Optimized Machine Learning Models Towards Intelligent Systems

MohammadNoor Ahmad Mohammad Injadat, *The University of Western Ontario*

Supervisor: Dr. Abdallah Shami, *The University of Western Ontario*
Co-Supervisor: Dr. Ali Bou Nassif, *The University of Western Ontario*

Abstract

The rapid growth of the Internet and related technologies has led to the collection of large amounts of data by individuals, organizations, and society in general [1]. However, this often leads to information overload which occurs when the amount of input (e.g. data) a human is trying to process exceeds their cognitive capacities [2]. Machine learning (ML) has been proposed as one potential methodology capable of extracting useful information from large sets of data [1]. This thesis focuses on two applications. The first is education, namely e-Learning environments. Within this field, this thesis proposes different optimized ML ensemble models to predict students' performance at earlier stages of the course delivery. Experimental results showed that the proposed optimized ML ensemble models accurately identified the weak students who needed help. More specifically, these models achieved an accuracy of up to 96% in the binary case and 93.1% in the multi-class case.

The second application is network security intrusion detection. Within this application field, this thesis proposes different optimized ML classification frameworks using a variety of optimization modeling algorithms and heuristics to improve the performance of the IDSs through anomaly detection while maintaining or reducing their time complexity. Experimental results showed that the developed models reduced the training sample size by up to 74%, reduced the feature set size by almost 60%, and improved the detection accuracy by up to 2%.

This thesis can be divided into two main parts. The first part analyzes different educational datasets and proposes different optimized ML classification ensemble models that accurately predict weak students who may need help. The second part proposes optimized ML classification frameworks that accurately detect network attacks while maintaining a low false alarm rate and time complexity. It is noteworthy that the developed models and frameworks could be generalized as follows:

- Optimized ML ensemble models proposed in the first part of this thesis can be generalized to many applications such as finance, network security, social media, and healthcare systems.
- Optimized ML classification models proposed in the second part of this thesis can be generalized to other applications that typically generate large datasets in terms of instances and feature set.

Keywords: Machine Learning, Data Analytics, Application Fields, e-Learning, Student Performance Prediction, Optimized Ensemble Learning Model Selection, Gini Index, p-value, Network Anomaly Detection, Hyper-parameter Optimization, Bayesian Optimization, Particle Swarm Optimization, Genetic Algorithm

Lay Summary

The rapid growth of the Internet and related technologies has led to the collection of large amounts of data by individuals, organizations, and society in general. However, these large amounts of data often lead to information overload which occurs when the amount of input (e.g. data) that a human is trying to process exceeds their cognitive capacities. In turn, this can lead to humans ignoring, overlooking, or misinterpreting crucial information. Machine learning (ML) has been proposed as one potential data analysis and prediction methodology capable of extracting useful information from large sets of data. ML allows computers to learn without being explicitly programmed. Accordingly, the computer can apply what it has learned to find the learned patterns in similar data. Furthermore, ML allows computer systems to adapt and learn from their experience.

This thesis focuses on two applications. The first is education, namely e-Learning environments. Within this field, this thesis proposes the use of different optimized ML models to predict students' performance at earlier stages of the course delivery. The second application is network security intrusion detection. Within this application field, this thesis proposes different optimized ML classification frameworks using a variety of optimization modeling algorithms and heuristics to improve the performance of the IDSs through anomaly detection while maintaining or reducing their time complexity.

Co-Authorship Statement

The following thesis contains material from manuscripts that are published or submitted for publication that have been co-authored by MohammadNoor Injadat, Dr. Abdallah Moubayed, Dr. Ali Bou Nassif, Dr. Abdallah Shami, Dr. Fadi Salo, and Dr. Aleksander Essex. All the research, developments, experiments, and work presented in this thesis were conducted by MohammadNoor Injadat under the supervision of Dr. Abdallah Shami and Dr. Ali Bou Nassif. Original manuscripts which make up parts of Chapters 3 through 7 in this thesis were also written by MohammadNoor Injadat.

Publications:

[J1] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Machine Learning Towards Intelligent Systems: Applications, Challenges, and Opportunities," *Submitted to Artificial Intelligence Review*, 2020.

[J2] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Systematic Ensemble Model Selection Approach for Educational Data Mining, " in *Elsevier: Knowledge-based Systems*, vol. 200, page 105992, 2020, DOI: 10.1016/j.knosys.2020.105992.

[J3] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-split Optimized Bagging Ensemble Model Selection for Multi-class Educational Datasets, " in *Springer: Applied Intelligence*, 2020, DOI: 10.1007/s10489-020-01776-3.

[J4] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-Stage Optimized Machine Learning Framework for Network Intrusion Detection," *Accepted in IEEE Transactions On Network and Service Management*, 2020, DOI: 10.1109/TNSM.2020.3014929 .

[C1] M. Injadat, F. Salo, A. B. Nassif, A. Essex, and A. Shami,"Bayesian optimization with machine learning algorithms towards anomaly detection," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.

Acknowledgments

In the Name of Allah, the Most Beneficent, the Most Merciful. First and foremost, my praise be to Allah, Lord of the Worlds, for providing me with the strength, determination and energy to complete this thesis. Without the grace of Almighty Allah, I would not be able to reach this stage.

Second, I would like to express my sincere thanks, appreciation and gratitude to my supervisors, Dr. Abdallah Shami and Dr. Ali Bou Nassif for their unlimited support, professional guidance and exemplary supervision while perusing my PhD. I could not seek better supervision than what I got during this process. I cannot give Dr. Shami and Dr. Bou Nassif the thanks they deserve for all their contributions, as well as their time, and support that made the PhD experience very fruitful, enjoyable and successful. Thank you for always encouraging me to move towards the goal, and your continuous support to overcome the obstacles I encountered during the study.

Third, I would like to express my thankfulness to the examination committee members: Dr. Luiz Fernando Capretz, Dr. Shaimaa Ali, Dr. Anwar Haque, and Dr. Ramiro Liscano. Thank you all for your time in reviewing and examining my thesis and for the discerning comments and suggestion that have been provided. Also, I would like to express my full thanks to Ms. Stephanie Tigert, Ms. Whitney Barrett, Ms. Courtney Harper, Ms. Yan Zhao, Ms. Michelle Wagler, Ms. Andrea Krasznai, and all the administrative staff at the faculty of Engineering and the University of Western Ontario for providing me with all the required help throughout this journey.

Fourth, I would like to express my heartfelt thanks and love to my family, first to my father, Ahmad Injadat, who taught and still is teaching me to persevere and work hard to achieve my goals, and secondly, to my brothers, Nidal, Moawieh, Khaldoun, and my sister Maram and their families and children who encouraged me in every step throughout this distinctive scientific journey.

Fifth, I am eternally grateful to my wife Rawia Zuod, my children Leen, Noor Aldeen, Ahmad and Raya, thank you all for your love, help, and being a great family that continuously encouraged and helped me to achieve this milestone in my educational and professional life.

Sixth, I would like to thank all of my amazing colleagues that I got the great chance to meet and know at the University of Western Ontario: Abdallah Moubayed, Anas Saci, Mohamed Khalil, Karim Hammad, Mohamed Hussein, Khaled Al Hazmi, Emad Aqeeli, Mohammad Abu Sharkh, Sara Zimmo, Hassan Hawilo, Manar Jammal, Elena Uchiteleva, Fadi Salo, Tamer Mohamed, Fuad Shamieh, Sam Aleyadeh, Li Yang , all the previous and current colleagues in Optimized Computing and Communications (OC2) Laboratory, and all the amazing people I have met at University of Western Ontario.

Finally, my special thanks to all friends from Jordan, UAE, Saudi Arabia, Malaysia, Europe, United States, and Canada and all around the world, thank you for the continuous encouragement to complete my study.
Without all of you, I could not be able to write this thesis.

MohammadNoor Injadat

To the memory of my lovely family (my wife Rawia and my children Leen, Noor Aldeen, Ahmad, and Raya).

To the memory of my father, brothers and their families, sister and her family.

To the memory of my late mother who could not wait to see this thesis is completed. I miss you Mum and I hope that you are rest in peace.

Contents

Abstract ii

Lay Summary iii

Co-Authorship Statement iv

Acknowledgements v

List of Figures xiii

List of Tables xvi

List of Abbreviations, Symbols, and Nomenclature xx

1 Introduction 1

1.1 Introduction . . . 1

1.2 Motivation . . . 2

1.2.1 Need for ML in e-Learning . . . 2

1.2.2 Need for ML in Network Security . . . 3

1.3 Thesis Objectives . . . 5

1.4 Thesis Organization . . . 5

1.5 Thesis Contributions . . . 6

1.5.1 Contributions of Chapter 3 . . . 6

1.5.2 Contributions of Chapter 4 . . . 6

1.5.3 Contributions of Chapter 5 . . . 7

1.5.4 Contributions of Chapter 6 . . . 7

1.5.5 Contributions of Chapter 7 . . . 7

2 Machine Learning Background 9

2.1 Machine Learning Mathematical Background . . . 9

2.2 Notation and Definitions . . . 10

2.3 Performance Evaluation Criteria . . . 11

2.4 Principal Component Analysis . . . 15

2.5 Logistic Regression . . . 16

2.6 Support Vector Machine . . . 18

2.7 Artificial Neural Network . . . 19

2.8 k-Nearest Neighbors (k-NN) . . . 20

2.9 Naïve Bayes . . . 21
2.10 Random Forest . . . 22
2.11 Ensemble Learning . . . 23

3 Machine Learning Towards Intelligent Systems: Applications, Challenges, and Opportunities 25
3.1 Introduction . . . 25
3.2 Education . . . 26
3.2.1 Essay Grading . . . 27
Challenge Description: . . . 27
Previous Works: . . . 27
Research Opportunities: . . . 28
3.2.2 Dropout Prevention . . . 29
Challenge Description: . . . 29
Previous Works: . . . 29
Research Opportunities: . . . 30
3.2.3 Intelligent Tutoring . . . 30
Challenge Description: . . . 30
Previous Works: . . . 30
Research Opportunities: . . . 31
3.2.4 Course Recommendation . . . 31
Challenge Description: . . . 31
Previous Works: . . . 32
Research Opportunities: . . . 32
3.2.5 Personalized Learning . . . 32
Challenge Description: . . . 32
Previous Works: . . . 32
Research Opportunities: . . . 34
3.3 Healthcare . . . 34
3.3.1 Drug Response Prediction . . . 36
Challenge Description: . . . 36
Previous Works: . . . 36
Research Opportunities: . . . 37
3.3.2 Diabetes Research . . . 37
Challenge Description: . . . 37
Previous Works: . . . 38
Research Opportunities: . . . 39
3.4 Network Security . . . 40
3.4.1 Network Intrusion Detection Systems . . . 41
Challenge Description: . . . 41
Previous Works: . . . 42
Research Opportunities: . . . 42
3.4.2 Botnets Detection . . . 43
Challenge Description: . . . 43
Previous Works: . . . 43

Research Opportunities: 44
3.4.3 Intrusion Detection in Vehicles 45
Challenge Description: 45
Previous Works: 45
Research Opportunities: 45
3.5 Banking & Finance 47
3.5.1 Credit Risk Assessment 47
Challenge Description: 47
Previous Works: 47
Research Opportunities: 48
3.5.2 Bankruptcy Prediction 48
Challenge Description: 48
Previous Works: 48
Research Opportunities: 50
3.5.3 Currency Crises Prediction 50
Challenge Description: 50
Previous Works: 50
Research Opportunities: 51
3.6 Social Media 51
3.6.1 Pharmacovigilance 52
Challenge Description: 52
Previous Works: 53
Research Opportunities: 54
3.6.2 Social Media and Vaccines 54
Challenge Description: 54
Previous Works: 55
Research Opportunities: 56
3.6.3 Social Media and Politics 56
Challenge Description: 56
Previous Works: 57
Research Opportunities: 58
3.7 Conclusion 59

4 Systematic Ensemble Model Selection Approach for Educational Data Mining 61
4.1 Introduction 61
4.2 Related Work 62
4.2.1 Limitations of Related Work 64
4.3 Contribution of Proposed Research 64
4.4 Methodology and the Research Framework 65
4.5 Datasets' Description and Analysis 67
4.5.1 Dataset Description 67
4.5.2 Dataset Visualization 71
4.6 ML application to the datasets 76
4.6.1 Dataset 1 - Stage 20% 78
4.6.2 Dataset 1 - Stage 50% 81

4.6.3 Dataset 2 - Stage 20% . . . 85
4.6.4 Dataset 2 - Stage 50% . . . 86
4.6.5 Proposed method Ensemble learners: a systematic approach . . . 87
4.7 Results and Discussion . . . 89
4.7.1 Results: Dataset 1 - Stage 20% . . . 90
4.7.2 Results: Dataset 1 - Stage 50% . . . 91
4.7.3 Results: Dataset 2 - Stage 20% . . . 93
4.7.4 Results: Dataset 2 - Stage 50% . . . 93
4.7.5 Confusion Matrices . . . 95
4.8 Conclusion and Research Limitations . . . 96

5 Multi-split Optimized Bagging Ensemble Model Selection for Multi-class Educational Datasets 98
5.1 Introduction . . . 98
5.2 Related Work and Limitations . . . 100
5.2.1 Related Work . . . 100
5.2.2 Limitations of Related Work . . . 101
5.3 Research Contribution . . . 102
5.4 Multi-class Target Variable Description . . . 102
5.5 Performance Evaluation Metrics Description . . . 105
5.6 Bias-Variance Tradeoff and Bagging . . . 105
5.7 Methodology . . . 107
5.8 ML Parameter Tuning and Application . . . 111
5.8.1 Neural Network Tutning . . . 111
5.8.2 ML Algorithms' Parameter Tuning . . . 112
5.8.3 Features importance: Dataset 1 - Stage 20% . . . 114
5.8.4 Features importance: Dataset 1 - Stage 50% . . . 116
5.8.5 Features importance: Dataset 2 - Stage 20% . . . 120
5.8.6 Features importance: Dataset 2 - Stage 50% . . . 120
5.9 Experimental Results and Discussion . . . 121
5.9.1 Results: Dataset 1 - Stage 20% . . . 122
5.9.2 Results: Dataset 1 - Stage 50% . . . 123
5.9.3 Results: Dataset 2 - Stage 20% . . . 125
5.9.4 Results: Dataset 2 - Stage 50% . . . 126
5.9.5 Results Summary . . . 127
5.10 Conclusion and Research Limitations . . . 128

6 Bayesian Optimization with Machine Learning Algorithms Towards Anomaly Detection 129
6.1 Introduction . . . 129
6.2 Related Work . . . 130
6.3 THEORETIC ASPECTS OF THE TECHNIQUES . . . 131
6.3.1 A. Support Vector Machines (SVM) . . . 131
6.3.2 k-Nearest Neighbors (k-NN) . . . 131
6.3.3 Random Forests (RF) . . . 132

6.3.4 Bayesian Optimization (BO) . . . 132
6.4 EXPERIMENTAL SETUP AND RESULT DISCUSSION . . . 133
6.4.1 Dataset Description . . . 133
6.4.2 Experimental setup and Data Pre-processing . . . 133
6.4.3 Prediction Performance Measures . . . 133
6.4.4 Results Discussion . . . 133
6.5 Conclusions . . . 138

7 Multi-Stage Optimized Machine Learning Framework for Network Intrusion Detection 139
7.1 Introduction . . . 139
7.2 Related Work and Limitations . . . 141
7.2.1 Related Work . . . 141
7.2.2 Limitations of Related Work . . . 141
7.3 Research Contributions . . . 142
7.4 Proposed Multi-Stage Optimized ML-based NIDS Framework . . . 142
7.4.1 General Framework Description: . . . 142
7.4.2 Data Pre-processing: . . . 143
Z-Score Normalization . . . 144
SMOTE Technique . . . 144
7.4.3 Feature Selection: . . . 144
Information Gain-based Feature Selection . . . 145
Correlation-based Feature Selection . . . 145
7.4.4 Hyper-parameter Optimization: . . . 145
Random Search . . . 146
Meta-heuristic Optimization Algorithms . . . 146
Bayesian Optimization . . . 148
7.4.5 Complexity: . . . 149
7.4.6 Security Considerations: . . . 150
7.5 Datasets Description . . . 150
7.5.1 CICIDS 2017 . . . 150
7.5.2 UNSW-NB 2015 . . . 151
7.5.3 Attack Types . . . 152
7.6 Experimental Performance Evaluation . . . 152
7.6.1 Experimental Setup . . . 152
7.6.2 Results and Discussion . . . 153
Impact of data pre-processing on training sample size . . . 153
Impact of feature selection on feature set size and training sample size . 155
Impact of optimization methods on the ML models' detection performance . . . 159
7.7 Conclusion . . . 161

8 Conclusion and Future Research Directions 163
8.1 Introduction . . . 163
8.2 Summary of Contributions . . . 164

8.3 Future Research Directions . . . 165
8.3.1 Research Opportunities Towards Intelligent e-Learning Environments . 165
8.3.2 Research Opportunities Towards More Effective Network Security Frameworks . . . 166

Bibliography **167**

A Appendix **191**
A.1 Ensemble Learners - Majority Voting Ensemble . . . 191
A.2 Ensemble Learners - Multi-Split Bagging Ensemble . . . 199

Curriculum Vitae **243**

List of Figures

1.1 Global ML Forecast [3] 2
1.2 Percentage of Student Enrollment in US Institutions 2017 [4] 3
1.3 Number of Accounts Hacked for Different Companies [5] 4
1.4 Monetary Damage Caused by Cybercrime Reported to Internet Crime Complaint Center (IC3) [6] 4

2.1 Example of Gini Index Computation 13
2.2 Example of Gini Index plot and Stats 14
2.3 Sigmoid function 17
2.4 Example of Artificial Neural Network 20
2.5 Ensemble Method 23

3.1 Potential Deployment of ML in LMS 34
3.2 Potential Deployment of ML in Diabetes Research 39
3.3 Potential Deployment of ML in Network Security 41
3.4 Potential ML-based Credit Risk and Bankruptcy Assessment Framework 49
3.5 Potential ML-based Social Media Analytics Framework 58
3.6 Summary of Challenges and ML Techniques 60

4.1 Learning Management System (LMS) Analytical Module 65
4.2 ML-Based Student Status Predictor 66
4.3 Dataset 1 - Features' distribution 68
4.4 Dataset 1 - Final Grade Distribution 69
4.5 Dataset 2 - Features' distribution 70
4.6 Dataset 2 - Final Grade Distribution 70
4.7 Target variable: Dataset 1 vs. Dataset 2 71
4.8 Decision boundaries for dataset 1 72
4.9 Decision boundaries for dataset 2 72
4.10 Dataset 1 - Stage 20% - Percentage of variance per principal component 73
4.11 Dataset 1 - Stage 20% - First two principal components 74
4.12 Dataset 1 - Stage 20% - First four principal components 75
4.13 Dataset 1 - Stage 20% - First and second component 75
4.14 Dataset 1 - Stage 20% - Third and fourth component 76
4.15 Dataset 1 - Stage 50% - Percentage of variance per principal component 76
4.16 Dataset 1 - Stage 50% - First four principal components 77
4.17 Dataset 1 - Stage 50% - First and second component 77
4.18 Dataset 1 - Stage 50% - Third and fourth component 78

4.19 Dataset 1 - Stage 20% - Random Forest, Accuracy vs. mtry value 79
4.20 Dataset 1 - Stage 20% - Performance vs. k 81
4.21 Dataset 1 - Stage 50% - Random Forest, Performance vs. number of features selected 82
4.22 Dataset 1 - Stage 50% - Performance vs. k 84
4.23 The procedure of generating ensemble learners and creating the score for each ensemble learner 88
4.24 Dataset 1 - Stage 20% - Performance of different classifiers on 5 splits 89
4.25 Dataset 1 - Stage 20% - Performance Ensemble Learner 90
4.26 Dataset 1 - Stage 20% - Performance Ensemble Learner 91
4.27 Dataset 1 - Stage 50% - Performance Ensemble Learner 92
4.28 Dataset 1 - Stage 50% - Performance Ensemble Learner 92
4.29 Dataset 2 - Stage 20% - Performance Ensemble Learner 93
4.30 Dataset 2 - Stage 20% - Performance Ensemble Learner 94
4.31 Dataset 2 - Stage 50% - Performance Ensemble Learner 94
4.32 Dataset 2 - Stage 50% - Performance Ensemble Learner 95

5.1 Dataset 1 and Dataset 2- Target Variables 103
5.2 Dataset 1 - Binary Target Variable vs. multi-class Target Variable 103
5.3 Dataset 2 - Binary Target Variable vs. multi-class Target Variable 104
5.4 Dataset 1 - multi-class target visualization 104
5.5 Dataset 2 - multi-class target visualization 104
5.6 Bias-Variance 106
5.7 Illustrative Example of Variance Reduction Using Bagging 107
5.8 Bagging Ensemble Model Building Methodology 108
5.9 Example - Averaged Gini Index Computation 109
5.10 NN with 1 hidden layer 112
5.11 NN with 2 hidden layers 112
5.12 NN with 3 hidden layers 112
5.13 Dataset 2 - Stage 20% - scatter plot 121
5.14 Dataset 1 - Stage 20% - Ensemble Learner 122
5.15 Dataset 1 - Stage 50% - Ensemble Learner 123
5.16 Dataset 2 - Stage 20% - Ensemble Learner 125
5.17 Dataset 2 - Stage 50% - Ensemble Learner 126

6.1 Optimized SVM Contour 134
6.2 Optimized SVM objective function model 135
6.3 Optimized k-NN Contour 135
6.4 Optimized k-NN Objective Function Model 136
6.5 BO-SVM Objective Function vs Number of Function Evaluations 136
6.6 BO-kNN Objective Function vs Number of Function Evaluations 137
6.7 BO-RF Objective Function vs Number of Function Evaluations 137

7.1 Proposed Multi-stage Optimized ML-based NIDS Framework 143
7.2 Principal Component Analysis of CICIDS 2017 Dataset 151

7.3 Principal Component Analysis of UNSW-NB 2015 Dataset 152
7.4 Learning Curve for CICIDS 2017 Dataset Before SMOTE 154
7.5 Learning Curve for UNSW-NB 2015 Dataset Before SMOTE 154
7.6 Learning Curve for CICIDS 2017 Dataset After SMOTE 155
7.7 Learning Curve for UNSW-NB 2015 Dataset After SMOTE 156
7.8 Mutual Information Score of Features for CICIDS 2017 Dataset 156
7.9 Mutual Information Score of Features for UNSW-NB 2015 Dataset 157
7.10 Learning Curve for CICIDS 2017 Dataset After IGBFS 157
7.11 Learning Curve for UNSW-NB 2015 Dataset After IGBFS 158
7.12 Learning Curve for CICIDS 2017 Dataset After CBFS 158
7.13 Learning Curve for UNSW-NB 2015 Dataset After CBFS 159

List of Tables

2.1 Confusion matrix corresponding to threshold $\tau = 0.38$ . . . 15
2.2 Performance corresponding to $\tau = 0.38$ . . . 15

3.1 Challenges, Previous Works, and Research Opportunities within Education Sector 35
3.2 Challenges, Previous Works, and Research Opportunities within Healthcare Sector . . . 40
3.3 Challenges, Previous Works, and Research Opportunities within Network Security Field . . . 46
3.4 Challenges, Previous Works, and Research Opportunities within Banking and Finance Field . . . 51
3.5 Challenges, Previous Works, and Research Opportunities within Social Media Field . . . 59

4.1 Dataset 1 - Features . . . 68
4.2 Dataset 2 - Features . . . 69
4.3 Dataset 1 - Stage 20% - Features' weights for Random Forest . . . 79
4.4 Dataset 1 - Stage 20% - Features' weights for SVM-RBF . . . 80
4.5 Dataset 1 - Stage 20% - Features' weights for MLP . . . 80
4.6 Dataset 1 - Stage 20% - Features' weights for k-NN . . . 80
4.7 Dataset 1 - Stage 20% - Features' weights for Logistic Regression . . . 81
4.8 Dataset 1 - Stage 20% - Features' weights for Naive Bayes . . . 82
4.9 Dataset 1 - Stage 50% - Features' weights for Random Forest . . . 83
4.10 Dataset 1 - Stage 50% - Features' weights for SVM-RBF . . . 83
4.11 Dataset 1 - Stage 50% - Features' weights for MLP . . . 84
4.12 Dataset 1 - Stage 50% - Features' weights for k-NN . . . 85
4.13 Dataset 1 - Stage 50% - Features' weights for Logistic Regression . . . 85
4.14 Dataset 1 - Stage 50% - Features' weights for Naïve Bayes . . . 86
4.15 Dataset 2 - Stage 20% - Features' weights . . . 86
4.16 Dataset 2 - Stage 50% - weights RF, SVM, MLP, k-NN and NB . . . 86
4.17 Dataset 2 - Stage 50% - Features' weights for LR . . . 86
4.18 Dataset 1 - Stage 20% - Best classifiers . . . 90
4.19 Dataset 1 - Stage 50% - Best Classifiers . . . 91
4.20 Dataset 2 - Stage 20% - Best Classifiers . . . 93
4.21 Dataset 2 - Stage 50% - Best Classifiers . . . 94
4.22 Base Classifiers' Performances . . . 95
4.23 Confusion Matrix $\tau = 0.35$ Ensemble: RF and BN Dataset 1 - stage 20% . . . 96
4.24 Confusion Matrix $\tau = 0.35$ Ensemble: RF, k-NN and SVM Dataset 1 - stage 50% 96

4.25 Confusion Matrix $\tau = 0.065$ Ensemble: LR and BN Dataset 2 - stage 20% . . . 96
4.26 Confusion Matrix $\tau = 0.2$ Ensemble: LR and MLP Dataset 2 - stage 50% . . . 96
4.27 Ensemble Learners' Performances . . . 96

5.1 Example - Predicting Classes . . . 110
5.2 Example . . . 110
5.3 Grid Search Parameter Tuning Range . . . 113
5.4 Dataset 1 - Stage 20% - Features' weights for Random Forest . . . 114
5.5 Dataset 1 - Stage 20% - Features' weights for SVM . . . 115
5.6 Dataset 1 - Stage 20% - Features' weights for NN with 1 hidden layer . . . 115
5.7 Dataset 1 - Stage 20% - Features' weights for NN with 2 hidden layers . . . 115
5.8 Dataset 1 - Stage 20% - Features' weights for NN with 3 hidden layers . . . 116
5.9 Dataset 1 - Stage 20% - Features' weights for k-NN . . . 116
5.10 Dataset 1 - Stage 20% - Features' weights for LR . . . 116
5.11 Dataset 1 - Stage 20% - Features' weights for BN . . . 117
5.12 Dataset 1 - Stage 50% - Features' weights for RF . . . 117
5.13 Dataset 1 - Stage 50% - Features' weights for SVM . . . 117
5.14 Dataset 1 - Stage 50% - Features' weights for NN with 1 hidden layer . . . 118
5.15 Dataset 1 - Stage 50% - Features' weights for NN with 2 hidden layer . . . 118
5.16 Dataset 1 - Stage 50% - Features' weights for NN with 3 hidden layer . . . 119
5.17 Dataset 1 - Stage 50% - Features' weights for k-NN . . . 119
5.18 Dataset 1 - Stage 50% - Features' weights for LR . . . 119
5.19 Dataset 1 - Stage 50% - Features' weights for Naïve Bayes . . . 120
5.20 Dataset 2 - Stage 20% - Features' weights . . . 120
5.21 Dataset 2 - Stage 50% - NN1, NN2, KNN, and BN, Features' weights . . . 121
5.22 Dataset 2 - Stage 50% - NN3, LR, RF, and SVM, Features' weights . . . 122
5.23 Confusion Matrix $\tau_F = 0.158, \tau_G = 0.310, \tau_W = 0.682$ Ensemble: NN 2 Dataset 1 - Stage 20% . . . 123
5.24 Dataset 1 - Stage 20% - multi-class Classification Problem - Ensemble Performances . . . 123
5.25 Confusion Matrix $\tau_F = 0.10, \tau_G = 0.29, \tau_W = 0.88$ Ensemble: BN and k-NN Dataset 1 - Stage 50% . . . 124
5.26 Dataset 1 - Stage 50% - multi-class Classification Problem - Ensemble Performances . . . 124
5.27 Confusion Matrix $\tau_F = 0.12, \tau_G = 0.62, \tau_W = 0.40$ Ensemble: BN, k-NN, LR, NN 2 and SVM Dataset 2 - Stage 20% . . . 125
5.28 Dataset 2 - Stage 20% - multi-class Classification Problem - Ensemble Performances . . . 126
5.29 Confusion Matrix $\tau_F = 0.12, \tau_G = 0.62, \tau_W = 0.30$ Ensemble: LR Dataset 2 - stage 50% . . . 127
5.30 Dataset 2 - Stage 50% - multi-class Classification Problem - Ensemble Performances . . . 127

6.1 Optimization parameters for each classifier . . . 134
6.2 Performance results of the three classifiers . . . 135

7.1 Optimal Parameter Values with IGBFS . . . 160
7.2 Optimal Parameter Values with CBFS . . . 160
7.3 Performance results of the Multi-stage Optimized ML-based NIDS Framework with IGBFS . . . 160
7.4 Performance results of the Multi-stage Optimized ML-based NIDS Framework with CBFS . . . 161

A.1 Dataset 1 - Stage 20% - Part1/2 . . . 191
A.2 Dataset 1 - Stage 20% - Part2/2 . . . 192
A.3 Dataset 1 - Stage 50% - Part1/2 . . . 193
A.4 Dataset 1 - Stage 50% - Part2/2 . . . 194
A.5 Dataset 2 - Stage 20% - Part1/2 . . . 195
A.6 Dataset 2 - Stage 20% - Part2/2 . . . 196
A.7 Dataset 2 - Stage 50% - Part1/2 . . . 197
A.8 Dataset 2 - Stage 50% - Part2/2 . . . 198
A.9 Dataset 1 at Stage 20% - Gini Indices Part 1/6 . . . 199
A.10 Dataset 1 at Stage 20% - Gini Indices Part 2/6 . . . 200
A.11 Dataset 1 at Stage 20% - Gini Indices Part 3/6 . . . 201
A.12 Dataset 1 at Stage 20% - Gini Indices Part 4/6 . . . 202
A.13 Dataset 1 at Stage 20% - Gini Indices Part 5/6 . . . 203
A.14 Dataset 1 at Stage 20% - Gini Indices Part 6/6 . . . 204
A.15 Dataset 1 at Stage 50% - Gini Indices Part 1/6 . . . 205
A.16 Dataset 1 at Stage 50% - Gini Indices Part 2/6 . . . 206
A.17 Dataset 1 at Stage 50% - Gini Indices Part 3/6 . . . 207
A.18 Dataset 1 at Stage 50% - Gini Indices Part 4/6 . . . 208
A.19 Dataset 1 at Stage 50% - Gini Indices Part 5/6 . . . 209
A.20 Dataset 1 at Stage 50% - Gini Indices Part 6/6 . . . 210
A.21 Dataset 2 at Stage 20% - Gini Indices Part 1/6 . . . 211
A.22 Dataset 2 at Stage 20% - Gini Indices Part 2/6 . . . 212
A.23 Dataset 2 at Stage 20% - Gini Indices Part 3/6 . . . 213
A.24 Dataset 2 at Stage 20% - Gini Indices Part 4/6 . . . 214
A.25 Dataset 2 at Stage 20% - Gini Indices Part 5/6 . . . 215
A.26 Dataset 2 at Stage 20% - Gini Indices Part 6/6 . . . 216
A.27 Dataset 2 at Stage 50% - Gini Indices Part 1/6 . . . 217
A.28 Dataset 2 at Stage 50% - Gini Indices Part 2/6 . . . 218
A.29 Dataset 2 at Stage 50% - Gini Indices Part 3/6 . . . 219
A.30 Dataset 2 at Stage 50% - Gini Indices Part 4/6 . . . 220
A.31 Dataset 2 at Stage 50% - Gini Indices Part 5/6 . . . 221
A.32 Dataset 2 at Stage 50% - Gini Indices Part 6/6 . . . 222
A.33 Dataset 1 at Stage 20% - p values splits- Part1/5 . . . 223
A.34 Dataset 1 at Stage 20% - p values splits- Part2/5 . . . 224
A.35 Dataset 1 at Stage 20% - p values splits- Part 3/5 . . . 225
A.36 Dataset 1 at Stage 20% - p values splits- Part 4/5 . . . 226
A.37 Dataset 1 at Stage 20% - p values splits- Part 5/5 . . . 227
A.38 Dataset 1 at Stage 50% - p values splits- Part 1/5 . . . 228

A.39 Dataset 1 at Stage 50% - p values splits- Part 2/5 229
A.40 Dataset 1 at Stage 50% - p values splits- Part 3/5 230
A.41 Dataset 1 at Stage 50% - p values splits- Part 4/5 231
A.42 Dataset 1 at Stage 50% - p values splits- Part 5/5 232
A.43 Dataset 2 at Stage 20% - p values splits- Part 1/5 233
A.44 Dataset 2 at Stage 20% - p values splits- Part 2/5 234
A.45 Dataset 2 at Stage 20% - p values splits- Part 3/5 235
A.46 Dataset 2 at Stage 20% - p values splits- Part 4/5 236
A.47 Dataset 2 at Stage 20% - p values splits- Part 4/5 237
A.48 Dataset 2 at Stage 50% - p values splits- Part 1/5 238
A.49 Dataset 2 at Stage 50% - p values splits- Part 2/5 239
A.50 Dataset 2 at Stage 50% - p values splits- Part 3/5 240
A.51 Dataset 2 at Stage 50% - p values splits- Part 4/5 241
A.52 Dataset 2 at Stage 50% - p values splits- Part 5/5 242

List of Abbreviations, Symbols, and Nomenclature

Acc	Accuracy
ANN	Artificial Neural Network
AUC	Area Under the Curve
AVA	All Versus All
BO	Bayesian Optimization
CAN	Controller Area Network
CBFS	Correlation Based Feature Selection
CICIDS	Canadian Institute of Cybersecurity Intrusion Detection System
DA	Discriminant Analysis
DBN	Deep Believe Network
DDoS	Distributed Denial of Service
DEEDS	Digital Electronics Education and Design Suite
DM	Data Mining
DNN	Deep Neural Network
DoS	Denial of Service
DR	Detection Rate
DT	Decision Tree
ECU	Electronics Control Unit
EDM	Educational Data Mining
EI	Expected Improvement
FAR	False Alarm Rate
FS	Feature Selection
FN	False Negative
FP	False Positive
GA	Genetic Algorithm
GB	Gaussian-Based
GP	Gaussian Process
HMM	Hidden Markov Model
IC3	Internet Crime Complaint Center
ID3	Iterative Dichotomiser 3
IDS	Intrusion Detection System
IG	Information Gain
IGBFS	Information Gain Based Feature Selection
ILSQ	Index Learning Style Questionnaire
ISCX	Information Security Centre of Excellence
K-NN	K-Nearest Neighbors
KPCA	Kernel Principal Component Analysis
LDA	Latent Dirichlet Allocation
LMS	Learning Management System
LOOCV	Leave One Out Cross Validation
LR	Logistic Regression
LSA	Latent Semantic Analysis
ML	Machine Learning
MLP	Multi-Layer Perceptron
MOOC	Massively Open Online Courses
NB	Naive Bayes
NIDS	Network Intrusion Detection System
NLP	Natural Language Processing
NN	Neural Networks
OVA	One Versus All
P2P	Peer-to-Peer
PCA	Principal Component Analysis
PSO	Particle Swarm Optimization
RBF	Radial Basis Function
RF	Random Forest
RFE	Recursive Feature Elimination
ROC	Receiver Operating Characteristic
RS	Random Search
SMOTE	Synthetic Minority Oversampling TEchnique
SSO	Simplified Swarm Optimization
STL	Self-Taught Learning
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
TPE	Tree Parzen Estimator
TPR	True Positive Rate
UNSW-NB	University of New South Wales-New Bot
WLS	Weighted Local Search

Chapter 1

Introduction

1.1 Introduction

The rapid growth of the Internet and related technologies has led to individuals, organizations, and society collecting large amounts of data [1]. However, these large amounts of data often lead to information overload which occurs when the amount of input (e.g. data) that a human is trying to process exceeds their cognitive capacities [2]. In turn, this can lead to humans ignoring, overlooking, or misinterpreting crucial information [7].

Humans do not have the cognitive capacity to process large amounts of data. To address this, the discipline of data science has emerged. Data science combines the classic disciplines of statistics, data mining, databases, and distributed systems in order to extract information from large sets of data [1]. Among the different data analysis methods that data scientists can implement is machine learning (ML). ML allows computers to learn without being explicitly programmed. Upon learning patterns from a training set of data, the computer can apply what it has learned to find these patterns in similar data [8]. Furthermore, ML allows computer systems to adapt and learn from their experience [9, 10].

ML has become an extremely popular topic within development organizations that are looking to adopt a data-driven approach to improve their business by gaining useful information from the data they collect. With ML models, organizations can continually predict changes in their business and make decisions accordingly. ML uses algorithms that iteratively learn from data to improve, describe data, and predict outcomes. Once an ML model has been trained, it can predict new data that is given as input. The output given by the model on the new data will depend on the data used to train the model.

The emerging growth of ML adoption in various fields is emphasized by the amount of financial resources being allocated to deploy ML models. As illustrated in Figure 1.1, the global ML market is expected to reach close to 42.5 billion CAD $ by the year 2024 [3]. Furthermore, as per McKinsey & Company's "Notes from the AI Frontier, Tackling Europe's Gap in Digital and AI" discussion paper, the ML market could boost economic activity growth throughout the EU by as much as 20% by the year 2030 [11]. Moreover, the World Economic Forum predicted that a net of 58 million jobs will be created in the coming years due to ML technologies [12]. This highlights the importance and positive potential impact that ML will have on the global economic market.

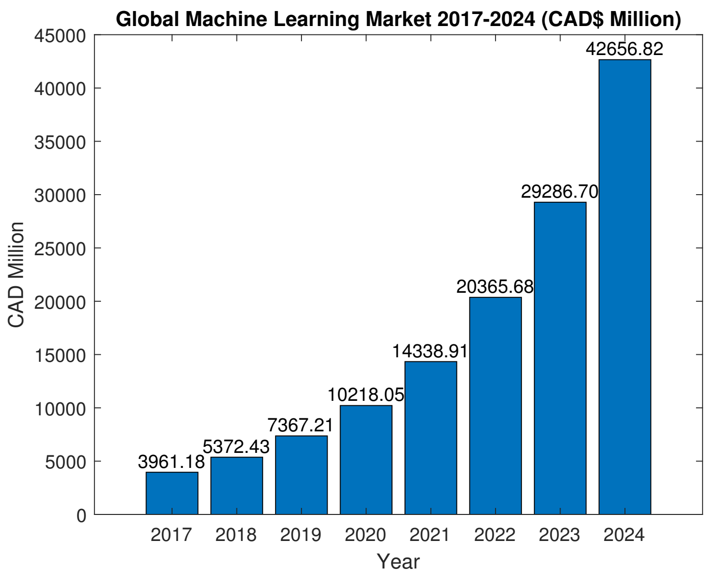


Figure 1.1: Global ML Forecast [3]

ML algorithms have several applications. This includes house pricing prediction, spam filtering, education, structuring of data in healthcare systems, drug response prediction, diabetes research, network security, banking and finance, and social media. This thesis focuses on two applications. The first is education, namely e-Learning environments. The second is network security intrusion detection.

1.2 Motivation

1.2.1 Need for ML in e-Learning

As mentioned earlier, the first application that this thesis focuses on is that of the education sector. Education can be delivered in three main modes: onsite, online, and blended learning [13]. Onsite education, or traditional education, refers to educational content delivery within a traditional classroom setting [13]. This requires that the educator and the students are simultaneously present in the same room. This allows the educator to deliver his/her lecture to the attending class. As such, traditional classrooms provide face-to-face interaction between the educator and the students [14].

In contrast, online education, one category of e-learning systems, refers to education that is provided over the Internet [13]. This allows students to access educational curriculum outside of a traditional classroom at any time from any geographical location [13]. However, there is no face-to-face interaction with the educator as all the content is delivered remotely [13].

The third delivery mode is the blended or hybrid learning. This mode is a combination of onsite and online education [13]. For the education delivery system to be considered blended, up to 30% of the course requirements must be completed face-to-face in a traditional classroom setting, while the remaining percentage of the course requirements can be completed online [13]. Blended learning allows students to have face-to-face interactions with the educator and

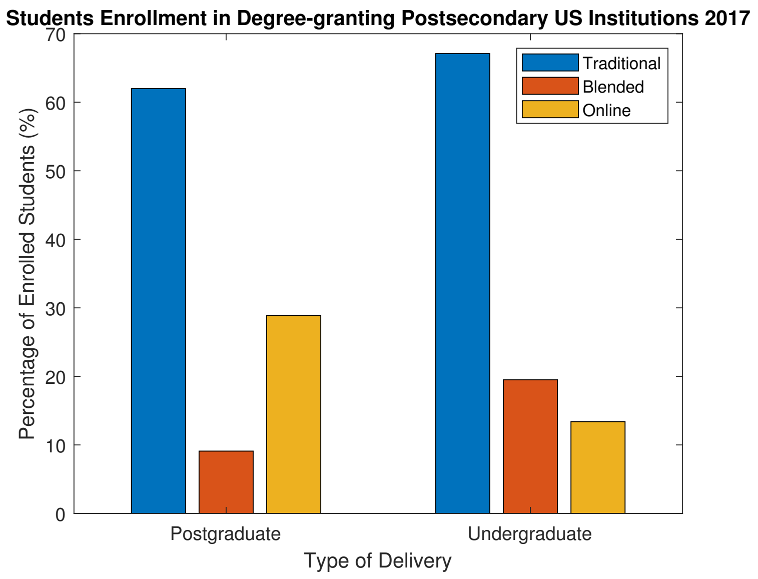


Figure 1.2: Percentage of Student Enrollment in US Institutions 2017 [4]

other students while also providing them with the opportunity to access course materials at any time from any location [13].

To highlight the continued emergence of e-learning, be it through blended or fully online learning, Figure 1.2 shows the percentage of students enrolled in degree-granting postsecondary US institutions in 2017. It can be seen that close to 33% of undergraduate students enrolled in some form of e-learning courses with around 20% enrolled in blended courses and 13% in online courses [4]. Similarly, around 38% of postgraduate students were enrolled in e-learning courses with around 9% registered in blended courses and around 29% in fully online courses [4]. This shows the transition from traditional programs to e-learning platforms. However, this entails dealing with a new set of challenges, some of which are pedagogical in nature and others that are technical. This includes automated essay grading procedures, intelligent tutoring mechanisms, dropout prevention, and personalized learning. As such, there is a need to analyze the plethora of data being generated and offer an intelligent automation mechanism that can help improve the educational process by tackling these challenges. This is where ML can play a vital role in addressing these different challenges facing e-learning environments.

1.2.2 Need for ML in Network Security

The second application considered in this thesis is the network security. Cisco Systems, Inc., an American multinational technology conglomerate who specializes in information technology, networking, and cybersecurity solutions, defines network security as any activity designed to protect the usability and integrity of a network and data [15]. According to Cisco Systems, Inc., network security allows authorized users to access a network while preventing outside threats from entering or spreading on a network [15, 16]. They listed fourteen types of network security. However, this thesis focuses on one network security aspect, namely Intrusion Detection Systems (IDS) [15]. IDSs analyze and monitor network traffic in order to

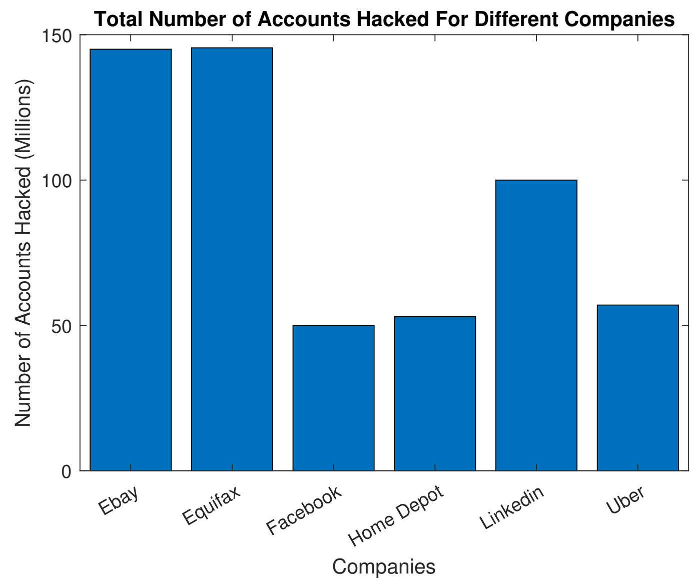


Figure 1.3: Number of Accounts Hacked for Different Companies [5]

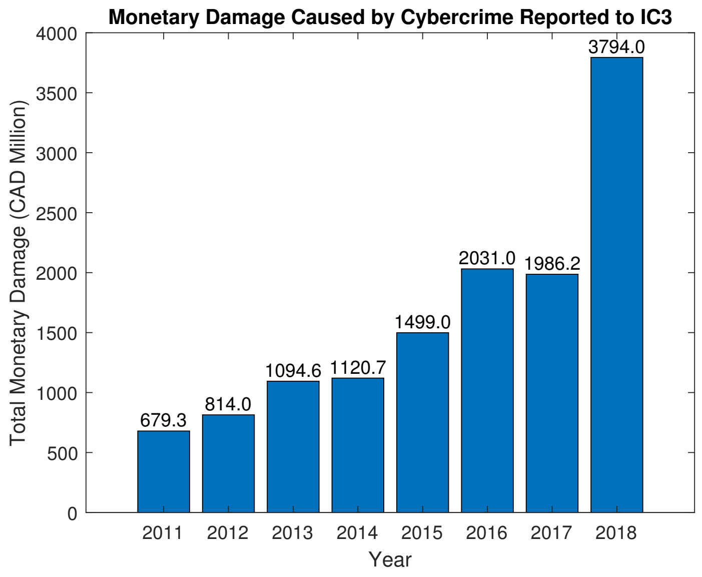


Figure 1.4: Monetary Damage Caused by Cybercrime Reported to Internet Crime Complaint Center (IC3) [6]

determine if the network traffic patterns show normal activity or if there are signs of malicious activity [17, 18].

The importance of having an effective and robust IDS is highlighted by the statistics about the number of accounts hacked for different leading international companies. As shown in Figure 1.3, companies such as Facebook, Uber, and Ebay had between 50 million and 145 million accounts hacked within the last five years [5]. This illustrates the significant privacy threat posed by not having intelligent and effective IDSs. Moreover, statistics reported by the Internet Crime Complaint Center (IC3) that the monetary damage resulting from Cybercrime rose from close to 679 million CAD $ in 2011 to around 3.8 billion CAD $ in 2018 [6] as shown in Figure 1.4. Hence, it can be seen that network security is an important challenge to address given the privacy threat it poses and the monetary damage it can cause. Thus, it is important to explore how ML models can improve the effectiveness and robustness of IDSs to reduce the privacy threat and resulting monetary losses incurred by the different individuals and organizations.

1.3 Thesis Objectives

This thesis can be divided into two main blocks. It focuses on different ML models and their hyper-parameter optimization to improve the performance of systems within two application fields. Accordingly, the first block focuses on improving e-learning environments using optimized ML classification models while the second block concentrates on the use of optimized ML models towards more effective IDSs.

The first block proposes using different ML classification algorithms to offer a more personalized learning experience in an e-learning environment. More specifically, Chapters 4 and 5 propose the use of different optimized ML models to predict student's performance at earlier stages of the course delivery. The developed models use different ensemble classification techniques to categorize the students and predict their final performance group.

On the other hand, the second block of this thesis addresses the use of ML models for more effective and robust IDSs. As such, Chapters 6 and 7 propose different optimized ML classification frameworks using various optimization modeling algorithms and heuristics to improve the performance of the IDS through anomaly detection while maintaining or reducing its time complexity.

Note that the move from e-Learning to network security was mainly motivated by the scarcity and difficulty of obtaining education datasets. As an example, obtaining one of the educational datasets required close to fourteen months of communication to get all the required approvals from the university's ethics office and information privacy office.

1.4 Thesis Organization

The thesis consists of eight chapters.

Chapter 1 briefly introduces the field of machine learning. Moreover, it illustrates the importance of using ML models in two different applications, namely e-Learning and network security. Moreover, it summarizes the thesis contributions and provides its outline.

Chapter 2 presents the basic mathematical background of the utilized ML algorithms as well as the performance metrics used for evaluation and the concept of ensemble learning.

Chapter 3 provides a brief literature review of the challenges facing different fields such as education, healthcare, network security, banking and finance, and social media. Moreover, it presents several research opportunities on the role and potential of using ML to address these challenges.

Chapter 4 uses the comparative analysis gained from various classification algorithms to predict student's performance at earlier stages of the course. The developed models use a majority voting-based ensemble classification techniques to categorize the students and predict their final performance group with the purpose of identifying the weak students that may need help at earlier stages of the course delivery.

Chapter 5 extends the work from the previous chapter by considering the multi-class problem instead of the binary case and explores bagging-based ensemble classifiers to categorize the students into one of the three classes identified.

Chapter 6 proposes an effective intrusion detection framework based on optimized machine learning classifiers using Bayesian Optimization (BO).

Chapter 7 extends the work from the previous chapter by proposing a multi-stage optimized ML-based network intrusion detection framework that reduces the computational complexity while maintaining the detection performance.

Finally, Chapter 8 concludes the thesis, summarizes the findings, and presents various potential future research directions worth exploring.

1.5 Thesis Contributions

The contributions of this thesis can be summarized as follows:

1.5.1 Contributions of Chapter 3

1. Describes briefly the different challenges facing a variety of modern fields including education, healthcare, network security, banking and finance, and social media.

2. Presents some previous works that focused on these challenges and their shortcomings.

3. Discusses the role and potential of ML in addressing these challenges and presents potential frameworks for its deployment.

1.5.2 Contributions of Chapter 4

1. Analyze the collected datasets and their corresponding features using multiple graphical, statistical, and quantitative techniques (e.g. probability density function, decision boundaries, feature variance, feature weights, principal component analysis,etc.)

2. Conduct hyper-parameter tuning to optimize the parameters of the different ML algorithms under consideration using grid search algorithm.

3. Eliminate any bias in the optimized models through the use of multiple splits of the training and testing data at both course delivery stages under consideration.

4. Propose a systemic approach for building an ensemble learner to choose the best model based on multiple performance metrics, namely the Gini index and the p-value.

5. Evaluate the performance of traditional classification techniques compared to the proposed ensemble learner.

6. Identify students who may need help with high accuracy using the proposed ensemble learner.

1.5.3 Contributions of Chapter 5

1. Analyze the collected datasets and their corresponding features using multiple graphical and quantitative techniques (e.g. dataset distribution visualization, target variable distribution, and feature importance).

2. Optimize hyper-parameters of the different ML algorithms under consideration using *grid search* algorithm.

3. Propose a systemic approach to build a multi-split-based (to reduce bias) bagging ensemble (to reduce variance) learner to choose the best model based on multiple performance metrics, namely the Gini index (for better statistical significance and robustness) and the target class score.

4. Study the performance of the proposed ensemble learning classification model on a multi-class dataset in comparison with a binary classification model.

5. Evaluate the performance of traditional classification techniques compared to the proposed ensemble learner.

1.5.4 Contributions of Chapter 6

1. Investigate the performance of the optimized machine learning algorithms using Bayesian Optimization to detect anomalies.

2. Enhances the performance of the classification models through the identification of the optimal parameters towards objective-function minimization.

3. UNB ISCX 2012, a benchmark intrusion dataset is used for experimentation and validation purposes through the visualization of the optimization process of the objective function of the considered machine learning models to select the best approach that identifies anomalous network traffic. To the best of our knowledge, no previous related work has adopted Bayesian Optimization on the utilized dataset towards anomaly detection.

1.5.5 Contributions of Chapter 7

1. Propose a novel multi-stage optimized ML-based NIDS framework that reduces computational complexity and enhances detection accuracy.

2. Study the impact of oversampling techniques and determine the minimum suitable training sample size for effective intrusion detection.

3. Explore the impact of different feature selection techniques on the NIDS detection performance and time (training and testing) complexity.

4. Propose and investigate different ML hyper-parameter optimization techniques and their corresponding enhancement of the NIDS detection performance.

5. Evaluate the performance of the optimized ML-based NIDS framework using two recent state-of-the-art datasets, namely the CICIDS 2017 dataset [19] and the UNSW-NB 2015 dataset [20].

6. Compare the performance of the proposed framework with other works from the literature and illustrate the improvement of detection accuracy, reduction of FAR, and a reduction of both the training sample size and feature set size.

Chapter 2

Machine Learning Background

The term *Machine Learning* was coined by Arthur Lee Samuels in 1959 when he published a paper in the *IBM Journal of Research and Development* with his approach for playing checkers using a *self-learning program*, [21]. Since then, Machine Learning techniques have been widely used and a huge amount of money has been invested in startup software companies.

Machine Learning has become a very popular topic within development organizations that are looking for a way to improve their business by gaining useful information from the data. With machine learning models, organizations can continually predict changes in their business and make decisions accordingly. In fact, as data is added to their database, the predictive models built using machine learning techniques ensure that the solution is constantly updated.

Machine Learning uses algorithms that iteratively learn from data to improve, describe data and predict outcomes. Once a machine learning model has been trained on some data, it is able to make predictions on new data provided as an input. The output given by the model on the new data will depend on the data used to train the model.

In this chapter we are going to have a taste of some algorithms from the fascinating and interesting Machine Learning Theory. The idea behind most of the algorithms that we are going to describe is quite complex and very beautiful at the same time.

Various techniques can be used in data mining: classification, clustering, regression, association rules etc. [22], [23]. For the purpose of this study, several classification techniques were used: Support Vector Machine (SVM), Logistic Regression (LR), Multi-layer perceptron (MLP), Decision Trees (DT), k-nearest neighbors (k-NN), Naive Bayes (NB), and ensemble learners.

In this section, we present basic details of the used algorithms, among which we will select the best subsets of classifiers that will form the ensemble learners. In the following two sections we fix the notation and define the concepts that we are going to use in the next chapters.

2.1 Machine Learning Mathematical Background

In this brief section we define some mathematical objects that we will use throughout the rest of the chapter. For further details regarding the contents of this section please refer to [24], [25], [26], [27].

The *gradient* is a generalization of the derivative to multiple variables. More precisely, let

$f : \mathbb{R}^{m\times n} \to \mathbb{R}$ a map from the $m\times n$ matrices to the real numbers $\mathbb{R}$, then we define the gradient of f with respect to A as the matrix whose elements are the derivatives of f with respect to A. In other words, the gradient $\nabla_A f(A)$ is defined as:

$$\nabla_A f(A) = \begin{bmatrix} \frac{\partial f}{A_{11}} & \cdots & \frac{\partial f}{A_{1n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f}{A_{m1}} & \cdots & \frac{\partial f}{A_{mn}} \end{bmatrix} \tag{2.1}$$

Example Let A be the matrix $A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$ and $f(A) = 5A_{11}^2 + 3A_{12} + 4A_{22}$ then $\nabla_A f(A) = \begin{bmatrix} 10A_{11} & 3 \\ 0 & 4 \end{bmatrix}$.

Given an $n \times n$ matrix A we define the *trace* of A, denoted by $\mathbf{tr}(A)$, to be the sum of its diagonal entries:

$$\mathbf{tr}(A) = \sum_{i=1}^{n} A_{ii} \tag{2.2}$$

The *transpose* of a matrix A, denoted by $A^{\mathbf{T}}$ is a matrix whose rows are the columns of the original matrix A. If A is the 2×2 matrix from the previous example, then its trace is $\mathbf{tr}(A) = A_{11} + A_{22}$ and its transpose is $A^{\mathbf{T}} = \begin{bmatrix} A_{11} & A_{21} \\ A_{12} & A_{22} \end{bmatrix}$.

Given a square matrix $A \in \mathbb{R}^{n\times n}$, we say that a (complex) number λ is an *eigenvalue* of A and v is the corresponding *eigenvector* if

$$Av = \lambda v \tag{2.3}$$

2.2 Notation and Definitions

In this section we define some notation and terminology that we will use in the next chapters, [28], [29]. *Machine Learning* can be defined as computational methods that use experience, i.e. past information, to make predictions or improve performance, [30]. Nowadays, Machine Learning algorithms are successfully used in a variety of applications such as Image Recognition, Speech Recognition, Text classification, Fraud Detection, Medical diagnosis etc.

There are several types of machine Learning algorithms, such as *supervised* algorithms, *unsupervised*, *semi-supervised*, *reinforcement*, etc, [13].
In this chapter we focus primarily on supervised machine learning algorithms, more specifically classification algorithms. A *supervised* Machine Learning algorithm is a machine learning algorithm for which both the input variables X and the output variable y are given and the algorithm is used to *learn* a map from X to y in such a way that if a new input data is given, then the map can be used to predict the output variable for that data.

With the terms *features*, *labels* and *hypotheses function* we will refer to the input variables, the target variables and the mapping function that approximizes the output variables, respectively, [28].

Consider the supervised problem where $X = \{X^{(1)}, \cdots, X^{(n)}\}$ is the input variables, and

$y = \{y^{(1)}, \cdots, y^{(n)}\}$ the target variable we aim to predict. Then we call each row X_i of X *observation*, or *example*, and we refer to a pair (X,y) as a *sample*, [28], [30].

If $y^{(1)}, \cdots, y^{(n)}$ take continuous values in $\mathbb{R}$ then the learning problem is a *regression* problem, otherwise if $y^{(1)}, \cdots, y^{(n)}$ take a small number of discrete values then the learning problem is a *classification* problem, [28]. A classification problem in which the target variable can take only two values, say 0 and 1, is called *binary classification* problem and the two possible values are often called *classes*. If y can take more than two values, say K, then the problem is called *multi-class classification* problem and the target variable for the i-th observation X_i is a vector of the form $(a_1(i)), \cdots, a_K(i))$ where only one of the entries is equal to 1 whereas the others are equal to 0, [28], [30].

The data used to to build a Supervised Machine Learning model is called *training sample*. The model developed is used to predict the responses for the observations in a second dataset called *test sample*. In particular, the output of a binary classification model consists of the assignment to each observation X_i of the probability that X_i belongs to a class C. For a multi-class classification problem with K classes $C_1, \cdots, C_K$, the model associates to each observation a K-vector whose enters are the K probabilities to belong to class C_1,..., class C_K respectively, [28].

There are several Machine Learning algorithms that can be used to solve classification (and regression) problems. Once an algorithm is chosen and the model is implemented, different *metrics* can be used to evaluate the perfomance of the model, such as Gini Index, accuracy, precision, recall, Log-Loss, AUC (Area Under the Curve), etc, [31]. The purpose of evaluating the model is to maximize the success rate of the predictions. The outcome for each test observation instance can be either *correct*, if the prediction agrees with the actual values or *incorrect*, if it does not. It is fundamental to keep in mind that the choice of the metric to be used to evaluate the performance is strictly related to the problem itself and its applications.

2.3 Performance Evaluation Criteria

Once a binary classification model is trained, its performance is tested on a test sample and this procedure is called *inference*. The output of the inference is a score associated to each observation that indicates the probability for the observation to be of class 1. Given the score vector, we set a *threshold* τ such that if the observation's score is greater or equal to τ then the observation is predicted as 1, otherwise its prediction is 0.

A *confusion matrix* is a matrix that compares the *actual* class and the *predicted* class of the target variable. More precisely, a confusion matrix is a matrix of the form [31]:

$$\left[\begin{array}{c|c} TP & FP \\ \hline FN & TN \end{array}\right] \tag{2.4}$$

and the terms associated are:

- *True Positives (TP)*: True positives are the cases when the actual class of the data point was 1 (True) and the predicted is also 1 (True)

- *True Negatives (TN)*: True negatives are the cases when the actual class of the data point was 0 (False) and the predicted is also 0 (False)

- *False Positives (FP)*: False positives are the cases when the actual class of the data point was 0 (False) and the predicted is 1 (True). In this case the model has predicted incorrectly and positive because the class predicted was a positive one.

- *False Negatives (FN)*: False negatives are the cases when the actual class of the data point was 1 (True) and the predicted is 0 (False). In this case the model has predicted incorrectly and negative because the class predicted was a negative one.

With the notation above we define the *accuracy* of the model as percentage of correct predictions out of all the predictions made:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.5}$$

The accuracy is a good metric when the target variable is balanced, i.e. the number of elements of each class is similar. On the contrary, when the target variable is unbalanced (for instance, consider a problem such as fraud detection where the number of cases of fraudulent transactions is very small compared with the total number of transactions) the accuracy should not be used as metric.
The *Precision* measures the proportion of the true positive out of the total number of elements labeled as belonging to the positive class. In other terms the precision can be calculated as:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.6}$$

In the previous example, as one would be only interested in knowing how many fraudulent transactions there are, precision would be a better estimate of the performance rather than the accuracy that would take into account also the true negative cases.
The *Recall* or *Sensitivity*, also called *TPR (True Positive Rate)*, measures the number of true positives divided by the total number of elements that actually belong to the positive class:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.7}$$

The choice of the metric depends on whether the objective is to minimize the false negative (then Recall is preferred as opposed to Precision) or to to minimize the false positives (in which case Precision is a better metric).
In analogy with the definition of Recall, we define the *Specificity* as:

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{2.8}$$

and the *FPR/FAR (False Positive/Alarm Rate)* as $1 - \text{Specificity}$.
A good compromise between Precision and Recall is the *F1 Score*, also called *F-Measure*, that is defined as:

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.9}$$

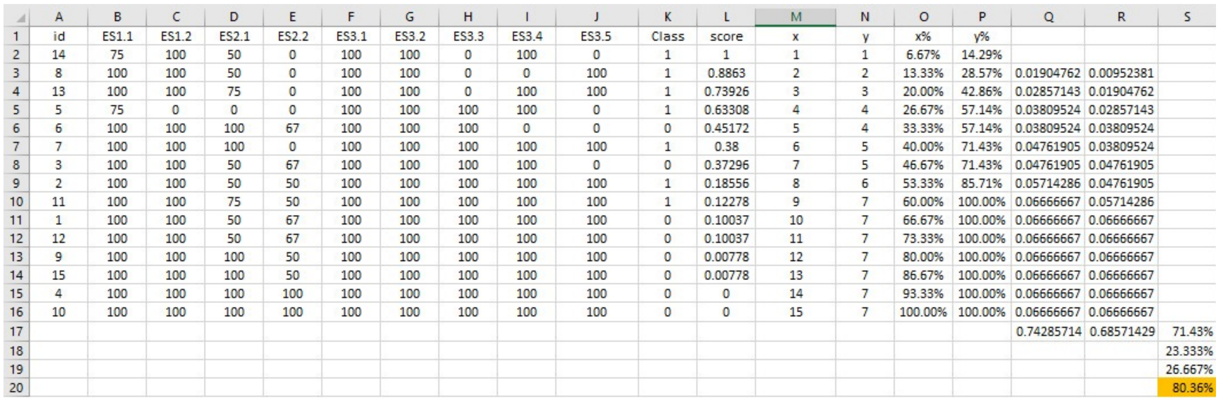
Sensitivity and Specificity are inversely proportional to each other. So when we increase Sensitivity, Specificity decreases and vice versa.

The metric that will be used to evaluate the performance of the classifiers is the *Gini Index*, [32]. The Gini Index can be seen geometrically as the area between the Lorenz curve [33] and the diagonal line representing perfect equality. The higher the Gini Index, the better the performance of the model. Formally the Gini index is defined as follows. Let $F(z)$ be the cumulative distribution of z and let a and b be the highest and the lowest value of z respectively, then the we can calculate half of Gini's expected mean difference as

$$2\int_a^b F(z)[1-F(z)]dz \tag{2.10}$$

To make the definition clear, we construct and calculate the Gini Index by hand in the following Example, then we fix a threshold and show the corresponding confusion matrix and the performance values associated with it.

Example Suppose we have built a model and we want to evaluate its performance. In Figure 2.1, the test sample is shown from column B to column J, the target variable is in column K and the output of the model consists of a score, as in column L. This score represents the probability for each observation to be of class 1. The table was ordered in a descending order with respect to the score, that is in such a way that on top we find the observations which are more likely to have target variable equal to 1 and at the bottom the less likely to have target variable equal to 1.

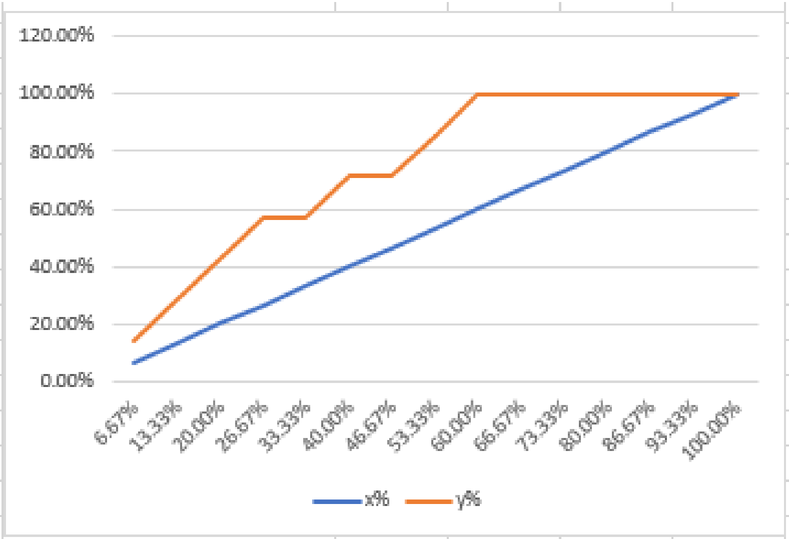


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	id	ES1.1	ES1.2	ES2.1	ES2.2	ES3.1	ES3.2	ES3.3	ES3.4	ES3.5	Class	score	x	y	x%	y%			
2	14	75	100	50	0	100	100	0	100	0	1	1	1	1	6.67%	14.29%			
3	8	100	100	50	0	100	100	0	0	100	1	0.8863	2	2	13.33%	28.57%	0.01904762	0.00952381	
4	13	100	100	75	0	100	100	0	100	100	1	0.73926	3	3	20.00%	42.86%	0.02857143	0.01904762	
5	5	75	0	0	0	100	100	100	100	0	1	0.63308	4	4	26.67%	57.14%	0.03809524	0.02857143	
6	6	100	100	100	67	100	100	100	0	0	0	0.45172	5	4	33.33%	57.14%	0.03809524	0.03809524	
7	7	100	100	100	0	100	100	100	100	100	1	0.38	6	5	40.00%	71.43%	0.04761905	0.03809524	
8	3	100	100	50	67	100	100	100	100	0	0	0.37296	7	5	46.67%	71.43%	0.04761905	0.04761905	
9	2	100	100	50	50	100	100	100	100	100	1	0.18556	8	6	53.33%	85.71%	0.05714286	0.04761905	
10	11	100	100	75	50	100	100	100	100	100	1	0.12278	9	7	60.00%	100.00%	0.06666667	0.05714286	
11	1	100	100	50	67	100	100	100	100	100	0	0.10037	10	7	66.67%	100.00%	0.06666667	0.06666667	
12	12	100	100	50	67	100	100	100	100	100	0	0.10037	11	7	73.33%	100.00%	0.06666667	0.06666667	
13	9	100	100	100	50	100	100	100	100	100	0	0.00778	12	7	80.00%	100.00%	0.06666667	0.06666667	
14	15	100	100	100	50	100	100	100	100	100	0	0.00778	13	7	86.67%	100.00%	0.06666667	0.06666667	
15	4	100	100	100	100	100	100	100	100	100	0	0	14	7	93.33%	100.00%	0.06666667	0.06666667	
16	10	100	100	100	100	100	100	100	100	100	0	0	15	7	100.00%	100.00%	0.06666667	0.06666667	
17																	0.74285714	0.68571429	71.43%
18																			23.333%
19																			26.667%
20																			80.36%

Figure 2.1: Example of Gini Index Computation

Column M is a count of the rows, and column N is a cumulative sum of the 1's; for example, the 8-th observation has value equal to 6 in column N because there are 6 1's counted from row 1 to row 8. Columns O and P are obtained by transforming columns M and N respectively into percentages and represent the points of the Lorenz curve. If we join the points in columns O and P we obtain the Lorenz curve as in Figure 2.2. In order to evaluate the performance of the model on the test sample we need to calculate the area between the orange curve (Lorenz curve) and the blue curve (the latter representing the absence of a model). To do so, we need to calculate the integral of the curve and subtract 1/2, i.e. the area underneath the blue curve: we obtain an approximation of the area by splitting the graph into several intervals, building rectangles above and below the red curve, and finally summing up the averages of the areas above and below the Lorenz curve. Columns Q and R show the cumulative sums of the averages of the areas of the rectangles. For instance in the figure we see that the rectangles built above the curve have area that sum up to 0.74 whereas the total area of the rectangles built below the curve is 0.68. The average of the areas of the two types of rectangles is 0.71. Half of the average of 1's in the target variable is expressed by 23.3% and so the Gini Index is calculated as

$$\frac{71.43\% - 50\%}{50\% - 23.3\%} = 80.4\%.$$



	1s	n	%1s	delta
top 30% (more likely W)	4	4	100.0%	**100%**
30-60%	3	5	60.0%	**20%**
60-80%	0	3	0.0%	**-100%**
bottom 20% (more likely G)	1	4	25.0%	**-50%**
Total	8	16	50.0%	

Figure 2.2: Example of Gini Index plot and Stats

In particular we can see that

- in the top 30% of the table, corresponding to four rows, we have four 1's
- in the portion 30%-60% of the sample, i.e. from row 5 to row 9, 60% of the corresponding target variable are 1's (three 1's out of five rows)
- in the portion 60%-80% of the sample, i.e. from row 10 to row 12, 0% were 1's (zero 1's out of three rows)
- in the bottom 20% of the table, corresponding to rows 13-16, we have one 1, i.e. 25% of the bottom 20% were 1's.

Considering that the average of the target variable in this example is 50% (half are zeros and half are ones), then the blue line, i.e. prediction made by chance, would expect 30% of 1's to be in the top 30%; using the model, 100% of 1's are identified in the top 30%, and this means that the 1's identified by the model are doubled with respect to the expected value.

Setting the threshold $\tau = 0.38$, we obtain that the first 6 rows are predicted as 1 and the rest are predicted as 0. The confusion matrix corresponding to τ is:

	1	**0**
1	5	2
0	1	7

Table 2.1: Confusion matrix corresponding to threshold $\tau = 0.38$

And we can calculate Accuracy, Precision, etc using the formulas in the section obtaining the valus in Table 2.2.

Accuracy	**Precision**	**Recall**	**F-Measure**	**False Positive Rate**
80.0%	71.4%	83.3%	76.9%	22.2%

Table 2.2: Performance corresponding to $\tau = 0.38$

It is crucial to note that the main reason to use the metric Gini Index instead of the accuracy is that the second is strongly affected by the threshold set on the probabilities.

All the metrics defined so far can be generalized to multi-class classification problems with K classes by using the "One vs ALL" approach, i.e. considering each time one of the possible outcomes (target=1) against all the other possible outcomes as if they were labelled in the same way (target=0). Then, depending on the problem, one strategy could be to average the K comparisons obtaining an average Gini Index, an average accuracy, average AUC - ROC curve, etc or, alternatively, to evaluate each individual performance and make judgments depending on the importance of class (if they are not equally important).

2.4 Principal Component Analysis

Principal Component Analysis (PCA) is a statistical technique that aims to identify the subspace in which the data approximately lies, [28]. PCA transforms a set of observations of possibly correlated n variables into a set of m *linearly uncorrelated* variables called *principal components*, with $m < n$. The first step consists of pre-processing the data:

1. Define $\mu = \frac{1}{n}\sum_{i=1}^{n} X^{(i)}$
2. Replace each $X^{(i)}$ with $X^{(i)} - \mu$
3. Define $\sigma_j^2 = \frac{1}{n}\sum_{i=1}^{n}(X_j^{(i)})^2$

4. Replace each $X_j^{(i)}$ with $\frac{X_j^{(i)}}{\sigma_j}$

Suppose we have some data X in $\mathbb{R}^2$ and let u be a unit vector whose direction is the one on which the data approximately lies, then the projection of a point $X^{(i)}$ onto u is given by $(X^{(i)})^{\mathbf{T}}u$. Consequently to maximize the variance of the projections, we need to find u such that the following quantity is maximized:

$$\frac{1}{n}\sum_{i=1}^{n}\left[(X^{(i)})^{\mathbf{T}}u\right]^2 = u^{\mathbf{T}}\left(\frac{1}{n}\sum_{i=1}^{n}X^{(i)}(X^{(i)})^{\mathbf{T}}\right)u \tag{2.11}$$

Such vector u is an eigenvector of $\Lambda = \frac{1}{n}\sum_{i=1}^{n} X^{(i)}(X^{(i)})^{\mathbf{T}}$, [28]. We can generalize this construction to data in $\mathbb{R}^n$, where we aim to find a k-dimensional subspace, with $k < n$, where we can project our data onto. To do so, let $u_1, \cdots, u_k$ be the top k eigenvector, also called *principal components*, then they form an orthogonal basis for the data, [28], and instead of the original n-dimensional data consider the k-dimensional approximation $\widehat{X}$ of X. Each vector $\widehat{X}^{(i)}$ of the new dataset $\widehat{X}$ can be calculated as

$$\widehat{X}^{(i)} = \begin{bmatrix} u_1^{\mathbf{T}}X^{(i)} \\ u_2^{\mathbf{T}}X^{(i)} \\ \vdots \\ u_k^{\mathbf{T}}X^{(i)} \end{bmatrix} \tag{2.12}$$

2.5 Logistic Regression

Logistic Regression is used for predicting multi-class dependent variables by building a model that predicts the odds of occurrence of the event [34]. Logistic regression can be used to analyze the given dataset, since the output is measured with a dichotomous variable. It is a good choice, since the goal of logistic regression is to find the best fitting model to describe the relationship between the dichotomous characteristic of interest and a set of independent (predictor) variables. Logistic regression generates the coefficients and their standard errors and significance measures in order to predict a logit transformation of the probability of presence of the characteristic of interest. It is a very powerful algorithm [35] that was used in various fields from agriculture [36] to prevention of accidents [37].

A *logistic function* or *sigmoid functiong* is defined as $g(z) = \frac{1}{1+e^{-z}}$. The sigmoid function tends to 0 as $z \to -\infty$ and tends to 1 as $z \to -\infty$, see Fig. 2.3. The derivative of g is equal to:

$$g'(z) = \frac{1}{1+e^{-z}}\left(1 - \frac{1}{1+e^{-z}}\right) = g(z)(1 - g(z)). \tag{2.13}$$

We define the hypothesis function for the logistic regression as

$$h_\theta(X) = g(\theta^{\mathbf{T}}X) \tag{2.14}$$

where g is the sigmoid function and note that h_θ is bounded between 0 and 1 by construction.

In this section we will use the convention that $X_0 = 1$ so that we can write $\theta^{\mathbf{T}}X = \theta_0 + \sum_{j=1}^{N}\theta_j X_j$ and we will assume that

$$\mathbf{P}(y = 1 \mid X; \theta) = h_\theta(X) \tag{2.15}$$

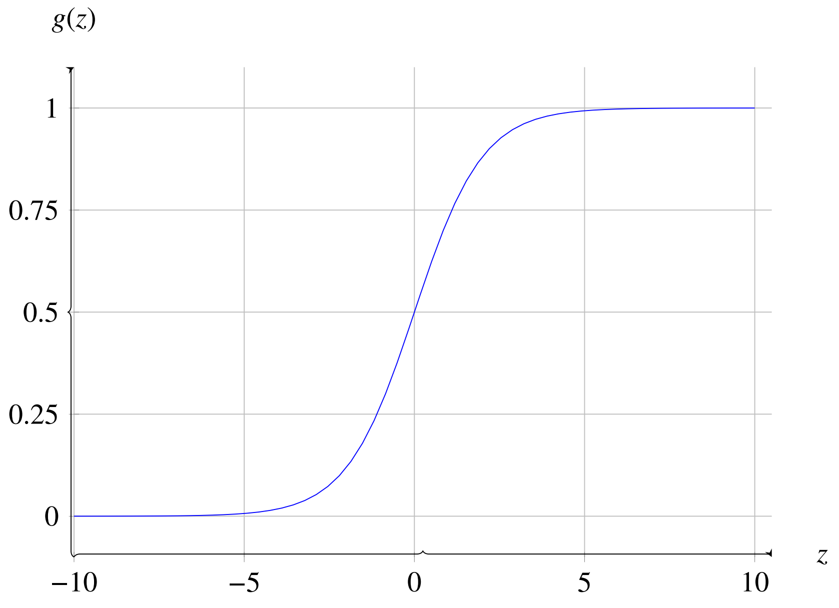


Figure 2.3: Sigmoid function

$$\mathbf{P}(y = 0 \mid X; \theta) = 1 - h_\theta(X) \tag{2.16}$$

which can be written as $\mathbf{P}(y \mid X; \theta) = (h_\theta(X))^y (1 - h_\theta(X))^{1-y}$. The goal is to maximize the following quantity that represents the likelihood of the parameters θ:

$$L(\theta) = \mathbf{P}(y \mid X; \theta) = \prod_{i=1}^{N} \mathbf{P}(y_i \mid X; \theta) = \prod_{i=1}^{N} (h_\theta(X^{(i)}))^y (1 - h_\theta(X^{(i)}))^{1-y}. \tag{2.17}$$

which is equalent to maximizing the following:

$$l(\theta) = \log L(\theta) = \sum_{i=1}^{N} y_i \log h(X^{(i)}) + (1 - y_i) \log(1 - h(X^{(i)})). \tag{2.18}$$

One can verify that the following identity holds [28]:

$$\frac{\partial}{\partial \theta_j} l(\theta) = (y - h_\theta(X)) X_j \tag{2.19}$$

Gradient ascent is a first-order iterative optimization algorithm for finding the maximum of a function. Let us use the gradient ascent algorithm, which starts at some initial θ, and repeatly performs the update for all $j = 1, \cdots, n$ simultaneously until convergence

$$\theta_j := \theta_j + \alpha(y_i - h_\theta(X^{(i)})) X_j^{(i)} \tag{2.20}$$

where α is called *learning rate*. If α is too small, then the gradient ascent is very slow, if α is too large then the gradient descent migh overshoot the maximum. Some advantages of logistic regression are that it is a very simple algorithm with low variance, it is not very prone to over-fitting and can directly optimize multinomial problems [38].

2.6 Support Vector Machine

This section presents the basic concepts of the *Support Vector Machine (SVM)* learning algorithm [39]. This algorithm performs well with small dataset and can be used for offline clustering [40]. A Support Vector Machine (SVM) is a classification algorithm that is based on the construction of N-dimensional hyperplane that separates the data in two categories [41]. SVM is a generalized linear model that is based on linear combination of features [34].

In this section, to make the notation easier, we assume the target variable y to be binary with values in $\{-1, 1\}$; let $X^{(1)}, \cdots, X^{(n)}$ be the set of features, and let $X_j = (X_j^{(1)}, \cdots, X_j^{(n)})$ be the j-th row of the sample. We want to find the *maximum-margin hyperplane* that divides the group of points $\{X_j\}$ for which $y_i = 1$ from the group of points for which $y_i = -1$, which is defined so that the distance between the hyperplane and the nearest point from either group is maximized [34]. Two sets of points $S_1$and S_2 in the N-dimensional Euclidean space are *linearly separable* if there exist $w_1, \cdots, w_N, k$ such that

$$\sum_{j=1}^{N} w_i X_i^{(j)} > k \quad \text{for every} \quad X_i \in S_1 \text{ and } \sum_{j=1}^{N} w_i X_i^{(j)} < k \quad \text{for every} \quad X_i \in S_2 \tag{2.21}$$

i.e. if there exists at least one line in the plane that separate them. In the case of *Linear SVM*, any hyperplane can be written as the set of points X satisfying $w \cdot X - b = 0$, where the vector w is orthogonal to the hyperplane. Let the training data be linearly separable, and select two parallel hyperplanes that separate the two classes of data in such a way that the distance between them is as large as possible. The region between the hyperplanes is called *margin*. If the dataset is normalised then the two hyperplane have equations

$$w \cdot x - b = 1 \qquad \text{and} \qquad w \cdot x - b = -1 \tag{2.22}$$

so we can define the classifier $h_{w,b}$ to be:

$$h_{w,b}(X_i) = \mathbf{sgn}(w \cdot X_i - b) \tag{2.23}$$

Note that the distance between the hyperplanes is given by $\frac{2}{||w||}$ so maximizing the marging is equavalent to minimizing $||w||$ and is equivalent to solving the optimization problem that consists of finding

$$\min_{w,b} \frac{1}{2} ||w|| \qquad \text{such that} \qquad \mathbf{sgn}(wX_i + b) \geq 1 \; i = 1, \cdots, N. \tag{2.24}$$

SVM is based on the assumption that the larger the distance between these hyperplanes, the lower the generalization error of the classifier will be [34], [39]. The concept of data separability can be generalized to the case where the training data can be separated in two regions by a curve that is not a line. In this case instead of the dot product between X and w, a function called *Kernel* is defined. The variables X, also named as *input attributes*, are mapped to some new set of quantities, called *input features*, that are then passed to the learning algorithm. The process of choosing the most suitable representation is called *feature selection*. Let φ be the *feature mapping* which maps from the attributes to the features, then instead of than applying SVMs using the original input attributes X, we may instead want to learn using some features $\varphi(X)$. Given a feature mapping φ, we define the corresponding *Kernel* to be:

$$\mathbf{K}(X_i, X_j) = \varphi(X_i)^{\mathbf{T}}\varphi(X_j) \tag{2.25}$$

Some common kernels include:

- Homogeneous Polynomial: $\mathbf{K}(X_i, X_j) = (X_i \cdot X_j)^d$
- Inhomogeneous Polynomial: $\mathbf{K}(X_i, X_j) = (X_i \cdot X_j + 1)^d$
- Hyperbolic tangent: $\mathbf{K}(X_i, X_j) = \tanh(\lambda X_i \cdot X_j + c)$ for some $\lambda > 0$ and $c < 0$
- Gaussian radial basis function (RBF): $\mathbf{K}(X_i, X_j) = e^{-\gamma||X_i - X_j||^2}$ for $\gamma > 0$. In particular the Gaussian radial basis function is a reasonable measure of X_i and X_j similarity: $\mathbf{K}(X_i, X_j)$ is close to 1 if the points X_i and X_j are close and is close to 0 if they are far apart.

2.7 Artificial Neural Network

Artificial Neural Network are a very complex and powerful approach used in machine learning algorithms. The concept of the artificial neural network was inspired by human biology and the way neurons of the human brain function together to understand inputs from human senses [42], [43].

An artificial neural network is a network of simple elements called *neurons*, which receive input, change their internal state (*activation*) according to that input, and produce output depending on the input and activation. To describe neural networks, we begin from the simplest example where there is only one neuron. This neuron takes as input the features $X^{(1)}, \cdots, X^{(n)}$ and an extra column $X^{(0)}$ called *intercept term* and computes the hypothesis function that is defined as follows

$$h_{\theta,b}(X_j) = g(\theta^{\mathbf{T}} X_j) = g\left(\sum_{i=i}^{n} \theta_i X_j^{(i)} + b\right) \tag{2.26}$$

where X_j is a point of the dataset, θ, b are two parameters, and g is the *activation function*. The most two common activation functions are:

- the sigmoid function: $g(z) = \frac{1}{1+e^{-z}}$ (see Section 2.5 for further details);
- the hyperbolic tangent: $g(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$

More generally, suppose we have more than one neuron: the input features are called collectively *input layer*. The network forms by connecting the output of certain neurons to the input of other neurons forming a directed, weighted graph, see figure 2.4. The weights as well as the functions that compute the activation can be modified by a process called learning which is governed by a *learning rule* [42]. The learning rule is a rule or an algorithm which modifies the parameters of the neural network, in order for a given input to the network to produce a favored output. This learning process typically amounts to modifying the weights and thresholds of the variables within the network [42].

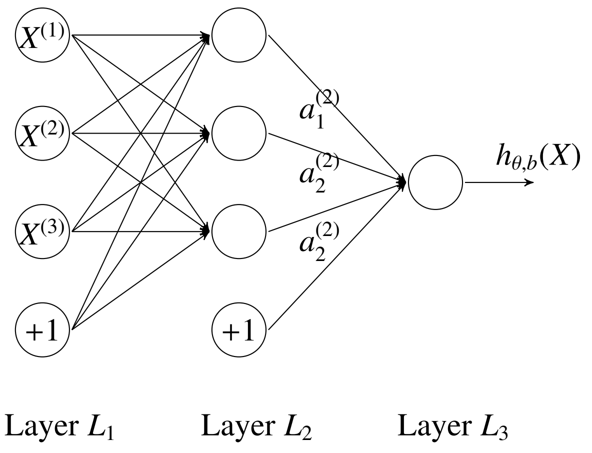


Figure 2.4: Example of Artificial Neural Network

In Figure 2.4, the circles that were labeled ”+1” are called *bias units*, and correspond to the intercept term; moreover, going from left to right in the image we see the input layer as first layer (already defined), the middle layer which is called *hidden layer*, and the last layer which is called *output layer*. Generally speaking, the number of hidden layers may vary, but usually only one is used [44].

Multi-Layer Perceptron (MLP) is an algorithm that belongs to the group of *feedforward neural network* classification, i.e. neural networks for which the connections between the nodes do not form a cycle [42].

2.8 k-Nearest Neighbors (k-NN)

The K-Nearest Neighbor algorithm (k-NN) is a classification technique that classifies an object “by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors” [45]. This algorithm is based on a *distance function* that is a function that defines a distance $d\colon X \times X \to \mathbb{R}^+$ between each pair of points X_i, X_j of a set X. A set endowed with a distance is called *metric space*. A distance satisfies the following conditions:

1. $d(X_i, X_j) \geq 0$ for all $X_i, X_j \in X$
2. $d(X_i, X_j) = 0$ if and only if $X_i = X_j$ for all $X_i, X_j \in X$
3. $d(X_i, X_j) = d(X_j, X_i)$ for all $X_i, X_j \in X$
4. $d(X_i, X_j) \leq d(X_j, X_k) + d(X_i, X_k)$ for all $X_i, X_j, X_k \in X$

An example of distance is the standard Euclidean distance [46]. The following formula represents the Euclidean distance between two points, X_i, X_j, with n attributes [47]:

$$d(X_i, X_j) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2} \tag{2.27}$$

The k-NN algorithm assume that similar attributes exist in close proximity, and so captures the idea of similarity between points by assigning to each point a certain label corresponding to one of k neighbor, where k is a fixed positive integer. Once the distance is established, for each point X_i in the data the algorithm computes the distance between X_i and every other point and sort the point by the distances from X_i, i.e. creates a chain of the type

$$d(X_i, X_1) \leq d(X_j, X_2) \leq d(X_j, X_3) \leq \cdots \tag{2.28}$$

and pick the first k points $X_1, \cdots, X_k$ from the sorted chain. The predicted neighbor for a point is identified as the most frequent label collected whilst computing the distances.

The accuracy of the k-NN algorithm can be easily affected by the presence of noise, with large values of k reducing the effect of noise on the classification.

The biggest advantage of k-NN algorithm is that it is a very simple classifier. On the other hand, computation cost is quite high because it needs to compute the distance of each instance to all training samples, it highly depends on distance measure [40], and needs to determine the value of parameter k, which is the number of nearest neighbors, [48].

2.9 Naïve Bayes

Two variables are *conditionally independent* if they have no direct impact on each other's values. A good way to represent the conditional independencies among variables is given by graphs. A *graphical model* shows any intermediary variables that saparate two conditionally independent variables. In a graphical model, a set of *nodes* represent the variables and the *edges* connecting the nodes represent the relationship between the corresponding variables. Edges can have a direction assigned which represent the causal relationship between the variables; for directed edges, the edge is said to be from *parent*, to *child* or, equivalently from the cause variable to the effect variable. A graphical model is *acyclic* if there are no cycles, i.e. there are no edges going in both directions between two nodes.

For each variable, there is a probability distribution function whose definition depends on the edges leading into the variable. We define graphical model to be a set of variables (nodes), dependencies (edges) and probability distribution functions (one for each variable). Examples of graphical models are causal networks, probabilistic independence networks, Markov Fields.

Another example of graphical models is represented by *Bayesian Networks* that are graphical models for which the graph is *directed* and *acyclic*. By the *Chain Rule for Bayesian Networks* [49], if $X = \{X^{(1)}, \cdots, X^{(n)}\}$ is the set of nodes for a Bayesian Network, then the *joint probability function* is given by

$$\mathbf{P}(X) = \prod_{i=1}^{n} \mathbf{P}(X^{(i)}|\text{parent}(X^{(i)})) \tag{2.29}$$

where the notation $\mathbf{P}(X^{(i)} \mid \text{parent}(X^{(i)}))$ is the called *conditional probability* of the event $x^{(i)}$ given the event parent$(X^{(i)})$ and is equal to the probability that both event occur over the probability that only parent$(X^{(i)})$ occurs. If the probability that both occur is zero, then also $\mathbf{P}(X^{(i)}|\text{parent}(X^{(i)})) = 0$.

One subclass of Bayesian Networks is the class called as *Naïve Bayes* [50]. The Naïve

Bayes algorithm is a classifier based on Bayes Theorem that calculates the probability by counting the frequency and combinations of values in a given dataset [51]. The Naive Bayes model makes the assumption that every pair of features $X^{(i)}$ and $X^{(j)}$ are conditionally independent, given the class. The classification decision is made based upon the *maximum-a-posteriori* rule i.e. the the most probable hypothesis is the one to be picked; in other words, this choice is made by defining a *Bayesian Classifier*. There are few steps for building a Bayesian classifier [51]:

1. Collecting class exemplars,
2. Estimating class a priori probability,
3. Estimating class means,
4. Forming covariance matrixes and finding the inverse and determinant for each, and
5. Forming the discriminant function for each class.

Formally, a Bayesian Classifier is a function h that assigns to each sample with features $X_i = \{X_i^{(1)}, \cdots, X_i^{(n)}\}_{i=1,\cdots,N}$, a class label $\bar{y}_i = C_j(i) \in \{C_1, \cdots, C_K\}$, where C_j are the possible classes for $j = 1, \cdots, K$:

$$h(X_i) = \text{argmax}_{j\in\{1,\cdots,K\}} \mathbf{P}(C_j)\mathbf{P}(X_i^{(1)}, \cdots, X_i^{(n)}|C_j) \tag{2.30}$$

and since the features are independent given the class value, $\mathbf{P}(X_i^{(1)}, \cdots, X_i^{(n)}|C_j)$ is equal to $\prod_{l=1}^{n} \mathbf{P}(X^{(l)}|C_j)$ and therefore the maximum posterior classification is given by:

$$h(X_i) = \text{argmax}_{j\in\{1,\cdots,K\}} \mathbf{P}(C_j) \prod_{l=1}^{n} \mathbf{P}(X_i^{(l)}|C_j). \tag{2.31}$$

One advantage of the Naïve Bayes algorithm is that it does not require large amount of data for accurate predictions. This algorithm is also fast and can handle discrete and continuous attributes, performs well in real-life problems and can be easily interpreted [51]. On the other hand, in situations where there is a strong dependency among attributes, Naïve Bayes usually has a deficient performance.

2.10 Random Forest

Random Forest as a modeling technique may also prove to be a good choice. Random Forest is a group of untrimmed classification trees made by utilizing bootstrap records from the training set and arbitrary feature assortment in tree initiation [52]. Forecasting is developed through combining (mainstream vote or averaging) the forecasts of the collective.

Random forest operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of average prediction of the individual trees, thus making this approach a possible candidate for good results as well. The random-forest algorithm brings extra randomness into the model, when it is growing the trees. Instead of searching for the best feature while splitting a node, it searches for the best feature among a random subset of features. This process creates a wide diversity, which generally results in a better model. Random Forest prevents over-fitting most of the time, by creating random subsets of the features and building smaller trees using these subsets.

The main limitation of Random Forest is that a large number of trees can make the algorithm too slow and ineffective for real-time predictions. From a training set X with answers Y, random forest recurrently chooses an arbitrary example with substitution of the training set and it is fitting trees to chosen samples:
For the specified amount of iterations N do

1. Perform sampling with substituting n training samples from the training and test sets

2. Denote that samples by $X1$, $Y1$ and train a classifier tree f_n on the sets $X1$, $Y1$.

2.11 Ensemble Learning

An *ensemble* consists of a set of individually trained classifiers whose predictions are combined when classifying novel instances [53]. Previous research has shown that an ensemble is often more accurate and robust than any of the single classifiers in the ensemble. Indeed, the combination of independent base learners will lead to a dramatic decrease of errors [54].

Bagging [54] and *Boosting* [55] are the most two popular methods for producing ensembles. Figure 2.5 gives a good description of how an ensemble works.

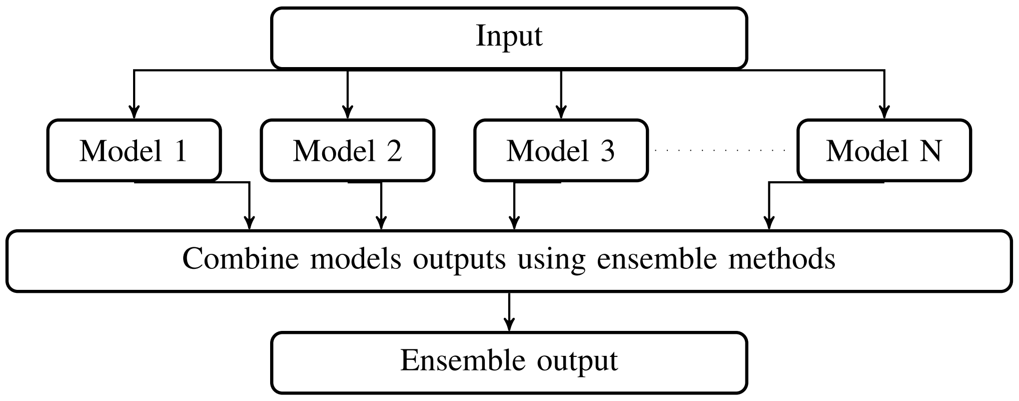


Figure 2.5: Ensemble Method

As shown in the figure, N models are trained, then for each example, the predicted output of each of these models is combined to produce the output of the ensemble. Some ensemble methods are:

- *Boosting* is a general method for improving the performance of a weak learner (such as decision trees) [55]. Intuitively a *weak learner* is very close to random guess, while a *strong learner* is very close to perfect performance. An example of boosting is *Adaboost*, [55]. Boosting consists of iteratively learning weak classifiers with respect to a distribution and adding them to a final strong classifier. When they are added, they are typically weighted in some way that is usually related to the weak learners' accuracy. After a weak learner is added, the data weights are readjusted [54].

- *Bagging* is another ensemble method. Given a training set X of size n, bagging generates m new training sets $\widehat{X_1}, \cdots, \widehat{X_m}$, by sampling from X uniformly and with replacement. By sampling with replacement, some observations may be repeated in each $\widehat{X_i}$. Then, m models are fitted using the m samples and combined by averaging the output [54].

- *Stacking of models* involves training a learning algorithm to combine the predictions of several other learning algorithms. First, all of the other algorithms are trained using the available data, then a combiner algorithm is trained to make a final prediction using all the predictions of the other algorithms as additional inputs, [56].

- A *Bucket of models* is an ensemble technique in which a model selection algorithm is used to choose the best model for each problem. When tested with only one problem, a bucket of models can produce no better results than the best model in the set, but when evaluated across many problems, it will typically produce much better results, on average, than any model in the set, [57].

- A *Voting based method* is an ensemble method that consists of counting and comparing, for every observation, the number of times that the observation was classified to belong to each class of the target variable. The ensemble learner will then assign to that observation the predicted class which correspond to the one that had appeared more frequently among the models forming the ensemble.

Chapter 3

Machine Learning Towards Intelligent Systems: Applications, Challenges, and Opportunities

3.1 Introduction

Large amounts of data are being collected by individuals, organizations, and society due to the rapid growth of the Internet and related technologies [1]. However, this often leads to information overload [2]. Information overload occurs when the amount of input (e.g. data) being processed by a human exceeds their cognitive capacities [2]. In turn, this can lead to humans ignoring, overlooking, or misinterpreting crucial information [7].

The discipline of data science has emerged as a potential paradigm to address this issue. This discipline combines the classic disciplines of statistics, data mining, databases, and distributed systems in order to extract information from large sets of data [1]. Within this discipline, machine learning (ML) is one method of data analysis that data scientists can implement. ML allows computers to learn without being explicitly programmed. Once the computer learns patterns from a training set of data, it can apply what it has learned to find these patterns in similar data [8]. Furthermore, ML allows computer systems to adapt and learn from their experience [9, 10].

ML algorithms have several applications. This includes house pricing prediction, spam filtering, education, structuring of data in healthcare systems, drug response prediction, diabetes research, network security, banking and finance, and social media. This work aims to provide a brief literature review of the challenges facing different fields such as education, healthcare, network security, banking and finance, and social media. Moreover, it presents several research opportunities on the role and potential of using ML to address these challenges. Hence, the contributions of this chapter are summarized as follows:

- Describes briefly the different challenges facing a variety of modern fields including education, healthcare, network security, banking and finance, and social media.
- Presents some previous works that focused on these challenges and their shortcomings.

A version of this chapter has been submitted in [58].

- Discusses the role and potential of ML in addressing these challenges and presents potential frameworks for its deployment.

Note that these fields were chosen since ML has been heavily explored and investigated to address many of the challenges observed within them. The remainder of this chapter is organized as follows: Section 3.2 discusses the education field. Section 3.3 focuses on the healthcare system and field. Section 3.4 presents the challenges in network security and the potential role of ML in addressing these challenges. Section 3.5 sheds light on the banking and finance sector. Section 3.6 focuses on the area of social media. Finally, Section 3.7 concludes the chapter.

3.2 Education

The first application considered is that of the education sector. There are three main ways that education can be delivered: onsite, online, and blended learning [13]. Onsite education, or traditional education, refers to educational content delivery within a traditional classroom setting [13]. The traditional classroom setting requires that the educator and the students are in the same room at the same time. This allows the educator to deliver his/her lecture to the attending class. As such, traditional classrooms provide face-to-face interaction between the educator and the students [14].

On the other hand, online education, one category of e-learning systems, refers to education that is provided over the Internet [13]. E-learning provides students with the opportunity to access educational curriculum outside of a traditional classroom at any time from any geographical location [13]. However, there is no face-to-face interaction with the educator as all the content is delivered remotely [13].

Last but not least, blended or hybrid learning is a combination of onsite and online education [13]. For the education delivery system to be considered blended, up to 30% of the course requirements must be conducted face-to-face in a traditional classroom setting, while the remaining percentage of the course requirements can be completed online [13]. Blended learning offers students the opportunity to have face-to-face interactions with the educator and other students while also providing them with the opportunity to access course materials at any time from any location [13].

However, the educational sector faces a variety of challenges, some of which are pedagogical and others being technical. This section identifies some of the challenges in the education sector. Moreover, it presents some of the previous literature works that tried to address each of these challenges. Furthermore, it discusses the role of ML in addressing them. challenges. More specifically, this section will discuss how ML can be used to grade essays, predict and prevent students from dropping-out, improve intelligent tutoring systems, recommend online courses, and provide personalized learning.

3.2.1 Essay Grading

Challenge Description:

Essays provide a tool for assessing students' critical thinking, analysis, and communication skills. However, it is time consuming for educators to grade essays [59, 60, 61]. Furthermore, when humans grade essays there is a great level of subjectivity which can lead to two different graders scoring an essay very differently [59, 60, 61]. Within this context, using ML algorithms to grade essays can reduce the workload of educators and provide more objectivity during the grading process. A common approach to creating an essay grading algorithm is to first collect a large pool of essays which have characteristics that are computationally measurable (e.g., sentence length, word frequency distributions, grammar, and spelling), and have been scored by humans [59]. This allows the algorithm to first learn the characteristics which are important for grading an essay. Then, when the algorithm is used to score essays, the algorithm's scores can be compared to those of the human graders in order to determine if the algorithm has properly learned the grading characteristics.

Previous Works:

Mahana *et al.* built an automated essay scoring system using essays from kaggle.com [60]. The authors selected roughly 13,000 essays from a pool of essays that were submitted to a competition by the William and Flora Hewlett Foundation. The essays were written by students from Grade 7 to Grade 10 and were approximately 150 to 550 words long. The selected essays were divided into eight sets, with each of the sets having unique grading characteristics. Eight different sets were selected to ensure that the automated grader was trained across different types of essays. Furthermore, each essay has one or more human scores. In the latter case, the essay also had a final resolved score which considered all human scores. After Mahana *et al.* selected the eight sets of training essays, they extracted several features from them (e.g., total word count per essay, sentence count, number of long words, part of speech counts, etc.) [60]. These features were selected because they are characteristics that a human grader would commonly look for when grading an essay. The authors then used a linear regression model to allow their algorithm to learn parameters for grading based on the selected features. After the algorithm learned the parameters for scoring the eight different essay types, the algorithm was used to score a distinct set of test essays. These scores were compared against human graded scores to arrive at an error metric (Quadratic Weighted Kappa). The average kappa score of the authors' algorithm across the eight essay types was 0.73. Essay set 8 had the lowest kappa at 0.68 and essay set 1 had the highest kappa at 0.80. Despite the fact that the proposed framework achieved good performance as seen with the high kappa values, this work has a limited contribution as it only considered one ML algorithm.

Ramalingam et al. also used ML techniques to develop an automated essay assessment system [61]. The authors selected essays from a pool of essays that were submitted to a competition by The Hewlett Foundation to kaggle.com. All of the essays had been graded by humans. Similar to the work in [60], the authors further segregated these essays into eight unique sets. This was followed by using Bayesian Linear Regression as their algorithm. The Bayesian essay scoring system used features like specific words, specific phrases, order in which certain noun-verb pair appears, and the order of the concepts explained to score the essays. In the end, the authors tested their algorithm on eighty essays divided into two groups of forty, which had also been scored by humans. The authors' algorithm was over 80% accurate at scoring the essays. Yet, this work also only considered one ML technique which limits its contribution.

Landauer *et al.* also discuss using ML to assess essays [59]. More specifically, the authors discuss the Intelligent Essay Assessor (IEA). The authors state that "IEA is based largely on Latent Semantic Analysis (LSA), a machine-learning model that induces the semantic similarity of words and passages by analysis of large bodies of domain-relevant text". The authors discuss how IEA has been validated across a wide variety of topics and test-takers. Landauer *et al.* also discussed a scenario wherein both IEA and humans graded essays which were written on neural conduction, in ten minutes, by a large undergraduate class [59]. These essays were graded by IEA, undergraduate teaching assistants, graduate teaching assistants, and/or the professor. IEA's grades correlated with the undergraduate teaching assistants' grades at 0.69, with the teaching assistants' grades at 0.78, and with the professor's grades at 0.80. In another example, the authors also presented a scenario wherein both IEA and highly trained experts graded over 800 creative narratives written by middle-schoolers. Experimental results showed that IEA's scores is highly correlated with the expert graders at r = 0.90 [59].

Research Opportunities:

As can be seen from this review of the literature, essays are an important tool for assessing students' comprehension and expression. However, grading essays is a time consuming task. Furthermore, essay grading is prone to subjectivity, which can lead to the same essay being scored differently by two graders. Hence, essay grading presents the challenges of time consumption and human subjectivity.

ML offers a potential solution to address these issues. Firstly, ML algorithms can be used for typed essays so that graders no longer need to spend time on grading. Secondly, such algorithms can be used to provide objective scores of typed essays. Although the previous subsection presented research that has shown how ML can be used to address the challenges of essay grading, there are still opportunities for further research.

One potential opportunity is exploring and evaluating different ML models (e.g. logistic model trees or deep neural networks). This is mainly due to the fact that most of the previous work only used one algorithm for essay grading. Therefore, it is important to explore and compare the performance of other ML models to obtain a more robust essay grading framework, especially given the effectiveness of other models such as deep neural networks in natural language processing problems. Another potential opportunity is studying the impact of more advanced Natural Language Processing features (e.g. N-grams, k-nearest neighbors (k-NN) in bag of words), selecting features that are grammar and usage specific, and exploring other polynomial basis functions like neural networks (NN) as part of the essay grading framework.

Such frameworks can be applied to any assessment task that contains an essay component. This includes exams and tests that contain essay sections. The application of essay grading algorithms to exam and test essays could increase the consistency of scoring while reducing the grader bias. Furthermore, there is the possibility to use essay grading algorithms as components of interactive knowledge and writing tutorial systems.

3.2.2 Dropout Prevention

Challenge Description:

Student dropout is another challenge that is prevailing in the education sector. The term dropout refers to the case when a student leaves/quits a course before completing it. In 2006, it was reported that students were 10% to 20% more likely to dropout of online courses than traditional classes [62]. High dropout rates can effect the future of colleges and universities, because policymakers, higher education funding bodies, and educators consider dropout rates to be an objective outcome-based measure of the quality of educational institutions [63]. Australia, the European Union, the United States of America, and South Africa all use dropout rates as an indicator of the quality of colleges and universities [64].

While the higher dropout rate of students in online classes is a known issue, there are many possible reasons for students to dropout. In turn, this makes predicting dropout challenging. From the student side, possible reasons for online course dropout include higher than expected workload, inability to manage academic responsibilities in a self-driven learning environment, unfamiliarity with the online educational delivery system, less student–teacher interaction, family and social obligations, and motivation level [65]. However, students may also dropout of online courses due to the course being poorly designed and delivered, which can occur when the professor who created and taught the online course is unfamiliar with technology and/or is provided with no training by their institution on how to teach in an online environment [65].

As can be seen, there are many possible reasons students may dropout, which can make predicting which students will dropout a complicated task. Even though this is a complicated task, it is important to identify students at risk of dropping out so that professors can address the needs of these students and take the appropriate actions to reduce their probability of dropping out [63]. One way to make the task of identifying at risk students easier is to use ML algorithms.

Previous Works:

Lykourentzou *et al.* used a combination of ML techniques in order to predict dropout in e-learning courses [63]. More specifically, the authors used three popular ML techniques (feed-forward NN, support vector machines (SVM) and probabilistic ensemble simplified fuzzy ARTMAP) on detailed student data to make their predictions. The authors' method of prediction combined the estimations of the three ML algorithms using three different decision schemes in order to overcome inaccuracies related to the individual ML techniques. The authors found that the most successful technique for predicting at-risk students was the decision scheme wherein a student who has been determined as at-risk by at least one of the three ML techniques is identified as being at-risk for dropping out. Looking at the data from two online courses, the authors found that from the first section of the two courses that this scheme resulted in a 75-85% overall student classification rate, and that in the final sections a 97-100% overall student classification rate was reached.

Kotsiantis *et al.* have also investigated if ML algorithms can be used to predict student dropout from online courses [66]. These authors tested six common ML techniques on data provided by the informatics course at Hellenic Open University; namely Decision Trees (DT), NN, Naive Bayes (NB) algorithm, Instance-Based Learning Algorithms, Logistic Regression (LR), and SVM. The authors found that at the beginning of the academic year, using only students' demographic data, that the NB algorithm was the most accurate (63.06%) of the six algorithms at predicting student dropout. By the middle of the academic period, the NB algorithm was still the most accurate (83.89%) of the six algorithms. Based on these results, the authors concluded that the NB algorithm was the best at predicting students at-risk for dropping out.

Research Opportunities:

Although ML has been proposed to predict dropout in e-learning courses, there are still research opportunities within this area. One potential opportunity is studying the performance of different ML dropout prediction frameworks and models in other course delivery settings such as blended learning, distance and classical education. This would highlight the generality of the dropout prediction framework. Another opportunity worth exploring is investigating the impact of different student attributes to create their dropout prediction method. This is essential as it can result in more accurate models. A third opportunity to consider is comparing the performance of different base and ensemble learning methods to achieve more accurate and robust prediction models and studying their impact on retention strategies through correlation and association rules mining.

3.2.3 Intelligent Tutoring

Challenge Description:

The third challenge facing modern education systems is that of providing and improving an intelligent tutor [67, 68, 69]. Project LISTEN's Reading Tutor is an intelligent tutor that helps students in 1st grade through 4th grade (6 to 9-year-olds) learn how to read English [69]. Students pick stories which they must read out-loud, one sentence at a time. The Reading Tutor uses speech recognition technology to determine which words the student has read incorrectly [69]. Furthermore, the student can also request help on words which s/he is uncertain about. This help can come in the form of sounding out the word, pronouncing the word, and/or providing rhyming hints. The Reading Tutor is able to assist children by constructing a model of the student (user) from voice input and mouse clicks. Although, the Reading Tutor can construct a user model from voice input and mouse clicks, this is difficult.

Previous Works:

Beck *et al.* aimed at strengthening the Reading Tutor's user model and determine which user characteristics are predictive of student behavior [67]. In order to achieve this goal, Beck *et al.* used the data of 88 students from the 2000-2001 school year, who used the Reading Tutor, to train a NB classifier to predict whether students would click on a particular word for help [67]. The data included when students started reading a story, when a new sentence was displayed, when a student read each word in the sentence, and when the student clicked on a word for help. In the end, the authors' classifier was able to predict whether students would click on a particular word for help with 83.2% accuracy. The improved ability of the Reading Tutor to predict how students will behave can be used to adapt the tutor's behavior and assess students in order to provide an enhanced learning environment.

Tam *et al.* were also interested in improving Project LISTEN's Reading Tutor [68, 69]. However, these authors were interested in estimating the probability that a word was read correctly. The authors' work led to the creation of a confidence measure that used a variety of features to estimate the probability that a word was read correctly. The confidence measure was built by investigating three kinds of features which were used to estimate confidence probabilities. These features were [68]:

1. Decoder-based features (obtained from a speech decoder).
2. Alignment-based features (derived from an alignment).
3. History-based features (extracted from the student's previous reading).

Furthermore, the authors trained two DT classifiers. The purpose of the first classifier was to try to fix insertion and substitution errors made by the speech decoder while the second classifier focused on fixing deletion errors. When the authors applied the two classifiers together and held the miscue detection rate constant, they were able to reduce the false alarm rate by 25.89%.

Research Opportunities:

Despite the fact that ML has been used to improve intelligent tutors, more research opportunities still exist. One such opportunity is considering more features (for example phonemic and history-based features) and investigating their impact on the performance of the developed model. Another potential opportunity to consider is comparing the performance of other classifiers such as NN and SVM. This comparison can help determine whether the classifiers previously proposed in the literature are biased. Moreover, such comparisons will lead to having a more adaptive and robust intelligent tutors.

3.2.4 Course Recommendation

Challenge Description:

Massively Open Online Courses (MOOCs) such us Coursera, Udacity, EdX, and MOOC.org are a form of online distance education/e-learning [13]. As such, MOOCs provide online courses that can be accessed by a student at any time from any geographical location [13]. MOOCs are open to anyone that is interested in enrolling and are often free or low-cost. However, the courses do not provide course credit and are not applied towards a degree [70]. Instead, MOOCs tend to be used by people who want to learn new skills, be it to advance their career or for fun [70]. MOOCs provide open access to a plethora of courses from various top-rated universities and institutions [70]. For example, the website mooc.org provides a course titled "Data Science: R Basics" from Harvard University, and a course titled "Introduction to Data Analysis using Excel" from Microsoft [70].

Since MOOCs are open access, hundreds of thousands of students can be enrolled in each course, with MOOCs platforms offering thousands of different courses. This means that

MOOC platforms are privy to mass amounts of data [71]. This data can then be used to improve the MOOC system. For example, having thousands of different courses available can be overwhelming for students. Therefore, if a student is looking to improve a specific skill, it would be beneficial for the MOOC system to recommend which courses are needed to acquire those skills [13, 71].

Previous Works:

Several previous literature works focused on the problem of course recommendation for students. One such example from Aher and Lobo [72]. The authors used prior student data and a combination of ML algorithms to recommend courses to students in an e-learning system [72]. The authors combined Simple K-means (a clustering technique) and Apriori (an association rule algorithm) to investigate prior students' data from Moodle.org in order to determine which courses to recommend to new students. The authors found that the results of their combination approach matched real world student course selection patterns. However, one limitation of this work is that it only considered one unsupervised clustering algorithm.

Research Opportunities:

ML algorithms can be further applied to the large amounts of data that MOOC platforms possess in order to determine which courses would be best for a student who is interested in improving a specific skill set. One potential research opportunity for students' course recommendation is evaluating the courses that other students have taken that are related to the skill that the student is interested in. Using that information can help build an effective course recommender. Another opportunity is consider multiple supervised classification algorithms. This is particularly important given the substantial impact that the classification process has on the overall performance of the recommender. Therefore, it is worth exploring the performance of different classification algorithms to study their impact on the effectiveness of the recommendation process.

3.2.5 Personalized Learning

Challenge Description:

Personalized learning is based on the individual students and how they learn. Each individual learns differently and has a unique learner profile. This profile is based on the individual's learning style [73, 74, 75], which consists of specific behaviors and attitudes [76]. Personalizing each learner's education can lead to better learning [77]. One way to personalize education is by using recommender systems that provide useful suggestions for users (books, movies, products, etc.) based on their preferences and their similarity to other students [77].

Previous Works:

Bourkoukou and Bachari tested the ability of LearnFitII to act as a recommender system [77]. LearnFitII is an adaptive learning system that automatically adapts to the dynamic preferences of learners [77]. By mining the server logs of students, LearnFitII was able to recognize the different learning styles and habits of students. Then, using the Felder-Silverman model of learning styles, LearnFitII proposed personalized learning scenarios [78]. The Felder-Silverman model of learning styles consists of four learning dimensions (1. Information Processing, 2. Information Perception, 3. Information Reception, and 4. Information Understanding) [78]. These dimensions can be accessed via the Index Learning Style Questionnaire (ILSQ) which consists of 44 questions [77, 78].

After proposing personalized learning scenarios, LearnFitII analyzed the habits and the preferences of learners by mining information about the learners' actions and interactions. After the mining of this information, the learning scenarios were revisited and updated using a hybrid recommender system which combined k-NN and association rule mining algorithms. The authors found that when LearnFitII was tested in real environments that learning quality increased and so did the learners' satisfaction with the learning process [77].

Another way that personalized learning can be beneficial is in helping students select the learning-pathway that is appropriate for them. Elfaki *et al.* investigated how student learning-pathways can be improved with ML [79]. The term learning-pathway can be understood as the path of academic courses that is appropriate for a student to achieve a degree. Ideally, one's learning-pathway is in their field of interest. Typically, students spend some time taking various courses in order to discover which topics they are interested in. However, this process of taking various courses can lead to a mismatch between a student's current and preferred learning pathway. When mismatches occur, the student may experience academic difficulties (e.g. weak performance, high absentee rate). These mismatches may lead students to lower their level of education or dropout of university altogether [79]. In order to improve students' levels of achievement it would be beneficial to help them determine their desired learning-pathway sooner. In order to achieve this goal sooner, Elfaki *et al.* first collected questionnaire data. The authors sent a questionnaire to 900 students from the Faculty of Computers and Information Technology at Tabuk University in Saudi Arabia with 450 students returning the questionnaire [79]. The questionnaire addressed four topics: basic information, personal information, academic information, and learning pathway information. After collecting this data, the authors applied a DT algorithm to the data. Then, induction rules were deduced from the tree paths in order to provide learning-pathway recommendations. In order to validate their results, the authors divided the questionnaires into two groups, a developing group (70%) and a test group (30%). Using these two groups of data, the authors found that their algorithm could accurately provide learning-pathway recommendations [79].

Taking a different approach to personalized learning, Lin et al. used ML to provide personalized learning paths for optimizing the performance of creativity [80]. The authors' personalized creativity learning system (PCLS) was developed based on data from ninety-two college students who completed a series of creativity tasks and a questionnaire that addressed key variables. Using a hybrid DT model, the authors were able to predict with 90% accuracy the probability of students obtaining an above-average creativity score.

Moubayed *et al.* [81, 82] studied the problem of identifying the student engagement level using K-means algorithm. In addition to that, Moreover, the authors extracted a set of rules that relate student engagement with academic performance. This was done using Apriori association rules algorithm. Their experimental results analysis showed that there is a positive correlation between students' engagement level and their academic performance in an e-learning

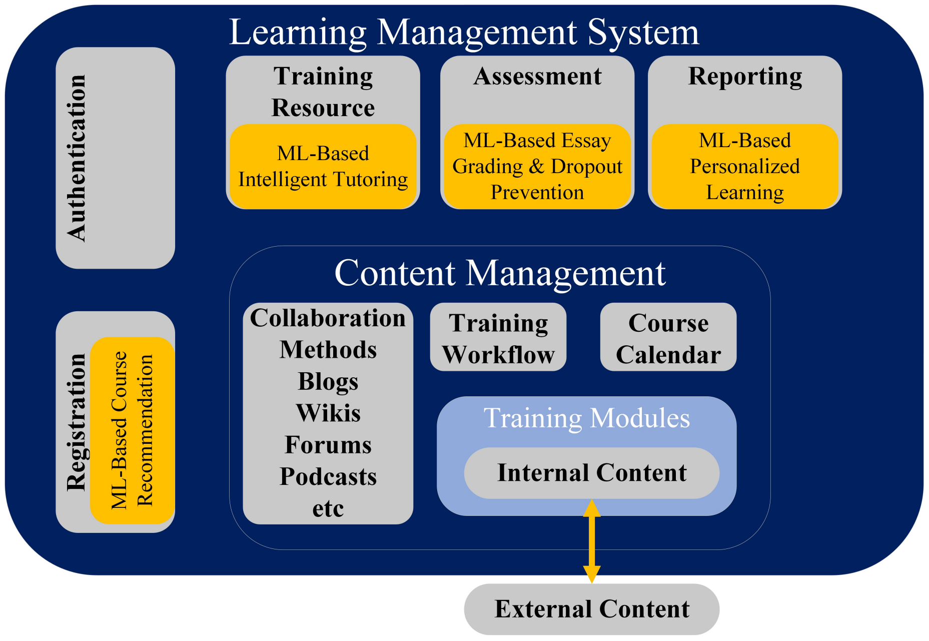

Learning Management System
Authentication
Training Resource
ML-Based Intelligent Tutoring
Assessment
ML-Based Essay Grading & Dropout Prevention
Reporting
ML-Based Personalized Learning
Registration
ML-Based Course Recommendation
Content Management
Collaboration Methods Blogs Wikis Forums Podcasts etc
Training Workflow
Course Calendar
Training Modules
Internal Content
External Content

Figure 3.1: Potential Deployment of ML in LMS

environment.

Research Opportunities:

There are still further research opportunities to use ML to provide personalized learning. One opportunity is to consider more complex recommendation approaches by including other factors such as learner motivation and knowledge level as well as additional personality traits. Another opportunity is to study the performance of different classification algorithms to predict student performance during the course delivery. This can help identify student who may need help and provide them with a personalized plan to improve their predicted performance.

Table 3.1 summarizes the challenges within the education sector, lists some of the previous works, and presents the different research opportunities. Furthermore, Figure 3.1 illustrates the potential deployment framework of the ML modules within the Learning Management System.

3.3 Healthcare

Another area where ML has shown promise is in the field of healthcare. Many modern medical organizations use electronic health records (EHRs) [7], EHRs consist of heterogeneous data elements, including patient demographic information, diagnoses, laboratory test results, medication prescriptions, clinical notes, and medical images [83]. Patient data can also include imaging, sensor and text data [84]. Furthermore, this data often comes in various formats, including structured, semi-structured and weakly structured data [85]. Originally it was thought that having access to more information about individual patients would lead to more informed medical decisions. However, often times health professionals are overwhelmed by the amount

Table 3.1: Challenges, Previous Works, and Research Opportunities within Education Sector

Challenge	Previous Work	Research Opportunity
Essay Grading	Regular linear regression is used for essay grading [60]	- Explore different ML models such as LR, DT, and DNN - Study the impact of more Language and usage specific features as well as other polynomial basis functions. - Compare the performance of the different models on various tasks to get a more accurate and robust essay grading framework
	Bayesian linear regression is used for essay grading [61]	
	Latent Semantic Analysis is used to study the domain-relevant text [59]	
Dropout Prevention	Studied the performance of three ML models to predict dropout [63]	- Study the performance of different models (base learners and ensemble learner models) in different course delivery settings. - Investigate the impact of different student attributes on the dropout prediction frameworks.
	Studied the performance of six ML models for dropout prediction[66]	
Intelligent Tutors	- NB classifier was used to predict whether students would click on a particular word for help [67]	- Consider more features such as phonemic and history-based features as well as investigate their impact on the performance of the developed model. - Study the performance of other classifiers such as NN and SVM to determine whether the classifiers previously proposed in the literature are biased.
	Trained two DT classifiers to estimate the probability that a word was read correctly [68]	
Course Recommendation	Combined k-means and apriori algorithm to recommend courses [72]	- Evaluate the courses that other students have taken that are related to the skill the student is interested in using multiple metrics. - Consider multiple supervised classification algorithms to study their impact on the effectiveness of recommendation process.
Personalized Learning	Combined a DT algorithm and induction rules algorithm to provide learning-pathway recommendations [79]	- Study the performance of different classification algorithms to predict student performance during the course delivery. - Consider more complex recommendation approaches by including other factors such as learner motivation and knowledge level as well as additional personality traits.
	Used a hybrid DT model to predict the probability of students obtaining an above-average creativity score [80]	
	Mined server logs to determine student learning style [77]	
	Used K-means and apriori algorithms to identify student engagement and their relation with academic performance [81, 82]	

of information that is now available to them [7]. Hence, a challenge with big data in healthcare is making the data easily interpretable for medical professionals. ML offers a solution to this problem because it can be used to identify relevant patterns in complex data. In this section how ML algorithms can be used to predict individual patient's responses to cancer drugs will be discussed. This section will also discuss how ML algorithms can be used in diabetes research.

3.3.1 Drug Response Prediction

Challenge Description:

One way that ML can be applied to medical data is to predict an individual patient's response to a drug or drugs [86]. For example, ML can be used to predict the responses of individual cancer patient to therapeutic drugs [87]. When working with cancer patients, it is possible to use precision cancer medicine. Precision cancer medicine aims to accurately predict the optimal drug therapies for a patient based upon the personalized molecular profiles of their tumors [88]. In order to provide precision cancer medicine, it is necessary to search for significant correlations between patient tumor profiles and the output predictions of optimal drug responses in cancer-relevant datasets [86]. Once these correlations are found in previously established datasets, they can be used to predict an individual patient's response to various series of therapeutic drugs [86]. As mentioned before, ML offers a solution to this problem, because it can be used to identify relevant patterns in complex data.

Previous Works:

Huang *et al.* applied their open-source SVM-based algorithm to the gene-expression profiles of 175 individual cancer patient's tumors [87]. The algorithm was able to predict the responses of these 175 individuals to a variety of standard-of-care chemotherapeutic drugs with > 80% accuracy [87].

Xia *et al.* also used ML to predict tumor cell line response to drug pairs [89]. The authors used a computational deep learning model to predict cell line response to a subset of drug pairs in the National Cancer Institute-ALMANAC database. When the authors ranked the drug pairs for each cell line based on the model's predicted combination effect, they were able to determine 80% of the top drug pairs.

Chiu *et al.* also used deep neural networks (DNN) and the genomic profiles of cancer tumors in order to predict the tumors' responses to therapeutic drugs [90]. The authors created DeepDR, a deep neural network model, then trained it to learn the genetic background of tumors based on data from The Cancer Genome Atlas (TCGA) [90]. DeepDR was also trained on pharmacogenomics data from human cancer cell lines provided by the Genomics of Drug Sensitivity in Cancer (GDSC) Project. After training on these data sets, DeepDR was applied to TCGA data again in order to predict the drug response of tumors. The authors' work provides insights into the ability of a deep neural network model to translate pharmacogenomics features identified from in vitro drug screening to predict the response of tumors.

Mucaki *et al.* used ML and genetic data to predict patients' responses to chemotherapy [91]. More specifically, the authors used supervised support vector ML to determine the gene sets whose expression was related to the specific tumor cell line GI50 [91]. The authors discovered that specific genes and functional pathways can be used to distinguish which tumor cell lines are sensitive to chemotherapy drugs and which tumor cell lines are resistant to chemotherapy drugs. They tested their algorithm on bladder, ovarian and colorectal cancer patient data from The Cancer Genome Atlas (TCGA) in order to determine the response of tumor cell line GI50 to three chemotherapy drugs (cisplatin, carboplatin and oxaliplatin) [91]. Through experimental results, the authors found that for cisplatin, their algorithm was 71.0% accurate at predicting disease recurrence and 59% accurate at predicting remission. In the case of carboplatin, their algorithm was 60.2% accurate at predicting disease recurrence and 61% accurate at predicting remission. Finally, for oxaliplatin, their algorithm was 54.5% accurate at predicting disease recurrence and 72% accurate at predicting remission. Furthermore, in patients who used cisplatin and had a specific genetic signature, the algorithm was able to predict 100% of recurrence in non-smoking bladder cancer patients and 79% recurrence in smokers.

Research Opportunities:

Many research opportunities still exist in applying ML for drug response prediction. One such opportunity is extending existing models to predict the drug responses of cancer patients who are receiving emerging immuno- and other targeted gene therapies. This will validate the comprehensiveness and generality of the considered frameworks. Another potential research opportunity is to build more comprehensive models by using more drug features (such as concentration, SMILES strings, molecular graph convolution and atomic convolution). This again will help extract more information and potentially uncover more correlations and interdependencies that can make the models more robust and accurate. A third opportunity is to investigate other methods and techniques including semi-supervised learning methods to encode molecular features with external gene expression and other types of data. This particularly would be helpful given that access to labeled data is not always possible. Therefore, having semi-supervised based ML models can help healthcare professionals gain insight from labeled data and apply it to the unlabeled data that they have. Last but not least, researchers should also investigate ways to adapt existing models to other drugs, cancer types, and diseases. This is essential as it would provide one adaptive system that can help healthcare professionals from different specializations make use of the available data.

3.3.2 Diabetes Research

Challenge Description:

As mentioned before, many modern medical organizations use electronic health records (EHRs) to store the medical data of patients. The large amounts of data present in EHRs can be a valuable source for researching diabetes mellitus (DM). Kavakiotis *et al.* discuss what DM is and why it is a medical concern [92]. DM is a group of metabolic disorders that are mainly caused by abnormal insulin secretion and/or action. Abnormal insulin secretion can result in a patient's body not producing enough insulin which causes the patient's metabolism of carbohydrates, fat and proteins to be impaired, which in turn results in elevated blood glucose levels (hyperglycaemia).

There are two major clinical types of DM, type 1 diabetes (T1D) and type 2 diabetes (T2D). T1D is linked to the auto-immunological destruction of the Langerhans islets; whereas T2D is linked to lifestyle, little physical activity, poor dietary habits and heredity. The main treatment for T1D is insulin administration which can applied to T2D patients. However, the main treatment for T2D is improved diet, weight loss, exercise and oral medication. DM affects more than 200 million people worldwide, with 10% of those affected with T1D and 90% affected with T2D. DM possess a health threat as chronic hyperglycaemia results in several complications, including diabetic nephropathy, retinopathy, neuropathy, diabetic coma and cardiovascular disease.

Previous Works:

DM has a high mortality and morbidity rate, therefore, detecting and treating DM is of high interest to the medical community as well as those who may or already do suffer from DM [92]. In recent years, researchers have been able to apply ML algorithms to the data of patients with DM in order to improve the methods of detecting and treating DM.

Hemoglobin is a substance in red blood cells that carries oxygen to tissues. However, it can also attach to sugar in the blood and form a substance called glycated hemoglobin (HbA1c) [93]. A patient's HbA1c level can be checked in order to determine if they have T2D. Alternatively, a patient's HbA1c level along with their fasting blood glucose level and oral glucose tolerance test results can be used to determine if they have T2D [94]. Currently, to diagnosis a patient with T2D their HbA1c value must be at or above 6.5% [94]. However, studies have shown that the cut-off value of 6.5% leads to inconsistencies in the diagnosis of T2D. Hence, using HbA1c with a 6.5% cut-off value as a single marker for T2D may lead to undiagnosed cases of diabetes [94].

Jelinek *et al.* applied ML algorithms to the data of 840 patients from the Diabetes Health screening (DiabHealth) in order to identify an optimal cut-off value for HbA1c and to identify whether additional biomarkers could be used along with HbA1c to increase the diagnosis of T2D [94]. Then the authors used T2D as the class feature and generated a conventional DT using an information gain (IG) measure. Using this algorithm, the authors found that if an oxidative stress marker (8-OhdG) was included in the model along with HbA1c that the accuracy of detecting T2D at the 6.5% HbA1c level increased from 78.71% to 86.64%. The authors also found that if interleukin-6 (IL-6) was included in the model along with HbA1c that the accuracy of detecting T2D increased from 78.71% to 85.63%. However, in this model, the optimal HbA1c range was between 5.73 and 6.22% [94].

Herrero *et al.* used ML to improve the treatment methods of T1D. Diabetics with T1D need to use the medication insulin in order to maintain normal blood sugar levels [95]. Diabetics must self-administer multiple daily injections of insulin, both before meals and basally, in order to mimic the natural insulin secretion of the pancreas. Before diabetics administer these injections, they must prick their fingertip to draw blood that is placed in an electronic glucose meter that determines the amount of glucose in the patient's blood [96, 97]. However, in recent years, an alternative form of therapy has become available. This alternative form of therapy is insulin pump therapy. In this therapy, injections are provided by continuous subcutaneous insulin infusion. The application of insulin by a machine allows diabetics to avoid multiple uncomfortable finger pricks and injections. Insulin that is taken at meal times is referred to as bolus insulin. Typically, bolus insulin doses are calculated by estimating carbohydrate intake and dividing this number by a fixed carbohydrate to insulin ratio, then adding a correction dose derived from the individual's insulin sensitivity factor [95]. Although several algorithms have been developed to calculate bolus insulin dose [98, 99, 100, 101, 102, 103], these algorithms have only been incorporated in commercially available insulin pumps and in some glucose meters [95, 104]. However, these algorithms have not been adopted widely commercially due to economic risk, security issues and inertia to change, and the lack of ease of use [95, 105]. Based on these challenges, Herrero *et al.* set out to create a more user-friendly bolus insulin calculating system. The authors used a decision support algorithm that incorporated Run-To-Run (R2R) control and case-based reasoning (CBR) [95]. They tested their algorithm via in-silico scenarios by using a simulator that emulated intra-subject insulin sensitivity variations and uncertainty in the capillarity measurements and carbohydrate intake [95]. Via these simulations, the authors found that the CBR(R2R) algorithm significantly reduced the mean blood glucose level and completely eliminated hypoglycemia. When the authors compared the CBR(R2R) algorithm to a standalone (R2R only) version of the algorithm, they found that the CBR(R2R) algorithm performed better in both adults and adolescent populations. The goal of the algorithm was to reduce blood glucose levels. Therefore, the CBR(R2R) algorithm performed better than the standalone R2R algorithm in both populations.

Zhang *et al.* used DNN for the automated identification and grading system of diabetic retinopathy [106]. The proposed system uses transfer learning and ensemble learning to detect the presence and severity of DR from fundus images. The authors' experimental results showed that their developed model has a high identification sensitivity of 97.5% and a specificity of 97.7%. On the other hand, the grading model achieved a sensitivity of 98.1% and a specificity of 98.9%.

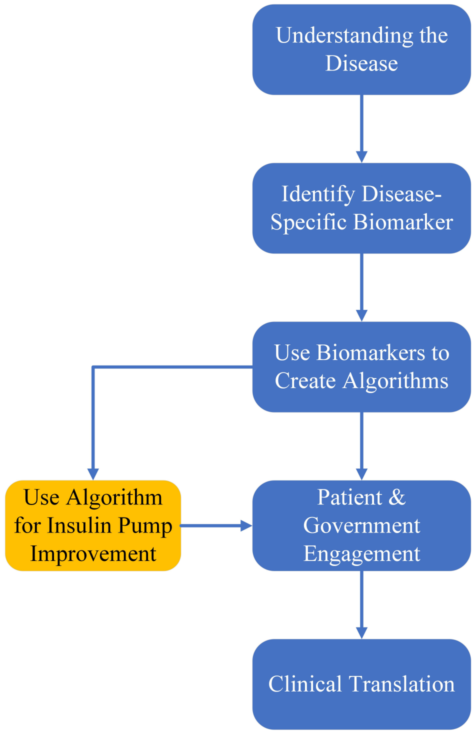

Understanding the Disease
Identify Disease-Specific Biomarker
Use Biomarkers to Create Algorithms
Use Algorithm for Insulin Pump Improvement
Patient & Government Engagement
Clinical Translation

Figure 3.2: Potential Deployment of ML in Diabetes Research

Research Opportunities:

Despite the fact that ML has been used in diabetes research, more opportunities still exist. One suggestion is testing existing algorithms in the real-world via clinical trials. This is particularly important given that simulation environment tend to over-estimate the benefits of an intervention and may not always provide an accurate representation of the behavior of the body.

Another opportunity worth exploring is investigating the performance of different ML classification models. This can help validate whether existing models have any bias. Therefore, it is important to compare the performance of different models to have a more accurate and sensitive model for insulin calculation.

Similar to the previous section on education, Table 3.2 summarizes some of the challenges facing the healthcare sector, lists some of the previous works, and presents the different research opportunities. Moreover, Figure 3.2 provides a visualization of how these topics fit into a precision medicine framework.

Table 3.2: Challenges, Previous Works, and Research Opportunities within Healthcare Sector

Challenge	Previous Work	Research Opportunity
Drug Response Prediction	Applied (SVM)-based algorithm to predict the responses of individuals to a variety of standard-of-care chemotherapeutic drugs [87]	- Extend existing models to to predict the drug responses of cancer patients who are receiving emerging immuno- and other targeted gene therapies.
	Developed a deep learning model to predict cell line response to a subset of drug pairs in the National Cancer Institute-ALMANAC database [89]	- Build more comprehensive models by using more drug features (such as concentration, SMILES strings, molecular graph convolution and atomic convolution)
	Used DNN and the genomic profiles of cancer tumors to predict the tumors' responses to therapeutic drugs [90]	- Investigate other methods and techniques including semi-supervised learning methods on other types of data.
	Used SVM to determine the gene sets whose expression was related to the specific tumor cell line GI50 [91]	- Investigate ways to adapt existing models to other drugs, cancer types, and diseases
Diabetes Research	Used conventional DT to identify an optimal cut-off value for HbA1c [94]	- Investigate the performance of different ML classification models to validate whether existing models have any bias.
	Used a decision support algorithm to calculate the bolus insulin levels [95]	- Test existing algorithms in the real-world via clinical trials.
	Used DNN for the automated identification and grading system of diabetic retinopathy [106]	- Use DNN techniques to build models that predict if a patient is diabetic or not.

3.4 Network Security

Turning to a different sector, ML can also be beneficial in network security. Cisco Systems, Inc., an American multinational technology conglomerate who specializes in information technology, networking, and cybersecurity solutions, defines network security as any activity designed to protect the usability and integrity of a network and data [15]. According to Cisco Systems, Inc., network security allows authorized users to access a network while preventing outside threats from entering or spreading on a network [15, 16]. Cisco Systems, Inc. lists fourteen types of network security. However, this section will focus on Intrusion Detection

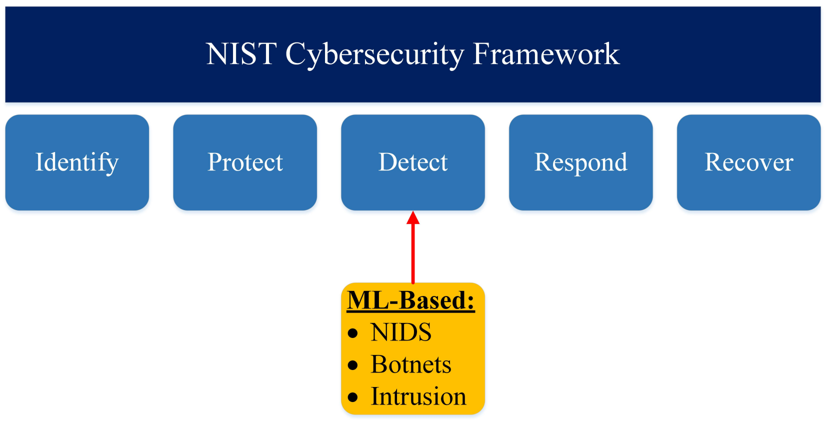


Figure 3.3: Potential Deployment of ML in Network Security

Systems (IDS) [15]. IDSs analyze and monitor network traffic in order to determine if the network traffic patterns show normal activity or if there are signs of malicious activity [17, 18]. More specifically, this section will discuss how ML can be used to improve network intrusion detection systems (NIDS) in general, how to better detect Botnets, and how to improve NIDS in vehicles. Figure 3.3 provides a visualization of how Network Intrusion Detection System (NIDS), Detecting Botnets, and Intrusion Detection in Vehicles fit into the Detect level of the NIST's Cybersecurity Framework.

3.4.1 Network Intrusion Detection Systems

Challenge Description:

A Network Intrusion Detection System (NIDS) helps system administrators to detect network security breaches in their organizations [17]. NIDSs are classified based on the style of detection that they use. Misuse-detection NIDSs use precise descriptions of known malicious behavior. Anomaly-detection NIDSs flag deviations from normal activity. Specification-based NIDSs define allowed types of activity and flag any other activity as forbidden. Behavioral detection NIDSs analyze patterns of activity and surrounding context to find secondary evidence of attacks [18]. Although there are many types of NIDS, misuse-detection and anomaly-detection NIDSs are the most common [18].

Misuse-detection NIDSs can also be referred to as signature (misuse) based NIDS (SNIDS). In SNIDS, attack signatures are pre-installed in the NIDS and pattern matching is then performed between network traffic and the installed signatures. When a mismatch is found, it is considered an intrusion [17]. There are advantages and disadvantages to both SNIDS and anomaly-detection NIDS (ADNIDS). SNIDSs are effective in the detection of known attacks and show high detection accuracy with less false-alarm rates, but is ineffective at detecting unknown or new attacks whose signatures have not been installed on the IDS [17]. On the other hand, ADNIDSs are the better option for the detection of unknown and new attacks, but produce high false-positive rates [17]. The current deployment framework and usage patters of NIDSs makes it hard for these systems to be efficient and flexible when considering unknown future attacks [17].

Previous Works:

Niyaz et al. propose a solution to the challenges of using NIDS to detect and unknown future attacks [17]. The authors' solution involved using a deep learning approach known as Self-Taught Learning (STL). STL consists of two stages, in the first stage a good feature representation is learned from a large collection of unlabeled data, then in the second stage the learned representation is applied to labeled data and used for the classification task [17]. The authors then verified their method on the benchmark intrusion dataset NSL-KDD, this dataset is an improved version of the former benchmark intrusion dataset KDD Cup 99. The authors present various metrics related to their algorithm, including accuracy, precision, recall, and f-measure values. For a detailed insight into these metrics the reader should reference the authors' paper; however, here it will be noted that the authors' algorithm achieved a classification accuracy rate above 98%.

Injadat *et al.* proposed using Bayesian optimization to hyper-tune the parameters of different supervised ML algorithms for anomaly-based IDSs [107]. More specifically, they tune the parameters of SVM, Random Forest (RF), and k-NN algorithms. Then, the authors evaluated the performance of the regular and optimized version of these classifiers in terms of accuracy, precision, and false alarm rate. Their experimental results showed that the proposed framework achieved a high accuracy rate and precision, and a low-false alarm rate and recall [107].

Salo *et al.* surveyed the literature and identified 19 different data mining techniques commonly used for intrusion detection [108]. Their review highlighted the need for more ML-based research to address real-time IDSs. Accordingly, the authors proposed an ensemble feature selection and an anomaly detection method for network intrusion detection [109]. The proposed framework combined unsupervised and supervised ML techniques to classify network traffic and identify previously unseen attack patterns. To that end, the authors used three different feature selection techniques that identified 8 common and representative features. Moreover, the authors adopted k-Means clustering to segregate the training instances into k clusters based on the Manhattan distance. Then, the classification model was developed using the aforementioned clusters. Their experimental results showed that the proposed framework was effective in detecting previously unseen attack patterns in comparison to the traditional classification approaches [109].

Wang *et al.* proposed the use of an SVM with augmented features for their intrusion detection framework [110]. More specifically, the author used the logarithm marginal density ratios transformation to get better-quality features. Using the NSL-KDD dataset, their experiments showed that the proposed framework achieved better performance in terms of accuracy, detection rate, false alarm rate and efficiency.

Research Opportunities:

Although the use of ML for network intrusion detection is popular, it still requires further research. One potential opportunity is to study the performance of more complex models such as bagging ensemble models or deep learning models. This is particularly important for real-time or near real-time network intrusion detection. Another opportunity is to study the impact of different optimization models and techniques in enhancing the current intrusion detection frameworks and models. A third research opportunity is to consider time-series analysis techniques to identify and detect temporal-based anomalies and intrusion attempts. This is crucial given that many attacks such as denial-of-service (DoS) attacks span a period of time rather than being instantaneous. Another opportunity is to investigate the performance of reinforcement learning and transfer learning techniques in IDSs. This is based on the fact that such techniques have the potential to make the IDSs more flexible and effective.

3.4.2 Botnets Detection

Challenge Description:

The term botnet refers to a network of computers (bots) which have been compromised by an attacker (aka botmaster) who has installed malicious software on the network via an attacking technique such as trojan horses, worms and viruses [111]. Botmasters often choose to attack computer networks that contain many computers due to the large amounts of bandwidth and powerful computing capabilities available for such networks [111]. Once the botmaster has control of a network, they use the network to initiate various malicious activities such as email spam, distributed denial-of-service (DDOS) attacks, password cracking, and key logging [111].

Leonard *et al.* divided the botnet life-cycle into four phases: 1. formation, 2. Command and control (C&C), 3. attack, and 4. post-attack [112]. In the formation phase, the botmaster infects other machines on the Internet, turning them into bots on the botnet. In the C&C phase, bots receive instructions from the bot master [113]. During the attack phase the bots carry out malicious activities based on the instructions of the botmaster [113]. In the post-attack phase, some bots might be detected and removed from the botnet. However, the botmaster will continue to probe the botnet for information about active bots and will plan to form a new botnet [113].

One common type of botnet is the Internet relay chat (IRC) botnet. This botnet uses IRC to facilitate command and control (C&C) communication between bots and botmasters [111]. IRC botnets can connect to one or more servers, making it easy for the botmaster to execute commands. However, IRC botnets can be stopped by shutting-down the IRC botnet's C&C server [111]. Once attackers realized this central flaw of IRC botnets, they began to utilize peer-to-peer (P2P) botnets [111]. In a P2P botnet, there is no centralized server and bots are connected to each other topologically and act as both C&C server and client [111]. Therefore, even if a P2P botnet loses some of its bots, its communication will not be disrupted [111]. According to Wang *et al.*, botnets have become one of the most significant threats to the Internet [111].

Previous Works:

Saad *et al.* explored the ability of five ML algorithms to perform network behavior analysis in order to characterize and detect P2P botnets [113]. The authors' work is also novel in that they focus on detecting bots during the C&C phase, while most other botnet detection systems detect bots during the formation or the attack phase [114]. In order to detect bots during the C&C phase, the authors studied and compared the ability of five different ML algorithms to both investigate network traffic behaviors, and to extract and analyze a set of traffic behavior characteristics. The five ML algorithms that the authors investigated were: SVM, Artificial

Neural Network (ANN), k-NN, Gaussian-Based (GB), and Naives Bayes (NB) [113]. In order to investigate the abilities of these five algorithms to characterize and detect botnets, the authors explored the algorithms' abilities to be adaptive, detect novelties, and provide early detection [113]. The authors used two datasets from the French chapter of the honeynet project which contains data about malicious traffic from the Storm botnet and the Walowdac botnet, the authors used data from the Traffic Lab at Ericsson Research in Hungary to represent non-malicious traffic [113]. The authors found that three of the algorithms (SVM, ANN, and NN) were able to detect P2P Botnet in the C&C phase with a true detection rate above 90% and a total error rate less than 7% [113]. Two of the algorithms (GB and NB) were able to detect P2P Botnet in the C&C phase with a true detection rate above 88% and a total error rate greater than 10%. From these results, the authors concluded that it is possible to identify P2P botnet C&C traffic by analyzing traffic behaviors with ML algorithms. However, the authors state that none of the algorithms were able to satisfy the novelty detection and the adaptability requirements of the online detection framework. Furthermore, the authors state that when considering both the training time and classification time metrics, the SVM and the ANN are not suitable for online detection. Based on these results, the authors concluded that while the five algorithms are promising for detecting and characterizing P2P Botnet in the C&C phase, that, currently, none of the algorithms can satisfy all of the requirements of an online botnet detection framework.

Pektaş and Acarman investigated the ability of three ML algorithms (RF, LR, and SVM) to effectively select features to use in botnet detection during network flow analysis [115]. More specifically, the authors investigated three different feature selection methods; linear models penalized with the L1 norm (aka Lasso), Recursive Feature Elimination (RFE), and tree-based feature selection (aka RF feature ranking), along with three different classifiers (LR, NB, and RF). The authors found that when the meta-classifier RF was applied on the features selected by RF that the model was nearly 99.9% accurate, making it the most accurate model that was tested. This model almost achieved perfect classification accuracy for identifying botnet and normal traffic.

Chen *et al.* also discussed using ML to detect botnets. According to the authors, in the past, signature-based and anomaly-based intrusion detection systems (IDS) were used to detect botnets [116]. However, as the speed of the Internet has increased, these methods are no longer as effective. Chen *et al.* proposed a method that uses conversation-based network traffic analysis and supervised ML to identify malicious botnet traffic [116]. The authors showed that their approach outperformed other approaches which are based on network flow analysis. More specifically, the authors' model resulted in a 13.2% decrease in the false positive rate of botnet traffic detection. Furthermore, it was shown that the RF algorithm had a high detection accuracy (93.6%) and a low false positive rate (0.3%).

Research Opportunities:

As mentioned earlier, there are still many research opportunities in the usage of ML for botnet detection that are worth exploring. One such opportunity is investigating the use of hybrid ML models to see if they can satisfy all the requirements of their proposed online botnet detection framework. Another potential opportunity is to consider non-numerical features as part of any botnet detection models since such features may contain valuable information. A third opportunity again is studying the impact of different optimization models and techniques on the performance of current botnet detection models.

3.4.3 Intrusion Detection in Vehicles

Challenge Description:

In recent years, the conventional mechanical controlling parts in cars have largely been replaced by Electronics Control Units (ECUs) [117]. ECUs are computing devices that are used for controlling and monitoring the subsystems of a vehicle for energy efficiency enhancement, and noise and vibration reduction [118]. The use of computing devices in vehicles has led to the use of automotive networking services such as Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) services. V2V automotive networking services require computing devices to perform intra-vehicular communication [119], while V2I automotive networking services require computing devices to perform inter-vehicular communication [120, 121].

One standard communication protocol for in-vehicle network communication is Controller Area Network (CAN) [122]. CAN connects sensors and actuators with ECUs [123]. Important information such as diagnostic, informative, and controlling data is delivered through a CAN bus and it is important that this information is secured in order to keep the driver safe [118, 124, 125]. However, whenever networks are used, there is a potential for significant security concerns. For in-vehicular networks there are several security flaws. For example, ECUs can obtain any ECU-to-ECU broadcasting messages in the same bus, but they are unable to identify a sender [118, 126].

Previous Works:

Based on their concerns about the security issues of in-vehicular networks, especially the CAN bus component, Kang and Kang created an intrusion detection system that uses a deep neural network (DNN) [118]. The authors' DNN was able to more accurately detect intrusions than a traditional ANN. According to the authors, this increased accuracy is due to the deep learning framework, which allows for the initialization of parameters through the unsupervised pre-training of deep belief networks (DBN). Finally, using experimental results, the authors showed that their algorithm can provide a real-time response to an attack with a detection ratio average of 98%.

In a similar manner, Li *et al.* proposed a DT-based IDS for autonomous and connected vehicles [127]. The goal of the IDS is to detect both intra-vehicle and external vehicle network attacks [127]. Experimental results showed that the authors' proposed framework improved the detection accuracy, detection rate, and F1 score by close to 2-3% and achieved lower false alarm rate than other traditional methods proposed in the literature. Moreover, the developed IDS was able to detect various attacks rather than only single type of attack on each run. The accuracy of CAN intrusion and CICIDS2017 data set reached 100% and 99.86%, while the computational time was reduced by 73.7% to 325.6s and by 38.6% to 2774.8s, respectively.

Research Opportunities:

There is still ample research opportunities to integrate ML as part of IDS systems for vehicular networks. For example, it is worth exploring the impact of different optimization techniques and meta-heuristics such as particle swarm optimization and Bayesian optimization to tune the hyper-parameters of existing IDS models. This should be done in order to improve the overall performance of such models. Another potential research opportunity is developing more complex and hybrid ML systems that can detect both the known and unknown attacks in vehicular networks. This is particularly important since more novel attacks are being introduced that are targeting autonomous and connected vehicles.

Table 3.3 summarizes the previously discussed challenges and present some of the literature work that has been conducted within this field. Moreover, they also list some of the potential research opportunities in which ML can play a role.

Table 3.3: Challenges, Previous Works, and Research Opportunities within Network Security Field

Challenge	Previous Work	Research Opportunity
Network Intrusion Detection Systems	Used a deep learning approach to detect network intrusions [17]	- Study the performance of more complex models such as bagging ensemble models, deep learning models, reinforcement learning, and transfer learning.
	Used Bayesian optimization to hyper-tune the parameters of three classification algorithms for anomaly-based IDSs [107]	- Study the impact of different optimization models and techniques in enhancing the current intrusion detection frameworks and models.
	Proposed an ensemble feature selection and an anomaly detection method for network intrusion detection [109]	- Consider time-series analysis techniques to identify and detect temporal-based anomalies and intrusion attempts.
	Used SVM with augmented features for their intrusion detection framework [110]	- Explore the performance of NIDS using recent datasets such as CICIDS2017, CSE-CIC-IDS2018 and Kyoto 2006+
Botnets Detection	Used five ML algorithms to detect P2P botnets [113]	- Investigate the use of hybrid ML models to see if they can satisfy all the requirements of their proposed online botnet detection framework.
	Used three ML algorithms to perform botnet detection during network flow analysis [115].	- Study the impact of different optimization models and techniques on current botnet detection frameworks and models.
	Used conversation-based network traffic analysis and supervised ML to identify malicious botnet traffic [116]	- Consider non-numerical features as part of any botnet detection models since such features may contain valuable information.
Intrusion Detection in Vehicles	Used a deep neural network (DNN) for intrusion detection in vehicles [118]	- Explore the impact of different optimization techniques and meta-heuristics such as particle swarm optimization and Bayesian optimization to tune the hyper-parameters of existing IDS models.
	Proposed a DT-based IDS for autonomous/connected vehicles [127]	- Develop more complex and hybrid ML systems that can detect both the known and unknown attacks in vehicular networks.

3.5 Banking & Finance

Moving on from network security, ML also has application in the sectors of banking and finance. After the financial crises of the 1980's and 90's, risk assessment of financial intermediaries became a hot topic [128]. "A financial intermediary is an entity that acts as the middleman between two parties in a financial transaction, such as a commercial bank, investment banks, mutual funds and pension funds" [129]. Researchers such as Galindo and Tamayo [128] believe that ML algorithms can be used to predict individual risk in the credit portfolios of institutions. In turn, this will help in determining who will and will not repay various forms of credit (e.g., loans, mortgages, and credit cards). Khandani *et al.* echo this sentiment as they discuss the importance of using "hard" information (e.g., characteristics contained in consumer credit files collected by credit bureau agencies) to determine the creditworthiness of consumers [130]. In the past, human discretion has been used to determine the creditworthiness of consumers. However, ML offers a way to determine the creditworthiness of consumers based on vast amounts of hard information. This section will discuss how ML can be used to assess the credit risk of potential borrowers, predict if borrowers will go bankrupt, and predict currency crises.

3.5.1 Credit Risk Assessment

Challenge Description:

With the increased dependency on mortgages and banks for financial support, credit risk assessment has garnered significant interest from both practitioners and researchers. Multiple factors are typically considered when using traditional assessment systems [131]. However, an applicant's dynamic transaction history, an important indicator of the applicant's trustworthiness and creditworthiness, is often not considered. Therefore, it is important for any credit assessment system to consider multiple factors to better utilize available resources.

Previous Works:

Galindo and Tamayo state that being able to accurately estimate the amount of individual risk in the credit portfolios of institutions would result in a more efficient use of resources [128]. In order to achieve this goal, it is necessary to determine accurate predictors of individual risk in the credit portfolios of institutions. The authors suggest that statistical and ML modeling can be used to accurately determine these predictors. To this end, the authors tested and compared the ability of various statistical and ML modeling methods to classify possible predictors in a mortgage loan dataset from a large commercial bank. The authors' modeling methodology was based on error curves and accordingly built more than 9,000 models. These models were trained on a sample of 2,000 records. Based on the results, Galindo and Tamayo determined that the CART decision-tree models provided the best estimation for default (failure to meet the legal obligations or conditions of a loan) [128]. These models had an average error rate of 8.31%. The authors believe that if they had more data, approximately 22,000 records, that they may have been able to achieve an error rate of 7.32%. The authors also found that the second-best model was NN which achieved an average error of 11.00%. The third best model was the k-NN algorithm which had an average error rate of 14.95%. The fourth best model was the standard Probit algorithm which had an average error rate of 15.13% [128].

Khandani *et al.* also used machine-learning algorithms to build consumer credit-risk models [130]. The authors used a proprietary dataset from a major commercial bank from January 2005 to April 2009. This dataset consisted of data about consumers' transactions, data from credit bureaus, and account balance data. Using linear regression, the authors were able to predict delinquencies with an accuracy rate of 85%. The authors believe that if this model is used to cut credit lines, that the bank could save 6% to 25% of their total losses. The authors also believe that the proportion of predicted delinquencies in the population can then be used as an indicator of systemic risk for consumer lending.

Research Opportunities:

Despite the literature showing that using ML has great potential for credit risk prediction, there are still research opportunities in this field. One potential research opportunity is considering more features. For example, Galindo and Tamayo [128] only used a mortgage loan dataset. However, evaluating the performance of their model as well as any other existing model on other risk factors and institution data would validate the effectiveness of their proposed model. Another potential opportunity is exploring the performance of different models that can make short, medium, and long term risk prediction rather than just on the short term as in the work of Khandani *et al.* which only focused on the short term prediction [130].

3.5.2 Bankruptcy Prediction

Challenge Description:

When selecting potential clients one aspect of their amount of credit risk is the probability that they will go bankrupt. Quantitative risk management systems, which are based on ML models, can provide financial institutions with early warning signs of clients whose potential business may fail [132]. Such failure can result in bankruptcy and the client defaulting on their bank payments. Prior work has used linear probability and multivariate conditional probability models, the recursive partitioning algorithm, artificial intelligence, multi-criteria decision making, and mathematical programming in order to predict a person's amount of credit risk [132]. Furthermore, in prior work, ANNs were commonly used since they have better power of prediction than other models. However, ANN models require a large amount of training data and tend to over-fit [132].

Previous Works:

Based on these observations, Min and Lee decided to use the SVM technique to predict bankruptcy [132]. The authors used data from Korea's largest credit guarantee organization. Originally, both the non-bankruptcy cases and the bankruptcy cases were selected from medium-sized heavy industry firms in 2002. However, due to a lack of data for the bankruptcy cases, additional data for these cases was selected from 2000, 2001, and 2002. In the end, the authors used a sample of 1,888 firms that included 944 bankruptcy and 944 non-bankruptcy cases. Once they selected their dataset, they applied SVM with an RBF kernel [132]. The authors then compared their model's ability to predict bankruptcy to three other common models (BNN, MDA, and Logit). For the hold-out data, SVM was 83.07% accurate, BNN 82.54% accurate, MDA 79.14% accurate, and Logit 78.31% accurate at predicting bankruptcy. Based on these results, the authors concluded that SVM outperformed the other models. Based on these accuracy results, SVM's ability to conduct classification learning with relatively small amount of data, and SVM's ability to prevent over-fitting, the authors concluded that SVM offers a promising model for predicting bankruptcy [132].

Kim *et al.* discuss how the financial sustainability of a company can maintain the soundness of the state and society [133]. They further discuss how the sustainability of financial institutions is directly dependent on the financial sustainability of the bank's borrowers. Hence, it is important for financial institutions to evaluate the sustainability of their borrowers, which is often done with the corporate financial distress prediction model. The authors in [133] agree with the statements of Min and Lee [132] that in previous studies of financial distress prediction models, various statistical and artificial intelligence techniques such as discriminant analysis (DA), LR, DT, case-based reasoning (CBR), and ANN have been studied. From these techniques, ANN has been one of the most frequently used techniques because of its high prediction accuracy [132, 133]. However, ANN does have its limitations [132, 133], which has resulted in researchers [132] considering SVM as an alternative to ANN. However, Kim *et al.* argue that the SVM model also has its limitations. That is why the authors propose a novel hybrid SVM model that uses globally optimized SVMs (GOSVM) and the genetic algorithm (GA) [133]. GOSVM optimizes feature selection, instance selection, and kernel parameters; while GA simultaneously optimizes multiple heterogeneous design factors of SVMs. The authors trained and tested their model on real-world data from H commercial bank in Korea [133]. The authors randomly chose 1,548 heavy industry companies, 774 of which had filed for financial distress between 1999 and 2002, and 774 which were non-bankrupt in this same time period. The authors found that their GOSVM model outperformed both non-SVM based models and other SVM-based models at accurately predicting financial distress during the hold-out phase [133]. Based on these results, the authors concluded that their model improves the prediction

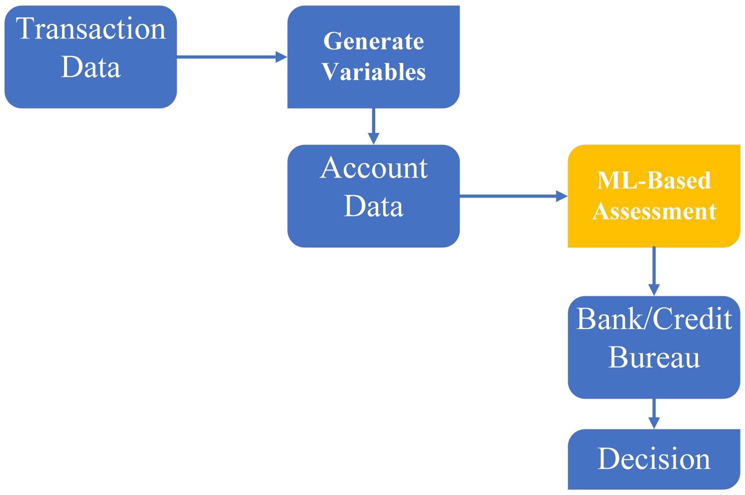


Figure 3.4: Potential ML-based Credit Risk and Bankruptcy Assessment Framework

accuracy of conventional SVMs.

Research Opportunities:

Again, there are still many research opportunities that would benefit from ML to better predict bankruptcy. For example, one open area is studying the impact of other ML models and kernels in performing the prediction. This is based on the fact that most previous work only focused on a single kernel or a single ML model. Another research opportunity that should be considered is investigating the performance of different optimization models and meta-heuristics such as simulated annealing, tabu search, or particle swarm optimization to study the potential trade-off between performance improvement and computational complexity. Figure 3.4 visualizes a potential ML-based credit risk and bankruptcy assessment framework.

3.5.3 Currency Crises Prediction

Challenge Description:

In the 90s, many countries suffered from a currency crisis wherein the value of their currency became unstable. Europe experienced a currency crisis in 1992, Mexico in 1994, Asia in 1997-98, and Russia in 1998 [134]. Currency crisis can damage the world economy, hence it would be beneficial to create an early warning system in order to prevent or at least to manage such events, particularly given the serious socio-economic impact that such events can have [134].

Previous Works:

Based on this knowledge, Lin *et al.* investigated the use of ML to predict currency crises [134]. When researching currency crisis, the authors found that prior work into the topic tended to fall into four categories [134]:

1. Emphasizing the change in some important indicators before the crisis.
2. Emphasizing the difference in values of the variables between the crisis period and the pre-crisis period.
3. Predicting the probability of a crisis according to a given theoretical model.
4. Signal approach to construct an early warning system.

After reviewing this prior work, the authors concluded that there may be an extensive non-linear relationship among the variables which can lead to a currency crisis [134]. Therefore, the authors decided to create an early warning system for currency crises using NN. More specifically, the authors used a hybrid model that combined the learning ability of neural network with the inference mechanism of fuzzy logic. The authors used a data set from Kaminsky and Reinhart (1999) which included data from twenty countries between June 1970 and June 1998. In the training set, during model building, the authors used data from 1970 through 1995. In the testing data set, during model validation, the authors used data from 1996 through 1998. The authors found that their neuro fuzzy model was able to achieve an average accuracy rate of 80.62% across the models for Indonesia, Malaysia, Philippine, and Thailand [134].

Research Opportunities:

Similar to other opportunities in the banking and finance sector, there are multiple research opportunities in which ML can play a role for financial crises prediction. One opportunity is investigating the performance of existing models in predicting financial crises in specific fields rather than just at the macro/country level. For example, study the effectiveness of existing models in predicting crises in the housing field since such models can be extremely helpful for real-estate developers and landlords. Another potential opportunity is exploring the effectiveness of different classification models in predicting such crises and studying their complexity.

Table 3.4 briefly summarizes the challenges, previous works, and potential research opportunities of ML within the banking and finance sector.

Table 3.4: Challenges, Previous Works, and Research Opportunities within Banking and Finance Field

Challenge	Previous Work	Research Opportunity
Credit Risk Assessment	Compare the performance of four classification algorithms to predict the individual risk in the credit portfolios of institutions [128]	- Consider more features and risk factors such as applicant's previous pay back history
	Used machine-learning algorithms to build consumer credit-risk models [130]	- Explore the performance of different models that can make short, medium, and long term risk prediction rather than just on the short term
Bankruptcy Prediction	Compared performance of SVM to three other ML models to predict bankruptcy [132]	- Study the impact of other ML models and kernels in performing the bankruptcy prediction.
	Compared performance of optimized SVM with ANN and other ML techniques to predict institution bankruptcy [133]	- Investigate the performance of different optimization models and meta-heuristics to study the potential trade-off between performance improvement and computational complexity.
Currency Crises Prediction	Investigated the use of ML to predict currency crises [134]	- Investigate the performance of existing models in predicting financial crises in specific fields rather than just at the macro/country level - Explore the effectiveness of different classification models in predicting such crises and studying their complexity.

3.6 Social Media

Another emerging area in which ML has been playing a major role is the area of social media. Tufts University [135] defines social media as “the means of interactions among people in which they create, share, and/or exchange information and ideas in virtual communities and networks”. The first form of social media appeared in 1979 when USENET created a decentralized system of discussion boards [136]. Since then, the Internet has advanced well beyond discussion boards where interactions occur in text only. In the early 21st century, many websites were launched that provide users with a platform to not only communicate via text, but also to share videos and/or photos [40].

According to Tufts University [135], eight of the most popular social media platforms are

Facebook, Twitter, Youtube, Vimeo, Flickr, Instagram, Snapchat, and LinkedIn. As of 2019, there are 2.77 billion social media users worldwide, with it being projected that in 2021 there will be 3.02 billion social media users worldwide [137]. There are 2.5 quintillion bytes of social media data created each day [138]. Furthermore, every minute of the day Snapchat users share 527,760 photos, 456,000 tweets are sent on Twitter, Instagram users post 46,740 photos, and Facebook users post 510,000 comments and 293,000 status updates [138]. Also, on Facebook more than 300 million photos are uploaded per day [138]. In conclusion, the vast amount of data produced by social media cannot be processed by humans. Hence, social media provides another area of opportunity for the use of ML. In this section, how ML techniques can be applied to social media data in order to make discoveries in the fields of pharmacovigilance, vaccine sentiment analysis, and politics will be discussed.

3.6.1 Pharmacovigilance

Challenge Description:

One way that social media data is being used is in pharmacovigilance. Pharmacovigilance (PhV) is defined as "the science and activities relating to the detection, assessment, understanding, and prevention of adverse effects or any other drug-related problem" [139]. Adverse effects, also known as Adverse Drug Reactions (ADRs) are harmful reactions that are caused by the intake of medication [140]. ADRs have led to millions of deaths and hospitalizations and cost nearly seventy-five billion dollars annually [140]. Governmental agencies such as the U.S. Food and Drug Administration (FDA) and the European Medicines Agency (EMA), along with international organizations such as the World Health Organization (WHO) engage in pharmacovigilance by requiring manufacturers to report adverse events [140, 141]. These agencies also encourage voluntary reporting by healthcare professionals and the public [140]. However, there is no guarantee that healthcare professionals or the public will report ADRs [140, 141]. Furthermore, when ADRs are voluntarily reported, the information may not be timely, may be incomplete, duplicated, under-reported or over-reported [140]. Due to the limited quantity and lack of quality of voluntarily reported ADRs, it has become necessary to supplement voluntary reports with other data forms. For example, information about ADRs can be acquired from health-related social networks such as DailyStrength or on social media sites such as Twitter and Facebook [140, 141].

Although these sites provide a vast amount of data for potential ADR detection, it is impossible for a human to analyze all of the data. Hence, natural language processing (NLP) and ML algorithms have been used to process the data [140, 141]. A survey of the literature shows that NLP techniques are commonly used to analyze social media data for ADRs via text classification using lexicon-based approaches [142, 143, 144, 145, 146, 146, 147, 148, 149, 150]. Furthermore, SVM [151, 152, 153, 154], NB [153], and Maximum Entropy algorithms [155] have been used to classify text. While these approaches provide a novel opportunity for collecting data about ADRs, there are still many challenges to using these approaches. For example, pure lexicon-based approaches are often impeded by consumers not using technical terms, misspelling words, using abbreviations, or sentence structure irregularities [140]. Furthermore, when supervised learning approaches are used, they require substantial amounts of data to be manually annotated, often by a domain expert [140, 156]. That being said, researchers have begun to use partially supervised (semi-supervised) algorithms in order to reduce the amount of annotated data that is required [156]. For example, Joachims [157] used the semi-supervised classification method of Transductive SVM in order to reduce the amount of annotated data the algorithm need.

Previous Works:

Nikfarjam and Gonzalez [143] created a new method that allowed them to use association rules for colloquial text mining. In their work, they realized that it is possible to set some parameters using more restrictive rules. However, while this results in higher precision, it also results in lower recall. Examples of parameter settings that lead to higher precision, but lower recall include: considering representative records with longer length in the Term-sequence file; and, limiting the minimum and maximum support and increasing the minimum number of terms in generating rules at Step 2 [143]. Furthermore, the authors [143] applied a pattern-based system in order to detect ADRs on health websites. It is not uncommon for health websites to have a large amount of text data for a single drug. For example, on the website DailyStrength, the authors observed that sometimes there were more than 10,000 free text comments for a single drug.

O'Connor *et al.* [144] utilized ML on tweets in order to discover mentions of ADRs. However, in their work the authors found that false positive errors were occurring due to non-ADR extracted terms being classified as ADRs. As an example, the authors discuss the username TScpCancer, which was classified as an ADR even though the word cancer is being used as a name in this context.

Benton et al. [145] used ML to identify potential ADRs that were mentioned on medical message boards. However, in their work the authors noticed that although their algorithm was able to extract potential ADRs that what the poster considered as an ADR might not actually be an ADR or might not be related to the drug being investigated.

Yang et al. [146] applied ML to posts on MedHelp in order to extract information about five FDA-alerted ADRs from ten drugs. Yeleswarapu *et al.* [147] built a semi-automated pipeline to extract adverse effect (AE) pairs from adverse event databases and non-traditional sources such as MEDLINE. Freifeld et al. [148] applied a semi-automated data filtering process to tweets in order to identify posts that may contain adverse events (AEs). Furthermore, the authors developed a dictionary, MedDRA, that was used to translate internet vernacular to a standardized regulatory ontology. Yang *et al.* [149] used an association rule mining approach and ADR alerts posted by the Food and Drug Administration on health-related social media content in order to identify the association between drugs and ADRs. Finally, Sampathkumar *et al.* [150] extracted information about adverse side-effects of drugs from healthcare forum messages using a Hidden Markov Model (HMM) Text Mining system.

Patki *et al.* [151] used ML techniques on social media data to automatically classify drugs into either a normal category or a blackbox category (blackbox is a category of drugs that the FDA has identified as having serious or life-threatening safety concerns). The authors' approach showed promise at classifying social media comments as ADRs or non-ADRs. However, their approach was only marginally successful at classifying drugs into the normal or blackbox categories. The authors believe that they encountered this challenge due to their limited annotated dataset. Furthermore, the authors found it challenging to distinguish true signals from the noisy social media text data.

Bian *et al.* [152] developed an analytic framework that combined natural language processing and ML methods to extract drug-related adverse events from Twitter messages. The authors discuss both the features that they used in their work and features that could be used in future research. Yang et al. [153] created a social media monitoring system that combined text mining and a partially supervised learning algorithm to identify consumer ADR messages from two Yahoo! Groups related to public health and wellness. More specifically, the authors tested two partially supervised learning algorithms, SVM and Naïve Bayes. The authors determined that SVM was a better algorithm than Naïve Bayes for their system.

Corley *et al.* [154] used ML to analyze blog posts that contained the keywords influenza or flu. The authors found that there was a strong co-occurrence between the amount of blog posts containing these keywords with the 2008-2009 flu season in the United States. Furthermore, the authors found a high correlation between the frequency of posts with these keywords and the Centers for Disease Control and Prevention's influenza-like-illness surveillance data.

Jiang and Zheng [155] developed a computational approach that collected, processed and analyzed Twitter data for drug effects. The authors tested the ability of three different ML classifiers to extract potential drug effects about five medications from Twitter data. The classifiers the authors tested were Naïve Bayes, SVM, and Maximum Entropy. Based on these results the authors concluded that the Maximum Entropy classifier was the best performer.

Research Opportunities:

Although many previous work has utilized ML for analyzing social media posts concerning drugs and medications, there still exist many further opportunities. One potential opportunity is examining the effectiveness of ML classification in modeling the contextual and semantic features of tweets. Another opportunity worth exploring is enriching the ADR lexicon datasets so that the sentiment analysis of tweets and social media posts becomes more accurate. Another potential research opportunity is performing temporal analyses to mine drug-ADR patterns and investigate ADRs related to the interaction of drugs taken by patients. Also, researchers can explore more complex classification models such as hidden markov models (HMM) to distinguish between symptoms and side-effects mentioned in the posts. Moreover, a transfer learning model can be explored to transfer knowledge from one classification domain to another, *i.e.* potentially from one drug to another or from one platform to another.

3.6.2 Social Media and Vaccines

Challenge Description:

Another way that ML can be used to gather information from social media data is by determining people beliefs, thoughts, and feelings about various vaccines. In recent years it has been observed that some individuals and/or groups have negative opinions about the safety and value of vaccines, and these negative opinions are being expressed online via social media. These negative opinions may influence some people's decisions to receive vaccines or to vaccinate their children [158, 159, 160, 161, 162]. In the past decade, in the United States and other countries, there has been an increase of parents refusing to vaccinate their children due to their concerns about the safety of vaccines [158]. Vaccine refusal for one's self or one's child can result in unnecessary harm or even death [159, 160]. One way that scientists and researchers are combating the anti-vaccination movement is by analyzing social media data with ML algorithms in order to understand how negative opinions about vaccines spread through social media. Once these patterns are understood, scientists and researchers hope that they can combat the spread of misinformation.

Previous Works:

Dunn *et al.* [158] hypothesized that when Twitter users were exposed to negative opinions about human papillomavirus (HPV) vaccines in Twitter communities that these users would subsequently express the negative opinions that they were exposed to by re-posting similar negative opinions. In order to examine their hypothesis, the authors analyzed temporal sequences of messages posted on Twitter (tweets) related to HPV vaccines and the social connections between users. The researchers' dataset was collected between October 2013 and April 2014. The dataset consisted of 83,551 tweets written in English that included terms related to HPV vaccines. Furthermore, the social connections (N = 957,865) of the 30,621 users who posted or reposted the tweets were examined to see if they also posted or reposted such tweets. In order to analyze this large dataset, the authors utilized a supervised ML approach to classify the tweets [158]. This approach required the researchers to first manually label a random sample of tweets. Then, the labeled tweets were used to train a ML classifier to recognize similar patterns in the remaining tweets. More specifically, the classifier was an ensemble of four classifiers that used the content of the tweets (the words and word combinations in the tweets themselves) or the social relations between users (the users followed by the user responsible for the tweet) in order to classify the sentiment of the tweets. The sentiment of the users' tweets about HPV vaccines was classified either as negative or neutral/positive. When the four classifiers were trained and tested in a 10-fold cross validation, their accuracy ranged between 87.6% and 94.0%. The researchers concluded that Twitter users who were more often exposed to negative opinions about HPV vaccines were more likely to subsequently post negative tweets about HPV vaccines.

Huang *et al.* [161] used natural language classifiers to examine and analyze data from Twitter in order to track flu vaccinations over time, as well as by geography and gender. The researchers collected a dataset of 1,007,582 tweets. From this dataset, the researchers created a training dataset by annotating a random sample of 10,000 tweets. After testing various classifiers, the researchers chose the best-performing classifier, namely LR, and used it in the rest of their experiments. When the researchers compared the results of their algorithm to a published government survey data about vaccination from the US Centers for Disease Control and Prevention (CDC), they found that their results were highly correlated with the CDC's data (r = 0.90). These results suggest that ML algorithms can be applied to Twitter data in order to track people's attitudes and behaviors about flu vaccinations.

Salathé and Khandelwal [162] also used ML and data from Twitter in order to assess flu vaccination sentiments. The researchers collected data about the novel influenza A(H1N1) vaccine during the second half of 2009. This time-frame covered the fall wave of the H1N1(swine flu) pandemic. The researchers' dataset came from 101,853 Twitter users and consisted of 477,768 tweets. The authors decided to use a supervised ML approach, and therefore, had students manually annotate 47,143 random tweets. The researchers then tested various classifiers. Based on their experiments, the authors determined that for sentiment classification, an ensemble method that combined NB and Maximum Entropy classifiers was the most suitable. The researchers used the NB classifier to determine the positive and negative tweets, and the Maximum Entropy classifier to determine the neutral and irrelevant tweets. The ensemble classifier had an accuracy rate of 84.29%. Furthermore, the researchers validated their approach by correlating their results with estimated vaccination rates by region provided by the US Centers for Disease Control and Prevention's (CDC). The researchers' results had a significantly strong correlation with the CDC's data. This study provides further support that ML algorithms can be applied to Twitter data in order to assess flu vaccination sentiments.

Research Opportunities:

As evident by the different research works discussed above, ML has great potential as it can be used to examine large amounts of social media data in order to track and determine how social media users may influence each other's opinions of vaccines. While these studies have shown great potential for the use of ML, there are still some limitations that offer possibilities for future research. One limitation that could use further development is that social media users' connections change over time and this may not be reflected in data that is taken from a set time period. Therefore, it is important to develop adaptive ML models that can change with the social connection changes of the social media platform. Another potential opportunity is creating new datasets with updated vocabulary to better track the sentiment of users based on the non-standard abbreviations, slang and phrases commonly used on social media platforms.

3.6.3 Social Media and Politics

Challenge Description:

Moving beyond health-related topics, ML techniques can also be applied to social media in order to collect, monitor, analyze, summarize, and visualize politically relevant information [163, 164]. In recent years, social media platforms such as Twitter and Facebook have been used to increase political participation [163]. For example, social media users publicly spread information about their political opinions on Twitter and political institutions have begun to use Facebook pages or groups to engage with citizens [163]. Furthermore, politicians and political parties are interested in social media data, because they can benefit from understanding what the public thinks about them [165]. Due to their interest in public opinion about politics, politicians or political parties may monitor social media data in order to detect social media content that is directly or indirectly associated with them [163]. Furthermore, the monitoring of political social media data is also important because it may provide information about potential political crises or scandals [163]. Additionally, the spread of political information through social networks can lead to administrative, political and societal changes [165]. For example, social media played a central role in shaping political debates in the Arab Spring [166] (a series of pro-democracy protests, uprisings, and armed rebellions that spread across North Africa and the Middle East beginning in the spring of 2011 [167]).

A common method that is used for the detection and analysis of political social media content is opinion mining (also known as sentiment analysis). The process of political opinion mining consists of collecting text that contains political opinions (or sentiments) and extracting attributes and components about a specific political feature from said text, then determine whether the text is positive, negative or neutral.

Previous Works:

Maynard *et al.* [165] discuss the challenges that arise when applying opinion mining to social media text, as well as the challenges that this process imposes on Natural Language Processing (NLP) systems. According to the authors [165], social media text data (especially short pieces such as those encountered on Twitter) provide challenges for opinion mining because tweets contain little contextual information and require implicit knowledge to process. Furthermore, tweets tend to be an ambiguous source of information because they do not typically follow a conversation thread, and may be isolated. Additionally, tweets tend to lack proper grammar. Instead, they contain more language variation, emoticons, abbreviations, hashtags, irony and sarcasm. All of these things are difficult for ML algorithms to detect and interpret. THe authors [165] note that traditional NLP approaches that involve full parsing tend not to be suitable for tweets due to these various issues. Therefore, they implemented an entity-centric approach that used rule-based NLP techniques based around entity and event recognition [165]. The authors show that their approach, which was shallower and more focused than traditional NLP approaches, was better able to interpret non-standard text (tweets). They utilized their approach in two example applications in very different domains, the Greek financial crisis and the 2010 Rock am Ring rock festival in Germany [165]. The authors used the freely available language processing toolkit GATE [168] in order to mine political opinions from tweets. After mining opinions from a political tweets dataset, the authors compared the performance of their algorithm to twenty Facebook posts about the Greek financial crisis that were manually annotated. The authors found that their system was able to identify sentiment-containing sentences with 86% Precision and 71% Recall. Furthermore, out of the correctly identified sentences, the authors' system was able to accurately determine the polarity (positive or negative) of the sentiment 66% of the time.

Jahanbakhsh and Moon [169] have also implemented sentiment analysis on tweets. However, they were interested in the predictive power of social media. In their study, the authors analyzed 32 million tweets related to the 2012 US presidential election using a combination of ML techniques. The authors created a ML and language processing (ML-NLP) engine that consists of four major components. These components and their functions are:

1. Statistical component (computes all basic statistics)
2. Text Analysis (runs basic text analysis tasks)
3. NB classifier (sentiment analysis)
4. Latent Dirichlet Allocation (LDA) algorithm (topic modeling).

The authors implemented a Twitter crawler from September 29, 2012 until November 16, 2012 using keywords such as Barack Obama, Mitt Romney, US election, Paul Ryan, and Joe Biden,

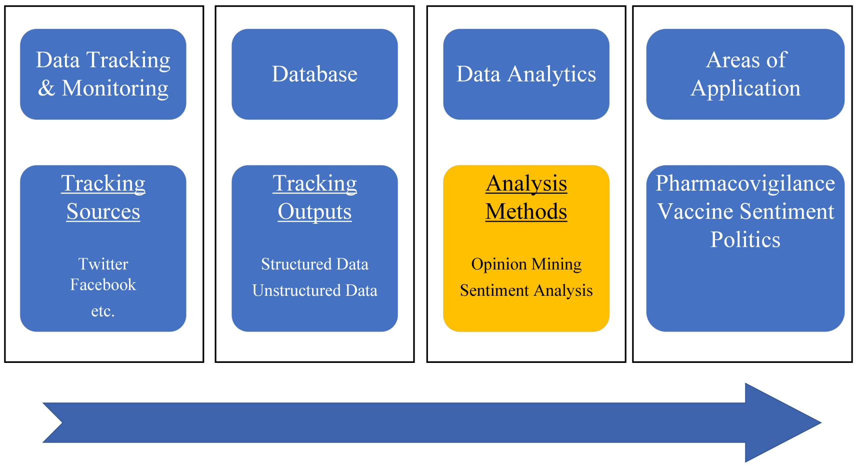


Figure 3.5: Potential ML-based Social Media Analytics Framework

in order to collect tweets related to the 2012 US presidential election. This resulted in the collection of around 39 million tweets. After collecting this data, the authors conducted statistical analysis, sentiment analysis, and analyzed the topics discussed on Twitter each day before the election [169]. Their results were numerous. Firstly, the authors' results (that Obama was leading in Twitter for the 2012 US presidential election) matched with the outcome of the election. Secondly, the authors found that negative advertising against one's competitor resulted in more negative tweets about one's competitor. This effect came from both political parties (Democratic and Republican) and shows that negative campaigning by either party results in negative tweets about the other party. As such, negative advertising is an effective method to reduce the popularity of one's competitor. Thirdly, the authors found that by analyzing geo-tweets (tweets with a geo-tag) with geographical sentiment analysis, they were able to uncover the popularity of candidates across the US states. Fourthly, the authors work demonstrated that LDA is a powerful unsupervised algorithm. The authors created a ML-NLP engine that implemented a NB classifier for sentiment analysis, and the Latent Dirichlet Allocation (LDA) algorithm for topic modeling [169]. This engine allowed them to run text analysis on a large number of tweets and "predict" the outcome of the 2012 US election. Hence, the authors have presented a system of mining social media data that may be used for predicting future events.

Research Opportunities:

Again, there are still many research opportunities in which ML can play a role as part of a politics sentiment analysis frameworks. One such opportunity is investigating other ML algorithms such as SVM and ANN given their previous success in determining linguistic features for opinion classification. Another potential opportunity is collecting more features such as swear words, sarcasm, and negative and conditional detection as well as contextual clues features to make the sentiment analysis framework more accurate and effective.

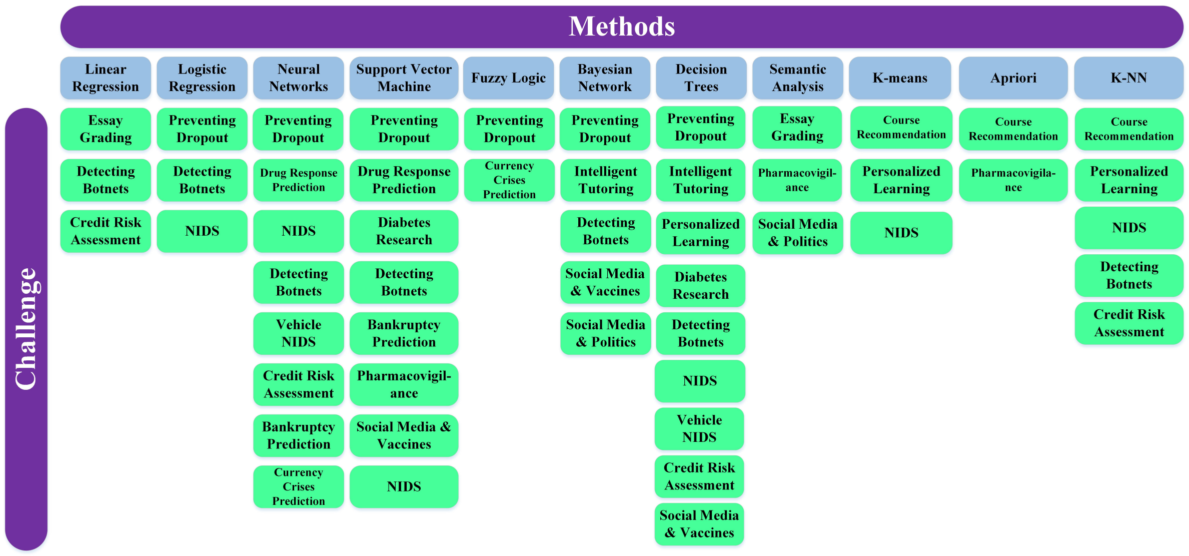
Figure 3.5 provides a visualization of how these topics fit into a Social Media Analytics Framework. Moreover, Table 3.5 summarizes some of the different challenges and research opportunities of ML within the social media field.

Table 3.5: Challenges, Previous Works, and Research Opportunities within Social Media Field

Challenge	Previous Work	Research Opportunity
Pharmacovigilance	Used association rules for colloquial text mining [143]	- Examine the effectiveness of ML classification in modeling the contextual and semantic features of tweets. - Enrich the ADR lexicon datasets so that the sentiment analysis of tweets and social media posts become more accurate. - Perform temporal analyses to mine drug-ADR patterns and investigate ADRs related to the interaction of drugs taken by patients. - Explore more complex classification models such as hidden markov models (HMM) to distinguish between symptoms and side-effects mentioned in the posts. - Explore transfer learning models to transfer knowledge from one classification domain to another.
	Utilized ML on tweets in order to discover mentions of ADRs [144]	
	Used ML to identify potential ADRs that were mentioned on medical message boards [145]	
	Used an association rule mining approach and ADR alerts to identify the association between drugs and ADRs [149]	
	Combined natural language processing and ML methods to extract drug-related adverse events from Twitter messages [152]	
Social Media and Vaccines	Utilized a supervised ML approach to classify vaccine-related tweets [158]	- Develop adaptive ML models that can change with the social connection changes of the social media platform. - Create new datasets with updated vocabulary to better track the sentiment of users based on the non-standard abbreviations, slang and phrases commonly used on social media platforms.
	Used natural language classifiers to examine and analyze data from Twitter in order to track flu vaccinations over time, geography, and gender [161]	
	Used ML and data from Twitter in order to assess flu vaccination sentiments [162]	
Social Media and Politics	Implemented an entity-centric approach that used rule-based NLP techniques based around entity and event recognition [165]	- Investigate other ML algorithms such as SVM and ANN given their previous success in determining linguistic features for opinion classification. - Collect more features such as swear words, sarcasm, and negative and conditional detection as well as contextual clues features to make the sentiment analysis framework more accurate and effective.
	Created a ML-NLP engine that implemented a NB classifier for sentiment analysis, and the Latent Dirichlet Allocation (LDA) algorithm for topic modeling [169]	

3.7 Conclusion

The availability and popularity of the Internet and related technologies has resulted in large amounts of data being available for analyses. However, humans do not possess the cognitive capabilities to understand such large amounts of data. Machine learning (ML) provides a way for humans to process large amounts of data and come to conclusions about the data. ML has applications in various fields. This review focused on some of the fields and applications such as education, healthcare, network security, banking and finance, and social media. These fields each have their own unique challenges. However, ML can provide solutions to these challenges, as well as create further research opportunities. Accordingly, this work briefly described some of the challenges facing the aforementioned fields and surveyed some of the previous literature works that focused on them. Moreover, it presented several research opportunities on the role and potential of using ML to address these challenges. Figure 3.6 summarizes the challenges and previous/potential ML techniques that addressed/can address them respectively.



Methods

Challenge

Linear Regression	Logistic Regression	Neural Networks	Support Vector Machine	Fuzzy Logic	Bayesian Network	Decision Trees	Semantic Analysis	K-means	Apriori	K-NN
Essay Grading	Preventing Dropout	Preventing Dropout	Preventing Dropout	Preventing Dropout	Preventing Dropout	Preventing Dropout	Essay Grading	Course Recommendation	Course Recommendation	Course Recommendation
Detecting Botnets	Detecting Botnets	Drug Response Prediction	Drug Response Prediction	Currency Crises Prediction	Intelligent Tutoring	Intelligent Tutoring	Pharmacovigil-ance	Personalized Learning	Pharmacovigila-nce	Personalized Learning
Credit Risk Assessment	NIDS	NIDS	Diabetes Research		Detecting Botnets	Personalized Learning	Social Media & Politics	NIDS		NIDS
		Detecting Botnets	Detecting Botnets		Social Media & Vaccines	Diabetes Research				Detecting Botnets
		Vehicle NIDS	Bankruptcy Prediction		Social Media & Politics	Detecting Botnets				Credit Risk Assessment
		Credit Risk Assessment	Pharmacovigil-ance			NIDS				
		Bankruptcy Prediction	Social Media & Vaccines			Vehicle NIDS				
		Currency Crises Prediction	NIDS			Credit Risk Assessment				
						Social Media & Vaccines				

Figure 3.6: Summary of Challenges and ML Techniques

Chapter 4

Systematic Ensemble Model Selection Approach for Educational Data Mining

4.1 Introduction

Data mining is rapidly becoming a part of software engineering projects, and standard methods are constantly revisited to integrate the software engineering point of view. Data mining is best defined as an extraction of data from a dataset and discovering useful information from it [170]. Data collected is then analyzed and used for enhancing the decision-making process [171]. Data mining uses different algorithms in an attempt to establish certain patterns from data [172].

After the development of data mining, a new subfield named Educational Data Mining (EDM) has emerged. Educational Data Mining specializes in analyzing educational results in order to understand and improve students' performance [174] and enhance learning and teaching [171]. Data used for Educational Data Mining includes administrative data and students' performance and activity data [175]. To be able to implement EDM methods, data needs to be collected from different databases and e-learning systems [171]. Online learning or e-learning has been changing recently along with the change of computer-mediated communication (CMC) [176].

The advancement of technology and the Internet also affected learning and education. In that matter, e-learning was developed and can be defined as "the use of computer network technology, primarily over an intranet or through the Internet, to deliver information and instruction to individuals" [177]. There are two types of e-learning: asynchronous and synchronous. Asynchronous e-learning is available to students at any time and from anywhere [178], while synchronous e-learning is "live" e-learning and requires all students to be connected at the same time, and it is similar to a virtual classroom [177].

There are various challenges regarding e-learning, such as the assorted styles of learning, and challenges arising from cultural differences [179]. Other challenges include pedagogical e-learning, technological and technical training, and the management of time [180]. That is why the need for more personalized learning has emerged. Personalized learning can be considered as one of the biggest challenges of this century [181], where the personalization of e-learning

A version of this chapter has been published in [173].

includes adaptation of courses to different individuals. One of the biggest learning differences includes the level of knowledge an individual has, and it is being accessed through the learner profile. Learner profile is the most crucial step of the personalization process [181]. To make learning more personalized, adaptive techniques can also be implemented [182], [183]. Data can be automatically collected from the e-learning environment [183] and then the learner's profile can be analyzed in order to customize the course according to the participant's location, language, currency, seasons etc. [183], [184], [185].

This chapter uses the comparative analysis gained from various classification algorithms to predict student's performance at earlier stages of the course. The developed models use ensemble classification techniques to categorize the students and predict their final performance group. The purpose is to identify the weak students that may need help at earlier stages of the course delivery. In these terms, few classification methods were used, such as K-nearest neighbor (k-NN), random forest (RF), Support Vector machine (SVM), Logistic Regression (LR), Multi-Layer Perceptron (MLP) and Naïve Bayes (NB). These techniques were used individually or as a part of an ensemble learner model to predict the final performance group during the course at two stages - at 20% and 50% of the coursework. The aim is to identify the best machine learning individual or ensemble classifier that performs well with e-Learning data. We intend to prove via experiments that ensemble learners model have the highest overall accuracy on e-learning data.

The chapter is organized as follows: the related work is described in Section 4.2; in Section 4.4 we describe the methodology used for the experiments; the utilized datasets are described, analyzed, and the method to evaluate our model is given in Section 4.5; experimental results and the discussion are given in Section 4.7 and finally Section 4.8 represents the conclusion and future work.

4.2 Related Work

DM methods have great potential when it comes to analyzing educational data. There is a big interest for understanding the needs of students and their actual level of knowledge. Many researchers have been interested in this problem during the last few years. In 2000, researchers tried to determine low-performing students by using association rules [184], so that they could involve them in additional courses. Luan [185, 186] tried investigating which students are most likely to fail the course by using clustering, neural networks and decision tree methods. In 2003 [187], Minaeli-Bidgoli et al. used classification for modeling online student grades, while in [188] authors were investigating how students' performance can be influenced by demographic characteristics and performance.

Pardos et al. [189] used LR to predict the test score in math based on students' individual characteristics, while Superby et al. [190] used decision tree techniques, RF method, Neural networks and Linear discriminant analysis for predicting students who will most likely dropout. Vandamme et al. [191] also used Decision tree methods, neural networks and linear discriminant analysis for their prediction of students who will fail the course by classifying them into three groups: low, intermediate and high-risk students. In 2008, Cortez and Silva [192] compared DM algorithms from four different approaches, namely Decision Tree, RF, Neural Network and SVM for prediction of students' failure.

Kovacic [193] developed a profile of students who would most likely fail or succeed by using classification techniques. He used socio-demographic and learning characteristics as variables for predicting students' success. Ramaswami et al. [194] tried developing a predictive model that will be used for identifying students who are slow at learning by using Chi-square Automatic Interaction Detector (CHAID) decision tree algorithm.

Pandey [195] used NB classification to accurately distinguish the bright students from the slow ones. Their model was able to predict students' grades based on their previous grades. In 2012, authors conducted a comparative research to make a best guess of the student's performance [196]. The study used decision tree algorithms and it was aimed at finding the best decision tree algorithm that can accurately predict students' grades. The authors found that CART algorithm that was designed as a decision tree algorithm was the most efficient as it produced the most desired results and concluded that it is desirable to try different classifiers first and then decide which one to use based on the precision and accuracy it gives. Kabakchieva in [197] used four DM algorithms – OneR Rule Learner, Decision Tree, Neural Network and k-NN. Results indicated that the highest accuracy was achieved using the Neural Network algorithm, where the most influencing factors on the classification process were students' score upon admission and the frequency of failures in the first-year examinations.

Yadav et al. [198] investigated how the marks from previous or first year exams impact the final grade of engineering students. In their experiments, the authors used classification algorithms such as ID3, J48 (C4.5) and CART and they found that J48 (C4.5) gives the most accurate results. In 2013, one research of secondary education data [199], performed by using NB and decision tree algorithms, concluded that decision tree classification algorithm was the best for predicting students' performance and that students' previous data can be used to predict their final grade.

Hung et al. [200] proposed the use of different classification algorithms such as SVM, RF, and neural networks to improve at-risk student identification. Experimental results performed on two datasets collected from both a school and university environments showed that the proposed approach had a higher accuracy and sensitivity than other works in the literature.

Similarly, Moubayed et al. [81][82] investigated the problem of identifying the student engagement level using K-means algorithm. Moreover, the authors derived a set of rulers that related student engagement with academic performance using Apriori association rules algorithm. Experimental results analysis showed that the students' engagement level and their academic performance have a positive correlation in an e-learning environment.

Helal *et al.* proposed different classification algorithms to predict student performance while taking into consideration multiple features including socio-demographic features, university admission basis, and attendance type [201]. The authors' experimental results showed that rule-based algorithms as well as tree-based algorithms provided the highest interpretability which made them more useful in an educational environment [201].

Zupanc and Bosnic extended an existing automated essay evaluation system by considering semantic coherence and consistency features [202]. Through their experimentation, the authors showed that their proposed system provided better semantic feedback to the writer. Moreover, it achieved higher grading accuracy when compared to other state-of-the-art automated essay evaluation systems [202].

Xu *et al.* proposed a two-layered machine learning model to track and predict student performance in degree programs [203]. Their simulation results showed that the proposed approach achieved superior performance to benchmark approaches [203].

Sekeroglu *et al.* compare the performance of five machine learning classification models to predict the performance of students in higher education [204]. Their experimental results showed that the prediction performance can be improved by applying data pre-processing mechanisms [204].

Khan *et al.* compared the performance of eleven machine learning models in terms of accuracy, F-measure, and true positive rate [205]. Their experimental results showed that decision tree algorithm outperformed other classifiers in terms of the aforementioned metrics [205].

4.2.1 Limitations of Related Work

The difference in the reported results of the previous research is due to multiple factors. First, the participants of the research in different models influence the decision of the studies and their preference. Different researchers have varying interpretation of the models. Moreover, researchers could be biased depending on the educational environment under consideration. Contradicting results could also be caused by prior knowledge of the researchers concerning the models. To carry out a research, one goes through literature from past studies and in doing so, their stand on the best model could have been biased. Also, the difference in results in related work is because they are not using the same dataset or the same sample in the case where the dataset is the same. The same models perform differently when evaluated using different datasets. Moreover, one major limitation that many of the previous works in the literature suffer from is the fact that they use data collected from one course/term to predict the performance of students in future courses/ terms. However, to the best of our knowledge, none of the previous works predict the student performance during the course delivery.

After going through the related work, our research aims to confirm the claims and clear any doubts concerning the best model that can identify students who may need help during a course at two stages. By conducting a practical research, our study aims to evaluate the prior findings and their authenticity. Our study will not be biased in any manner and it will look into the nature of datasets. Moreover, our research explores all the six algorithms equally, and any possible ensemble learner that might be developed using these algorithms. The study design predicts the students' grades during the course as opposed to other designs that prefer to conduct it at the end of the course because it is a more accurate predictor. The research assumes that the efforts and seriousness of a student are directly proportional to the final course performance and grade. Therefore, assuming that all factors are constant, the performance of a student can be accurately predicted in the course of the semester.

4.3 Contribution of Proposed Research

Based on the discussion of the related work limitations, the contributions of this work can be summarized as follows:

- Analyze the collected datasets and their corresponding features using multiple graphical, statistical, and quantitative techniques (e.g. probability density function, decision boundaries, feature variance, feature weights, principal component analysis,etc.)

- Conduct hyper-parameter tuning to optimize the parameters of the different ML algorithms under consideration using grid search algorithm.
- Eliminate any bias in the optimized models through the use of multiple splits of the training and testing data at both course delivery stages under consideration.
- Propose a systemic approach for building an ensemble learner to choose the best model based on multiple performance metrics, namely the Gini index and the p-value.
- Evaluate the performance of traditional classification techniques compared to the proposed ensemble learner.
- Identify students who may need help with high accuracy using the proposed ensemble learner.

4.4 Methodology and the Research Framework

The purpose of this study is to predict students' final grades in order to identify students who may need help at earlier stages of the course. Figure 4.1 shows the analytical process. In grey, the step of the analytic process our research focuses on. More in detail, *ML Based student*

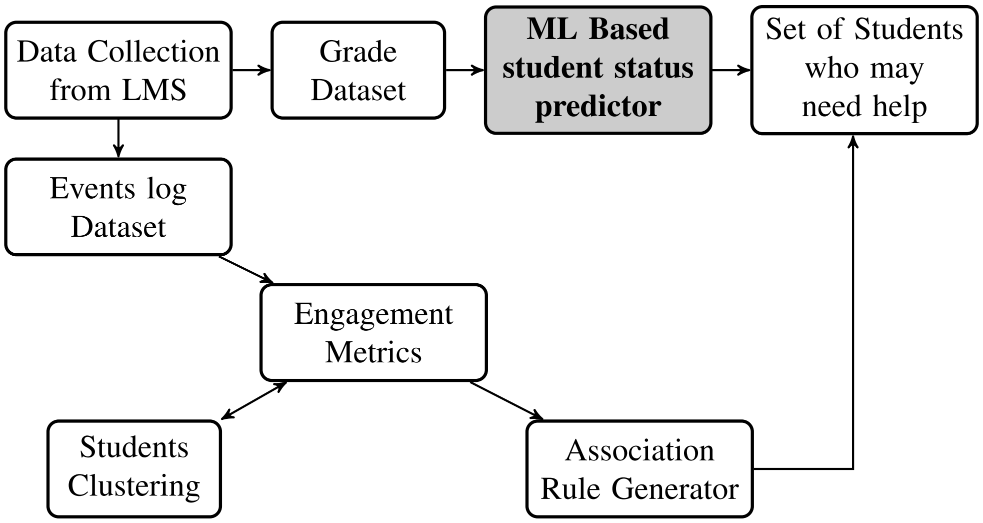

Data Collection from LMS
Grade Dataset
ML Based student status predictor
Set of Students who may need help
Events log Dataset
Engagement Metrics
Students Clustering
Association Rule Generator

Figure 4.1: Learning Management System (LMS) Analytical Module

status predictor is structured as in Figure 4.2.

Two datasets were used in this experiment. Dataset 1 consists of records of 115 first year students who attended undergraduate engineering course at the University of Genoa [206]. The students were selected in a random manner to avoid selection bias. Their prior performance remained unknown to the research team. The initial dataset was considered to speed up the research and from these 115 students, only 52 completed the course. Students' individual marks were used in the analysis. Dataset 2 consists of records of 486 students who attended undergraduate science course at University of Western Ontario, Canada. The two datasets comprised

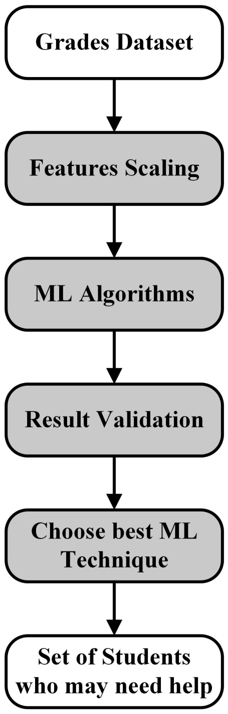


Figure 4.2: ML-Based Student Status Predictor

of students in approximately the same proportions of male-female ratio to avoid conducting a gender-biased research. Students' individual marks were used in the analysis while the event logs were investigated in other research works within the group.

These experiments predict the final grade based on the individual marks during the course at two stages: 20% and 50% of the coursework.

To improve the accuracy of the prediction, the event log system was substituted with individual marks of quizzes, exams and assignments. The method used was the conversion of marks to percentage as this scaling of scores (grades) improved the experimental results accuracy. The scaling of scores was also important when it came to compare the performance of students. Furthermore, if a student was absent for certain mark and it was empty in the dataset it was replaced with the value of zero. This also improves the experimental results accuracy across the six used techniques.

This experiment predicts the final letter grade based on the individual marks during the course at two stages 20% and 50% of the coursework for both dataset.

The final grade was classified into two categories (classes):

- Good (G) – the student will finish the course with a good grade (60% – 100%);

- Weak (W) – the student will finish the course with a weak grade ($\leq$ 59%).

The second class represents the targeted learners, i.e. students who need additional assistance and concentration in order to improve their performance.

4.5 Datasets' Description and Analysis

In the 1950's, an American Educational psychologist, Benjamin Bloom, developed his taxonomy of cognitive objectives. According to *Bloom's Taxonomy*, thinking skills and objectives can be categorized and ordered following the thinking process, [207]. Bloom's Taxonomy was revised years later when the categories or taxonomic elements were associated with it Lower Order Thinking Skills (LOTS):

- Remembering - Recognizing, listing, describing, identifying, retrieving, naming, locating, finding
- Understanding - Interpreting, Summarizing, inferring, paraphrasing, classifying, comparing, explaining, exemplifying
- Applying - Implementing, carrying out, using, executing
- Analyzing - Comparing, organizing, deconstructing, Attributing, outlining, finding, structuring, integrating
- Evaluating - Checking, hypothesizing, critiquing, Experimenting, judging, testing, Detecting, Monitoring
- Creating - designing, constructing, planning, producing, inventing, devising, making

In this section we describe two distinct datasets at two separate stages each, consisting of the results of a collection of tasks performed by University students, and we conduct some Principal Components Analysis. Interestingly, the first four principal components for Dataset 1, stage 20% and 50%, correspond to four of the categories above.

4.5.1 Dataset Description

In this chapter, R was used for numerical analysis, machine learning techniques and data virtualization [208]. R is a language and environment for statistical computing and graphics.

- *Dataset 1*: The experiment has been conducted with a group of 115 students of first-year, undergraduate engineering major of the University of Genoa. The dataset contains data collected using a simulation environment named Deeds (Digital Electronics Education and Design Suite) and it is used in e-Learning courses. The e-Learning platform offers the courses' contents using a special browser which will ask the students to solve problems that lied under different complexity levels.
  The records were summarized in Table 4.1 in order to be analyzed. Only 52 students completed the course.
  Features ES.1.1 to ES 3.5 were used in the 20% stage. Features ES. 1.1 to ES 5.1 which used at the 50% stage.
  Any empty mark replaced with 0, also all features converted to mark out of 100 which improves the accuracy of all classifiers. Any mark that consist of decimal point number was rounded to the nearest 1.

Feature	Description	Type	Value/s
Id	Student Id.	Nominal	Student 1,..,Student 52
ES 1.1	Exercise 1.1 Mark	Numeric	0..2
ES 1.2	Exercise 1.2 Mark	Numeric	0..3
ES 2.1	Exercise 2.1 Mark	Numeric	0..2
ES 2.2	Exercise 2.2 Mark	Numeric	0..3
ES 3.1	Exercise 3.1 Mark	Numeric	0..1
ES 3.2	Exercise 3.2 Mark	Numeric	0..2
ES 3.3	Exercise 3.3 Mark	Numeric	0..2
ES 3.4	Exercise 3.4 Mark	Numeric	0..2
ES 3.5	Exercise 3.5 Mark	Numeric	0..3
ES 4.1	Exercise 4.1 Mark	Numeric	0..15
ES 4.2	Exercise 4.2 Mark	Numeric	0..10
ES 5.1	Exercise 5.1 Mark	Numeric	0..2
ES 5.2	Exercise 5.2 Mark	Numeric	0..10
ES 5.3	Exercise 5.3 Mark	Numeric	0..3
ES 6.1	Exercise 6.1 Mark	Numeric	0..25
ES 6.2	Exercise 6.2 Mark	Numeric	0..15
Final Grade	Total Final Mark	Numeric	0..100
Total	Final Grade	Symbolic	G,W

Table 4.1: Dataset 1 - Features

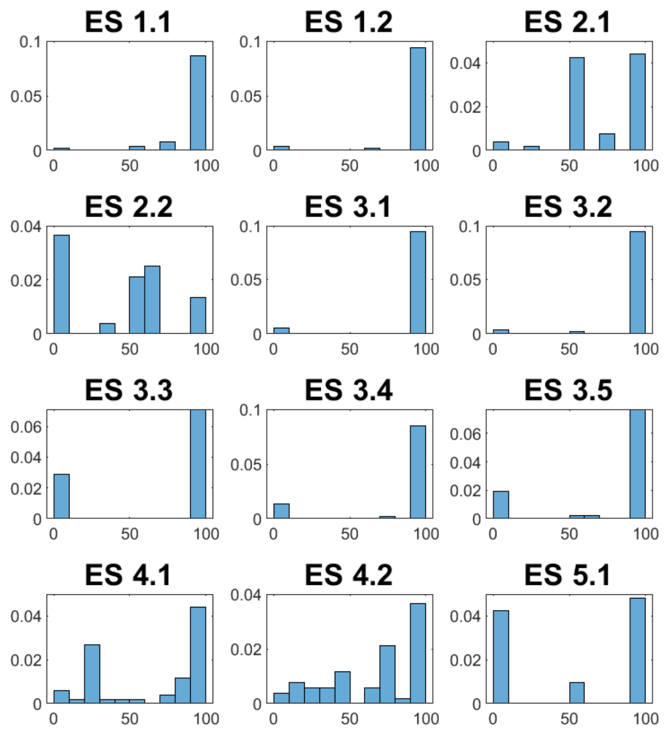


Figure 4.3: Dataset 1 - Features' distribution

In particular, the Final Grade has distribution as in Fig. 4.4:

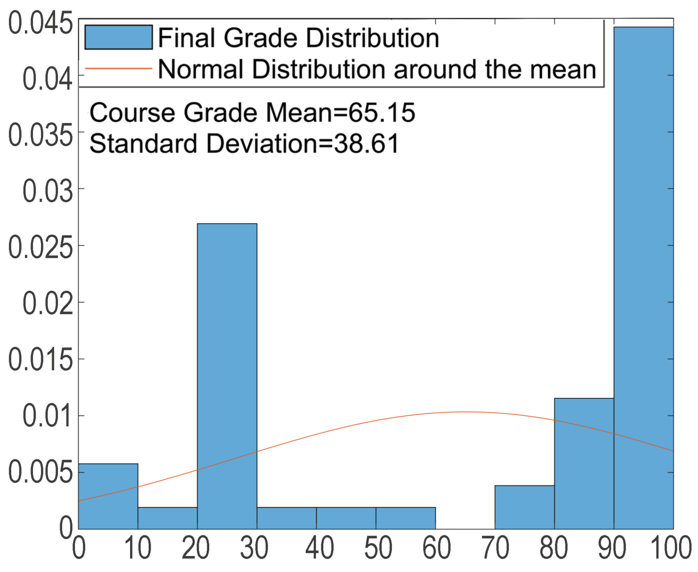


Figure 4.4: Dataset 1 - Final Grade Distribution

- *Dataset 2*: The collected dataset is from a second year undergraduate Science course offered in The University of Western Ontario. The dataset consists of two parts. The first part is an event log of 486 enrolled students and has a total of 305933 records collected from the university's learning management system (LMS). Note that this event log was used in other research works within the group. The second part which is used in this experiment is the obtained grades of the 486 students in the different assignments, quizzes, and exams. Features Quiz 01 and Assignment 01 were used in the 20% stage. Features Quiz 01 to Assignment 02 were used at the 50% stage. Any empty mark replaced with 0, also all featured converted to mark out of 100 which improves the accuracy of all classifiers. Any mark that consist of decimal point number was rounded to the nearest 1. The total course mark was counted out of 110, the additional 10% were implemented in assignment 03 as curving to help students in the course final grade. In Table. 4.2 we show the list of features corresponding to dataset 2.

Feature	**Description**	**Type**	**Value/s**
Id	Student Id.	Nominal	student00000,..,student00485
Quiz01	Quiz1 Mark	Numeric	0..10
Assignment01	Assignment 1 Mark	Numeric	0..8
Midterm	Midterm Exam Mark	Numeric	0..20
Assignment02	Assignment 2 Mark	Numeric	0..12
Assignment03	Assignment 3 Mark	Numeric	0..25
Final Exam	Final Exam Mark	Numeric	0..35
Final Grade	Total Final Mark	Numeric	0..100
Total	Final Grade	Symbolic	G,W

Table 4.2: Dataset 2 - Features

And the distribution of the variables in Table 4.2 is shown in Figure 4.5.

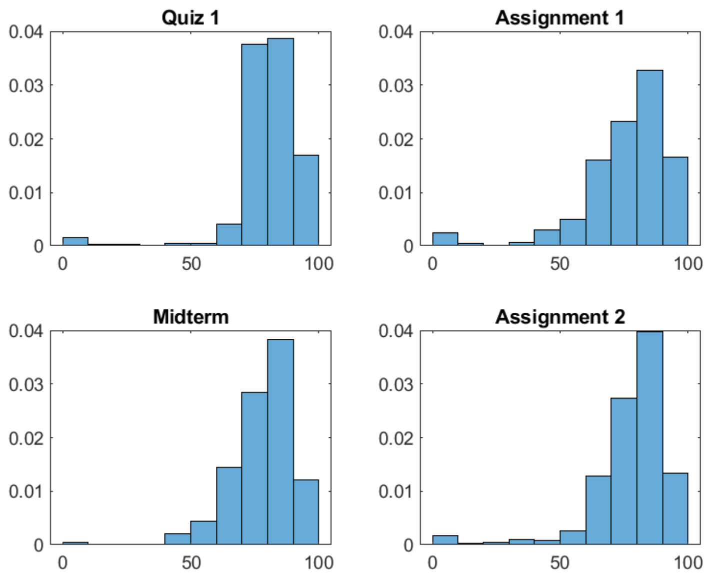


Figure 4.5: Dataset 2 - Features' distribution

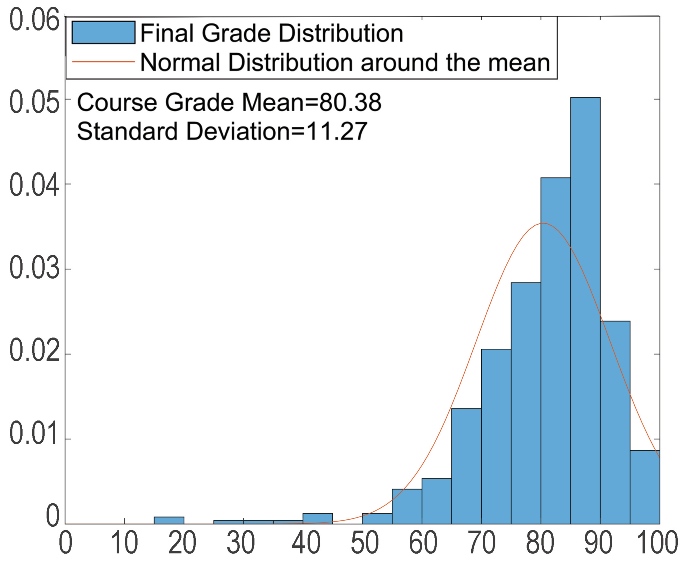


Figure 4.6: Dataset 2 - Final Grade Distribution

The distribution of the Final Grade of the first dataset is shown in Fig. 4.6.

Figures 4.4 and 4.6 show that both datasets are not normally distributed hence some classifiers are unlikely to have a good performance on the given datasets. For example, Naive

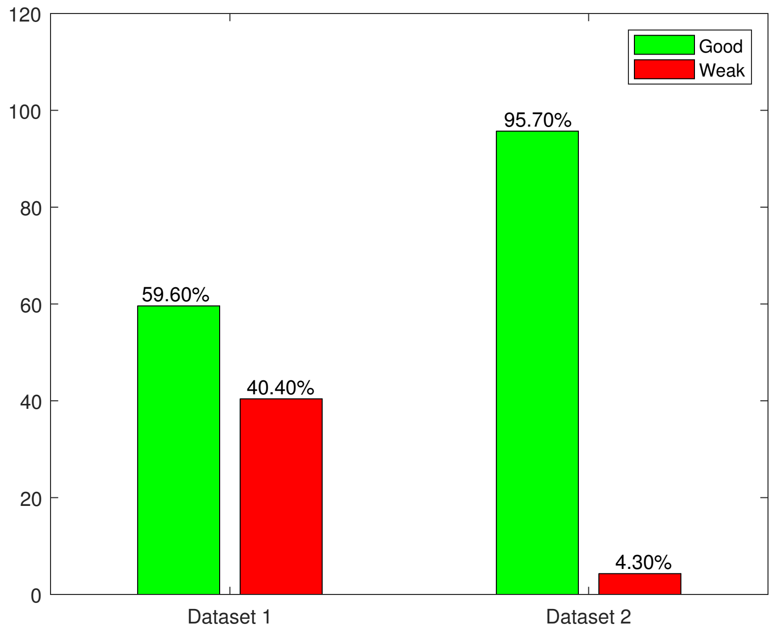


Figure 4.7: Target variable: Dataset 1 vs. Dataset 2

Bayes, is a technique that performs very well in case of normally distributed numerical input (not categorical), and this is not the case of our datasets.

Note that the Dataset 2 is unbalanced (4.3% of Weak students) whereas Dataset 1 has 40.4% of Weak students, as summarized in Figure 4.7.

4.5.2 Dataset Visualization

In machine learning problems, it is very important to visualize the dataset in order to get more understanding of the nature of data. In Section 2.4, we saw that Principal Component Analysis (PCA) can be used to reduce the number of features to two principle components and this enables us to visualize the dataset. The first and second principle components resulted from PCA were used to train SVM-RBF to plot the decision boundaries in order to understand the behavior of SVM with the given dataset.

Fig. 4.8 shows that the dataset 1 at 50% is not linearly separable because there are outlier data points. Indeed, if we were to train a linear classifier, we would be likely to obtain miss-classified points in the test sample.

Fig. 4.9 illustrates the behavior of SVM in building the decision boundary with Gaussian kernel (RBF) of dataset 2 at 50% stage. In both cases SVM-RBF model gives a better performance and it is clear that it outperforms the linear kernel (since the data is not linear) and that it is more likely to well-perform in classifying new instances.

We know from Section 2.4 that PCA shows the overall "shape" of the data, identifying which samples are similar to one another and which are very different. In other words, PCA can enable us to identify groups of samples that are similar and work out which variables make one group different from another.

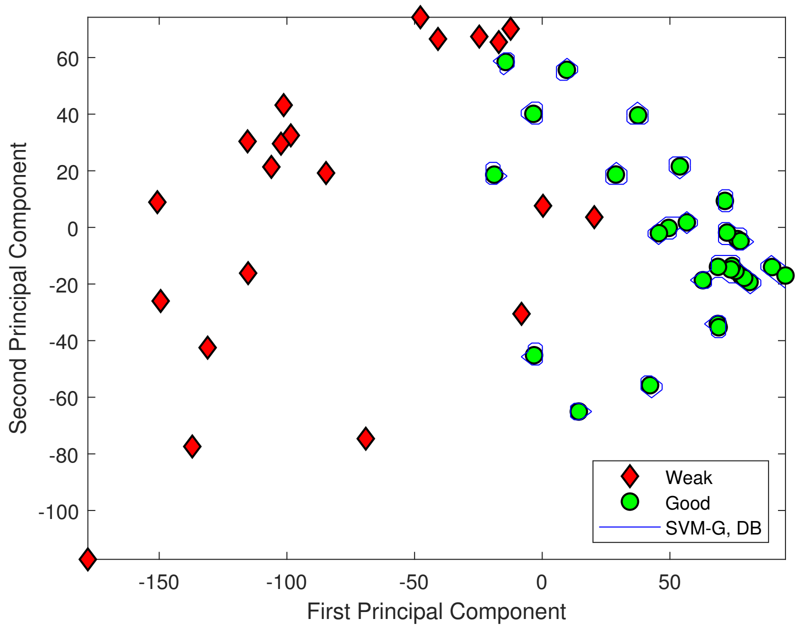


Figure 4.8: Decision boundaries for dataset 1

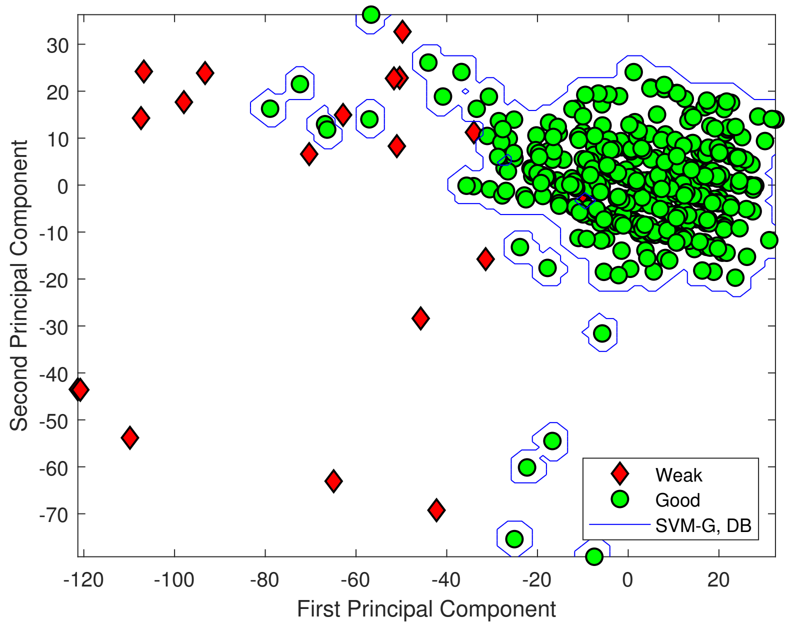


Figure 4.9: Decision boundaries for dataset 2

Performing PCA on Dataset 1 at stage 20%, we obtain that the percentage of variance for every component is as in Figure 4.10. Each component explains a percentage of the total variation in the dataset. In particular the first four components can explain 85% of the variance.

For instance, the first principal component PC1 explains 42.1% of the total variance, which means that almost 1/2 of the information in the dataset can be encapsulated by just that one

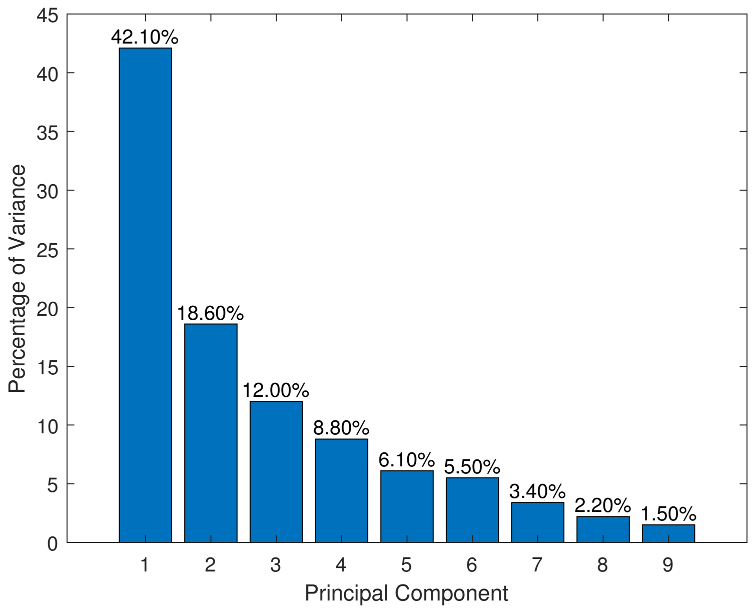


Figure 4.10: Dataset 1 - Stage 20% - Percentage of variance per principal component

Principal Component. PC1 and PC2 together can explain 60.7% of the variance with PC1 explaining 42.1% and PC2 explaining an additional 18.6%, see Figures 4.10 and 4.11.

More generally, we can plot the first four components 2 by 2 obtaining the following plot that shows in particular that there are many outliers, see Figure 4.12.

We visualize the variable contributions to the principal PC1 - PC4, aiming to give an interpretation of each principal component (see Figures 4.13, 4.14).

So we deduce that:

- PC1 corresponds to the *Analyze Task* cluster
- PC2 corresponds to the *Apply Task* cluster
- PC3 corresponds to the *Understand Task* cluster
- PC4 corresponds to the *Evaluate Task* cluster

And all these tasks are in Boolean Algebra.

Analogously, we perform PCA on Dataset 1 at stage 50%, obtaining the percentage of variance for every component as in Figure 4.10. In particular the first four components can explain 76% of the variance.

In particular, the first principal component PC1 in this case explains 40.9% of the total variance, which means that about 2/5 of the information in the dataset is described by just the first Principal Component.

PC1 and PC2 together can explain 57.8% of the variance, see Figure 4.15.

More generally, we can plot the first four components 2 by 2 obtaining the following plot that shows in particular that there are many outliers, see Figure 4.16.

We visualize the variable contributions to the principal PC1 - PC4 (see Figures 4.17, 4.18).

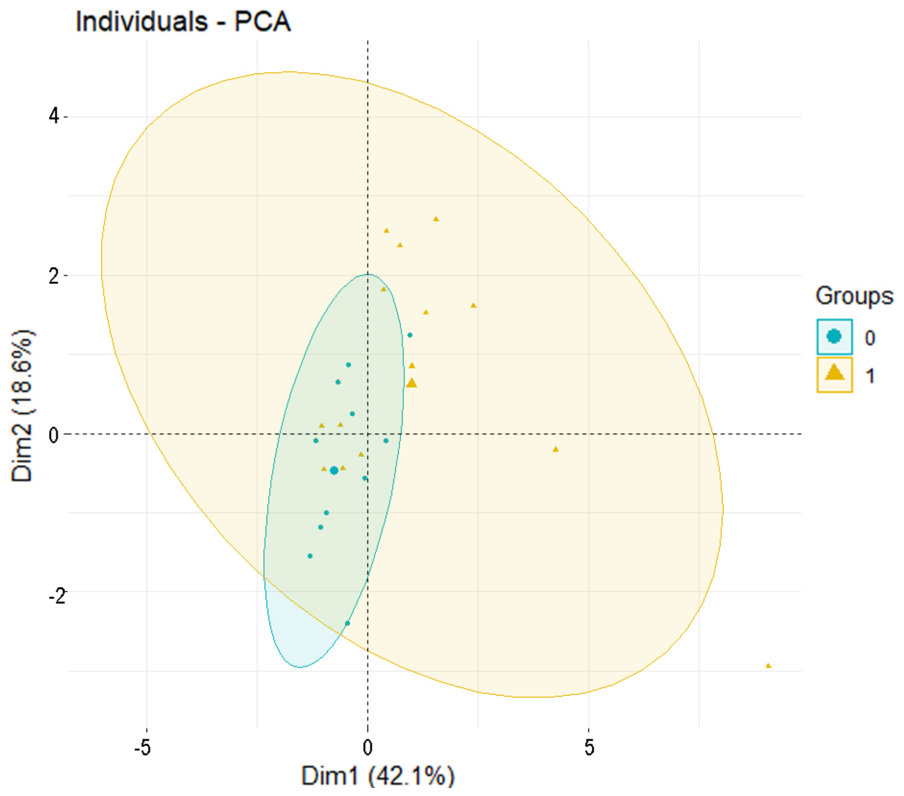


Figure 4.11: Dataset 1 - Stage 20% - First two principal components

- PC1 corresponds to the *Evaluate Task* cluster
- PC2 corresponds to the *Apply Task* cluster
- PC3 corresponds to the *Analyze Task* cluster
- PC4 corresponds to the *Understand Task* cluster

Note that the mapping between each principal component and the corresponding task cluster was performed based on the nature of the tasks/exercises forming the principal component. More specifically, the actual task content was available for dataset 1 and hence determining to which task cluster they belong to is straightforward. On the other hand, this information was not available for dataset 2.

Based on the above results, it can be inferred that tasks that fall under the *Evaluate* and *Analyze* categories based on Bloom's taxonomy (PC1 and PC2 in this case) are better indicators and predictors of student performance. This is because these task categories show the highest level of comprehension of the course material from the educational point of view. Hence, the performance of students in these tasks can provide us with intuitive insights into their overall projected performance in the course.

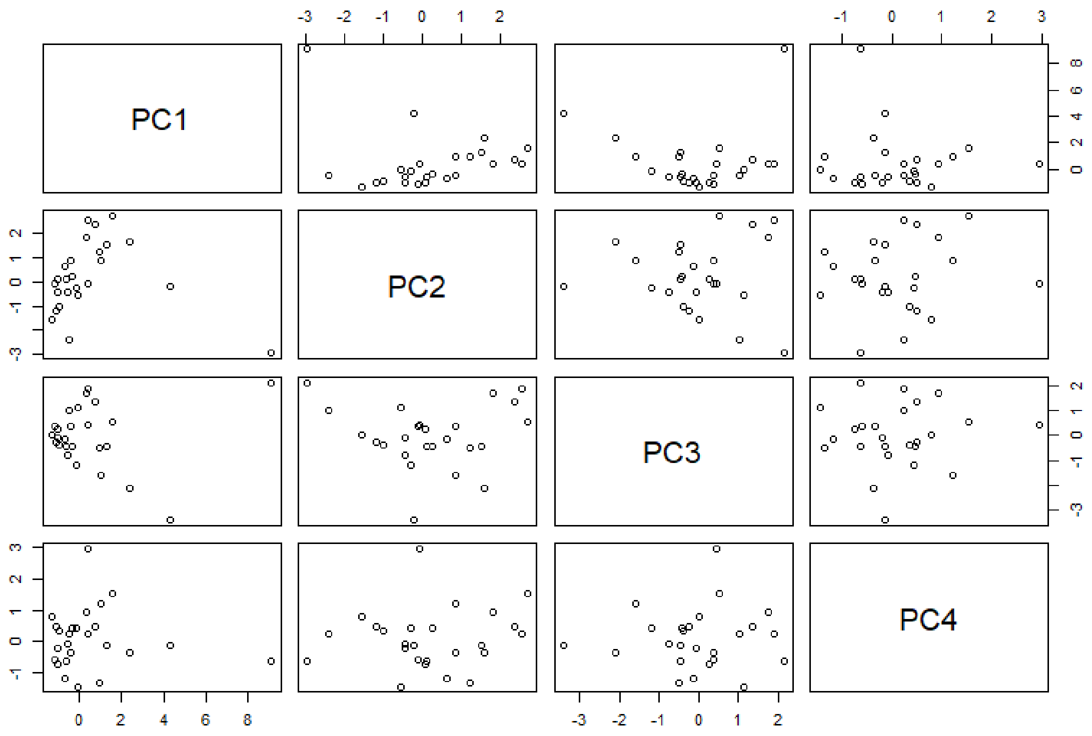


Figure 4.12: Dataset 1 - Stage 20% - First four principal components

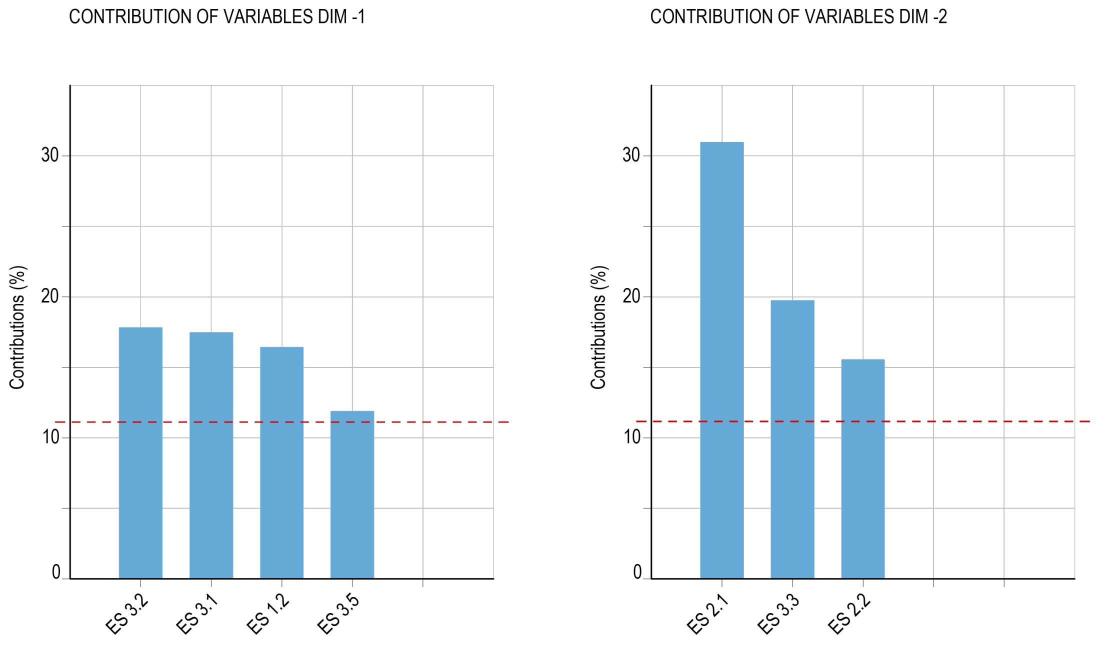


Figure 4.13: Dataset 1 - Stage 20% - First and second component

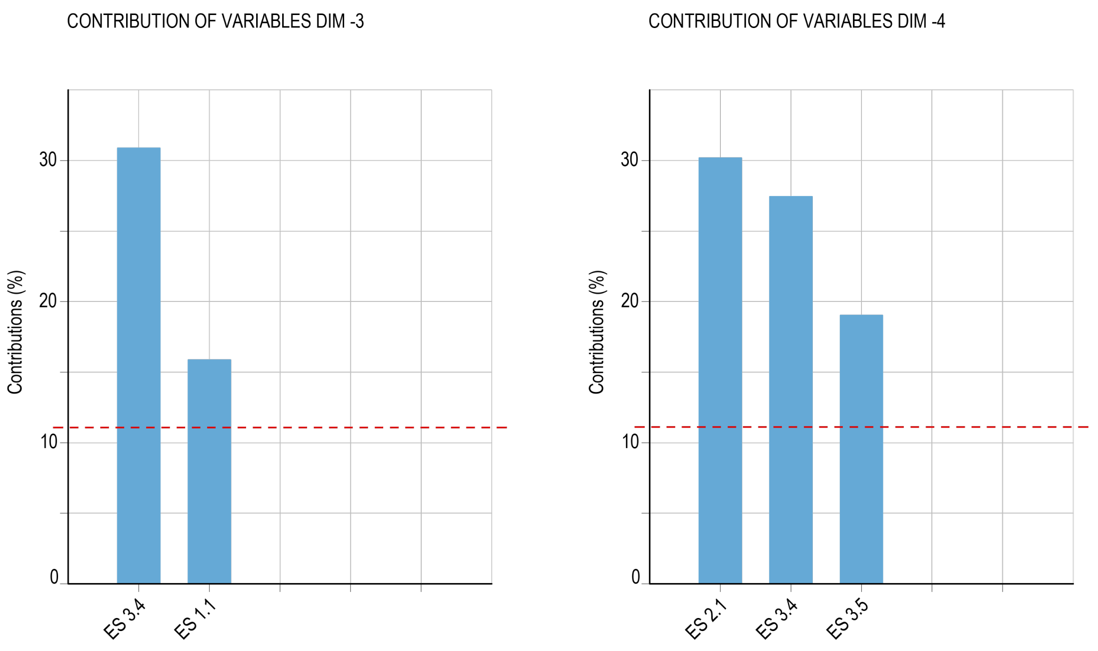


Figure 4.14: Dataset 1 - Stage 20% - Third and fourth component

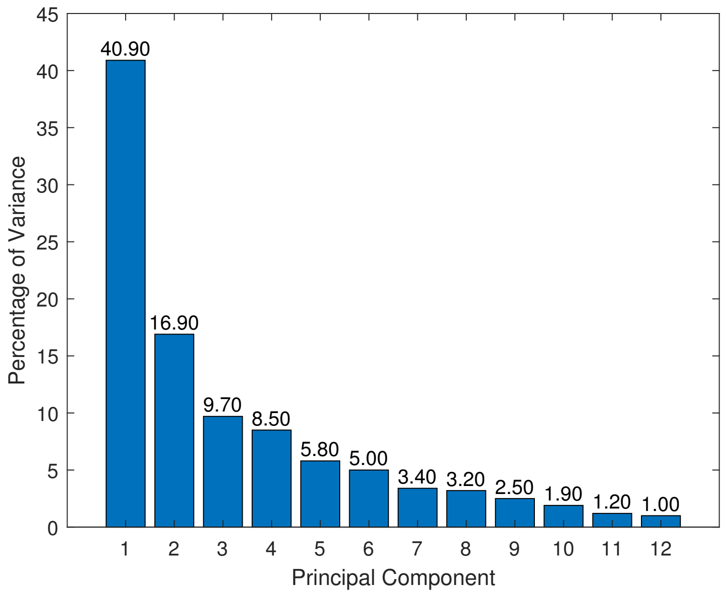


Figure 4.15: Dataset 1 - Stage 50% - Percentage of variance per principal component

4.6 ML application to the datasets

In this section we describe the classifiers we built for each of the four datasets, then we explain the approach used to select the best ensemble learners for the four datasets. Note that the models were trained on the four *raw* datasets and not on the principal components. R was used to implement six classifiers and the ensemble learners using an Intel® Core^{TM} i7 processor @

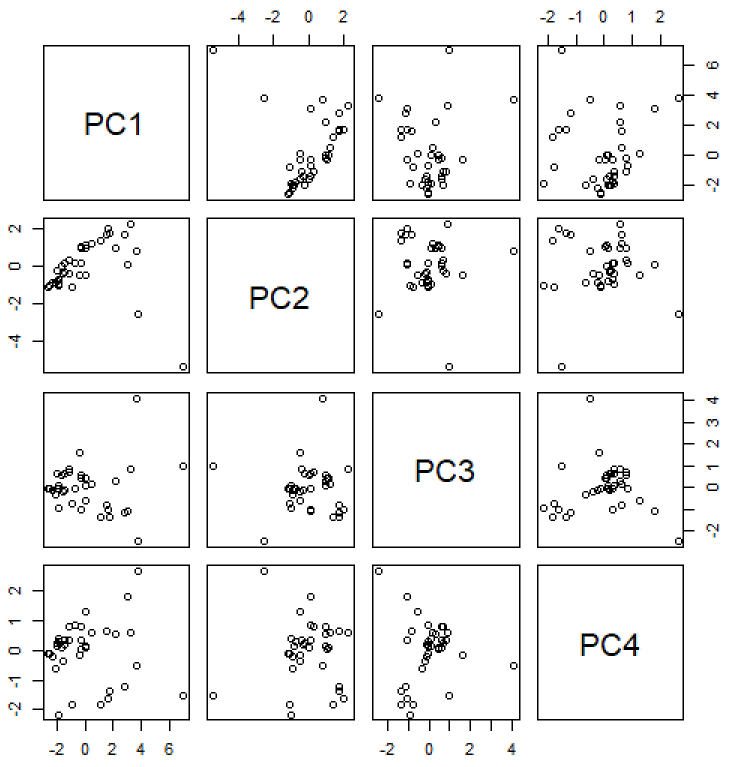


Figure 4.16: Dataset 1 - Stage 50% - First four principal components

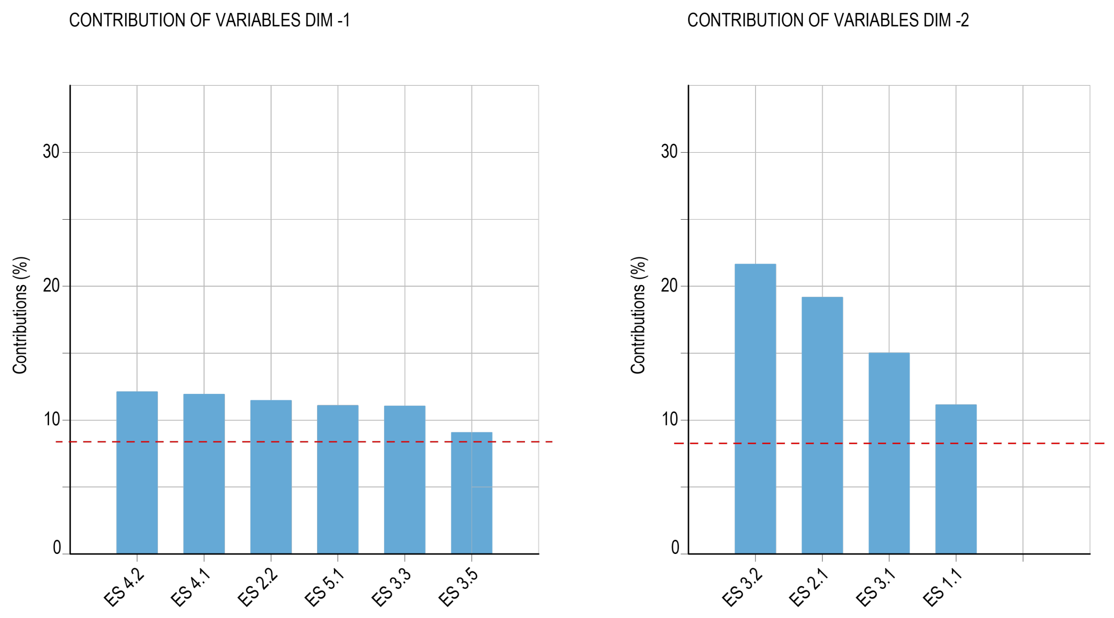


Figure 4.17: Dataset 1 - Stage 50% - First and second component

3.40 GHz system with 16GB RAM running Windows 10 operating system. The six classifiers that we trained are SVM-RBF, Logistic Regression, Naive Bayes, k-NN, Random Forest, Artificial Neural Networks. All the classifiers were trained using all the variables available and

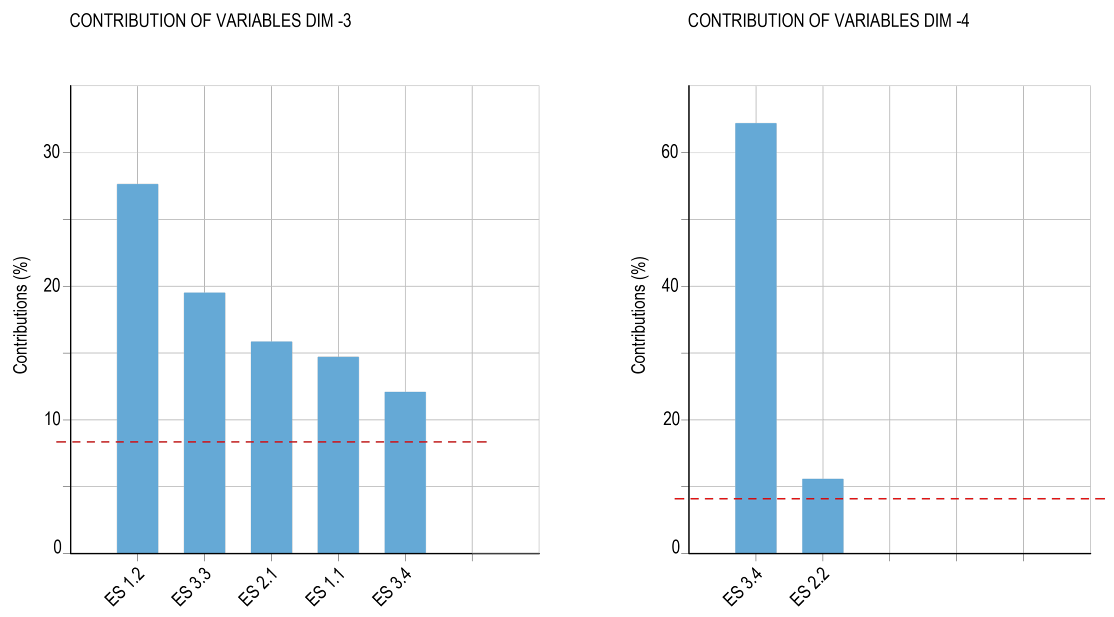


Figure 4.18: Dataset 1 - Stage 50% - Third and fourth component

maximizing the Gini Index of a 3-fold cross validation [209]. The parameters used for each model are selected from a *grid* of parameters and the set of parameters is chosen so that the Gini Index is maximized.

For each algorithm and each dataset we show the list of the features ordered by their importance, i.e. their impact on the predictions. This is meant to give only a rough idea of what the most important features are for each algorithm and each dataset, as the ordering, for such small datasets, heavily depends on the split in Train-Test samples chosen. For this reason, the weights of the predictors will not be specified.

The final step was to select, for each problem, the best ensemble learner among all the possible ensemble learners that could be produced with the six classifiers.

4.6.1 Dataset 1 - Stage 20%

- *Random Forest:* The classifier was trained using k-fold cross-validation with $k = 3$. Figure 4.19 shows how the performance changes by choosing a different *mtry* value, i.e. the number of variables available for splitting at each tree node.
  The variables' importance in terms of predictivity is described in Table 4.3 that shows that the most relevant feature is ES3.3, followed by ES3.5, whereas ES1.1 does not have much impact on the predictions.

- *SVM-RBF*: SVM Algorithm was trained with radial basis function kernel. Table 4.4 shows the list of the predictors ordered by their impact on the output. In particular we can see from the table that the top three variables are ES2.2, ES3.3 and ES3.5 and that ES1.1 and ES3.2 do not have much impact on the predictions.

- *MLP:* The variables' importance for MLP classifier is shown in Table 4.5.

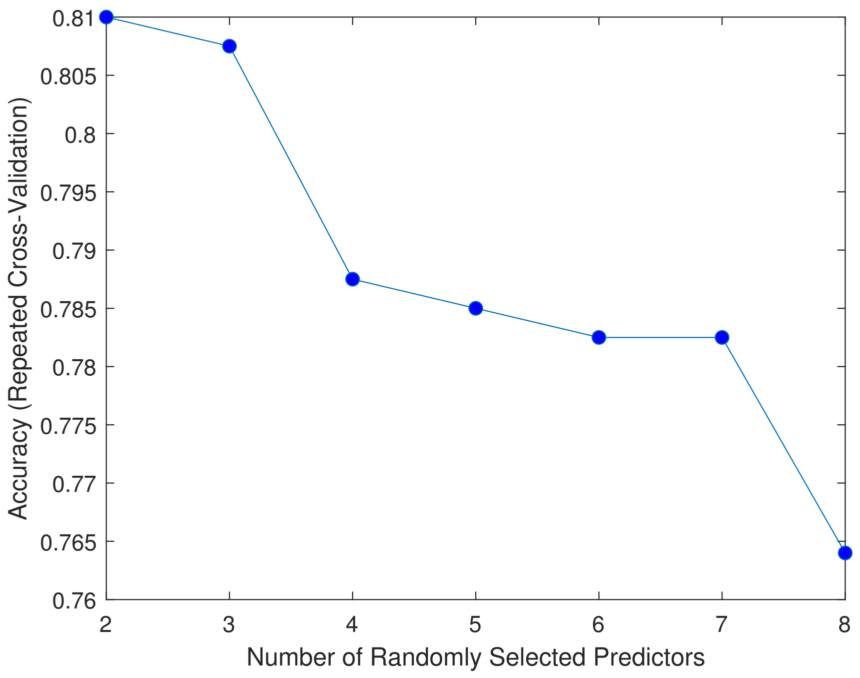


Figure 4.19: Dataset 1 - Stage 20% - Random Forest, Accuracy vs. mtry value

Ranking	**Feature**
1	ES3.3
2	ES3.5
3	ES2.2
4	ES3.4
5	ES3.1
6	ES1.2
7	ES3.2
8	ES2.1
9	ES1.1

Table 4.3: Dataset 1 - Stage 20% - Features' weights for Random Forest

- *k-NN* We trained k-NN classifier trying different values for k. Figure 4.20 shows how the performance changes as k changes. For Dataset 1, stage 20%, the best performance was obtained for $k = 9$ and the list of variables ordered by their importance is shown in Table 4.6, which is the same as the one obtained for MLP and SVM classifiers.

- *Logistic Regression* For the Logistic Regression classifier, variables ES3.1, ES3.2 and ES1.2 have no impact on the predictions. The most important variables for this algorithm are ES2.2 and ES3.3, as it is shown in Table 4.7.

- *Naive Bayes* The variables' importance for the Naive Bayes classifier is the same ob-

Ranking	**Feature**
1	ES2.2
2	ES3.3
3	ES3.5
4	ES3.1
5	ES3.4
6	ES2.1
7	ES1.2
8	ES1.1
9	ES3.2

Table 4.4: Dataset 1 - Stage 20% - Features' weights for SVM-RBF

Ranking	**Feature**
1	ES2.2
2	ES3.3
3	ES3.5
4	ES3.1
5	ES3.4
6	ES2.1
7	ES1.2
8	ES1.1
9	ES3.2

Table 4.5: Dataset 1 - Stage 20% - Features' weights for MLP

Ranking	**Feature**
1	ES2.2
2	ES3.3
3	ES3.5
4	ES3.1
5	ES3.4
6	ES2.1
7	ES1.2
8	ES1.1
9	ES3.2

Table 4.6: Dataset 1 - Stage 20% - Features' weights for k-NN

tained for MLP, SVM, and k-NN classifiers, see Table 4.8.

In general, the most important features for all the classifiers are ES2.2, ES3.3 and ES3.5, that

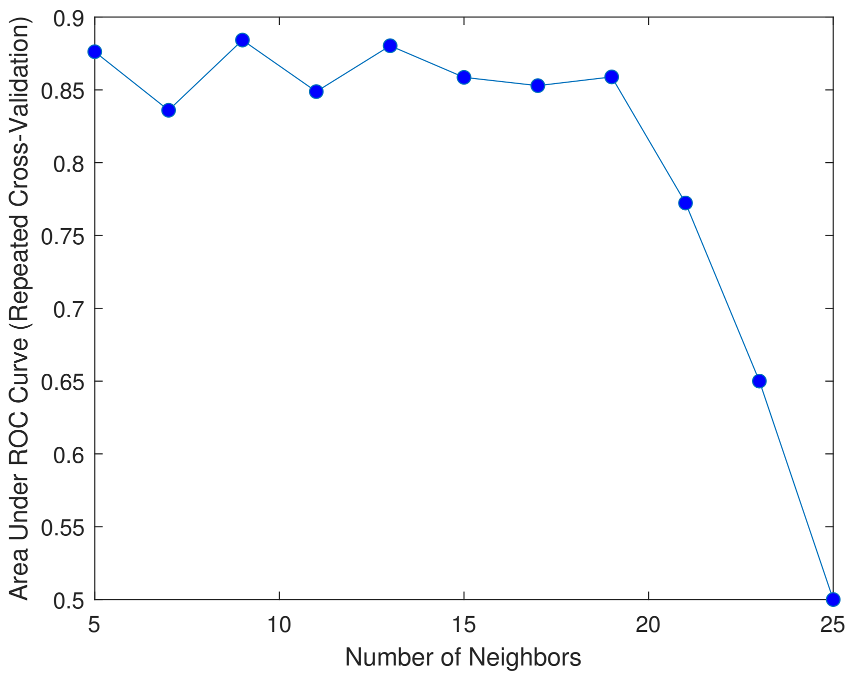


Figure 4.20: Dataset 1 - Stage 20% - Performance vs. k

Ranking	**Feature**
1	ES2.2
2	ES3.3
3	ES3.5
4	ES3.4
5	ES2.1
6	ES1.1
7	ES3.1
8	ES3.2
9	ES1.2

Table 4.7: Dataset 1 - Stage 20% - Features' weights for Logistic Regression

contributed to the first and second principal components, as we saw in Figure 4.13.

4.6.2 Dataset 1 - Stage 50%

- *Random Forest:* Figure 4.21 shows how the performance changes by choosing a different *mtry* value, i.e. the number of variables available for splitting at each tree node. The variables' importance in terms of predictivity is described in Table 4.9 that shows that the most relevant feature is ES4.2, followed by ES4.1 , whereas ES3.2 does not have much impact on the predictions. Also note that the bottom 3 variables are the same as

Ranking	**Feature**
1	ES2.2
2	ES3.3
3	ES3.5
4	ES3.1
5	ES3.4
6	ES2.1
7	ES1.2
8	ES1.1
9	ES3.2

Table 4.8: Dataset 1 - Stage 20% - Features' weights for Naive Bayes

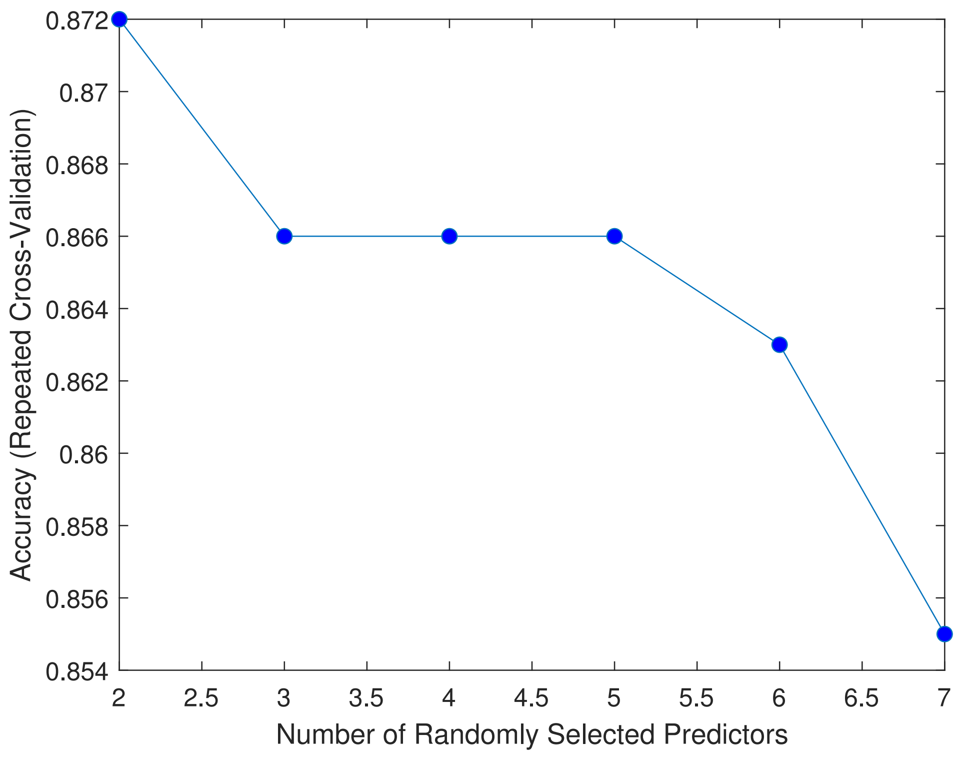


Figure 4.21: Dataset 1 - Stage 50% - Random Forest, Performance vs. number of features selected

the ones shown for Random Forest classifier on Dataset 1, at stage 20%.

- *SVM-RBF:* The list of predictors for SVM classifier, ordered by their importance is shown in Table 4.10

- *MLP:* MLP classifier shares with SVM classifier the same table of variables' importance, see Table 4.11.

Ranking	Feature
1	ES4.2
2	ES4.1
3	ES5.1
4	ES3.5
5	ES3.3
6	ES2.2
7	ES3.1
8	ES3.4
9	ES2.1
10	ES1.2
11	ES1.1
12	ES3.2

Table 4.9: Dataset 1 - Stage 50% - Features' weights for Random Forest

Ranking	Feature
1	ES4.1
2	ES4.2
3	ES5.1
4	ES2.2
5	ES3.3
6	ES3.5
7	ES3.1
8	ES3.4
9	ES2.1
10	ES1.2
11	ES1.1
12	ES3.2

Table 4.10: Dataset 1 - Stage 50% - Features' weights for SVM-RBF

- *k-NN:* We tried different values for k when we trained k-NN classifier and Table 4.22 shows how the performance changes as k changes. $k = 5$ was our final choice fo Dataset 1 at stage 50%. As forDataset 1 at stage 20%, the list of variables ordered by their importance, shown in Table 4.12, is the same as the one obtained for MLP and SVM classifiers.

- *Logistic Regression* For the Logistic Regression classifier, variables ES2.2, ES3.2 and ES1.2 have no impact on the predictions. The most important variable for this algorithm is ES4.1 that weighs almost 41% on the predictions, as Table 4.13 shows.

- *Naive Bayes* The variables' importance for the Naïve Bayes classifier is the same ob-

Ranking	**Feature**
1	ES4.1
2	ES4.2
3	ES5.1
4	ES2.2
5	ES3.3
6	ES3.5
7	ES3.1
8	ES3.4
9	ES2.1
10	ES1.2
11	ES1.1
12	ES3.2

Table 4.11: Dataset 1 - Stage 50% - Features' weights for MLP

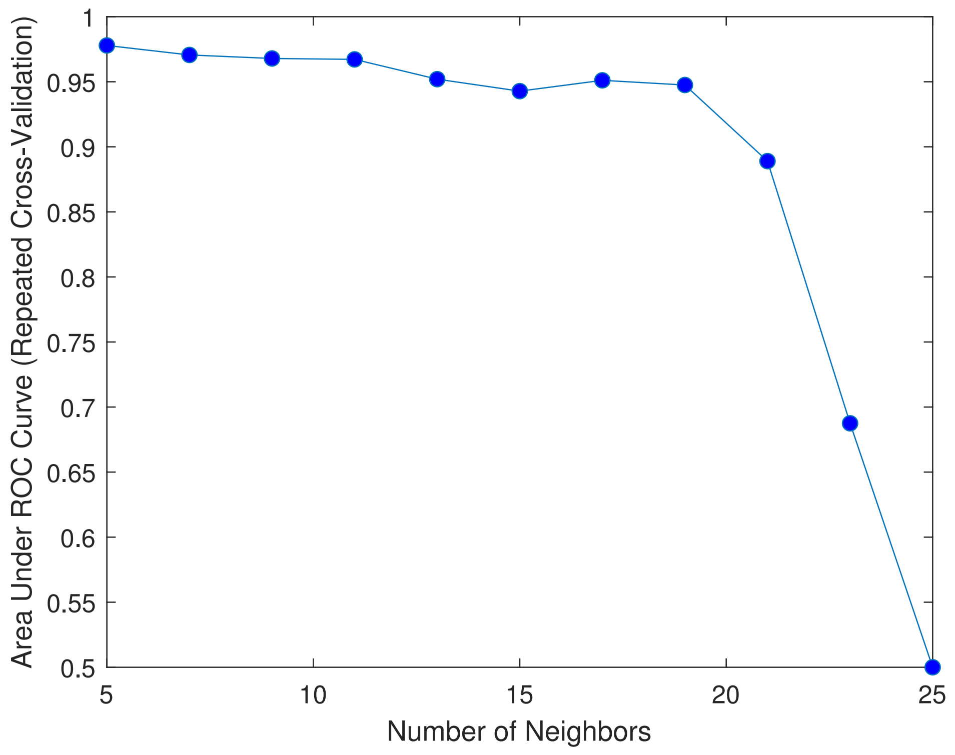


Figure 4.22: Dataset 1 - Stage 50% - Performance vs. k

tained for MLP, SVM, and k-NN classifiers, see Table 4.14.

In general, the most important features for almost all the classifiers are ES4.1, ES4.2 and ES5.1, that contributed to the first principal components, as we saw in Figure 4.17. The only classifier

Ranking	**Feature**
1	ES4.1
2	ES4.2
3	ES5.1
4	ES2.2
5	ES3.3
6	ES3.5
7	ES3.1
8	ES3.4
9	ES2.1
10	ES1.2
11	ES1.1
12	ES3.2

Table 4.12: Dataset 1 - Stage 50% - Features' weights for k-NN

Ranking	**Feature**
1	ES4.1
2	ES4.2
3	ES3.3
4	ES1.1
5	ES3.4
6	ES5.1
7	ES3.1
8	ES1.2
9	ES2.1
10	ES3.5
11	ES3.2
12	ES2.2

Table 4.13: Dataset 1 - Stage 50% - Features' weights for Logistic Regression

that does not have ES5.1in the top three variables is te Logistic Regression classifier which has ES3.3 in third position instead. Also variable ES3.3 belongs to the first principal component.

4.6.3 Dataset 2 - Stage 20%

We have only two features for Dataset 2, Stage 20% and for all the classifiers, the list of features ordered by importance is the same, see Table 4.15.

Ranking	Feature
1	ES4.1
2	ES4.2
3	ES5.1
4	ES2.2
5	ES3.3
6	ES3.5
7	ES3.1
8	ES3.4
9	ES2.1
10	ES1.2
11	ES1.1
12	ES3.2

Table 4.14: Dataset 1 - Stage 50% - Features' weights for Naïve Bayes

Ranking	Feature
1	Assignment01 [8]
2	Quiz01 [10]

Table 4.15: Dataset 2 - Stage 20% - Features' weights

4.6.4 Dataset 2 - Stage 50%

For Random Forest, SVM, MLP, k-NN and Naive Bayes the lists of features are ordered in the same way, as shown in Table 4.16.
For Logistic Regression the list order by importance is slightly different, see Table 4.17

Ranking	Feature
1	Assignment02 [12]
2	Assignment01 [8]
3	Quiz01 [10]
4	Midterm Exam [20]

Table 4.16: Dataset 2 - Stage 50% - weights RF, SVM, MLP, k-NN and NB

Ranking	Feature
1	Assignment02 [12]
2	Quiz01 [10]
3	Midterm Exam [20]
4	Assignment01 [8]

Table 4.17: Dataset 2 - Stage 50% - Features' weights for LR

4.6.5 Proposed method Ensemble learners: a systematic approach

For each dataset and each stage, a systematic approach was used to select the best subset of classifiers to consider for the ensemble learner. More specifically, the procedure was to evaluate the performance of every possible combination of the classifiers that we trained.

The performance of each model was measured in terms of Gini Index. We inferred each model on the test sample producing a score. Each score corresponds to the probability of being a Weak student.

Although confusion matrices present a clear picture of correct and incorrect classifications for each class of objects, they are affected by the choice of a threshold on the probability score. For this reason, instead of evaluating every model basing our considerations on the confusion matrices, we will rely on the Gini Index instead.

The statistical significance of our results is determined by computing the p-values. The general approach is to test the validity of a claim, called the *null hypothesis*, made about a population. An alternative hypothesis is the one you would believe if the null hypothesis is concluded to be untrue.

Hypothesis tests use a p-value to weigh the strength of the evidence:

- A small p-value (≤ 0.05) indicates strong evidence against the null hypothesis, so you reject the null hypothesis.
- A large p-value (> 0.05) indicates weak evidence against the null hypothesis, so you fail to reject the null hypothesis.
- p-values very close to the cutoff (0.05) are considered to be marginal (could go either way).

For our purposes, the null hypothesis states that the Gini Indices were obtained by chance. We generated 1 million random scores from normal distribution and calculated the p-value. *The ensemble learners selected have p-value* ≤ 0.05*, indicating that there is strong evidence against the null hypothesis.*

For each model we created a matrix consisting of the target variable and the score produced by the model, then we ordered the matrix by the score in descending order. In this way, on top we can find the students which are more likely to be Weak students as opposed to the bottom of the matrix where we find the students who are less likely to be Weak students.
To create a confusion matrix we set a threshold for each model, as explained in Section 2.3. Finally, for each dataset, we generated all the possible combinations of the six models and calculated the corresponding Gini Index.The procedure followed to produce each ensemble learner can be summarized in Figure 4.23.

Since the training and test samples are small sized, many of the ensemble learners produced had the same Gini Index and the performances seemed to depend on the split into training and test samples that was chosen at the beginning. For instance, Figure 4.24 shows the performance of the different classifiers on Dataset 1 at stage 20% on different splits. For example, LR performance is really good on the first two splits considered (with Gini Index 88.9% and 76% respectively), whereas on the fourth split it performs very poorly (Gini Index is only 17.8%). In contrast, the RF classifier shows a more consistent performance across the multiple splits.

Since the performance of the classifiers depends very much on the split, and so do all the ensembles, instead of considering only one split we considered 5 additional splits of the dataset, namely split1= (Training1, Test1), split2 = (Training2, Test2), split3 = (Training3, Test3), split4 = (Training4, Test4), split5 = (Training5, Test5), and ran the 6 algorithms on each split, training a total of 6×5 models, each one run using a 3-fold method and keeping track of the performances of each model also on the folds. We average of the performance across the splits to remove any potential bias. Note that every split was created randomly, like for the initial training and test samples, and in such a way that the target variable of each training and test sample was representative of the entire dataset.

Afterwards, we compared the performances of the models of each split, and produced one table of all possible ensembles for each one.
Although from the literature we expect the ensemble learners to perform better than the individual classifiers, we included in the comparison also the individual classifiers and considered

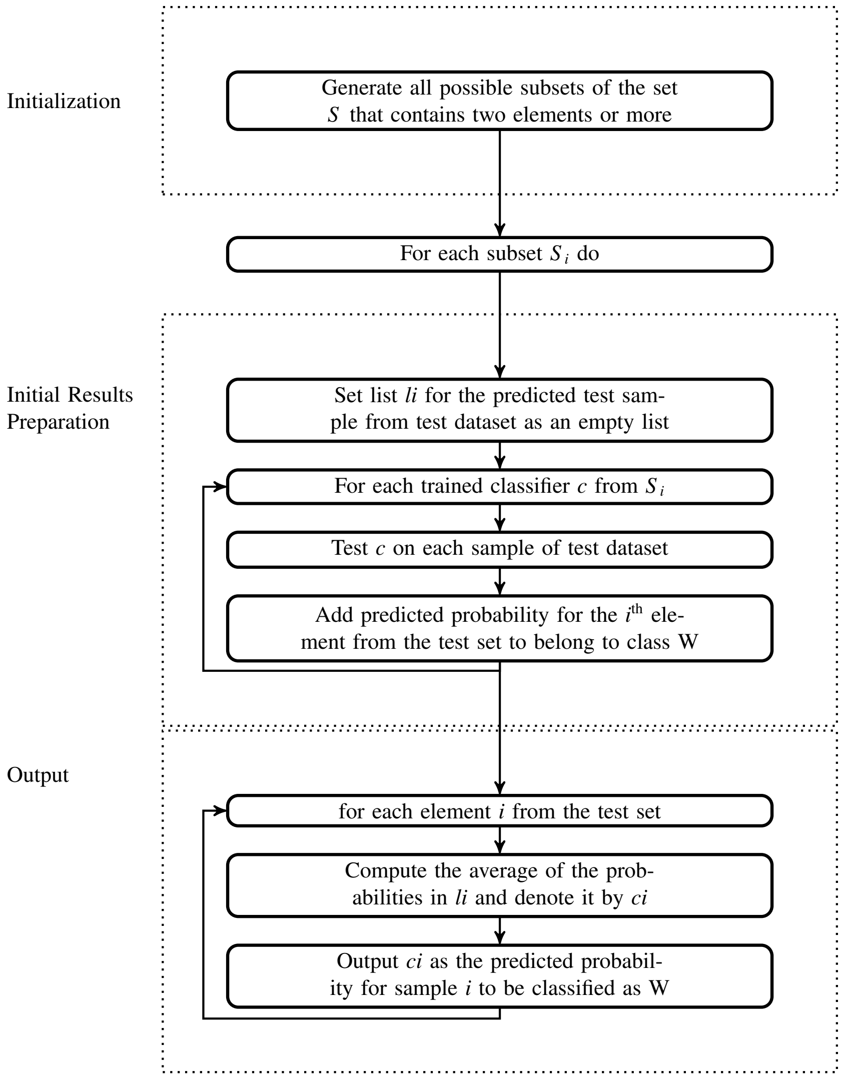

Initialization
Generate all possible subsets of the set S that contains two elements or more
For each subset S_i do
Initial Results Preparation
Set list li for the predicted test sample from test dataset as an empty list
For each trained classifier c from S_i
Test c on each sample of test dataset
Add predicted probability for the i^{th} element from the test set to belong to class W
Output
for each element i from the test set
Compute the average of the probabilities in li and denote it by ci
Output ci as the predicted probability for sample i to be classified as W

Figure 4.23: The procedure of generating ensemble learners and creating the score for each ensemble learner

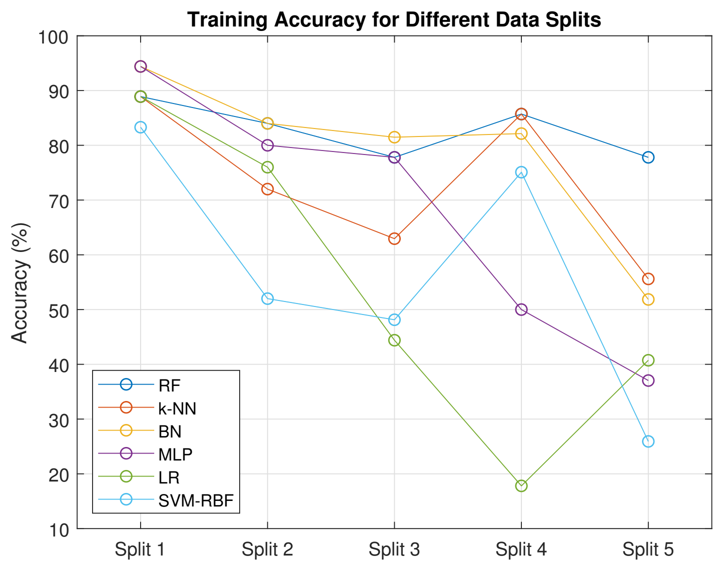


Figure 4.24: Dataset 1 - Stage 20% - Performance of different classifiers on 5 splits

64 combinations of classifiers as opposed to 57 (the actual ensemble learners).

Finally we created a table consisting of 64 rows, each one representing a specific ensemble, where on each row we have, (see Appendix A):

- the first 6 entries, one for each algorithm, with 1's and 0's corresponding to presence or absence of the algorithm in the ensemble. In particular, the individual classifiers correspond to those rows for which the sum of the first six entries is one.

- entries 7,8,9,10,11,12 corresponding to the Gini index, namely G, G1, G2, G3, G4, G5, associated to the initial split, split1, split2, split3, split4, split5.

- entry 13, called Avg, corresponding to the average of G, G1, G2, G3, G4 and G5.

The table was ordered with respect to Avg, in descending order, and the top ensemble was selected as best classifier. Moreover, the p-values were calculated and shown in the table in the 14-th column called p, and prove that the ensemble selected is statistically significant.

For every model, we will show the corresponding confusion matrices associated to the Training and Test samples. They can be used to get an insight into the number of correct and failed predictions of one class opposed to other class.

4.7 Results and Discussion

In this section we discuss the results and select an ensemble learner for each of the four experiments. Finally we set a threshold and produce and compare the corresponding confusion matrices.

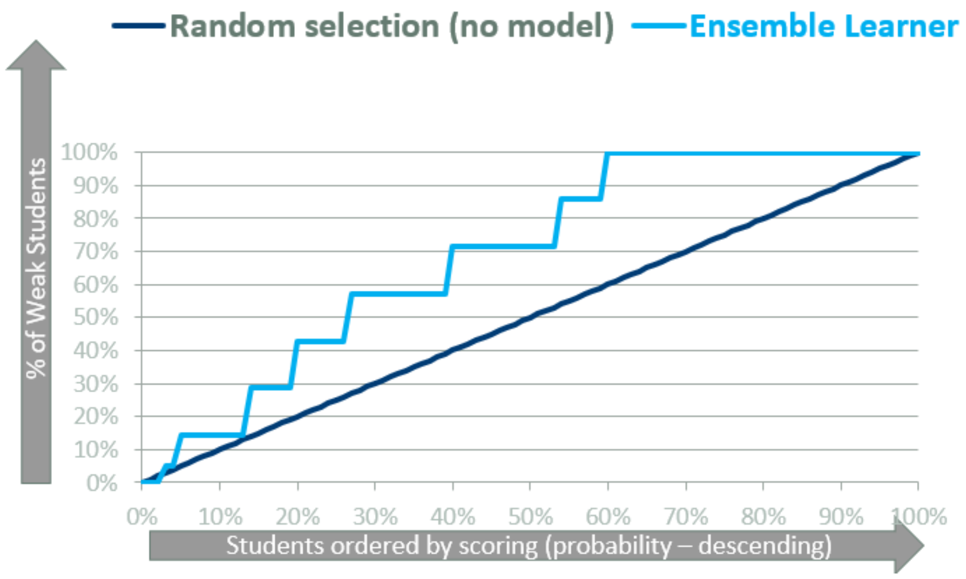


Figure 4.25: Dataset 1 - Stage 20% - Performance Ensemble Learner

4.7.1 Results: Dataset 1 - Stage 20%

As explained, 30 models were trained, 6 on each of the 5 splits. The top 3 models in terms of Gini Index are Random Forest, Naive Bayes and k-NN.
Once created the matrix with all possible ensemble learners as explained before, we ordered it with respect to Average Gini Index.
Table 4.18 consists of the best 3 classifiers (one of each row). For the complete table of all the possible combinations and corresponding Gini Index, please see Appendix A.

rf	mlp	bn	knn	lreg	svm	G	G1	G2	G3	G4	G5	Avg	p
1	0	1	0	0	0	0.750	0.899	0.880	0.815	0.857	0.778	0.828	0.0034
1	0	1	1	0	0	0.679	0.944	0.840	0.852	0.821	0.815	0.825	0.0034
1	0	1	0	0	0	0.786	0.899	0.840	0.778	0.857	0.778	0.821	0.0045

Table 4.18: Dataset 1 - Stage 20% - Best classifiers

In the table, the 1's correspond to the presence of the model in the ensemble whereas the 0's indicate that the corresponding model should not be included. We select, for Dataset 1 at stage 20%, the ensemble corresponding to the first row, i.e. the ensemble formed by Random Forest and Naïve Bayes. The Gini Index associated with this ensemble learner for the original split is 75%.
Although this Gini Index is not the highest reached on the initial dataset, we believe that the ensemble chosen is more robust as it comes from the test on 6 different splits. The Gini Index of the ensemble chosen corresponds to the area between the model curve (light blue) and the straight line (in dark blue – no model), in Figure 4.25.

Figure 4.26 shows that the number of Weak Students decreases by 100 times, moving from Highest scoring to Lowest Scoring. Setting a threshold for the probabilities we can built confusion matrices for every classifier and for the ensemble learner we selected, see Appendix

A. Although the ensemble we selected does not show either the lowest false positive rate or the highest accuracy, it is more robust than each individual classifier, i.e. depends less on the split, hence it is more reliable when inferred on a new dataset.

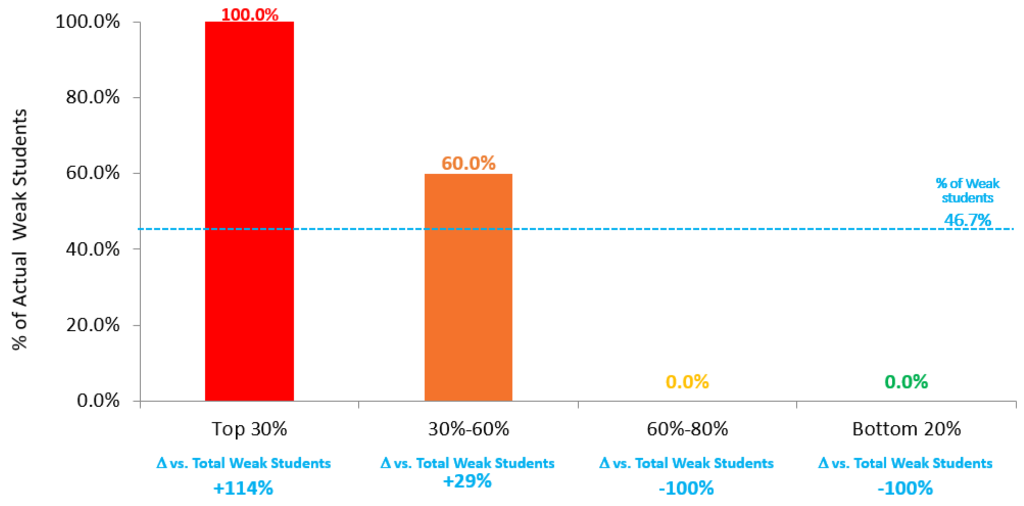


Figure 4.26: Dataset 1 - Stage 20% - Performance Ensemble Learner

4.7.2 Results: Dataset 1 - Stage 50%

Following the same procedure, we trained 30 models and compared the performances of the inferences on each test. Random Forest and k-NN have the best performances, whereas Logistic Regression has the worst performance on average on the datasets. The results obtained for the 3-folds agree with the ones obtained on the test samples: the top 3 models in terms of Gini Index are Random Forest, Naive Bayes and k-NN. The best 3 ensembles are shown in Table 4.19.

rf	mlp	bn	knn	lreg	svm	G	G1	G2	G3	G4	G5	Avg	p
1	0	0	0	0	1	0.929	1	1	1	0.929	1	0.976	0.00036
1	0	0	1	0	1	0.929	1	1	1	0.929	1	0.976	0.00036
1	1	0	1	0	1	0.929	1	1	1	0.929	1	0.976	0.00036

Table 4.19: Dataset 1 - Stage 50% - Best Classifiers

The top three rows of the matrix of all possible ensemble learners, that was ordered with respect to the Avg, have same Gini Index and same p-value.

Although they all are good candidates to be selected, we decide not to choose the third ensemble that includes MLP classifier as it performed very poorly on certain splits. Since k-NN had very good performances on all splits, we decide to include it in the ensemble. In conclusion, we choose the ensemble corresponding to the second row of Table 4.19, i.e. the ensemble formed by Random Forest, k-NN and Support Vector Machine classifiers.

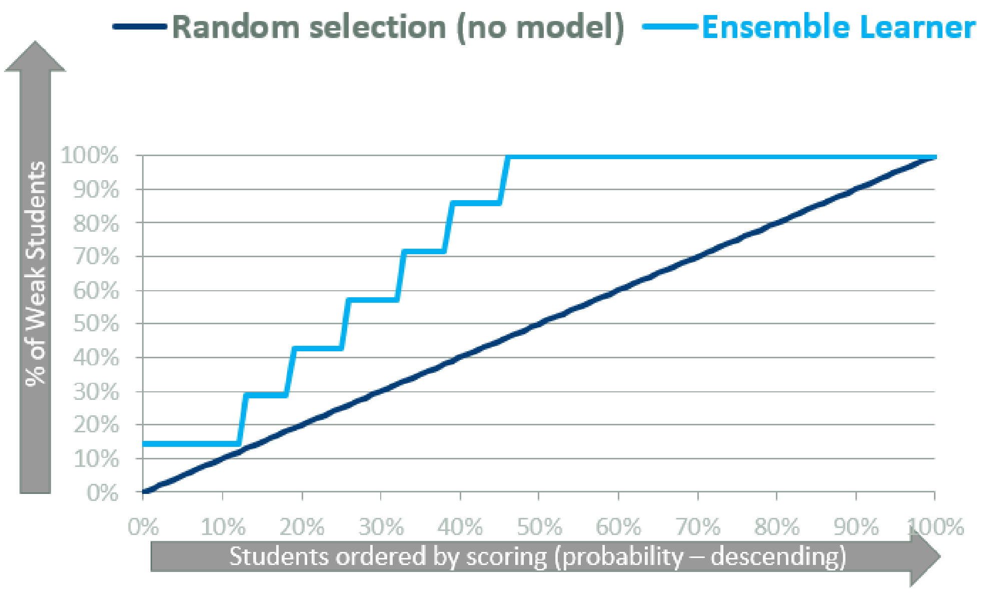


Figure 4.27: Dataset 1 - Stage 50% - Performance Ensemble Learner

The ensemble learner has Gini Index = 92.9%, represented by the area between the model curve and the straight line in Figure 4.27. In particular, we can see from Figure 4.27 that if we order the students by their probability of being a Weak student, we get 60% of Weak students in the first 30% of students, and 100% of Weak students in the first 50%, as opposed to only 30% and 50% respectively if we were not to use the model. Figure 4.28 shows that the number of Weak Students decreases by 100 times, moving from Highest scoring to Lowest Scoring.

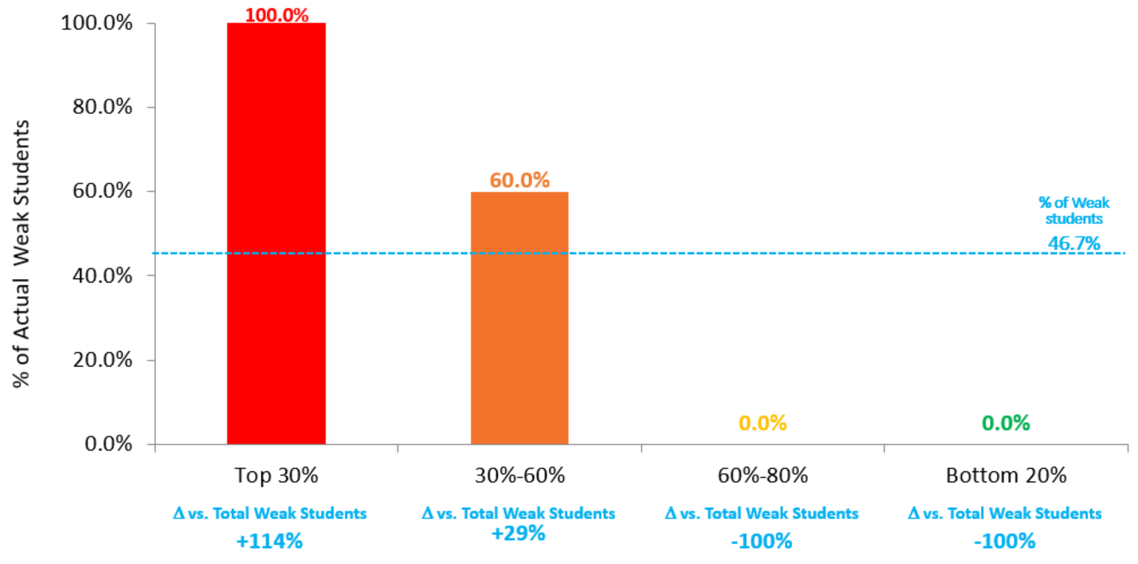


Figure 4.28: Dataset 1 - Stage 50% - Performance Ensemble Learner

4.7.3 Results: Dataset 2 - Stage 20%

In the same way, we trained 30 models and compared the performances of the inferences on each test. For this dataset, Random Forest, SVM and k-NN do not have good perfomances. The best classifiers in this case are Logistic Regression, MLP and Naïve Bayes, and the results obtained for the 3-folds agree with the ones obtained on the test samples. The best 3 ensemble learners for Dataset 2 at stage 20% are shown in Table 4.20.

rf	mlp	bn	knn	lreg	svm	G	G1	G2	G3	G4	G5	Avg	p
0	0	1	0	1	0	0.89	0.698	0.872	0.846	0.849	0.863	0.8363	0.0000024
0	0	0	0	1	0	0.888	0.702	0.876	0.84	0.856	0.854	0.8360	0.0000024
0	1	1	0	0	0	0.882	0.667	0.872	0.834	0.867	0.894	0.8360	0.0000032

Table 4.20: Dataset 2 - Stage 20% - Best Classifiers

Hence, for this dataset we select the ensemble learner formed by Naïve Bayes and Logistic Regression. The Gini Index of the ensemble selected is 89% and is represented by the area between the model curve and the straight line.

In particular, 100% of Weak students are identified by the ensemble learner in the first

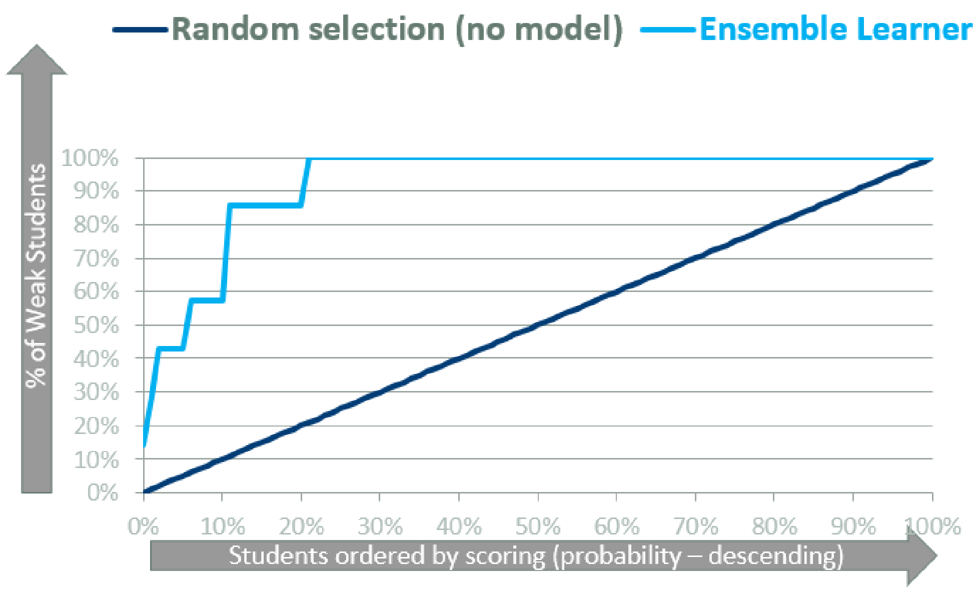


Figure 4.29: Dataset 2 - Stage 20% - Performance Ensemble Learner

21.4% of students ordered by scoring, in descending order.

4.7.4 Results: Dataset 2 - Stage 50%

For this dataset, Random Forest, SVM and k-NN do not have good performances. The best classifiers in this case are Logistic Regression, MLP, followed by Naive Bayes. For Dataset 2 at stage 50% we select the ensemble learner formed by MLP and Logistic Regression as per Table 4.21.

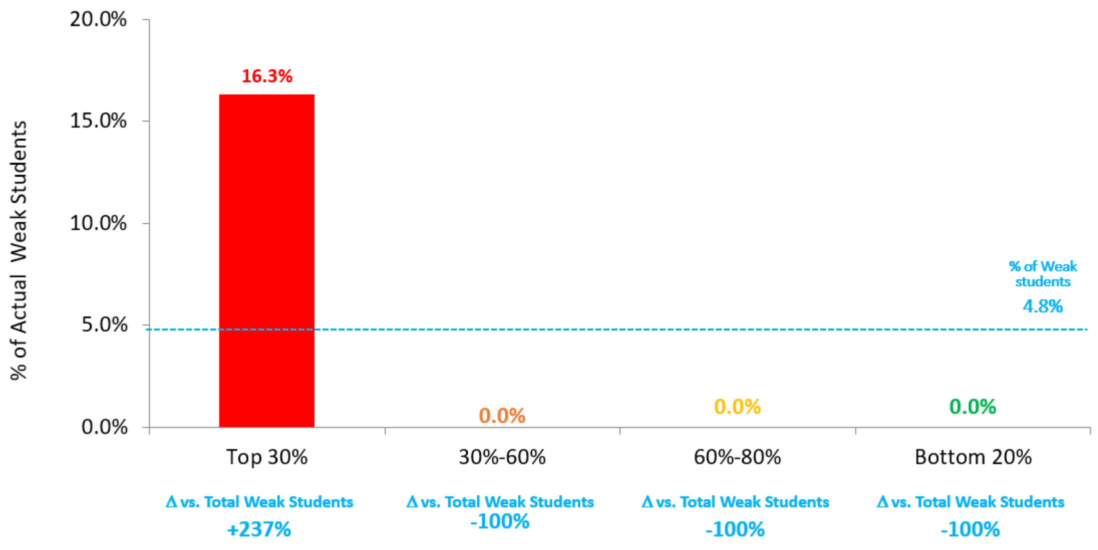


Figure 4.30: Dataset 2 - Stage 20% - Performance Ensemble Learner

rf	mlp	bn	knn	lreg	svm	G	G1	G2	G3	G4	G5	Avg	p
0	1	0	0	1	0	0.899	0.888	0.925	0.977	0.929	0.988	0.934	0.00012
0	0	0	0	1	0	0.934	0.876	0.923	0.98	0.902	0.983	0.933	0.00001
0	1	0	0	0	0	0.859	0.886	0.932	0.98	0.929	0.981	0.928	0.00075

Table 4.21: Dataset 2 - Stage 50% - Best Classifiers

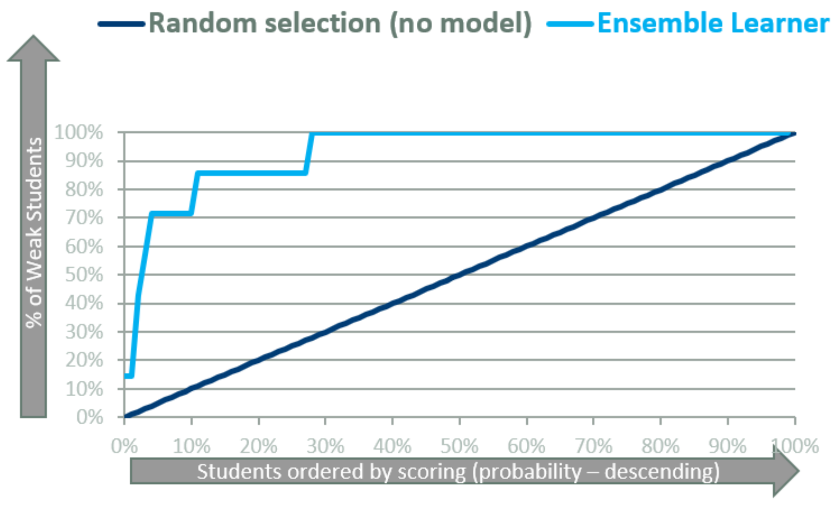


Figure 4.31: Dataset 2 - Stage 50% - Performance Ensemble Learner

The Gini Index of the ensemble selected is 89.9% and is represented by the area between the model curve and the straight line, see Figure 4.31. In particular, 100% of Weak students are identified by the ensemble learner in the first 28.28% of students ordered by their probability of being a Weak student.

The Gini Index of the ensemble selected is 89.9% and is represented by the area between the model curve and the straight line, see Figure 4.31. In particular, 100% of Weak students are

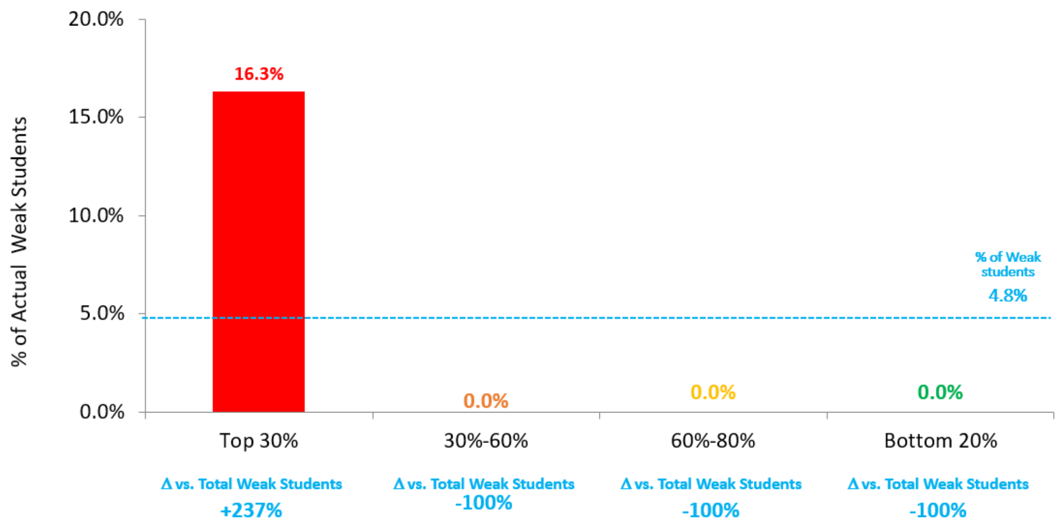


Figure 4.32: Dataset 2 - Stage 50% - Performance Ensemble Learner

identified by the ensemble learner in the first 28.28% of students ordered by their probability of being a Weak student.

Table 4.22 summarize the specificity and sensitivity results of all the base learners for the two datasets. It is worth noting that the performance was evaluated using the specificity and sensitivity due to the fact that the datasets studied are imbalanced. This is a regular occurrence in educational datasets.

Table 4.22: Base Classifiers' Performances

Specificity				
Technique	**Dataset 1**		**Dataset 2**	
	Stage 20%	Stage 50%	Stage 20%	Stage 50%
RF	0.75	0.875	0.891	0.956
MLP	0.875	0.75	0.949	0.992
K-NN	0.75	0.875	0.949	0.992
NB	0.875	1	0.942	0.963
LR	0.75	0.75	0.949	0.992
SVM	0.75	0.75	0.949	0.956
Sensitivity				
RF	1	0.857	0.857	0.714
MLP	0.714	1	0.714	0.714
K-NN	0.857	0.857	0.571	0.428
NB	0.428	0.857	0.714	0.857
LR	0.857	0.857	0.714	0.714
SVM	0.714	0.857	0.571	0.714

4.7.5 Confusion Matrices

The ensemble learners selected are formed by:

1. Random Forest and Naïve Bayes for Dataset 1 at stage 20%

2. Random Forest, k-NN and SVM for Dataset 1 at stage 50%

3. Naïve Bayes and Logistic Regression for Dataset 2 at stage 20%

4. MLP and Logistic Regression for Dataset 2 at stage 50%

For each dataset, we fix a threshold τ and compute the confusion matrices:

	W	**G**
W	5	1
G	2	7

Table 4.23: Confusion Matrix
$\tau = 0.35$
Ensemble: RF and BN
Dataset 1 - stage 20%

	W	**G**
W	6	1
G	1	7

Table 4.24: Confusion Matrix
$\tau = 0.35$
Ensemble: RF, k-NN and SVM
Dataset 1 - stage 50%

	W	**G**
W	6	11
G	1	127

Table 4.25: Confusion Matrix
$\tau = 0.065$
Ensemble: LR and BN
Dataset 2 - stage 20%

	W	**G**
W	5	3
G	2	135

Table 4.26: Confusion Matrix
$\tau = 0.2$
Ensemble: LR and MLP
Dataset 2 - stage 50%

Table 4.27 illustrates the performances of the ensemble learners in terms of accuracy, precision, recall, F-measure and false positive rate. These quantities depend on the *threshold* τ.

	τ	**Accuracy**	**Precision**	**Recall**	**F-Measure**	**False Positive Rate**
Dataset 1 - stage 20%	0.35	0.800	0.833	0.714	0.769	0.125
Dataset 2 - stage 50%	0.35	0.867	0.857	0.857	0.857	0.125
Dataset 2 - stage 20%	0.065	0.966	0.625	0.714	0.667	0.022
Dataset 2 - stage 50%	0.2	0.917	0.353	0.857	0.500	0.080

Table 4.27: Ensemble Learners' Performances

4.8 Conclusion and Research Limitations

In this chapter we discussed the focus of our experiments and the techniques used. We described the theory behind the algorithms used, the datasets and pointed out their main characteristics. For each classifier we showed the results, we compared their performances and explained how we selected the best ensemble learner.

There are a few factors that affected the results.

A For Dataset 1, the main issue was the size: only 52 students could be considered for our experiment, and the models were trained on only 70% of them and tested on the remaining 30%, corresponding to a number of students which is not statistically relevant.

B For Dataset 2, in the 20% case, the number of student was not an issue, but we could only use two features to buid the classifiers.

In both cases it was not possible to obtain additional data: indeed, for the second dataset it took almost a year to get the data because of the privacy.

C Another main factor is that there are many outliers, i.e. points that have very different characteristics from all the other points of the dataset, see Figures 4.9 and 4.8. These points correspond to those students who had a good performance at all tasks except for one, where they did not perform it, getting zero grade, e.g. getting a zero grade in the midterm. The classifiers are more likely to give a wrong prediction for these students.

D Another issue encountered was that Dataset 2 is unbalanced, i.e. the percentage of weak students in the target variable is very low.

E As we have seen in Section 4.5.2, the datasets are non-linear and consequently any linear classifier would not perform well on such given datasets.

Despite all the issues encountered, we highlight that *the classifier was able to predict correctly the weak students*, as it was shown in Section 4.7.

Chapter 5

Multi-split Optimized Bagging Ensemble Model Selection for Multi-class Educational Datasets

5.1 Introduction

Data mining is rapidly becoming a part of software engineering projects, and standard methods are constantly revisited to integrate the software engineering point of view. Data mining can be defined as an extraction of data from a dataset and discovering useful information from it [170]. This is followed by the analysis of the collected data in order to enhance the decision-making process [171]. Data mining uses different algorithms and tries to uncover certain patterns from data [172].

Educational Data Mining (EDM), a sub-field of data mining, has emerged that specializes in educational data. This is done to better understand and improve students' performance [174] and enhance learning and teaching [171]. Data used for EDM includes administrative data, students' performance data, and student activity data [175].

The advancement of technology and the Internet has also significantly impacted learning and education. As an example, e-learning was developed and can be defined as "the use of computer network technology, primarily over an intranet or through the Internet, to deliver information and instruction to individuals" [177]. There are various challenges facing e-learning platforms and environment. This includes the assorted styles of learning, and challenges arising from cultural differences [179]. Other challenges also exist such as pedagogical e-learning, technological and technical training, and e-learning time management [180]. To this end, the need for more personalized learning has emerged. Personalized learning can be considered as one of the biggest challenges of this century [181], where the personalization of e-learning includes adaptation of courses to different individuals. To make learning more personalized, adaptive techniques can also be implemented [182], [183]. Data can be automatically collected from the e-learning environment [183] with the learner's profile being analyzed in order to customize the course according to the participant's location, language, currency, seasons etc. [183], [184], [185].

A version of this chapter has been published in [210].

The previous chapter focused on a binary classification problem meant to identify Weak Students and Good students for two datasets at two different stages of the course, namely at 20% and 50% of the coursework. However, some educators prefer to identify not only two classes of students (i.e. Good vs. Weak), but instead they divide the students into several groups and consider the associated multi-class classification problem, [211]. This is usually done because the binary model often identifies a large number weak students, many of which are not truly at risk of failing the course. Accordingly, this chapter considers the same datasets previously investigated in Chapter 4 by dividing the students into three groups, namely Weak, Fair, and Good students. Furthermore, the modified datasets are then analyzed as a set of multi-class classification problems.

Multi-class classification problems can be solved by naturally extending the binary classification techniques for some algorithms, [212]. Similar to Chapter 4, in this chapter we consider various classification algorithms, compare their performances, and use Machine Learning techniques aiming to predict the students' performance in the most accurate way. Indeed, we consider K-nearest neighbor (k-NN), random forest (RF), Support Vector Machine (SVM), Multinomial Logistic Regression (LR), Naive Bayes (NB) and Neural Networks (NN) and use an optimized systematic ensemble model selection approach.

Although the approach used is similar to the one followed in Chapter 4, due to the increase of complexity of the problem, additional tools are used to predict the three groups of students. In Chapter 4 we trained one model of each type, then we selected the best ensemble among all the possible ensembles that could be generated with them. In this case, we produced a bagging of each type of model and the bagging was used for the ensembles as opposed to single models as for the binary classification problem. Bagging is itself an ensemble algorithm as it consists of ensembling several models of the same type and defining a linear combination of the individual predictions as the final prediction on an external test sample, as explained in Section 5.7. Bagging is one of the best procedures to improve the performance of classifiers as it helps reducing the variance in many hard decision problems, [213]. The empirical fact that bagging improves the classifiers' performance is widely documented, [214], and in fact ensemble methods placed first in many prestigious machine learning competitions, such as the Netflix Competition, KDD 2009, and Kaggle, [215], [216], [217]. Furthermore, a multi-split framework is considered for the studied datasets in order to reduce the bias of the ML models investigated as part of the bagging ensemble models.

The main disadvantage of bagging, and other ensemble algorithms, is the lack of interpretation. For instance, a linear combination of decision trees is much harder to interpret than a single tree. In the same way, bagging several variable selections gives little clues about which of the predictor variables are actually important. In this chapter, in order to have a rough idea of which variables are the best predictors for each algorithm, we decided to average, for each variable, its importance in every model and this average is assigned to the variable and defined to be its *averaged importance*. This was done in order to better highlight the features that are truly important across the multiple splits under consideration.

The remainder of this chapter is organized as follows: Section 5.2 presents some of the previous related work and their limitations; Section 5.3 summarizes the research contributions of this chapter; Section 5.4 defines and describes the new target variables for Dataset 1 and Dataset 2; Section 5.5 describes the performance measurement approach adopted; Section 5.6 briefly discusses the trade-off between Bias and Variance; Section 5.7 presents the methodology used to choose the best classifiers for the multi-class classification problem; Section 5.8 discusses the architecture used for training NN and shows the features' importance for each classifier for each dataset; Section 5.9 presents and discusses the experimental results both in terms of Gini Indices and by using confusion matrices; and finally, Section 5.10 lists the research limitations, proposes multiple future research opportunities, and concludes the chapter.

5.2 Related Work and Limitations

5.2.1 Related Work

Educational data mining has become a rich field of research with the demand for empirical studies and research by academia increasing in recent years. This is due to the competitive advantages that can be gained from such kind of research. Data mining can be used to evaluate and analyze the different factors that improve the knowledge gaining, skills improvement of the learners, and makes the educational institution offer a better learning experience with highly qualified students or trainees [218].

Several researchers have explored the use of data mining techniques in an educational setting. Authors of [219] used data mining techniques to analyze the learner's web usage and content-based profiles to have an on-line automatic recommendation system. In contrast, Chang et al. proposed a k-NN classification model to classify the learner's style [220]. The results of this model was used to help the educational institution management and faculties to improve the courses' contents to satisfy the learner's needs [220].

Another related study that used simple leaner regression to check the effect of the student mother's education level and the family's income in learner's academic level was presented in [221].

On the other hand, Baradwaj and Pal used classification methods to evaluate the students' performance using decision trees [222]. The study was conducted using collected data from previous year's database to predict the student result at the end of the current semester. Their study aimed to provide a prediction that will help the next term instructors identify students that they may need help.
Other researchers [223] applied Naïve Bayes classification algorithm to predict students' grades based on their previous performance and other important factors. The authors discovered that, other than students' efforts, factors such as residency, the qualification standards of the mother, hobbies and activities, the total income of the family, and the state of the family had a significant effect on the students' performance.

Later, the same authors used Iterative Dichotomiser 3 (ID3) decision tree algorithm and if-then rules to accurately predict the performance of the students at the end of the semester [224] based on different variables like Previous Semester Marks, Class Test Grades, Seminar Performance, Assignments, Attendance, Lab Work, General Proficiency, and End Semester Marks.

Similarly, Moubayed et al. [81][82] investigated the problem of identifying the student engagement level using K-means algorithm. Moreover, the authors derived a set of rulers that related student engagement with academic performance using Apriori association rules algorithm. Experimental results analysis showed that the students' engagement level and their academic performance have a positive correlation in an e-learning environment.

Prasad et al. [225] used J48 (C4.5) algorithm and concluded that this algorithm is the best choice for making the best decision about the students' performance. The algorithm was also preferred because of its accuracy and speed.

Ahmed and Elaraby conducted a similar research in 2014 [226] using classification rules. They analyzed data from a course program across 6 years and were able to predict students' final grades. In similar fashion, Khan et al. [227] used J48 (C4.5) algorithm for predicting the final grade of Secondary School Students based on their previous marks.

Kostiantis et al. [228] proposed an incremental majority voting-based ensemble classifier based on 3 base classifiers, namely NB, k-NN, and Winnow algorithms. The authors' experimental results showed that the proposed ensemble model outperformed the single base models in a binary classification environment.

Saxena [229] used k-means clustering and J48 (C4.5) algorithms and compared their performance in predicting students' grades. The author concluded that J48 (C4.5) algorithm is more efficient, since it gave higher accuracy values than k-means algorithm. Authors in [230] used and compared K-Means and Hierarchical clustering algorithms. They concluded that K-means algorithm is more preferred to hierarchical clustering due to better performance and faster model building time.

Wang et al. proposed an e-Learning recommendation framework using deep learning neural networks model [231]. Their experiments showed that the proposed framework offered a better personalized e-learning experience. Similarly, Fok et al. proposed a deep learning model using TensorFlow to predict the performance of students using both academic and non-academic subjects [232]. Experimental results showed that the proposed model had a high accuracy in terms of student performance prediction.

5.2.2 Limitations of Related Work

The limitations of the related work can be summarized as follows:

- Do not analyze the features before applying any machine learning model. Any classification model is directly applied without studying the nature of the data being considered.

- Only consider the binary classification case. Such cases often lead to identifying too many students which are not truly in danger of failing the course and hence would not need as much help and attention.

- Often use a single classification model or an ensemble model built upon randomly chosen group of base classifiers. Moreover, to the best of our knowledge, only majority voting-based ensemble models are considered.

- Often predict the performance of students from one course to the other or from one year to the other. Performance prediction is rarely considered during the course delivery.

- Often use the default parameters of the utilized algorithms/techniques without optimization.

5.3 Research Contribution

As explained in Section 4.2, our research aims to predict the students' grades during the course as opposed to other that prefer doing it at the end of the course. Similar to the binary classification problem, the multi-class classification problem assumes that the efforts and seriousness of each student are directly proportional to the final course performance and grade.

Having proved in the Chapter 4.8 that the classifiers for the binary classification problem could identify Good and Weak students with high accuracy, in this chapter we aim to:

- Analyze the collected datasets and their corresponding features using multiple graphical and quantitative techniques (e.g. dataset distribution visualization, target variable distribution, and feature importance).
- Optimize hyper-parameters of the different ML algorithms under consideration using *grid search* algorithm.
- Propose a systemic approach to build a multi-split-based (to reduce bias) bagging ensemble (to reduce variance) learner to choose the best model based on multiple performance metrics, namely the Gini index (for better statistical significance and robustness) and the target class score.
- Study the performance of the proposed ensemble learning classification model on a multi-class dataset in comparison with a binary classification model.
- Evaluate the performance of traditional classification techniques compared to the proposed ensemble learner.

5.4 Multi-class Target Variable Description

In Chapter 4, we saw that Dataset 1 and Dataset 2 were characterized by being small sized and unbalanced respectively. These two issues have even more impact on the multi-class classification problems.

Figure 5.1 shows the comparison of the two target variables. In the first case, the three classes are almost evenly distributed but each class consists of only a few students, as opposed to the second dataset that is not small sized but that is strongly unbalanced, having only 8 Weak students out of 486 students.

The target variables were constructed, as well as the target variables for the binary classification problem in Chapter 4, by considering the final grade. For the multi-class classification problem, the target variable is defined as:

1. Good (G) – the student will finish the course with a good grade (70 – 100%);
2. Fair (F) – the student will finish the course with a fair grade (51 – 69%);
3. Weak (W) – the student will finish the course with a weak grade ($\leq$ 50%).

We recall that the two classes for the binary classification problem from the previous chapter were defined as:

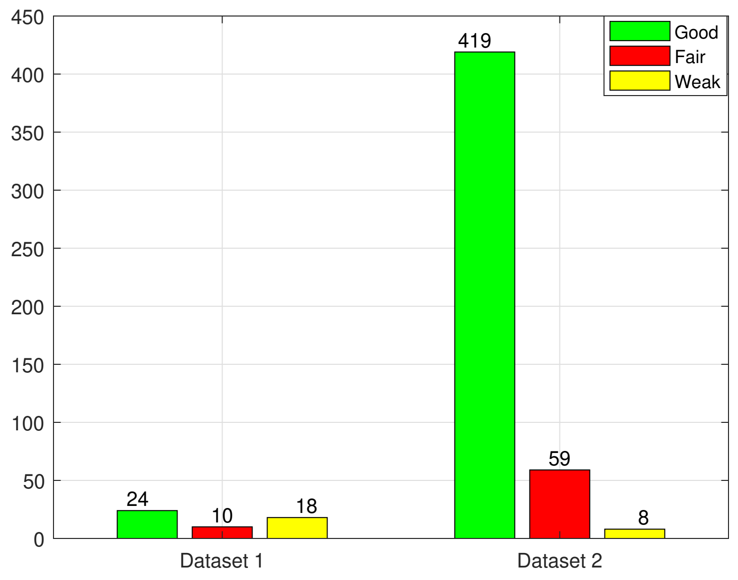


Figure 5.1: Dataset 1 and Dataset 2- Target Variables

1. Good (G) – the student will finish the course with a good grade (60 – 100%);
2. Weak (W) – the student will finish the course with a weak grade ($\leq$ 59%).

So, the Fair students are a mixture of Good and Weak students from the binary problem as shown in Figures 5.2 and 5.3. These figures illustrate how the classes are represented in the binary case (left side) and the multi-class case (right side) along with their correspondence in both datasets.

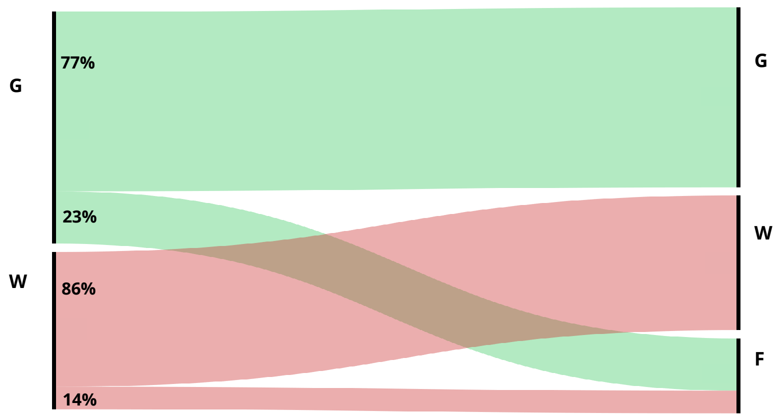


Figure 5.2: Dataset 1 - Binary Target Variable vs. multi-class Target Variable

Finally, we applied PCA to the datasets (both considered at Stage 50%), in order to visualize the three classes of students, see Figures 5.4 and 5.5.

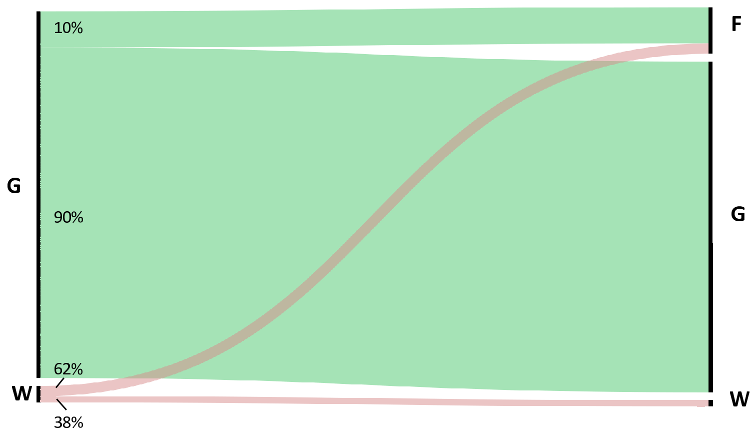


Figure 5.3: Dataset 2 - Binary Target Variable vs. multi-class Target Variable

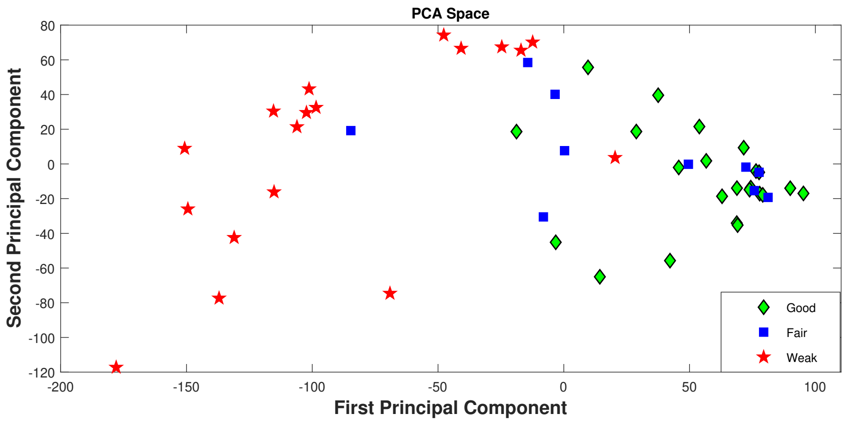


Figure 5.4: Dataset 1 - multi-class target visualization

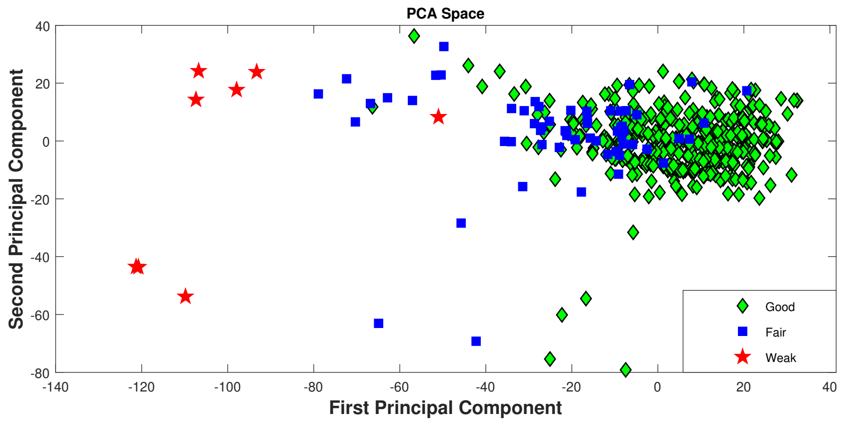


Figure 5.5: Dataset 2 - multi-class target visualization

Looking at the two figures, we note that it can be possible to draw a boundary that separates

Weak Students from the rest of the students, whereas Fair and Good students are too close and not separable by a boundary. We will see in the next sections that the performance of the models is affected by this distribution and that most of the algorithms fail in distinguishing between Fair and Good students, especially for Dataset 1.

5.5 Performance Evaluation Metrics Description

In general there are two standard approaches to choosing multiple class performance measures, [212], [233]. One approach, namely *OVA (One-versus-all)* is to reduce the problem of classifying among N classes into N binary problems. In this case, every class is discriminated from the other classes. In the other approach, called *AVA (All-versus-all)*, each class is compared to each other class. In other words, it is necessary to build a classifier for every pair of classes, i.e. building $\frac{N(N-1)}{2}$ classifiers, while discarding the rest of the classes.

Due to the size of our datasets, we chose to follow the first method as opposed to the second one. In fact, if we were to use the second approach for Dataset 1 we would need to train three binary models, one for each pair of classes (G,F), (F,W), (G,W). In particular the database for the model (F,W) would consist of only 28 students, which would be split into Training Sample (70%) and Test Sample (30%), i.e. we would train a model using 20 students and test it on 8 students. We conclude that this approach is not suitable, given the size of the dataset.

We have seen in Section 2.3 that the Gini Index metric, as well as the other metrics (Accuracy, ROC curve etc.) can be generalized to the multi-class classification problems.
As explained in the previous chapter, of all the metrics *we choose the Gini Index metric instead of the Accuracy because the latter depends on the choice of a threshold whereas the Gini Index metric does not*. This makes it statistically more significant and robust than the accuracy, particularly given that it provides a measure of the statistical dispersion of the classes [234].
In particular, we implemented a generalization of Gini index metric: during the training phase, that computes the Gini Index of each one of the three binary classifications and *optimizes (i.e. maximizes) the average of the 3 performances*, i.e. the performances corresponding to classes G, F, W.

5.6 Bias-Variance Tradeoff and Bagging

The bias-variance decomposition is a useful theoretical tool to understand the performance of a classifier, [235]. This method, used both for regression and classification problems, consists of decomposing the error generated by a learning algorithm into a *bias* component, which gives information about the accuracy of the model on average across different possible training sets, and a *variance* component, which tells us how sensitive the learning algorithm is to small changes. When either bias or variance or even both are too high, the error generated by the classifier is high and consequently the performance is very poor. Bias and variance are strictly related to the complexity of the algorithm (e.g. number of neurons for Neural Networks classifier, number of parameters, etc.) and the concepts of *over-fitting* and *under-fitting*. To better understand bias, variance, over-fitting and under-fitting we analyze Figure 5.6 that shows four cases:

1. High bias and low variance: the model underfits the data, i.e. it is not able to fit the data as it is too simple too explain the variance

2. Low bias and high variance: the model overfits the data, i.e. it fits too well the data and is not able to generalize some underlying reality, so it fails on new data

3. High bias and high variance: the model is too simple to explain the variance, and also not able to be generalized to different split

4. Low bias and low variance: appropriate-fitting

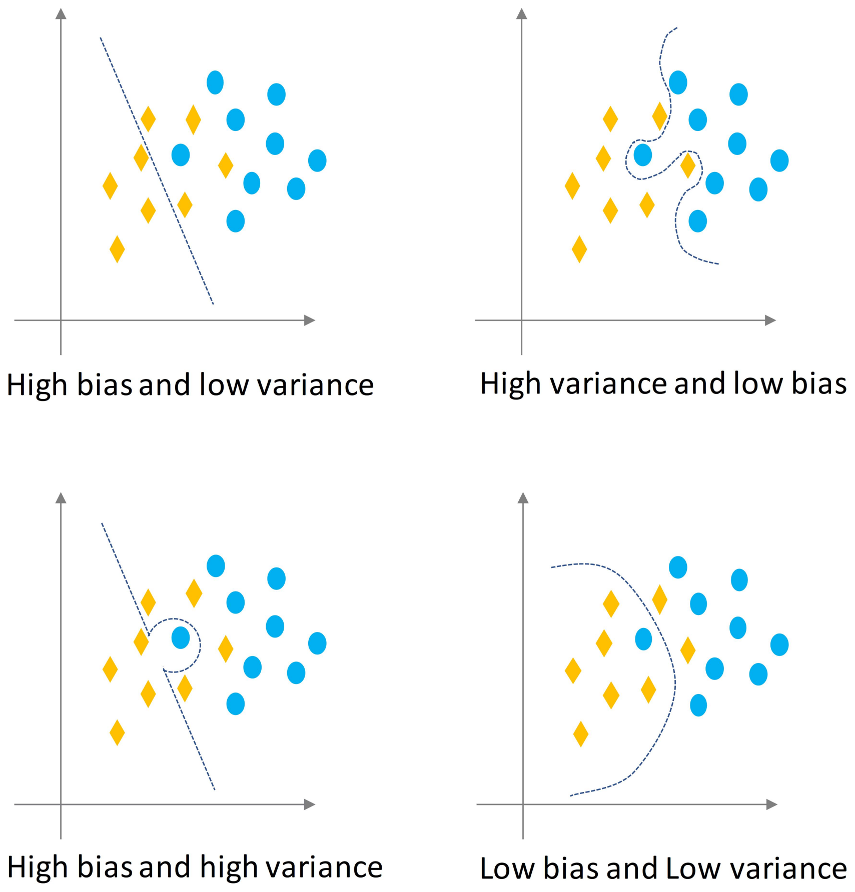

High bias and low variance
High variance and low bias
High bias and high variance
Low bias and Low variance

Figure 5.6: Bias-Variance

It is a hard task to reduce both bias and variance as usually reducing bias implies an increase of the variance and vice versa. By bagging models it is possible to average many low bias, high variance predictors, reducing consequently the variance while retaining the low bias, as shown in Figure 5.7 As shown in Figure 5.7, the learners' errors are not correlated, and consequently their average tends to zero as the number of learners increases.

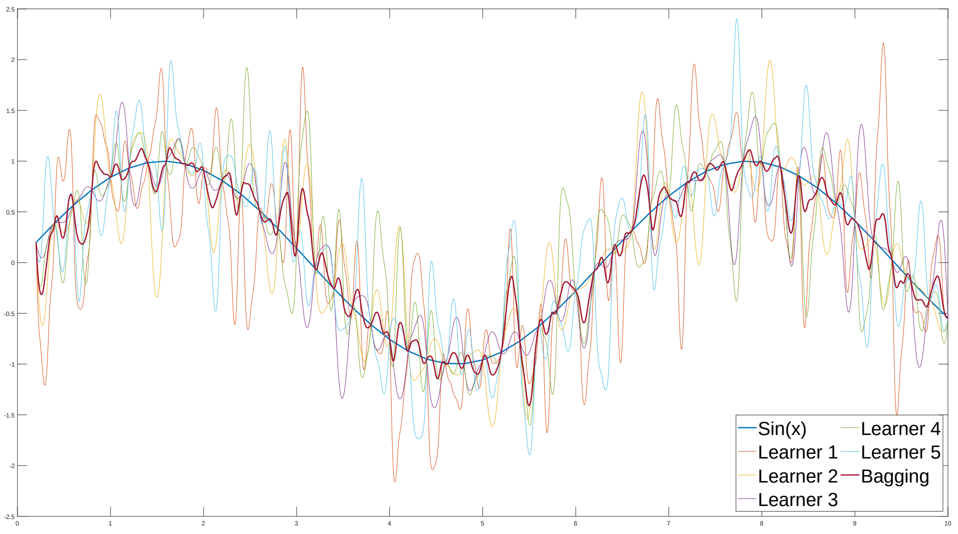


Figure 5.7: Illustrative Example of Variance Reduction Using Bagging

5.7 Methodology

For the multi-class classification problem we used several algorithms, some of which are the same as the ones used for the binary classification problems whilst the others were here introduced. More specifically we explored Random Forest, SVM - RBF, k-NN, Naïve Bayes, Multinomial Logistic Regression, and Neural Networks with 1, 2 and 3 layers, for a total of 8 classifiers per dataset.

In order to achieve better performances, we did not build only one individual model for each algorithm, instead we constructed baggings of classifiers. In fact, as explained in the previous section, bagging reduces the variance.

We built a bagging of models for each algorithm in the following way: we started by splitting each dataset into Training and Test samples in proportions 70%-30% then we used the training sample to build baggings of models. More precisely the Training sample was split 200 times into sub-Training and sub-Test samples randomly but forcing the percentages of Fair, Good and Weak students to be the same as the ones in the entire datasets.

Two hundred models were trained on the sub-Training samples and inferred on the corresponding sub-Test samples. If the Average Gini Index was above a certain fixed threshold (lowest acceptable Gini Index) then the model was kept otherwise it was discarded. For each algorithm we obtained in this way a set of models having the best performances, and we averaged their scores on the (external) Test sample, class by class. This procedure is explained in Figure 5.8.

Once for each dataset we had the eight baggings of models (one for each algorithm), we considered all the possible ensembles that could be constructed with them and compared their performances in terms of Gini Index, as explained in Section 5.5. Moreover, for each dataset, we computed the p-values corresponding to each one of the 256 possible ensembles and aimed to choose as the final ensemble the one that had best Gini Index and, at the same time, that was

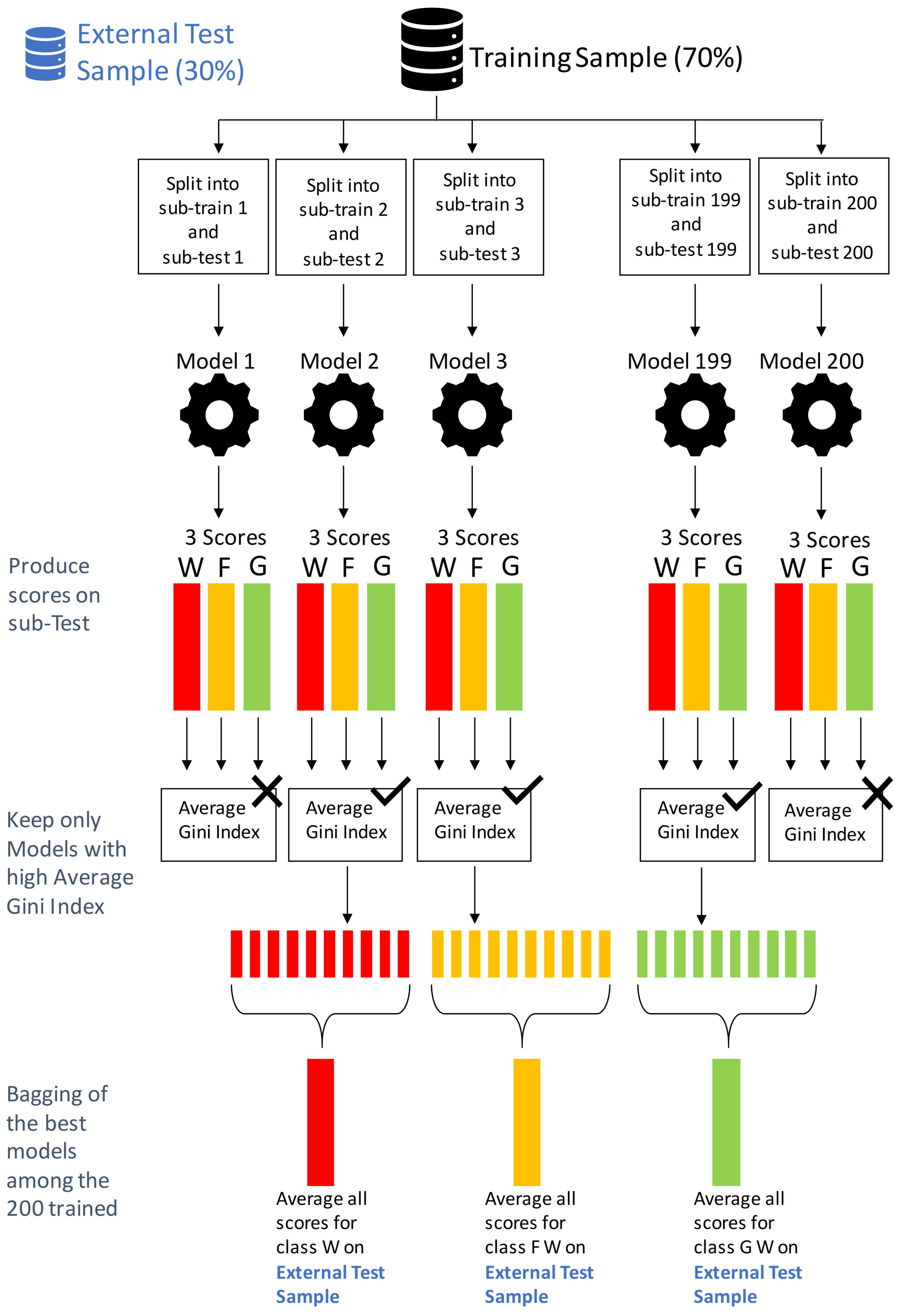

External Test Sample (30%)
Training Sample (70%)
Split into sub-train 1 and sub-test 1
Split into sub-train 2 and sub-test 2
Split into sub-train 3 and sub-test 3
Split into sub-train 199 and sub-test 199
Split into sub-train 200 and sub-test 200
Model 1
Model 2
Model 3
Model 199
Model 200
3 Scores
W F G
Produce scores on sub-Test
Average Gini Index
Keep only Models with high Average Gini Index
Bagging of the best models among the 200 trained
Average all scores for class W on External Test Sample
Average all scores for class F W on External Test Sample
Average all scores for class G W on External Test Sample

Figure 5.8: Bagging Ensemble Model Building Methodology

statistically significant.

As for Chapter 4, all the performances were strongly dependent on the split that had been chosen. For this reason we repeated the whole process (baggings, ensembles, p-values) 5 times for 5 extra splits choosing the same splits that were used in the previous Chapter for the binary classification problem in order to be able to compare the results obtained from the binary classification problem with the ones obtained for the multi-class-classification problem.

The classifiers were inferred on the test sample, giving as output three vectors of predictions to be analyzed. These three vectors express the chance that each student is classified as Weak, Fair and Good. In order to build the confusion matrices, we fixed a threshold for each class, namely τ_F, τ_G, and τ_W and, for each student belonging to the Test sample, we defined the predicted class according to the following steps:

1. The 3 scores corresponding to the 3 classes were normalized in order to make them comparable.

2. For each class, if the probability is higher than the corresponding threshold then the target variable for the binary classification problem associated to that class is predicted to be 1, otherwise it's 0.

3. In this way we obtained a 3-column matrix taking values 1's and 0's. Comparing the 3 predictions, if a student has only one possible outcome (i.e. only one 1, and two 0's) then the student is predicted to belong to the corresponding class. Otherwise, if there is uncertainty about the prediction because there is more than one 1 predicted for the student, then the class with the highest score is chosen to be the predicted one.

For instance, consider the following example.

Example Suppose we have trained a classifier using 70% of Dataset 1. When we infer the model on the test sample (remaining 30%, consisting of 15 students), we obtain 3 vectors of scores, one for each class and we can compute their Gini Indices, see Figure 5.9.

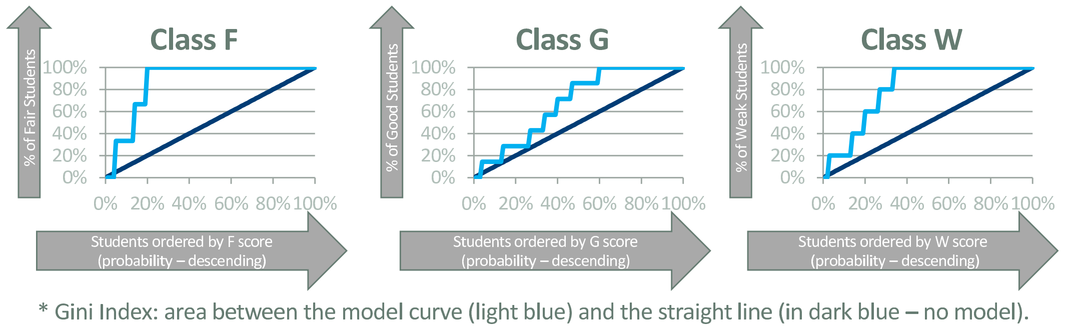


Figure 5.9: Example - Averaged Gini Index Computation

In this example the Gini Indices of Classes F, G, W are 97.2%, 76.8%, 98% respectively, hence the Averaged Gini Index is 90.7%.
We map the three scores linearly to the interval $[0, 1]$, i.e. we normalize them to make them comparable. The normalized scores are represented in Table 5.1 in columns *score F*, *score G*, *score W*.

Column *Actual Class* corresponds to the actual target variable that we aim to predict. Treating each score as if it was the score associated to a binary classification problem, we need to set a threshold for each class such that if the score is greater than the threshold then the student belongs to such class otherwise he/she doesn't (i.e., he/she belongs to one of the other two classes). Therefore we set three thresholds τ_F, τ_G, and τ_W for Class, F, G, and W respectively. For instance, let $\tau_F = 0.267$, $\tau_G = 0.323$, and $\tau_W = 0.740$. For student 1 in Table 5.1, the chance to be classified as F is $0.365 \geq \tau_F$, whereas the probabilities to belong to classes G and W are less than τ_G and τ_W respectively. In conclusion, once the three thresholds are set, we can claim that student 1 is a Fair student.

Student 6 has score $F = 0.389 \geq \tau_F$ and score $G = 0.620 \geq \tau_G$ so he/she belongs either to Class F or to class G. Since the scores are normalized and are comparable, we set the predicted class to be the one corresponding to the highest score, hence we predict student ID=6 to belong to class G.
For student 2 (7 and 14) note that the three scores are all below the thresholds so the predicted class is the one corresponding to the greatest score, i.e. the student is predicted as Weak.

ID	Actual Class	score F	score G	score W	Max Pred.	F	G	W	Predicted Class
1	F	0.365	0.1	0.707	W	1	0	0	F
2	F	0.015	0.25	0.647	W	0	0	0	W
3	G	0.828	0.232	0.337	F	1	0	0	F
4	G	0.085	0.13	0.758	W	0	0	1	W
5	W	0.663	0.038	0.853	W	1	0	1	W
6	G	0.389	0.62	0.142	G	1	1	0	G
7	G	0.234	0.078	0.723	W	0	0	0	W
8	W	0	0.054	0.793	W	0	0	1	W
9	W	0.009	0	0.944	W	0	0	1	W
10	G	0.33	0.797	0	G	1	1	0	G
11	F	0.266	0.818	0.01	G	0	1	0	G
12	G	0.33	0.797	0	G	1	1	0	G
13	G	0.248	0.58	0.22	G	0	1	0	G
14	W	0.18	0.167	0.648	W	0	0	0	W
15	W	0.061	0.186	0.745	W	0	0	1	W

Table 5.1: Example - Predicting Classes

The max probability associated to each student is expressed in column *Max Pred.*, and if we compare this column with column *Actual Class* we note that taking the max score as the predicted class would not have been a good strategy.

By setting the three thresholds τ_F, τ_G, and τ_W and considering the max score only in case of uncertainty we obtained for each student a predicted class, expressed in column *Predicted Class*. If we compare the actual class with the predicted class we can build the corresponding confusion matrix:

	F	**G**	**W**
F	1	1	1
G	1	4	2
W	0	0	5

Table 5.2: Example

5.8 ML Parameter Tuning and Application

We chose one algorithm for each area of Machine Learning aiming to cover all types of classification methods including tree-based (Random Forest), vector-based (SVM-RBF), distance-based (k-NN), regression-based (Multinomial logistic regression), probabilistic (Naïve Bayes), and neural network-based (Neural Networks). Specifically, we trained 8 baggings of classifiers using Random Forest, SVM - RBF, Neural Networks (with 1, 2, and 3 layers with 5 neurons per layer), k-NN, Multinomial Logistic Regression, and Naïve Bayes).

In Section 5.8.1, we explain how we trained a NN and in the following sections we show, for each dataset, the impact that each variable has on each classifier. As explained in Section 5.1, in order to understand which variables are the best predictors for each algorithm, we decided to average, for each variable, its importance on every model and this average is assigned to the variable and defined to be its *averaged importance*. In Section 5.9 we will show that the most important variables affect the performances of some classifiers.

5.8.1 Neural Network Tutning

Finding the optimal number of neurons for NN is still an open field of research and requires a lot of computational resources. [236] summarizes some formulas (previously proved by Li, Chow and Yu, Shibata and Ikeda, Sheela and Deepa) for the computation of the optimal number of hidden neurons N_h:

- $N_h = \frac{\sqrt{1+8N_i}-1}{2}$
- $N_h = \sqrt{N_i N_o}$
- $N_h = \frac{4N_i^2+3}{N_i^2-8}$

where N_i is the number of input neurons (number of variables) and N_o is the number of output neurons (3 classes). Applying the latter formulas to our datasets at the two different stages, we obtained a number of neurons between 2 and 6. Considering that we adopted the early stopping technique in order to prevent over-fitting and reduce variance, we decided to choose this number in the high range of the interval $[2, 6]$ and set it to be equal to 5 instead of performing a full optimization (i.e., brute force searching).
The results obtained by using 1 hidden layer with 5 neurons were so promising that we decided to stress our hypothesis about early stopping and tried NN with 2 and 3 hidden layers with 5 neurons each, obtaining similar results.
The NN models we built are as in Figure 5.10, 5.11, 5.12.

The initialization of the weights of neural networks was implemented by using the Nguyen-Widrow Initialization Method [237] whose goal is to speed up the training process by choosing the initial weights instead of generating them randomly. The main idea is to assign to each hidden node its own interval at the start of training. By doing so, during the training each hidden layer has to adjust its interval size and location less than if the initial weights are chosen randomly. Consequently, the computational cost is reduced.

Levenberg-Marquardt backpropagation was used to train the models: this algorithm was

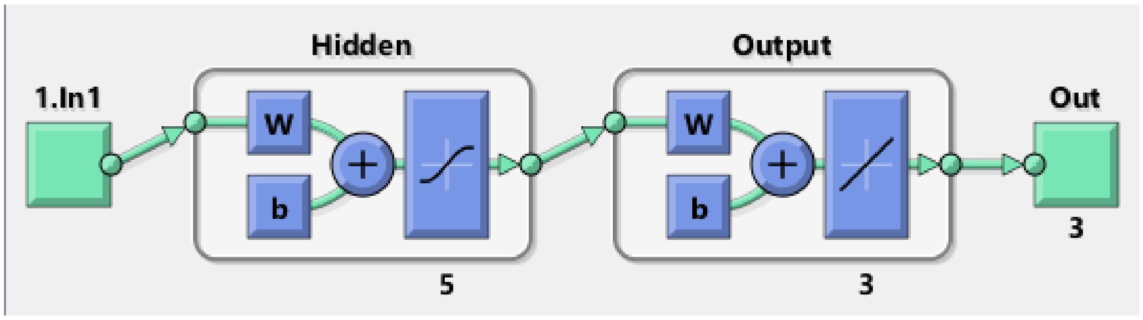


Figure 5.10: NN with 1 hidden layer

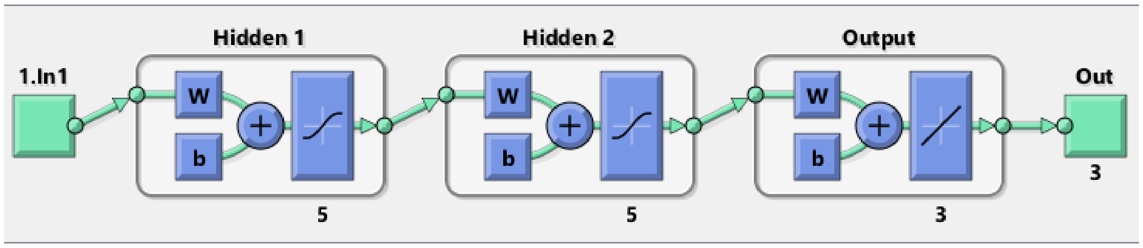


Figure 5.11: NN with 2 hidden layers

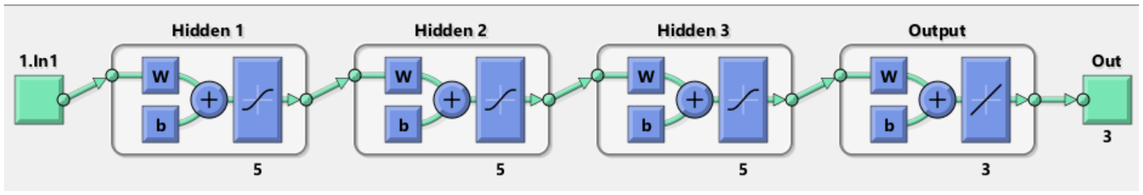


Figure 5.12: NN with 3 hidden layers

introduced for the first time by Levenberg and Marquardt in [238], and is derived from Newton's method that was designed for minimizing functions that are sums of squares of nonlinear functions [239]. This method is confirmed to be the best choice in various learning scenarios, both in terms of time spent and performance achieved, [240] . Moreover, the datasets were normalized in input by mapping linearly to $[-1, 1]$ (the activation function used in the input layer is the hyperbolic tangent) and in output to $[0, 1]$ (the activation function in the output layer is linear) in order to avoid saturation of neurons and make the training smoother and faster.

5.8.2 ML Algorithms' Parameter Tuning

Hyper-parameter tuning has become an essential step to improve the performance of ML algorithms. This is due to the fact that each ML algorithm is governed by a set of parameters that dictate its predictive performance [241]. Several methods have been proposed in the literature to optimize and tune these parameters such as grid search algorithm, random search, evolutionary algorithms, and Bayesian optimization method [241, 107].

This work adopts the *grid search* method to perform hyper-parameter tuning. Grid search optimization method is a common method used to hyper tune the parameters of ML classification algorithms. Simply put, it discretizes the values for the parameter set [241]. Models are then trained and assessed for all possible combinations of these values for all the parameters of the ML model used. Mathematically speaking, this can be formulated as follows:

$$\max_{parm} f(parm) \tag{5.1}$$

where f is an objective function to be maximized (typically the accuracy of the model) and *parm* is the set of parameters to be tuned. Despite the fact that this may seem computationally heavy, grid search method benefits from the ability to perform the optimization in parallel, which results in a lower computational complexity [241].

In contrast to traditional hyper-parameter tuning algorithms that perform the optimization with the objective of maximizing the accuracy of the ML model, this work tunes the parameters used for each model using the *grid search* optimization method to maximize the average Gini index (for more statistical significance and robustness [234]) over multiple splits [32]. More specifically, the objective function is:

$$\max_{parm} Average\ Gini\ Index = \max_{parm} \frac{1}{N} \sum_{i=1}^{N} Gini\ Index_i(parm) \tag{5.2}$$

where *parm* is the set of parameters to be tuned for each ML algorithm and N is the number of different splits considered. For example, in the case of K-NN algorithm, $parm = \{K\}$ which is the number of neighbors used to determine the class of the data point.

R was used to implement the six classifiers and the ensemble learners. As mentioned above, the six classifiers considered in this work are SVM-RBF, LR, NB, k-NN, RF, and NN. All the classifiers were trained using all the variables available. Moreover, the parameters of the algorithms were tuned by maximizing the Gini Index of each split. Furthermore, 200 different splits of data were used to reduce the bias of the models under consideration.

Table 5.3 summarizes the range of values for the parameters of the different ML algorithms considered in this work. Note the following:

Table 5.3: Grid Search Parameter Tuning Range

Algorithm	**Parameter Range in Dataset 1**	**Parameter Range in Dataset 2**
SVM-RBF	C=[0.25, 0.5, 1] & sigma = [0.05-0.25]	C=[0.25, 0.5, 1] & sigma = [0.5-3.5]
NB	usekernel=[True,False]	usekernel=[True,False]
K-NN	k=[5,7,9,...,43]	k=[5,7,9,...,43]
RF	mtry=[2,3,...,12]	mtry=[2,3,4]

- For the NB algorithm, the *usekernel* parameter represents the choice of the density estimator used. More specifically, *usekernel=false* implies that the data distribution is Gaussian while *usekernel = true* implies that the data distribution is non-Gaussian.

- The LR algorithm was not included in the table. This is due to the fact that it has no parameters to optimize. The sigmoid function, which is the default function, was used by the grid search method to maximize the Gini index.

- The NN method was not included in the table because it was explained in the previous Section 5.8.1.

For each algorithm and each dataset we show the list of the features ordered by their importance, i.e. their impact on the predictions. This is meant to give only a rough idea of what the most important features are for each algorithm and each dataset, as the ordering, for such small datasets, heavily depends on the split in Train-Test samples chosen. For this reason, the weights of the predictors will not be specified. The importance of the features is determined using the CARET package available for R language [242]. Depending on the classification model adopted, the importance is calculated in one of multiple ways. For example, when using RF method, the prediction accuracy on the out-of-bag portion of the data is recorded. This is iteratively done after permuting each predictor variable. The difference between the two accuracy values is then averaged over all trees and normalized by the standard error [242]. In contrast, when the k-NN method is used, the difference between the class centroid and the overall centroid is used to measure the variable influence. Accordingly, the separation between the classes is larger whenever the difference between the class centroids is larger [242]. On the other hand, when using the NN method, the CARET package uses the same feature importance method proposed in Gevrey *et al.* which uses combinations of the absolute values of the weights [243]. This importance is reflected in the weights calculated for each feature for each classification model with more important features contributing more towards the prediction.

The final step was to select, for each problem, the best ensemble learner among all the possible ensemble learners that could be produced with the six classifiers.

5.8.3 Features importance: Dataset 1 - Stage 20%

- RF: The variables' importance in terms of predictivity is described in Table 5.4 that shows that the most relevant features are ES2.2 and ES3.3.

Ranking	**Feature**
1	ES2.2
2	ES3.3
3	ES2.1
4	ES1.1
5	ES3.5
6	ES1.2
7	ES3.4
8	ES3.1
9	ES3.2

Table 5.4: Dataset 1 - Stage 20% - Features' weights for Random Forest

- SVM: The variables' importance for SVM is described in Table 5.5, that shows that the most relevant features are ES2.2 and ES3.3.

- NN with 1 hidden layer: For NN1, the variables' importance in terms of predictivity is described in Table 5.6 that shows that the most relevant features are ES2.2 and ES3.5.

- NN with 2 hidden layer: The most important variables for NN2 are ES2.2 and ES3.2, as shown in Table 5.7.

Ranking	**Feature**
1	ES2.2
2	ES3.3
3	ES2.1
4	ES3.5
5	ES1.2
6	ES3.4
7	ES1.1
8	ES3.1
9	ES3.2

Table 5.5: Dataset 1 - Stage 20% - Features' weights for SVM

Ranking	**Feature**
1	ES2.2
2	ES3.5
3	ES3.3
4	ES3.2
5	ES1.1
6	ES2.1
7	ES1.2
8	ES3.4
9	ES3.1

Table 5.6: Dataset 1 - Stage 20% - Features' weights for NN with 1 hidden layer

Ranking	**Feature**
1	ES2.2
2	ES3.3
3	ES3.5
4	ES2.1
5	ES3.2
6	ES1.1
7	ES3.4
8	ES1.2
9	ES3.1

Table 5.7: Dataset 1 - Stage 20% - Features' weights for NN with 2 hidden layers

- NN with 3 hidden layers: The variables' importance in terms of predictivity is described in Table 5.8 that shows that the most relevant features are ES2.2 and ES3.2.
- (k-NN) Table 5.9 shows that the most relevant features for k-NN are ES2.2 and ES3.3.
- LR: The variables' importance in terms of predictivity is described in Table 5.10 that shows that the most relevant features are ES1.1 and ES1.2.
- NB: Table 5.11 shows that the most relevant features are ES2.2 and ES3.3.

Ranking	**Feature**
1	ES2.2
2	ES3.3
3	ES3.5
4	ES2.1
5	ES3.2
6	ES1.1
7	ES3.4
8	ES1.2
9	ES3.1

Table 5.8: Dataset 1 - Stage 20% - Features' weights for NN with 3 hidden layers

Ranking	**Feature**
1	ES2.2
2	ES3.3
3	ES2.1
4	ES3.5
5	ES3.4
6	ES1.2
7	ES1.1
8	ES3.1
9	ES3.2

Table 5.9: Dataset 1 - Stage 20% - Features' weights for k-NN

Ranking	**Feature**
1	ES1.1
2	ES1.2
3	ES3.5
4	ES3.3
5	ES3.4
6	ES3.2
7	ES3.1
8	ES2.2
9	ES2.1

Table 5.10: Dataset 1 - Stage 20% - Features' weights for LR

5.8.4 Features importance: Dataset 1 - Stage 50%

It is important to point out that, for Dataset 1 at stage 50%, features ES4.1 and ES4.2 are the most important for every classifier.

- RF: For RF, the variables' importance in terms of predictivity is described in Table 5.12 that shows that the most relevant features are ES4.1 and ES4.2.

- SVM: The variables' importance in terms of predictivity is described in Table 5.13 that

Ranking	**Feature**
1	ES2.2
2	ES3.3
3	ES2.1
4	ES3.5
5	ES3.4
6	ES1.2
7	ES1.1
8	ES3.2
9	ES3.1

Table 5.11: Dataset 1 - Stage 20% - Features' weights for BN

Ranking	**Feature**
1	ES4.1
2	ES4.2
3	ES2.2
4	ES5.1
5	ES2.1
6	ES1.1
7	ES3.5
8	ES3.3
9	ES3.4
10	ES3.1
11	ES3.2
12	ES1.2

Table 5.12: Dataset 1 - Stage 50% - Features' weights for RF

shows that the most relevant features are ES4.1 and ES4.2.

Ranking	**Feature**
1	ES4.1
2	ES4.2
3	ES2.2
4	ES5.1
5	ES3.3
6	ES2.1
7	ES3.5
8	ES1.2
9	ES3.4
10	ES1.1
11	ES3.1
12	ES3.2

Table 5.13: Dataset 1 - Stage 50% - Features' weights for SVM

- NN with 1 hidden layer: The variables' importance in terms of predictivity is described in Table 5.14 that shows that the most relevant features are ES4.1 and ES4.2.

Ranking	**Feature**
1	ES4.1
2	ES4.2
3	ES3.3
4	ES3.5
5	ES2.1
6	ES5.1
7	ES1.1
8	ES3.4
9	ES3.2
10	ES2.2
11	ES1.2
12	ES3.1

Table 5.14: Dataset 1 - Stage 50% - Features' weights for NN with 1 hidden layer

- NN with 2 hidden layers: The variables' importance in terms of predictivity is described in Table 5.15 that shows that the most relevant features are ES4.1 and ES4.2.

Ranking	**Feature**
1	ES4.1
2	ES4.2
3	ES3.5
4	ES3.3
5	ES5.1
6	ES2.1
7	ES3.4
8	ES2.2
9	ES1.1
10	ES3.1
11	ES3.2
12	ES1.2

Table 5.15: Dataset 1 - Stage 50% - Features' weights for NN with 2 hidden layer

- NN with 3 hidden layers: The variables' importance in terms of predictivity is described in Table 5.16 that shows that the most relevant features are ES4.1 and ES4.2.
- k-NN: Table 5.17 shows that the most relevant features for k-NN are ES4.1 and ES4.2.
- LR: Table 5.18 shows that the most relevant features for LR are ES4.1 and ES4.2.
- NB: The variables' importance in terms of predictivity is described in Table 5.19 that shows that the most relevant features are ES4.1 and ES4.2.

Ranking	Feature
1	ES4.1
2	ES4.2
3	ES5.1
4	ES3.5
5	ES3.3
6	ES2.1
7	ES2.2
8	ES1.1
9	ES3.4
10	ES3.2
11	ES3.1
12	ES1.2

Table 5.16: Dataset 1 - Stage 50% - Features' weights for NN with 3 hidden layer

Ranking	Feature
1	ES4.1
2	ES4.2
3	ES2.2
4	ES5.1
5	ES3.3
6	ES2.1
7	ES3.5
8	ES3.4
9	ES1.2
10	ES1.1
11	ES3.1
12	ES3.2

Table 5.17: Dataset 1 - Stage 50% - Features' weights for k-NN

Ranking	Feature
1	ES4.1
2	ES4.2
3	ES1.1
4	ES2.1
5	ES1.2
6	ES3.3
7	ES3.4
8	ES5.1
9	ES3.5
10	ES2.2
11	ES3.1
12	ES3.2

Table 5.18: Dataset 1 - Stage 50% - Features' weights for LR

Ranking	Feature
1	ES4.1
2	ES4.2
3	ES1.1
4	ES2.1
5	ES1.2
6	ES3.3
7	ES3.4
8	ES5.1
9	ES3.5
10	ES2.2
11	ES3.1
12	ES3.2

Table 5.19: Dataset 1 - Stage 50% - Features' weights for Naïve Bayes

In general, the most important features for almost all the classifiers are ES4.1 and ES4.2. These features correspond to the *Evaluate* category as per Bloom's taxonomy which represents one of the highest level of comprehension of the course material from the educational point of view. Therefore, it makes sense for these features to be suitable indicators and predictors of student performance.

5.8.5 Features importance: Dataset 2 - Stage 20%

We have only two features for Dataset 2, Stage 20% and for all the classifiers, the list of features ordered by importance is the same as for the binary classification problem, see Table 5.20.

Ranking	Feature
1	Assignment01 [8]
2	Quiz01 [10]

Table 5.20: Dataset 2 - Stage 20% - Features' weights

Since Dataset 2 at stage 20% has only two variables we can represent it graphically in order to have a better understanding of the situation and to explain why all the algorithms agree that Assignment01 [8] is the most important predictor.

Figure 5.13 shows that it is easy to identify the categories of students by setting some thresholds on the variable Assignment01 [8]. For instance, most of the Weak students have grade zero to Assignment01 [8]

5.8.6 Features importance: Dataset 2 - Stage 50%

The variables' importance for NN1, NN2, KNN, and BN is described in Table 5.21 whereas the variables' importance for NN3, LR, RF, and SVM is described in Table 5.22:

Based on the aforementioned results, it can be seen that assignments are better indicators of the student performance. This is because students tend to have more time to complete assignments. Moreover, they are often allowed to discuss issues and problems among themselves.

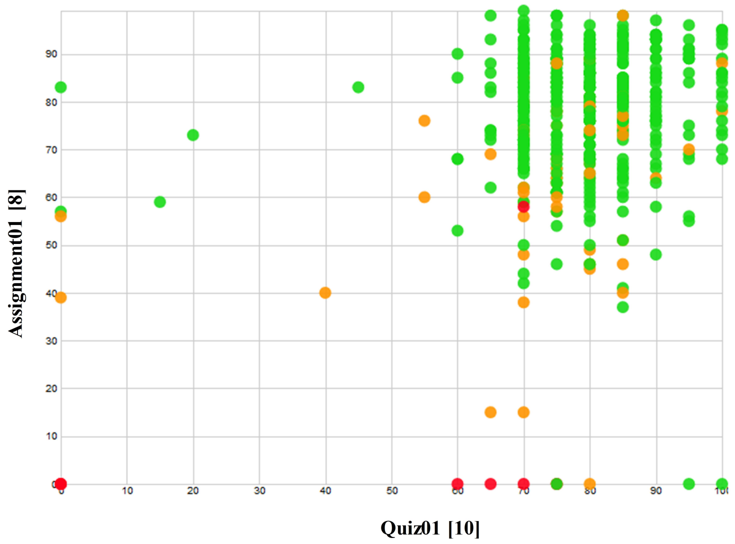


Figure 5.13: Dataset 2 - Stage 20% - scatter plot

Ranking	Feature
1	Assignment02 [8]
2	Assignment01 [12]
3	Midterm Exam [20]
4	Quiz01 [10]

Table 5.21: Dataset 2 - Stage 50% - NN1, NN2, KNN, and BN, Features' weights

Thus, students not performing well in the assignments can be an indication that they are not fully comprehending the material, resulting in potentially lower overall final course grade.

5.9 Experimental Results and Discussion

A Matlab [244] library was used to build the Neural Networks classifiers which was integrated with R. All other experiments were built in R. This was done on an Intel® Core^{TM} i7 processor @ 3.40 GHz system with 16GB RAM running Windows 10 operating system. All possible combinations of ensembles of 8 baggings of models (256 in total) were computed for the initial Train-Test split and for the 5 extra splits. In particular, we chose the same 6 splits into Training and Test samples that we have worked with in the binary classification problem in order to compare the performances split-wise.

For each dataset, the average of the performances, namely *averaged Gini Index*, on the 6 splits was used to select the most robust ensemble learner. In addition, we computed the p-values of all the ensembles for all the splits aiming to select the ensemble learner *with highest averaged Gini index that was also statistically significant on every split*.

In the following sections we will see the results obtained for the two datasets at each stage.

Ranking	Feature
1	Assignment01 [12]
2	Assignment02 [8]
3	Midterm Exam [20]
4	Quiz01 [10]

Table 5.22: Dataset 2 - Stage 50% - NN3, LR, RF, and SVM, Features' weights

5.9.1 Results: Dataset 1 - Stage 20%

If we based our choice only on the Gini index corresponding to the initial split, the ensemble learner we would have selected for Dataset 1 at Stage 20% would have been formed by Naïve Bayes, Neural Networks (1 layer) and SVM, see Appendix A. Instead, the ensemble learner that appears to be the most stable on every split and with statistical significance is the one formed by a bagging of Neural Networks with two layers. The performances obtained by selecting this ensemble learner on each split are shown in Appendix A.

Figure 5.14 shows the results obtained by inferring the ensemble on the initial test sample.

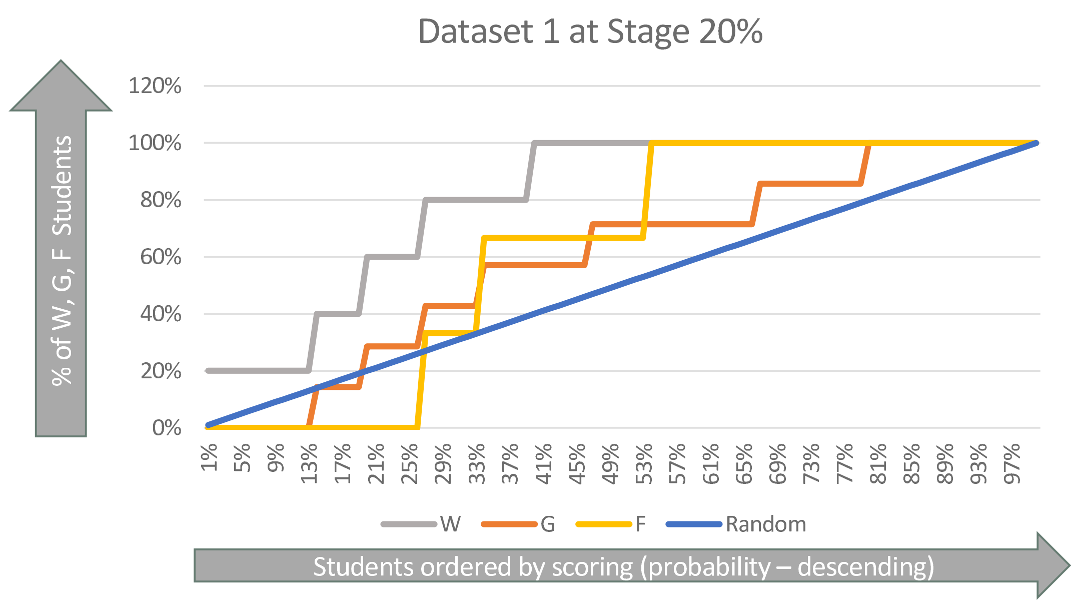


Figure 5.14: Dataset 1 - Stage 20% - Ensemble Learner

Classes G, F, W have Gini Indices equal to 46.4%, 38.9% and 94.0% respectively. Hence, the Averaged Gini Index is 59.8%. On average, on Test sample and the 5 extra splits the Averaged Gini Index is 62.1%. The corresponding p-values are all less than 0.03 (see Appendix A).

The confusion matrix for the Test sample (consisting of 15 students), obtained as explained in Section 5.7, is the following:

Table 5.24 illustrates the performances of the ensemble learner in terms of precision, recall, F-measure and false positive rate per class and on average. These quantities depend on the thresholds τ_F, τ_G and τ_W and the way we defined the predictions. The Accuracy is 66.7%.

	F	**G**	**W**
F	1	1	1
G	1	4	2
W	0	0	5

Table 5.23: Confusion Matrix
$\tau_F = 0.158, \tau_G = 0.310, \tau_W = 0.682$
Ensemble: NN 2
Dataset 1 - Stage 20%

	Precision	**Recall**	**F-measure**	**False Positive Rate**
F	0.33	0.50	0.40	0.50
G	0.57	0.80	0.67	0.20
W	1.00	0.63	0.77	0.38
Avg	0.64	0.64	0.61	0.36

Table 5.24: Dataset 1 - Stage 20% - multi-class Classification Problem - Ensemble Performances

Furthermore, if we compare this performance with the one obtained on the same splits in the binary classification problem, see Table 4.27, we note that the performance is worse in the multi-class classification problem even if we have used a more sophisticated approach (ensemble of baggings).

5.9.2 Results: Dataset 1 - Stage 50%

For Dataset 1 at Stage 50%, *none of the ensembles we constructed is statistically significant* even if their Averaged Gini Indices are on average higher than the ones obtained for Dataset 1

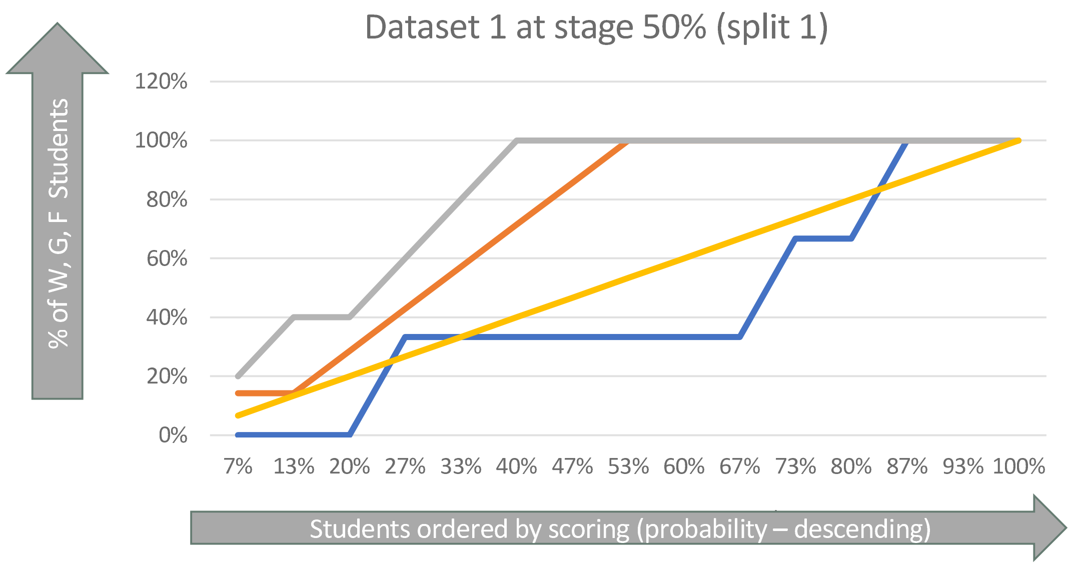


Figure 5.15: Dataset 1 - Stage 50% - Ensemble Learner

at Stage 20%. In fact, the performance for class F gets worse when we add the three variables. More precisely, when we add Features *ES4.1*, *ES4.2* and *ES5.1* to Dataset 1 at stage 20% obtaining Dataset 1 at stage 50%, they end up being the ones that have the main impact on the predictions. These features were the most important ones also in the binary classification case, as we saw in Section 4.6.2, and were crucial in distinguishing between Good and Weak Students. Also in the multi-class classification problem these variables help distinguishing between W and G and in fact the performance corresponding to these two classes improve, but since Fair students are defined as a mixture of Weak and Good students from the binary classification problem, see Section 5.4, the classifier becomes less confident in predicting the Fair students.

The best ensemble in terms of performance is the one obtained from a bagging of Naïve Bayes and bagging of k-NN. The Averaged Gini Index on 6 splits is 74.9% and on the initial test sample the Averaged Gini Index is 86.5%. Figure 5.15 shows the performance obtained on Split 1, having Averaged Gini Index equals 50%, with Gini Indices -22.2%, 76.8%, 86.0% respectively on Classes F, G and W. On a different split, the ensemble formed by a bagging of Naïve Bayes and bagging of k-NN on Dataset 1 at stage 20% gives Gini Indices 77.8%, 53.6% and 48.0% respectively on Classes F, G and W, proving that the performance depends heavely on the split. In general, when we add the new three features (obtaining Dataset1 at stage 50%) the performance improves on classes G and W whereas it gets much worse for class F.
The confusion matrix obtained is the following:

	F	**G**	**W**
F	0	2	1
G	0	7	0
W	1	1	3

Table 5.25: Confusion Matrix
$\tau_F = 0.10, \tau_G = 0.29, \tau_W = 0.88$
Ensemble: BN and k-NN
Dataset 1 - Stage 50%

Table 5.26 illustrates the performances of the ensemble learner in terms of precision, recall, F-measure and false positive rate per class and on average. These quantities depend on the thresholds τ_F, τ_G and τ_W and the way we defined the predictions. The Accuracy is 66.7%. Note that we cannot compute the F-measure for class F as Precision and Recall are zero.

	Precision	**Recall**	**F-measure**	**False Positive Rate**
F	0.00	0.00	-	1.00
G	1.00	0.70	0.82	0.30
W	0.60	0.75	0.67	0.25
Avg	0.53	0.48	-	0.52

Table 5.26: Dataset 1 - Stage 50% - multi-class Classification Problem - Ensemble Performances

It is worth noting the low average Gini can be attributed to 2 main reasons: the first is that this dataset is a small dataset. The second reason is, as per figure 5.3, the Fair class is a combination of students from the Good and the Weak (when using the binary model). Hence, this is causing some confusion to the models being trained. This is further highlighted by the large false positive rate obtained for the Fair class.

5.9.3 Results: Dataset 2 - Stage 20%

The ensemble learner selected for Dataset 2 at Stage 20% is formed by bagging of Naïve Bayes, bagging of k-NN, bagging of LR, bagging of Neural Networks with two layers, and bagging of SVM. For instance, we show the results corresponding to the initial test sample. For each class, we normalized the scores obtained by the five baggings of models on the test sample in order to make these probabilities comparable, then we averaged them. The performances obtained are shown in Figure 5.16

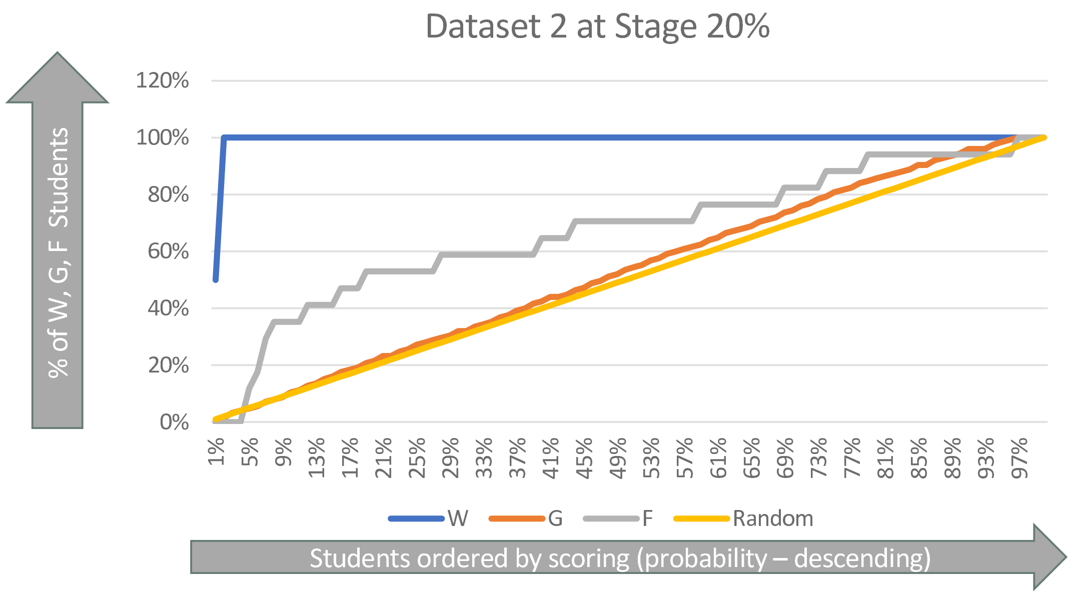


Figure 5.16: Dataset 2 - Stage 20% - Ensemble Learner

Classes G, F, W have Gini Indices equal to 48.1%, 38.6% and 99.7% respectively.
The confusion matrix associated is the following: Table 5.28 illustrates the performances of the

	F	**G**	**W**
F	5	11	1
G	5	120	0
W	0	0	2

Table 5.27: Confusion Matrix
$\tau_F = 0.12, \tau_G = 0.62, \tau_W = 0.40$
Ensemble: BN, k-NN, LR, NN 2 and SVM
Dataset 2 - Stage 20%

ensemble learner in terms of precision, recall, F-measure and false positive rate per class and on average. These quantities depend on the thresholds τ_F, τ_G and τ_W and the way we defined the predictions. The Accuracy is 88.2%, which is very good compared with the performances obtained for Dataset 1.

	Precision	Recall	F-measure	False Positive Rate
F	0.29	0.50	0.37	0.50
G	0.96	0.92	0.94	0.08
W	1.00	0.67	0.80	0.33
Avg	0.75	0.69	0.70	0.31

Table 5.28: Dataset 2 - Stage 20% - multi-class Classification Problem - Ensemble Performances

5.9.4 Results: Dataset 2 - Stage 50%

The ensemble learner selected for Dataset 2 at Stage 50% is formed by a bagging of Multinomial Logistic Regression models only. The performances obtained on the initial test sample are shown in Figure 5.17. On average, almost all the ensembles we constructed have very good performances and are statistically significant. The ensemble we selected is very robust on every split.

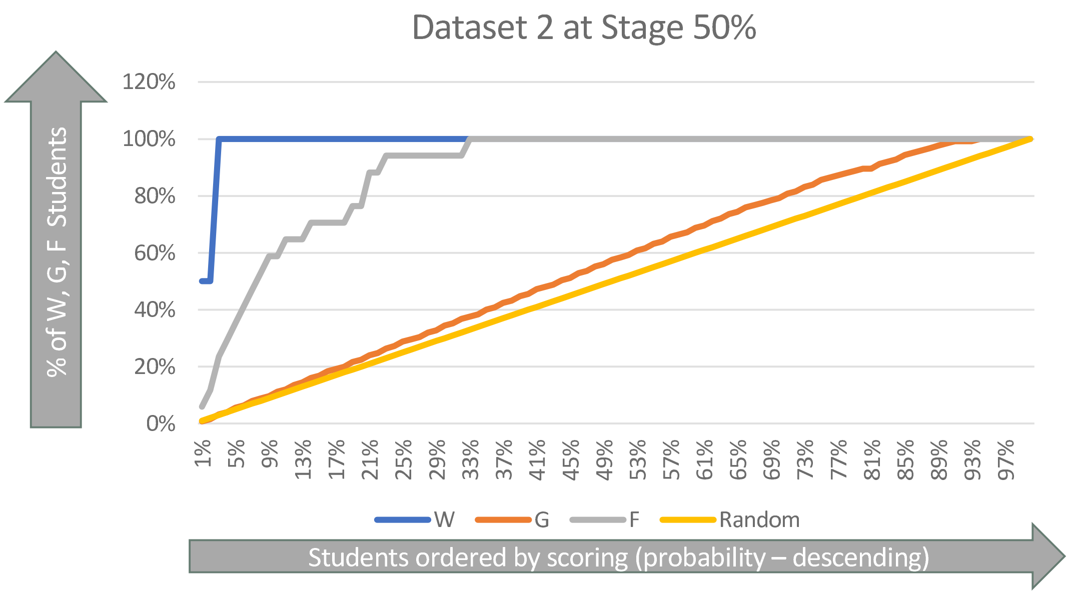


Figure 5.17: Dataset 2 - Stage 50% - Ensemble Learner

Classes G, F, W have Gini Indices equal to 92.3%, 90.7% and 99.3% respectively. The confusion matrix obtained is the following:

Table 5.30 illustrates the performances of the ensemble learner in terms of precision, recall, F-measure and false positive rate per class and on average. These quantities depend on the thresholds τ_F, τ_G and τ_W and the way we defined the predictions. The Accuracy is 93.1%.

	F	**G**	**W**
F	11	5	1
G	3	122	0
W	1	0	1

Table 5.29: Confusion Matrix
$\tau_F = 0.12, \tau_G = 0.62, \tau_W = 0.30$
Ensemble: LR
Dataset 2 - stage 50%

	Precision	**Recall**	**F-measure**	**False Positive Rate**
F	0.65	0.73	0.69	0.27
G	0.98	0.96	0.97	0.04
W	0.50	0.50	0.50	0.50
Avg	0.71	0.73	0.72	0.27

Table 5.30: Dataset 2 - Stage 50% - multi-class Classification Problem - Ensemble Performances

5.9.5 Results Summary

The performances obtained for Dataset 1 and Dataset 2 are very different. For Dataset 1, the models performances depend strongly on the splits. For instance, the same ensemble might perform very well on certain splits but have very low Averaged Gini Index on others, due to a negative Gini index on class F. For Dataset 1 at stage 20%, only 25% of the ensembles had averaged Gini Index above 50% and of all the ensembles only one of them is statistically significant, the one corresponding to a bagging of Neural Networks with 2 hidden layers. Although the evidence shows that this ensemble performs decently on each split we have considered for our experiments, we cannot assume that this is true on every possible other split we might have chosen instead. The problem is so dependent on the split selected, that even the ensemble we chose results in lack of robustness and poor performances.

For Dataset 1 at stage 50%, the averaged Gini Index is in general higher that the averaged Gini Index obtained at stage 20% because the Gini Indices corresponding to classes G (Good students) and class W (Weak students) improve when we add the three features *ES4.1, ES4.2, ES5.1*, in the same way as for the binary classification problem. Indeed, these three features were observed to be the most important ones when we were predicting Good students vs. Weak students in the binary problem, see Section 4.6.2. Since the target variable for the multi-class classification problem, as we saw in Section 5.4, is defined in such a way that some of the Fair students are Weak students in the binary problem, and the remaining Fair students are Good students in the binary problem, the consequence is that when we add the best predictors from the binary problem, they predict incorrectly the Fair students and the Gini Index for class F, for each ensemble and for almost every split is negative or very low, leading to not statistically significant results. In particular there is not even an ensemble among the 256 constructed such that the p-value corresponding to class F is lower than 0.03 on every split .

For this reason, even though for completeness we are going to show the results for Dataset 1 at both stages, it is important to point out that if we were aiming to classify correctly the students for Dataset 1 and to use the classifier for applications in real world, we should not include the last three features, i.e. we should use Dataset 1 at stage 20%.

Dataset 2 was easier to deal with and also the choice of the best ensemble was straightforward. For Dataset 2 at stage 50%, 88% of the ensembles have averaged Gini Indices above 90%, and 96% of the ensembles were statistically significant.
For Dataset 2, the highest averaged Gini Index led us to choose:

- the ensemble of Naive Bayes bagging, k-NN bagging, Logistic Regression bagging and Neural Networks with 2 hidden layers bagging for the 20% stage.

- the ensemble consisting of Logistic Regression bagging only for stage 50%.

This improvement from stage 20% to stage 50% confirms the results obtained for Dataset 2 for the binary classification case, see Section 4.7.2.

5.10 Conclusion and Research Limitations

In this chapter we tackled the multi-class classification problem for the two datasets at stages 20% and 50%. We trained 8 baggings of models for each dataset and considered all the possible ensembles that could be generated by considering the scores produced by inferring them on a test sample.
We compared the performances, and concluded that the ensemble learners to be selected are formed by:

- a bagging of Neural Networks with two layers for Dataset 1 at Stage 20%.

- a bagging of Naïve Bayes classifiers, a bagging of k-NN classifiers, a bagging of Logistic Regression classifiers and a bagging of Neural Networks with 2 hidden layers for the for Dataset 2 at Stage 20%.

- a bagging of Multinomial Logistic Regression classifiers for Dataset 2 at stage 50%.

whereas it was not possible to select a good ensemble for Dataset 1 at stage 50% as none of the ensembles was statistically significant.

The results are good for Dataset 2 both in terms of Averaged Gini Index and p-values, especially if we consider the issues encountered. All the issues described in Section 4.8, such as the size of Dataset 1 and Dataset 2 being unbalanced, were worse when considering the multi-class classification problem, to the point where it was impossible to find a good classifier for Dataset 1 at stage 50% and that the performance obtained for Dataset 1 at stage 20% was poor due to the small sample size and hence the low number of instances of the target class.

Note that optimized ML ensemble models proposed in chapters 4 and 5 can be generalized to other applications such as finance, network security, social media, and healthcare systems. However, the opposite may not necessarily hold true, *i.e.* models originally developed for other applications like those proposed for social media may not be suitable to be applied to the education domain. This is mainly due to the unique nature of educational datasets in which different courses with different tasks and evaluation criteria may result in different ensemble models being selected. Furthermore, educational datasets normally suffer from low number of instances and features, presenting a new set of challenges that are often not considered in traditional ML models developed for other applications.

Chapter 6

Bayesian Optimization with Machine Learning Algorithms Towards Anomaly Detection

6.1 Introduction

Computer networks and the Internet have become an essential component of any organization in this high-tech world. Organizations heavily depend on their networks to conduct their daily work. Moreover, individuals are also dependent on the Internet as a means to communicate, conduct business, and store their personal information [245]. The topic of Cyber-security has garnered significant attention as it greatly impacts many entities including individuals, organizations, and governmental agencies. Organizations have become more concerned with their network security and are allocating more resources to protect it against potential attacks or anomalous activities. Traditional network protection mechanisms have been proposed such as adopting firewalls, authenticating users, and integrating antivirus and malware programs as a first line of defense [246]. Nonetheless, these mechanisms have not been as efficient in providing complete protection for the organizations' networks, especially with contemporary attacks [247].

Typical intrusion detection systems (IDSs) can be categorized into two main types, namely signature-based detection systems (misused detection) and anomaly-based detection systems [248]. Signature-based detection systems compare the observed data with pre-defined attack patterns to detect intrusion. Such systems are effective for attacks with well-known signatures and patterns. However, these systems miss new attacks due to the ever-changing nature of intrusion attacks [249]. On the other hand, anomaly-based detection systems rely on the hypothesis that abnormal behavior differs from normal behavior. Therefore, any deviation from what is considered as normal is classified as anomalous or intrusive. Such systems typically build models based on normal patterns and hence are capable of detecting unknown behaviors or intrusions [250]. Although previous work on IDSs has shown promising improvement, intrusion detection problem remains a prime concern, especially given the high volume of network traffic data generated, the continuously changing environments, the plethora of features

A version of this chapter has been published in [107].

collected as part of training datasets (high dimensional datasets), and the need for real-time intrusion detection [251]. For instance, high dimensional datasets can have irrelevant, redundant, or highly correlated features. This can have a detrimental impact on the performance of IDSs as it can slow the model training process. Additionally, choosing the most suitable subset of features and optimizing the corresponding parameters of the detection model can help improve its performance significantly [252].

In this chapter, we propose an effective signature-based intrusion detection framework based on optimized machine learning classifiers including Support Vector Machine with Gaussian kernel (SVM-RBF), Random Forest (RF), and k-Nearest Neighbors (k-NN) using Bayesian Optimization (BO). These techniques have been selected based on the nature of the selected dataset, i.e. SVM-RBF is selected because the data is not linearly separable. Additional details about the utilized techniques are presented in Section 6.3. This is done to provide a robust and accurate methodology to detect network attacks. The considered methods are titled BO-SVM, BO-RF, and BO-kNN respectively.

The performance is evaluated and compared by conducting different experiments with the ISCX 2012 dataset that was collected from University of New Brunswick [253]. As mentioned in Wu and Banzhaf [249], a robust IDS should have a high detection rate/recall and a low false alarm rate (FAR). Despite the fact that most of intrusion detection methods have high detection rate (DR), they suffer from higher FAR. Thus, this work utilizes optimized machine learning models to minimize the objective function that will maximize the effectiveness of the considered methods. Totally, the feasibility and efficiency of these optimized methods is compared using various evaluation metrics such as accuracy (acc), precision, recall, and FAR. Furthermore, the performance of the three optimized methods in parameter setting are compared with the standard approaches. The main contributions of this chapter include the following:

- Investigate the performance of the optimized machine learning algorithms using Bayesian Optimization to detect network attacks.
- Enhances the performance of the classification models through the identification of the optimal parameters towards objective-function minimization.
- UNB ISCX 2012, a benchmark intrusion dataset is used for experimentation and validation purposes through the visualization of the optimization process of the objective function of the considered machine learning models to select the best approach that identifies malicious network traffic. To the best of our knowledge, no previous related work has adopted Bayesian Optimization on the utilized dataset towards network attack detection.

The remainder of this chapter is organized as follows. Section 6.2 presents the related work. Section 6.3 gives a brief overview of SVM, RF, and k-NN algorithms along with the utilized optimization method. Section 6.4 discusses the research methodology and the experimental results. Finally, Section 6.5 concludes the chapter and provides future research directions.

6.2 Related Work

The intrusion detection problem has been addressed as a classification problem by researchers. Different data mining-based methodologies have been posited to tackle this problem including, SVM [254], Decision Trees [255], k-NN [256], and Naive Bayes [257] classifiers as shown in the short review presented in Tsai et al. [245]. Later, noteworthy research have been implemented and acquired promising results through proposing novel approaches based data mining techniques Wu and Banzhaf [249].

Recently, many research adopted optimization techniques to improve the performance of their approach. For instance, a hybrid approach proposed by Chung and Wahid [258] including feature selection and classification with simplified swarm optimization (SSO). The performance of SSO was further improved by using weighted local search (WLS) to obtain better solutions from the neighborhood [258]. Their experimental results yielded accuracy of 93.3% in detecting intrusions.

Similarly, Kuang et al. [259] proposed a hybrid method incorporating genetic algorithm (GA) and multi-layered SVM with kernel principal component analysis (KPCA) to enhance the performance of the proposed methodology. Another technique introduced by Zhang et al. [260] combining misuse and anomaly detection using RF. A novel algorithm applied catfish effect named, Catfish-BPSO, had been used to select features and enhance the model performance [261]. Authors used leave-one-out cross-validation (LOOCV) with k-NN for fitness evaluation.

6.3 THEORETIC ASPECTS OF THE TECHNIQUES

6.3.1 A. Support Vector Machines (SVM)

SVM algorithm is a supervised machine learning classification technique that identifies the class positive and negative sample by determining the maximum separation hyperplane between the two classes [262]. Depending on the nature of the dataset, different kernels can be used as part of the SVM technique since the kernel determines the shape of the separating hyperplane. For example, a linear kernel can be used in cases where the data is linearly separable by providing a linear equation to represent the hyperplane. However, other kernels are needed in cases where the data is not linearly separable. One such kernel is the Gaussian Kernel . This kernel maps the data points from their original input space into a high-dimensional feature space. The output of the SVM with Gaussian kernel (also known as SVM-RBF) is [263]:

$$f(x) = w^T \Phi(x) + b \tag{6.1}$$

where $\Phi(x)$ represents the used kernel. The goal is to determine the weight vector w^T and intercept b that minimizes the following objective function:

$$\min_{w,b} \frac{1}{2} w^2 + C \sum_{i=1}^{m} [y_i \times cost_1(f(x_i)) + (1 - y_i) \times cost_0(f(x_i))] \tag{6.2}$$

where C is a regularization parameter that penalizes incorrectly classified instances, $cost_i$is the squared error over the training dataset.

6.3.2 k-Nearest Neighbors (k-NN)

k-NN is a simple classification algorithm that determines the class of an instance based on the majority class of its k nearest neighboring points. This is done by first evaluating the distance from the data point to all other points within the training dataset. Different distance measures can be used such as the Euclidean distance or Mahalanoblis distance. After determining the distance, the k nearest points are identified and a majority voting-based decision is made on the class of the considered data point [264].

6.3.3 Random Forests (RF)

RF classifier is an ensemble learning classifier that combines several decision tree classifiers to predict the class [40]. Each tree is independently and randomly sampled with their results combined using majority rule. The RF classifier sends any new incoming data point to each of its trees and chooses the class that is classified by the most trees. RF algorithm works as follows [265]:

1. Choose T number of trees to grow.
2. Choose m number of variables used to split each node.$m \ll M$, where M is the number of input variables.
3. Grow trees; While growing each tree, do the following:
    - Construct a sample of size N from N training cases with replacement and grow a tree from this new sample.
    - When growing a tree at each node, select m variables at random from M and use them to find the best split.
    - Grow tree to maximum size without pruning.
4. To classify point X, collect votes from every tree in the forest and then use majority voting to decide on the class label.

6.3.4 Bayesian Optimization (BO)

Bayesian optimization algorithm [266] tries to minimize a scalar objective function $f(x)$ for x. Depending on whether the function is deterministic or stochastic, the output will be different for the same input x. The minimization process is comprised of three main components: a Gaussian process model for the objective function $f(x)$, a Bayesian update process that modifies the Gaussian model after each new evaluation of the objective function, and an acquisition function $a(x)$. This acquisition function is maximized in order to identify the next evaluation point. The role of this function is to measure the expected improvement in the objective function while discarding values that would increase it [266]. Hence, the expected improvement (EI) is calculated as:

$$EI(x,Q) = E_Q\Big[\max(0, \mu_Q(x_{best}) - f(x))\Big] \tag{6.3}$$

where x_{best} is the location of the lowest posterior mean and $\mu_Q(x_{best})$ is the lowest value of the posterior mean.

6.4 EXPERIMENTAL SETUP AND RESULT DISCUSSION

6.4.1 Dataset Description

In this chapter, the Information Security Centre of Excellence (ISCX) 2012 dataset was used to perform the experiments and evaluate the performance of the proposed approach to detect network attacks. The entire dataset comprises nearly 1.5 million network traffic packets, with 20 features and covered seven days of network activity (i.e. normal and intrusion). Additional information about the dataset are available in [253]. A random subset has been extracted from the original dataset. The training data contains 30,814 normal traces and 15,375 attack traces while the testing data contains 13,154 normal traces and 6,580 additional attack traces.

6.4.2 Experimental setup and Data Pre-processing

The proposed techniques were implemented using MATLAB 2018a. Experiments were carried out in an Intel® Core^{TM} i7 processor @ 3.40 GHz system with 16GB RAM running Windows 10 operating system. The selected dataset was transformed from their original format into a new dataset consisting of 14 features. We eliminated the payload features which include the actual packet as most of their contents were empty, while start time, and end time features have been replaced by duration feature. In the data normalization stage, attributes were scaled between the range [0,1] by using Min-Max method to eliminate the bias of features with greater values, the mathematical computation is as follows:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{6.4}$$

As most of the classifiers do not accept categorical features [267], data mapping technique was used to transform the non-numeric values of the features into numeric ones, named *categorical* in MATLAB.

6.4.3 Prediction Performance Measures

To evaluate and compare prediction models quantitatively, four metrics are used, namely the accuracy, precision, recall, and false alarm rate as per the equations provided in Section 2.3.

6.4.4 Results Discussion

The aim of the work is to discover the optimized models' parameters of the utilized classifiers to classify the network intrusion data with the selected parameters. The experimental scheme has been done for each technique to reduce the cost function by tuning all possible parameters to obtain the highest classification accuracy and the minimum FAR. To that end, BO technique is used to determine the optimal parameters for the considered machine learning models. For instance, the optimal values of C and γ (for SVM), the depth of trees and the adopted ensemble method (for RF), and the value of k and the distance measure method (for k-NN) are determined.

For example, if we have a set of machine learning model parameters $P* = P_1, P_2, \ldots, P_n$ where P_i is a parameter of the parameters subset that needs tuning, then BO tries to minimize the following cost function:

$$P* = \min J(P) \tag{6.5}$$

where $J(P)$ is the associated cost function.

To visualize the behavior of the BO technique combined with the machine learning technique on the training dataset, Figures 6.1 and 6.2 as well as Table 6.1 depict how BO tunes the parameters towards the global minimum value of the SVM cost function with respect to C and γ as parameters subset. According to the figures, a unique global minimum is obtained for $C = 433.32$ and $\gamma = 1.0586$. This in turn leads to improving the model's training accuracy as shown in Table 6.2 from 99.58% without optimization to 99.95% after optimization. Additionally, the testing accuracy increases from 99.59% to 99.84%. On the other side, the FAR had promising results with a reduction of 0.01 and 0.007 in the training and testing datasets respectively. Table 2 also shows more details about the optimization processing time. Note that the training accuracy refers to the ML models' accuracy during the training stage with the training sample dataset. In contrast, the testing accuracy refers to the ML models' accuracy during the testing stage with the testing sample dataset which differs from the training sample dataset.

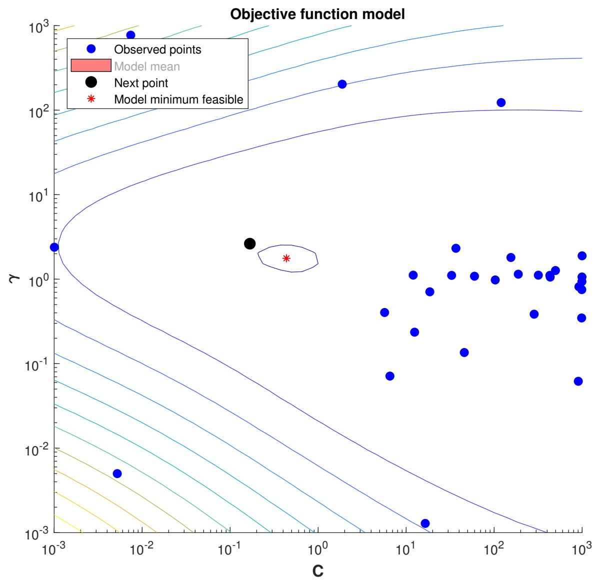


Figure 6.1: Optimized SVM Contour

Table 6.1: Optimization parameters for each classifier

	BO-SVM		BO-k-NN		BO- RF	
Best Parameters	BoxConstraint (C)	433.32	NumNeighbors	1	Method	AdaBoost
	KernelScale (γ)	1.0586	Distance	Mahalanobis	MaxNumSplits	1004
Total function evaluations	30		30		30	
Total elapsed time in seconds	6175.78		2272.50		771.24	

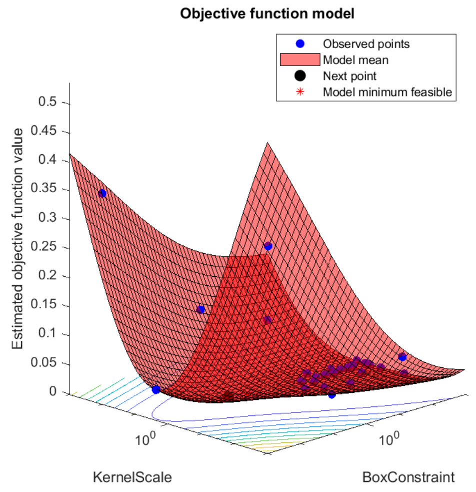


Figure 6.2: Optimized SVM objective function model

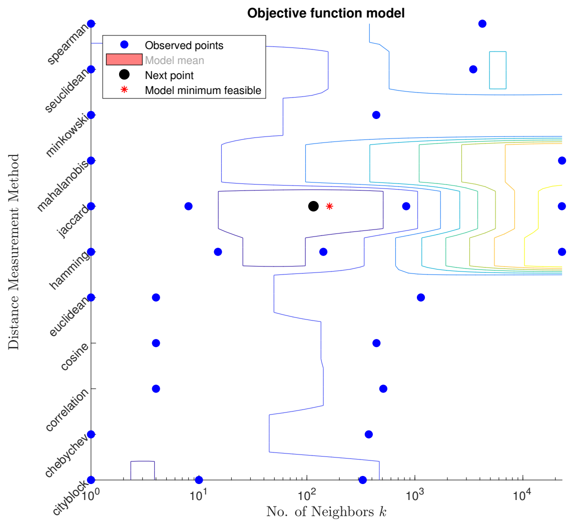


Figure 6.3: Optimized k-NN Contour

Table 6.2: Performance results of the three classifiers

	Training				Testing			
Classifier	Acc(%)	Precision	Recall	FAR	Acc(%)	Precision	Recall	FAR
SVM-RBF	99.58	0.994	0.999	0.011	99.59	0.995	0.999	0.010
K-NN (k=5)	99.59	0.9965	0.998	0.008	99.36	0.994	0.996	0.012
RF	99.96	0.999	1.00	0.001	99.88	0.998	0.999	0.002
BO-SVM	99.95	0.999	1.00	0.001	99.84	0.998	0.999	0.003
BO-k-NN	99.98	0.999	1.00	0.001	99.93	0.999	0.999	0.001
BO-RF	99.98	0.999	1.00	0.001	99.92	0.999	0.999	0.001

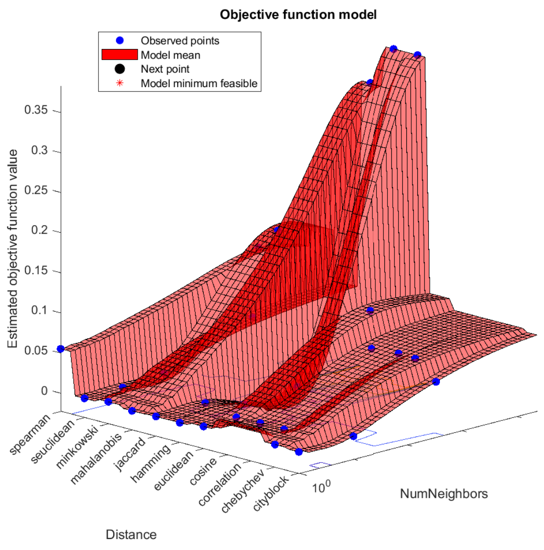


Figure 6.4: Optimized k-NN Objective Function Model

Similarly, Figures 6.3 and 6.4 and Table 6.1 show how the BO technique is minimizing the cost function $J(P)$ for k-NN algorithm with respect to the number of neighbors k and the distance measuring method. A unique global minimum is achieved for the values of $k = 1$ and Mahalanobis distance as the distance measuring method.

Figures 6.5, 6.6, and 6.7 visualize the change in the objective function value vs the number of function evaluations for BO-SVM, BO-RF, and BO-kNN respectively. It can be observed that the objective function reaches its global minimum within 30 iterations at most. This reiterates the efficiency of the BO technique in optimizing the considered algorithms.

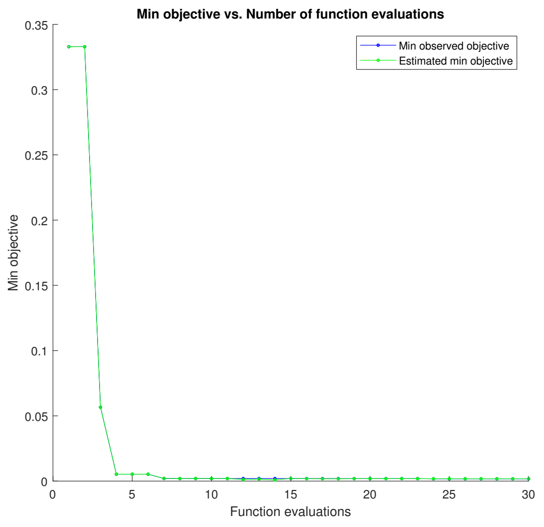


Figure 6.5: BO-SVM Objective Function vs Number of Function Evaluations

By applying BO-RF, a unique global minimum is achieved with 1004 tree splits (Tree

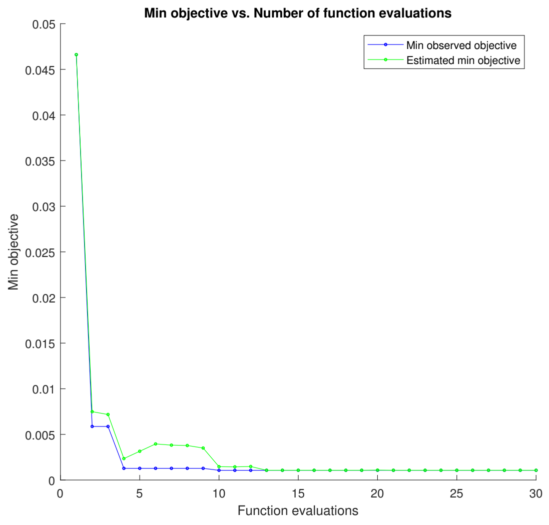


Figure 6.6: BO-kNN Objective Function vs Number of Function Evaluations

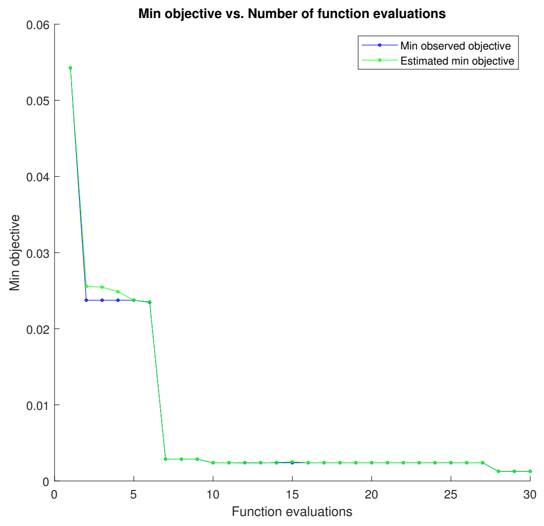


Figure 6.7: BO-RF Objective Function vs Number of Function Evaluations

Depth) and AdaBoost as a tree method. The BO improves the training accuracy from 99.97% to 99.98% while the testing accuracy improves from 99.88% to 99.92%. The FAR remains steady in the training dataset and is reduced by 0.001 in the testing dataset. Furthermore, Table 6.1 indicates that the BO find that AdaBoost is the best ensemble method to build the tree.

It is also worth mentioning that Naïve Bayes classifier was utilized at the initial stage of the experiment. However, due to the fact that the dataset's features are not fully independent, the classifier shows a low accuracy of 87.23% and 87.65% on the training and testing datasets respectively. Hence, the Naïve Bayes classifier was excluded from the experiment.

Based on the previous publications, our results outperform the results of previous experiments conducted using ISCX 2012 such as the results shown in [268] with their model acheiving about 95% as overall accuracy using their proposed technique. Additionally, [269] reported the highest accuracy of 99.8% and 99.0% for the training and testing phases respectively.

6.5 Conclusions

In this chapter, we utilized a Bayesian optimization method to enhance the performance of intrusion detection methodology based on three conventional classifiers; Support Vector Machine with Gaussian kernel (SVM-RBF), Random Forest (RF), and k-Nearest Neighbor (k-NN). The BO optimization method has been applied to set the parameters of these classifiers by finding the global minimum of the corresponding objective function. In order to have an efficient machine learning-based intrusion detection system with high accuracy rate and a low false positive rate, BO was able to improve the utilized classifiers. The experimental results show not only is the proposed optimization method more accurate in detecting intrusions, but also it can find the global minimum of the objective function which leads to better classification results. Overall, k-NN with Bayesian optimization has achieved the optimum performance on ISCX 2012 dataset in terms of accuracy, precision, recall, and false alarm rate.

Chapter 7

Multi-Stage Optimized Machine Learning Framework for Network Intrusion Detection

7.1 Introduction

The Internet has become an essential block in our current times with individuals as well as organizations heavily dependent on it to facilitate their communication, conduct business, and store their information [245]. This dependence is coupled with these individuals and organizations' concern about the security and privacy of their activities. Accordingly, the area of cyber-security has garnered significant attention from both the industry and academia. To that end, more resource are being deployed and allocated to protect modern Internet-based networks to protect them against potential attacks or anomalous activities. Several protection mechanisms have been proposed such as firewalls, user authentication, and the deployment of antivirus and malware programs as a first line of defense [246]. However, these mechanisms have not been able to completely protect the organizations' networks, particularly with contemporary attacks [247].

Typically, network intrusion detection systems (NIDSs) can be divided into two main categories: signature-based detection systems (misused detection) and anomaly-based detection systems [248]. Signature-based detection systems base their detection on the observation of pre-defined attack patterns. Thus, they have proven to be effective for attacks with well-known signatures and patterns. However, such systems are vulnerable against new attacks as they do not have any previous observations to enable them to detect these new attacks [249]. In contrast, anomaly-based detection systems base their detection on the observation of any behavior or pattern that deviates from what is considered to be normal. Therefore, these systems are capable of detecting unknown attacks or intrusions based on the built models that characterize normal behavior [250].

Despite the continuous improvements in NIDS performance, there still remains room for further enhancements. This is particularly evident given the high volume of generated network traffic data, the continuously evolving environments, the vast amount of features collected that

A version of this chapter has been accepted in [270].

form the training datasets (high dimensional datasets), and the need for real-time intrusion detection [251]. For example, having redundant or irrelevant features can have a negative impact on the detection capabilities of NIDSs as it slows down the model training process. Therefore, it is important to choose the most suitable subset of features and optimize the parameters of the machine learning (ML)-based detection models to enhance their performance [252].

This chapter extends our previous work in Chapter 6 published in [107] by proposing a novel multi-stage optimized ML-based NIDS framework that reduces the computational complexity while maintaining its signature-based intrusion detection performance. To that end, this work first studies the impact of oversampling techniques on the models' training sample size and determines the minimum suitable training size for effective intrusion detection. Furthermore, it compares between two different feature selection techniques, namely information gain and correlation based feature selection, and explores their effect on the models' detection performance and time complexity. Moreover, different ML hyper-parameter optimization techniques are investigated to enhance the NIDS's performance and ensure its effectiveness and robustness. The main differences between this work and our previous work in [107] are as follows:

- This work first studies the suitable training sample size by investigating the training accuracy and cross-validation accuracy illustrated using the learning curve.
- This work addresses the class imbalance problem by using an oversampling technique. This was not considered in the conference paper.
- This work investigated the use of feature selection and studied its impact on the training sample size, feature set size, and consequently on the overall training complexity.
- This work compares the performance of five different optimization techniques and studies their impact on the overall detection performance of the proposed ML-models. In the conference paper, only one optimization technique was considered.
- This work considered more recent datasets, namely the CICIDS 2017 (which is an extension of the previously used ISCX 2012 dataset) and the UNSW-NB 2015 dataset.

To evaluate the performance of the proposed optimized ML-based NIDS framework, two recent state-of-the-art intrusion detection datasets are used, namely the CICIDS 2017 dataset [19] (which is the updated version of the ISCX 2012 dataset [253] used in our previous work [107]) and the UNSW-NB 2015 dataset [20]. The performance evaluation is conducted using various evaluation metrics such as accuracy (acc), precision, recall, and false alarm rate (FAR).

The remainder of this chapter is organized as follows: Section 7.2 briefly summarizes some of the previous literature works that focused on this research problem and presents its limitations. Section 7.3 summarizes the contributions of this work. Section 7.4 presents the proposed multi-stage optimized ML-based NIDS framework and discusses its theoretical background. Section 7.5 describes the two datasets under consideration in more details. Section 7.6 presents and discusses the experimental results obtained. Finally, Section 7.7 concludes the chapter and proposes potential future research endeavors.

7.2 Related Work and Limitations

7.2.1 Related Work

ML classification techniques have been proposed as part of various network attack detection frameworks and applications using different classification models such as Support Vector Machines (SVM) [254], Decision Trees [255], KNN [256], and Naive Bayes [257] as illustrated in [245]. One such application is the DNS typo-squatting attack detection framework presented in [16, 271]. Also, ML techniques have been proposed to detect zero-day attacks as illustrated by the probabilistic Bayesian network model presented in [272]. On the other hand, hybrid ML-fuzzy logic based system that focuses on distributed denial of service (DDoS) attack detection has been proposed in [273]. Additionally, these ML classification techniques have also been proposed for bot net detection [274] as well as for mobile phone malware detection [275].

Along the same line, several previous works focused on the use of ML classification techniques for network intrusion detection. For example, Salo *et al.* conducted a literature survey and identified 19 different data mining techniques commonly used for intrusion detection [108]. The result of this review highlighted the need for more ML-based research to address real-time IDSs. To that end, the authors proposed an ensemble feature selection and an anomaly detection method for network intrusion detection [109]. On the other hand, Li *et al.* proposed a decision tree (DT)-based IDS model for autonomous and connected vehicles [127]. The goal of the IDS is to detect both intra-vehicle and external vehicle network attacks [127].

In a similar fashion, several previous research works proposed the use of various optimization techniques to enhance the performance of their NIDSs. For example, Chung and Wahid proposed a hybrid approach that included feature selection and classification with simplified swarm optimization (SSO) in addition to using weighted local search (WLS) to further enhance its performance [258]. In a similar vein, Kuang *et al.* presented a hybrid GA-SVM model associated with kernel principal component analysis (KPCA) to improve the performance [259]. On the other hand, Zhang *et al.* combined misuse and anomaly detection using RF [260]. In contrast, our previous work in [107] proposed a Bayesian optimization model to hyper-tune the parameters of different supervised ML algorithms for signature-based IDSs [107]. More specifically, they tune the parameters of SVM, Random Forest (RF), and K-nearest neighbors (KNN) algorithms.

7.2.2 Limitations of Related Work

Despite the many previous works in the literature that focused on the intrusion detection problem, the previously proposed models suffer from various shortcomings. For example, many of these works do not focus on the class imbalance issue often encountered in intrusion detection datasets which typically results in misleadingly high accuracies around the 99% mark. Also, the training sample size is often selected randomly rather than using a systematic approach. Another shortcoming is that they used outdated datasets such as NLS KDD99. Additionally, the results reported are usually only done using one dataset rather than being validated using multiple datasets. Furthermore, few works considered the hyper-parameter optimization using different techniques as they tend to use only one method. Also, only a handful research works studied the time complexity of their proposed framework, a metric that is often overlooked.

7.3 Research Contributions

The main contributions of this chapter can be summarized as follows:

- Propose a novel multi-stage optimized ML-based NIDS framework that reduces computational complexity and enhances detection accuracy.
- Study the impact of oversampling techniques and determine the minimum suitable training sample size for effective intrusion detection.
- Explore the impact of different feature selection techniques on the NIDS detection performance and time (training and testing) complexity.
- Propose and investigate different ML hyper-parameter optimization techniques and their corresponding enhancement of the NIDS detection performance.
- Evaluate the performance of the optimized ML-based NIDS framework using two recent state-of-the-art datasets, namely the CICIDS 2017 dataset [19] and the UNSW-NB 2015 dataset [20].
- Compare the performance of the proposed framework with other works from the literature and illustrate the improvement of detection accuracy, reduction of FAR, and a reduction of both the training sample size and feature set size.

To the best of our knowledge, no previous works in the literature proposed such a novel multi-stage optimized ML-based NIDS framework and evaluated it using these datasets.

7.4 Proposed Multi-Stage Optimized ML-based NIDS Framework

7.4.1 General Framework Description:

This work focuses on building a multi-stage optimized ML-based NIDS framework that achieves high detection accuracy, low FAR, and has a low time complexity. To achieve this goal, the proposed framework is divided into three main stages. The first stage includes the data pre-processing which includes performing Z-score normalization and Synthetic Minority Oversampling TEchnique (SMOTE). This is done to improve the performance of the training model and reduce the class-imbalance often observed in network traffic data [276]. In turn, this can reduce the training sample size since the ML model would have enough samples to understand the behavior of each class [277].

The second stage of the proposed framework is conducting a feature selection process to reduce the number of features needed for the ML classification model. This is done to reduce

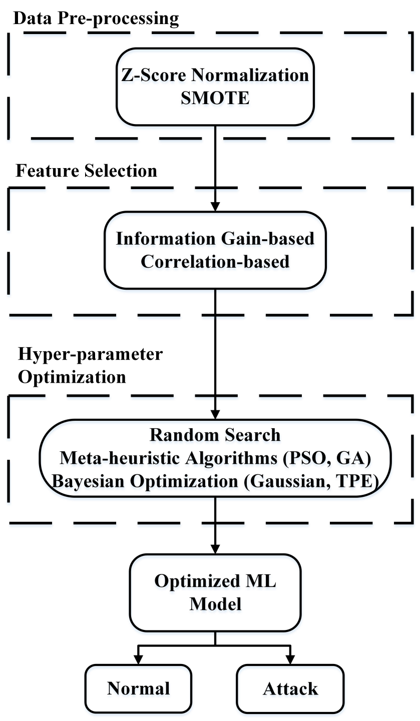


Figure 7.1: Proposed Multi-stage Optimized ML-based NIDS Framework

the time complexity of the classification model and consequently decrease its training time without sacrificing its performance [278]. With that in mind, two different methods are compared withing this stage of the framework.

The third stage of the framework involves the optimization of the hyper-parameters of the different ML classification models considered. To that end, three different hyper-parameter tuning/optimization models are investigated, namely random search, meta-heuristic optimization algorithms including particle swarm optimization (PSO) and genetic algorithm (GA), and Bayesian Optimization (BO) algorithm. These models represent three different hyper-parameter tuning/optimization categories which are heuristics [279], meta-heuristics [280], and probabilistic global optimization [281] models respectively.

The results of the aforementioned optimization stages are combined to build the optimized ML classification model for effective NIDS system that classifies new instances as either normal or attack instances. Figure 7.1 illustrates the different stages of the proposed framework.

7.4.2 Data Pre-processing:

As mentioned earlier, the data pre-processing stage involves performing data normalization using the Z-score method and minority class oversampling using the SMOTE algorithm. In what follows, these two algorithms are briefly discussed.

Z-Score Normalization

The first step of the data pre-processing stage is performing Z-score data normalization. However, to be able to do so, the data is first encoded using a label encoder to transform any categorical features into numerical ones. Then, data normalization is performed by calculating the normalized value x_{norm} of each data sample x_i as follows:

$$x_{norm} = \frac{x_i - \mu}{\sigma} \tag{7.1}$$

where μ being the mean vector of the features and σ being the standard deviation. It is worth mentioning that the Z-score data normalization is performed given that ML classification models tend to perform better with normalized datasets [282].

SMOTE Technique

The second step of the data pre-processing stage is performing minority class oversampling using the SMOTE algorithm. The minority class is typically the class with the lowest number of instances in the dataset. SMOTE algorithm aims at synthetically creating more instances of the minority class to reduce the class-imbalance which often negatively impacts the ML classification model's performance [276]. Therefore, it is important to perform minority class oversampling, especially for network traffic datasets which typically suffer from this issue by having a lower number of attack instances when compared to normal traffic instances, to improve the performance of the training model [277].

Upon analyzing the original minority class instances, SMOTE algorithm synthesizes new instances using the k-nearest neighbors concept. Accordingly, the algorithm groups all the instances of the minority class into one set $X_{minority}$. For each instance X_{inst} within $X_{minority}$, a new synthetic instance X_{new} is determined as follows [283]:

$$X_{new} = X_{inst} + rand(0,1) * \left(X_j - X_{inst}\right), j = 1, 2, ..., k \tag{7.2}$$

where $rand(0,1)$ is a random value in the range [0,1] and X_j is a randomly selected sample from the set $\{X_1, X_2, ..., X_k\}$ of k nearest neighbors of X_{inst}. Note that unlike other oversampling algorithms that merely replicate minority class instances, the SMOTE algorithm generates new high quality instances that statistically resemble the samples of the minority class [277, 283].

7.4.3 Feature Selection:

This work compares between two different feature selection techniques, namely information gain-based and correlation-based feature selection, and explores their effect on the models' detection performance and time complexity. More specifically, feature selection aims at reducing the complexity of the classification model and consequently decrease the model's training time without sacrificing its performance [278]. This is particularly relevant when designing ML models for large scale systems that generate high dimensional data [278]. In what follows, a brief overview of these two methods is given.

Information Gain-based Feature Selection

The first algorithm considered in this work is the information gain-based feature selection (IGBFS) algorithm. This algorithm belongs to the group of "Information Theory" feature selection techniques [284]. Accordingly, it uses information theory concepts such as entropy and mutual information to select the relevant features [285]. Simply put, the IGBFS ranks features based on the amount of information (in bits) that can be gained from them about the target class and selects the ones with the highest amount of information. Therefore, the feature evaluation function is [285]:

$$\begin{aligned} I(S;C) &= H(S) - H(S|C) \\ &= \sum_{s_i \in S} \sum_{c_j \in C} P(s_i, c_j) log \frac{P(s_i, c_j)}{P(s_i) \times P(c_j)} \end{aligned} \tag{7.3}$$

where $I(S;C)$ is the mutual information between feature subset S and class C, $H(S)$ is the entropy/uncertainty of discrete feature subset S, $H(S|C)$ is the conditional entropy/uncertainty of discrete feature subset S given class C, $P(s_i, c_j)$ is the joint probability of feature having a value s_i and class being c_j, $P(s_i)$ is the probability of feature having a value s_i, and $P(c_j)$ is the probability of class being c_j. The information gained from each feature about the target class is calculated using these values. Then, the ones with the highest amount of information are chosen as part of the feature subset provided for the ML classification model.

Correlation-based Feature Selection

The second feature selection algorithm considered in this work is the correlation-based feature selection (CBFS) algorithm. This algorithm belongs to the group of "Traditional Statistical" feature selection techniques [286]. It is often used due to its simplicity since it ranks features based on their correlation with the class to be predicted and selects the highest ones [287]. In essence, CBFS includes a feature as part of the subset if it is considered to be relevant (*i.e.* if it is highly correlated with or predictive of the class [287, 288]). When using CBFS, the Pearson's correlation coefficient is used as the feature subset evaluation function. Therefore, the evaluation function is [287]:

$$Merit_S = \frac{k \times \overline{r_{cf}}}{\sqrt{k + k \times (k-1) \times \overline{r_{ff}}}} \tag{7.4}$$

where $Merit_S$ is the merit of the feature subset S, k is the number of features in feature subset S, $\overline{r_{cf}}$ is the average class-feature Pearson correlation, and $\overline{r_{cf}}$ is the average feature-feature Pearson correlation. This equation is used to rank the feature subsets with the subset having the highest correlation with the target class being chosen for the ML training model.

7.4.4 Hyper-parameter Optimization:

As mentioned earlier, the third stage of the framework focuses on optimizing the hyper-parameters of the different ML classification models considered. This is done in an attempt to improve the performance of ML algorithms. This is because each classification algorithm is governed by a set of parameters that dictate its predictive performance [241]. Therefore, optimizing the ML classification model's hyper-parameters can improve their intrusion detection performance.

This work explores different hyper-parameter tuning/optimization methods, namely random search (RS), PSO and GA meta-heuristic algorithms, and Bayesian optimization algorithm [107, 241, 289]. These methods are briefly described in the following subsections.

Random Search

The first hyper-parameter optimization technique considered is the RS method. This method belongs to the class of heuristic optimization models [279]. Similar to grid search algorithm, RS also tries different combinations of the parameters to be optimized. In mathematical terms, this translates to the following optimization model:

$$\max_{parm} f(parm) \tag{7.5}$$

where f is an objective function to be maximized (typically the accuracy of the model) and *parm* is the set of parameters to be tuned. However, in contrast to the grid search method, the RS method does not perform an exhaustive search by trying all possible combinations, but rather only randomly chooses a subset of combinations to test [279]. Therefore, RS tends to outperform grid search method, especially when the number of hyper-parameters to be optimized is small [279]. Additionally, this method also allows for the optimization to be performed in parallel, further reducing its computational complexity [241].

Meta-heuristic Optimization Algorithms

The second class of hyper-parameter optimization methods considered in this work is meta-heuristic optimization algorithms. Such algorithms aim at identifying or generating a heuristic algorithm that may provide a sufficiently good solution to the optimization problem at hand [290]. Such algorithms tend to find suitable solutions for combinatorial optimization problems with a lower computational complexity [290]. Thus, they are good candidates for ML hyper-parameter optimization.

Within this group of algorithms, this work considers two well-known meta-heuristics for hyper-parameter optimization, namely Particle Swarm Optimization and Genetic Algorithms. In what follows, a brief description of each algorithm is provided.

1. Particle Swarm Optimization (PSO): PSO is a well-known meta-heuristic algorithm that aimed at simulating the social behavior such as flocks of birds traveling to a "promising position" [291]. In the case of hyper-parameter optimization, the desired "position" is the suitable values for the hyper-parameters. In general, PSO algorithm uses a population or a set of particles to search for a suitable solution by iteratively updating these particles' position within the search space.
   More specifically, each particle looks at its own best previous experience *pbest* (representing the cognition part) and the best experience of other particles *gbest* (representing the social part) to determine how it will change its searching direction. Mathematically speaking, the position of the particle at each iteration t is represented as a vector

$x_i^t = \{x_{i1}^t, x_{i2}^t, ..., x_{iD}^t\}$ and its velocity as $v_i^t = \{v_{i1}^t, v_{i2}^t, ..., v_{iD}^t\}$ where D represents the number of parameters to be optimized. Assuming that $pbest_i^t$ = is particle i's best solution until iteration t and $gbest^t$ is the best solution within the population at iteration t, each particle changes its velocity as follows [291]:

$$\begin{aligned} & v_{id}^t = v_{id}^{t-1} + c_1 r_1 (pbest_{id}^t - x_{id}^t) + c_2 r_2 (gbest_d^t - x_{id}^t) \\ & d = 1, 2, ...D \end{aligned} \tag{7.6}$$

where c_1 is the particle's cognition learning factor, c_2 the social learning factor, and r_1 and r_2 are random numbers in the range [0,1]. Based on that, the particle's new position becomes [291]:

$$x_{id}^{t+1} = x_{id}^t + v_{id}^t \; d = 1, 2, ...D \tag{7.7}$$

Within the context of hyper-parameter optimization, $x_i^t = parm$ where $parm$ is the specific set of parameters for the ML model under consideration that is being optimized. For example, in the case of SVM, the parameters are C and γ.

2. Genetic Algorithm (GA): GA is another well-known meta-heuristic algorithm that is inspired by the evolution and the process of natural selection [292]. This algorithm is often used to identify high-quality solutions to combinatorial optimization problems using biologically inspired operations including mutation, crossover, and selection [292]. Using these operators, GA algorithms can search the potential solution space efficiently [292].
   Within the context of ML hyper-parameter optimization, GA algorithm works as follows [292]:

   a) Initialize a population of random solutions denoted as chromosomes. Each chromosome is a vector of potential combinations of values of the hyper-parameters to be optimized.
   b) Determine the fitness of each chromosome using a fitness function. This function is typically the accuracy of the ML model when using the vector of hyper-parameter values of each chromosome.
   c) Rank the chromosomes according to their relative fitness in descending order.
   d) Replace least-fit chromosomes with new chromosomes that are generated through the process of crossover and mutation. Crossover refers to the process of generating new off-springs from two parent chromosomes by exchanging their "genes" (each gene represents a value for one of the hyper-parameters). On the other hand, mutation refers to the processing of altering a chromosome randomly. Within this process, a gene is randomly selected and altered.
   e) Repeat steps b)-d) until the performance is no longer improving or some stopping criterion is met.

   It is worth mentioning that each chromosome is represented as an encoded string of bits of length l which depends on the range of values for each hyper-parameter.
   Due to its effectiveness in identifying very good solutions (near-optimal in many cases),

this meta-heuristic has been used in a variety of applications including workflow scheduling [293], photovoltaic systems [294], wireless networking [295], and in this case machine learning [296].

Bayesian Optimization

The third hyper-parameter optimization method considered in this work is the Bayesian Optimization method. As mentioned earlier, this method belongs to the class of probabilistic global optimization models [281]. This method aims at minimizing a scalar objective function $f(x)$ for some value x. The output of this optimization process for the same input x differs based on whether the function is deterministic or stochastic [266]. The minimization process is divided into three main parts: an surrogate model that fits all the points of the objective function $f(x)$, a Bayesian update process that modifies the surrogate model after each new evaluation of the objective function, and an acquisition function $a(x)$. Different surrogate models can be assumed for the first part, namely the Gaussian Process and the Tree Parzen Estimator.

1. Gaussian Process (GP): In this case, the model is assumed to follow a Gaussian distribution. Hence, the model is of the form [297]:

$$p(f(x)\Big|\, x, parm) = N(f(x)\Big|\, \hat{\mu}, \hat{\sigma}^2) \tag{7.8}$$

   where *parm* is the configuration space of the hyper-parameters to be optimized and $f(x)$ the value of the objective function with $\hat{\mu}$ and $\hat{\sigma}^2$ being its mean and variance respectively. It is worth noting that such a model is effective when the number of hyper-parameters to be optimized is small, but is ineffective for conditional hyper-parameters [298].

2. Tree Parzen Estimator (TPE): In this case, the model is assumed to follow one of two density functions, $l(x)$ or $g(x)$ depending on some pre-defined threshold $f^*(x)$ as follows [297]:

$$p(x\Big|\, f(x), parm) = \begin{cases} l(x) & \text{if } f(x) < f^*(x) \\ g(x) & \text{if } f(x) > f^*(x) \end{cases} \tag{7.9}$$

   where again *parm* is the configuration space of the hyper-parameters to be optimized and $f(x)$ the value of the objective function. Accordingly, it can be observed that the TPE estimators follow a tree-structure. Note that the TPE method is capable of optimizing all hyper-parameter types [298].

Based on the surrogate model assumption, the acquisition function is maximized to determine the subsequent evaluation point. The role of the function is to measure the expected improvement in the objective while avoiding values that would increase it [266]. Therefore, the expected improvement (EI) can be determined as follows:

$$EI(x, Q) = E_Q\Big[\max(0, \mu_Q(x_{best}) - f(x))\Big] \tag{7.10}$$

where x_{best} is the location of the lowest posterior mean and $\mu_Q(x_{best})$ is the lowest value of the posterior mean.

7.4.5 Complexity:

To determine the time complexity of the proposed multi-stage optimized ML-based NIDS framework, we need to determine the complexity of each algorithm used in each stage. Given that this work compares the performance of different algorithms within the different stages of the framework, the overall time complexity is determined by the combination of algorithms that results in the highest aggregate complexity.

It is assumed that the data is composed of M samples and N features. Starting with the first stage, the complexity of the Z-score normalization process is $O(N)$ since we need to normalize all the samples of the N features within the dataset. On the other hand, the complexity of the SMOTE algorithm is $O(M_{min}^2 N)$ where M_{min} is the number of samples belonging to the minority class [299]. Thus, the complexity of the first stage is $O(M_{min}^2 N)$.

The complexity of the second stage is depended on the complexity of the different feature selection algorithms considered. The complexity of Correlation-based feature selection is $O(MN^2)$. This is based on the fact that this method needs to calculate all the class-feature and feature-feature correlations [287]. In contrast, the complexity of the information gain-based feature selection method is $O(MN)$. This is due to the fact that this method has to calculate the joint probabilities of the class-feature interaction [285]. Therefore, the overall complexity of the second stage is $O(MN^2)$.

In a similar fashion, the complexity of the third stage depends on the complexity of each of the hyper-parameter optimization methods and the underlying ML model. Starting with the RS method, its complexity is $O(N_{parm} log N_{parm})$ where N_{parm} is the number of parameters to be optimized [300]. On the other hand, the complexity of the PSO algorithm is $O(N_{parm} N_{pop})$ where N_{pop} is the population size, *i.e.* the number of swarm particles or potential solutions that we start with [301]. In a similar fashion, it can be shown that the complexity of the GA algorithm is also $O(N_{parm} N_{pop})$ where N_{pop} is the population size, *i.e.* the number of chromosomes/potential solutions at the initialization stage [302]. For the GP-based BO algorithm, the complexity is $O(M_{red}^3)$ where M_{red} is the size of the reduced training sample. This is because the optimization process is carried on the training sample chosen after pre-processing and feature selection. In contrast, the time complexity of the TPE-based BO model is $O(M_{red} log M_{red})$ since this model follows a tree-like structure when performing the optimization [303].

Based on the aforementioned discussion, the overall complexity of the proposed framework is $O(MN^2)$. This is because the second stage will dominate the complexity as it would still use the complete dataset rather than the reduced training dataset. As such, even if we consider the complexity of the potential ML classification model (for example the complexity of KNN classifier can be estimated as $O(M_{red} N_{red})$ [304, 305] where N_{red} is the size of the reduced feature set) is dependent on the reduced training sample dataset with reduced feature size. Hence, the multi-stage optimized ML-based NIDS framework's complexity is $O(MN^2)$. Note that determining the overall time complexity of the complete framework including the optimized ML model training is essential since the model will be frequently re-trained to learn new attack patterns. This is based on the fact that network intrusion attacks continue to evolve and thus organizations need to have a flexible and dynamic NIDSs to keep up with these new attacks.

7.4.6 Security Considerations:

The proposed multi-stage optimized ML-based NIDS framework is a signature-based NIDS system. This is illustrated by the fact that the framework oversamples the minority class, which typically is the attack class in network traffic [108, 306]. Thus, the framework learns from the observed patterns of the known initiated attacks [108, 306]. However, it is worth noting that the framework can work as an anomaly-based NIDS since it is trained by adopting a binary classification model so that it can classify any anomalous behavior as an attack.

This framework can be deployed as one module within a more comprehensive security framework/policy that an individual or organization can adopt. This security framework/policy can include other mechanisms such as firewalls, deep packet inspection, user access control, and user authentication mechanisms [307][308]. This would offer a multi-layer secure framework that can preserve the privacy and security of the users' data and information.

7.5 Datasets Description

This work uses two state-of-the-art intrusion datasets to evaluate the performance of the proposed multi-stage optimized ML-based NIDS framework. In what follows, a brief description of the two datasets is given.

7.5.1 CICIDS 2017

The first dataset under consideration is the Canadian Institute of Cybersecurity's IDS 2017 (CICIDS2017) dataset [19]. This dataset is an extension of the ISCX 2012 dataset used in our previous work [107]. The dataset was generated with the goal of it resembling realistic background traffic [19]. As such, the dataset contains benign and 14 of the most up-to-date common network attacks. The data collection process span a duration of five days from Monday July 3 till Friday July 7, 2017. Within this period, different attacks where generated during different time windows using two separate networks, namely the victim network and the attack network. Each network consisted of all the necessary equipment such as routers, computers running different operating systems (Linux, Windows, and Macintosh), servers, switches, and firewalls. Interested readers are encouraged to check the testbed architecture details in [19]. The resulting dataset contained **3,119,345 instances** and **83 features** (1 class feature and 82 statistical features) representing the different characteristics of a network traffic request such as duration, protocol used, packet size, as well as source and destination details. However, it was noticed that nearly 300,000 samples were unlabeled and hence were discarded. Therefore, the refined dataset considered in this work contains **2,830,540 instances in total** with **2,359,087** being **BENIGN** and **471,453** being **ATTACK**. Note that the attack instances represent various types of real-world network traffic attacks such as denial-of-service (DoS) and port scanning. However, this work merged all attacks into one label as the goal is to detect an attack regardless of its nature.

Fig. 7.2 shows the first and second principal components for the CICIDS 2017 dataset. It can be clearly seen that the two classes are intertwined. Moreover, it can be observed that the features of the dataset are non-linear. Hence, we would expect a non-linear kernel to perform better in classifying the instances of this dataset.

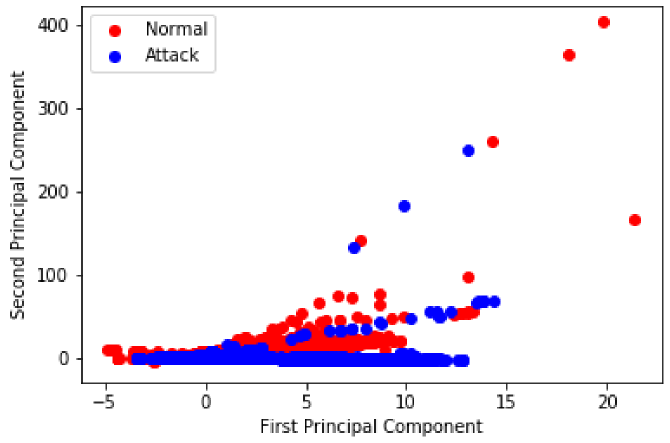


Figure 7.2: Principal Component Analysis of CICIDS 2017 Dataset

7.5.2 UNSW-NB 2015

The second dataset considered is the University of New South Wales's network intrusion dataset (UNSW-NB 2015) generated in 2015 [20]. The dataset is a hybrid of real modern network normal activities and synthetic attack behaviors [20]. The data was collected through two different simulations conducted on two different days, namely January 22 and February 17, 2015. The testbed used to generate the dataset consisted of all the necessary equipment such as routers, computers, servers, and firewalls. Interested readers are encouraged to review the detailed description provided in [20]. The resulting dataset consists of **2,540,044 instances** and **49 features** (1 class feature and 48 statistical features) representing the different characteristics of a network traffic request such as source and destination details, duration, protocol used, and packet size [20]. These instances are labeled as follows: **2,218,761 normal instances** and **521,283 attack instances**. In this case, no merging of attacks was needed since the dataset was originally labeled in a binary fashion.

In a similar fashion, Fig. 7.3 shows the first and second principal components for the UNSW-NB 2015 dataset. Again, we can observe that the features are non-linear. However, it can be noticed that the level of intertwining between the two classes is lower. Accordingly, it is easier to separate between the two classes.

Note that there are other network intrusion detection datasets that can be studied such as the NSL KDD 99 dataset and the Kyoto 2006+ datasets. However, these datasets have already been extensively studied. Moreover, they are outdated and may not have recent attack patterns. In contrast, the two datasets considered in this work are more recent and have more attack patterns. As such, studying them will provide better equipped NIDSs that are trained to detect more attack types.

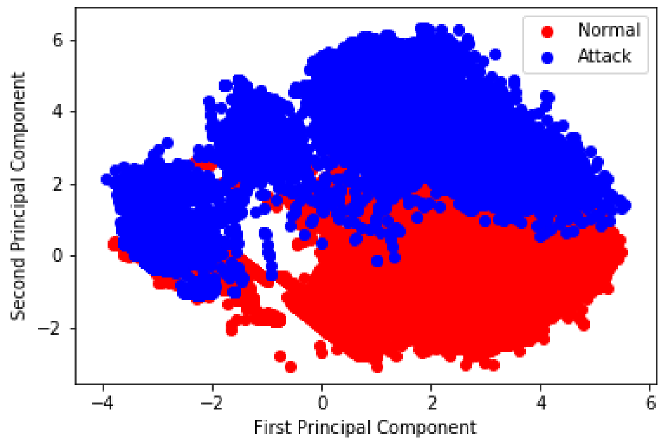


Figure 7.3: Principal Component Analysis of UNSW-NB 2015 Dataset

7.5.3 Attack Types

The two datasets considered in this work contain some similar attacks and some that are different. For example, the CICIDS 2017 dataset contains the following attacks: Denial-of-Service (DoS), port scanning, brute-force, web-attacks, botnets, and infiltration [19]. In contrast, the UNSW-NB 2015 dataset contains the following attacks: fuzzers, analysis, backdoors, DoS, exploits, generic, reconnaissance, shellcode, and worms [20]. Accordingly, it can be deduced that the proposed framework learns the patterns of various attack types.

Note that the proposed framework adopts a binary classification model by labeling all attack types as "attack". The goal is to develop a NIDS that can detect various attacks rather than just a finite group of common attacks such as DoS. This reiterates the idea that the proposed multi-stage optimized ML-based NIDS can work as an anomaly-based NIDS despite its training as a signature-based NIDS.

7.6 Experimental Performance Evaluation

7.6.1 Experimental Setup

The experiments conducted for this work were completed using **Python 3.7.4** running on Anaconda's Jupyter Notebook. This was run on a virtual machine having a 3 processors Intel (R) Xeon (R) CPU E5-2660 v3 2.6 GHz and 64GB of memory running Windows Server 2016. The experimental results are divided into three main subsections, namely the impact of data pre-processing on training sample size, impact of feature selection on feature set size and training sample size, and impact of optimization methods on the ML models' detection performance.

The classification models used in this work are KNN classifier and the RF classifier. These classifiers were chosen due to two main reasons. Firstly, these classifiers where the top performing classifiers in our previous work as they showed their effectiveness with network intrusion detection [107]. Secondly, these classifiers have lower computational complexities when compared to other classifiers. For example, the KNN classifier has a complexity of $O(MN)$ where M is the number of instances and N is the number of features [304, 305]. Similarly, the complexity of the RF classifier is $O(M^2\sqrt{N}t)$ where t is the number of trees within the RF classifier. However, since this classifier allows for multi-threading, its training time is significantly reduced to approximately $O(\frac{N^2\sqrt{M}t}{threads})$ where *threads* is the maximum number of participating threads [127]. In contrast, the complexity of SVM can reach an order of $O(M^3N)$ [309]. Therefore, training such a model would be computationally prohibitive, especially given the dataset sizes used in this work. Note that the parameters to be tuned are:

- KNN: number of neighbors K.
- RF: Splitting criterion (Gini or Entropy) and Number of trees.

It is worth noting that the runtime complexity of KNN and RF optimized models is $O(MN)$ and $O(Nt)$ respectively where M is the number of training samples, N is the number of features, and t is the number of decision trees forming the RF classifier [310, 311]. In the case of KNN, any new instance is classified after calculating the distance between itself and all other instances in the training sample and identifying its K nearest neighbors [310]. On the other hand, when using the RF classifier, the new instance is fed to the t different decision trees, each of which uses N splits based on the N features considered, and the class is determined based on the majority vote among these t trees.

7.6.2 Results and Discussion

Impact of data pre-processing on training sample size

Starting with the impact of data pre-processing stage on the training sample size, the learning curve showing the variation of training accuracy and the cross-validation accuracy as the training sample size changes. Both datasets were split randomly into training and testing samples after normalization using a 70%/30% split criterion.

Using the SMOTE technique, the number of instances of each type in each dataset's training sample is as follows:

- CICIDS 2017: 1,818,477 **benign** instances (denoted as 0) and 1,800,000 **attack** instances (denoted as 1).
- UNSW-NB 2015: 1,775,010 **normal** instances (denoted as 0) and 1,500,000 **attack** instances (denoted as 1).

It can be seen from Fig. 7.4 that the number of training samples needed for the CICIDS 2017 dataset for the training accuracy and cross-validation accuracy to converge is close to 2.3 million samples. Similarly, for the UNSW-NB 2015 dataset, the number of training samples needed is close to 1.3 million samples as can be seen from Fig. 7.5. This can be attributed to the fact that both datasets are originally imbalanced with much fewer attack samples when compared to normal samples. Hence, the model struggles to learn the attack patterns and behaviors. Note that the grey area represents the standard deviation between the learning accuracy and the cross-validation accuracy during the training stage.

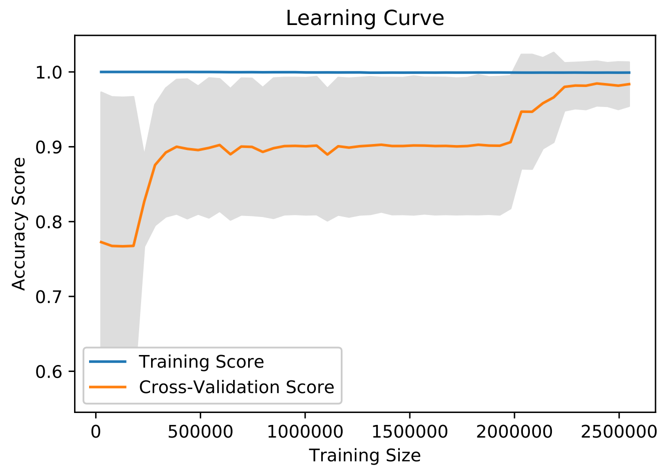


Figure 7.4: Learning Curve for CICIDS 2017 Dataset Before SMOTE

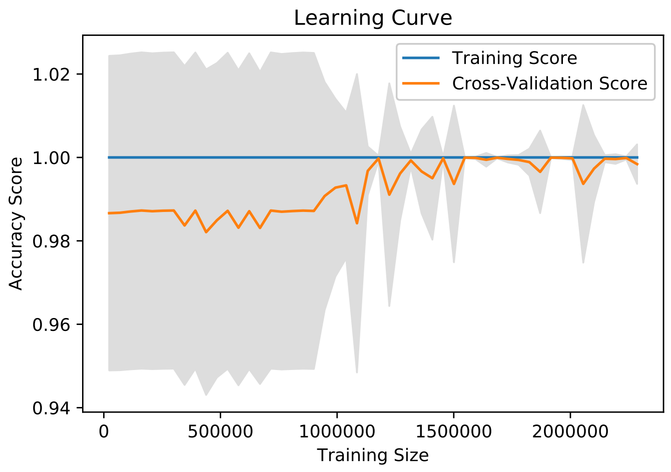


Figure 7.5: Learning Curve for UNSW-NB 2015 Dataset Before SMOTE

In contrast, it can be seen from Figs. 7.6 and 7.7 that the number of training samples needed is around 600,000 samples and 800,000 samples for the CICIDS 2017 and UNSW-NB 2015 respectively. This represents a drop of approximately 74% and 39% in the training sample size for the two datasets respectively. This highlights the positive impact of using SMOTE technique as it was able to significantly reduce the size of the training sample needed without sacrificing the detection performance. This is mainly due to the introduction of more attack samples that allow the ML model to better learn their patterns and behaviors. To further highlight the impact of using data pre-processing phase, the time needed to build the learning curve was determined. For example, building the learning curve for the UNSW-NB 2015 dataset needed close to 600 minutes prior to applying SMOTE. In contrast, it required around 90 minutes after implementing SMOTE. This highlights the time complexity reduction associated with adopting an oversampling technique.

Moreover, it can be seen from all these figures that the models developed before and after

SMOTE for both datasets do not suffer from overfitting as illustrated by the relatively small error gap between the training and cross-validation accuracy in Figs. 7.4 and 7.5 and the zero error gap seen in Figs. 7.6 and 7.7. As per [312], overfitting can be observed from the learning curve whenever the error gap between the training accuracy and the cross-validation accuracy is large. Thus, a small or zero error gap implies that the developed model is not too specific to the training dataset but can perform equally well on the testing and cross-validation sets.

Impact of feature selection on feature set size and training sample size

The second stage of analysis involves studying the impact of the different feature selection algorithms on the feature set size and training sample size.

1. *Impact of feature selection on feature set size:* Starting with the IGBFS method, Figs. 7.8 and 7.9 show the mutual information score for each of the features for the CICIDS 2017 and UNSW-NB 2015 datasets respectively. For example, for the CICIDS 2017 dataset, some of the most informative features include the average packet size and packet length variance. Similarly, for the UNSW-NB 2015 dataset, some of the most informative features are also the packet size (denoted by sbyte and dbyte features) and the time to live values. This illustrates the tendency of attacks to have different packet sizes when compared to normal traffic. Moreover, the figures also show that some IPs may have a higher tendency to initiate attacks, which means they are more likely to be compromised.
   Based on the figures, the number of features selected for the CICIDS 2017 and UNSW-NB 2015 datasets is 31 features and 19 features respectively. This represents a reduction of 62% and 61% in the feature set size for the two datasets respectively. This is due to the fact that the IGBFS method chooses the relevant features that provide the most information about the class.
   In contrast, when using the CBFS method, the number of features selected for the CICIDS 2017 and UNSW-NB 2015 datasets is 41 and 32 features respectively. This represents a reduction of 50% and 33.3% for each of the datasets respectively. This reduction

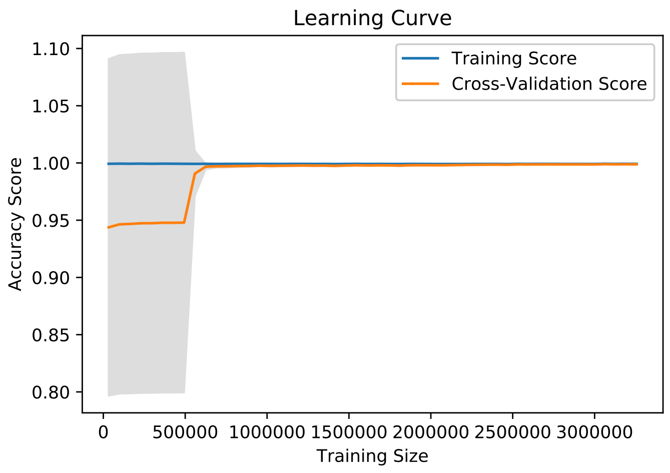


Figure 7.6: Learning Curve for CICIDS 2017 Dataset After SMOTE

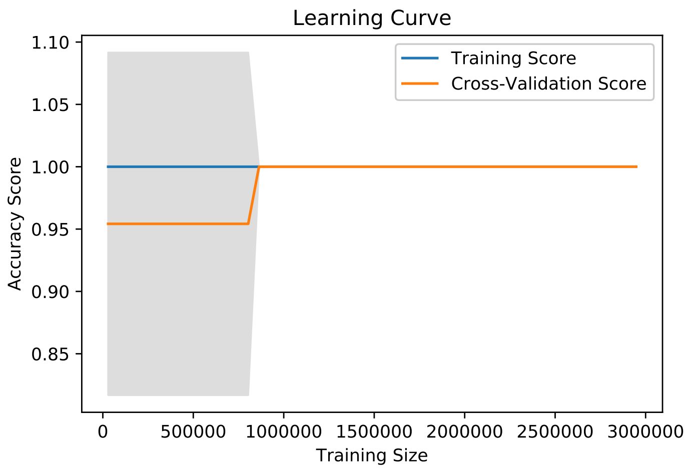


Figure 7.7: Learning Curve for UNSW-NB 2015 Dataset After SMOTE

is because the CBFS method chooses the relevant features that are highly correlated with the class feature, *i.e.* the features whose variation is also reflected in a variation in the corresponding class.

It is worth noting that the IGBFS method tends to choose a lower number of features when compared to the CBFS method. This is because the CBFS method relies on the correlation. Thus, two features may be chosen that are highly correlated with the class because they have a high correlation between them and one of them is highly correlated with the class. On the other hand, the IGBFS method studies the features one by one with respect to the class and selects the features that provide the highest amount of information about the class without considering the mutual information between the features themselves. Hence, a lower number of features is typically chosen by the IGBFS method.

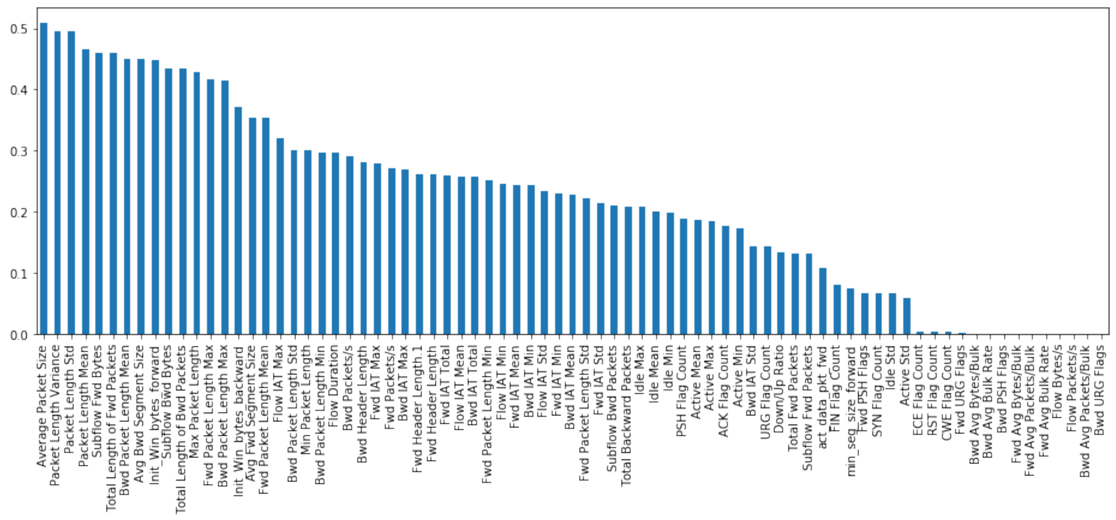


Figure 7.8: Mutual Information Score of Features for CICIDS 2017 Dataset

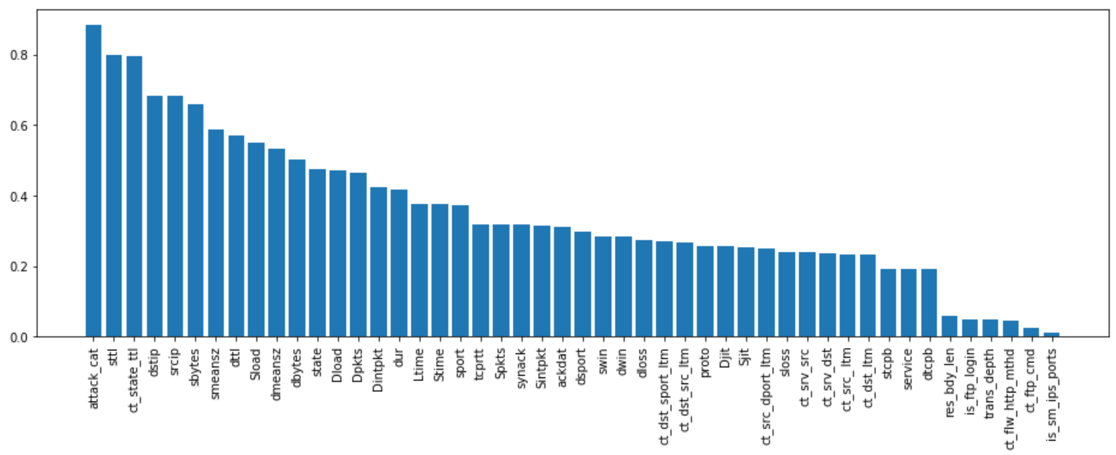


Figure 7.9: Mutual Information Score of Features for UNSW-NB 2015 Dataset

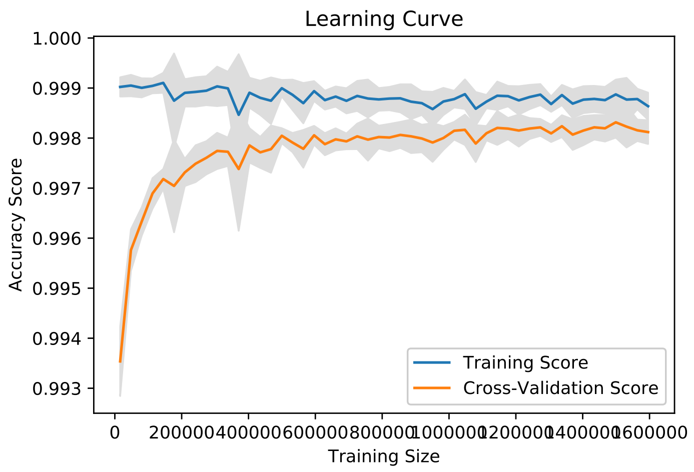


Figure 7.10: Learning Curve for CICIDS 2017 Dataset After IGBFS

2. *Impact of feature selection on training sample size:* In addition to the impact of the feature selection process on the feature set size, this work also studies its impact on the training sample size. Starting with the IGBFS method, it can be seen from Figs. 7.10 and 7.11 that the training sample size was reduced to 250,000 and 110,000 samples for the CICIDS 2017 and UNSW-NB 2015 datasets respectively. This represents a reduction of 59% and 86% when compared to the required training sample size after SMOTE technique is applied. This shows that the IGBFS method is able to keep the features that provide the most information about the class and discard any feature that may be negatively impacting the learning process.

Similarly for the case of using CBFS method, it can be observed from Figs. 7.12 and 7.13 that the required training sample size for the CICIDS 2017 and UNSW-NB 2015 datasets is reduced to 500,000 and 200,000 respectively. This represents a reduction of 17% and 75% when compared to the required training sample size after SMOTE technique is applied. This shows

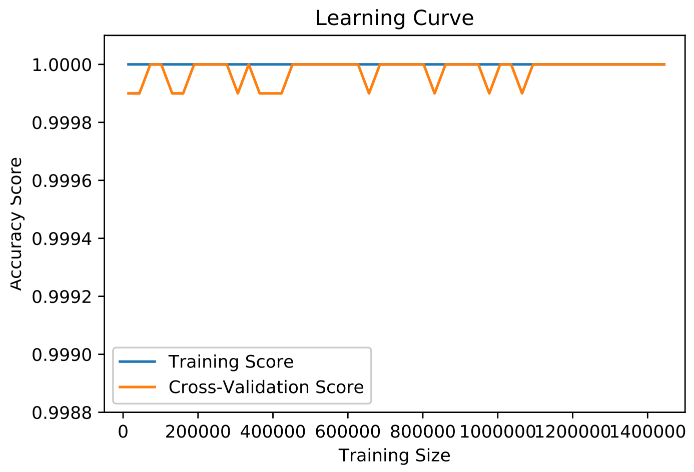


Figure 7.11: Learning Curve for UNSW-NB 2015 Dataset After IGBFS

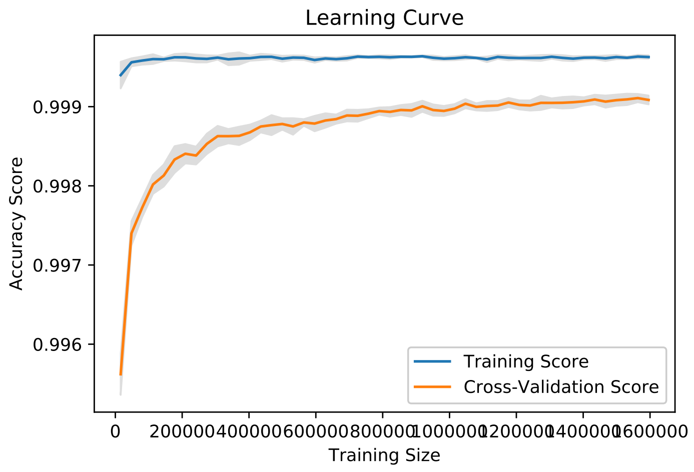


Figure 7.12: Learning Curve for CICIDS 2017 Dataset After CBFS

that the CBFS method is also able to select relevant features that have a positive impact on the learning process. However, due to the fact that some of the features selected may be redundant, this may be slightly impacting the learning process negatively when compared to that of the IGBFS. To further highlight the impact of the feature selection on the reduction of time complexity, the time needed to build the learning curve using the two feature selection methods was determined. For example, building the learning curve for the CICIDS 2017 dataset required between 60 minutes to 77 minutes for the CBFS and IGBFS respectively compared to almost 151 minutes without SMOTE and feature selection. Similarly, building the learning curve for the UNSW-NB 2015 dataset required around 21 minutes and 25 minutes for the IGBFS and CBFS methods respectively compared to almost 48 minutes without SMOTE and feature selection. Accordingly, applying either of the two feature selection methods will have a positive impact on the feature set size and training sample size with the IGBFS method having a slight advantage over the CBFS method.

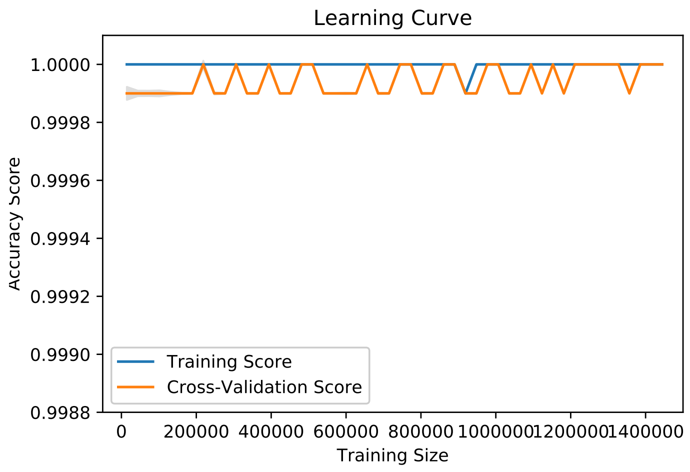


Figure 7.13: Learning Curve for UNSW-NB 2015 Dataset After CBFS

Figs. 7.10, 7.11, 7.12, and 7.13 describe a relatively small or zero error gap between the training accuracy and the cross-validation accuracy. This indicates that the model is suitable to be generalized for testing and cross-validation datasets and is not being overfit to the training dataset [312].

Impact of optimization methods on the ML models' detection performance

To evaluate the performance of the different classifiers and study the impact of the different optimization methods on them, we determine four evaluation metrics, namely the accuracy (acc), precision, recall, and false alarm rate (FAR) as per [107].

Table 7.1 gives the optimal parameter values for the two different classifiers when the IGBFS technique is used. In the case of KNN method, it is noticed that the RS and PSO methods tend to choose smaller values for the number of neighbors when compared to the GA, BO-GP, and BO-TPE methods. For the RS method, this can be attributed to the fact that the algorithm's stopping criterion is typically the number of iterations and does not test all potential values. Accordingly, it is possible for it to miss the optimal number of neighbors. Similarly, one of the stopping criterion in the PSO algorithm is also the number of evaluations which can also lead to it missing the optimal value. In contrast, the GA, BO-GP, and BO-TPE all resulted in a similar number of neighbors for both the CICIDS 2017 and UNSW-NB 2015 datasets. For the GA algorithm, the number of generations is typically set high enough for it to reach the optimal value for the number of neighbors. In a similar manner, the BO-GP and BO-TPE determine the actual optimal value based on the assumed model.

In the case of the RF method, again it is noticed that the RS and PSO algorithms tend to choose a lower number of trees as compared to the GA, BO-GP, and BO-TPE. As mentioned above, this is due to the algorithms' stopping criterion which often leads to a pre-mature stoppage. In contrast, the GA, BO-GP, and BO-TPE determine that the number of trees needed is higher as they explore more potential values, allowing them to select more optimal values for the number of trees. In terms of the splitting criterion, it noticed that the entropy criterion is mostly selected. This is expected given the fact that the IGBFS method selects features based on their information gain which is determined using the entropy of each feature. As such, this criterion would be more suitable when using IGBFS.

Looking at Table 7.2, similar observations about the hyper-parameter optimization performance of the different algorithms can be made for both the KNN and RF methods. The only difference is that for the RF method, the splitting criterion is chosen to be the Gini index. This is due to the fact that the CBFS method uses the correlation as the selection criterion rather than the entropy. Therefore, the features chosen might result in low amount of information (equivalent to having a high entropy with respect to the class) and thus would be overlooked if the entropy splitting criterion is chosen. That is why the Gini splitting criterion is chosen when

Table 7.1: Optimal Parameter Values with IGBFS

	CICIDS 2017	UNSW-NB 2015
Classifier	Parameter Values	Parameter Values
RS-KNN	Number of Neighbors= 3	Number of Neighbors= 11
PSO-KNN	Number of Neighbors= 5	Number of Neighbors= 11
GA-KNN	Number of Neighbors= 29	Number of Neighbors= 13
BO-GP-KNN	Number of Neighbors= 29	Number of Neighbors= 13
BO-TPE-KNN	Number of Neighbors= 29	Number of Neighbors= 13
RS-RF	Splitting Criterion= Gini, Number of trees= 40	Splitting Criterion= Entropy, Number of trees= 30
PSO-RF	Splitting Criterion= Gini, Number of trees= 21	Splitting Criterion= Entropy, Number of trees= 81
GA-RF	Splitting Criterion= Gini, Number of trees= 219	Splitting Criterion= Entropy, Number of trees= 168
BO-GP-RF	Splitting Criterion= Entropy, Number of trees= 200	Splitting Criterion= Entropy, Number of trees= 171
BO-TPE-RF	Splitting Criterion= Entropy, Number of trees= 90	Splitting Criterion= Entropy, Number of trees= 50

Table 7.2: Optimal Parameter Values with CBFS

	CICIDS 2017	UNSW-NB 2015
Classifier	Parameter Values	Parameter Values
RS-KNN	Number of Neighbors= 3	Number of Neighbors= 3
PSO-KNN	Number of Neighbors= 11	Number of Neighbors= 5
GA-KNN	Number of Neighbors= 25	Number of Neighbors= 25
BO-GP-KNN	Number of Neighbors= 29	Number of Neighbors= 25
BO-TPE-KNN	Number of Neighbors= 29	Number of Neighbors= 29
RS-RF	Splitting Criterion= Gini, Number of trees= 20	Splitting Criterion= Gini, Number of trees= 10
PSO-RF	Splitting Criterion= Gini, Number of trees= 87	Splitting Criterion= Gini, Number of trees= 54
GA-RF	Splitting Criterion= Gini, Number of trees= 219	Splitting Criterion= Gini, Number of trees= 219
BO-GP-RF	Splitting Criterion= Gini, Number of trees= 164	Splitting Criterion= Entropy, Number of trees= 78
BO-TPE-RF	Splitting Criterion= Entropy, Number of trees= 50	Splitting Criterion= Gini, Number of trees= 20

Table 7.3: Performance results of the Multi-stage Optimized ML-based NIDS Framework with IGBFS

	CICIDS 2017				UNSW-NB 2015			
Classifier	Acc(%)	Precision	Recall	FAR	Acc(%)	Precision	Recall	FAR
RS-KNN	99.63%	0.99	0.99	0.001	99.96%	0.99	0.99	0.001
PSO-KNN	99.09%	0.98	0.99	0.001	99.91%	0.99	0.99	0.001
GA-KNN	99.09%	0.98	0.99	0.001	99.91%	0.99	0.99	0.001
BO-GP-KNN	99.11%	0.98	0.99	0.001	99.91%	0.99	0.99	0.001
BO-TPE-KNN	99.11%	0.98	0.99	0.001	99.91%	0.99	0.99	0.001
RS-RF	99.72%	0.99	0.99	0.001	100%	1.0	1.0	0.0
PSO-RF	99.98%	0.99	0.99	0.001	100%	1.0	1.0	0.0
GA-RF	99.98%	0.99	0.99	0.001	100%	1.0	1.0	0.0
BO-GP-RF	99.83%	0.99	0.99	0.001	100%	1.0	1.0	0.0
BO-TPE-RF	99.99%	0.99	0.99	0.001	100%	1.0	1.0	0.0

CBFS method is used.

Tables 7.3 and 7.4 show the performance of the two classification algorithms when using IGBFS and CBFS methods, respectively. Several observations can be made. The first observation is that the optimized models outperform the regular models reported in [19][127][313] by 1-2% on average in terms of accuracy and a reduction of 1-2% in FAR for both datasets. For example, the authors in [19] had an average detection accuracy of 96%-98% while the authors in [127] achieved an average accuracy between 96%-99% for the CICIDS 2017 dataset. Similarly, the authors in [313] achieved an accuracy between 96%-98% on the UNSW-NB 2015 dataset. This is expected since one of the main goals of hyper-parameter optimization is to improve the performance of the ML models. The second observation is that the RF classifier outperforms the KNN classifier for both the IGBFS and CBFS methods as seen in the CICIDS 2017 and UNSW-NB 2015 datasets. This reiterates the previously obtained results in [107] with ISCX 2012 dataset and the reported results in [19][127][313] in which the RF classifier also outperformed the KNN model. This can be attributed to the RF classifier being an ensemble model. Accordingly, it is effective with non-linear and high-dimensional datasets like the datasets under consideration in this work. The third observation is that the BO-TPE-RF method had the highest detection accuracy for both the CICIDS 2017 and UNSW-NB 2015 datasets for both feature selection algorithms with a detection accuracy of 99.99% and 100%, respectively. This proves the effectiveness and robustness of the proposed multi-stage optimized ML-based NIDS framework as it outperformed other NIDS frameworks while using a significantly reduced dataset size (reduced the training sample size by 74% and reduced the feature set size by 60%).

7.7 Conclusion

With the increased dependency of individuals and organizations on the Internet and their concern about the security and privacy of their activities, the area of cyber-security has garnered significant attention from both the industry and academia. To that end, more resource are being deployed and allocated to protect modern Internet-based networks to protect them against potential attacks or anomalous activities. Accordingly, different types of network intru-

Table 7.4: Performance results of the Multi-stage Optimized ML-based NIDS Framework with CBFS

	CICIDS 2017				UNSW-NB 2015			
Classifier	Acc(%)	Precision	Recall	FAR	Acc(%)	Precision	Recall	FAR
RS-KNN	99.70%	0.99	0.99	0.001	99.96%	0.99	0.99	0.001
PSO-KNN	99.28%	0.99	0.99	0.001	99.88%	0.99	0.99	0.001
GA-KNN	99.28%	0.99	0.99	0.001	99.88%	0.99	0.99	0.001
BO-GP-KNN	99.23%	0.99	0.99	0.001	99.88%	0.99	0.99	0.001
BO-TPE-KNN	99.23%	0.99	0.99	0.001	99.88%	0.99	0.99	0.001
RS-RF	99.61%	0.99	0.99	0.001	100%	1.0	1.0	0.0
PSO-RF	99.88%	0.99	0.99	0.001	100%	1.0	1.0	0.0
GA-RF	99.88%	0.99	0.99	0.001	100%	1.0	1.0	0.0
BO-GP-RF	99.88%	0.99	0.99	0.001	100%	1.0	1.0	0.0
BO-TPE-RF	99.99%	0.99	0.99	0.001	100%	1.0	1.0	0.0

sion detection systems (NIDSs) have been proposed in the literature. Despite the continuous improvements in NIDS performance, there still remains room for further enhancements. More specifically, more information can be learned from the high volume of network traffic data generated, the continuously changing environments, the plethora of features collected as part of training datasets (high dimensional datasets), and the need for real-time intrusion detection.

Based on the aforementioned information, it can be concluded that choosing the most suitable subset of features and optimizing the parameters of the machine learning (ML)-based detection models is essential to enhance their performance. Accordingly, this chapter extended our previous work by proposing a novel multi-stage optimized ML-based NIDS framework that reduced the computational complexity while maintaining its detection performance. To achieve that, and using two recent state-of-the-art intrusion detection datasets (CICIDS 2017 dataset and the UNSW-NB 2015 dataset) for performance evaluation, this work first studied the impact of oversampling techniques on the models' training sample size and determined the minimum suitable training size for effective intrusion detection. Experimental results showed that using the SMOTE oversampling technique can reduce the training sample size between 39% and 74% of the original datasets' size. In addition to that, this work then compared between two different feature selection techniques, namely information gain (IGBFS) and correlation-based feature selection (CBFS), and explored their impact on the feature set size, the training sample size, and the models' detection performance. The experimental results showed that the feature selection methods were able to reduce the feature set size by almost 60%. Moreover, they further reduced the required training sample size between 33% and 50% when compared to the training sample after SMOTE. Finally, this work investigated the impact of different ML hyper-parameter optimization techniques on the NIDS's performance using two ML classification models, namely the K-nearest neighbors (KNN) and the Random Forest (RF) classifiers. Experimental results showed that the optimized RF classifier with Bayesian Optimization using Tree Parzen Estimator (BO-TPE-RF) had the highest detection accuracy when compared to the other optimization techniques. Additionally, it was also observed that using the IGBFS method achieved better detection accuracy when compared to the CBFS method. Furthermore, it is seen that the optimized models outperform the regular models reported in [19][127][313] by 1-2% on average in terms of accuracy and a reduction of 1-2% in FAR for both datasets while using a significantly reduced dataset size (reduced the training sample size by 74% and reduced the feature set size by 60%).

Chapter 8

Conclusion and Future Research Directions

8.1 Introduction

The rapid growth of the Internet and related technologies has led to the collection of large amounts of data by individuals, organizations, and society in general [1]. However, these large amounts of data often lead to information overload which occurs when the amount of input (e.g. data) that a human is trying to process exceeds their cognitive capacities [2]. In turn, this can lead to humans ignoring, overlooking, or misinterpreting crucial information [7]. To address this issue, the discipline of data science has emerged. Data science combines the classic disciplines of statistics, data mining, databases, and distributed systems in order to extract useful information from large sets of data [1]. Among the different data analysis methods that data scientists can implement is machine learning (ML). ML allows computers to learn without being explicitly programmed. Upon learning patterns from a training set of data, the computer can apply what it has learned to find these patterns in similar data [8]. Furthermore, ML allows computer systems to adapt and learn from their experience [9, 10]. With ML models, organizations can continually predict changes in their business and make data-driven decisions accordingly. ML uses algorithms that iteratively learn from data to improve, describe data, and predict outcomes. Once an ML model has been trained, it can predict new data that is given as input. The output given by the model on the new data will depend on the data used to train the model.

ML algorithms have several applications. This includes house pricing prediction, spam filtering, education, structuring of data in healthcare systems, drug response prediction, diabetes research, network security, banking and finance, and social media. This thesis focused on two of the aforementioned applications. The first is education, namely e-Learning environments. Within this field, this thesis proposed the use of different optimized ML models to predict students' performance at earlier stages of the course delivery. The developed models use different ensemble classification techniques to categorize the students and predict their final performance group. The second application is network security intrusion detection. Within this application field, this thesis proposed different optimized ML classification frameworks using a variety of optimization modeling algorithms and heuristics to improve the performance of the IDSs through anomaly detection while maintaining or reducing their time complexity.

The remainder of this chapter briefly summarizes the contributions within the thesis and presents some future research directions worth exploring.

8.2 Summary of Contributions

Chapter 3 briefly described the different challenges facing a variety of modern fields including education, healthcare, network security, banking and finance, and social media. Moreover, it presented some previous works that focused on these challenges and their shortcomings. Furthermore, it discussed the role and potential of ML in addressing these challenges and presented potential frameworks for its deployment.

Chapter 4 first analyzed the two educational datasets under consideration using multiple graphical, statistical, and quantitative techniques (e.g. probability density function, decision boundaries, feature variance, feature weights, principal component analysis,etc.). It then conducted hyper-parameter tuning using grid search algorithm to optimize the parameters of the different ML models investigated. This was followed by a systematic multi-split based approach (to eliminate bias) for building a majority voting ensemble learner to choose the best model for student performance prediction based on multiple metrics, namely the Gini index and the p-value. The performance of the proposed approach was evaluated in comparison to traditional ML classification techniques. Experimental results showed that the proposed optimized ML ensemble models accurately identified the weak students who needed help.

Chapter 5 extended the previous work by proposing a systemic approach to build a multi-split-based (to reduce bias) bagging ensemble (to reduce variance) learner to choose the best model based on multiple performance metrics, namely the Gini index (for better statistical significance and robustness) and the target class score. It studied the performance of the proposed ensemble learning classification model on a multi-class dataset in comparison with the previously developed binary classification model. Similar to the previous chapter, the experiments showed that the proposed model accurately detected the target class despite slightly suffering in identifying the fair students due to the fact that they were at the border between the good and the weak students.

Chapter 6 investigated the performance of the optimized ML models using Bayesian Optimization to detect network intrusion anomalies. The proposed optimization method enhanced the performance of the classification models through the identification of the optimal parameters towards objective-function minimization. The performance of the optimized ML models was evaluated using UNB ISCX 2012, a benchmark intrusion dataset commonly used for experimentation and validation purposes. The experimental results showed that not only is the proposed optimization method more accurate in detecting intrusions, but it was also successful in finding the global minimum of the objective function which led to better classification results.

Chapter 7 extended the work from the previous chapter by studying the impact of oversampling techniques and determining the minimum suitable training sample size for effective intrusion detection. It also explored the impact of different feature selection techniques on the NIDS detection performance and time complexity. Additionally, it investigated different ML hyper-parameter optimization techniques and their corresponding enhancement of the NIDS detection performance. Finally, it evaluated the performance of the optimized ML-based NIDS framework using two recent state-of-the-art datasets, namely the CICIDS 2017 dataset and the UNSW-NB 2015 dataset. Experimental results showed that using the SMOTE oversampling technique reduced the training sample size between 39% and 74% of the original datasets' size. Furthermore, the feature selection methods were able to reduce the feature set size by almost 60%. Also, they further reduced the required training sample size between 33% and 50% when compared to the training sample after SMOTE. Finally, experimental results showed that the optimized RF classifier with Bayesian Optimization using Tree Parzen Estimator (BO-TPE-RF) had the highest detection accuracy when compared to the other optimization techniques. Additionally, it was also observed that using the information gain feature selection method achieved better detection accuracy when compared to the correlation-based feature selection method. Furthermore, it is seen that the optimized models outperform the regular models reported in [19][127][313] by 1-2% on average in terms of accuracy and a reduction of 1-2% in FAR for both datasets while using a significantly reduced dataset size (reduced the training sample size by 74% and reduced the feature set size by 60%).

Note that the developed models and frameworks in this thesis could be generalized to multiple application fields. More specifically:

- The optimized ML ensemble models proposed in chapters 4 and 5 of this thesis can be generalized to applications such as finance, network security, social media, and healthcare systems.

- The optimized ML classification models proposed in chapters 6 and 7 of this thesis can be generalized to other applications that typically generate large datasets in terms of instances and feature set such as wireless sensor networks and autonomous vehicles.

8.3 Future Research Directions

This thesis explored the use of optimized ML models to introduce intelligence and improve the performance of two different systems in two application fields. Despite the innovative solutions and frameworks developed in this thesis, there are many future research directions worth exploring. The subsequent subsections discuss some of these future research directions, particularly in the fields of education and network security, to achieve more effective, robust, and intelligent systems.

8.3.1 Research Opportunities Towards Intelligent e-Learning Environments

The work presented in Chapters 4 and 5 can be extended in many different directions. One potential extension is collecting further datasets with more features to overcome the challenge of the scarcity of educational datasets. Furthermore, more sophisticated and optimized ML models need to be trained using these datasets to further improve their ability to identify weak students during the course delivery duration. Another potential extension worth exploring is studying the impact of language and usage specific features as well as different kernel functions, particularly on the performance of essay grading frameworks. A third extension is investigating more phonemic and history-based features as well as more ML models and studying the resulting performance enhancement for intelligent tutoring frameworks.

8.3.2 Research Opportunities Towards More Effective Network Security Frameworks

The work presented in Chapters 6 and 7 can be extended in many different directions. One research direction worth exploring is developing hybrid IDS systems that combine both supervised and unsupervised ML models to detect known and unknown attack patterns. This would help improve the performance and robustness of the IDS further by flagging previously unseen attack patterns as anomalous and learning from them for identifying future attacks. Additionally, more sophisticated models such as deep learning classifiers and ensemble models should be considered given that they have proved to perform admirably on non-linear and high-dimensional datasets. Moreover, the performance of different time-series techniques should be studied in an attempt to identify and detect temporal-based anomalies and intrusion attempts.

Bibliography

[1] W. Van Der Aalst, “Data science in action,” in *Process mining*. Springer, 2016, pp. 3–23.

[2] G. S. Halford, R. Baker, J. E. McCredden, and J. D. Bain, “How many variables can humans process?” *Psychological science*, vol. 16, no. 1, pp. 70–76, 2005.

[3] L. Columbus, “Roundup of machine learning forecasts and market estimates, 2020,” *Forbes*, 2020.

[4] T. D. Snyder, C. de Brey, and S. A. Dillow, “Digest of education statistics 2018,” *National Center for Education Statistics*, 2019.

[5] M. Powell, “11 eye opening cyber security statistics for 2019,” *CPO Magazine*, 2019.

[6] K. Xie, “Four key challenges for cybersecurity leaders.” World Economic Forum, 2020.

[7] J. J. Caban and D. Gotz, “Visual analytics in healthcare–opportunities and research challenges,” 2015.

[8] M. J. Kearns, U. V. Vazirani, and U. Vazirani, *An introduction to computational learning theory*. MIT press, 1994.

[9] R. A. Wilson and F. C. Keil, *The MIT encyclopedia of the cognitive sciences*. MIT press, 2001.

[10] T. M. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, 1997.

[11] J. Bughin, J. Seong, J. Manyika, L. Hämäläinen, E. Windhagen, and E. Hazan, “Notes from the ai frontier: Tackling europe?s gap in digital and ai,” *McKinsey&Company: New York, NY, USA*, 2019.

[12] W. E. Forum, “The future of jobs report 2018.” World Economic Forum Geneva, 2018.

[13] A. Moubayed, M. Injadat, A. B. Nassif, H. Lutfiyya, and A. Shami, “E-learning: Challenges and research opportunities using machine learning data analytics,” *IEEE Access*, vol. 6, pp. 39 117–39 138, 2018.

[14] G. Black, “A comparison of traditional, online, and hybrid methods of course delivery,” *Journal of Business Administration Online*, vol. 1, no. 1, pp. 1–9, 2002.

[15] Cisco, “What is network security?” 2019, Accessed on: Feb. 1, 2020. [Online]. Available: https://www.cisco.com/c/en/us/products/security/what-is-network-security.html

[16] A. Moubayed, M. Injadat, A. Shami, and H. Lutfiyya, “Dns typo-squatting domain detection: A data analytics & machine learning based approach,” in *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2018, pp. 1–7.

[17] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, “A deep learning approach for network intrusion detection system,” in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, 2016, pp. 21–26.

[18] R. Sommer and V. Paxson, “Outside the closed world: On using machine learning for network intrusion detection,” in *2010 IEEE symposium on security and privacy*. IEEE, 2010, pp. 305–316.

[19] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization.” in *ICISSP*, 2018, pp. 108–116.

[20] N. Moustafa and J. Slay, “Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set),” in *2015 Military Communications and Information Systems Conference (MilCIS)*, Nov 2015, pp. 1–6.

[21] A. L. Samuel, “Some studies in machine learning using the game of checkers,” *IBM Journal of research and development*, vol. 3, no. 3, pp. 210–229, 1959.

[22] A. Ahmed and I. S. Elaraby, “Data mining: A prediction for student’s performance using classification method,” *World Journal of Computer Application and Technology*, vol. 2, no. 2, pp. 43–47, 2014.

[23] A. B. Nassif, D. Ho, and L. F. Capretz, “Regression model for software effort estimation based on the use case point method,” in *2011 International Conference on Computer and Software Modeling*, vol. 14, 2011, pp. 106–110.

[24] G. A. Korn and T. M. Korn, *Mathematical handbook for scientists and engineers: definitions, theorems, and formulas for reference and review*. Courier Corporation, 2000.

[25] D. Bachman, *Advanced calculus demystified*. McGrawHill, 2007.

[26] Z. Kolter, “Linear algebra review and reference,” *Available online: http://backspaces.net/temp/ML/CS229Math.pdf*, 2008.

[27] S. L. Algebra, “Graduate texts in mathematics,” New York, USA, pp. 572–585, 2002.

[28] A. Ng., “Cs229 lecture notes,” 2012. [Online]. Available: http://cs229.stanford.edu/notes/

[29] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[30] M. Mohri, A. Rostamizadeh, and A. Talwalkar, "Foundations of machine learning. chapter 8," 2012.

[31] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.

[32] R. I. Lerman and S. Yitzhaki, "A note on the calculation and interpretation of the gini index," *Economics Letters*, vol. 15, no. 3-4, pp. 363–368, 1984.

[33] M. O. Lorenz, "Methods of measuring the concentration of wealth," *Publications of the American statistical association*, vol. 9, no. 70, pp. 209–219, 1905.

[34] D. L. Olson, D. Delen, and Y. Meng, "Comparative analysis of data mining methods for bankruptcy prediction," *Decision Support Systems*, vol. 52, no. 2, pp. 464–473, 2012.

[35] T.-S. Lee, C.-C. Chiu, Y.-C. Chou, and C.-J. Lu, "Mining the customer credit using classification and regression tree and multivariate adaptive regression splines," *Computational Statistics & Data Analysis*, vol. 50, no. 4, pp. 1113–1130, 2006.

[36] C. Bielza, P. Barreiro, M. Rodrıguez-Galiano, and J. Martın, "Logistic regression for simulating damage occurrence on a fruit grading line," *Computers and Electronics in Agriculture*, vol. 39, no. 2, pp. 95–113, 2003.

[37] A. S. Al-Ghamdi, "Using logistic regression to estimate the influence of accident factors on accident severity," *Accident Analysis & Prevention*, vol. 34, no. 6, pp. 729–741, 2002.

[38] Y. Shi, W. Qian, W. Yan, and J. Li, "Adaptive depth control for autonomous underwater vehicles based on feedforward neural networks," in *Intelligent Control and Automation*. Springer, 2006, pp. 207–218.

[39] N. Cristianini, J. Shawe-Taylor *et al.*, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[40] M. Injadat, F. Salo, and A. B. Nassif, "Data mining techniques in social media: A survey," *Neurocomputing*, vol. 214, pp. 654 – 670, 2016.

[41] M. Kiran, A. Kumar, S. Mukherjee, and R. G. Prakash, "Verification and validation of mapreduce program model for parallel support vector machine algorithm on hadoop cluster," *International Journal of Computer Science Issues (IJCSI)*, vol. 10, no. 3, p. 317, 2013.

[42] A. Zell, *Simulation neuronaler netze*. Addison-Wesley Bonn, 1994, vol. 1, no. 5.3.

[43] A. B. Nassif, L. F. Capretz, and D. Ho, "Estimating software effort using an ann model based on use case points," in *2012 11th International Conference on Machine Learning and Applications*, vol. 2, 2012, pp. 42–47.

[44] W. I. D. Mining, "Data mining: Concepts and techniques," *Morgan Kaufinann*, vol. 10, pp. 559–569, 2006.

[45] D. Kabakchieva, “Predicting student performance by using data mining methods for classification,” *Cybernetics and information technologies*, vol. 13, no. 1, pp. 61–72, 2013.

[46] L. Jiang, Z. Cai, D. Wang, and S. Jiang, “Survey of improving k-nearest-neighbor for classification,” in *Fourth international conference on fuzzy systems and knowledge discovery (FSKD 2007)*, vol. 1. IEEE, 2007, pp. 679–683.

[47] M. Steinbach and P.-N. Tan, “knn: k-nearest neighbors,” in *The top ten algorithms in data mining*. Chapman and Hall/CRC, 2009, pp. 165–176.

[48] W. Zheng, H. Wang, L. Ma, and R. Wang, “An improved k-nearest neighbor classification algorithm using shared nearest neighbor similarity.” *Metallurgical & Mining Industry*, no. 10, 2015.

[49] F. V. Jensen and T. D. Nielsen, “Probabilistic decision graphs for optimization under uncertainty,” *Annals of Operations Research*, vol. 204, no. 1, pp. 223–248, 2013.

[50] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[51] P. Bhargavi and S. Jyothi, “Applying naive bayes data mining technique for classification of agricultural land soils,” *International journal of computer science and network security*, vol. 9, no. 8, pp. 117–122, 2009.

[52] A. Liaw, M. Wiener *et al.*, “Classification and regression by randomforest,” *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[53] D. Opitz and R. Maclin, “Popular ensemble methods: An empirical study,” *Journal of artificial intelligence research*, vol. 11, pp. 169–198, 1999.

[54] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*, 1st ed. Chapman & Hall, 2012.

[55] L. Rokach, “Ensemble-based classifiers,” *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 1–39, 2010.

[56] D. H. Wolpert, “Stacked generalization,” *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.

[57] S. Dzeroski and B. Zenko, “Is combining classifiers better than selecting the best one?” in *ICML*, vol. 2002. Citeseer, 2002, p. 123e30.

[58] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, “Machine Learning Towards Intelligent Systems: Applications, Challenges, and Opportunities,” *Submitted to Artificial Intelligence Review*, 2020.

[59] T. K. Landauer, “Automatic essay assessment,” *Assessment in education: Principles, policy & practice*, vol. 10, no. 3, pp. 295–308, 2003.

[60] M. Mahana, M. Johns, and A. Apte, "Automated essay grading using machine learning," *Mach. Learn. Session, Stanford University*, 2012.

[61] V. Ramalingam, A. Pandian, P. Chetry, and H. Nigam, "Automated essay grading using machine learning algorithm," in *Journal of Physics: Conference Series*, vol. 1000, no. 1. IOP Publishing, 2018, p. 012030.

[62] M. Herbert, "Staying the course: A study in online student satisfaction and retention," *Online Journal of Distance Learning Administration*, vol. 9, no. 4, pp. 300–317, 2006.

[63] I. Lykourentzou, I. Giannoukos, V. Nikolopoulos, G. Mpardis, and V. Loumos, "Dropout prediction in e-learning courses through the combination of machine learning techniques," *Computers & Education*, vol. 53, no. 3, pp. 950–965, 2009.

[64] K. Blom, D. Meyers *et al.*, *Quality indicators in vocational education and training: International perspectives*. National Centre for Vocational Education Research, 2003.

[65] P. Bawa, "Retention in online courses: Exploring issues and solutions a literature review," *Sage Open*, vol. 6, no. 1, 2016.

[66] S. B. Kotsiantis, C. Pierrakeas, and P. E. Pintelas, "Preventing student dropout in distance learning using machine learning techniques," in *International conference on knowledge-based and intelligent information and engineering systems*. Springer, 2003, pp. 267–274.

[67] J. E. Beck, P. Jia, J. Sison, and J. Mostow, "Predicting student help-request behavior in an intelligent tutor for reading," in *International Conference on User Modeling*. Springer, 2003, pp. 303–312.

[68] Y.-C. Tam, J. Mostow, J. E. Beck, and S. Banerjee, "Training a confidence measure for a reading tutor that listens," in *Eighth European Conference on Speech Communication and Technology*, 2003.

[69] J. Mostow *et al.*, "Evaluating tutors that listen: An overview of project listen." 2001.

[70] Mooc.org, "Massive open online courses: An edx site," 2019, Accessed on: Dec. 15, 2019. [Online]. Available: https://www.mooc.org/

[71] P. Symeonidis and D. Malakoudis, "Moocrec. com: Massive open online courses recommender system." in *RecSys Posters*, 2016.

[72] S. B. Aher and L. Lobo, "Combination of machine learning algorithms for recommendation of courses in e-learning system based on historical data," *Knowledge-Based Systems*, vol. 51, pp. 1–14, 2013.

[73] A. Klašnja-Milićević, B. Vesin, M. Ivanović, and Z. Budimac, "E-learning personalization based on hybrid recommendation strategy and learning style identification," *Computers & Education*, vol. 56, no. 3, pp. 885–899, 2011.

[74] P. Dwivedi and K. K. Bharadwaj, "e-learning recommender system for a group of learners based on the unified learner profile approach," *Expert Systems*, vol. 32, no. 2, pp. 264–276, 2015.

[75] O. Bourkoukou and E. El Bachari, "E-learning personalization based on collaborative filtering and learner's preference," *Journal of Engineering Science and Technology*, vol. 11, no. 11, pp. 1565–1581, 2016.

[76] J. C. Reiff, "Learning styles. what research says to the teacher series." 1992.

[77] O. Bourkoukou and E. El Bachari, "Toward a hybrid recommender system for e-learning personnalization based on data mining techniques," *JOIV: International Journal on Informatics Visualization*, vol. 2, no. 4, pp. 271–278, 2018.

[78] R. M. Felder, L. K. Silverman *et al.*, "Learning and teaching styles in engineering education," *Engineering education*, vol. 78, no. 7, pp. 674–681, 1988.

[79] A. O. Elfaki, K. M. Alhawiti, Y. M. AlMurtadha, O. A. Abdalla, and A. A. Elshiekh, "Rule-based recommendation for supporting student learning-pathway selection," *Recent Advances in Electrical Engineering and Educational Technologies*, pp. 155–160, 2014.

[80] C. F. Lin, Y.-c. Yeh, Y. H. Hung, and R. I. Chang, "Data mining for providing a personalized learning path in creativity: An application of decision trees," *Computers & Education*, vol. 68, pp. 199–210, 2013.

[81] A. Moubayed, M. Injadat, A. Shami, and H. Lutfiyya, "Relationship between student engagement and performance in e learning environment using association rules," in *2018 IEEE World Engineering Education Conference (EDUNINE)*, March 2018, pp. 1–6.

[82] A. Moubayed, M. Injadat, A. Shami, and H. Lutfiyya, "Student engagement level in e learning environment: Clustering using k-means," *American Journal of Distance Education*, vol. 34, no. 2, 2019.

[83] C. Xiao, E. Choi, and J. Sun, "Opportunities and challenges in developing deep learning models using electronic health records data: a systematic review," *Journal of the American Medical Informatics Association*, vol. 25, no. 10, pp. 1419–1428, 2018.

[84] R. Miotto, F. Wang, S. Wang, X. Jiang, and J. T. Dudley, "Deep learning for healthcare: review, opportunities and challenges," *Briefings in bioinformatics*, vol. 19, no. 6, pp. 1236–1246, 2018.

[85] A. Holzinger, M. Dehmer, and I. Jurisica, "Knowledge discovery and interactive data mining in bioinformatics-state-of-the-art, future challenges and research directions," *BMC bioinformatics*, vol. 15, no. 6, p. I1, 2014.

[86] M. Vidyasagar, "Identifying predictive features in drug response using machine learning: opportunities and challenges," *Annual review of pharmacology and toxicology*, vol. 55, pp. 15–34, 2015.

[87] C. Huang, E. A. Clayton, L. V. Matyunina, L. D. McDonald, B. B. Benigno, F. Vannberg, and J. F. McDonald, “Machine learning predicts individual cancer patient responses to therapeutic drugs with high accuracy,” *Scientific reports*, vol. 8, no. 1, pp. 1–8, 2018.

[88] V. Prasad, T. Fojo, and M. Brada, “Precision oncology: origins, optimism, and potential,” *The Lancet Oncology*, vol. 17, no. 2, pp. e81–e86, 2016.

[89] F. Xia, M. Shukla, T. Brettin, C. Garcia-Cardona, J. Cohn, J. E. Allen, S. Maslov, S. L. Holbeck, J. H. Doroshow, Y. A. Evrard *et al.*, “Predicting tumor cell line response to drug pairs with deep learning,” *BMC bioinformatics*, vol. 19, no. 18, pp. 71–79, 2018.

[90] Y.-C. Chiu, H.-I. H. Chen, T. Zhang, S. Zhang, A. Gorthi, L.-J. Wang, Y. Huang, and Y. Chen, “Predicting drug response of tumors from integrated genomic profiles by deep neural networks,” *BMC medical genomics*, vol. 12, no. 1, p. 18, 2019.

[91] E. J. Mucaki, J. Z. Zhao, D. J. Lizotte, and P. K. Rogan, “Predicting responses to platin chemotherapy agents with biochemically-inspired machine learning,” *Signal transduction and targeted therapy*, vol. 4, no. 1, pp. 1–12, 2019.

[92] I. Kavakiotis, O. Tsave, A. Salifoglou, N. Maglaveras, I. Vlahavas, and I. Chouvarda, “Machine learning and data mining methods in diabetes research,” *Computational and structural biotechnology journal*, vol. 15, pp. 104–116, 2017.

[93] Michael Dansinger, “What is a glycated hemoglobin test (hba1c)?” 2019. [Online]. Available: https://www.webmd.com/diabetes/qa/what-is-a-glycated-hemoglobin-test-hba1c

[94] H. F. Jelinek, A. Stranieri, A. Yatsko, and S. Venkatraman, “Data analytics identify glycated haemoglobin co-markers for type 2 diabetes mellitus diagnosis,” *Computers in biology and medicine*, vol. 75, pp. 90–97, 2016.

[95] P. Herrero, P. Pesl, M. Reddy, N. Oliver, P. Georgiou, and C. Toumazou, “Advanced insulin bolus advisor based on run-to-run control and case-based reasoning,” *IEEE journal of biomedical and health informatics*, vol. 19, no. 3, pp. 1087–1096, 2014.

[96] A. Schiffrin and M. Belmonte, “Multiple daily self-glucose monitoring: its essential role in long-term glucose control in insulin-dependent diabetic patients treated with pump and multiple subcutaneous injections.” *Diabetes Care*, vol. 5, no. 5, pp. 479–484, 1982.

[97] C. C. A. Flyvbjerg, G. Holt and B. Goldstein, *Textbook of Diabetes: A Clinical Approach*, 4th ed. New Jersey, USA: Wiley, 2010.

[98] L. Jovanovic and C. M. Peterson, “Optimal insulin delivery for the pregnant diabetic patient.” *Diabetes Care*, vol. 5, pp. 24–37, 1982.

[99] L. H. Chanoch, L. Jovanovic, and C. M. Peterson, “The evaluation of a pocket computer as an aid to insulin dose determination by patients,” *Diabetes Care*, vol. 8, no. 2, pp. 172–176, 1985.

[100] A. Schiffrin, M. Mihic, B. S. Leibel, and A. M. Albisser, "Computer-assisted insulin dosage adjustment," *Diabetes Care*, vol. 8, no. 6, pp. 545–552, 1985.

[101] F. Chiarelli, S. Tumini, G. Morgese, and A. M. Albisser, "Controlled study in diabetic children comparing insulin-dosage adjustment by manual and computer algorithms," *Diabetes Care*, vol. 13, no. 10, pp. 1080–1084, 1990.

[102] A. M. Albisser, "Analysis: Toward algorithms in diabetes self-management," *Diabetes technology & therapeutics*, vol. 5, no. 3, pp. 371–373, 2003.

[103] C. Owens, H. Zisser, L. Jovanovic, B. Srinivasan, D. Bonvin, and F. J. Doyle, "Run-to-run control of blood glucose concentrations for people with type 1 diabetes mellitus," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 6, pp. 996–1005, 2006.

[104] H. Zisser, L. Robinson, W. Bevier, E. Dassau, C. Ellingsen, F. J. Doyle III, and L. Jovanovic, "Bolus calculator: a review of four smart insulin pumps," *Diabetes technology & therapeutics*, vol. 10, no. 6, pp. 441–444, 2008.

[105] R. Bellazzi, "Telemedicine and diabetes management: current challenges and future research directions," *Journal of diabetes science and technology*, vol. 2, no. 1, pp. 98–104, 2008.

[106] W. Zhang, J. Zhong, S. Yang, Z. Gao, J. Hu, Y. Chen, and Z. Yi, "Automated identification and grading system of diabetic retinopathy using deep neural networks," *Knowledge-Based Systems*, vol. 175, pp. 12–25, 2019.

[107] M. Injadat, F. Salo, A. B. Nassif, A. Essex, and A. Shami, "Bayesian optimization with machine learning algorithms towards anomaly detection," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.

[108] F. Salo, M. Injadat, A. B. Nassif, A. Shami, and A. Essex, "Data mining techniques in intrusion detection systems: A systematic literature review," *IEEE Access*, vol. 6, pp. 56 046–56 058, 2018.

[109] F. Salo, M. Injadat, A. Moubayed, A. B. Nassif, and A. Essex, "Clustering enabled classification using ensemble feature selection for intrusion detection," in *2019 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2019, pp. 276–281.

[110] H. Wang, J. Gu, and S. Wang, "An effective intrusion detection framework based on svm with feature augmentation," *Knowledge-Based Systems*, vol. 136, pp. 130–139, 2017.

[111] P. Wang, B. Aslam, and C. C. Zou, "Peer-to-peer botnets," in *Handbook of Information and Communication Security*. Springer, 2010, pp. 335–350.

[112] J. Leonard, S. Xu, and R. Sandhu, "A framework for understanding botnets," in *2009 International Conference on Availability, Reliability and Security*. IEEE, 2009, pp. 917–922.

[113] S. Saad, I. Traore, A. Ghorbani, B. Sayed, D. Zhao, W. Lu, J. Felix, and P. Hakimian, "Detecting p2p botnets through network behavior analysis and machine learning," in *2011 Ninth annual international conference on privacy, security and trust*. IEEE, 2011, pp. 174–180.

[114] G. Gu, P. A. Porras, V. Yegneswaran, M. W. Fong, and W. Lee, "Bothunter: Detecting malware infection through ids-driven dialog correlation." in *USENIX Security Symposium*, vol. 7, 2007, pp. 1–16.

[115] A. Pektaş and T. Acarman, "Effective feature selection for botnet detection based on network flow analysis," 2017.

[116] R. Chen, W. Niu, X. Zhang, Z. Zhuo, and F. Lv, "An effective conversation-based botnet detection method," *Mathematical Problems in Engineering*, vol. 2017, 2017.

[117] T.-J. Park, C.-S. Han, and S.-H. Lee, "Development of the electronic control unit for the rack-actuating steer-by-wire using the hardware-in-the-loop simulation system," *Mechatronics*, vol. 15, no. 8, pp. 899–918, 2005.

[118] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PloS one*, vol. 11, no. 6, 2016.

[119] S. Tuohy, M. Glavin, C. Hughes, E. Jones, M. Trivedi, and L. Kilmartin, "Intra-vehicle networks: A review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 534–545, 2014.

[120] S. Biswas, R. Tatchikou, and F. Dion, "Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety," *IEEE communications magazine*, vol. 44, no. 1, pp. 74–82, 2006.

[121] F. Yu, D.-F. Li, and D. Crolla, "Integrated vehicle dynamics control-state-of-the art review," in *2008 IEEE Vehicle Power and Propulsion Conference*. IEEE, 2008, pp. 1–6.

[122] M. Farsi, K. Ratcliff, and M. Barbosa, "An overview of controller area network," *Computing & Control Engineering Journal*, vol. 10, no. 3, pp. 113–120, 1999.

[123] K. H. Johansson, M. Törngren, and L. Nielsen, "Vehicle applications of controller area network," in *Handbook of networked and embedded control systems*. Springer, 2005, pp. 741–765.

[124] A. Moubayed, A. Shami, P. Heidari, A. Larabi, and R. Brunner, "Edge-enabled v2x service placement for intelligent transportation systems," *IEEE Transactions on Mobile Computing*, 2020.

[125] A. Moubayed and A. Shami, "Softwarization, virtualization, & machine learning for intelligent & effective v2x communications," *IEEE Intelligent Transportation Systems Magazine*, 2020.

[126] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham *et al.*, "Experimental security analysis of a modern automobile," in *2010 IEEE Symposium on Security and Privacy*. IEEE, 2010, pp. 447–462.

[127] L. Yang, A. Moubayed, I. Hamieh, and A. Shami, "Tree-based intelligent intrusion detection system in internet of vehicles," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019.

[128] J. Galindo and P. Tamayo, "Credit risk assessment using statistical and machine learning: basic methodology and risk modeling applications," *Computational Economics*, vol. 15, no. 1-2, pp. 107–143, 2000.

[129] J. Chen, "Financial intermediary," May 2019, Accessed on: Nov. 1, 2019. [Online]. Available: https://www.investopedia.com/terms/f/financialintermediary.asp

[130] A. E. Khandani, A. J. Kim, and A. W. Lo, "Consumer credit-risk models via machine-learning algorithms," *Journal of Banking & Finance*, vol. 34, no. 11, pp. 2767–2787, 2010.

[131] T. Zhang, W. Zhang, X. Wei, and H. Haijing, "Multiple instance learning for credit risk assessment with transaction data," *Knowledge-Based Systems*, vol. 161, pp. 65–77, 2018.

[132] J. H. Min and Y.-C. Lee, "Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters," *Expert systems with applications*, vol. 28, no. 4, pp. 603–614, 2005.

[133] K.-j. Kim, K. Lee, and H. Ahn, "Predicting corporate financial sustainability using novel business analytics," *Sustainability*, vol. 11, no. 1, p. 64, 2019.

[134] C.-S. Lin, H. A. Khan, R.-Y. Chang, and Y.-C. Wang, "A new approach to modeling early warning systems for currency crises: Can a machine-learning fuzzy expert system predict the currency crises effectively?" *Journal of International Money and Finance*, vol. 27, no. 7, pp. 1098–1121, 2008.

[135] Communications and Marketing Office, Tufts University, "Social media overview," 2019, Accessed on: Jan. 19, 2020. [Online]. Available: https://communications.tufts.edu/marketing-and-branding/social-media-overview/

[136] A. Carvin, "Timeline: The life of the blog," Dec 2007, Accessed on: Jan. 5, 2020. [Online]. Available: https://www.npr.org/templates/story/story.php?storyId=17421022

[137] J. Clement, "Number of social network users worldwide from 2010 to 2021," Aug 2019, Accessed on: Dec. 1, 2019. [Online]. Available: https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/

[138] B. Marr, "How much data do we create every day? the mind-blowing stats everyone should read," Sep. 2019, Accessed on: Nov. 20, 2019. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/#259844e160ba

[139] P.-L. Lezotre, "Part iii - recommendations to support the next phase of international cooperation, convergence, and harmonization in the pharmaceutical domain," in *International Cooperation, Convergence and Harmonization of Pharmaceutical Regulations*, P.-L. Lezotre, Ed. Boston: Academic Press, 2014, pp. 221 – 294. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780128000533000045

[140] A. Sarker, R. Ginn, A. Nikfarjam, K. O'Connor, K. Smith, S. Jayaraman, T. Upadhaya, and G. Gonzalez, "Utilizing social media data for pharmacovigilance: a review," *Journal of biomedical informatics*, vol. 54, pp. 202–212, 2015.

[141] A. Nikfarjam, A. Sarker, K. O'connor, R. Ginn, and G. Gonzalez, "Pharmacovigilance from social media: mining adverse drug reaction mentions using sequence labeling with word embedding cluster features," *Journal of the American Medical Informatics Association*, vol. 22, no. 3, pp. 671–681, 2015.

[142] R. Leaman, L. Wojtulewicz, R. Sullivan, A. Skariah, J. Yang, and G. Gonzalez, "Towards internet-age pharmacovigilance: extracting adverse drug reactions from user posts to health-related social networks," in *Proceedings of the 2010 workshop on biomedical natural language processing*. Association for Computational Linguistics, 2010, pp. 117–125.

[143] A. Nikfarjam and G. H. Gonzalez, "Pattern mining for extraction of mentions of adverse drug reactions from user comments," in *AMIA annual symposium proceedings*, vol. 2011. American Medical Informatics Association, 2011, p. 1019.

[144] K. O'Connor, P. Pimpalkhute, A. Nikfarjam, R. Ginn, K. L. Smith, and G. Gonzalez, "Pharmacovigilance on twitter? mining tweets for adverse drug reactions," in *AMIA annual symposium proceedings*, vol. 2014. American Medical Informatics Association, 2014, pp. 924–933.

[145] A. Benton, L. Ungar, S. Hill, S. Hennessy, J. Mao, A. Chung, C. E. Leonard, and J. H. Holmes, "Identifying potential adverse effects using the web: A new approach to medical hypothesis generation," *Journal of biomedical informatics*, vol. 44, no. 6, pp. 989–996, 2011.

[146] C. C. Yang, H. Yang, L. Jiang, and M. Zhang, "Social media mining for drug safety signal detection," in *Proceedings of the 2012 international workshop on Smart health and wellbeing*, 2012, pp. 33–40.

[147] S. Yeleswarapu, A. Rao, T. Joseph, V. G. Saipradeep, and R. Srinivasan, "A pipeline to extract drug-adverse event pairs from multiple data sources," *BMC medical informatics and decision making*, vol. 14, no. 1, p. 13, 2014.

[148] C. C. Freifeld, J. S. Brownstein, C. M. Menone, W. Bao, R. Filice, T. Kass-Hout, and N. Dasgupta, "Digital drug safety surveillance: monitoring pharmaceutical products in twitter," *Drug safety*, vol. 37, no. 5, pp. 343–350, 2014.

[149] C. C. Yang, H. Yang, and L. Jiang, "Postmarketing drug safety surveillance using publicly available health-consumer-contributed content in social media," *ACM Transactions on Management Information Systems (TMIS)*, vol. 5, no. 1, pp. 1–21, 2014.

[150] H. Sampathkumar, X.-w. Chen, and B. Luo, "Mining adverse drug reactions from online healthcare forums using hidden markov model," *BMC medical informatics and decision making*, vol. 14, no. 1, p. 91, 2014.

[151] A. Patki, A. Sarker, P. Pimpalkhute, A. Nikfarjam, R. Ginn, K. O'Connor, K. Smith, and G. Gonzalez, "Mining adverse drug reaction signals from social media: going beyond extraction," *Proceedings of BioLinkSig*, vol. 2014, pp. 1–8, 2014.

[152] J. Bian, U. Topaloglu, and F. Yu, "Towards large-scale twitter mining for drug-related adverse events," in *Proceedings of the 2012 international workshop on Smart health and wellbeing*, 2012, pp. 25–32.

[153] M. Yang, X. Wang, and M. Y. Kiang, "Identification of consumer adverse drug reaction messages on social media." in *Pacific Asia Conference on Information Systems (PACIS)*, 2013, p. 193.

[154] C. D. Corley, D. J. Cook, A. R. Mikler, and K. P. Singh, "Using web and social media for influenza surveillance," in *Advances in computational biology*. Springer, 2010, pp. 559–564.

[155] K. Jiang and Y. Zheng, "Mining twitter data for potential drug effects," in *International conference on advanced data mining and applications*. Springer, 2013, pp. 434–443.

[156] R. Sloane, O. Osanlou, D. Lewis, D. Bollegala, S. Maskell, and M. Pirmohamed, "Social media and pharmacovigilance: a review of the opportunities and challenges," *British journal of clinical pharmacology*, vol. 80, no. 4, pp. 910–920, 2015.

[157] T. Joachims, "Transductive inference for text classification using support vector machines," in *Icml*, vol. 99, 1999, pp. 200–209.

[158] A. G. Dunn, J. Leask, X. Zhou, K. D. Mandl, and E. Coiera, "Associations between exposure to and expression of negative opinions about human papillomavirus vaccines on social media: an observational study," *Journal of medical Internet research*, vol. 17, no. 6, p. e144, 2015.

[159] Centers for Disease Control and Prevention (CDC), "If you choose not to vaccinate your child, understand the risk and responsibilities," 2019, Accessed on: Jan. 13, 2020. [Online]. Available: https://www.cdc.gov/vaccines/parents/vaccine-decision/no-vaccination.html

[160] ——, “Attention adults: You need vaccines too!” 2019, Accessed on: Jan. 13, 2020. [Online]. Available: https://www.cdc.gov/features/adultimmunizations/index.html

[161] X. Huang, M. C. Smith, M. J. Paul, D. Ryzhkov, S. C. Quinn, D. A. Broniatowski, and M. Dredze, “Examining patterns of influenza vaccination in social media,” in *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[162] M. Salathé and S. Khandelwal, “Assessing vaccination sentiments with online social media: implications for infectious disease dynamics and control,” *PLoS computational biology*, vol. 7, no. 10, 2011.

[163] S. Stieglitz and L. Dang-Xuan, “Social media and political communication: a social media analytics framework,” *Social network analysis and mining*, vol. 3, no. 4, pp. 1277–1291, 2013.

[164] A. I. E. Hosni and K. Li, “Minimizing the influence of rumors during breaking news events in online social networks,” *Knowledge-Based Systems*, p. 105452, 2019.

[165] D. Maynard, K. Bontcheva, and D. Rout, “Challenges in developing opinion mining tools for social media,” in *Proceedings of the Language Resources and Evaluation Conference (LREC)*, 2012, pp. 15–22.

[166] P. N. Howard, A. Duffy, D. Freelon, M. M. Hussain, W. Mari, and M. Maziad, “Opening closed regimes: what was the role of social media during the arab spring?” *Available at SSRN 2595096*, 2011.

[167] Editors of History.com Website, “Arab spring,” Jan 2018, Accessed on: Jan. 21, 2020. [Online]. Available: https://www.history.com/topics/middle-east/arab-spring

[168] H. Cunningham, D. Maynard, K. Bontcheva, H. Maynard, and V. Tablan, “Gate: A framework and graphical development environment for robust nlp tools and applications,” pp. 168–175, 2002.

[169] K. Jahanbakhsh and Y. Moon, “The predictive power of social media: On the predictability of us presidential elections using twitter,” *arXiv preprint arXiv:1407.0622*, 2014.

[170] G. Kaur and W. Singh, “Prediction of student performance using weka tool,” *An International Journal of Engineering Sciences*, vol. 17, pp. 8–16, 2016.

[171] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, “From data mining to knowledge discovery in databases,” *AI magazine*, vol. 17, no. 3, pp. 37–37, 1996.

[172] A. Abdul Aziz, N. H. Ismail, and F. Ahmad, “Mining students’ academic performance,” *Journal of Theoretical and Applied Information Technology*, vol. 53, no. 3, pp. 485–485, 2013.

[173] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, “Systematic ensemble model selection approach for educational data mining,” *Knowledge-Based Systems*, vol. 200, p. 105992, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950705120302999

[174] X. Chen, M. Vorvoreanu, and K. Madhavan, “Mining social media data for understanding students’ learning experiences,” *IEEE Transactions on Learning Technologies*, vol. 7, no. 3, pp. 246–259, July 2014.

[175] B. Kehrwald, “Understanding social presence in text-based online learning environments,” *Distance Education*, vol. 29, no. 1, pp. 89–106, 2008.

[176] E. T. Welsh, C. R. Wanberg, K. G. Brown, and M. J. Simmering, “E-learning: emerging uses, empirical results and future directions,” *international Journal of Training and Development*, vol. 7, no. 4, pp. 245–258, 2003.

[177] M. J. Rosenberg and R. Foshay, “E-learning: Strategies for delivering knowledge in the digital age,” *Performance Improvement*, vol. 41, no. 5, pp. 50–51, 2002. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/pfi.4140410512

[178] N. Islam, M. Beer, and F. Slack, “E-learning challenges faced by academics in higher education,” *Journal of Education and Training Studies*, vol. 3, no. 5, pp. 102–112, 2015.

[179] F. Essalmi, L. J. B. Ayed, M. Jemni, S. Graf, and Kinshuk, “Generalized metrics for the analysis of e-learning personalization strategies,” *Computers in Human Behavior*, vol. 48, pp. 310 – 322, 2015.

[180] R. Klamma, M. A. Chatti, E. Duval, H. Hummel, E. T. Hvannberg, M. Kravcik, E. Law, A. Naeve, and P. Scott, “Social software for life-long learning,” *Journal of Educational Technology & Society*, vol. 10, no. 3, pp. 72–83, 2007.

[181] J. Daniel, E. Vázquez Cano, and M. Gisbert Cervera, “The future of moocs: Adaptive learning or business model?” *International Journal of Educational Technology in Higher Education*, vol. 12, no. 1, pp. 64–73, Jan 2015.

[182] T. Daradoumis, R. Bassi, F. Xhafa, and S. Caballe, “A review on massive e-learning (mooc) design, delivery and assessment,” in *2013 Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, Oct 2013, pp. 208–213.

[183] K. Buffardi and S. H. Edwards, “Introducing codeworkout: An adaptive and social learning environment,” in *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, ser. SIGCSE ’14. ACM, 2014, pp. 724–724. [Online]. Available: http://doi.acm.org/10.1145/2538862.2544317

[184] Y. Ma, B. Liu, C. K. Wong, P. S. Yu, and S. M. Lee, “Targeting the right students using data mining,” in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2000, pp. 457–464.

[185] J. Luan, “Data mining and its applications in higher education,” *New Directions for Institutional Research*, vol. 2002, no. 113, pp. 17–36, 2002.

[186] ——, “Data mining and its applications in higher education,” *New Directions for Institutional Research*, vol. 2002, no. 113, pp. 17–36, 2002.

[187] B. Minaei-Bidgoli, D. A. Kashy, G. Kortemeyer, and W. F. Punch, "Predicting student performance: an application of data mining methods with an educational web-based system," in *33rd Annual Frontiers in Education, 2003. FIE 2003.*, vol. 1, Nov 2003, pp. T2A–13.

[188] S. Kotsiantis, C. Pierrakeas, and P. Pintelas, "Predicting students' performance in distance learning using machine learning techniques," *Applied Artificial Intelligence*, vol. 18, no. 5, pp. 411–426, 2004.

[189] Z. A. Pardos, N. T. Heffernan, B. Anderson, C. L. Heffernan, and W. P. Schools, "Using fine-grained skill models to fit student performance with bayesian networks," *Handbook of educational data mining*, vol. 417, 2010.

[190] J.-F. Superby, J. Vandamme, and N. Meskens, "Determination of factors influencing the achievement of the first-year university students using data mining methods," in *Workshop on educational data mining*, vol. 32, 2006, p. 234.

[191] J.-P. Vandamme, N. Meskens, and J.-F. Superby, "Predicting academic performance by data mining methods," *Education Economics*, vol. 15, no. 4, pp. 405–419, 2007.

[192] P. Cortez and A. M. G. Silva, "Using data mining to predict secondary school student performance," in *5th Annual Future Business Technology Conference*. EUROSIS-ETI, 2008, pp. 5–12.

[193] Z. Kovacic, "Early prediction of student success: Mining students' enrolment data." in *Proceedings of Informing Science & IT Education Conference (InSITE)*, 2010, pp. 647–665.

[194] M. Ramaswami and R. Bhaskaran, "A CHAID based performance prediction model in educational data mining," *CoRR*, vol. abs/1002.1144, 2010.

[195] U. K. Pandey and S. Pal, "Data mining : A prediction of performer or underperformer using classification," *CoRR*, vol. abs/1104.4163, 2011.

[196] S. K. Yadav, B. Bharadwaj, and S. Pal, "Data mining applications: A comparative study for predicting student's performance," *International Journal of Innovative Technology and Creative Engineering*, vol. abs/1202.4815, 2012.

[197] D. Kabakchieva, "Student performance prediction by using data mining classification algorithms," *International journal of computer science and management research*, vol. 1, no. 4, pp. 686–690, 2012.

[198] S. K. Yadav and S. Pal, "Data mining: A prediction for performance improvement of engineering students using classification," *CoRR*, vol. abs/1203.3832, 2012.

[199] V. Ramesh, P. Parkavi, and K. Ramar, "Predicting student performance: a statistical and data mining approach," *International journal of computer applications*, vol. 63, no. 8, 2013.

[200] J. Hung, B. E. Shelton, J. Yang, and X. Du, "Improving predictive modeling for at-risk student identification: A multistage approach," *IEEE Transactions on Learning Technologies*, vol. 12, no. 2, pp. 148–157, April 2019.

[201] S. Helal, J. Li, L. Liu, E. Ebrahimie, S. Dawson, D. J. Murray, and Q. Long, "Predicting academic performance by considering student heterogeneity," *Knowledge-Based Systems*, vol. 161, pp. 134 – 146, 2018.

[202] K. Zupanc and Z. Bosnic, "Automated essay evaluation with semantic analysis," *Knowledge-Based Systems*, vol. 120, pp. 118 – 132, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950705117300072

[203] J. Xu, K. H. Moon, and M. van der Schaar, "A machine learning approach for tracking and predicting student performance in degree programs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 5, pp. 742–753, Aug 2017.

[204] B. Sekeroglu, K. Dimililer, and K. Tuncal, "Student performance prediction and classification using machine learning algorithms," in *Proceedings of the 2019 8th International Conference on Educational and Information Technology*, ser. ICEIT 2019. New York, NY, USA: Association for Computing Machinery, 2019, pp. 7–11. [Online]. Available: https://doi.org/10.1145/3318396.3318419

[205] I. Khan, A. Al Sadiri, A. R. Ahmad, and N. Jabeur, "Tracking student performance in introductory programming by means of machine learning," in *2019 4th MEC International Conference on Big Data and Smart City (ICBDSC)*, Jan 2019, pp. 1–6.

[206] M. Vahdat, L. Oneto, D. Anguita, M. Funk, and M. Rauterberg, "A learning analytics approach to correlate the academic achievements of students with interaction data from an educational simulator," in *Design for Teaching and Learning in a Networked World*. Cham: Springer International Publishing, 2015, pp. 352–366.

[207] A. Churches, "Bloom's taxonomy blooms digitally," *Tech & Learning*, vol. 1, pp. 1–6, 2008.

[208] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2013.

[209] S. Geisser, *Predictive inference*. New York, NY, USA: NY: Chapman and Hall, 1993.

[210] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-split Optimized Bagging Ensemble Model Selection for Multi-class Educational Datasets," *Applied Intelligence*, Jul. 2020.

[211] M. Ramaswami, "Validating predictive performance of classifier models for multiclass problem in educational data mining," *International Journal of Computer Science Issues (IJCSI)*, vol. 11, no. 5, p. 86, 2014.

[212] M. Aly, "Survey on multiclass classification methods," *Neural Network*, vol. 19, pp. 1–9, 2005.

[213] P. Bühlmann, B. Yu *et al.*, “Analyzing bagging,” *The Annals of Statistics*, vol. 30, no. 4, pp. 927–961, 2002.

[214] P. Bühlmann, “Bagging, boosting and ensemble methods,” in *Handbook of Computational Statistics*. Springer, 2012, pp. 985–1022.

[215] Netflix Inc., “Netflix competition,” 2009. [Online]. Available: https://www.netflixprize.com/

[216] I. Guyon, V. Lemaire, M. Boullé, G. Dror, and D. Vogel, “Design and analysis of the kdd cup 2009: fast scoring on a large orange customer database,” *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 2, pp. 68–76, 2010.

[217] Kaggle Inc., “Kaggle,” 2019. [Online]. Available: https://www.kaggle.com/

[218] C. Romero and S. Ventura, “Educational data mining: A survey from 1995 to 2005,” *Expert systems with applications*, vol. 33, no. 1, pp. 135–146, 2007.

[219] M. K. Khribi, M. Jemni, and O. Nasraoui, “Automatic recommendations for e-learning personalization based on web usage mining techniques and information retrieval,” in *2008 Eighth IEEE International Conference on Advanced Learning Technologies*, July 2008, pp. 241–245.

[220] Y.-C. Chang, W.-Y. Kao, C.-P. Chu, and C.-H. Chiu, “A learning style classification mechanism for e-learning,” *Computers & Education*, vol. 53, no. 2, pp. 273–285, 2009.

[221] S. T. Hijazi and S. Naqvi, “Factors affecting students’performance.” *Bangladesh e-journal of Sociology*, vol. 3, no. 1, 2006.

[222] B. K. Baradwaj and S. Pal, “Mining educational data to analyze students’ performance,” *arXiv preprint arXiv:1201.3417*, 2012.

[223] B. K. Bhardwaj and S. Pal, “Data mining: A prediction for performance improvement using classification,” *arXiv preprint arXiv:1201.3418*, 2012.

[224] S. Pal, “Mining educational data to reduce dropout rates of engineering students,” *International Journal of Information Engineering and Electronic Business*, vol. 4, no. 2, p. 1, 2012.

[225] G. N. R. Prasad and A. V. Babu, “Mining previous marks data to predict students performance in their final year examinations,” *International Journal of engineering research and technology*, vol. 2, no. 2, pp. 1–4, 2013.

[226] A. B. E. D. Ahmed and I. S. Elaraby, “Data mining: A prediction for student’s performance using classification method,” *World Journal of Computer Application and Technology*, vol. 2, no. 2, pp. 43–47, 2014.

[227] B. Khan, M. S. H. Khiyal, and M. D. Khattak, “Final grade prediction of secondary school student using decision tree,” *International Journal of Computer Applications*, vol. 115, no. 21, 2015.

[228] S. Kotsiantis, K. Patriarcheas, and M. Xenos, “A combinational incremental ensemble of classifiers as a technique for predicting students’ performance in distance education,” *Knowledge-Based Systems*, vol. 23, no. 6, pp. 529–535, 2010.

[229] R. Saxena, “Educational data mining: Performance evaluation of decision tree and clustering techniques using weka platform,” *International Journal of Computer Science and Business Informatics*, vol. 15, no. 2, pp. 26–37.

[230] S. Rana and R. Garg, “Evaluation of student’s performance of an institute using clustering algorithms,” *International Journal of Applied Engineering Research*, vol. 11, no. 5, pp. 3605–3609, 2016.

[231] X. Wang, Y. Zhang, S. Yu, X. Liu, Y. Yuan, and F. Wang, “E-learning recommendation framework based on deep learning,” in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2017, pp. 455–460.

[232] W. W. Fok, Y. He, H. A. Yeung, K. Law, K. Cheung, Y. Ai, and P. Ho, “Prediction model for students’ future development by deep learning and tensorflow artificial intelligence engine,” in *2018 4th International Conference on Information Management (ICIM)*. IEEE, 2018, pp. 103–106.

[233] D. J. Hand and R. J. Till, “A simple generalisation of the area under the roc curve for multiple class classification problems,” *Machine learning*, vol. 45, no. 2, pp. 171–186, 2001.

[234] A. Hosseinzadeh, M. Izadi, A. Verma, D. Precup, and D. Buckeridge, “Assessing the predictability of hospital readmission using machine learning,” in *Twenty-Fifth IAAI Conference*, 2013.

[235] C. Sammut and G. I. Webb, *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.

[236] T. Vujicic, T. Matijevic, J. Ljucovic, A. Balota, and Z. Sevarac, “Comparative analysis of methods for determining number of hidden neurons in artificial neural network,” in *Central european conference on information and intelligent systems*. Faculty of Organization and Informatics Varazdin, 2016, p. 219.

[237] D. Nguyen and B. Widrow, “Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights,” in *1990 IJCNN International Joint Conference on Neural Networks*. IEEE, 1990, pp. 21–26.

[238] D. W. Marquardt, “An algorithm for least-squares estimation of nonlinear parameters,” *Journal of the society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[239] C. Lv, Y. Xing, J. Zhang, X. Na, Y. Li, T. Liu, D. Cao, and F.-Y. Wang, “Levenberg–marquardt backpropagation training of multilayer neural networks for state estimation of a safety-critical cyber-physical system,” *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3436–3446, 2017.

[240] V. Dhar, A. Tickoo, R. Koul, and B. Dubey, "Comparative performance of some popular artificial neural network algorithms on benchmark and function approximation problems," *Pramana*, vol. 74, no. 2, pp. 307–324, 2010.

[241] P. Koch, B. Wujek, O. Golovidov, and S. Gardner, "Automated hyperparameter tuning for effective machine learning," in *Proceedings of the SAS Global Forum 2017 Conference*, 2017, pp. 1–23.

[242] M. Kuhn *et al.*, "Building predictive models in r using the caret package," *Journal of statistical software*, vol. 28, no. 5, pp. 1–26, 2008.

[243] M. Gevrey, I. Dimopoulos, and S. Lek, "Review and comparison of methods to study the contribution of variables in artificial neural network models," *Ecological modelling*, vol. 160, no. 3, pp. 249–264, 2003.

[244] MATLAB, "R2018b)," Natick, Massachusetts, 2018.

[245] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, "Intrusion detection by machine learning: A review," *Expert Systems with Applications*, vol. 36, no. 10, pp. 11 994–12 000, 2009.

[246] M. B. Salem, S. Hershkop, and S. J. Stolfo, "A survey of insider attack detection research," in *Insider Attack and Cyber Security*. Springer, 2008, pp. 69–90.

[247] W. Bul'ajoul, A. James, and M. Pannu, "Improving network intrusion detection system performance through quality of service configuration and parallel technology," *Journal of Computer and System Sciences*, vol. 81, no. 6, pp. 981–999, 2015.

[248] S. M. H. Bamakan, B. Amiri, M. Mirzabagheri, and Y. Shi, "A new intrusion detection approach using pso based multiple criteria linear programming," *Procedia Computer Science*, vol. 55, pp. 231–237, 2015.

[249] S. X. Wu and W. Banzhaf, "The use of computational intelligence in intrusion detection systems: A review," *Applied soft computing*, vol. 10, no. 1, pp. 1–35, 2010.

[250] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, 2013.

[251] S. Suthaharan, "Big data classification: Problems and challenges in network intrusion prediction with machine learning," *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, no. 4, pp. 70–73, 2014.

[252] J. Zhang and M. Zulkernine, "Anomaly based network intrusion detection with unsupervised outlier detection," in *Communications, 2006. ICC'06. IEEE International Conference on*, vol. 5. IEEE, 2006, pp. 2388–2393.

[253] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers & Security*, vol. 31, no. 3, pp. 357 – 374, 2012.

[254] F. Kuang, W. Xu, and S. Zhang, "A novel hybrid kpca and svm with ga model for intrusion detection," *Applied Soft Computing*, vol. 18, pp. 178–184, 2014.

[255] A. S. Eesa, Z. Orman, and A. M. A. Brifcani, "A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems," *Expert Systems with Applications*, vol. 42, no. 5, pp. 2670–2679, 2015.

[256] W. Li, P. Yi, Y. Wu, L. Pan, and J. Li, "A new intrusion detection system based on knn classification algorithm in wireless sensor network," *Journal of Electrical and Computer Engineering*, vol. 2014, 2014.

[257] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," *Journal of Computational Science*, 2017.

[258] Y. Y. Chung and N. Wahid, "A hybrid network intrusion detection system using simplified swarm optimization (sso)," *Applied Soft Computing*, vol. 12, no. 9, pp. 3014–3022, 2012.

[259] F. Kuang, W. Xu, and S. Zhang, "A novel hybrid kpca and svm with ga model for intrusion detection," *Applied Soft Computing*, vol. 18, pp. 178–184, 2014.

[260] J. Zhang, M. Zulkernine, and A. Haque, "Random-forests-based network intrusion detection systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 5, pp. 649–659, 2008.

[261] A. J. Malik and F. A. Khan, "A hybrid technique using multi-objective particle swarm optimization and random forests for probe attacks detection in a network," in *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*. IEEE, 2013, pp. 2473–2478.

[262] I. S. Thaseen and C. A. Kumar, "Intrusion detection model using fusion of chi-square feature selection and multi class svm," *Journal of King Saud University-Computer and Information Sciences*, vol. 29, no. 4, pp. 462–472, 2017.

[263] H. Bostani and M. Sheikhan, "Modification of supervised opf-based intrusion detection systems using unsupervised learning and social network concept," *Pattern Recognition*, vol. 62, pp. 56–72, 2017.

[264] W. Meng, W. Li, and L.-F. Kwok, "Design of intelligent knn-based alarm filter using knowledge-based alert verification in intrusion detection," *Security and Communication Networks*, vol. 8, no. 18, pp. 3883–3895, 2015.

[265] A. J. Malik, W. Shahzad, and F. A. Khan, "Binary pso and random forests algorithm for probe attacks detection in a network," in *2011 IEEE Congress of Evolutionary Computation (CEC)*, June 2011, pp. 662–668.

[266] E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599*, 2010.

[267] M. Salem and U. Buehler, "Mining techniques in network security to enhance intrusion detection systems," *arXiv preprint arXiv:1212.2414*, 2012.

[268] H. Huang, R. S. Khalid, W. Liu, and H. Yu, "Work-in-progress: a fast online sequential learning accelerator for iot network intrusion detection," in *Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2017 International Conference on*. IEEE, 2017, pp. 1–2.

[269] W. Yassin, N. I. Udzir, Z. Muda, M. N. Sulaiman *et al.*, "Anomaly-based intrusion detection through k-means clustering and naives bayes classification," in *Proc. 4th Int. Conf. Comput. Informatics, ICOCI*, no. 49, 2013, pp. 298–303.

[270] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-Stage Optimized Machine Learning Framework for Network Intrusion Detection," *IEEE Transactions On Network and Service Management*, Aug. 2020.

[271] A. Moubayed, E. Aqeeli, and A. Shami, "Ensemble-based Feature Selection and Classification Model for DNS Typo-squatting Detection," in *33rd Canadian Conference on Electrical and Computer Engineering (CCECE'20)*. IEEE, 2020, pp. 1–6.

[272] X. Sun, J. Dai, P. Liu, A. Singhal, and J. Yen, "Using bayesian networks for probabilistic identification of zero-day attack paths," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, pp. 2506–2521, Oct 2018.

[273] A. Alsirhani, S. Sampalli, and P. Bodorik, "Ddos detection system: Using a set of classification algorithms controlled by fuzzy logic system in apache spark," *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 936–949, Sep. 2019.

[274] A. A. Daya, M. A. Salahuddin, N. Limam, and R. Boutaba, "Botchase: Graph-based bot detection using machine learning," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2020.

[275] T. Kim, B. Kang, M. Rho, S. Sezer, and E. G. Im, "A multimodal deep learning method for android malware detection using various features," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 3, pp. 773–788, Mar 2019.

[276] Z. Chen, Q. Yan, H. Han, S. Wang, L. Peng, L. Wang, and B. Yang, "Machine learning based mobile malware detection using highly imbalanced network traffic," *Information Sciences*, vol. 433, pp. 346–364, 2018.

[277] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[278] M. B. Catalkaya, O. Kalipsiz, M. S. Aktas, and U. O. Turgut, "Data feature selection methods on distributed big data processing platforms," in *2018 3rd International Conference on Computer Science and Engineering (UBMK)*, Sep. 2018, pp. 133–138.

[279] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of machine learning research*, vol. 13, no. Feb, pp. 281–305, 2012.

[280] D. Ashlock, *Evolutionary computation for modeling and optimization*. Springer Science & Business Media, 2006.

[281] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, 2012, pp. 2951–2959.

[282] K. M. Ali Alheeti and K. McDonald-Maier, "Intelligent intrusion detection in external communication systems for autonomous vehicles," *Systems Science & Control Engineering*, vol. 6, no. 1, pp. 48–56, 2018.

[283] X. Tan, S. Su, Z. Huang, X. Guo, Z. Zuo, X. Sun, and L. Li, "Wireless sensor networks intrusion detection based on smote and the random forest algorithm," *Sensors*, vol. 19, no. 1, p. 203, 2019.

[284] R. S. B. Krishna and M. Aramudhan, "Feature Selection Based on Information Theory for Pattern Classification," in *2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, Jul. 2014, pp. 1233–1236.

[285] B. Bonev, "Feature selection based on information theory," Ph.D. dissertation, University of Alicante, Jun. 2010.

[286] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM Computing Surveys (CSUR)*, vol. 50, no. 6, p. 94, 2018.

[287] M. A. Hall, "Correlation-based feature selection for machine learning," Ph.D. dissertation, University of Waikato Hamilton, 1999.

[288] J. H. Gennari, P. Langley, and D. Fisher, "Models of incremental concept formation," *Artificial intelligence*, vol. 40, no. 1-3, pp. 11–61, 1989.

[289] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231220311693

[290] L. Bianchi, M. Dorigo, L. M. Gambardella, and W. J. Gutjahr, "A survey on metaheuristics for stochastic combinatorial optimization," *Natural Computing*, vol. 8, no. 2, pp. 239–287, 2009.

[291] S.-W. Lin, K.-C. Ying, S.-C. Chen, and Z.-J. Lee, "Particle swarm optimization for parameter determination and feature selection of support vector machines," *Expert Systems with Applications*, vol. 35, no. 4, pp. 1817 – 1824, 2008.

[292] G. Cohen, M. Hilario, and A. Geissbuhler, "Model selection for support vector classifiers via genetic algorithms. an application to medical decision support," in *International Symposium on Biological and Medical Data Analysis*. Springer, 2004, pp. 200–211.

[293] S. G. Ahmad, C. S. Liew, E. U. Munir, T. F. Ang, and S. U. Khan, "A hybrid genetic algorithm for optimization of scheduling workflow applications in heterogeneous computing systems," *Journal of Parallel and Distributed Computing*, vol. 87, pp. 80–90, 2016.

[294] S. Blaifi, S. Moulahoum, I. Colak, and W. Merrouche, "An enhanced dynamic model of battery using genetic algorithm suitable for photovoltaic applications," *Applied Energy*, vol. 169, pp. 888–898, 2016.

[295] U. Mehboob, J. Qadir, S. Ali, and A. Vasilakos, "Genetic algorithms in wireless networking: techniques, applications, and issues," *Soft Computing*, vol. 20, no. 6, pp. 2467–2501, 2016.

[296] A. Rikhtegar, M. Pooyan, and M. T. Manzuri-Shalmani, "Genetic algorithm-optimised structure of convolutional neural network for face recognition applications," *IET Computer Vision*, vol. 10, no. 6, pp. 559–566, 2016.

[297] I. Dewancker, M. McCourt, and S. Clark, "Bayesian optimization primer," 2015.

[298] K. Eggensperger, M. Feurer, F. Hutter, J. Bergstra, J. Snoek, H. Hoos, and K. Leyton-Brown, "Towards an empirical foundation for assessing bayesian optimization of hyperparameters," in *NIPS workshop on Bayesian Optimization in Theory and Practice*, vol. 10, 2013, p. 3.

[299] F. Hu and H. Li, "A novel boundary oversampling algorithm based on neighborhood rough set model: Nrsboundary-smote," *Mathematical Problems in Engineering*, vol. 2013, 2013.

[300] A. Lissovoi, P. S. Oliveto, and J. A. Warwicker, "On the time complexity of algorithm selection hyper-heuristics for multimodal optimisation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 2322–2329.

[301] R. Cheng and Y. Jin, "A social learning particle swarm optimization algorithm for scalable optimization," *Information Sciences*, vol. 291, pp. 43–60, 2015.

[302] P. S. Oliveto and C. Witt, "Improved time complexity analysis of the simple genetic algorithm," *Theoretical Computer Science*, vol. 605, pp. 21–41, 2015.

[303] M. Feurer and F. Hutter, "Hyperparameter optimization," in *Automated Machine Learning*. Springer, 2019, pp. 3–33.

[304] The Kernel Trip, "Computational complexity of machine learning algorithms," Apr. 2018.

[305] C.-T. Chu, S. K. Kim, Y.-A. Lin, Y. Yu, G. Bradski, K. Olukotun, and A. Y. Ng, "Map-reduce for machine learning on multicore," in *Advances in neural information processing systems*, 2007, pp. 281–288.

[306] F. Salo, M. Injadat, A. B. Nassif, and A. Essex, "Data mining with big data in intrusion detection systems: A systematic literature review," in *International Symposium on Big Data Management and Analytics 2019*, Apr. 2019.

[307] A. Moubayed, A. Refaey, and A. Shami, "Software-defined perimeter (sdp): State of the art secure solution for modern networks," *IEEE Network*, vol. 33, no. 5, pp. 226–233, 2019.

[308] P. Kumar, A. Moubayed, A. Refaey, A. Shami, and J. Koilpillai, "Performance analysis of sdp for secure internal enterprises," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1–6.

[309] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[310] O. Veksler, "Cs434a/541a class notes prof. olga veksler," 2015. [Online]. Available: http://www.csd.uwo.ca/courses/CS9840a/Lecture2_knn.pdf

[311] X. Solé, A. Ramisa, and C. Torras, "Evaluation of random forests on large-scale classification problems using a bag-of-visual-words representation." in *CCIA*, 2014, pp. 273–276.

[312] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*. Springer, 2013, vol. 112.

[313] N. Moustafa, B. Turnbull, and K. R. Choo, "An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4815–4830, 2019.

Appendix A

Appendix

A.1 Ensemble Learners - Majority Voting Ensemble

rf	mlp	bn	knn	lreg	svm	G	G1	G2	G3	G4	G5	Avg	p
1	0	1	0	0	0	0.75	0.889	0.88	0.815	0.857	0.778	0.828	0.0034
1	0	1	1	0	0	0.679	0.944	0.84	0.852	0.821	0.815	0.825	0.0034
1	0	0	0	0	0	0.786	0.889	0.84	0.778	0.857	0.778	0.821	0.00452
1	1	1	1	1	0	0.75	0.944	0.88	0.852	0.786	0.704	0.819	0.00452
1	0	1	0	0	1	0.821	0.944	0.88	0.778	0.821	0.667	0.819	0.00452
1	1	1	1	1	1	0.786	0.944	0.8	0.852	0.821	0.704	0.818	0.00452
1	0	1	1	0	1	0.821	0.944	0.8	0.815	0.857	0.667	0.817	0.00452
1	1	1	0	1	0	0.786	0.944	0.88	0.778	0.75	0.741	0.813	0.00452
1	0	1	1	1	0	0.75	0.944	0.88	0.852	0.786	0.667	0.813	0.00452
1	1	1	1	0	0	0.679	0.944	0.84	0.852	0.786	0.778	0.813	0.00452
1	0	0	0	1	0	0.786	0.944	0.88	0.778	0.786	0.704	0.813	0.00452
1	1	0	1	1	0	0.75	0.944	0.84	0.852	0.786	0.704	0.813	0.00452
1	1	1	0	0	1	0.786	0.944	0.84	0.778	0.786	0.741	0.812	0.00452
1	1	1	1	0	1	0.786	0.944	0.8	0.815	0.821	0.704	0.812	0.00452
0	0	1	1	0	0	0.607	0.944	0.88	0.852	0.821	0.741	0.808	0.00452
1	1	1	0	0	0	0.714	0.944	0.88	0.778	0.75	0.778	0.807	0.00452
1	0	0	1	1	0	0.75	0.944	0.84	0.852	0.786	0.667	0.806	0.00452
1	1	0	1	0	0	0.75	0.944	0.8	0.852	0.821	0.667	0.806	0.00452
1	0	1	1	1	1	0.786	0.944	0.8	0.815	0.821	0.667	0.806	0.00452
1	0	1	0	1	0	0.786	0.944	0.88	0.778	0.75	0.667	0.801	0.00567
1	0	1	0	1	1	0.786	0.944	0.88	0.778	0.786	0.63	0.801	0.00567
1	1	1	0	1	1	0.786	0.944	0.84	0.815	0.714	0.704	0.8	0.00567
1	1	0	0	0	0	0.786	0.944	0.8	0.778	0.786	0.704	0.8	0.00567
1	0	0	1	0	0	0.75	0.944	0.8	0.852	0.821	0.63	0.8	0.00567
1	1	0	1	1	1	0.786	0.944	0.8	0.815	0.786	0.667	0.8	0.00567
0	1	1	1	1	0	0.679	0.944	0.84	0.852	0.75	0.704	0.795	0.00567
1	0	0	0	1	1	0.786	0.944	0.8	0.778	0.75	0.704	0.794	0.00567

Table A.1: Dataset 1 - Stage 20% - Part1/2

rf	mlp	bn	knn	lreg	svm	G	G1	G2	G3	G4	G5	Avg	p
1	0	0	1	1	1	0.786	0.944	0.8	0.815	0.786	0.63	0.793	0.00567
1	1	0	1	0	1	0.786	0.944	0.76	0.815	0.786	0.667	0.793	0.00567
0	0	1	1	1	0	0.714	0.944	0.84	0.852	0.75	0.63	0.788	0.00567
0	0	1	0	0	1	0.75	0.944	0.84	0.778	0.786	0.63	0.788	0.00567
1	1	0	0	0	1	0.786	0.944	0.76	0.778	0.786	0.667	0.787	0.00567
1	1	0	0	1	0	0.786	0.944	0.84	0.778	0.679	0.667	0.782	0.00732
0	1	1	1	1	1	0.679	0.944	0.8	0.852	0.75	0.667	0.782	0.00732
0	1	1	1	0	1	0.679	0.944	0.8	0.815	0.786	0.667	0.782	0.00732
0	1	1	0	0	1	0.75	0.944	0.8	0.778	0.75	0.667	0.781	0.00732
1	0	0	1	0	1	0.786	0.944	0.72	0.815	0.786	0.63	0.78	0.00732
1	0	0	0	0	1	0.821	0.889	0.76	0.704	0.786	0.704	0.777	0.00732
0	0	1	0	1	0	0.75	0.944	0.84	0.778	0.714	0.63	0.776	0.00732
0	1	1	0	1	1	0.714	0.944	0.8	0.815	0.714	0.667	0.776	0.00732
1	1	0	0	1	1	0.786	0.944	0.8	0.778	0.679	0.667	0.776	0.00732
0	0	1	1	1	1	0.679	0.944	0.8	0.815	0.821	0.593	0.775	0.00732
0	0	1	1	0	1	0.714	0.889	0.8	0.815	0.821	0.593	0.772	0.00732
0	1	1	1	0	0	0.464	0.944	0.84	0.852	0.714	0.778	0.765	0.00901
0	1	1	0	1	0	0.714	0.944	0.84	0.778	0.643	0.667	0.764	0.00901
0	0	1	0	1	1	0.714	0.944	0.84	0.778	0.679	0.63	0.764	0.00901
0	0	1	0	0	0	0.643	0.944	0.84	0.815	0.821	0.519	0.764	0.00901
0	0	0	1	1	0	0.714	0.944	0.72	0.852	0.75	0.519	0.75	0.00979
0	1	1	0	0	0	0.5	0.944	0.84	0.778	0.679	0.741	0.747	11.57
0	1	0	1	0	1	0.679	0.944	0.64	0.815	0.786	0.556	0.737	11.57
0	1	0	0	1	1	0.714	0.944	0.76	0.741	0.679	0.556	0.732	13.98
0	0	0	1	1	1	0.679	0.944	0.68	0.815	0.75	0.519	0.731	13.98
0	1	0	1	1	1	0.679	0.944	0.68	0.815	0.679	0.556	0.725	13.98
0	1	0	0	0	1	0.714	0.944	0.72	0.778	0.786	0.407	0.725	13.98
0	0	0	1	0	0	0.679	0.889	0.72	0.63	0.857	0.556	0.722	13.98
0	1	0	1	1	0	0.607	0.944	0.72	0.852	0.607	0.556	0.714	17.12
0	0	0	1	0	1	0.679	0.889	0.64	0.741	0.786	0.519	0.709	17.12
0	0	0	0	1	1	0.714	0.944	0.76	0.741	0.679	0.407	0.708	17.12
0	1	0	1	0	0	0.321	0.944	0.68	0.852	0.714	0.556	0.678	24.61
0	1	0	0	1	0	0.643	0.944	0.84	0.593	0.393	0.556	0.661	24.61
0	1	0	0	0	0	0.357	0.944	0.8	0.778	0.5	0.37	0.625	35.24
0	0	0	0	0	1	0.786	0.833	0.52	0.481	0.75	0.259	0.605	47.27
0	0	0	0	1	0	0.75	0.889	0.76	0.444	0.179	0.407	0.572	54.23
0	0	0	0	0	0	0	0	0	0	0	0	0	991.39

Table A.2: Dataset 1 - Stage 20% - Part2/2

rf	mlp	bn	knn	lreg	svm	G	G1	G2	G3	G4	G5	Avg	p
1	0	0	0	0	1	0.929	1	1	1	0.929	1	0.976	0.00036
1	0	0	1	0	1	0.929	1	1	1	0.929	1	0.976	0.00036
1	1	0	1	0	1	0.929	1	1	1	0.929	1	0.976	0.00036
1	0	0	1	0	0	0.929	1	1	1	0.893	1	0.97	0.00036
1	1	0	1	0	0	0.929	1	1	1	0.893	1	0.97	0.00036
1	1	0	0	0	1	0.893	1	1	1	0.929	1	0.97	0.00036
1	0	0	0	1	1	0.929	1	1	0.963	0.929	1	0.97	0.00036
0	0	0	1	1	1	0.929	1	1	0.963	0.929	1	0.97	0.00036
1	0	0	1	1	1	0.929	1	1	0.963	0.929	1	0.97	0.00036
0	1	0	1	1	1	0.929	1	1	0.963	0.929	1	0.97	0.00036
1	1	0	1	1	1	0.929	1	1	0.963	0.929	1	0.97	0.00036
1	0	0	0	0	0	0.929	1	1	1	0.857	1	0.964	0.00054
0	0	0	1	0	0	0.893	1	1	1	0.893	1	0.964	0.00036
0	1	0	1	0	0	0.929	1	1	1	0.893	0.963	0.964	0.00054
0	0	1	1	0	0	0.929	1	1	0.963	0.893	1	0.964	0.00054
1	0	0	1	1	0	0.929	1	1	0.963	0.893	1	0.964	0.00054
0	1	0	1	1	0	0.929	1	1	0.963	0.893	1	0.964	0.00054
1	1	0	1	1	0	0.929	1	1	0.963	0.893	1	0.964	0.00054
0	1	0	1	0	1	0.929	1	1	1	0.929	0.926	0.964	0.00054
1	1	0	0	1	1	0.893	1	1	0.963	0.929	1	0.964	0.00054
1	1	0	0	0	0	0.893	1	1	1	0.857	1	0.958	0.00054
1	0	1	1	0	0	0.929	1	1	0.963	0.857	1	0.958	0.00054
0	1	1	1	0	0	0.929	1	1	0.963	0.857	1	0.958	0.00054
1	1	1	1	0	0	0.929	1	1	0.963	0.857	1	0.958	0.00054
0	0	0	1	1	0	0.929	1	1	0.926	0.893	1	0.958	0.00054
0	0	1	1	1	0	0.929	1	1	0.926	0.893	1	0.958	0.00054
0	0	0	1	0	1	0.929	1	1	1	0.929	0.889	0.958	0.00054
0	0	1	1	0	1	0.929	1	1	0.963	0.857	1	0.958	0.00054
1	0	1	1	0	1	0.929	1	1	0.963	0.857	1	0.958	0.00054
0	1	1	1	0	1	0.929	1	1	0.963	0.857	1	0.958	0.00054
1	1	1	1	0	1	0.929	1	1	0.963	0.857	1	0.958	0.00054

Table A.3: Dataset 1 - Stage 50% - Part1/2

rf	mlp	bn	knn	lreg	svm	G	G1	G2	G3	G4	G5	Avg	p
1	0	1	0	0	0	0.893	1	1	0.963	0.857	1	0.952	0.00054
1	1	1	0	0	0	0.893	1	1	0.963	0.857	1	0.952	0.00054
1	0	0	0	1	0	0.929	1	1	0.926	0.857	1	0.952	0.00054
1	1	0	0	1	0	0.893	1	1	0.963	0.857	1	0.952	0.00054
1	0	1	1	1	0	0.929	1	1	0.926	0.857	1	0.952	0.00054
0	1	1	1	1	0	0.929	1	1	0.926	0.857	1	0.952	0.00054
1	1	1	1	1	0	0.929	1	1	0.926	0.857	1	0.952	0.00054
1	0	1	0	0	1	0.893	1	1	0.963	0.857	1	0.952	0.00054
1	1	1	0	0	1	0.893	1	1	0.963	0.857	1	0.952	0.00054
0	1	0	0	1	1	0.893	1	1	0.963	0.857	1	0.952	0.00054
0	0	1	1	1	1	0.929	1	1	0.926	0.857	1	0.952	0.00054
1	0	1	1	1	1	0.929	1	1	0.926	0.857	1	0.952	0.00054
0	1	1	1	1	1	0.929	1	1	0.926	0.857	1	0.952	0.00054
1	1	1	1	1	1	0.929	1	1	0.926	0.857	1	0.952	0.00054
1	0	1	0	1	0	0.893	1	1	0.926	0.857	1	0.946	0.00076
1	1	1	0	1	0	0.893	1	1	0.926	0.857	1	0.946	0.00076
0	1	0	0	0	1	0.893	1	1	1	0.857	0.926	0.946	0.00076
0	0	0	0	1	1	0.893	1	1	0.926	0.857	1	0.946	0.00076
1	0	1	0	1	1	0.893	1	1	0.926	0.857	1	0.946	0.00076
1	1	1	0	1	1	0.893	1	1	0.926	0.857	1	0.946	0.00076
0	1	1	0	0	0	0.857	1	1	0.926	0.857	1	0.94	0.00076
0	1	1	0	1	0	0.857	1	1	0.926	0.857	1	0.94	0.00076
0	1	1	0	0	1	0.893	1	1	0.963	0.786	1	0.94	0.00076
0	0	1	0	1	0	0.821	1	1	0.889	0.893	1	0.934	0.00076
0	0	1	0	0	1	0.857	1	1	0.963	0.786	1	0.934	0.00076
0	1	1	0	1	1	0.893	1	1	0.926	0.786	1	0.934	0.00076
0	0	1	0	1	1	0.857	1	1	0.926	0.786	1	0.928	0.00097
0	0	1	0	0	0	0.821	1	1	0.815	0.893	1	0.922	0.00097
0	0	0	0	0	1	0.929	1	1	1	0.857	0.63	0.903	0.00124
0	1	0	0	1	0	0.857	0.944	1	0.852	0.679	1	0.889	0.00177
0	1	0	0	0	0	0.857	0.944	1	0.889	0.679	0.926	0.882	0.00177
0	0	0	0	1	0	0.893	0.778	0.8	0.593	0.714	1	0.796	0.00588
0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A.4: Dataset 1 - Stage 50% - Part2/2

rf	mlp	bn	knn	lreg	svm	G	G1	G2	G3	G4	G5	Avg	p
0	0	1	0	1	0	0.89	0.698	0.872	0.846	0.849	0.863	0.836	0.0000024
0	0	0	0	1	0	0.888	0.702	0.876	0.84	0.856	0.854	0.836	0.0000024
0	1	1	0	0	0	0.882	0.667	0.872	0.834	0.867	0.894	0.836	0.0000032
0	1	1	0	1	0	0.886	0.677	0.872	0.837	0.864	0.88	0.836	0.000003
0	0	1	0	0	0	0.89	0.687	0.849	0.88	0.819	0.882	0.835	0.0000024
0	1	0	0	1	0	0.88	0.679	0.876	0.823	0.865	0.875	0.833	0.000004
0	1	1	0	1	1	0.874	0.677	0.865	0.843	0.862	0.873	0.832	0.0000051
0	1	1	0	0	1	0.867	0.665	0.863	0.846	0.862	0.887	0.832	0.0000064
0	0	1	0	1	1	0.876	0.689	0.861	0.854	0.84	0.859	0.83	0.0000046
0	1	0	0	1	1	0.855	0.675	0.863	0.834	0.867	0.868	0.827	0.0000105
0	0	0	0	1	1	0.857	0.7	0.859	0.829	0.847	0.849	0.824	0.0000105
0	1	0	0	0	1	0.824	0.652	0.863	0.831	0.871	0.882	0.821	0.0000355
0	0	1	0	0	1	0.874	0.675	0.851	0.891	0.727	0.897	0.819	0.0000051
1	1	1	0	1	0	0.876	0.648	0.843	0.834	0.828	0.854	0.814	0.0000046
1	0	1	0	1	0	0.882	0.658	0.836	0.843	0.812	0.837	0.811	0.0000032
1	1	1	0	1	1	0.861	0.648	0.841	0.837	0.83	0.849	0.811	0.0000081
1	1	1	0	0	0	0.874	0.638	0.836	0.829	0.825	0.863	0.811	0.0000051
1	1	0	0	1	0	0.878	0.648	0.839	0.82	0.824	0.854	0.81	0.0000041
1	1	1	0	0	1	0.855	0.634	0.832	0.84	0.821	0.863	0.807	0.0000105
1	0	1	0	0	0	0.876	0.636	0.814	0.874	0.781	0.861	0.807	0.0000046
0	1	1	1	1	0	0.89	0.675	0.839	0.797	0.819	0.82	0.807	0.0000024
1	0	1	0	1	1	0.863	0.654	0.832	0.849	0.807	0.832	0.806	0.0000078
0	0	1	1	1	0	0.894	0.689	0.839	0.797	0.807	0.811	0.806	0.0000018
1	1	0	0	1	1	0.849	0.644	0.834	0.829	0.828	0.851	0.806	0.0000129
0	1	1	1	0	0	0.882	0.671	0.836	0.797	0.822	0.82	0.805	0.0000032
1	0	0	0	1	0	0.878	0.656	0.83	0.837	0.804	0.823	0.805	0.0000041
0	1	0	1	1	0	0.882	0.677	0.836	0.791	0.818	0.82	0.804	0.0000032
0	1	1	1	1	1	0.87	0.675	0.832	0.797	0.819	0.82	0.802	0.0000058
0	1	1	1	0	1	0.865	0.669	0.834	0.806	0.818	0.82	0.802	0.0000067
0	0	0	1	1	0	0.882	0.685	0.836	0.789	0.81	0.803	0.801	0.0000032
0	1	0	1	1	1	0.853	0.675	0.834	0.791	0.822	0.82	0.799	0.0000121
0	0	1	1	1	1	0.872	0.683	0.834	0.797	0.803	0.806	0.799	0.0000051
0	1	0	1	0	0	0.853	0.667	0.832	0.797	0.83	0.815	0.799	0.0000121

Table A.5: Dataset 2 - Stage 20% - Part1/2

rf	mlp	bn	knn	lreg	svm	G	G1	G2	G3	G4	G5	Avg	p
1	1	0	0	0	0	0.843	0.619	0.828	0.823	0.824	0.851	0.798	0.0000175
1	0	0	0	1	1	0.851	0.656	0.826	0.826	0.806	0.818	0.797	0.0000128
1	1	0	0	0	1	0.832	0.617	0.826	0.826	0.819	0.851	0.795	0.0000259
0	0	1	1	0	0	0.892	0.65	0.824	0.803	0.787	0.811	0.794	0.0000019
0	1	0	1	0	1	0.834	0.66	0.834	0.797	0.824	0.815	0.794	0.0000256
0	0	0	1	1	1	0.855	0.685	0.834	0.78	0.809	0.801	0.794	0.0000105
1	0	1	0	0	1	0.859	0.65	0.818	0.889	0.69	0.859	0.794	0.0000088
0	1	0	0	0	0	0.64	0.656	0.872	0.823	0.877	0.887	0.792	0.0027613
1	1	1	1	1	0	0.876	0.648	0.812	0.797	0.803	0.815	0.792	0.0000046
1	1	0	1	1	0	0.876	0.65	0.814	0.786	0.801	0.815	0.79	0.0000046
1	1	1	1	0	0	0.87	0.644	0.807	0.791	0.806	0.818	0.789	0.0000058
1	1	1	1	1	1	0.855	0.648	0.81	0.797	0.804	0.815	0.788	0.0000105
1	0	1	1	1	0	0.878	0.658	0.81	0.797	0.794	0.789	0.788	0.0000041
1	1	0	1	1	1	0.851	0.646	0.812	0.789	0.806	0.815	0.786	0.0000128
1	1	1	1	0	1	0.849	0.642	0.805	0.803	0.801	0.818	0.786	0.0000129
1	0	0	1	1	0	0.874	0.656	0.807	0.789	0.79	0.782	0.783	0.0000051
1	0	1	1	1	1	0.857	0.656	0.805	0.8	0.791	0.787	0.783	0.0000105
1	1	0	1	0	0	0.834	0.636	0.803	0.791	0.807	0.813	0.781	0.0000256
1	0	1	1	0	0	0.874	0.638	0.803	0.803	0.77	0.796	0.781	0.0000051
1	0	0	1	1	1	0.849	0.658	0.801	0.777	0.793	0.784	0.777	0.0000129
0	0	1	1	0	1	0.872	0.654	0.824	0.803	0.696	0.813	0.777	0.0000051
1	1	0	1	0	1	0.822	0.629	0.801	0.794	0.803	0.811	0.777	0.0000359
1	0	1	1	0	1	0.853	0.652	0.801	0.803	0.68	0.794	0.764	0.0000121
0	0	0	1	0	0	0.859	0.654	0.762	0.729	0.397	0.719	0.687	0.0000088
1	0	0	1	0	0	0.841	0.634	0.72	0.726	0.421	0.727	0.678	0.0000202
1	0	0	1	0	1	0.828	0.573	0.752	0.666	0.507	0.681	0.668	0.000029
0	0	0	1	0	1	0.843	0.615	0.778	0.666	0.404	0.688	0.666	0.0000175
1	0	0	0	0	1	0.772	0.6	0.692	0.683	0.517	0.703	0.661	0.000156
1	0	0	0	0	0	0.787	0.66	0.528	0.789	0.434	0.748	0.658	0.0001109
0	0	0	0	0	1	0.77	0.46	0.499	0.48	0.329	0.542	0.513	0.0001693
0	0	0	0	0	0	0.021	-0.213	0.101	-0.143	0.056	0.206	0.005	0.9247

Table A.6: Dataset 2 - Stage 20% - Part2/2

rf	mlp	bn	knn	lreg	svm	G	G1	G2	G3	G4	G5	Avg	p
0	1	0	0	1	0	0.899	0.888	0.925	0.977	0.929	0.988	0.934	0.00012
0	0	0	0	1	0	0.934	0.876	0.923	0.98	0.902	0.983	0.933	0.00001
0	1	0	0	0	0	0.859	0.886	0.932	0.98	0.929	0.981	0.928	0.00075
0	1	1	0	1	0	0.901	0.857	0.928	0.966	0.919	0.988	0.926	0.0001
0	1	1	0	0	0	0.876	0.861	0.925	0.966	0.92	0.988	0.923	0.0003
0	1	0	1	1	0	0.894	0.88	0.925	0.92	0.927	0.983	0.922	0.00013
0	0	1	0	1	0	0.934	0.82	0.923	0.966	0.896	0.983	0.92	0.00001
0	1	1	1	1	0	0.896	0.855	0.928	0.914	0.917	0.983	0.916	0.00013
0	0	0	1	1	0	0.921	0.867	0.923	0.886	0.904	0.981	0.914	0.00005
1	1	1	0	1	0	0.888	0.834	0.913	0.957	0.901	0.986	0.913	0.0002
0	1	0	1	0	0	0.872	0.878	0.932	0.886	0.927	0.981	0.913	0.00037
1	1	0	0	1	0	0.886	0.834	0.911	0.963	0.895	0.986	0.912	0.0002
0	0	1	0	0	0	0.936	0.812	0.884	0.957	0.895	0.976	0.91	0.00001
1	0	1	0	1	0	0.915	0.795	0.907	0.954	0.89	0.988	0.908	0.00005
0	1	1	1	0	0	0.884	0.859	0.925	0.88	0.917	0.981	0.908	0.00021
1	1	1	0	0	0	0.857	0.834	0.905	0.954	0.901	0.983	0.906	0.00076
1	1	1	1	1	0	0.886	0.834	0.913	0.911	0.899	0.986	0.905	0.0002
1	1	0	1	1	0	0.888	0.834	0.911	0.917	0.893	0.986	0.905	0.0002
1	0	0	0	1	0	0.913	0.793	0.907	0.96	0.868	0.988	0.905	0.00007
1	0	1	0	0	0	0.894	0.787	0.905	0.954	0.893	0.986	0.903	0.00013
0	0	1	1	1	0	0.921	0.818	0.921	0.88	0.898	0.981	0.903	0.00005
1	1	0	0	0	0	0.841	0.83	0.905	0.963	0.895	0.978	0.902	0.00161
1	1	1	1	0	0	0.872	0.832	0.909	0.911	0.901	0.981	0.901	0.00037
1	0	1	1	1	0	0.909	0.795	0.911	0.909	0.89	0.983	0.9	0.00007
1	1	0	1	0	0	0.861	0.83	0.909	0.917	0.895	0.981	0.899	0.00072
0	0	1	1	0	0	0.925	0.816	0.901	0.871	0.895	0.981	0.898	0.00003
0	1	1	0	1	1	0.903	0.818	0.896	0.966	0.799	0.988	0.895	0.00009
0	1	0	0	1	1	0.899	0.822	0.899	0.966	0.781	0.988	0.892	0.00012
1	0	0	1	1	0	0.905	0.793	0.911	0.886	0.868	0.983	0.891	0.00007
1	0	1	1	0	0	0.899	0.791	0.903	0.877	0.89	0.986	0.891	0.00012
0	1	1	0	0	1	0.892	0.818	0.882	0.966	0.799	0.988	0.891	0.00013
1	1	1	0	1	1	0.896	0.799	0.894	0.957	0.797	0.986	0.888	0.00013

Table A.7: Dataset 2 - Stage 50% - Part1/2

rf	mlp	bn	knn	lreg	svm	G	G1	G2	G3	G4	G5	Avg	p
0	1	0	0	0	1	0.872	0.824	0.882	0.963	0.784	0.988	0.885	0.00037
0	1	1	1	1	1	0.896	0.818	0.896	0.914	0.797	0.983	0.884	0.00013
0	1	0	1	1	1	0.894	0.82	0.899	0.914	0.779	0.986	0.882	0.00013
1	1	0	0	1	1	0.89	0.793	0.892	0.954	0.775	0.986	0.882	0.00016
1	1	1	0	0	1	0.878	0.797	0.878	0.954	0.796	0.983	0.881	0.00029
1	1	1	1	1	1	0.888	0.799	0.894	0.911	0.794	0.983	0.878	0.0002
1	1	0	0	0	1	0.865	0.795	0.878	0.951	0.775	0.983	0.875	0.00057
0	1	1	1	0	1	0.886	0.818	0.884	0.88	0.799	0.981	0.875	0.0002
1	1	0	1	1	1	0.888	0.793	0.894	0.911	0.773	0.983	0.874	0.0002
1	1	1	1	0	1	0.876	0.797	0.878	0.911	0.794	0.981	0.873	0.0003
0	0	1	0	1	1	0.932	0.735	0.894	0.966	0.72	0.986	0.872	0.00001
0	1	0	1	0	1	0.876	0.822	0.884	0.877	0.784	0.981	0.871	0.0003
1	0	1	0	1	1	0.921	0.716	0.89	0.954	0.736	0.986	0.867	0.00005
1	1	0	1	0	1	0.863	0.795	0.878	0.906	0.773	0.981	0.866	0.00061
1	0	1	0	0	1	0.901	0.71	0.857	0.954	0.738	0.983	0.857	0.0001
0	0	1	1	1	1	0.919	0.735	0.896	0.88	0.72	0.981	0.855	0.00005
0	0	1	0	0	1	0.921	0.731	0.849	0.926	0.717	0.983	0.854	0.00005
1	0	1	1	1	1	0.911	0.716	0.89	0.889	0.735	0.983	0.854	0.00007
0	0	0	0	1	1	0.928	0.737	0.896	0.966	0.59	0.986	0.85	0.00003
1	0	0	0	1	1	0.909	0.696	0.89	0.954	0.664	0.986	0.85	0.00007
1	0	1	1	0	1	0.896	0.71	0.857	0.88	0.736	0.983	0.844	0.00013
0	0	1	1	0	1	0.923	0.729	0.857	0.849	0.716	0.981	0.842	0.00004
1	0	0	1	1	1	0.903	0.696	0.89	0.88	0.661	0.983	0.835	0.00009
1	0	0	1	0	0	0.88	0.723	0.818	0.849	0.75	0.986	0.834	0.00027
0	0	0	1	1	1	0.917	0.735	0.899	0.88	0.59	0.981	0.834	0.00005
1	0	0	0	0	0	0.774	0.716	0.824	0.92	0.753	0.978	0.828	0.01431
1	0	0	1	0	1	0.886	0.66	0.851	0.777	0.649	0.983	0.801	0.0002
1	0	0	0	0	1	0.776	0.663	0.851	0.817	0.65	0.983	0.79	0.01347
0	0	0	1	0	1	0.894	0.522	0.853	0.403	0.51	0.981	0.694	0.00013
0	0	0	1	0	0	0.888	0.497	0.847	0.406	0.44	0.981	0.676	0.0002
0	0	0	0	0	1	0.499	0.522	0.847	0.406	0.511	0.983	0.628	23.97
0	0	0	0	0	0	0.021	-0.213	0.101	-0.143	0.056	0.206	0.005	92.83

Table A.8: Dataset 2 - Stage 50% - Part2/2

A.2 Ensemble Learners - Multi-Split Bagging Ensemble

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
1	0	0	1	0	1	0	0	0	0.608	0.619	0.662	0.516	0.434	0.757	0.599
2	0	0	1	0	0	0	0	0	0.57	0.651	0.633	0.565	0.709	0.435	0.621
3	0	0	1	0	1	0	0	1	0.528	0.516	0.551	0.487	0.504	0.929	0.586
4	1	0	1	0	0	0	0	0	0.519	0.553	0.676	0.45	0.603	0.566	0.561
5	0	0	1	0	0	1	0	1	0.504	0.514	0.571	0.543	0.506	0.701	0.557
6	1	0	1	0	0	1	0	0	0.551	0.514	0.558	0.55	0.53	0.635	0.556
7	0	0	1	0	1	1	0	1	0.578	0.514	0.595	0.543	0.219	0.886	0.556
8	1	0	1	0	1	0	0	0	0.534	0.534	0.576	0.462	0.467	0.738	0.552
9	0	0	1	0	0	0	0	1	0.453	0.534	0.627	0.407	0.696	0.595	0.552
10	0	0	1	0	1	1	1	1	0.517	0.472	0.565	0.506	0.44	0.807	0.551
11	1	0	1	0	1	0	0	1	0.528	0.497	0.515	0.401	0.455	0.898	0.549
12	0	0	1	0	0	1	1	0	0.573	0.453	0.568	0.524	0.694	0.475	0.548
13	1	0	1	0	1	1	0	1	0.578	0.496	0.565	0.488	0.302	0.849	0.546
14	1	0	1	0	0	1	0	1	0.504	0.514	0.571	0.506	0.52	0.659	0.546
15	0	0	1	1	1	1	0	1	0.512	0.477	0.551	0.55	0.358	0.807	0.542
16	0	0	1	0	1	0	1	0	0.463	0.521	0.608	0.432	0.659	0.57	0.542
17	0	0	1	1	1	0	0	1	0.424	0.442	0.546	0.469	0.505	0.812	0.533
18	1	0	1	0	1	1	0	0	0.578	0.514	0.595	0.525	0.213	0.763	0.531
19	0	1	1	0	1	0	0	1	0.485	0.502	0.534	0.487	0.31	0.843	0.527
20	0	0	1	0	1	1	1	0	0.573	0.484	0.6	0.536	0.305	0.659	0.526
21	1	1	1	0	0	0	0	0	0.496	0.521	0.62	0.425	0.588	0.505	0.526
22	1	0	1	0	1	1	1	1	0.498	0.453	0.534	0.464	0.454	0.751	0.526
23	1	0	1	0	1	1	1	0	0.498	0.472	0.565	0.499	0.427	0.677	0.523
24	0	0	1	0	1	0	1	1	0.436	0.467	0.516	0.401	0.573	0.731	0.521
25	1	0	1	1	1	1	0	0	0.512	0.459	0.595	0.525	0.339	0.689	0.52
26	0	0	1	0	0	0	1	0	0.451	0.565	0.602	0.43	0.746	0.318	0.519
27	0	0	1	1	0	1	0	1	0.443	0.44	0.583	0.506	0.625	0.498	0.516
28	1	0	1	1	1	1	1	1	0.448	0.453	0.522	0.457	0.484	0.721	0.514
29	1	1	1	0	0	0	0	1	0.477	0.484	0.564	0.438	0.57	0.548	0.514
30	1	0	1	0	0	0	0	1	0.453	0.476	0.547	0.401	0.571	0.627	0.513
31	1	1	1	0	1	0	0	1	0.504	0.484	0.527	0.432	0.353	0.775	0.512
32	0	1	1	0	0	0	0	1	0.459	0.484	0.59	0.475	0.551	0.509	0.511
33	1	1	1	1	1	1	0	1	0.53	0.459	0.553	0.494	0.321	0.709	0.511
34	1	0	1	1	0	1	0	0	0.4	0.496	0.583	0.506	0.533	0.542	0.51
35	1	0	1	0	1	0	1	1	0.443	0.467	0.497	0.396	0.536	0.719	0.51
36	0	0	1	0	0	1	0	0	0.588	0.533	0.63	0.604	0.135	0.566	0.509
37	0	0	1	1	1	0	0	0	0.352	0.497	0.551	0.504	0.499	0.652	0.509
38	1	0	1	1	1	1	0	1	0.493	0.453	0.565	0.439	0.322	0.782	0.509
39	0	0	1	1	0	1	0	0	0.461	0.496	0.613	0.58	0.469	0.424	0.507
40	0	1	1	0	0	0	0	0	0.504	0.533	0.642	0.531	0.45	0.381	0.507
41	0	0	1	1	1	1	1	1	0.473	0.435	0.541	0.526	0.465	0.598	0.506
42	0	1	1	1	1	1	0	1	0.53	0.459	0.576	0.55	0.233	0.69	0.506
43	1	0	1	1	1	1	1	0	0.467	0.435	0.553	0.482	0.454	0.64	0.505
44	1	1	1	0	1	1	0	1	0.559	0.477	0.565	0.45	0.214	0.763	0.505
45	1	1	1	0	1	1	1	1	0.498	0.447	0.516	0.482	0.395	0.69	0.505
46	1	0	1	0	0	1	1	0	0.461	0.435	0.526	0.481	0.57	0.549	0.504

Table A.9: Dataset 1 at Stage 20% - Gini Indices Part 1/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
47	1	1	1	0	1	0	1	0	0.528	0.472	0.497	0.42	0.491	0.61	0.503
48	0	1	1	0	1	0	1	0	0.468	0.49	0.565	0.475	0.424	0.591	0.502
49	1	0	1	0	1	0	1	0	0.424	0.467	0.509	0.413	0.573	0.627	0.502
50	0	0	1	1	1	1	0	0	0.549	0.477	0.607	0.53	0.198	0.647	0.501
51	1	1	1	0	1	1	1	0	0.583	0.435	0.528	0.494	0.342	0.616	0.5
52	0	1	1	0	1	1	1	1	0.517	0.416	0.528	0.501	0.342	0.69	0.499
53	0	0	1	1	1	1	1	0	0.51	0.416	0.571	0.506	0.44	0.549	0.499
54	1	0	1	1	1	0	0	1	0.369	0.423	0.504	0.432	0.468	0.794	0.498
55	0	1	1	0	0	1	0	1	0.565	0.477	0.602	0.543	0.271	0.53	0.498
56	0	1	1	0	0	0	1	0	0.407	0.435	0.575	0.464	0.738	0.362	0.497
57	0	1	1	1	1	0	0	1	0.438	0.428	0.539	0.506	0.403	0.665	0.497
58	1	0	1	0	0	1	1	1	0.424	0.435	0.514	0.464	0.602	0.53	0.495
59	1	0	0	0	1	0	0	1	0.43	0.442	0.459	0.413	0.305	0.917	0.494
60	1	1	1	0	0	1	0	1	0.533	0.477	0.565	0.45	0.357	0.579	0.493
61	1	1	1	0	0	1	0	0	0.57	0.496	0.582	0.536	0.189	0.586	0.493
62	0	0	1	0	1	1	0	0	0.615	0.563	0.644	0.555	-0.204	0.782	0.493
63	0	0	1	0	0	1	1	1	0.424	0.416	0.526	0.487	0.602	0.498	0.492
64	1	0	1	1	1	0	0	0	0.424	0.46	0.534	0.413	0.45	0.671	0.492
65	0	0	1	1	0	0	0	0	0.283	0.534	0.657	0.425	0.696	0.355	0.492
66	1	1	1	0	1	0	1	1	0.443	0.453	0.485	0.408	0.491	0.665	0.491
67	1	0	1	1	0	1	0	1	0.443	0.435	0.553	0.439	0.514	0.561	0.491
68	1	1	1	1	1	0	0	1	0.456	0.447	0.497	0.45	0.379	0.714	0.491
69	1	1	1	0	1	0	0	0	0.51	0.502	0.553	0.425	0.278	0.671	0.49
70	0	1	1	1	1	1	1	0	0.529	0.379	0.546	0.593	0.34	0.549	0.489
71	1	0	0	1	1	1	0	1	0.419	0.435	0.472	0.494	0.259	0.849	0.488
72	1	0	0	0	1	1	1	1	0.455	0.416	0.442	0.464	0.321	0.825	0.487
73	0	0	1	1	0	0	0	1	0.343	0.439	0.571	0.457	0.627	0.484	0.487
74	1	1	1	1	1	1	1	1	0.406	0.447	0.51	0.457	0.439	0.66	0.486
75	0	1	1	1	1	0	0	0	0.45	0.465	0.546	0.518	0.359	0.579	0.486
76	1	1	1	0	0	1	1	0	0.535	0.416	0.538	0.494	0.451	0.481	0.486
77	0	0	1	1	0	1	1	1	0.438	0.398	0.528	0.506	0.59	0.45	0.485
78	1	0	0	0	1	1	0	1	0.522	0.477	0.502	0.45	0.036	0.917	0.484
79	1	0	1	0	0	0	1	0	0.35	0.467	0.519	0.4	0.701	0.467	0.484
80	1	1	1	1	0	0	0	1	0.393	0.484	0.522	0.438	0.565	0.493	0.482
81	1	1	1	1	1	1	1	0	0.461	0.416	0.534	0.482	0.414	0.586	0.482
82	0	1	1	1	0	1	0	1	0.467	0.459	0.565	0.506	0.412	0.48	0.481
83	0	0	1	1	0	1	1	0	0.444	0.398	0.526	0.506	0.602	0.412	0.481
84	1	1	1	1	1	0	0	0	0.438	0.484	0.546	0.469	0.341	0.61	0.481
85	1	1	1	1	0	1	0	1	0.448	0.44	0.565	0.494	0.389	0.549	0.481
86	0	0	0	1	1	1	1	1	0.424	0.398	0.479	0.489	0.358	0.733	0.48
87	0	0	0	0	1	0	0	1	0.479	0.404	0.452	0.487	0.131	0.922	0.479
88	0	0	1	1	1	0	1	0	0.352	0.49	0.504	0.413	0.573	0.534	0.478
89	0	1	1	0	1	0	1	1	0.399	0.435	0.497	0.39	0.472	0.671	0.477
90	1	1	0	0	1	0	0	1	0.504	0.447	0.411	0.432	0.171	0.898	0.477
91	0	0	0	0	1	0	1	1	0.357	0.393	0.46	0.427	0.418	0.805	0.476
92	1	0	1	1	0	1	1	0	0.424	0.398	0.514	0.469	0.59	0.463	0.476

Table A.10: Dataset 1 at Stage 20% - Gini Indices Part 2/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
93	0	0	0	0	0	1	0	0	0.567	0.594	0.574	0.661	-0.194	0.653	0.476
94	0	1	1	0	0	1	1	1	0.48	0.398	0.484	0.538	0.469	0.481	0.475
95	1	1	1	1	1	1	0	0	0.512	0.477	0.576	0.469	0.196	0.616	0.474
96	0	1	1	1	1	1	1	1	0.455	0.398	0.522	0.507	0.377	0.586	0.474
97	0	1	1	0	1	1	0	1	0.596	0.477	0.565	0.525	-0.142	0.819	0.473
98	0	0	0	1	1	1	0	1	0.419	0.422	0.514	0.55	0.065	0.868	0.473
99	0	1	1	1	0	1	1	1	0.443	0.398	0.497	0.519	0.501	0.468	0.471
100	0	1	1	1	0	0	0	0	0.338	0.465	0.588	0.469	0.577	0.379	0.469
101	0	0	1	1	1	0	1	1	0.35	0.43	0.528	0.364	0.536	0.608	0.469
102	1	0	1	1	1	0	1	0	0.337	0.448	0.504	0.395	0.536	0.596	0.469
103	1	1	1	1	0	1	0	0	0.448	0.44	0.583	0.506	0.362	0.475	0.469
104	1	0	0	0	1	0	1	1	0.399	0.393	0.418	0.408	0.362	0.831	0.468
105	1	1	1	0	0	0	1	1	0.406	0.453	0.49	0.396	0.602	0.463	0.468
106	1	0	0	1	1	1	1	1	0.436	0.398	0.46	0.438	0.339	0.738	0.468
107	1	1	1	1	0	0	0	0	0.337	0.484	0.564	0.45	0.533	0.438	0.468
108	1	0	0	1	1	0	0	1	0.357	0.423	0.404	0.408	0.332	0.88	0.467
109	1	1	1	0	0	0	1	0	0.387	0.453	0.527	0.401	0.608	0.426	0.467
110	0	0	0	0	1	1	1	1	0.443	0.398	0.46	0.482	0.219	0.8	0.467
111	1	1	1	1	1	0	1	0	0.35	0.435	0.485	0.445	0.485	0.598	0.466
112	0	1	1	1	1	0	1	0	0.389	0.416	0.516	0.445	0.479	0.554	0.466
113	1	1	1	0	0	1	1	1	0.443	0.416	0.484	0.445	0.488	0.518	0.466
114	0	0	0	0	1	1	0	0	0.598	0.476	0.587	0.642	-0.321	0.812	0.466
115	1	0	0	0	0	1	0	1	0.387	0.416	0.46	0.469	0.326	0.733	0.465
116	0	1	1	1	0	1	0	0	0.541	0.445	0.613	0.599	0.184	0.406	0.465
117	0	0	0	1	1	0	0	1	0.332	0.386	0.423	0.445	0.3	0.898	0.464
118	1	1	1	1	1	0	1	1	0.362	0.435	0.473	0.445	0.467	0.598	0.463
119	0	1	1	1	0	1	1	0	0.443	0.379	0.508	0.556	0.494	0.394	0.462
120	1	0	1	1	0	0	0	1	0.35	0.439	0.522	0.39	0.548	0.522	0.462
121	1	0	1	1	1	0	1	1	0.355	0.411	0.504	0.396	0.48	0.622	0.461
122	1	0	0	1	1	0	1	1	0.387	0.393	0.455	0.427	0.406	0.696	0.461
123	1	1	1	1	0	1	1	0	0.461	0.398	0.484	0.501	0.483	0.431	0.46
124	1	0	1	1	0	1	1	1	0.387	0.393	0.516	0.445	0.516	0.5	0.459
125	0	0	0	1	1	0	1	1	0.382	0.374	0.448	0.445	0.462	0.645	0.459
126	1	0	1	0	0	0	1	1	0.369	0.448	0.484	0.359	0.59	0.498	0.458
127	0	0	1	0	0	0	1	1	0.278	0.448	0.502	0.356	0.738	0.424	0.458
128	0	1	1	1	0	0	0	1	0.325	0.41	0.522	0.438	0.583	0.467	0.458
129	1	0	0	1	1	1	1	0	0.418	0.379	0.467	0.482	0.339	0.659	0.457
130	0	0	0	0	0	1	0	1	0.412	0.496	0.49	0.587	0	0.757	0.457
131	1	0	0	1	1	0	0	0	0.32	0.442	0.453	0.457	0.319	0.738	0.455
132	1	1	0	0	1	0	1	1	0.399	0.416	0.387	0.427	0.39	0.708	0.455
133	0	1	1	1	1	0	1	1	0.313	0.416	0.492	0.445	0.485	0.573	0.454
134	0	1	1	0	1	1	0	0	0.633	0.459	0.613	0.543	-0.205	0.677	0.453
135	1	1	0	0	1	1	1	1	0.455	0.416	0.43	0.464	0.245	0.709	0.453
136	0	1	1	0	0	0	1	1	0.301	0.435	0.527	0.396	0.608	0.45	0.453
137	1	1	0	1	1	1	0	1	0.419	0.422	0.46	0.513	0.134	0.77	0.453
138	1	1	1	1	0	1	1	1	0.369	0.398	0.497	0.464	0.483	0.505	0.452

Table A.11: Dataset 1 at Stage 20% - Gini Indices Part 3/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
139	0	0	1	1	0	0	1	0	0.328	0.453	0.506	0.364	0.757	0.306	0.452
140	0	1	1	0	1	0	0	0	0.566	0.496	0.613	0.499	-0.097	0.634	0.452
141	0	1	0	0	1	0	1	1	0.36	0.416	0.442	0.445	0.303	0.743	0.451
142	1	0	1	1	0	0	0	0	0.276	0.457	0.559	0.364	0.571	0.475	0.45
143	0	1	1	0	0	1	1	0	0.535	0.398	0.5	0.519	0.308	0.426	0.448
144	0	0	0	1	0	1	0	1	0.296	0.422	0.502	0.525	0.326	0.615	0.448
145	1	0	0	0	1	1	1	0	0.436	0.379	0.472	0.457	0.208	0.733	0.448
146	0	1	1	1	0	0	1	0	0.352	0.453	0.502	0.396	0.651	0.33	0.447
147	0	0	0	0	1	1	0	1	0.548	0.44	0.514	0.58	-0.279	0.88	0.447
148	1	0	0	0	1	0	0	0	0.448	0.442	0.447	0.407	0.096	0.843	0.447
149	1	0	0	0	1	0	1	0	0.399	0.393	0.418	0.408	0.381	0.682	0.447
150	1	1	0	1	1	1	1	1	0.399	0.398	0.418	0.457	0.351	0.648	0.445
151	1	0	0	1	1	0	1	0	0.35	0.393	0.485	0.464	0.376	0.603	0.445
152	0	1	0	1	1	0	0	1	0.345	0.41	0.416	0.432	0.248	0.819	0.445
153	1	0	0	1	0	1	0	1	0.308	0.398	0.509	0.494	0.365	0.59	0.444
154	1	1	0	1	1	0	0	1	0.413	0.41	0.386	0.42	0.239	0.794	0.443
155	0	1	1	1	0	0	1	1	0.315	0.416	0.509	0.415	0.602	0.399	0.443
156	1	1	0	1	1	0	1	0	0.369	0.416	0.436	0.445	0.39	0.596	0.442
157	0	0	0	0	0	1	1	1	0.406	0.398	0.371	0.482	0.451	0.536	0.441
158	0	0	1	1	0	0	1	1	0.291	0.448	0.509	0.364	0.627	0.404	0.44
159	1	1	0	1	1	0	1	1	0.35	0.398	0.436	0.39	0.397	0.671	0.44
160	0	1	0	0	1	1	0	0	0.616	0.44	0.501	0.599	-0.23	0.714	0.44
161	0	0	0	0	1	0	1	0	0.429	0.404	0.434	0.432	0.322	0.619	0.44
162	1	1	1	1	0	0	1	0	0.301	0.453	0.509	0.378	0.583	0.412	0.439
163	1	0	0	0	0	0	0	1	0.295	0.383	0.404	0.315	0.444	0.794	0.439
164	0	1	0	0	1	1	0	1	0.542	0.427	0.472	0.562	-0.23	0.856	0.438
165	0	0	0	1	1	0	1	0	0.315	0.379	0.473	0.469	0.443	0.545	0.437
166	0	1	1	1	1	1	0	0	0.586	0.459	0.588	0.599	-0.193	0.579	0.436
167	1	0	0	0	0	1	1	1	0.382	0.416	0.378	0.427	0.432	0.567	0.434
168	1	1	1	0	1	1	0	0	0.578	0.477	0.576	0.506	-0.214	0.677	0.433
169	0	1	0	1	1	1	1	1	0.387	0.379	0.448	0.489	0.282	0.611	0.433
170	1	0	0	0	0	0	0	0	0.283	0.439	0.448	0.253	0.468	0.701	0.432
171	0	1	0	0	1	0	0	1	0.511	0.442	0.435	0.501	-0.232	0.929	0.431
172	1	0	0	1	0	1	1	1	0.352	0.374	0.423	0.464	0.452	0.518	0.43
173	1	1	1	1	0	0	1	1	0.276	0.416	0.49	0.396	0.553	0.45	0.43
174	1	0	0	1	1	1	0	0	0.345	0.44	0.502	0.506	0.022	0.763	0.43
175	1	1	0	0	1	0	1	0	0.394	0.416	0.411	0.445	0.278	0.628	0.429
176	1	0	0	0	0	1	1	0	0.401	0.398	0.371	0.457	0.414	0.53	0.428
177	0	1	0	1	1	0	1	1	0.359	0.398	0.418	0.427	0.408	0.561	0.428
178	0	0	0	1	1	1	1	0	0.394	0.398	0.502	0.524	0.166	0.585	0.428
179	1	0	0	1	0	1	0	0	0.289	0.422	0.479	0.488	0.331	0.561	0.428
180	1	1	0	1	1	0	0	0	0.37	0.428	0.472	0.469	0.159	0.671	0.428
181	1	1	0	0	1	1	0	1	0.559	0.44	0.472	0.525	-0.278	0.849	0.428
182	1	0	1	1	0	0	1	1	0.259	0.448	0.467	0.359	0.571	0.45	0.426
183	0	1	0	1	1	0	1	0	0.397	0.416	0.43	0.464	0.295	0.548	0.425
184	1	1	0	1	1	1	1	0	0.455	0.379	0.46	0.482	0.178	0.593	0.425

Table A.12: Dataset 1 at Stage 20% - Gini Indices Part 4/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
185	1	1	0	0	0	1	1	1	0.406	0.398	0.373	0.464	0.389	0.518	0.424
186	0	1	1	0	0	1	0	0	0.62	0.482	0.6	0.555	-0.206	0.493	0.424
187	1	1	0	0	0	0	0	1	0.355	0.428	0.411	0.383	0.338	0.628	0.424
188	0	1	1	0	1	1	1	0	0.602	0.435	0.514	0.531	-0.127	0.586	0.424
189	0	1	0	0	0	1	0	0	0.59	0.432	0.496	0.605	-0.146	0.544	0.42
190	1	1	0	0	0	1	0	1	0.424	0.422	0.435	0.525	0.041	0.672	0.42
191	0	1	0	0	1	0	0	0	0.568	0.477	0.416	0.587	-0.291	0.757	0.419
192	1	0	1	1	0	0	1	0	0.291	0.448	0.485	0.327	0.534	0.412	0.416
193	1	0	0	1	0	1	1	0	0.315	0.416	0.403	0.464	0.415	0.481	0.416
194	0	1	0	0	0	0	0	1	0.443	0.378	0.44	0.395	0.191	0.645	0.415
195	1	1	0	1	0	0	0	1	0.312	0.404	0.43	0.364	0.439	0.541	0.415
196	0	0	0	1	1	0	0	0	0.259	0.381	0.465	0.518	0.11	0.757	0.415
197	0	0	0	0	1	0	0	0	0.517	0.472	0.462	0.587	-0.385	0.824	0.413
198	1	1	0	1	0	1	1	1	0.381	0.361	0.386	0.464	0.383	0.5	0.412
199	0	0	0	1	0	1	1	1	0.309	0.361	0.416	0.482	0.452	0.45	0.412
200	1	1	0	0	1	1	0	0	0.559	0.44	0.465	0.506	-0.267	0.763	0.411
201	1	0	0	0	1	1	0	0	0.504	0.477	0.514	0.525	-0.365	0.812	0.411
202	0	0	0	1	0	1	1	0	0.296	0.398	0.395	0.487	0.464	0.412	0.409
203	0	0	0	0	0	0	1	1	0.32	0.356	0.353	0.388	0.571	0.465	0.409
204	1	1	0	0	0	0	1	1	0.307	0.398	0.385	0.378	0.477	0.5	0.407
205	1	1	0	1	0	1	0	1	0.355	0.403	0.423	0.476	0.241	0.536	0.406
206	1	1	0	1	0	1	1	0	0.362	0.379	0.373	0.464	0.389	0.463	0.405
207	0	0	0	0	1	1	1	0	0.544	0.41	0.488	0.499	-0.227	0.701	0.402
208	1	0	0	0	0	0	1	0	0.279	0.393	0.339	0.334	0.534	0.527	0.401
209	0	0	0	0	0	0	0	1	0.332	0.364	0.349	0.327	0.407	0.618	0.4
210	1	0	0	0	0	0	1	1	0.32	0.374	0.353	0.309	0.511	0.529	0.399
211	0	1	0	1	1	1	0	1	0.475	0.39	0.484	0.513	-0.255	0.788	0.399
212	0	1	0	1	0	1	1	1	0.32	0.361	0.386	0.464	0.407	0.45	0.398
213	0	1	0	1	0	0	1	1	0.296	0.361	0.398	0.415	0.509	0.399	0.396
214	1	0	0	1	0	0	0	1	0.227	0.364	0.423	0.322	0.487	0.553	0.396
215	0	0	0	0	0	1	1	0	0.493	0.41	0.431	0.519	0.057	0.461	0.395
216	0	1	0	0	0	0	1	1	0.278	0.379	0.339	0.415	0.494	0.461	0.394
217	1	0	0	1	0	0	1	1	0.266	0.334	0.393	0.366	0.511	0.487	0.393
218	0	1	0	0	1	1	1	1	0.498	0.379	0.46	0.501	-0.217	0.728	0.392
219	1	0	0	0	0	1	0	0	0.369	0.459	0.447	0.562	-0.179	0.69	0.391
220	1	1	0	1	0	0	1	1	0.276	0.398	0.398	0.359	0.465	0.45	0.391
221	1	1	0	0	0	0	0	0	0.44	0.447	0.442	0.382	0.073	0.562	0.391
222	0	0	0	1	1	1	0	0	0.327	0.408	0.545	0.599	-0.291	0.745	0.389
223	1	1	0	0	0	0	1	0	0.271	0.416	0.321	0.401	0.476	0.444	0.388
224	1	1	0	1	0	0	0	0	0.337	0.41	0.394	0.358	0.389	0.438	0.387
225	1	1	0	1	0	1	0	0	0.43	0.39	0.467	0.506	-0.026	0.549	0.386
226	1	1	0	1	0	0	1	0	0.308	0.361	0.361	0.39	0.445	0.426	0.382
227	1	1	0	0	1	0	0	0	0.504	0.46	0.411	0.462	-0.291	0.738	0.381
228	0	1	0	1	1	1	0	0	0.451	0.39	0.502	0.562	-0.237	0.61	0.38
229	0	1	0	1	0	1	0	1	0.394	0.371	0.455	0.506	0.033	0.517	0.379
230	0	1	0	0	0	1	1	1	0.406	0.379	0.359	0.482	0.14	0.481	0.375

Table A.13: Dataset 1 at Stage 20% - Gini Indices Part 5/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
231	1	1	0	1	1	1	0	0	0.438	0.403	0.49	0.525	-0.286	0.665	0.373
232	1	0	0	1	0	0	1	0	0.224	0.337	0.379	0.366	0.511	0.418	0.372
233	0	1	0	0	0	1	0	1	0.45	0.39	0.435	0.543	-0.225	0.633	0.371
234	1	0	0	1	0	0	0	0	0.197	0.327	0.423	0.297	0.487	0.493	0.371
235	1	1	0	0	1	1	1	0	0.473	0.398	0.442	0.494	-0.249	0.665	0.371
236	0	0	0	1	0	0	0	1	0.178	0.327	0.392	0.341	0.45	0.531	0.37
237	0	1	0	1	0	0	0	1	0.271	0.359	0.443	0.383	0.333	0.428	0.37
238	0	1	0	0	1	1	1	0	0.53	0.379	0.434	0.531	-0.249	0.579	0.367
239	0	1	0	1	0	0	1	0	0.321	0.31	0.37	0.408	0.494	0.293	0.366
240	1	1	0	0	0	1	0	0	0.509	0.39	0.447	0.518	-0.28	0.611	0.366
241	1	1	0	0	0	1	1	0	0.418	0.379	0.359	0.457	0.055	0.518	0.364
242	0	1	0	0	1	0	1	0	0.5	0.416	0.428	0.457	-0.194	0.578	0.364
243	0	0	0	1	0	1	0	0	0.277	0.445	0.522	0.593	-0.16	0.473	0.359
244	0	1	0	1	1	1	1	0	0.438	0.379	0.46	0.538	-0.23	0.556	0.357
245	0	0	0	1	0	0	1	1	0.249	0.3	0.379	0.371	0.511	0.317	0.354
246	0	1	0	0	0	1	1	0	0.461	0.371	0.419	0.556	-0.181	0.444	0.345
247	0	1	0	1	0	1	1	0	0.357	0.371	0.346	0.482	0.086	0.426	0.345
248	0	1	0	1	1	0	0	0	0.352	0.428	0.44	0.481	-0.274	0.634	0.344
249	0	0	0	1	0	0	1	0	0.194	0.319	0.346	0.371	0.571	0.249	0.342
250	0	1	0	0	0	0	1	0	0.321	0.385	0.351	0.42	0.28	0.289	0.341
251	0	0	0	0	0	0	1	0	0.281	0.258	0.311	0.27	0.689	0.227	0.339
252	0	1	0	1	0	1	0	0	0.443	0.377	0.455	0.543	-0.232	0.424	0.335
253	0	1	0	1	0	0	0	0	0.347	0.328	0.393	0.376	0.097	0.305	0.307
254	0	1	0	0	0	0	0	0	0.462	0.403	0.378	0.464	-0.273	0.407	0.307
255	0	0	0	1	0	0	0	0	0.087	0.297	0.389	0.352	0.431	0.253	0.302
256	0	0	0	0	0	0	0	0	0.049	0.042	0.016	0.042	-0.019	0.05	0.03

Table A.14: Dataset 1 at Stage 20% - Gini Indices Part 6/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
1	0	0	0	0	1	1	0	0	0.5	0.738	0.577	0.927	0.885	0.865	0.749
2	1	0	0	0	1	1	0	0	0.591	0.775	0.577	0.86	0.818	0.833	0.742
3	1	0	1	0	1	1	0	0	0.603	0.757	0.558	0.878	0.818	0.827	0.74
4	0	0	0	0	0	1	0	0	0.5	0.597	0.681	0.786	0.903	0.939	0.734
5	1	0	1	0	1	0	0	0	0.64	0.784	0.563	0.897	0.755	0.759	0.733
6	0	0	0	1	1	1	0	0	0.525	0.664	0.644	0.934	0.732	0.889	0.731
7	1	0	0	1	1	1	0	0	0.628	0.701	0.614	0.872	0.732	0.84	0.731
8	0	0	1	0	1	0	0	0	0.628	0.735	0.481	0.952	0.774	0.798	0.728
9	1	0	1	1	1	1	0	0	0.658	0.669	0.558	0.86	0.78	0.827	0.725
10	0	0	0	0	1	1	1	0	0.621	0.674	0.449	0.927	0.823	0.852	0.724
11	1	0	0	0	1	0	0	0	0.658	0.811	0.595	0.89	0.692	0.698	0.724
12	1	0	1	0	1	1	0	1	0.591	0.738	0.54	0.885	0.818	0.759	0.722
13	0	0	0	0	1	1	0	1	0.512	0.775	0.54	0.922	0.855	0.722	0.721
14	0	0	0	0	1	0	0	0	0.572	0.657	0.62	0.909	0.694	0.847	0.717
15	1	1	1	0	1	1	0	0	0.621	0.737	0.54	0.804	0.799	0.79	0.715
16	1	1	0	0	1	1	0	0	0.61	0.787	0.595	0.737	0.78	0.771	0.713
17	1	0	0	0	1	1	0	1	0.53	0.775	0.577	0.836	0.823	0.729	0.712
18	1	0	1	1	1	0	0	0	0.677	0.73	0.549	0.878	0.676	0.759	0.711
19	1	0	1	0	1	1	1	0	0.658	0.643	0.5	0.897	0.767	0.803	0.711
20	0	0	1	1	1	1	0	0	0.598	0.614	0.506	0.897	0.769	0.875	0.71
21	1	0	0	1	1	0	0	0	0.677	0.706	0.607	0.921	0.631	0.717	0.71
22	1	1	0	0	1	1	1	0	0.628	0.711	0.5	0.823	0.767	0.821	0.708
23	0	1	0	0	1	1	0	0	0.549	0.738	0.54	0.755	0.829	0.838	0.708
24	0	0	0	1	1	1	0	1	0.549	0.701	0.57	0.903	0.755	0.771	0.708
25	0	0	1	0	1	1	0	1	0.475	0.775	0.508	0.91	0.818	0.764	0.708
26	1	0	0	0	1	1	1	0	0.64	0.693	0.481	0.86	0.759	0.815	0.708
27	1	0	1	0	1	0	0	1	0.591	0.786	0.545	0.873	0.73	0.722	0.708
28	0	1	1	0	1	1	0	0	0.561	0.706	0.526	0.866	0.78	0.807	0.708
29	0	0	1	0	1	1	0	0	0.475	0.688	0.508	0.934	0.78	0.84	0.704
30	0	0	0	1	1	1	1	0	0.677	0.6	0.461	0.909	0.718	0.857	0.704
31	0	0	0	1	1	0	0	0	0.647	0.569	0.579	0.89	0.681	0.84	0.701
32	0	0	1	0	1	1	1	0	0.64	0.643	0.431	0.915	0.749	0.821	0.7
33	1	0	0	1	1	1	1	0	0.714	0.6	0.486	0.89	0.705	0.803	0.7
34	0	0	0	0	1	1	1	1	0.542	0.711	0.449	0.903	0.804	0.784	0.699
35	1	0	0	0	1	1	1	1	0.628	0.711	0.5	0.86	0.74	0.754	0.699
36	0	0	1	1	1	0	0	0	0.635	0.606	0.529	0.897	0.694	0.833	0.699
37	1	0	1	0	1	0	1	0	0.647	0.752	0.486	0.86	0.712	0.735	0.699
38	1	0	1	1	1	1	0	1	0.61	0.614	0.545	0.836	0.78	0.801	0.698
39	0	0	1	1	1	1	0	1	0.567	0.651	0.525	0.873	0.75	0.813	0.696
40	0	1	1	1	1	1	0	0	0.616	0.651	0.525	0.823	0.75	0.813	0.696
41	1	0	0	0	1	0	1	0	0.665	0.734	0.505	0.89	0.666	0.717	0.696
42	0	1	0	0	1	1	1	0	0.628	0.711	0.431	0.786	0.767	0.852	0.696
43	1	0	0	1	0	1	0	0	0.695	0.585	0.614	0.786	0.718	0.776	0.696
44	0	0	1	0	1	0	1	0	0.647	0.709	0.431	0.915	0.73	0.742	0.696
45	0	0	0	1	0	1	0	0	0.543	0.479	0.705	0.804	0.732	0.901	0.694
46	1	1	0	1	1	1	0	0	0.665	0.651	0.558	0.767	0.75	0.771	0.694

Table A.15: Dataset 1 at Stage 50% - Gini Indices Part 1/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
47	1	1	1	1	1	1	0	0	0.647	0.651	0.54	0.786	0.743	0.795	0.693
48	0	0	1	1	1	1	1	0	0.714	0.587	0.443	0.878	0.718	0.82	0.693
49	1	1	0	0	1	0	0	0	0.647	0.835	0.577	0.712	0.742	0.648	0.693
50	0	1	1	0	1	1	0	1	0.542	0.725	0.526	0.799	0.799	0.764	0.693
51	0	1	1	0	1	0	1	1	0.647	0.803	0.431	0.841	0.73	0.69	0.69
52	1	0	0	0	1	0	0	1	0.572	0.812	0.614	0.873	0.666	0.599	0.689
53	0	0	1	0	1	1	1	1	0.598	0.698	0.431	0.897	0.767	0.746	0.689
54	1	1	1	0	1	1	1	0	0.647	0.68	0.449	0.804	0.762	0.791	0.689
55	1	0	0	0	0	1	0	0	0.621	0.585	0.595	0.725	0.823	0.783	0.689
56	0	1	0	1	1	1	0	0	0.586	0.651	0.552	0.755	0.769	0.82	0.689
57	0	1	1	1	1	1	1	0	0.647	0.606	0.461	0.86	0.75	0.808	0.689
58	0	1	1	0	1	0	0	1	0.572	0.821	0.449	0.799	0.762	0.727	0.688
59	0	1	1	0	1	0	0	0	0.542	0.808	0.468	0.841	0.725	0.746	0.688
60	0	1	0	0	1	1	0	1	0.512	0.775	0.558	0.762	0.818	0.702	0.688
61	1	0	1	0	1	1	1	1	0.616	0.68	0.481	0.86	0.767	0.722	0.688
62	1	1	1	0	1	1	0	1	0.591	0.706	0.54	0.78	0.799	0.709	0.687
63	0	1	1	0	1	1	1	0	0.647	0.68	0.431	0.823	0.78	0.764	0.687
64	0	1	0	0	1	0	0	0	0.616	0.821	0.526	0.651	0.767	0.741	0.687
65	1	1	1	0	1	0	0	0	0.598	0.771	0.513	0.786	0.725	0.722	0.686
66	1	0	1	1	1	1	1	0	0.695	0.624	0.468	0.841	0.7	0.784	0.685
67	0	0	1	0	1	0	0	1	0.542	0.68	0.463	0.929	0.762	0.734	0.685
68	0	1	1	0	1	0	1	0	0.598	0.808	0.431	0.841	0.725	0.704	0.684
69	1	0	0	1	1	0	1	0	0.702	0.643	0.535	0.89	0.599	0.735	0.684
70	1	0	0	1	1	1	0	1	0.604	0.609	0.577	0.836	0.718	0.759	0.684
71	1	1	0	0	1	1	0	1	0.591	0.738	0.558	0.725	0.786	0.704	0.684
72	0	0	0	0	0	1	1	0	0.647	0.508	0.449	0.804	0.823	0.87	0.683
73	0	1	1	0	1	1	1	1	0.616	0.68	0.431	0.841	0.799	0.727	0.682
74	0	0	1	1	1	1	1	1	0.635	0.624	0.443	0.878	0.737	0.776	0.682
75	1	0	1	1	1	0	0	1	0.665	0.637	0.53	0.848	0.663	0.746	0.681
76	0	0	0	1	1	1	1	1	0.616	0.606	0.48	0.878	0.718	0.79	0.681
77	1	1	0	0	1	0	1	0	0.647	0.789	0.486	0.786	0.68	0.698	0.681
78	1	1	0	0	1	1	1	1	0.616	0.693	0.481	0.792	0.772	0.722	0.679
79	1	1	1	0	1	0	1	0	0.61	0.771	0.449	0.767	0.725	0.754	0.679
80	1	1	1	1	1	1	0	1	0.628	0.651	0.526	0.799	0.743	0.727	0.679
81	1	0	1	0	1	0	1	1	0.635	0.729	0.468	0.86	0.68	0.704	0.679
82	1	0	1	0	0	1	0	1	0.628	0.634	0.526	0.792	0.818	0.672	0.678
83	0	0	0	1	1	0	1	0	0.702	0.55	0.498	0.89	0.649	0.778	0.678
84	0	1	1	1	1	1	0	1	0.561	0.619	0.525	0.799	0.769	0.795	0.678
85	0	1	0	1	1	1	0	1	0.549	0.651	0.575	0.78	0.75	0.758	0.677
86	1	0	1	1	0	1	0	0	0.695	0.534	0.526	0.749	0.78	0.776	0.677
87	0	0	0	0	1	0	1	0	0.647	0.609	0.456	0.927	0.649	0.772	0.677
88	0	1	0	0	1	0	1	0	0.635	0.789	0.431	0.767	0.681	0.754	0.676
89	0	1	0	0	1	1	1	1	0.598	0.711	0.431	0.774	0.786	0.752	0.675
90	1	1	1	0	1	1	1	1	0.616	0.68	0.449	0.804	0.799	0.704	0.675
91	0	0	1	1	1	0	0	1	0.647	0.624	0.48	0.873	0.676	0.752	0.675
92	0	1	0	1	1	1	1	0	0.665	0.6	0.443	0.823	0.718	0.801	0.675

Table A.16: Dataset 1 at Stage 50% - Gini Indices Part 2/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
93	1	1	0	1	1	1	1	0	0.665	0.637	0.468	0.804	0.686	0.79	0.675
94	0	0	1	1	1	0	1	0	0.684	0.606	0.461	0.86	0.681	0.759	0.675
95	1	0	1	1	1	0	1	0	0.647	0.698	0.443	0.841	0.663	0.754	0.674
96	1	1	1	0	1	0	0	1	0.647	0.759	0.481	0.762	0.743	0.653	0.674
97	1	1	1	1	1	1	1	1	0.653	0.624	0.468	0.823	0.743	0.727	0.673
98	1	0	0	1	1	0	0	1	0.647	0.688	0.594	0.855	0.599	0.653	0.672
99	0	1	1	1	1	0	0	0	0.616	0.68	0.498	0.78	0.713	0.746	0.672
100	1	1	1	1	1	1	1	0	0.647	0.606	0.468	0.804	0.725	0.778	0.671
101	1	1	0	1	1	1	0	1	0.591	0.688	0.577	0.762	0.718	0.69	0.671
102	1	0	1	1	0	1	0	1	0.647	0.565	0.526	0.799	0.78	0.709	0.671
103	1	0	0	0	1	0	1	1	0.635	0.729	0.486	0.878	0.648	0.643	0.67
104	0	0	0	1	0	1	0	1	0.481	0.578	0.663	0.78	0.742	0.771	0.669
105	0	1	0	1	1	1	1	1	0.635	0.619	0.461	0.829	0.718	0.752	0.669
106	0	1	1	1	1	0	1	0	0.635	0.692	0.443	0.841	0.663	0.741	0.669
107	1	1	0	0	1	0	0	1	0.598	0.809	0.558	0.743	0.712	0.592	0.669
108	1	0	0	1	1	1	1	1	0.635	0.563	0.486	0.878	0.673	0.772	0.668
109	1	0	1	0	0	1	0	0	0.64	0.585	0.521	0.718	0.818	0.715	0.666
110	1	0	1	1	1	0	1	1	0.672	0.624	0.498	0.841	0.649	0.709	0.666
111	0	0	0	0	0	1	0	1	0.5	0.689	0.577	0.818	0.855	0.555	0.665
112	0	0	1	1	0	1	0	0	0.604	0.497	0.547	0.73	0.769	0.844	0.665
113	1	1	1	1	1	0	1	0	0.647	0.654	0.468	0.804	0.676	0.741	0.665
114	0	0	1	0	1	0	1	1	0.628	0.661	0.412	0.878	0.73	0.678	0.665
115	1	1	1	0	1	0	1	1	0.635	0.729	0.449	0.804	0.698	0.672	0.664
116	1	1	0	0	1	0	1	1	0.616	0.747	0.486	0.804	0.698	0.629	0.664
117	1	1	1	1	1	0	0	0	0.635	0.654	0.498	0.774	0.688	0.727	0.663
118	0	0	0	0	0	1	1	1	0.579	0.588	0.449	0.848	0.791	0.721	0.663
119	0	0	0	0	1	0	0	1	0.512	0.674	0.55	0.866	0.692	0.68	0.662
120	1	1	1	0	0	1	0	1	0.598	0.62	0.54	0.706	0.799	0.709	0.662
121	0	0	1	1	1	0	1	1	0.684	0.606	0.461	0.848	0.663	0.709	0.662
122	1	0	0	0	0	1	1	0	0.684	0.54	0.449	0.749	0.759	0.784	0.661
123	0	1	0	1	0	1	0	0	0.574	0.571	0.629	0.614	0.769	0.807	0.661
124	0	1	0	1	1	0	0	0	0.616	0.637	0.555	0.737	0.676	0.741	0.66
125	0	0	1	1	0	1	0	1	0.586	0.546	0.517	0.799	0.769	0.746	0.66
126	0	1	0	1	1	0	1	0	0.635	0.617	0.461	0.841	0.663	0.741	0.66
127	1	1	1	0	0	1	0	0	0.621	0.62	0.54	0.663	0.78	0.727	0.659
128	0	1	1	1	1	1	1	1	0.635	0.569	0.461	0.823	0.737	0.727	0.659
129	0	0	1	0	0	1	0	1	0.475	0.634	0.545	0.818	0.818	0.66	0.658
130	1	0	0	0	0	1	1	1	0.653	0.588	0.468	0.804	0.74	0.692	0.658
131	1	1	0	1	1	0	0	0	0.665	0.686	0.562	0.7	0.663	0.667	0.657
132	1	1	0	0	0	1	0	0	0.616	0.634	0.577	0.632	0.767	0.715	0.657
133	1	1	0	1	0	1	0	0	0.653	0.553	0.558	0.681	0.755	0.739	0.657
134	0	0	1	0	0	1	1	1	0.635	0.575	0.431	0.841	0.767	0.69	0.657
135	1	0	1	0	0	1	1	1	0.653	0.557	0.431	0.841	0.754	0.704	0.657
136	1	1	0	1	1	1	1	1	0.635	0.656	0.486	0.804	0.668	0.69	0.657
137	1	0	1	1	1	1	1	1	0.653	0.569	0.468	0.841	0.7	0.709	0.657
138	1	0	0	1	0	1	0	1	0.623	0.578	0.558	0.799	0.686	0.69	0.656

Table A.17: Dataset 1 at Stage 50% - Gini Indices Part 3/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
139	0	1	0	0	0	1	1	0	0.628	0.54	0.449	0.712	0.786	0.82	0.656
140	0	1	1	1	1	0	0	1	0.591	0.624	0.517	0.78	0.694	0.727	0.656
141	1	0	0	1	1	0	1	1	0.653	0.643	0.498	0.86	0.599	0.68	0.655
142	0	1	1	1	1	0	1	1	0.647	0.624	0.461	0.811	0.681	0.709	0.655
143	1	1	0	1	1	0	1	0	0.647	0.636	0.486	0.804	0.649	0.704	0.654
144	1	0	0	0	0	1	0	1	0.567	0.634	0.54	0.762	0.804	0.618	0.654
145	0	0	0	0	1	0	1	1	0.628	0.624	0.431	0.885	0.636	0.705	0.651
146	0	0	0	1	1	0	0	1	0.554	0.587	0.567	0.866	0.631	0.704	0.651
147	1	1	1	1	1	0	0	1	0.647	0.661	0.486	0.78	0.663	0.672	0.651
148	0	0	0	1	0	1	1	0	0.665	0.439	0.443	0.804	0.718	0.838	0.651
149	0	1	1	1	0	1	0	0	0.537	0.571	0.554	0.681	0.75	0.813	0.651
150	0	1	1	1	0	1	1	0	0.628	0.545	0.461	0.749	0.75	0.771	0.651
151	1	0	1	0	0	1	1	0	0.647	0.526	0.449	0.749	0.767	0.766	0.651
152	1	0	0	1	0	1	1	0	0.684	0.502	0.449	0.786	0.692	0.784	0.649
153	0	0	1	1	0	1	1	1	0.653	0.52	0.443	0.823	0.737	0.721	0.649
154	1	1	0	0	0	1	1	0	0.628	0.577	0.468	0.73	0.754	0.735	0.649
155	1	1	1	1	1	0	1	1	0.635	0.643	0.449	0.804	0.649	0.709	0.648
156	0	1	0	1	0	1	0	1	0.574	0.57	0.581	0.706	0.755	0.702	0.648
157	0	1	1	0	0	1	0	1	0.512	0.588	0.518	0.743	0.799	0.727	0.648
158	0	0	0	1	0	1	1	1	0.653	0.52	0.461	0.804	0.705	0.739	0.647
159	0	1	0	0	1	0	0	1	0.524	0.816	0.489	0.725	0.73	0.598	0.647
160	1	1	0	1	1	0	0	1	0.635	0.674	0.532	0.762	0.644	0.635	0.647
161	0	0	1	0	0	1	1	0	0.665	0.476	0.431	0.786	0.749	0.771	0.646
162	1	1	0	1	1	0	1	1	0.635	0.661	0.486	0.811	0.649	0.635	0.646
163	1	1	1	1	0	1	0	0	0.647	0.553	0.54	0.681	0.725	0.727	0.645
164	1	0	0	1	0	1	1	1	0.653	0.521	0.468	0.841	0.673	0.709	0.644
165	0	1	1	0	0	1	0	0	0.5	0.571	0.55	0.651	0.78	0.812	0.644
166	1	1	0	0	0	1	0	1	0.598	0.615	0.558	0.706	0.786	0.598	0.643
167	1	1	0	1	0	1	0	1	0.616	0.59	0.577	0.706	0.718	0.653	0.643
168	1	1	1	0	0	1	1	1	0.616	0.575	0.449	0.73	0.799	0.69	0.643
169	1	1	1	1	0	1	0	1	0.616	0.553	0.513	0.725	0.743	0.709	0.643
170	0	0	0	1	1	0	1	1	0.665	0.531	0.443	0.878	0.617	0.722	0.643
171	0	1	1	1	0	1	0	1	0.567	0.551	0.517	0.743	0.75	0.727	0.643
172	0	1	0	0	1	0	1	1	0.579	0.747	0.412	0.78	0.668	0.667	0.642
173	0	1	1	0	0	1	1	1	0.598	0.575	0.431	0.767	0.767	0.709	0.641
174	0	1	0	1	0	1	1	0	0.647	0.54	0.443	0.712	0.718	0.783	0.64
175	1	0	1	1	0	1	1	0	0.684	0.508	0.431	0.749	0.7	0.771	0.64
176	0	1	0	1	0	1	1	1	0.653	0.54	0.48	0.774	0.705	0.69	0.64
177	1	0	1	0	0	0	0	0	0.61	0.624	0.532	0.706	0.755	0.611	0.64
178	0	0	1	1	0	1	1	0	0.684	0.457	0.443	0.73	0.718	0.801	0.639
179	1	0	0	1	0	0	0	0	0.714	0.571	0.599	0.749	0.599	0.598	0.638
180	1	0	1	1	0	0	0	0	0.647	0.577	0.517	0.718	0.676	0.69	0.637
181	1	1	1	1	0	1	1	1	0.635	0.526	0.468	0.767	0.717	0.709	0.637
182	0	1	0	0	0	1	0	1	0.5	0.615	0.558	0.706	0.818	0.623	0.637
183	0	1	0	0	0	1	1	1	0.635	0.57	0.449	0.737	0.772	0.653	0.636
184	0	1	1	0	0	0	1	1	0.61	0.661	0.412	0.749	0.73	0.653	0.636

Table A.18: Dataset 1 at Stage 50% - Gini Indices Part 4/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
185	0	1	0	1	1	0	1	1	0.635	0.569	0.443	0.848	0.649	0.667	0.635
186	1	1	0	0	0	1	1	1	0.635	0.57	0.468	0.718	0.772	0.635	0.633
187	0	1	1	1	0	1	1	1	0.635	0.526	0.461	0.767	0.718	0.69	0.633
188	0	0	1	0	0	1	0	0	0.475	0.528	0.545	0.755	0.78	0.712	0.633
189	0	1	1	0	0	1	1	0	0.591	0.545	0.431	0.693	0.78	0.752	0.632
190	1	1	1	0	0	1	1	0	0.61	0.582	0.449	0.712	0.743	0.69	0.631
191	1	1	0	1	0	1	1	0	0.647	0.54	0.449	0.712	0.686	0.752	0.631
192	1	1	1	1	0	1	1	0	0.628	0.526	0.468	0.712	0.725	0.727	0.631
193	0	1	1	1	0	0	0	1	0.598	0.545	0.554	0.713	0.694	0.672	0.629
194	0	0	1	0	0	0	0	1	0.579	0.594	0.486	0.78	0.743	0.592	0.629
195	0	1	0	1	1	0	0	1	0.579	0.6	0.53	0.762	0.663	0.635	0.628
196	1	0	0	0	0	0	0	1	0.653	0.671	0.595	0.762	0.629	0.458	0.628
197	1	0	0	0	0	0	0	0	0.689	0.602	0.614	0.712	0.673	0.477	0.628
198	0	1	1	1	0	0	1	1	0.628	0.545	0.48	0.774	0.681	0.653	0.627
199	0	0	1	0	0	0	1	1	0.665	0.557	0.394	0.786	0.712	0.641	0.626
200	1	1	1	0	0	0	0	1	0.616	0.654	0.468	0.706	0.73	0.579	0.626
201	0	1	0	0	0	1	0	0	0.432	0.566	0.614	0.558	0.829	0.751	0.625
202	0	0	1	1	0	0	0	1	0.635	0.501	0.535	0.743	0.676	0.66	0.625
203	0	1	1	0	0	0	1	0	0.579	0.606	0.431	0.712	0.712	0.704	0.624
204	1	0	1	0	0	0	1	0	0.61	0.592	0.449	0.712	0.698	0.68	0.623
205	1	0	1	0	0	0	0	1	0.598	0.625	0.463	0.762	0.73	0.561	0.623
206	1	0	1	1	0	1	1	1	0.653	0.508	0.431	0.767	0.686	0.69	0.623
207	1	1	1	0	0	0	1	1	0.598	0.643	0.449	0.749	0.698	0.598	0.622
208	1	1	0	1	0	1	1	1	0.635	0.558	0.468	0.749	0.668	0.653	0.622
209	0	0	1	0	0	0	1	0	0.579	0.567	0.431	0.73	0.749	0.673	0.621
210	0	1	1	0	0	0	0	1	0.542	0.649	0.486	0.651	0.762	0.635	0.621
211	1	1	1	1	0	0	0	1	0.653	0.582	0.498	0.713	0.663	0.616	0.621
212	0	0	1	1	0	0	1	1	0.665	0.501	0.461	0.755	0.663	0.672	0.619
213	0	1	1	1	0	0	0	0	0.518	0.57	0.584	0.62	0.713	0.709	0.619
214	1	0	1	1	0	0	0	1	0.653	0.551	0.48	0.737	0.663	0.616	0.617
215	1	0	1	1	0	0	1	0	0.628	0.563	0.443	0.712	0.649	0.704	0.616
216	1	0	0	1	0	0	1	0	0.684	0.545	0.498	0.749	0.599	0.606	0.613
217	1	1	0	1	0	0	1	0	0.616	0.575	0.48	0.749	0.631	0.629	0.613
218	1	1	1	1	0	0	1	1	0.635	0.582	0.449	0.749	0.649	0.616	0.613
219	1	0	1	0	0	0	1	1	0.635	0.587	0.412	0.767	0.666	0.611	0.613
220	1	0	0	0	0	0	1	1	0.672	0.587	0.468	0.804	0.629	0.513	0.612
221	0	0	1	0	0	0	0	0	0.529	0.628	0.403	0.706	0.774	0.633	0.612
222	0	0	1	1	0	0	0	0	0.572	0.439	0.547	0.688	0.694	0.727	0.611
223	1	0	0	0	0	0	1	0	0.695	0.543	0.468	0.767	0.611	0.577	0.61
224	1	0	1	1	0	0	1	1	0.653	0.526	0.461	0.767	0.636	0.616	0.61
225	0	1	0	1	0	0	1	1	0.653	0.489	0.48	0.755	0.649	0.616	0.607
226	0	1	1	1	0	0	1	0	0.542	0.575	0.443	0.693	0.663	0.722	0.606
227	1	1	0	0	0	0	0	1	0.598	0.668	0.54	0.651	0.693	0.487	0.606
228	1	1	0	1	0	0	0	1	0.653	0.595	0.525	0.651	0.644	0.561	0.605
229	1	0	0	1	0	0	0	1	0.635	0.583	0.525	0.762	0.58	0.542	0.605
230	1	1	1	0	0	0	1	0	0.561	0.599	0.449	0.693	0.725	0.598	0.604

Table A.19: Dataset 1 at Stage 50% - Gini Indices Part 5/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
231	1	0	0	1	0	0	1	1	0.672	0.526	0.461	0.786	0.599	0.579	0.604
232	0	0	0	1	0	0	1	1	0.684	0.445	0.443	0.792	0.617	0.635	0.603
233	1	1	0	0	0	0	1	0	0.579	0.617	0.468	0.675	0.661	0.606	0.601
234	1	1	0	0	0	0	0	0	0.579	0.644	0.577	0.558	0.705	0.537	0.6
235	0	0	1	1	0	0	1	0	0.635	0.407	0.443	0.712	0.681	0.722	0.6
236	1	1	0	1	0	0	1	1	0.635	0.563	0.468	0.737	0.631	0.561	0.599
237	1	1	0	0	0	0	1	1	0.635	0.624	0.431	0.712	0.685	0.505	0.598
238	1	1	1	0	0	0	0	0	0.549	0.599	0.486	0.614	0.725	0.616	0.598
239	0	0	0	1	0	0	0	1	0.609	0.464	0.535	0.75	0.599	0.629	0.598
240	1	1	1	1	0	0	0	0	0.549	0.575	0.48	0.651	0.688	0.635	0.596
241	1	1	1	1	0	0	1	0	0.579	0.575	0.461	0.675	0.649	0.635	0.596
242	0	0	0	1	0	0	0	0	0.592	0.37	0.559	0.681	0.594	0.778	0.596
243	0	0	0	0	0	0	1	1	0.684	0.501	0.375	0.811	0.636	0.567	0.596
244	0	1	0	1	0	0	0	1	0.598	0.509	0.572	0.669	0.644	0.579	0.595
245	1	1	0	1	0	0	0	0	0.586	0.588	0.543	0.602	0.644	0.598	0.594
246	0	0	0	1	0	0	1	0	0.69	0.37	0.443	0.712	0.617	0.704	0.589
247	0	0	0	0	0	0	1	0	0.704	0.382	0.382	0.73	0.592	0.741	0.588
248	0	1	0	1	0	0	1	0	0.586	0.469	0.443	0.693	0.649	0.667	0.584
249	0	1	0	0	0	0	1	1	0.598	0.606	0.394	0.706	0.668	0.524	0.582
250	0	1	0	0	0	0	1	0	0.599	0.557	0.38	0.656	0.668	0.611	0.578
251	0	1	0	1	0	0	0	0	0.525	0.465	0.598	0.546	0.663	0.653	0.575
252	0	1	0	0	0	0	0	1	0.524	0.632	0.486	0.595	0.749	0.444	0.572
253	0	1	1	0	0	0	0	0	0.518	0.487	0.449	0.521	0.725	0.695	0.566
254	0	0	0	0	0	0	0	1	0.554	0.588	0.468	0.762	0.673	0.309	0.559
255	0	1	0	0	0	0	0	0	0.432	0.456	0.449	0.405	0.767	0.524	0.505
256	0	0	0	0	0	0	0	0	0.049	0.042	0.016	0.042	-0.019	0.05	0.03

Table A.20: Dataset 1 at Stage 50% - Gini Indices Part 6/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
1	0	0	0	0	1	1	0	0	0.5	0.738	0.577	0.927	0.885	0.865	0.749
2	1	0	0	0	1	1	0	0	0.591	0.775	0.577	0.86	0.818	0.833	0.742
3	1	0	1	0	1	1	0	0	0.603	0.757	0.558	0.878	0.818	0.827	0.74
4	0	0	0	0	0	1	0	0	0.5	0.597	0.681	0.786	0.903	0.939	0.734
5	1	0	1	0	1	0	0	0	0.64	0.784	0.563	0.897	0.755	0.759	0.733
6	0	0	0	1	1	1	0	0	0.525	0.664	0.644	0.934	0.732	0.889	0.731
7	1	0	0	1	1	1	0	0	0.628	0.701	0.614	0.872	0.732	0.84	0.731
8	0	0	1	0	1	0	0	0	0.628	0.735	0.481	0.952	0.774	0.798	0.728
9	1	0	1	1	1	1	0	0	0.658	0.669	0.558	0.86	0.78	0.827	0.725
10	0	0	0	0	1	1	1	0	0.621	0.674	0.449	0.927	0.823	0.852	0.724
11	1	0	0	0	1	0	0	0	0.658	0.811	0.595	0.89	0.692	0.698	0.724
12	1	0	1	0	1	1	0	1	0.591	0.738	0.54	0.885	0.818	0.759	0.722
13	0	0	0	0	1	1	0	1	0.512	0.775	0.54	0.922	0.855	0.722	0.721
14	0	0	0	0	1	0	0	0	0.572	0.657	0.62	0.909	0.694	0.847	0.717
15	1	1	1	0	1	1	0	0	0.621	0.737	0.54	0.804	0.799	0.79	0.715
16	1	1	0	0	1	1	0	0	0.61	0.787	0.595	0.737	0.78	0.771	0.713
17	1	0	0	0	1	1	0	1	0.53	0.775	0.577	0.836	0.823	0.729	0.712
18	1	0	1	1	1	0	0	0	0.677	0.73	0.549	0.878	0.676	0.759	0.711
19	1	0	1	0	1	1	1	0	0.658	0.643	0.5	0.897	0.767	0.803	0.711
20	0	0	1	1	1	1	0	0	0.598	0.614	0.506	0.897	0.769	0.875	0.71
21	1	0	0	1	1	0	0	0	0.677	0.706	0.607	0.921	0.631	0.717	0.71
22	1	1	0	0	1	1	1	0	0.628	0.711	0.5	0.823	0.767	0.821	0.708
23	0	1	0	0	1	1	0	0	0.549	0.738	0.54	0.755	0.829	0.838	0.708
24	0	0	0	1	1	1	0	1	0.549	0.701	0.57	0.903	0.755	0.771	0.708
25	0	0	1	0	1	1	0	1	0.475	0.775	0.508	0.91	0.818	0.764	0.708
26	1	0	0	0	1	1	1	0	0.64	0.693	0.481	0.86	0.759	0.815	0.708
27	1	0	1	0	1	0	0	1	0.591	0.786	0.545	0.873	0.73	0.722	0.708
28	0	1	1	0	1	1	0	0	0.561	0.706	0.526	0.866	0.78	0.807	0.708
29	0	0	1	0	1	1	0	0	0.475	0.688	0.508	0.934	0.78	0.84	0.704
30	0	0	0	1	1	1	1	0	0.677	0.6	0.461	0.909	0.718	0.857	0.704
31	0	0	0	1	1	0	0	0	0.647	0.569	0.579	0.89	0.681	0.84	0.701
32	0	0	1	0	1	1	1	0	0.64	0.643	0.431	0.915	0.749	0.821	0.7
33	1	0	0	1	1	1	1	0	0.714	0.6	0.486	0.89	0.705	0.803	0.7
34	0	0	0	0	1	1	1	1	0.542	0.711	0.449	0.903	0.804	0.784	0.699
35	1	0	0	0	1	1	1	1	0.628	0.711	0.5	0.86	0.74	0.754	0.699
36	0	0	1	1	1	0	0	0	0.635	0.606	0.529	0.897	0.694	0.833	0.699
37	1	0	1	0	1	0	1	0	0.647	0.752	0.486	0.86	0.712	0.735	0.699
38	1	0	1	1	1	1	0	1	0.61	0.614	0.545	0.836	0.78	0.801	0.698
39	0	0	1	1	1	1	0	1	0.567	0.651	0.525	0.873	0.75	0.813	0.696
40	0	1	1	1	1	1	0	0	0.616	0.651	0.525	0.823	0.75	0.813	0.696
41	1	0	0	0	1	0	1	0	0.665	0.734	0.505	0.89	0.666	0.717	0.696
42	0	1	0	0	1	1	1	0	0.628	0.711	0.431	0.786	0.767	0.852	0.696
43	1	0	0	1	0	1	0	0	0.695	0.585	0.614	0.786	0.718	0.776	0.696
44	0	0	1	0	1	0	1	0	0.647	0.709	0.431	0.915	0.73	0.742	0.696
45	0	0	0	1	0	1	0	0	0.543	0.479	0.705	0.804	0.732	0.901	0.694
46	1	1	0	1	1	1	0	0	0.665	0.651	0.558	0.767	0.75	0.771	0.694

Table A.21: Dataset 2 at Stage 20% - Gini Indices Part 1/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
47	1	1	1	1	1	1	0	0	0.647	0.651	0.54	0.786	0.743	0.795	0.693
48	0	0	1	1	1	1	1	0	0.714	0.587	0.443	0.878	0.718	0.82	0.693
49	1	1	0	0	1	0	0	0	0.647	0.835	0.577	0.712	0.742	0.648	0.693
50	0	1	1	0	1	1	0	1	0.542	0.725	0.526	0.799	0.799	0.764	0.693
51	0	1	1	0	1	0	1	1	0.647	0.803	0.431	0.841	0.73	0.69	0.69
52	1	0	0	0	1	0	0	1	0.572	0.812	0.614	0.873	0.666	0.599	0.689
53	0	0	1	0	1	1	1	1	0.598	0.698	0.431	0.897	0.767	0.746	0.689
54	1	1	1	0	1	1	1	0	0.647	0.68	0.449	0.804	0.762	0.791	0.689
55	1	0	0	0	0	1	0	0	0.621	0.585	0.595	0.725	0.823	0.783	0.689
56	0	1	0	1	1	1	0	0	0.586	0.651	0.552	0.755	0.769	0.82	0.689
57	0	1	1	1	1	1	1	0	0.647	0.606	0.461	0.86	0.75	0.808	0.689
58	0	1	1	0	1	0	0	1	0.572	0.821	0.449	0.799	0.762	0.727	0.688
59	0	1	1	0	1	0	0	0	0.542	0.808	0.468	0.841	0.725	0.746	0.688
60	0	1	0	0	1	1	0	1	0.512	0.775	0.558	0.762	0.818	0.702	0.688
61	1	0	1	0	1	1	1	1	0.616	0.68	0.481	0.86	0.767	0.722	0.688
62	1	1	1	0	1	1	0	1	0.591	0.706	0.54	0.78	0.799	0.709	0.687
63	0	1	1	0	1	1	1	0	0.647	0.68	0.431	0.823	0.78	0.764	0.687
64	0	1	0	0	1	0	0	0	0.616	0.821	0.526	0.651	0.767	0.741	0.687
65	1	1	1	0	1	0	0	0	0.598	0.771	0.513	0.786	0.725	0.722	0.686
66	1	0	1	1	1	1	1	0	0.695	0.624	0.468	0.841	0.7	0.784	0.685
67	0	0	1	0	1	0	0	1	0.542	0.68	0.463	0.929	0.762	0.734	0.685
68	0	1	1	0	1	0	1	0	0.598	0.808	0.431	0.841	0.725	0.704	0.684
69	1	0	0	1	1	0	1	0	0.702	0.643	0.535	0.89	0.599	0.735	0.684
70	1	0	0	1	1	1	0	1	0.604	0.609	0.577	0.836	0.718	0.759	0.684
71	1	1	0	0	1	1	0	1	0.591	0.738	0.558	0.725	0.786	0.704	0.684
72	0	0	0	0	0	1	1	0	0.647	0.508	0.449	0.804	0.823	0.87	0.683
73	0	1	1	0	1	1	1	1	0.616	0.68	0.431	0.841	0.799	0.727	0.682
74	0	0	1	1	1	1	1	1	0.635	0.624	0.443	0.878	0.737	0.776	0.682
75	1	0	1	1	1	0	0	1	0.665	0.637	0.53	0.848	0.663	0.746	0.681
76	0	0	0	1	1	1	1	1	0.616	0.606	0.48	0.878	0.718	0.79	0.681
77	1	1	0	0	1	0	1	0	0.647	0.789	0.486	0.786	0.68	0.698	0.681
78	1	1	0	0	1	1	1	1	0.616	0.693	0.481	0.792	0.772	0.722	0.679
79	1	1	1	0	1	0	1	0	0.61	0.771	0.449	0.767	0.725	0.754	0.679
80	1	1	1	1	1	1	0	1	0.628	0.651	0.526	0.799	0.743	0.727	0.679
81	1	0	1	0	1	0	1	1	0.635	0.729	0.468	0.86	0.68	0.704	0.679
82	1	0	1	0	0	1	0	1	0.628	0.634	0.526	0.792	0.818	0.672	0.678
83	0	0	0	1	1	0	1	0	0.702	0.55	0.498	0.89	0.649	0.778	0.678
84	0	1	1	1	1	1	0	1	0.561	0.619	0.525	0.799	0.769	0.795	0.678
85	0	1	0	1	1	1	0	1	0.549	0.651	0.575	0.78	0.75	0.758	0.677
86	1	0	1	1	0	1	0	0	0.695	0.534	0.526	0.749	0.78	0.776	0.677
87	0	0	0	0	1	0	1	0	0.647	0.609	0.456	0.927	0.649	0.772	0.677
88	0	1	0	0	1	0	1	0	0.635	0.789	0.431	0.767	0.681	0.754	0.676
89	0	1	0	0	1	1	1	1	0.598	0.711	0.431	0.774	0.786	0.752	0.675
90	1	1	1	0	1	1	1	1	0.616	0.68	0.449	0.804	0.799	0.704	0.675
91	0	0	1	1	1	0	0	1	0.647	0.624	0.48	0.873	0.676	0.752	0.675
92	0	1	0	1	1	1	1	0	0.665	0.6	0.443	0.823	0.718	0.801	0.675

Table A.22: Dataset 2 at Stage 20% - Gini Indices Part 2/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
93	1	1	0	1	1	1	1	0	0.665	0.637	0.468	0.804	0.686	0.79	0.675
94	0	0	1	1	1	0	1	0	0.684	0.606	0.461	0.86	0.681	0.759	0.675
95	1	0	1	1	1	0	1	0	0.647	0.698	0.443	0.841	0.663	0.754	0.674
96	1	1	1	0	1	0	0	1	0.647	0.759	0.481	0.762	0.743	0.653	0.674
97	1	1	1	1	1	1	1	1	0.653	0.624	0.468	0.823	0.743	0.727	0.673
98	1	0	0	1	1	0	0	1	0.647	0.688	0.594	0.855	0.599	0.653	0.672
99	0	1	1	1	1	0	0	0	0.616	0.68	0.498	0.78	0.713	0.746	0.672
100	1	1	1	1	1	1	1	0	0.647	0.606	0.468	0.804	0.725	0.778	0.671
101	1	1	0	1	1	1	0	1	0.591	0.688	0.577	0.762	0.718	0.69	0.671
102	1	0	1	1	0	1	0	1	0.647	0.565	0.526	0.799	0.78	0.709	0.671
103	1	0	0	0	1	0	1	1	0.635	0.729	0.486	0.878	0.648	0.643	0.67
104	0	0	0	1	0	1	0	1	0.481	0.578	0.663	0.78	0.742	0.771	0.669
105	0	1	0	1	1	1	1	1	0.635	0.619	0.461	0.829	0.718	0.752	0.669
106	0	1	1	1	1	0	1	0	0.635	0.692	0.443	0.841	0.663	0.741	0.669
107	1	1	0	0	1	0	0	1	0.598	0.809	0.558	0.743	0.712	0.592	0.669
108	1	0	0	1	1	1	1	1	0.635	0.563	0.486	0.878	0.673	0.772	0.668
109	1	0	1	0	0	1	0	0	0.64	0.585	0.521	0.718	0.818	0.715	0.666
110	1	0	1	1	1	0	1	1	0.672	0.624	0.498	0.841	0.649	0.709	0.666
111	0	0	0	0	0	1	0	1	0.5	0.689	0.577	0.818	0.855	0.555	0.665
112	0	0	1	1	0	1	0	0	0.604	0.497	0.547	0.73	0.769	0.844	0.665
113	1	1	1	1	1	0	1	0	0.647	0.654	0.468	0.804	0.676	0.741	0.665
114	0	0	1	0	1	0	1	1	0.628	0.661	0.412	0.878	0.73	0.678	0.665
115	1	1	1	0	1	0	1	1	0.635	0.729	0.449	0.804	0.698	0.672	0.664
116	1	1	0	0	1	0	1	1	0.616	0.747	0.486	0.804	0.698	0.629	0.664
117	1	1	1	1	1	0	0	0	0.635	0.654	0.498	0.774	0.688	0.727	0.663
118	0	0	0	0	0	1	1	1	0.579	0.588	0.449	0.848	0.791	0.721	0.663
119	0	0	0	0	1	0	0	1	0.512	0.674	0.55	0.866	0.692	0.68	0.662
120	1	1	1	0	0	1	0	1	0.598	0.62	0.54	0.706	0.799	0.709	0.662
121	0	0	1	1	1	0	1	1	0.684	0.606	0.461	0.848	0.663	0.709	0.662
122	1	0	0	0	0	1	1	0	0.684	0.54	0.449	0.749	0.759	0.784	0.661
123	0	1	0	1	0	1	0	0	0.574	0.571	0.629	0.614	0.769	0.807	0.661
124	0	1	0	1	1	0	0	0	0.616	0.637	0.555	0.737	0.676	0.741	0.66
125	0	0	1	1	0	1	0	1	0.586	0.546	0.517	0.799	0.769	0.746	0.66
126	0	1	0	1	1	0	1	0	0.635	0.617	0.461	0.841	0.663	0.741	0.66
127	1	1	1	0	0	1	0	0	0.621	0.62	0.54	0.663	0.78	0.727	0.659
128	0	1	1	1	1	1	1	1	0.635	0.569	0.461	0.823	0.737	0.727	0.659
129	0	0	1	0	0	1	0	1	0.475	0.634	0.545	0.818	0.818	0.66	0.658
130	1	0	0	0	0	1	1	1	0.653	0.588	0.468	0.804	0.74	0.692	0.658
131	1	1	0	1	1	0	0	0	0.665	0.686	0.562	0.7	0.663	0.667	0.657
132	1	1	0	0	0	1	0	0	0.616	0.634	0.577	0.632	0.767	0.715	0.657
133	1	1	0	1	0	1	0	0	0.653	0.553	0.558	0.681	0.755	0.739	0.657
134	0	0	1	0	0	1	1	1	0.635	0.575	0.431	0.841	0.767	0.69	0.657
135	1	0	1	0	0	1	1	1	0.653	0.557	0.431	0.841	0.754	0.704	0.657
136	1	1	0	1	1	1	1	1	0.635	0.656	0.486	0.804	0.668	0.69	0.657
137	1	0	1	1	1	1	1	1	0.653	0.569	0.468	0.841	0.7	0.709	0.657
138	1	0	0	1	0	1	0	1	0.623	0.578	0.558	0.799	0.686	0.69	0.656

Table A.23: Dataset 2 at Stage 20% - Gini Indices Part 3/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
139	0	1	0	0	0	1	1	0	0.628	0.54	0.449	0.712	0.786	0.82	0.656
140	0	1	1	1	1	0	0	1	0.591	0.624	0.517	0.78	0.694	0.727	0.656
141	1	0	0	1	1	0	1	1	0.653	0.643	0.498	0.86	0.599	0.68	0.655
142	0	1	1	1	1	0	1	1	0.647	0.624	0.461	0.811	0.681	0.709	0.655
143	1	1	0	1	1	0	1	0	0.647	0.636	0.486	0.804	0.649	0.704	0.654
144	1	0	0	0	0	1	0	1	0.567	0.634	0.54	0.762	0.804	0.618	0.654
145	0	0	0	0	1	0	1	1	0.628	0.624	0.431	0.885	0.636	0.705	0.651
146	0	0	0	1	1	0	0	1	0.554	0.587	0.567	0.866	0.631	0.704	0.651
147	1	1	1	1	1	0	0	1	0.647	0.661	0.486	0.78	0.663	0.672	0.651
148	0	0	0	1	0	1	1	0	0.665	0.439	0.443	0.804	0.718	0.838	0.651
149	0	1	1	1	0	1	0	0	0.537	0.571	0.554	0.681	0.75	0.813	0.651
150	0	1	1	1	0	1	1	0	0.628	0.545	0.461	0.749	0.75	0.771	0.651
151	1	0	1	0	0	1	1	0	0.647	0.526	0.449	0.749	0.767	0.766	0.651
152	1	0	0	1	0	1	1	0	0.684	0.502	0.449	0.786	0.692	0.784	0.649
153	0	0	1	1	0	1	1	1	0.653	0.52	0.443	0.823	0.737	0.721	0.649
154	1	1	0	0	0	1	1	0	0.628	0.577	0.468	0.73	0.754	0.735	0.649
155	1	1	1	1	1	0	1	1	0.635	0.643	0.449	0.804	0.649	0.709	0.648
156	0	1	0	1	0	1	0	1	0.574	0.57	0.581	0.706	0.755	0.702	0.648
157	0	1	1	0	0	1	0	1	0.512	0.588	0.518	0.743	0.799	0.727	0.648
158	0	0	0	1	0	1	1	1	0.653	0.52	0.461	0.804	0.705	0.739	0.647
159	0	1	0	0	1	0	0	1	0.524	0.816	0.489	0.725	0.73	0.598	0.647
160	1	1	0	1	1	0	0	1	0.635	0.674	0.532	0.762	0.644	0.635	0.647
161	0	0	1	0	0	1	1	0	0.665	0.476	0.431	0.786	0.749	0.771	0.646
162	1	1	0	1	1	0	1	1	0.635	0.661	0.486	0.811	0.649	0.635	0.646
163	1	1	1	1	0	1	0	0	0.647	0.553	0.54	0.681	0.725	0.727	0.645
164	1	0	0	1	0	1	1	1	0.653	0.521	0.468	0.841	0.673	0.709	0.644
165	0	1	1	0	0	1	0	0	0.5	0.571	0.55	0.651	0.78	0.812	0.644
166	1	1	0	0	0	1	0	1	0.598	0.615	0.558	0.706	0.786	0.598	0.643
167	1	1	0	1	0	1	0	1	0.616	0.59	0.577	0.706	0.718	0.653	0.643
168	1	1	1	0	0	1	1	1	0.616	0.575	0.449	0.73	0.799	0.69	0.643
169	1	1	1	1	0	1	0	1	0.616	0.553	0.513	0.725	0.743	0.709	0.643
170	0	0	0	1	1	0	1	1	0.665	0.531	0.443	0.878	0.617	0.722	0.643
171	0	1	1	1	0	1	0	1	0.567	0.551	0.517	0.743	0.75	0.727	0.643
172	0	1	0	0	1	0	1	1	0.579	0.747	0.412	0.78	0.668	0.667	0.642
173	0	1	1	0	0	1	1	1	0.598	0.575	0.431	0.767	0.767	0.709	0.641
174	0	1	0	1	0	1	1	0	0.647	0.54	0.443	0.712	0.718	0.783	0.64
175	1	0	1	1	0	1	1	0	0.684	0.508	0.431	0.749	0.7	0.771	0.64
176	0	1	0	1	0	1	1	1	0.653	0.54	0.48	0.774	0.705	0.69	0.64
177	1	0	1	0	0	0	0	0	0.61	0.624	0.532	0.706	0.755	0.611	0.64
178	0	0	1	1	0	1	1	0	0.684	0.457	0.443	0.73	0.718	0.801	0.639
179	1	0	0	1	0	0	0	0	0.714	0.571	0.599	0.749	0.599	0.598	0.638
180	1	0	1	1	0	0	0	0	0.647	0.577	0.517	0.718	0.676	0.69	0.637
181	1	1	1	1	0	1	1	1	0.635	0.526	0.468	0.767	0.717	0.709	0.637
182	0	1	0	0	0	1	0	1	0.5	0.615	0.558	0.706	0.818	0.623	0.637
183	0	1	0	0	0	1	1	1	0.635	0.57	0.449	0.737	0.772	0.653	0.636
184	0	1	1	0	0	0	1	1	0.61	0.661	0.412	0.749	0.73	0.653	0.636

Table A.24: Dataset 2 at Stage 20% - Gini Indices Part 4/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
185	0	1	0	1	1	0	1	1	0.635	0.569	0.443	0.848	0.649	0.667	0.635
186	1	1	0	0	0	1	1	1	0.635	0.57	0.468	0.718	0.772	0.635	0.633
187	0	1	1	1	0	1	1	1	0.635	0.526	0.461	0.767	0.718	0.69	0.633
188	0	0	1	0	0	1	0	0	0.475	0.528	0.545	0.755	0.78	0.712	0.633
189	0	1	1	0	0	1	1	0	0.591	0.545	0.431	0.693	0.78	0.752	0.632
190	1	1	1	0	0	1	1	0	0.61	0.582	0.449	0.712	0.743	0.69	0.631
191	1	1	0	1	0	1	1	0	0.647	0.54	0.449	0.712	0.686	0.752	0.631
192	1	1	1	1	0	1	1	0	0.628	0.526	0.468	0.712	0.725	0.727	0.631
193	0	1	1	1	0	0	0	1	0.598	0.545	0.554	0.713	0.694	0.672	0.629
194	0	0	1	0	0	0	0	1	0.579	0.594	0.486	0.78	0.743	0.592	0.629
195	0	1	0	1	1	0	0	1	0.579	0.6	0.53	0.762	0.663	0.635	0.628
196	1	0	0	0	0	0	0	1	0.653	0.671	0.595	0.762	0.629	0.458	0.628
197	1	0	0	0	0	0	0	0	0.689	0.602	0.614	0.712	0.673	0.477	0.628
198	0	1	1	1	0	0	1	1	0.628	0.545	0.48	0.774	0.681	0.653	0.627
199	0	0	1	0	0	0	1	1	0.665	0.557	0.394	0.786	0.712	0.641	0.626
200	1	1	1	0	0	0	0	1	0.616	0.654	0.468	0.706	0.73	0.579	0.626
201	0	1	0	0	0	1	0	0	0.432	0.566	0.614	0.558	0.829	0.751	0.625
202	0	0	1	1	0	0	0	1	0.635	0.501	0.535	0.743	0.676	0.66	0.625
203	0	1	1	0	0	0	1	0	0.579	0.606	0.431	0.712	0.712	0.704	0.624
204	1	0	1	0	0	0	1	0	0.61	0.592	0.449	0.712	0.698	0.68	0.623
205	1	0	1	0	0	0	0	1	0.598	0.625	0.463	0.762	0.73	0.561	0.623
206	1	0	1	1	0	1	1	1	0.653	0.508	0.431	0.767	0.686	0.69	0.623
207	1	1	1	0	0	0	1	1	0.598	0.643	0.449	0.749	0.698	0.598	0.622
208	1	1	0	1	0	1	1	1	0.635	0.558	0.468	0.749	0.668	0.653	0.622
209	0	0	1	0	0	0	1	0	0.579	0.567	0.431	0.73	0.749	0.673	0.621
210	0	1	1	0	0	0	0	1	0.542	0.649	0.486	0.651	0.762	0.635	0.621
211	1	1	1	1	0	0	0	1	0.653	0.582	0.498	0.713	0.663	0.616	0.621
212	0	0	1	1	0	0	1	1	0.665	0.501	0.461	0.755	0.663	0.672	0.619
213	0	1	1	1	0	0	0	0	0.518	0.57	0.584	0.62	0.713	0.709	0.619
214	1	0	1	1	0	0	0	1	0.653	0.551	0.48	0.737	0.663	0.616	0.617
215	1	0	1	1	0	0	1	0	0.628	0.563	0.443	0.712	0.649	0.704	0.616
216	1	0	0	1	0	0	1	0	0.684	0.545	0.498	0.749	0.599	0.606	0.613
217	1	1	0	1	0	0	1	0	0.616	0.575	0.48	0.749	0.631	0.629	0.613
218	1	1	1	1	0	0	1	1	0.635	0.582	0.449	0.749	0.649	0.616	0.613
219	1	0	1	0	0	0	1	1	0.635	0.587	0.412	0.767	0.666	0.611	0.613
220	1	0	0	0	0	0	1	1	0.672	0.587	0.468	0.804	0.629	0.513	0.612
221	0	0	1	0	0	0	0	0	0.529	0.628	0.403	0.706	0.774	0.633	0.612
222	0	0	1	1	0	0	0	0	0.572	0.439	0.547	0.688	0.694	0.727	0.611
223	1	0	0	0	0	0	1	0	0.695	0.543	0.468	0.767	0.611	0.577	0.61
224	1	0	1	1	0	0	1	1	0.653	0.526	0.461	0.767	0.636	0.616	0.61
225	0	1	0	1	0	0	1	1	0.653	0.489	0.48	0.755	0.649	0.616	0.607
226	0	1	1	1	0	0	1	0	0.542	0.575	0.443	0.693	0.663	0.722	0.606
227	1	1	0	0	0	0	0	1	0.598	0.668	0.54	0.651	0.693	0.487	0.606
228	1	1	0	1	0	0	0	1	0.653	0.595	0.525	0.651	0.644	0.561	0.605
229	1	0	0	1	0	0	0	1	0.635	0.583	0.525	0.762	0.58	0.542	0.605
230	1	1	1	0	0	0	1	0	0.561	0.599	0.449	0.693	0.725	0.598	0.604

Table A.25: Dataset 2 at Stage 20% - Gini Indices Part 5/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
231	1	0	0	1	0	0	1	1	0.672	0.526	0.461	0.786	0.599	0.579	0.604
232	0	0	0	1	0	0	1	1	0.684	0.445	0.443	0.792	0.617	0.635	0.603
233	1	1	0	0	0	0	1	0	0.579	0.617	0.468	0.675	0.661	0.606	0.601
234	1	1	0	0	0	0	0	0	0.579	0.644	0.577	0.558	0.705	0.537	0.6
235	0	0	1	1	0	0	1	0	0.635	0.407	0.443	0.712	0.681	0.722	0.6
236	1	1	0	1	0	0	1	1	0.635	0.563	0.468	0.737	0.631	0.561	0.599
237	1	1	0	0	0	0	1	1	0.635	0.624	0.431	0.712	0.685	0.505	0.598
238	1	1	1	0	0	0	0	0	0.549	0.599	0.486	0.614	0.725	0.616	0.598
239	0	0	0	1	0	0	0	1	0.609	0.464	0.535	0.75	0.599	0.629	0.598
240	1	1	1	1	0	0	0	0	0.549	0.575	0.48	0.651	0.688	0.635	0.596
241	1	1	1	1	0	0	1	0	0.579	0.575	0.461	0.675	0.649	0.635	0.596
242	0	0	0	1	0	0	0	0	0.592	0.37	0.559	0.681	0.594	0.778	0.596
243	0	0	0	0	0	0	1	1	0.684	0.501	0.375	0.811	0.636	0.567	0.596
244	0	1	0	1	0	0	0	1	0.598	0.509	0.572	0.669	0.644	0.579	0.595
245	1	1	0	1	0	0	0	0	0.586	0.588	0.543	0.602	0.644	0.598	0.594
246	0	0	0	1	0	0	1	0	0.69	0.37	0.443	0.712	0.617	0.704	0.589
247	0	0	0	0	0	0	1	0	0.704	0.382	0.382	0.73	0.592	0.741	0.588
248	0	1	0	1	0	0	1	0	0.586	0.469	0.443	0.693	0.649	0.667	0.584
249	0	1	0	0	0	0	1	1	0.598	0.606	0.394	0.706	0.668	0.524	0.582
250	0	1	0	0	0	0	1	0	0.599	0.557	0.38	0.656	0.668	0.611	0.578
251	0	1	0	1	0	0	0	0	0.525	0.465	0.598	0.546	0.663	0.653	0.575
252	0	1	0	0	0	0	0	1	0.524	0.632	0.486	0.595	0.749	0.444	0.572
253	0	1	1	0	0	0	0	0	0.518	0.487	0.449	0.521	0.725	0.695	0.566
254	0	0	0	0	0	0	0	1	0.554	0.588	0.468	0.762	0.673	0.309	0.559
255	0	1	0	0	0	0	0	0	0.432	0.456	0.449	0.405	0.767	0.524	0.505
256	0	0	0	0	0	0	0	0	0.049	0.042	0.016	0.042	-0.019	0.05	0.03

Table A.26: Dataset 2 at Stage 20% - Gini Indices Part 6/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
1	0	0	0	0	1	1	0	0	0.5	0.738	0.577	0.927	0.885	0.865	0.749
2	1	0	0	0	1	1	0	0	0.591	0.775	0.577	0.86	0.818	0.833	0.742
3	1	0	1	0	1	1	0	0	0.603	0.757	0.558	0.878	0.818	0.827	0.74
4	0	0	0	0	0	1	0	0	0.5	0.597	0.681	0.786	0.903	0.939	0.734
5	1	0	1	0	1	0	0	0	0.64	0.784	0.563	0.897	0.755	0.759	0.733
6	0	0	0	1	1	1	0	0	0.525	0.664	0.644	0.934	0.732	0.889	0.731
7	1	0	0	1	1	1	0	0	0.628	0.701	0.614	0.872	0.732	0.84	0.731
8	0	0	1	0	1	0	0	0	0.628	0.735	0.481	0.952	0.774	0.798	0.728
9	1	0	1	1	1	1	0	0	0.658	0.669	0.558	0.86	0.78	0.827	0.725
10	0	0	0	0	1	1	1	0	0.621	0.674	0.449	0.927	0.823	0.852	0.724
11	1	0	0	0	1	0	0	0	0.658	0.811	0.595	0.89	0.692	0.698	0.724
12	1	0	1	0	1	1	0	1	0.591	0.738	0.54	0.885	0.818	0.759	0.722
13	0	0	0	0	1	1	0	1	0.512	0.775	0.54	0.922	0.855	0.722	0.721
14	0	0	0	0	1	0	0	0	0.572	0.657	0.62	0.909	0.694	0.847	0.717
15	1	1	1	0	1	1	0	0	0.621	0.737	0.54	0.804	0.799	0.79	0.715
16	1	1	0	0	1	1	0	0	0.61	0.787	0.595	0.737	0.78	0.771	0.713
17	1	0	0	0	1	1	0	1	0.53	0.775	0.577	0.836	0.823	0.729	0.712
18	1	0	1	1	1	0	0	0	0.677	0.73	0.549	0.878	0.676	0.759	0.711
19	1	0	1	0	1	1	1	0	0.658	0.643	0.5	0.897	0.767	0.803	0.711
20	0	0	1	1	1	1	0	0	0.598	0.614	0.506	0.897	0.769	0.875	0.71
21	1	0	0	1	1	0	0	0	0.677	0.706	0.607	0.921	0.631	0.717	0.71
22	1	1	0	0	1	1	1	0	0.628	0.711	0.5	0.823	0.767	0.821	0.708
23	0	1	0	0	1	1	0	0	0.549	0.738	0.54	0.755	0.829	0.838	0.708
24	0	0	0	1	1	1	0	1	0.549	0.701	0.57	0.903	0.755	0.771	0.708
25	0	0	1	0	1	1	0	1	0.475	0.775	0.508	0.91	0.818	0.764	0.708
26	1	0	0	0	1	1	1	0	0.64	0.693	0.481	0.86	0.759	0.815	0.708
27	1	0	1	0	1	0	0	1	0.591	0.786	0.545	0.873	0.73	0.722	0.708
28	0	1	1	0	1	1	0	0	0.561	0.706	0.526	0.866	0.78	0.807	0.708
29	0	0	1	0	1	1	0	0	0.475	0.688	0.508	0.934	0.78	0.84	0.704
30	0	0	0	1	1	1	1	0	0.677	0.6	0.461	0.909	0.718	0.857	0.704
31	0	0	0	1	1	0	0	0	0.647	0.569	0.579	0.89	0.681	0.84	0.701
32	0	0	1	0	1	1	1	0	0.64	0.643	0.431	0.915	0.749	0.821	0.7
33	1	0	0	1	1	1	1	0	0.714	0.6	0.486	0.89	0.705	0.803	0.7
34	0	0	0	0	1	1	1	1	0.542	0.711	0.449	0.903	0.804	0.784	0.699
35	1	0	0	0	1	1	1	1	0.628	0.711	0.5	0.86	0.74	0.754	0.699
36	0	0	1	1	1	0	0	0	0.635	0.606	0.529	0.897	0.694	0.833	0.699
37	1	0	1	0	1	0	1	0	0.647	0.752	0.486	0.86	0.712	0.735	0.699
38	1	0	1	1	1	1	0	1	0.61	0.614	0.545	0.836	0.78	0.801	0.698
39	0	0	1	1	1	1	0	1	0.567	0.651	0.525	0.873	0.75	0.813	0.696
40	0	1	1	1	1	1	0	0	0.616	0.651	0.525	0.823	0.75	0.813	0.696
41	1	0	0	0	1	0	1	0	0.665	0.734	0.505	0.89	0.666	0.717	0.696
42	0	1	0	0	1	1	1	0	0.628	0.711	0.431	0.786	0.767	0.852	0.696
43	1	0	0	1	0	1	0	0	0.695	0.585	0.614	0.786	0.718	0.776	0.696
44	0	0	1	0	1	0	1	0	0.647	0.709	0.431	0.915	0.73	0.742	0.696
45	0	0	0	1	0	1	0	0	0.543	0.479	0.705	0.804	0.732	0.901	0.694
46	1	1	0	1	1	1	0	0	0.665	0.651	0.558	0.767	0.75	0.771	0.694

Table A.27: Dataset 2 at Stage 50% - Gini Indices Part 1/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
47	1	1	1	1	1	1	0	0	0.647	0.651	0.54	0.786	0.743	0.795	0.693
48	0	0	1	1	1	1	1	0	0.714	0.587	0.443	0.878	0.718	0.82	0.693
49	1	1	0	0	1	0	0	0	0.647	0.835	0.577	0.712	0.742	0.648	0.693
50	0	1	1	0	1	1	0	1	0.542	0.725	0.526	0.799	0.799	0.764	0.693
51	0	1	1	0	1	0	1	1	0.647	0.803	0.431	0.841	0.73	0.69	0.69
52	1	0	0	0	1	0	0	1	0.572	0.812	0.614	0.873	0.666	0.599	0.689
53	0	0	1	0	1	1	1	1	0.598	0.698	0.431	0.897	0.767	0.746	0.689
54	1	1	1	0	1	1	1	0	0.647	0.68	0.449	0.804	0.762	0.791	0.689
55	1	0	0	0	0	1	0	0	0.621	0.585	0.595	0.725	0.823	0.783	0.689
56	0	1	0	1	1	1	0	0	0.586	0.651	0.552	0.755	0.769	0.82	0.689
57	0	1	1	1	1	1	1	0	0.647	0.606	0.461	0.86	0.75	0.808	0.689
58	0	1	1	0	1	0	0	1	0.572	0.821	0.449	0.799	0.762	0.727	0.688
59	0	1	1	0	1	0	0	0	0.542	0.808	0.468	0.841	0.725	0.746	0.688
60	0	1	0	0	1	1	0	1	0.512	0.775	0.558	0.762	0.818	0.702	0.688
61	1	0	1	0	1	1	1	1	0.616	0.68	0.481	0.86	0.767	0.722	0.688
62	1	1	1	0	1	1	0	1	0.591	0.706	0.54	0.78	0.799	0.709	0.687
63	0	1	1	0	1	1	1	0	0.647	0.68	0.431	0.823	0.78	0.764	0.687
64	0	1	0	0	1	0	0	0	0.616	0.821	0.526	0.651	0.767	0.741	0.687
65	1	1	1	0	1	0	0	0	0.598	0.771	0.513	0.786	0.725	0.722	0.686
66	1	0	1	1	1	1	1	0	0.695	0.624	0.468	0.841	0.7	0.784	0.685
67	0	0	1	0	1	0	0	1	0.542	0.68	0.463	0.929	0.762	0.734	0.685
68	0	1	1	0	1	0	1	0	0.598	0.808	0.431	0.841	0.725	0.704	0.684
69	1	0	0	1	1	0	1	0	0.702	0.643	0.535	0.89	0.599	0.735	0.684
70	1	0	0	1	1	1	0	1	0.604	0.609	0.577	0.836	0.718	0.759	0.684
71	1	1	0	0	1	1	0	1	0.591	0.738	0.558	0.725	0.786	0.704	0.684
72	0	0	0	0	0	1	1	0	0.647	0.508	0.449	0.804	0.823	0.87	0.683
73	0	1	1	0	1	1	1	1	0.616	0.68	0.431	0.841	0.799	0.727	0.682
74	0	0	1	1	1	1	1	1	0.635	0.624	0.443	0.878	0.737	0.776	0.682
75	1	0	1	1	1	0	0	1	0.665	0.637	0.53	0.848	0.663	0.746	0.681
76	0	0	0	1	1	1	1	1	0.616	0.606	0.48	0.878	0.718	0.79	0.681
77	1	1	0	0	1	0	1	0	0.647	0.789	0.486	0.786	0.68	0.698	0.681
78	1	1	0	0	1	1	1	1	0.616	0.693	0.481	0.792	0.772	0.722	0.679
79	1	1	1	0	1	0	1	0	0.61	0.771	0.449	0.767	0.725	0.754	0.679
80	1	1	1	1	1	1	0	1	0.628	0.651	0.526	0.799	0.743	0.727	0.679
81	1	0	1	0	1	0	1	1	0.635	0.729	0.468	0.86	0.68	0.704	0.679
82	1	0	1	0	0	1	0	1	0.628	0.634	0.526	0.792	0.818	0.672	0.678
83	0	0	0	1	1	0	1	0	0.702	0.55	0.498	0.89	0.649	0.778	0.678
84	0	1	1	1	1	1	0	1	0.561	0.619	0.525	0.799	0.769	0.795	0.678
85	0	1	0	1	1	1	0	1	0.549	0.651	0.575	0.78	0.75	0.758	0.677
86	1	0	1	1	0	1	0	0	0.695	0.534	0.526	0.749	0.78	0.776	0.677
87	0	0	0	0	1	0	1	0	0.647	0.609	0.456	0.927	0.649	0.772	0.677
88	0	1	0	0	1	0	1	0	0.635	0.789	0.431	0.767	0.681	0.754	0.676
89	0	1	0	0	1	1	1	1	0.598	0.711	0.431	0.774	0.786	0.752	0.675
90	1	1	1	0	1	1	1	1	0.616	0.68	0.449	0.804	0.799	0.704	0.675
91	0	0	1	1	1	0	0	1	0.647	0.624	0.48	0.873	0.676	0.752	0.675
92	0	1	0	1	1	1	1	0	0.665	0.6	0.443	0.823	0.718	0.801	0.675

Table A.28: Dataset 2 at Stage 50% - Gini Indices Part 2/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
93	1	1	0	1	1	1	1	0	0.665	0.637	0.468	0.804	0.686	0.79	0.675
94	0	0	1	1	1	0	1	0	0.684	0.606	0.461	0.86	0.681	0.759	0.675
95	1	0	1	1	1	0	1	0	0.647	0.698	0.443	0.841	0.663	0.754	0.674
96	1	1	1	0	1	0	0	1	0.647	0.759	0.481	0.762	0.743	0.653	0.674
97	1	1	1	1	1	1	1	1	0.653	0.624	0.468	0.823	0.743	0.727	0.673
98	1	0	0	1	1	0	0	1	0.647	0.688	0.594	0.855	0.599	0.653	0.672
99	0	1	1	1	1	0	0	0	0.616	0.68	0.498	0.78	0.713	0.746	0.672
100	1	1	1	1	1	1	1	0	0.647	0.606	0.468	0.804	0.725	0.778	0.671
101	1	1	0	1	1	1	0	1	0.591	0.688	0.577	0.762	0.718	0.69	0.671
102	1	0	1	1	0	1	0	1	0.647	0.565	0.526	0.799	0.78	0.709	0.671
103	1	0	0	0	1	0	1	1	0.635	0.729	0.486	0.878	0.648	0.643	0.67
104	0	0	0	1	0	1	0	1	0.481	0.578	0.663	0.78	0.742	0.771	0.669
105	0	1	0	1	1	1	1	1	0.635	0.619	0.461	0.829	0.718	0.752	0.669
106	0	1	1	1	1	0	1	0	0.635	0.692	0.443	0.841	0.663	0.741	0.669
107	1	1	0	0	1	0	0	1	0.598	0.809	0.558	0.743	0.712	0.592	0.669
108	1	0	0	1	1	1	1	1	0.635	0.563	0.486	0.878	0.673	0.772	0.668
109	1	0	1	0	0	1	0	0	0.64	0.585	0.521	0.718	0.818	0.715	0.666
110	1	0	1	1	1	0	1	1	0.672	0.624	0.498	0.841	0.649	0.709	0.666
111	0	0	0	0	0	1	0	1	0.5	0.689	0.577	0.818	0.855	0.555	0.665
112	0	0	1	1	0	1	0	0	0.604	0.497	0.547	0.73	0.769	0.844	0.665
113	1	1	1	1	1	0	1	0	0.647	0.654	0.468	0.804	0.676	0.741	0.665
114	0	0	1	0	1	0	1	1	0.628	0.661	0.412	0.878	0.73	0.678	0.665
115	1	1	1	0	1	0	1	1	0.635	0.729	0.449	0.804	0.698	0.672	0.664
116	1	1	0	0	1	0	1	1	0.616	0.747	0.486	0.804	0.698	0.629	0.664
117	1	1	1	1	1	0	0	0	0.635	0.654	0.498	0.774	0.688	0.727	0.663
118	0	0	0	0	0	1	1	1	0.579	0.588	0.449	0.848	0.791	0.721	0.663
119	0	0	0	0	1	0	0	1	0.512	0.674	0.55	0.866	0.692	0.68	0.662
120	1	1	1	0	0	1	0	1	0.598	0.62	0.54	0.706	0.799	0.709	0.662
121	0	0	1	1	1	0	1	1	0.684	0.606	0.461	0.848	0.663	0.709	0.662
122	1	0	0	0	0	1	1	0	0.684	0.54	0.449	0.749	0.759	0.784	0.661
123	0	1	0	1	0	1	0	0	0.574	0.571	0.629	0.614	0.769	0.807	0.661
124	0	1	0	1	1	0	0	0	0.616	0.637	0.555	0.737	0.676	0.741	0.66
125	0	0	1	1	0	1	0	1	0.586	0.546	0.517	0.799	0.769	0.746	0.66
126	0	1	0	1	1	0	1	0	0.635	0.617	0.461	0.841	0.663	0.741	0.66
127	1	1	1	0	0	1	0	0	0.621	0.62	0.54	0.663	0.78	0.727	0.659
128	0	1	1	1	1	1	1	1	0.635	0.569	0.461	0.823	0.737	0.727	0.659
129	0	0	1	0	0	1	0	1	0.475	0.634	0.545	0.818	0.818	0.66	0.658
130	1	0	0	0	0	1	1	1	0.653	0.588	0.468	0.804	0.74	0.692	0.658
131	1	1	0	1	1	0	0	0	0.665	0.686	0.562	0.7	0.663	0.667	0.657
132	1	1	0	0	0	1	0	0	0.616	0.634	0.577	0.632	0.767	0.715	0.657
133	1	1	0	1	0	1	0	0	0.653	0.553	0.558	0.681	0.755	0.739	0.657
134	0	0	1	0	0	1	1	1	0.635	0.575	0.431	0.841	0.767	0.69	0.657
135	1	0	1	0	0	1	1	1	0.653	0.557	0.431	0.841	0.754	0.704	0.657
136	1	1	0	1	1	1	1	1	0.635	0.656	0.486	0.804	0.668	0.69	0.657
137	1	0	1	1	1	1	1	1	0.653	0.569	0.468	0.841	0.7	0.709	0.657
138	1	0	0	1	0	1	0	1	0.623	0.578	0.558	0.799	0.686	0.69	0.656

Table A.29: Dataset 2 at Stage 50% - Gini Indices Part 3/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
139	0	1	0	0	0	1	1	0	0.628	0.54	0.449	0.712	0.786	0.82	0.656
140	0	1	1	1	1	0	0	1	0.591	0.624	0.517	0.78	0.694	0.727	0.656
141	1	0	0	1	1	0	1	1	0.653	0.643	0.498	0.86	0.599	0.68	0.655
142	0	1	1	1	1	0	1	1	0.647	0.624	0.461	0.811	0.681	0.709	0.655
143	1	1	0	1	1	0	1	0	0.647	0.636	0.486	0.804	0.649	0.704	0.654
144	1	0	0	0	0	1	0	1	0.567	0.634	0.54	0.762	0.804	0.618	0.654
145	0	0	0	0	1	0	1	1	0.628	0.624	0.431	0.885	0.636	0.705	0.651
146	0	0	0	1	1	0	0	1	0.554	0.587	0.567	0.866	0.631	0.704	0.651
147	1	1	1	1	1	0	0	1	0.647	0.661	0.486	0.78	0.663	0.672	0.651
148	0	0	0	1	0	1	1	0	0.665	0.439	0.443	0.804	0.718	0.838	0.651
149	0	1	1	1	0	1	0	0	0.537	0.571	0.554	0.681	0.75	0.813	0.651
150	0	1	1	1	0	1	1	0	0.628	0.545	0.461	0.749	0.75	0.771	0.651
151	1	0	1	0	0	1	1	0	0.647	0.526	0.449	0.749	0.767	0.766	0.651
152	1	0	0	1	0	1	1	0	0.684	0.502	0.449	0.786	0.692	0.784	0.649
153	0	0	1	1	0	1	1	1	0.653	0.52	0.443	0.823	0.737	0.721	0.649
154	1	1	0	0	0	1	1	0	0.628	0.577	0.468	0.73	0.754	0.735	0.649
155	1	1	1	1	1	0	1	1	0.635	0.643	0.449	0.804	0.649	0.709	0.648
156	0	1	0	1	0	1	0	1	0.574	0.57	0.581	0.706	0.755	0.702	0.648
157	0	1	1	0	0	1	0	1	0.512	0.588	0.518	0.743	0.799	0.727	0.648
158	0	0	0	1	0	1	1	1	0.653	0.52	0.461	0.804	0.705	0.739	0.647
159	0	1	0	0	1	0	0	1	0.524	0.816	0.489	0.725	0.73	0.598	0.647
160	1	1	0	1	1	0	0	1	0.635	0.674	0.532	0.762	0.644	0.635	0.647
161	0	0	1	0	0	1	1	0	0.665	0.476	0.431	0.786	0.749	0.771	0.646
162	1	1	0	1	1	0	1	1	0.635	0.661	0.486	0.811	0.649	0.635	0.646
163	1	1	1	1	0	1	0	0	0.647	0.553	0.54	0.681	0.725	0.727	0.645
164	1	0	0	1	0	1	1	1	0.653	0.521	0.468	0.841	0.673	0.709	0.644
165	0	1	1	0	0	1	0	0	0.5	0.571	0.55	0.651	0.78	0.812	0.644
166	1	1	0	0	0	1	0	1	0.598	0.615	0.558	0.706	0.786	0.598	0.643
167	1	1	0	1	0	1	0	1	0.616	0.59	0.577	0.706	0.718	0.653	0.643
168	1	1	1	0	0	1	1	1	0.616	0.575	0.449	0.73	0.799	0.69	0.643
169	1	1	1	1	0	1	0	1	0.616	0.553	0.513	0.725	0.743	0.709	0.643
170	0	0	0	1	1	0	1	1	0.665	0.531	0.443	0.878	0.617	0.722	0.643
171	0	1	1	1	0	1	0	1	0.567	0.551	0.517	0.743	0.75	0.727	0.643
172	0	1	0	0	1	0	1	1	0.579	0.747	0.412	0.78	0.668	0.667	0.642
173	0	1	1	0	0	1	1	1	0.598	0.575	0.431	0.767	0.767	0.709	0.641
174	0	1	0	1	0	1	1	0	0.647	0.54	0.443	0.712	0.718	0.783	0.64
175	1	0	1	1	0	1	1	0	0.684	0.508	0.431	0.749	0.7	0.771	0.64
176	0	1	0	1	0	1	1	1	0.653	0.54	0.48	0.774	0.705	0.69	0.64
177	1	0	1	0	0	0	0	0	0.61	0.624	0.532	0.706	0.755	0.611	0.64
178	0	0	1	1	0	1	1	0	0.684	0.457	0.443	0.73	0.718	0.801	0.639
179	1	0	0	1	0	0	0	0	0.714	0.571	0.599	0.749	0.599	0.598	0.638
180	1	0	1	1	0	0	0	0	0.647	0.577	0.517	0.718	0.676	0.69	0.637
181	1	1	1	1	0	1	1	1	0.635	0.526	0.468	0.767	0.717	0.709	0.637
182	0	1	0	0	0	1	0	1	0.5	0.615	0.558	0.706	0.818	0.623	0.637
183	0	1	0	0	0	1	1	1	0.635	0.57	0.449	0.737	0.772	0.653	0.636
184	0	1	1	0	0	0	1	1	0.61	0.661	0.412	0.749	0.73	0.653	0.636

Table A.30: Dataset 2 at Stage 50% - Gini Indices Part 4/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
185	0	1	0	1	1	0	1	1	0.635	0.569	0.443	0.848	0.649	0.667	0.635
186	1	1	0	0	0	1	1	1	0.635	0.57	0.468	0.718	0.772	0.635	0.633
187	0	1	1	1	0	1	1	1	0.635	0.526	0.461	0.767	0.718	0.69	0.633
188	0	0	1	0	0	1	0	0	0.475	0.528	0.545	0.755	0.78	0.712	0.633
189	0	1	1	0	0	1	1	0	0.591	0.545	0.431	0.693	0.78	0.752	0.632
190	1	1	1	0	0	1	1	0	0.61	0.582	0.449	0.712	0.743	0.69	0.631
191	1	1	0	1	0	1	1	0	0.647	0.54	0.449	0.712	0.686	0.752	0.631
192	1	1	1	1	0	1	1	0	0.628	0.526	0.468	0.712	0.725	0.727	0.631
193	0	1	1	1	0	0	0	1	0.598	0.545	0.554	0.713	0.694	0.672	0.629
194	0	0	1	0	0	0	0	1	0.579	0.594	0.486	0.78	0.743	0.592	0.629
195	0	1	0	1	1	0	0	1	0.579	0.6	0.53	0.762	0.663	0.635	0.628
196	1	0	0	0	0	0	0	1	0.653	0.671	0.595	0.762	0.629	0.458	0.628
197	1	0	0	0	0	0	0	0	0.689	0.602	0.614	0.712	0.673	0.477	0.628
198	0	1	1	1	0	0	1	1	0.628	0.545	0.48	0.774	0.681	0.653	0.627
199	0	0	1	0	0	0	1	1	0.665	0.557	0.394	0.786	0.712	0.641	0.626
200	1	1	1	0	0	0	0	1	0.616	0.654	0.468	0.706	0.73	0.579	0.626
201	0	1	0	0	0	1	0	0	0.432	0.566	0.614	0.558	0.829	0.751	0.625
202	0	0	1	1	0	0	0	1	0.635	0.501	0.535	0.743	0.676	0.66	0.625
203	0	1	1	0	0	0	1	0	0.579	0.606	0.431	0.712	0.712	0.704	0.624
204	1	0	1	0	0	0	1	0	0.61	0.592	0.449	0.712	0.698	0.68	0.623
205	1	0	1	0	0	0	0	1	0.598	0.625	0.463	0.762	0.73	0.561	0.623
206	1	0	1	1	0	1	1	1	0.653	0.508	0.431	0.767	0.686	0.69	0.623
207	1	1	1	0	0	0	1	1	0.598	0.643	0.449	0.749	0.698	0.598	0.622
208	1	1	0	1	0	1	1	1	0.635	0.558	0.468	0.749	0.668	0.653	0.622
209	0	0	1	0	0	0	1	0	0.579	0.567	0.431	0.73	0.749	0.673	0.621
210	0	1	1	0	0	0	0	1	0.542	0.649	0.486	0.651	0.762	0.635	0.621
211	1	1	1	1	0	0	0	1	0.653	0.582	0.498	0.713	0.663	0.616	0.621
212	0	0	1	1	0	0	1	1	0.665	0.501	0.461	0.755	0.663	0.672	0.619
213	0	1	1	1	0	0	0	0	0.518	0.57	0.584	0.62	0.713	0.709	0.619
214	1	0	1	1	0	0	0	1	0.653	0.551	0.48	0.737	0.663	0.616	0.617
215	1	0	1	1	0	0	1	0	0.628	0.563	0.443	0.712	0.649	0.704	0.616
216	1	0	0	1	0	0	1	0	0.684	0.545	0.498	0.749	0.599	0.606	0.613
217	1	1	0	1	0	0	1	0	0.616	0.575	0.48	0.749	0.631	0.629	0.613
218	1	1	1	1	0	0	1	1	0.635	0.582	0.449	0.749	0.649	0.616	0.613
219	1	0	1	0	0	0	1	1	0.635	0.587	0.412	0.767	0.666	0.611	0.613
220	1	0	0	0	0	0	1	1	0.672	0.587	0.468	0.804	0.629	0.513	0.612
221	0	0	1	0	0	0	0	0	0.529	0.628	0.403	0.706	0.774	0.633	0.612
222	0	0	1	1	0	0	0	0	0.572	0.439	0.547	0.688	0.694	0.727	0.611
223	1	0	0	0	0	0	1	0	0.695	0.543	0.468	0.767	0.611	0.577	0.61
224	1	0	1	1	0	0	1	1	0.653	0.526	0.461	0.767	0.636	0.616	0.61
225	0	1	0	1	0	0	1	1	0.653	0.489	0.48	0.755	0.649	0.616	0.607
226	0	1	1	1	0	0	1	0	0.542	0.575	0.443	0.693	0.663	0.722	0.606
227	1	1	0	0	0	0	0	1	0.598	0.668	0.54	0.651	0.693	0.487	0.606
228	1	1	0	1	0	0	0	1	0.653	0.595	0.525	0.651	0.644	0.561	0.605
229	1	0	0	1	0	0	0	1	0.635	0.583	0.525	0.762	0.58	0.542	0.605
230	1	1	1	0	0	0	1	0	0.561	0.599	0.449	0.693	0.725	0.598	0.604

Table A.31: Dataset 2 at Stage 50% - Gini Indices Part 5/6

ID	rf	NN1	NN2	NN3	bn	knn	LR	svm	SP1	SP2	SP3	SP4	SP5	test	avg
231	1	0	0	1	0	0	1	1	0.672	0.526	0.461	0.786	0.599	0.579	0.604
232	0	0	0	1	0	0	1	1	0.684	0.445	0.443	0.792	0.617	0.635	0.603
233	1	1	0	0	0	0	1	0	0.579	0.617	0.468	0.675	0.661	0.606	0.601
234	1	1	0	0	0	0	0	0	0.579	0.644	0.577	0.558	0.705	0.537	0.6
235	0	0	1	1	0	0	1	0	0.635	0.407	0.443	0.712	0.681	0.722	0.6
236	1	1	0	1	0	0	1	1	0.635	0.563	0.468	0.737	0.631	0.561	0.599
237	1	1	0	0	0	0	1	1	0.635	0.624	0.431	0.712	0.685	0.505	0.598
238	1	1	1	0	0	0	0	0	0.549	0.599	0.486	0.614	0.725	0.616	0.598
239	0	0	0	1	0	0	0	1	0.609	0.464	0.535	0.75	0.599	0.629	0.598
240	1	1	1	1	0	0	0	0	0.549	0.575	0.48	0.651	0.688	0.635	0.596
241	1	1	1	1	0	0	1	0	0.579	0.575	0.461	0.675	0.649	0.635	0.596
242	0	0	0	1	0	0	0	0	0.592	0.37	0.559	0.681	0.594	0.778	0.596
243	0	0	0	0	0	0	1	1	0.684	0.501	0.375	0.811	0.636	0.567	0.596
244	0	1	0	1	0	0	0	1	0.598	0.509	0.572	0.669	0.644	0.579	0.595
245	1	1	0	1	0	0	0	0	0.586	0.588	0.543	0.602	0.644	0.598	0.594
246	0	0	0	1	0	0	1	0	0.69	0.37	0.443	0.712	0.617	0.704	0.589
247	0	0	0	0	0	0	1	0	0.704	0.382	0.382	0.73	0.592	0.741	0.588
248	0	1	0	1	0	0	1	0	0.586	0.469	0.443	0.693	0.649	0.667	0.584
249	0	1	0	0	0	0	1	1	0.598	0.606	0.394	0.706	0.668	0.524	0.582
250	0	1	0	0	0	0	1	0	0.599	0.557	0.38	0.656	0.668	0.611	0.578
251	0	1	0	1	0	0	0	0	0.525	0.465	0.598	0.546	0.663	0.653	0.575
252	0	1	0	0	0	0	0	1	0.524	0.632	0.486	0.595	0.749	0.444	0.572
253	0	1	1	0	0	0	0	0	0.518	0.487	0.449	0.521	0.725	0.695	0.566
254	0	0	0	0	0	0	0	1	0.554	0.588	0.468	0.762	0.673	0.309	0.559
255	0	1	0	0	0	0	0	0	0.432	0.456	0.449	0.405	0.767	0.524	0.505
256	0	0	0	0	0	0	0	0	0.049	0.042	0.016	0.042	-0.019	0.05	0.03

Table A.32: Dataset 2 at Stage 50% - Gini Indices Part 6/6

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
1	0.008	0.085	0.209	0.434	0.028	0.002	0.065	0.0123	0.008	0.434	0.086	0.029	0.449	0.124	0.076	0.137	0.018	0
2	0.008	0.0106	0.026	0.0192	0.009	0.002	0.027	0.0123	0.027	0.0282	0.0064	0.029	0.0135	0.0065	0.001	0.00959	0.0139	0.005
3	0.128	0.064	0.209	0.639	0.107	0.002	0.295	0.0123	0.008	0.452	0.124	0.02	0.637	0.124	0.002	0	0.006	0
4	0.07	0.064	0.311	0.452	0.107	0.002	0.065	0.123	0.005	0.636	0.124	0.02	0.449	0.065	0.001	0.634	0.065	0.001
5	0.063	0.154	0.255	0.846	0.047	0.002	0.433	0.108	0.003	0.434	0.086	0.013	0.96	0.021	0.004	0.192	0.03	0.001
6	0.063	0.064	0.255	0.846	0.047	0.002	0.433	0.108	0.005	0.452	0.064	0.008	0.828	0.021	0.007	0.432	0.065	0
7	0.063	0.064	0.168	0.846	0.047	0.002	0.433	0.064	0.003	0.434	0.086	0.013	1	0.048	0.17	0.007	0.018	0
8	0.07	0.085	0.209	0.622	0.107	0.002	0.295	0.108	0.005	0.636	0.108	0.02	0.844	0.124	0.002	0.192	0.018	0
9	0.128	0.154	0.311	0.622	0.107	0.002	0.065	0.262	0.008	0.827	0.155	0.029	0.135	0.065	0.001	0.432	0.065	0.003
10	0.063	0.154	0.209	0.96	0.084	0.002	0.636	0.048	0.001	0.434	0.124	0.02	1	0.048	0.002	0.032	0.03	0
11	0.128	0.064	0.209	0.828	0.107	0.002	0.451	0.156	0.005	0.845	0.155	0.02	1	0.108	0.001	0.007	0.01	0
12	0.008	0.154	0.209	1	0.084	0.002	0.433	0.064	0.008	0.295	0.124	0.02	0.28	0.021	0.001	0.959	0.139	0.001
13	0.063	0.064	0.168	0.96	0.047	0.002	0.451	0.085	0.003	0.636	0.086	0.013	1	0.048	0.007	0.027	0.018	0
14	0.063	0.154	0.255	0.846	0.047	0.002	0.433	0.108	0.003	0.619	0.086	0.013	0.96	0.021	0.002	0.432	0.03	0
15	0.07	0.154	0.168	1	0.047	0.002	0.451	0.085	0.005	0.295	0.108	0.013	1	0.048	0.007	0.032	0.03	0
16	0.136	0.192	0.168	0.828	0.064	0.002	0.282	0.108	0.003	0.619	0.193	0.029	0.28	0.065	0.001	0.616	0.048	0.003
17	0.282	0.154	0.209	1	0.107	0.002	0.433	0.123	0.005	0.619	0.124	0.02	0.844	0.108	0.001	0.065	0.018	0
18	0.063	0.064	0.168	0.846	0.047	0.002	0.433	0.064	0.003	0.452	0.086	0.013	1	0.029	0.17	0.128	0.021	0
19	0.282	0.064	0.168	0.846	0.064	0.002	0.433	0.156	0.005	0.452	0.124	0.02	1	0.048	0.057	0.032	0.01	0
20	0.008	0.154	0.209	0.96	0.064	0.002	0.451	0.029	0.003	0.295	0.108	0.02	1	0.029	0.17	0.432	0.03	0
21	0.136	0.064	0.255	0.828	0.064	0.002	0.137	0.123	0.005	0.845	0.108	0.02	0.637	0.021	0.001	0.846	0.107	0.001
22	0.07	0.154	0.209	1	0.084	0.002	0.829	0.064	0.001	0.636	0.124	0.013	1	0.048	0.001	0.128	0.03	0
23	0.07	0.154	0.209	0.96	0.084	0.002	0.636	0.048	0.001	0.452	0.108	0.02	1	0.048	0.004	0.294	0.03	0
24	0.282	0.122	0.209	0.846	0.123	0.002	0.636	0.108	0.003	0.636	0.262	0.029	0.619	0.086	0.001	0.072	0.048	0.002
25	0.07	0.154	0.168	1	0.047	0.002	0.433	0.064	0.003	0.452	0.086	0.013	1	0.048	0.007	0.294	0.021	0
26	0.136	0.262	0.168	0.452	0.084	0.002	0.071	0.108	0.04	0.619	0.155	0.041	0.069	0.065	0	1	0.193	0.005
27	0.128	0.262	0.255	1	0.047	0.002	0.433	0.085	0.003	0.619	0.086	0.013	0.449	0.021	0.001	0.828	0.139	0.002
28	0.136	0.154	0.255	1	0.084	0.002	0.829	0.085	0.001	0.619	0.155	0.02	1	0.029	0.001	0.137	0.048	0
29	0.192	0.064	0.255	0.96	0.064	0.002	0.295	0.123	0.005	0.636	0.155	0.02	0.828	0.021	0.001	0.828	0.065	0.001
30	0.128	0.154	0.311	0.639	0.107	0.013	0.282	0.262	0.005	0.845	0.155	0.02	0.828	0.029	0.001	0.45	0.03	0.001
31	0.192	0.064	0.168	0.96	0.064	0.002	0.451	0.123	0.005	0.827	0.124	0.02	1	0.048	0.004	0.128	0.018	0
32	0.282	0.064	0.255	0.96	0.064	0.002	0.192	0.156	0.005	0.452	0.155	0.02	0.844	0.021	0.001	0.959	0.048	0.002
33	0.063	0.154	0.168	1	0.047	0.002	0.451	0.108	0.003	0.619	0.108	0.013	1	0.048	0.007	0.137	0.065	0
34	0.136	0.373	0.255	0.96	0.047	0.002	0.433	0.085	0.003	0.619	0.086	0.013	0.96	0.021	0.001	0.634	0.107	0.001
35	0.192	0.154	0.209	0.846	0.123	0.002	0.829	0.108	0.003	0.827	0.262	0.02	0.828	0.086	0.001	0.137	0.03	0.001
36	0.026	0.064	0.255	0.828	0.047	0.002	0.137	0.085	0.008	0.282	0.048	0.013	1	0.029	0.874	0.634	0.065	0.001
37	0.434	0.315	0.209	0.828	0.107	0.002	0.295	0.123	0.008	0.434	0.108	0.029	0.828	0.124	0.001	0.45	0.021	0
38	0.128	0.154	0.168	1	0.084	0.002	0.451	0.085	0.003	0.845	0.108	0.013	1	0.065	0.004	0.072	0.021	0
39	0.07	0.262	0.255	0.96	0.047	0.002	0.295	0.064	0.003	0.282	0.086	0.013	1	0.021	0.004	1	0.139	0.002
40	0.128	0.106	0.209	0.828	0.047	0.002	0.071	0.108	0.012	0.434	0.108	0.013	0.828	0.021	0.076	1	0.156	0.003
41	0.136	0.122	0.209	1	0.084	0.002	0.636	0.085	0.001	0.295	0.155	0.013	1	0.029	0.001	0.432	0.107	0.001
42	0.063	0.154	0.168	1	0.047	0.002	0.451	0.064	0.003	0.295	0.108	0.013	1	0.048	0.142	0.192	0.065	0
43	0.128	0.154	0.255	1	0.084	0.002	0.636	0.064	0.001	0.619	0.124	0.013	1	0.048	0.001	0.45	0.03	0
44	0.07	0.064	0.168	1	0.047	0.002	0.451	0.085	0.003	0.845	0.086	0.013	1	0.048	0.142	0.128	0.021	0
45	0.07	0.154	0.209	1	0.064	0.002	0.846	0.064	0.001	0.619	0.124	0.013	1	0.048	0.007	0.192	0.065	0
46	0.136	0.154	0.209	1	0.084	0.002	0.636	0.064	0.005	0.619	0.108	0.02	0.828	0.021	0.001	0.616	0.139	0.001
47	0.128	0.064	0.209	0.96	0.084	0.002	0.829	0.108	0.003	0.827	0.155	0.02	1	0.048	0.001	0.616	0.048	0
48	0.296	0.085	0.141	0.846	0.084	0.002	0.451	0.085	0.003	0.452	0.155	0.02	1	0.048	0.019	0.634	0.048	0
49	0.282	0.154	0.209	0.846	0.123	0.002	0.829	0.085	0.003	0.636	0.193	0.029	0.619	0.086	0.001	0.45	0.03	0.001
50	0.03	0.154	0.168	1	0.047	0.002	0.433	0.048	0.003	0.434	0.086	0.02	1	0.048	0.313	0.432	0.048	0
51	0.03	0.064	0.209	1	0.084	0.002	0.846	0.048	0.001	0.619	0.108	0.013	1	0.048	0.04	0.45	0.065	0

Table A.33: Dataset 1 at Stage 20% - p values splits- Part1/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
52	0.063	0.154	0.209	1	0.084	0.002	0.846	0.048	0.001	0.452	0.124	0.013	1	0.048	0.04	0.192	0.065	0
53	0.07	0.122	0.209	1	0.084	0.002	0.618	0.064	0.001	0.434	0.124	0.02	1	0.048	0.002	0.828	0.074	0
54	0.451	0.154	0.209	1	0.107	0.002	0.451	0.194	0.005	0.827	0.124	0.02	1	0.108	0.001	0.072	0.018	0
55	0.063	0.064	0.209	1	0.047	0.002	0.295	0.085	0.003	0.434	0.086	0.013	1	0.014	0.313	0.634	0.139	0.001
56	0.296	0.192	0.168	1	0.084	0.002	0.295	0.085	0.008	0.452	0.193	0.02	0.189	0.014	0	1	0.156	0.003
57	0.282	0.154	0.168	1	0.064	0.002	0.451	0.108	0.005	0.434	0.124	0.02	1	0.048	0.002	0.294	0.048	0
58	0.282	0.154	0.209	1	0.084	0.002	0.636	0.085	0.005	0.636	0.124	0.013	0.637	0.021	0.001	0.634	0.139	0.001
59	0.451	0.064	0.168	1	0.107	0.002	0.961	0.085	0.005	0.845	0.124	0.02	1	0.124	0.004	0	0.01	0
60	0.07	0.064	0.255	1	0.047	0.002	0.451	0.085	0.003	0.845	0.086	0.013	1	0.014	0.007	0.45	0.107	0.001
61	0.03	0.064	0.255	0.96	0.047	0.002	0.433	0.064	0.005	0.452	0.064	0.013	1	0.014	0.458	0.432	0.139	0.001
62	0.026	0.064	0.168	0.639	0.028	0.002	0.282	0.048	0.003	0.434	0.064	0.013	1	0.029	1	0.072	0.021	0
63	0.282	0.154	0.209	1	0.084	0.002	0.636	0.064	0.005	0.452	0.124	0.02	0.637	0.021	0.001	0.828	0.139	0.002
64	0.282	0.154	0.209	0.96	0.107	0.002	0.433	0.156	0.005	0.845	0.124	0.02	1	0.108	0.001	0.432	0.021	0
65	0.451	0.424	0.311	0.622	0.107	0.002	0.032	0.194	0.008	0.636	0.155	0.029	0.135	0.065	0.001	1	0.193	0.005
66	0.192	0.154	0.209	1	0.084	0.002	0.829	0.123	0.003	0.827	0.193	0.02	1	0.048	0.001	0.294	0.048	0
67	0.128	0.262	0.255	1	0.084	0.002	0.451	0.108	0.003	0.845	0.108	0.013	1	0.021	0.001	0.616	0.107	0.001
68	0.192	0.154	0.168	1	0.064	0.002	0.618	0.156	0.005	0.636	0.124	0.02	1	0.048	0.001	0.192	0.03	0
69	0.136	0.085	0.168	0.846	0.064	0.002	0.295	0.156	0.005	0.845	0.108	0.02	1	0.048	0.076	0.432	0.021	0
70	0.063	0.122	0.209	1	0.084	0.002	0.829	0.048	0.001	0.137	0.124	0.013	1	0.029	0.057	0.828	0.074	0
71	0.296	0.154	0.168	1	0.084	0.002	0.961	0.085	0.003	0.619	0.108	0.013	1	0.065	0.012	0.027	0.018	0
72	0.192	0.122	0.209	1	0.084	0.002	1	0.064	0.001	0.636	0.124	0.013	1	0.048	0.007	0.027	0.03	0
73	0.296	0.315	0.311	0.846	0.107	0.013	0.282	0.156	0.005	0.619	0.155	0.02	0.449	0.029	0.001	0.828	0.107	0.005
74	0.296	0.154	0.209	1	0.064	0.002	0.829	0.108	0.001	0.619	0.155	0.02	1	0.029	0.004	0.281	0.074	0
75	0.282	0.122	0.168	1	0.064	0.002	0.433	0.123	0.005	0.434	0.108	0.02	1	0.029	0.057	0.634	0.065	0
76	0.03	0.154	0.209	1	0.084	0.002	0.636	0.048	0.005	0.619	0.108	0.013	1	0.021	0.004	0.846	0.156	0.001
77	0.136	0.262	0.209	1	0.084	0.002	0.636	0.085	0.003	0.434	0.124	0.02	0.637	0.029	0.001	0.959	0.156	0.002
78	0.136	0.064	0.168	1	0.047	0.002	0.846	0.064	0.003	0.845	0.086	0.013	1	0.086	0.458	0	0.01	0
79	0.622	0.154	0.209	0.846	0.123	0.002	0.433	0.123	0.012	0.636	0.193	0.041	0.189	0.029	0.001	0.846	0.139	0.003
80	0.296	0.154	0.255	0.96	0.064	0.002	0.433	0.194	0.005	0.636	0.155	0.02	0.844	0.021	0.001	0.846	0.139	0.001
81	0.136	0.154	0.209	1	0.084	0.002	0.829	0.064	0.001	0.619	0.124	0.013	1	0.048	0.007	0.616	0.074	0
82	0.128	0.154	0.255	1	0.047	0.002	0.451	0.085	0.003	0.619	0.086	0.013	1	0.014	0.007	0.846	0.139	0.002
83	0.128	0.315	0.209	1	0.084	0.002	0.636	0.064	0.005	0.434	0.124	0.02	0.637	0.021	0.001	1	0.156	0.002
84	0.282	0.154	0.168	0.96	0.064	0.002	0.433	0.123	0.005	0.619	0.124	0.02	1	0.065	0.004	0.616	0.048	0
85	0.136	0.154	0.255	1	0.047	0.002	0.451	0.085	0.003	0.619	0.108	0.013	1	0.014	0.004	0.616	0.139	0.001
86	0.282	0.154	0.209	1	0.084	0.002	1	0.064	0.001	0.452	0.155	0.013	1	0.048	0.007	0.072	0.065	0.001
87	0.296	0.048	0.168	1	0.107	0.002	1	0.064	0.005	0.452	0.124	0.02	1	0.155	0.1	0.007	0.005	0
88	0.622	0.192	0.168	0.846	0.084	0.002	0.636	0.123	0.003	0.636	0.193	0.029	0.619	0.086	0.001	0.828	0.065	0.002
89	0.434	0.122	0.209	1	0.084	0.002	0.829	0.108	0.003	0.845	0.193	0.02	1	0.048	0.001	0.281	0.048	0.001
90	0.192	0.064	0.168	1	0.064	0.002	1	0.123	0.003	0.827	0.124	0.02	1	0.029	0.142	0.007	0.01	0
91	0.639	0.122	0.168	1	0.123	0.002	1	0.064	0.001	0.636	0.193	0.02	1	0.108	0.001	0.009	0.048	0.002
92	0.136	0.262	0.255	1	0.084	0.002	0.636	0.085	0.005	0.619	0.124	0.02	0.637	0.029	0.001	0.959	0.156	0.001
93	0.063	0.106	0.141	0.434	0.047	0.004	0.636	0.014	0.005	0.128	0.064	0.008	1	0.01	1	0.294	0.065	0
94	0.128	0.154	0.209	1	0.084	0.002	0.829	0.108	0.005	0.295	0.124	0.013	1	0.021	0.004	0.846	0.156	0.001
95	0.07	0.154	0.168	1	0.047	0.002	0.451	0.064	0.003	0.827	0.086	0.013	1	0.048	0.142	0.45	0.065	0
96	0.192	0.122	0.209	1	0.084	0.002	0.829	0.085	0.001	0.434	0.155	0.013	1	0.048	0.007	0.616	0.074	0
97	0.03	0.064	0.168	1	0.047	0.002	0.451	0.085	0.003	0.452	0.086	0.013	1	0.065	1	0.032	0.021	0
98	0.296	0.154	0.168	1	0.047	0.002	0.846	0.048	0.003	0.295	0.108	0.013	1	0.048	0.528	0.009	0.018	0
99	0.192	0.154	0.209	1	0.084	0.002	0.829	0.108	0.003	0.434	0.124	0.013	1	0.021	0.002	0.846	0.156	0.002
100	0.434	0.315	0.255	1	0.064	0.002	0.192	0.123	0.008	0.619	0.124	0.02	0.844	0.014	0.001	1	0.139	0.005
101	0.622	0.154	0.209	1	0.123	0.002	0.636	0.085	0.003	0.845	0.262	0.029	0.828	0.086	0.001	0.432	0.065	0.002
102	0.622	0.154	0.255	0.96	0.123	0.002	0.636	0.123	0.003	0.827	0.193	0.029	0.828	0.086	0.001	0.616	0.048	0.001

Table A.34: Dataset 1 at Stage 20% - p values splits- Part2/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
103	0.136	0.154	0.255	1	0.047	0.002	0.433	0.085	0.003	0.619	0.086	0.013	1	0.014	0.012	0.959	0.139	0.001
104	0.434	0.122	0.209	1	0.123	0.002	1	0.108	0.001	0.827	0.193	0.02	1	0.108	0.001	0.009	0.03	0.001
105	0.296	0.154	0.209	1	0.084	0.002	0.636	0.123	0.005	0.827	0.262	0.02	0.637	0.021	0.001	0.959	0.156	0.001
106	0.282	0.122	0.209	1	0.084	0.002	1	0.064	0.001	0.636	0.155	0.02	1	0.048	0.007	0.128	0.03	0.001
107	0.622	0.154	0.255	0.96	0.064	0.002	0.295	0.123	0.005	0.636	0.124	0.02	0.96	0.021	0.001	1	0.139	0.001
108	0.639	0.122	0.168	1	0.107	0.002	1	0.156	0.005	0.827	0.193	0.02	1	0.124	0.001	0.009	0.01	0
109	0.434	0.154	0.209	1	0.084	0.002	0.451	0.123	0.005	0.845	0.155	0.02	0.619	0.029	0.001	1	0.156	0.001
110	0.192	0.154	0.209	1	0.084	0.002	1	0.064	0.001	0.619	0.124	0.013	1	0.048	0.17	0.027	0.048	0.001
111	0.622	0.154	0.209	1	0.084	0.002	0.829	0.123	0.003	0.619	0.193	0.02	1	0.048	0.001	0.616	0.065	0
112	0.434	0.192	0.168	1	0.084	0.002	0.636	0.108	0.003	0.619	0.193	0.02	1	0.029	0.001	0.634	0.074	0.001
113	0.192	0.154	0.209	1	0.084	0.002	0.829	0.108	0.005	0.827	0.124	0.013	1	0.021	0.004	0.634	0.156	0.001
114	0.03	0.085	0.141	1	0.028	0.004	0.636	0.014	0.003	0.137	0.064	0.008	1	0.086	1	0.065	0.018	0
115	0.282	0.262	0.255	1	0.084	0.002	0.961	0.108	0.003	0.827	0.086	0.013	1	0.021	0.007	0.137	0.03	0
116	0.026	0.154	0.255	1	0.047	0.004	0.295	0.064	0.003	0.192	0.086	0.013	1	0.01	0.818	1	0.139	0.002
117	0.846	0.106	0.168	1	0.107	0.002	0.961	0.156	0.005	0.619	0.193	0.02	1	0.124	0.002	0.007	0.01	0
118	0.622	0.122	0.209	1	0.084	0.002	0.829	0.156	0.003	0.619	0.193	0.02	1	0.048	0.001	0.616	0.065	0
119	0.192	0.154	0.209	1	0.084	0.002	0.829	0.064	0.005	0.282	0.124	0.013	1	0.014	0.002	1	0.156	0.002
120	0.282	0.373	0.311	0.846	0.107	0.013	0.433	0.194	0.005	0.845	0.193	0.02	0.828	0.065	0.001	0.828	0.074	0.002
121	0.451	0.154	0.255	1	0.123	0.002	0.636	0.123	0.003	0.827	0.262	0.02	1	0.086	0.001	0.432	0.065	0.001
122	0.622	0.106	0.168	1	0.123	0.002	1	0.108	0.001	0.636	0.193	0.02	1	0.086	0.001	0.137	0.065	0.001
123	0.136	0.154	0.209	1	0.084	0.002	0.829	0.108	0.005	0.452	0.124	0.013	1	0.021	0.002	1	0.156	0.002
124	0.282	0.262	0.255	1	0.123	0.002	0.636	0.108	0.003	0.827	0.124	0.013	1	0.029	0.001	0.828	0.156	0.001
125	0.639	0.106	0.141	1	0.123	0.002	1	0.085	0.001	0.619	0.193	0.02	1	0.086	0.001	0.281	0.065	0.002
126	0.451	0.154	0.209	0.96	0.123	0.002	0.618	0.156	0.008	0.96	0.262	0.02	0.637	0.029	0.001	0.828	0.139	0.002
127	0.961	0.192	0.168	0.96	0.123	0.002	0.451	0.156	0.008	0.827	0.262	0.058	0.126	0.029	0.001	0.959	0.193	0.003
128	0.622	0.192	0.255	1	0.064	0.002	0.433	0.194	0.005	0.636	0.155	0.02	0.828	0.021	0.001	0.846	0.139	0.003
129	0.296	0.122	0.209	1	0.084	0.002	1	0.085	0.001	0.619	0.124	0.013	1	0.048	0.007	0.432	0.03	0
130	0.282	0.192	0.209	0.96	0.047	0.002	0.846	0.085	0.003	0.295	0.064	0.008	1	0.014	1	0.072	0.03	0.001
131	1	0.085	0.141	1	0.107	0.002	0.846	0.123	0.005	0.619	0.155	0.02	1	0.124	0.002	0.192	0.018	0
132	0.434	0.122	0.209	1	0.084	0.002	1	0.123	0.001	0.636	0.193	0.02	1	0.048	0.004	0.137	0.048	0.001
133	0.829	0.154	0.209	1	0.084	0.002	0.636	0.156	0.003	0.619	0.193	0.02	1	0.048	0.001	0.616	0.074	0.001
134	0.008	0.064	0.168	1	0.047	0.002	0.295	0.064	0.003	0.434	0.086	0.013	1	0.048	1	0.294	0.03	0
135	0.192	0.122	0.209	1	0.084	0.002	1	0.085	0.001	0.636	0.124	0.013	1	0.029	0.142	0.137	0.065	0
136	0.829	0.192	0.209	1	0.084	0.002	0.451	0.123	0.005	0.827	0.262	0.02	0.619	0.029	0.001	0.959	0.156	0.002
137	0.296	0.154	0.168	1	0.047	0.002	0.961	0.108	0.003	0.452	0.108	0.013	1	0.021	0.398	0.072	0.03	0
138	0.451	0.154	0.209	1	0.084	0.002	0.829	0.108	0.003	0.636	0.124	0.013	1	0.021	0.002	0.634	0.156	0.002
139	0.622	0.315	0.168	1	0.084	0.002	0.433	0.123	0.019	0.845	0.262	0.029	0.069	0.029	0.001	1	0.263	0.005
140	0.063	0.085	0.168	0.96	0.047	0.002	0.192	0.108	0.005	0.452	0.108	0.02	1	0.029	1	0.616	0.021	0
141	0.639	0.192	0.099	1	0.084	0.002	1	0.064	0.001	0.619	0.193	0.02	1	0.029	0.057	0.072	0.03	0.002
142	0.622	0.373	0.311	0.828	0.107	0.013	0.282	0.194	0.005	1	0.155	0.02	0.828	0.029	0.001	0.959	0.139	0.001
143	0.03	0.154	0.209	1	0.084	0.002	0.846	0.029	0.008	0.434	0.124	0.013	1	0.014	0.313	1	0.156	0.001
144	0.451	0.424	0.255	1	0.047	0.002	0.846	0.064	0.003	0.452	0.086	0.013	1	0.021	0.007	0.294	0.074	0.002
145	0.282	0.122	0.209	1	0.084	0.002	1	0.048	0.001	0.827	0.108	0.013	1	0.065	0.17	0.137	0.03	0
146	0.622	0.192	0.168	1	0.084	0.002	0.451	0.156	0.008	0.827	0.262	0.02	0.449	0.014	0.001	1	0.156	0.005
147	0.07	0.085	0.168	1	0.047	0.002	0.846	0.048	0.003	0.282	0.086	0.013	1	0.048	1	0.009	0.01	0
148	0.434	0.064	0.168	1	0.107	0.002	0.961	0.108	0.005	0.96	0.108	0.02	1	0.155	0.259	0.032	0.01	0
149	0.622	0.085	0.168	1	0.123	0.002	1	0.108	0.001	0.827	0.193	0.02	1	0.108	0.001	0.281	0.03	0.001
150	0.434	0.122	0.209	1	0.084	0.002	1	0.108	0.001	0.619	0.155	0.02	1	0.029	0.007	0.281	0.107	0
151	0.829	0.106	0.168	1	0.123	0.002	0.961	0.085	0.001	0.452	0.193	0.02	1	0.108	0.001	0.45	0.065	0.001
152	0.639	0.154	0.168	1	0.064	0.002	1	0.123	0.005	0.827	0.124	0.02	1	0.029	0.057	0.032	0.021	0
153	0.451	0.373	0.255	1	0.084	0.002	0.829	0.085	0.003	0.619	0.108	0.013	1	0.029	0.004	0.45	0.065	0.002

Table A.35: Dataset 1 at Stage 20% - p values splits- Part 3/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
154	0.434	0.122	0.168	1	0.064	0.002	1	0.156	0.005	0.827	0.155	0.02	1	0.048	0.019	0.072	0.018	0
155	0.829	0.192	0.168	1	0.084	0.002	0.618	0.123	0.005	0.636	0.262	0.02	0.637	0.021	0.001	1	0.156	0.003
156	0.639	0.106	0.168	1	0.084	0.002	1	0.108	0.001	0.619	0.193	0.02	1	0.048	0.004	0.616	0.048	0.001
157	0.296	0.154	0.209	1	0.084	0.002	1	0.085	0.008	0.619	0.124	0.013	1	0.021	0.004	0.616	0.139	0.002
158	0.829	0.315	0.168	0.96	0.123	0.002	0.618	0.123	0.005	0.845	0.262	0.029	0.449	0.029	0.001	1	0.156	0.005
159	0.622	0.154	0.209	1	0.084	0.002	1	0.108	0.001	0.845	0.193	0.02	1	0.029	0.002	0.281	0.048	0.001
160	0.026	0.085	0.141	1	0.047	0.002	1	0.021	0.003	0.192	0.086	0.013	1	0.029	1	0.192	0.03	0
161	0.622	0.122	0.056	1	0.107	0.002	1	0.029	0.003	0.619	0.193	0.029	1	0.065	0.076	0.281	0.065	0.005
162	0.829	0.192	0.209	1	0.084	0.002	0.618	0.123	0.005	0.845	0.262	0.02	0.828	0.021	0.001	1	0.156	0.002
163	0.639	0.262	0.255	1	0.107	0.013	1	0.156	0.005	1	0.193	0.02	1	0.155	0.001	0.072	0.018	0
164	0.128	0.085	0.141	1	0.047	0.004	0.961	0.085	0.003	0.295	0.086	0.013	1	0.029	1	0.009	0.021	0
165	0.961	0.122	0.141	1	0.084	0.002	1	0.064	0.001	0.434	0.193	0.029	1	0.086	0.001	0.616	0.065	0.005
166	0.008	0.154	0.168	1	0.047	0.002	0.451	0.048	0.003	0.192	0.086	0.013	1	0.029	1	0.634	0.065	0
167	0.296	0.262	0.209	1	0.084	0.002	1	0.108	0.008	0.845	0.124	0.013	1	0.021	0.004	0.45	0.139	0.001
168	0.063	0.064	0.168	1	0.047	0.002	0.451	0.064	0.003	0.619	0.086	0.013	1	0.065	1	0.294	0.03	0
169	0.434	0.154	0.209	1	0.084	0.002	1	0.085	0.001	0.452	0.155	0.013	1	0.029	0.142	0.432	0.107	0
170	0.639	0.315	0.255	0.846	0.107	0.013	0.961	0.123	0.003	1	0.155	0.02	0.844	0.155	0.001	0.294	0.018	0
171	0.282	0.048	0.141	1	0.064	0.001	1	0.085	0.003	0.452	0.124	0.013	1	0.029	1	0	0.006	0
172	0.434	0.315	0.209	1	0.123	0.002	1	0.108	0.003	0.636	0.124	0.013	1	0.029	0.002	0.634	0.156	0.001
173	0.961	0.154	0.209	1	0.084	0.002	0.636	0.123	0.005	0.827	0.262	0.02	0.844	0.029	0.001	0.959	0.156	0.002
174	0.639	0.154	0.168	1	0.047	0.002	0.846	0.064	0.003	0.619	0.086	0.013	1	0.065	0.603	0.128	0.021	0
175	0.451	0.122	0.168	1	0.084	0.002	1	0.085	0.001	0.619	0.193	0.02	1	0.048	0.076	0.45	0.048	0
176	0.282	0.262	0.209	1	0.084	0.002	1	0.085	0.008	0.827	0.108	0.013	1	0.021	0.004	0.634	0.139	0.001
177	0.846	0.106	0.099	1	0.084	0.002	1	0.108	0.001	0.636	0.193	0.02	1	0.048	0.004	0.616	0.107	0.001
178	0.451	0.122	0.168	1	0.084	0.002	1	0.029	0.001	0.295	0.124	0.02	1	0.048	0.398	0.616	0.065	0.001
179	0.622	0.373	0.255	1	0.047	0.002	0.846	0.108	0.003	0.636	0.086	0.013	1	0.021	0.012	0.616	0.107	0.001
180	0.639	0.122	0.141	1	0.064	0.002	0.961	0.085	0.003	0.619	0.124	0.02	1	0.048	0.142	0.432	0.021	0
181	0.07	0.064	0.168	1	0.047	0.002	0.961	0.085	0.003	0.452	0.086	0.013	1	0.029	1	0.027	0.018	0
182	0.639	0.424	0.255	0.96	0.123	0.002	0.636	0.194	0.005	0.96	0.262	0.02	0.828	0.029	0.001	0.959	0.156	0.002
183	0.639	0.122	0.075	1	0.084	0.002	1	0.085	0.001	0.452	0.193	0.02	1	0.029	0.1	0.828	0.065	0.001
184	0.192	0.122	0.209	1	0.084	0.002	1	0.064	0.001	0.619	0.124	0.013	1	0.029	0.398	0.45	0.107	0
185	0.296	0.154	0.209	1	0.084	0.002	1	0.108	0.005	0.636	0.124	0.013	1	0.014	0.004	0.634	0.156	0.001
186	0.008	0.064	0.209	0.96	0.047	0.004	0.295	0.064	0.005	0.434	0.064	0.013	1	0.014	1	0.846	0.139	0.001
187	0.451	0.154	0.255	1	0.064	0.002	1	0.123	0.003	0.96	0.155	0.02	1	0.014	0.007	0.294	0.074	0.001
188	0.026	0.064	0.209	1	0.084	0.002	0.846	0.048	0.003	0.434	0.108	0.013	1	0.029	1	0.616	0.074	0
189	0.026	0.085	0.209	1	0.047	0.008	0.961	0.048	0.003	0.282	0.064	0.008	1	0.008	1	0.634	0.139	0
190	0.282	0.154	0.209	1	0.047	0.002	1	0.085	0.003	0.452	0.086	0.013	1	0.021	0.977	0.281	0.065	0
191	0.128	0.064	0.099	1	0.047	0.002	1	0.085	0.003	0.192	0.108	0.013	1	0.065	1	0.137	0.018	0
192	0.622	0.424	0.209	0.96	0.123	0.002	0.618	0.194	0.005	1	0.262	0.029	0.96	0.029	0.001	1	0.156	0.002
193	0.622	0.315	0.209	1	0.084	0.002	1	0.085	0.005	0.636	0.124	0.013	1	0.029	0.002	0.846	0.156	0.001
194	0.451	0.064	0.141	1	0.064	0.004	1	0.048	0.003	0.96	0.124	0.02	1	0.014	0.259	0.281	0.065	0.002
195	0.829	0.122	0.255	1	0.107	0.002	1	0.123	0.003	1	0.155	0.02	1	0.014	0.002	0.634	0.074	0.002
196	1	0.122	0.141	1	0.107	0.001	0.846	0.108	0.005	0.434	0.108	0.02	1	0.155	0.211	0.137	0.018	0
197	0.296	0.029	0.099	1	0.047	0.001	1	0.029	0.008	0.192	0.108	0.013	1	0.426	1	0.065	0.01	0
198	0.451	0.122	0.209	1	0.084	0.002	1	0.108	0.003	0.636	0.124	0.013	1	0.029	0.004	0.828	0.156	0.001
199	0.451	0.424	0.209	1	0.084	0.002	1	0.085	0.003	0.619	0.124	0.013	1	0.029	0.002	0.959	0.156	0.002
200	0.07	0.064	0.168	1	0.047	0.002	1	0.064	0.003	0.619	0.086	0.013	1	0.029	1	0.128	0.021	0
201	0.192	0.064	0.168	1	0.047	0.002	0.846	0.048	0.003	0.452	0.086	0.013	1	0.065	1	0.065	0.018	0
202	0.639	0.315	0.209	1	0.084	0.002	1	0.048	0.008	0.452	0.124	0.02	1	0.021	0.002	1	0.156	0.002
203	0.846	0.122	0.168	1	0.123	0.002	1	0.085	0.008	0.636	0.262	0.041	0.828	0.029	0.001	0.634	0.156	0.008
204	0.846	0.122	0.209	1	0.084	0.002	1	0.085	0.005	0.845	0.262	0.02	1	0.021	0.001	0.828	0.156	0.001

Table A.36: Dataset 1 at Stage 20% - p values splits- Part 4/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
205	0.451	0.154	0.255	1	0.047	0.002	1	0.108	0.003	0.636	0.108	0.013	1	0.021	0.057	0.616	0.139	0.002
206	0.622	0.122	0.209	1	0.084	0.002	1	0.108	0.005	0.636	0.124	0.013	1	0.014	0.004	0.959	0.156	0.001
207	0.063	0.154	0.141	1	0.064	0.002	1	0.01	0.003	0.452	0.108	0.02	1	0.029	1	0.192	0.03	0.001
208	0.961	0.262	0.141	1	0.123	0.002	1	0.085	0.012	1	0.193	0.02	0.96	0.029	0.001	0.634	0.074	0.003
209	0.451	0.262	0.255	1	0.107	0.013	1	0.156	0.005	1	0.262	0.029	1	0.155	0.001	0.128	0.03	0.058
210	0.639	0.192	0.209	1	0.123	0.002	1	0.123	0.012	1	0.262	0.029	0.96	0.065	0.001	0.634	0.107	0.002
211	0.136	0.154	0.168	1	0.047	0.004	0.961	0.064	0.003	0.452	0.108	0.013	1	0.021	1	0.065	0.03	0
212	0.639	0.192	0.209	1	0.084	0.002	1	0.108	0.003	0.636	0.124	0.013	1	0.014	0.004	0.959	0.156	0.002
213	1	0.122	0.141	1	0.084	0.002	1	0.085	0.003	0.636	0.262	0.02	1	0.021	0.001	1	0.156	0.003
214	0.961	0.315	0.255	1	0.107	0.013	0.961	0.156	0.005	1	0.262	0.02	0.96	0.108	0.001	0.45	0.107	0.003
215	0.128	0.154	0.168	1	0.064	0.002	1	0.014	0.008	0.434	0.124	0.013	1	0.014	1	0.959	0.139	0.002
216	0.961	0.192	0.168	1	0.084	0.002	1	0.085	0.012	0.636	0.262	0.02	1	0.014	0.002	0.959	0.139	0.002
217	0.829	0.373	0.209	1	0.123	0.013	1	0.123	0.003	0.845	0.317	0.02	0.96	0.065	0.001	0.828	0.156	0.002
218	0.07	0.154	0.209	1	0.084	0.002	1	0.064	0.001	0.452	0.124	0.013	1	0.029	1	0.128	0.065	0
219	0.296	0.262	0.255	1	0.047	0.002	0.961	0.108	0.005	0.452	0.048	0.008	1	0.014	1	0.192	0.065	0
220	0.961	0.154	0.209	1	0.084	0.002	1	0.085	0.003	0.96	0.262	0.02	1	0.029	0.001	0.959	0.156	0.002
221	0.296	0.064	0.255	1	0.064	0.002	1	0.064	0.001	1	0.086	0.02	1	0.021	0.874	0.616	0.139	0
222	0.829	0.154	0.168	1	0.047	0.004	0.829	0.029	0.003	0.192	0.086	0.013	1	0.065	1	0.137	0.021	0
223	1	0.154	0.168	1	0.084	0.002	1	0.085	0.012	0.845	0.155	0.02	1	0.014	0.002	1	0.156	0.001
224	0.622	0.154	0.255	1	0.064	0.002	1	0.156	0.001	1	0.124	0.02	1	0.014	0.004	1	0.139	0.001
225	0.192	0.154	0.255	1	0.047	0.004	1	0.085	0.001	0.619	0.086	0.013	1	0.021	1	0.616	0.139	0.001
226	1	0.106	0.141	1	0.084	0.002	1	0.123	0.005	0.845	0.193	0.02	1	0.021	0.002	1	0.156	0.001
227	0.192	0.064	0.168	1	0.064	0.001	1	0.123	0.003	0.636	0.108	0.02	1	0.029	1	0.192	0.018	0
228	0.282	0.154	0.141	1	0.047	0.004	0.846	0.064	0.003	0.295	0.086	0.013	1	0.021	1	0.616	0.048	0
229	0.296	0.192	0.209	1	0.047	0.004	1	0.108	0.001	0.619	0.086	0.013	1	0.021	1	0.634	0.139	0.002
230	0.296	0.154	0.209	1	0.084	0.002	1	0.108	0.008	0.619	0.124	0.013	1	0.014	0.874	0.846	0.156	0.001
231	0.282	0.154	0.168	1	0.047	0.002	0.846	0.085	0.003	0.452	0.086	0.013	1	0.029	1	0.294	0.048	0
232	1	0.262	0.141	1	0.123	0.002	1	0.123	0.005	0.845	0.317	0.02	0.96	0.065	0.001	1	0.156	0.003
233	0.136	0.192	0.209	1	0.047	0.004	1	0.085	0.003	0.434	0.086	0.013	1	0.014	1	0.281	0.074	0.002
234	1	0.373	0.255	1	0.107	0.013	0.961	0.156	0.005	1	0.193	0.02	0.96	0.108	0.001	0.846	0.139	0.001
235	0.136	0.122	0.209	1	0.084	0.002	1	0.064	0.001	0.619	0.108	0.013	1	0.029	1	0.294	0.048	0
236	1	0.373	0.255	1	0.107	0.013	1	0.194	0.005	1	0.262	0.02	1	0.108	0.001	0.432	0.107	0.013
237	1	0.154	0.168	1	0.064	0.004	1	0.123	0.001	0.96	0.155	0.02	1	0.014	0.004	1	0.107	0.005
238	0.063	0.154	0.168	1	0.084	0.002	1	0.029	0.003	0.434	0.108	0.013	1	0.029	1	0.634	0.065	0
239	0.846	0.154	0.141	1	0.084	0.004	1	0.064	0.012	0.827	0.193	0.02	1	0.014	0.002	1	0.156	0.005
240	0.136	0.064	0.209	1	0.047	0.004	1	0.064	0.003	0.619	0.064	0.013	1	0.014	1	0.432	0.107	0
241	0.296	0.122	0.209	1	0.084	0.002	1	0.108	0.008	0.827	0.108	0.013	1	0.014	1	0.634	0.156	0.001
242	0.434	0.048	0.075	1	0.084	0.002	1	0.029	0.001	0.619	0.155	0.02	1	0.048	1	0.634	0.048	0.001
243	0.622	0.424	0.255	1	0.047	0.004	0.829	0.085	0.001	0.282	0.086	0.008	1	0.014	1	0.959	0.107	0.002
244	0.282	0.154	0.168	1	0.084	0.002	1	0.064	0.001	0.295	0.124	0.013	1	0.029	1	0.634	0.107	0
245	1	0.315	0.141	1	0.123	0.002	1	0.123	0.005	0.827	0.317	0.029	0.96	0.065	0.001	1	0.263	0.013
246	0.136	0.154	0.209	1	0.084	0.008	1	0.021	0.008	0.282	0.124	0.013	1	0.01	1	1	0.156	0.001
247	0.451	0.192	0.209	1	0.084	0.008	1	0.108	0.012	0.619	0.124	0.013	1	0.01	1	1	0.156	0.001
248	0.829	0.122	0.141	1	0.064	0.002	1	0.085	0.005	0.619	0.108	0.02	1	0.021	1	0.616	0.021	0
249	1	0.472	0.141	1	0.123	0.002	1	0.108	0.012	0.827	0.317	0.029	0.828	0.029	0.001	1	0.317	0.013
250	0.846	0.154	0.141	1	0.084	0.004	1	0.029	0.008	0.827	0.155	0.02	1	0.01	0.528	1	0.317	0.003
251	0.961	0.315	0.099	1	0.315	0.004	1	0.064	0.04	0.96	0.426	0.077	0.292	0.021	0	1	0.515	0.008
252	0.192	0.154	0.209	1	0.047	0.008	1	0.108	0.001	0.434	0.086	0.013	1	0.01	1	1	0.139	0.002
253	0.846	0.122	0.099	1	0.064	0.008	1	0.123	0.003	1	0.124	0.02	1	0.01	0.874	1	0.139	0.005
254	0.434	0.064	0.141	1	0.084	0.004	1	0.064	0.005	0.845	0.086	0.008	1	0.065	1	1	0.156	0.001
255	1	0.424	0.209	1	0.123	0.013	0.961	0.376	0.003	1	0.193	0.02	1	0.108	0.001	1	0.156	0.019
256	0.296	1	0.819	0.282	0.991	1	0.451	0.426	1	0.137	0.426	1	1	0.475	1	0.137	0.887	1

Table A.37: Dataset 1 at Stage 20% - p values splits- Part 5/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
1	1	0.008	0.003	0.192	0.014	0	1	0.008	0	0.008	0.001	0	0.027	0.005	0	0.072	0.001	0
2	0.961	0.001	0.003	0.129	0.014	0	1	0.008	0	0.063	0.003	0	0.069	0.008	0	0.128	0.001	0.001
3	0.961	0.001	0.003	0.138	0.014	0	1	0.008	0	0.031	0.003	0	0.069	0.008	0	0.072	0.001	0.002
4	1	0.008	0.003	0.828	0.02	0	0.618	0.005	0	0.192	0.003	0	0.009	0.005	0	0.009	0.001	0
5	0.829	0.001	0.003	0.129	0.005	0.001	1	0.008	0.001	0.026	0.003	0	0.28	0.005	0	0.192	0.003	0.002
6	1	0.01	0.003	0.452	0.014	0	0.829	0.005	0	0.006	0.003	0	0.28	0.01	0	0.032	0.001	0.001
7	0.829	0.001	0.003	0.295	0.014	0	0.846	0.008	0	0.063	0.001	0	0.28	0.01	0	0.072	0.001	0.001
8	0.829	0.001	0.003	0.129	0.014	0.002	1	0.008	0.003	0	0.003	0	0.189	0.005	0	0.128	0.005	0.001
9	0.639	0.001	0.003	0.434	0.014	0.001	1	0.008	0	0.063	0.003	0	0.135	0.008	0	0.072	0.001	0.002
10	0.846	0.001	0.003	0.295	0.014	0.001	1	0.008	0.005	0.008	0.001	0	0.063	0.008	0.001	0.072	0.001	0.001
11	0.639	0.001	0.003	0.129	0.005	0	0.961	0.008	0	0.031	0.001	0	0.431	0.005	0.001	0.45	0.003	0.001
12	0.961	0.001	0.003	0.192	0.014	0	1	0.008	0	0.026	0.005	0	0.069	0.008	0	0.192	0.003	0.002
13	1	0.005	0.003	0.129	0.014	0	1	0.008	0	0.006	0.005	0	0.031	0.008	0	0.294	0.003	0.002
14	1	0.001	0.003	0.622	0.01	0	0.829	0.01	0	0.026	0.001	0	0.431	0.01	0	0.128	0.001	0
15	0.846	0.001	0.003	0.192	0.01	0.001	1	0.008	0	0.137	0.003	0	0.126	0.008	0	0.137	0.001	0.002
16	0.846	0.001	0.003	0.129	0.01	0	0.961	0.008	0	0.295	0.005	0	0.135	0.008	0	0.192	0.001	0.002
17	1	0.005	0.003	0.129	0.014	0	1	0.008	0	0.063	0.008	0	0.063	0.008	0.001	0.281	0.005	0.002
18	0.622	0.001	0.003	0.138	0.014	0.001	1	0.005	0.003	0.031	0.003	0	0.449	0.01	0	0.192	0.003	0.002
19	0.639	0.001	0.003	0.434	0.014	0.002	1	0.008	0.003	0.026	0.003	0	0.135	0.008	0.001	0.137	0.001	0.001
20	0.846	0.003	0.003	0.639	0.014	0.001	1	0.005	0.001	0.026	0.003	0	0.135	0.01	0	0.032	0.001	0.002
21	0.622	0.001	0.003	0.282	0.014	0.001	0.961	0.005	0	0.026	0.001	0	0.619	0.01	0.001	0.432	0.003	0.001
22	0.829	0.001	0.003	0.192	0.014	0.001	1	0.008	0.003	0.128	0.003	0	0.135	0.008	0.001	0.128	0.001	0.001
23	1	0.005	0.003	0.192	0.014	0	1	0.008	0	0.282	0.005	0	0.069	0.005	0	0.072	0.001	0.002
24	1	0.005	0.003	0.295	0.014	0	1	0.005	0	0.008	0.005	0	0.135	0.01	0.001	0.192	0.001	0.002
25	1	0.005	0.003	0.129	0.014	0	1	0.008	0.001	0.006	0.008	0	0.069	0.008	0	0.137	0.003	0.003
26	0.829	0.001	0.003	0.282	0.014	0.001	1	0.008	0.003	0.063	0.003	0	0.126	0.008	0.002	0.137	0.001	0.001
27	0.961	0.001	0.003	0.063	0.014	0.001	1	0.008	0.001	0.026	0.008	0	0.28	0.008	0.001	0.294	0.003	0.002
28	1	0.003	0.003	0.282	0.014	0.001	1	0.008	0.001	0.031	0.005	0	0.135	0.008	0	0.128	0.001	0.003
29	1	0.005	0.003	0.295	0.014	0.001	1	0.008	0.001	0.006	0.003	0	0.135	0.008	0	0.072	0.001	0.001
30	0.622	0.001	0.003	0.639	0.014	0.001	1	0.005	0.005	0.026	0.001	0	0.28	0.01	0.001	0.065	0.001	0.002
31	0.639	0.001	0.003	0.828	0.014	0.002	0.961	0.003	0.003	0.031	0.001	0	0.431	0.01	0.001	0.072	0.001	0.001
32	0.829	0.001	0.003	0.434	0.014	0.002	1	0.008	0.005	0.008	0.003	0	0.189	0.008	0.001	0.128	0.001	0.001
33	0.434	0.001	0.003	0.639	0.014	0.001	1	0.008	0.005	0.031	0.001	0	0.28	0.01	0.001	0.137	0.001	0.001
34	1	0.003	0.003	0.192	0.014	0.001	1	0.008	0.005	0.008	0.005	0	0.069	0.008	0.001	0.192	0.001	0.001
35	0.829	0.001	0.003	0.192	0.014	0.001	1	0.008	0.003	0.063	0.003	0	0.135	0.008	0.002	0.281	0.003	0.001
36	0.639	0.003	0.003	0.622	0.014	0.002	1	0.003	0.005	0.026	0.003	0	0.431	0.01	0	0.065	0.003	0.002
37	0.639	0.001	0.003	0.138	0.005	0.002	1	0.008	0.005	0.063	0.003	0	0.292	0.008	0.001	0.294	0.003	0.001
38	0.846	0.001	0.003	0.639	0.014	0.001	1	0.008	0.001	0.063	0.008	0	0.135	0.008	0	0.072	0.003	0.003
39	0.961	0.005	0.003	0.452	0.014	0.001	1	0.005	0.001	0.026	0.008	0	0.189	0.01	0	0.072	0.001	0.003
40	0.829	0.003	0.003	0.452	0.014	0.001	1	0.005	0.001	0.128	0.003	0	0.189	0.01	0	0.072	0.001	0.003
41	0.622	0.001	0.003	0.192	0.005	0.002	1	0.008	0.005	0.031	0.001	0	0.431	0.008	0.002	0.432	0.003	0.001
42	0.829	0.001	0.003	0.192	0.014	0.001	1	0.008	0.005	0.192	0.003	0	0.135	0.008	0.001	0.072	0.001	0.001
43	0.451	0.001	0.003	0.828	0.028	0	0.846	0.008	0	0.192	0.003	0	0.28	0.01	0.001	0.137	0.001	0.003
44	0.639	0.001	0.003	0.192	0.007	0.004	1	0.008	0.005	0.008	0.003	0	0.28	0.008	0.001	0.281	0.005	0.001
45	0.961	0.01	0.003	1	0.028	0.001	0.618	0.001	0	0.137	0.003	0	0.28	0.01	0	0.032	0	0.001
46	0.622	0.001	0.003	0.452	0.014	0.001	1	0.008	0	0.282	0.003	0	0.189	0.01	0	0.192	0.001	0.002
47	0.639	0.001	0.003	0.452	0.014	0.001	1	0.008	0	0.192	0.003	0	0.28	0.008	0	0.128	0.001	0.003
48	0.434	0.001	0.003	0.639	0.014	0.002	1	0.005	0.005	0.031	0.003	0	0.28	0.01	0.001	0.128	0.001	0.002
49	0.639	0.001	0.003	0.063	0.005	0.001	1	0.008	0	0.452	0.003	0	0.28	0.005	0.001	0.634	0.003	0.002
50	1	0.003	0.003	0.192	0.014	0.001	1	0.008	0.001	0.128	0.008	0	0.126	0.008	0	0.137	0.003	0.003
51	0.639	0.001	0.003	0.031	0.01	0.002	1	0.008	0.005	0.07	0.003	0	0.28	0.008	0.001	0.432	0.003	0.003

Table A.38: Dataset 1 at Stage 50% - p values splits- Part 1/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
52	1	0.001	0.003	0.063	0.014	0	0.846	0.008	0	0.026	0.008	0	0.431	0.008	0.002	0.846	0.005	0.002
53	0.846	0.003	0.003	0.192	0.014	0.002	1	0.008	0.005	0.026	0.003	0	0.135	0.008	0.001	0.192	0.003	0.003
54	0.639	0.001	0.003	0.282	0.014	0.002	1	0.008	0.005	0.137	0.003	0	0.189	0.008	0	0.137	0.003	0.001
55	0.846	0.001	0.003	0.828	0.028	0	0.961	0.008	0	0.295	0.008	0	0.063	0.008	0.001	0.192	0.001	0.002
56	0.846	0.005	0.003	0.452	0.014	0.001	1	0.005	0	0.282	0.005	0	0.135	0.01	0	0.128	0.001	0.002
57	0.639	0.001	0.003	0.622	0.014	0.002	1	0.005	0.005	0.063	0.003	0	0.189	0.01	0	0.128	0.001	0.002
58	1	0.001	0.003	0.027	0.01	0.002	1	0.008	0.005	0.128	0.008	0	0.189	0.008	0	0.281	0.003	0.003
59	1	0.003	0.003	0.063	0.005	0.002	1	0.008	0.005	0.07	0.003	0	0.292	0.008	0	0.192	0.003	0.003
60	1	0.005	0.003	0.129	0.014	0	1	0.008	0	0.192	0.008	0	0.069	0.008	0	0.432	0.001	0.003
61	0.829	0.003	0.003	0.282	0.014	0.002	1	0.008	0.003	0.063	0.003	0	0.135	0.008	0.001	0.294	0.003	0.002
62	0.961	0.001	0.003	0.282	0.014	0.001	1	0.008	0	0.137	0.008	0	0.126	0.008	0	0.294	0.003	0.003
63	0.639	0.001	0.003	0.282	0.014	0.002	1	0.008	0.005	0.128	0.003	0	0.135	0.008	0	0.281	0.001	0.002
64	0.829	0.003	0.003	0.063	0.005	0.001	1	0.008	0.001	0.636	0.008	0	0.28	0.003	0	0.281	0.003	0.002
65	0.846	0.003	0.003	0.129	0.005	0.002	1	0.008	0.001	0.192	0.003	0	0.292	0.008	0	0.294	0.003	0.002
66	0.451	0.001	0.003	0.452	0.014	0.002	1	0.008	0.005	0.07	0.003	0	0.292	0.01	0.001	0.192	0.001	0.001
67	1	0.003	0.003	0.282	0.014	0.002	1	0.008	0.003	0	0.008	0	0.189	0.008	0	0.192	0.005	0.003
68	0.846	0.003	0.003	0.063	0.005	0.002	1	0.008	0.005	0.07	0.003	0	0.292	0.008	0	0.432	0.003	0.002
69	0.434	0.001	0.003	0.434	0.014	0.002	1	0.005	0.005	0.031	0.001	0	0.637	0.01	0.002	0.294	0.003	0.001
70	0.829	0.005	0.003	0.828	0.014	0	1	0.008	0	0.063	0.008	0	0.28	0.01	0.001	0.192	0.003	0.002
71	0.961	0.001	0.003	0.192	0.014	0	1	0.008	0	0.295	0.008	0	0.126	0.008	0.001	0.432	0.003	0.002
72	0.639	0.001	0.003	0.96	0.028	0.002	1	0.008	0.005	0.137	0.003	0	0.063	0.008	0.001	0.065	0.001	0.001
73	0.829	0.003	0.003	0.282	0.014	0.002	1	0.008	0.005	0.07	0.003	0	0.126	0.008	0	0.281	0.003	0.003
74	0.639	0.003	0.003	0.452	0.014	0.002	1	0.005	0.005	0.031	0.003	0	0.189	0.01	0.001	0.137	0.001	0.003
75	0.622	0.001	0.003	0.452	0.014	0.001	1	0.005	0.003	0.063	0.005	0	0.449	0.01	0.001	0.192	0.003	0.003
76	0.829	0.003	0.003	0.622	0.014	0.002	1	0.005	0.005	0.031	0.003	0	0.28	0.01	0.001	0.137	0.001	0.002
77	0.639	0.001	0.003	0.071	0.005	0.002	1	0.008	0.005	0.192	0.003	0	0.431	0.008	0.001	0.45	0.003	0.001
78	0.829	0.003	0.003	0.282	0.014	0.001	1	0.008	0.003	0.137	0.005	0	0.126	0.008	0.001	0.294	0.003	0.002
79	0.846	0.001	0.003	0.129	0.005	0.002	1	0.008	0.005	0.282	0.003	0	0.292	0.008	0	0.281	0.003	0.001
80	0.829	0.001	0.003	0.452	0.014	0.001	1	0.008	0.001	0.128	0.008	0	0.28	0.008	0	0.281	0.003	0.003
81	0.639	0.003	0.003	0.138	0.01	0.002	1	0.008	0.005	0.063	0.003	0	0.431	0.008	0.001	0.432	0.003	0.002
82	0.829	0.001	0.003	0.622	0.02	0	1	0.008	0.001	0.137	0.005	0	0.069	0.008	0	0.45	0.003	0.003
83	0.434	0.001	0.003	0.846	0.014	0.002	1	0.005	0.005	0.031	0.001	0	0.449	0.01	0.001	0.137	0.003	0.002
84	1	0.003	0.003	0.622	0.014	0.001	1	0.005	0.001	0.128	0.008	0	0.135	0.01	0	0.128	0.001	0.003
85	1	0.005	0.003	0.452	0.014	0.001	1	0.005	0.001	0.137	0.008	0	0.189	0.01	0	0.192	0.001	0.003
86	0.451	0.001	0.003	0.96	0.028	0.001	1	0.008	0.001	0.295	0.003	0	0.135	0.008	0	0.137	0.001	0.003
87	0.639	0.001	0.003	0.639	0.005	0.004	1	0.01	0.005	0.008	0.001	0	0.449	0.01	0.001	0.192	0.003	0.001
88	0.639	0.003	0.003	0.071	0.005	0.002	1	0.008	0.005	0.282	0.003	0	0.431	0.01	0.001	0.281	0.003	0.001
89	0.846	0.003	0.003	0.192	0.014	0.001	1	0.008	0.005	0.192	0.005	0	0.126	0.008	0.001	0.281	0.001	0.002
90	0.829	0.003	0.003	0.282	0.014	0.002	1	0.008	0.005	0.137	0.003	0	0.126	0.008	0	0.432	0.003	0.002
91	0.639	0.001	0.003	0.452	0.014	0.002	1	0.005	0.005	0.026	0.008	0	0.449	0.01	0	0.137	0.005	0.003
92	0.622	0.001	0.003	0.639	0.014	0.001	1	0.005	0.005	0.128	0.003	0	0.28	0.01	0.001	0.137	0.001	0.002
93	0.622	0.001	0.003	0.452	0.014	0.001	1	0.008	0.005	0.137	0.003	0	0.292	0.01	0.001	0.137	0.001	0.002
94	0.451	0.001	0.003	0.622	0.014	0.002	1	0.005	0.005	0.063	0.003	0	0.431	0.01	0.001	0.192	0.003	0.002
95	0.639	0.001	0.003	0.192	0.014	0.002	1	0.005	0.005	0.07	0.003	0	0.449	0.01	0.001	0.281	0.003	0.001
96	0.639	0.001	0.003	0.129	0.007	0.002	1	0.008	0.003	0.192	0.008	0	0.28	0.008	0	0.616	0.003	0.003
97	0.622	0.003	0.003	0.452	0.014	0.002	1	0.008	0.005	0.128	0.003	0	0.28	0.008	0	0.281	0.003	0.003
98	0.639	0.001	0.003	0.295	0.014	0.001	0.961	0.005	0.001	0.031	0.008	0	0.637	0.01	0.002	0.616	0.003	0.003
99	0.829	0.003	0.003	0.282	0.014	0.002	1	0.005	0.005	0.137	0.008	0	0.292	0.01	0	0.192	0.003	0.003
100	0.639	0.001	0.003	0.622	0.014	0.002	1	0.008	0.005	0.137	0.003	0	0.292	0.008	0	0.137	0.003	0.002
101	0.961	0.001	0.003	0.295	0.014	0.001	1	0.008	0	0.192	0.008	0	0.28	0.01	0.001	0.432	0.003	0.003
102	0.639	0.001	0.003	0.846	0.02	0.001	1	0.008	0.001	0.128	0.008	0	0.135	0.008	0	0.294	0.003	0.003

Table A.39: Dataset 1 at Stage 50% - p values splits- Part 2/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
103	0.639	0.003	0.003	0.138	0.01	0.002	1	0.008	0.005	0.031	0.003	0	0.449	0.008	0.002	0.828	0.003	0.001
104	1	0.008	0.003	0.846	0.02	0	0.636	0.005	0	0.137	0.008	0	0.135	0.01	0.001	0.192	0.001	0.002
105	0.639	0.003	0.003	0.622	0.014	0.001	1	0.005	0.005	0.07	0.005	0	0.28	0.01	0.001	0.281	0.001	0.002
106	0.639	0.003	0.003	0.282	0.01	0.002	1	0.005	0.005	0.07	0.003	0	0.449	0.01	0.001	0.281	0.003	0.002
107	0.846	0.003	0.003	0.063	0.007	0.001	1	0.008	0	0.282	0.008	0	0.292	0.008	0.001	0.959	0.003	0.002
108	0.639	0.003	0.003	0.846	0.014	0.001	1	0.008	0.005	0.031	0.003	0	0.292	0.01	0.002	0.192	0.003	0.001
109	0.829	0.001	0.003	0.828	0.028	0	1	0.008	0	0.434	0.005	0	0.069	0.008	0	0.432	0.001	0.002
110	0.451	0.003	0.003	0.452	0.014	0.002	1	0.005	0.005	0.07	0.003	0	0.449	0.01	0.001	0.294	0.003	0.003
111	1	0.008	0.003	0.295	0.02	0	1	0.008	0	0.07	0.008	0	0.031	0.008	0	0.959	0.007	0.003
112	0.829	0.005	0.003	1	0.028	0.001	1	0.003	0.005	0.434	0.003	0	0.135	0.01	0	0.065	0.001	0.003
113	0.639	0.001	0.003	0.434	0.01	0.002	1	0.008	0.005	0.137	0.003	0	0.449	0.01	0	0.281	0.003	0.002
114	0.829	0.001	0.003	0.295	0.014	0.002	1	0.008	0.005	0.031	0.003	0	0.28	0.008	0.001	0.432	0.005	0.003
115	0.639	0.003	0.003	0.138	0.01	0.002	1	0.008	0.005	0.137	0.003	0	0.292	0.008	0.001	0.45	0.003	0.003
116	0.829	0.003	0.003	0.129	0.01	0.002	1	0.008	0.005	0.137	0.003	0	0.292	0.008	0.001	0.828	0.003	0.002
117	0.639	0.003	0.003	0.434	0.01	0.002	1	0.005	0.005	0.192	0.005	0	0.449	0.008	0	0.281	0.003	0.003
118	0.961	0.003	0.003	0.639	0.02	0.001	1	0.008	0.005	0.063	0.005	0	0.069	0.008	0.001	0.294	0.001	0.003
119	1	0.005	0.003	0.295	0.014	0.001	1	0.008	0.001	0.031	0.005	0	0.28	0.01	0.002	0.432	0.007	0.002
120	0.846	0.003	0.003	0.622	0.02	0.001	1	0.008	0	0.434	0.008	0	0.126	0.008	0	0.294	0.003	0.003
121	0.451	0.001	0.003	0.622	0.014	0.002	1	0.005	0.005	0.063	0.005	0	0.449	0.01	0.001	0.294	0.003	0.003
122	0.451	0.001	0.003	0.846	0.028	0.001	1	0.008	0.005	0.295	0.003	0	0.126	0.008	0.002	0.192	0.001	0.001
123	0.846	0.008	0.003	0.828	0.028	0.001	0.829	0.003	0.001	0.845	0.008	0	0.135	0.01	0	0.128	0.001	0.003
124	0.829	0.003	0.003	0.452	0.014	0.001	1	0.003	0.001	0.295	0.005	0	0.449	0.01	0	0.281	0.003	0.002
125	0.846	0.005	0.003	0.96	0.02	0.001	1	0.005	0.005	0.128	0.008	0	0.135	0.01	0	0.192	0.003	0.003
126	0.639	0.003	0.003	0.622	0.01	0.002	1	0.005	0.005	0.07	0.003	0	0.449	0.01	0.001	0.281	0.003	0.002
127	0.846	0.001	0.003	0.622	0.02	0.001	1	0.008	0	0.636	0.005	0	0.135	0.008	0	0.281	0.003	0.003
128	0.639	0.003	0.003	0.828	0.014	0.002	1	0.005	0.005	0.128	0.003	0	0.189	0.01	0.001	0.281	0.003	0.003
129	1	0.005	0.003	0.622	0.02	0	1	0.008	0.001	0.07	0.008	0	0.069	0.008	0	0.45	0.005	0.003
130	0.622	0.003	0.003	0.639	0.02	0.001	1	0.008	0.005	0.137	0.003	0	0.135	0.008	0.002	0.432	0.005	0.002
131	0.622	0.001	0.003	0.295	0.01	0.001	1	0.005	0.001	0.452	0.005	0	0.449	0.01	0.001	0.616	0.003	0.002
132	0.829	0.003	0.003	0.622	0.02	0	1	0.008	0	0.827	0.008	0	0.135	0.008	0.001	0.432	0.001	0.002
133	0.622	0.003	0.003	0.846	0.028	0.001	1	0.008	0	0.619	0.005	0	0.135	0.01	0.001	0.281	0.001	0.003
134	0.639	0.003	0.003	0.639	0.02	0.002	1	0.008	0.005	0.07	0.003	0	0.135	0.008	0.001	0.432	0.003	0.003
135	0.622	0.003	0.003	0.828	0.02	0.002	1	0.008	0.005	0.07	0.003	0	0.135	0.008	0.001	0.432	0.003	0.002
136	0.639	0.003	0.003	0.434	0.014	0.001	1	0.008	0.005	0.137	0.003	0	0.431	0.01	0.001	0.432	0.003	0.003
137	0.622	0.003	0.003	0.828	0.014	0.002	1	0.008	0.005	0.07	0.003	0	0.292	0.01	0.001	0.294	0.003	0.003
138	0.639	0.005	0.003	0.846	0.02	0	1	0.008	0	0.128	0.008	0	0.292	0.01	0.001	0.432	0.003	0.003
139	0.829	0.001	0.003	0.846	0.028	0.001	1	0.008	0.005	0.452	0.003	0	0.126	0.008	0.001	0.128	0.001	0.002
140	0.961	0.001	0.003	0.452	0.014	0.002	1	0.005	0.005	0.137	0.008	0	0.431	0.01	0	0.281	0.003	0.003
141	0.622	0.003	0.003	0.434	0.014	0.002	1	0.005	0.005	0.063	0.003	0	0.637	0.01	0.002	0.616	0.003	0.001
142	0.639	0.001	0.003	0.452	0.014	0.002	1	0.005	0.005	0.128	0.005	0	0.431	0.01	0.001	0.294	0.003	0.003
143	0.639	0.001	0.003	0.452	0.01	0.002	1	0.008	0.005	0.137	0.003	0	0.449	0.01	0.001	0.432	0.003	0.002
144	0.961	0.005	0.003	0.622	0.02	0	1	0.008	0	0.192	0.008	0	0.069	0.008	0.001	0.828	0.005	0.002
145	0.829	0.001	0.003	0.452	0.014	0.002	1	0.008	0.005	0.026	0.005	0	0.449	0.01	0.002	0.432	0.005	0.001
146	0.961	0.005	0.004	0.639	0.014	0.002	0.961	0.005	0.003	0.031	0.005	0	0.619	0.01	0.001	0.432	0.003	0.002
147	0.639	0.001	0.003	0.295	0.014	0.002	1	0.008	0.005	0.137	0.008	0	0.449	0.01	0.001	0.45	0.003	0.003
148	0.622	0.001	0.003	1	0.028	0.004	1	0.005	0.005	0.137	0.003	0	0.28	0.01	0.001	0.072	0.001	0.002
149	1	0.008	0.003	0.828	0.028	0.001	0.961	0.005	0.005	0.619	0.005	0	0.189	0.01	0	0.072	0.001	0.003
150	0.829	0.001	0.003	0.828	0.028	0.002	1	0.005	0.005	0.295	0.003	0	0.189	0.01	0	0.192	0.001	0.002
151	0.639	0.001	0.003	0.846	0.028	0.002	1	0.008	0.005	0.295	0.003	0	0.135	0.008	0.001	0.281	0.001	0.001
152	0.451	0.001	0.003	1	0.028	0.001	1	0.008	0.005	0.192	0.003	0	0.28	0.01	0.002	0.192	0.001	0.001
153	0.622	0.003	0.003	0.96	0.02	0.002	1	0.005	0.005	0.128	0.003	0	0.189	0.01	0.001	0.294	0.001	0.003

Table A.40: Dataset 1 at Stage 50% - p values splits- Part 3/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
154	0.829	0.001	0.003	0.639	0.028	0.001	1	0.008	0.005	0.434	0.003	0	0.135	0.008	0.001	0.294	0.003	0.001
155	0.639	0.003	0.003	0.434	0.014	0.002	1	0.008	0.005	0.137	0.003	0	0.449	0.01	0.001	0.294	0.003	0.003
156	0.846	0.008	0.003	0.828	0.02	0.001	0.961	0.005	0.001	0.434	0.008	0	0.135	0.01	0.001	0.432	0.001	0.003
157	1	0.005	0.003	0.639	0.02	0.001	1	0.008	0.003	0.282	0.008	0	0.126	0.008	0	0.281	0.003	0.003
158	0.622	0.003	0.003	0.96	0.02	0.002	1	0.005	0.005	0.137	0.003	0	0.28	0.01	0.001	0.281	0.001	0.003
159	1	0.003	0.003	0.031	0.01	0.001	1	0.008	0.001	0.295	0.008	0	0.28	0.008	0.001	0.846	0.003	0.003
160	0.639	0.003	0.003	0.295	0.014	0.001	1	0.008	0.001	0.192	0.008	0	0.619	0.01	0.001	0.634	0.003	0.003
161	0.622	0.001	0.003	1	0.028	0.004	1	0.008	0.005	0.192	0.003	0	0.189	0.008	0.001	0.192	0.001	0.002
162	0.639	0.003	0.003	0.295	0.014	0.002	1	0.008	0.005	0.128	0.005	0	0.449	0.01	0.001	0.634	0.003	0.003
163	0.639	0.001	0.003	0.846	0.028	0.001	1	0.008	0	0.619	0.005	0	0.292	0.008	0	0.281	0.003	0.003
164	0.622	0.003	0.003	0.96	0.028	0.001	1	0.008	0.005	0.07	0.003	0	0.292	0.01	0.002	0.294	0.003	0.003
165	1	0.008	0.003	0.828	0.028	0.001	1	0.008	0.001	0.636	0.008	0	0.135	0.008	0	0.072	0.001	0.005
166	0.846	0.003	0.003	0.639	0.02	0	1	0.008	0	0.434	0.008	0	0.126	0.008	0.001	0.846	0.003	0.003
167	0.829	0.003	0.003	0.639	0.028	0.001	1	0.008	0	0.434	0.008	0	0.28	0.01	0.001	0.616	0.003	0.003
168	0.829	0.003	0.003	0.639	0.02	0.002	1	0.008	0.005	0.434	0.003	0	0.126	0.008	0	0.432	0.003	0.003
169	0.829	0.003	0.003	0.846	0.028	0.001	1	0.008	0.001	0.295	0.008	0	0.28	0.008	0	0.294	0.003	0.003
170	0.622	0.001	0.003	0.96	0.014	0.002	1	0.005	0.005	0.031	0.003	0	0.619	0.01	0.002	0.294	0.003	0.002
171	0.961	0.005	0.003	0.846	0.02	0.001	1	0.005	0.005	0.282	0.008	0	0.189	0.01	0	0.281	0.003	0.003
172	0.961	0.003	0.003	0.129	0.01	0.002	1	0.008	0.005	0.137	0.008	0	0.431	0.01	0.001	0.616	0.003	0.002
173	0.846	0.003	0.003	0.639	0.02	0.002	1	0.008	0.005	0.282	0.003	0	0.135	0.008	0.001	0.294	0.003	0.003
174	0.639	0.001	0.003	0.846	0.028	0.001	1	0.005	0.005	0.452	0.003	0	0.28	0.01	0.001	0.192	0.001	0.002
175	0.451	0.001	0.003	0.96	0.028	0.002	1	0.008	0.005	0.295	0.003	0	0.292	0.01	0.001	0.192	0.001	0.002
176	0.622	0.003	0.003	0.846	0.028	0.001	1	0.005	0.005	0.192	0.005	0	0.28	0.01	0.001	0.432	0.003	0.003
177	0.846	0.001	0.003	0.639	0.007	0.001	1	0.008	0.001	0.434	0.008	0	0.28	0.005	0	0.846	0.003	0.002
178	0.451	0.001	0.003	1	0.028	0.004	1	0.005	0.005	0.434	0.003	0	0.28	0.01	0.001	0.137	0.001	0.002
179	0.434	0.001	0.003	0.828	0.028	0.001	0.846	0.005	0.001	0.295	0.003	0	0.637	0.01	0.002	0.846	0.003	0.003
180	0.639	0.001	0.003	0.639	0.028	0.001	1	0.005	0.005	0.434	0.005	0	0.449	0.01	0	0.432	0.003	0.003
181	0.639	0.003	0.003	0.846	0.028	0.002	1	0.008	0.005	0.282	0.003	0	0.28	0.008	0.001	0.294	0.003	0.003
182	1	0.008	0.003	0.639	0.02	0	1	0.008	0	0.434	0.008	0	0.069	0.008	0	0.634	0.005	0.003
183	0.639	0.003	0.003	0.828	0.02	0.001	1	0.008	0.005	0.295	0.005	0	0.126	0.008	0.001	0.616	0.003	0.003
184	0.846	0.001	0.003	0.295	0.014	0.002	1	0.008	0.005	0.295	0.003	0	0.28	0.008	0.001	0.616	0.003	0.003
185	0.639	0.003	0.003	0.828	0.014	0.002	1	0.005	0.005	0.063	0.005	0	0.449	0.01	0.001	0.616	0.003	0.002
186	0.639	0.003	0.003	0.828	0.02	0.001	1	0.008	0.005	0.434	0.005	0	0.126	0.008	0.001	0.634	0.003	0.003
187	0.639	0.003	0.003	0.846	0.028	0.002	1	0.005	0.005	0.282	0.003	0	0.28	0.01	0.001	0.432	0.003	0.003
188	1	0.005	0.003	1	0.02	0.001	1	0.008	0.001	0.282	0.005	0	0.135	0.008	0	0.281	0.001	0.008
189	0.961	0.001	0.003	0.828	0.028	0.002	1	0.008	0.005	0.619	0.003	0	0.135	0.008	0	0.281	0.001	0.002
190	0.846	0.001	0.003	0.622	0.028	0.002	1	0.008	0.005	0.452	0.003	0	0.28	0.008	0	0.432	0.003	0.003
191	0.639	0.001	0.003	0.846	0.028	0.001	1	0.008	0.005	0.452	0.003	0	0.292	0.01	0.001	0.281	0.001	0.002
192	0.829	0.001	0.003	0.846	0.028	0.002	1	0.008	0.005	0.452	0.003	0	0.292	0.008	0	0.281	0.003	0.003
193	0.846	0.003	0.003	0.828	0.028	0.002	0.961	0.005	0.005	0.295	0.01	0	0.431	0.01	0	0.45	0.003	0.003
194	0.961	0.003	0.003	0.622	0.02	0.002	1	0.008	0.005	0.137	0.008	0	0.28	0.008	0	0.828	0.007	0.003
195	0.961	0.003	0.003	0.639	0.014	0.001	1	0.005	0.003	0.192	0.008	0	0.449	0.01	0.001	0.634	0.003	0.003
196	0.622	0.003	0.003	0.434	0.02	0	0.961	0.008	0	0.192	0.008	0	0.619	0.008	0.002	1	0.007	0.002
197	0.622	0	0.003	0.846	0.01	0	0.846	0.008	0	0.452	0.003	0	0.449	0.005	0.001	1	0.021	0.002
198	0.829	0.001	0.003	0.828	0.028	0.002	1	0.005	0.005	0.192	0.005	0	0.431	0.01	0.001	0.616	0.003	0.003
199	0.622	0.001	0.003	0.828	0.02	0.002	1	0.008	0.005	0.192	0.003	0	0.292	0.008	0.001	0.616	0.005	0.003
200	0.829	0.003	0.003	0.434	0.01	0.002	1	0.008	0.005	0.434	0.008	0	0.28	0.008	0.001	0.959	0.003	0.003
201	1	0.01	0.003	0.846	0.028	0	0.846	0.008	0	1	0.008	0	0.069	0.005	0	0.281	0.001	0.003
202	0.639	0.003	0.003	1	0.02	0.002	1	0.005	0.005	0.282	0.008	0	0.449	0.01	0	0.45	0.005	0.003
203	0.961	0.003	0.003	0.622	0.014	0.002	1	0.008	0.005	0.452	0.003	0	0.292	0.008	0.001	0.432	0.003	0.002
204	0.846	0.001	0.003	0.828	0.007	0.002	1	0.008	0.005	0.452	0.003	0	0.292	0.008	0.001	0.616	0.003	0.001

Table A.41: Dataset 1 at Stage 50% - p values splits- Part 4/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
205	0.846	0.003	0.003	0.452	0.02	0.001	1	0.008	0.003	0.192	0.008	0	0.28	0.008	0.001	1	0.003	0.003
206	0.622	0.003	0.003	0.96	0.028	0.002	1	0.008	0.005	0.282	0.003	0	0.292	0.01	0.001	0.432	0.003	0.003
207	0.846	0.003	0.003	0.434	0.014	0.002	1	0.008	0.005	0.295	0.003	0	0.292	0.008	0.001	0.846	0.003	0.003
208	0.639	0.003	0.003	0.828	0.028	0.001	1	0.008	0.005	0.295	0.003	0	0.431	0.01	0.001	0.616	0.003	0.003
209	0.961	0.003	0.003	0.828	0.01	0.004	1	0.008	0.005	0.434	0.003	0	0.189	0.008	0.001	0.45	0.005	0.002
210	1	0.003	0.003	0.295	0.02	0.002	1	0.008	0.005	0.636	0.008	0	0.189	0.008	0	0.634	0.003	0.003
211	0.622	0.003	0.003	0.622	0.028	0.002	1	0.005	0.005	0.295	0.01	0	0.449	0.01	0.001	0.828	0.003	0.003
212	0.622	0.001	0.003	1	0.02	0.002	1	0.005	0.005	0.282	0.005	0	0.449	0.01	0.001	0.45	0.003	0.003
213	1	0.008	0.003	0.622	0.047	0.002	0.846	0.003	0.005	0.827	0.01	0	0.292	0.01	0	0.294	0.003	0.003
214	0.622	0.003	0.003	0.846	0.02	0.001	1	0.005	0.005	0.295	0.005	0	0.449	0.01	0.001	0.828	0.003	0.003
215	0.829	0.001	0.003	0.639	0.028	0.002	1	0.005	0.005	0.452	0.003	0	0.449	0.01	0.001	0.432	0.003	0.002
216	0.451	0.001	0.003	0.828	0.028	0.002	1	0.005	0.005	0.295	0.003	0	0.637	0.01	0.002	0.959	0.003	0.001
217	0.829	0.003	0.003	0.639	0.02	0.002	1	0.005	0.005	0.295	0.003	0	0.619	0.01	0.001	0.828	0.003	0.002
218	0.639	0.003	0.003	0.622	0.028	0.002	1	0.008	0.005	0.295	0.003	0	0.449	0.01	0.001	0.828	0.003	0.003
219	0.639	0.003	0.003	0.639	0.014	0.002	1	0.008	0.005	0.282	0.003	0	0.431	0.008	0.002	0.846	0.003	0.002
220	0.451	0.003	0.003	0.639	0.014	0.002	1	0.008	0.005	0.137	0.003	0	0.619	0.008	0.002	1	0.007	0.002
221	1	0.001	0.003	0.622	0.005	0.004	1	0.005	0.019	0.434	0.008	0	0.189	0.005	0	0.45	0.005	0.008
222	1	0.001	0.003	1	0.028	0.004	1	0.003	0.005	0.452	0.008	0	0.431	0.01	0	0.281	0.003	0.003
223	0.451	0.001	0.003	0.96	0.01	0.002	1	0.008	0.005	0.282	0.003	0	0.637	0.008	0.002	0.846	0.014	0.001
224	0.622	0.003	0.003	0.846	0.028	0.002	1	0.005	0.005	0.282	0.003	0	0.449	0.01	0.002	0.828	0.003	0.003
225	0.622	0.003	0.003	1	0.028	0.002	1	0.005	0.005	0.282	0.005	0	0.449	0.01	0.001	0.828	0.003	0.003
226	1	0.003	0.003	0.639	0.02	0.002	1	0.005	0.005	0.619	0.003	0	0.449	0.01	0.001	0.294	0.003	0.002
227	0.846	0.003	0.003	0.434	0.01	0.001	1	0.008	0	0.636	0.008	0	0.431	0.008	0.001	1	0.003	0.003
228	0.622	0.003	0.003	0.622	0.028	0.001	1	0.005	0.001	0.636	0.008	0	0.619	0.01	0.001	1	0.003	0.003
229	0.639	0.003	0.003	0.828	0.02	0.001	1	0.005	0.001	0.192	0.008	0	0.828	0.01	0.002	1	0.003	0.003
230	1	0.003	0.003	0.639	0.01	0.002	1	0.008	0.005	0.619	0.003	0	0.292	0.008	0	0.846	0.003	0.003
231	0.451	0.003	0.003	0.846	0.028	0.002	1	0.005	0.005	0.192	0.003	0	0.637	0.01	0.002	0.959	0.003	0.003
232	0.451	0.001	0.003	1	0.02	0.002	1	0.005	0.005	0.137	0.005	0	0.619	0.01	0.002	0.634	0.003	0.003
233	0.961	0.003	0.003	0.622	0.01	0.002	1	0.008	0.005	0.636	0.003	0	0.449	0.008	0.001	0.959	0.003	0.001
234	0.961	0.003	0.003	0.622	0.01	0.001	1	0.008	0	1	0.008	0	0.431	0.005	0.001	1	0.003	0.002
235	0.639	0.003	0.003	1	0.028	0.008	1	0.005	0.005	0.452	0.003	0	0.431	0.01	0.001	0.294	0.003	0.002
236	0.639	0.003	0.003	0.639	0.028	0.002	1	0.008	0.005	0.295	0.005	0	0.619	0.01	0.001	1	0.003	0.003
237	0.639	0.003	0.003	0.452	0.014	0.002	1	0.008	0.005	0.452	0.003	0	0.292	0.008	0.002	1	0.003	0.003
238	1	0.005	0.003	0.639	0.01	0.002	1	0.008	0.005	0.845	0.008	0	0.292	0.008	0	0.828	0.003	0.003
239	0.639	0.005	0.004	1	0.02	0.002	1	0.005	0.005	0.192	0.01	0	0.637	0.01	0.002	0.828	0.003	0.002
240	1	0.005	0.003	0.639	0.02	0.002	1	0.005	0.005	0.636	0.008	0	0.449	0.008	0	0.634	0.003	0.003
241	0.961	0.003	0.003	0.639	0.02	0.002	1	0.005	0.005	0.636	0.003	0	0.449	0.01	0.001	0.634	0.003	0.003
242	0.622	0.014	0.004	1	0.028	0.008	1	0.001	0.005	0.619	0.005	0	0.828	0.01	0.001	0.281	0.001	0.001
243	0.451	0.001	0.003	1	0.02	0.002	1	0.014	0.008	0.128	0.005	0	0.449	0.01	0.002	0.959	0.005	0.003
244	0.846	0.003	0.003	0.96	0.047	0.001	0.846	0.005	0.005	0.619	0.008	0	0.619	0.01	0.001	0.959	0.003	0.003
245	0.846	0.005	0.003	0.639	0.02	0.001	1	0.005	0.001	0.845	0.01	0	0.619	0.01	0.001	0.846	0.003	0.003
246	0.434	0.003	0.003	1	0.028	0.008	1	0.005	0.005	0.452	0.003	0	0.619	0.01	0.002	0.432	0.003	0.002
247	0.434	0.003	0.001	1	0.01	0.004	1	0.048	0.012	0.434	0.003	0	0.619	0.014	0.004	0.281	0.003	0.002
248	0.846	0.005	0.003	1	0.02	0.004	1	0.005	0.005	0.619	0.003	0	0.449	0.01	0.001	0.616	0.003	0.002
249	0.846	0.003	0.003	0.622	0.014	0.002	1	0.008	0.005	0.434	0.008	0	0.431	0.01	0.001	1	0.003	0.003
250	0.846	0.005	0.001	0.828	0.02	0.002	1	0.008	0.008	0.827	0.003	0	0.431	0.01	0.001	0.846	0.003	0.002
251	1	0.01	0.003	1	0.064	0.002	0.846	0.003	0.003	1	0.01	0	0.449	0.01	0.001	0.616	0.003	0.003
252	1	0.003	0.003	0.434	0.028	0.001	1	0.008	0.005	0.96	0.008	0	0.189	0.008	0.001	1	0.007	0.003
253	1	0.008	0.003	0.622	0.263	0.002	1	0.008	0.005	1	0.008	0	0.292	0.008	0	0.294	0.003	0.005
254	0.961	0.005	0.004	0.639	0.02	0.001	1	0.014	0.008	0.192	0.008	0	0.292	0.01	0.002	1	0.014	0.003
255	1	0.01	0.056	0.639	0.315	0.002	1	0.008	0.005	1	0.014	0	0.28	0.003	0	1	0.003	0.003
256	0.296	1	0.819	0.282	0.991	1	0.451	0.426	1	0.137	0.426	1	1	0.475	1	0.137	0.934	1

Table A.42: Dataset 1 at Stage 50% - p values splits- Part 5/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
1	0.00002	0	0.00094	0.00326	0.00023	0.00785	0.00001	0	0	0.09429	0.02237	0.00775	0	0	0.01374	0.00724	0.00096	0
2	0.00006	0	0.00182	0.00343	0.00052	0.00536	0	0	0	0.04654	0.01222	0.01337	0.00005	0	0.00799	0.01762	0.00188	0
3	0.00029	0.00001	0.00094	0.00303	0.00018	0.00785	0.00002	0.00003	0.00035	0.02662	0.00443	0.00939	0.00007	0	0.00799	0.00949	0.00063	0
4	0.00029	0	0.00182	0.00422	0.00024	0.00785	0.00001	0.00001	0	0.03531	0.00611	0.00665	0.00004	0	0.01079	0.01493	0.00103	0
5	0.00029	0.00001	0.0031	0.00297	0.00023	0.00545	0.00001	0.00002	0	0.0324	0.00481	0.01188	0.00005	0	0.01074	0.01559	0.00115	0
6	0.00029	0.00001	0.00182	0.00265	0.00018	0.00545	0.00004	0.00004	0	0.01826	0.00317	0.01337	0.00007	0	0.00799	0.00928	0.00065	0
7	0.00002	0	0.00094	0.00581	0.00056	0.02577	0.00001	0	0	0.05374	0.01349	0.00939	0.00002	0	0.02088	0.00862	0.00078	0
8	0.00021	0.00001	0.0019	0.00401	0.00026	0.02568	0.00001	0.00002	0	0.0324	0.00486	0.01188	0.00003	0	0.01383	0.01016	0.00071	0
9	0.00032	0	0.0019	0.00384	0.00025	0.00785	0	0	0	0.04654	0.00749	0.00665	0.00006	0	0.01374	0.02574	0.00209	0
10	0.00029	0.00001	0.00094	0.00343	0.00018	0.00785	0.00001	0.00002	0	0.0443	0.00629	0.00539	0.00007	0	0.01079	0.0244	0.00139	0
11	0.00021	0	0.00094	0.00483	0.00034	0.00545	0.00001	0	0.00035	0.06332	0.01259	0.00939	0.00006	0	0.00799	0.0191	0.00191	0
12	0.00029	0.00001	0.0019	0.00431	0.00034	0.01061	0	0	0	0.04994	0.0076	0.00665	0.00004	0	0.01374	0.02037	0.00163	0
13	0.00035	0	0.0019	0.00454	0.00025	0.00794	0.00001	0	0	0.04861	0.00772	0.00665	0.00004	0	0.01374	0.02249	0.00209	0
14	0.00047	0	0.00094	0.0051	0.00034	0.00545	0	0	0	0.07094	0.01395	0.01188	0.00004	0	0.00805	0.02791	0.00256	0
15	0.00035	0.00001	0.0019	0.00265	0.00023	0.00785	0.00001	0.00001	0	0.03474	0.00619	0.00775	0.00007	0.00001	0.01383	0.0191	0.00153	0
16	0.00023	0.00001	0.0019	0.00269	0.00021	0.02577	0.00009	0.00003	0	0.02138	0.00365	0.00939	0.00002	0	0.02088	0.00553	0.00039	0.00024
17	0.00035	0.00001	0.00094	0.00269	0.00018	0.00785	0.00017	0.00012	0.00035	0.03186	0.0045	0.00662	0.00009	0	0.0056	0.00911	0.00063	0
18	0.00032	0	0.00094	0.0051	0.00036	0.00545	0	0	0	0.07024	0.01372	0.00665	0.00004	0	0.01074	0.03176	0.0027	0
19	0.00026	0.00001	0.00094	0.00231	0.00018	0.00545	0.00006	0.00003	0	0.03883	0.00535	0.00665	0.00007	0	0.01074	0.0244	0.00133	0
20	0.00039	0.00001	0.0019	0.00371	0.00024	0.00794	0.00001	0.00001	0	0.03745	0.0064	0.00665	0.00011	0.00001	0.01383	0.0172	0.00158	0
21	0.00029	0.00001	0.0019	0.00413	0.00026	0.01378	0.00001	0.00001	0	0.03686	0.00589	0.00775	0.00005	0	0.01706	0.01233	0.00127	0.00024
22	0.00034	0	0.0019	0.00504	0.00032	0.01751	0.00001	0	0	0.05301	0.00809	0.00665	0.00003	0	0.01383	0.0191	0.00163	0
23	0.00023	0	0.0019	0.00469	0.00031	0.05025	0.00001	0.00001	0	0.03825	0.00619	0.00665	0.00001	0	0.01383	0.01098	0.00065	0
24	0.00034	0.00001	0.0019	0.00401	0.00024	0.00794	0.00001	0.00001	0	0.06077	0.00884	0.00539	0.00004	0	0.01079	0.02934	0.0022	0
25	0.00019	0	0.00182	0.00699	0.00053	0.02568	0.00001	0	0	0.07486	0.01493	0.01188	0.00001	0	0.01374	0.02137	0.00144	0
26	0.00021	0.00001	0.00094	0.00384	0.00026	0.05709	0.00009	0.00003	0	0.02662	0.0045	0.00665	0.00004	0	0.02502	0.00535	0.00038	0
27	0.00031	0.00001	0.0019	0.00454	0.00034	0.01751	0.00002	0.00001	0	0.03948	0.0064	0.00665	0.00005	0	0.02073	0.0116	0.00136	0
28	0.00048	0.00001	0.0038	0.00562	0.00025	0.00785	0.00003	0.00003	0	0.03134	0.00508	0.00665	0.00007	0	0.01079	0.02577	0.00124	0
29	0.00039	0.00001	0.0019	0.00322	0.00023	0.00794	0.00002	0.00002	0	0.04654	0.00676	0.00539	0.00014	0.00001	0.01079	0.02249	0.00174	0
30	0.00037	0.00001	0.0019	0.00413	0.00023	0.00785	0.00001	0.00001	0	0.06077	0.00884	0.00539	0.00007	0	0.01374	0.04007	0.0026	0
31	0.00039	0.00001	0.00094	0.0026	0.00018	0.00794	0.00086	0.00095	0.00241	0.0228	0.00365	0.00775	0.00003	0	0.01383	0.00249	0.00017	0.00024
32	0.00048	0.00001	0.0019	0.00265	0.00018	0.00794	0.00006	0.00003	0	0.04792	0.00693	0.00539	0.00007	0	0.01383	0.0116	0.00116	0.00024
33	0.00016	0	0.00094	0.00974	0.00026	0.00545	0.00001	0	0.00035	0.06433	0.01222	0.00662	0.00011	0.00001	0.0056	0.03316	0.00422	0
34	0.00031	0	0.00094	0.00562	0.00035	0.00536	0.00001	0	0	0.09104	0.01688	0.00539	0.00004	0	0.01074	0.05173	0.00526	0
35	0.00029	0	0.0019	0.00401	0.00035	0.00785	0	0	0	0.07024	0.01493	0.00775	0.00008	0.00001	0.01383	0.03483	0.00422	0
36	0.00039	0.00001	0.0019	0.00289	0.00023	0.00785	0.00001	0.00002	0	0.05773	0.00836	0.00662	0.00007	0	0.01374	0.04672	0.0027	0
37	0.00047	0.00001	0.0019	0.00198	0.00015	0.00785	0.00002	0.00003	0	0.0443	0.00629	0.00539	0.00013	0.00001	0.01079	0.02489	0.00198	0
38	0.00016	0	0.00094	0.00652	0.00058	0.05709	0.00001	0	0	0.06166	0.01281	0.00665	0.00001	0	0.02502	0.00928	0.00078	0
39	0.00035	0	0.0019	0.00641	0.00059	0.01061	0	0	0	0.0862	0.01665	0.00665	0.00002	0	0.01079	0.04463	0.0037	0
40	0.00016	0	0.0019	0.00413	0.00032	0.00794	0.00001	0	0	0.13046	0.02576	0.00539	0.00004	0	0.01383	0.03483	0.00645	0
41	0.00035	0.00001	0.0019	0.00384	0.00025	0.02568	0.00007	0.00002	0	0.05842	0.00859	0.00539	0.00005	0	0.01699	0.0116	0.0013	0
42	0.0007	0.00001	0.0038	0.0043	0.0003	0.00794	0.00001	0.00001	0	0.05074	0.00703	0.00665	0.00008	0	0.01374	0.04666	0.00225	0
43	0.00029	0.00001	0.0019	0.00439	0.00028	0.0329	0.00001	0.00001	0	0.06755	0.01128	0.00374	0.00004	0	0.01079	0.02752	0.00217	0
44	0.00034	0	0.0019	0.0051	0.00036	0.00794	0.00001	0	0	0.07404	0.01522	0.00665	0.00008	0.00001	0.01383	0.03322	0.00383	0
45	0.00063	0.00001	0.0019	0.00513	0.00036	0.01751	0.00002	0.00001	0	0.04729	0.0076	0.00665	0.00005	0	0.01383	0.02705	0.00172	0
46	0.00074	0	0.0019	0.00537	0.00042	0.00785	0	0	0	0.08457	0.01665	0.00665	0.00004	0	0.01374	0.04463	0.00482	0
47	0.00019	0	0.0019	0.00641	0.00058	0.01378	0.00001	0	0	0.07988	0.01586	0.00775	0.00007	0	0.01706	0.02623	0.00318	0
48	0.00024	0	0.00094	0.00652	0.00058	0.05025	0.00001	0	0	0.07094	0.01416	0.00665	0.00001	0	0.01374	0.02249	0.00161	0
49	0.00051	0	0.0019	0.00537	0.00038	0.00794	0.00001	0	0	0.08098	0.01586	0.00665	0.00004	0	0.01374	0.04188	0.00472	0
50	0.00125	0.00002	0.0031	0.00389	0.00021	0.00785	0.00041	0.00012	0.00035	0.02315	0.00327	0.00939	0.00017	0.00001	0.00799	0.00974	0.00063	0
51	0.00078	0.00001	0.0038	0.00589	0.0005	0.01061	0.00001	0.00001	0	0.04134	0.0076	0.00665	0.00004	0	0.01374	0.04392	0.00174	0

Table A.43: Dataset 2 at Stage 20% - p values splits- Part 1/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
52	0.00099	0.00001	0.00389	0.00371	0.00024	0.00545	0.00014	0.00003	0	0.02576	0.00405	0.01188	0.00009	0	0.01074	0.03316	0.00128	0
53	0.00024	0	0.0019	0.00626	0.00058	0.01751	0.00001	0	0	0.07988	0.01493	0.00665	0.00004	0	0.02073	0.0244	0.00268	0
54	0.0008	0.00001	0.0019	0.00504	0.00035	0.01751	0.00014	0.00002	0	0.03323	0.00554	0.00665	0.00007	0.00001	0.02073	0.014	0.00111	0
55	0.00032	0	0.00094	0.00492	0.00028	0.00785	0.00001	0	0	0.09214	0.01716	0.00539	0.00014	0.00001	0.01079	0.04473	0.00614	0
56	0.00111	0.00001	0.0019	0.00439	0.00026	0.00794	0.00006	0.00003	0	0.03323	0.00523	0.00665	0.00026	0.00001	0.01383	0.02577	0.00163	0
57	0.00021	0	0.00094	0.01375	0.00032	0.00369	0.00009	0	0	0.10177	0.02146	0.00665	0.00003	0	0.00805	0.06137	0.00614	0
58	0.00134	0.00001	0.0038	0.00401	0.00034	0.00785	0.00001	0.00001	0	0.04384	0.00676	0.00665	0.00007	0.00001	0.01374	0.06503	0.00214	0
59	0.00058	0.00001	0.0031	0.00562	0.00035	0.05025	0.00012	0.00002	0	0.03371	0.00535	0.00665	0.00004	0	0.01383	0.01327	0.00061	0
60	0.00072	0.00001	0.0038	0.0051	0.00034	0.02568	0.00014	0.00003	0	0.02616	0.00456	0.01188	0.00004	0	0.01383	0.0165	0.00065	0
61	0.00035	0	0.0019	0.00616	0.00053	0.01751	0.00001	0	0	0.08884	0.018	0.00665	0.00004	0	0.01374	0.03959	0.00349	0
62	0.00048	0	0.0019	0.00439	0.00035	0.00785	0.00001	0	0	0.1032	0.02039	0.00539	0.00005	0	0.01079	0.0758	0.00686	0
63	0.00029	0	0.0019	0.00524	0.00047	0.00794	0.00001	0	0	0.13521	0.02742	0.00539	0.00004	0	0.01074	0.08358	0.00936	0
64	0.00018	0	0.0019	0.01468	0.00025	0.00785	0	0	0	0.11732	0.024	0.00539	0.00011	0.00001	0.01079	0.05478	0.00788	0
65	0.00037	0.00001	0.00094	0.01401	0.00119	0.00794	0.00001	0.00002	0	0.0408	0.00693	0.00665	0.00049	0.00002	0.01079	0.02791	0.00137	0
66	0.00031	0	0.0019	0.00482	0.0003	0.02568	0.00002	0	0	0.10688	0.02194	0.00539	0.00004	0	0.01699	0.02934	0.00463	0
67	0.00125	0.00002	0.0038	0.00265	0.00025	0.00785	0.0001	0.00003	0	0.03134	0.00473	0.00775	0.00026	0.00001	0.01383	0.02928	0.00174	0.00024
68	0.00066	0.00001	0.0019	0.00482	0.00024	0.00785	0.00001	0.00002	0	0.05514	0.00822	0.00539	0.00014	0.00001	0.01374	0.09942	0.00376	0
69	0.00035	0.00001	0.00094	0.01187	0.0013	0.00785	0.00001	0.00002	0	0.04134	0.00599	0.01188	0.00054	0.00002	0.01079	0.02884	0.00137	0
70	0.00482	0	0.0019	0.00681	0.00061	0.00545	0.00001	0.00001	0	0.06597	0.01349	0.00665	0.00007	0	0.01074	0.06613	0.00282	0
71	0.00051	0.00001	0.00094	0.01401	0.00148	0.00794	0.00009	0.00003	0	0.0324	0.00554	0.00665	0.00072	0.00004	0.00799	0.0172	0.00098	0
72	0.00031	0.00001	0.0019	0.01233	0.00125	0.01051	0.00001	0.00001	0	0.05074	0.0087	0.00665	0.00039	0.00002	0.01383	0.03322	0.00188	0
73	0.00046	0.00001	0.00182	0.01138	0.00077	0.00794	0.00001	0.00001	0	0.05074	0.0092	0.00665	0.00046	0.00002	0.01383	0.03767	0.00225	0
74	0.00111	0.00002	0.0019	0.00431	0.00026	0.05709	0.00063	0.00005	0	0.0228	0.00398	0.00665	0.00007	0	0.02502	0.00402	0.00034	0
75	0.00023	0.00001	0.0019	0.00322	0.00017	0.17018	0.00006	0.00003	0	0.03948	0.00599	0.00662	0.00001	0	0.01079	0.0112	0.00065	0
76	0.00058	0	0.0019	0.00504	0.00035	0.00785	0.00001	0	0	0.11864	0.024	0.00662	0.00007	0	0.01079	0.11223	0.0095	0
77	0.00029	0	0.0019	0.00513	0.00041	0.0329	0.00001	0	0	0.11362	0.02327	0.00374	0.00004	0	0.01074	0.06137	0.00735	0
78	0.00153	0.00001	0.0038	0.0043	0.00035	0.01378	0.00014	0.00003	0	0.03104	0.00486	0.00775	0.00013	0.00001	0.01706	0.01873	0.00137	0.00069
79	0.00035	0.00001	0.0019	0.0114	0.00103	0.00785	0.00001	0.00001	0	0.05154	0.00859	0.00665	0.00055	0.00002	0.01383	0.04392	0.00225	0
80	0.00035	0.00001	0.00094	0.01716	0.00156	0.00785	0.00012	0.00003	0	0.0249	0.00473	0.00939	0.00086	0.00005	0.00799	0.01868	0.00103	0
81	0.00056	0.00001	0.00094	0.01468	0.00124	0.00794	0.00001	0.00001	0	0.04654	0.00722	0.00665	0.00064	0.00003	0.01374	0.0244	0.00198	0
82	0.00074	0.00001	0.0019	0.00581	0.00024	0.00785	0.00021	0.00012	0.00035	0.03531	0.00523	0.00539	0.00014	0	0.01079	0.06883	0.00163	0
83	0.00072	0.00001	0.0019	0.00431	0.00025	0.0329	0.00006	0.00002	0	0.06077	0.00884	0.00374	0.00007	0	0.01079	0.0769	0.00286	0
84	0.0003	0.00001	0.00094	0.01533	0.0017	0.02149	0.00002	0.00002	0	0.0408	0.0064	0.00939	0.00041	0.00002	0.01383	0.02278	0.00103	0
85	0.00046	0.00001	0.0019	0.01256	0.00124	0.01751	0.00001	0.00001	0	0.05213	0.00904	0.00665	0.00027	0.00002	0.01383	0.03132	0.00188	0
86	0.00048	0.00001	0.00094	0.01187	0.00077	0.00785	0.00003	0.00002	0	0.05154	0.00798	0.00374	0.00066	0.00002	0.01079	0.04589	0.00191	0
87	0.00012	0	0.00094	0.00504	0.00034	0.17008	0.00001	0	0	0.11604	0.0248	0.00539	0	0	0.01074	0.04007	0.00376	0
88	0.00074	0	0.0019	0.01233	0.0013	0.05025	0.00009	0.00001	0	0.04994	0.0111	0.00665	0.00011	0.00001	0.02502	0.0085	0.0008	0
89	0.00113	0	0.0019	0.00919	0.00078	0.05025	0.00002	0.00001	0	0.06685	0.01328	0.00665	0.00003	0	0.01374	0.04122	0.00172	0
90	0.00056	0.00001	0.00094	0.01233	0.00132	0.00785	0.00002	0.00002	0	0.04516	0.00703	0.00775	0.00064	0.00004	0.01374	0.0275	0.00198	0
91	0.00805	0	0.0038	0.0168	0.00194	0.02568	0	0	0	0.06433	0.01222	0.01188	0.00002	0	0.01374	0.05248	0.00186	0
92	0.00048	0.00001	0.00094	0.01536	0.00152	0.01751	0.00002	0.00001	0	0.0443	0.00736	0.00665	0.00059	0.00002	0.01699	0.02278	0.00146	0
93	0.00091	0.00004	0.00389	0.00343	0.00024	0.02577	0.00211	0.00088	0.00125	0.01271	0.00234	0.00939	0.00005	0	0.02088	0.00466	0.00031	0.0008
94	0.00035	0.00001	0.00094	0.014	0.00136	0.0502	0.00003	0.00002	0	0.04007	0.00693	0.00665	0.00034	0.00002	0.01383	0.02132	0.00085	0
95	0.00102	0	0.0019	0.00732	0.00098	0.01751	0.00006	0	0	0.07486	0.01539	0.00665	0.00011	0.00001	0.02073	0.03068	0.00318	0
96	0.00041	0.00001	0.0019	0.01468	0.00144	0.01061	0.00003	0.00002	0	0.04994	0.00703	0.00775	0.00059	0.00002	0.01699	0.02178	0.00152	0
97	0.018	0	0.00389	0.00772	0.00156	0.00545	0	0	0	0.06524	0.01281	0.01188	0.00007	0	0.00805	0.09448	0.00306	0
98	0.00082	0.00001	0.0038	0.00389	0.00025	0.00794	0.00015	0.00003	0	0.05514	0.00884	0.00539	0.00007	0	0.01079	0.12394	0.00336	0
99	0.00041	0.00001	0.00094	0.01117	0.00077	0.00785	0.00006	0.00003	0	0.04516	0.00693	0.00662	0.00064	0.00005	0.01079	0.05546	0.00188	0
100	0.00269	0	0.0019	0.00836	0.00089	0.01751	0.00001	0	0	0.08098	0.01565	0.00665	0.00004	0	0.01374	0.07165	0.00436	0
101	0.00054	0.00001	0.00094	0.01043	0.00065	0.00785	0.00001	0.00001	0	0.06914	0.01091	0.00539	0.00064	0.00002	0.01383	0.06613	0.00296	0
102	0.00454	0	0.0019	0.00712	0.00063	0.00794	0.00001	0	0	0.08098	0.01539	0.00665	0.00009	0.00001	0.01374	0.09817	0.00482	0

Table A.44: Dataset 2 at Stage 20% - p values splits- Part 2/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
103	0.00035	0.00001	0.00094	0.01877	0.00216	0.02149	0.00017	0.00003	0	0.02576	0.00523	0.00939	0.00059	0.00004	0.01374	0.01433	0.00065	0
104	0.00078	0.00005	0.0008	0.01785	0.00136	0.00359	0.00062	0.00012	0	0.03745	0.00611	0.00291	0.00094	0.00007	0.00371	0.01873	0.00063	0
105	0.00048	0.00001	0.0019	0.01161	0.00114	0.00794	0.00001	0.00002	0	0.06838	0.0092	0.00374	0.00057	0.00002	0.01374	0.06883	0.00256	0
106	0.00072	0.00002	0.0008	0.0114	0.00089	0.00785	0.00004	0.00002	0	0.05592	0.00836	0.00539	0.00088	0.00005	0.01374	0.03641	0.00241	0
107	0.00048	0.00001	0.00094	0.01039	0.00089	0.00785	0.00001	0.00002	0	0.06166	0.00979	0.00539	0.00066	0.00002	0.01374	0.08948	0.00306	0
108	0.00599	0.00001	0.0019	0.00641	0.00059	0.00785	0.00001	0	0	0.07094	0.01395	0.00665	0.00046	0.00002	0.01383	0.04672	0.00455	0
109	0.00074	0.00001	0.00094	0.01536	0.00144	0.00794	0.00007	0.00003	0	0.03575	0.00619	0.00665	0.00064	0.00003	0.01079	0.06055	0.00152	0
110	0.00063	0.00003	0.0008	0.01289	0.00102	0.00794	0.00025	0.00003	0.00035	0.0443	0.00611	0.00662	0.0011	0.00007	0.00799	0.02319	0.00119	0
111	0.00066	0.00003	0.0008	0.01187	0.00119	0.00794	0.00009	0.00003	0	0.05301	0.00736	0.00539	0.00086	0.00004	0.01374	0.02985	0.00195	0
112	0.018	0.00001	0.00094	0.0114	0.00132	0.00545	0.00001	0.00001	0.00035	0.04586	0.01004	0.00939	0.00067	0.00004	0.00799	0.02282	0.00209	0
113	0.00037	0.00001	0.0008	0.0168	0.002	0.05025	0.00015	0.00003	0	0.03104	0.00583	0.00665	0.00055	0.00002	0.01699	0.01327	0.00063	0
114	0.0003	0.00001	0.0008	0.01842	0.00164	0.00794	0.00004	0.00002	0	0.06597	0.01091	0.00665	0.00058	0.00004	0.01079	0.05327	0.00286	0
115	0.00479	0	0.0038	0.01161	0.00139	0.01061	0.00001	0	0	0.08337	0.01616	0.00665	0.00004	0	0.01079	0.10298	0.00599	0
116	0.00942	0	0.0038	0.00699	0.00092	0.00785	0	0	0	0.08337	0.01565	0.00665	0.00007	0	0.01374	0.13653	0.00559	0
117	0.00066	0.00003	0.0008	0.01082	0.00083	0.00785	0.00006	0.00003	0	0.05301	0.00749	0.00539	0.0011	0.00008	0.01079	0.04392	0.0024	0
118	0.00029	0.00001	0.00017	0.0204	0.00205	0.00794	0.0001	0.00003	0	0.05688	0.0095	0.00665	0.00119	0.00011	0.00799	0.03641	0.00237	0
119	0.01115	0	0.00816	0.0475	0.00422	0.02577	0.00001	0	0	0.03531	0.01004	0.00939	0.00014	0	0.02511	0.01098	0.00091	0
120	0.00078	0.00001	0.0019	0.01253	0.00104	0.00794	0.00001	0.00002	0	0.05154	0.00786	0.00665	0.00064	0.00002	0.01383	0.08358	0.00233	0
121	0.00063	0.00002	0.0008	0.014	0.00134	0.01757	0.0001	0.00002	0	0.05773	0.00836	0.00539	0.00061	0.00002	0.01383	0.02577	0.00177	0
122	0.00221	0.00004	0.0019	0.00454	0.00028	0.02568	0.00067	0.00005	0	0.04307	0.00736	0.00539	0.00014	0	0.01699	0.02752	0.0013	0
123	0.00227	0.00005	0.0019	0.00469	0.00023	0.00794	0.00037	0.00011	0.00035	0.04233	0.00611	0.00539	0.00039	0.00001	0.01079	0.0651	0.00195	0
124	0.00016	0	0.00094	0.02188	0.00231	0.00785	0.00006	0.00003	0	0.05842	0.01128	0.01188	0.00064	0.00006	0.01079	0.05253	0.00282	0
125	0.00011	0.00001	0.00017	0.02142	0.00341	0.00785	0.00012	0.00003	0	0.0408	0.00935	0.00939	0.00122	0.00017	0.00799	0.03641	0.00217	0
126	0.00125	0.00001	0.0038	0.00389	0.00023	0.00785	0.00036	0.00003	0	0.05213	0.0076	0.00662	0.00014	0	0.01374	0.14359	0.0041	0
127	0.00048	0.00001	0.0019	0.01161	0.00104	0.0329	0.00003	0.00001	0	0.07205	0.01091	0.00374	0.00052	0.00002	0.01374	0.05546	0.00243	0
128	0.00087	0.00001	0.00094	0.0168	0.00181	0.00785	0.00014	0.00003	0	0.03186	0.00508	0.01188	0.00066	0.00004	0.01079	0.07165	0.00152	0
129	0.00013	0	0.0008	0.02339	0.00284	0.02149	0.0001	0.00003	0	0.06077	0.01171	0.00939	0.00053	0.00004	0.01374	0.04392	0.00191	0
130	0.00482	0	0.0038	0.01651	0.00205	0.01378	0.00001	0	0	0.07205	0.01462	0.00775	0.00017	0.00001	0.01706	0.05478	0.00436	0
131	0.00037	0.00001	0.00094	0.01536	0.00152	0.00794	0.00002	0.00002	0	0.07789	0.01328	0.00665	0.00061	0.00005	0.01383	0.06883	0.00495	0
132	0.00087	0.00001	0.0019	0.01401	0.00148	0.00785	0.00002	0.00002	0	0.04384	0.00772	0.00665	0.0007	0.00003	0.01383	0.11223	0.0022	0
133	0.00074	0.00001	0.0019	0.0168	0.0021	0.01051	0.00002	0.00002	0	0.04516	0.00809	0.00665	0.00061	0.00002	0.01383	0.0769	0.00225	0
134	0.0008	0.00001	0.0019	0.01533	0.00148	0.01751	0.00004	0.00002	0	0.05074	0.00809	0.00665	0.00059	0.00002	0.01383	0.06613	0.0022	0
135	0.00026	0	0.00094	0.02002	0.00186	0.00785	0.00002	0.00002	0	0.07205	0.01281	0.00665	0.00061	0.00006	0.01383	0.0727	0.00463	0
136	0.00016	0	0.0019	0.02293	0.00274	0.01051	0.00004	0.00003	0	0.07404	0.01281	0.00665	0.00059	0.00004	0.01374	0.06613	0.00422	0
137	0.00035	0.00001	0.00094	0.01877	0.00227	0.01751	0.00004	0.00002	0	0.08098	0.01328	0.00665	0.00052	0.00002	0.01374	0.06055	0.00422	0
138	0.00019	0	0.0008	0.01842	0.00227	0.00785	0.00007	0.00003	0	0.06433	0.01171	0.00775	0.0013	0.00017	0.01374	0.05022	0.00422	0
139	0.00035	0.00001	0.0008	0.01785	0.0021	0.00794	0.00006	0.00002	0	0.07024	0.01259	0.00665	0.00118	0.00011	0.01374	0.05248	0.0041	0
140	0.00035	0.00001	0.00182	0.00326	0.00017	0.00545	0.00067	0.06198	0.00125	0.01794	0.00256	0.07011	0.00011	0.00001	0.00371	0.01016	0.00068	0
141	0	0.00005	0.00094	0.02986	0.00459	0.00369	0	0.00067	0.00035	0.06524	0.01349	0.07011	0.00004	0	0.00371	0.03483	0.00526	0
142	0.00158	0.00004	0.00094	0.0197	0.00156	0.01051	0.00045	0.00004	0.00035	0.03104	0.0045	0.00665	0.00122	0.00009	0.00799	0.02388	0.00115	0
143	0.00139	0.00003	0.0019	0.01651	0.0016	0.00794	0.0001	0.00003	0	0.04173	0.00676	0.00665	0.00118	0.00009	0.01374	0.03955	0.00217	0
144	0.00074	0.00001	0.00094	0.01939	0.00227	0.0502	0.00017	0.00003	0	0.03418	0.00599	0.00665	0.00052	0.00002	0.01383	0.03322	0.00103	0
145	0.00086	0.00001	0.0019	0.02105	0.00227	0.02149	0.00017	0.00003	0	0.03104	0.00486	0.00939	0.00058	0.00003	0.01383	0.04672	0.00124	0
146	0.00072	0.00005	0.0008	0.01536	0.00114	0.00785	0.00048	0.00013	0	0.03287	0.00481	0.01853	0.0013	0.00011	0.00544	0.02388	0.00128	0
147	0.00542	0.00001	0.00094	0.02188	0.00063	0.00369	0.00001	0.00001	0.00035	0.08884	0.01539	0.00539	0.00026	0.00001	0.01074	0.1686	0.0095	0
148	0.00037	0.00001	0.0008	0.01877	0.0016	0.00545	0.00006	0.00003	0	0.07566	0.01307	0.00374	0.00098	0.00008	0.01074	0.08958	0.00614	0
149	0.00039	0.00001	0.00094	0.01233	0.00102	0.0779	0.0001	0.00003	0	0.04516	0.00693	0.00662	0.00052	0.00003	0.01079	0.03698	0.00137	0
150	0.00013	0.00001	0.00017	0.02842	0.004	0.02149	0.00021	0.00003	0	0.04516	0.00904	0.00939	0.00088	0.00014	0.01374	0.02928	0.00156	0
151	0.00139	0.00003	0.0019	0.01975	0.00216	0.01751	0.00017	0.00002	0	0.04134	0.00693	0.00665	0.00075	0.00003	0.01699	0.03316	0.00156	0
152	0.00116	0.00003	0.0019	0.01716	0.00186	0.00794	0.00014	0.00003	0	0.04134	0.00583	0.00775	0.00118	0.00007	0.01374	0.04871	0.00214	0
153	0.0393	0	0.0038	0.00854	0.00144	0.00785	0	0	0	0.07566	0.01436	0.00775	0.00046	0.00002	0.01383	0.06503	0.00503	0

Table A.45: Dataset 2 at Stage 20% - p values splits- Part 3/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
154	0.00143	0.00005	0.00597	0.00326	0.00025	0.00545	0.00258	0.0466	0.00125	0.01188	0.00218	0.01337	0.00019	0.00001	0.00799	0.01249	0.00065	0.00024
155	0.00521	0.00001	0.0019	0.00815	0.00052	0.00545	0.00001	0.00001	0	0.09666	0.01908	0.00539	0.00017	0.00001	0.01079	0.24718	0.01484	0
156	0.0003	0.00001	0.0008	0.0204	0.00247	0.01751	0.00012	0.00002	0	0.07205	0.01222	0.00665	0.001	0.00009	0.01699	0.04194	0.00359	0
157	0.00023	0.00001	0.0008	0.02149	0.00233	0.0502	0.00009	0.00002	0	0.06685	0.0115	0.00665	0.00046	0.00004	0.01374	0.05022	0.0024	0
158	0.00019	0.00001	0.0008	0.02293	0.00295	0.01061	0.00012	0.00003	0	0.06914	0.0115	0.00775	0.00107	0.00017	0.01699	0.04871	0.00365	0
159	0.00013	0.00001	0.00017	0.02516	0.00233	0.00545	0.00014	0.00003	0	0.07682	0.01416	0.00662	0.00075	0.00012	0.01074	0.1018	0.00624	0
160	0.00102	0.00003	0.0019	0.02002	0.00233	0.01378	0.00017	0.00003	0	0.04233	0.00574	0.00775	0.00094	0.00005	0.01699	0.04871	0.00163	0.00024
161	0.00026	0.00001	0.00182	0.00422	0.00023	0.4975	0.00006	0.00002	0	0.05457	0.00749	0.00374	0.00003	0	0.01374	0.01331	0.00063	0
162	0.00048	0.00001	0.0008	0.01536	0.00139	0.00785	0.00006	0.00002	0	0.10575	0.01908	0.00539	0.00085	0.00006	0.01374	0.12416	0.00788	0
163	0.00116	0.00002	0.0019	0.0133	0.00095	0.00785	0.00004	0.00002	0	0.06077	0.0095	0.00539	0.00107	0.00004	0.01383	0.1686	0.00422	0
164	0.00181	0.00001	0.0019	0.00699	0.00056	0.0329	0.00003	0.00001	0	0.10688	0.02107	0.00374	0.00006	0	0.01074	0.28797	0.01647	0
165	0.00099	0.00003	0.0019	0.01289	0.0013	0.00794	0.00009	0.00003	0	0.05842	0.00859	0.00374	0.00083	0.00002	0.01374	0.18637	0.00401	0
166	0.00023	0.00001	0.0008	0.02332	0.00247	0.05025	0.00021	0.00003	0	0.04861	0.00884	0.00665	0.00083	0.00009	0.01699	0.03274	0.00161	0
167	0.0012	0.00003	0.00094	0.01533	0.00104	0.00794	0.0004	0.00003	0	0.04729	0.0064	0.00374	0.001	0.00007	0.01079	0.13012	0.00241	0
168	0.00143	0.00005	0.00094	0.02188	0.00233	0.00794	0.0014	0.00017	0	0.0208	0.00335	0.00939	0.00119	0.00011	0.00799	0.02928	0.00124	0
169	0.00268	0.00005	0.0038	0.00291	0.00023	0.00794	0.00168	0.0007	0.00241	0.04134	0.00589	0.00539	0.00017	0.00001	0.01383	0.0589	0.00191	0.0008
170	0.00026	0.00001	0.00017	0.02332	0.00181	0.00359	0.00067	0.00012	0	0.07205	0.01259	0.00291	0.00198	0.00032	0.00371	0.06982	0.00286	0
171	0.00099	0.00001	0.0019	0.00681	0.0005	0.02568	0.00014	0.00001	0	0.10688	0.02146	0.00539	0.00036	0.00001	0.01699	0.11373	0.01743	0
172	0.00037	0.00001	0.00017	0.0204	0.0021	0.00794	0.00019	0.00003	0.00035	0.06685	0.00971	0.00662	0.00205	0.00025	0.00799	0.0589	0.00463	0
173	0.00125	0.00005	0.0008	0.01939	0.00227	0.05709	0.00067	0.00003	0	0.02938	0.0045	0.00665	0.00078	0.00005	0.01699	0.01997	0.00064	0
174	0.00063	0.00001	0.0008	0.02332	0.00231	0.00794	0.00016	0.00003	0	0.05993	0.01068	0.00665	0.00122	0.00012	0.01079	0.11223	0.00383	0
175	0.00032	0.00001	0.0008	0.01569	0.00156	0.0329	0.0001	0.00002	0	0.1032	0.01837	0.00374	0.00057	0.00004	0.01079	0.10847	0.00703	0
176	0.00039	0.00001	0.0008	0.0168	0.0016	0.00785	0.0001	0.00003	0	0.1032	0.01539	0.00539	0.00169	0.00017	0.01374	0.08358	0.00671	0
177	0.0012	0.00005	0.00094	0.02432	0.00238	0.02568	0.00168	0.00012	0	0.02247	0.0031	0.00939	0.00088	0.00005	0.01374	0.02438	0.00065	0.00024
178	0.00125	0.00002	0.0019	0.01256	0.00112	0.0329	0.0001	0.00002	0	0.05993	0.00904	0.00374	0.00078	0.00002	0.01374	0.14359	0.00401	0
179	0.00029	0.00001	0.0008	0.02142	0.00174	0.00785	0.00009	0.00003	0	0.09918	0.01638	0.00539	0.00098	0.00016	0.01079	0.17059	0.01056	0
180	0.00134	0.00003	0.0019	0.01375	0.00102	0.00794	0.00013	0.00003	0	0.06433	0.0076	0.00539	0.00105	0.00006	0.01374	0.22015	0.00472	0
181	0.00023	0.00001	0.0008	0.02105	0.00227	0.00794	0.0001	0.00003	0	0.11732	0.01908	0.00374	0.00083	0.00012	0.01079	0.16116	0.00905	0
182	0.00078	0.00001	0.0019	0.02149	0.00181	0.00794	0.00009	0.00003	0	0.07205	0.01349	0.00665	0.00098	0.00012	0.01383	0.15194	0.00516	0
183	0.00269	0.00005	0.0008	0.01425	0.00102	0.00794	0.00041	0.00003	0	0.05514	0.0076	0.00539	0.00153	0.00016	0.01374	0.08358	0.00286	0
184	0.00048	0.00001	0.00094	0.03249	0.00354	0.00785	0.00014	0.00003	0	0.05213	0.00884	0.01188	0.00114	0.00017	0.01079	0.15742	0.00376	0
185	0.0071	0.00001	0.0019	0.02227	0.00058	0.00785	0.00001	0.00001	0.00035	0.08766	0.01638	0.00539	0.00067	0.00002	0.01079	0.17276	0.01429	0
186	0.00021	0.00001	0.0008	0.02332	0.00194	0.00785	0.00009	0.00003	0	0.09214	0.01395	0.00539	0.0032	0.00033	0.01079	0.09817	0.00775	0
187	0.00479	0	0.0038	0.00694	0.00058	0.00794	0.00019	0	0	0.13976	0.02776	0.00539	0.00005	0	0.01074	0.4352	0.02538	0
188	0.00035	0.00001	0.0008	0.01683	0.00156	0.01757	0.00019	0.00003	0	0.10422	0.01665	0.00539	0.0013	0.00013	0.01383	0.08106	0.00645	0
189	0.00066	0.00001	0.0019	0.02227	0.00238	0.01751	0.00015	0.00002	0	0.07306	0.01222	0.00665	0.00078	0.00011	0.01374	0.13492	0.00482	0
190	0.00023	0.00001	0.0008	0.01846	0.00205	0.00794	0.00015	0.00003	0	0.10941	0.01871	0.00539	0.00205	0.00025	0.01374	0.08958	0.00788	0
191	0.00024	0	0.00094	0.00524	0.00035	0.4777	0.00002	0	0	0.10847	0.02146	0.00374	0.00001	0	0.01079	0.03767	0.00318	0
192	0.00074	0.00001	0.0008	0.02293	0.00216	0.00794	0.00025	0.00003	0	0.07094	0.00979	0.00665	0.00205	0.00027	0.01374	0.08948	0.00503	0
193	0.00066	0.00001	0.0019	0.02785	0.00344	0.00785	0.00009	0.00003	0	0.07094	0.01128	0.00665	0.00122	0.00016	0.01383	0.19284	0.00582	0
194	0.00019	0	0.0008	0.00439	0.00035	0.69932	0.00003	0	0	0.08232	0.01586	0.00374	0.00001	0	0.02511	0.0085	0.00064	0
195	0.00048	0.00001	0.0019	0.03249	0.005	0.01051	0.00014	0.00003	0	0.06838	0.01068	0.00665	0.00098	0.00016	0.01374	0.18637	0.00545	0
196	0.00031	0	0.0008	0.04411	0.0056	0.02149	0.00021	0.00003	0	0.05688	0.00904	0.00939	0.00086	0.00016	0.01374	0.14145	0.00237	0
197	0.00261	0.00005	0.00182	0.01683	0.00144	0.02137	0.00052	0.00003	0	0.05074	0.00809	0.00539	0.00136	0.00009	0.01383	0.07896	0.00241	0
198	0.00555	0.00014	0.00094	0.00483	0.00023	0.00785	0.01229	0.10055	0.00125	0.02178	0.00275	0.00662	0.00039	0.00001	0.0056	0.02109	0.00051	0
199	0.00048	0.00001	0.0008	0.02785	0.00278	0.0502	0.00037	0.00003	0	0.05842	0.00971	0.00665	0.00086	0.00012	0.01374	0.10682	0.00243	0
200	0.00125	0.00001	0.00017	0.03242	0.00322	0.01051	0.00048	0.00004	0.00035	0.0443	0.00693	0.00665	0.0043	0.00052	0.00799	0.06055	0.0026	0
201	0.0007	0.00001	0.00182	0.02432	0.00284	0.01751	0.00044	0.00003	0	0.06524	0.00904	0.00665	0.00189	0.00017	0.01699	0.07676	0.00422	0
202	0.00013	0.00001	0.0008	0.02227	0.00233	0.07782	0.00015	0.00003	0	0.09429	0.018	0.00662	0.00064	0.0001	0.01074	0.09584	0.00449	0
203	0.0003	0.00001	0.00094	0.00303	0.00018	0.72291	0.00041	0.00012	0	0.03883	0.00554	0.00374	0.00004	0	0.02955	0.00268	0.00018	0
204	0.00153	0.00004	0.00389	0.00354	0.00016	0.00545	0.01755	0.05345	0.00035	0.0284	0.00342	0.00665	0.00014	0	0.01074	0.09817	0.00172	0

Table A.46: Dataset 2 at Stage 20% - p values splits- Part 4/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
205	0.00164	0.00005	0.00182	0.01651	0.00119	0.00794	0.00148	0.00015	0	0.03825	0.00629	0.00662	0.00119	0.00012	0.01079	0.17276	0.0026	0
206	0.00755	0.00001	0.00389	0.02188	0.00069	0.00785	0.00034	0	0	0.11604	0.02294	0.00662	0.00014	0.00001	0.01079	0.36058	0.01871	0
207	0.00542	0.00001	0.0019	0.00699	0.00056	0.00794	0.00002	0	0	0.14326	0.02866	0.00539	0.00033	0.00001	0.01383	0.58573	0.04465	0
208	0.00116	0.00002	0.0038	0.00401	0.00023	0.17018	0.00374	0.00076	0.00241	0.03104	0.00421	0.00662	0.00004	0	0.01079	0.06872	0.00085	0
209	0.00289	0.00006	0.0038	0.00297	0.00017	0.00785	0.0017	0.04728	0.00035	0.03745	0.00473	0.00539	0.00041	0.00001	0.01079	0.10823	0.00241	0.00024
210	0.00019	0.00001	0.0008	0.04327	0.00247	0.00545	0.00036	0.00012	0	0.05842	0.0111	0.01853	0.00384	0.00054	0.00544	0.0727	0.00545	0
211	0.00072	0.00001	0.0008	0.03763	0.00369	0.00794	0.00016	0.00003	0	0.05993	0.0115	0.00775	0.00341	0.00048	0.01374	0.11389	0.00545	0
212	0.00208	0.00006	0.00182	0.01842	0.00148	0.01051	0.00081	0.00012	0	0.04792	0.00703	0.00539	0.00174	0.00016	0.01374	0.11968	0.0027	0.00024
213	0.00047	0.00001	0.00182	0.03484	0.00483	0.01378	0.00036	0.00003	0	0.05907	0.01053	0.00775	0.00265	0.00032	0.01699	0.10847	0.00526	0
214	0.00289	0.00007	0.00182	0.01651	0.00124	0.00794	0.00074	0.00015	0	0.04516	0.00676	0.00539	0.00203	0.00017	0.01079	0.13861	0.00359	0
215	0.00074	0.00001	0.0008	0.02842	0.00227	0.00545	0.0004	0.00004	0	0.07094	0.01053	0.00374	0.00205	0.00026	0.01074	0.27912	0.0095	0
216	0.00521	0.00024	0.00017	0.01846	0.00144	0.01051	0.00387	0.00041	0.00035	0.03474	0.00563	0.00662	0.00209	0.00016	0.00799	0.06055	0.00111	0
217	0.00102	0.00001	0.0008	0.0197	0.00186	0.00785	0.00014	0.00003	0	0.10177	0.01493	0.00539	0.00186	0.00021	0.01374	0.31095	0.01318	0
218	0.00072	0.00001	0.0019	0.00504	0.0003	0.4975	0.00057	0.00008	0	0.0443	0.00629	0.00374	0.00004	0	0.01374	0.03641	0.00063	0
219	0.00512	0	0.0038	0.00772	0.00058	0.17008	0.0003	0.00001	0	0.11604	0.02194	0.00539	0.00001	0	0.01074	0.28764	0.01429	0
220	0.00087	0.00001	0.00017	0.03078	0.00362	0.05709	0.00098	0.00004	0	0.04586	0.00749	0.00665	0.00209	0.00017	0.01699	0.06137	0.00164	0
221	0.00113	0.00005	0.00094	0.01683	0.00152	0.08608	0.00106	0.00012	0	0.04516	0.00629	0.00662	0.00086	0.00004	0.01079	0.16116	0.00186	0
222	0.00047	0.00001	0.00017	0.04411	0.00523	0.02568	0.00206	0.00023	0	0.04173	0.00809	0.00939	0.00308	0.00038	0.01374	0.07394	0.00156	0
223	0.00039	0.00001	0.00094	0.0133	0.00112	0.4975	0.00009	0.00002	0	0.05213	0.00772	0.00374	0.00046	0.00002	0.01374	0.03641	0.00124	0
224	0.00426	0.00034	0.00017	0.02142	0.00174	0.00536	0.0319	0.00289	0.00125	0.03104	0.00351	0.00291	0.00234	0.00017	0.00371	0.06247	0.00063	0.00024
225	0.00048	0.00001	0.00182	0.02561	0.00257	0.00794	0.00024	0.00003	0	0.1032	0.02004	0.00374	0.00138	0.00026	0.01079	0.4605	0.02257	0
226	0.0008	0	0.0019	0.00641	0.0005	0.47755	0.00009	0.00001	0	0.10177	0.01757	0.00298	0.00004	0	0.01079	0.14359	0.00624	0
227	0.00072	0.00001	0.00182	0.02188	0.00216	0.0329	0.00025	0.00003	0	0.10688	0.01493	0.00374	0.00136	0.00017	0.01079	0.35379	0.01522	0
228	0.0007	0.00001	0.00182	0.032	0.00295	0.00794	0.00048	0.00003	0	0.09322	0.01665	0.00539	0.00189	0.00032	0.01079	0.3833	0.01674	0
229	0.0006	0.00003	0.0008	0.01496	0.00124	0.55736	0.00048	0.00003	0	0.04384	0.0064	0.00291	0.00057	0.00003	0.01079	0.01762	0.00061	0
230	0.00289	0.00003	0.0008	0.01039	0.00071	0.68779	0.00021	0.00001	0.00035	0.06524	0.01004	0.00374	0.00059	0.00002	0.02511	0.00119	0.00036	0
231	0.00143	0.00003	0.0008	0.02944	0.00181	0.00794	0.00041	0.00004	0	0.08232	0.01416	0.00539	0.00638	0.00059	0.01374	0.24456	0.01171	0
232	0.00426	0.00001	0.0008	0.07285	0.01138	0.00794	0.00167	0.00026	0	0.03825	0.00703	0.00939	0.00587	0.00078	0.00799	0.0758	0.0027	0
233	0.00116	0.00002	0.0008	0.02293	0.00194	0.02137	0.00067	0.00003	0	0.08766	0.01539	0.00539	0.00449	0.00033	0.01383	0.29976	0.01113	0
234	0.00029	0.00001	0.0008	0.01785	0.00156	0.4777	0.00015	0.00003	0	0.07566	0.01307	0.00374	0.00057	0.00004	0.01079	0.08106	0.00436	0
235	0.00074	0.00001	0.0008	0.07285	0.00388	0.00785	0.00757	0.00013	0	0.07306	0.01091	0.00662	0.00259	0.00046	0.01074	0.33528	0.01287	0
236	0.00116	0.00003	0.00094	0.01536	0.00125	0.4975	0.00051	0.00003	0	0.0443	0.00736	0.00374	0.00064	0.00002	0.01374	0.11373	0.00153	0
237	0.00464	0.00008	0.0019	0.00371	0.00018	0.72291	0.01357	0.00234	0.01045	0.02212	0.00306	0.00374	0.00009	0	0.02955	0.00206	0.00009	0
238	0.00047	0.00001	0.00017	0.03484	0.00369	0.0779	0.00134	0.00013	0	0.09666	0.01462	0.00662	0.00221	0.00025	0.01074	0.55276	0.0107	0
239	0.03217	0.00006	0.00606	0.09952	0.01777	0.00359	0.00015	0.00154	0.00035	0.10575	0.01969	0.00665	0.00007	0	0.00805	0.26834	0.01379	0
240	0.00035	0.00001	0.00017	0.0197	0.00181	0.55736	0.00045	0.00003	0	0.07024	0.01193	0.00291	0.00144	0.00021	0.01079	0.06055	0.00296	0
241	0.00227	0.00005	0.00017	0.05855	0.00344	0.00794	0.00081	0.00015	0	0.08337	0.01259	0.00539	0.01484	0.00251	0.01079	0.33528	0.01463	0
242	0.00164	0.00004	0.00182	0.03713	0.00362	0.01051	0.00119	0.00013	0	0.10941	0.01616	0.00539	0.00927	0.00097	0.01374	0.49137	0.01825	0
243	0.00805	0.00006	0.00017	0.0691	0.00344	0.00785	0.00291	0.0004	0.00035	0.04654	0.00859	0.00662	0.01902	0.00267	0.00799	0.17629	0.00717	0
244	0.00521	0.00044	0.0008	0.04059	0.00186	0.00794	0.04342	0.09097	0.00125	0.02247	0.00265	0.01853	0.0027	0.00022	0.00544	0.08461	0.00128	0
245	0.00482	0.00021	0.0008	0.02002	0.0016	0.57804	0.00631	0.0003	0	0.03575	0.00563	0.00291	0.00153	0.00011	0.01079	0.03547	0.00051	0
246	0.04365	0.00118	0.00094	0.23342	0.02042	0.00369	0.00006	0.00327	0.00035	0.05907	0.00971	0.00662	0.00563	0.00068	0.0056	0.08948	0.00735	0
247	0.00072	0.00001	0.0008	0.02332	0.00194	0.4777	0.00073	0.00004	0	0.07306	0.01307	0.00374	0.00169	0.00017	0.01079	0.3479	0.01035	0
248	0.17064	0.00005	0.00606	1	0.09824	0.00536	0	0.00067	0.00035	0.03531	0.00859	0.01337	0.00067	0.00002	0.00799	0.02577	0.0022	0
249	0.06797	0.00057	0.00597	0.08305	0.01327	0.00545	0.00019	0.00124	0.00035	0.12148	0.02294	0.00539	0.00072	0.00011	0.01079	0.38695	0.01825	0
250	0.00379	0.00006	0.00017	0.03242	0.00284	0.57804	0.00622	0.00035	0	0.06597	0.00904	0.00291	0.01157	0.00099	0.01079	0.27655	0.00516	0
251	0.00965	0.01228	0.00829	0.05224	0.00705	0.00785	0.79578	1	0.03733	0.00858	0.00111	0.07011	0.00054	0.00002	0.00371	0.04392	0.00063	0.00024
252	0.0251	0.00088	0.00017	0.08305	0.00893	0.00545	0.04977	0.00389	0	0.21452	0.01688	0.00291	0.18651	0.02332	0.00371	0.63279	0.00845	0
253	0.16491	0.00623	0.0008	0.67846	0.09951	0.01061	0.09028	0.09552	0.00035	0.02704	0.00398	0.01853	0.1522	0.03219	0.00544	0.32606	0.01056	0
254	0.0119	0.00089	0.00389	0.00292	0.00018	0.84461	0.47636	0.85367	0.7939	0.01188	0.00178	0.77214	0.00007	0	0.74551	0.00645	0.00005	0.00165
255	1	1	0.01561	1	1	0.00545	1	1	0.01045	0.00432	0.00035	0.07011	0.02562	0.00366	0.00371	0.30781	0.01364	0
256	1	0.86311	0.19099	1	0.2228	1	1	0.93691	0.66389	1	0.35278	1	1	0.02358	1	0.9952	1	0.12798

Table A.47: Dataset 2 at Stage 20% - p values splits- Part 4/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
1	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.008	0	0	0.0002
2	0	0	0.0002	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0.0022	0	0	0.002	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
10	0	0	0	0	0	0	0	0	0.0002	0	0	0.0012	0	0	0.001	0	0	0.0002
11	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
16	0	0	0.0002	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
17	0	0	0	0	0	0.0003	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
19	0	0	0.0002	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0007
20	0	0	0.0002	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
23	0	0	0.0002	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0.0008
24	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0.0002
35	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
36	0	0	0.0002	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
37	0	0	0	0	0	0	0	0	0.0002	0	0	0.0011	0	0	0.0003	0	0	0
38	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.0003	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
41	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
42	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
43	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
44	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
45	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
46	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
47	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
48	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0007
49	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
51	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0

Table A.48: Dataset 2 at Stage 50% - p values splits- Part 1/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
52	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0002
53	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
54	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
55	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
56	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
57	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
58	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
59	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
60	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
61	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
62	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
63	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
64	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
65	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
66	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.0003	0	0	0
67	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
68	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
69	0	0	0	0	0	0	0	0	0.0002	0	0	0.0005	0	0	0.001	0	0	0
70	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
71	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
72	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
73	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
74	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
75	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
76	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
77	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
78	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
79	0	0	0	0	0	0	0	0	0	0	0	0.0022	0	0	0.001	0	0	0
80	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
81	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
82	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0.0002
83	0	0	0.0002	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0007
84	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
85	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
86	0	0	0	0	0	0	0	0	0	0	0	0.0022	0	0	0.001	0	0	0
87	0	0	0	0	0	0	0	0	0	0	0	0.0023	0	0	0.0022	0	0	0.0002
88	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
89	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
90	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.002	0	0	0
91	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
92	0	0	0	0	0	0	0	0	0	0	0	0.0023	0	0	0.002	0	0	0
93	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
94	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
95	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
96	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
97	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
98	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
99	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
100	0	0	0	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
101	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
102	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0

Table A.49: Dataset 2 at Stage 50% - p values splits- Part 2/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
103	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
104	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
105	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
106	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
107	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
108	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0.0002
109	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
110	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
111	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
112	0	0	0	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
113	0	0	0	0	0	0	0	0	0	0	0	0.0022	0	0	0.001	0	0	0
114	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
115	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
116	0	0	0	0	0	0.0003	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
117	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
118	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0002
119	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
120	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
121	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
122	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
123	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0002
124	0	0	0	0	0	0	0	0	0	0	0	0.0022	0	0	0.001	0	0	0
125	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
126	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
127	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
128	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
129	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
130	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
131	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0002
132	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
133	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
134	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.0003	0	0	0
135	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
136	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
137	0	0	0	0	0	0	0	0	0	0	0	0.0023	0	0	0.001	0	0	0
138	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0.0002
139	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
140	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
141	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
142	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0.0002
143	0	0	0.0002	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0.0007
144	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
145	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
146	0	0	0	0	0	0	0	0	0	0	0	0.0036	0	0	0.0036	0	0	0
147	0	0	0	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
148	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
149	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
150	0	0	0	0	0	0.0005	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0.0007
151	0	0	0	0	0	0.0003	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
152	0	0	0.0002	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
153	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0

Table A.50: Dataset 2 at Stage 50% - p values splits- Part 3/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
154	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
155	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
156	0	0	0	0	0	0.0003	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
157	0	0	0.0002	0	0	0.0003	0	0	0.0002	0	0	0.0005	0	0	0.001	0	0	0.0002
158	0	0	0.0002	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0008
159	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
160	0	0	0	0	0	0.0005	0	0	0	0	0	0.0011	0	0	0.001	0	0	0.0007
161	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
162	0	0	0	0	0	0	0	0	0.0007	0	0	0.0022	0	0	0.002	0	0	0
163	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
164	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
165	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
166	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
167	0	0	0	0	0	0.0003	0	0	0	0	0	0.0022	0	0	0.0003	0	0	0
168	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
169	0	0	0	0	0	0.0003	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
170	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
171	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
172	0	0	0	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
173	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
174	0	0	0	0	0	0.0003	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
175	0	0	0	0	0	0.0011	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
176	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0007
177	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
178	0	0	0	0	0	0.0003	0	0	0	0	0	0.0012	0	0	0.002	0	0	0
179	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
180	0	0	0	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
181	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
182	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
183	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
184	0	0	0	0	0	0.0003	0	0	0	0	0	0.0023	0	0	0.001	0	0	0
185	0	0	0	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
186	0	0	0	0	0	0.0003	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
187	0	0	0	0	0	0.0005	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
188	0	0	0	0	0	0	0	0	0	0	0	0.0023	0	0	0.002	0	0	0
189	0	0	0	0	0	0.0003	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
190	0	0	0	0	0	0	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
191	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
192	0	0	0	0	0	0	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
193	0	0	0	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
194	0	0	0.0002	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
195	0	0	0	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0002
196	0	0	0	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.001	0	0	0.0007
197	0	0	0	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
198	0	0	0	0	0	0.0005	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
199	0	0	0	0	0	0.0005	0	0	0	0	0	0.0012	0	0	0.0003	0	0	0.0007
200	0	0	0	0	0	0.0003	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
201	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
202	0	0	0	0	0	0.0005	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
203	0	0	0	0	0	0.0011	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
204	0	0	0	0	0	0.0003	0	0	0	0	0	0.0011	0	0	0.001	0	0	0

Table A.51: Dataset 2 at Stage 50% - p values splits- Part 4/5

ID	SP1 p F	SP1 p G	SP1 p W	SP2 p F	SP2 p G	SP2 p W	SP3 p F	SP3 p G	SP3 p W	SP4 p F	SP4 p G	SP4 p W	SP5 p F	SP5 p G	SP5 p W	test p F	test p G	test p W
205	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
206	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
207	0	0	0	0	0	0.0011	0	0	0	0	0	0.0005	0	0	0.001	0	0	0.0002
208	0	0	0	0	0	0.0003	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
209	0	0	0	0	0	0.0003	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
210	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
211	0	0	0	0	0	0.0011	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0002
212	0	0	0	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
213	0	0	0	0	0	0.0003	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
214	0	0	0	0	0	0.0011	0	0	0	0	0	0.0005	0	0	0.001	0	0	0.0002
215	0	0	0	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0002
216	0	0	0	0	0	0.0003	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
217	0	0	0	0	0	0.0011	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0.0008
218	0	0	0	0	0	0.0003	0	0	0	0	0	0.0023	0	0	0.001	0	0	0
219	0	0	0	0	0	0.0003	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
220	0	0	0	0	0	0.0005	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
221	0	0	0	0	0	0	0	0	0.0002	0	0	0.0022	0	0	0.0036	0	0	0
222	0	0	0	0	0	0.0005	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
223	0	0	0	0	0	0.0005	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
224	0	0	0	0	0	0.0003	0	0	0	0	0	0.0022	0	0	0.001	0	0	0.0007
225	0	0	0	0	0	0.0003	0	0	0	0	0	0.0023	0	0	0.0036	0	0	0
226	0	0	0.0009	0	0	0.0003	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
227	0	0	0	0	0	0.0011	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0.0002
228	0	0	0	0	0	0.0003	0	0	0.0002	0	0	0.0012	0	0	0.0003	0	0	0.0002
229	0	0	0.0002	0	0	0	0	0	0	0	0	0.0011	0	0	0.002	0	0	0
230	0	0	0	0	0	0.0011	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
231	0	0	0	0	0	0.0005	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
232	0	0	0	0	0	0.0005	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
233	0	0	0	0	0	0	0	0	0.0002	0	0	0.0005	0	0	0.0003	0	0	0
234	0	0	0	0	0	0.0005	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
235	0	0	0	0	0	0.0005	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
236	0	0	0	0	0	0.0005	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0.0002
237	0	0	0	0	0	0.0005	0	0	0	0	0	0.0012	0	0	0.002	0	0	0
238	0	0	0	0	0	0.0003	0	0	0	0	0	0.0022	0	0	0.001	0	0	0
239	0	0	0	0	0	0.0011	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
240	0	0	0	0	0	0.0005	0	0	0	0	0	0.0023	0	0	0.001	0	0	0.0002
241	0	0	0	0	0	0.0005	0	0	0	0	0	0.0012	0	0	0.001	0	0	0
242	0	0	0	0	0	0.0005	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
243	0	0	0	0	0	0.0011	0	0	0	0	0	0.0022	0	0	0.0003	0	0	0.0002
244	0	0	0	0	0	0.0005	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
245	0	0	0	0	0	0	0	0	0.0002	0	0	0.0012	0	0	0.001	0	0	0
246	0	0	0	0	0	0.0011	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0
247	0	0	0	0	0	0.0003	0	0	0	0	0	0.9242	0	0	0.0003	0	0	0.0007
248	0	0	0	0	0	0	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
249	0	0	0.0002	0.4229	0.6218	0.6638	0	0	0	0	0	0.0011	0	0	0.001	0	0	0
250	0	0	0.0002	0.4807	0.606	0.951	0	0	0	0	0	0.0005	0	0	0.0003	0	0	0
251	0	0	0	0.477	0.6807	0.8701	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0.0008
252	0	0	0	0.4586	0.6935	0.747	0	0	0	0	0	0.0011	0	0	0.001	0	0	0.0007
253	0	0	0	0.4441	0.6508	0.8201	0	0	0	0	0	0.0011	0	0	0.0003	0	0	0.0002
254	0	0	0	0.5537	0.7243	1	0	0	0	0	0	0.9242	0	0	0.0002	0	0	0.0017
255	0	0	1	0.4117	0.5777	1	0	0	0	0	0	0.0005	0	0	0.001	0	0	0
256	1	0.8631	0.191	1	0.8639	0.1942	1	0.9365	0.6639	1	0.3528	1	1	0.0236	1	0.9952	1	0.128

Table A.52: Dataset 2 at Stage 50% - p values splits- Part 5/5

Curriculum Vitae

Name: MohammadNoor A. M. Injadat

Post-Secondary Education and Degrees:

- **2015-2020 Ph.D**: Electrical and Computer Engineering- Software Engineering, Faculty of Engineering, University of Western Ontario, London, ON, Canada.
- **2014-2015 MEng.**: Electrical and Computer Engineering-Software Engineering- Software Engineering, Faculty of Engineering, University of Western Ontario, London, ON, Canada.
- **2013 Edexcel- BTEC Certified Trainer**: Edexcel UK, UK.
- **2011 Certificate 4 in Training and Assessment (TAFE)**: Challenger TAFE WA, Australia .
- **2000-2002 MSc.**: Computer Science, Faculty of Computer Science and Information Technology, University Putra Malaysia, Serdang, Selangore, Malaysia.
- **1995-2000 BSc**: Computer Science, Faculty of Arts and Science, Al al-Bayt University, Mafraq, Jordan.

Honours and Awards :

- **1993 Crown Prince Awards- International Youth Awward (IYA)**: Bronze Award, Crown Prince Award Office, Amman, Jordan.
- **1995 Crown Prince Awards (IYA)**: Silver Award, Crown Prince Award Office, Amman, Jordan.
- **1998 Crown Prince Awards (IYA)**: Golden Award, Crown Prince Award Office, Amman, Jordan.
- **2007 Appreciation Certificate, New Academic programs Development**: Emirates College of Technology, Abu Dhabi, UAE.
- **2011 Appreciation Certificate, Applied Technology Youth Camp**: Institute of Applied Technology (IAT), Abu Dhabi, UAE.

- **2013 Certificate of Contribution, Cyber Security Conference**: Abu Dhabi Polytechnic, organizing member, Abu Dhabi, UAE .
- **2013 Certificate of Excellence, Emirates Skills Competition**: Judge (Software Solution for Business Category), ACTVET, Abu Dhabi, UAE.
- **2015 R.K. Swartman Master of Engineering Program Award**: Faculty of Engineering, University of Western Ontario, London, Ontario, Canada.
- **2017-2018 Ontario Graduate Scholarship (OGS)**: Faculty of Engineering, University of Western Ontario, London, Ontario, Canada.
- **2018-2019 Ontario Graduate Scholarship (OGS)**: Faculty of Engineering, University of Western Ontario, London, Ontario, Canada.

Related Work Experience:

- **2020 LDA- Teaching SE3310B**: Faculty of Engineering, University of Western Ontario, London, ON, Canada.
- **2015-2019 Teaching Assistant**: Faculty of Engineering, University of Western Ontario, London, ON, Canada.
- **2008-2014 IT and Networking Curriculum Specialist**: Abu Dhabi Vocational Education and Training Institutes (ADVETI), Abu Dhabi, UAE.
- **2010-2013 Part Time Lecturer**: Al Ain University of Science and Technology, Abu Dhabi, UAE.
- **2005-2008 University Instructor and College Registrar**: Emirates College of Technology, Abu Dhabi, UAE.
- **2004-2005 Part Time Lecturer**:Al-Balqa'a Applied University, Ajlun University College, Ajlun, Jordan.
- **2002-2004 Lecturer**: Tiba University, Teacher College in Al-Medina Al-Monawwara, Medina, Saudi Arabia.
- **2002 Part Time Lecturer**: Jordan University of Science and Technology, Irbid, Jordan.
- **2002 Lecturer**:Al al-Bayt University, Mafraq, Jordan.

Publications:

- **Journals Publications:**

  [1] **M. Injadat**, F. Salo, and A. B. Nassif "Data mining techniques in social media: A survey," in *Neurocomputing 214 (2016): 654-670*, 2016.

[2] A. Moubayed, **M. Injadat**, A. B. Nassif, H. Lutfiyya, and A.shami, "E-learning: Challenges and research opportunities using machine learning& Data Analytics," in *IEEE Access 6 (2018): 39117-39138*, 2018.

[3] F. Salo, **M. Injadat**, A. B. Nassif, A.shami, and A. Essex, "Data Mining Techniques in Intrusion Detection Systems: A Systematic Literature Review," in *IEEE Access 6 (2018): 56046-56058*, 2018.

[4] A. Moubayed, **M. Injadat**, A.shami, and H. Lutfiyya, "Student Engagement Level in e-learning Environment: Clustering Using K-means," in *American Journal of Distance Education 2 (2020): 0892-3647*, 2020.

[5] **M. Injadat**, A. Moubayed, A. B. Nassif, and A. Shami, "Systematic Ensemble Model Selection Approach for Educational Data Mining," in *Elsevier: Knowledge-Based Systems*, vol. 200, page 105992, 2020, DOI: 10.1016/j.knosys.2020.105992.

[6] **M. Injadat**, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-split Optimized Bagging Ensemble Model Selection for Multi-class Educational Datasets," *Springer: Applied Intelligence*, 2020, DOI: 10.1007/s10489-020-01776-3.

[7] **M. Injadat**, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-Stage Optimized Machine Learning Framework for Network Intrusion Detection," *IEEE Transactions On Network and Service Management*, 2020, DOI:10.1109/TNSM.2020.3014929.

[8] **M. Injadat**, A. Moubayed, A. B. Nassif, and A. Shami, "Machine Learning Towards Intelligent Systems: Applications, Challenges, and Opportunities," *Submitted to Artificial Intelligence Review*, 2020.

- **Conferences Publications:**

[1] A. Moubayed, **M. Injadat**, A.shami, and H. Lutfiyya, "Relationship Between Student Engagement and Performance in E-Learning Environment Using Association Rules," in *2018 IEEE World Engineering Education Conference (EDUNINE), IEEE*, 2018, pp. 1-6.

[2] **M. Injadat**, F. Salo, A. B. Nassif, A. Essex, and A. Shami,"Bayesian optimization with machine learning algorithms towards anomaly detection," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.

[3] A. Moubayed, **M. Injadat**, A.shami, and H. Lutfiyya, " DNS Typo-Squatting Domain Detection: A Data Analytics & Machine Learning Based Approach," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1-7.

[4] F. Salo, **M. Injadat**, Abdallah Moubayed, A. B. Nassif, and A. Essex, "Clustering Enabled Classification using Ensemble Feature Selection for Intrusion Detection," in *2019 International Conference on Computing, Networking and Communications (ICNC), IEEE*, 2019, pp. 276-281.

[5] F. Salo, **M. Injadat**, A. B. Nassif, and A. Essex, "Data Mining with Big Data in Intrusion Detection Systems: A Systematic Literature Review," in *International Symposium on Big Data Management and Analytics 2019 (BIDMA), Calgary, Canada*, 2019.