Recursive Feature Addition: a Novel Feature Selection Technique, Including a Proof of Concept in Network Security

by

Tarfa Hamed

A Thesis
presented to
The University of Guelph

In partial fulfilment of requirements
for the degree of
Doctor of Philosophy
in
Computer Science

Guelph, Ontario, Canada
©Tarfa Hamed, March, 2017
ABSTRACT

Recursive Feature Addition: a Novel Feature Selection Technique,
Including a Proof of Concept in Network Security

Tarfa Hamed
University of Guelph, 2017

Advisor:
Dr. Stefan C. Kremer
Coadvisor:
Dr. Rozita Dara

Protecting the information on the Internet is a significant objective and this significance is increasing with time. The Internet experiences many attacks every day and that is attracting all participant parties to find a solution. One of the most common solutions to provide network security is called a Network Intrusion Detection System (NIDS). Usually a NIDS utilizes a classifier to classify the data extracted from the incoming traffic to either a normal or an attack connection. The feature selection phase is an early phase that needs to be carefully achieved prior to classification; otherwise the entire performance of the NIDS will tremendously deteriorate. However, two challenges face the NIDS that deals with new attacks is the availability of only a small number of examples with many features. The first challenge that NIDS experiences is the overfitting. Feature selection methods in turn face the second challenge of finding interdependent features.

This thesis focuses on feature selection to address the above challenges that face NIDS. The contributions of this thesis involve proposing and implementing a new feature selection method which is called Recursive Feature Addition (RFA). The RFA depends on Support Vector Machines classifier and works in a forward recursive fashion in selecting the features.
The RFA method has been tested on the synthetic data set and proved its ability to detect interdependent features, and tested on real-world high-dimensional data sets and proved its superiority over RFE in performance. Applying RFA on the ISCX 2012 data set is another contribution since RFA outperformed RFE in detecting intrusions. A new metric is also proposed to be used in evaluating NIDS applications that combines three well-known metrics. Furthermore, four ranking coefficients have been proposed to be used with RFA beside the original one, and have been tested with RFA on the ISCX data set. The statistical test confirmed the RFA’s superiority over RFE using different performance metrics and on different data sets. Moreover, the work involved implementing the multi-class intrusion detection, where the NIDS identifies the specific attack type instead of raising an alarm only in case of an intrusion.
Acknowledgements

In the name of God, the Most Beneficent, the Most Merciful

At the end of this exciting and at times frustrating process, having finished this thesis, I would like to take the opportunity to express my gratitude to all that have supported me through this long journey.

I would like to express my deepest gratitude and respect to my research advisor, Dr. Stefan C. Kremer and my co-advisor Dr. Rozita Dara. I have always benefited from their expertise supervision, brilliant ideas, continuous enthusiasm, rigorous attitude to science, and valuable advice and extensive knowledge.

I also thank my advisory committee members Dr. Fangju Wang and Dr. Radu Muresan for their valuable comments during this work.

My family, to whom I dedicate this dissertation, my beloved wife Rasha and my precious children Aya, Abdulrahman and Malak, thank you for all you’ve sacrificed during this doctoral journey. Thank you for giving me the time, place, and quiet I needed to study, read, and write. I could not have accomplished this without your support, unconditional encouragement and patience. My parents who instilled in me a passion for learning and a desire to do my personal best, my most sincere and deepest gratitude for your never ending, unconditional love, encouragement, guidance, wisdom and support. My brothers Qutayba, Saif, Wail and Huthayfa for all their prayers for me and continuous encouragement.

I am also grateful to my colleagues Jamileh and Masood, who enriched me with their comments discussions.
# Contents

1 Introduction ............................... 1

1.1 Internet’s Threats ........................ 2

1.2 Threats targets ........................... 3

1.3 Intrusion Detection ....................... 4

1.4 Intrusion Detection Systems ............ 4

1.5 Machine Learning ........................ 7

1.6 Challenges to IDS ....................... 7

1.7 Feature Selection ........................ 8

1.7.1 Connection between Feature Selection and Intrusion Detection .... 11

1.7.2 Feature types and Feature interdependency ........... 11

1.8 Motivation .............................. 12

1.9 Organization of the thesis ............... 12

1.10 Thesis statement ........................ 13

1.11 Summary of the work .................... 13

1.12 Goals of the work ....................... 15
### 2 Literature Review

2.1 Feature Selection .................................................. 16

2.2 Feature selection types .......................................... 22

2.2.1 Filter Methods .................................................. 23

2.2.2 Wrapper Methods .............................................. 26

2.2.3 Embedded Methods ............................................ 28

2.2.4 Hybrid methods .............................................. 29

2.3 Utilizing feature selection in intrusion detection ............ 29

2.4 Major Key-Steps of Feature Selection .......................... 31

2.5 Data Classification .............................................. 35

2.5.1 Using SVM as a Classifier .................................... 35

2.6 Classifier Performance Evaluation ............................. 38

2.7 Summary of the chapter .......................................... 39

### 3 Analysis of Recursive Feature Elimination RFE and a Proposal for a RFA Alternative

3.1 Objectives .......................................................... 41

3.2 Analysis and Limitation of RFE .................................. 41

3.2.1 Description of the data set used in the analysis ............ 42

3.2.2 Experimental design ........................................... 45

3.2.3 Preliminary Results of RFE analysis .......................... 46

3.3 Deep analysis on RFE ............................................. 48

3.4 Thesis Statement ................................................... 53

3.5 Methodology ....................................................... 54
## 3.5.1 Recursive Feature Addition (RFA) ................................. 56

## 3.6 The proposed feature selection method and approach ................. 57

### 3.6.1 Recursive Feature Addition (RFA) with SVMs .................. 57

### 3.6.2 Training the Classifier During the Feature Selection ........ 60

## 3.7 Methods ................................................................. 61

### 3.7.1 Selection of Data sets ............................................ 61

### 3.7.2 Choosing a Classifier ............................................ 63

### 3.7.3 Data set Splitting ................................................ 64

### 3.7.4 Performance Evaluation ......................................... 64

## 3.8 Methodology Description .............................................. 65

### 3.8.1 Measurements .................................................... 65

### 3.8.2 Methodology Steps ................................................ 66

## 3.9 Conclusion and Summary .............................................. 68

### 4 Experimental work and Results ..................................... 70

#### 4.1 Results of deep analysis on RFA ................................. 70

#### 4.2 Comparison of RFE and RFA in applying feature selection on the synthetic Majority problem data sets ................................. 77

#### 4.3 Results of applying feature selection on the small, real-world benchmark data sets ................................................. 80

#### 4.4 Results of applying feature selection on the large real-world benchmark data sets ................................................ 86

#### 4.5 Statistical analysis .................................................. 89

#### 4.6 Summary of the chapter ............................................. 92
5 Intrusion Detection Application

5.1 ISCX data set ................................................................. 95
5.2 Feature extraction and data set preparation ......................... 96
5.3 Feature Selection on ISCX data set ................................... 102
5.4 Feature selection evaluation metrics for IDS ....................... 103
5.5 RFA application on ISCX data set ..................................... 107
5.6 RFE application on ISCX data set .................................... 114
5.7 Distribution of the performance of RFA and RFE .................. 124
5.8 Statistical Analysis .......................................................... 130
5.9 New ranking coefficients for RFA ....................................... 134
5.10 Multi-class intrusion detection problem .............................. 171
  5.10.1 Binary tree structure for multi-class classification ........... 172
  5.10.2 Data set preparation ................................................... 173
  5.10.3 Results of applying feature selection on the balanced data sets ... 174
  5.10.4 Results of applying feature selection on the imbalanced data sets . 175
  5.10.5 Calculating the final confusion matrix for multi-class intrusion detection 177
  5.10.6 Discussing the results of the multi-class intrusion detection .... 184
5.11 Summary of the chapter .................................................... 185

6 Conclusions and Future Work .............................................. 188

6.1 Thesis statement ........................................................... 192
6.2 Limitations and future work ............................................. 193
6.3 Summary ................................................................. 193
6.4 Work implications ....................................................... 194
List of Tables

2.1 A Confusion Matrix for Binary classification ........................................ 38

3.1 Synthetic Data Set Design ................................................................. 43

3.2 Description of the Synthetic Problem Data Sets .................................. 44

3.3 Performance Metrics before and after Applying RFE on the Synthetic Majority Data Sets ................................................................. 47

3.4 Description of the small-size data sets .............................................. 61

3.5 Description of the real-world data sets ............................................. 63

4.1 Performance metrics before and after applying both RFA and RFE on the Synthetic Majority problem data sets ............................................. 78

4.2 The maximum classification accuracy obtained from RFA, RFE, two wrapper methods and two filter methods on small real-world benchmark data sets ........................................... 81

4.3 The total time taken for applying feature selection for RFA compared to RFE, two wrapper methods and two filter methods on small real-world benchmark data sets (Time in seconds) ........................................... 82

4.4 The total time taken to reach the maximum classification accuracy for both RFA and RFE methods (in seconds) ........................................... 83

4.5 Performance metrics before and after applying both RFA and RFE on the real-world benchmark data sets ........................................... 87
4.6 Statistical tests and the resulting cases .................................................. 90

4.7 Mann-whitney U test for RFA and RFE results on the synthetic majority data
sets using both metrics Accuracy and F-measure ........................................ 91

4.8 Mann-whitney U test for RFA and RFE results on the small data sets using
Accuracy ........................................................................................................ 91

4.9 Mann-whitney U test for RFA and RFE results on the real-world data sets
using both metrics Accuracy and F-measure ................................................. 92

5.1 Description of ISCX Data Set ................................................................. 95

5.2 Bigram representation for the three payload features in the example ........ 100

5.3 Performance metrics for three different scenarios with equal accuracy ...... 105

5.4 Performance metrics for three different scenarios with equal detection rate . 105

5.5 Performance metrics for three different scenarios with equal false alarm rate 106

5.6 Performance metrics without and with bigram features after applying RFA
on the ISCX data sets ..................................................................................... 107

5.7 Performance Metrics before and after Applying RFE on the ISCX Data Sets 115

5.8 All the performance metrics after applying both RFA and RFE on the ISCX
data sets ......................................................................................................... 122

5.9 Mann-whitney U test for RFA and RFE results on the ISCX data sets using
both metrics Accuracy and F-measure ......................................................... 133

5.10 Mann-whitney U test for RFA and RFE results on the ISCX data sets using
Detection rate, False Alarm Rate and overall metric .................................... 133

5.11 Maximum obtained accuracy by each of the proposed ranking coefficient and
the original one for all the ISCX data sets ................................................... 136

5.12 Maximum obtained F-measure by each of the proposed ranking coefficients
and the original one for all the ISCX data sets ............................................. 136
5.13 Maximum obtained detection rate by each of the proposed ranking coefficients and the original one for all the ISCX data sets
5.14 Minimum obtained false alarm rate by each of the proposed ranking coefficients and the original one for all the ISCX data sets
5.15 Maximum obtained overall metric by each of the proposed and original ranking coefficients for all the ISCX data sets
5.16 Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 25 data set on detection rate
5.17 Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 50 data set on detection rate
5.18 Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 100 data set on detection rate
5.19 Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 500 data set on detection rate
5.20 Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 25 data set on false alarm rate
5.21 Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 50 data set on false alarm rate
5.22 Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 100 data set on false alarm rate
5.23 Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 500 data set on false alarm rate
5.24 Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 25 data set on overall metric
5.25 Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 50 data set on overall metric
5.26 Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 100 data set on overall metric .................................................. 168

5.27 Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 500 data set on overall metric .................................................. 170

5.28 Data sets’ distribution for the balanced approach ........................................ 173

5.29 Data sets’ distribution for the imbalanced approach ........................................ 173

5.30 Results of applying RFA on the balanced data sets ........................................ 174

5.31 Results of applying RFE on the balanced data sets ........................................ 175

5.32 Results of applying RFA on the imbalanced data sets ........................................ 176

5.33 Results of applying RFE on the imbalanced data sets ........................................ 176

5.34 The confusion matrix for normal vs. attack sub-tree using RFA on the balanced ISCX data sets .................................................. 177

5.35 Percentages of normal attack calculated from the corresponding confusion matrix .................................................. 178

5.36 The confusion matrix for Non-DoS vs. DoS sub-tree using RFA on the balanced ISCX data sets .................................................. 178

5.37 Percentages of Non DoS vs DoS attacks calculated from the corresponding confusion matrix .................................................. 179

5.38 The confusion matrix for internal vs. brute-force attacks sub-tree using RFA on the balanced ISCX data sets .................................................. 179

5.39 Percentages of Internal vs Brute-force attacks calculated from the corresponding confusion matrix .................................................. 180

5.40 The confusion matrix for HTTP DoS vs. DDoS attacks sub-tree using RFA on the balanced ISCX data sets .................................................. 180
5.41 Percentages of HTTP DoS vs DDoS attacks calculated from the corresponding confusion matrix ................................................... 181

5.42 The final confusion matrix after applying RFA on the balanced ISCX data sets ....................................................... 182

5.43 The final confusion matrix after applying RFE on the balanced ISCX data sets ...................................................... 183

5.44 The final confusion matrix after applying RFA on the imbalanced ISCX data sets ......................................................... 183

5.45 The final confusion matrix after applying RFE on the imbalanced ISCX data sets ....................................................... 184
# List of Figures

2.1 AUC in three cases of classifications: Good, Random and Prefect . . . . . . 19

2.2 The wrapper subset selection approach. The induction algorithm is used as a black box by the subset selection algorithm (Kohavi & John, 1997) . . . . 27

2.3 Four key-steps of feature selection . . . . . . . . . . . . . . . . . . . . . . . . 31

2.4 Subset generation in different feature selection methods (a) Filter (b) Wrapper (c) Embedded (Saeys, Inza, & Larrañaga, 2007) . . . . . . . . . . . . . 33

2.5 Decision boundary and margin of SVM classifier . . . . . . . . . . . . . . . 36

3.1 Performance of RFE feature selection against random selection for Majority1000 data set (a) The number of good features eliminated (b) The number of bad features remaining (c) The percentage of good features remaining . . 50

3.2 Performance of RFE feature selection against random selection for Majority800 data set (a) The number of good features eliminated (b) The number of bad features remaining (c) The percentage of good features remaining . . 51

3.3 Performance of RFE feature selection against random selection for Majority600 data set (a) The number of good features eliminated (b) The number of bad features remaining (c) The percentage of good features remaining . . 52

3.4 Performance of RFE feature selection against random selection for Majority300 data set (a) The number of good features eliminated (b) The number of bad features remaining (c) The percentage of good features remaining . . 53
3.5 Overall Model based on Embedded Feature Selection

3.6 Embedded forward-feature selection in a four features with the order of selection in bold (2,4,3,1), where 1 means presence of a feature and 0 means absence

3.7 A block diagram for training and testing steps of the proposed model

4.1 Performance of RFA feature selection against random selection for the Majority300 data set (a) The number of good features selected (b) The number of bad features selected (c) The percentage of good features selected

4.2 Performance of RFA feature selection against random selection for the Majority600 data set (a) The number of good features selected (b) The number of bad features selected (c) The percentage of good features selected

4.3 Performance of RFA feature selection against random selection for the Majority800 data set (a) The number of good features selected (b) The number of bad features selected (c) The percentage of good features selected

4.4 Performance of RFA feature selection against random selection for the Majority1000 data set (a) The number of good features selected (b) The number of bad features selected (c) The percentage of good features selected

4.5 Δ%Accuracy vs. Δ%F-measure for both RFA and RFE on Majority problem data sets (a) Majority300 (b) Majority600 (c) Majority800 (d) Majority1000

4.6 The classifier accuracy after adding/removing one feature at a time for RFA and RFE for Diabetes data set

4.8 The classifier accuracy after adding/removing one feature at a time for RFA and RFE for Ionosphere data set

4.7 The classifier accuracy after adding/removing one feature at a time for RFA and RFE for Spect data set
4.9 The classifier accuracy after adding/removing one feature at a time for RFA and RFE for QSAR data set .............................. 85

4.10 The classifier accuracy after adding/removing one feature at a time for RFA and RFE for Hill-valley data set .............................. 86

4.11 $\Delta$%Accuracy vs. $\Delta$%F-measure for both RFA and RFE on real-world data sets (a) Breast (b) CNS (c) Colon (d) DB_emails (e) DLBCL (f) DLBCL_small (g) Leukemia (h) Lung Michigan (i) Lung Ontario (j) Lymph (k) Pomeroy . 88

5.1 Feature extraction process for ISCX data set using bigram technique ....... 96

5.2 Dictionary construction stage during feature extraction for ISCX data set using bigram technique ........................................... 98

5.3 Feature vector extraction for ISCX data set using bigram technique ....... 99

5.4 Feature vector extraction and data set preparation for the feature selection for ISCX data set using bigram technique ...................... 101

5.5 RFA bigram and non-bigram selection on ISCX data set 25 examples compared to random selection ........................................... 110

5.6 RFA bigram and non-bigram selection on ISCX data set 50 examples compared to random selection ........................................... 112

5.7 RFA bigram and non-bigram selection on ISCX data set 100 examples compared to random selection ....................................... 114

5.8 RFA bigram and non-bigram selection on ISCX data set 500 examples compared to random selection ....................................... 115

5.9 RFE bigram and non-bigram elimination on ISCX data set 25 examples compared to random selection ....................................... 118

5.10 RFE bigram and non-bigram elimination on ISCX data set 50 examples compared to random selection ....................................... 120
5.11 RFE bigram and non-bigram elimination on ISCX data set 100 examples compared to random selection ........................................ 123
5.12 RFE bigram and non-bigram elimination on ISCX data set 500 examples compared to random selection ........................................ 124
5.13 The simplest box-plot with its five components ........................................ 125
5.14 Boxplot for ISCX 25 (a) Detection rate (b) False Alarm Rate (c) Overall ........ 126
5.15 Boxplot for ISCX 50 (a) Detection rate (b) False Alarm Rate (c) Overall ........ 128
5.16 Boxplot for ISCX 100 (a) detection rate (b) False Alarm Rate (c) Overall .......... 129
5.17 Boxplot for ISCX 500 (a) detection rate (b) False Alarm Rate (c) Overall .......... 131
5.18 Box-plot for all of the proposed ranking coefficients in addition to the original one for detection rate for ISCX 25 data set ............... 140
5.19 Box-plot for all of the proposed ranking coefficients in addition to the original one for false alarm rate for ISCX 25 data set ............... 141
5.20 Box-plot for all of the proposed ranking coefficients in addition to the original one for overall metric for ISCX 25 data set ............... 142
5.21 Box-plot for all the proposed ranking coefficients in addition to the original one for detection rate for ISCX 50 data set ............... 143
5.22 Box-plot for all the proposed ranking coefficients in addition to the original one for false alarm rate for ISCX 50 data set ............... 144
5.23 Box-plot for all the proposed ranking coefficients in addition to the original one for overall metric for ISCX 50 data set ............... 145
5.24 Box-plot for all the proposed ranking coefficients in addition to the original one for detection rate on ISCX 100 data set ............... 146
5.25 Box-plot for all the proposed ranking coefficients in addition to the original one for false alarm rate on ISCX 100 data set ............... 148
5.26 Box-plot for all the proposed ranking coefficients in addition to the original one for overall metric on ISCX 100 data set ........................................ 149
5.27 Box-plot for all the proposed ranking coefficients in addition to the original one for detection rate on ISCX 500 data set ........................................ 150
5.28 Box-plot for all the proposed ranking coefficients in addition to the original one for false alarm rate on ISCX 500 data set ........................................ 151
5.29 Box-plot for all the proposed ranking coefficients in addition to the original one for the overall metric on ISCX 500 data set ........................................ 152
5.30 Partial ordering of the ranking coefficients with RFA on detection rate for the ISCX 25 data set ............................................................... 155
5.31 Partial ordering of the ranking coefficients with RFA on detection rate for the ISCX 50 data set ............................................................... 156
5.32 Partial ordering of the ranking coefficients with RFA on detection rate for the ISCX 100 data set ............................................................... 158
5.33 Partial ordering of the ranking coefficients with RFA on detection rate for the ISCX 500 data set ............................................................... 159
5.34 Partial ordering of the ranking coefficients with RFA on false alarm rate for the ISCX 25 data set ............................................................... 160
5.35 Partial ordering of the ranking coefficients with RFA on false alarm rate for the ISCX 50 data set ............................................................... 162
5.36 Partial ordering of the ranking coefficients with RFA on false alarm rate for the ISCX 100 data set ............................................................... 163
5.37 Partial ordering of the ranking coefficients with RFA on false alarm rate for the ISCX 500 data set ............................................................... 165
5.38 Partial ordering of the ranking coefficients with RFA on the overall metric for the ISCX 25 data set ............................................................... 166 

xix
5.39 Partial ordering of the ranking coefficients with RFA on the overall metric
for the ISCX 50 data set ........................................... 168

5.40 Partial ordering of the ranking coefficients with RFA on the overall metric
for the ISCX 100 data set ........................................... 169

5.41 Partial ordering of the ranking coefficients with RFA on the overall metric
for the ISCX 500 data set ........................................... 171

5.42 Binary tree structure for multi-class classification ............................... 172
Chapter 1

Introduction

Since the emergence of the Internet, it has dramatically impacted the whole world. Currently, the Internet is considered a significant element in our daily life. More and more, all fields in all disciplines strongly depend on the Internet. Due to the huge amount of services provided by the Internet, it has made a revolution in the world of communication and now many individuals’ needs are being fulfilled using the Internet. Those needs span several areas (e.g., communication, marketing, education, banking, and surveillance). According to Statistics Canada, in 2013, Canadian enterprises sold almost $136 Billion of goods and services through the Internet and about $122 Billion in 2012. That reflects the size of depending on the Internet in the marketing sector only. For interaction with governments online, 2013 statistics show that 46% of enterprises completed or returned tax forms online (Canada, 2014). For the individuals, in 2012 statistics show that 83% of Canadian households had Internet access at home, compared with 79% in 2010 (Canada, 2013). Ultimately, what has made the Internet as widespread as it is today is its ability to save effort and time of its users. Moreover, the Internet has helped in developing new disciplines and opened the door for new fields for research, for example cloud computing. Recent statistics indicate that the expected number of cloud-based online service users worldwide in 2018 to be approximately 3.6 billion internet users, up from 2.4 billion users in 2013 (Research, 2014).
However, that growth is not without risks and it has made the Internet a target for many types of threats. In the next section, we will discuss these Internet threats.

The rest of the chapter is organized as follows: Section 1.1 is specified for explaining the Internet’s threats, the threats targets are discussed in Section 1.2, Intrusion detection is explained in Section 1.3, while Intrusion Detection Systems are explained in Section 1.4. Section 1.5 presents the application machine learning and IDS, while the challenges that face IDS are explained in Section 1.6. In Section 1.7, we present the feature selection along with connection between feature selection and intrusion detection, in addition to the feature types and feature interdependency. We also explained the motivations of this work in Section 1.8 and the organization is explained in Section 1.9. The thesis statement is presented in Section 1.10. The summary of the work is presented in Section 1.11 and the goals of the work are explained in Section 1.12.

1.1 Internet’s Threats

Recently, attackers are targeting the Internet with many types of threats and attacks. These attacks are increasing as Internet usage is increasing as well. According to the 2013 Internet Security Threat Report published by Symantec Corporation, targeted attacks have increased by 42% from 2011 to 2012. These attacks vary in their severity from low damage to high damage. Viruses, malwares, worms, and intrusions are some of the Internet threats that threaten Internet security. In general, these threats are defined as follows:

(a) **A virus:** Is a harmful piece of code that has the ability to bind itself to host programs and reproduce when an infected program is executed.

(b) **A malware (i.e., malicious software):** Is any code added, changed, or discarded from a software system that leads to damages to the desired function of the system.

(c) **Worms:** Are malicious programs that target computer networks. Worms differ from viruses in strategy (i.e., they do not attach themselves to a host program).
An intrusion: Is any action that attempts to violate one or more of the network security goals (i.e., Confidentiality, Integrity, and Availability) (Alonso-Betanzos, Sanchez-Maron, Carballal-Fortes, Suarez-Romero, & Perez-Sanchez, 2007). Confidentiality means that the system is accessible only by authorized individuals. Integrity refers to the goal of protecting data in such a way to achieve accuracy and consistency. Lastly, availability means a data item, service, or system must be available upon request to authorized parties within a reasonable time slot (Pfleeger & Pfleeger, 2006).

In recent years, Internet users have suffered from many types of attacks. These cyber attacks are sometimes so damaging and cost billions of dollars every year (Roberto Di Pietro, 2008). Some of these attacks were able to access sensitive information and reveal credit cards numbers, delete entire domains, or even prevent legitimate users from being served by servers such as in the case of Denial of Service (DoS) attacks. One of the most common types of Internet attacks is intrusion. These days, the most popular Internet services are being attacked by many intrusion attempts that affect the network’s performance.

1.2 Threats targets

In general, there are three main targets for attackers in their attacking any computing system (Creutzburg, 2016):

(a) Data: target systems may be used to store personal/important data; they may be a good source for information to the attackers such as: credit card numbers, private information (i.e., driving license number, birth date) or any other important piece of information (i.e., pictures). Therefore, attackers usually try to access this information in order to remove, alter, gain monetary benefits, blackmail or any other malicious purpose.

(b) Identity: target systems can contain authentication information associated to its owner. Such information can compromise the identity of the owner/organization.
An attacker may impersonate the owner/organization to commit some other misbe- 
behaviour.

(c) **Availability:** attackers may limit the access to the system and prevent its legitimate 
users from getting any services from it causing a Denial of Service.

### 1.3 Intrusion Detection

Intrusion Detection (ID) is one of the sub-areas of computer and network security. ID is 
responsible for detecting most inappropriate activities on systems. The entire system which 
collects the information, monitors the computer systems, analyzes the incoming data, and 
makes the decision (whether there is an intrusion or not) is called an Intrusion Detection 
System (IDS). If the IDS is designed to detect attacks targeting computer networks then it 
is called a Network Intrusion Detection System (NIDS)(Debar, 2002).

### 1.4 Intrusion Detection Systems

In this section, a brief explanation about Intrusion Detection Systems is provided. As a con-
sequence of the aforementioned damages of intrusions on the Internet, exploring an effective 
countermeasure has become a necessity to protect Internet users against these intrusions. 
The evolution of these attacks has attracted many researchers around the globe to inves-
tigate and find a solution for such a problem. Therefore, the efforts of network security 
researchers from different universities, organizations or even governments have resulted in 
a countermeasure called an Intrusion Detection System (IDS). The IDS is responsible for 
detecting improper activities on a system (computer or network) and raising an alarm in 
the case of detecting an intrusion. IDSs can be classified into two types according to their 
place of operation: Host-based IDS and Network-based IDS (NIDS).

(a) **A Host-based IDS (HIDS)** resides in a host computer, where it monitors and 
analyzes log entries for particular information. Its operation requires periodically
looking for new log entries and comparing them with pre-defined rules. If the HIDS finds a match between a log entry and a rule, it raises an alarm. Some of the new versions of HIDS examine system calls to look for certain attack signatures. The HIDS takes action if it finds a match between a system call and any of the signatures (Singh & Silakari, 2009).

(b) A Network-based IDS (NIDS), on the other hand, involves monitoring all the traffic passing through a network card to and from the network. The NIDS then examines the traffic according to a set of rules and attack signatures, IP and transport layer headers of packets, and even the content of these packets to determine if the traffic contains an intrusion. If it finds one, an alarm is raised. One of the main problems of the NIDS is the huge amount of data collected from the network that needs to be analyzed and tested for any potential intrusion (Kayacik, Zincir-Heywood, & Heywood, 2005); (Singh & Silakari, 2009). Therefore, in order to deal with this huge amount of data, it will require a system that can identify which features in the data are relevant and also to quickly identify new threats as they are introduced. Identifying those relevant features to any problem from the other irrelevant features is called feature selection.

The focus of this thesis will be on NIDS and feature selection. In general there are two common types of IDSs according to their model nature: Anomaly-based IDS and signature-based IDS (or misuse-based IDS).

(a) An anomaly-based IDS involves learning the base-line or the normal behaviour of a system and any deviation in the observed behaviour from the normal behaviour will be considered an intrusion. The advantage of this type is its ability to detect novel attacks (because it depends on comparing to normal behaviour) while its disadvantage is that it suffers from a high false positive rate (Joo, Hong, & Han, 2003).

(b) A misuse-based IDS on the other hand, uses attack signatures of known intrusions, compares them with the observed signatures, and considers any match between them
as an intrusion. The advantage of this type is that it has high detection rate but its
disadvantage is that it cannot detect novel attacks (because it depends on comparing
to known signatures) (Singh & Silakari, 2009).

The focus of this work will be on the anomaly based IDS and feature selection using the
most recent used data set in this field (the ISCX 2012 data set), which will be discussed in
detail in Chapter 5.

The last few years have proved that the world requires continuous research, design and
implementation in the area of NIDS to protect the Internet from these evolving attacks
every day. The attacks, in the last few years, have been so sophisticated in a way that they
are highly destructive and hard to detect at the same time. Consequently, it has become
quite important to build an effective NIDS to protect the Internet from those attacks.

IDS/NIDS have been developed in the computer and network security fields due to the
fact that we cannot ensure a system is immune from security flaws. In general, the term
system or target system referred to here is the information system under monitoring by the
IDS, which can be a workstation, a network terminal, a server, a web server, or any other
computer system (Debar, 2002).

As stated above, a NIDS is responsible for detecting intrusions on a network and this is
performed by keeping network traffic under continuous monitoring. A NIDS collects its data
by using sensors to capture network traffic and examines individual packets to determine
whether they are normal or attack (Sathya, Ramani, & K.Sivaselvi, 2011).

In general, in network security there are two types of protection systems: passive systems
which are Intrusion Detections Systems (IDS) and reactive systems which are Intrusion
Prevention Systems (IPS). An IDS alarms the user when discovering malicious activity from
the observed activities. An IPS on the other hand, has a different reaction than IDS when
detecting suspicious activities, by auto-responding to that activity, resetting the connection,
or reprogramming the firewall to prevent network traffic from a suspicious source (Sathya
et al., 2011). In this thesis, we are looking at intrusion detection (which is also the first step in IPS).

1.5 Machine Learning

Recently, the study of IDS has gained a great deal of attention from the machine learning community. This attention came from the fact that an ID is considered an interesting classification task due to the increasing number of attacks on computer networks everyday (Bolon-Canedo, Sanchez-Marono, & Alonso-Betanzos, 2011). According to (Bolon-Canedo et al., 2011), the IDS’s classification task involves building a classifier which can make a distinction between intrusions, attacks, and normal connections. This classification problem can be of two types: a two-class problem if the goal is to distinguish attacks from normal connections, or a multiple-class problem if the goal is to distinguish between different types of attacks. The ultimate goal in both cases is to build a classifier that can generalize, i.e., achieve the classification task precisely on new (unseen) examples (Bolon-Canedo et al., 2011).

1.6 Challenges to IDS

There are always new threats, and when the threat is new, there are not enough examples to train a learning machine to distinguish between the normal traffic from threats instances. Moreover, network traffic contains many many features from different types. Therefore, the most two greatest challenges that face the IDSs are the small number of examples and the presence of many many features. These challenges tend to make any learning machine suffer from overfitting. Overfitting in turn may have a negative impact on the IDS performance. One of the biggest difficulties of network-based intrusion detection is the huge amount of data available for collection from the network. Accordingly, the raw network traffic that has been collected from the network needs to be distilled into higher-level events
(such as connection records) before feeding the data to a machine learning algorithm. This abstraction is done by describing each high-level event with a set of features (Kayacik et al., 2005).

1.7 Feature Selection

In the last decade, the area of feature selection has received a great amount of attention by machine learning researchers (Shanab, Khoshgoftaar, & Wald, 2011). It can be noticed that, in many pattern recognition and machine learning applications, the range of features have grown from tens to hundreds and thousands of features. These features may contain many irrelevant features which may affect application performance. Therefore, researchers have been looking for techniques to handle the problem of reducing irrelevant features which can negatively affect the given application (Chandrashekar & Sahin, 2014), while at the same time preserving features that are useful to the task at hand. An example of a classification task might be to separate healthy patients from cancer patients based on their gene expression (binary classification), where every gene is represented by a single feature (Bolon-Canedo, Sanchez-Maroo, Alonso-Betanzos, Bentez, & Herrera, 2014). A classifier that solves this problem needs to deal with thousands of features (genes) in the form of micro-array expression data to make its decision (i.e., healthy or cancerous). These features may contain many irrelevant features which can affect the classifier performance. Other features may be interdependent, when they are only useful in combination with other features (Yang, Salehi, & Gras, 2010).

Feature selection aims at finding the best subset of features from the entire number of features that can represent the input data efficiently and can still provide good prediction results (Chandrashekar & Sahin, 2014);(Moustakidis & Theocharis, 2012). Feature selection uses a search algorithm to find one or more informative subsets of features according to predefined criteria. This process can be defined in the following way:
Let \( F = \{F_1, F_2, ..., F_n\} \) be the entire set of features; \( S = \{F_{\tau(1)}, F_{\tau(2)}, ..., F_{\tau(m)}\} \) \((S \subseteq F)\) be a selected subset of features from the entire set, where \( m < n \). The goal of feature subset selection is to select the most informative subset \( S_{optimal} \subseteq F \) that represents the original data according to some criteria \( J \) (Zeng, Zhang, Zhang, & Yin, 2015). However, the original set of features may contain some irrelevant features. According to John, Kohavi, & Pfleger (1994), a feature \( F_i \) is considered relevant if and only if \( p(Y = y|F_i = f_i) \neq p(Y = y) \) where \( Y \) is the label, or output. According to this definition, feature \( F_i \) is relevant if its value can change the prediction for \( Y \). In other words \( Y \) is conditionally dependent of \( F_i \).

Feature selection should be able to find the important or relevant features (i.e., the features that are relevant to the given prediction task) (Yang et al., 2010). If a problem has \( N \) features, the number of all subsets of features is equal to \( 2^N \). Therefore, the optimal set of features is one (or could be more) of an exponential number of possible subsets, and comparing all of these subsets to find the best is intractable for \( N > 20 \). Consequently, it is necessary to find a good subset of features. Feature selection determines the feature relevance according to an evaluation criterion associated with the given method. Basically, each feature has two possible cases: present or absent. Therefore, a feature selection algorithm has to find a good subset from \( 2^N \) possible subsets of features for a data set that consists of \( N \) features. Indeed, the best subset of features refers to the subset of features that gives the best classification performance among the \( 2^N \) feature sets. This makes the problem an NP-hard (Non-deterministic polynomial-time hard) problem as the number of features grow (Chandrashekar & Sahin, 2014).

The presence of extra features usually increases the difficulty of training a classifier. In addition, irrelevant features can confuse the classifier, or can lead to overfitting. Overfitting refers to the phenomenon where perfect prediction can be made by a classifier on a training data set, while achieving poor prediction on test data (Guyon & Elisseeff, 2003). This typically occurs when there are so many parameters in the model that the system is able to learn the data, rather than the trends in the data. This is especially common when there are many parameters in the system compared to the numbers of training examples. Indeed,
the number of parameters in the classifier is typically a function of the number of features. Another advantage of feature selection is that it helps in getting insight into the useful features and helps in understanding what the classifier is doing (Moustakidis & Theocharis, 2012); (Wang, Pedrycz, Zhu, & Zhu, 2015); (Yao, Mao, Goodison, Mai, & Sun, 2015). Feature selection also helps to decrease the impact of the curse of dimensionality problem (Bolon-Canedo et al., 2014); (Chandrashekar & Sahin, 2014). The curse of dimensionality refers to a common phenomenon that excessively restricts the performance of different algorithms such as: searching, clustering, classification, and other in major operations used by applications of data mining applied on high-dimensional data (Bernecker, Houle, Kriegel, Kröger, Renz, Schubert, & Zimek, 2011). In addition to mitigating the curse of dimensionality, feature selection may considerably reduce the computational complexity and the cost of learning with these irrelevant features. It may also help in providing a great insight to the nature of the given problem (Yao et al., 2015). Therefore, feature selection has been employed to alleviate the above problems in many applications. However, some feature selection methods still struggle with the overfitting problem since this problem may lead to inaccurate results (i.e., not better than without feature selection).

It is also worth mentioning here that feature selection might fail sometimes in improving the performance when the selected features produce equal or less performance than the performance obtained without feature selection. This might occur when all features are required and important to make accurate classifications or the features have been already chosen carefully to be relevant to the given problem.

Practically, feature selection is now being used in many applications such as gene microarray analysis (Sharma, Paliwal, Imoto, & Miyano, 2014), network security (Wahba, ElSalamouny, & ElTaweel, 2015), medical diagnosis (Guyon, Weston, Barnhill, & Vapnik, 2002); (Liu, Liu, Gu, Chen, & Chen, 2015), object recognition (Rainey & Stastny, 2011), and many other applications.
Usually, feature selection algorithms are categorized into three different types: filter, wrapper and embedded methods (Saeys et al., 2007). More details will be provided in Chapter 2.

1.7.1 Connection between Feature Selection and Intrusion Detection

After converting the ID problem into a data set and using machine learning with the intrusion detection problem, researchers have incorporated feature selection with intrusion detection. Ultimately, the intrusion detection problem is a two-class classification problem that is capable of distinguishing between intrusions and normal connections (Uday Babu P., 2014). One of the recent useful techniques in NIDS is the feature selection technique. An accurate detection of network intrusions by NIDS relies greatly on feature selection, which can depict the pattern of the network packets. The goal of using feature selection with NIDS is to improve the detection accuracy by removing irrelevant/redundant features from the input data. As a result, a new set of features that consists of only a subset of the original features will be selected (Mzila & Dube, 2013).

1.7.2 Feature types and Feature interdependency

As stated before, feature selection aims at selecting the features that contribute the most to classification task. However, the process of evaluating the importance (relevance) of a feature could be different from one technique to another. According to Guyon & Elisseeff (2006), individual relevance is one of the feature ranking techniques that is used to rank features according to their individual importance. Other techniques consider a feature informative in combination with other features but not alone (Draminski, Rada-Iglesias, Enroth, Wadelius, Koronacki, & Komorowski, 2008). These features are called interdependent features and the technique is called feature interdependency discovery. The reasons behind emergence of the second technique are that features that are individually irrelevant may become relevant in combination with others, and features that are individually relevant may not all be relevant due to potential redundancies (Guyon & Elisseeff, 2006). The
importance of this technique can be clearly shown in bioinformatics in finding cooperative genes (Dramiński, Kierczak, Koronacki, & Komorowski, 2010).

1.8 Motivation

Feature selection has been a vital preprocessing step in many machine learning applications. As stated before, a data set that comprises of many many features and few examples can lead to overfitting. In general, tracing new attacks on the Internet may reveal only few examples. However, from those few examples many, many features can be extracted from the network traffic. These attacks could be devastating to the networks and to the underlying systems. Therefore, finding a solution for this problem has become a necessity. In addition, the collected data set may contain some interdependent features that work better when they are combined together. Therefore, the motivation of this work is to design a feature selection method that can detect interdependent features among a large set of potential features and can be used in intrusion detection to detect any new threat that may compromise the Internet and its users based only on a limited number of examples.

1.9 Organization of the thesis

This thesis is divided into six chapters as follows: In Chapter 2, a comprehensive literature review about feature selection and its types including filter, wrapper and embedded methods are provided. Next, the major key-steps of feature selection are discussed. Also, the data classification and use of SVMs for classification are explained. The chapter concludes with how to evaluate the classifier’s performance. In Chapter 3, an analysis of the limitation of RFE is discussed. This chapter includes conducting a deep analysis experiment on RFE. The experiment involves proposing a synthetic data set to simulate interdependent features. The results on this data set motivates our search for a superior technique. The chapter also explains the proposed features selection method in detail. Next, the chapter describes the
methodology and its steps for the conducted experiments. In Chapter 4, the results of applying feature selection on three groups of data sets are explained in detail. The results of applying both the proposed feature selection method and RFE will be shown. The feature selection will be applied on real-world small data sets, synthetic data sets, and real-world large data sets. The statistical significance of the results are shown also in this chapter using statistical analysis. Since one of the objectives of this thesis is to develop a feature selection system that is suitable for intrusion detection, Chapter 5 starts by introducing the ISCX 2012 intrusion detection data set. The chapter then discusses the feature extraction and data set preparation. Next, all the details of applying feature selection on the ISCX 2012 intrusion detection data set including RFA and RFE are provided. The chapter shows also the statistical test of the results for both RFA and RFE. Lastly, Chapter 6 provides a discussion about all the results of the thesis.

1.10 Thesis statement

In this thesis, the limitation of a very successful feature selection method called RFE is diagnosed and studied. A new feature selection method called RFA is proposed to deal with irrelevant interdependent features. The existence of irrelevant features and interdependent features both affect the effectiveness of some feature selection methods especially in high dimensional data sets. We are going to study if we can improve the intrusion detection performance with the existence of interdependent features, along with many features and few examples.

1.11 Summary of the work

In this work we introduce a novel feature selection method called RFA and we show that RFA is a useful feature selection tool that overcomes the observed limitations of RFE. RFA outperforms RFE on real-world data sets with large numbers of features and small numbers
of examples. We examine the ability of the proposed method to detect interdependent features. We also show that RFA is a good system for intrusion detection problem since usually new attacks have small numbers of examples and large numbers extracted features. We apply our proposed method RFA on the most recent intrusion detection data set: ISCX 2012.

It is also worth mentioning here the list of publications from this work:


(c) A limitation of RFE and A practical Alternative (Submitted journal paper to Pattern Recognition, 2017), Tarfa Hamed, Rozita Dara, and Stefan C. Kremer.

(d) Network Intrusion Detection System Based on Feature Selection and Bigram Technique (journal paper ready for submission, Tarfa Hamed, Rozita Dara, and Stefan C. Kremer).


1.12 Goals of the work

This thesis presents to the reader a novel method for feature selection called RFA. It shows that RFA can outperform the leading alternative (RFE) on a number of real-world problems. The thesis shows that these results are statistically significant. Additionally, both RFE and RFA are applied to the most recent intrusion detection data set: ISCX 2012. This thesis will equip readers with a new tool to include in their feature selection toolbox.

The goal of this work is to develop feature selection systems used for IDS particularly when there are few examples and many features. The new tool for feature selection can be used in ID to make a more secure Internet, bioinformatics and many other applications. The goal is to make the Internet more safe and to maintain the privacy of all who benefit from the Internet. As it is known, the Internet experiences many attacks every day. Analyzing these attacks usually requires extracting many features from the network traffic. These many features may contain interdependent features and irrelevant features. Therefore, identifying the interdependent features from the irrelevant features can help in detecting the new attacks in the early stages of the attack. Using the RFA tool to identify the relevant features can help in detecting new attacks. Since for every new attack, there are only few examples and many features, then RFA can be the most suitable tool to detect these attacks. By applying this strategy with every new attack, we can obtain a better and more secure Internet environment.
Chapter 2

Literature Review

The purpose of this chapter is to discuss what has been done in the literature in the field of feature selection and how feature selection. How feature selection is used with noisy features is covered in Section 2.1. In addition, the chapter explores feature selection types in some detail in Section 2.2. Major key-steps of feature selection are explained in Section 2.4. The chapter also presents data classification and using SVM as a classifier in Section 2.5. Measuring a classifier performance is explained in Section 2.6. Section 2.7 gives a summary of this chapter.

2.1 Feature Selection

Feature selection has been attracting the attention of the machine learning community for the last 15 years. One of the major challenges faced by any feature selection technique is that it must be able to distinguish features that contain valuable information from those that contain only or primarily noise. However, relevant features could be independent (i.e., they have importance when they are taken individually), or could be interdependent (i.e. they are important when they are together and they are useless when they are individual). Identifying noisy features is also a crucial preprocessing step as the performance of the classifier constructed on such noisy data will highly depend on the quality of the training data.
Noisy features will act in complicating the problem that would require complex decision boundaries for data separation (García, de Carvalho, & Lorena, 2015). Therefore, several algorithms have been proposed to deal with noise in feature selection. Feature selection is the process of selecting the best (most useful) subset of features and discarding the irrelevant ones from the set of features of a dataset under investigation. The goal behind feature selection is to obtain a subset of features that describes correctly the given problem (Bolon-Canedo et al., 2011). This process has multiple advantages (Guyon & Elisseeff, 2003):

- Producing faster and lower cost predictors due to data reduction. Reducing the dimensionality of the problem, means less data to process and less computation time taken by predictors.

- Improving the prediction performance of the machine learning predictors by avoiding overfitting. As stated before, having a data set with so many features and only few examples may lead to overfitting during the learning process. Therefore, selecting only the informative features from the entire set of features can improve the prediction performance and avoid overfitting.

- Providing a clear understanding of the implicit process that generates the data. Feature selection also leads to transparency in understanding the data set and what the informative features to the given problem are, and, at the same time, what the irrelevant features to that problem are.

This process is necessary since some datasets contain irrelevant or redundant features and the existence of those features can have a negative impact on the used classifier in terms of computation time and resources required, even if the used classifier was resistant to irrelevant or redundant features. One effort in dealing with noise was in Wei, Ma, Hu, Su, & Ma (2014), which considered the feature noise for the sake of testing the robustness of several feature selection algorithms. Every algorithm was tested by adding 5% and 10% of Gaussian noise to the raw data. They computed the differences of classification
performance between different data sets with varying noise levels. The smaller the difference in classification performance is, the more robust the feature selection algorithm is. They found that Brownboost loss function is more robust to noise than other loss functions. They also found that greedy feature search method to minimize classification loss (GFS-MCL) selects less number of features and produces higher classification performance and better robustness compared to other methods with gene data. In another study, the authors in Liu et al. (2015) proposed a new feature selection method with certain noise tolerance. Their objective was to design a feature selection method resistant to the inevitable noise in some data sets. Their method was called FECS (FEature Clustering with Selection Strategies), which consisted of two phases: a feature clustering phase and a feature selection phase. During the feature selection phase, two types of noise were injected: class noise and feature noise to simulate noisy data sets. To tolerate different kinds of noise, the authors used three different heuristic strategies.

- **Strategy 1:** for each cluster \( C_i \), select the feature that has the most feature-to-class relevance.

- **Strategy 2:** for each cluster \( C_i \), select the medoid (cluster centre) that has the maximum total similarity to the other clusters based on feature-to-feature correlation.

- **Strategy 3:** for each cluster \( C_i \), select the feature that has the minimum total similarity to the medoids of all clusters except \( C_i \).

The proposed method was assessed using the AUC (Area Under the ROC Curve) metric and RLA (Relative Loss of Accuracy). AUC is an evaluation metric used to measure the classification performance of a binary classifier based on the ROC (Receiver Operating Characteristic) Curve by making a balance between true positive rate and false positive rate. AUC values are in the range 0-1, where an ideal classifier has a value of 1 as shown in Figure 2.1. In Figure 2.1, we show three curves for AUC for three cases of classifications: good, random and perfect. In the good case, a classifier aimed at finding a solution as close to true positive rate and far from false positive rate as possible as shown in the red
In addition to affecting the accuracy of the resultant classifier, noise has also been observed to affect the robustness of feature selection. Therefore, researchers used RLA which is an evaluation metric used in measuring the loss of classification accuracy of a binary classifier built on noisy data.

\[
RLA_{x\%} = \frac{AUC_0 - AUC_{x\%}}{AUC_0}
\]  

(2.1)

where \(x\%\) represents \(x\%\) noise injected in the data set and \(AUC_0\) represents the \(AUC\) value that is measured before injecting the noise in the data set. The proposed method was compared to other feature selection algorithms (Correlation based Feature Selection, Information Gain, and Consistency subset evaluation), and showed better performance than most of them.

In Huang, Yoo, Yu, & Qin (2014), the authors proposed a Noise Resistant Feature Selection (NRFS). Their objective was to build an effective and robust unsupervised fea-
ture selection technique for both noisy observations and noisy features. Their proposed method utilized a multi-perspective correlation measurement in designing the feature selection method. Their NRFS method selected features based on examining the feature’s effect through a local perspective of representative instances and a global spectrum. In this way, it was able to successfully identify diverse and informative features from the remaining ones. To decrease the impact of noisy features, they extracted a multi-perspective correlation for each feature with respect to each instance and global spectrum. They compared their method to Multi-Cluster Feature Selection (MCFS), Nonnegative Discriminative Feature Selection (NDFS), Spectral Feature Selection (SPEC), Laplacian Score (LS), and a $k$-means method and their method showed better performance and robustness than these algorithms on microarray and text data sets.

According to Altidor, Khoshgoftaar, & Napolitano (2011), stability can be defined as the degree of agreement between a feature selection output (i.e., the set of selected features) on a data set corrupted with different combinations of noise level and noise distribution and its output on the corresponding clean data. Their objective was designing an evaluation approach for feature selection techniques. The proposed stability measure was used to assess 11 Threshold-Based Feature Selection techniques (TBFS) on six real-world binary classification data sets. Their results showed some methods outperformed other methods with regard to their insensitivity to noise. Their observation also showed that the stability performance of feature selection is proportional to the size of a data set - the more examples in the data set, the higher the stability of the TBFS methods.

One application area in which feature selection is especially important is bioinformatics since most of bioinformatics data sets are characterized by high dimensionality and relatively few examples which may cause overfitting. In Van Hulse, Khoshgoftaar, & Napolitano (2011), the authors provided a comparative evaluation of feature ranking methods for bioinformatics data sets. In their work, they compared their proposed method named Threshold-Based Feature Selection technique with the AUC metric (TBFS-AUC), with three other filter methods: $\chi^2$, Information Gain (IG), and ReliefF (RF) on 17 high-dimensional bioin-
formatics data sets. Their method involved normalizing each feature between 0 and 1, then evaluating each feature against the class independently of all other features. A threshold value $t \in [0,1]$ was defined to be used in building a Nave Bayes classifier. To evaluate the performance of the classification model, they used three metrics, area under curve (AUC), area under precision-recall (APRC), and F-measure. The AUC metric is an evaluation metric used in evaluating classification models, using the posterior probabilities computed by these models in classifying examples as either positive, $P$, or negative, $N$, based on the classification threshold. It was assumed that each example $x$ would belong to one of two classes $c(x) \in P, N$.

The predicted class of example $x$, referred as $\hat{c}(x)$ is decided based on a threshold $t \in [0,1]$. If $p(P|x) > t$, then $\hat{c}(x) = P$; otherwise $\hat{c} = N$. The AUC was calculated as the area under the curve resulting from the true positive and false positive rates. The ARPC curve was generated by plotting the recall and precision while varying the threshold from 0 to 1. Recall is simply the true positive rate, while precision is the percentage of the truly classified positive examples relative to the total number of positive examples. The last metric is the F-measure which is the harmonic mean of precision and recall. Their TBFS-AUC method outperformed the other three methods in all three of the metrics.

In Liu, Chen, Zhang, & Hu (2011), the authors proposed a SVM with a RBF kernel based on the RFE algorithm (SVM-RBF-RFE). Their algorithm expanded the non-linear RBF kernel into its Maclaurin series and the weight vector $w$ is calculated from the series depending on each feature’s contribution to the classification hyperplane. The proposed algorithm used $w_i^2$ as a ranking criterion and was designed to start with all the features and keeps eliminating one feature with the least squared weight at each iteration until all the features are ranked. Their approach used SVM and KNN classifiers to evaluate the nested subset of features selected by SVM-RBF-RFE. Their experiments involved applying 3 UCI and 3 microarray data sets and the results showed that SVM-RBF-RFE generally performed better than RFE and information gain.
A regularized linear discriminate analysis technique was another feature selection method proposed in Sharma et al. (2014), which considered sensitivity analysis to check the robustness of the method. Their objective was to find the significant genes for a cancer classification task. They applied their technique on three DNA microarray gene expression data sets. The analysis involved contaminating the input data set with Gaussian noise, then applying their proposed method to find the top 100 features (genes). The injected noise levels were varied between 1, 2, and 5% of the standard deviation of the original gene expression values. The contamination of data and gene selection were repeated 20 times and the performance was averaged over 20 iterations. The proposed method was compared to other feature selection algorithms (information gain, towing rule, sum minority, max minority, Gini index and sum of variances), and showed better performance than all of them. In Yao et al. (2015), an unsupervised feature selection was proposed to deal with high dimensional biological data. The authors’ goal was to develop a feature selection algorithm to (1) distinguish features facilitating intrinsic geometry of high dimensional data without explicitly supposing the form of data structures, and (2) prepare some significant statistics to point out the absence of data structures if data lacks to provide any meaningful structure. Their approach involved using gap statistics to estimate the parameters and permutation tests to evaluate the statistical significance of the existence of a detected data pattern. Their algorithm showed its outperformance over seven feature selection algorithms.

2.2 Feature selection types

In general, feature selection methods can be divided into three types: Filter methods, Wrapper methods, and Embedded methods (Saeys et al., 2007).

(a) Filter methods involve the methods that perform feature selection before building the classifier and do not incorporate learning.
(b) Wrapper methods incorporate a learning machine in measuring the quality of the subsets of features without incorporating knowledge about the specific structure of the classification or regression function.

(c) Embedded methods are different from Filter and Wrapper methods, in that with embedded methods, the learning part and the feature selection part cannot be separated (Guyon & Elisseeff, 2003).

In the next sections, a detailed description about each of the aforementioned methods is provided.

2.2.1 Filter Methods

Filter methods include the methods where feature selection is independent of the classifier to be applied to the selected features. These methods determine the feature importance by inspecting the intrinsic properties of the data. Commonly, filter methods calculate a feature relevance score for all the features and remove low-scoring features. The main advantage of filter methods is their ability to deal with high-dimensional datasets (i.e., not affected by the curse of dimensionality). In addition, they are independent of the classification algorithm, faster than other feature selection methods as they are not computationally intensive. An inherent shortcoming of filter methods is that they neglect the interaction with the classifier. In other words, they ignore the impact of the selected subset of features on the performance of the induction algorithm (Kohavi & John, 1997); (Saeys et al., 2007). Since we used a filter method for pre-ranking and also compared our proposed method to two filter methods: Correlation-based Feature selection (CFS) and Consistency Subset evaluation, we will explain those methods in the next sections.

Correlation-based Feature selection (CFS)

The CFS method was proposed by Hall (1999) and has been used in many machine learning applications. The CFS method is based on looking for the features that are highly correlated
with the class but uncorrelated with each other. The CFS algorithm utilizes a correlation-based heuristic evaluation function to rank the given features. A feature will be selected if it can predict classes in the instance space that were not already predicted by other features. This function considers the goodness of each subset of features in conjunction with the degree of inter-correlation among these features in predicting the class label. The core of the heuristic function is based on the Equation 2.2.

\[ Merit_s = \frac{k\overline{r_{cj}}}{\sqrt{k + k(k - 1)\overline{r_{ff}}}} \]  

(2.2)

Where \( Merit_s \) represents the heuristic merit of a feature subset, \( S \), consisting of \( k \) features, \( \overline{r_{cj}} \) is the average feature-class correlation \( (f \in S) \), and \( \overline{r_{ff}} \) represents the average feature-feature inter-correlation. The search space is all the possible feature subsets. The CFS method uses a forward best-first search method to search the feature subset space and keeps adding features to an empty feature set until a stopping criterion is met. The stopping criterion is five successive subsets with no improvement.

**Consistency Subset Feature Selection Method**

The consistency subset feature selection method is based on the probabilistic approach of the Las Vegas Filter (LVF). This method was proposed, as heuristic search methods suffer with datasets of high order correlations. The algorithm starts by specifying the current number of features \( C_{best} \) from the original number of features \( N \) as in step 3. The method iterates for \( MAX\_TRIES \) iterations to produce a subset of \( M \) features satisfying criterion \( \gamma \) at each iteration. The method generates a random subset \( S \) that consists of \( C \) features as in step 6. If the number of features \( C \) is less than the current best (i.e., \( C < C_{best} \)), then data set \( D \) of \( S \) features is checked against the inconsistency criterion as in steps 8 and 9 respectively. If the inconsistency rate is less than an acceptable inconsistency rate \( \gamma \), then \( C_{best} \) and \( S_{best} \) are replaced by \( C \) and \( S \) respectively as in step 10. Then the new current best \( S \) is printed as in step 11. If \( C = C_{best} \) and the inconsistency criterion is acceptable, then a good current best \( S \) is found and printed as in steps 13 and 14 respectively (Liu & Setiono, 1996).
Algorithm 1 Las Vegas Algorithm (LVF)

1: procedure Consistency Subset evaluation
2: \begin{align*}
3: \textbf{Input:} & \text{ MAX\_TRIES, data set } D, \text{ number of features } N, \text{ acceptable inconsistency rate } \gamma \\
4: & \text{ } \quad C_{\text{best}} \leftarrow N \\
5: & \text{ } \quad i \leftarrow 1 \\
6: \end{align*}

7: \begin{align*}
8: \text{ while } i \leq \text{MAX\_TRIES do} & \\
9: & \quad S \leftarrow \text{randomset} (\text{seed}); \\
10: & \quad C \leftarrow \text{numOfFeatures}(S) \\
11: & \quad \text{if } (C < C_{\text{best}}) \text{ then} \\
12: & \quad \quad \text{if } (\text{InconCheck}(S, D) < \gamma) \text{ then} \\
13: & \quad \quad \quad S_{\text{best}} \leftarrow S; C_{\text{best}} \leftarrow C; \\
14: & \quad \quad \quad \text{print\_Current\_Best}(S) \\
15: & \quad \quad \text{else} \\
16: & \quad \quad \quad \text{if } ((C = C_{\text{best}} \text{ and } (\text{InconCheck}(S, D) < \gamma))) \text{ then} \\
17: & \quad \quad \quad \quad \text{print\_Current\_Best}(S) \\
18: & \quad \quad i \leftarrow i + 1. \\
19: \end{align*}

20: \textbf{endwhile}

21: \textbf{Output: } subsets $S$ of $M$ features satisfying the inconsistency criterion .

The effectiveness of the above algorithm is based on the inconsistency criterion when it is checked in this step ($\text{InconCheck}(S, D) < \gamma$). This criterion controls the accepted range of dimensionality reduction of the input data. The inconsistency rate is calculated for the selected features and checked against ($\gamma$). If the obtained value is less than $\gamma$, this means this dataset is acceptable. The algorithm sets $\gamma$ to 0 as the default value, though other values can be specified. The inconsistency rate of a dataset is calculated as follows:

1. Each two examples are considered inconsistent if they match except for their class labels.
2. For all those matching examples, the inconsistency value is the total number of the matching examples minus the number of examples of the biggest class. For example: assume there were \( n \) matching examples, with \( c_1 \) examples among them belonging to class label 1, \( c_2 \) examples belonging to class label 2, and \( c_3 \) examples belonging to class label 3 (where \( c_1 + c_2 + c_3 = n \)). Suppose that \( c_3 \) was the biggest among the three, then the inconsistency value is \( z = n - c_3 \).

3. The inconsistency rate is the sum of all the inconsistency values divided by the total number of examples as shown in Equation 2.3:

\[
Inconsistency\ rate = \frac{\sum_{i=1}^{n} z_i}{n}
\]  

(2.3)

where \( n \) is the total number of examples.

2.2.2 Wrapper Methods

Wrapper methods incorporate a learning machine in measuring the quality of the subsets of features without incorporating knowledge about the specific structure of the classification or regression function. Wrapper methods involve wrapping the feature selection around the classifier construction (Lal, Chapelle, Weston, & Elisseeff, 2006);(Draminski et al., 2008). However, wrapper methods are more computationally intensive than filter methods since wrapper methods require constructing a new predictor for every candidate feature subset (May, Bannach, Davey, Ruppert, & Kohlhammer, 2011). A common shortcoming of the wrapper methods is that they are prone to overfitting and very computationally intensive. The wrapper subset evaluation process is illustrated in Figure 2.2, below:

Since we also compared our proposed method to two wrapper methods: Wrapper subset evaluation and Classifier subset evaluation, we will explain those methods in the next sections.

1. Wrapper Subset Evaluation Method
The wrapper subset selection method was proposed by Kohavi & John (1997). As stated above, the general wrapper feature selection methods involve using a learning algorithm for evaluating the goodness of the generated subset of features. This interaction between the feature selection and the learning algorithm is applied to obtain the best possible performance with a specific learning algorithm on a specific training set. The feature subset selection works by applying a wrapper around the learning or induction algorithm. The search for a good subset of features is performed by the feature subset selection algorithm using the induction algorithm as shown in Figure 2.2.

The wrapper approach deals with the induction algorithm as a black box. The data is fed to the induction algorithm, and then split into training and holdout sets with different subsets of features. The feature selection algorithm chooses the feature subset that recorded the maximum evaluation as the final set on which to execute the induction algorithm. The last step involves evaluating the resulting classifier on an independent test set that was not involved in the search process.
2. Classifier Subset Evaluation Method

The Classifier Subset Evaluation method is also a wrapper method as it uses a classifier in the evaluation of the generated subsets of features on the training data or on a separate holdout set. This method is similar to the Wrapper Subset Evaluation but it does not involve cross-validation (Witten & Frank, 2005);(Kohavi & John, 1997).

2.2.3 Embedded Methods

Embedded methods are different from Filter and Wrapper methods, in that with embedded methods, the learning part and the feature selection part cannot be separated (Guyon & Elisseeff, 2003). Embedded methods perform the feature selection in the process of learning, which saves the time required for two-step induction as in the wrapper methods (Guyon, 2008). The search for the best subset of features is built into the classifier construction (Saeyes et al., 2007). Embedded methods have more efficiency over wrappers in better utilizing the available data without the need to split the training data into training and validation sets (i.e., efficient use of the available data). Thus, embedded methods are more of a “white box” method, since the feature selection is based directly on the classifier. In addition, they find a solution faster since there is no need to retrain a predictor from scratch for every examined subset of variables (Guyon & Elisseeff, 2003). One of the most well-known embedded methods is SVM-RBF-RFE or (RFE) which was proposed by Guyon et al. (2002). The method tries to find the best subset of features of size $S$ by employing a greedy backward selection. The method is based on SVM classifier as it performs feature selection in the process of learning. It works by trying to find the $S$ features that generate the largest margin of class separation. The method operates in a recursive fashion by removing the feature that causes minimum margin class separation until only $S$ input features are left (Guyon et al., 2002);(Lal et al., 2006).
2.2.4 Hybrid methods

Interestingly, there have been some efforts that adopted combining both Filter and Wrapper methods in one approach to select the best subset of features for classification as in Liu & Zheng (2006). The authors proposed Filtered and Supported Sequential Forward Search (FS_SFS) with Support Vector Machines (SVMs). The method consisted of two parts: a filter part and a wrapper part. The filter part was used first to reduce the number of features, and then the resulting subset of features was fed into the wrapper part. The wrapper part’s task involved evaluating those features by calculating the accuracy of the classification by training a SVM classifier. In our methodology we will do something similar with a Filter method and our RFA Embedded method.

2.3 Utilizing feature selection in intrusion detection

Many studies have been conducted on applying feature selection to improve the IDS performance. Those studies used different IDS data sets for testing their models. However, in this thesis we use the ISCX 2012 data set (which will be explained in details in Chapter 5. In Vasudevan & Selvakumar (2015), the authors applied the intraclass correlation coefficient and interclass correlation coefficient to obtain class-specific subset of features. The interclass and intraclass correlation coefficients were used to measure the validity and the reliability of features respectively. The authors tested their model on the ISCX 2012 data set. They observed that the above combination between interclass and interclass correlation coefficients led to increase the detection rate and to decrease both execution time and false alarm rate.

In other studies Heidarian, Movahedinia, Moghim, & Mahdinia (2015), the authors opted to build their intrusion detection system based on the normal traffic to detect unseen intrusions using the ISCX 2012 data set. The authors employed one-class Support Vector Machines (SVMs) classifier the learn http regular traffic attributes for anomaly detection task. Their approach involved extracting appropriate attributes from normal and abnormal
http traffic. The system was trained on the normal traffic examples to learn the normal behaviour. The system generates an alert if it finds any deviation from the normal traffic model. The authors obtained 80% accuracy and 8.6% false alarm rate in detecting attacks on port 80.

Other studies such as Yassin, Udzir, Abdullah, Abdullah, Zulzalil, & Muda (2014) used Signature-based Anomaly Detection Scheme (SADS) in examining packet headers to extract behaviour patterns more accurately. The authors combined both Signature-based Detection System (SDS) and Anomaly-based Detection System (SDS) as to overcome their limitations the formal suffers from its inability to detect novel attacks while the latter produces high false positive rate. Their proposed model was built by integrating both Naive Bayes (NB) classifier and Random Forest (RF) to minimize false alarms and to generate signatures. The model was tested on the ISCX 2012 data set and they observed that their model was able to outperform the conventional anomaly detections system in terms of detection rate and false alarm rate.

The authors in Kumar & Kumar (2013) designed a new multi-objective optimization approach for efficient intrusion detection. Their proposed approach involved encoding of chromosomes that offer the best subset of features. Those features could be later used to train varied instances of Naive Bayes (NB) classifier for ensembles. Their proposed approach was tested on the ISCX 2012 data set to validate its performance. For the ensemble, the authors used bagging and boosting to make use of multiple models at the same time. Their approach was able to achieve 95.2% detection rate for the normal data and 92.7% for the attack data respectively. However, the observed drawback of that model is that very computationally expensive in computing fitness functions in different generations.

The concept of using ensemble features selection was used by Milliken, Bi, Galway, & Hawe (2015) also to detect pairing between features. They built their hypothesis on combining and pairing features could improve the performance. They made four different combinations between features of the ISCX 2012 data set and checked the performance to see if the pairing has a positive impact on the performance of IDS. They also observed
the ability of the ensemble to distinguish between those examples that are actually Normal from the other instances.

The authors in Tan, Jamdagni, He, Nanda, Liu, & Hu (2015) proposed an IDS that handles traffic records as images to detect Denial of Service (DoS) attacks using computer vision techniques. The proposed approach involved utilizing multivariate correlation analysis to describe network traffic records and convert them to images. Then those images were used to detect DoS attacks by utilizing dissimilarity measure called Earth Mover’s Distance (EMD). EMD is based on taking cross-bin matching and producing a more precise evaluation on the dissimilarity between distributions than other dissimilarity measures. They found that some pairings achieved better performance in terms of false alarms and false negatives. The model was tested using ten-fold cross-validation and achieved 90.12% accuracy in detecting DoS attacks.

2.4 Major Key-Steps of Feature Selection

In general, a feature selection process consists of four major key-steps as shown in Figure 2.3 (Chen, Li, Cheng, & Guo, 2006); (Liu & Yu, 2005):

![Figure 2.3: Four key-steps of feature selection](image)

(a) **Subset Generation**: This step involves specifying a candidate subset of features for evaluation. Usually this step deals with two basic issues. First, the search starting
point(s) must be selected, which will determine the search direction. The search direction can be forward selection, backward selection, or bidirectional. The forward selection (addition) starts with an empty set and keeps adding features iteratively until a stopping criterion is met. The backward selection (elimination) starts with the full set of features and successively removes one feature or more until a stopping criterion is met. The third one is the bidirectional selection which starts from both ends and adds and removes features simultaneously during the iteration until a stopping criterion is met. The second issue that the subset generation needs to deal with is deciding the search strategy. If we have a dataset with $N$ features, then we have $2^N - 1$ candidate subsets. Therefore, the search space grows exponentially with $N$ and can take months or even years to evaluate all the candidate subsets even with super-computers. In general, numerous techniques can be explored for subset generation: complete, sequential, greedy, and random search (Chen et al., 2006); (Lal et al., 2006).

The filter, wrapper, and embedded methods differ in terms of the search method that they utilize. The filter feature selection methods usually use either ranking methods to rank individual features (not subsets) according to their evaluation or one of the search methods such as best-first to search for the subsets of features. The wrapper methods use only search methods as they only generate subsets of features (Witten & Frank, 2005). In the embedded methods, the search for the best subset of features is included into the classifier construction; evidently the search is performed in the combined space of feature subsets and hypotheses (Beniwal & Arora, 2012); (Saeys et al., 2007). A better view on the subset generation of the three methods can be obtained from Figure 2.4 (Saeys et al., 2007).

The FS in Figure 2.4a means Feature Selection, the blank oval represents the feature space and the outer arrow represents the selected subset of features resulting from the feature selection. In Figure 2.4b the hypothesis space refers to the search space, and since the search space is within the feature space and the classifier is used to evaluate each candidate subset of features, the two ovals and the classifier are nested.
Since embedded feature selection methods perform feature selection in the process of learning (in the process of classifier construction), the feature space and the search space are united as shown in Figure 2.4c.

(b) **Subset Evaluation**: This step is for evaluating the candidate subset using a certain evaluation criterion (Liu & Yu, 2005). According to Chen et al. (2006) and Liu & Yu (2005), two types of evaluation criteria can be identified based on their dependency on the learning algorithms that will be applied on the selected subset of features, independent criteria and dependent criteria. Usually, the independent criterion is used with filter feature selection methods. This criterion is used to evaluate the goodness of a feature or subset of features by utilizing the underlying characteristics of the training data without using any learning algorithm. Some examples of the independent criteria are: distance measures, information measures, dependency measures, and
consistency measures. The dependent criterion, on the other hand, is used with the wrapper feature selection methods and requires a predefined learning algorithm to be applied on each selected subset of features for evaluation using the estimated accuracy of that classifier (Blum & Langley, 1997). For this reason, the wrapper method is computationally expensive (Liu & Yu, 2005). The embedded methods differ in the way that feature selection is performed in the process of learning, i.e. without the need for a two-step induction process (Guyon, 2008).

(c) **Stopping Criteria:** This part determines when the feature selection process should stop. Some examples of stopping criteria are (Liu & Yu, 2005):

- The search completes (like filter methods).
- Minimum number of features or maximum number of iterations is reached (Filter).
- Consecutive addition (or deletion) of any feature does not produce a better subset (Wrapper, Embedded).
- An adequately good subset is selected where a subset is considered as a good subset if its classification error rate is less than a predetermined error rate threshold (Wrapper, Embedded).

(d) **Result Validation:** This is the stage of measuring the result of the classifier after stopping the feature selection. The most popular way of validating the feature selection results is to conduct a before-and-after experiment. This classification approach involves measuring the classification error rate for the classifier trained on the whole set and comparing it with the classifier trained on the selected subset of features (Liu & Yu, 2005);(Chen et al., 2006). However, we will observe later that sometimes feature selection can make the result worse instead of better. Filter, wrapper and embedded methods all validate the results in the same way.
2.5 Data Classification

When the data comprises different categories and there is a need to distinguish each example, a classifier is used to accomplish this task. A classifier task is to assign (predict) the class for each example. When a dataset contains only two classes, a binary classifier is used. When the dataset contains more than two classes, a multi-class classifier is used. Different classification algorithms have been proposed and explained in the data mining literature. In this thesis, we used the SVM classifier which is implemented in WEKA in LibSVM library. We used the SVM classifier inside the feature selection process of both our proposed method and RFE. In addition, we used the SVMs classifier to evaluate the results of the feature selection process. The SVM classifiers have been chosen in this work since SVM classifiers have been recently used in many applications due to their remarkable classification performance Chamasemani & Singh (2011). Researchers have been using SVMs in regression, binary classification and multi-class classification. In applications, it has been used in medical fields, network security, and other machine learning tasks and many real-world applications Goodfellow, Erhan, Carrier, Courville, Mirza, Hamner, Cukierski, Tang, Thaler, Lee, Zhou, Ramaiah, Feng, Li, Wang, Athanasakis, Shawe-Taylor, Milakov, Park, Ionescu, Popescu, Grozea, Bergstra, Xie, Romaszko, Xu, Chuang, & Bengio (2015). According to Goodfellow et al. (2015), the winner in the ICML 2013 in the facial expression challenge used the primal objective of a SVM as the loss function for training. The SVM classifier is trained according to Algorithm 1.

The focus of this thesis is on feature selection for SVM classifiers (for reasons we will explain shortly); the next sections will focus on: using SVM as a classifier, using a SVM with IDS, and using SVM with feature selection in IDS.

2.5.1 Using SVM as a Classifier

The Support Vector Machine (SVM) is a machine learning method that has been extensively used in solving machine learning problems due to its accurate classification results. This
classifier finds the maximum margin between training examples and the decision boundary (Boser, Guyon, & Vapnik, 1992). The decision boundary of a SVM classifier is illustrated in Figure 2.5.

![Diagram of SVM decision boundary and margin]

Researchers have been using SVMs in regression, classification and other machine learning tasks (Chang & Lin, 2011). The winner in ICML 2013 in the facial expression challenge used the primal objective of a SVM as the loss function for training (Goodfellow et al., 2015). In the following, more details about SVM classifiers are provided: Given training examples $x_i \in \mathbb{R}^n$, $i=1,..,l$ that belong to two classes, and a class label vector $y \in \mathbb{R}^l$ such that $y_i \in \{1, -1\}$, according to Boser et al. (1992), Support Vector Classification (C-SVC) can solve this optimization problem:

$$
\min_{w,b,\xi} \left( \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \right)
$$

subject to

$$
y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i
$$

$$
\xi_i \geq 0, i = 1, ..., l
$$
where $\mathcal{O}(x_i)$ projects $x_i$ into a higher dimensional space and $C > 0$ is used as a regularization parameter. Since the vector variable $w$ can be of high dimensionality, the dual problem is solved as follows:

$$\min_\alpha \left( \frac{1}{2} \alpha^T Q \alpha - e^T \alpha \right)$$ (2.5)

subject to $y^T \alpha = 0$, $0 \leq \alpha_i \leq C$, $i = 1, ..., l$

where $e = [1, ..., 1]^T$ is a vector of all ones, $Q$ is an $l \times l$ positive matrix, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$, and $K(x_i, x_j) \equiv \mathcal{O}(x_i)^T \mathcal{O}(x_j)$ is the kernel function.

Once the above problem is solved, the primal – dual relationship can be used to find the optimal $w$ which satisfies:

$$w = \sum_{i=1}^l y_i \alpha_i \mathcal{O}(x_i)$$ (2.6)

Then the decision function will be:

$$\text{sgn}(w^T \mathcal{O}(x) + b) = \text{sgn}(\sum_{i=1}^l y_i \alpha_i K(x_i, x) + b)$$ (2.7)

The SVM classifier is trained according to procedure 1 below:

1: procedure SVM$_\text{train}$(Training examples $\{x_1, x_2, ..., x_l\}$, class labels $\{y_1, y_2, ..., y_l\}$)

2: Minimize over $\alpha_k$:

$$J = \left( \frac{1}{2} \right) \sum_{hk} y_h y_k \alpha_h \alpha_k (x_h.x_k + \lambda \delta_{hk}) - \sum_k \alpha_k$$ (2.8)

subject to: $0 \leq \alpha_k \leq C$ and $\sum_k \alpha_k y_k = 0$

3: Outputs: Parameters $\alpha_k$

where $x_h.x_k$ denotes the scalar product, $y_k$ represents the class label as a binary value $+1$ or $-1$, the summations are applied on all training examples $x_k$ which are $n$ dimensional feature vectors each, $\delta_{hk}$ is the Kronecker symbol ($\delta_{hk} = 1$ if $h = k$ and 0 otherwise), and $\lambda$ and $C$ are positive constants. In general, the decision function of an input vector $x$ using SVMs is:

$$D(x) = w.x + b$$ (2.9)
with $w = \sum_k \alpha_k y_k x_k$ and $b = (y_k - w.x_k)$ where $w$ is the weight vector which consists of a linear combination of training examples. The training examples that have non-zero weights are support vectors (Guyon et al., 2002). In addition, using an SVM classifier in the feature selection is an excellent idea due to the great classification performance of this classifier and due to the ubiquitous use of SVMs in a vast array of application domains and problems (Chamasemani & Singh, 2011);(Goodfellow et al., 2015).

2.6 Classifier Performance Evaluation

After training a classifier, usually a matrix is produced which is called a confusion matrix. The confusion matrix is a square matrix ($n \times n$), where $n$ corresponds to the number of classes in the problem. The rows represent the actual class and the columns represent the prediction of the classifier for that class. And the cells represent the number of co-occurrences of a particular actual class and predicted class. Usually, the correct classified instances should be on the main diagonal of the confusion matrix. The confusion matrix for a binary classification problem is a $2 \times 2$ matrix as shown in Table 2.1.

<table>
<thead>
<tr>
<th>Class \ Recognized</th>
<th>As Negative</th>
<th>As Positive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Negative</td>
<td>TN</td>
<td>FP</td>
</tr>
<tr>
<td>Positive</td>
<td>FN</td>
<td>TP</td>
</tr>
</tbody>
</table>

where True Negative (TN) is the total number of correctly classified examples from the negative class, False Positive (FP) is the total number of examples from the negative class incorrectly classified as from the positive class, False Negative (FN) is the total number of examples from the positive class incorrectly classified as from the negative class, and True Positive (TP) is the total number of correctly classified examples from the positive class. Therefore, the diagonal cells of the confusion matrix represent the total number of
examples that are correctly classified by the given classifier from both classes, while the off diagonal cells represent the total number of examples that are incorrectly classified by that classifier. Usually, some metrics are extracted from the confusion matrix as performance measurement for the classifier, such as accuracy. The accuracy of a classifier can be defined as ratio between the total number of correctly classified examples to the total number of examples as shown in Equation 2.10:

\[
Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.10}
\]

Specificity or (True negative Rate) is another measurement which is the ratio between the total number of correctly classified examples from the negative class to the total number of examples of the negative class as shown in Equation 2.11:

\[
Specificity = \frac{TN}{TN + FP} \tag{2.11}
\]

### 2.7 Summary of the chapter

This chapter introduced feature selection in more detail and how it can be used with noisy data sets. In addition, the chapter presented the main feature selection types: filter, wrapper, and embedded. The chapter also discussed the main key-steps of any feature selection method. Then, the chapter moved into the data classification stage which comes after feature selection and focused on SVM classifier in particular. The chapter concluded by explaining how the performance of a classifier is evaluated and explained two evaluation metrics. This chapter is a connection between the feature selection and the proposed methodology that will be explained in the next chapter.
Chapter 3

Analysis of Recursive Feature Elimination RFE and a Proposal for a RFA Alternative

The purpose of this chapter is to present the proposed methodology of this thesis in detail. The chapter starts with the objectives of this work in Section 3.1. Next, an analysis and limitation of RFE are explained in Section 3.2. Moreover, a deep analysis on RFE is explained in Section 3.3. The thesis statement is explained in Section 3.4. Then, the methodology is explained in Section 3.5. The proposed solution, a new feature selection method, is explained in Section 3.6. The data sets used, selection of the classifier, and the method for splitting the data sets are explained in the Methods Section, 3.7. In addition, a detailed description of the proposed methodology is explained in Section 3.8.

In the previous chapter we reviewed filter, wrapper and embedded feature selection methods. Here we focus on embedded methods. These methods involve learning as wrapper methods do, which leads to obtaining more accurate results in most cases. The remarkable point about embedded methods is that they are faster than wrapper methods (Lal et al., 2006). We aim to use SVM with an embedded method in this study for the task of Intrusion...
Detection. That would ultimately lead to selecting the best features to detect the attacks and removing the irrelevant/redundant features as their existence in a data set would lead to misclassification of the observed connections.

3.1 Objectives

The objective of this research is to improve the final classification performance of a Support Vector Machine on interdependent features and on benchmark problems by using a new embedded feature selection method. The research will show that the new embedded feature selection method is superior to an existing embedded method (RFE) in terms of the classification performance of the resulting classifier on the target problems (problems with many features and few examples).

The classification performance will be evaluated based on two standard metrics: accuracy and $F$-measure in addition to two joint metrics: $\Delta\%Accuracy$ and $\Delta\%F-measure$ as that will be explained in Section 3.2.2.

3.2 Analysis and Limitation of RFE

The authors in Chen & Jeong (2007) made some observations regarding RFE which peaked our curiosity. These observations concerned RFE's tendency to remove some features that work in combination with other features. In Chen & Jeong (2007), the authors observed that RFE removed the feature 45 from the given data set as it was ranked 19th best feature. However, they discovered keeping this feature with other features led to an increase to the accuracy from 62% to 74%. The removed features were discovered that they do not help classification independently. However, those removed features may produce good performance when they are taken together. This observation about RFE sparked our curiosity to investigate and study this phenomenon before possibly proposing a solution.
In order to analyze the performance and investigate this limitation of RFE, we designed a small problem designed to require features that work in combinations, and used RFE to solve it. We checked the resilience of RFE to irrelevant noisy features and tested its ability to find interdependent features using a majority problem.

The majority problem consists of a group of binary features (bits), among which there are a group of relevant features. Those relevant bits, when taken together, they deterministically define the target output. Specifically, we sum the number of bits and determine whether or not it is larger than the number of relevant bits (i.e., The target output is “1” if a majority of bits are “1”). Additionally, we added a number of irrelevant features which contain random bits. What makes this problem interesting are two aspects of this problem. First, the relevant bits are meaningless independently. An individual bit’s state has almost no predictive power with respect to the majority. Only when combined do they provide information about the target output. Second, it is possible to add an arbitrary number of irrelevant features (distractors) to the problem. In the next section, we give the complete details of the data set we used to perform this investigation. In addition, we will describe our experimental design that we used in this analysis in detail in Section 3.2.2.

### 3.2.1 Description of the data set used in the analysis

As indicated above, to analyze the performance of RFE, we used a synthetic data set. The synthetic data set consisted of binary features that embed a majority problem with linear decision with extra irrelevant features and varying numbers of examples. As introduced in Section 3.2, the majority problem consists of 20 features which are taken collectively to generate the required target. The detailed description of the synthetic data sets is provided in Table 3.2.

We used four different sizes of data sets and 3-fold cross-validation. Each data set consists of 100 dimensions - only 20 dimensions are related to the prediction task and the remaining 80 are irrelevant features. The irrelevant features (noise) represent 4 times the relevant features. All the features have been generated as binary features (0, 1) and the distribution
of each feature has been made uniform (50% for 0s and 50% for 1s), however the sum of the bits represents a binomial distribution. Each example consisted of 100 randomly-generated binary values. By counting the number of ones in each example and drawing the histogram of the number of examples for number of ones (from 0-100), we made sure that the data set generation follows a binomial distribution. The problem design is illustrated in Table 3.1. The last column of the table represents the class label. It is important to indicate that from Table 3.1, the target can not be predicted by using only one relevant feature as each feature provides no information by itself only, rather, all of them have to be present together in order to predict the class label. The synthetic majority problem is illustrated as follows:

The number of the relevant features \( r \) is 20. \( S \) represents the summation of the relevant features' values. Since the features are binary features, the summation gives the number of the 1’s in the first 20 features. The target value of the problem depends on \( S \), if \( S \) was greater than half of the relevant features, then the target is 1, and 0 otherwise as shown in Equation 3.1.

\[
S = \sum_{i=0}^{r} x_i \quad (3.1)
\]

\[
Target = \begin{cases} 
1 & \text{if } S > (r/2) \\
0 & \text{otherwise}
\end{cases}
\]
Table 3.2: Description of the Synthetic Problem Data Sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>#features</th>
<th>#examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Majority1000</td>
<td>20 + 80 noise</td>
<td>1000</td>
</tr>
<tr>
<td>Majority800</td>
<td>20 + 80 noise</td>
<td>800</td>
</tr>
<tr>
<td>Majority600</td>
<td>20 + 80 noise</td>
<td>600</td>
</tr>
<tr>
<td>Majority300</td>
<td>20 + 80 noise</td>
<td>300</td>
</tr>
</tbody>
</table>

One of the benefits of feature selection is that it can help to prevent overfitting by reducing model parameters (Guyon & Elisseeff, 2003). Overfitting in turn is more common when there is a limited amount of training data versus situations in which there is more training data. For this reason, in this experiment we decided to experiment with different amounts of training data. Therefore, we varied the size of the data sets by generating four different data sets to make the problem more difficult for the classifier and for the feature selection algorithms. The generated data sets consisted of 1000, 800, 600, and 300 examples respectively. This variation helped us to challenge the feature selection algorithm. It also assisted us to check if it can work with different number of examples, while still produce a good subset of features without overfitting. The description of the synthetic problem data sets is shown in Table 3.2.

As stated before, each example consists of 100 binary features in which just 20 are related to the majority problem. It is important to note that the first 20 features are not represented any differently to the feature selection system or the classification system. Therefore, we could place these relevant features anywhere within the feature set and get the same result. We only place the relevant features in the beginning to allow for easy observation of how they are selected by the selection algorithm. The RFE technique works in a greedy fashion removing the worst feature at each step. Therefore, we want to check the implications that these features will have for RFE in terms of generalization ability and classification accuracy. The advantage of using this synthetic problem is that it is a controlled data set since the relevant features are already known to us. The remaining features (the irrelevant...
ones) stand as noisy features here. We want to see if the decision of feature selection is
distracted by the additional irrelevant features. According to Huang et al. (2014), these
noisy features may have a negative impact on the classifier as they make it difficult for
the classifier to identify informative features, and increasingly so as the number of example
patterns is reduced. Therefore, we can evaluate whether the given feature selection methods
are able to detect the relevant features in this set of problems, or not.

3.2.2 Experimental design

Our experiment involved measuring two kinds of metrics (standard and joint metrics) before
and after feature selection in order to observe the impact of applying the feature selection
on a given problem (data set). Interestingly, the classification performance on some of the
problems does not always improve after applying feature selection. Therefore, we measured
the accuracy of the classifier and the $F$-measure before and after feature selection. The
accuracy of a classifier is calculated using the following formula:

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

(3.2)

where $TP$, $TN$, $FP$, and $FN$ are True Positive, True negative, False Positive, and False
Negative values respectively. The classifiers’ accuracy is the percentage of the correctly
classified examples to the total number of examples. $F$-measure on the other hand, is a
harmonic mean of precision and recall which can be calculated using the following formula:

$$F - \text{measure} = \frac{2 \times TP}{2 \times TP + FP + FN}$$

(3.3)

In addition, we used two joint metrics to evaluate the feature selection performance by
considering the difference in the accuracy and $F$-measure before and after feature selection.
These metrics can give us an indication about the impact of applying feature selection, if
it was good (improving the performance) or bad (decreasing the performance). The first
joint metric is to check the accuracy change and can be calculated as follows:

$$\Delta\%\text{Accuracy} = \left(\frac{\text{Accuracy}_2 - \text{Accuracy}_1}{\text{Accuracy}_1}\right) \times 100\%$$

(3.4)
where $Accuracy_1$ and $Accuracy_2$ are the accuracy values before and after applying feature selection, respectively. The value of this metric could be positive (if the feature selection is improving the accuracy), or could be negative (if the feature selection is deteriorating the accuracy). In addition, this metric’s value could be more than 100 sometimes if the improvement was sufficiently effective (i.e., more than doubles the accuracy).

The second joint metric is to check the $F$-measure change and can be calculated as follows (in fact it is calculated the same way as 3.4):

$$\Delta\%F\text{-measure} = \frac{(F\text{-measure}_2 - F\text{-measure}_1)}{F\text{-measure}_1} \times 100\%$$  \hspace{1cm} (3.5)

where $F\text{-measure}_1$ and $F\text{-measure}_2$ are the $F$-measure values before and after applying feature selection, respectively. Same as above, this metric could be positive or negative or could be more than 100.

### 3.2.3 Preliminary Results of RFE analysis

In Table 3.3, below, we show the RFE performance using the four mentioned metrics in columns 4-7. The reported performance here is the maximum obtained value for accuracy and $F$-measure across each iteration of the RFE algorithm. Since we repeat the experiment 30 times and in each iteration we have 100 features to remove and report the performance metrics. Therefore, in each iteration, a vector of 100 values is generated. Consequently, 30 vectors are produced with 100 values each. This procedure is done for both accuracy and $F$-measure. To report the maximum, we averaged all the iterations to obtain one vector for accuracy and one vector for $F$-measure, then we report the maximum from the resulting vectors according to Equation 3.6.

$$\max_{ACC} = \max(\text{avg}_i(ACC_{ij}))$$ \hspace{1cm} (3.6)

where $i$ is the number of features, $j$ is the experiment iteration, and ACC is the accuracy metric.
Thus, RFE will show superior performance to the full-featured SVM if, and only if, removing features in the order specified by RFE improves the SVM performance. In addition, we show the performance of the SVM classifier without feature selection in columns 2 and 3. What we would expect from applying RFE is that it would eliminate the bad features and that would increase the performance in terms of accuracy and $F$-measure. We can see from Table 3.3 that the accuracy of RFE is comparable to (or sometimes worse than) the accuracy obtained without feature selection as in the Majority1000 data set. The same is true for the $F$-measure. With the other data sets, RFE showed some improvement on all the metrics especially with the Majority300 data set.

The RFE showed a little good performance (and sometimes poor performance) as it was not able to improve the performance, and this is due to its inability to detect the interdependent features (i.e., the 20 important features of the problem), and its deficiency in the resilience to the irrelevant features.

Our speculation for this poor performance of RFE, is that due to the fact that the performance decreases for the regular SVM as data set size increases (probably due to over generalization), consequently the RFE does not improve performance.

Table 3.3: Performance Metrics before and after Applying RFE on the Synthetic Majority Data Sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>Without feature selection</th>
<th>RFE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACC</td>
<td>$F$-measure</td>
</tr>
<tr>
<td>Majority1000</td>
<td>0.75</td>
<td>0.73</td>
</tr>
<tr>
<td>Majority800</td>
<td>0.65</td>
<td>0.56</td>
</tr>
<tr>
<td>Majority600</td>
<td>0.60</td>
<td>0.47</td>
</tr>
<tr>
<td>Majority300</td>
<td>0.58</td>
<td>0.43</td>
</tr>
</tbody>
</table>
3.3 Deep analysis on RFE

In order to analyze the results of the majority problem for RFE in more depth, we conducted a more thorough analysis of the results. The main objective of this additional analysis is that we wanted to know when the RFE starts eliminating one of the entangled features (relevant features). For RFE, the later it breaks the entangled features, the better. Once RFE eliminates one of the entangled features, it will make the problem more difficult to solve. At each iteration, we report how many times one of the relevant features has been eliminated. Since we have 100 features, then we have 100 iterations. In addition, since we repeat the experiment 30 times and we do 3-folds cross-validation at each iteration, then the maximum count that can be reported at each iteration is 90. This analysis has been done for the four Majority problem data sets that we have. This analysis involved extracting three graphs for each data set as follows:

(a) The number of good features eliminated compared to random selection

We compared the performance of RFE against a random feature selection process (which incrementally selects one feature at random to eliminate). That is, we eliminate features one by one, then we calculate the chance to eliminate the important features at each iteration $i$. Since we have already counted the number of good features eliminated $Count_i$ by RFE, then we can calculate the remaining number of good features $R_i$ after each iteration from: $R_i = R_{i-1} - Count_i / 90$, where the initial value of $R$ is 20. The random selection works by subtracting $(1/5)$ from the initial value of good features (which are 20, and that corresponds to $1/5$ of the total number of features–100), at each iteration. Therefore, plotting the random selection will be a straight line that starts from the top (20 good features) and ends at 0 (when there are no more features to eliminate). This graph gives us an indication about the performance of RFE if it eliminates good features worse than random selection (worse than chance) or not. If it performs better than random, then the curve should be over the line of the random
selection, but if RFE performs worse than random, then the curve should be under the line of the random selection.

(b) **The number of bad features remaining**

This graph in fact represents the complement of the first graph. We also compared the performance of RFE in terms of keeping bad features at each iteration \( i \) against the random selection. In this case, we have 80 bad features out of 100 features and hence the chance to eliminate a bad feature is \( 4/5 \). The number of remaining bad features \( B_i \) is calculated as: \( B_i = B_{i-1} - (1 - \text{Count}_i/90) \). Again, the random selection will be a straight line starts from the top (80 bad features) and ends at the bottom when there are no more features to eliminate. This graph gives us an indication about the performance of RFE if it keeps bad features worse than random selection (worse than chance) or not. If it performs better than random, then the curve should be under the line of the random selection and vice verse.

(c) **The percentage of good features remaining**

The third graph that we extracted here represents the percentage of the good features remaining at each iteration compared to random selection. The probability to eliminate a good feature is 20% for the whole time since the total number of good features is 20 out of 100 features. Therefore, the random selection of good features will be a straight line on the 20% from the beginning till the end. However, the percentage of good features remaining \( G_i \) will be calculated as: \( G_i = R_i / (R_i + B_i) \), where \( R_i \) is the number remaining good features and \( B_i \) is the number of remaining bad features respectively. The graph gives us an indication of the RFE performance in terms of keeping good features at each iteration and if it works better than chance in this context or not.

The analysis of the Majority1000 data set is shown in Figure 3.1. With the increase in the data set size, RFE is slightly showing better performance in terms of keeping the relevant features as shown in Figure 3.1a. For the number of bad features remaining, the pattern is
Figure 3.1: Performance of RFE feature selection against random selection for Majority1000 data set (a) The number of good features eliminated (b) The number of bad features remaining (c) The percentage of good features remaining

The analysis of Majority800 data set is shown in Figure 3.2. The pattern repeats itself for the number good features remaining and the number bad features remaining as shown in Figures 3.2a and 3.2b respectively. The percentage of the good features remaining has increased after the last 10% of the features to 63% compared to 56% in Majority600 data set and 47% in Majority300 data set. However, in general, the percentage of good features remaining is still below the random elimination as shown in Figure 3.2c.
Figure 3.2: Performance of RFE feature selection against random selection for Majority800 data set (a) The number of good features eliminated (b) The number of bad features remaining (c) The percentage of good features remaining

The analysis of Majority600 data set is shown in Figure 3.3. Again, the RFE is showing poor performance in terms of keeping good features when it removes the good features worse than chance until the last 10% of the features when it slightly overcomes the random elimination as shown in Figure 3.3a. The second graph of the bad features elimination shows poor performance worse than chance as shown in Figure 3.3b. For the percentage of the good features remaining, RFE is not performing better than chance until the last 10% of the features when the red curve goes up over the blue line as shown in Figure 3.3c.

In Figure 3.4 we are showing the RFE performance for the Majority300 data set. We can notice that RFE is showing poor performance in terms of keeping the good features as can be observed from the red curve which descends below the random elimination which is
(a) The number of good features eliminated
(b) The number of bad features remaining
(c) The percentage of good features remaining

Figure 3.3: Performance of RFE feature selection against random selection for Majority600 data set (a) The number of good features eliminated (b) The number of bad features remaining (c) The percentage of good features remaining

represented by the blue line as shown in Figure 3.4a. For the bad features remaining, RFE is also showing bad performance (which is represented by the red curve) when it keeps bad features worse than random elimination, which is represented by the blue line as shown in Figure 3.4b. The pattern is repeated with the percentage of good features remaining when RFE removes good features worse than chance until the last 10% of the iterations when recovers from its earlier poor performance by eliminating enough poor features for it to keep up with chance as shown in Figure 3.4c.

From this deep analysis, we diagnosed the limitation of RFE experimentally regarding its inability to keep the interdependent features. The method showed its limitation also
in removing relevant features instead of removing irrelevant ones as we noticed from the figures of the deep analysis of all the synthetic data sets.

### 3.4 Thesis Statement

The RFE method has been shown to be a very powerful feature selection method. However, as indicated by previous studies (Chen & Jeong, 2007) and, now, from our own experiments, we found that this method suffers from a significant limitation. This limitation is represented in its inability to avoid noisy features and to detect feature interdependency, which are both common problems of feature selection that need to be addressed.
Some researchers have pointed out that there might be two weak features that are independently useless, but they can produce high performance when used together (Guyon & Elisseeff, 2003). The effectiveness of some feature selection methods is affected by the existence of irrelevant features or with their inability to find interdependent features especially in high dimensional data sets.

Based on these observations, we now:

1. Develop a solution to the limitations of RFE.

2. Evaluate the solution with respect to both generated and real-world data sets.

### 3.5 Methodology

The overall model of our approach is depicted in Figure 3.5. The input data represent data sets training records including their targets. These records have been collected and organized as a standard benchmark data set. In our work, we will use several benchmark data sets that we chose according to a specific criterion that will be explained in Section 3.7.1. If the data set does not have a pre-defined testing set, the data set will be split into training and testing sets; otherwise the testing phase will work on the pre-defined testing data. The training records are then entered into the feature selection module to select the best subset of features according to the employed feature selection method.
Next, the training phase works on the selected features by training a classifier on the training records (with the selected features only). The employed classifier will produce learning parameters which will be saved to be used later in the testing phase. The testing phase tests the trained classifier (which results from the training phase) on a separate test
data set using the same selected features to extract the evaluation metrics as shown in Figure 3.5.

The goal of this study is to select the best subset of features to increase the performance of the classification process. In this work, a novel embedded feature selection method is proposed. The proposed method uses a new technique called Recursive Feature Addition (RFA) that was developed specifically for this research and utilizes machine learning based on Support Vector Machines (SVMs).

### 3.5.1 Recursive Feature Addition (RFA)

The novel feature of RFA is that it is a forward-feature selection method, as opposed to previous work which worked backwards. RFA starts from an empty feature set and keeps adding one feature at a time until a stopping criterion is met. The method utilizes a SVM classifier as a core classifier to rank the features, and based on that ranking, it adds the feature that attains the best ranking criterion. The general model of this embedded forward-feature selection method is illustrated in Figure 3.6.

In the example shown in Figure 3.6, the problem has four features only. Each feature is represented as a gray code value (0 or 1) Gray (1953), where 0 means absence of this feature and 1 means presence. When the feature is selected, its location will be 1 otherwise will be 0. The RFA method starts with an empty feature set (0 0 0 0). In the shown example, the feature 2 was selected as the most relevant one among the other features therefore it was selected first. The algorithm continues until it ranks all the features according to the ranking coefficient.
The bold arrow represents the route that the method follows, while the dotted arrows represent all the possible cases from the current case. The final ranking of this example was (2, 4, 3, 1) respectively.

3.6 The proposed feature selection method and approach

In this section we explain our proposed feature selection method in detail. We propose this technique to overcome the observed limitations of RFE that are explained in the previous sections. The algorithm of the proposed method also will be explained in the next section.

3.6.1 Recursive Feature Addition (RFA) with SVMs

RFA is a new feature selection from the embedded feature selection family. It works in a forward fashion and employs the SVMs classifier. The method initializes an empty set of
features to be used for the selected features by recursively adding one feature at a time to that set according to the calculated ranking coefficients of the remaining features.

Since SVMs are an effective technique and to draw a fair comparison, an SVM classifier has been utilized in the proposed feature selection algorithm here. The SVMs will calculate the weights, \( w_i \), of the decision function \( D(x) \) for only the support vectors (which basically represent a small subset of the training examples) as shown in Figure 2.5.

The support vectors are the training examples that are closest to the decision boundary and provide the maximum separation between the classes. Ranking the features in the (RFA) depends on the weight magnitude as ranking coefficients. Our algorithm trains the SVM classifier according to its well known algorithm. Our algorithm performs by adding one feature upon the maximum change in the cost function. According to Boser et al. (1992), the cost function of the SVMs that needs to be minimized is:

\[
J = (1/2)\alpha^T H \alpha - \alpha 1
\]  

(3.7)

where \( H \) is the matrix that can be calculated as:

\[
H = y_h y_k K(x_h, x_k)
\]  

(3.8)

and 1 is an \( n \) dimensional vector of ones only, \( x_h \) and \( x_k \) are training examples. In Equation 3.8, \( K \) is a kernel function used for measuring the similarity between \( x_h \) and \( x_k \) examples, \( h = 1..N, k = 1..N \), \( N \) is the number of features, and \( y \) is a vector of class labels. The RBF kernel function has been used in this algorithm which can be calculated as:

\[
K(x_h, x_k) = \exp(-\gamma ||x_h - x_k||^2)
\]  

(3.9)

where \( \gamma \) is a constant and usually chosen as \( \frac{1}{\text{number of features}} \). In order to calculate the change in the cost function resulting from adding one feature \( i \), the \( H \) matrix needs to be re-calculated and, therefore, it is called \( H(+i) \) where the notation \((+i)\) corresponds to adding feature \( i \). This involves calculating \( K(x_h(+i), x_k(+i)) \). The final ranking coefficient \( DJ \) is calculated from:

\[
DJ = (1/2)\alpha^T H \alpha - (1/2)\alpha^T H(+i) \alpha
\]  

(3.10)
When the algorithm starts, since there is no feature selected yet, the ranking coefficient $DJ$ at this point is simply reduced to the second term. The feature corresponding to the maximum difference $DJ(i)$ is added to the ranked feature list. The algorithm is executed iteratively to perform Recursive Feature Addition (RFA). The result of the algorithm will be a ranked list of features from the most important to the least important. The algorithm of the proposed method is illustrated in Algorithm 2.

**Algorithm 2** Recursive Feature Addition (RFA)

1: **Input:** Data set $D$, original features set $F$, no. of features $N$
2: Set an empty ranked feature set $S$
3: $i \leftarrow 1$
4: **while** $i \leq N$ **do**
5: Train SVMs classifier
6: Get the resulted $\alpha$ vector, $\alpha \leftarrow$ alpha vector
7: Get the resulted support vectors $X \leftarrow$ support vectors
8: Calculate the ranking coefficients for the remaining features from 3.10
9: Add the feature $f$ that has the maximum ranking coefficient to the set $S$, $S \leftarrow S + f$
10: Remove feature $f$ from original features set $F$, $F \leftarrow F - f$
11: $i \leftarrow i + 1$.
12: **endwhile**
13: **Output:** Ranked features set $S$.

One of our objectives in this technique is to find combinations of features, that although do not work especially well independently, they work very well when part of the same set of selected features. In this sense, we are looking for interdependent features in addition to independent features. To achieve this, our approach involves ranking one feature at a time by starting from an empty set of features and keeps adding features until ranking all the features. At each iteration, the algorithm finds the best feature in terms of how well it works with the current feature(s).
The RFA finds the feature that makes the maximum change in the ranking coefficient. A small example of using embedded forward feature selection is shown in Figure 3.6.

Therefore, the feature that corresponds to the maximum $DJ$ is added to the selected features. In other words, the feature that causes the maximum change in the cost function is the most important feature that should be selected. This technique is performed recursively and called Recursive Feature Addition (RFA). We wonder how RFA will perform with interdependent features as compared to RFE. In addition, we wonder if RFA is affected by the noisy features in selecting features as we will see in the majority problem.

### 3.6.2 Training the Classifier During the Feature Selection

As stated in Chapter 2, the embedded feature selection carries out feature selection in the process of learning. This learning is usually accomplished by using one of the machine learning algorithms. In the proposed approach (RFA), the SVM classifier is used during the feature selection process. To train the SVM classifier, the LibSVM library is used which was implemented by Chang & Lin (2011). The LibSVM has become a de facto standard library for training SVMs classifiers. It can be used in Java code and with the Weka GUI directly. The training of the SVMs using LibSVM produces the alpha vector and the support vectors. These results are used in calculating the ranking coefficients as in Equation 3.10.

Since the proposed feature selection method is a forward-selection, the SVM training has been applied after adding each new feature in order to calculate the ranking coefficients for the remaining features. The ranking coefficients are calculated according to Equation 3.10. The result after finishing this step is a ranked list of the features of the given data set from the most important to the least important one.

For the sake of consistency, we used the same kernel function and same kernel settings in both RFA and RFE in calculating the ranking coefficients. The stopping criterion for both methods was the number of features (i.e., the algorithm keeps adding or removing until all the features are ranked).
3.7 Methods

In this section, first we will explain the selected the data sets and then we will explain selection of classifier.

3.7.1 Selection of Data sets

We tested our proposed method on three categories of data sets, synthetic, small-size, and large-size real-world data sets. Since we talked about the synthetic data sets in previous sections, in the next sections we are going to explain the small-size, and real world data sets.

Small-size data sets

Table 3.4 below shows the benchmark data set details including the number of features, number of examples, class distribution and source or data set repository. The data sets used in the experiment vary in terms of the number of features and number of examples, but all have fewer features than examples.

Table 3.4: Description of the small-size data sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>#features</th>
<th>#examples</th>
<th>Class distribution</th>
<th>Source (Repository)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diabetes</td>
<td>9</td>
<td>768</td>
<td>500+268</td>
<td>UCI</td>
</tr>
<tr>
<td>Spect</td>
<td>23</td>
<td>267</td>
<td>212+55</td>
<td>UCI</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>35</td>
<td>351</td>
<td>126+225</td>
<td>UCI</td>
</tr>
<tr>
<td>QSAR</td>
<td>42</td>
<td>1055</td>
<td>356+699</td>
<td>UCI</td>
</tr>
<tr>
<td>Hill_valley</td>
<td>101</td>
<td>1212</td>
<td>600+612</td>
<td>UCI</td>
</tr>
</tbody>
</table>

Real-world data sets

For the real-world data sets, also we chose a variety of data sets in terms of number of examples and number of features. The data sets have been chosen according to specific
criteria. We are looking for data sets that exhibit high dimensionality (large number of features), and relatively few examples, to check the resilience of the two methods against overfitting. Since Cancer detection is an interesting topic these days as it assists in distinguishing between cancer tissues from their gene expression from normal tissues or other types of cancer tissues Guyon & Elisseff (2003), we explored multiple cancer data sets repositories as follows:

1. The Cancer Program Data set repository from the Broad institute (Bolon-Canedo et al., 2014). This repository contains many data sets for bioinformatics and computational biology research. We chose four data sets from this repository matching the criteria that we specified.

2. The LibSVM repository is another recognized repository which contains many data sets for classification, regression, and multi-label problems (Chandrashekar & Sahin, 2014). We found two data sets matching our criteria.

3. The Bioinformatics Research group which we found in Bolon-Canedo et al. (2014), we chose one data set matching our specified criteria. This repository also has several benchmark bioinformatics data sets in Weka format ready for machine learning research.

4. The Kent Ridge Bio-medical Data set repository is an online repository of high-dimensional biomedical data sets including protein profiling data, gene expression, and data genomic sequence for classification tasks (Bolon-Canedo et al., 2014). We chose two lung cancer data sets from this repository matching our criteria.

5. The Gene Expression Model Selector (GEMS) is another repository we found in Bolon-Canedo et al. (2014) for different kinds of benchmark cancer data sets. We chose only one data set that we found matching our criteria.

6. The UCI repository which is a well-known machine learning repository that contains 335 data sets for different machine learning tasks: classification, regression or clustering. We chose only one data set from this repository matching our criteria.
Therefore, in total, we chose 11 benchmark data sets most of which are microarray data sets except the Double world email bodies data set. All the details of the real-world data sets that we used in this thesis are shown in Table 3.5. Note that the number of features in most of these data sets is two orders of magnitude larger than the number of examples.

<table>
<thead>
<tr>
<th>Data set</th>
<th>#features</th>
<th>#examples</th>
<th>Class distribution</th>
<th>Source (Repository)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast Cancer</td>
<td>7129</td>
<td>44</td>
<td>23+21</td>
<td>Cancer Program Data Sets</td>
</tr>
<tr>
<td>Double world email bodies</td>
<td>4702</td>
<td>64</td>
<td>35+29</td>
<td>UCI repository</td>
</tr>
<tr>
<td>Colon cancer</td>
<td>2000</td>
<td>62</td>
<td>22+40</td>
<td>LibSVM repository</td>
</tr>
<tr>
<td>CNS</td>
<td>7129</td>
<td>60</td>
<td>21+39</td>
<td>Cancer Program Data Sets</td>
</tr>
<tr>
<td>Pomeroy</td>
<td>7129</td>
<td>60</td>
<td>39+21</td>
<td>Cancer Program Data Sets</td>
</tr>
<tr>
<td>Leukemia</td>
<td>7129</td>
<td>38</td>
<td>27+11</td>
<td>LibSVM repository</td>
</tr>
<tr>
<td>Lymph (Harvard)</td>
<td>7129</td>
<td>77</td>
<td>58+19</td>
<td>Bioinformatics Research Group</td>
</tr>
<tr>
<td>DLBCL (smaller)</td>
<td>5469</td>
<td>77</td>
<td>58+19</td>
<td>Gene Expression Model Selector</td>
</tr>
<tr>
<td>DLBCL</td>
<td>7129</td>
<td>58</td>
<td>32+26</td>
<td>Cancer Program Data Sets</td>
</tr>
<tr>
<td>Lung_Michigan</td>
<td>7129</td>
<td>96</td>
<td>86+10</td>
<td>Kent Ridge Biomedical Data sets</td>
</tr>
<tr>
<td>Lung_Ontario</td>
<td>4702</td>
<td>39</td>
<td>24+15</td>
<td>Kent Ridge Biomedical Data sets</td>
</tr>
</tbody>
</table>

### 3.7.2 Choosing a Classifier

For the classifier, we used the SVM classifier which was explained in Chapter 2.
3.7.3 Data set Splitting

The data set splitting in this study is performed as three-fold cross-validation. The splitting involved taking two thirds of the data set for a training phase and the remaining third of the data set is used for a testing phase and repeating this process for 3 times as the model used in Draminski et al. (2008). The splitting procedure has been applied on data sets which do not have separate test sets. However, the proposed method has been applied on other data sets which have already separate test sets. For the data sets that have already separate test sets, no cross-validation has been performed and the testing phase used the testing set to evaluate the trained classifier. In all the cases the testing step must employ unseen data to the classifier to measure its performance as shown in Figure 3.7 below:

\[
\begin{align*}
\text{Training set} & \rightarrow \text{Feature Selection} & \rightarrow \text{Ranked features} \\
\text{Testing set} & \rightarrow \text{Classifier} & \rightarrow \text{Classification performance}
\end{align*}
\]

Figure 3.7: A block diagram for training and testing steps of the proposed model

3.7.4 Performance Evaluation

After obtaining the ranked features’ list, the model performance needs to be evaluated to check the performance of the proposed feature selection method. The accuracy of the classifier has been calculated after adding each new feature to the ranked features list.
As a result, an $N$-dimensional vector is generated, and each $i$–th element in the vector corresponds to the classifier’s accuracy after adding the $i$–th feature from the ranked list. The same is true for the $F$-measure of the classifier. In both cases, we report the maximum obtained value from the $N$ values.

Testing the classifier has been performed on the third fold of the input data (which is unseen by the classifier) as shown in Figure 3.7. The three-fold cross-validation requires repeating the whole process three times in order to test the classifier on an unseen portion of the data each time. Since three-fold cross-validation is used in the testing as stated in Section 3.7.3, three accuracy vectors and three $F$-measure vectors (one from each fold) will result. These vectors are averaged to obtain the final accuracy and $F$-measure vectors. For the RFE, we measured the same metrics as we did with RFA, except we measured them after eliminating a feature instead of adding.

### 3.8 Methodology Description

In this section, we will explain the measurements that we used in our experiments as well as the detailed steps of our methodology.

#### 3.8.1 Measurements

The two metrics have been measured before and after adding a feature (in RFA), and after removing a feature (in RFE), and this was done iteratively over all the features. We measured all the metrics that we stated in Section 3.2.2 for both synthetic and real-world data sets to study the behaviour of feature selection regarding these metrics. In particular, we are aiming at investigating the conditions under which each method (RFA and RFE) achieves higher results in terms of the accuracy and $F$-measure and on which group of data sets.
3.8.2  Methodology Steps

We designed a methodology to study the behaviour (in terms of accuracy and $F$-measure) of our method (RFA) with real-world high dimensional data sets as compared to RFE. Our methodology consisted of the following steps:

1. Ranking the data set features using CFS
2. Taking a subset of the ranked features
3. Rank the features from (2) using RFE
4. Rank the features from (2) using RFA
5. Measure the performance of RFE
6. Measure the performance of RFA

In the next sections, we explain each part of the methodology in detail.

**Ranking the data set features using CFS**

Our goal in this step is to create a set of benchmark problems that would allow us to more quickly compare the two methods on many data sets with multiple folds. To do this we needed to create data sets with smaller numbers of features, but including both useful and less useful features. We want to make sure that the reduced-features are still informative and yet, still contained distracting features as well, as will be described in (b). To this end, we used a very fast feature selection method to pre-rank the features which is called Correlation-based Feature Selection (CFS). By using this method, we were able to rank the features in order to perform controlled experiments by manipulating relevant and irrelevant features. The CFS method was explained in more detail in Section 2.2.1. The result of this step is a ranked features list, which will be used in the next steps.
Taking a subset of the ranked features

After obtaining the ranked features list from the CFS method, a smaller subset of this ranked list is extracted. This subset of features consists of one portion of the top ranked features and nine portions of lowest ranked features. We specified the portion size as 35 features; therefore the subset size is 350 features in total. The wisdom behind applying this strategy is twofold. First, it will decrease the computation time required to rank the features by RFA and RFE considerably which allowed us to expedite our testing and conduct more tests. Second, by taking a subset of the features according to this specification, we will know if feature selection will work or not (i.e., improve the performance) for each this data set without providing the entire set of features. In addition, we include 9 times as many bad features compared to good features to challenge the feature selection algorithm and check if it can extract the good features among the much larger number of bad features.

Rank the features from (2) using RFE

Next, we take the 350 features that resulted from Step (2), and rank them using the RFE method proposed by Guyon et al. (2002). Since most of the data sets that we are using have a few number of examples, we used 3-fold cross validation for the evaluation. Therefore, each data set is divided into three parts, 2 parts are used for training and one for testing. The process is repeated three times so that each part is used for testing exactly once.

Rank the features from (2) using RFA

As with RFE, we use our method, RFA, described in Section 3.6.1 to rank the features that result from step (2). We followed the same testing strategy that we used with RFE for the testing process of RFA.
Measure the performance of RFE

After ranking the features using RFE, the ranked list is evaluated using the SVM classifier. To evaluate the performance, we measured four metrics (accuracy, $F$-measure, $\Delta\%$ accuracy, and $\Delta\%$ $F$-measure) after removing each feature from the ranked list that results from step (3).

Measure the performance of RFA

As with RFE, we evaluated the performance of RFA using the four aforementioned metrics. We calculated the metrics after adding each feature from the ranked list that results from step (4). In both cases (RFA and RFE), we obtain 4 vectors, each of which consists of 350 values for each metric. We repeated the above steps 30 times for each data set and evaluated the results using 3-fold cross validation at each iteration. In order to give a meaning to the experiment repetition, we randomly shuffled the data so that the training and testing data sets will be different at each iteration. At the end, each performance metric is averaged over the 30 iterations. We followed the above methodology with the real-world data sets. However, with the synthetic data sets, we started from step 2 as we do not have to pre-rank the features using CFS method since synthetic data sets do not have as many features as the real-world data sets.

3.9 Conclusion and Summary

In this chapter we analyzed the RFE feature selection method and diagnosed its limitation. We discovered that RFE has a limitation related to discovering interdependent features. We showed that limitation by applying RFE on a synthetic data set that embeds some interdependent binary features with four times noisy features. In addition, we studied the behaviour of RFE in removing good and bad features, and we discovered that RFE lacks in keeping good (informative) features during the elimination process. Therefore, we proposed
our method, RFA, to tackle the problems of RFE. We explained our method, RFA, and the employed methodology. In order to test the proposed method, we chose some real-world data sets (according to a predefined criteria) in addition to the synthetic data sets to apply on.
Chapter 4

Experimental work and Results

In this chapter we study the advantage of applying feature selection on different data sets. First, we conduct a deep analysis on RFA to observe the behaviour of RFA with the interdependent features in Section 4.1. In addition, we establish a baseline for our experiments by measuring the classifier performance without feature selection. This is performed by simply feeding each data set to the SVM classifier and measuring the performance metrics. After obtaining the baseline, we show the results of applying feature selection on the three groups of data sets that we used for the experiments (i.e., synthetic, real-world data sets, and small data sets). In Section 4.2, we compare the results of applying RFA and RFE on the synthetic majority data sets. We also compare the results of applying RFA and RFE on the small real-world data sets in Section 4.3. In Section 4.4, the results of applying both RFA and RFE are compared. The statistical analysis of the results on synthetic and real-world data sets using two metrics: accuracy and $F$-measure is presented in Section 4.5. Lastly, the summary of the chapter is presented in Section 4.6.

4.1 Results of deep analysis on RFA

In order to analyze the results of the majority problem for RFA, we conducted (the same deep analysis that we conducted on RFE) on the RFA results. The main objective of this
additional analysis is that we wanted to know when RFA finds the twenty entangled features in terms of number of steps. For RFA, because it incrementally adds features, the earlier it finds these good features, the better.

At each iteration, we report how many times one of the relevant features has been selected. Since we have 100 features, then we have 100 iterations. In addition, since we repeat the experiment 30 times and we do 3-folds cross-validation at each iteration, then the maximum count that can be reported at each iteration is 90.

We generated the same three figures that we generated with RFE, with the difference that we generate them from RFA as follows:

(a) **The number of good features selected compared to random selection**

We compared the performance of RFA against a random feature selection process (which incrementally selects one feature at random). That is, if we select features one by one, then we calculate the chance to select the important features at each iteration $i$. Since we have already counted the number of good features selected $Count_i$ by RFA, then we can calculate the number of good features selected $R_i$ after each iteration from: $R_i = R_{i-1} + (Count_i / 90)$, where the initial value of $R$ is 0. The random selection works by adding $1/5$ (the ratio of good features to the entire number of features) to the initial value of good features (which is 0, since we start with an empty set of features), at each iteration. Therefore, plotting the random selection will be a straight line that starts from the bottom (0 good features) and ends at 20 (when there are no more features to add). This graph gives us an indication about the performance of RFA – if it adds good features better than random selection (better than chance) or not. If it performs better than random, then the curve should be over the line of the random selection and vice verse.

(b) **The number of bad features selected**

We also compared the performance of RFA in terms of selecting bad features at each iteration $i$ against the random selection. In this case, we have 80 bad features out
of 100 features and hence the chance to select a bad feature is 4/5. The number of selected bad features, $B_i$, is calculated as: $B_i = B_{i-1} + (1 - \text{Count}_i/90)$. Again, the random selection will be a straight line that starts from the bottom (0 bad features) and ends at the top when there are no more features to add. This graph gives us an indication about the performance of RFA – if it selects bad features worse than random selection (worse than chance) or not. If it performs better than random, then the curve should be under the line of the random selection and vice verse.

(c) The percentage of good features selected

The third graph that we generated here represents the percentage of the good features selected at each iteration compared to random selection. The probability to select a good feature is 20% for the whole time since the total number of good features is 20 out of 100 features. Therefore, the random selection of good features will be a straight line on the 20% from the beginning until the end. However, the percentage of selected good features will be calculated as: $G_i = R_i/(R_i + B_i)$. The graph gives us an indication of the RFA performance in terms of selecting good features at each iteration and if it works better than chance in this context or not.

This analysis has been done for the four Majority problem data sets as follows: First we did the analysis for the Majority300 data set as shown in Figure 4.1.
(a) The number of good features selected compared to random
(b) The number of bad features selected compared to random
(c) The percentage of good features selected

Figure 4.1: Performance of RFA feature selection against random selection for the Majority300 data set (a) The number of good features selected (b) The number of bad features selected (c) The percentage of good features selected.

Figure 4.2 shows the RFA performance for Majority600 data set.
Figure 4.2: Performance of RFA feature selection against random selection for the Majority600 data set (a) The number of good features selected compared to random (b) The number of bad features selected compared to random (c) The percentage of good features selected.

Figure 4.3 shows the RFA performance for Majority800 data set.
Figure 4.3: Performance of RFA feature selection against random selection for the Majority800 data set (a) The number of good features selected compared to random (b) The number of bad features selected compared to random (c) The percentage of good features selected.

Figure 4.4 shows the RFA performance for Majority1000 data set.
As shown from the Figures 4.1, 4.2, 4.3 and 4.4, our RFA method is consistently able to find the good feature earlier than random as shown in (a) in all the figures where the red line rises high above the blue line early in the feature selection process. In addition, RFA is showing better performance than random in selecting fewer bad features also consistently as shown in (b) in all the figures where the red line remains below the blue line. Interestingly, as the data set size increases, RFA performs better in avoiding selecting bad features as we can see the red curve diverges from the blue line as the data set size increases. Lastly, Figure (c) in all the figures shows the percentage of good features selected compared to random selection. Again, RFA is outperforming random selection in this metric, as evidenced by the red line being above the blue line, and it also improves with increasing data set size.
4.2 Comparison of RFE and RFA in applying feature selection on the synthetic Majority problem data sets

We measured the performance of feature selection on the synthetic group data sets (Majority problem data sets), where each problem contains 80 irrelevant features. Recall that, we chose this kind of data set to compare the performance of RFA and RFE with the presence of irrelevant features (we would like to check whether irrelevant features will mislead the feature selection methods or not). Table 4.1 shows the performance evaluation results before and after applying feature selection on the four synthetic Majority data sets. Columns 2 and 3 show the accuracy and $F$-measure of the classifier without feature selection for these synthetic data sets (the baseline). These accuracy and $F$-measure values are calculated after feeding the entire data set to the SVM classifier.

Columns 4-7 represent the RFE performance for all four metrics. These metrics are calculated iteratively after removing one feature at a time from the original set of features. When there is an improvement in applying the RFE in terms of the accuracy and the $F$-measure, the $\Delta\%$accuracy and the $\Delta\%F$-measure will be positive values. Otherwise they will have negative values. Columns 8-11 represent the RFA performance in all the four metrics. These metrics are calculated iteratively after adding a feature to the selected features. Similar to RFE, the $\Delta\%$accuracy and the $\Delta\%F$-measure could be positive or negative. For each of the four metrics, we show the score of the feature selection method with the superior performance by using bold-face.

In order to have a better picture of the performance of both methods, we projected the $\Delta\%$accuracy vs. $\Delta\%F$-measure on an x-y axis. The projection is plotted for all majority problem data sets in Figure 4.5.
Table 4.1: Performance metrics before and after applying both RFA and RFE on the Synthetic Majority problem data sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>Without feature selection</th>
<th>RFE</th>
<th>RFA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACC</td>
<td>F-measure</td>
<td>ACC</td>
</tr>
<tr>
<td>Majority1000</td>
<td>0.75</td>
<td>0.73</td>
<td>0.75</td>
</tr>
<tr>
<td>Majority800</td>
<td>0.65</td>
<td>0.56</td>
<td>0.67</td>
</tr>
<tr>
<td>Majority600</td>
<td>0.60</td>
<td>0.47</td>
<td>0.63</td>
</tr>
<tr>
<td>Majority300</td>
<td>0.58</td>
<td>0.43</td>
<td>0.62</td>
</tr>
</tbody>
</table>
Figure 4.5: Δ%Accuracy vs. Δ%F-measure for both RFA and RFE on Majority problem data sets (a) Majority300 (b) Majority600 (c) Majority800 (d) Majority1000

The top right quartile of the graph represents the best performance in both metrics (Δ%accuracy and Δ%F-measure). The top left quartile would represent good performance in Δ%F-measure and bad performance in Δ%accuracy. The bottom right quartile would represent good performance in Δ%accuracy and poor performance in Δ%F-measure. The last quartile (bottom left) represents poor performance in both metrics.

It can be noticed from Figure 4.5 that all of the blue diamonds (which represent the RFA) are in the right hand-side quartiles. Interestingly, all of the blue diamonds are above and to the right of their red squares counterparts (which represent the RFE). These results show the superior performance of the RFA over the RFE on the synthetic majority data sets using Δ%accuracy and Δ%F-measure metrics.

We can see here from the Figure 4.5 that RFA is showing consistent improvement on both metrics and outperforming RFE in all of the majority problem data sets. However,
for the RFE, the performance showed a clear degrading with increasing the data set size. As it can be noticed from the Figure 4.5d, the RFE performance is located at the bottom left quartile (which means negative value in both metrics).

4.3 Results of applying feature selection on the small, real-world benchmark data sets

In addition to the synthetic data sets, we applied feature selection on 5 small, real-world benchmark data sets. In this experiment, we compared the proposed method (RFA) with RFE, two wrapper methods and two filter methods. As a base line, we measured the SVM accuracy of each data set before applying feature selection. In Table 4.2 the results of applying feature selection on the those data sets are shown. In Column 2, the SVM accuracy without feature selection is shown. In Columns 3 and 4, we show the maximum obtained accuracy for RFA and RFE respectively as embedded methods. The accuracy is calculated after adding a feature in RFA, or removing a feature in RFE. Then, we report the maximum obtained value from each method. In Columns 5 and 6, we show the maximum obtained accuracy for Wrapper subset evaluation and Classifier subset evaluation respectively as Wrapper methods. In Columns 7 and 8, we show the maximum obtained accuracy for CFS subset evaluation and Consistency subset evaluation respectively as Filter methods.
Table 4.2: The maximum classification accuracy obtained from RFA, RFE, two wrapper methods and two filter methods on small real-world benchmark data sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>SVM accuracy</th>
<th>Embedded methods</th>
<th>Wrapper methods</th>
<th>Filter methods</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>RFA</td>
<td>RFE</td>
</tr>
<tr>
<td>Diabetes</td>
<td>74.86</td>
<td>71.87</td>
<td>74.32</td>
<td>69.92</td>
</tr>
<tr>
<td>Spect</td>
<td>80.14</td>
<td>91.97</td>
<td>81.98</td>
<td>91.97</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>90.31</td>
<td>94.87</td>
<td>91.39</td>
<td>92.59</td>
</tr>
<tr>
<td>QSAR</td>
<td>82.46</td>
<td>83.12</td>
<td>82.41</td>
<td>85.68</td>
</tr>
<tr>
<td>Hill_valley</td>
<td>51.73</td>
<td>52.97</td>
<td>49.55</td>
<td>51.32</td>
</tr>
</tbody>
</table>

From the Table 4.2, we can notice that RFA is improving the performance in four out of five data sets (80%) in terms of the classification accuracy as compared to the accuracy without feature selection. The only data set where RFA could not improve the performance is diabetes where it degrades the performance, but that is the case with the other feature selection methods in the experiment. The reason for this degradation is that in some cases feature selection does not improve the performance and diabetes is an example of those cases. For the Spect data set, RFA improved the accuracy of the SVM classifier by 11% same as the other methods except RFE. For Ionosphere data set, RFA outperformed all the other methods. For the QSAR data set, RFA was able to outperform RFE but not the other methods. In the last data set, Hill_valley, RFA outperformed all the other methods. We also measured the total time taken for applying feature selection on each data set for all the data sets as shown in Table 4.3. Interestingly, CFS method also showed great performance compared with the performance without feature selection although it is a filter method as it was able to improve the performance in four out of five data sets. It can be noticed that CFS method sometimes performs as good as RFA and RFE.
Table 4.3: The total time taken for applying feature selection for RFA compared to RFE, two wrapper methods and two filter methods on small real-world benchmark data sets (Time in seconds)

<table>
<thead>
<tr>
<th>Data set</th>
<th>Embedded methods</th>
<th>Wrapper methods</th>
<th>Filter methods</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RFA</td>
<td>RFE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Wrapper subset evaluation</td>
<td>Classifier subset evaluation</td>
<td>CFS subset evaluation</td>
</tr>
<tr>
<td>Diabetes</td>
<td>2.0</td>
<td>2.85</td>
<td>5.36</td>
</tr>
<tr>
<td>Spect</td>
<td>2.69</td>
<td>3.80</td>
<td>3.43</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>13.09</td>
<td>21.30</td>
<td>41.04</td>
</tr>
<tr>
<td>QSAR</td>
<td>109.91</td>
<td>191.62</td>
<td>235.44</td>
</tr>
<tr>
<td>Hill_valley</td>
<td>51.73</td>
<td>52.97</td>
<td>49.55</td>
</tr>
</tbody>
</table>

Filter methods are known to be fast feature selection methods. Therefore, they outperformed the other methods in most of the data sets in the total time taken for applying feature selection as shown in Table 4.3. However, RFA was able to outperform RFE in this metric in all (100%) of the data sets and wrapper subset evaluation in four of the data sets (80%). For the classifier subset evaluation, since it does not involve cross-validation, it performs faster than wrapper subset evaluation and faster than both RFA and RFE.

Since the accuracy may change upon adding or removing features, there will be some optimal classification accuracy that can be achieved by a given method for some specific number of features. That number of features that lead to the optimal accuracy differs and depends on the features list produced during the feature selection. Therefore, we also measured the total time taken to reach the maximum accuracy for the ranked features produced by RFA and RFE methods and for all the data sets as shown in Table 4.4.
Table 4.4: The total time taken to reach the maximum classification accuracy for both RFA and RFE methods (in seconds)

<table>
<thead>
<tr>
<th>Data set</th>
<th>RFA</th>
<th>RFE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diabetes</td>
<td>3.63</td>
<td>0.11</td>
</tr>
<tr>
<td>Spect</td>
<td>1.88</td>
<td>0.10</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>35.20</td>
<td>0.65</td>
</tr>
<tr>
<td>QSAR</td>
<td>509.21</td>
<td>0.65</td>
</tr>
<tr>
<td>Hill_valley</td>
<td>612.55</td>
<td>558.54</td>
</tr>
</tbody>
</table>

In order to show the classifier performance in terms of the classification accuracy upon adding (or removing in the case of RFE) one feature at a time, we measured the classification accuracy after each addition/elimination process. Figure 4.6 shows the classifier accuracy after adding/removing one feature at a time for both RFA and RFE respectively for the diabetes data set. In all of the coming figures in this section, the X-axis represents the number of features and the y-axis represents the accuracy which ranges from 0 to 1. The accuracy for the RFA is shown from left to right as the features added, and from right to left as the features are removed as in RFE.

Figure 4.6 shows the classifier accuracy after adding/removing one feature for diabetes data set.

As we can notice from the Figure 4.6, RFE was consistently performing better than RFA as the blue curve was above the red curve from the beginning until the end.

Figure 4.7 shows the classifier accuracy after adding/removing one feature for Spect data set. It can be noticed from this figure that RFA found the most important feature and outperformed RFE from the first iteration (when the red curve was above the blue curve) then it degraded after adding the other features.
Figure 4.6: The classifier accuracy after adding/removing one feature at a time for RFA and RFE for Diabetes data set

Figure 4.8: The classifier accuracy after adding/removing one feature at a time for RFA and RFE for Ionosphere data set

Figure 4.8 shows the classifier accuracy after adding/removing one feature for Ionosphere data set. From this figure, it can be noticed that RFA was consistently performing better that RFE as the red curve was above the blue curve most of the time.

Figure 4.9 shows the classifier accuracy after adding/removing one feature for QSAR data set. In this figure, it is difficult to find which method is predominating as the both
Figure 4.7: The classifier accuracy after adding/removing one feature at a time for RFA and RFE for Spect data set

curves were matching most of the times. However, RFA was able to record a slightly better accuracy as depicted in Table 4.2.

Figure 4.9: The classifier accuracy after adding/removing one feature at a time for RFA and RFE for QSAR data set

Figure 4.10 shows the classifier accuracy after adding/removing one feature for Hill-valley data set. In Hill_valley data set, it can be clearly noticed that RFA is prevailing most of the time as the red curve was above the blue curve most of the time as shown in Figure 4.10.
4.4 Results of applying feature selection on the large real-world benchmark data sets

In addition to the synthetic and the small real-world data sets, we measured the performance of feature selection on real-world, high dimensional data sets. Table 4.5 shows the results before and after applying feature selection on these data sets. Columns two and three in Table 4.5, below, show the accuracy and $F$-measure respectively of the classifier without feature selection for the real-world benchmark data sets (the baseline).

Columns 4-7 represent the RFE performance in all the four metrics. As with the synthetic data, these values could be positive or negative.

Columns 8-11 represent the RFA performance in all the four metrics.

We also projected the results of $\Delta$%accuracy vs. $\Delta$%$F$-measure on an x-y axis to have better understanding of the performance of both RFA and RFE. The projection for all real-world data sets is plotted in Figure 4.11.

Similar to Figure 4.5, the top right quartile of the graph represents the best performance in both metrics. The last quartile (bottom left) would represent bad performance in both metrics.
<table>
<thead>
<tr>
<th>Data set</th>
<th>Without feature selection</th>
<th>RFE</th>
<th>RFA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACC</td>
<td>F-measure</td>
<td>ACC</td>
</tr>
<tr>
<td>Breast Cancer</td>
<td>0.52</td>
<td>0.35</td>
<td>0.71</td>
</tr>
<tr>
<td>DB emails</td>
<td>0.56</td>
<td>0.44</td>
<td>0.90</td>
</tr>
<tr>
<td>Colon cancer</td>
<td>0.64</td>
<td>0.50</td>
<td>0.67</td>
</tr>
<tr>
<td>CNS</td>
<td>0.65</td>
<td>0.51</td>
<td>0.65</td>
</tr>
<tr>
<td>Pomeroy</td>
<td>0.65</td>
<td>0.51</td>
<td>0.65</td>
</tr>
<tr>
<td>Leukemia</td>
<td>0.71</td>
<td>0.59</td>
<td>0.78</td>
</tr>
<tr>
<td>Lymph</td>
<td>0.75</td>
<td>0.64</td>
<td>0.82</td>
</tr>
<tr>
<td>DLBCL (small)</td>
<td>0.75</td>
<td>0.64</td>
<td>0.88</td>
</tr>
<tr>
<td>DLBCL</td>
<td>0.55</td>
<td>0.39</td>
<td>0.59</td>
</tr>
<tr>
<td>Lung Michigan</td>
<td>0.89</td>
<td>0.84</td>
<td>0.99</td>
</tr>
<tr>
<td>Lung Ontario</td>
<td>0.61</td>
<td>0.64</td>
<td>0.65</td>
</tr>
</tbody>
</table>

Table 4.5: Performance metrics before and after applying both RFA and RFE on the real-world benchmark data sets.
Figure 4.11: $\Delta$%Accuracy vs. $\Delta$%F-measure for both RFA and RFE on real-world data sets (a) Breast (b) CNS (c) Colon (d) DB_emails (e) DLBCL (f) DLBCL_small (g) Leukemia (h) Lung Michigan (i) Lung Ontario (j) Lymph (k) Pomeroy

It can be noticed from Figure 4.11 that there are no points in the left hand-side quartiles. Interestingly, most of the blue diamonds (which represent the RFA) are above and to the right of their red squares counterparts (which represent the RFE).
These results show the superior performance of the RFA over the RFE on the real-world data sets using $\Delta$%accuracy and $\Delta$%F-measure metrics.

4.5 Statistical analysis

In order to test the significance of the results of the two methods (RFA and RFE) since they show close performance with some data sets- we performed statistical analysis to make the comparison. We used the non-parametric Mann-Whitney U test to examine the significance of the results of the two methods on both of the metrics (accuracy and $F$-measure). We used the Mann-Whitney U test since this test is more suitable with microarray data sets than t-test, in that it is robust against outliers and does not assume a normal distribution Guile & Wang (2008). Our testing approach consists of the following steps:

**Step 1:** State the hypothesis and identify the claim

$H_0$: The RFE performance is better than RFA performance

$H_1$: The RFE performance is less than or equal to the RFA performance

**Step 2:** Apply the unpaired test and obtain the $p$-value to draw a conclusion. If the $p$-value is less than 0.05, then we reject the $H_0$ and accept the $H_1$. Otherwise, we cannot reject any hypothesis as we do not have enough evidence to make a decision. In such cases, we get $p$-value greater than 0.05, which indicates that there is not enough evidence that RFA is superior to RFE.

**Step 3:** If the null hypothesis $H_0$ is not rejected, then no conclusion is drawn. Therefore, we decided to apply a second Mann-Whitney U test in order to try to draw a different conclusion: that RFE is superior to RFA. To do this we reverse the hypothesis.

$H_0$: The RFA performance is better than RFE performance

$H_1$: The RFA performance is less than or equal to the RFE performance
Step 4: Apply the unpaired test on the second version of the null-hypothesis and obtain the $p$-value to make the decision.

Therefore, we have three different cases in total when we apply the two tests:

**Case 1:** $p$-value is less than 0.05, RFA is superior to RFE

**Case 2:** $p$-value is greater than or equal to 0.05, no conclusion. Therefore, to check if RFE is superior to RFA, we reversed the $H_0$ hypothesis to be RFE is less than or equal to RFA, if the $p$-value is less than 0.05, RFE is superior to RFA. If neither test returns a significant result, we have Case 3

**Case 3:** when we there is not enough evidence that RFA is superior to RFE or vice versa, each $p$-value is greater than 0.05 in both directions. We draw no conclusion in this case.

The above cases can be viewed better through Table 4.6. When the first $p$-value is less than 0.05 (Case1), that is evidence that RFA is superior to RFE, as depicted in the bottom left cell of Table 4.6. In the above case, we would not conduct Test 2 as it is obvious that it would be not significant. However, when the first $p$-value is greater than or equal to 0.05 (Case 2), we have to do Test 2, then if we get a value of less than 0.05 for the second $p$-value, then this is evidence that RFE is superior to RFA, as in the top right cell of the table. In Case 3, even after we do Test 2, we get both $p$-values greater than or equal to 0.05, that means there is no conclusion that can be drawn from these tests, as in the bottom right cell of the table.

As can be noticed from the Table 4.6, there is one "Not Applicable" case when both tests are significant, which is mathematically impossible.

<table>
<thead>
<tr>
<th>Tests</th>
<th>Test1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test2</td>
<td></td>
</tr>
<tr>
<td>Significant</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Not significant</td>
<td>Case2 (RFE &gt; RFA)</td>
</tr>
</tbody>
</table>

Table 4.6: Statistical tests and the resulting cases
We applied the statistical test on the synthetic and real-world data sets using both metrics Accuracy and $F$-measure as shown in Table 4.7.

Table 4.7: Mann-whitney U test for RFA and RFE results on the synthetic majority data sets using both metrics Accuracy and $F$-measure

<table>
<thead>
<tr>
<th>Data set</th>
<th>Test decision (Accuracy)</th>
<th>Test decision ($F$-measure)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Majority1000</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>Majority800</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>Majority600</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>Majority300</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
</tbody>
</table>

In addition, we applied the same statistical test on the small data sets’ results to check the significance of the results and to compare RFA and RFE at the same time. In Table 4.8, the results of Mann-Whitney U test on the small data sets’ results are shown.

Table 4.8: Mann-whitney U test for RFA and RFE results on the small data sets using Accuracy

<table>
<thead>
<tr>
<th>Data set</th>
<th>Test decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diabetes</td>
<td>RFE &gt; RFA</td>
</tr>
<tr>
<td>Spect</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>QSAR</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>Hill_valley</td>
<td>No conclusion</td>
</tr>
</tbody>
</table>

From the Table 4.8 we can notice that RFA outperformed RFE in three data sets: spect, ionosphere and QSAR (60%), while RFE was better in one data set (diabetes) with percentage of 20% of the total number of data sets. In only one data set (Hill_valley) there was no conclusion since the $p$-value was greater than 0.05 in both directions which is only 10% of the data sets, therefore we could not draw any conclusion.
In addition, we applied the same tests on the results of feature selection on the real-world high dimensional data sets for both metrics accuracy and $F$-measure and the results are stated in Table 4.9.

Table 4.9: Mann-whitney U test for RFA and RFE results on the real-world data sets using both metrics Accuracy and $F$-measure

<table>
<thead>
<tr>
<th>Data set</th>
<th>Test decision (Accuracy)</th>
<th>Test decision ($F$-measure)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast cancer</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>Double world email bodies</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>Colon cancer</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>CNS</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>Pomeroy</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>Leukemia</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>Lymph</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>DLBCL (small)</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>DLBCL</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>Lung Michigan</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
<tr>
<td>Lung Ontario</td>
<td>RFA &gt; RFE</td>
<td>RFA &gt; RFE</td>
</tr>
</tbody>
</table>

From both Tables 4.7 and 4.9 we can notice that RFA is always superior to RFE in terms Accuracy and $F$-measure.

4.6 Summary of the chapter

This chapter presented the results of applying the deep analysis on RFA, the same way that we applied the test on RFE. We showed the behaviour of RFA in selecting the entangled features among the other features on all the synthetic data sets and compared that to random selection. The results of synthetic data sets shows that RFA selects good features well and does not select bad features.
In addition, this chapter presented the results of applying the proposed feature selection RFA, on three groups of data sets: synthetic data sets, small real-world data sets, and large real-world data sets. In the first group, the synthetic data sets, we showed the performance of RFA on the majority problem data sets compared to RFE and the results showed that RFA outperformed RFE on this group of data sets. In the second group, we showed the results of applying RFA to five small real-world data sets and compared the results to one embedded method (RFE), two wrapper methods, and two filter methods. We noticed that RFA showed poor performance on one of the data sets when it was not able to achieve better than SVM’s accuracy and showed better performance on the other four data sets. However, when we compared the RFA performance with other embedded, wrapper and filter methods, RFA showed better performance than RFE on four data sets and better than (sometimes comparable to) the other wrapper and filter methods. However, by comparing the total time taken to reach the maximum accuracy, RFA outperformed RFE in all the data sets and outperformed the wrapper subset evaluation method in four data sets, while it was slower than the other methods.

In the last section, we tested the statistical significance of the results of the three groups of data sets using Mann-Whitney U test. For the synthetic data set, the statistical test showed a superior performance of RFA over RFE in all the data sets and using the two metrics: accuracy and F-measure. For the small real-world data sets, the statistical test showed that RFA did better in all data sets except for diabetes data set when RFE outperformed RFA, and with Hill-valley where we could not draw any conclusion. In the third group, the large real-world data sets, again, the statistical test showed that there is an enough evidence that RFA was statistically better than RFE on this group of data sets using accuracy and F-measure.
Chapter 5

Intrusion Detection Application

The purpose of this chapter is to demonstrate the role of feature selection in intrusion detection as an application and specifically using the ISCX 2012 data set. Feature selection has been used with intrusion detection recently to improve the detection performance. In this chapter we introduce the ISCX data set and its structure in Section 5.1. In addition, we explain the feature extraction and data set preparing process for intrusion detection in Section 5.2. Then, we discuss feature selection on the ISCX data set in Section 5.3. The evaluation metrics that we use in this work are explained in Section 5.4. In Section 5.5, we discuss applying RFA on the ISCX data set. We also apply RFE on the ISCX data set and discussed the results in Section 5.6. We compared the distribution of the performance of RFA and RFE visually in Section 5.7. In Section 5.8, we evaluate the statistical significance of the results of RFA and RFE on ISCX 2012 data set. We also proposed four ranking coefficients for RFA in addition to the original one and provided a comprehensive analysis on their performance in Section 5.9. We conclude the chapter by summarizing the results of the chapter in Section 5.11.
Table 5.1: Description of ISCX Data Set

<table>
<thead>
<tr>
<th>Day</th>
<th>#features</th>
<th>#examples</th>
<th>#normal</th>
<th>#attacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>June11th</td>
<td>19</td>
<td>378,667</td>
<td>378,667</td>
<td>0</td>
</tr>
<tr>
<td>June12th</td>
<td>19</td>
<td>133,197</td>
<td>131,110</td>
<td>2087</td>
</tr>
<tr>
<td>June13th</td>
<td>19</td>
<td>110,588</td>
<td>632</td>
<td>109,956</td>
</tr>
<tr>
<td>June14th</td>
<td>18</td>
<td>171,380</td>
<td>167,594</td>
<td>3786</td>
</tr>
<tr>
<td>June15th</td>
<td>20</td>
<td>572,410</td>
<td>534,830</td>
<td>3840</td>
</tr>
<tr>
<td>June16th</td>
<td>20</td>
<td>523,160</td>
<td>523,134</td>
<td>26</td>
</tr>
<tr>
<td>June17th</td>
<td>19</td>
<td>398,516</td>
<td>393,181</td>
<td>5335</td>
</tr>
</tbody>
</table>

5.1 ISCX data set

In this section, we will explain the ISCX 2012 data set. This data set has been generated by the Information Security Centre of Excellence (ISCX) at University of New Brunswick in 2012 (Shiravi, Shiravi, Tavallaee, & Ghorbani, 2012). The data set involves real traces analyzed to create profiles for agents that generate real traffic for HTTP, SMTP, SSH, IMAP, POP3, and FTP. The generated data set contains different features including including full packet payloads in addition to other relevant features such as total number of bytes sent or received. The full data set has been captured in a period of seven days (From Friday June 11th at 00:01:06 s to Friday June 18th at 00:01:06 s) and involved about more than one million network trace packets and 20 features, and every data example has been labeled as one of two classes (normal or attack) (Yassin et al., 2014). The ISCX data set has acquired the security community’s attention and become a benchmark data set for intrusion detection research purposes due to its realistic traffic, labeled connections, and its multiple attack scenarios (Shiravi et al., 2012). The data set has been designed to overcome the technical limitations of other intrusion detection data sets, and to prepare network traces by capturing contemporary legitimate and intrusive network behaviours and patterns (Tan et al., 2015). A summary of the ISCX data set is provided in Table 5.1.
5.2 Feature extraction and data set preparation

In this section, we will explain how we minimized the size of data set in both (number of features and number of examples). We start with how we encoded the string features to 4k features. Then we talk about how we used CFS method to decrease the number of features to 350. Next we explain how we generated different sizes (in terms of number of examples) of data sets. The ISCX data set consists of different types of features: numeric, categorical, datetime, and strings. Usually the packet header information are represented by a mixture of the above types, but the payload features are usually represented by long string values such as the `sourcePayloadAsBase64` feature which contains very long strings that makes it difficult to deal with in machine learning. We chose to summarize these features by using a bigram technique. Figure 5.1 illustrates the main steps of the feature extraction process that we employed to extract features using a bigram technique (which will we explain in more detail later). We used this bigram technique with payload features to investigate if the payload features include informative features or not. We opted to do this since many researches ignore these features due to their long strings, which makes them not easy to utilize in machine learning.

![Feature extraction process for ISCX data set using bigram technique](image)

Figure 5.1: Feature extraction process for ISCX data set using bigram technique

Now the first step in the feature extraction process for payload features is the Dictionary construction. In order to extract the feature vector for each payload feature, we need to build a dictionary that contains all the bigrams from all the examples. The detailed steps for the dictionary construction are shown in Algorithm 3.
Algorithm 3 Dictionary construction for long payload features

1: procedure Dictionary Construction
2: Input: long payload features
3: Initialize an empty dictionary $D$
4: $i \leftarrow 1$
5: while $i \leq \text{No. of features}$ do
6: Take one feature $f_i$
7: $j \leftarrow 1$
8: while $j \leq \text{No. of bigrams in } S_i$ do
9: Take one bigram $b_j$
10: if (dictionary $D$ contains bigram $b_j$) then
11: Add feature $b_j$ to $D$
12: $j \leftarrow j + 1$
13: End while
14: $i \leftarrow i + 1$
15: End while
16: Output: Dictionary $D$.

This algorithm generates the dictionary that is used in the next step to extract the feature vectors for the long payload features. Figure 5.2 shows the block diagram of dictionary construction. The next step is the feature vector extraction for the long payload features which is outlined in Figure 5.3.

The feature vector extraction step is also explained in more detail in Algorithm 4. The result of Algorithm 4 is the feature vectors for all the payload string features from the ISCX data set. After this step, the number of features can expand to thousands, in our case the number of the resulting features is 4123 features including the original features (i.e., non-payload features). Let’s take a small example on how the payload features are converted to bigrams according to the aforementioned technique. In our example we have only three
training instances and each one contains a payload feature with different contents. The original three payload features are: "B7z2", "Vud3j" and "z2nB7" respectively. By feeding
those features to the dictionary generation according to the Algorithm 3, the resulting dictionary will consist of 9 words (bigrams) as follows: B7 | 7z | z2 | Vu | ud | d3 | 3j | 2n | nB. By applying the feature vector extraction on the given three payload features according
Algorithm 4 Feature vector extraction for long payload features

1: procedure Feature vector extraction
2: Input: Long payload features, Dictionary
3: Initialize all feature vectors to zero
4: \( i \leftarrow 1 \)
5: while \( i \leq \text{No. of Strings} \) do
6: \( S_i \)
7: \( j \leftarrow 1 \)
8: while \( j \leq \text{No. of bigrams in String } S_j \) do
9: \( b_j \)
10: Take the index \( \text{idx} \) of bigram \( b_j \) from dictionary
11: Increment the corresponding location in the feature vector: \( \text{feature vector}[\text{idx}]++ \)
12: \( j \leftarrow j + 1 \)
13: End while
14: Finish feature vector \( i \)
15: \( i \leftarrow i + 1 \)
16: End while
17: Output: Feature vectors.

Algorithm 4, the resulting bigram representation for each of the above three features will be as depicted in Table 5.2.

Table 5.2: Bigram representation for the three payload features in the example

<table>
<thead>
<tr>
<th>original payload</th>
<th>B7</th>
<th>7z</th>
<th>z2</th>
<th>Vu</th>
<th>ud</th>
<th>d3</th>
<th>3j</th>
<th>2n</th>
<th>nB</th>
</tr>
</thead>
<tbody>
<tr>
<td>B7z2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Vud3j</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>z2nB7</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

The three payload features in this example are converted to bigram features as in Table 5.2. The table’s header represents the standard feature vector for all the payload features.
in this example. The payload features appeared in the table according to the order of their presentation to the feature extraction algorithm (i.e., the first row corresponds the first payload feature discovered during dictionary generation and so on). Note, that the order in which the bigrams occur within a given payload is not preserved by this encoding.

Now in order to prepare the resulting data set for feature selection, we conducted a pre-ranking step for the features using a quick Filter selection algorithm which is (CFS). Since the current number of features is huge and the feature selection can be very time consuming in this case, we took a subset of 350 features from the original features. Those 350 features are divided into 10 chunks of 35 features each. We took 9 chunks of the bad features and 1 chunk of the good features. The intentional inclusion of a large number of bad features was done to test the efficacy of our feature selection algorithm. Those resulting 350 features are used to generate data sets of size 25, 50, 100, and 500 examples respectively. We generated those data sets as balanced data sets (i.e. we made them of equal number of normal and attack examples). We used different numbers of examples to monitor the behaviour of feature selection with each size of data set. All the steps from the feature extraction until ranking the features using feature selection algorithms are outlined in Figure 5.4

For the final step (feature selection), we will apply both RFA and RFE on the four data sets so that we can compare these two methods with ISCX data set.
5.3 Feature Selection on ISCX data set

As mentioned before, this step involves applying feature selection methods on the ISCX data set. In the next sections, we will apply both RFA and RFE on the ISCX data set and analyze the results of each method on the four generated data sets. For each method we repeated the experiment 30 times and used 3 folds cross-validation for testing.
5.4 Feature selection evaluation metrics for IDS

For most IDS applications some additional metrics are usually calculated to check how effective the system is. Therefore, we added three metrics to the accuracy and $F$-measure -that we used with other data sets- as follows:

(a) **Detection rate or Recall**

The detection rate ($DR$) or *Recall* of any IDS represents the percentage of the correctly detected attacks to the total number of attacks according to the following formula:

$$\text{Detection Rate (DR)} = \frac{TP}{TP + FN} \quad (5.1)$$

(b) **Precision**

The precision represents the percentage of correctly classified attacks $TP$ to the total number of classified examples as attacks (both $TP$ and $FP$) according to the following formula:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (5.2)$$

(c) **False Alarm Rate (FAR)**

False Alarm Rate ($FAR$) represents the percentage of normal instances that have been incorrectly classified as attacks to the total number of normal instances according to the following formula:

$$FAR = \frac{FP}{FP + TN} \quad (5.3)$$

(d) **Proposed overall performance**

In addition to the previous mentioned metrics, we propose an *overall* metric in this
thesis so that it can compare multiple systems by incorporating three metrics together (Accuracy, Detection rate, and FAR) according to the following formula:

\[
Overall = \left\{ \frac{ACC + DR}{2} \right\} - FAR
\]  

(5.4)

The result of this metric will be a real value between -1 and 1, \( overall \in [-1, 1] \), where -1 corresponds to the worst overall system performance and 1 corresponds the best overall system performance. Accordingly, 0 corresponds to 50% overall system performance. To clarify the importance of our proposed overall metric, assume that we have a data set with 200 instances split into 100 normal instances and 100 attack instances. Now lets discuss three different scenarios that have the same accuracy 50%.

**Scenario 1:**

\[
\text{Confusion matrix} = \begin{bmatrix} 1 & 99 \\ 1 & 99 \end{bmatrix}
\]

**Scenario 2:**

\[
\text{Confusion matrix} = \begin{bmatrix} 50 & 50 \\ 50 & 50 \end{bmatrix}
\]

**Scenario 3:**

\[
\text{Confusion matrix} = \begin{bmatrix} 70 & 30 \\ 70 & 30 \end{bmatrix}
\]

For the above three different confusion matrices, we can calculate the performance metrics as shown in Table 5.3.

All of the above mentioned scenarios have equal accuracy metric. However, the three scenarios have different detection rates and false alarm rates. That makes it difficult to decide which system performs the best among the three systems that produced the three scenarios. However, if we calculate the proposed overall metric, we can conclude that the system that generates the third scenario is the best system among the others, and that is due to its highest overall metric value (0.1). 

104
Similarly, there might be different systems with equal detection rates and different accuracies and false alarm rates as shown in the three scenarios below which all have the same detection rate 50%.

**Scenario 4:**

\[
\text{Confusion matrix} = \begin{bmatrix}
1 & 99 \\
50 & 50
\end{bmatrix}
\]

**Scenario 5:**

\[
\text{Confusion matrix} = \begin{bmatrix}
99 & 1 \\
50 & 50
\end{bmatrix}
\]

**Scenario 6:**

\[
\text{Confusion matrix} = \begin{bmatrix}
50 & 50 \\
50 & 50
\end{bmatrix}
\]

When we can calculate the performance metrics for the above three different confusion matrices, we can conclude the best system as shown in Table 5.4.

Table 5.4: Performance metrics for three different scenarios with equal detection rate

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Accuracy</th>
<th>DR</th>
<th>FAR</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scenario 4</td>
<td>0.255</td>
<td>0.5</td>
<td>0.99</td>
<td>-0.6125</td>
</tr>
<tr>
<td>Scenario 5</td>
<td>0.745</td>
<td>0.5</td>
<td>0.01</td>
<td>0.6125</td>
</tr>
<tr>
<td>Scenario 6</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.0</td>
</tr>
</tbody>
</table>

From the Table 5.4, we can conclude that the best system that has the maximum overall metric which is the system that generates the second scenario.
In the same way, there might be different systems that generate equal false alarm rates with different accuracies and detection rates as shown in the three scenarios below which all have the same false alarm rate 50%.

**Scenario 7:**

\[
\text{Confusion matrix} = \begin{bmatrix}
50 & 50 \\
10 & 90
\end{bmatrix}
\]

**Scenario 8:**

\[
\text{Confusion matrix} = \begin{bmatrix}
50 & 50 \\
90 & 10
\end{bmatrix}
\]

**Scenario 9:**

\[
\text{Confusion matrix} = \begin{bmatrix}
50 & 50 \\
50 & 50
\end{bmatrix}
\]

By calculating the performance metrics for the above three scenarios, we can see the differences as shown in Table 5.5.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Accuracy</th>
<th>DR</th>
<th>FAR</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scenario 7</td>
<td>0.7</td>
<td>0.9</td>
<td>0.5</td>
<td>0.3</td>
</tr>
<tr>
<td>Scenario 8</td>
<td>0.3</td>
<td>0.1</td>
<td>0.5</td>
<td>-0.3</td>
</tr>
<tr>
<td>Scenario 9</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
</tr>
</tbody>
</table>

By considering the overall performance, we can conclude that the system that corresponds to the first scenario is the best system since its overall metric is 0.3 which is the highest among the others.

Therefore, we can conclude from the 9 scenarios above, that none of the accuracy, detection rate, and false alarm rate can be expressive enough in measuring the performance of the intrusion detection system when they are taken individually. However, our proposed overall metric can combine the information given by all of the above three metrics to measure the performance more thoroughly as we noticed from the aforementioned scenarios. By using the proposed overall metric, we can choose the system that has the best obtained maximum accuracy, maximum detection rate and minimum false alarm rate.
5.5 RFA application on ISCX data set

In order to observe the effect in including the payload features in improving the detection accuracy, we conducted a crucial experiment. We measured the SVM’s classification accuracy and $F$-measure on all the ISCX data sets without the payload features and with payload features. We measured the performance metrics after converting the payload features to bigram features and applying RFA feature selection. The goal of this experiment is to show that these payload features include important and helpful information in improving the detection accuracy. Many of the previous studies excluded payload features from the original set of features before looking for intrusions due to the difficulty involved in dealing with those features. We applied our RFA method on the four generated data sets of ISCX and reported the maximum obtained accuracy and $F$-measure as shown in Table 5.6.

Table 5.6: Performance metrics without and with bigram features after applying RFA on the ISCX data sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>Without bigram features</th>
<th>RFA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACC</td>
<td>$F$-measure</td>
</tr>
<tr>
<td>25 examples</td>
<td>67%</td>
<td>67%</td>
</tr>
<tr>
<td>50 examples</td>
<td>77%</td>
<td>77%</td>
</tr>
<tr>
<td>100 examples</td>
<td>80%</td>
<td>80%</td>
</tr>
<tr>
<td>500 examples</td>
<td>82%</td>
<td>82%</td>
</tr>
</tbody>
</table>

Columns 2 and 3 of Table 5.6 represent the classifier’s performance on each data set with the payload features removed from that data set. Columns 4 and 5 represent the maximum obtained performance from the classifier after including the payload features, expanding them using the bigram technique and applying RFA on the resulting data set. We can notice that there is a significant improvement (between 8%-11%) after applying RFA on
the bigram features compared to the performance of the SVMs classifier on the same data sets without bigram features.

In addition, we wanted to visually study the behaviour of feature selection with bigram and non-bigram features. Therefore we extracted the following graphs:

(a) The number of selected bigrams compared to random and CFS with random selection

This graph consist of three curves. In the first curve (represented by the red line), the goal is to analyze the behaviour of RFA in selecting the bigrams. First, we reported the number of bigrams selected at each feature index, then we summed up these numbers. Since we repeat the experiment 30 times and we do 3-folds cross-validation at each iteration, then the maximum number of bigrams that can be selected is 90. Therefore, we averaged the summation over 90 to check the probability of selecting bigram features at each iteration. The second curve (the blue line) represents probability of selecting bigram features randomly at each iteration. If RFA selected all the bigram features first and then the non-bigram ones, the curve will be increasing with a slop of 1 until feature 345 then will be flat until the end. The other case, if RFA selected all the non-bigrams first and then the bigram ones, then the curve will be flat from 0-5 then will be increasing with a slop of 1 until the end.

Since the total number of features is 4123 features and the number of non-bigram features is 11 only, then the remaining features are 4112 bigram features. Therefore, the probability of selecting a bigram feature is 4112/4123 at each feature index, and the expected cumulative number of features selected is 4112/4123 multiplied by the feature index. The third graph (the green line) represents the probability of selecting bigram features at random from the features pre-selected by the CFS. After applying CFS, we have 350 features in total, 345 features are bigram features and only 5 are non-bigram features. Therefore, the probability of selecting a bigram feature is 345/350 and the expected cumulative number of bi-grams selected at each index is 345/350 times the feature index. The goal of the first figure is to give us an indication
about the RFA performance compared to random selection before and after applying CFS.

(b) The number of selected Non-bigrams compared to random and CFS with random selection

This graph consists of three curves related to non-bigram features. The goal of this graph is to analyze the behaviour of RFA in selecting non-bigram features. The first curve (the red line) represents the probability of RFA selecting non-bigram features. First we calculated the number of non-bigram features selected at each iteration. Since we have calculated the number of selected bigram features $B$, then the number of non-bigram features is $90-B$. After that, we summed up these numbers. Then we averaged the summation over 90 to check the probability of selecting non-bigram features at each iteration. If RFA selected all the bigram features first and then the non-bigrams, the curve will be flat from 0 until 345 then it will be increasing with a slop of 1 until the end. The other case, if RFA selected all the non-bigram features first and then the bigrams, then the curve will start from 0 and increase with a slop of 1 until 5 then will flat until the end.

The second curve (the blue line) represents selecting non-bigram features at random before applying the CFS. Since we have only 11 non-bigram features before applying the CFS, the probability of selecting non-bigrams is $11/4123$ at each feature index, and the expected cumulative number of features selected is $11/4123$ multiplied by the feature index. The third graph (the green line) represents the probability of selecting non-bigram features at random from the features pre-selected by CFS. Since we have only 5 non-features after applying the CFS, then the probability is $5/350$ and the expected cumulative number of bigram features selected at each index $5/350$ multiplied by feature index.

(c) Zoomed number of selected Non-bigrams compared to random and CFS with random selection
The third graph is just a zoomed version of the second graph for only the first 60 features. This graph gives a closer picture on how the non-bigram features are being selected in the first 60 iterations.

We generated the above three figures for all the ISCX data sets after applying RFA. In Figure 5.5, we show the RFA behaviour in selecting bigram and non-bigram features with ISCX 25 examples.

Figure 5.5: RFA bigram and non-bigram selection on ISCX data set 25 examples compared to random selection.
As we can notice from Figure 5.5a, there is no difference between all the of curves, which means the probability of RFA selecting bigram features, the probability of randomly selecting bigram features and the probability of selecting random bigram features after CFS are almost the same. Therefore, we chose to analyze the results in a different way and look at the results from a different angle. In Figure 5.5b, selecting non-bigram features by RFA is depicted (the red line) and compared to random selection before and after CFS. It can be noticed that all of the non-bigram features (which are 5 only) are selected early always within the first 58 iterations. As compared to random non-bigram selection (the blue line), RFA selects the non-bigram features much earlier than random. In addition, RFA selects non-bigram features earlier than the random non-bigram selection after CFS (the green line).

By zooming the picture of RFA non-bigram selection to the first 60 features only as shown in Figure 5.5c, we can notice that non-bigram features are never selected in the first iteration, for this reason, the red curve starts from 0. However, quickly after the first iteration, RFA starts selecting some non-bigram features as shown in Figure 5.5c, where the red curve goes to 1 in the second iteration. However, this is not the case with random non-bigram selection before and after CFS (blue and green lines respectively), where their probabilities do not reach even to 1 feature in the first 60 iterations.

The previous figure was about ISCX 25 examples, in Figure 5.6 we show the results of RFA on ISCX data set with 50 examples only. Interestingly, the pattern is repeated with selecting bigram features with ISCX 50 examples as shown in Figure 5.6a. We can notice that there is no difference between the probability of RFA selecting bigram features, random selection and random selection after CFS as all the three curves are conforming all the time. However, selecting non-bigrams is different as RFA quickly selects non-bigrams, but again not from the first iteration as shown in Figure 5.6b as the red curve starts from 0. Again, RFA selects non-bigram features much earlier than random non-bigram selection before and after CFS as the red curve goes up before the green and the blue curves. By getting insight into RFA non-bigram selection, we can notice that RFA finished selecting all non-bigram
features by iteration 54 as shown in Figure 5.6c, where the red curve reaches the maximum (5 features) at iteration 54 while both the green and blue curves are still below 1.

Figure 5.6: RFA bigram and non-bigram selection on ISCX data set 50 examples compared to random selection

We generated the same three figures also for the ISCX 100 examples data set as shown in Figure 5.7. As with the ISCX 25 and ISCX 50 examples, we cannot notice any difference between RFA bigram selection, and random selection before and after CFS as shown in Figure 5.7a as all the curves are conforming with each other all the time. However, with non-bigram selection, RFA starts from the second iteration and the probability shows that
RFA guarantees selecting one of the non-bigram features in the second iteration as the red curve goes to 1 in the second iteration. In addition, RFA is still selecting non-bigram features much earlier than random before and after CFS as shown in Figure 5.7b, where the red curve goes up and reaches the maximum (5) much earlier than the blue and green curves. The picture becomes clearer when zooming in the non-bigram selection to the first 60 iterations only as shown in Figure 5.7c. However, RFA could not select all the non-bigram features in the first 60 iterations as with ISCX 25 and 50 examples where the red curve did not reach 5 by the end of 60 iterations. In fact, RFA took 76 iterations to finish selecting all the non-bigram features as it can be noticed from the red curve of Figure 5.7b.

The last data set is the ISCX 500 examples. In Figure 5.8, we show the results of RFA bigram and non-bigram selection on the ISCX data set with 500 examples. As with previous data sets, the RFA did not differ from the random bigram selection and random bigram selection after CFS as shown in Figure 5.8a. For the non-bigram selection, RFA started selecting non-bigram features from the second iteration and securely selected one feature in the second iteration. The probability of selecting non-bigram features by RFA then increases consistently until selecting all of them. However, although the RFA starts selecting non-bigram features early, with ISCX 25 examples RFA was able to select all of them earlier than with 500 examples as the red curve in Figures 5.5b reached the maximum 5 in iteration 58, while with ISCX 500 examples the red curve reached the maximum number of non-bigram features in 174 iterations. In all the cases, the RFA is selecting non-bigram features faster than random selection before and after CFS as shown in Figure 5.8b. By zooming in into the second graph, we can notice that RFA was not able to select all of the non-bigram features in the first 60 iterations as shown in Figure 5.8c. In fact, it took the RFA 174 iterations to select all the non-bigram features as shown in Figure 5.8b.
Figure 5.7: RFA bigram and non-bigram selection on ISCX data set 100 examples compared to random selection

5.6 RFE application on ISCX data set

The previous section was about applying RFA on the ISCX data set, in this section we focus on applying the RFE method on the four generated data sets of ISCX and reporting the maximum obtained accuracy and $F$-measure as shown in Table 5.7.
Figure 5.8: RFA bigram and non-bigram selection on ISCX data set 500 examples compared to random selection

Table 5.7: Performance Metrics before and after Applying RFE on the ISCX Data Sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>Without feature selection</th>
<th>RFE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACC</td>
<td>$F$-measure</td>
</tr>
<tr>
<td>25 examples</td>
<td>67%</td>
<td>67%</td>
</tr>
<tr>
<td>50 examples</td>
<td>77%</td>
<td>77%</td>
</tr>
<tr>
<td>100 examples</td>
<td>80%</td>
<td>80%</td>
</tr>
<tr>
<td>500 examples</td>
<td>82%</td>
<td>82%</td>
</tr>
</tbody>
</table>
Again, we notice that there is a significant improvement on the performance (between 8%-11%) after applying RFE on the bigram features compared to the performance of the SVMs classifier on the same data sets without bigram features. From the results of RFA and RFE we get a clear indication about the importance of the payload features of the ISCX data set (that many works ignore).

We wanted to visually study the behaviour of feature selection with bigram and non-bigram features. Therefore we extracted the following graphs:

(a) The number of eliminated bigrams compared to random and CFS with random selection

Once again, our graph consist of three curves. In the first curve (represented by the red line), the goal is, once again, to analyze the behaviour of feature selection –in this case RFE–in eliminating the bigram features. First, we reported the number of bigrams eliminated (as opposed to added in the previous section) at each feature index, then we summed up these numbers. We again repeat the experiment 30 times and we do 3-folds cross-validation at each iteration. Therefore, we averaged the summation over 90 to check the probability of eliminating (as opposed to adding) bigram features at each iteration. Next, we calculate the remaining bigrams by subtracting them from 345, since we have 345 bigrams in total. As usual, the second curve (the blue line) represents probability of eliminating bigram features by RFE at each iteration. Since the total number of features is 4123 features and the number of non-bigram features is 11, the remaining are 4112 bigram features. Therefore, the probability of eliminating a bigram feature is 4112/4123 at each feature index, and the expected cumulative number of features eliminated is 4112/4123 multiplied by the feature index. Then, we subtract them from 350 since we have 350 features after the CFS. The third graph (the green line) once again represents the probability of eliminating bigram features at random after the CFS. After applying CFS, we have 350 features in total, 345 features are bigram features and only 5 are non-bigram features. Therefore, the probability of eliminating a bigram feature is 345/350 and the expected cumulative number of
bi-grams eliminated at each index is 345/350 times the feature index. Therefore, we subtract the resulting probability from 345. The first figure gives us an indication about the RFA performance compared to random selection before and after applying CFS.

(b) **The number of eliminated Non-bigrams compared to random and CFS with random selection**

As discussed in Section 5.5 (b), this graph consists of three curves related to non-bigram features. However, in RFE we are dealing with elimination instead of addition. The goal of this graph is to analyze the behaviour of RFE in eliminating non-bigram features. The difference in generating the first curve (the red line) is that since we have 5 non-bigram features, then we subtract the resulting probability from 5. The second curve (the blue line) represents eliminating non-bigram features at random before applying the CFS. The only difference in generating this curve with its corresponding graph in RFA is that we subtract the resulting probability from 1. The third graph (the green line) represents the probability of eliminating non-bigram features at random after applying CFS. The only difference from its corresponding graph in RFA is that we subtract the resulting value from 5.

(c) **Zoomed number of eliminated Non-bigrams compared to random and CFS with random selection**

The third graph is just a zoomed version of the second graph for only the first 60 features. This graph gives a closer picture on how the non-bigram features are being eliminated in the first 60 iterations.

We generated the above three figures for all the ISCX data sets after applying RFE. In Figure 5.9, we show the RFE behaviour in eliminating bigram and non-bigram features with ISCX 25 examples.
Figure 5.9: RFE bigram and non-bigram elimination on ISCX data set 25 examples compared to random selection

We can clearly notice that there is no difference between RFE in eliminating bigrams, random elimination and random elimination after CFS as shown in Figure 5.9a where all the curves are conforming with each other all the time. That consistency means the probability of RFE eliminating bigram features, the probability of randomly eliminating bigram features and the probability of eliminating random bigram features after CFS are almost the same. However, the situation is different when we analyze the results in a different way and look...
at the results from a different angle. In Figure 5.9b, eliminating non-bigram features by RFE is illustrated (the red line) and compared to random elimination before and after CFS. It can be noticed that the non-bigram features (which are 5 only), that the probability of eliminating non-bigram features starts decreasing consistently after the first iteration until iteration 20 as the red curve descends from 5 to 4. Then, it remains at the same level until iteration 57. This pattern repeats itself between iteration 56 until iteration 70, and between 213 until iteration 234. Then it decreases until the end. When compared to random non-bigram elimination (the blue line), RFE tends to eliminate non-bigram features faster than random elimination as the red curve reaches 0 by the end of 350 iterations while the blue curve is still at 4. Moreover, RFE eliminates non-bigram features earlier than the random non-bigram elimination after CFS (the green line) until iteration 72, when the RFE tends to keep non-bigram features more than random non-bigram elimination after CFS until the end as we can see the red curve stays on the top of the green curve after iteration 72 until the end.

By zooming in to the second graph, we can notice that RFE is eliminating the non-bigram features in the first 60 iterations faster than the random elimination (the blue line) and faster than random elimination after CFS as shown in Figure 5.9c where the red curve is below both the green and blue curves until iteration 60.

Figure 5.10 shows the results of RFE on ISCX data set with 50 examples only. Again, the pattern is the same in eliminating bigram features as shown in Figure 5.10a. We can notice that there is no difference between the probability of RFE eliminating bigram features, random selection and random selection after CFS as the three curves conform consistently all the time. On the other hand, eliminating non-bigrams is different, as the RFE starts slowly to eliminate non-bigrams, especially after iteration 20 and stays slow until the end as shown in Figure 5.10b. In general, the RFE tends to remove non-bigram features earlier than random elimination until the end, and earlier than random elimination after CFS until iteration 252.
By zooming in to the first 60 iterations, we can notice that RFE eliminates non-bigram features much earlier than random elimination (the blue line) and random elimination after CFS (the green line) as shown in Figure 5.10c.

Figure 5.10: RFE bigram and non-bigram elimination on ISCX data set 50 examples compared to random selection

The third generated data set is ISCX 100 examples. Figure 5.11 shows the results of RFE and random elimination for bigram features. Again we can notice from Figure 5.11a that there is no difference between RFE, random elimination and random elimination after CFS in eliminating bigram features. With the non-bigram elimination, RFE quickly starts
eliminating them until iteration 22 when it stays the same until iteration 42. After that it continues to eliminate non-bigram features very slowly until the end. In this size of data set, RFE also tends to eliminate non-bigram features earlier than random elimination always and earlier than random elimination after CFS until iteration 255 when the later precedes RFE in eliminating non-bigram features as shown in Figure 5.11b. By zooming in to the first 60 iterations, we can notice that RFE eliminates non-bigram features much earlier than random elimination (the blue line) and random elimination after CFS (the green line) even when RFE stays in a steady state as shown in Figure 5.10c.

The last data set is the ISCX 500 examples. In Figure 5.12 we show the results of RFE bigram and non-bigram elimination on ISCX data set with 500 examples.

Same as with previous data sets, the RFE, the random bigram elimination and random elimination after CFS did not differ from each other in eliminating bigram features as shown in Figure 5.12a.

For the non-bigram elimination, it is interesting to notice that the RFE (the red line) tends to eliminate non-bigram features more slowly than random elimination after CFS (green line) especially after iteration 55. In other words, RFE here tends to keep non-bigram features as long as possible. However, RFE did not precede random elimination (the blue line) in keeping non-bigram features as shown in Figure 5.12b.

By zooming in to the first 60 iterations, we can notice that at the beginning RFE (the red line) eliminates non-bigram features earlier than random elimination (the blue line) and random elimination after CFS (the green line) until iteration 55 as stated above when RFE becomes slower than random elimination after CFS as shown in Figure 5.12c.

We also measured detection rate, false alarm rate, and the proposed overall metric for all the data sets after applying both RFA and RFE. In Table 5.8 we reported the maximum for all the metrics, except for the false alarm rate where we reported the minimum, for both methods and for all the ISCX data sets.
Table 5.8: All the performance metrics after applying both RFA and RFE on the ISCX data sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>RFA</th>
<th>RFE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACC</td>
<td>F-measure</td>
</tr>
<tr>
<td>25 examples</td>
<td>77.6%</td>
<td>77.0%</td>
</tr>
<tr>
<td>50 examples</td>
<td>88.6%</td>
<td>88.6%</td>
</tr>
<tr>
<td>100 examples</td>
<td>88.8%</td>
<td>88.7%</td>
</tr>
<tr>
<td>500 examples</td>
<td>92.9%</td>
<td>92.9%</td>
</tr>
</tbody>
</table>


Figure 5.11: RFE bigram and non-bigram elimination on ISCX data set 100 examples compared to random selection

From the Table 5.8, we can notice how RFA is outperforming RFE in accuracy, $F$-measure and detection rate. However, in the false alarm rate RFE outperformed RFA in the 500 examples data set, but when we take the overall performance metric, we can notice that RFA is outperforming RFE consistently in all the data sets.
Figure 5.12: RFE bigram and non-bigram elimination on ISCX data set 500 examples compared to random selection

5.7 Distribution of the performance of RFA and RFE

In order to have a better picture about the distribution of the performance of RFA and RFE, we generated the box plot for their metrics. A box-plot is a visual way of showing the distribution of data using five numbers summary: minimum, first quartile, median, third quartile, and maximum. The box-plot is formed from the ordered data points. Then these ordered points are split into four groups (25% of all the data points are placed in each
The virtual lines that divide the groups are called quartiles. Usually a box-plot has four quartiles labelled from the bottom. In the simplest box-plot the central rectangle ranges from the first quartile to the third quartile (also called the interquartile range or IQR). The horizontal line that splits the rectangle represents the median, and the upper and lower whiskers represent the maximum and minimum respectively. However, in some cases, a data set may have high maximums or low minimums called outliers. These outliers could be either $3 \times IQR$ or more above the third quartile or $3 \times IQR$ below the first quartile.

The simplest box-plot with its main components is shown in Figure 5.13.

![Figure 5.13: The simplest box-plot with its five components](image-url)
We generated the box-plot for each of the generated ISCX data set for the three metrics: detection rate, false alarm rate and overall metric. Figure 5.14 shows the box-plot of all the three metrics for RFA and RFE for the ISCX 25 data set for both RFA and RFE.

(a) Detection rate  
(b) False Alarm Rate  
(c) Overall metric

Figure 5.14: Boxplot for ISCX 25 (a) Detection rate (b) False Alarm Rate (c) Overall
Figure 5.14a shows the box-plot for the ISCX 25 data set for both methods RFA and RFE. It can be clearly noticed that RFA is generating higher detection rate than RFE since the red box with its both whiskers is above blue box as shown in Figure 5.14a.

Figure 5.14b shows the false alarm rate box-plot for the ISCX 25 data set for both methods RFA and RFE. As it is already known that this metric represents the percentage of the false alarms that the system generates. Therefore, the goal is to minimize this metric as much as possible. In this metric, RFA has proved its superiority on RFE as shown in Figure 5.14b, where the red box (which represents RFA) is below the blue box (which represents RFE).

For the overall metric, Figure 5.14c shows the box-plot for ISCX 25 data set for both methods RFA and RFE for this metric.

We also generated the box-plot for each of the generated ISCX 50 data set for the three metrics: detection rate, false alarm rate and overall metric. Figure 5.15 shows the box-plot of the three metrics for RFA and RFE.

As it can be noticed from Figure 5.15a, RFA generated higher detection rate than RFE for the ISCX 50 data set since the entire red box is above the blue box.

However, for the false alarm rate, RFA performed poorly in this metric on ISCX 50 data set. It can be noticed from Figure 5.15b that RFA generated higher false alarm rate than RFE as the entire box-plot of RFA represented by red box (which is shown in the figure as a thick line since the data is not distributed enough around the median) is above the box-plot of RFE represented by blue box. This result matches the second row in Table 5.10 as RFE outperformed RFA on false alarm rate for the ISCX 50 data set.

However, for the overall metric, RFA showed clear superiority over RFE as it is shown in Figure 5.15c where the red box is much above the blue box.

We also generated the box-plot for the ISCX 100 data set for the three metrics: detection rate, false alarm rate and overall metric for both RFA and RFE. Figure 5.16 shows the box-plot of the detection rate for RFA and RFE.
From the Figure 5.16a, it is clear that RFA generated higher detection rate than RFE as the red box is above the blue box and the median of RFA is about 84% while the median of RFE is about 80%.
Figure 5.16: Boxplot for ISCX 100 (a) detection rate (b) False Alarm Rate (c) Overall

For the false alarm rate, again RFA performed poorly on this metric as shown in Figure 5.16b as the red box is over the blue box and the RFA median is 6.5% while the median of RFE is 5%. This result also matches the result shown in row 3 of Table 5.10.
The third metric, the overall metric, RFA outperformed RFE as the median of RFA was around 79% while the median of RFE was around 77% and the whole red box is above the blue box as shown in Figure 5.16c.

The last data set is the ISCX 500 data set. We also generated the box-plot for this data set for the three metrics: detection rate, false alarm rate and overall metric on both methods RFA and RFE. Figure 5.17 shows the box-plot of the three metrics for RFA and RFE.

For the detection rate, again RFA outperformed RFE in this metric as shown in Figure 5.17a where the entire red box is above the blue box and the median of RFA is around 89% while the median of RFE is around 88%.

In this data set, the pattern of RFA in generating false alarm has improved. As we can notice that there is a full overlap between the red box and the blue box, where entire blue box (which is the smaller) overlaps with the red box (which is bigger) as shown in Figure 5.17b. Although the median of RFA and RFE was almost the same, RFA was statistically better than RFE as shown in row 4 of Table 5.10.

We also generated the box plot for the last metric, the overall metric. Figure 5.17c shows the box-plot of the overall metric for ISCX 500 data set for both methods RFA and RFE. RFA showed its superiority on RFE in this metric by producing higher overall metric than RFE as almost the entire red box is above the blue box and there is a clear superiority from the side of RFA in terms of the median of the overall metric.

5.8 Statistical Analysis

Since the results of RFA and RFE were sometimes close to each other, we decided to analyze the results statistically. We measured the statistical significance of the RFA and RFE results using the Mann-Whitney U test. The statistical test involved both accuracy and $F$-measure of both methods RFA and RFE. We set the same null and alternative hypotheses that we explained in Section 4.5 for accuracy, $F$-measure, detection rate and
Figure 5.17: Boxplot for ISCX 500 (a) detection rate (b) False Alarm Rate (c) Overall metric. However, stating the hypothesis for false alarm rate is different since for this metric the lower the value, the better the performance. Stating the hypothesis for false alarm rate is explained as follows:

**Step 1:** State the hypothesis and identify the claim
$H_0$: The false alarm rate of RFE is less than or equal to the false alarm rate of RFA

$H_1$: The false alarm rate of RFE is greater than the false alarm rate of RFA

**Step 2:** Apply the unpaired test on the results of RFA and RFE and obtain the $p$-value to draw a conclusion. If the $p$-value is less than 0.05, then we reject the $H_0$ and accept the $H_1$. Otherwise, we cannot reject any hypothesis as we do not have enough evidence to make a decision. In such cases, we get $p$-value greater than 0.05, which indicates that there is not enough evidence that RFA is superior to RFE.

**Step 3:** If the null hypothesis $H_0$ is not rejected, then no conclusion is drawn. Therefore, we decided to apply a second Mann-Whitney U test in order to try to draw a different conclusion: that RFE is superior to RFA. To do this we reverse the hypothesis.

$H_0$: The false alarm rate of RFA is less than or equal to the false alarm rate of RFE

$H_1$: The false alarm rate of RFA is greater than the false alarm rate of RFE

**Step 4:** Apply the unpaired test on the second version of the null-hypothesis and obtain the $p$-value to make the decision.

Therefore, we have three different cases in total when we apply the two tests:

**Case 1:** $p$-value is less than 0.05, RFA is superior to RFE in terms of False Alarm Rate

**Case 2:** $p$-value is greater than or equal to 0.05, no conclusion. Therefore, to check if RFE is superior to RFA (in terms of false alarm rate), we reversed the $H_0$ hypothesis to be the false alarm rate of RFA is less than or equal to the false alarm rate of RFE, if the $p$-value is less than 0.05, RFE is superior to RFA. If neither test returns a significant result, we have Case 3

**Case 3:** when we there is not enough evidence that RFA is superior to RFE or vice versa, each $p$-value is greater than 0.05 in both directions. We draw no conclusion in this case.
The results of the statistical test for accuracy and $F$-measure for RFA and RFE are shown in Table 5.9.

Table 5.9: Mann-whitney U test for RFA and RFE results on the ISCX data sets using both metrics Accuracy and $F$-measure

<table>
<thead>
<tr>
<th>Data set</th>
<th>Test decision (Accuracy)</th>
<th>Test decision ($F$-measure)</th>
</tr>
</thead>
<tbody>
<tr>
<td>25 examples</td>
<td>RFA&gt;RFE</td>
<td>RFA&gt;RFE</td>
</tr>
<tr>
<td>50 examples</td>
<td>RFA&gt;RFE</td>
<td>RFA&gt;RFE</td>
</tr>
<tr>
<td>100 examples</td>
<td>RFA&gt;RFE</td>
<td>RFA&gt;RFE</td>
</tr>
<tr>
<td>500 examples</td>
<td>RFA&gt;RFE</td>
<td>RFA&gt;RFE</td>
</tr>
</tbody>
</table>

As it can be noticed from the Table 5.9, the superior performance of RFA was statistically significant on all the generated ISCX data sets. In all of the above tests we obtained $p$-values less than 0.05, therefore, the conclusion was that RFA is better than RFE in these metrics.

In addition, we applied the same statistical test on the other metrics that we added for intrusion detection (detection rate, false alarm rate and the overall metric). The results of the Mann-Whitney U test for those three metrics of RFA and RFE are shown in Table 5.10.

Table 5.10: Mann-whitney U test for RFA and RFE results on the ISCX data sets using Detection rate, False Alarm Rate and overall metric

<table>
<thead>
<tr>
<th>Data set</th>
<th>Test decision (Detection rate)</th>
<th>Test decision (False Alarm Rate)</th>
<th>Test decision (Overall)</th>
</tr>
</thead>
<tbody>
<tr>
<td>25 examples</td>
<td>RFA&gt;RFE</td>
<td>RFA&lt;RFE</td>
<td>RFA&gt;RFE</td>
</tr>
<tr>
<td>50 examples</td>
<td>RFA&gt;RFE</td>
<td>RFE&lt;RFA</td>
<td>RFA&gt;RFE</td>
</tr>
<tr>
<td>100 examples</td>
<td>RFA&gt;RFE</td>
<td>RFE&lt;RFA</td>
<td>RFA&gt;RFE</td>
</tr>
<tr>
<td>500 examples</td>
<td>RFA&gt;RFE</td>
<td>RFA&lt;RFE</td>
<td>RFA&gt;RFE</td>
</tr>
</tbody>
</table>
From the Table 5.10, it can be noticed that RFA outperformed RFE statistically in the detection rate metric for all the ISCX data sets. Regarding the second metric, false alarm rate, RFA outperformed RFE by producing less false alarms in 25 and 500 examples, while RFE outperformed RFA in the 50 and 100 examples by producing less false alarms. However, by looking at the overall metric, we can notice that RFA outperformed RFE statistically in the overall performance metric for all the ISCX data sets.

5.9 New ranking coefficients for RFA

As stated in Chapter 3, the ranking coefficients that we use with RFA depends on selecting the feature that causes the maximum change in the objective function. According to Equation 3.10, the feature that corresponds to the maximum value of \( DJ \) is chosen and added to the set of selected features. However, that ranking coefficient depends on one measure which is the change of the objective function \( DJ \). In order to make the ranking coefficient depend on more than one source in selecting the features, we propose the following forms of ranking coefficients:

(a) **Adding the \( F \)-measure to the original ranking coefficient \( DJ \)**

This proposed ranking coefficient depends on calculating the \( F \)-measure \( FM_j \) after adding the feature \( j \) and adding it to the original ranking coefficient \( DJ_j \) as shown in Equation 5.5.

\[
R_j = FM_j + DJ_j \tag{5.5}
\]

where \( R_j \) is the new ranking coefficient. The feature that corresponds to the maximum \( R_j \) is selected and added to the set of selected features.

(b) **Multiplying the \( F \)-measure by the original ranking coefficient \( DJ \)**

The other proposed ranking coefficient is similar to the previous one except it multiplies the \( F \)-measure by \( DJ \) instead of adding them as shown in Equation 5.6.
\[ R_j = FM_j \times DJ_j \]  
(5.6)

Same as the previous ranking coefficient, the feature that has the maximum \( R_j \) is selected.

(c) **Adding 70% of the F-measure to 30% of the original ranking coefficient \( DJ \)**

The third proposed ranking coefficient is based on 70%:30% split of the F-measure and original ranking coefficient \( DJ \) as shown in Equation 5.7.

\[ R_j = (0.7 \times FM_j) + (0.3 \times DJ_j) \]  
(5.7)

Again, the feature that gains the maximum \( R_j \) is chosen and added to the set of selected features.

(d) **Taking F-measure only instead of the original ranking coefficient \( DJ \)**

We also opted to choose the F-measure only without the original ranking coefficient as shown in Equation 5.8.

\[ R_j = FM_j \]  
(5.8)

In this case, the feature that generates the maximum F-measure is selected and added to set of selected features.

For simplicity, from now on, we will use a short term for each of the above ranking coefficients. We will refer to (a) as FM+R, (b) as FM*R, (c) as FM only, (d) as FM perc R respectively.

In order to test the above proposed ranking coefficients, we used each one of them with RFA on all of the ISCX data sets. We repeated the experiment for 30 times, we took the average and reported the maximum obtained accuracy from each one as shown in Table 5.11.
Table 5.11: Maximum obtained accuracy by each of the proposed ranking coefficient and the original one for all the ISCX data sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISCX 25</td>
<td>77.03</td>
<td>77.65</td>
<td>77.26</td>
<td>75.80</td>
<td>77.65</td>
</tr>
<tr>
<td>ISCX 50</td>
<td>88.80</td>
<td>88.62</td>
<td>88.25</td>
<td>88.50</td>
<td>88.62</td>
</tr>
<tr>
<td>ISCX 100</td>
<td>88.77</td>
<td>88.89</td>
<td>88.57</td>
<td>88.80</td>
<td>88.80</td>
</tr>
<tr>
<td>ISCX 500</td>
<td>92.99</td>
<td>92.92</td>
<td>92.59</td>
<td>92.89</td>
<td>92.89</td>
</tr>
</tbody>
</table>

As it can be noticed from the Table 5.11, that FM*R and the original coefficients outperformed the other ones on the ISCX 25 data set by producing the maximum accuracy. With the ISCX 50 data set, FM+R outperformed the other ranking coefficients, while FM*R and the original coefficient performed equally on this data set. With the ISCX 100 data set, the differences between the ranking coefficients have become very small. Although the FM*R outperformed the other rankings, the other ones are also close to this ranking especially the FM perc R and the original one. With the last data, ISCX 500 data set, FM+R outperformed the other coefficients in the maximum accuracy while FM perc R and the original one were equal in this metric but less than FM+R and FM*R coefficients.

In addition to the accuracy, we measured the maximum obtained $F$-measure also for all the data sets and for all the proposed and original ranking coefficients as shown in Table 5.12.

Table 5.12: Maximum obtained $F$-measure by each of the proposed ranking coefficients and the original one for all the ISCX data sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISCX 25</td>
<td>76.31</td>
<td>77.06</td>
<td>76.56</td>
<td>75.07</td>
<td>77.06</td>
</tr>
<tr>
<td>ISCX 50</td>
<td>88.80</td>
<td>88.66</td>
<td>88.29</td>
<td>88.53</td>
<td>88.66</td>
</tr>
<tr>
<td>ISCX 100</td>
<td>88.75</td>
<td>88.88</td>
<td>88.56</td>
<td>88.78</td>
<td>88.78</td>
</tr>
<tr>
<td>ISCX 500</td>
<td>92.98</td>
<td>92.90</td>
<td>92.58</td>
<td>92.88</td>
<td>92.88</td>
</tr>
</tbody>
</table>
It can be noticed from Tables 5.11 and 5.12 that the new proposed ranking coefficients have also achieved a competitive performance (if not better) especially those defined by FM+R and FM*R. In the ISCX 25 data set, the ranking coefficient defined by FM*R was able to achieve the same performance to the original one in both accuracy and F-measure. In the ISCX 50 data set, the ranking coefficient defined by FM+R achieved slightly better than the other ranking coefficients in both accuracy and F-measure. Again, the ranking coefficient defined by FM*R outperformed the other ones in both accuracy and F-measure in the ISCX 100 data set. Lastly, FM+R ranking coefficient outperformed all the other ones in the ISCX 500 data set in both accuracy and F-measure.

Since this work is applied on intrusion detection data sets, we measured the other three metrics (i.e., detection rate, false alarm rate and the proposed overall metric) for all the proposed ranking coefficients. In order to have a better comparison on the performance of the proposed ranking coefficients, we reported the results of each metric in a separate table. Table 5.13 shows the maximum obtained detection rate by each one of the proposed ranking coefficients with RFA in addition to the original one on all the ISCX data sets.

Table 5.13: Maximum obtained detection rate by each of the proposed ranking coefficients and the original one for all the ISCX data sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISCX 25</td>
<td>71.28</td>
<td>71.54</td>
<td>71.95</td>
<td>69.45</td>
<td>71.54</td>
</tr>
<tr>
<td>ISCX 50</td>
<td>88.84</td>
<td>89.14</td>
<td>88.53</td>
<td>88.87</td>
<td>89.14</td>
</tr>
<tr>
<td>ISCX 100</td>
<td>83.88</td>
<td>83.98</td>
<td>83.67</td>
<td>83.82</td>
<td>83.90</td>
</tr>
<tr>
<td>ISCX 500</td>
<td>89.50</td>
<td>89.30</td>
<td>93.08</td>
<td>89.60</td>
<td>89.61</td>
</tr>
</tbody>
</table>

We can notice from Table 5.13, that the FM only ranking coefficient outperformed the other ranking coefficients, while FM*R and the original one came in the second place and produced the same detection rate on the ISCX 25 data set. For the ISCX 50 data set, FM*R and the original coefficient outperformed the other ranking coefficients. However, with the ISCX 100, the FM*R ranking coefficients outperformed the other ones. On the
last data set, the ISCX 500 data set, the FM ranking coefficients noticeably outperformed the other ones.

The other metric is the false alarm rate which intrusion detection systems aim to minimize as much as possible as the false alarms also threaten these systems’ reliability. Table 5.14 shows the minimum obtained false alarm rate by each one of the proposed ranking coefficients with RFA in addition to the original one on all the ISCX data sets.

Table 5.14: Minimum obtained false alarm rate by each of the proposed ranking coefficients and the original one for all the ISCX data sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISCX 25</td>
<td>10.41</td>
<td>10.01</td>
<td>11.08</td>
<td>10.14</td>
<td>10.01</td>
</tr>
<tr>
<td>ISCX 50</td>
<td>07.97</td>
<td>8.35</td>
<td>10.47</td>
<td>7.97</td>
<td>08.35</td>
</tr>
<tr>
<td>ISCX 100</td>
<td>1.44</td>
<td>2.33</td>
<td>6.35</td>
<td>1.44</td>
<td>01.11</td>
</tr>
<tr>
<td>ISCX 500</td>
<td>0.72</td>
<td>02.73</td>
<td>04.26</td>
<td>0.39</td>
<td>02.63</td>
</tr>
</tbody>
</table>

From Table 5.14, we can notice that for the ISCX 25, FM*R ranking coefficient outperformed all the other ranking coefficients by producing the minimum false alarm rate among the other ranking coefficients. For the ISCX 50 data set, FM+R and FM perc R share the minimum obtained false alarm rate among the other ranking coefficients. The pattern is repeated in the ISCX 100 data set, when FM+R and FM perc R produced the minimum false alarm rate among the other ranking coefficients. The last data set, ISCX 500, FM perc R outperformed the other coefficients noticeably by generating the minimum false alarm rate.

Lastly, we reported the proposed overall metric to compare the overall performance of all the proposed ranking coefficients with RFA in addition to the original one on all the ISCX data sets as shown in Table 5.15.
Table 5.15: Maximum obtained overall metric by each of the proposed and original ranking coefficients for all the ISCX data sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISCX 25</td>
<td>62.61</td>
<td>63.84</td>
<td>62.92</td>
<td>60.98</td>
<td>63.84</td>
</tr>
<tr>
<td>ISCX 50</td>
<td>78.04</td>
<td>77.41</td>
<td>77.08</td>
<td>77.36</td>
<td>77.41</td>
</tr>
<tr>
<td>ISCX 100</td>
<td>80.21</td>
<td>80.67</td>
<td>79.76</td>
<td>80.33</td>
<td>80.45</td>
</tr>
<tr>
<td>ISCX 500</td>
<td>87.65</td>
<td>87.65</td>
<td>86.89</td>
<td>87.60</td>
<td>87.82</td>
</tr>
</tbody>
</table>

From the Table 5.15, it can be noticed that FM*R and the original ranking coefficients produced the maximum overall metric and outperformed the other coefficients on ISCX 25 data set. For the ISCX 50 data set, FM+R outperformed the other ranking coefficients in the overall metric. With the ISCX 100 data set, FM*R produced the maximum overall metric. With the last data set, ISCX 500, the original ranking coefficient outperformed the other ones on the overall metric.

To have a better picture of the performance of all the proposed ranking coefficients beside the original one -as the maximum can be tricky sometimes-, we generated the box-plot of all of them for each metric individually and for each data set. Therefore, we used the box-plot to visually evaluate the distribution of all the proposed and the original ranking coefficients with RFA for the three metrics: detection rate, false alarm rate, and overall metric using each ranking coefficient. In Figure 5.18, the box-plots of all the proposed ranking coefficients in addition to the original one for detection rate on ISCX 25 data set are shown.
Figure 5.18: Box-plot for all of the proposed ranking coefficients in addition to the original one for detection rate for ISCX 25 data set

From the Figure 5.18, we can notice that FM+R, FM*R and original ranking coefficients are close to each other with ISCX 25 data set by generating the highest detection rate. However, the median of FM+R was higher than the median of both FM*R and the original one.
For the false alarm rate, we also generated the box-plots for all the proposed and original ranking coefficients on ISCX data set as shown in Figure 5.19. In this metric, all of the ranking coefficients were close to each other except for the FM only ranking coefficient (represented by pink box) which badly generated the highest false alarm rate. Although,
the median of the other ranking coefficients were close to each other, the box size (spread) of FM+R (represented by red box) was bigger than the others.

Similarly, we generated the box-plots for the overall metric for the ISCX 25 data set as shown in Figure 5.20. It can be noticed from the figure, that the worst overall performance was by FM only ranking coefficient, while FM+R, FM*R and the original one were close to each other ad their median were close too.

Figure 5.20: Box-plot for all of the proposed ranking coefficients in addition to the original one for overall metric for ISCX 25 data set
For the ISCX 50 data set, we also generated the box-plot for all the proposed and original ranking coefficients for detection rate as shown in Figure 5.21. The figure shows that all the ranking coefficients (except FM only) were close to each other in the detection rate and all of them exhibit many outliers.

Figure 5.21: Box-plot for all the proposed ranking coefficients in addition to the original one for detection rate for ISCX 50 data set
We also generated the box-plot for all the proposed and original ranking coefficients for false alarm rate for the same ISCX 50 data set as shown in Figure 5.19. Again the FM only ranking coefficient produced the highest false alarms among the other ranking coefficients which showed substantial convergence so that we could not find which one is the best one.

Figure 5.22: Box-plot for all the proposed ranking coefficients in addition to the original one for false alarm rate for ISCX 50 data set
The overall metric shows the final performance evaluation on each ranking coefficient on ISCX 50 data set as shown in Figure 5.23. This figure shows that all the ranking coefficients generated a median around 77% except the FM only in which the median is about 72%. All of the ranking coefficients exhibit many outliers as with the previous data set.

<table>
<thead>
<tr>
<th>Coefficient</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM+R</td>
<td>0.70</td>
</tr>
<tr>
<td>FM*R</td>
<td>0.72</td>
</tr>
<tr>
<td>FM</td>
<td>0.74</td>
</tr>
<tr>
<td>FM prec R</td>
<td>0.76</td>
</tr>
<tr>
<td>Original</td>
<td>0.78</td>
</tr>
</tbody>
</table>

Figure 5.23: Box-plot for all the proposed ranking coefficients in addition to the original one for overall metric for ISCX 50 data set
We generated the same figures above for the third data set, ISCX 100 data set. Figure 5.24 shows the box-plot of all the proposed and original ranking coefficients for detection rate on ISCX 100 data set.

Figure 5.24: Box-plot for all the proposed ranking coefficients in addition to the original one for detection rate on ISCX 100 data set
As shown in Figure 5.24, the FM ranking coefficient showed the worst performance among the other ones. However, among the other four ranking coefficients, FM*R showed a slightly better performance than the others by producing a higher median than the others.

For the false alarm rate, we generated also the box-plot for all the proposed and generated ranking coefficients on ISCX 100 data set as shown in Figure 5.25. Again the pattern is repeated with the false alarm rate as the FM only ranking coefficient producing the highest false alarms, while the others were very close to each other.
Figure 5.25: Box-plot for all the proposed ranking coefficients in addition to the original one for false alarm rate on ISCX 100 data set

Same as with the other data sets, we generated the box-plot for the overall metric for all the ranking coefficients on the ISCX 100 data set as shown in Figure 5.26
It can be noticed from the Figure 5.26, that the FM only ranking coefficient generated the worst overall performance among the other coefficients with median less than 75% while all the other ones have medians close to 80% but in general all of them have many outliers.

For the last data set, ISCX 500 data set, we generated the box-plot for all the ranking coefficients for the detection rate as shown in Figure 5.27.
Figure 5.27: Box-plot for all the proposed ranking coefficients in addition to the original one for detection rate on ISCX 500 data set

As with the ISCX 100 data set, all of the ranking coefficients except FM only generated close detection rates, but with a median around 90% while the FM only produced a median around 80%. We also generated the box-plot for all the ranking coefficients for the false alarm rate on the ISCX 500 data set as shown in Figure 5.28. Similar to the previous data
sets, FM only also produced the worst overall performance among the other ones with a median around 13% while the other were close to each other with a median less than 5%.

![Box-plot for all the proposed ranking coefficients in addition to the original one for false alarm rate on ISCX 500 data set](image)

Figure 5.28: Box-plot for all the proposed ranking coefficients in addition to the original one for false alarm rate on ISCX 500 data set

For the third metric, the overall metric, we generated the box-plot also for all the ranking coefficients with ISCX 500 data set as shown in Figure 5.29.
Figure 5.29: Box-plot for all the proposed ranking coefficients in addition to the original one for the overall metric on ISCX 500 data set

The same pattern is repeated with the overall metric since all the ranking coefficients (except the FM only) generated overall performance with a median close to 87% while the FM only coefficient produced a median close to 69%.

From all of the previous figures we noticed that there are some cases where it is difficult to infer which ranking coefficient is better (if any) as they showed a very close performance...
sometimes. Therefore, we conducted a comprehensive pair-wise statistical test on all the proposed ranking coefficients in addition to the original one with RFA on the three metrics: detection rate, false alarm rate, and overall metric and for all the ISCX data sets. The alternative hypothesis on each row $i$ represents that ranking coefficients $i$ is greater than the ranking coefficients on each column $j$. Each cell in the table represents the intersection between ranking coefficients $i$ and ranking coefficient $j$. We left the intersection of each ranking coefficients with itself ($i = j$) empty as there is no importance in comparing the results of a ranking coefficient with itself. In each relevant cell $C_{i,j}$, we put the result of the statistical test as “T” when there is enough evidence that the ranking coefficient $i$ is greater than the result of the ranking coefficient $j$. In addition, we provided the $p$-value of each statistical test, and when the $p$-value is less than 0.05 the result is True, otherwise it is False. We also coloured each cell with green when the result is True, coloured it red when its converse was True, and left it un-coloured when it is False (i.e., no significant result is obtained). Table 5.16 shows the results of Mann-Whitney test on all the ranking coefficients with RFA for ISCX 25 data set on detection rate.
We can notice from the Table 5.16 that \( DR_{FM+R} \) outperformed both \( DR_{FM only} \) and \( DR_{FM perc R} \) as it can be noticed the reported p-values in these two cases are less than 0.05. The same pattern is repeated with \( DR_{FM*R} \) which outperformed both \( DR_{FM only} \) and \( DR_{FM perc R} \). However, none of the \( DR_{FM perc R} \) and \( DR_{FM perc R} \) were able to outperform any of the other ranking coefficients. With the \( DR_{original} \), it showed its superiority over both \( DR_{FM only} \) and \( DR_{FM perc R} \).

In order to know the order of the ranking coefficients, we extracted the partial ordering of all the ranking coefficients on detection rate for the ISCX 25 data set as shown in Figure 5.30. We can notice that each of the FM+R, FM*R and original coefficients are the same in their superiority to both FM only and FM perc R. Therefore, selecting any of the best three will give the same results with this data set and with detection rate.

For the ISCX 50 data set, Table 5.17 shows the results of Mann-Whitney test on all the ranking coefficients with RFA for the detection rate.
Figure 5.30: Partial ordering of the ranking coefficients with RFA on detection rate for the ISCX 25 data set

Table 5.17: Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 50 data set on detection rate

<table>
<thead>
<tr>
<th></th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
</table>
| FM+R   | -      | DR_{FM+R} > DR_{FM+R:R}  
  p=1    |         |         |          |
| FM*R   | DR_{FM+R} > DR_{FM+R:T}  
  p=5.5e^{-11} | -      | DR_{FM+R} > DR_{FM+R:T}  
  p=1.3e^{-81} |         |
| FM only| DR_{FM only} > DR_{FM+R:F}  
  p=1    | DR_{FM only} > DR_{FM+R:F}  
  p=1    | DR_{FM perc R} > DR_{FM perc R:T}  
  p=8.2e^{-11} | DR_{FM perc R} > DR_{FM perc R:T}  
  p=0.5  |
| FM perc R | DR_{FM perc R} > DR_{FM perc R:T}  
  p=0.43 | DR_{FM perc R} > DR_{FM perc R:T}  
  p=0.43 | DR_{FM perc R} > DR_{FM perc R:T}  
  p=1    | DR_{FM perc R} > DR_{FM perc R:T}  
  p=1    |
| original| DR_{original} > DR_{original:R}  
  p=5.03e^{-11} | DR_{original} > DR_{original:R}  
  p=0.5  | DR_{original} > DR_{original:T}  
  p=1.3e^{-84} | DR_{original} > DR_{original:T}  
  p=8.2e^{-11} | -        |

The results in the Table 5.17 are interesting. The first ranking coefficient $DR_{FM+R}$ was statistically better than $DR_{FM only}$. The second ranking coefficient, $DR_{FM+R}$, was statistically better than all the others except the $DR_{original}$. The $DR_{FM only}$ was not able to outperform any of the other ranking coefficients as we can notice that all the resulting $p$-values were 1’s. The $DR_{FM perc R}$ was statistically better than $DR_{FM only}$. The last one, $DR_{original}$ was statistically better than all the other ranking coefficients except $DR_{FM+R}$.  

155
The partial ordering of the ranking coefficients for the ISCX 50 data set shows that both original and FM*R ranking coefficients share the same level in their superiority on all the other ranking coefficients. In the second level, we find both FM+R and FM perc R, while FM only comes in the last level since it did not outperform any of the ranking coefficients as shown in Figure 5.31.

Figure 5.31: Partial ordering of the ranking coefficients with RFA on detection rate for the ISCX 50 data set

Table 5.18 shows the results of the Mann-Whitney test on all the ranking coefficients with RFA for ISCX 100 data set on detection rate.
As can be noticed from Table 5.18, $DR_{FM+R}$ was statistically more significant than $DR_{FM\ only}$ only. The second ranking coefficient, $DR_{FM*R}$ was more statistically significant than all the other ranking coefficients. Again, the $DR_{FM\ only}$ did not show any statistical significance over any of the other ranking coefficients. The $DR_{FM\ perc\ R}$ showed its statistical significance over $DR_{FM\ only}$ only. The last one, $DR_{original}$, was statistically more significant than all the ranking coefficients except $DR_{FM*R}$.

The partial ordering of the ranking coefficients on detection rate for the ISCX 100 data set shows FM*R ranking coefficient outperformed all the other ranking coefficients, while the original one came in the second level. In the third level, both FM+R and FM perc R shared the same performance, while FM only came in the last level.

Table 5.19 shows the results of Mann-Whitney test on all the ranking coefficients with RFA for ISCX 500 data set on detection rate.
Figure 5.32: Partial ordering of the ranking coefficients with RFA on detection rate for the ISCX 100 data set

Table 5.19: Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 500 data set on detection rate

<table>
<thead>
<tr>
<th></th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM+R</td>
<td>-</td>
<td>$DR_{FM+R} &gt;$</td>
<td>$DR_{FM+R} &gt;$</td>
<td>$DR_{FM+R} &gt;$</td>
<td>$DR_{FM+R} &gt;$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$DR_{FM+R:T}$</td>
<td>$DR_{FM only:T}$</td>
<td>$DR_{FM perc R:F}$</td>
<td>$DR_{original:F}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>p=8.2$^{-14}$</td>
<td>p=6.9–92</td>
<td>p=0.24</td>
<td>p=1</td>
</tr>
<tr>
<td>FM*R</td>
<td>$DR_{FM+R} &gt;$</td>
<td>-</td>
<td>$DR_{FM+R} &gt;$</td>
<td>$DR_{FM+R} &gt;$</td>
<td>$DR_{FM+R} &gt;$</td>
</tr>
<tr>
<td></td>
<td>$DR_{FM+R:F}$</td>
<td></td>
<td>$DR_{FM only:T}$</td>
<td>$DR_{FM perc R:F}$</td>
<td>$DR_{original:F}$</td>
</tr>
<tr>
<td></td>
<td>p=1</td>
<td></td>
<td>p=2.5$^{-91}$</td>
<td>p=1</td>
<td>p=1</td>
</tr>
<tr>
<td>FM only</td>
<td>$DR_{FM only}$ &gt;</td>
<td>$DR_{FM only}$ &gt;</td>
<td>-</td>
<td>$DR_{FM only}$ &gt;</td>
<td>$DR_{FM only}$ &gt;</td>
</tr>
<tr>
<td></td>
<td>$DR_{FM+R:F}$</td>
<td>$DR_{FM+R:F}$</td>
<td></td>
<td>$DR_{FM perc R:F}$</td>
<td>$DR_{original:F}$</td>
</tr>
<tr>
<td></td>
<td>p=1</td>
<td>p=1</td>
<td></td>
<td>p=1</td>
<td>p=1</td>
</tr>
<tr>
<td>FM perc R</td>
<td>$DR_{FM perc R}$ &gt;</td>
<td>$DR_{FM perc R}$ &gt;</td>
<td>$DR_{FM perc R}$ &gt;</td>
<td>-</td>
<td>$DR_{FM perc R}$ &gt;</td>
</tr>
<tr>
<td></td>
<td>$DR_{FM+R:T}$</td>
<td>$DR_{FM+R:T}$</td>
<td>$DR_{FM only:T}$</td>
<td></td>
<td>$DR_{original:T}$</td>
</tr>
<tr>
<td></td>
<td>p=7.7$^{-11}$</td>
<td>p=2.5$^{-27}$</td>
<td>p=1.9$^{-92}$</td>
<td></td>
<td>p=0.311</td>
</tr>
<tr>
<td>original</td>
<td>$DR_{original}$ &gt;</td>
<td>$DR_{original}$ &gt;</td>
<td>$DR_{original}$ &gt;</td>
<td>$DR_{original}$ &gt;</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>$DR_{FM+R:T}$</td>
<td>$DR_{FM+R:T}$</td>
<td>$DR_{FM only:T}$</td>
<td></td>
<td>$DR_{FM perc R}$ &gt;</td>
</tr>
<tr>
<td></td>
<td>p=1.5$^{-8}$</td>
<td>p=1.2$^{-26}$</td>
<td>p=4.1$^{-93}$</td>
<td></td>
<td>p=0.68</td>
</tr>
</tbody>
</table>
The partial ordering of the ranking coefficients on detection rate for the ISCX 500 data set shows that both original and FM perc R share the highest level as they both outperform the other three ones. FM+R came in the second place as it outperformed both FM*R and FM only, while FM*R came in the third place as it only outperformed FM only coefficient. As with the previous data sets, FM only came in the last place as shown in Figure 5.33.

Figure 5.33: Partial ordering of the ranking coefficients with RFA on detection rate for the ISCX 500 data set

The second metric that we tested with RFA is the false alarm rate. The goal always is to minimize this metric as much as possible. Therefore, we applied the statistical test on the results of all the ranking coefficients with RFA to detect which one generates minimum false alarm rate among the others. Accordingly, we changed the alternative hypothesis in testing this metric to be false alarm rate generated by ranking coefficient from row $i$ is less than the one generated by ranking coefficient from column $j$. Table 5.20 shows the results of Mann-Whitney test on all the ranking coefficients with RFA for ISCX 25 data set on false alarm rate.
Table 5.20: Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 25 data set on false alarm rate

<table>
<thead>
<tr>
<th></th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM+R</td>
<td>-</td>
<td>FM+R &gt; FARFM+R? p=0.16</td>
<td>FM only &gt; FARFM+R? p=1.04\textsuperscript{-92}</td>
<td>FM perc R &gt; FARFM+R? p=0.25</td>
<td>original &gt; FARFM+R? p=0.16</td>
</tr>
<tr>
<td>FM*R</td>
<td>FM+R &gt; FARFM+R? p=0.83</td>
<td>-</td>
<td>FARFM+R &gt; FARFM only? p=1.02\textsuperscript{-95}</td>
<td>FARFM+R &gt; FARFM perc R? p=0.54</td>
<td>FAR original? p=0.5</td>
</tr>
<tr>
<td>FM only</td>
<td>FARFM only &gt; FARFM+R? p=1</td>
<td>FARFM only &gt; FARFM perc R? p=1</td>
<td>-</td>
<td>FARFM+R &gt; FARFM perc R? p=1</td>
<td>FAR original? p=1</td>
</tr>
<tr>
<td>FM perc R</td>
<td>FARFM perc R &gt; FARFM+R? p=0.74</td>
<td>FARFM perc R &gt; FARFM+R? p=0.46</td>
<td>FARFM perc R &gt; FARFM+R? p=1.01\textsuperscript{-91}</td>
<td>-</td>
<td>FARFM perc R? p=0.46</td>
</tr>
<tr>
<td>original</td>
<td>FARFM perc R? p=0.83</td>
<td>FARFM perc R? p=0.5</td>
<td>FARFM perc R? p=1.02\textsuperscript{-95}</td>
<td>FARFM perc R? p=0.54</td>
<td>-</td>
</tr>
</tbody>
</table>

Figure 5.34: Partial ordering of the ranking coefficients with RFA on false alarm rate for the ISCX 25 data set

The partial ordering of the ranking coefficients with RFA on false alarm rate for the ISCX 25 data set shows that FM*R, FM+R, FM perc R and original ranking coefficients share the first place and they all have better performance than FM only as shown in Figure 5.34.
For the second data set, ISCX 50, Table 5.21 shows the results of Mann-Whitney test on all the ranking coefficients with RFA for ISCX 50 data set on false alarm rate.

Table 5.21: Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 50 data set on false alarm rate

<table>
<thead>
<tr>
<th></th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM+R</td>
<td>-</td>
<td>$FAR_{FM+R} &gt; FAR_{FM*R}$</td>
<td>$FAR_{FM+R} &gt; FAR_{FM only,R}$</td>
<td>$FAR_{FM+R} &gt; FAR_{FM perc R}$</td>
<td>$FAR_{original}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$p=0.49$</td>
<td>$p=2.2^{85}$</td>
<td>$p=0.52$</td>
<td>$p=0.49$</td>
</tr>
<tr>
<td>FM*R</td>
<td>$FAR_{FM+R} &gt; FAR_{FM*R}$</td>
<td>-</td>
<td>$FAR_{FM+R} &gt; FAR_{FM only,R}$</td>
<td>$FAR_{FM+R} &gt; FAR_{FM perc R}$</td>
<td>$FAR_{original}$</td>
</tr>
<tr>
<td></td>
<td>$p=0.5$</td>
<td></td>
<td>$p=8.2^{85}$</td>
<td>$p=0.51$</td>
<td>$p=0.5$</td>
</tr>
<tr>
<td>FM only</td>
<td>$FAR_{FM only} &gt; FAR_{FM*R}$</td>
<td>$FAR_{FM only} &gt; FAR_{FM*R}$</td>
<td>-</td>
<td>$FAR_{FM only} &gt; FAR_{FM perc R}$</td>
<td>$FAR_{original}$</td>
</tr>
<tr>
<td></td>
<td>$p=1$</td>
<td>$p=1$</td>
<td></td>
<td>$p=1$</td>
<td>$p=1$</td>
</tr>
<tr>
<td>FM perc R</td>
<td>$FAR_{FM perc R} &gt; FAR_{FM*R}$</td>
<td>$FAR_{FM perc R} &gt; FAR_{FM*R}$</td>
<td>$FAR_{FM perc R} &gt; FAR_{FM only,R}$</td>
<td>-</td>
<td>$FAR_{FM perc R}$</td>
</tr>
<tr>
<td></td>
<td>$0.47$</td>
<td>$0.46$</td>
<td>$p=3.2^{85}$</td>
<td></td>
<td>$p=0.46$</td>
</tr>
<tr>
<td>original</td>
<td>$FAR_{original} &gt; FAR_{FM*R}$</td>
<td>$FAR_{original} &gt; FAR_{FM*R}$</td>
<td>$FAR_{original} &gt; FAR_{FM perc R}$</td>
<td>-</td>
<td>$FAR_{original}$</td>
</tr>
<tr>
<td></td>
<td>$p=0.5$</td>
<td>$p=0.5$</td>
<td>$p=8.2^{85}$</td>
<td></td>
<td>$p=0.53$</td>
</tr>
</tbody>
</table>

The partial ordering of the ranking coefficients with RFA on false alarm rate for the ISCX 50 data set shows exactly the same pattern on the ISCX 25 data set as shown in Figure 5.35.

Table 5.22 shows the results of Mann-Whitney test on all the ranking coefficients with RFA for ISCX 100 data set on false alarm rate.
Figure 5.35: Partial ordering of the ranking coefficients with RFA on false alarm rate for the ISCX 50 data set

Table 5.22: Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 100 data set on false alarm rate

<table>
<thead>
<tr>
<th></th>
<th>FM*+R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM+R</td>
<td>-</td>
<td>(FAR_{FM+R} &gt; FAR_{FM \times R})</td>
<td>(FAR_{FM+R} &gt; FAR_{FM \times R})</td>
<td>(FAR_{FM+R} &gt; FAR_{FM \times R})</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(p=0.63)</td>
<td>(p=1)</td>
<td>(p=1)</td>
</tr>
<tr>
<td>FM*+R</td>
<td>(FAR_{FM+R} &gt; FAR_{FM \times R})</td>
<td>-</td>
<td>(FAR_{FM+R} &gt; FAR_{FM \times R})</td>
<td>(FAR_{FM+R} &gt; FAR_{FM \times R})</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(p=0.36)</td>
<td>(p=2.02^{-116})</td>
<td>(p=3.7^{-12})</td>
</tr>
<tr>
<td>FM only</td>
<td>(FAR_{FM \times R} &gt; FAR_{FM \times R})</td>
<td>(FAR_{FM \times R} &gt; FAR_{FM \times R})</td>
<td>-</td>
<td>(FAR_{FM \times R} &gt; FAR_{FM \times R})</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(p=1)</td>
<td>(p=1)</td>
<td>(p=1)</td>
</tr>
<tr>
<td>FM perc R</td>
<td>(FAR_{FM perc R} &gt; FAR_{FM perc R})</td>
<td>(FAR_{FM perc R} &gt; FAR_{FM perc R})</td>
<td>-</td>
<td>(FAR_{FM perc R} &gt; FAR_{FM perc R})</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.3)</td>
<td>(1.2^{-116})</td>
<td>(4.3^{-14})</td>
</tr>
<tr>
<td>original</td>
<td>(FAR_{original} &gt; FAR_{original})</td>
<td>(FAR_{original} &gt; FAR_{original})</td>
<td>(FAR_{original} &gt; FAR_{original})</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(p=1)</td>
<td>(p=1)</td>
<td>(p=1)</td>
</tr>
</tbody>
</table>

The partial ordering of the ranking coefficients with RFA on false alarm rate for the ISCX 100 data set shows that FM*+R, FM perc R and FM+R outperform the other ranking coefficients, while the original ranking coefficient came in the second place. FM only ranking
coefficient came in last place as it did not show any significance over any of the other ranking coefficients as shown in Figure 5.36.

Figure 5.36: Partial ordering of the ranking coefficients with RFA on false alarm rate for the ISCX 100 data set

Table 5.23 shows the results of Mann-Whitney test on all the ranking coefficients with RFA for ISCX 500 data set on false alarm rate.
Table 5.23: Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 500 data set on false alarm rate

<table>
<thead>
<tr>
<th></th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM+R</td>
<td>-</td>
<td>(\text{FAR}<em>{\text{FM+R}} &gt; \text{FAR}</em>{\text{FM+R+:T}}) (p=0.21)</td>
<td>(\text{FAR}<em>{\text{FM+R}} &gt; \text{FAR}</em>{\text{FM only:T}}) (p=3.4^{-114})</td>
<td>(\text{FAR}<em>{\text{FM+R}} &gt; \text{FAR}</em>{\text{FM perc R:T}}) (p=2.4^{-9})</td>
<td>(\text{FAR}<em>{\text{FM+R}} &gt; \text{FAR}</em>{\text{original:T}}) (p=0.41)</td>
</tr>
<tr>
<td>FM*R</td>
<td>(\text{FAR}<em>{\text{FM+R}} &gt; \text{FAR}</em>{\text{FM+R+:T}}) (p=0.78)</td>
<td>-</td>
<td>(\text{FAR}<em>{\text{FM+R}} &gt; \text{FAR}</em>{\text{FM only:T}}) (p=1.5^{-110})</td>
<td>(\text{FAR}<em>{\text{FM+R}} &gt; \text{FAR}</em>{\text{FM perc R:T}}) (p=5.4^{-3})</td>
<td>(\text{FAR}<em>{\text{FM+R}} &gt; \text{FAR}</em>{\text{original:T}}) (p=0.65)</td>
</tr>
<tr>
<td>FM only</td>
<td>(\text{FAR}<em>{\text{FM only}} &gt; \text{FAR}</em>{\text{FM+R:F}}) (p=1)</td>
<td>(\text{FAR}<em>{\text{FM only}} &gt; \text{FAR}</em>{\text{FM+R:T}}) (p=1)</td>
<td>-</td>
<td>(\text{FAR}<em>{\text{FM only}} &gt; \text{FAR}</em>{\text{FM perc R:F}}) (p=1)</td>
<td>(\text{FAR}<em>{\text{FM only}} &gt; \text{FAR}</em>{\text{original:F}}) (p=1)</td>
</tr>
<tr>
<td>FM perc R</td>
<td>(\text{FAR}<em>{\text{FM perc R}} &gt; \text{FAR}</em>{\text{FM+R:F}}) (p=1)</td>
<td>(\text{FAR}<em>{\text{FM perc R}} &gt; \text{FAR}</em>{\text{FM+R:T}}) (p=0.99)</td>
<td>(\text{FAR}<em>{\text{FM perc R}} &gt; \text{FAR}</em>{\text{FM only:T}}) (p=1.12^{-112})</td>
<td>-</td>
<td>(\text{FAR}<em>{\text{FM perc R}} &gt; \text{FAR}</em>{\text{original:F}}) (p=0.99)</td>
</tr>
<tr>
<td>original</td>
<td>(\text{FAR}<em>{\text{original}} &gt; \text{FAR}</em>{\text{FM+R+:T}}) (p=0.58)</td>
<td>(\text{FAR}<em>{\text{original}} &gt; \text{FAR}</em>{\text{FM+R+:T}}) (p=0.34)</td>
<td>(\text{FAR}<em>{\text{original}} &gt; \text{FAR}</em>{\text{FM+R:T}}) (p=9.04^{-111})</td>
<td>(\text{FAR}<em>{\text{original}} &gt; \text{FAR}</em>{\text{FM perc R:T}}) (p=4.1^{-4})</td>
<td>-</td>
</tr>
</tbody>
</table>

The partial ordering of the ranking coefficients with RFA on false alarm rate for the ISCX 500 data set shows that FM*R, FM+R and original ranking coefficients share the first place as they all outperform both FM perc R and FM only ranking coefficients. However, since FM perc R outperformed the FM only ranking coefficient, then FM perc R came in the second place and FM only was the last one as shown in Figure 5.37.

The last metric is the overall metric which we also conducted the same statistical test on. Table 5.24 shows the results of Mann-Whitney test on all the ranking coefficients with RFA for ISCX 25 data set on overall metric.
Figure 5.37: Partial ordering of the ranking coefficients with RFA on false alarm rate for the ISCX 500 data set

Table 5.24: Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 25 data set on overall metric

<table>
<thead>
<tr>
<th></th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM+R</td>
<td>-</td>
<td>overall_{FM+R} &gt; overall_{FM only}; T p=1.4^{-83}</td>
<td>overall_{FM+R} &gt; overall_{FM perc R}; T p=9.4^{-80}</td>
<td>overall_{FM+R} &gt; overall_{original}; p=0.28</td>
<td></td>
</tr>
<tr>
<td>FM*R</td>
<td>overall_{FM+R} &gt; overall_{FM only}; T p=8.6^{-86}</td>
<td>T p=9.4^{-80}</td>
<td>overall_{FM+R} &gt; overall_{FM perc R}; T p=4.7^{-90}</td>
<td>overall_{FM+R} &gt; overall_{original}; p=0.5</td>
<td></td>
</tr>
<tr>
<td>FM only</td>
<td>overall_{FM only} &gt; overall_{FM+R}; F p=1</td>
<td>overall_{FM only} &gt; overall_{FM perc R}; F p=1</td>
<td>overall_{FM only} &gt; overall_{FM perc R}; F p=1</td>
<td>overall_{FM only} &gt; overall_{original}; F p=1</td>
<td></td>
</tr>
<tr>
<td>FM perc R</td>
<td>overall_{FM perc R} &gt; overall_{FM+R}; F p=1</td>
<td>overall_{FM perc R} &gt; overall_{FM+R}; F p=1</td>
<td>overall_{FM perc R} &gt; overall_{FM+R}; F p=1</td>
<td>overall_{FM perc R} &gt; overall_{original}; F p=1</td>
<td></td>
</tr>
<tr>
<td>original</td>
<td>overall_{original} &gt; overall_{FM+R};? p=0.7</td>
<td>overall_{original} &gt; overall_{FM+R};? p=0.5</td>
<td>overall_{original} &gt; overall_{FM+R};? p=0.5</td>
<td>overall_{original} &gt; overall_{original};? p=0.5</td>
<td></td>
</tr>
</tbody>
</table>

165
The partial ordering of the ranking coefficients with RFA on the overall metric for the ISCX 25 data set shows that FM+R, FM*R and original ranking coefficients outperformed the other ones and they all share the first place. FM perc R came in the second place as it outperformed only the FM only ranking coefficient which came in the last place as shown in Figure 5.38.

![Diagram of partial ordering of ranking coefficients](image)

Figure 5.38: Partial ordering of the ranking coefficients with RFA on the overall metric for the ISCX 25 data set

Table 5.25 shows the results of Mann-Whitney test on all the ranking coefficients with RFA for ISCX 50 data set on overall metric.
Table 5.25: Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 50 data set on overall metric

<table>
<thead>
<tr>
<th></th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM+R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>overallFM+R &gt; overallFM*R: F</td>
<td>overallFM+R &gt; overallFM only: T; p=2.7^{-106}</td>
<td>overallFM+R &gt; overallFM perc R: p=0.82</td>
<td>overallFM+R &gt; overalloriginal:F p=1</td>
</tr>
<tr>
<td>FM*R</td>
<td>overallFM+R &gt; overallFM*R: T; p=3.1^{-9}</td>
<td>-</td>
<td>overallFM+R &gt; overallFM only: T; p=2.19^{-105}</td>
<td>overallFM+R &gt; overallFM perc R: p=1.19^{-10}</td>
<td>overallFM+R &gt; overalloriginal: p=0.5</td>
</tr>
<tr>
<td>FM only</td>
<td>overallFM only &gt; overallFM+R: F; p=1</td>
<td>overallFM only &gt; overallFM+R: F; p=1</td>
<td>-</td>
<td>overallFM perc R &gt; overallFM only: T; p=1.8^{-106}</td>
<td>overallFM perc R &gt; overalloriginal: p=1</td>
</tr>
<tr>
<td>FM perc R</td>
<td>overallFM perc R &gt; overallFM+R: ?; p=0.17</td>
<td>overallFM perc R &gt; overallFM+R: F; p=1</td>
<td>overallFM perc R &gt; overallFM only: T; p=1.8^{-106}</td>
<td>overallFM perc R &gt; overalloriginal: T</td>
<td>overallFM perc R &gt; overalloriginal: p=1</td>
</tr>
<tr>
<td>original</td>
<td>overalloriginal &gt; overallFM+R: T; p=3.1^{-9}</td>
<td>overalloriginal &gt; overallFM+R: ?; p=0.5</td>
<td>overalloriginal &gt; overallFM only: T; p=2.19^{-105}</td>
<td>overalloriginal &gt; overallFM perc R: p=1.19^{-10}</td>
<td>overalloriginal &gt; overallFM perc R: p=1</td>
</tr>
</tbody>
</table>

The partial ordering of the ranking coefficients with RFA on the overall metric for the ISCX 50 data set shows that both FM*R and the original ranking coefficient share the first place as they both outperform the other ranking coefficients. FM+R and FM perc R share the second place as they both outperform only the FM only ranking coefficient as shown in Figure 5.39.

Table 5.26 shows the results of Mann-Whitney test on all the ranking coefficients with RFA for ISCX 100 data set on overall metric.
Figure 5.39: Partial ordering of the ranking coefficients with RFA on the overall metric for the ISCX 50 data set

Table 5.26: Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 100 data set on overall metric

<table>
<thead>
<tr>
<th></th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM+R</td>
<td>-</td>
<td></td>
<td>overall_{FM+R} &gt; overall_{FM+R}, F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>p=1</td>
<td>overall_{FM+R} &gt; overall_{FM only}, F</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>p=4.08e-109</td>
<td>overall_{FM perc R} &gt; overall_{original}, F</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>p=0.71</td>
<td>overall_{FM perc R} &gt; overall_{original}, F</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>p=1</td>
<td>overall_{FM perc R} &gt; overall_{original}, F</td>
<td></td>
</tr>
<tr>
<td>FM*R</td>
<td>overall_{FM+R}, F &gt; overall_{FM+R}, T</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=1.3e-27</td>
<td></td>
<td></td>
<td>overall_{FM perc R} &gt; overall_{FM perc R}, T</td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=3.8e-110</td>
<td></td>
<td></td>
<td>overall_{FM perc R} &gt; overall_{FM perc R}, T</td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=3.7e-109</td>
<td></td>
<td></td>
<td>overall_{FM perc R} &gt; overall_{original}, F</td>
<td></td>
</tr>
<tr>
<td>FM only</td>
<td>overall_{FM only} &gt; overall_{FM only}, F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=1</td>
<td></td>
<td>overall_{FM only} &gt; overall_{FM only}, F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=1</td>
<td></td>
<td>overall_{FM perc R} &gt; overall_{FM perc R}, T</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>p=7.4e-26</td>
<td>overall_{FM perc R} &gt; overall_{FM perc R}, T</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>p=3.7e-109</td>
<td>overall_{FM perc R} &gt; overall_{FM perc R}, T</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>p=1</td>
<td>overall_{FM perc R} &gt; overall_{original}, F</td>
<td></td>
</tr>
<tr>
<td>FM perc R</td>
<td>overall_{FM perc R}, F &gt; overall_{FM perc R}, T</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=0.28</td>
<td></td>
<td>overall_{FM perc R} &gt; overall_{FM perc R}, T</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=3.3e-109</td>
<td></td>
<td></td>
<td>overall_{FM perc R} &gt; overall_{FM perc R}, T</td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=1</td>
<td></td>
<td>overall_{FM perc R} &gt; overall_{original}, F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>original</td>
<td>overall_{original} &gt; overall_{original}, F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=7.5e-9</td>
<td></td>
<td>overall_{original} &gt; overall_{original}, F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=4.5e-110</td>
<td></td>
<td>overall_{original} &gt; overall_{original}, F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=7.1e-8</td>
<td></td>
<td>overall_{original} &gt; overall_{original}, F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=0.99</td>
<td></td>
<td>overall_{original} &gt; overall_{original}, F</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The partial ordering of the ranking coefficients with RFA on the overall metric for the ISCX 100 data set shows exactly the same pattern on the ISCX 50 data set as shown in Figure 5.40.

Figure 5.40: Partial ordering of the ranking coefficients with RFA on the overall metric for the ISCX 100 data set

Table 5.27 shows the results of Mann-Whitney test on all the ranking coefficients with RFA for ISCX 500 data set on overall metric.
Table 5.27: Mann-Whitney U test for all the ranking coefficients with RFA for ISCX 500 data set on overall metric

<table>
<thead>
<tr>
<th></th>
<th>FM+R</th>
<th>FM*R</th>
<th>FM only</th>
<th>FM perc R</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM+R</td>
<td>-</td>
<td></td>
<td>overall_FM+R (&gt;) overall_FM+R;T</td>
<td>overall_FM+R (&gt;) overall_FM only;T</td>
<td>overall_FM+R (&gt;) overall_FM perc R;T</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>p=8.9(^{-14})</td>
<td>p=8.3(^{-107})</td>
<td>p=2.7(^{-6})</td>
</tr>
<tr>
<td>FM*R</td>
<td>overall_FM only (&gt;) overall_FM+R;F</td>
<td>-</td>
<td>overall_FM only (&gt;) overall_FM only;T</td>
<td>overall_FM only (&gt;) overall_FM perc R;F</td>
<td>overall_FM only (&gt;) overall_FM perc R;F</td>
</tr>
<tr>
<td></td>
<td>p=1</td>
<td></td>
<td>p=3.8(^{-105})</td>
<td>p=0.99</td>
<td>p=1</td>
</tr>
<tr>
<td>FM only</td>
<td>overall_FM perc R (&gt;) overall_FM+R;F</td>
<td>overall_FM perc R (&gt;) overall_FM perc R;F</td>
<td>-</td>
<td>overall_FM perc R (&gt;) overall_FM perc R;F</td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=0.99</td>
<td>p=1</td>
<td>p=1</td>
<td>p=1</td>
<td></td>
</tr>
<tr>
<td>FM perc R</td>
<td>overall_original (&gt;) overall_FM+R;T</td>
<td>overall_original (&gt;) overall_FM perc R;T</td>
<td>overall_original (&gt;) overall_original;F</td>
<td>overall_original (&gt;) overall_FM perc R;F</td>
<td></td>
</tr>
<tr>
<td></td>
<td>p=0.01</td>
<td>p=4.1(^{-17})</td>
<td>p=2.4(^{-107})</td>
<td>p=5.1(^{-8})</td>
<td>p=0.99</td>
</tr>
</tbody>
</table>

The partial ordering of the ranking coefficients with RFA on the overall metric for the 500 data set shows that both FM+R and original ranking coefficients perform the same in their superior performance over the other ranking coefficients. FM perc R came in second place at it outperformed both FM+R and FM only. As with the previous data sets and metrics, the FM only came in the last place as shown in Figure 5.41.
Multi-class intrusion detection problem

In intrusion detection, there are two common approaches: binary classification and multi-class classification. In the previous sections, we discussed the binary classification approach where the system tells if the incoming traffic there is an intrusion or not (two possible cases only: normal and intrusion). However, there is another approach where the intrusion detection system identifies the specific kind of intrusion instead of just raising an alarm of intrusion only without revealing its type. The advantage of this approach is that when it identifies the type of intrusion, it makes it easier for administrators to use the proper treatment for that intrusion. Therefore, for the sake of completeness, we opted here to add the multi-class intrusion detection to the thesis.

Since we are using the ISCX 2012 data set which already has five classes (four attack types and normal connections), we decided to use the same data set for this part. The section is organized as follows: In Section 5.10.1 we explain the binary tree structure for multi-class classification. Preparing the data set to this approach is explained in Section 5.10.2. Next,
we present the results of applying both RFA and RFE on the new data sets in Sections 5.10.3 and 5.10.4 respectively. Lastly, we discuss the obtained results in Section 5.10.6.

5.10.1 Binary tree structure for multi-class classification

Since our feature selection method RFA is based on binary SVMs classifier, we designed a binary tree structure in order to perform multi-class classification using binary classifier. The proposed binary tree structure is shown in Figure 5.42. The first sub-tree is to distinguish between normal and attack (in general) categories only. The attack category includes two sub-categories: DoS attacks and non-DoS attacks. For the DoS attacks, there are two sub-categories HTTP DoS and DDoS attacks, while for the non-DoS attacks there are internal attacks and brute-force attacks. Therefore, there are four total sub-trees that we used in constructing the binary tree structure: normal and attack, DoS and non-DoS, HTTP DoS and DDoS, and internal and brute-force attacks.

![Binary tree structure for multi-class classification](image)

Figure 5.42: Binary tree structure for multi-class classification
5.10.2 Data set preparation

To prepare the data set for our experiments, we used two approaches in splitting the data set for training and testing purposes. The first approach is to use 10% size for the training and 90% for the testing. The second approach is to use balanced data sets for training and testing. Table 5.28 shows the data set’s distribution for the balanced approach. In terms of features, we used the same features that were used in Nyakundi (2015). We used Application Name, Total Source Bytes, Total Destination Bytes, Total Destination Packets, Total Source Packets, Direction, Source TCP Flags Description, Destination TCP Flags Description, Protocol Name, Source Port, Destination Port, and Class. The other features such as TimeStart, TimeEnd and Base64 have been excluded from this experiment.

Table 5.28: Data sets’ distribution for the balanced approach

<table>
<thead>
<tr>
<th>Data set</th>
<th>Training</th>
<th>Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal vs. attacks</td>
<td>19107</td>
<td>19111</td>
</tr>
<tr>
<td>DoS vs non-DoS attacks</td>
<td>10154</td>
<td>9501</td>
</tr>
<tr>
<td>HTTP DoS vs DDoS attacks</td>
<td>5066</td>
<td>4409</td>
</tr>
<tr>
<td>internal vs brute-force attacks</td>
<td>5088</td>
<td>5092</td>
</tr>
</tbody>
</table>

The second approach for data set’s distribution is the imbalanced distribution (10% for training and 90% for testing) as shown in Table 5.29.

Table 5.29: Data sets’ distribution for the imbalanced approach

<table>
<thead>
<tr>
<th>Data set</th>
<th>Training</th>
<th>Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal vs. attacks</td>
<td>1910</td>
<td>17199</td>
</tr>
<tr>
<td>DoS vs non-DoS attacks</td>
<td>1015</td>
<td>9501</td>
</tr>
<tr>
<td>HTTP DoS vs DDoS attacks</td>
<td>4114</td>
<td>37035</td>
</tr>
<tr>
<td>internal vs brute-force attacks</td>
<td>2555</td>
<td>23006</td>
</tr>
</tbody>
</table>
We used two different approaches in data set’s distribution in order to study the behaviour of feature selection with each distribution to observe with which one feature selection performs the best.

5.10.3 Results of applying feature selection on the balanced data sets

In this section, we are going to present the obtained results of applying both feature selection methods RFA and RFE on the ISCX data sets. We measured seven performance metrics after applying feature selection. The performance metrics are: accuracy, $F$ – measure, detection rate (recall), False Alarm Rate (FAR), the overall metric, the precision and the time to reach the optimum. Since we have four sub-trees in the overall binary structure, we report all the performance metrics for each sub-tree.

(a) Results of applying RFA on the balanced data sets

In Table 5.30, the performance metrics after applying RFA on the balanced ISCX data sets according to the sub-tree structure are shown.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Normal vs attack</th>
<th>HTTP DoS vs DDoS</th>
<th>DoS vs non-DoS</th>
<th>internal vs brute-force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>97.7%</td>
<td>99.9%</td>
<td>80.6%</td>
<td>100%</td>
</tr>
<tr>
<td>$F$ – measure</td>
<td>97.6%</td>
<td>99.9%</td>
<td>77.6%</td>
<td>100%</td>
</tr>
<tr>
<td>Detection rate</td>
<td>98.8%</td>
<td>100%</td>
<td>99.6%</td>
<td>100%</td>
</tr>
<tr>
<td>FAR</td>
<td>1%</td>
<td>0%</td>
<td>1%</td>
<td>0%</td>
</tr>
<tr>
<td>Overall</td>
<td>95.8%</td>
<td>99.8%</td>
<td>68.4%</td>
<td>100%</td>
</tr>
<tr>
<td>Precision</td>
<td>98.5%</td>
<td>100%</td>
<td>96.5%</td>
<td>100%</td>
</tr>
<tr>
<td>Time (sec)</td>
<td>733.5</td>
<td>23</td>
<td>269.7</td>
<td>30.9</td>
</tr>
</tbody>
</table>

(b) Results of applying RFE on the balanced data sets

In Table 5.31, the performance metrics after applying RFE on the balanced ISCX data sets according to the sub-tree structure are shown.
Table 5.31: Results of applying RFE on the balanced data sets

<table>
<thead>
<tr>
<th>Metric</th>
<th>Normal vs attack</th>
<th>HTTP DoS vs DDoS</th>
<th>DoS vs non-DoS</th>
<th>internal vs brute-force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>97.7%</td>
<td>99.9%</td>
<td>80.6%</td>
<td>100%</td>
</tr>
<tr>
<td>F-measure</td>
<td>97.6%</td>
<td>99.9%</td>
<td>77.6%</td>
<td>100%</td>
</tr>
<tr>
<td>Detection rate</td>
<td>98.8%</td>
<td>100%</td>
<td>99.6%</td>
<td>100%</td>
</tr>
<tr>
<td>FAR</td>
<td>1%</td>
<td>$7.9 \times 10^{-4}$%</td>
<td>1%</td>
<td>0%</td>
</tr>
<tr>
<td>Overall</td>
<td>95.8%</td>
<td>99.8%</td>
<td>68.4%</td>
<td>100%</td>
</tr>
<tr>
<td>Precision</td>
<td>98.5%</td>
<td>99.8%</td>
<td>96.5%</td>
<td>100%</td>
</tr>
<tr>
<td>Time (sec)</td>
<td>770</td>
<td>2.7</td>
<td>348</td>
<td>4.1</td>
</tr>
</tbody>
</table>

5.10.4 Results of applying feature selection on the imbalanced data sets

Similar to Section 5.10.3, we present in this section the results of applying feature selection but on the imbalanced ISCX data sets. We also measure the same performance metrics that we measured with the previous data sets after applying both RFA and RFE. All of the four sub-trees in the overall binary structure have been evaluated and all the performance metrics for each sub-tree have been reported.

(a) Results of applying RFA on the imbalanced data sets

In Table 5.32, the performance metrics after applying RFA on the imbalanced ISCX data sets according to the sub-tree structure are shown.
Table 5.32: Results of applying RFA on the imbalanced data sets

<table>
<thead>
<tr>
<th>Metric</th>
<th>Normal vs attack</th>
<th>HTTP DoS vs DDoS</th>
<th>DoS vs non-DoS</th>
<th>internal vs brute-force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>97.2%</td>
<td>99.5%</td>
<td>76.1%</td>
<td>99.9%</td>
</tr>
<tr>
<td>$F - measure$</td>
<td>97.2%</td>
<td>97.3%</td>
<td>78.9%</td>
<td>99.9%</td>
</tr>
<tr>
<td>Detection rate</td>
<td>100%</td>
<td>95.4%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>FAR</td>
<td>1%</td>
<td>5.9 $\cdot e^{-4}$%</td>
<td>39%</td>
<td>0%</td>
</tr>
<tr>
<td>Overall</td>
<td>95.2%</td>
<td>97.4%</td>
<td>45.2%</td>
<td>99.9%</td>
</tr>
<tr>
<td>Precision</td>
<td>95.2%</td>
<td>99.5%</td>
<td>67.1%</td>
<td>100%</td>
</tr>
<tr>
<td>Time (sec)</td>
<td>6.7</td>
<td>8</td>
<td>5.5</td>
<td>7.4</td>
</tr>
</tbody>
</table>

(b) Results of applying RFE on the imbalanced data sets

In Table 5.33, the performance metrics after applying RFE on the imbalanced ISCX data sets according to the sub-tree structure are shown.

Table 5.33: Results of applying RFE on the imbalanced data sets

<table>
<thead>
<tr>
<th>Metric</th>
<th>Normal vs attack</th>
<th>HTTP DoS vs DDoS</th>
<th>DoS vs non-DoS</th>
<th>internal vs brute-force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>92.2%</td>
<td>99.5%</td>
<td>76%</td>
<td>99.9%</td>
</tr>
<tr>
<td>$F - measure$</td>
<td>91.8%</td>
<td>97.3%</td>
<td>78.9%</td>
<td>99.9%</td>
</tr>
<tr>
<td>Detection rate</td>
<td>98.8%</td>
<td>95.4%</td>
<td>96.5%</td>
<td>100%</td>
</tr>
<tr>
<td>FAR</td>
<td>14%</td>
<td>5.9 $\cdot e^{-4}$%</td>
<td>39%</td>
<td>0%</td>
</tr>
<tr>
<td>Overall</td>
<td>86.7%</td>
<td>97.4%</td>
<td>45.2%</td>
<td>99.9%</td>
</tr>
<tr>
<td>Precision</td>
<td>96.4%</td>
<td>99.5%</td>
<td>67%</td>
<td>100%</td>
</tr>
<tr>
<td>Time (sec)</td>
<td>2.1</td>
<td>2.1</td>
<td>7</td>
<td>4.8</td>
</tr>
</tbody>
</table>
5.10.5 Calculating the final confusion matrix for multi-class intrusion detection

Since we used the binary tree approach in evaluating the performance of feature selection, we obtained multiple confusion matrices, each of which corresponds one sub-tree and it is 2x2 cells. Therefore, we conducted some calculations in order to construct the final confusion matrix that consists of (5x5) cells. We constructed two final confusion matrices, one for each data set distribution as shown below:

(a) Constructing the final confusion matrix for the balanced data sets

As stated before, the binary tree structure that we used consists of four binary sub-trees. Therefore, four binary confusion matrices are produced from applying feature selection on the four binary sub-trees. The final confusion matrix is constructed form the four confusion matrices as we will explain. The same approach is followed in constructing the final confusion matrix for RFE. A step-by-step in calculating the final confusion matrix for the multi-class intrusion detection is shown below.

In Table 5.34, the confusion matrix of the normal and attack sub-tree using RFA on the balanced ISCX data sets is shown.

Table 5.34: The confusion matrix for normal vs. attack sub-tree using RFA on the balanced ISCX data sets

<table>
<thead>
<tr>
<th></th>
<th>Predicted Normal</th>
<th>Predicted Attack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Normal</td>
<td>9417</td>
<td>138</td>
</tr>
<tr>
<td>Actual Attack</td>
<td>299</td>
<td>9257</td>
</tr>
</tbody>
</table>

First, we need to calculate the percentage of each cell to the corresponding class. For example, by dividing 9417 by the total of normal class (which is 9555), we obtain 98.56% and by dividing 138 by 9555, we obtain 1.44%. Similarly, for the attack class,
by dividing 299 by the total of attack class (which is 9556), we obtain 3.13% and by dividing 9257 by 9556, we obtain 96.87%. Table 5.35 shows the calculated percentages of normal and attacks classes.

Table 5.35: Percentages of normal attack calculated from the corresponding confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Percentage of predicted Normal</th>
<th>Percentage of predicted Attack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Normal</td>
<td>98.56%</td>
<td>1.44%</td>
</tr>
<tr>
<td>Actual Attack</td>
<td>3.13%</td>
<td>96.87%</td>
</tr>
</tbody>
</table>

Similarly, for the non-DoS vs. DoS attacks sub-tree, the resulting confusion matrix is shown in Table 5.36.

Table 5.36: The confusion matrix for Non-DoS vs. DoS sub-tree using RFA on the balanced ISCX data sets

<table>
<thead>
<tr>
<th></th>
<th>Predicted Non-DoS</th>
<th>Predicted DoS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Non-DoS</td>
<td>4964</td>
<td>128</td>
</tr>
<tr>
<td>Actual DoS</td>
<td>1706</td>
<td>2703</td>
</tr>
</tbody>
</table>

We also calculated the percentages of each class as shown in Table 5.37. In this table we also calculated the average of each column of the since we will need it in estimating the final confusion matrix of the multi-class intrusion detection.
Table 5.37: Percentages of Non DoS vs DoS attacks calculated from the corresponding confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Percentage of predicted Non-DoS</th>
<th>Percentage of predicted DoS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Non-DoS</td>
<td>97.49%</td>
<td>2.51%</td>
</tr>
<tr>
<td>Actual DoS</td>
<td>38.69%</td>
<td>61.31%</td>
</tr>
<tr>
<td>Average:</td>
<td>68.09%</td>
<td>31.91%</td>
</tr>
</tbody>
</table>

The third sub-tree is the internal vs. brute-force attacks (both are non-DoS attacks). The resulting confusion matrix for that sub-tree is shown in Table 5.38.

Table 5.38: The confusion matrix for internal vs. brute-force attacks sub-tree using RFA on the balanced ISCX data sets

<table>
<thead>
<tr>
<th></th>
<th>Predicted Internal</th>
<th>Predicted Brute-force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Internal</td>
<td>2547</td>
<td>0</td>
</tr>
<tr>
<td>Actual Brute-force</td>
<td>0</td>
<td>2545</td>
</tr>
</tbody>
</table>

By calculating the obtained percentages from the above confusion matrix, we can notice the perfect results as shown in Table 5.39.
Table 5.39: Percentages of Internal vs Brute-force attacks calculated from the corresponding confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Percentage of predicted Internal</th>
<th>Percentage of predicted Brute-force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Internal</td>
<td>100%</td>
<td>0%</td>
</tr>
<tr>
<td>Actual Brute-force</td>
<td>0%</td>
<td>100%</td>
</tr>
<tr>
<td>Average:</td>
<td>50%</td>
<td>50%</td>
</tr>
</tbody>
</table>

Lastly, the fourth sub-tree is for DoS attacks (HTTP and DDoS) and the obtained confusion matrix is shown in Table 5.40.

Table 5.40: The confusion matrix for HTTP DoS vs. DDoS attacks sub-tree using RFA on the balanced ISCX data sets

<table>
<thead>
<tr>
<th></th>
<th>Predicted HTTP DoS</th>
<th>Predicted DDoS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual HTTP DoS</td>
<td>2521</td>
<td>2</td>
</tr>
<tr>
<td>Actual DDoS</td>
<td>0</td>
<td>1886</td>
</tr>
</tbody>
</table>

Similar to the previous sub-tree, we calculated the percentages from the DoS attacks confusion matrix as shown in Table 5.41.
Table 5.41: Percentages of HTTP DoS vs DDoS attacks calculated from the corresponding confusion matrix

<table>
<thead>
<tr>
<th>Actual HTTP DoS</th>
<th>Percentage of predicted HTTP DoS</th>
<th>Percentage of predicted DDoS</th>
</tr>
</thead>
<tbody>
<tr>
<td>99.92%</td>
<td>0.08%</td>
<td></td>
</tr>
<tr>
<td>Actual DDoS</td>
<td>0%</td>
<td>100%</td>
</tr>
<tr>
<td>Average:</td>
<td>49.96%</td>
<td>50.04%</td>
</tr>
</tbody>
</table>

Now, to construct the final confusion matrix, we start from the top of the tree structure shown in Figure 5.42. For the first row, the first cell corresponds to the normal connections that are correctly classified as normal. This value is taken directly from the top left cell of Table 5.35. Now to calculate the other cell in the first row, we will use a way to estimate that since we do not have the exact numbers for them. In general, those cells correspond to the normal connections that are misclassified as attacks. The next cell after the top left corresponds to the normal connections that are misclassified as internal attacks. Recalling the binary tree structure, we need to use the percentage of the normal connections that are misclassified as attacks (in general) first which is the top right cell of Table 5.35 and its value here is 1.44%. Then, we need to use average percentage of classifying any attack to non-DoS attacks which is the bottom left cell of Table 5.37 and its value here is 68.09%. Next, we multiply the previous two values by the percentage of any non-DoS attack as an internal attack, which is the bottom left cell of Table 5.39 and its value here is 50%. By multiplying the those three values, we obtain 0.049% for the normal connections that are misclassified as internal attacks.

For the normal connections that are misclassified as brute-force attacks, we also the follow the binary tree structure but we take the right branch of the non-DoS sub-tree instead of the left one. Therefore, we multiply the previous two values (1.44% and
68.09%) by the average of classifying any non-DoS attack as brute-force attack which is the bottom right cell of Table 5.39 and its value here is 50% to obtain 0.49%.

Similarly, we calculate the third cell of the first row which corresponds to the percentage of misclassifying normal connections as HTTP DoS attacks. It should also start from misclassifying normal connections as attacks from the top of the binary tree structure. Now, the route is to take the DoS sub-tree and then taking the left branch of that tree. To calculate the corresponding cell, we multiply the top right cell of Table 5.35 and its value here is 1.44% by the percentage of classifying any attack as DoS attack which is bottom right cell of Table 5.37 which is 31.91%. Next, the previous product is multiplied by the percentage of classifying any DoS attack as HTTP DoS attack which is the bottom left cell of Table 5.41 which its value is 49.96%. The product of the above multiplication is 0.23%.

The other rows are calculated the same way as the Normal row. The final confusion matrix is shown in Table 5.42. We can notice that each row adds up to 100% since the percentage of that particular row may be distributed over all the columns (in case of misclassification). For all the upcoming tables, the grey cells represent the estimated values for the corresponding class.

Table 5.42: The final confusion matrix after applying RFA on the balanced ISCX data sets

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
<th>Normal</th>
<th>Internal</th>
<th>Brute-force</th>
<th>HTTP</th>
<th>DDoS</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>Normal</td>
<td>98.56%</td>
<td>0.49%</td>
<td>0.49%</td>
<td>0.23%</td>
<td>0.23%</td>
<td>100%</td>
</tr>
<tr>
<td>Internal</td>
<td>Internal</td>
<td>3.13%</td>
<td>94.44%</td>
<td>0.00%</td>
<td>1.22%</td>
<td>1.22%</td>
<td>100%</td>
</tr>
<tr>
<td>Brute-force</td>
<td>Brute-force</td>
<td>3.13%</td>
<td>0.00%</td>
<td>94.44%</td>
<td>1.22%</td>
<td>1.22%</td>
<td>100%</td>
</tr>
<tr>
<td>HTTP</td>
<td>HTTP</td>
<td>3.13%</td>
<td>18.74%</td>
<td>18.74%</td>
<td>59.34%</td>
<td>0.05%</td>
<td>100%</td>
</tr>
<tr>
<td>DoS</td>
<td>DoS</td>
<td>3.13%</td>
<td>18.74%</td>
<td>18.74%</td>
<td>0.00%</td>
<td>59.39%</td>
<td>100%</td>
</tr>
</tbody>
</table>

We followed the same steps in constructing the final confusion matrix after applying RFE on the balanced ISCX data sets. The final confusion matrix for RFE is shown in Table 5.43.
Table 5.43: The final confusion matrix after applying RFE on the balanced ISCX data sets

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Normal</th>
<th>Internal</th>
<th>Brute-force</th>
<th>HTTP</th>
<th>DDoS</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>98.56%</td>
<td>0.49%</td>
<td>0.49%</td>
<td>0.23%</td>
<td>0.23%</td>
<td>100%</td>
</tr>
<tr>
<td>Internal</td>
<td>3.13%</td>
<td>94.44%</td>
<td>0.00%</td>
<td>1.22%</td>
<td>1.22%</td>
<td>100%</td>
</tr>
<tr>
<td>Brute-force</td>
<td>3.13%</td>
<td>0.00%</td>
<td>94.44%</td>
<td>1.22%</td>
<td>1.22%</td>
<td>100%</td>
</tr>
<tr>
<td>HTTP</td>
<td>3.13%</td>
<td>18.74%</td>
<td>18.74%</td>
<td>59.34%</td>
<td>0.05%</td>
<td>100%</td>
</tr>
<tr>
<td>DoS</td>
<td>3.13%</td>
<td>18.74%</td>
<td>18.74%</td>
<td>0.00%</td>
<td>59.39%</td>
<td>100%</td>
</tr>
</tbody>
</table>

(b) Constructing the final confusion matrix for the imbalanced data sets

For the imbalanced ISCX data sets, we constructed the final confusion matrix the same way that we did with the balanced data sets. Since we followed the same steps that we followed with the balanced data sets, we are not showing the intermediate steps, instead we are only showing the final confusion matrix using RFA on the imbalanced data sets in Table 5.44.

Table 5.44: The final confusion matrix after applying RFA on the imbalanced ISCX data sets

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Normal</th>
<th>Internal</th>
<th>Brute-force</th>
<th>HTTP</th>
<th>DDoS</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>98.65%</td>
<td>0.21%</td>
<td>0.21%</td>
<td>0.49%</td>
<td>0.45%</td>
<td>100%</td>
</tr>
<tr>
<td>Internal</td>
<td>4.09%</td>
<td>55.98%</td>
<td>0.00%</td>
<td>20.88%</td>
<td>19.05%</td>
<td>100%</td>
</tr>
<tr>
<td>Brute-force</td>
<td>4.09%</td>
<td>0.02%</td>
<td>55.96%</td>
<td>20.88%</td>
<td>19.05%</td>
<td>100%</td>
</tr>
<tr>
<td>HTTP</td>
<td>4.09%</td>
<td>1.65%</td>
<td>1.65%</td>
<td>92.55%</td>
<td>0.06%</td>
<td>100%</td>
</tr>
<tr>
<td>DoS</td>
<td>4.09%</td>
<td>1.65%</td>
<td>1.65%</td>
<td>4.31%</td>
<td>88.30%</td>
<td>100%</td>
</tr>
</tbody>
</table>

For the RFE, we also constructed the final confusion matrix as shown is Table 5.45.
Table 5.45: The final confusion matrix after applying RFE on the imbalanced ISCX data sets

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Normal</th>
<th>Internal</th>
<th>Brute-force</th>
<th>HTTP</th>
<th>DDoS</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>96.81%</td>
<td>0.52%</td>
<td>0.51%</td>
<td>1.13%</td>
<td>1.03%</td>
<td>100%</td>
</tr>
<tr>
<td>Internal</td>
<td>12.37%</td>
<td>52.25%</td>
<td>0.00%</td>
<td>18.48%</td>
<td>16.90%</td>
<td>100%</td>
</tr>
<tr>
<td>Brute-force</td>
<td>12.37%</td>
<td>0.02%</td>
<td>52.23%</td>
<td>18.48%</td>
<td>16.90%</td>
<td>100%</td>
</tr>
<tr>
<td>HTTP</td>
<td>12.37%</td>
<td>2.15%</td>
<td>2.15%</td>
<td>83.29%</td>
<td>0.05%</td>
<td>100%</td>
</tr>
<tr>
<td>DoS</td>
<td>12.37%</td>
<td>2.15%</td>
<td>2.15%</td>
<td>3.78%</td>
<td>79.56%</td>
<td>100%</td>
</tr>
</tbody>
</table>

5.10.6 Discussing the results of the multi-class intrusion detection

As mentioned before, we added the multi-class intrusion detection for the sake of completeness. However, there has been a significant difference in the results when added the multi-class than we used the binary class problem. We studied the behaviour of the intrusion detection system using two approaches for data set splitting. We used a balanced approach which involves using equal sizes of data sets for training and testing. In addition, we used an imbalanced approach which involves using bigger size data set for testing (9 times as the size of training set). For the both approaches, we used the binary tree structure as shown in Figure 5.42 for processing all the classes that we have in this data set since our feature selection method is based on binary SVMs classifier. We followed the two approaches in data sets splitting with RFA and RFE and recorded their performance metrics. From the Table 5.30, we can notice that RFA was able to identify 98.56% of the normal connections with very little percentage of misclassification of normal connections as attacks. The second class is the internal attack which was correctly classified by 94.44%. The same percentage was achieved with brute-force attack. However, with the DoS attacks (HTTP DoS and DDoS), RFA was able to achieve 59.34% and 59.39% respectively. With both of the HTTP DoS and DDoS attacks, RFA misclassified HTTP DoS as internal or brute-force by 18.74%. The same percentage was achieved with DDoS attack which was also misclassified as internal and brute-force.
For the RFE, exactly the same confusion matrix was constructed with all the classes which means there is no difference between RFA and RFE on the balanced ISCX data sets.

However, with imbalanced data sets, the results were different (and sometimes significantly different) from the results of the balanced data sets. Starting with RFA, the method was able to classify normal connections by 98.65% correctly. However, the internal attacks were correctly classified by 55.98% and incorrectly classified as HTTP and DDoS by 20.88% and 19.05% respectively. The same happened with brute-force attacks when they were incorrectly classified as HTTP and DDoS by the same percentages. The significant improvement was achieved on classifying DDoS attacks as they were correctly classified by 88.30%.

For the RFE, the normal class was correctly classified by 96.81%. For the internal and brute force attacks, the RFE was able to achieve only 52.25% and 52.23% respectively correctly classified by SVMs. However, with the HTTP DoS attacks, the percentage was improved to 83.29% compared to the previous attacks. The last attack is the DDoS attack, which 79.56% of the examples were correctly classified by SVMs.

5.11 Summary of the chapter

To sum up, in this chapter we chose one of the most recent data sets for intrusion detection - ISCX 2012 - to apply our proposed feature selection on. In addition, we conducted an important step to perform features extraction by employing a bigram technique prior to applying feature selection. We applied the bigram technique on the payload features of the ISCX 2012 data set. We have noticed in the literature that these features are neglected by researchers as they contain a lot of text and hence they are difficult to deal with. However, these features contain important information that may lead to detecting an incoming intrusion. Encoding these payload features using the bigram technique produced too many bigram features. Given the fact that usually there are only few examples of new attacks, and since the bigram features are too many, that might lead to overfitting in the
learning process. Therefore, employing a feature selection method resilient to overfitting is important to detect these new attacks quickly by identifying only the relevant features. Therefore, we generated four different sized data sets: 25, 50, 100, and 500 examples to test the resilience of feature selection to overfitting. Then, we applied both RFA and RFE on the four generated ISCX data sets and monitored the behaviour of each method on bigram selection/elimination, non-bigram selection/elimination, and non-bigram selection for the first 60 features only and compared them with random selection. We found that the non-bigram features are selected by both methods as they contain important information to intrusion detection. In addition, we found that RFA method outperformed random selection on non-bigram features in all sizes of the ISCX data set. However, that was not the case with RFE which was able to keep the non-bigram features just in the 500 examples data set, while with the other data sets it performed worse than the random selection. We also reported the accuracy and the \( F \)-measure for both RFA and RFE before and after feature selection but the two methods showed close performance.

Furthermore, since it is an intrusion detection application, we also measured the detection rate and false alarm rate. In addition, we proposed an overall metric that combines three metrics: accuracy, detection rate and false alarm rate. To observe the distribution of the performance of both RFA and RFE, we plotted the box-plot for the detection rate, false alarm rate and the overall metric for both methods RFA and RFE and for the four data sets. From the visual comparison, we can notice that RFA has outperformed RFE in the overall metric although RFA did not do better than RFE in some data sets especially with the false alarm rate. To have better insight into the results, we conducted a statistical analysis to compare the results of RFA and RFE. The statistical analysis showed that RFA outperformed RFE in all the data sets using accuracy and \( F \)-measure. We also conducted the same statistical analysis on the detection rate, false alarm rate and the overall metric. RFA showed a superior performance over RFE on those three metrics and on all the data sets except for the 50 and 100 examples where RFE outperformed RFA in the false alarm rate.
The chapter also involved proposing four additional ranking coefficients in addition to the original one and measuring the performance of the accuracy, $F$-measure, detection rate, false alarm rate and the overall metric using those proposed ranking coefficients. The obtained results of the proposed ranking coefficients have been compared also using box-plots. In addition, we conducted a statistical, pair-wise comparison on all the ranking coefficients and on all the data sets using detection rate, false alarm rate and the overall metric. We also provided the partial ordering for each case, since sometimes there are more than one ranking coefficients at the same level of performance.

The last part of the chapter presented the multi-class intrusion detection where the system identifies that particular class of attack in addition to the normal connections. For the sake of completeness, we added the multi-class detection in addition to the binary classification that we originally have. Since our feature selection algorithm is based on binary classifier, we converted the problem into multiple binary classifiers by adopting a binary tree for each two categories (classes). In addition, we prepared the data sets in two ways in terms of splitting training and testing data sets: balanced and imbalanced. For the balanced way, we made both the training and testing data sets of equal sizes, while the imbalanced way involved making the testing data set as 9 times as the training set. In addition, we excluded all the payload and datetime features from the data sets. Next, we applied both RFA and RFE on the data sets and tested the performance. We noticed that RFA outperformed RFE on the imbalanced data sets while performed the same on the balanced data sets.
Chapter 6

Conclusions and Future Work

This thesis has developed a novel feature selection method applicable to the domain of intrusion detection. This method can outperform the leading RFE approach, particularly on problems where there are many, inter-related features and few examples. The thesis was presented in three main parts: 1) deep analysis and diagnosing the limitation of RFE, 2) developing an algorithm to solve the RFE limitation, and 3) the intrusion detection application part with two folds contribution: binary and multi-class detection. The thesis started with an introduction chapter which talked about the Internet and its threats, followed by intrusion detection systems as one of the countermeasures. Next, the chapter discussed how machine learning is employed in intrusion detection. The chapter also introduced feature selection as one of the disciplines that can be used with intrusion detection. The last part of the chapter talked about feature types and feature interdependency. In this way, the first chapter introduced all the main concepts of the thesis. Chapter Two provided a literature review about feature selection and its types. The chapter also discussed the major key-steps of feature selection followed by data classification and using SVMs in classification. The chapter concluded by explaining how the performance of a classifier is evaluated. In summary, the second chapter reviewed the relevant previous work in feature selection, classification and machine learning.
The first significant contribution of the thesis was the identification of a limitation that surrounds RFE regarding interdependent features, which is discussed in part one of the thesis (which starts in Chapter Three). Previous work had reported that RFE feature selection suffers from a limitation that makes RFE remove relevant interdependent features instead of keeping them. The interdependent features are the features that are useless when they are taken individually and useful when they are combined together. Therefore, in order to analyze the above mentioned limitation, we proposed a small problem, called the majority problem. This majority problem comprises 20 interdependent features (used in generating the class label) and 80 noisy features. All the features are generated as random binary bits, so that the number of bits that are “on” follows a binomial distribution. We generated four sizes of majority data sets 300, 600, 800 and 1000 examples to study the behaviour of RFE with different data set sizes. We applied the RFE on all the majority problem data sets. The experiment revealed that RFE really suffers from its inability to keep interdependent features. This test problem and the difficulties that RFE exhibited in solving it, motivated us to further study feature selection for problems where features are interdependent. All of the that was discussed in Chapter Three.

The second contribution is represented in developing a new feature selection method that can work with many features and relatively few examples. Specifically, we developed a new algorithm that works in a recursive fashion in ranking the features. That was the beginning of part two (which was the continuation of Chapter Three). The new algorithm is based on SVMs (same as RFE), and starts from an empty set of features and adds features in a greedy fashion until the stopping criterion is satisfied. The new method is called Recursive Feature Addition (RFA) and it is, like RFE, an example of an embedded feature selection approach. The proposed method was applied on the synthetic majority problem data sets and on 11 benchmark, real-world data sets. We proposed a chronological methodology to test all the real-world data sets in shorter time. We used a fast filter method to pre-rank the features at the beginning. Then, we chose a subset of the resulting ranked features list and the subset consisted of ten equal parts of features (9 parts of badly ranked features and one
part of well ranked features). We set the size of each part to 35 features, therefore the total
is 350 features instead of the entire feature set of each data set. This allowed us to conduct
more experiments and to run multiple iterations of each experiment and conduct statistical
tests. We challenged the feature selection algorithm with extra noisy features to check its
resilience to those noisy features. This challenge involved providing the feature selection
method with 9 times as many bad as good features in the real-world data sets and 4 times
as many noisy as relevant features in the synthetic data sets. In Chapter Four, we presented
the experimental work and the obtained results. The performance has been measured using
accuracy, $F$-measure, $\Delta\%$Accuracy, and $\Delta\%$ $F$-measure. Our method, RFA, showed its
superiority on RFE on all of the data sets of both synthetic and real-world. The results
were then subjected to a significance test using the Mann-Whitney U test. Our method,
RFA, was able to show a statistically significant improvement over RFE results for both
synthetic and real-world data sets, using both accuracy and $F$-measure measures.

The third significant contribution of our work is included in Chapter Five. The chap-
ter focusses on applying the proposed method on the ISCX 2012 intrusion detection data
set using binary and multi-class detection approach. While most previous approaches to
performing feature selection have focused only on the statistical properties of packets, we
opted instead to also try to extract useful features from the payloads. We were able to do
this, because, our method is able to quickly and effectively identify useful features for clas-
sification. Therefore this stage required building a dictionary in order to be used to extract
the feature vector for each example. Then, we followed the same methodology that we did
with the previous real-world data sets as the resulting data set is also a high-dimensional
data set. Motivated by the need of identifying new threats based only on a small number
of observations, we generated four sizes of data sets 25, 50, 100 and 500 examples to check
the behaviour of feature selection with different sizes of data sets. Next, we applied feature
selection on the resulting data sets and we studied the behaviour of both RFA and RFE in
selecting bigram and non-bigram features. By analyzing the results of both RFA and RFE
on the data sets with bigram and non-bigram features, we were able to find a significant
difference on the performance after we expanded the payload features to bigram features using both RFA and RFE methods. The results showed that bigram features are useful in improving the performance. However, both methods were equal in improving the accuracy and $F$-measure of all of the ISCX data sets. Therefore, we decided to test the significance of the results of both methods. By doing the Mann-Whitney U test, we observed that RFA was statistically better than RFE in both metrics and on all the data sets even with small numbers of examples.

The fourth contribution is proposing four additional ranking coefficients in addition to the original one. We implemented RFA using each one of the proposed ranking coefficients and applied RFA on the ISCX data sets. We also reported the performance of RFA using those proposed ranking coefficients after running RFA on the ISCX data set for 30 times. We measured five metrics after we applied RFA using those proposed ranking coefficients: accuracy, $F$-measure, detection rate, false alarm rate and the overall metric. We compared the maximum obtained values for accuracy and $F$-measure, detection rate and the overall metric using all the proposed and the original ranking coefficients. We measured also the minimum obtained value for the false alarm rate for all the ranking coefficients. To provide a visual comparison for the ranking coefficients, we generated the box plot for the detection rate, false alarm rate and for the overall metric for all the ISCX data sets. Furthermore, we conducted a comprehensive statistical significance test of the results all the ranking coefficients on detection rate, false alarm rate, and the overall metric for all the data sets. Since the statistical test could sometimes give no conclusion, could show the results variation, or could give equal results for the ranking coefficients, we generated the partial ordering figure for all the statistical tests. This partial ordering gave us a visual indication about the performance ordering of all the ranking coefficients, for each metric, and for all the ISCX data sets.

The fifth contribution is implementing the multi-class intrusion detection. This approach is important, since knowing the attack class makes it easier to find its treatment. Therefore, we utilized our feature selection algorithm in improving the performance of the intrusion
detection system. Since our feature selection algorithm is based on a binary SVMs classifier, we converted the five class problem into binary tree to solve the multi-class problem. The problem is converted to multiple binary sub-trees so that the incoming traffic is classified multiple times to reach its final classification. We adopted the same approach with RFE to compare the results of both methods. In addition, we employed two ways in preparing the training and testing data sets: balanced and imbalanced ways. For the balanced way, we made the training and testing data sets of equal sizes. The imbalanced way involved making the size of the testing set as 9 times as the size of the training set. The final confusion matrix has been constructed after applying both RFA and RFE using both ways of data splitting. From our experiments, we noticed that RFA performs much better with the imbalanced way of splitting the data sets than the balanced one and it outperformed RFE as well. However, with balanced data sets, RFA and RFE performed exactly the same with all the classes.

6.1 Thesis statement

In this thesis, we experimentally diagnosed a limitation of a very successful feature selection method called RFE. We found in the literature that this method suffers from a considerable limitation regarding interdependent features. In Chen & Jeong (2007), the authors observed that RFE removed a feature from the given data set as it was ranked a bad feature. However, they discovered keeping that feature with other features led to a noticeable improvement in accuracy. This anecdotal observation, sparked our curiosity to study RFE and find a solution for this limitation.

Other studies have indicated that there might be features that are individually useless, but may become useful when combined with others, and they can produce high performance when used together (Guyon & Elisseeff, 2003); (Guyon & Elisseeff, 2006).

The existence of irrelevant features and interdependent features both affect the effectiveness of some feature selection methods especially in high dimensional data sets.
We conducted a deep analysis on RFE to check if it selects interdependent features or not. In addition, we provided a solution to the RFE’s inability to avoid noisy features and to detect feature interdependency, which are both common problems of feature selection methods. Our new method is called Recursive Feature Addition (RFA) and it is an embedded feature selection method.

6.2 Limitations and future work

The limitation of our work lies in that the work has focused only on data sets that contain interdependent features. In addition, the way of evaluating each feature that RFA follows in selecting features makes it time consuming especially with high-dimensional data sets, since it needs to train a SVMs classifier for each feature. Therefore, we opted to use a pre-ranking treatment for these kinds of data sets. We applied RFA on hundreds of features rather than thousands of features due to the time consuming barrier in finishing the analysis. If we had a lot more time, we would have conducted more experiments on more data sets that have criteria different from the criteria that we used in choosing our data sets.

In the future, we plan to apply the RFA on more bioinformatics data sets without taking a subset of the features (without the pre-ranking that we applied in this work). Furthermore, we plan to use GPUs with many cores to take advantage of the parallelism that GPUs offer in order to train multiple SVMs at the same time. In this case we can collect multiple ranking coefficients at the same time. In addition, we plan to use a stopping criterion for RFA such as 50% of the entire number of features.

6.3 Summary

In this thesis we developed a new feature selection method to be used as a new tool and added to the toolbox of any machine learning researcher. We showed experimentally and statistically how RFA outperformed RFE in data sets with many features and relatively
few examples (where there is a possibility for overfitting). That helped us in understanding the problems that fit to both RFA and RFE and which one to use. Therefore, now we can recommend when to use RFA and when to use RFE with a given problem. From our experiments, we can recommend to use RFA whenever the problem has many features and few examples. Conversely, we suggest to use RFE whenever the problem has more examples than features. In addition, in the field of IDS, we showed how important the payload features are in detecting new intrusions. We showed the importance of payload features by expanding them to bigram features and using them with the other header features in detecting intrusions. Both RFA and RFE were able to identify and select bigram features carefully and successfully. Again, RFA showed its superiority to RFE in this intrusion detection problem with many features and only few examples. Our new method, RFA, outperformed RFE in intrusion detection using the ISCX 2012 data set experimentally and statistically. Another interesting contribution of this thesis is proposing four additional ranking coefficients for the RFA feature selection method. We tried all the proposed and original ranking coefficients with RFA on the ISCX 2012 data set and we measured all the evaluation metrics. We conducted different comparisons on the performance of RFA using all the ranking coefficients on the ISCX 2012 data set. Furthermore, we proposed an overall metric that integrates accuracy, detection rate and false alarm rate to evaluate the performance of any IDS. We measured the performance of RFA using five metrics: accuracy, $F$-measure, detection rate, false alarm rate and the overall metric.

6.4 Work implications

One of the implications of this work can be viewed as adding a new feature selection tool to the feature selection toolbox. We have shown that the proposed method can improve the performance of an IDS, if the feature selection method is utilized in an IDS to detect new attacks. Usually with new attacks, there are only few examples and many, many features extracted from the network traffic. Therefore, we have conducted experiments that show that this method will improve the performance of IDS due to the method ability to detect
useful features from huge number of features and few examples. This study helped us in understanding with which problems we use RFA or RFE. In addition, we showed the importance of using payload features in improving the IDS performance. We encoded those payload features using bigram technique and added them to the original set of features. By using RFA, it was able to identify those bigram features successfully and extract some informative features among them. Furthermore, we think bigram technique is effective in representing payload features and converting them to a format acceptable by machine learning algorithms.

It is also worth mentioning here that this work can have a good impact on the intrusion detection due to the ever growing number of new threats and the ever increasing use of the Internet. Therefore, we think that this area will still need much work to be done. Accordingly, we hope future work will build on the successes of this thesis.
Bibliography


URL http://dx.doi.org/10.1007/978-3-642-22922-0_25


198


URL http://dx.doi.org/10.1007/978-3-540-35488-8_1


URL http://dx.doi.org/10.1023/A%3A1012487302797


URL [http://dx.doi.org/10.1007/978-3-540-35488-8_6](http://dx.doi.org/10.1007/978-3-540-35488-8_6)


URL [http://dx.doi.org/10.1007/s10462-011-9205-2](http://dx.doi.org/10.1007/s10462-011-9205-2)


URL http://dx.doi.org/10.1007/s10044-012-0293-7


URL http://dx.doi.org/10.1007/s00138-013-0577-y


URL http://dx.doi.org/10.1002/sec.1269


URL http://arxiv.org/abs/1507.06692


URL http://dx.doi.org/10.1007/s13042-013-0164-6

URL http://dx.doi.org/10.1007/978-3-642-13208-7_61


Appendix A

Mathematical Symbols Used in Thesis

The following are the symbols used to describe the various stages of calculation within the thesis:

\( t \in [0,1] \) a threshold value

\( \hat{c} \) predicted class

\( \hat{c}(x) \) predicted class of example \( x \)

\( r_{cf} \) average feature-class correlation

\( r_{ff} \) average feature-feature inter-correlation

\( \gamma \) acceptable inconsistency rate

\( \xi_i \geq 0 \), ‘chi’ \( i = 1,\ldots,l \)

\( \Theta(x_i) \) ‘Phi’ function that projects \( x_i \) into a higher dimensional space

\( \alpha \) ‘alpha’ a control value
α ‘alpha’ vectors that results from training SVMs classifier

δ ‘delta’ the Kronecker symbol

λ ‘lambda’ positive constant