AN OPTIMIZED FEATURE SET FOR ANOMALY-BASED INTRUSION DETECTION

WASSWA HASSAN

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2019

AN OPIMIZED FEATURE SET FOR ANOMALY-BASED INTRUSION DETECTION

WASSWA HASSAN

DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF COMPUTER SCIENCE (APPLIED COMPUTING)

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2019

UNIVERSITY OF MALAYA

ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: Wasswa Hassan

Matric No: WOA170012

Name of Degree: Master of Computer Science (Applied Computing)

Title of Project Paper/Research Report/Dissertation/Thesis ("this Work"): An Optimized Feature Set for Anomaly-Based Intrusion Detection

Field of Study: Information Security

I do solemnly and sincerely declare that:

(1) I am the sole author/writer of this Work;
(2) This Work is original;
(3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the Work and its authorship have been acknowledged in this Work;
(4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;
(5) I hereby assign all and every rights in the copyright to this Work to the University of Malaya ("UM"), who henceforth shall be owner of the copyright in this Work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained;
(6) I am fully aware that if in the course of making this Work I have infringed any copyright whether intentionally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature Date:

Subscribed and solemnly declared before,

Witness's Signature Date:

Name:

Designation:

AN OPTIMIZED FEATURE SET FOR ANOMALY-BASED INTRUSION DETECTION

ABSTRACT

The ubiquity of the internet and its enhanced transmission speed has led to establishment of many networks by various businesses across the vertical market. Currently, a huge number of organizations across the globe conduct business transactions over the internet. This has amplified the volume of network traffic flowing in and out of business information systems making real-time analysis a very hectic task for network administrators. Consequently, the escalated number of business transactions has allured an outrageous number of cyber attackers to the business' information systems. The hackers use advanced techniques and tools to launch new and well refined attacks every day. To enable detection of new and unknown attacks, various research efforts have focused towards enhancing anomaly-based network intrusion detection systems (ANIDS). One way to optimize the performance of ANIDSs is to identify only relevant features for training the intrusion detection system (IDS). This is since modern traffic constitutes a large number of attributes many of which are irrelevant for classification of traffic as either benign or anomaly. Having only relevant features can greatly reduce model complexity making it more interpretable, improve IDS performance in terms of speed and accuracy and avoid over fitting. To this end, this research proposed a feature set that optimizes the performance of ANIDSs by utilizing various feature selection techniques, i.e. filter, wrapper and embedded methods, for enhanced information security. The proposed feature set is evaluated using five machine learning classifiers trained and tested on UNSW-NB15 dataset. The proposed feature set recorded better detection results with regard to accuracy, precision, recall, false positive rate (FPR) and detection time compared to feature sets obtained by application of a single feature election method. Random forest classifier outperformed the other four classifiers used in this research i.e.

Decision tree (DT), AdaBoost, Extra trees classifier and Gradient boosting classifier with regard to accuracy, precision, recall and false positive rate (FPR) while DT recorded shortest detection time.

Keywords: intrusion detection, intrusion detection systems, machine learning, feature selection, UNSW-NB15

SET CIRI YANG OPTIMUM UNTUK PENGESANAN PENCEROBOHAN BERASASKAN ANOMALI

ABSTRAK

Keadaan internet dan kelajuan transmisi yang dipertingkatkan telah membawa kepada penubuhan sejumlah besar rangkaian oleh pelbagai perniagaan di seluruh pasaran menegak. Pada masa ini, sebilangan besar organisasi di seluruh dunia menjalankan transaksi perniagaan melalui internet. Ini telah meningkatkan jumlah trafik rangkaian yang mengalir keluar dan masuk dari sistem maklumat perniagaan, yang membuatkan analisis masa semasa, tugas yang sangat sibuk untuk pengurus rangkaian. Akibatnya, bilangan transaksi perniagaan yang semakin meningkat telah menarik sejumlah besar penyerang siber ke sistem maklumat perniagaan. Penggodam menggunakan teknik dan alat canggih untuk melancarkan serangan baru dan terancang setiap hari. Untuk membolehkan pengesanan serangan novel, pelbagai usaha penyelidikan yang tertumpu ke arah peningkatan sistem pengesanan pencerobohan rangkaian berasaskan anomali (ANIDS) telah diadakan. Satu cara untuk mengoptimumkan prestasi ANIDS adalah dengan mengenal pasti hanya ciri-ciri yang berkaitan untuk melatih IDS. Ini disebabkan hakikat bahawa lalu lintas moden merupakan sebilangan besar sifat yang banyak tidak relevan untuk klasifikasi lalu lintas sama ada benigna atau anomali. Hanya dengan mempunyai ciri-ciri yang berkaitan yang akan dapat mengurangkan kompleksiti model menjadikannya lebih mudah difahami, prestasi IDS dipertingkatkan dari segi kelajuan dan ketepatan dan lebih muatan dielakkan. Untuk meningkatkan kawalan keselamatan maklumat, penyelidikan ini mencadangkan satu set ciri yang mengoptimumkan prestasi ANIDS dengan menggunakan pelbagai teknik pemilihan ciri, iaitu penapisan, kaedah pembungkusan dan kaedah terbenam.Set ciri yang dicadangkan dinilai menggunakan lima kelas pembelajaran komputer yang dilatih dan dinilai pada dataset UNSW-NB15. Set ciri yang dicadangkan adalah dengan mencatatkan hasil pengesanan yang lebih baik dari segi ketepatan, ketelitian, mengingat, kadar positif palsu dan masa pengesanan berbanding dengan set ciri yang diperolehi dengan menggunakan kaedah pemilihan ciri tunggal. Pengelas hutan secara rawak mengatasi empat pengelas yang lain yang digunakan dalam penyelidikan ini iaitu DT, AdaBoost, pengelas pokok tambahan dan penggred menaikkan pengelas dari segi ketepatan, ketelitian, mengingat dan FPR manakala DT mencatatkan masa pengesanan terpendek

Kata kunci: pengesanan pencerobohan, sistem pengesanan pencerobohan, pembelajaran mesin, pemilihan ciri, UNSW-NB15

ACKNOWLEDGEMENTS

I begin by thanking the Almighty Allah for having blessed me with good health, strength and wisdom throughout my master’s study.

I feel immense pleasure to extend my sincerest gratitude to my supervisor Assoc. Prof. Dr. Ainuddin Wahid Abdul Wahab for the invaluable support and encouragement you have always extended to me through this research. Your endless guidance has availed me the opportunity to broaden my knowledge and professional experience while giving me a sense of direction for future challenges. May the Almighty Allah always bless your work.

Let me utilize this opportunity to extend my sincere gratitude and appreciation to Islamic Development Bank (IsDB) Group for offering me a full scholarship for master’s study. This opportunity has not only let me broaden my knowledge base but also allowed me to experience the other side of the world. I pray that the Almighty Allah rewards you abundantly. In particular I would like to extend my gratitude to Dr. Nazar Elhilali for the endless guidance and support you have always given me whenever I contacted you. May Allah grant you good health.

To my beloved father Haji Sulaiman Ssemambo, mother Nalongo Aidah Nakibuuka and siblings, I can’t thank you enough for the invaluable love, support and encouragement you have tirelessly extended to me during my study. No words can express my feelings and how grateful I’m for having people like you in my life. May Allah always guide and protect you.

Finally, let me extend my heartfelt appreciation to all my friends for the support and guidance through this study. In a special way I would like to extend my appreciation to my two special friends Ma Yuzhan and Mohamed EL Gohary. Your invaluable friendship has always made me feel at home. May Allah be pleased with you.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

AI	:	Artificial Intelligence
ANIDS	:	Anomaly-based Network Intrusion Detection
AWID	:	Aegean Wi-Fi Intrusion Detection dataset
CART	:	Classification and Regression Tree
CFS	:	Correlation-based Feature Selection
DT	:	Decision Tree
EM	:	Expectation Maximization
ET	:	Extra Trees
FPR	:	False Positive Rate
GA	:	Genetic Algorithm
GBC	:	Gradient Boosting Classifier
GR	:	Gain Ratio
IA	:	Information Assurance
IDS	:	Intrusion Detection System
IG	:	Information Gain
IoT	:	Internet of Things
KNN	:	K-Nearest Neighbor
LDA	:	Linear Discriminant Analysis
LR	:	Logistic Regression
MI	:	Mutual Information
ML	:	Machine Learning
MLA	:	Machine Learning Algorithm
NB	:	Naïve Bayes
NIDS	:	Network Intrusion Detection System

NP	:	Non-polynomial
ONS	:	Office of National Security
OOB	:	Out-Of-Bag
PCA	:	Principal Component Analysis
RF	:	Random Forest
RFE	:	Recursive Feature Elimination
RO	:	Research Objective
SCAN	:	Spatial Clustering of Applications with Noise
SMOTE	:	Synthetic Minority Oversampling Technique
SOM	:	Self-Organizing Map
SVM	:	Support Vector Machine
TPR	:	True Positive Rate
χ^2	:	Chi-Square

LIST OF APPENDICES

CHAPTER 1: INTRODUCTION

1.1 Background

The ubiquity of internet with its enhanced data transmission speeds has resulted into an increasing number of networks established by a myriad of business entities across the vertical market. Currently many businesses store, process and perform all sorts of transactions over highly interlinked information systems via the internet (Bendovschi, 2015). According to the "IoT: number of connected devices worldwide 2012-2025" report, the number of interconnected devices is estimated to rise from 15.41 billion in 2015 to 75.44 billion by 2025. This fast growth of internet has amplified the complexity of securing information systems in terms of realizing the information assurance principles of integrity, availability and confidentiality across the globe. This is because many cyber criminals are taking advantage of the fast growth of the internet to launch a very huge number of well refined attacks to the various entities' networks that are difficult to detect (Janarthanan & Zargari, 2017). In addition, some of the cyber criminals are highly skilled experts with highly sophisticated tools and advanced systems that can easily penetrate any network or information system if not well protected. This makes many networks and information systems vulnerable to cyber-attack (Wang, Xu, Lee, & Lee, 2018).

To guard network infrastructure and information systems against cyber-attacks, intrusion detection systems (IDSs) with varying detection methods are deployed by network administrators. The prime role of any IDS is to detect and signal the presence of a break-in attempt into an information system. An IDS serves a vital role in shielding any network infrastructure and information system from malicious attacks. It also serves a crucial part in stopping any illegal access to computers and network systems hence enhancing the security of these systems (Aburomman & Reaz, 2016). Intrusion detection systems study, observe and check the behavior of the network or information system and system users to identify or detect possible security threats and attacks (Al-Jarrah, et al., 2014). It provides a second layer of defense to signal break-in attempts that the firewall fails to stop.

The two types of IDSs are Network-based Intrusion detection systems (NIDSs) and Host-based IDSs (HIDSs). In addition, IDSs are categorized into two classes based on detection methods; anomaly-based IDSs and signature-based or misuse IDSs. The signature-based IDSs detect break-ins by comparing the incoming traffic with a record of already known attack signatures for a match. On the other hand, anomaly-based IDSs analyze the characteristics of, and establish a profile for benign traffic. Any divergence from this profile is marked as a break-in attempt (Hajisalem & Babaie, 2018; Idowu, Maroosi, Muniyandi, & Othman, 2013). Anomaly-based NIDSs have an advantage of detecting unknown attacks over signature-based IDSs (Moustafa & Slay, 2016). On the other hand, signature-based NIDSs produce better detection accuracy with very low false detection signals but for only known attacks (Shah & Issac, 2018). However, despite the high detection accuracy of signature-based IDSs, it is practically impossible for the administrator to know all the possible attack signatures and for this reason a lot of research has focused at improving anomaly-based IDSs and is the main reason for selecting anomaly-based NIDSs for this research.

Considering their ability to detect novel attacks or break-ins attempts, more research efforts have been dedicated to anomaly-based NIDSs over signature-based NIDSs in the area of information and cyber security (Moustafa & Slay, 2016; Min, Long, Liu, Cui, & Chen, 2018). However, for most of the proposed anomaly-based NIDSs the performance has been assessed using the famous KDD'99 intrusion dataset which were generated 20 years ago and seems too old and obsolete for use in the current attack environment. This is due to the fact that the features used in this dataset and its attack categories do not fairly represent state-of-the-art attack traffic features (Moustafa & Slay, 2016). According to

(Janarthanan & Zargari, 2017), out of the 41 features in the KDD’99 and out of 42 features in the UNSW-NB15 intrusion datasets, there are only five common features (i.e. duration, service, protocol type, source bytes and destination bytes). UNSW-NB15 dataset is a more recent dataset generated in 2015 using a tool known as IXIA PerfectStorm at the Cyber range lab of the Australian Centre for Cyber Security (ACCS) (Moustafa & Slay, 2015) and better represents the state-of-the-art attack traffic as compared to KDD’99 or NSL-KDD dataset. In addition, the number of novel attack threats to information systems is overwhelmingly increasing day-by-day. Hackers design new attack strategies every day and the attack traffic come with new attributes that are not included in older datasets like KDDCup99 or NSL-KDD datasets (Janarthanan & Zargari, 2017; Moustafa & Slay, 2015).

To improve the detection performance of any intrusion detection model in terms of detection accuracy with low false detection signals, there is a need to identify and select attributes that are pertinent in separating attack traffic from benign traffic. Identifying relevant features helps to reduce the risk of model over fitting and improves detection performance while reducing system resource requirements like classification time, model training time, CPU and memory usage among others (Gul & Adali, 2017). Furthermore, building a model based on relevant features reduces its complexity and consequently improving its interpretability. To this end this research proposed a feature set for optimizing intrusion detection performance by examining various feature combinations obtained using various feature selection techniques.

1.2 Problem statement

Today high dimensional network traffic flows in and out of business information systems in large volume and high velocity. Analyzing all traffic for attack detection has become very hectic and costly in terms of detection performance, detection time and system requirements. Various efforts have been invested in improving the detection performance of IDSs by training the detection models on a subset of features from the intrusion datasets. However, most of the models have been trained based on the famous KDDCup99 benchmark dataset which is almost twenty years old and does not perfectly represent new generation network traffic (Kulariya, Saraf, Ranjan, & Gupta, 2016; Chang, Li, & Yang, 2017; Aljawarneh, Aldwairi, & Yassein, 2018). This has made most of these models obsolete for use in the production environment. In addition, studies based on newer datasets, like UNSW-NB15 dataset, have not balanced between detection accuracy and detection time. In their work (Janarthanan & Zargari, 2017), proposed a subset of five features from the UNSW-NB15 dataset which would greatly improve detection time owing to its small dimensionality but registered low detection accuracy with both training and test datasets. Similarly, (Anwer, Farouk, & Abdel-Hamid, 2018) used the wrapper method and GR ranking and reported J48 classifier to produce the best detection accuracy of 88% with 18 features based on UNSW-NB15 dataset. In their work with all 42 features, which is an inefficient approach, of the UNSW-NB15 dataset and two traffic classes, (Belouch, Hadaj, & Idhammad, 2018) the best detection accuracy of 97.49% was registered by random forest while decision tree classifier registered least detection time of 0.13s. Therefore, the problem statement of this research can be summarized as:

There is a need to identify an optimized feature set for anomaly-based intrusion detection systems based on state-of-the-art network traffic.

1.3 Research Motivation

Cybercrime has become the biggest threat to business over the last decade and many businesses have closed because of cyber-attacks while others are still struggling to recover from cyber-attack incidents. The government of UK in its report revealed that 74% of small businesses in the UK were faced with a cyber-security breach while 90% of big enterprises were potential targets in 2014 (Nguyen, et al., 2018). Similarly, according to a report from Crime Survey for England and Wales (CSEW) and National Fraud Intelligence Bureau (NFIB), there was a fall in reported fraud incidents from 3.6 million in 2016 to 3.2 million in 2017. However, it also reports that 56% of fraud was cyber related. The ONS report also revealed that regardless of the general decrease in fraud incidents in 2017, malware and fraud incidents against business rose up to 63% in 2018 (http://www.computerweekly.com November 1, 2018). This threat has been amplified by the new trend of computing which involves cloud computing services, big data computing environment and internet of things (IoT). According to (*https://www.interpol.int*, November 1, 2018) Interpol stated that:

"*Cybercrime is a fast-growing area of crime. More and more criminals are exploiting the speed, convenience and anonymity of the Internet to commit a diverse range of criminal activities that know no borders, either physical or virtual, cause serious harm and pose very real threats to victims worldwide*".

Owing to the above reports, it can be concluded that there is a need for optimizing ANIDSs to enable real-time detection of the state-of-the-art intrusions as they attempt to flow in and out of information systems. This continuously rising need was the main motivation for undertaking this research.

1.4 Research Questions

RQ1: What feature selection methods are commonly used for attribute selection in machine learning models?

RQ2: What feature set can optimize an intrusion detection system's performance in terms of detection rate?

RQ3: Does combining multiple feature selection methods produce better detection performance than using features from a single selection method?

1.5 Research Objectives

RO1: To study the various feature selection methods commonly used for attribute selection in machine learning models

RO2: To propose a feature set for optimizing anomaly-based intrusion detection systems in terms of detection rate.

RO3: To evaluate the performance of the proposed feature set against feature sets selected by single selection methods.

1.6 Research scope

This research focused on identifying an optimized feature set for anomaly-based intrusion detection system using three feature selection methods which include wrapper, embedded and filter methods. The scope is limited to features in the UNSW-NB15 data set. In addition, supervised learning algorithms are used for this research since the training dataset is already labelled. The evaluation of all feature subsets in this research is limited to five machine learning algorithms which include decision tree (DT), random forest (RF), extra trees (ET), gradient boosting (GB) and AdaBoost classifiers. These classifiers were selected after examining nine classifiers including five single classifiers (i.e. KNN, SVM, LR, NB and DT) and four ensemble classifiers (i.e. RF, Extra trees, Gradient boost and adaBoost classifier). Performance evaluation of all feature subsets is limited to five metrics including detection accuracy, precision, recall, FPR and detection time.

1.7 Research significance

The significance of this research is two-fold. First, the research identified and examined the different feature selection methods commonly deployed for intrusion detection models. This will enable the various researchers in the field of machine learning and intrusion detection to easily determine which feature selection methods to deploy in their studies since the research highlights the benefits and drawbacks of each approach. Second, the research identified a feature set for anomaly-based intrusion detection models that can give rise to better detection performance of anomaly-based IDSs and consequently will enhance the security of information systems.

1.8 Thesis Structure

The thesis organization is as follows.

Chapter 2: Gives a detailed review of intrusion detection systems, highlighting the types of IDSs and the various attack detection methods. The chapter also discusses the different conventional methods and machine learning techniques. Furthermore, the various feature selection methods deployed in previous studies for intrusion detection are reviewed hence answering RQ1.

Chapter 3: In this chapter, a detailed explanation of the entire research design/process is presented. It also highlights how data pre-processing was conducted and highlights the overall experimental environment setup. It further details the criterion for selection the best classifiers and how the optimized feature set was proposed at hence answering RQ2.

Chapter 4: This chapter presents major research results.

Chapter 5: This chapter discusses the core discoveries/outcomes of this research in comparison with findings of previous related studies. RQ3 of this research is answered in this chapter.

Chapter 6: This chapter gives the conclusive remarks about the overall study by giving an account of how the various research objectives were achieved, the major contributions and limitations of the research and finally gives the direction of future work.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

This chapter provides a detailed formal assessment on the relevant literature for gaining an insight into the related work done in the area of intrusion detection, machine learning and feature selection methods. The review is broadly classified into four sections including intrusion/attack detection techniques, machine learning classifiers, feature selection schemes and evaluation metrics.

An information system security breach can be either internal or external. In other words, a malicious attack can be launched from within the organization's private network by an internal network user or from outside the network (Syam, & Venkata, 2017; Roshan, Miche, Akusok, & Lendasse, 2018). Over the past two decades, a variety of security measures and policies have been deployed to realize the triad of information assurance (IA) principles i.e. integrity, confidentiality and availability. Some of these measures include firewalls, access control approaches like use of passwords, antivirus software and constantly updating and upgrading both system as well as application software among others (Li, Zhang, Peng, & Yang, 2018). However, these conventional measures have constantly exhibited many weaknesses and have continuously exposed information systems to various cyber-attacks. In addition, none of the traditional measures is capable of detecting both external and internal intrusions at the same time. This is where IDSs become a better choice for deployment.

2.2 Intrusion Detection System

The idea of IDSs was initially put across by James Anderson in 1980. Intrusion detection is a process which involves analyzing network traffic or an information system's behavior in order to identify malicious traffic or break-in attempts. Kabir, Hu, Wang, & Zhuo, (2018), defined intrusion detection as a technique of identifying illegal activities on an information system. Intrusion detection is also defined as the act of tracking the user, system and network activities in order to distinguish attack activities from normal behavior (Roshan, Miche, Akusok, & Lendasse, 2018). In their work, (Bhosale & Mane, 2015) defined this concept as a process that involves inspecting events that take place on a network for revealing any break-in attempt. Intrusion detection is accomplished through use of intrusion detection systems commonly abbreviated as IDSs.

An IDS is a hardware or software component which studies and oversees the activities that are carried out on a computer system or network, examines the characteristics of network traffic and user behavior and signals the presence of a break-in attempt, an information security policy violation or any potential malware or cyber-attack (Kabir, Hu, Wang, & Zhuo, 2018; Colom, Gil, Mora, Volckaert, & Jimeno, 2018). It examines the information or network system for possible intrusion attempts. Because of their ability to detect both internal and external attacks, IDSs serve to provide a second layer of protection to an organization's information system (Roshan, Miche, Akusok, & Lendasse, 2018). This way, IDS's are indispensable in deterring any illegal access to computers and network systems hence enhancing the security of these systems (Aburomman & Reaz, 2016).

2.2.1 Types of IDS

Intrusion detection systems are broadly categorized into host-based intrusion detection systems (HIDSs) and network-based intrusion detection systems (NIDSs).

Host-Based Intrusion Detection System (HIDS)

This type of IDS runs on a single computer or device and monitors the activities running on that particular system for potential malware, attack activities or information assurance policy violations (Sun, Hahn, & Liu, 2018). It detects unauthorized access to systems on which it is installed and generate alerts to the system or security administrators (Wagner & Soto, 2002). In simple terms HIDS sensors, often referred to as agents are typically installed on individual devices that are considered susceptible to potential attacks. These sensors alert the system/security administrators in case of any suspicious traffic or event on the host on which they are installed.

Host-based IDSs monitor kernel events, log files, system files and connections to the systems (Niksefat, Kaghazgaran, & Sadeghiyan, 2017).

In kernel based intrusion detection, HIDSs analyze the arguments passed to system calls and their sequences, patterns of calls to system processes and their execution durations, user access and system use patterns including access sequences, duration and time (Sun, Hahn, & Liu, 2018). In addition, the pattern and density of alterations made to system binaries in terms of system logins are also examined and recorded for possible malicious attacks. Kernel based intrusion detection is crucial in detecting operating system based security threats.

The other aspect of HIDS is file system monitoring. Here the different attributes of the files stored on the system are examined and recorded for future reference in case of any modifications (Sun, Hahn, & Liu, 2018). The files' attributes are constantly compared with the recorded attributes and any suspicious event affecting the status of the file is reported. Some of the file attributes may include file permissions, file owner and/or group, file size, creation date, last accessed date, last modified date, file location, number of files in the location, file type, access frequency or pattern among other attributes. Any deviation in the established or known file profile is considered an attack and the authorities are signaled about the change.

Host based IDSs can also operate by monitoring system log files for unusual or abnormal events (Niksefat, Kaghazgaran, & Sadeghiyan, 2017; Sun, Hahn, & Liu, 2018). Log files store a record of all events that take place on a particular system. The IDS examine the log files on a regular basis and alerts the system administrator in case an unusual event is detected. The analysis of log files may be based on pattern matching which can be achieved using regular expressions or can be based on the correlation between the various events that take place on the host system.

Lastly, the fourth aspect of HIDSs is connection analysis. Here the HIDS examines network packets flowing to and from the computer running the IDS (Colom, Gil, Mora, Volckaert, & Jimeno, 2018). However, it is not concerned about packets directed to other hosts on the network and does not perform any pattern matching on them for attack detection. This implies that for effective intrusion detection using HIDS approach each host must have an updated IDS installed on it (Sun, Hahn, & Liu, 2018). In this approach of intrusion detection, the IDS monitor only activities that take place on its network interfaces. An IDS which monitors all activities running on the network is known as a network based intrusion detection system and is the main focus of this research.

Due to increased reliance of many business operations on networks, HIDSs have proved insufficient in protecting information systems against quite a large number of attacks. This is because many of the modern attacks target the network as a whole instead of a single host (Bijani & A., 2008). Therefore, a host-based IDS will not detect attacks like ping of death, DNS spoofing, TCP hijacking and many others which do not target individual machines on the network but the whole network.

Network-based Intrusion Detection System (NIDS)

This type of IDS captures and examines network packets being transmitted on a network segment (Bhosale & Mane, 2015). Unlike HIDS, a NIDS collects information from the network itself and its operation is independent of the underlying OS. The NIDS sensors inspect or analyze the attributes of traffic packets flowing in and out of the network segment. They are installed at strategic locations within a network segment where they can easily capture all packets flowing in and out of the network (Naik, Diao, & Shen, 2016). They copy the wiretapping approach by listening to communication links for incoming and outgoing packets (Syam, & Venkata, 2017). NIDSs analyze traffic using two approaches which include flow-based and packet-based analysis. In packet-based approach the entire packet is examined by inspecting the header information together with the payload. On the other hand, flow-based analysis covers only header information like source and destination IP addresses, source and destination port numbers, among other flow-based features. A full comparison between the two IDS types is given in Table 2.1.

Table 2. 1: Comparison between HIDSs and NIDSs

Host-based Intrusion detection system (HIDS)	**Network-based Intrusion detection system (NIDS)**
Installed on a single host and cannot detect attacks directed to other hosts on the network.	Installed at strategic spots on the network and inspects network packets to and from all devices on a network segment
Often affected by the underlying operating system	Independent of the operating systems running on individual network devices

Table 2. 1 Continued

Host-based Intrusion detection system (HIDS)	**Network-based Intrusion detection system (NIDS)**
Inspects and collects its data from activities and files on a single host	Inspects and collects its data from the network itself
Not ideal for large networks since installation, configuration and monitoring the performance of each individual HIDS consumes a lot of production time.	Ideal for both small and large networks since it is installed and configured once for all network devices.

2.2.2 Intrusion Detection Methods

The two main methods include;

Signature-based detection

In this mode of detection records of all known malicious signatures are maintained in a database often referred to as a rule set (Bhosale & Mane, 2015; Karami, 2018). Any traffic instance that flows in or out of the network is compared with the stored attack signatures for a match. If it matches any of the known bad activities, it is labelled as a potential break-in and the administrator is signaled about the event (Naik, Diao, & Shen, 2016). It can be noted that this method of attack detection is attack oriented, that is, it has knowledge about attack patterns but not normal traffic. Some of common examples of signature-based IDSs include Snort, Suricata and Bro-IDS among others. Figure 2.1 shows the mechanism of operation of a signature-based IDS.

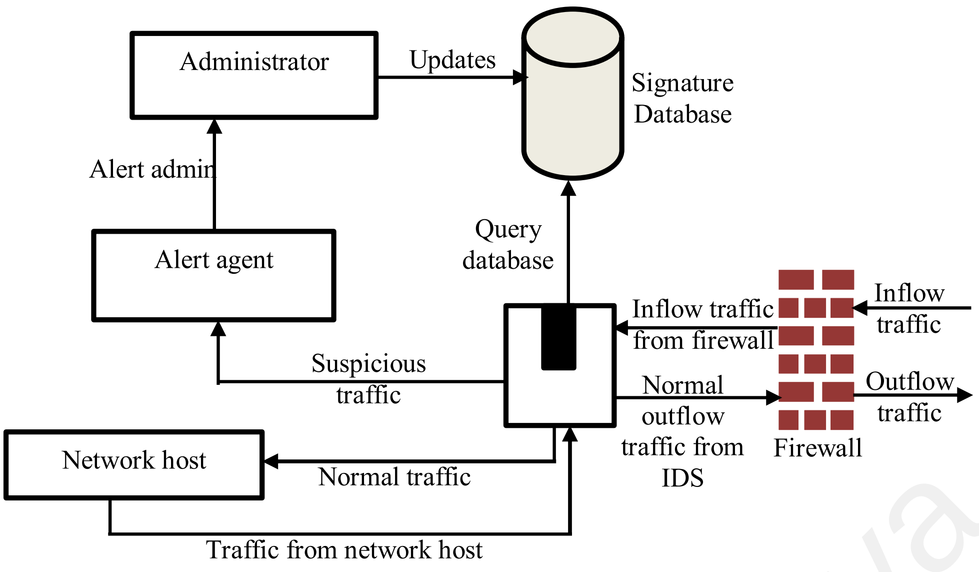

Figure 2. 1: Signature-based IDS

Misuse-based detection is considerably accurate for detection of attacks whose signatures are already known and recorded in the signature database and hardly generate false attack warnings (Xin, et al., 2018; Min, Long, Liu, Cui, & Chen, 2018). However, this method has two major weaknesses; one is that they cannot detect intrusions which span more than one packets considering their mode of operation. Nonetheless, today's intrusion traffic is very sophisticated and sometimes calls for examining signatures of many packets (Hubballi & Suryanarayanan, 2014). The second weakness is the inability to detect novel attacks like zero-day malware which are not contained in the database (Min, Long, Liu, Cui, & Chen, 2018). This implies that the security/system administrator has to constantly and manually add new attack signatures to the database in order to ensure the security of the network (Xin, et al., 2018). This is impractical in the real world as it is hard for the administrator to know patterns of all potential attack.

To worsen matters today's intruders or hackers use advanced tools to develop new and well refined attacks every day. Therefore, misuse-based detection is not an ideal approach for intrusion detection in the modern computing environment where a large volume of new traffic signatures, both malicious and normal, flow in and out of the network infrastructure at astounding speeds (Syam, & Venkata, 2017). This is because it is hard for the security administrator to study all types of traffic in order to update the database on the fly and therefore, it leaves the network or information system vulnerable to the modern hostile attack. In addition, due to the advanced nature of the modern attack generation tools and the level of expertise of today's hackers, attack traffic can easily be engineered to masquerade as normal traffic by hiding known attack signatures from the IDS. This means that many malicious attacks can be cleared by the IDS as clean traffic and since a single undetected attack can be extremely disastrous to any information system today, many enterprises are investing in anomaly-based IDSs.

Anomaly-based detection

In this method, the IDS studies the characteristic features of normal traffic and creates a normal profile (Hubballi & Suryanarayanan, 2014). Each traffic instance that flows in or out of the network is compared with the established normal profile for attack detection. Any traffic event that fails to concur with the established benign traffic profile is flagged as a potential break-in and the administrator is notified (Karami, 2018). Owing to the fact that this method works by considering normal traffic behavior, it is also referred to as behavior-based detection.

Figure 2.2 shows a simple illustration of how an anomaly-based IDS operates in the detection of break-in attempts.

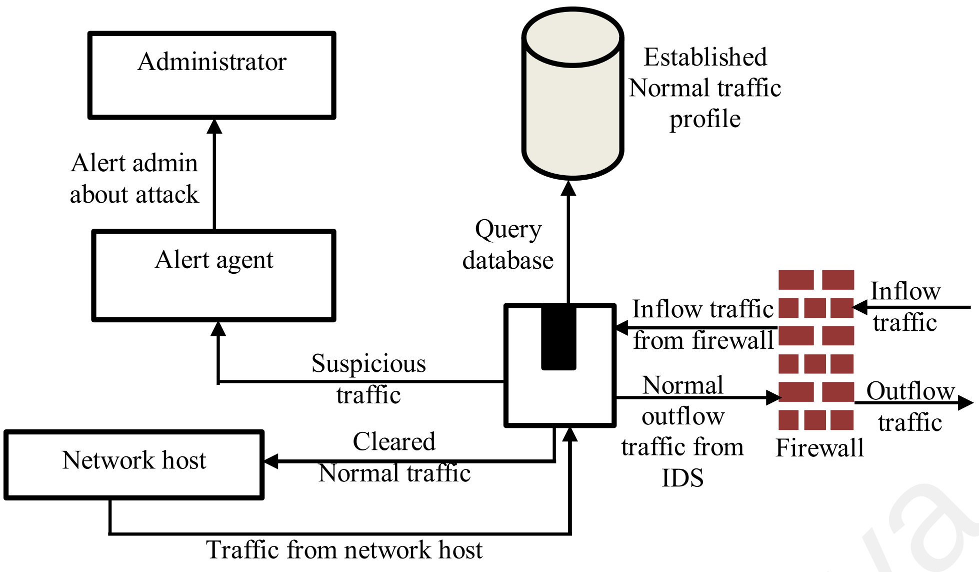

Figure 2. 2: Anomaly-based IDS

Unlike misuse-based detection which is attack oriented, anomaly-based detection mainly focusses on normal traffic to create a normal behavior pattern (Naik, Diao, & Shen, 2016). And because any deviation from the normal behavior is considered as an outlier and marked as an attack, an anomaly-based network intrusion detection system (ANIDS) has an advantage of the ability to detect novel attacks over knowledge-based IDS (Min, Long, Liu, Cui, & Chen, 2018). However, owing to the dynamic nature of the current day traffic, ANIDSs cannot establish a comprehensive normal profile for all normal traffic. This implies that even normal traffic with a slight variation from the created harmless profile is tagged as an intrusion attempt which results into a high rate of false alarms (Karami, 2018).

Table 2. 2: Comparison between signature-based and Anomaly-based IDS

Signature-based IDS	Anomaly-based IDS
Constitutes a database of known attack signatures which must be regularly updated for new attacks	Creates a profile for normal traffic by studying patterns of normal user events/behavior
Misuse/attack oriented i.e. compares traffic instances with stored attack patterns or signatures.	Normal behavior/traffic oriented i.e. it matches traffic instances with the created normal behavior profile.
More accurate for known signature detection but cannot detect unknown/new attacks whose pattern is not stored in the database	Less accurate with a high false detection rate but can detect both known and unknown/new attacks
Signature definition is dependent on administrator's knowledge and experience of the different types of attacks and their attack patterns	Creation of normal profile depends on statistical models, correlation between data packets attributes and the traffic class, data mining and machine learning algorithms used.

In anomaly-based intrusion detection a machine learning algorithm learns the behavior of normal traffic from a training intrusion data sample and uses the results to detect intrusions.

2.3 Overview of Machine Learning

Over the years it has been known that for some task to be accomplished using a computer, an algorithm has to be developed and fed into the computer for that particular task. Many algorithms have been developed and implemented for various tasks like sorting a list of items, searching for a particular object from a list of objects among others. In all these algorithms input and output are known, and the programmer has prior knowledge of how to convert the input into the desired output (Alpaydin, 2010). For example, in a search algorithm the input is a list of items and a particular item to search for. While the output is the location of the item if found or some message to indicate that the item is not a subset of the list if not found.

However, due to advancement in technology and the high level of interconnectivity between devices, businesses collect huge volumes of data flowing into their information systems in various formats but there are no specific algorithms to convert data into the desired output for many of the collected data instances. In addition, the programmers and system administrators have no any idea of how the input can be converted into desired output. For instance, in intrusion detection, a large number of traffic instances can be collected from a networked environment. The administrator knows that each of the traffic instances represents either normal traffic or some kind of attack but he doesn't know how to program the computer to distinguish between the two classes of traffic and there is no particular algorithm developed to perform this task (Alpaydin, 2010). This is where ML comes to our rescue.

The idea of machine learning was introduced way back in 1950's and Arthur Samuel (1959) defined machine learning as a field of study which allows computers to learn without being explicitly programmed. Tom Mitchel (1989) came up with a more formal definition of machine learning. His definition goes as "a computer program is said to learn from experience E with respect to a task T and some performance measure P if its performance on task T, as measured by P improves with experience E". Machine learning can further be defined as a branch of artificial intelligence (AI) that allows computer algorithms performance to improve with experience (Nguyen, Costa, Cios, & Gardiner, 2011).

Therefore, in simple terms machine learning can be defined as an application of AI that gives computers the power to automatically extract knowledge and enhance their performance from example data or experience without any explicit programming (Smola, & Vishwanathan, 2008). ML targets implementation of models or creation of programs that can extract knowledge from the available example data and use the extracted knowledge for making decision on new and unseen data. That is, it aims at identifying the underlying structure within huge, high dimensional datasets (Lou & Tsai, 2008; Stimpson & Cummings, 2014). This way, models for both labelled and non-labelled datasets can be automatically established without programming the machine.

Basing on a combination of statistical measures like mean, standard deviation, correlation between the target output field and the different input attributes, distance between the various data samples in the input vector space among other factors, machines with the help of learning algorithms study input data and extract important patterns. Using the extracted patterns, the machine automatically generates an algorithm or model for transforming the input into desired output without any explicit programming. The learning algorithms automatically change their default parameters and draw inferences basing on the identified patterns in highly complex and large datasets (Xin, et al., 2018; Al-Jarrah, et al., 2014). It is worth noting that the accuracy of the generated algorithm or model strongly depends on the correctness and volume of the supplied example data (commonly referred to as training data), independent attributes, learning algorithm and learning method used.

Machine learning plays a very crucial role in the area of artificial intelligence and is widely applied in various fields for various purposes ranging from intrusion detection in information security, weather forecasting, customer recommendation systems, search engines, spam filtering applications, cancer detection in medicine, fraud detection in finance and banking industry among other fields (Alpaydin, 2010).

Machine learning is broadly divided into four classes which include supervised learning, semi-supervised learning, unsupervised learning and reinforcement learning.

2.3.1 Supervised Learning

This type of learning is concerned with predicting a specific value in case of regression (i.e. when the target output is continuous) or assignment of a label to a new data instance for classification (i.e. when the target output is categorical). Its focus is learning the association between independent (input) variables and a dependent (response) variable (Kaneko, 2018). Each of the training instance is already assigned corresponding response value for regression or label for classification (Liu, et al., 2018; Lou & Tsai, 2008). In other words, the training dataset is well labelled with the correct output responses. For instance, in fraud detection system the training dataset can be a collection of online transaction records with each transaction already labelled as either fraudulent or non-fraudulent.

Figure 2.3 illustrates the supervised learning process for classification

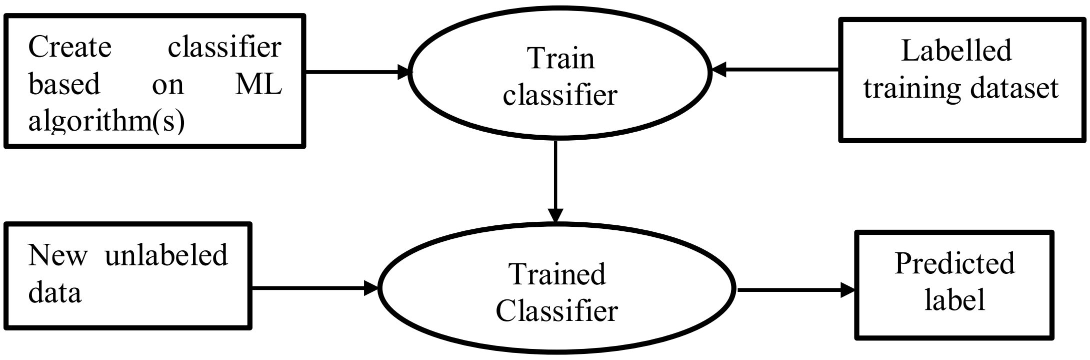
Create classifier based on ML algorithm(s)
Train classifier
Labelled training dataset
New unlabeled data
Trained Classifier
Predicted label

Figure 2.3: Supervised learning process

2.3.2 Unsupervised Learning

This learning type is often used in data mining applications. Its main target is to explore unlabeled datasets so as to understand the underlying structure or distribution (Cao, Qian, Wu, & Wong, 2019; Kaneko, 2018). It is used in high-dimensional reduction and clustering. In clustering, instances with the same underlying patterns are grouped together to form multiple categories (Cao, Qian, Wu, & Wong, 2019). The most common clustering method for unsupervised learning is K-Means clustering.

In highly complex datasets, there may be a need to reduce on the dimensions of the dataset and this too can be achieved using the dimensional reduction methods of unsupervised learning like self-organizing map (SOM).

2.3.3 Semi-supervised Learning

This type of learning combines the features of supervised and unsupervised learning. SSL extracts the relational pattern between input variables and the target/dependent variable with both labelled input instances and unlabeled instances. Using SSL a regression or classification model, depending on whether the target variable is continuous or categorical, is built on both labelled and unlabeled instances of the input dataset

(Kaneko, 2018). This type of learning has proved very useful in scenarios where the labelled data is considerably small compared to unlabeled data (Pei, Wang, Lin, & Zhong, 2018; Zhang, et al., 2018). SSL focuses on applying the inferred knowledge from unlabeled training instances with the extracted knowledge from the labelled training instances for enhanced prediction results (Tanha, 2018).

2.3.4 Reinforcement Learning

This type of learning involves interaction of the learning agent with the environment. Reinforcement learning has some kind of supervision. However, unlike supervised, semi-supervised and unsupervised leaning where the algorithm is provided with training data, in reinforcement learning the learning algorithm interacts with and takes responsive actions to, a dynamic environment. After it has made a prediction for an input instance, the desired response is supplied with either a reward or a punishment (Zhang, et al., 2018). There are two major components in this type of learning i.e. learning agent and environment.

The learning agent takes an action in response to the state of the environment and gets a reward for every action taken. The major goal of the agent is to realize the maximum possible reward while reducing the size of the punishment as much as possible in a changing environment (Lou & Tsai, 2008). This type of learning is used in areas which require sequential decision making especially in cases that are highly uncertain.

Therefore, it can be said that reinforcement learning lies between supervised and unsupervised learning since during training its training data is not labelled like the case of unsupervised learning but desired output for training instances are known like the case of supervised learning.

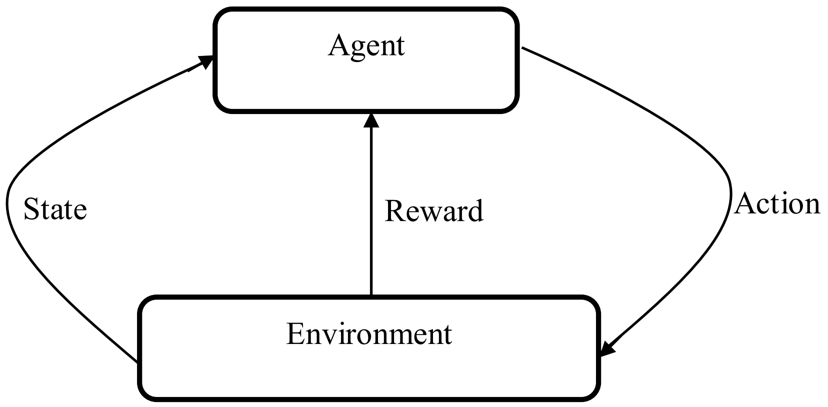

Figure 2. 3: Reinforcement learning model

2.4 Machine Learning Algorithms

This section gives a detailed overview of the machine learning algorithms used in this research. These are broadly divided into two categories i.e. single classifiers and ensemble classifiers. The single classifiers used in this research include Naïve Bayes (NB), decision tree (DT), Logistic Regression (LR), K-nearest neighbor (KNN) and Support Vector Machine (SVM). On the other hand, ensemble classifiers include Random Forest (RF), AdaBoost classifier, Extra Trees classifier (ET) and Gradient boosting (GBC) classifier. It should be noted that this research uses labelled datasets and therefore its scope is limited to supervised machine learning and classification algorithm.

2.4.1 Naïve Bayes Classifier

Naïve Bayes (NB) classifier is a Bayes' theorem based classifier. The Bayes' theorem is a result of Thomas Bayes' (172-1761) work which is based on the assumption that events are independently distributed. To classify an input instance into a given class of the output variable, for each probable class, the NB classifier determines the probability that the input instance belongs to that class. The instance is then classified into the class that gets the highest probability (Harzevili & Alizadeh, 2018; Maitra, Madan, Kandwal, & Mahajan, 2018).

NB classifier operates on the assumption that the probability distribution of every variable with a dataset is independent of the probability distributions of all other variables. This implies that the removal or addition of a particular attribute does not lead to any effect whatsoever on the other attributes in the dataset provided the attribute class information is given. For a training dataset containing n-attributes, a naïve Bayes model is built on 2n! assumptions of attribute independency. However, this assumption is not applicable in a large number of real-world scenarios. As a result, the performance of the classifier appreciably degrades due to biasness in the estimated probabilities (Harzevili & Alizadeh, 2018). According to (Mukherjee & Sharma, 2012), the inaccuracy of a naïve Bayes classifier is brought about by three factors. These factors are bias as a result of a large number of independent assumptions made on a high-dimensional dataset, noise due to presence of irrelevant and redundant variables and variance. Despite these factors however, NB classifier has always given significantly good classification results. This is because the assumption issues of the classifier have been handled by previous researchers through feature selection and reduction, feature weighting and local learning among others (Harzevili & Alizadeh, 2018). Various intrusion detection studies have deployed NB for classification.

In their study, (Varuna & Natesan, 2015) deployed K-means to create five distinct clusters with each cluster representing one class label of the KDD'99 dataset and used NB classifier for classification of the traffic instances. Their approach used only five out of the 41 features of the dataset to lower detection overhead. Their approach outperformed other classifiers used in the study in detection of R2L, probe attacks and U2R attacks.

In their study, (Han, Xu, Ren, & Gu, 2015) proposed an intrusion detection approach based on PCA for dimension reduction, which reduced the dimensionality of the dataset from 41-features to 15-features, and NB classifier for traffic classification on the KDD'99 dataset. Their proposed technique outperformed the traditional NB and neural networks in detecting all the five traffic categories in terms of detection accuracy.

Other studies where NB classifier has been put to use include (Li & Li, 2010) in which NB was used as a weak learner with AdaBoost classifier for intrusion detection on the KDDCup99 dataset, (Panda, Abraham, & Patra, 2010) where discriminative multinomial NB was utilized along with different filtering approaches so as to establish a network IDS which was evaluated using NSL-KDD dataset among others studies.

2.4.2 Support Vector Machine

Support Vector Machine (SVM) was proposed by (Vapnik, 1998) as a new statistical technique that utilizes the principle of structural risk minimization (SRM) and is among the powerful ML classifiers that can deal with both linear and non-linear cases. SVM is a supervised ML classifier and can perform both regression and classification (Chen, Hsu, & Shen, 2005).

SVM's core operations are based on a kernel that transforms data into dimensions that provide clear decision boundaries between instance classes (Sabar, Yi, & Song, 2018). The boundaries are in such a way that instances belonging to the same class are put together. The plane separating groups or class instances is known as a hyper-plane (Gao, Tian, & Xia, 2009). SVM aims at establishing an optimal hyper-plane. This is achieved by working out a restricted optimization equation expressed in quadratic form (Ahmad, Basheri, Iqbal, & Rahim, 2018). This problem is generated by utilizing SRM. The optimal hyper-plane establishes the highest possible distance between the closest data points as shown in Figure 2.8.

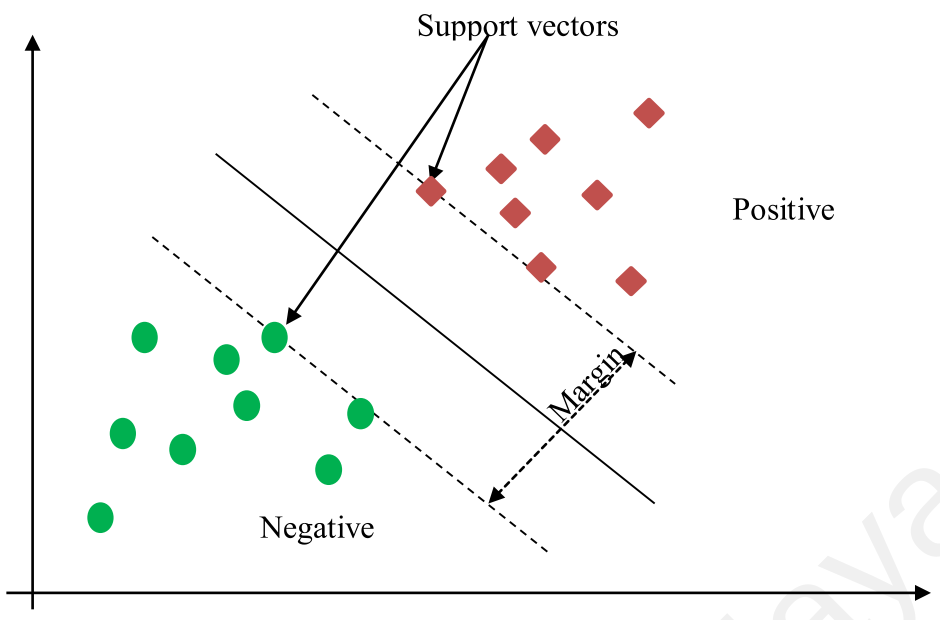

Figure 2. 4: SVM binary classification with a negative and a positive class

An outrageous number of network intrusion detection studies have utilized SVM for both binary and multiclass classification.

In (Jianhong, 2015) a SVM-based IDS algorithm that utilized the hybrid anti-colony technique was proposed. The KDDCup99 dataset was utilized to evaluate the proposed approach to evaluate and it was able to detect the four distinct anomaly traffic classes with good accuracy. In their study (Chang, Li, & Yang, 2017) proposed an intrusion detection approach based on SVM (for classification) and RF (for variable selection). Their approach was evaluated using 14 independent variables against the 41 independent variables of the KDD’99 dataset. In another study (Zhou, Yi, & Luo, 2013) presented a method for intrusion detection that combined density-based SCAN and GA for feature selection and incremental SVM classification. Evaluated using KDD’99 dataset, the algorithm indicated good detection results with respect to accuracy compared to other proposed SVM approaches by other researchers.

2.4.3 K-Nearest Neighbor (KNN)

KNN classifier was first introduced in the 1950s. However, due to its processing requirement never gained much attention until in the 1960s when processing power had been considerably enhanced (Han, Kamber, & Pei, 2012). It is one of the simplest and basic ML algorithms based on distance between data points in the vector space. It uses lazy learning for classification and is commonly referred to as a lazy classifier (Chellam, L, & S, 2018; Verma & Ranga, 2018). In addition to its ability of performing both binary and multiclass classification, KNN has the advantage of being simple and easy to implement and requires few parameters for its execution (Li, Zhang, Peng, & Yang, 2018). The KNN algorithm operates on the idea that instances belonging to the same class or instances with similar feature patterns are distributed close to one another in the data space.

To predict the class of a new unlabeled instance q, k labelled instances that are nearest to q are selected. The class to which the majority of the k-selected instances belong is predicted as the class to which q belongs (Aburomman & Reaz, 2016; Verma & Ranga, 2018). Although the distance between instances can be determined using various methods in order to identify and select the closest neighbors to the input instance, Euclidean distance is the standard measure in the KNN classifier (Li & Guo, 2007). The Euclidean distance, d(a, b) between two data points a and b is given as;

$$d(a,b) = \sqrt{\sum_{j=1}^{m} \left(a_j - b_j\right)^2}$$

Where;

a_j represents the jth attribute of instance a

b_j represents the j^{th} attribute of instance b

m represents the number of attributes.

The major drawback of the KNN classifier is that it does not perform well with unbalanced data (Li, Zhang, Peng, & Yang, 2018).

Since its invention KNN has been used in a number of classification studies including anomaly-based IDSs. In their study (Aung & Min, 2018) built a model that utilized k-means approach for grouping related instances and KNN classifier for classification of new instances with the aim of lowering detection time complexity while realizing high detection accuracy.

In their study (Malhotra, Bali, & Paliwal, 2017) built an IDS model based on genetic programming and KNN classifier. Their main objective was to apply genetic programming technique to KNN in order to enhance its intrusion detection rate. The model was evaluated using KDD'99 dataset and achieved up to 99.6% detection accuracy.

In (Wang, Zhang, & Zheng, 2016) an algorithm based on KNN combined with clustering and density feature was proposed for anomaly detection. The proposed method was assessed on KDDCup99 and NSL-KDD datasets and recorded improved performance.

2.4.4 Logistic Regression (LR)

Basing on Cox's (1958) work, Duncan and Walker (1967) introduced the idea of logistic regression. LR establishes a function or model that best fits the pattern between target variable and the attributes. However, unlike linear regression which is used for only regression, logistic regression can perform both regression and classification. Logistic regression is one of the probability based learning algorithms that can perform both binary and multiclass predictions (Ghosh & Mitra, 2015; Subba, Biswas, & Karmakar, 2015). To transform the continuous result from the regression phase, the result, y, of regression is passed to a sigmoid function which gives a discrete value. The discrete value is then used to predict the class labels of the input instances.

Logistic regression's strength is in its light-weightiness, easy interpretability and easy to implement and takes up low systems resources. These features make LR fit for deployment in environments that involve analysis of huge amount data in real time processing like intrusion detection (Bapat, et al., 2018).

In their study, in an effort to establish a basis for anomaly-based IDS and lower human intervention in the detection of botnets, (Bapat, et al., 2018) used LR for both important feature identification and traffic instance classification. Using LR model, a detection accuracy of 95% and TPR of 96.7% were recorded.

In (Subba, Biswas, & Karmakar, 2015) a LR model was built for intrusion detection. The authors in this study used LDA for reducing the dimensionality of the dataset and LR was used for detecting whether a particular traffic instance was benign or an anomaly. The proposed model was assessed on NSL-KDD dataset and recorded better detection performance as compared to SVM, C4.5 and NB classifier.

2.4.5 Decision Tree Classifier

The decision tree (DT) classifier was first put across by Quinlan (1986) and is defined as a directed tree made up of a series of nodes in which one or more nodes are created at each node in a hierarchical approach until an end criteria is satisfied/met. At the end of each branch is a leaf that represents a particular category of the dependent variable. The leaf can be chosen basing on the values of the input instance at each internal node of the DT (Quinlan, 1986; Wang, Li, Yu, & Liu, 2018). In addition to their ability of handling categorical and continuous inputs, their simplicity in terms of readability and interpretability for both computers and humans has made them very popular in the area of AI. DTs are mainly used in supervised ML environments for construction of prediction models (Guggari, Kadappa, & Umadevi, 2018; Kuzey, Karaman, & Akman, 2019).

DT classifiers have the ability of solving highly sophisticated problems and can provide a simple graphical visualization of the entire problem solving process for both humans and computer systems (Trabelsi, Elouedi, & Lefevre, 2018). Conventional DTs are based on a hierarchical top-down approach that utilizes the famous divide-and-conquer approach where the dataset is divided into subsets at each internal node until a stopping condition is reached (Wang, Yang, & Ren, 2009). At each internal node an attribute that gives more information on how to classify the training data instances into the appropriate class is selected basing on some attribute evaluation method which may be either Gini index, information gain or gain ratio. Since its invention DT classifier has been deployed in a number of IDS models owing to its simplicity, interpretability and classification performance.

In (Li, 2017) a DT algorithm called CART was applied together with PCA to build an IDS model. PCA was deployed for dimension reduction while CART performed the traffic classification task. The model was implemented using python and testing and evaluation was done using the famous KDD'99 dataset. In their study, (Jabbar & Samreen, 2016) deployed DT based algorithm known as alternate decision tree. Their model produced good accuracy for detection of the four distinct anomaly traffic classes of NSL-KDD dataset. In another study, (Sahu & Mehtre, 2015) used J48, a DT classifier that uses C4.5 algorithm, to build an intrusion detection model. Their aim was to implement a NIDS model based on a newer dataset other than the famous KDD'99 dataset. The model was tested and assessed using the KYOTO2006 dataset and a detection accuracy of up to 97.2% was achieved. Other studies where DT has been deployed for intrusion detection include (Katkar & Bhatia, 2013) where an IDS model was built using REPTree classifier for detection of DoS attacks and in (Rathore, et al., 2016) where DT based classifiers (J48 and REPTree classifiers) outperformed other classifiers used in the study with a detection accuracy of up to 99.9% on the KDD dataset.

2.4.6 Random Forest

In many cases single classifiers have not yielded reliable predictions. For this reason, many classification applications are based on a collection of base-classifiers to form what is known as an ensemble classifier. Random Forest is one of the most efficient and reliable ensemble classifiers based on a large number of decision trees. The final class of any input instance is determined based on majority vote where every tree casts one vote (Breiman, 2001). To train a random forest classifier, multiple unpruned trees are generated with each particular tree being trained independently on a bootstrap sample drawn from the input training dataset.

RF uses bagging where for each unpruned tree, a random sample is drawn from the original dataset with replacement. When a bootstrap sample is drawn, the remaining instances, normally about 30% of the whole training set, are known as out-of-bag (OOB) records and are used for evaluating the resulting tree (Zhang & Zulkernine, 2006). In addition, instead of using all features in the bootstrapped sample, each tree uses a random subset of features for the split at each internal node. This therefore means that RF has two phases of randomization for every tree, that is, during bootstrapping and during splitting at tree nodes (Malik, Shahzad, & Khan, 2011; Chang, Li, & Yang, 2017). These two randomization phases help to remove any correlation between trees in the forest which makes the resulting random forest classifier more stable by reducing variance in classification of new instances. It is worth noting that the classification accuracy of RF varies inversely with the correlation between the individual trees in the forest. This is because the classification error grows directly with correlation. The accuracy and correlation of trees in a forest depend on the number of features used for its growth (Tesfahun & Bhaskari, 2013).

If n bootstrap samples are drawn, then n independent decision tree classifiers are built; each corresponding to one bootstrapped sample. The resulting n-trees form the final RF classifier. During classification each individual tree makes its independent classification and the class with the highest number of predictions (votes) is considered as the final class for the input instance (Malik & Khan, 2013).

All unpruned trees only stop growing when the maximum depth is achieved. Pruning is sometimes used to stop the growth of a DT. This can be achieved by defining a stopping condition that can be based on the resources or performance or by explicitly defining a maximum tree depth. Pruning helps to control tree complexity and reduces the chances of over-fitting.

In addition, RF naturally performs some feature selection by examining the relative importance of each attribute which is a vital step for dimensionality reduction (Prashanth, Prashanth, Jayashree, & Srinivasan, 2008). To obtain the relative importance of a particular attribute, RF removes the input variable while keeping others and computes either the average increase in the classification error or the average decrease in the classification accuracy. It then assigns a rank value to each feature indicating how important it is in classifying input instances. A large number of intrusion detection studies have deployed RF in the recent past and it has shown better detection performance than many existing machine learning classifiers (Janarthanan & Zargari, 2017; Belouch et al, 2018).

In their study (Park, Song, & Cheong, 2018) deployed RF classifier on the Kyoto2006 dataset to detect the various attack types. Their aim was to study and analyze the extent to which the different attack classes can be detected using machine learning algorithms. In (Choi, Ko, Hwang, & Choi, 2018) a RF-based algorithm was proposed to solve the issue of explicitly specifying the number of trees when developing an ordinary RF model for IDSs. The NSL-KDD dataset was utilized to evaluate the proposed algorithm and better results were realized than the ordinary RF classifier. In another study, (Tesfahun & Bhaskari, 2013) built an IDS model based on RF for enhanced detection of minority attack classes in the NSL-KDD dataset. Their study utilized SMOTE to overcome the imbalances between the attack classes. Other studies where RF has been used for intrusion detection include (Janarthanan & Zargari, 2017) where it was used in evaluating important attributes of the UNSW-NB15 dataset, (Zhang & Zulkernine, 2006) where it was used to propose an IDS framework to combine the benefits of both signature-based and anomaly-based detection, among others.

2.4.7 AdaBoost Classifier

Adaptive boosting classifier commonly referred to AdaBoost is one of the popular and robust ensemble ML algorithms based on a weak learner. It employs a strategy which selects one weak learning algorithm e.g. Bayesian Net, NB, SVM or DT to use as the base learner. The weak learning algorithm is then executed iteratively in order to produce better classification results (Natesan & Rajesh, 2012). In each call the training dataset is slightly changed and utilized for a new model. Finally, weak models are integrated to form one model that is more robust with enhanced performance. The accuracy of the resulting model improves significantly owing to the fact that after each iteration, the algorithm tries to learn the data again and learns instances which were wrongly classified by the previous model. This way it tries to place these instances in their right classes in the subsequent models, consequently producing uplifted classification results (Li & Li, 2010). The classifier iterates until generation of a new model does not cause any further improvement in classification performance. This research used the default sklearn's SAMME.R algorithm for this classifier. Stagewise Adaptive Modelling using a Multi-class Exponential loss function (SAMME) is a weak learner which can perform both binary and multiclass classification (Jin, Hou, & Liu, 2010).

In their study (Yadahalli & Nighot, 2017) used AdaBoost classifier to build two IDS models for detection of online malicious activities in sensor networks. In one model the authors utilized decision stumps as weak learners and in the second GMM was used. In another study, (Hu, Hu, & Maybank, 2008) proposed an algorithm which used AdaBoost as the base classifier and decision stumps as AdaBoost's base learners. Evaluated using KDD dataset, the algorithm reported improved performance. To improve the detection of rare attacks like U2R and R2L, (Natesan & Rajesh, 2012) combined AdaBoost with NB classifier to propose an intrusion detection model. The model recorded improved detection of rare attacks when evaluated using KDD'99 dataset.

2.4.8 Extra Trees Classifier

Extra Trees (ET) is an ensemble classier which deploys the top-down strategy to build unpruned randomized DTs as the base classifier (Geurts, Ernst, & Wehenkel, 2006). It is a form of RF that was proposed to introduce an extra layer of randomness. In other words, it also uses DT as a base classifier. However, unlike in RF where a DT searches for an optimal split at each node, ET utilizes a random split threshold in DT training process. Like RF, the DT models are built on dissimilar random samples of the train dataset. This classifier deploys the Gini index where the highest number of variables is considered while selecting the optimum split node for the DT (Arjunan & Modi, 2017). The random DT is only built when the Gini index is 1 which reduces model variance and consequently avoiding the issue of over-fitting. However, not many researchers have used ET classifier for intrusion detection, despite the fact that it can produce good classification accuracy. In their effort to build a WLAN intrusion detection system, (Alotaibi & Elleithy, 2016) examined the detection performance of Bagging, RF and ET classifier against their proposed voting technique in traffic classification using the AWID dataset. In addition to traffic classification, ET was used to select 20 features that were deemed pertinent for traffic classification. Evaluated on the reduced feature set, ET and proposed method outperformed bagging and random forest with a detection accuracy of up to 96.31% and 96.32% respectively. In another study (Arjunan & Modi, 2017) proposed an IDS framework based on misuse and signature based detection. The researchers combined various ensemble classifiers including ET classifier in a hierarchical approach with the aim of enhancing the efficiency of detection of VLAN layer intrusions in the cloud without violating any of the cloud requirements for IDSs.

2.4.9 Gradient Boosting Classifier

The Gradient Boosting classifier (GBC), often referred to as a gradient boosted DT classifier, is an ensemble designed to build a model on a collection DTs as base classifiers. It draws a generalized boosting approach to random differentiable loss functions (Kulariya, Saraf, Ranjan, & Gupta, 2016). Like RF, GBC makes predictions based on decisions trees. However, the two deploy different modes of operation. For RF, each decision tree's prediction is independent of other trees' predictions in the forest. On the contrary however, with GBC, DT training is iterative, and a particular tree utilizes the prediction made by the previous trees to make its prediction. This way GBC classifier produces high prediction performance and is highly robust against outliers. The major drawback of this classifier is the long training time due to their sequential mode of operation.

2.5 Feature selection

Feature selection among the key phases in the process of building anomaly-based IDSs. Today's network traffic constitutes a large number of features. Some of the features are relevant and adding them improves the performance of the classifier (Li, Guo, Wu, & Li, 2018; Guo, et al., 2010). However, many of the traffic attributes are noisy and irrelevant for classification purposes. Using irrelevant features may either have no effect on the classifier accuracy or degrades the accuracy of the IDS. However, having only features that are pertinent for classification of a given input instance does not only improve detection performance with respect to accuracy and other performance metrics but also reduces the chances of over-fitting, reduces model complexity while improving its interpretability (Gul & Adali, 2017).

Feature selection is a phase aims at identifying an optimal set of pertinent features of size "p" out of the "q" independent attributes of a dataset such that p<q and that the subset of size "p" gives better prediction performance of the target variable than the entire attribute set of size "q" (Sheen & Rajesh, 2008). Feature selection involves examining every independent attribute in the feature vector to know its degree of association with the target i.e. how it affects the class variable. The main target of feature selection is to reduce the dimensionality of a dataset by identifying and removing noisy, irrelevant and redundant features from the attribute space. This consequently leads to a subset of attributes that better represent the patterns of the different classes in the target variable (Ambusaidi, He, Nanda, & Tan, 2016). The association between a given variable and the target variable can be used as a basis to either drop or retain a variable during feature selection.

Despite being a vital phase for building ANIDSs, feature selection is a complex task as it deals with a large number of features some of which are dependent on each other (Vijayanand, Devaraj, & Kannapiran, 2018).

The feature selection process is made up of four steps as illustrated in Figure 2.12 below;

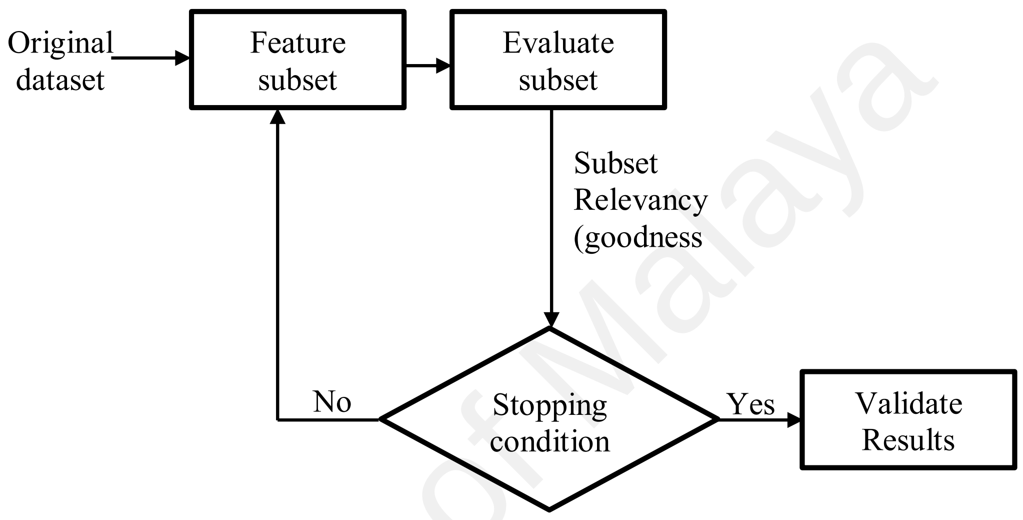

Figure 2. 5: Attribute selection process

Feature selection approaches are majorly divided into three categories which include; embedded, wrapper and embedded methods as described in the subsequent subsections.

2.5.1 Filter methods

In this method the predictive power of each feature is independently evaluated by applying some statistical or mathematical calculation e.g. distance, consistency, standard deviation, information measure among others. This selection method examines and provides the degree of relevancy of an attribute in predicting the target class without relying on any classification algorithm (Li, Guo, Wu, & Li, 2018; B & K, 2019). The most common statistical measures in filter method include correlation between the predictor variable and the target class, chi-square test and mutual information. Every predictor is assigned a score or rank value that indicates its predictive power. Basing on the rank or score, a variable may be considered important or redundant and thus retained or dropped respectively.

Some threshold or condition is set as an exclusion/inclusion criterion. For instance, inclusion rule can be that any feature with score/rank above some threshold value should be retained (Anwer, Farouk, & Abdel-Hamid, 2018). The selection process starts with an empty subset of features. Then features that can provide considerable information about the categories in the target variable are added successively until a threshold (or stopping condition) is reached. The filter method is cheaper in terms of system resource requirements (e.g. memory, processor and time) than wrapper and embedded methods since no classification model is built during the selection process.

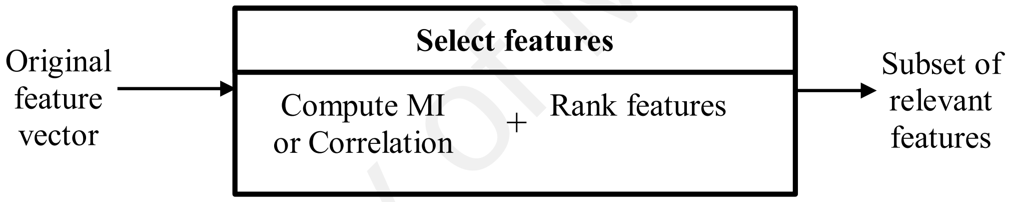

Figure 2. 6: Filter method process

The three common statistical approaches used to compute the relevancy of features in the filter method are;

Mutual Information

Mutual information (MI) commonly known as Info-Gain is a stochastic measure that provides information about the various features in relation to the target class. A feature that contains useful information in relation to the classes of the target variable is given a high score or rank while the one with low or no information is given low or zero score (Ambusaidi, et al., 2014; B & K, 2019). Mutual information is correlated with the entropy of attributes. The entropy is the measure of the extent of randomness of a given attribute in relation to the target class. Given an independent variable A{a_1, a_2, …, a_k} and a target variable Q{$q_1, q_2, \ldots, q_k$} where k is the number of instances, the mutual information of A in relation to Q can be obtained as (Aljawarneh, Aldwairi, & Yassein, 2018);

$$I(A,Q) = \sum_{q \in Q} \sum_{a \in A} p(A = a, Q = q) \log \frac{p(A = a, Q = q)}{p(A = a) p(Q = q)};\ for\ discrete\ variables$$

And

$$I(A,Q) = E(A) + E(Q) - E(A,Q);\ for\ continuous\ variables$$

Where; $p(A = a, Q = q)$, $p(A = a)$ and $p(Q = q)$ are the joint probability distribution and marginal probabilities of A and Q respectively while $E(A), E(Q)$ and $E(A,Q)$ are the entropies of A and Q and the joint entropy of A and Q respectively.

The entropy of a variable A with marginal probability $p(a)$ is defined as;

$$E(A) = -\sum_{a \in A} p(a) \log p(a)$$

And the joint entropy of a predictor variable A and the target variable Q is defined as;

$$E(A,Q) = -\sum_{a \in A} \sum_{q \in Q} p(a,q) \log p(a,q)$$

The role of MI is to show how much information is connecting the dependent variable to the independent variable. If A and Q are independent variables, then MI is zero since they share no information. MI measures the degree to which knowing a variable A removes or lowers the uncertainty of variable Q. In other words, MI gives the degree of how much knowing the value of an independent variable A tells about the dependent variable Q.

Correlation-based Feature selection (CFS)

Filter methods that utilize correlation between the predictor and the target variables for feature selection are based on the core principle that variables which are pertinent in the prediction of the target variable meet some significant level of correlation to the target variable (Kushwaha, Buckchash, & Raman, 2017; Mohammadi, Mirvaziri, Ghazizadeh-Ahsaee, & Karimipour, 2019). This implies that features with a high correlation value are assigned a high weight or rank and vice-versa. Pearson's correlation is the most commonly used strategy. Each feature is ranked depending on how much it associates with the target/dependent variable. It should be noted that Pearson's correlation works with numeric data type and nominal values must be encoded as numeric.

The Pearson correlation, r between a predictor variable A and target/dependent variable Q can be obtained from;

$$r(A,Q) = \frac{\sum_{i=1}^{n}(A_i - \bar{A})(Q_i - \bar{Q})}{\sqrt{\sum_{i=1}^{n}(A_i - \bar{A})^2}\sqrt{\sum_{i=1}^{n}(Q_i - \bar{Q})^2}}$$

Chi- Squared based selection

The chi-square (χ^2) is a statistical value usually utilized as a yardstick in establishing the statistical connection between attributes. The χ^2 test is a very fundamental test in cases involving categorical variables. For the case of feature selection, features with a high χ^2 value are ranked high and consequently considered to be of high relevancy in giving information about the variable (Onpans, Rasmequan, Jantarakongkul, Chinnasarn, & Rodtook, 2013; Mohammadi, Mirvaziri, Ghazizadeh-Ahsaee, & Karimipour, 2019). This way noisy and redundant features, i.e. features with very low χ^2 values, can be neglected.

The χ^2 value can be obtained from;

$$\chi^2 = \sum_{all\ i}\sum_{all\ j}\frac{(O_{ij} - A_{ij})^2}{A_{ij}}$$

Where O_{ij} represents the observed value

And A_{ij} represents the actual value

In their study, (Belouch, Hadaj, & Idhammad, 2017) applied the filter method (information gain) to filter out irrelevant features in the UNSW-NB15 dataset. In their study, (Gul & Adali, 2017) proposed an algorithm for selecting important intrusion detection attributes based on filter feature selection method and evaluated their approach using NSL-KDD dataset for a triad of intrusion classes. In another study, (Ullah & Mahmoud, 2017) proposed a feature selection model based on filter method. Their approach was evaluated using ISCX and NSL-KDD datasets.

2.5.2 Wrapper methods

Different from filter method, wrapper methods select features by evaluating all possible subsets of features from the original feature vector and evaluating them against a particular classification algorithm (Mohammadi, Mirvaziri, Ghazizadeh-Ahsaee, & Karimipour, 2019). The subset that produces the best classification results is selected for the model. For a dataset with a feature vector containing "m" features, $2^m - 1$ subsets are generated. Since a model is built for evaluating each of the subsets, wrapper methods are very expensive in terms of systems resource like C.P.U and memory. In addition, wrapper methods are highly inefficient in terms of computation time and their time complexity is in the order of NP-hard tasks. This makes wrapper methods inappropriate for deployment in cases involving extremely high dimensional datasets.

A wrapper method is composed of three fundamental components which include selecting a search algorithm, a classifier and an evaluator. The search algorithm decides which feature subset to evaluate. Many search algorithms have been proposed for this task including exhaustive, sequential, random search, generic search, and many more (Aljawarneh, Aldwairi, & Yassein, 2018). This is followed by instantiating a classifier and training and evaluating the selected subset using the subset evaluator method. The evaluator attaches a score to each of the subsets and the best scoring subset of features is returned by the selection method. This method produces considerably accurate results for the classifier deployed during the selection process. However, the features selected by this method do not in many cases produce good results for other classifiers. This implies that for every classification algorithm the entire selection process must be repeated (Anwer, Farouk, & Abdel-Hamid, 2018).

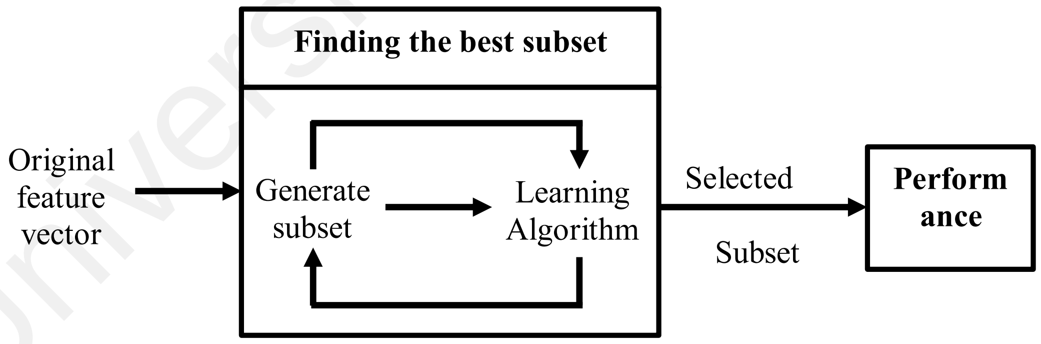

Figure 2. 7: Feature selection process by the wrapper method

In their work (Anwer, Farouk, & Abdel-Hamid, 2018) proposed a framework for feature selection using the UNSW-NB15 dataset. Their approach utilized filter and wrapper selection methods while J48 and NB classifiers were deployed for performance evaluation of the various feature combinations. The authors proposed 18 features, using

J48 algorithm and GR ranking, a filter-based technique, as the best UNSW-NB15 feature combination for intrusion detection with a detection accuracy of 88%

In (Moustafa & Slay, 2015) the features of UNSW-NB15 and KDD were examined to find out the important attributes for detecting of every distinct attack category in the two datasets using both filter and wrapper methods. The authors based their work on the degree of dependency/association of a particular attack category on a particular independent variable to determine the variable's relevancy in detection of the attack. The association rule mining approach was deployed in this research and the various features were evaluated using Naïve Bayes and EM clustering algorithms.

In their work, (Janarthanan & Zargari, 2017) proposed a subset of 5 features for intrusion detection and evaluated their proposed features against a subset of 8 most common features from (Moustafa & Slay, 2015) work using random forest classifier. Although the proposed subset in their work performs better than the extracted features from (Moustafa & Slay, 2015) work, it registers considerably low detection rates of up to 82.992% and 81.6175% with UNSW-NB15 training and test datasets respectively. This detection rate is lower than the detection rate recorded if the full dataset are used without feature selection.

In their work, (Belouch et al, 2018) compared the performance of machine learning algorithm using UNSW-NB15 dataset without dimension reduction. Their work was conducted in the apache spark environment and the results indicated random forest to record the best performance. However, using all 42 features of the dataset is not feasible since there many redundant features and this greatly degrades the performance of detection models not only in terms of prediction accuracy but also in detection time, system memory requirements among other system resources according to (Gul & Adali, 2017).

2.5.3 Embedded methods

This feature selection approach exhibits characteristics of filter and wrapper selection approaches. Unlike filter and wrapper methods, the embedded method combines the feature selection phase and the learning phase. In other words, the two are inseparable for this approach (Hamed, Dara, & Kremer, 2014). Because the learning and attribute selection are done in one go, embedded methods eliminate the two-step approach utilized by wrapper method making it quicker. However, because it combines feature selection with learning, it is slower than the filter method since the latter is independent of any learning algorithm.

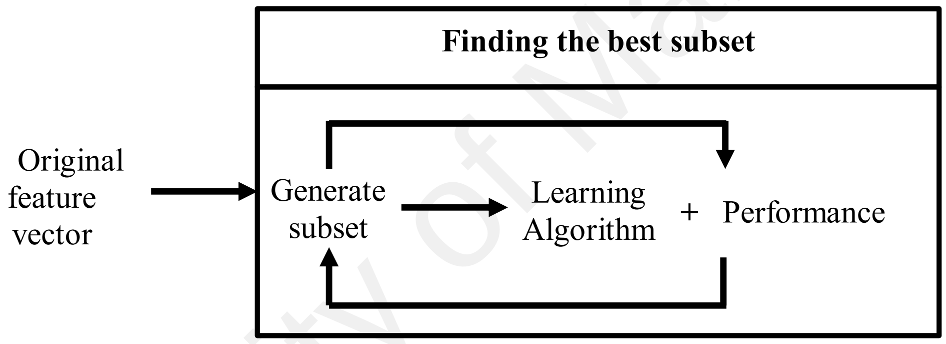

Figure 2. 8: Feature selection by embedded method

Table 2.3 gives a full comparison of the three selection approaches highlight the characteristics, merits and demerits of each individual method.

Table 2. 3: Comparison of filter, wrapper and embedded selection methods

Method	Characteristics	Advantages	Disadvantages
Filter Method	• Learning algorithm independent. • Uses a statistical approach to determine each feature's relevancy. • Assigns a rank or score to each feature and stops adding features to the subset when a stopping condition is reached.	• Faster than other methods. • Can work efficiently with datasets constituting a very high number of features • Cheaper in terms of computational resources like RAM, C.P.U etc.	• Often produces a large feature subset • Hard to determine the stopping condition or threshold • Does give optimum subsets for many learning algorithm
Wrapper method	• Based on a learning and search algorithm for subset generation	• Produces better selection results than filter method	• Very expensive in terms of computation resources and selection time is too long with an NP-Hard complexity.

Table 2.3 Continued

Method	Characteristics	Advantages	Disadvantages
Wrapper Method	• Evaluates the relevancy of all possible feature combinations using a particular classifier to determine a subset of relevant features		• Does not work well with high dimensional datasets. • The selection step has to be repeated for each learning algorithm
Embedded Method	• Combines some features of filter and wrapper method • Feature selection and the learning process are performed simultaneously i.e. the two operations are inseparable.	• Eliminates the two-step selection-learning process. • Relatively faster than wrapper method since learning and feature selection are performed at the same time	• Considerably slower than filter method • Expensive in terms of system resources compared to filter methods. • The resulting feature set is dependent of the classier used.

2.6 Evaluation metrics

There are various metrics deployed in machine learning and data mining applications to tell the goodness of an algorithm. The choice of the metrics depends on three main factors. First, the application field (e.g. weather prediction, intrusion detection, spam detection, price prediction among other). Second, the dataset used and type of task (i.e. classification or regression). This section gives an insight of some of the metrics used for evaluating the efficiency and detection performance of IDS.

2.6.1 Confusion Matrix

This shows the overall distribution of classification results. An incoming packet can be either predicted as an attack or as a normal traffic instance. Any classified packet in an intrusion detection task can take one of the four categories i.e. true positive, true negative, false positive or false negative

True positive (TP)

TP represents a malicious traffic instance that is rightly predicted as attack (i.e. positive predicted as positive).

True negative (TN)

TN represents a benign traffic instance that is rightly predicted as normal traffic (i.e. negative predicted as negative).

False positive (FP)

FP represents a normal traffic instance that is wrongly predicted as attack (i.e. negative predicted as positive).

False negative (FN)

FN represents a malicious traffic instance that is wrongly predicted as benign (i.e. positive predicted as negative).

Table 2. 4: Confusion matrix for an instance that is predicted as positive or negative

		Predicted	
Actual		Negative (Normal)	Positive (Anomaly)
	Negative(Normal)	TN	FP
	Positive (Anomaly)	FN	TP

It is worth noting that for any confusion matrix the main diagonal (grey-shaded cells) represents rightly classified instances. Below and above the main diagonal are false negatives and false positives respectively

2.6.2 Accuracy

The accuracy of a classifier in the field of intrusion detection is the fraction of rightly marked traffic instances to the total number of instances.

That is;

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{(TP + TN + FP + FN)}$$

In intrusion detection systems however, accuracy is not a sufficient metric for performance evaluation. This is because accuracy does not provide any information about the detective power of the system. In cases that involve significantly small attack instances with a large number of benign traffic instances, accuracy may be high even when a large proportion of attack packets have not been detected.

Since a single undetected attack instance can be highly disastrous and can result into huge losses to a business ranging from financial to loss customer trust, it is vital to consider other performance metrics for intrusion detection system in addition to accuracy.

2.6.3 Sensitivity

Sensitivity often referred to as true positive rate (TPR), recall, hit rate or detection rate gives the fraction of rightly detected anomalies out of the total anomaly instances. This metric is very important in intrusion detection applications as it shows the efficiency of the system in detection of attacks. In other words, it explicitly gives the probability of detecting possible attacks/intrusions to the network.

Sensitivity is defined by the equation below;

$$Sensitivity\ (or\ Recall) = \frac{TP}{TP + FN}$$

2.6.4 Precision

This is also known as the positive predictive rate (PPR). It gives the fraction of correctly detected attacks out of the total instances detected as break-in attempts.

Precision is defined as;

$$Precision = \frac{TP}{TP + FP}$$

2.6.5 False Positive rate

This is also known as fall-out rate and is defined by the equation;

$$FPR = \frac{FP}{TN + FP}$$

2.6.6 Specificity

Specificity often referred to as selectivity or true negative rate (TNR), gives the fraction of correctly classified benign traffic instances out of all benign traffic instances.

Specificity is defined by the equation;

$$Specificity\ (or\ TNR) = \frac{TN}{TN + FP}$$

2.6.7 False Negative Rate

This is sometimes referred to as miss rate and is given by the equation;

$$FNR = \frac{FN}{TP + FN}$$

2.6.8 F-measure

The F-measure is one of the statistical approaches that gives the level of balance between precision and recall in relation to accuracy. The F-measure is defined as;

$$F - measure = \frac{2(Recall * Precision)}{(Recall + Precision)}$$

2.6.9 Detection time

This is the time the classifier takes to classify a single traffic instance as either normal or an attack in milliseconds.

2.7 Conclusion

This chapter answers RQ1 by presenting a detailed formal assessment on the relevant literature for gaining an insight into the related work done in the area of intrusion detection, machine learning and feature selection methods. It presents a detailed account of the feature selection methods commonly used for attribute selection in machine learning models for anomaly-based intrusion detection highlighting the strengths and limitations of each method and giving a comprehensive comparison between them. Furthermore, a critical analysis of the existing studies which deployed the various machine learning and feature selection methods has been presented in this chapter highlighting the methodology, contribution and limitation of each approach. Finally, the various performance metrics commonly used in intrusion detection studies are explained.

CHAPTER 3: METHODOLOGY

3.1 Introduction

This chapter describes how the overall research was conducted by explaining the different phases and tasks that were performed. It reflects the sequential flow of the various activities starting by giving details of the different datasets and why the used dataset was considered as the best choice for this research. It then delves into the process of data pre-processing, feature selection all the way up to evaluation of the identified optimized feature set for anomaly detection.

3.2 Dataset

A good dataset for training an anomaly-based IDS must have two vital features i.e. must comprehensively represent the various types of potential attacks and must reflect a significantly large number of all possible traffic instances (Moustafa & Slay, 2015b). Therefore, this research considered three of the famous intrusion datasets which include KDDCup99, NSL-KDD and UNSW-NB15 datasets. In order to choose the best representative dataset for this research, three labeled benchmark datasets were compared as described in the subsequent sections.

3.2.1 KDD Cup 99 dataset

This dataset is the most famous and has been utilized for almost two decades by an outrageously large number of researchers in the field of intrusion detection. It was prepared as a standard anomaly detection dataset for training and testing IDSs (Stolfo, Fan, Lee, Prodromidis, & Chan, 2000). This dataset is an upgraded version of the DARPA'98 dataset that was generated at the Lincoln Laboratories at MIT University for IDS assessment projects. The DARPA'98 is a simulated dataset constituting tcp_dump files in form of compressed raw binary records collected for 9weeks.

To obtain the KDDCup99 dataset, the tcpdump files were processed into 7weeks of nearly 4GB records to form a training dataset of nearly five million TCP/IP connection instances with each instance comprising of 41 attributes. Each of these connection instances was marked as either normal or abnormal traffic with the help of Bro-IDS simulation tool. The remaining 2weeks' files were processed producing close to two million records of connection instances to form the test dataset. The training and test dataset comprise of 22 and 15 types of attacks respectively and for each abnormal record the attack type is precisely stated.

The attack traffic in the KDDCup99 dataset is broadly classified into four categories;

- **R2L (Remote to local) attacks**: In this type of attack, an external attacker attempts to access system/network resources by exploiting a vulnerability on an internal network machine/device or a weak user's authentication credentials. This way the attacker accesses all resources that are accessible to the exploited/compromised system by masquerading as a legitimate user.
- **DoS (Denial of service) attack**: In this type of attack, the attacker prevents authentic users to access system or network resources. This is reached at constantly transmitting a huge volume of traffic/requests to the system that consumes all/most of the computing resources like memory. These packets constantly keep the system busy making it unable to process requests from authentic users.
- **Probe attacks**: This type of attack aims at collecting as much information as possible about the network. Here the attacker scans the network in order to discover exploitable security vulnerabilities.

- **U2R (User to root) Attacks**: In this type of attack, the attacker gains access to an internal user's system and uses the compromised system to gain access to the system super user.

Due to the large number of instances in both the training and test datasets, KDDCup99 dataset can be considered a good choice for training and evaluating IDSs. However, many studies indicate that IDSs trained on KDDCup99 dataset have not produced reliable results because of two major inherent limitations;

- Redundant records in the training and test datasets. In their statistical analysis of the KDDCup99 dataset, (Tavallaee, Bagheri, Lu, & Ghorbani, 2009) revealed that duplicate entries constituted 0.78 and ¾ of the training and test datasets respectively. This causes biasness in the learning process as the records that are repeated more often tend to bar the learning algorithm from learning about infrequent records like R2L and U2R which can be very disastrous if undetected. In addition, the repeated instances in the test often translate into biased evaluation values for algorithms with good detection rates for highly frequent instances.

- Tavallaee, Bagheri, Lu, & Ghorbani, (2009) also indicated that comparing the performance of IDSs trained on the KDDCup99 dataset is difficult owing to the fact that a large number of attack instances in the training set are duplicated in test set making even a simple learning algorithm to give detection accuracy of at least 86%.

- Another commonly raised deficiency of this dataset by researchers is the existence of a significantly high number of missing values and outliers.

In an attempt to cub the shortfalls of this dataset and remove the imbalances between the various attack categories, (Tavallaee, Bagheri, Lu, & Ghorbani, 2009) created the NSL-KDD dataset which is a reduced, better and upgraded version of the KDDCup99.

3.2.2 NSL-KDD dataset

The NSL-KDD dataset is an enhancement of the KDDCup99 dataset and was created offline by (Tavallaee, Bagheri, Lu, & Ghorbani, 2009) after a comprehensive statistical study and evaluation of the KDDCup99 dataset. Like the KDD'99 dataset, each instance in the NSL-KDD dataset is composed of 42 attributes with the 42nd attribute indicating the class of the connection instance i.e. benign or attack instance. Similarly, the attacks in this dataset can be classified into four categories as in the KDDCup99 dataset i.e. U2R, R2L, Probe and DoS attacks. The complete NSL-KDD dataset is composed of three components namely KDDTrain+, KDDTest+ and KDDTest-21. The KDDTrain+ was derived from training set of the KDD'99 dataset by eliminating duplicate instances and is made up of 125,973 traffic instances. To eliminate repeated entries without losing information about them, only one instance was retained for each group of duplicate instances. The KDDTest+ is a derivation of the test set component of the KDD'99 dataset. This data set is composed of 22,544 traffic records. The third component of the NSL-KDD dataset was created as a collection of all incorrectly classified instances by all the 21 learners used by Tavallaee, Bagheri, Lu, & Ghorbani (2009) and comprises 11,850 instances.

Like in the KDDCup99 dataset, this dataset has 22 kinds of distinct attacks in the training set while the test set has an additional 14 attacks that are not found in the training set. A detailed description of the 42 attributes of the KDD datasets (KDDCup99 and NSL-KDD) is given in Appendix A.

Despite the fact that the NSL-KDD seems to work out the two major issues of the KDDCup99 dataset, various research efforts in the field of intrusion detection have marked this dataset unfit for evaluation of anomaly-based IDSs for two major reasons; first, addition of new attacks in the test set that are not in the training causes the probability distributions of the two sets to differ. Secondly, the dataset is very old since it contains network records that were collected over twenty years ago. Due to the evolution of technology, many researchers have stressed that this dataset is not a perfect representation of the current attack environment since it does not carry any information about modern attacks.

Owing to the above issues of both datasets this research considered the datasets inappropriate since the main reason for adopting anomaly-based IDSs is their ability to detect the current state-of-the-art attacks that are created using advanced tools and techniques.

3.2.3 UNSW-NB15 dataset

UNSW-NB15 is a more recent intrusion dataset generated in 2015 to provide researchers in the field of intrusion detection a modern comprehensive dataset that eliminates the major issues of the famous KDDCup99 and NSL-KDD datasets like lack of traceable information about modern attack styles and types among others as described in the previous sections. This dataset was created by Nour Moustafa and Jill Slay at the Cyber range lab of the Australian Centre for Cyber Security (ACCS) with the help of a modern network traffic generating tool known as IXIA PerfectStorm tool (Moustafa & Slay, 2014; Moustafa & Slay, 2015a; Moustafa & Slay, 2015b; Moustafa & Slay, 2016).

According to (Moustafa & Slay, 2016), the reference datasets at the time had two main limitations that would degrade the correctness of detection results produced by IDSs evaluated using such datasets; first since the creation of these datasets, many hard to detect attacks have been engineered. These attacks use attack styles that make them hard to trace and old datasets do not provide any information about these attacks. Secondly, the difference in distributions between the training and test datasets leads to detection results being skewed towards certain attack categories while producing poor detection for others and often leading to high false detection rates.

Using the IXIA tool, a series of realistic and modern traffic composing of normal, together with the co-existing current attack traffic instances were captured making up to 100Gigabytes of raw data stored as Pcap files created with the tcpdump tool in which each Pcap file contains up to 1GB of data. Feature extraction and writing to the Pcap files were achieved by deploying Bro-IDS (a tool for monitoring network activities) and Argus tool (for handling and analyzing raw packets transmitted with the network) respectively.

Each of the attack instances can be placed in one of nine attack classes as described below;

- **Analysis attacks:** This encompasses the various attacks that penetrate the network via web-based applications. Examples of such attacks are spams which penetrate the network through emails, web-scripts which use HTML files, among others.
- **DoS attacks:** these attacks prevent authentic users from accessing system or network resources. This is achieved by constantly sending a huge number of traffic/requests to the system that consumes all/most of the computing resources like memory. These packets constantly keep the system busy making it unable to process requests from authentic users.

- **Fuzzers:** These are intended to identify exploitable security weaknesses or loopholes in an application, OS or network. They often attempt to crash the system through feeding it with a huge volume of data in a random manner.
- **Generic attacks:** these attacks utilize an approach that intends to cause collision of a block-cipher with the help of a harsh function.
- **Backdoor attacks:** These attacks are intended to bypass the established network authentication policies and standards in order to remotely access network assets/devices without being detected.
- **Exploits:** these are instructions that make use of a weakness on the system or network e.g. a bug or a security weakness caused an authentic user unintentionally.
- **Worms:** These attacks replicate themselves like viruses to other network devices or systems but only spread via computer networks.
- **Shell code:** These attacks are intended to obtain root privileges on the compromised system. The attacker uses instructions that attempt to access the shell in order to acquire full control of the victim system.
- **Reconnaissance:** these are similar to probe attacks i.e. they attempt to collect as much information as possible concerning the victim system. The attacker tries to inspect the activities carried out on the network in order to discover any exploitable vulnerability.

The feature vector of the UNSW-NB15 datasets is composed of 49 attributes which are grouped into six categories namely flow features, basic features, content features, time features, additional generated features and label features as described in Appendix B;

However, after a comprehensive analysis, (Moustafa & Slay, 2016) created a training set consisting of 175,341 distinct traffic instances and a test set comprising of 82,332 distinct instances with no record appearing in both datasets. Each traffic instance in these datasets is composed of 45 attributes with the first attribute representing the instance id and the last two attributes giving class information; that is, attribute 44 provides the precise traffic category while attribute 45 tells whether the traffic instance is normal traffic or is an anomaly. Therefore, only 42 attributes form the independent feature vector for traffic classification in the UNSW-NB15 dataset and were considered for feature selection in this research.

Table 3. 1: Final features of UNSW-NB15 dataset with their associated data types

Name	Data type	Name	Data type
Dur	Real	Dwin	Integer
proto	Nominal	Tcprtt	Real
service	Nominal	Synack	Real
State	Nominal	Ackdat	Real
spkts	Integer	Smean	Integer
dpkts	Integer	Dmean	Integer
sbytes	Integer	trans_depth	Integer
dbytes	Integer	response_body_len	Integer
Rate	Real	ct_srv_src	Integer
Sttl	Integer	ct_state_ttl	Integer

Table 3.1 Continued

Name	Data type	Name	Data type
Dttl	Integer	ct_dst_ltm	Integer
sload	Real	ct_src_dport_ltm	Integer
dload	Real	ct_dst_sport_ltm	Integer
Sloss	Integer	ct_dst_src_ltm	Integer
dloss	Integer	is_ftp_login	Binary
sinpkt	Real	ct_ftp_cmd	Integer
dinpkt	Real	ct_flw_http_mthd	Integer
Sjit	Real	ct_src_ltm	Integer
Djit	Real	ct_srv_dst	Integer
swin	Integer	is_sm_ips_ports	Binary
stcpb	Integer	attack_cat	Nominal
dtcpb	Integer	Label	Binary

A comprehensive comparison of the datasets that were initially considered as primary candidates for this study is summarized in Table 3.2.

Table 3. 2: Comparison between UNSW-NB15 and KDDCup99 datasets

UNSW-NB15 dataset	KDDCup99 dataset
Created by simulation using three networks with forty-five distinct IP addresses	Created by simulation using two networks with only eleven distinct IP addresses
Collected for two days; 16hrs on the first day and 15hrs on the second day	The training data is collected for 5weeks while the training data is collected for 2weeks
Has no repeated or redundant instances in both training and test datasets	Contains repeated and redundant traffic instances in both training and test datasets
Traffic instances are classified into ten attack categories e.g. Normal, DoS, Worms, Exploits, etc.	Traffic instances are classified into five categories e.g. Normal, DoS, probe, U2R and R2L attacks
Feature vector contains forty-nine features	Feature vector contains forty-two features
Argus and Bro-IDS tools used for instance labelling and feature extraction	Only Bro-IDS tool used
Data collected using only one format i.e. pcap file	Data collected using three formats i.e. tcpdump files, BSM files and dump files.

It should be noted that out of the 41 and 42 attributes of the final KDDCup99 and final UNSW-NB15 datasets respectively, there are only five common features i.e. duration, service, protocol type, number of source bytes and number of destination bytes. Therefore, one can hardly compare these two datasets on a feature-by-feature basis (Janarthanan & Zargari, 2017).

3.3 Data pre-processing

Data pre-processing involved identifying and removing outliers, dealing with missing values and transforming data into a format that the tools and algorithms used can understand or work with. Both the training and test datasets of the UNSW-NB15 dataset are presented as csv (comma separated value) files and were found to have no missing values.

3.3.1 Working with string variables

As reflected in Table 3.1, the dataset's feature vector is composed of attributes with different data types. Since some tools (e.g. SPSS 24.0 which was used for data analysis) and algorithms don't work well with string data type, all independent attributes were transformed to have real/numeric data types as follows;

All integer, floating-value and binary attributes were left intact. For a nominal attribute with *m* distinct values; if *m* <= 10, one-hot encoder was used to present the values numerically. On the other hand, if *m > 10,* the distinct values were assigned discrete integers starting from 1 for the first distinct value up to *m* for the *m*th distinct value.

One-hot encoder works with binary digits i.e. 0 for off/low and 1 for on/high (Mitra, Avra, & Mccluskey, 1997). It basically creates a column for each of the distinct values of a given attribute. For any given instance/record it sets the entry of the column corresponding to that value of the instance to 1 and the rest to 0. This approach is better than the traditional approach of representing the string values as counting integers i.e. 1 for the first value, 2 for the second, up to the last distinct nominal value since there is no possibility of algorithms interpreting the entry of one distinct value of an attribute as numerically higher than the other. However, this approach does not work well with nominal/categorical attributes with a high number of distinct values as it leads to high dataset dimensionality and consequently affecting the performance of the algorithm in terms of prediction/detection time and other system resources. This is the main reason why one-hot encoding was applied to nominal attributes having less than or equal to ten distinct values.

3.3.2 Creating training and test datasets

Considering the fact that the attack class of UNSW-NB15 dataset is fairly balanced between the normal and attack instances in addition to its being comprehensive and with a considerably large number of traffic instances, this research deployed the percentage split approach for creating training and test datasets appropriate. Here sklearn's random train_test_split module was used in Python 3.7 to create a training and test set of 70% and 30% of the full dataset respectively.

3.4 Environmental setup

Two open source tools, WEKA 3.8 and Python 3.7 were installed on Intel® CORE™ i7-4770 PC, 4GB RAM, CPU @ 3.40GHz running windows 10 for all experiments in this research. WEKA 3.8 is freely available and can be downloaded from *https://www.cs.waikato.ac.nz/ml/weka* while python 3.7 can be got free of charge from *https://www.python.org/downloads/release/python-370/*. The whole research started with nine machine learning classifiers which were evaluated using the entire UNSW_NB15 dataset in order to select the best five classifiers for further experiments. The nine classifiers constituted five single classifiers (i.e. DT, Naïve Bayes, KNN, LR and SVM) and four ensemble classifiers (i.e. RF, AdaBoost classifier, ET Classifier and GBC classifier).

3.5 Selecting the five classifiers

To identify the best five classifiers for this research, each one of nine classifiers was trained and evaluated using the entire UNSW-NB15 dataset before feature selection. Accuracy, precision, recall, FPR and detection time in milliseconds per traffic instance were utilized as a yardstick for performance evaluation as shown in Table 4.1.

3.6 Identifying the optimal feature set

Here the three major attribute selection approaches i.e. wrapper, embedded and filter, were applied. In each method, each of the selected attribute subset was evaluated on the five identified classifiers to determine its goodness.

3.6.1 Filter method

The three statistical approaches deployed by the filter selection method were applied for this work. These included correlation-based FS (Pearson correlation), chi-square approach and mutual information approach (InfoGain and Gain ratio). Since all these methods attach scores (showing the predictive power or relevancy of the attribute) to each of the independent variables in relation to the target variable, a threshold value has to be set as a stopping criterion for attribute selection. However, it is hard to determine this value appropriately and often these approaches do not result into optimal results (Anwer, Farouk, & Abdel-Hamid, 2018). To avoid elimination of relevant features, the cut off value was set to select the 25 top ranking attributes (which more than half of the total independent features) in each of the filter approaches used. For all approaches, the selected set of features was evaluated for all five classifiers and results compared with features selected by other approaches.

3.6.2 Wrapper methods

For the wrapper method, the recursive feature elimination (RFE) approach which iteratively creates subsets of features by utilizing the greedy search algorithm was used together with classifiers that have the ability of attaching feature importance, scores or coefficients to the independent features. The classifiers used include LR, RF, DT (tree based algorithms were including C4.5 algorithms, J48 algorithm and REPTree), gradient boosting classifier, extra trees classifier and AdaBoost classifier. In addition, the Boruta method (Kursa & Rudnicki, 2010), a wrapper method based on random forest, was used with the help of Boruta-py module in python. Again, each of the five algorithms was evaluated on each of the selected subset of attributes.

3.6.3 Embedded methods

Like for wrapper methods, algorithms with the ability to attach/assign feature importance scores/coefficients to independent features were deployed in this approach. The Algorithms used here include DT, RF, AdaBoost, Extra tree classifier, LR and gradient boosting classifier.

3.6.4 Identifying common features

The whole process of applying the three attribute selection methods resulted into 18 subsets of features. Some features were selected by majority of the selection approaches, some others were selected by very few while others were not selected by any selection approach at all. In this research, all features that appeared in at least eight subsets were considered for the common features set for final selection. Eight was considered as threshold value after conducting a series of experiments. The selected subset of frequent features was then evaluated for all five classifiers used in this research.

3.6.5 Final optimal feature set

The final optimal feature set for intrusion detection was obtained by examining every individual attribute in the common feature's subset obtained as described above. This involved iteratively removing features from the subset and evaluating the remaining subset against all the five algorithms. If removing a feature from the subset results into better detection performance or has no any effect on the detection rate, such feature was considered redundant and discarded from the final subset. Otherwise the feature was considered relevant and added to the final optimal feature set. A pseudo code for obtaining the final optimal feature set is given below.

Pseudo code for obtaining optimized feature set

```
1. final_feature_set ←empty arraylist
2. Data ←read dataset
3. Common_features ←list_of_frequent_features_from_various_selection_methods
4. accuracy1← build and evaluate model on Common_features and return accuracy
5. for feature_f in common_features do:
       Temp_fs ←common_features – feature_f
       accuracy2 ← build and evaluate model on Temp_fs and return new accuracy
       if accuracy1<accuracy2 do:
           accuracy1← accuracy2
       Else:
           add feature f to final_feature_set
        End if:
6. End for:
7. Return final_fs
```

The overall research design/process is shown in Figure 3.1

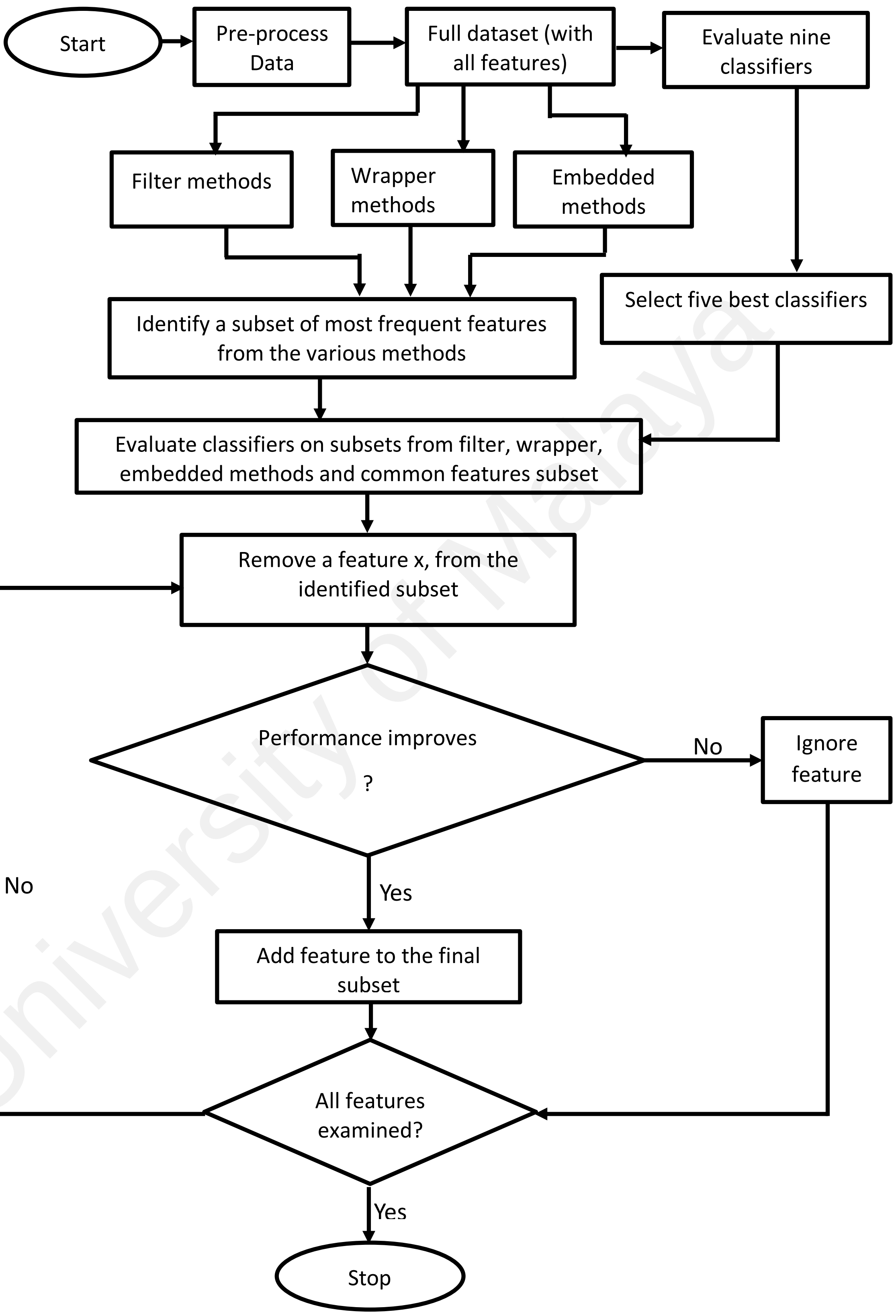
Start
Pre-process Data
Full dataset (with all features)
Evaluate nine classifiers
Filter methods
Wrapper methods
Embedded methods
Identify a subset of most frequent features from the various methods
Select five best classifiers
Evaluate classifiers on subsets from filter, wrapper, embedded methods and common features subset
Remove a feature x, from the identified subset
Performance improves ?
No
Ignore feature
Yes
Add feature to the final subset
All features examined?
No
Yes
Stop

Figure 3. 1: Flowchart of the research design

It is worth noting that classifiers that are incapable of attaching coefficients/scores to attributes like NB, SVM and KNN were neither used in the wrapper method nor embedded method.

To obtain the best k-value for KNN classifier, the classifier was executed iteratively for values of k ranging from 1 to 20 while recording the mean classification error for each k-value. The k-value with least classification error was considered for this research.

3.7 Conclusion

This chapter has presented the overall research process and design clearly describing the three famous malware benchmark datasets and highlighting the limitations of each. The chapter also presents a clear description of the environmental setup clearly stating the various tools deployed for data analysis and feature selection. In addition, the various approaches deployed for data preprocessing are described giving an account for the deployment of each. The chapter also details how the best five classifiers for the experiments were selected and how the various feature selection methods were used to arrive at the optimized feature set proposed in this study.

CHAPTER 4: RESULTS

4.1 Introduction

Results from the data analysis and all experiments conducted for instance experiment without features selection, to the selected features in all phases and the evaluation results from the different subsets of features are presented in this chapter.

4.2 Descriptive statistics

The dataset used for this research is composed of 82,332 distinct traffic instances with 37,000 (44.9%) normal instances and 45,332 (55.1%) attack instances. In addition, the dataset constitutes six variables that were treated as nominal variables during data analysis i.e. protocol_type (134 distinct values), service used (13 distinct values), state (11 distinct values), is_ftp_cmd (binary: 1 for yes and 0 otherwise), is_sm_ips_ports (binary: 1 for yes and 0 otherwise) and label (binary: 1 for attack and 0 normal traffic). Table 4.1 gives the statistical description for the rest of numeric variables precisely giving mean, standard deviation, minimum and maximum value for each variable. For purposes of consistence all floating values have been written to two decimal places.

Table 4. 1 Descriptive statistics of numeric variables

Variable	Mean	Standard Dev.	Minimum	Maximum
Dur	0.00	59.99	1.01	4.71
Spkts	1	10646	18.67	133.92
Dpkts	0	11018	17.55	115.57
Sbytes	24	14355774	7993.91	171642.26
Dbytes	0	14657531	13233.79	151471.46
Rate	0.00	1000000.00	82410.89	148620.37
Sttl	0	255	180.97	101.51
Dttl	0	253	95.71	116.67
Sload	0.00	52680002560.0	64549016.91	179861832.60
Dload	0.00	208211080.00	630546.96	2393000.56
Sloss	0	5319	4.75	64.65
Dloss	0	5507	6.31	55.71
Sinpkt	0.00	60009.99	755.39	6182.62
Dinpkt	0.00	57739.24	121.70	1292.31
Sjit	0.00	1483830.92	6363.08	56724.02
Djit	0.00	463199.24	535.18	3635.31
swin	0	255	133.46	127.36
stcpb	0	4294949667	1084641551.00	1390859762.00
dtcpb	0	4294880717	1073464671.00	1381996192.00
dwin	0	255	128.29	127.49
tcprtt	0.00	3.82	0.06	0.13
synack	0.00	3.23	0.03	0.08
ackdat	0.00	2.93	0.03	0.06
smean	24	1504	139.53	208.47
dmean	0	1500	116.28	244.60
trans_depth	0	131	0.09	0.54
response_body_len	0	5242880	1595.37	38066.97
ct_srv_src	1	63	9.55	11.09
ct_state_ttl	0	6	1.37	1.07
ct_dst_ltm	1	59	5.74	8.42
ct_src_dport_ltm	1	59	4.93	8.39
ct_dst_sport_ltm	1	38	3.66	5.92
ct_dst_src_ltm	1	63	7.46	11.42
ct_ftp_cmd	0	2	0.01	0.09
ct_flw_http_mthd	0	16	0.13	0.64
ct_src_ltm	1	60	6.47	8.54
ct_srv_dst	1	62	9.16	11.12

4.3 Without feature selection

The first experiment was executed on the entire dataset without any feature selection or elimination done. As indicated in chapter three UNSW-NB15 dataset's feature vector contains 41 independent variables and a dependent variable that marks the traffic instance as either normal or an anomaly.

Table 4. 2: Detection performance of classifiers before feature selection

Classifier	Accuracy (%)	Precision	Recall	FPR	Detection time ($\times 10^{-3}$ms)
NB	71.53	72.62	78.00	36.70	1.26
LR	75.34	85.96	66.12	12.99	0.63
DT	96.64	96.80	97.00	3.92	0.63
KNN	92.99	95.90	91.22	4.855	768.67
SVM	81.43	99.58	67.00	0.343	57027
RF	97.78	98.41	98.02	2.45	12.84
AdaBoost	95.49	96.27	98.10	4.22	11.87
Extra Trees	97.32	97.60	79.07	2.87	2.10
GBC	97.38	98.00	98.02	2.41	5.47

For KNN classifier the value of k that gives the best classification accuracy was determined by iteratively checking the error for values of k ranging from 1 to 20. k=5 gave the least classification error as shown in Figure 4.1.

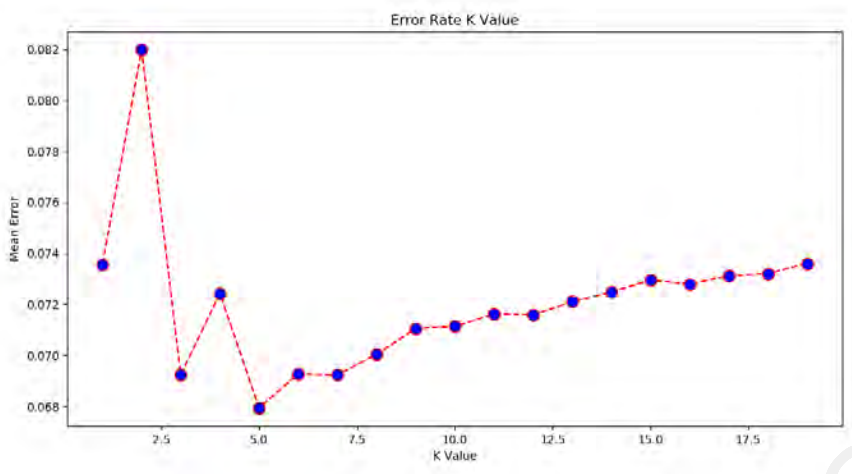

Figure 4. 1: Determining the value of k for KNN classifier

4.4 Feature selection

Finding an optimal feature set for anomaly-based IDS having been the major aim of this research, feature selection by application of different methods formed the core of this research. Table 4.2 shows the features that were considered pertinent for intrusion detection by the various selection methods. The features are sorted based on their frequency of selection starting from the most frequent to the least frequent. In addition, a tick indicates that the feature was selected by a particular selection technique. The selection techniques are grouped under filter, embedded and embedded methods. **Total** indicates how many times a particular attribute was selected i.e. frequency of selection of a given attribute in the attribute space.

Table 4. 3: Attributes selected by various selection methods

	FILTER			WRAPPER									EMBEDDED						
Features	**IG**	**Pearson**	**Chi-2**	**LR**	**RF**	**DT**	**GBC**	**Ada Boost**	**Extra Trees**	**J48**	**REP Tree**	**RF Boruta**	**LR**	**RF**	**GBC**	**Ada Boost**	**Extra Trees**	**DT**	**Total**
ct_srv_dst	√	√	√	√	√	√	√	√	√	√	-	√	√	√	√	√	√	√	16
ct_dst_src_ltm	-	√	√	√	√	√	√	√	√	-	√	√	√	√	√	√	√	√	16
ct_srv_src	√	√	√	-	√	√	√	√	√	-	√	√	√	√	√	√	√	√	15
sttl	-	√	√	-	√	√	√	√	√	√	√	√	-	√	√	√	√	√	15
tcprtt	-	√	√	√	√	√	√	√	√	-	-	√	√	√	√	√	-	√	14
ct_dst_sport_ltm	√	√	√	√	√	-	√	√	√	-	-	√	√	√	√	√	√	-	13
sbytes	√	-	-	-	√	√	√	√	√	√	√	√	-	√	√	√	√	√	13
synack	√	√	√	√	√	√	√	-	√	-	-	√	√	√	√	√	-	√	13
ct_state_ttl	√	√	√	√	√	-	√	√	√	-	-	√	√	√	√	√	√	-	13
dload	√	√	√	√	√	√	√	-	-	-	-	√	√	√	√	√	√	√	13
service	√	√	√	-	-	√	√	√	√	√	√	√	-	-	√	√	√	-	12
rate	√	√	√	-	√	√	√	√	√	-	-	√	-	√	√	-	√	√	12
dmean	√	√	√	√	√	-	√	√	√	-	-	√	√	√	√	-	-	-	11
smean	√	-	-	-	√	√	√	√	√	-	-	√	-	√	√	√	√	√	11
dbytes	√	-	-	-	√	√	√	√	-	√	√	√	-	√	√	√	-	√	11
dttl	√	√	√	√	-	-	√	-	√	-	-	√	√	√	√	-	√	-	10
ct_dst_ltm	-	√	√	-	-	√	-	√	-	√	√	√	√	-	-	√	-	√	10
sload	√	√	√	√	√	√	-	-	√	-	-	√	-	√	-	-	-	√	9
proto	√	-	-	√	-	√	√	√	-	-	-	√	√	-	√	√	-	√	9
ct_src_dport_ltm	√	√	√	√	-	√	-	-	√	-	√	√	-	-	-	-	√	√	9
sinpkt	√	√	√	√	√	√	-	-	-	-	-	√	√	-	-	-	-	√	8
state	-	√	√	√	√	-	-	-	√	-	-	√	√	-	-	-	√	-	8
swin	-	√	√	√	-	-	-	-	√	-	√	√	√	-	-	-	√	-	8
ct_src_ltm	-	√	√	-	-	√	-	√	-	√	-	√	-	-	-	√	√	-	8
dur	√	-	-	-	√	√	-	√	-	-	-	√	-	√	-	√	-	-	6

Table 4.3 Continued

	FILTER			WRAPPER									EMBEDDED						
Features	**IG**	**Pearson**	**Chi-2**	**LR**	**RF**	**DT**	**GBC**	**Ada Boost**	**Extra Trees**	**J48**	**REP Tree**	**RF Boruta**	**LR**	**RF**	**GBC**	**Ada Boost**	**Extra Trees**	**DT**	**Total**
response_body_len	-	-	-	-	-	-	√	√	-	-	√	√	-	-	√	√	-	-	6
dinpkt	√	-	-	-	-	-	√	-	-	√	-	√	-	√	√	-	-	-	5
dpkts	√	-	-	-	√	-	-	-	-	-	√	√	-	√	-	√	-	-	5
ct_flw_http_mthd	-	-	√	√	-	-	-	-	-	-	√	-	√	-	-	-	-	-	4
is_sm_ips_ports	-	√	√	√	-	-	-	-	-	-	-	-	√	-	-	-	-	-	4
spkts	-	-	-	√	-	-	-	-	-	√	-	√	√	-	-	-	-	-	4
dwin	-	√	√	-	-	-	-	-	√	-	-	-	-	-	-	-	√	-	4
sloss	-	-	-	√	-	-	√	-	-	-	-	√	-	-	√	-	-	-	4
dloss	-	-	-	-	-	-	-	√	-	-	√	√	-	-	-	√	-	-	4
dtcpb	-	√	√	-	-	√	-	-	-	-	-	-	-	-	-	-	-	-	3
ackdat	-	√	-	√	-	-	-	-	-	-	-	√	-	-	-	-	-	-	3
is_ftp_login	-	-	-	-	-	-	-	-	-	√	√	-	-	-	-	-	-	-	2
stcpb	-	√	√	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2
sjit	-	-	-	-	√	-	-	-	-	-	-	√	-	-	-	-	-	-	2
djit	-	-	-	-	-	-	-	-	-	-	-	√	√	-	-	-	-	-	2
ct_ftp_cmd	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0
trans_depth	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0

Our set of common features constituted all features that were selected by at least eight methods and these features are; *ct_srv_dst, ct_dst_src_ltm, ct_srv_src, sttl, tcprtt, ct_dst_sport_ltm, sbytes, synack, ct_state_ttl, dload, service, rate, dmean, smean, dbytes, dttl, ct_dst_ltm, sload, proto, ct_src_dport_ltm, sinpkt, state, swin and ct_src_ltm.*

The final optimal set was determined by iteratively examining each of the features that was a member of the common features subset and the final optimal feature set constitutes sixteen features which include; *service, sbytes, dbytes, rate, sttl, dload, tcprtt, synack, proto, smean, dmean, ct_srv_src, ct_state_ttl, ct_dst_sport_ltm, ct_dst_src_ltm and ct_srv_dst*

The performance of the five classifiers was evaluated for each selected subset of features from the various selection techniques based on accuracy, precision, recall, FPR and detection time per instance in milliseconds.

4.4.1 Detection Accuracy

Accuracy is a measure of the total number of correctly classified attack traffic instances out of all traffic instances flowing in and out of the network.

Tables 4.4, 4.5, 4.6 and 4.7 indicate the accuracy recorded by the five classifiers when trained and evaluated using subsets of features selected by the filter methods, wrapper methods, embedded methods and the common features subset and proposed feature set respectively.

Table 4. 4: Accuracy of the five algorithms based on features selected using filters methods

Classifier	Pearson Correlation	Chi-Square	Info Gain
DT	95.71	95.58	93.6
RF	97.036	97.1	95.45
AdaBoost	93.34	93.13	91.7
ET	96.66	96.67	94.4
GBC	96.8	96.77	94.98

Table 4. 5: Accuracy of the five algorithms based on features selected using wrapper methods based on RFE

Classifier	LR	RF	DT	GBC	Ada Boost	ET	J48	REP Tree	RF Boruta
DT	95.38	96.37	96.43	96.70	96.76	96.41	93.85	96.83	96.52
RF	96.83	97.50	97.71	97.99	97.94	97.74	95.16	97.16	97.81
AdaBoost	93.64	94.59	95.36	95.46	95.43	94.28	90.87	94.41	95.53
ET	96.53	96.79	96.98	97.34	97.63	97.43	94.14	97.08	97.34
GBC	96.43	97.25	97.24	97.60	97.62	97.37	94.87	97.23	97.59

Table 4. 6: Accuracy of the five algorithms based on features selected using embedded methods

Classifier	LR	RF	GBC	Ada Boost	Extra Trees	DT
DT	95.27	96.37	96.68	96.44	96.56	96.22
RF	96.49	97.35	97.70	97.97	97.77	97.49
AdaBoost	93.74	94.58	95.78	95.52	94.11	94.55
ET	96.19	96.92	97.24	97.32	97.44	97.10
GBC	96.27	97.18	97.46	97.69	97.45	97.51

Table 4. 7: Accuracy of the five algorithms based on most frequently selected features and proposed feature set

Classifier	Common (or most frequent) Features	Proposed feature set
DT	96.53	96.79
RF	97.90	98.12
AdaBoost	95.16	95.54
ET	96.88	97.45
GBC	97.53	97.83

The results in Tables 4.4, 4.5, 4.6 and 4.7 are visualized in Figure 4.2 for better comparison of detection accuracy for the various feature sets with the proposed set.

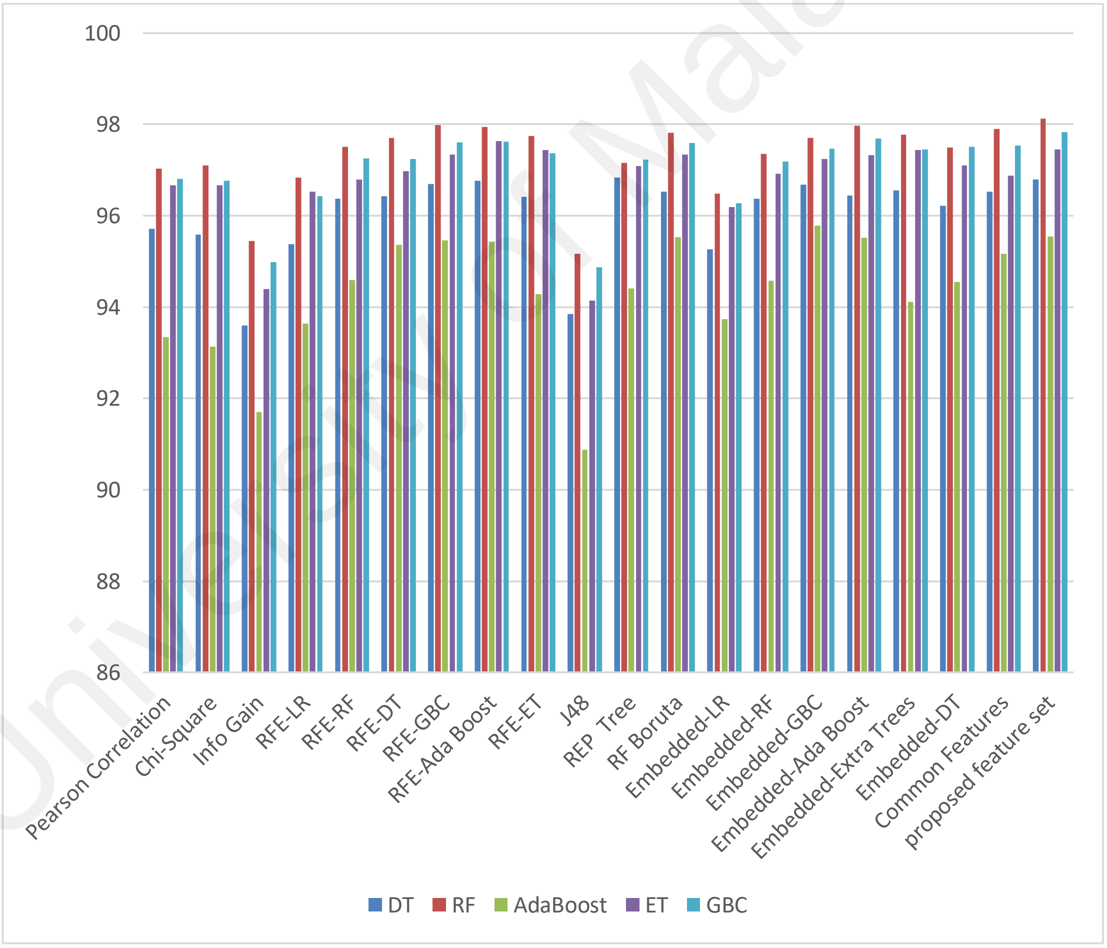

Figure 4. 2: Detection accuracy for the various classifiers evaluated with various subsets of features from the UNSW-NB15 dataset

From Figure 4.2 it can be clearly seen that the proposed feature set outperforms all other feature sets in terms of detection accuracy with random forest having the highest accuracy of 98.12%. This therefore implies that RF is the best choice for optimum detection accuracy.

4.4.2 Precision

The precision gives a fraction of the correctly detected attacks out of the total instances detected as attacks. A high precision indicates good attack detection.

Tables 4.8, 4.9, 4.10 and 4.11 indicate the precision results of the five classifiers used in this research based on features selected by the filter methods, wrapper methods, embedded methods and the common and proposed feature sets respectively.

Table 4. 8: Precision of the five algorithms based on features selected using filter methods

Classifier	Pearson Correlation	Chi-Square	Info Gain
DT	95.91	96.00	94.10
RF	97.32	97.32	97.06
AdaBoost	93.53	93.20	94.15
ET	96.62	96.74	95.31
GBC	97.23	97.16	96.21

Table 4. 9: Precision of the five algorithms based on features selected using wrapper methods

Classifier	LR	RF	DT	GBC	Ada Boost	ET	J48	REP Tree	RF Boruta
DT	95.92	96.50	96.71	97.10	96.92	96.92	94.32	97.14	96.91
RF	97.35	98.23	98.31	98.47	98.46	98.22	96.53	97.75	98.26
Ada Boost	93.38	94.84	95.74	96.09	96.32	94.51	93.24	94.50	96.20
ET	96.63	97.00	97.00	97.81	97.92	97.80	94.81	97.24	97.72
GBC	96.52	97.81	98.13	98.20	98.00	97.92	96.21	97.53	98.33

Table 4. 10: Precision of the five algorithms based on features selected using embedded methods

Classifier	LR	RF	GBC	Ada Boost	Extra Trees	DT
DT	95.61	96.82	96.93	96.91	97.00	96.63
RF	96.36	98.07	98.19	98.62	98.23	98.10
AdaBoost	93.86	94.68	96.58	96.22	94.10	94.95
ET	96.00	97.00	97.41	97.64	97.23	97.23
GBC	96.50	97.70	97.92	98.11	98.00	98.00

Table 4. 11: Precision of the five algorithms based on most frequently selected features and the proposed feature set

Classifier	Common Features	Proposed feature set
DT	96.91	97.55
RF	98.41	98.67
AdaBoost	95.49	95.64
ET	97.00	97.93
GBC	98.10	98.30

For better comparison of precision, the results from Tables 4.8, 4.9, 4.10 and 4.11 are presented graphically by a column chart in Figure 4.3.

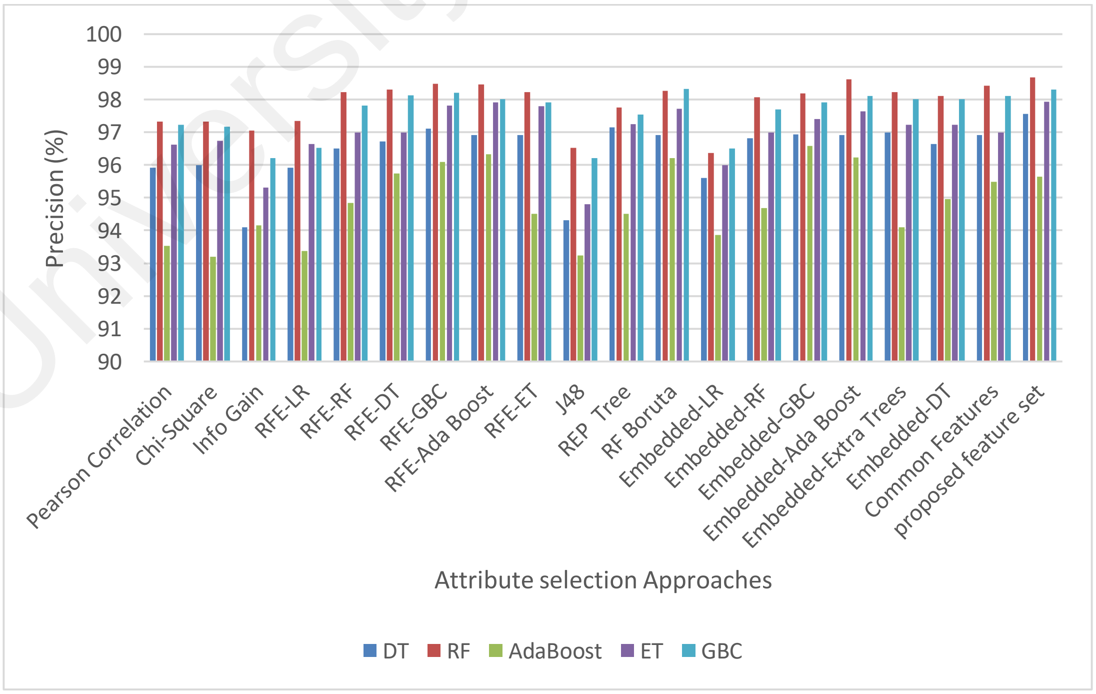

Figure 4. 3: Precision of the various classifiers based on various subsets of features from the UNSW-NB15 dataset

From Figure 4.3 it can be clearly seen that the proposed feature set outperforms all other feature sets in terms of precision with random forest having the highest accuracy of 98.67%. This therefore implies that RF is more precise than all the other classifiers in detection of attacks when trained on the proposed feature set.

4.4.3 Recall

Also known as TPR or sensitivity gives the fraction of detected attacks out of the total attacks. A high recall value implies good intrusion detection.

The recall values achieved by the five classifiers used in this research when trained and evaluated on the various attribute sets selected by the different attribute selection techniques are given in Table 4.12, 4.13, 4.14 and 4.15.

Table 4. 12: Recall values by the five algorithms based on features selected using filter methods

Classifier	Pearson Correlation	Chi-Square	Info Gain
DT	96.25	95.93	94.24
RF	97.29	97.41	94.64
AdaBoost	94.48	94.38	90.60
ET	97.34	97.21	94.73
GBC	97.13	97.01	94.66

Table 4. 13: Recall values by the five algorithms based on features selected using wrapper methods based on RFE

Classifier	LR	RF	DT	GBC	Ada Boost	ET	J48	REP Tree	RF Boruta
DT	95.70	96.94	96.83	96.92	97.18	96.58	94.53	97.20	96.83
RF	96.85	97.19	97.05	97.89	97.78	97.89	94.64	97.12	97.73
Ada Boost	95.21	95.35	95.81	95.58	95.31	93.26	89.93	95.40	95.63
ET	97.18	97.16	97.46	97.33	97.88	97.59	94.61	97.50	97.46
GBC	96.99	97.16	96.89	97.47	97.68	97.29	94.47	97.49	97.40

Table 4. 14: Recall values by the five algorithms based on features selected using embedded methods

Classifier	LR	RF	GBC	Ada Boost	Extra Trees	DT
DT	95.81	96.72	97.15	96.58	96.78	96.57
RF	97.03	97.08	97.62	98.32	97.72	97.33
AdaBoost	94.83	95.48	95.79	95.63	95.24	95.11
ET	97.12	97.44	97.62	97.51	97.75	97.56
GBC	96.76	97.16	97.47	97.69	97.41	97.47

Table 4. 15: Recall values achieved by the five algorithms based on most frequent selected features and the proposed feature set

Classifier	Common Features	Proposed feature set
DT	96.76	96.81
RF	97.75	97.71
AdaBoost	95.77	95.26
ET	97.36	97.68
GBC	97.43	97.91

For better comparison of sensitivity/recall, the results from Tables 4.12, 4.13, 4.14 and 4.15 are represented graphically by a column chart in Figure 4.4.

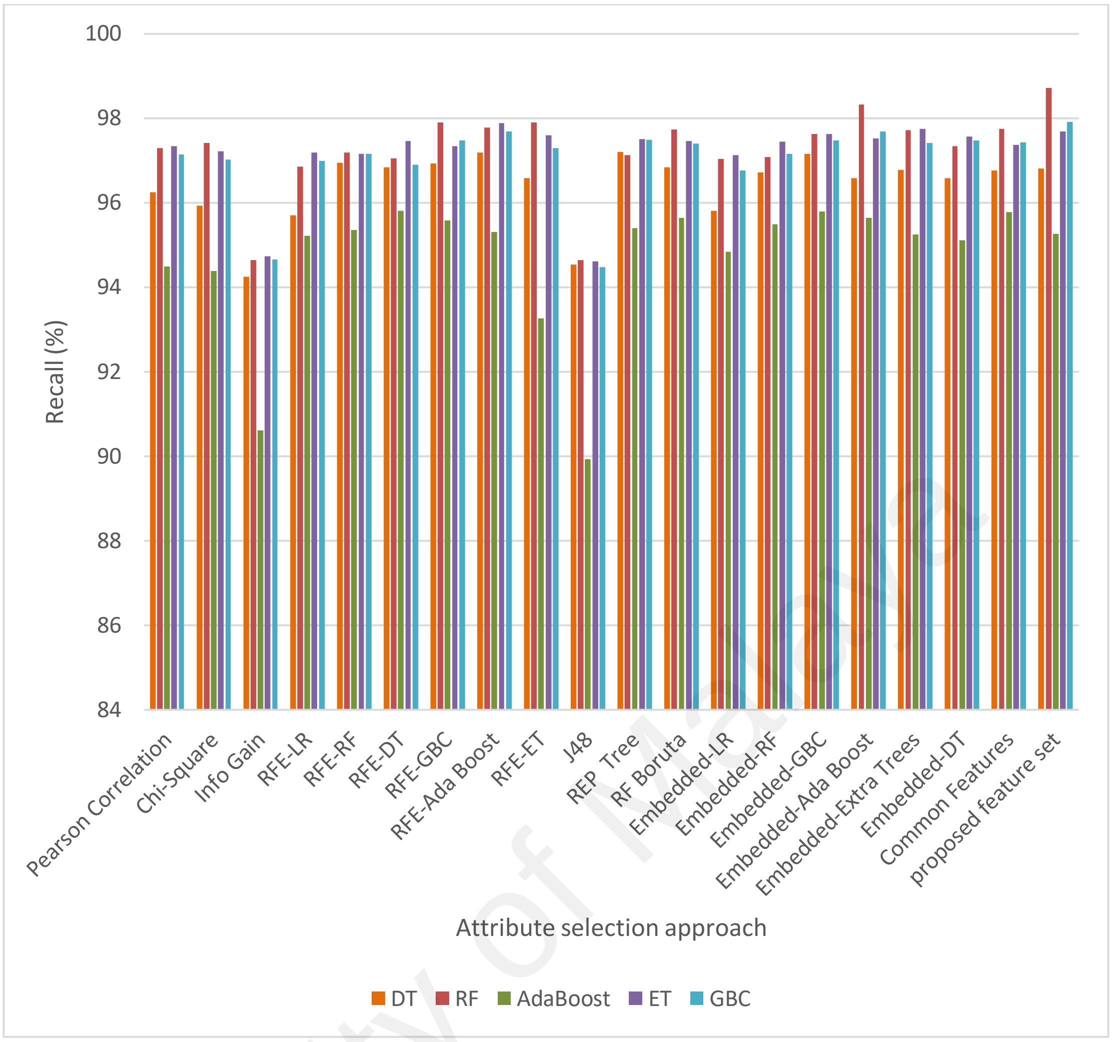

Figure 4. 4: Recall achieved from the various classifiers based on subsets of features of the UNSW-NB15 dataset

It can be seen from Figure 4.4 that the proposed feature set is more sensitive to attacks than all other feature sets with random forest having the highest recall of 98.71%. This therefore implies that RF can accurately detect attacks with significantly very low false negatives than all the other classifiers in detection of attacks when trained on the proposed feature set.

4.4.4 FPR

This, also often referred to as false alarm rate, gives the proportion of normal instances that are falsely reported or marked as attacks out of all normal traffic instances. A lower FPR value implies better detection performance. The FPR values obtained in this research for the different classifiers trained on the various subsets of features selected by the different feature selection techniques are given in Table 4.16, 4.17, 4.18 and 4.19.

Table 4. 16: FPR values by the five algorithms based on features selected using filter methods

Classifier	Pearson Correlation	Chi-Square	Info Gain
DT	4.94	4.84	7.19
RF	3.28	3.27	3.54
AdaBoost	8.06	8.38	6.93
ET	4.19	3.99	6.01
GBC	3.62	3.53	4.61

Table 4. 17: FPR values by the five algorithms based on features selected using wrapper methods based on RFE

Classifier	LR	RF	DT	GBC	Ada Boost	ET	J48	REP Tree	RF Boruta
DT	5.01	4.33	4.05	3.57	3.75	3.79	6.97	3.62	3.86
RF	3.20	2.13	2.04	1.87	1.88	2.11	4.20	2.79	2.11
AdaBoost	8.28	6.33	5.20	4.69	4.43	6.74	7.98	6.79	4.59
ET	2.82	3.68	3.60	2.65	2.68	2.77	6.43	3.43	2.82
GBC	4.24	2.65	2.33	2.24	2.46	2.53	4.65	3.09	2.16

Table 4. 18: FPR values by the five algorithms based on features selected using embedded methods

Classifier	LR	RF	GBC	Ada Boost	Extra Trees	DT
DT	5.4	4.06	3.91	3.74	3.72	4.22
RF	4.16	2.32	2.19	1.68	2.18	2.32
AdaBoost	7.61	6.5	4.21	4.61	7.24	6.133
ET	4.97	3.7	3.23	2.9	2.94	3.48
GBC	4.33	2.79	2.55	2.31	2.50	2.44

Table 4. 19: FPR values from the five algorithms based on most frequent selected features and the proposed feature set

Classifier	Common Features	Proposed feature set
DT	3.76	3.41
RF	1.92	1.62
AdaBoost	5.58	3.38
ET	3.70	2.52
GBC	2.33	2.10

The results in Tables 4.16, 4.17, 4.18 and 4.19 are visualized in Figure 4.5 below for clearer comparison of the performance of the different classifiers against the various feature sets selected by the different attribute selection approaches.

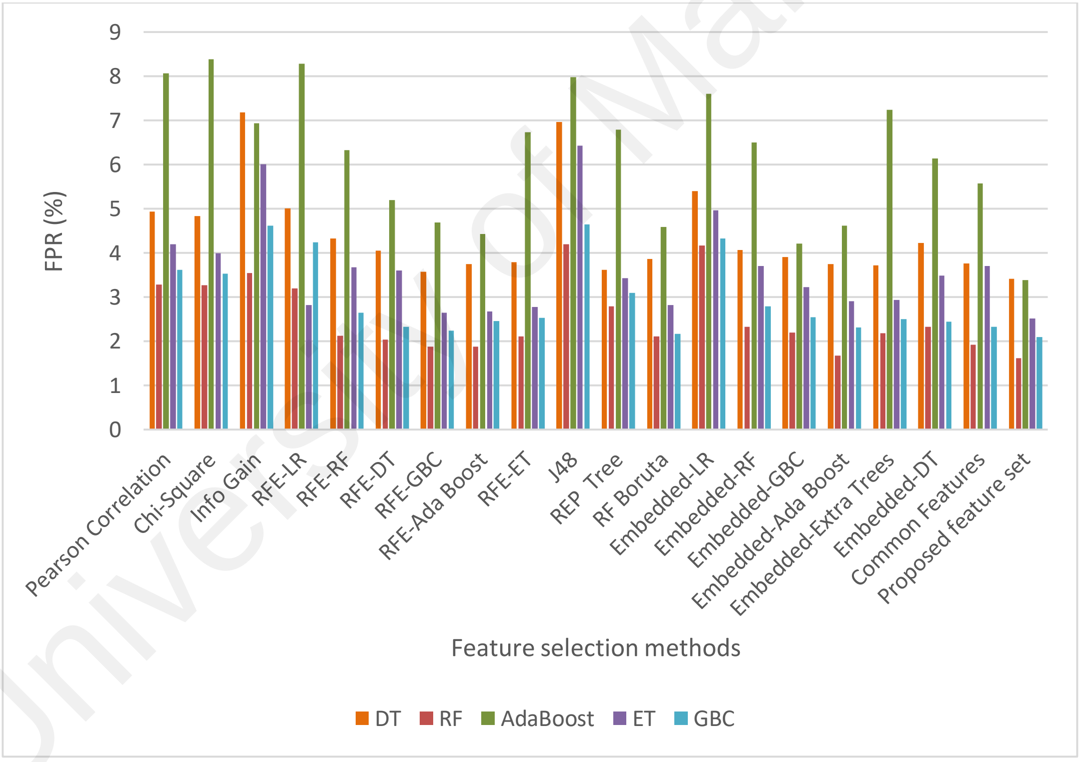

Figure 4. 5: FPR values from the various classifiers based on various subsets of features of the UNSW-NB15 dataset

It can be seen from Figure 4.5 that the proposed feature set produces low false detections than all other feature sets with random forest having the lowest TPR value of 3.38%. This therefore implies that the IDS will produce less false alarms when trained on the proposed features set.

4.4.5 Detection time

This is a measure of the time in milliseconds a classifier takes to determine the class (i.e. whether an attack or a normal instance) of a single traffic instance. Detection time depends on many factors like algorithm used, capacity of the computer system in terms of RAM, CPU, number of features to examine among other factors. This research focused on examining number of attributes so as to establish the class of any given traffic instance given a particular classifier. Table 4.20, 4.21, 4.22 and 4.23 give the detection time for the different algorithms trained and evaluated on the various subsets of features obtained from the various attribute selection techniques. It is worth noting that the detection time is in the order of $\times 10^{-3} ms$

Table 4. 20: detection time per instance by the five algorithms based on features selected using filter methods

Classifier	Pearson Correlation	Chi-Square	Info Gain
DT	0.41	0.33	0.41
RF	11.8	11.48	11.48
AdaBoost	13.12	14.32	13.03
ET	1.64	1.64	1.64
GBC	4.51	4.18	4.52

Table 4. 21: Detection time by the five algorithms based on features selected using wrapper methods based on RFE

Classifier	LR	RF	DT	GBC	Ada Boost	ET	J48	REP Tree	RF Boruta
DT	0.41	0.08	0.33	0.33	0.41	0.00	0.00	0.41	0.41
RF	11.88	10.58	10.74	10.25	10.25	10.66	10.99	11.07	11.16
Ada Boost	12.30	13.51	13.13	13.01	13.53	13.05	12.94	13.11	13.86
ET	1.64	1.66	1.64	1.55	1.63	1.23	1.63	1.62	2.45
GBC	4.10	4.88	3.78	4.92	3.25	4.92	5.70	5.41	4.47

Table 4. 22: Detection time by the five algorithms based on features selected using embedded methods

Classifier	LR	RF	GBC	Ada Boost	Extra Trees	DT
DT	0.00	0.33	0.32	0.33	0.00	0.33
RF	11.89	10.67	10.66	10.57	11.03	10.66
AdaBoost	13.13	13.03	13.54	12.95	12.72	12.72
ET	1.64	1.64	1.64	1.31	1.64	1.64
GBC	4.88	5.25	5.15	3.74	4.07	4.50

Table 4. 23: Detection time by the five algorithms based on most frequently selected features and the proposed feature set

Classifier	Common Features	Proposed feature set
DT	0.33	0.00
RF	10.59	10.10
AdaBoost	13.92	11.01
ET	1.64	1.02
GBC	3.60	0.23

The detection time for the five classifiers under the different features sets from the various selection techniques are visualized by the column graph in Figure 4.6.

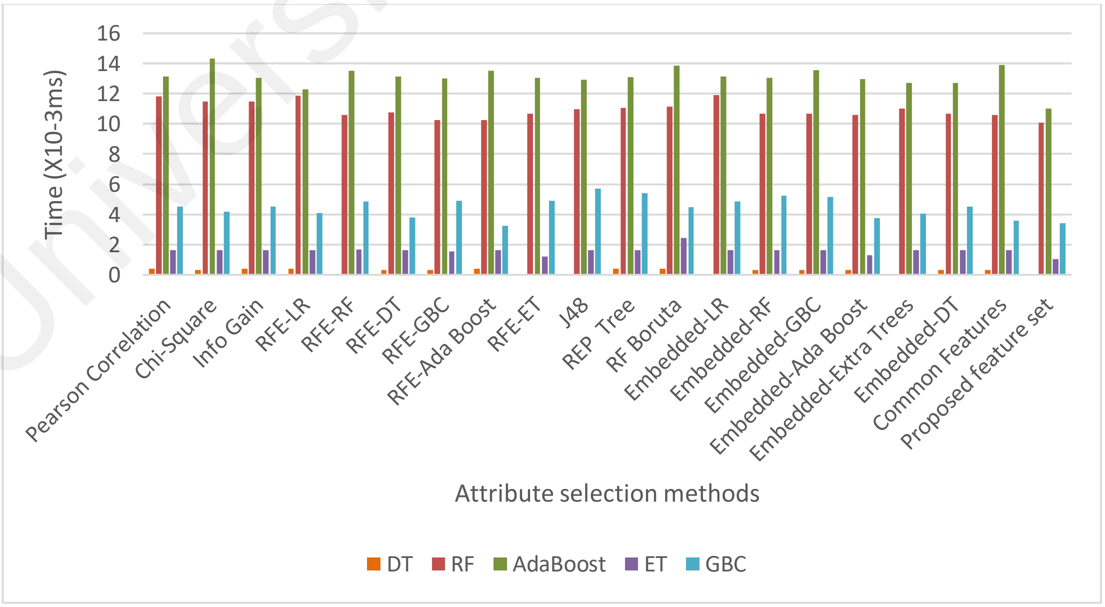

Figure 4. 6: Detection time per traffic instance by the various classifiers based on subsets of features from the UNSW-NB15 dataset

It can be seen from Figure 4.5 that the proposed feature set outperforms all other feature sets in terms of detection time with decision tree almost detecting instances in real time. This therefore clearly shows that the proposed feature set is optimized for anomaly-based intrusion detection.

4.5 Conclusion

This chapter presented the experimental results of this research. It clearly gives a visual presentation for the performance of various machine learning algorithms trained and test on different feature sets used in this study. In addition, each graph is accompanied by an explanation of the implication of the results on the IDS's performance.

CHAPTER 5: DISCUSSION

In the field of information security, a single undetected traffic instance can be extremely disastrous and can lead to an unrecoverable loss to the business. Therefore, any business must highly prioritize realization of the fundamental principles of confidentiality, integrity and availability in order to remain competitive in the current enterprise environment. This can be achieved by establishing and constantly maintaining a comprehensive defense-in-depth strategy. One of the famous measures that have proved to be highly vital in realizing a secure information system is deployment of anomaly-based network IDSs owing to their ability to detect both internal and external, known and novel attacks.

The efficiency of any anomaly-based IDS is dependent on three major factors i.e. the processing capacity of the hardware on which the system is installed, the choice of classifier and the dimensionality and relevancy of features on which the detection model is trained. However, once hardware is installed, it does not scale with the quantity of traffic instances flowing through the system and therefore for a given hardware system, it is important to identify and optimize the most efficient classification algorithm for ANIDS. Classifier optimization can be realized by training it on only features that provide pertinent information for traffic classification. In addition, the training must be conducted on a considerably comprehensive dataset that gives a good reflection of the state-of-the-art network traffic.

For the last two decades various ANIDSs have been proposed in an effort to embed novel attack detection capabilities in NIDSs. However, the training of many of these detection systems has been conducted on the famous KDD'99 dataset. Though seems comprehensive because of the considerably large number of both normal and attack traffic instances, this dataset does not represent the state-of-the-art attack traffic owing to its age.

This is due to the fact that between its creation and this day, a large number of traffic attributes have been engineered which are not reflected in this dataset rendering it obsolete and unfit for training modern ANIDSs.

In addition, finding the best classification algorithm and relevant features for building ANIDSs is one of the most challenging and time consuming tasks for system administrators today. To find the best algorithm for this study, nine most popular machine learning algorithms were identified from literature and trained them on the full UNSW-NB15 dataset without feature selection. Four out of the nine classifiers were then discarded including NB which produced the lowest detection accuracy, precision and highest FPR of 71.53%, 72.62% and 36.70% respectively, LR which produced the lowest recall of 66.12% and SVM and KNN which had the longest detection time of 5.70ms and 0.77ms per traffic instance respectively. The fact that SVM produced the longest detection time and that NB produced the worst detection performance is supported by (Belouch et al, 2018) where NB classifier had the worst classification accuracy of 74.19% while SVM produced the longest training and prediction time of 38.91s and 0.20s respectively.

The five classifiers that were chosen for further experiments include DT, RF, AdaBoost, ET Classifier and Gradient Boosting classifier in order to find an optimal feature set for anomaly-based attack detection which was the main goal of this research.

The three feature selection methods i.e. filter, wrapper and embedded methods were each separately deployed to produce subsets of features that were deemed relevant for classification of traffic as either normal or anomaly. It is however, worth noting that no two approaches produced the same set of relevant features and different classifiers had their best performance for different sets of features. In other words, a certain algorithm would produce its best performance on one subset and another algorithm would perform better on another set.

For instance, before combining the three selection techniques to produce our optimal feature set, random forest (RF) recorded its best accuracy of 97.99% on features selected using wrapper method (Recursive Feature Elimination using GBC as a learning algorithm), GBC recorded its best accuracy of 97.69% on features selected using Embedded method (using AdaBoost as a learning algorithm) while DT, AdaBoost and ET recorded best accuracy of 96.83% on features selected using wrapper method (based on REP Tree), 95.78% on features selected using Embedded method (Recursive Feature Elimination using GBC as a learning algorithm) and 97.63% on features selected using wrapper method (Recursive Feature Elimination using AdaBoost as a learning algorithm) respectively.

The above scenario was recorded for all five performance metrics used in this research i.e. a classifier would produce the best recall on one feature set and record a lower FPR or detection time on another feature set. This made drawing inferences of which approach performs better in identifying the best pertinent set of features for intrusion detection hard. Therefore, this would lead to wastage of productive time and resources in determining which of the selection approach to deploy in order to find the best feature set for training an anomaly-based IDS. In addition, the administrator can hardly find a best-classifier-best-feature-set combination for building an IDS.

This research solved the above problem by proposing an optimized feature set for building and training ANIDSs. The performance of the proposed feature set was compared with the performance of each individual feature set obtained by deployment of single attribute selection methods against all five classifiers. The proposed feature set outperformed all feature sets from single selection methods with all the five classifiers as measured by accuracy, precision, recall, FPR and detection time. Using the proposed optimized feature set, RF recorded the best accuracy, precision, recall and lowest FPR of 98.12%, 98.67%, 98.71% and 1.62% respectively. The fact that random forest outperformed other classifiers is support by (Belouch et al, 2018) where the authors reported that RF recorded the best accuracy, sensitivity and recall of 97.49%, 93.53% and 97.75% respectively. In addition, (Janarthanan & Zargari 2017) reported that after evaluating various classifiers, RF was selected for their study as it outperformed other classification algorithms.

Decision tree recorded shortest detection time of approximately 0.0ms per traffic instance. This can be attributed to the fact that each of the other four classifiers use a collection of weak classifiers to predict the final class which requires more processing time. This is again supported by (Belouch et al, 2018) who reported DT as the fastest classifier in their study with a detection time of 0.13s.

The uniqueness of this research is the combination of all three feature selection techniques (filter, wrapper and embedded methods) to find a feature set that optimizes ANIDSs performance. To ensure optimal performance of the feature set, performance evaluation was performed on the best five classifiers identified after examining nine most popular classification algorithms in the field of anomaly-based intrusion detection.

5.1 Conclusion

In this chapter a detailed analysis of the research results is presented. The results are evaluated by comparing them with results from existing studies. The chapter also clearly highlight the relevancy of the research results with regard to intrusion detection systems and network security enhancement. Finally, the chapter highlights the applicability of the proposed feature set by stating its relevance in information assurance in relation to other approaches.

CHAPTER 6: CONCLUSION

This section concludes this work by highlighting how the research objectives, as highlighted in section 1.4 of this work, were achieved, contributions of the study and a brief description of future work.

6.1 Achievement of research objectives

This study aimed at achieving three research objectives as in section 1.4. Objective one was realized through conducting a review of the related literature where a clear insight of the operational mechanisms of various feature selection methods, their characteristics, advantages and weaknesses were highlighted. This can greatly help other researchers and systems administrators in determining which feature selection approach to apply under what circumstance.

The second objective, which happens to be the core of this research, was achieved by combining results from the three feature selection methods i.e. filter, wrapper and embedded methods to propose an optimized feature set.

Finally, the last objective was realized by examining the performance of each of the feature set produced by various single selection techniques and the performance of the optimized feature set proposed in this research using five classifiers. The goodness of the feature sets was compared with the goodness of other sets on the basis of five performance metrics as highlighted in section 4.3. Random forest recorded the highest detection accuracy of 98.12% which is higher than 97.49% (by 0.63%), the highest achieved so far in literature using UNSW-NB15 dataset (Belouch, Hadaj, & Idhammad, 2018) precision and lowest FPR of 1.62% with our optimized feature set.

6.2 Contribution of the research

This research has two major contributions. First, the research introduces the idea of combining all three feature selection methods (filter, wrapper and embedded) for identification of relevant features. This approach can be applied for feature selection in building other ML based systems and models. Secondly, it proposed an optimized set of features for ANIDS. This will greatly save time that would otherwise be wasted in finding relevant features for building intrusion detection systems. As indicated before, the optimized feature, evaluated using UNSW-NB15 dataset and RF classifier, produced good detection accuracy of up to 98.12%, precision of 98.67%, recall of 97.71% and a very low FPR of 1.62%. This outperformed all previous studies which deployed the feature selection and UNSW-NB15 dataset for building and evaluation of IDS models as highlighted in chapter 2.

6.3 Limitations of the research

Although this research produced outstanding results over all existing studies, it has two major limitations. First, due to absence of recent malware datasets, its evaluation was done using only one dataset which might make it had to generalise to other datasets. Second, the study is limited to only supervised learning where the training dataset has to be labelled which is not always the case.

6.4 Future work

Owing to the fact that signature-based IDS are very accurate for known attacks and having identified RF as an outstanding classifier in terms of intrusion detection when trained on our proposed optimal feature set, our future research will focus on finding a mechanism of integrating the two detection methods for enhanced information security. We intend to examine the various signature-based IDSs available to select the best that can be integrated with RF so as to realise higher intrusion detection accuracy with very low FPR.

REFERENCES

Aburomman, A. A., & Reaz, M. B. (2016). A novel SVM-kNN-PSO ensemble method for intrusion detection system. *Applied Soft Computing,38*, 360-372., doi:10.1016/j.asoc.2015.10.011

Ahmad, I., Basheri, M., Iqbal, M. J., & Rahim, A. (2018). Performance Comparison of Support Vector Machine, Random Forest, and Extreme Learning Machine for Intrusion Detection. *IEEE Access,6*, 33789-33795., doi: 10.1109/ACCESS.2018.2841987

Al-Jarrah, O., Siddiqui, A., Elsalamouny, M., Yoo, P., Muhaidat, S., & Kim, K. (2014). Machine-Learning-Based Feature Selection Techniques for Large-Scale Network Intrusion Detection. In *2014 IEEE 34th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, Madrid, 2014, pp. 177-181. doi: 10.1109/ICDCSW.2014.14

Aljawarneh, S., Aldwairi, M., & Yassein, M. B. (2018). Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. *Journal of Computational Science,25*, 152-160., doi: 10.1016/j.jocs.2017.03.006

Alotaibi, B., & Elleithy, K. (2016). A majority voting technique for Wireless Intrusion Detection Systems. *2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*. Farmingdale, NY, 2016, doi: 10.1109/lisat.2016.7494133

Alpaydin, E. (2010). *Introduction to machine learning* (2nd ed.). Cambridge, MA.

Ambusaidi, M. A., He, X., Nanda, P., & Tan, Z. (2016). Building an Intrusion Detection System Using a Filter-Based Feature Selection Algorithm. *IEEE Transactions on Computers,65*(10), 2986-2998., doi: 10.1109/TC.2016.2519914

Ambusaidi, M. A., He, X., Tan, Z., Nanda, P., Lu, L. F., & Nagar, U. T. (2014). A Novel Feature Selection Approach for Intrusion Detection Data Classification. *2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications*, *TrustCom 2014* (pp. 82-89). IEEE Computer Society. https://doi.org/10.1109/TrustCom.2014.15

Anwer, H. M., Farouk, M., & Abdel-Hamid, A. (2018). A framework for efficient network anomaly intrusion detection with features selection. *2018 9th International Conference on Information and Communication Systems (ICICS)*, Irbid, 2018., doi: 10.1109/iacs.2018.8355459

Arjunan, K., & Modi, C. N. (2017). An enhanced intrusion detection framework for securing network layer of cloud computing. *2017 ISEA Asia Security and Privacy (ISEASP)*., doi: 10.1109/iseasp.2017.7976988

Aung, Y. Y., & Min, M. M. (2018). Hybrid Intrusion Detection System using K-means and K-Nearest Neighbors Algorithms. *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*, Singapore, 2017., doi: 10.1109/icis.2018.8466537

B, S., & K, M. (2019). Firefly algorithm based feature selection for network intrusion detection. *Computers & Security,81*, 148-155., doi: 10.1016/j.cose.2018.11.005

Bapat, R., Mandya, A., Liu, X., Abraham, B., Brown, D. E., Kang, H., & Veeraraghavan, M. (2018). Identifying malicious botnet traffic using logistic regression. In *2018 Systems and Information Engineering Design Symposium, SIEDS* (pp 266-271). (*2018 Systems and Information Engineering Design Symposium, SIEDS 2018). Institute of Electrical and Electronics Engineers Inc.*, https://doi: 10.1109/sieds.2018.8374749

Belouch, M., Hadaj, S. E., & Idhammad, M. (2018). Performance evaluation of intrusion detection based on machine learning using Apache Spark. *Procedia Computer Science,127*, 1-6., doi: 10.1016/j.procs.2018.01.091

Bendovschi, A. (2015). Cyber-Attacks – Trends, Patterns and Security Countermeasures. *Procedia Economics and Finance,28*, 24-31., doi: 10.1016/s2212-5671(15)01077-1

Bhosale, D. A., & Mane, V. M. (2015). Comparative study and analysis of network intrusion detection tools. *2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*. Davangere, 2015., doi: 10.1109/icatcct.2015.7456901

Bijani, S., & A., M. K. (2008). HIDMN: A Host and Network-Based Intrusion Detection for Mobile Networks. *2008 International Conference on Computer and Electrical Engineering (ICCEE),* Phutek, 2008 pp. 204-208., doi: 10.1109/iccee.2008.183

Breiman, L., 2001. Random forests. Mach. Learn. 45 (1).

Business cyber crime up 63%, UK stats show. (2018, November 1). Retrieved from *http://www.computerweekly.com/news/252433873/Business-cyber-crime-up-63-UK-stats-show*

Cao, W., Qian, S., Wu, S., & Wong, H. (2019). Unsupervised Multi-task Learning with Hierarchical Data Structure. *Pattern Recognition,86*, 248-264., doi: 10.1016/j.patcog.2018.08.021

Chang, Y., Li, W., & Yang, Z. (2017). Network Intrusion Detection Based on Random Forest and Support Vector Machine. *22017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC),* Guangzhou, 2017., doi: 10.1109/cse-euc.2017.118

Chellam, A., L, R., & S, R. (2018). Intrusion Detection in Computer Networks using Lazy Learning Algorithm. *Procedia Computer Science,132*, 928-936., doi: 10.1016/j.procs.2018.05.108

Chen, W., Hsu, S., & Shen, H. (2005). Application of SVM and ANN for intrusion detection. *Computers & Operations Research,32*(10), 2617-2634., doi: 10.1016/j.cor.2004.03.019

Choi, S., Ko, D., Hwang, S., & Choi, Y. (2018). Memory-Efficient Random Forest Generation Method for Network Intrusion Detection. *2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)*, Prague, 2018, pp. 305-307., doi: 10.1109/icufn.2018.8436590

Colom, J. F., Gil, D., Mora, H., Volckaert, B., & Jimeno, A. M. (2018). Scheduling framework for distributed intrusion detection systems over heterogeneous network architectures. *Journal of Network and Computer Applications,108*, 76-86., doi: 10.1016/j.jnca.2018.02.004

Cybercrime report. (2018, November 1). Retrieved from *https://www.interpol.int/Crime-areas/Cybercrime/Cybercrime*

Gao, M., Tian, J., & Xia, M. (2009). Intrusion Detection Method Based on Classify Support Vector Machine. Proceedings of *2009 Second International Conference on Intelligent Computation Technology and Automation,02,* Changsha, Hunan, 2009., doi: 10.1109/icicta.2009.330

Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning,63*(1), 3-42., doi: 10.1007/s10994-006-6226-1

Ghosh, P., & Mitra, R. (2015). Proposed GA-BFSS and logistic regression based intrusion detection system. *Proceedings of the 2015 Third International Conference on Computer, Communication, Control and Information Technology (C3IT),* Hooghly, 2015, pp. 1-6., doi: 10.1109/c3it.2015.7060117

Guggari, S., Kadappa, V., & Umadevi, V. (2018). Non-sequential partitioning approaches to decision tree classifier. *Future Computing and Informatics Journal,3*(2), 275-285., doi: 10.1016/j.fcij.2018.06.003

Gul, A., & Adali, E. (2017). A feature selection algorithm for IDS. *2017 International Conference on Computer Science and Engineering (UBMK),* Antalya, 2017, pp 816-820., doi: 10.1109/ubmk.2017.8093538

Guo, Y., Wang, B., Zhao, X., Xie, X., Lin, L., & Zhou, Q. (2010). Feature selection based on Rough set and modified genetic algorithm for intrusion detection. *2010 5th International Conference on Computer Science & Education*, Hefei, 2010, pp. 1441-1446., doi: 10.1109/iccse.2010.5593765

Hajisalem, V., & Babaie, S. (2018). A hybrid intrusion detection system based on ABC-AFS algorithm for misuse and anomaly detection. *Computer Networks,136*, 37-50., doi: 10.1016/j.comnet.2018.02.028

Hamed, T., Dara, R., & Kremer, S. C. (2014). An Accurate, Fast Embedded Feature Selection for SVMs. *2014 13th International Conference on Machine Learning and Applications*, Detroit, MI, 2014, pp. 145-140., doi: 10.1109/icmla.2014.104

Han, J., Kamber, M., & Pei, J. (2012). *Data mining: Concepts and techniques*. Waltham, MA: Morgan Kaufmann.

Han, X., Xu, L., Ren, M., & Gu, W. (2015). A Naive Bayesian Network Intrusion Detection Algorithm Based on Principal Component Analysis. *2015 7th International Conference on Information Technology in Medicine and Education (ITME)*, Huangshan, 2015, pp. 325-328., doi: 10.1109/itme.2015.29

Harzevili, N. S., & Alizadeh, S. H. (2018). Mixture of latent multinomial naive Bayes classifier. *Applied Soft Computing,69*, 516-527., doi: 10.1016/j.asoc.2018.04.020

Hu, W., Hu, W., & Maybank, S. (2008). AdaBoost-Based Algorithm for Network Intrusion Detection. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics),38*(2), 577-583., doi: 10.1109/TSMCB.2007.914695

Hubballi, N., & Suryanarayanan, V. (2014). False alarm minimization techniques in signature-based intrusion detection systems: A survey. *Computer Communications,49*, 1-17., doi: 10.1016/j.comcom.2014.04.012

Idowu, R. K., Maroosi, A., Muniyandi, R. C., & Othman, Z. A. (2013). An Application of Membrane Computing to Anomaly-based Intrusion Detection System. *Procedia Technology,11*, 585-592., doi: 10.1016/j.protcy.2013.12.232

IoT: Number of connected devices worldwide 2012-2025. (2018, November 1). Retrieved from https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

Jabbar, M. A., & Samreen, S. (2016). Intelligent network intrusion detection using alternating decision trees. *2016 International Conference on Circuits, Controls, Communications and Computing (I4C)*, Bangalore, 2016, pp. 1-6., doi: 10.1109/cimca.2016.8053265

Janarthanan, T., & Zargari, S. (2017). Feature selection in UNSW-NB15 and KDDCUP99 datasets. *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, Edinburgh, 2017, pp. 1881-1886., doi: 10.1109/isie.2017.8001537

Jianhong, H. (2015). Network Intrusion Detection Algorithm Based on Improved Support Vector Machine. *2015 International Conference on Intelligent Transportation, Big Data and Smart City,* Halong Bay, 2015., doi: 10.1109/icitbs.2015.135

Jin, X., Hou, X., & Liu, C. (2010). Multi-class AdaBoost with Hypothesis Margin. *2010 20th International Conference on Pattern Recognition*, Istanbul, 2010, pp. 65-68., doi: 10.1109/icpr.2010.25

Kabir, E., Hu, J., Wang, H., & Zhuo, G. (2018). A novel statistical technique for intrusion detection systems. *Future Generation Computer Systems,79*, 303-318., doi: 10.1016/j.future.2017.01.029

Kaneko, H. (2018). Illustration of merits of semi-supervised learning in regression analysis. *Chemometrics and Intelligent Laboratory Systems,182*, 47-56., doi: 10.1016/j.chemolab.2018.08.015

Karami, A. (2018). An anomaly-based intrusion detection system in presence of benign outliers with visualization capabilities. *Expert Systems with Applications,108*, 36-60., doi: 10.1016/j.eswa.2018.04.038

Katkar, V. D., & Bhatia, D. S. (2013). Experiments on detection of Denial of Service attacks using REPTree. *2013 International Conference on Green Computing, Communication and Conservation of Energy (ICGCE)*, Chennai, 2013, pp. 713-718., doi: 10.1109/icgce.2013.6823527

Kulariya, M., Saraf, P., Ranjan, R., & Gupta, G. P. (2016). Performance analysis of network intrusion detection schemes using Apache Spark. *2016 International Conference on Communication and Signal Processing (ICCSP)*, Melmaruvathur, 2016, pp 1973-1977., doi: 10.1109/iccsp.2016.7754517

Kursa, M. B., & Rudnicki, W. R. (2010). Feature Selection with theBorutaPackage. *Journal of Statistical Software,36*(11)., doi: 10.18637/jss.v036.i11

Kushwaha, P., Buckchash, H., & Raman, B. (2017). Anomaly based intrusion detection using filter based feature selection on KDD-CUP 99. *TENCON 2017 - 2017 IEEE Region 10 Conference*, Penang, 2017, pp. 839-844., doi: 10.1109/tencon.2017.8227975

Kuzey, C., Karaman, A. S., & Akman, E. (2019). Elucidating the impact of visa regimes: A decision tree analysis. *Tourism Management Perspectives,29*, 148-156., doi: 10.1016/j.tmp.2018.11.008

Li, H., Guo, W., Wu, G., & Li, Y. (2018). A RF-PSO Based Hybrid Feature Selection Model in Intrusion Detection System. *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, Guangzhou, 2018, pp. 795-802., doi: 10.1109/dsc.2018.00128

Li, L., Zhang, H., Peng, H., & Yang, Y. (2018). Nearest neighbors based density peaks approach to intrusion detection. *Chaos, Solitons & Fractals,110*, 33-40., doi: 10.1016/j.chaos.2018.03.010

Li, M. (2017). Application of CART decision tree combined with PCA algorithm in intrusion detection. *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*., doi: 10.1109/icsess.2017.8342859

Li, W., & Li, Q. (2010). Using Naive Bayes with AdaBoost to Enhance Network Anomaly Intrusion Detection. *2010 Third International Conference on Intelligent Networks and Intelligent Systems*, Shenyang, 2010, pp. 486-489., doi: 10.1109/icinis.2010.133

Li, Y., & Guo, L. (2007). An active learning based TCM-KNN algorithm for supervised network intrusion detection. *Computers & Security,26*(7-8), 459-467., doi: 10.1016/j.cose.2007.10.002

Liu, Q., Li, P., Zhao, W., Cai, W., Yu, S., & Leung, V. C. (2018). A Survey on Security Threats and Defensive Techniques of Machine Learning: A Data Driven View. *IEEE Access,6*, 12103-12117., doi: 10.1109/access.2018.2805680

Liu, Y., Xu, Z., Yang, J., Wang, L., Song, C., & Chen, K. (2016). A Novel Meta-Heuristic-Based Sequential Forward Feature Selection Approach for Anomaly Detection Systems. *2016 International Conference on Network and Information Systems for Computers (ICNISC)*, Wuhan, 2016, pp. 218-227., doi: 10.1109/icnisc.2016.056

Lou, Y., & Tsai, J. J. (2008). A Framework for Extrusion Detection Using Machine Learning. *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, Orlando, FL, 2008, pp. 83-88., doi: 10.1109/isorc.2008.70

Maitra, S., Madan, S., Kandwal, R., & Mahajan, P. (2018). Mining authentic student feedback for faculty using Naïve Bayes classifier. *Procedia Computer Science,132*, 1171-1183., doi: 10.1016/j.procs.2018.05.032

Malhotra, S., Bali, V., & Paliwal, K. K. (2017). Genetic programming and K-nearest neighbour classifier based intrusion detection model. *2017 7th International Conference on Cloud Computing, Data Science & Engineering - Confluence*, Noida, 2017., doi: 10.1109/confluence.2017.7943121

Malik, A. J., & Khan, F. A. (2013). A Hybrid Technique Using Multi-objective Particle Swarm Optimization and Random Forests for PROBE Attacks Detection in a Network. *2013 IEEE International Conference on Systems, Man, and Cybernetics*, Manchester, 2013, pp. 2473-2478., doi: 10.1109/smc.2013.422

Malik, A. J., Shahzad, W., & Khan, F. A. (2011). Binary PSO and random forests algorithm for PROBE attacks detection in a network. *2011 IEEE Congress of Evolutionary Computation (CEC)*, New Orleans, LA, 2011, pp. 662-668., doi: 10.1109/cec.2011.5949682

Min, E., Long, J., Liu, Q., Cui, J., & Chen, W. (2018). TR-IDS: Anomaly-Based Intrusion Detection through Text-Convolutional Neural Network and Random Forest. *Security and Communication Networks,2018*, 1-9., doi: 10.1155/2018/4943509

Mitra, S., Avra, L., & Mccluskey, E. (n.d.). Scan synthesis for one-hot signals. *Proceedings International Test Conference 1997*, Washington, DC, 1997., doi: 10.1109/test.1997.639684

Mohammadi, S., Mirvaziri, H., Ghazizadeh-Ahsaee, M., & Karimipour, H. (2019). Cyber intrusion detection by combined feature selection algorithm. *Journal of Information Security and Applications,44*, 80-88., doi: 10.1016/j.jisa.2018.11.007

Moustafa, N., & Slay, J. (2015a). The Significant Features of the UNSW-NB15 and the KDD99 Data Sets for Network Intrusion Detection Systems. *2015 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS),* Kyoto, 2015, pp. 25-31., doi: 10.1109/badgers.2015.014

Moustafa, N., & Slay, J. (2015b). UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). *2015 Military Communications and Information Systems Conference (MilCIS)*, Canberra, ACT, 2015, pp.1-6., doi: 10.1109/milcis.2015.7348942

Moustafa, N., & Slay, J. (2016). The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Information Security Journal: A Global Perspective,25*(1-3), 18-31., doi: 10.1080/19393555.2015.1125974

Mukherjee, S., & Sharma, N. (2012). Intrusion Detection using Naive Bayes Classifier with Feature Reduction. *Procedia Technology,4*, 119-128., doi: 10.1016/j.protcy.2012.05.017

Naik, N., Diao, R., & Shen, Q. (2016). Application of dynamic fuzzy rule interpolation for intrusion detection: D-FRI-Snort. *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, Vancouver, BC, 2016, pp. 78-85., doi: 10.1109/fuzz-ieee.2016.7737671

Natesan, P., & Rajesh, P. (2012). Cascaded classifier approach based on Adaboost to increase detection rate of rare network attack categories. *2012 International Conference on Recent Trends in Information Technology*, Chennai, Tamil Nadu, 2012, pp. 417-422., doi: 10.1109/icrtit.2012.6206789

Nguyen, C. D., Costa, A. C., Cios, K. J., & Gardiner, K. J. (2011). Machine Learning Methods Predict Locomotor Response to MK-801 in Mouse Models of Down Syndrome. *Journal of Neurogenetics,25*(1-2), 40-51., doi: 10.3109/01677063.2011.558606

Nguyen, K. K., Hoang, D. T., Niyato, D., Wang, P., Nguyen, D., & Dutkiewicz, E. (2018). Cyberattack detection in mobile cloud computing: A deep learning approach. *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, Barcelona, 2018, pp. 1-6., doi: 10.1109/wcnc.2018.8376973

Niksefat, S., Kaghazgaran, P., & Sadeghiyan, B. (2017). Privacy issues in intrusion detection systems: A taxonomy, survey and future directions. *Computer Science Review,25*, 69-78., doi: 10.1016/j.cosrev.2017.07.001

Onpans, J., Rasmequan, S., Jantarakongkul, B., Chinnasarn, K., & Rodtook, A. (2013). Intrusion feature selection using Modified Heuristic Greedy Algorithm of Itemset. *2013 13th International Symposium on Communications and Information Technologies (ISCIT)*, Surat Thani, 2013, pp. 627-632., doi: 10.1109/iscit.2013.6645936

Panda, M., Abraham, A., & Patra, M. R. (2010). Discriminative multinomial Naïve Bayes for network intrusion detection. *2010 Sixth International Conference on Information Assurance and Security*, Atlanta, GA, 2010, pp. 5-10., doi: 10.1109/isias.2010.5604193

Park, K., Song, Y., & Cheong, Y. (2018). Classification of Attack Types for Intrusion Detection Systems Using a Machine Learning Algorithm. *2018 IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService)*, Bamberg, 2018, pp. 282-286., doi: 10.1109/bigdataservice.2018.00050

Pei, H., Wang, K., Lin, Q., & Zhong, P. (2018). Robust semi-supervised extreme learning machine. *Knowledge-Based Systems,159*, 203-220., doi: 10.1016/j.knosys.2018.06.029

Prashanth, G., Prashanth, V., Jayashree, P., & Srinivasan, N. (2008). Using Random Forests for Network-based Anomaly detection at Active routers. *2008 International Conference on Signal Processing, Communications and Networking*, Chennai, 2008, pp. 93-96., doi: 10.1109/icscn.2008.4447167

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning,1*(1), 81-106.

Rathore, M. M., Paul, A., Ahmad, A., Rho, S., Imran, M., & Guizani, M. (2016). Hadoop Based Real-Time Intrusion Detection for High-Speed Networks. *2016 IEEE Global Communications Conference (GLOBECOM)*, Washington, DC, 2016, pp. 1-6., doi: 10.1109/glocom.2016.7841864

Roshan, S., Miche, Y., Akusok, A., & Lendasse, A. (2018). Adaptive and online network intrusion detection system using clustering and Extreme Learning Machines. *Journal of the Franklin Institute,355*(4), 1752-1779., doi: 10.1016/j.jfranklin.2017.06.006

Sabar, N. R., Yi, X., & Song, A. (2018). A Bi-objective Hyper-Heuristic Support Vector Machines for Big Data Cyber-Security. *IEEE Access,6*, 10421-10431., doi: 10.1109/access.2018.2801792

Sahu, S., & Mehtre, B. M. (2015). Network intrusion detection system using J48 Decision Tree. *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Kochi, 2015, pp. 2023-2026., doi: 10.1109/icacci.2015.7275914

Shah, S. A., & Issac, B. (2018). Performance comparison of intrusion detection systems and application of machine learning to Snort system. *Future Generation Computer Systems,80*, 157-170., doi: 10.1016/j.future.2017.10.016

Sheen, S., & Rajesh, R. (2008). Network intrusion detection using feature selection and Decision tree classifier. *TENCON 2008 - 2008 IEEE Region 10 Conference*, Hyderabad, 2008, pp. 1-4., doi: 10.1109/tencon.2008.4766847

Shenfield, A., Day, D., & Ayesh, A. (2018). Intelligent intrusion detection systems using artificial neural networks. *ICT Express,4*(2), 95-99., doi: 10.1016/j.icte.2018.04.003

Smola, J. A., & Vishwanathan, S. V. N. (2008). *Introduction to machine learning*. Cambridge university press

Stimpson, A. J., & Cummings, M. L. (2014). Assessing Intervention Timing in Computer-Based Education Using Machine Learning Algorithms. *IEEE Access,2*, 78-87., doi: 10.1109/access.2014.2303071

Stolfo, S., Fan, W., Lee, W., Prodromidis, A., & Chan, P. (2000). Cost-based modeling for fraud and intrusion detection: Results from the JAM project. *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX00*,02, Hilton Head, SC, 2000, pp. 130-144., doi: 10.1109/discex.2000.821515

Subba, B., Biswas, S., & Karmakar, S. (2015). Intrusion Detection Systems using Linear Discriminant Analysis and Logistic Regression. *2015 Annual IEEE India Conference (INDICON)*, New Delhi, 2015, pp. 1-6., doi: 10.1109/indicon.2015.7443533

Sun, C., Hahn, A., & Liu, C. (2018). Cyber security of a power grid: State-of-the-art. *International Journal of Electrical Power & Energy Systems,99*, 45-56., doi: 10.1016/j.ijepes.2017.12.020

Syam A. R., & Venkata R. K. (2017). Intrusion Detection System using AI and Machine Learning Algorithm. *International Research Journal of Engineering and Technology (IRJET),04*(12), 1709-1715

Tanha, J. (2018). MSSBoost: A new multiclass boosting to semi-supervised learning. *Neurocomputing,314*, 251-266., doi: 10.1016/j.neucom.2018.06.047

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications,* pp. 1-6., doi: 10.1109/cisda.2009.5356528

Tesfahun, A., & Bhaskari, D. L. (2013). Intrusion Detection Using Random Forests Classifier with SMOTE and Feature Reduction. *2013 International Conference on Cloud & Ubiquitous Computing & Emerging Technologies*, Pune, 2013, pp. 127-132., doi: 10.1109/cube.2013.31

Trabelsi, A., Elouedi, Z., & Lefevre, E. (2018). Decision tree classifiers for evidential attribute values and class labels. *Fuzzy Sets and Systems.*, doi: 10.1016/j.fss.2018.11.006

Ullah, I., & Mahmoud, Q. H. (2017). A filter-based feature selection model for anomaly-based intrusion detection systems. *2017 IEEE International Conference on Big Data (Big Data)*, Boston, MA, 2017, pp. 2151-2159., doi: 10.1109/bigdata.2017.8258163

Vapnik, V. N. (1998). *Statistical learning theory*. New York: J. Wiley.

Varuna, S., & Natesan, P. (2015). An integration of k-means clustering and naïve bayes classifier for Intrusion Detection. *2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN)*, Chennai, 2015, pp. 1-5., doi: 10.1109/icscn.2015.7219835

Verma, A., & Ranga, V. (2018). Statistical analysis of CIDDS-001 dataset for Network Intrusion Detection Systems using Distance-based Machine Learning. *Procedia Computer Science,125*, 709-716., doi: 10.1016/j.procs.2017.12.091

Vigneswaran, K. R., Vinayakumar, R., Soman, K., & Poornachandran, P. (2018). Evaluating Shallow and Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security. *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Bangalore, 2018, pp. 1-6., doi: 10.1109/icccnt.2018.8494096

Vijayanand, R., Devaraj, D., & Kannapiran, B. (2018). Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection. *Computers & Security,77*, 304-314., doi: 10.1016/j.cose.2018.04.010

Wagner, D., & Soto, P. (2002). Mimicry attacks on host-based intrusion detection systems. *Proceedings of the 9th ACM Conference on Computer and Communications Security - CCS 02*, Washington, DC, 2002, pp. 18-22., doi: 10.1145/586110.586145

Wang, C., Xu, R., Lee, S., & Lee, C. (2018). Network intrusion detection using equality constrained-optimization-based extreme learning machines. *Knowledge-Based Systems,147*, 68-80., doi: 10.1016/j.knosys.2018.02.015

Wang, J., Yang, Q., & Ren, D. (2009). An Intrusion Detection Algorithm Based on Decision Tree Technology. *2009 Asia-Pacific Conference on Information Processing*,2, Shenzhen, 2009, pp. 333-335., doi: 10.1109/apcip.2009.218

Wang, L., Li, Q., Yu, Y., & Liu, J. (2018). Region compatibility based stability assessment for decision trees. *Expert Systems with Applications,105*, 112-128. doi: 10.1016/j.eswa.2018.03.036

Wang, X., Zhang, C., & Zheng, K. (2016). Intrusion detection algorithm based on density, cluster centers, and nearest neighbors. *China Communications,13*(7), 24-31., doi: 10.1109/cc.2016.7559072

Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y., Zhu, H., . . . Wang, C. (2018). Machine Learning and Deep Learning Methods for Cybersecurity. *IEEE Access,6*, 35365-35381., doi: 10.1109/ACCESS.2018.2836950

Yadahalli, S., & Nighot, M. K. (2017). Adaboost based parameterized methods for wireless sensor networks. *2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon)*, Bangalore, 2017, pp. 17-19., doi: 10.1109/smarttechcon.2017.8358590

Zhang, D., Han, X., & Deng, C. (2018). Review on the research and practice of deep learning and reinforcement learning in smart grids. *CSEE Journal of Power and Energy Systems,4*(3), 362-370., doi: 10.17775/cseejpes.2018.00520

Zhang, J., & Zulkernine, M. (2006). A hybrid network intrusion detection technique using random forests. *First International Conference on Availability, Reliability and Security (ARES06)*,8, Vienna, Austria, 2006, pp. 269., doi: 10.1109/ares.2006.7

Zhang, Z., Jia, L., Zhao, M., Ye, Q., Zhang, M., & Wang, M. (2018). Adaptive non-negative projective semi-supervised learning for inductive classification. *Neural Networks,108*, 128-145., doi: 10.1016/j.neunet.2018.07.017

Zhou, X., Yi, Y., & Luo, D. (2013). Improved Incremental Support Vector Machine with Hybrid Feature Selection for Network Intrusion Detection. *2013 International Conference on Information and Network Security (ICINS 2013)*, Beijing, 2013., doi:10.1049/cp.2013.2450