Machine Learning Based Intrusion Detection System for Cyberattacks on Corporate Networks

by Srikanth Ambatipudi

B.A. in Mathematics & Statistics, June 1995, Osmania University
Post Graduate Diploma in Business Management, June 2015, Osmania University
Master of Engineering (Cybersecurity), October 2019, George Washington University

A Praxis submitted to

The Faculty of
The School of Engineering and Applied Science
of The George Washington University
in partial fulfillment of the requirements
for the degree of Doctor of Engineering

January 10, 2025

Praxis directed by

Arthiraiyan Appulingam,
Professorial Lecturer of Engineering Management and Systems Engineering

The School of Engineering and Applied Science of The George Washington University certifies that Srikanth Venkata Ambatipudi has passed the Final Examination for the degree of Doctor of Engineering as of December 3, 2024. This is the final and approved form of the Praxis.

Machine Learning based Intrusion Detection System for Cyber Attacks on Corporate Networks

Srikanth Ambatipudi

Praxis Research Committee:

Arthiraiyan Appulingam, Professorial Lecturer in Engineering and Applied Science, Praxis Director

Amir Etemadi, Associate Professor of Online Engineering Programs, Praxis Chair

Joseph Blackford, Associate Professor of Engineering and Applied Science, Committee Member

© Copyright 2024 by Srikanth Ambatipudi
All rights reserved.

Dedication

The author dedicates this research to his parents, the late Rajeswaram Ambatipudi and Rajyalakshmi Ambatipudi, and to his in-laws, the late Kesava Rao Nanduri and the late Sitadevi Nanduri, who have always provided tremendous support and strived for the well-being and education of their children. They have worked hard their entire lives to provide quality life and a learning environment.

The author also dedicates it to his incredible wife, Krishnapriya, and two lovely daughters, Sripriya and Sriya, for their continuous love, affection, support, and encouragement for his success.

Acknowledgements

The author wants to thank Dr. Arthiraiyan Appulingam for great support and mentorship, and for pointing him in the right direction throughout the research phase of the Doctor of Engineering program. He also thanks all the other professors who taught him during the classroom phase of the program, for providing him with incredible knowledge and equipping him with the needed skills to progress through this research.

Abstract of Praxis

Machine Learning Based Intrusion Detection System for Cyberattacks on Corporate Networks

Business organizations have benefited significantly from technology explosion that resulted in a rapid expansion of their computer network infrastructure. This led to the introduction of new vulnerabilities that could be exploited by cyber attackers to gain unauthorized entry into corporate networks resulting in data theft and other related threats. These threats include, but are not limited to, ransomware, credential misuse, and data breaches. All these attacks are significant in terms of the resources required for rectification efforts, and the resulting reputational damage caused to organizations. Cyberattacks target corporate networks, which maintain their customers' personally identifiable information (PII) for day-to-day operations, thus causing data theft, reputation, and financial losses. This praxis research provided a machine learning (ML)-based intrusion detection system (IDS) to detect and classify a potential cyberattack, based on network traffic analysis. The IDS can be implemented on a server that monitors the network traffic. This praxis research used a portion from the network traffic data set that is over 16.5 GB in size with over nine (9) million records of simulated network traffic that includes Packet size, Source and Destination IP (Internet Protocol) Addresses. Random Forest, eXtreme Gradient Boosting, k-Nearest Neighbor were the primary ML models studied.

Table of Contents

Dedication ..... **iv**

Acknowledgements ..... **v**

Abstract of Praxis ..... **vi**

List of Figures ..... **x**

List of Tables ..... **xii**

List of Equations ..... **xiii**

List of Symbols ..... **xiv**

List of Acronyms ..... **xv**

Glossary of Terms ..... **xvii**

Chapter 1: Introduction **1**

1.1 Background 1

1.2 Research Motivation 5

1.3 Problem Statement 6

1.4 Thesis Statement 6

1.5 Research Objectives 6

1.6 Research Questions and Hypotheses 7

1.7 Scope of Research 7

1.8 Research Limitations 8

1.9 Organization of Praxis 8

Chapter 2: Literature Review **10**

2.1 Introduction 10

2.2 Cyber-Attacks on Corporate Networks 10

2.3 Common Types of Attack Attempts 15

2.4 Types of Intrusion Detection Systems (IDS) 17
2.5 Machine Learning based Intrusion Detection System 20
2.6 Summary of Literature Review 26
Chapter 3: Methodology 28
3.1 Introduction 28
3.2 Machine Learning Models 31
3.2.1 Random Forest (RF) 31
3.2.2 eXtreme Gradient Boosting (XGBoost) 31
3.2.3 k-Nearest Neighbor (KNN) 32
3.3 Data Description 34
3.4 Data Acquisition 36
3.4.1 Data Structure 37
3.5 Exploratory Data Analysis 37
3.6 Feature Engineering 38
3.7 Data Preprocessing 38
3.7.1 Data Filtering and Cleaning 38
3.7.2 Data Encoding 39
3.7.3 Data Standardization 40
3.7.4 Data Balancing and Splitting 41
3.7.5 Feature Selection 41
3.7.6 Hyperparameter Tuning 43
3.8 Model Evaluation Metrics 44
3.8.1 Confusion Matrix 45
3.9 Summary of Methodology 47
Chapter 4: Results 48

4.1 Introduction 48
4.2 Exploratory Data Analysis (EDA) 49
4.3 Data Preprocessing 61
4.3.1 Categorical Data Encoding 61
4.3.2 Data Standardization 62
4.3.3 Data Balancing 64
4.3.4 Feature Selection 65
4.3.5 Training of RF, XGBoost, and KNN ML Models 69
4.4 Machine Learning Model Development 72
4.4.1 Machine Learning Model Hyperparameter Tuning 72
4.4.2 Performance Testing 74
4.4.3 Machine Learning Model Performance Testing Results 80
Chapter 5: Discussion and Conclusions 83
5.1 Discussion 83
5.1.1 Hypothesis 1 83
5.1.2 Hypothesis 2 84
5.1.3 Hypothesis 3 85
5.2 Conclusion 86
5.3 Research Contributions 87
5.4 Recommendations for Future Work 87
References 89

List of Figures

Figure 1 *Cost of Cybercrime in U.S. Dollars* 14

Figure 2 *Supervised Machine Learning Process* 22

Figure 3 *Random Forest Algorithm Process* 25

Figure 4 *Overview of the Research Methodology* 30

Figure 5 *XGBoost Process* 32

Figure 6 *Classification with KNN Algorithm* 34

Figure 7 *A 2 X 2 Confusion Matrix* 45

Figure 8 *Distribution of flow of Bytes* 54

Figure 9 *Distribution of flow of Packets* 61

Figure 10 *Summary Correlation Heatmap with all Data set Features Included* 57

Figure 11 *Summary Correlation Heatmap with Non-Relevant Features Excluded* 58

Figure 12 *Correlation Heatmaps for the Data set Features* 59

Figure 13 *Distribution of Benign and Infiltration Traffic for the Training Data Set* 60

Figure 14 *Distribution of Benign and Infiltration Traffic for the Test Data Set* 61

Figure 15 *Feature Distribution Changes after Z-Score Standardization – IDS* 63

Figure 16 *Distribution of Label Feature Before Balancing* 64

Figure 17 *Distribution of Label Feature After Balancing* 65

Figure 18 *XGBoost Feature Importance Scores Bar Chart for Data set Features* 66

Figure 19 *RF Feature Importance Scores Bar Chart for Data set Features* 67

Figure 20 *KNN Feature Importance Scores Bar Chart for Data set Features* 67

Figure 21 *Feature Importance Rank Comparison between XGBoost, RF and KNN* 68

Figure 22 *Accuracy Comparison plotted against Number of Features* 69

Figure 23 *Confusion Matrix for the Trained RF Model* 70

Figure 24 *Confusion Matrix for the Trained XGBoost Model* 71

Figure 25 *Confusion Matrix for Trained KNN Model* 72

Figure 26 *AUC-ROC curves for KNN, RF, and XGBoost* 75

Figure 27 *Confusion Matrix Obtained from Tested RF Model* 76

Figure 28 *Confusion Matrix Obtained from Tested XGBoost Model* 77

Figure 29 *Confusion Matrix Obtained from Tested KNN Model* 78

Figure 30 *Confusion Matrix for XGBoost with Multiclass Data set* 80

Figure 31 *Bar Chart of Performance Test Results of the ML Models* 81

List of Tables

Table 1 *Top 5 Cyberattacks in 2023* 13

Table 2 *Summary Statistics for the Data set Features* 49

Table 3 *Summary Statistics After Removing infinity values* 51

Table 4 *Data Encoding Map for Categorical Attack Type* 62

Table 5 *Hyperparameter Tuning Results for RF, XGBoost, and KNN* 73

Table 6 *Data Encoding Map for Multi-Class Label Feature* 79

Table 7 *XGBoost ML Model Classification Times (In Seconds)* 81

Table 8 *Summary of Hypotheses Testing* 86

List of Equations

Equation 1 *Formula for Accuracy*.....46

Equation 2 *Formula for Precision* 46

Equation 3 *Formula for Recall* 46

Equation 4 *Formula for F1 Score* 46

Equation 5 *Formula for False Positive Rate* 47

Equation 6 *Formula for AUC-ROC Value* 47

Equation 7 *Formula for Average Classification Time* 47

Equation 8 *Formula for Z-Score Calculation*.....65

List of Symbols

X	A single raw data value
μ	Population Mean
σ	Population Standard Deviation

List of Acronyms

AUC-ROC	Area Under the Curve of the Receiver Operator Characteristics
CIA	Confidentiality, Integrity, and Availability
CSV	Comma-Separated Values
DoS	Denial of Service
DDoS	Distributed Denial of Service
DT	Decision Trees
EDA	Exploratory Data Analysis
FN	Number of False Negative instances
FP	Number of False Positive instances
FPR	False Positive Rate
FNR	False Negative Rate
HIDS	Host-based IDS
IDS	Intrusion Detection System
ICMPv6	Internet Control Message Protocol for IPv6
IPv6	Internet Protocol version 6
IMF	International Monetary Fund
KNN	k-Nearest Neighbors
LAN	Local Area Network
ML	Machine Learning
NIDS	Network-based IDS
NIST	National Institute of Standards and Technology
OS	Operating System

RF	Random Forest
SIDS	Signature-based IDS
SMOTE	Synthetic Minority Oversampling Technique
TN	Number of True Negative instances
TP	Number of True Positive instances
TPR	True Positive Rate
TNR	True Negative Rate
XGBoost	eXtreme Gradient Boosting

Glossary of Terms

Critical Infrastructure	It comprises of 16 sectors whose assets, systems, and networks, are considered vital to the functioning and economy of the United States (CISA, 2023).
Cyber Resilience	It is defined as the ability of an organization to continue operating despite attacks on their computer systems (NIST, 2023).
Machine Learning	Computer systems that are designed to self-learn and predict based on data provided (CISA, 2023)
Threat	A potential to exploit the vulnerability (CISA, 2023)
Threat Landscape	Collection of threats in a domain with information on vulnerable assets (CISA, 2023)
Vulnerability	A weakness within the system (CISA, 2023).
CICFlowMeter	Network traffic flow generator and analyzer also known as ISCXFlowMeter. (https://www.unb.ca/cic/research/applications.html).

Chapter 1: Introduction

1.1 Background

The Internet and related technologies, such as cloud computing, have enabled businesses to expand rapidly and operate more efficiently. This has put organizations that are part of the U.S. national critical infrastructure on the radar of cyber attackers. Organizations from every industry sector are progressively adopting the use of latest digital technologies for managing their key business processes, which store and process substantial amounts of sensitive data of their customers, making them remunerative for cyber attackers. The Internet security threats, namely, network attacks and ransomware attacks, have increased manyfold. These attacks impact business operations, as well as causing reputational damage that cannot be accurately measured in terms of lost business.

Computer networks that belong to the government, large corporations, banking, financial services, and insurance sectors are the prime targets for cybercrime groups. The substantial amounts of personal and financial information that they store, and process daily, make them an attractive target (Rawass, 2019). Connections with rich financial investment institutions, debt issuers, and capital fundraising firms make them more lucrative and vulnerable to cyberattacks. Due to the nature of their operations, these companies cannot afford to have any downtime, which makes them ideal candidates for cyberattacks. Organizations have been shifting to new innovative methods to analyze their data for introducing new products, sustaining competition, and growing their business. Modern technology brings in additional vulnerabilities adding up to the existing ones, as most of the large corporations use legacy platforms, such as mainframe computers for their core information systems. The most common cyber threats facing

these companies are: (1) ransomware; (2) compromise/theft of personally identifiable information (PII), and trade secrets; (3) state-sponsored attacks, such as Advanced Persistent Threat (APT); (4) COVID-19 related exploits, and (5) hacktivists who oppose or support a government or corporate with the use of computer technology (CrowdStrike., 2024).

An APT is a sophisticated and targeted cyberattack on the computer network of a large corporation. The attackers behind an APT are often well-funded nation-state threat actors. It starts with system infiltration and is typically executed over an extended period which sometimes lasts more than 6 months. During the infiltration phase, the attackers start by adopting the organization's policies and procedures which helps them evade security measures designed to defend the company against cyberattacks. The COVID-19 pandemic opened novel methods and opportunities for cyber attackers to target insurance and healthcare organizations, which are the two most vulnerable industry sectors to commit fraud and identity theft. Hacktivists are ideologically motivated cybercriminals. They perform cyberattacks on companies that are easily susceptible and fulfill their political or economic goals.

The very first virus program was created to measure the efficiency of a computer system. However, it has evolved into a more sophisticated threat called ransomware attack. Ransomware attacks have become a significant cyberattack technique targeting individuals and enterprises (Chakkaravarthy et al., 2020). During a ransomware attack, the data gets encrypted and access to the data is denied until the demanded amount is paid as ransom. Ransomware is categorized into two primary types: locker-ransomware, which blocks users' interactions, and crypto-ransomware, which encrypts users' data (Su

et al., 2018). Despite a 21% year-over-year global decrease in ransomware during the year 2022, there was a jump across geographies in 2023. In Europe, the number of ransomware attacks jumped by 83%, including a 112% increase in the U.K. Targets on the education and finance industries in the U.S. have increased by 275% and 41%, respectively (Sonicwall, 2023).

In a corporate network, the most important asset is data and there should be security assurance that data critical for ongoing business operations is kept safe from third-party intrusions and attacks. Government and corporate networks are primary targets for cyber attackers as they contain proprietary information, such as trade secrets, financial information, and PII. There are a variety of cyber threats that attempt to exploit system weaknesses and launch their attacks.

Cyberattacks are the biggest emerging risk that corporate computer networks have been facing, given their vulnerability to data exfiltration attempts on their vast amounts of PII. Corporate computer networks today face the downside of technological advancements in the form of technology-aided theft, fraudulent wire transfers, and loss of sensitive customer data.

The traditional protection strategies against cyberattacks involve deploying firewalls on the network perimeter and holding the data within a trusted network. As a result of remote work arrangements, securing the network perimeter has become an obsolete solution as intruders can easily bypass these measures and infect network devices and endpoints. The increase in remote work and communication during the pandemic period due to corona virus disease (COVID-19), has enabled more customers and employees to access corporate networks remotely and use information services. This new method of

working has also opened the door for malicious attacks and intrusions. Organizations must evolve their security strategy and solutions to counter this latest threat and implement new defenses that not only secure the gateway of the corporate network but also secure the network devices, servers, endpoints, and, most importantly, their data. An Intrusion Detection System (IDS) ensures a network's security from various cyberattacks and related threats. An IDS monitors network traffic for anomalies and matches them against a known list of suspicious activities, and threats, and alerts the information security personnel as soon as any such activities are discovered in the corporate network.

The current network security monitoring with the use of IDS along with antivirus software requires a significant amount of time to process the network traffic and hence fails in detecting cyberattacks. Machine-learning (ML)-based IDSs are better suited for the task because they are scientific, fast, scalable, and detect zero-day threats as they learn from data with the use of sophisticated statistical algorithms. Network and host-based IDSs are a good protection strategy to actively monitor network traffic without taking any protective action but fire an alert to the responsible personnel who will implement necessary actions needed for mitigation.

Predictive models are useful in implementing protection strategies that perform better than other models. They are suitable for and work better in static systems and are difficult for ransomware attacks to overcome due to the dynamic behavior aided by obfuscation strategies, such as polymorphism and metamorphism (Gazzan & Sheldon, 2023). Polymorphism refers to the altering of the malware code or behavior to avoid being detected and metamorphism is the process of rewriting the entire malware code after each iteration to evade detection. To counterattack these threats, security researchers have

proposed static and dynamic analysis techniques. Cyberattacks that use obfuscation techniques such as polymorphism and metamorphism can avoid being detected by static analysis. Dynamic analysis employs monitoring and analysis of suspicious processes to identify behavior that matches different formats of cyberattacks (Venturini et al., 2023).

By using the tremendous technological advancements to their benefit, cyber attackers are ahead of the security professionals in the race to steal organizational data. To ensure that organizations reap the benefits of these advancements in newer technologies, ML-based IDS serves security professionals to achieve an edge. The ML-based IDS can sort network traffic as malicious or benign and help identify intrusion attacks and attempts aimed at data theft. This would afford an efficient and early detection of intrusion attacks, enabling cybersecurity personnel to resort to the best possible mitigation methods by using the available infrastructure within the corporate network (Vinayakumar et al., 2019).

This praxis focuses on comparing reliable ML-based classification algorithms namely, extreme Gradient Boosting (XGBoost), Random Forest (RF), and K-Nearest Neighbor (KNN) to find out the best-suited algorithm based on metrics like accuracy, precision, recall, F1 score of the model. The model with the best metrics will predict network intrusion attack attempts accurately.

1.2 Research Motivation

Corporate computer networks have become prime targets of cyberattacks on their customers' PII data, which they store on their application systems. Some companies have the financial capability to pay massive amounts in ransom should the data be held hostage due to a ransomware-type attack. Insurance companies are also affected when they are

required to pay ransom amounts as part of the cybersecurity liability insurance coverage that they provide to these organizations. Small and medium-sized enterprises, which provide various information technology (IT) and non-IT third-party services are also impacted due to cyberattacks. This research focused on using ML in offline mode and may not provide insights based on network traffic analysis in real-time.

1.3 Problem Statement

Corporate networks are prime targets of cyberattacks resulting in an average of $1.85 million spent on rectification efforts per attack (Han et al., 2019). Cyberattack impacts include theft or permanent loss of sensitive information, disruption to operations, cost to restore IT services, and damage to reputation. There is a cascading effect on third-party organizations, as 78% of attacks target the supply chain of businesses (Han et al., 2019).

1.4 Thesis Statement

A model based on machine learning algorithms performing analytics will detect cyberattacks on corporate networks.

1.5 Research Objectives

The aim of this research was to build an ML-based IDS to classify and identify potential cyberattacks on corporate networks by accomplishing the following objectives:

1. Obtain network traffic data set and identify feature names and data types.
2. Perform data pre-processing to locate and treat missing values, identify correlation coefficient between different data set features and find out the distribution of benign and infiltration traffic within the data set.

3. Identify data set features that can significantly impact ML model's performance and build the model.
4. Using the network traffic data set obtained, measure and compare performance of each of the ML model built using the algorithms, namely: RF, XGBoost, and KNN.

1.6 Research Questions and Hypotheses

To aid the development of an ML-based IDS to identify and classify potential cyberattack attempts, the following research questions and hypotheses were proposed:

RQ1: Can a machine learning model accurately detect and classify a cyberattack on a corporate network?

RQ2: Does an ML model trained using the top seven features of the data set detect cyberattacks with an accuracy of at least 80%?

RQ3: Which of the ML algorithms RF, KNN, or XGBoost can most accurately detect a potential cyberattack on a corporate network?

H1: A machine-learning-based model can accurately detect and classify cyberattack attempts on the corporate network.

H2: A machine learning model can be built to detect a potential cyberattack solely based on the top seven features with an accuracy of at least 80%.

H3: An ML-based model developed using XGBoost will most accurately detect cyberattacks on a corporate network.

1.7 Scope of Research

The objective of this study is to develop a model based on ML algorithms, which performs evolutionary-based analytics on historical network traffic data. The chosen

methodology was based on statistical data analysis using models based on machine learning algorithms. The model analyzes network traffic data to deliver insights to be used by the information security team to identify and act upon potential cyberattack attempts.

1.8 Research Limitations

The following are the limitations of this praxis research:

1. The performance and efficiency of an ML-based model is heavily dependent on the diversity and quality of the data set used. Due to non-availability of real-time data, synthetically generated and publicly available network traffic data set was used.
2. ML models based on three algorithms namely: RF, KNN, and XGBoost were assessed using the Python programming language. Use of these three specific algorithms might produce results different from models that use other algorithms.

1.9 Organization of Praxis

This praxis is divided into five chapters with Chapter 1 providing introduction and includes details of the problem and thesis statements, research motivation, research objectives, research questions, research hypotheses, scope, and limitations of research. Chapter 2 elaborates on the literature that was already published on ML-based IDS including the methodology adopted by those researchers.

Chapter 3 describes the proposed methodology to build the ML-based IDS for detecting cyberattacks on corporate networks and includes details of all the activities for preparing the data set used.

Chapter 4 discusses the study's results, based on the previously discussed methodology in Chapter 3. Chapter 5 summarizes the research by describing research contribution and pointers for further research.

Chapter 2: Literature Review

2.1 Introduction

Significant amount of research has been performed on developing intrusion detection systems which are designed to detect atypical traffic and anomalous behavior of network devices (Yeruva et al., 2022). However, present day IDS still experience challenges in terms of high false positive rates which leads to alert fatigue among the security professionals (Liu & Lang, 2019). Although some IDS models have achieved high accuracy rates, they have also resulted in higher false negative rates which indicates lower model efficiency and weak network security (Mijakovic & Spognardi, 2022). Chapter 2 is divided into six sections offering context of the problem, outlining the background of the issue being addressed, along with a thorough overview of the proposed solutions. Section 2.1 contains the introduction for this chapter. Section 2.2 describes cyberattacks on corporate networks. Section 2.3 informs us about the usual categories of attack attempts on corporate networks. Section 2.4 explains the various categories of intrusion detection systems (IDS), how each of them works, and the strengths and weaknesses of each type of system. Section 2.5 describes the development of a machine learning (ML)-based IDS for cyberattacks. Finally, Section 2.6 summarizes the review of literature performed for the area and emphasizes the gap which called for the development of an ML-based IDS.

2.2 Cyber-Attacks on Corporate Networks

Cyberattacks have been spreading rapidly in recent years. They have the potential to cause huge financial losses for a wide range of targets across many industry sectors, including but not limited to insurance, banking and financial services, defense,

government, healthcare, and other corporations. The attackers have been using sophisticated tools and have tailored their methods to fit known vulnerabilities of the targeted victim organizations. The effects of cyberattacks are sometimes irreversible, causing the targeted organization to become defunct. Ransomware and APT attacks are hard to mitigate without obtaining the help of the intruder, attacker, or original author of the encryption keys. In addition to the rectification costs, organizations suffer loss of revenue due to downtime, data theft, and damage to reputation (Al-Rimy et al., 2018).

The chances for a complete recovery of the data are few in the case of a ransomware attack, even after the ransom amounts are paid. This is due to data being lost because of technical faults in the ransomware itself (Savage et al., 2015). Corporations that store and process copious amounts of sensitive data make it a prime target for all types of cyberattacks. The threat actors include those that rely on remote tactics, using remote encryption to impersonate third parties (Microsoft Digital Defense Report [MDDR], 2023). Cyberattacks are a severe threat to the protection of information assets as they successfully exploit vulnerabilities in the target computer systems and then encrypt or exfiltrate data. Access to the encryption key is denied to the victim to create further damage due to sensitive data exposure or given only after the ransom amounts are paid within a limited time. The attacks target larger organizations rather than individuals or smaller organizations because larger sums can be extracted from large corporations, which prefer to pay the ransom instead of taking a hit on their business due to halted operations.

The growth of crypto currencies has made it easier for attackers to demand and obtain ransom amounts anonymously in the form of bitcoins. The developments in technology,

economics, security, and culture have contributed to the evolution of cyberattacks into sophisticated formats of threats affecting companies worldwide, especially in developed nations with high-tech services sector (Damien et al., 2020). During the three-year period ending July 2023, $4.45 million is the average cost incurred in the form of rectification efforts per every data breach, and an increase of 15% was reported in corporate network security breaches (IBM Security Report, 2023). Present day cyberattacks are evolving, exhibiting a high degree of adaptability in the high-tech environment they are trying to survive in. In case of a ransomware type of cyberattack, there are two main forms: (1) locker ransomware that denies access to data, and (2) crypto ransomware, which prevents access to data (Symantec, 2015).

Both types aim at denying what their targets need for continued business operations. The end-to-end process of digital extortion using ransomware as a means requires months of planning and careful execution during which cybercriminals try to establish all possible paths to achieve ransomware infection. All ransomware attacks adversely affect organizations by disrupting regular operations and creating the need to activate system safeguards and formal responses from leadership. (Symantec, 2015). The disruptions have a bigger monetary impact and include rectification costs, losses due to halted business operations, and damage to reputation (Han et al., 2019).

Previously, attackers had adopted techniques to encrypt enormous amounts of the target organization's data and held the decryption key. Present-day attackers are using novel methods and are threatening to expose data if ransom demands are not met (Chauhan & Kshetri, 2023). Ransomware as a service (RaaS) is a new business technique being used by novice cyber criminals to engage in attacks by deploying RaaS kits. The

RaaS kits are available in the Dark Web for inexperienced attackers to try out attacks on individuals and smaller organizations. Microsoft refers to this as human-operated ransomware (MDDR, 2023). Due to the human element, the attack pattern can vary depending on the target's vulnerabilities, as well as maximizing the returns by repeated data theft until the intruder is identified and evicted.

When ransomware attacks hit large organizations, they can perform "short and distort" activities, resulting in manipulating the stock market by using the access obtained to confidential information. Manipulative tactics include short-selling securities to disrupt operations and cause the stock price to go down. In this way, vast amounts of stock are purchased at these lower prices causing the shareholders to lose millions of dollars (Chauhan & Kshetri, 2023).

Cyberattacks are growing exponentially as companies have been rapidly adopting digital technologies to attract and create better customer relationships and increase their product portfolios. Cyberattacks on corporate networks result in significant losses in the form of fines, legal fees, lawsuits, fraud monitoring costs, and the cost to recover from the cyber incident in addition to the damage on brand and market value due to loss of reputation (Deloitte., 2023). The world economic forum lists the top five (5) cyberattacks in 2023, and the losses incurred (weforum.org, 2024) as shown in Table 1.

Table 1

Top 5 Cyberattacks in 2023

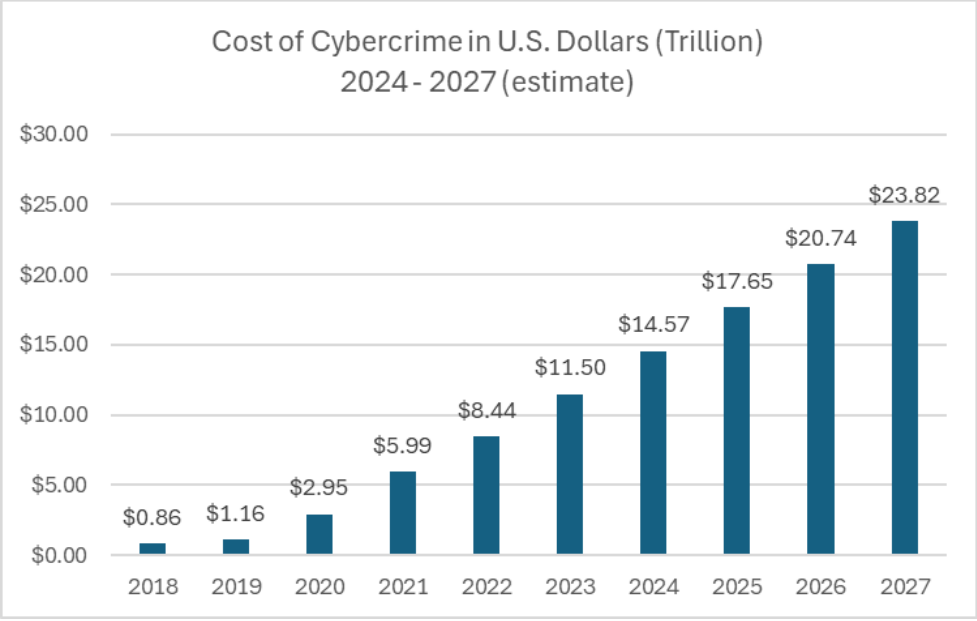
Organization / Entity	Exposure
U.S State Department	> 55,000 email messages stolen from federal government personnel.

Organization / Entity	Exposure
Darkbeam	3.8 Billion records from the digital protection firm
U.K Royal Mail	Ransomware attack demanding $80 million
MOVEit	60 million individuals' data from the file transfer software databases
Indonesia Passport	34 million individual passport records stolen

The World Economic Forum's Centre for Cybersecurity estimates that the cost of cybercrime will reach $23.84 trillion by 2027. Figure 1 below shows the trend of this cost captured in partnership with Statista Technology Market Outlook, FBI, and the IMF between the years 2018 – 2027. The numbers for the years 2024 – 2027 are estimates based on historical data collected between the years 2018 - 2023.

Figure 1

Cost of Cybercrime in U.S. Dollars



The World Economic Forum's Centre for Cybersecurity captured the top ten cyberattacks in 2023 across the world on organizations from multiple business sectors. The cyberattacks impacted organizations from different business sectors like cybersecurity, IT service and software**, c**onstruction and real estate management, healthcare, media, telecom, and manufacturing from various countries like India, Kazakhstan, Hong Kong, Israel, Bulgaria, Italy, UK, and the U.S. The number of records breached ranged between 70 million to 3.8 billion records (weforum.org, 2024).

2.3 Common Types of Attack Attempts

Network security threats fall into four broad categories: (1) unstructured, (2) structured, (3) internal, and (4) external (Wang et al., 2016).

1. **Unstructured threats:** Threats from inexperienced hackers, using tools such as password crackers and shell scripts. The intent is intellectual challenge rather than causing harm.
2. **Structured threats:** Experienced hackers hired by organized crime syndicates and equipped with skills and technology that can reveal system, code, and script vulnerabilities. An advanced persistent threat (APT) is a structured attack attempt that is deliberate, organized, sophisticated, and targeted method executed in multiple phases.
3. **External threats:** Threats that originate from outside the organization and include structured and unstructured threats, as well as nation-state threat actors.
4. **Internal threats:** Threats from within the organization that include disgruntled former or current employees.

With the rapid growth of technology and innovation in Internet connectivity, devices on the corporate network infrastructure are now interconnected, opening numerous avenues for bad actors to intrude into corporate networks. This has created the need for more network-based security and intrusion detection (Wang et al., 2016). Eighty percent of successful cyberattacks originate from unmanned devices (MDDR, 2023).

Neprash et al. (2022) documented 374 cyberattacks on the healthcare industry during five-year period between 2016 and 2021 that exposed the personal health information (PHI) of 41,987,751 individuals. The number of cyberattacks per annum doubled, from 43 to 91 during the same five-year period and significantly increased in the level of sophistication (Neprash et al., 2022).

The systems that attackers have been using to deploy cyberattacks have evolved into a complex, continuously growing infrastructure. The primary attack methods include:

1. Social engineering tactics: this is accomplished via enhanced reconnaissance and helps tailor phishing emails to lure victims and make them more engaged in the attack.
2. The initial attack vectors are usually emails as most users do not have the skills or awareness to identify social engineering tactics.
3. Users are lured into booby-trapped websites after the initial attack or made to download attachments laced with ransomware.
4. A scan is performed on the victim's computer and any vulnerabilities identified are exploited to infect the computer with ransomware.
5. Payload delivery (ransomware) is the next step of the process.

6. Command and control server calls are made to establish a communication path for crypto ransomware encryption keys.

Social engineering combined with technology forms a blended attack and is an essential component of the cyberattack strategy. Social engineering exploits human behavior and emotions. There is a rise in blended attacks using aggressive social engineering and sophisticated technology, from fake antivirus to WannaCry in 2017 (O'Kane et al., 2018). A vast majority of current attacks can be categorized as fileless attacks, which have become prominent for their stealthiness and difficulty to detect. In a fileless attack, programs that were developed for legitimate use, in which users have confidence, are repurposed to run viruses. The security company, CrowdStrike refers to the technique by which legal software, such as the Microsoft Windows PowerShell is used for fileless attack as "living off the land." The Ponemon Institute report sponsored by IBM Security stated that the fileless attacks are 10 times more successful than traditional attack methods (Balabanau, 2021).

There are three main techniques that adversaries have recently been adopting: (1) Living-off-the-land, which leverages existing systems and tools for the campaign, (2) self-propagation malware, where the infected entity attempts to spread locally within the Local Area Network (LAN) or the corporate network as well as globally (Internet), and (3) Hypertext Transfer Protocol (HTTP)-based malware, which uses communication protocols for spreading locally and globally (Ongun, 2023).

2.4 Types of Intrusion Detection Systems (IDS)

An IDS monitors inbound and outbound network traffic to identify security policy breaches and suspicious activity and alerts responsible personnel. An IDS is either a

software or hardware component which acts as the first line of defense. The main objective of an IDS was to decrease the mean time needed to detect whenever network traffic anomalies occur so that appropriate security measures can be put in place. Traditionally, network anomaly detection was achieved by deep packet inspection.

A critical challenge in protecting organizations from cyberattacks is to increase the resiliency capacity. The ability to continue business operations and to provide continuous support with essential functionality even during an ongoing attack is called cyber resiliency. It is one among the diverse levels of resilience that an organization should possess to continue to strive during adverse situations (Sun et al., 2018).

Cyber resilience is the combination of business continuity, information security, and the overall organizational ability to continue business operations and deliver predefined outcomes. It is the preparedness of an organization and its ability to recover from cyberattacks and other adverse situations which halts normal business operations resulting in loss of revenue (NIST, 2023).

A cyber resilient organization succeeds in continuing its business operations despite the adverse effects of a cyber-attack. The goal of a cyber resilient system is to prepare for the worst (anticipate), support essential business functions (withstand), revert to normal operations (recover), and make changes based on lessons learned (adapt).

The cost of building enough resilience depends on the size and type of the IT landscape. The cybersecurity systems needed for an organization’s critical infrastructure are complex, possessing a layered structure. The tools used for protecting the information systems need to be deployed and managed as per the business requirements by assigning security controls based on the risk level for theft and/or exposure (Selifanov et al., 2022).

Maintaining confidentiality, integrity, and availability (CIA) of their data is the primary objective of organizations. The process of identifying attempts to compromise a computer system or network's CIA is called intrusion detection. The goal of an organization's cybersecurity systems is to manage information security by adopting a strategic approach that addresses risks, data breaches and other security incidents, which threaten the CIA of the corporate information resources (Zammani et al., 2021).

An IDS processes network traffic and detects malicious network packets so that they are prevented from reaching their intended destinations on the network (Elrawy et al., 2018). An IDS is a combination of hardware and software applications that observes network activities and analyzes them to identify intrusion attempts and is designed to send alerts to information security personnel when required (Latha & Prakash, 2018). Cyberattackers have become aware of the security measures of their targets and are willing to invest time and effort to perform reconnaissance before moving to other phases of the attack and finally deliver and deploy their payloads. The security tools are identified during the initial phases of the attack and are disabled to avoid detection. Hence, the need to implement an advanced ML-based IDS is inevitable to protect organizational data (Suramwar & Bansode, 2015).

Based on their operating environment, IDS can be classified as host-based or network-based. Based on the detection method, they fall into anomaly-based or signature-based (Khraisat et al., 2019). A host-based IDS (HIDS) collects operating system logs from network devices. Due to limited storage capacity, they might become flooded with these logs as the number of network devices increases. A NIDS will perform better and provide better network protection due to its ability to handle more network data packets

when compared to HIDS (Khan et al., 2020). The NIDSs perform better when compared to HIDS and provide security coverage for all network devices rather than providing security to each host individually. Cloud-based IDS are better than NIDS due to the storage, processing capacity, and scalability that they provide and the ability to manage numerous network devices. The other types of IDS in use are:

- Network Node IDS: A variation of a NIDS that analyzes packets that pass through the node monitoring each node of the network.
- Protocol-based IDS: Monitors protocols in use and notifies if there is a deviation from the approved list of protocols to be allowed.
- Application-based IDS: For software application security associated with a HIDS.

Additionally, the methods used by IDS can be classified into:

- Signature-based: Identifies patterns to match them against known signs of intrusion.
- Anomaly-based: Identifies network traffic that deviates from normal behavior.
- Hybrid: Looks at patterns as well as one-off situations to identify intrusion attempts.

2.5 Machine Learning based Intrusion Detection System

An organization's IT landscape proliferates with numerous network devices and computers, which has led to the drastic increase in network traffic within corporate networks when such devices exchange data internally and externally, there is an increase in Internet traffic for external communication. With the massive increase in network traffic levels, it is difficult for security professionals to detect intrusion attempts, and security engineers and researchers have therefore made attempts to identify a solution.

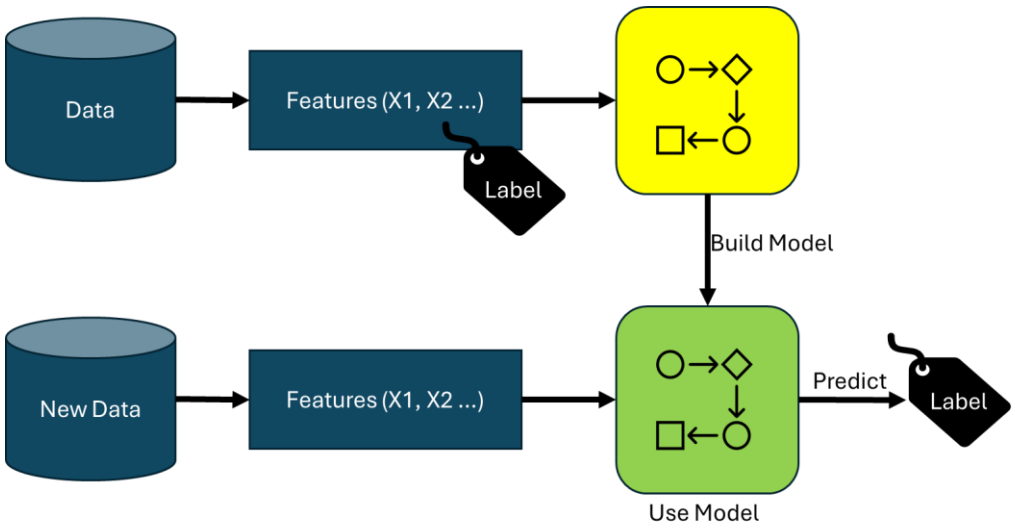
Due to the dynamic nature of network traffic, any security solution designed for intrusion detection needs to be as dynamic. An ML-based IDS can monitor network traffic in real-time, working in conjunction with the firewalls and other perimeter security devices.

The costs incurred by organizations that suffered ransomware attacks have reached the $1 billion mark as early as 2017. This can be attributed to the evolution of new malware versions that behave differently from predecessors by evading traditional static detection methods (Fernando et al., 2020). Detection of malware in Windows systems can be carried out efficiently by using machine learning techniques. Research in the use of ML has shown effectiveness in detection compared to traditional signature-based methods. An ML approach is adaptable in detecting newer ransomware types (Fernando et al., 2020).

Machine learning algorithms are useful in improving the functionality of IDS since they enable it to use historical network traffic and predict the nature of current traffic (Tait et al., 2021). In a supervised ML model, the algorithm utilizes data to train and identify patterns in the data set that includes labels along with the features. Then, the trained model is used on the features of a new data set to predict the labels. Figure 2 below illustrates the high-level process of a supervised learning model.

Figure 2

Supervised Machine Learning Process



Feature engineering and selection allows detection of ransomware using either ML or deep learning by following specific patterns. With the use of algorithms, the optimal set of features is identified. A subset of features is used in the ML model, whereby the model is trained and tested, allowing it to learn by itself. Learning methods are either supervised or unsupervised. Supervised learning requires a training set where training data samples are labeled as benign and malicious, and the ML model differentiates between them. Unsupervised learning methods involve data sets that are not labeled, and the model locates patterns and distinguishes among the data types (Fernando et al., 2020).

The primary challenge in cyberattack detection is the disparity in speed between the traditional methods for detection and the speed at which cyberattacks are evolving. The IDSs that are not ML-based do not have the capacity to evolve and change based on the malware strains. Early detection of cyberattacks is critical. In case of a ransomware attack, once the files are encrypted, it is not possible to decrypt without getting the original decryption key from the author after paying the ransom amount. Early detection

should consider the variety of methods used by ransomware to propagate. Ransomware has a high success rate in evading predictions, as attackers have begun using artificial intelligence (AI)-based techniques and the counter measures must take AI-based attack vectors into consideration (Fernando et al., 2020).

It is not feasible to have one algorithm that is applicable across all the domains (Wolpert & Macready, 1997). A variety of ML-based algorithms must be tested to identify the best one suited for the purpose. For the problem discussed in this praxis, which is intrusion detection or classification of network traffic data, three different classifiers have been selected for testing.

An ML-based IDS can detect a cyberattack in its preliminary stages of execution by following a multilayered approach with minimal system overhead. By following the mutual information criterion technique, discrimination added by each feature to the classifier can be quantified providing the system with a measure of dependency of the features on classification and labeling (Verma, 2020). The feature selection phase of the ML model uses mutual information criteria allowing for the most discriminating features to be selected.

Verma (2020) considered both ensemble and single classifier types of classification algorithms. AdaBoost achieved the highest average value of 97.5% for the specificity metric. Random Forest and XGBoost achieved higher values of sensitivity, with 97.31%. For AUC, XGBoost achieved the best result at 98.77% (Verma, 2020).

Gradient Tree Boosting: a machine learning algorithm that builds the models progressively by starting with a simple basic model, which has a finite amount of residual error. The error gets minimized to an acceptable level by lowering the gradient which

enhances the model by reducing the gap between observed and expected values (Fernando et al., 2020).

Decision Tree: data structure that is shaped like a tree. Training data is used to make predictions on test data by constructing a tree and aims at achieving accuracy with minimum number of trees. A J48 decision tree enables solutions for classification and regression problems and relies on entropy (uncertainty) and information gain. Decision trees predict the label with a series of combination-based questions and evaluate the responses for classification into a category, or regression for a continuous variable (Nopour et al., 2021).

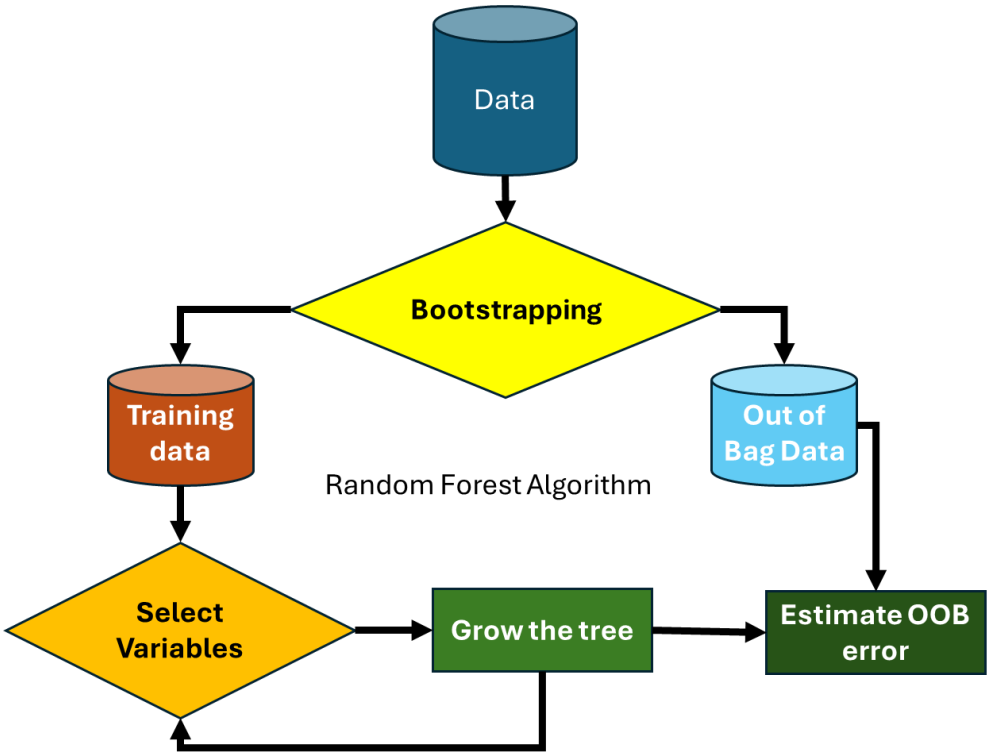
Random Forest is based on the J48 decision trees that requires minimal data preparation. Random forest (RF) produces reliable results by using clusters of decision trees that are grouped randomly (Fernando et al., 2020). RF is the most used ML algorithm that operates as an ensemble ML method. It is based on decision trees, and bagging which is the process of training using trees obtained randomly from a subset of the data. Ensemble learning is an advanced ML technique in which multiple learners are trained to produce the same result. Traditional ML models learn one hypothesis at a time from the available training data, whereas ensemble methods involve a set of hypotheses in combination (Fernando et al., 2020).

The RF algorithm produces a single result by combining output from multiple decision trees. Random Forest contains weak estimators in the form of binary decision trees. In ensemble learning, several weak estimators or learners are trained, and the results are aggregated to create a strong estimator (Zhou, 2009).

RF is a two-step algorithm. During the first step, the data is sampled randomly with replacement using bootstrap technique. In the second step, the algorithm constructs a decision tree to select the most voted prediction result. Bootstrapping helps in measuring the statistical accuracy of the model. The RF keeps growing randomly to reduce variance and the data that remains unselected by the bootstrap sampling technique is called 'out-of-bag' (OOB) data and is used to test the accuracy of the tree. The mean squared error calculated for the OOB data is called the OOB error. Figure 3 illustrates a simple RF algorithm.

Figure 3

Random Forest Algorithm Process



RF is easy to use due to the smaller number of parameters (three) that require tuning by the user, which are: *ntree* (number of trees), *mtry* (number of variables), and *nodesize*

(minimum depth of the tree). Adjusting the *ntree* parameter will decrease variance and does not cause overfitting or increase in bias. The other two parameters *mtry* and *nodesize* do not have an impact on the accuracy value because the RF algorithm internally validates OOB data and minimizes OOB error. Random Forest is popular because of its resistance to overfitting, faster computation, and its ability to uncover nonlinear complex relationships among different variables of the data set (Stevens et al, 2015).

The biggest strength of the XGBoost algorithm is its scalability and speed because of adjustable parameters that can be defined from the available set of standard parameters. XGBoost is widely used by data scientists for achieving predefined desired results because of its optimization capabilities aided by the gradient boosting library. An advanced implementation of the XGBoost algorithm that is based on regularization reduces overfitting, providing better performance. The XGBoost has been widely recognized as a go-to algorithm during ML and data mining challenges because of the low computing power required. This is due to the algorithm's methodology of building a sequence of shallow trees that manage errors that have been produced by previous trees, by making corrections on the residual errors (Sheridan et al., 2016).

2.6 Summary of Literature Review

Chapter 2 has outlined various solutions for ML-based IDS to detect cyberattack traffic within a corporate network. Studies by researchers have contributed knowledge that has led to significant progress on the topic of using ML algorithms. While some of the existing solutions have been tested in real-time, there are many that are theoretical, whose performance has not been tested and documented. Based on the literature review

conducted, the problem described in this praxis will allow the research community to advance and develop ML-based IDS to protect corporate networks from cyberattacks. An ML-based IDS for detecting cyberattacks on companies will protect their customers' and employees' sensitive data, thus saving millions of dollars in rectification efforts.

Chapter 3: Methodology

3.1 Introduction

The purpose of this praxis was to design an ML-based IDS that analyzes network traffic and categorizes them as normal traffic versus intrusion attempts on corporate networks. Random Forest (RF), k-Nearest Neighbors (KNN), and eXtreme Gradient Boosting (XGBoost), were the three (3) ML algorithms that were used to build the models using Python programming language, which is simple to use even for researchers with no computer programming background.

During the testing phase, the performance of the three ML models built was assessed and compared to determine the most effective ML model. The metrics used for the evaluation include accuracy, precision, recall, F1 score, False Positive Rate (FPR), and Area Under the Curve of the Receiver Operator Characteristic (AUC-ROC).

The RF algorithm's output is the aggregate of multiple decision trees that provide a single result. RF is the most frequently used ML algorithm due to its ease of use, flexibility, and the ability to oversee both classification and regression. The KNN algorithm useful for both classification and regression is a nonparametric supervised learner. The XGBoost is scalable and is based on gradient-boosted decision trees (DT). Due to its ability to provide parallel tree boosting, it is the most preferred ML library for solving classification, regression, and ranking.

This chapter details the methodology applied in this praxis for developing ML-based IDS to analyze network data and identify cyberattacks on a corporate network. Exploratory Data Analysis (EDA) was conducted on the entire data set to understand its characteristics. The goal of EDA is to pinpoint features in the dataset that do not provide

value to the model and identify categorical textual features which can only be used in the model after encoding. Summary statistics was the most important deliverable from EDA which helped determine the variedness in scales between different features along with the correlation coefficients between them. Data standardization solves problems due to differences in scales among different features. EDA helps in discovering patterns, relationships, and identifying features that provide insights into the data. In summary, the deliverables from EDA are: (1) feature identification, (2) clean data, (3) encoded categorical data, (4) standardized data, and (5) data split into training and test subsets (Sahoo et al., 2019).

The EDA uncovers specific findings that are sometimes missed by traditional analytics approaches. The primary challenge facing ML-based IDS is its inability to directly use the network traffic data as an input. The variety of data types and value ranges in the data captured from network packets, and the data fields derived from them, makes it an even more complex and challenging task (Maxwell et al., 2019).

Contemporary network environments employ different protocols, interfaces, and a wide variety of organization-specific proprietary controls. Accuracy and speed in detecting cyberattacks with optimal use of resources is the desired state for every organization, enabled by adopting feature engineering for detecting anomalies in corporate network traffic. To enhance predictive capability, feature engineering involves generating new features from existing ones by using transformation functions on existing features to produce new and unique features (Nargesian et al., 2017).

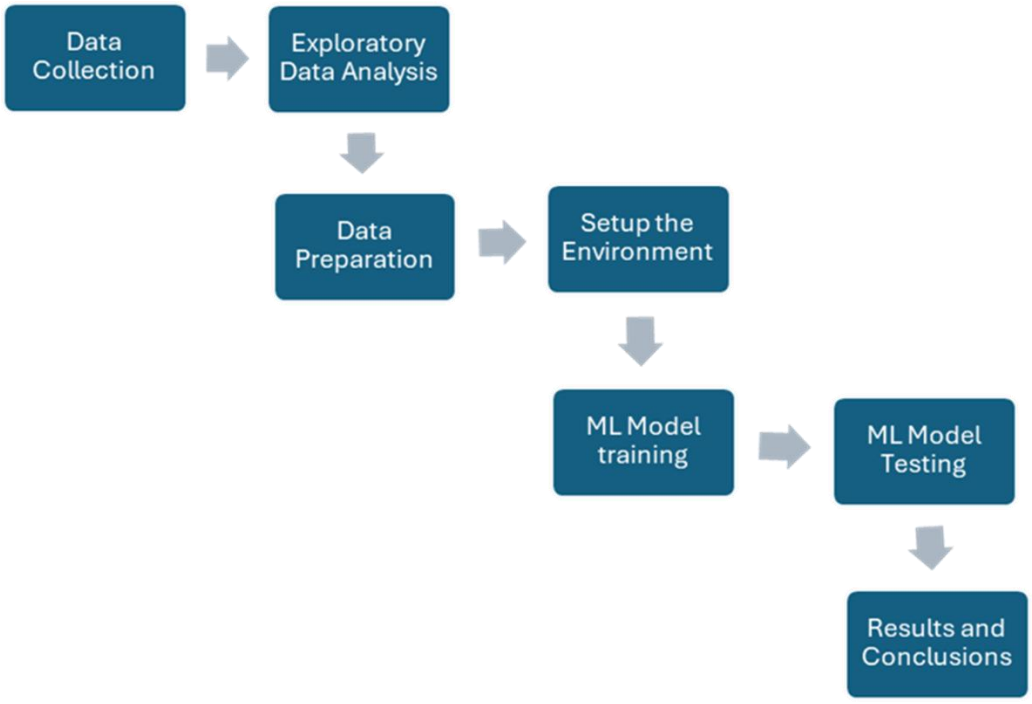
Researchers have been adopting simplistic feature engineering methods for IDS and significant improvements were made in this domain for generating the required features

without compromising quality, and without the risk of losing the features that possess pertinent data qualities (Maxwell et al., 2019).

Feature selection is an essential component of this praxis research to create an efficient ML model that has both speed and accuracy. Figure 4 provides a high-level overview of the praxis methodology.

Figure 4

Overview of the Research Methodology



The training data set was used to train and tune various parameters for accurately recording the model performance. A separate testing data set was used to measure

performance to determine which of the three models is the most efficient, accurate, and best satisfies the goals and objectives of this research.

3.2 Machine Learning Models

During this praxis research, ML-based models using the RF, XGBoost, and KNN algorithms were evaluated.

3.2.1 Random Forest (RF)

ML model based on the RF algorithm works using Decision Trees (DT). The RF algorithm-based ML model works well with large data sets that have high dimensionality (Liyang et al., 2021). RF is best to take advantage of humongous modern data sets while maintaining efficiencies that are statistically required. It combines results from multiple decision trees trained on randomly selected subsets to reduce overfitting the data points and to improve accuracy. RF predictor is a collection of trees and uses 'bootstrap aggregating,' or 'bagging,' a process of training randomly obtained trees from a subset of the data and averaging output to improve accuracy of the estimator (Lee et al., 2019).

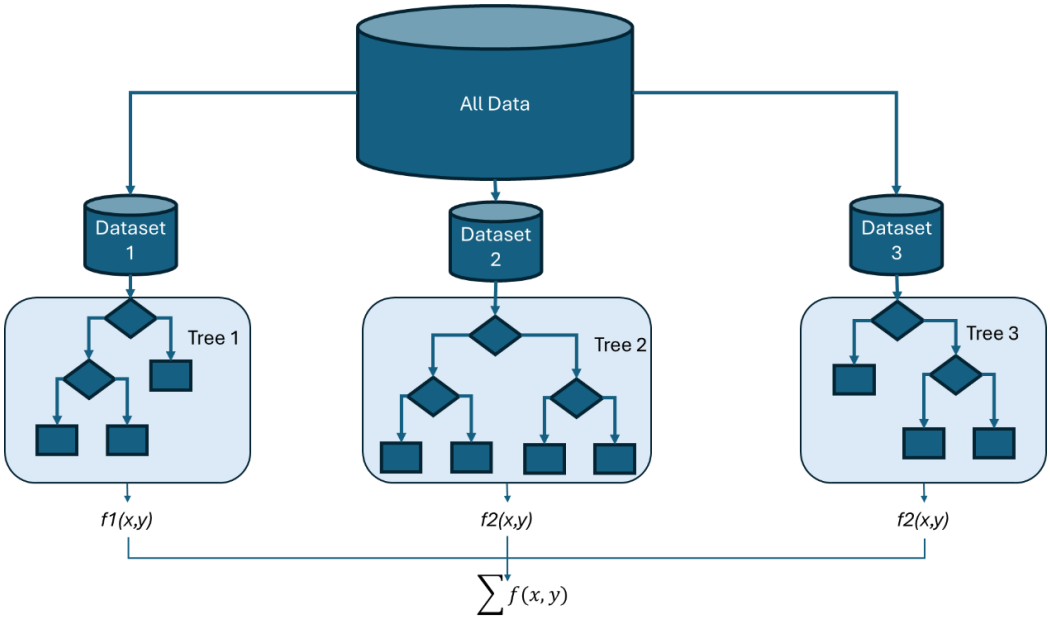
3.2.2 eXtreme Gradient Boosting (XGBoost)

ML model based on the XGBoost algorithm is known for its superior speed, scalability, and accuracy compared to other models. While the RF model requires trees to be generated and run parallelly on GPU clusters for greater efficiency, XGBoost can be run on a single CPU (Sheridan et al., 2016). XGBoost is the most prevalent model used by scientists to address machine learning challenges (Chen & Guestrin, 2016). It enables control over variable weights and helps reduce overfitting by generating trees and with each new tree, correcting errors generated by the predecessor tree (Carmona et al., 2019).

The XGBoost model is capable of summarizing and evaluating large data sets, and the results are easily interpretable providing deeper insights with greater predictive power than conventional models. The XGBoost process is presented in Figure 5.

Figure 5

XGBoost Process



The XGBoost model included a wide range of hyperparameters that allowed for maximum optimization based on the width and depth of the trees along with the ability to handle missing data without compromising the accuracy of the model (Aydin & Ozturk, 2021). The XGBoost process works with a set of multiple ‘weak’ learners combining them into a strong learner (Han et al., 2019).

3.2.3 k-Nearest Neighbor (KNN)

The KNN process performs binary classification and has several advantages along with some limitations due to its susceptibility to high dimensionality and noise in large

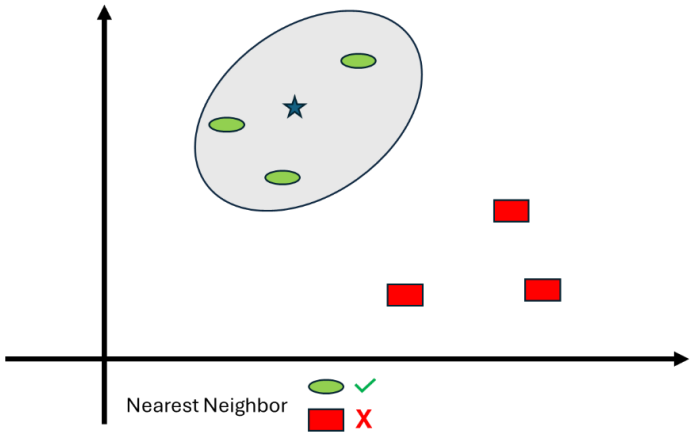
data sets. KNN is a nonparametric classifier that classifies data among its k-nearest neighbors. In an IDS, network traffic is classified into normal or malicious traffic based on knowledge. The classification is based on closeness of the behavior and the KNN model is suitable for IDS because of its accuracy, speed, simple output, and predictive power (Jadhav & Vidyullatha, 2021).

A comparison of KNN, Support Vector Machines (SVM), DT, and Naive Bayes classification algorithms was conducted to predict ICMPv6-based Distributed Denial of Service (DDoS) attacks. It was found that KNN was the fastest compared to SVM, DT, and Naive Bayes algorithms (Suangli et al., 2023).

The ‘k’ in KNN equals number of neighbors, which is a major consideration for the majority voting process. The data points that are in proximity are considered neighbors. Large values of ‘k’ produce smooth decision boundaries but are not compatible for all types of data sets and may lead to overfitting. The other parameters for a KNN classification model are distance, weights, algorithm type, leaf size, power (p), and an additional parameter called ‘*metric_params*’ for passing keyword arguments. ‘*n_jobs*’ is the number of jobs that are run in parallel for searching the neighbors of the datapoint (Zhang, 2016). In Figure 6 below, K=3 and hence the classification algorithm considers the three green circles as neighbors of the blue star.

Figure 6

Classification with KNN Algorithm



3.3 Data Description

This praxis utilized a data set collected by the Canadian Institute for Cybersecurity at the University of New Brunswick (CICUNB) in 2018 to assess new techniques for categorizing network traffic Google LLC (CSE-CIC-IDS2018) (Amazon Web Services [AWS], 2018). The data set had eighty (80) features and the value for each feature is generated by a device on the network. Each record holds crucial information of a packet flow, namely, source and destination IP addresses, ports, and application protocols along with other information. There are features that are numeric, but the data set also contains non-numeric and date type features. The statistics for network packet flow were obtained using *ntopng*, a web-based traffic monitoring application that monitors network traffic by passively capturing network traffic flows. The tool uses deep packet inspection techniques for obtaining this information. Google LLC (CSE-CIC-IDS2018) (Amazon Web Services [AWS], 2018).

The data set has over nine (9) million records of network traffic stored as ten comma separated value (CSV) files. It was originally created in 2018 by CICUNB in collaboration with Communications Security Establishment (CSE). Two (2) files from the ten (10) individual files were split up by CIC which are representative of all data, were used as training, and testing data sets for the ML models investigated in this praxis research. Google LLC (CSE-CIC-IDS2018) (Amazon Web Services [AWS], 2018).

Anomaly detection helps in identifying a variety of novel attacks. However, anomaly detection requires complex systems that need significant amounts of pre-work prior to deployment. The challenge here was the nonavailability of data sets due to two major reasons: (1) The data is internal to the organization and due to privacy concerns, could not be shared, and (2) anonymized data does not represent current trends or lacks key statistical characteristics, making the availability of truly representative data impractical. As a result, researchers often resort to using suboptimal data sets. This results in a suboptimal model since ML models are as good as the data they are trained and tested upon. Hence there is a need to use dynamically generated data sets that are more flexible, which include the current network traffic trends and intrusion attempts improving overall reproducibility (Chakraborty et al., 2023).

It has become imperative to devise a systematic approach to generate data sets that can be used for evaluating IDS. The data set includes a ‘Label’ field that classifies the traffic either as ‘Benign’ or as an ‘Infiltration’ attempt on the network. The infrastructure used for simulating the attacks was created using 50 machines. The target organization was created and consists of 420 systems along with 30 servers deployed among five departments. The 80 features of the data set contained network traffic information and

system logs captured by using CICFlowMeter-V3 tool. Google LLC (CSE-CIC-IDS2018) (Amazon Web Services [AWS], 2018).

During the simulation of an attack to generate the data set, a ‘Kali Linux’ system was used as the attack system and the victim machine was running a ‘Ubuntu 16.4 (Web Server).’ The network traffic was labeled as benign or infiltration attempts which were deployed using tools like FTP, SSH, Nmap, and portscan. Google LLC (CSE-CIC-IDS2018) (Amazon Web Services [AWS], 2018).

The testbed contained Windows and Linux-based workstations that were interconnected. Older versions of the Windows service packs were deployed to make use of diverse vulnerabilities that existed under each service pack. The Linux machines were running Metasploitable, developed for use by penetration testers. The data set was a combination of data from multiple days and included the start time, finish time, and duration of attacks.

The attack traffic originated from the darknet, the unused portion of the Internet. There are no legitimate hosts present in the darknet and any traffic coming from it could be malicious. Analysis of traffic from the darknet provides valuable insights that enable early detection of malicious activity which can be utilized to prevent cyberattacks (Sharafaldin et al., 2018).

3.4 Data Acquisition

The CICUNB has made the data set publicly available on ‘Kaggle,’ which is a web-based data science environment for use by ML practitioners under Google LLC (CSE-CIC-IDS2018) (Amazon Web Services [AWS], 2018). There are 9,332,829 records of network traffic data. The data is divided and is available as ten individual files. A file

containing 331,100 records was acquired and used for training. For testing purposes, another file containing 613,071 records was acquired. Pandas is a powerful library from the Python environment designed for data analysis including manipulation, cleaning, visualization along file input/output for reading and writing CSV, and Excel file formats. The training and test data sets were imported into the Python environment using the '*read'* function of the Pandas library.

3.4.1 Data Structure

Each row of the data set contains eighty (80) features. The last column of the data set is 'Label' which indicates whether the network traffic packets are a cyberattack attempt or if they represent standard traffic by using the value 'benign' for denoting standard traffic, and 'infiltration' to denote atypical traffic. The binary variables in the 'Label' column of the data set are the dependent variables of the data set, which are the output variables that this ML model must detect. The other variables in the data set which are used to predict the dependent variable used as inputs into the ML model are independent.

3.5 Exploratory Data Analysis

Exploratory Data Analysis (EDA) summarizes and provides visualization for gaining more insights into the data. The EDA deliverables like summary statistics, correlation heatmap, count of individual values of the target feature, and distribution of features provide valuable information about the data set. Python programming language is simple to learn and easy to implement because of the ease of access to its tools and libraries for exploratory data analysis of all types of data sets utilized in data science and machine learning activities (Singh et al., 2022). During this praxis, EDA was performed to obtain

information like basic statistics, correlation heatmaps, and relationships among various data set features.

3.6 Feature Engineering

Feature engineering is an ML technique to identify the most noteworthy features that have an impact on the 'Label' feature of the data set. The technique identifies significant features of a data set or transforms existing features to enhance their importance. The process involves transforming raw features into formats that are easily interpretable by an ML model. This praxis research was guided by the three research questions and hypotheses that explored the ML models' usability for classifying network traffic data to identify potential cyberattacks. Feature engineering can be effectively used to measure ML models' performance in cyberattack detection (Yadav & Kalpana, 2019).

3.7 Data Preprocessing

Data preprocessing plays a critical role during the construction of an ML model since training on quality data significantly boosts model performance during the testing phase. It also aids in organizing the data set to ensure that the ML model runs efficiently by focusing on the data set features that have the most significant influence on its performance (Yadav & Kalpana, 2019).

3.7.1 Data Filtering and Cleaning

The explainability and understandability of intrusion detection system (IDS) alerts is essential for making informed decisions based on those alerts. Network traffic data size is ever increasing overwhelmingly challenging the computational power of the IDS and with the introduction of high false positive rates. Data preprocessing activities like

cleaning, filtering, summarization, and visualization are critical in obtaining high quality data from the network traffic data before its use in the ML model (Liu et al., 2021).

The data filtering and cleaning phase which followed the EDA phase focused on treating the data set for efficient model performance by applying methods that address poor-quality data as well as eliminating those that are unnecessary or irrelevant for the ML model. From the network traffic data set used in this praxis, rows that contained no data were either eliminated or replaced. Some of the features that were specific to the record in the data set were 'Dst_Port,' 'Protocol,' and 'Flow_Duration,' denoting features with destination port number, transmission protocol, and flow duration (the time taken for transferring all the data packets), respectively.

The focus of this praxis research was on classifying the network traffic and detecting potential cyberattacks by using the 'Label' feature which indicates whether the traffic is benign or an infiltration attempt on the corporate network.

3.7.2 Data Encoding

Converting data from one form into another for easy storage, processing, and analysis is called encoding. Machine Learning models manage numerical data better, compared to their ability to manage categorical data (Venu et al., 2022).

ML models can only understand categorical variables when represented numerically. In this praxis, the one-hot encoder function was used to convert categorical data into numeric data. When categorical data was converted into numerical data, it was easier for the ML model to process, given the expectation for input data to be in numerical format.

Categorical data is of two major types: nominal and ordinal. Statistical operations cannot be performed on nominal data since they are labels that represent categories. Ordinal data has a well-defined ranking or order. The features which contained categorical data were converted into numerical data for the ML models used in this praxis research to process the data set. One-hot encoder utilizing Python libraries was used to convert categorical text data into numerical values which resulted in a table with a one-one relationship between the nominal data and an assigned numerical value.

3.7.3 Data Standardization

Data standardization refers to the process of converting data into a standard format that is uniform and usable across the board. Data standardization examines whether data is in the required data type and format, any data that lies outside of the acceptable range, and any data uses shortened forms or abbreviations. Python's *'sklearn'* library provides a variety of methods for scaling of which StandardScaler is the most primarily used. StandardScaler is for normally distributed data and suitable with ML models like RF, XGBoost, and KNN being evaluated during this praxis. By using the StandardScaler method for data standardization, ML model metrics like accuracy, precision, recall and F1 score have improved (Raju et al., 2020). The basic statistics obtained during the EDA phase showed that many features from the data set used in the praxis have their means and standard deviations spread across wide ranges. ML models are sensitive to large variations in scales of the feature scales which adds computational complexity and hence do not produce desired results (L'Heureux et al., 2017). Standardization of the data is achieved by using the StandardScaler() function in Python which uses the Z-Score standardization method, by which the standardized value is

obtained by dividing the difference between the raw value and mean with standard deviation (Basu et al., 2023).

3.7.4 Data Balancing and Splitting

Data balancing was achieved by following the Synthetic Minority Oversampling Technique (SMOTE) technique, which is considered the universal standard for learning from imbalanced data. ML models seek to minimize classification errors irrespective of distribution resulting in bias towards majority and reduction in accuracy of the minority. SMOTE adopts oversampling techniques to counter the effects of minority classes in the data set (Elreedy & Atiya, 2019). SMOTE creates instances of minority class by within existing samples by using interpolation techniques without duplicating data points and by creating new samples. Two separate data sets were used for training and testing the model. During the initial run, the training data set was used. Then, for the testing phase a new data set was used. These subsets of data are crucial in developing the ML model as each data set is created for and serves a specific purpose. Using the training data set, the ML model identifies the patterns, and testing data set is useful for testing and scoring the model performance.

3.7.5 Feature Selection

Feature selection is crucial for building an efficient model that is fast and accurate. Shapley values is a concept derived from the cooperative game theory and is concerned with assigning a value to a player based on their participation corresponding to the result of the game. The concept characterized by collection of desirable properties was introduced by Lloyd Shapely in 1953 (Tan et al., 2018). The advantage of deploying Shapley values is that it uses average prediction and has a fair distribution among all the

feature values. Shapley values facilitate contrastive explanations, which enable comparing them with the entire data set, a smaller subset of the data, or even a single datapoint. However, there are limitations associated with using Shapley values due to the large quantity of computing power needed, as well as the susceptibility of the values to minute misinterpretations (Tan et al., 2018).

Feature selection is the method of picking a set or subcategory in a correlated feature for producing a result model, if the data contains redundant and irrelevant information. Feature selection or exclusion should have zero impact on classification process and should not cause any information loss and is an important aspect during preprocessing step for the data set (Bakro et al., 2023).

Feature selection techniques using ML fall into two categories, supervised techniques (for labeled data set), and unsupervised techniques (for unlabeled data set). Feature selection methods in ML are of three types, filter, wrapper, and embedded (Bakro et al., 2023). The filter methods are faster and computationally less expensive. They are based on the inherent properties of the features. Statistic values like Mean Absolute Difference, Fisher's Score, Chi-Square, variance, and correlation coefficient are some examples of filter methods for feature selection (Bakro et al., 2023).

Although a popular choice for feature selection process, there are several limitations and challenges with the use of the Shapley values method, namely: 1) computational complexity, 2) ambiguity in measuring feature interactions, 3) sensitivity to the models being used, 4) complex to interpret, and 5) representativeness of the samples used (Li et al., 2024). The type of feature selection process selected depends on the ML algorithm used. This is done by comparing the available feature combinations

with the criteria that is followed for model evaluation. Wrapper methods produce predictive accuracy, which is superior to that derived from filter methods. Incremental addition of new features termed as forward feature selection is a wrapper method. Eliminating features based on feature importance score is called backward feature elimination, a common wrapper method. Embedded method which is an iterative approach for training the model is the third type of feature selection. During each iteration, the feature which contributes the most is extracted.

The data set used during this praxis contained eighty (80) features making it essential to identify the top features that have an impact on the performance of the model. The three ML models RF, XGBoost, and KNN being evaluated during this praxis include functions that plot feature importances. RF, XGBoost, and KNN algorithms can be used for feature selection and achieve greater accuracy in results thus avoiding the limitations due to Shapley values (Upadhyay et al., 2021). By excluding the features being studied and measuring the model performance again by including the same subset of features is how the feature importance scores were obtained. The in-built functions available within the ML models evaluated during this praxis were used to plot the feature importance scores. A ranking of the features by importance was created with highly correlated features by assigning values based on their contribution to the model. After ranking, the optimal number of features to be used was determined based on the model performance metrics.

3.7.6 Hyperparameter Tuning

There are two categories of ML model parameters: (1) parameters, estimated by the model by using available data, and (2) hyperparameters, which cannot be estimated

by using available data and are useful for estimating model parameters. Tuning the hyperparameters results in providing increased control over the process. There are two methods available for hyperparameter tuning: (1) random search, and (2) grid search. A random search creates a grid of possible values for hyperparameters, and each iteration uses a random combination from the grid providing best combination towards the end. In a grid search, the combination iterations are performed in a specific order and the model is fitted with every combination returning a model with the best hyperparameters (Chowdhury et al., 2020).

The Python library Scikit-learn can be used for random search and grid search using k-fold cross-validation for creating the best model. The three ML models used in this praxis included a variety of hyperparameters that had an impact on the model performance, including speed and accuracy. Hyperparameter tuning is a crucial step in the ML model development, in which different sets of values are used for testing to determine the best combination that produces desired results. In this praxis, Python's *GridSearchCV()* was used to identify the best set of hyperparameter values to be used in evaluating the ML model performance.

3.8 Model Evaluation Metrics

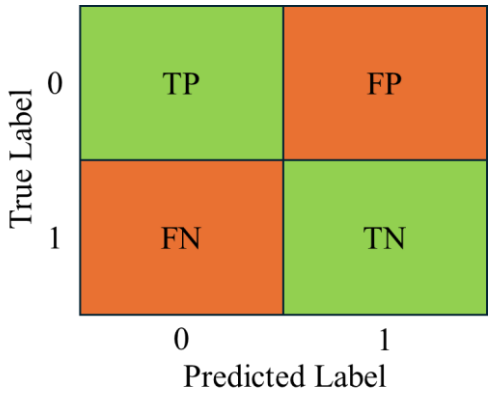
ML models were evaluated by collecting metrics during the training and then during the testing phase for comparison. The metrics collected during the training and testing phases were accuracy, precision, recall, F1 score and AUC-ROC values which were compared among the three (3) models evaluated during this praxis to determine the best ML model.

3.8.1 Confusion Matrix

The confusion matrix summarizes the performance of an ML model. The 'n' possible outcomes can be easily represented using a 'n x n' confusion matrix table (Tharwat, 2020). Figure 7 below illustrates the confusion matrix for a 2 X 2 outcome scenario.

Figure 7

A 2 X 2 Confusion Matrix



The equations to calculate each of the performance metrics discussed are as follows:

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

Where:

TP = Number of True Positives for a class

TN = Number of True Negatives for the same class

FP = Number of False Positives for the same class

FN = Number of False Negatives for the same class

Equation 1 - Formula for Accuracy (Brown, 2018)

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

Where:

TP = Number of True Positives for a class

FP = Number of False Positives for the same class

Equation 2 - Formula for Precision (Brown, 2018)

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

Where:

TP = Number of True Positives for a class

FP = Number of False Positives for the same class

Equation 3 - Formula for Recall (Brown, 2018)

$$F1\ Score = 2* \frac{Precision*Recall}{Precision+Recall}$$

Where:

Precision = Precision from Equation 2

Recall = Recall from Equation 2

Equation 4 - Formula for F1 Score (Brown, 2018)

$$False\ Positive\ Rate = \frac{FP}{FP+TN}$$

Where:

FP = Number of False Positives for a class

TN = Number of True Negatives for the same class

Equation 5 - Formula for False Positive Rate (FPR) (Brown, 2018)

$$AUC\text{-}ROC\ Value = \frac{1+TPR+FPR}{2}$$

Where:

TPR = True Positive rate

FPR = False Positive rate

Equation 6 - Formula for AUC-ROC value (Brown, 2018)

$$Average\ Classification\ Time = \frac{Total\ Classification\ Time}{Number\ of\ records\ classified}$$

Equation 7 - Formula for Average Classification Time (Brown, 2018)

3.9 Summary of Methodology

Chapter 3 explained the methodology that is followed for the ML-based IDS to detect cyberattack traffic within a corporate network. Next, the chapter introduces the data set used in the praxis including the source, and features of the data set. The remaining sections explain the data preprocessing including cleaning, filtering, categorical encoding, standardization, balancing, feature selection, hyperparameter tuning, and model evaluation metrics with the use of confusion matrix. The chapter concludes with description of the confusion matrix which contains useful metrics for performance measurement of the ML models investigated.

Chapter 4: Results

4.1 Introduction

This research is aimed at developing a machine learning (ML)-based intrusion detection system (IDS) to prevent cyberattacks on corporate networks. The system can identify and classify infiltration traffic in the network. An important part of building this model is training the model to detect cyberattack attempts, analyze the training phase results and test the model after refining and tuning the data set features. Experimental analysis on three (3) ML-based models namely: Random Forest (RF), eXtreme Gradient Boosting (XGBoost), and k-Nearest Neighbors (KNN) was performed during this praxis by systematically documenting the results obtained for each model during the training as well as testing phases.

To ascertain which ML model fulfilled the objectives of this praxis, performance metrics were collected and empirically evaluated. To ensure proper flow in explaining the results obtained, Chapter 4 includes sections that correlate directly with methodology explained in Chapter 3.

The subsequent sections of Chapter 4 describe the details of exploratory data analysis (EDA), data preprocessing, which includes data encoding, data standardization, data balancing, and feature selection which are results from each step explained as part of methodology in Chapter 3. Finally, the results from the training and test phases for the ML models being evaluated during the praxis concludes this chapter.

4.2 Exploratory Data Analysis (EDA)

The data set used in this praxis contained numeric and alphanumeric features. Firstly, summary statistics were collected for the numerical features in the data set, which are captured in Table 2.

Table 2

Summary Statistics for the Data set Features

Feature_Name	Count	Mean	Std. Dev.	Min.	25th%	50th%	75th%	Max.
Dst_Port	331100	8753.37	17985.94	0	53	443	3389	65535
Protocol	331100	9.56	5.36	0	6	6	17	17
Flow_Duration	331100	14060152.61	34666653.84	0	270	3023	1748169	119999992
Tot_Fwd_Pkts	331100	5.55	74.02	1	1	2	7	18291
Tot_Bwd_Pkts	331100	5.65	158.04	0	1	1	5	35547
TotLen_Fwd_Pkts	331100	470.90	24746.82	0	0	42	373	9789747
TotLen_Bwd_Pkts	331100	3307.01	220349.43	0	0	71	384	41900000
Fwd_Pkt_Len_Max	331100	164.17	263.72	0	0	40	194	1968
Fwd_Pkt_Len_Min	331100	16.28	41.18	0	0	0	35	1460
Fwd_Pkt_Len_Mean	331100	46.21	61.65	0	0	35	52	1460
Fwd_Pkt_Len_Std	331100	49.76	89.53	0	0	0	70	1324
Bwd_Pkt_Len_Max	331100	320.64	519.16	0	0	67	252	65160
Bwd_Pkt_Len_Min	331100	33.57	54.83	0	0	0	62	1212
Bwd_Pkt_Len_Mean	331100	107.82	176.08	0	0	58	144	33879
Bwd_Pkt_Len_Std	331100	109.25	208.01	0	0	0	76	21326
Flow_Byts_s	329266	inf		0	0	1067	95156	inf
Flow_Pkts_s	331100	inf		0	7	776	7547	inf
Flow_IAT_Mean	331100	2484116.62	11841316.24	0	261	2471	155783	119000000
Flow_IAT_Std	331100	1178967.10	3723009.17	0	0	0	245728	69200000
Flow_IAT_Max	331100	5755644.61	16556442.59	0	269	2968	953513	120000000
Flow_IAT_Min	331100	1923565.10	11755077.91	0	6	50	500	119000000
Fwd_IAT_Tot	331100	13805592.17	34593961.11	0	0	136	1672873	120000000
Fwd_IAT_Mean	331100	2890604.19	12095885.41	0	0	104	265182	120000000
Fwd_IAT_Std	331100	1362720.29	4504616.23	0	0	0	341236	84700000
Fwd_IAT_Max	331100	5636548.63	16499342.27	0	0	129	1109419	120000000
Fwd_IAT_Min	331100	2029919.97	11918782.05	0	0	3	170	120000000
Bwd_IAT_Tot	331100	9926704.62	29901034.51	0	0	0	731751	120000000
Bwd_IAT_Mean	331100	977011.20	4176240.96	0	0	0	220846	120000000
Bwd_IAT_Std	331100	1065749.61	3915000.83	0	0	0	26396	84700000
Bwd_IAT_Max	331100	3272654.87	11563005.83	0	0	0	311159	120000000

Feature_Name	Count	Mean	Std. Dev.	Min.	25th%	50th%	75th%	Max.
Bwd_IAT_Min	331100	301195.86	3453485.03	0	0	0	19	120000000
Fwd_PSH_Flags	331100	0.06	0.25	0	0	0	0	1
Bwd_PSH_Flags	331100	0.00	0.00	0	0	0	0	0
Fwd_URG_Flags	331100	0.00	0.05	0	0	0	0	1
Bwd_URG_Flags	331100	0.00	0.00	0	0	0	0	0
Fwd_Header_Len	331100	102.72	1405.66	0	8	24	132	332000
Bwd_Header_Len	331100	114.49	2926.03	0	8	20	112	531332
Fwd_Pkts_s	331100	55685.21	243082.93	0	4	162	3802	3000000
Bwd_Pkts_s	331100	30570.17	121426.71	0	0	25	2710	2000000
Pkt_Len_Min	331100	16.04	39.15	0	0	0	35	1460
Pkt_Len_Max	331100	338.31	522.55	0	0	70	465	65160
Pkt_Len_Mean	331100	73.96	106.97	0	0	48	89	17345
Pkt_Len_Std	331100	101.43	162.86	0	0	22	125	22788
Pkt_Len_Var	331100	36815.84	905414.25	0	0	470	15537	519000000
FIN_Flag_Cnt	331100	0.01	0.08	0	0	0	0	1
SYN_Flag_Cnt	331100	0.06	0.25	0	0	0	0	1
RST_Flag_Cnt	331100	0.20	0.40	0	0	0	0	1
PSH_Flag_Cnt	331100	0.43	0.50	0	0	0	1	1
ACK_Flag_Cnt	331100	0.21	0.40	0	0	0	0	1
URG_Flag_Cnt	331100	0.05	0.22	0	0	0	0	1
CWE_Flag_Count	331100	0.00	0.05	0	0	0	0	1
ECE_Flag_Cnt	331100	0.20	0.40	0	0	0	0	1
Down_Up_Ratio	331100	0.63	1.04	0	0	1	1	237
Pkt_Size_Avg	331100	86.87	112.69	0	0	70	120	17478
Fwd_Seg_Size_Avg	331100	46.21	61.65	0	0	35	52	1460
Bwd_Seg_Size_Avg	331100	107.82	176.08	0	0	58	144	33879
Fwd_Byts_b_Avg	331100	0.00	0.00	0	0	0	0	0
Fwd_Pkts_b_Avg	331100	0.00	0.00	0	0	0	0	0
Fwd_Blk_Rate_Avg	331100	0.00	0.00	0	0	0	0	0
Bwd_Byts_b_Avg	331100	0.00	0.00	0	0	0	0	0
Bwd_Pkts_b_Avg	331100	0.00	0.00	0	0	0	0	0
Bwd_Blk_Rate_Avg	331100	0.00	0.00	0	0	0	0	0
Subflow_Fwd_Pkts	331100	5.55	74.02	1	1	2	7	18291
Subflow_Fwd_Byts	331100	470.90	24746.82	0	0	42	373	9789747
Subflow_Bwd_Pkts	331100	5.65	158.04	0	1	1	5	35547
Subflow_Bwd_Byts	331100	3306.44	220294.58	0	0	71	384	41866587
Init_Fwd_Win_Byts	331100	4737.22	10792.16	-1	-1	258	8192	65535
Init_Bwd_Win_Byts	331100	8818.84	21249.15	-1	-1	0	252	65535
Fwd_Act_Data_Pkts	331100	2.18	41.68	0	0	0	3	18290
Fwd_Seg_Size_Min	331100	16.34	7.12	0	8	20	20	48
Active_Mean	331100	289864.63	3182353.79	0	0	0	0	114000000

Feature_Name	Count	Mean	Std. Dev.	Min.	25th%	50th%	75th%	Max.
Active_Std	331100	130707.21	1903213.66	0	0	0	0	72200000
Active_Max	331100	418268.96	4191694.89	0	0	0	0	114000000
Active_Min	331100	201525.62	2701849.36	0	0	0	0	114000000
Idle_Mean	331100	5222464.83	16130322.74	0	0	0	0	120000000
Idle_Std	331100	275698.84	2082252.11	0	0	0	0	55200000
Idle_Max	331100	5483134.15	16569711.89	0	0	0	0	120000000
Idle_Min	331100	4968130.32	15931901.60	0	0	0	0	120000000

The two features of the data set namely, 'Flow_Byts_s' and 'Flow_Pkts_s' have a maximum value of Infinity which resulted in the mean and maximum values as 'inf' in the descriptive statistics summarized in Table 2 above. Using the '*replace*' function of the *numpy* library of Python, these values have been replaced with zeroes and descriptive statistics generated again as summarized in Table 3.

Table 3

Summary Statistics After Removing infinity values

Feature_Name	Count	Mean	Std. Dev.	Min.	25th%	50th%	75th%	Max.
Dst_Port	331100	8753.37	17985.94	0	53	443	3389	65535
Protocol	331100	9.56	5.36	0	6	6	17	17
Flow_Duration	331100	14060152.61	34666653.84	0	270	3023	1748169	119999992
Tot_Fwd_Pkts	331100	5.55	74.02	1	1	2	7	18291
Tot_Bwd_Pkts	331100	5.65	158.04	0	1	1	5	35547
TotLen_Fwd_Pkts	331100	470.90	24746.82	0	0	42	373	9789747
TotLen_Bwd_Pkts	331100	3307.01	220349.43	0	0	71	384	41900000
Fwd_Pkt_Len_Max	331100	164.17	263.72	0	0	40	194	1968
Fwd_Pkt_Len_Min	331100	16.28	41.18	0	0	0	35	1460
Fwd_Pkt_Len_Mean	331100	46.21	61.65	0	0	35	52	1460
Fwd_Pkt_Len_Std	331100	49.76	89.53	0	0	0	70	1324
Bwd_Pkt_Len_Max	331100	320.64	519.16	0	0	67	252	65160
Bwd_Pkt_Len_Min	331100	33.57	54.83	0	0	0	62	1212
Bwd_Pkt_Len_Mean	331100	107.82	176.08	0	0	58	144	33879
Bwd_Pkt_Len_Std	331100	109.25	208.01	0	0	0	76	21326
Flow_Byts_s	331100	285396.86	3546117.72	0	0	1021	90838	708000000

Feature_Name	Count	Mean	Std. Dev.	Min.	25th%	50th%	75th%	Max.
Flow_Pkts_s	331100	86255.39	317738.51	0	7	287	7380	3000000
Flow_IAT_Mean	331100	2484116.62	11841316.24	0	261	2471	155783	119000000
Flow_IAT_Std	331100	1178967.10	3723009.17	0	0	0	245728	69200000
Flow_IAT_Max	331100	5755644.61	16556442.59	0	269	2968	953513	120000000
Flow_IAT_Min	331100	1923565.10	11755077.91	0	6	50	500	119000000
Fwd_IAT_Tot	331100	13805592.17	34593961.11	0	0	136	1672873	120000000
Fwd_IAT_Mean	331100	2890604.19	12095885.41	0	0	104	265182	120000000
Fwd_IAT_Std	331100	1362720.29	4504616.23	0	0	0	341236	84700000
Fwd_IAT_Max	331100	5636548.63	16499342.27	0	0	129	1109419	120000000
Fwd_IAT_Min	331100	2029919.97	11918782.05	0	0	3	170	120000000
Bwd_IAT_Tot	331100	9926704.62	29901034.51	0	0	0	731751	120000000
Bwd_IAT_Mean	331100	977011.20	4176240.96	0	0	0	220846	120000000
Bwd_IAT_Std	331100	1065749.61	3915000.83	0	0	0	26396	84700000
Bwd_IAT_Max	331100	3272654.87	11563005.83	0	0	0	311159	120000000
Bwd_IAT_Min	331100	301195.86	3453485.03	0	0	0	19	120000000
Fwd_PSH_Flags	331100	0.06	0.25	0	0	0	0	1
Bwd_PSH_Flags	331100	0.00	0.00	0	0	0	0	0
Fwd_URG_Flags	331100	0.00	0.05	0	0	0	0	1
Bwd_URG_Flags	331100	0.00	0.00	0	0	0	0	0
Fwd_Header_Len	331100	102.72	1405.66	0	8	24	132	332000
Bwd_Header_Len	331100	114.49	2926.03	0	8	20	112	531332
Fwd_Pkts_s	331100	55685.21	243082.93	0	4	162	3802	3000000
Bwd_Pkts_s	331100	30570.17	121426.71	0	0	25	2710	2000000
Pkt_Len_Min	331100	16.04	39.15	0	0	0	35	1460
Pkt_Len_Max	331100	338.31	522.55	0	0	70	465	65160
Pkt_Len_Mean	331100	73.96	106.97	0	0	48	89	17345
Pkt_Len_Std	331100	101.43	162.86	0	0	22	125	22788
Pkt_Len_Var	331100	36815.84	905414.25	0	0	470	15537	519000000
FIN_Flag_Cnt	331100	0.01	0.08	0	0	0	0	1
SYN_Flag_Cnt	331100	0.06	0.25	0	0	0	0	1
RST_Flag_Cnt	331100	0.20	0.40	0	0	0	0	1
PSH_Flag_Cnt	331100	0.43	0.50	0	0	0	1	1
ACK_Flag_Cnt	331100	0.21	0.40	0	0	0	0	1
URG_Flag_Cnt	331100	0.05	0.22	0	0	0	0	1
CWE_Flag_Count	331100	0.00	0.05	0	0	0	0	1
ECE_Flag_Cnt	331100	0.20	0.40	0	0	0	0	1
Down_Up_Ratio	331100	0.63	1.04	0	0	1	1	237
Pkt_Size_Avg	331100	86.87	112.69	0	0	70	120	17478

Feature_Name	Count	Mean	Std. Dev.	Min.	25th%	50th%	75th%	Max.
Fwd_Seg_Size_Avg	331100	46.21	61.65	0	0	35	52	1460
Bwd_Seg_Size_Avg	331100	107.82	176.08	0	0	58	144	33879
Fwd_Byts_b_Avg	331100	0.00	0.00	0	0	0	0	0
Fwd_Pkts_b_Avg	331100	0.00	0.00	0	0	0	0	0
Fwd_Blk_Rate_Avg	331100	0.00	0.00	0	0	0	0	0
Bwd_Byts_b_Avg	331100	0.00	0.00	0	0	0	0	0
Bwd_Pkts_b_Avg	331100	0.00	0.00	0	0	0	0	0
Bwd_Blk_Rate_Avg	331100	0.00	0.00	0	0	0	0	0
Subflow_Fwd_Pkts	331100	5.55	74.02	1	1	2	7	18291
Subflow_Fwd_Byts	331100	470.90	24746.82	0	0	42	373	9789747
Subflow_Bwd_Pkts	331100	5.65	158.04	0	1	1	5	35547
Subflow_Bwd_Byts	331100	3306.44	220294.58	0	0	71	384	41866587
Init_Fwd_Win_Byts	331100	4737.22	10792.16	-1	-1	258	8192	65535
Init_Bwd_Win_Byts	331100	8818.84	21249.15	-1	-1	0	252	65535
Fwd_Act_Data_Pkts	331100	2.18	41.68	0	0	0	3	18290
Fwd_Seg_Size_Min	331100	16.34	7.12	0	8	20	20	48
Active_Mean	331100	289864.63	3182353.79	0	0	0	0	114000000
Active_Std	331100	130707.21	1903213.66	0	0	0	0	72200000
Active_Max	331100	418268.96	4191694.89	0	0	0	0	114000000
Active_Min	331100	201525.62	2701849.36	0	0	0	0	114000000
Idle_Mean	331100	5222464.83	16130322.74	0	0	0	0	120000000
Idle_Std	331100	275698.84	2082252.11	0	0	0	0	55200000
Idle_Max	331100	5483134.15	16569711.89	0	0	0	0	120000000
Idle_Min	331100	4968130.32	15931901.60	0	0	0	0	120000000

Descriptive statistics generated the minimum, maximum, mean, and standard deviation values for the numeric features in the data set. The‘Flow_Byts_s’ and ‘Flow_Pkts_s’ features which denote the flow of bytes per second and the flow of packets per second have a minimum value of 0 and maximum values of 708000000 and 3000000, respectively. Standard deviation denotes the spread of data points over the range and a high value of standard deviation means the data has lot of variability, less consistency, and is unpredictable. The differences in the summary statistics values for the

‘Flow_Duration’ feature was much more significant when compared with the same values for the forward and backward packet length features. The standard deviation value for the ‘Flow_Duration’ feature is highest among all the other features ranked first with a value of 34666653.84. Thus, the features with higher values will not introduce Skewness and Kurtosis into the data and adversely affect the model performance because of the disproportion of the weights carried by each feature. Skewness is a statistic that describes the horizontal pull on the distribution of the data and Kurtosis describes the vertical pull. Additionally, Table 3 shows several other features whose minimum and maximum values have been distributed across wider ranges of values. The two features ‘Flow_Byts_s,’ and ‘Flow_Pkts_s’ have been further examined in Figure 8 and Figure 9 which illustrates the distribution of these two features’ values for the data set.

Figure 8

Distribution of flow of Bytes

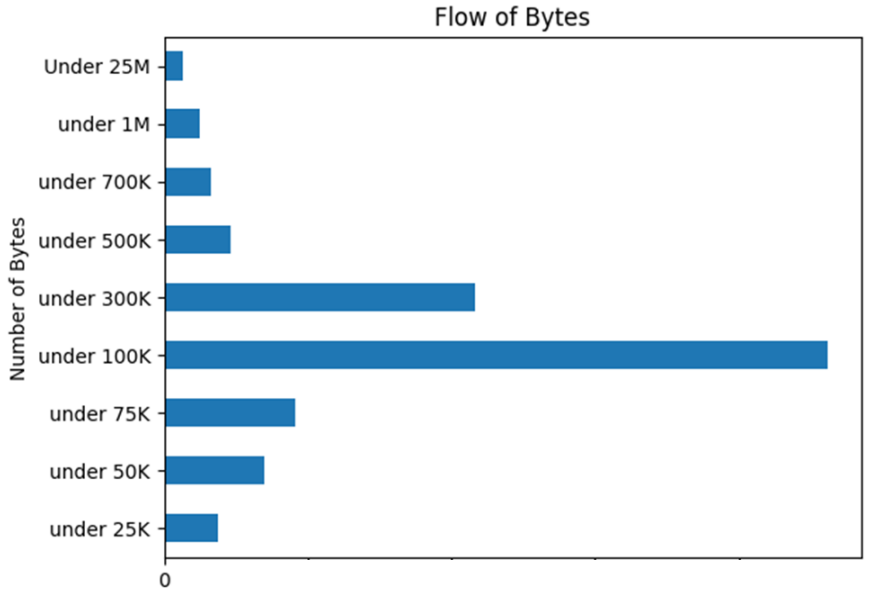
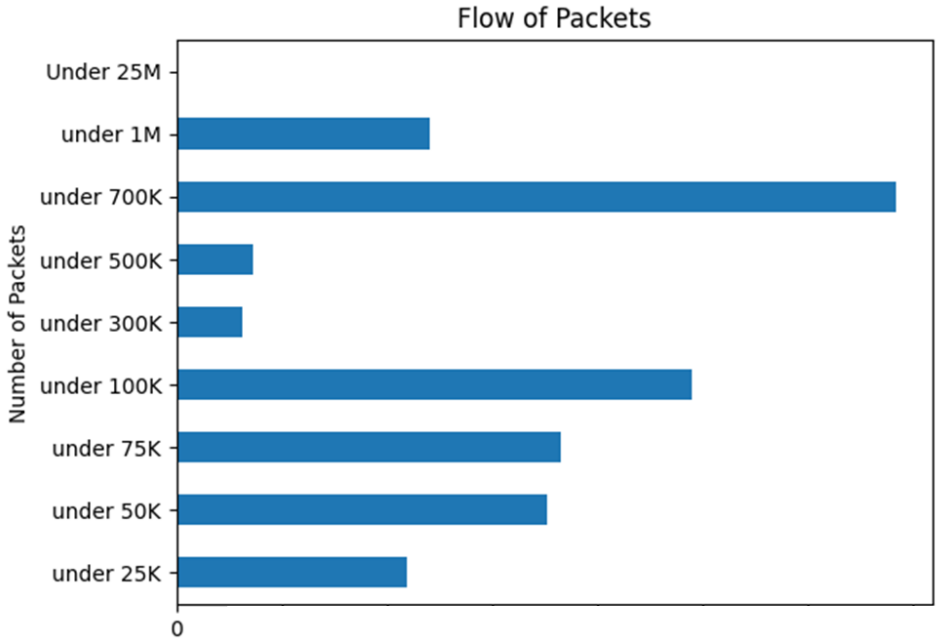


Figure 9

Distribution of flow of Packets



The comparison of the duration for the source-to-destination with the byte count and packet count values prove that data standardization must be performed on the data set. Python '*pandas'* library has inbuilt method '*isna()'* to read the data set and count the number of values that are not applicable (NA) There were no rows or columns of the data set that had NA value. The '*info()'* method was used to identify the data types of each individual feature of the data set. There were two (2) features namely: *Timestamp* and *Label* that had the data type '*object'* and the remaining 78 features of the data set had the '*int64'*data type. The EDA was performed using Python, which categorizes alphanumeric characters as object data type, and 64-bit integers as int64. Most ML models and especially the three models tested as part of this praxis could only handle numeric data. Hence, only those features in the data set that are int64 were suitable for

testing purposes of this praxis. The ‘Timestamp’ and ‘Label’ are the only two features of the data set used in this praxis that were of the data type object. The ‘Label’ feature was mapped to numerical values for the purposes of evaluating the three ML models efficiently. Categorical data encoding was clearly the method that needed to be used after ascertaining the features’ data type and the need to convert the object data type to numeric data type.

The statistical measure, correlation, is used to calculate the strength and direction of relationship between features of a data set. Figure 10 shows the correlation heatmap created including all features of the data set. Several features in the data set have considerable amounts of correlation representing strong interrelationships between them. For example, the flow of bytes per second, ‘Flow_Byts_s’ and the flow of packets per second, ‘Flow_Pkts_s’ demonstrated a strong relationship. However, there were features that did not exhibit any correlation and were not included in the correlation heatmap due to their values being zero as shown in the white area/gaps in the correlation heatmap, in Figure 10.

A second correlation heatmap was generated by eliminating features that did not contribute due to the values in them as shown in Figure 11. There are data set features that showed 100% correlation among them. This mostly included the features related to flow of bytes and flow of packets, since each packet contained a specific number of bytes and there was direct proportionality between these features where the number of source-to-destination packets increased or decreased, which can also be seen in the number of bytes for the destination-to-source path.

Figure 10

Summary Correlation Heatmap with all Data set Features Included

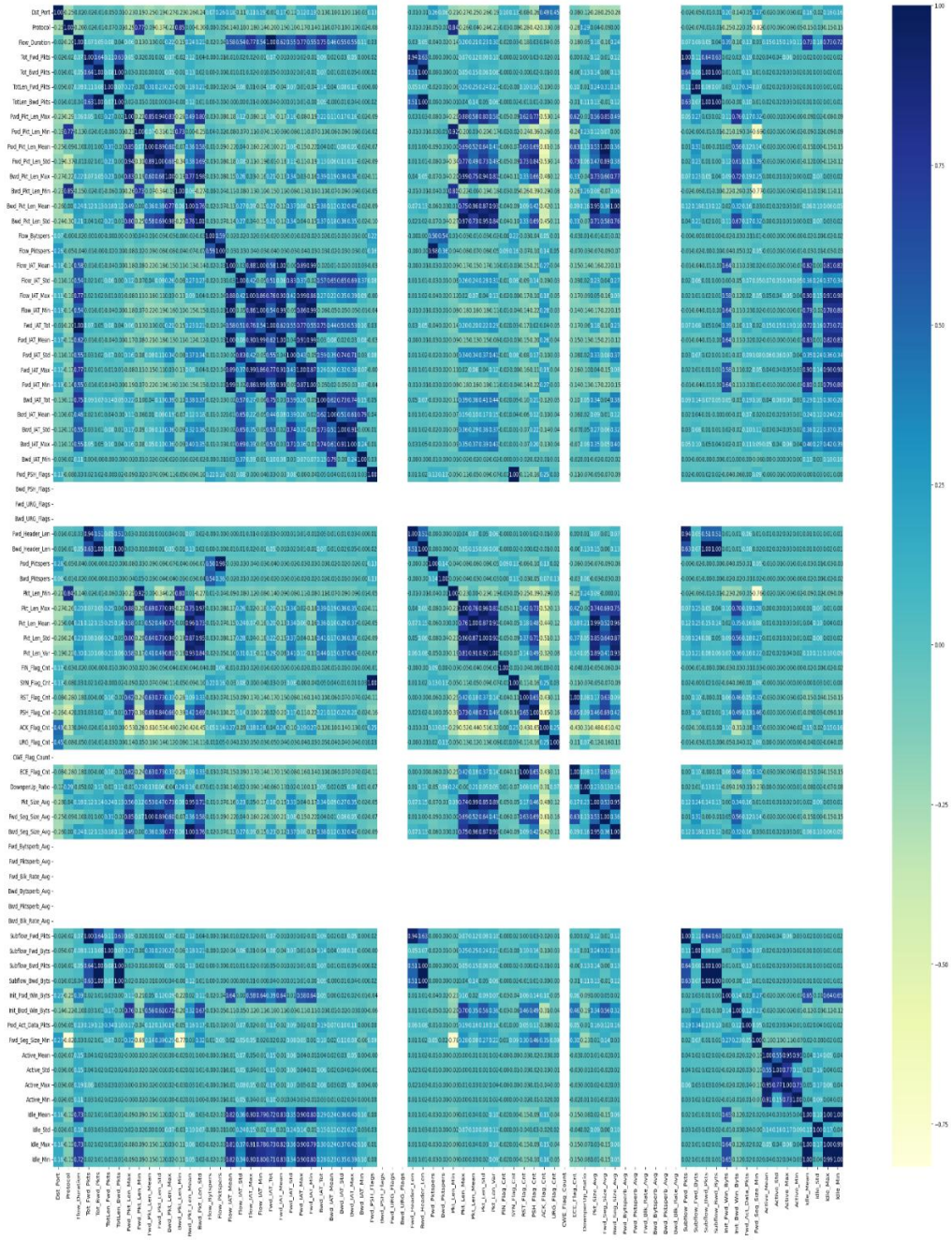
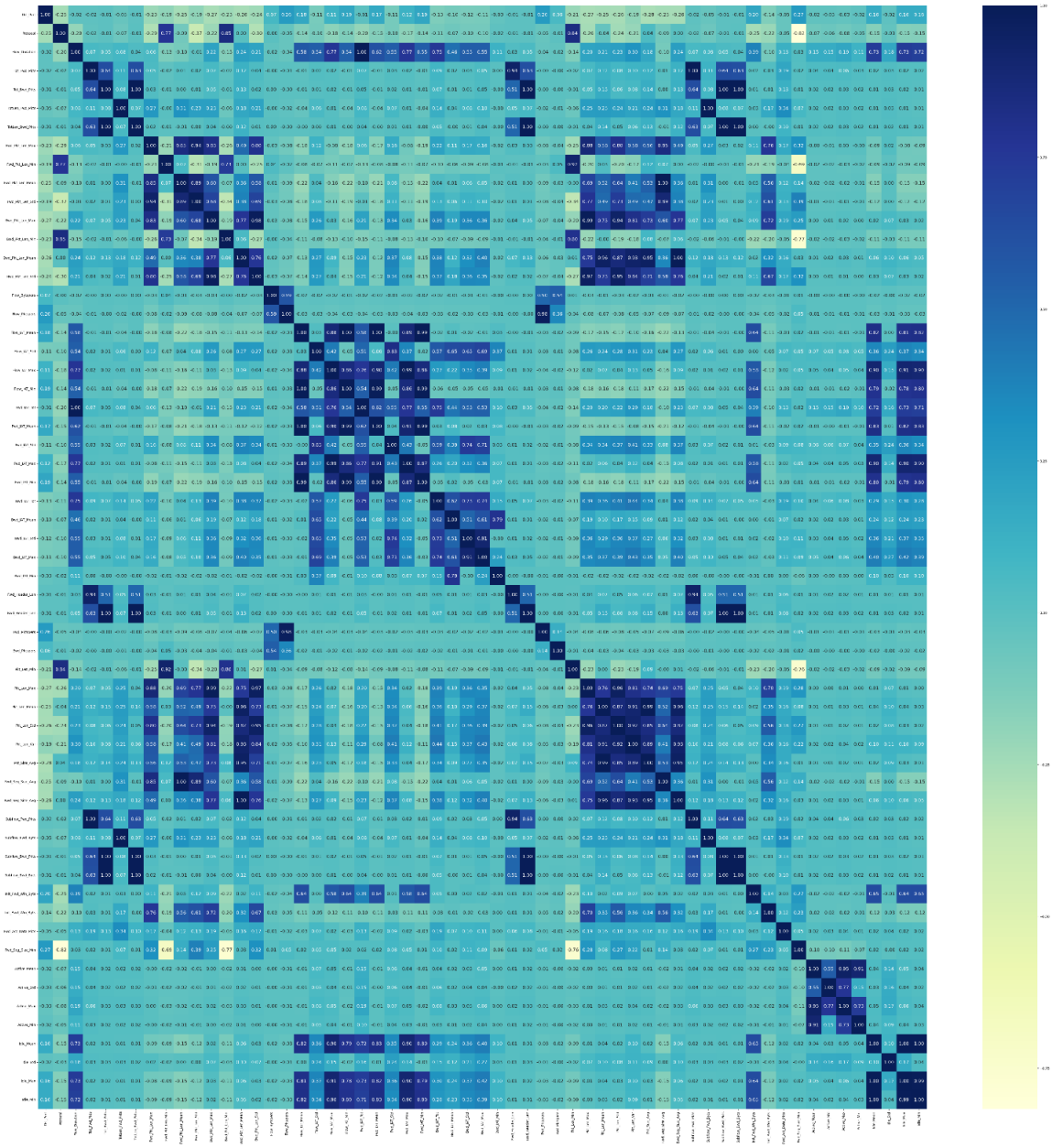


Figure 11

Summary Correlation Heatmap with Non-Relevant Features Excluded

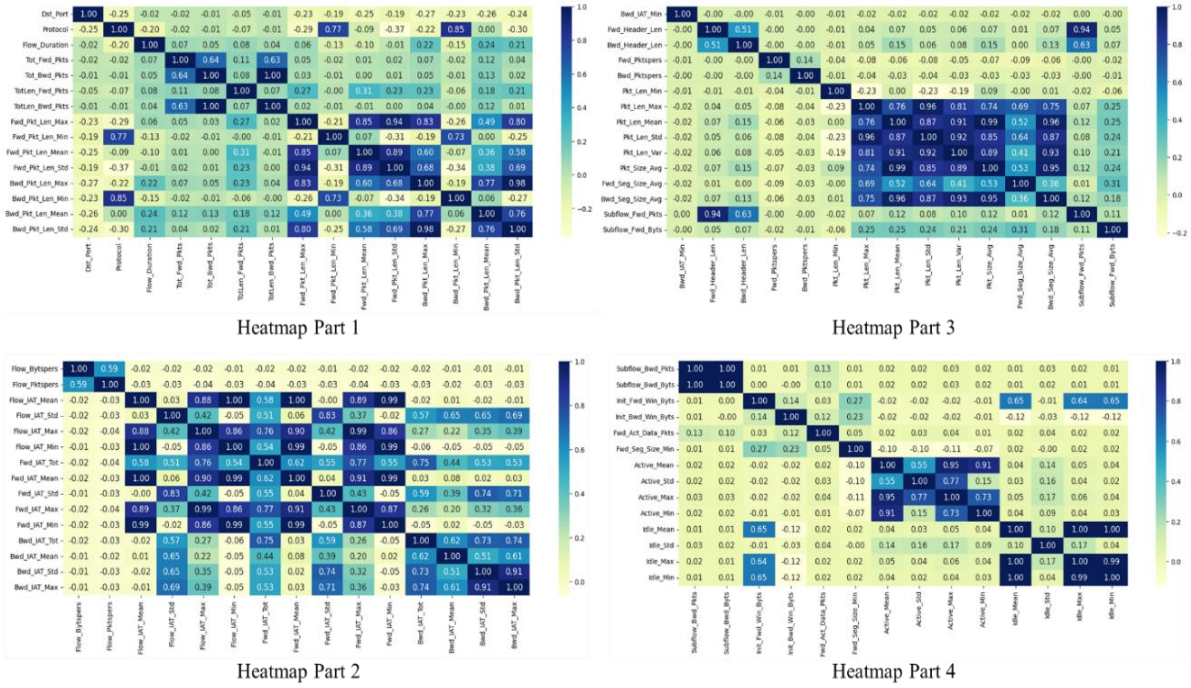


The data set contained eighty (80) features. The second correlation heatmap was generated after eliminating twenty (20) unwanted features. The heatmap is not readable

because it contains correlation coefficients of sixty (60) features. The data set is divided into four (4) parts to generate correlation heatmaps shown in Figure 12.

Figure 12

Correlation Heatmaps for the Data set Features



The sb.heatmap function of Python's sklearn library was used to generate the correlation heatmaps. The correlation heatmap produced a matrix that displayed the 2-dimensional interrelationships between data set features numerically. A value of '1' which is seen in the diagonal of the heat demonstrates perfect correlations between two features. Strong correlation between each other was exhibited by destination port, packet length and packet size, packets per second, and bytes per second features of the data set. These correlations between features of the dataset significantly impact the prediction of

the label. The values from the correlation heatmap helped in identifying those features which can impact the ‘Label’ feature.

The test data set was created with 150,000 records randomly selected from a data set containing 613,071 records by using the *sample* function of Python’s Pandas library. The file was saved separately and used as the test data set for all three models being evaluated. The distribution of the benign and infiltration traffic for the training and test data sets used in this praxis is shown in Figure 13 and Figure 14 below.

Figure 13

Distribution of Benign and Infiltration Traffic for the Training Data Set

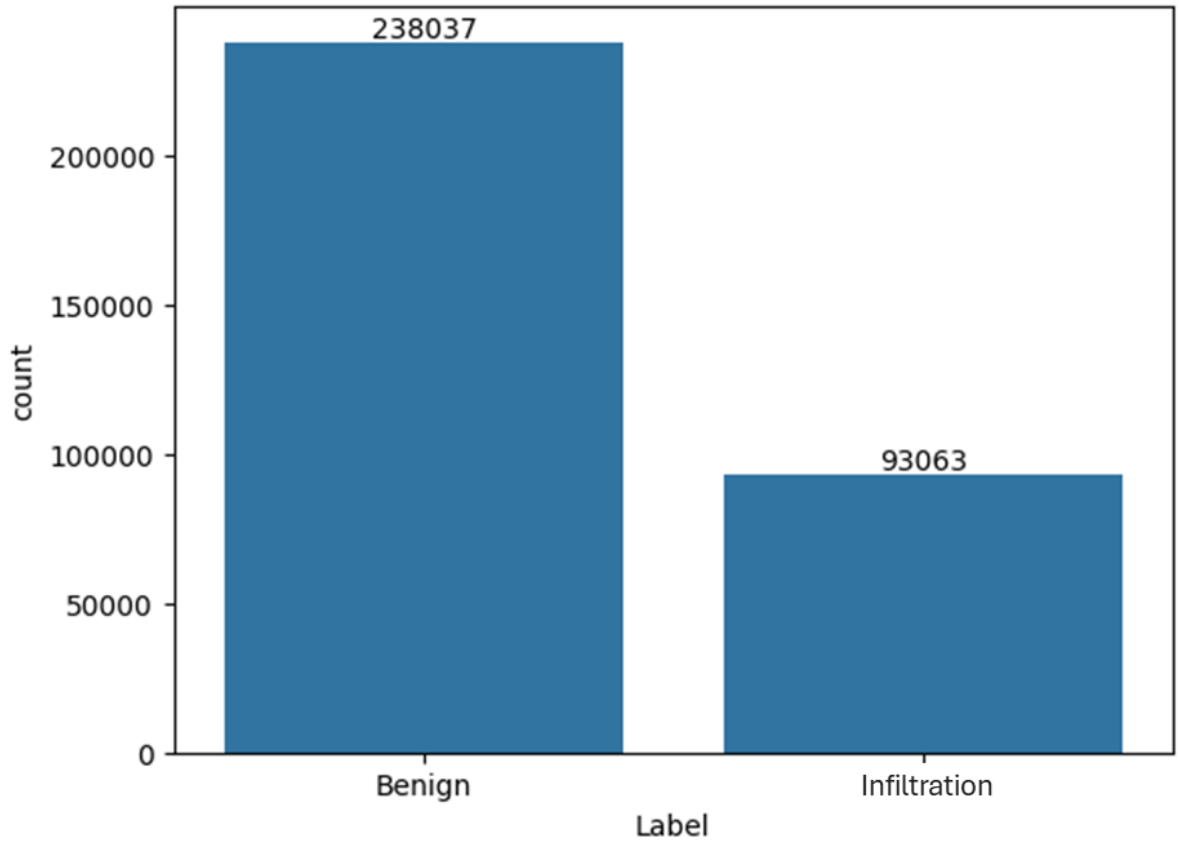
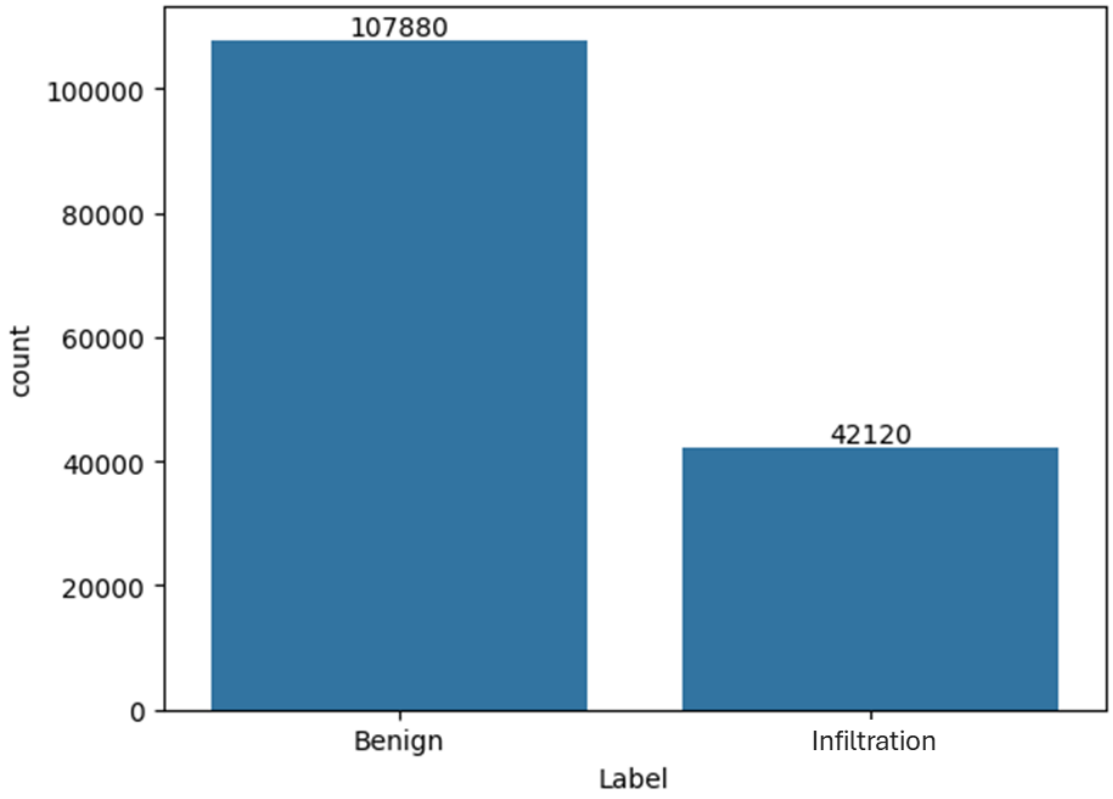


Figure 14

Distribution of Benign and Infiltration Traffic for the Test Data Set



4.3 Data Preprocessing

Data preprocessing directly impacts efficiency and is essential for transforming raw data set making it suitable for use by the ML model. It involves improving the quality of data to ensure compatibility with the ML model by encoding categorical data, standardization, balancing and selecting features that significantly influence model outcomes.

4.3.1 Categorical Data Encoding

The data set used in this praxis contained two (2) features of the data type *object*, 78 of the *int64* type. The two features that belonged to the data type *object* were: (1)

label, and (2) timestamp. ‘Label’ is categorical data that classifies the network traffic as ‘Benign,’ or ‘Infiltration.’ Timestamp was loaded as *object* data type because of the forward slash symbol (‘/’) and the colon symbol (‘:’) that are used as a date separator. ‘Timestamp’ feature was dropped from both training and test data sets as it did not have any impact on the ‘Label’ feature.

The purpose of converting these features into numerical values was to use them in the ML models investigated in this praxis. Table 4 describes the mapping of the categorical values in the Label feature of the data set.

Table 4

Data Encoding Map for Categorical Attack Type

Categorical Attack Type	Numerical Value Assigned
Benign	0
Infiltration	1

4.3.2 Data Standardization

EDA phase has identified features that have a wider spread in terms of the ranges for their values. ML models do not perform well when those features are used and produce skewed results. The Z-score standardization technique was used to standardize all numerical values in the data sets so that the mean was set to zero and minimized standard deviation and variance to unit. The equation below shows how Z-score was calculated.

$$Z = (X - \mu)/\sigma$$

- X = A single raw data value

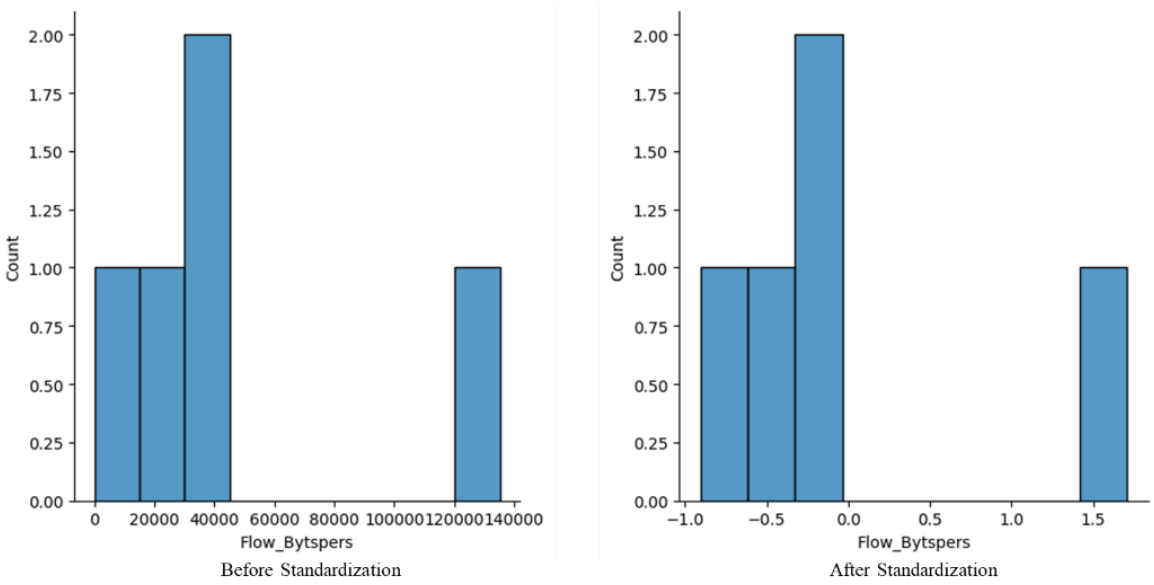
- μ = Population Mean
- σ = Population Standard Deviation

Equation 8 - Formula for Z-Score Calculation (Andrade, 2021)

The ‘Flow_Byts_s’ which denotes the flow of bytes per second feature of the data set has a range distributed between 0 and 708,000,000. The distribution of the values of the ‘Flow_Byts_s’ feature has been adjusted around zero with unit standard deviation by using the Z-Score Standardization technique. Figure 15 shows how the mean is standardized around zero using Z-Score technique. The graph on the right side of Figure 15 illustrates how the same set of feature values are within unit standard deviation on either side of the mean.

Figure 15

Feature Distribution Changes after Z-Score Standardization – IDS



The data preprocessing technique using Z-Score Standardization method ensures that features are centered and scaled so that range bias can be avoided, and the ML model

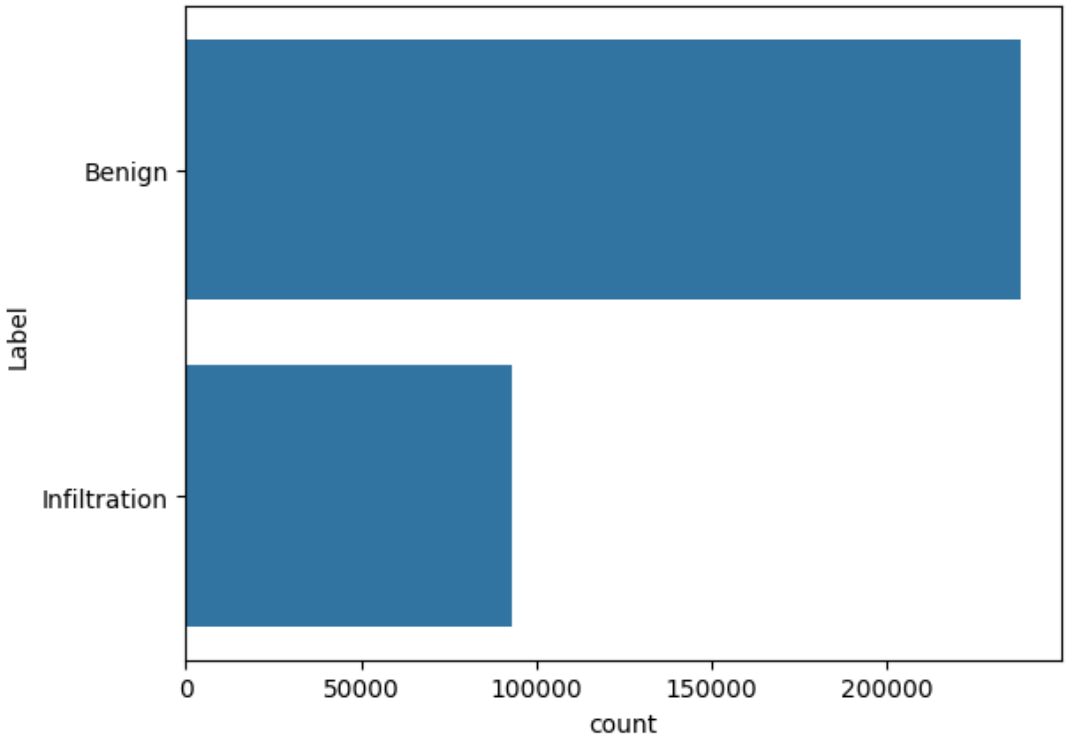
performs accurately. Standardization using the Z-score values is applied on both training and test data set features that were numeric.

4.3.3 Data Balancing

The Label feature of the data set used in this praxis had different frequencies as shown in Figure 16 below.

Figure 16

Distribution of Label Feature Before Balancing

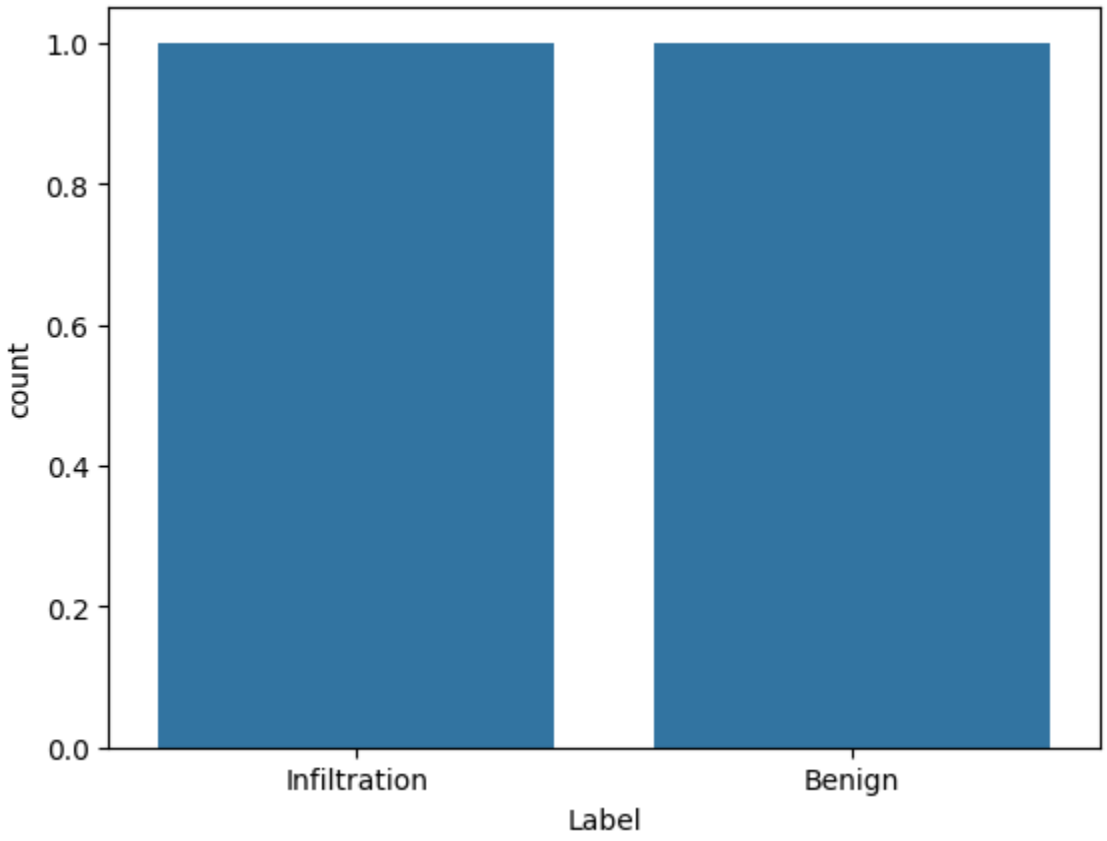


Synthetic Minority Oversampling Technique (SMOTE) which was described in Chapter 3 was used for balancing the data set. The ‘Infiltration’ value of the ‘Label’ feature was the minority class. By using SMOTE synthetic instances of the minority class were generated by interpolating between existing samples without duplicating data

points by creating new samples. Figure 17 shows the distribution of the Label feature after balancing by using the SMOTE technique. Both ‘Benign’ and ‘Infiltration’ classes were evenly distributed.

Figure 17

Distribution of Label Feature After Balancing



4.3.4 Feature Selection

Feature selection is a critical data preprocessing activity for building an efficient model that is both fast and accurate. The inbuilt functions available within the ML

models being evaluated during this praxis were used for feature selection process as described in section 3.7.5 of Chapter 3.

With regards to the data set used in this praxis, the primary challenge aimed at solving was to find out how each feature affected the determination of whether the network traffic was benign or infiltration. For a given feature, the feature importance score is the average change it has caused in the prediction, which was determined by the features already present in the prediction process. The results of feature selection based on XGBoost, RF, and KNN classifier importance scores are summarized in Figures 18, 19 and 20 below.

Figure 18

XGBoost Feature Importance Scores Bar Chart for Data set Features

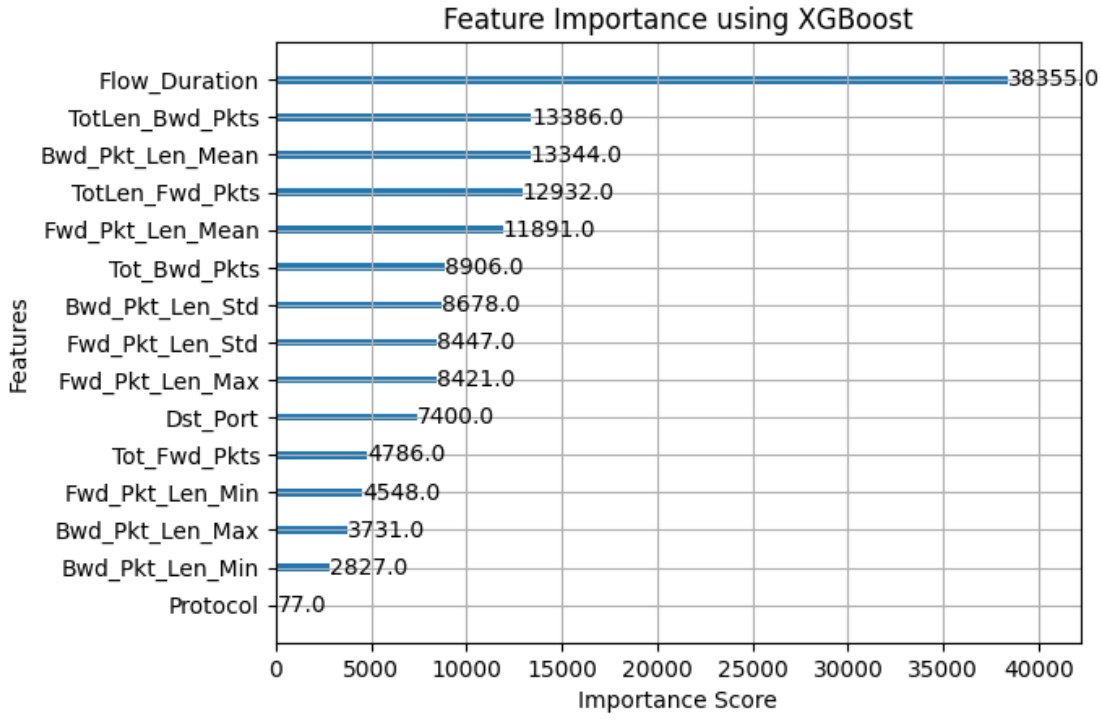


Figure 19

RF Feature Importance Scores Bar Chart for Data set Features

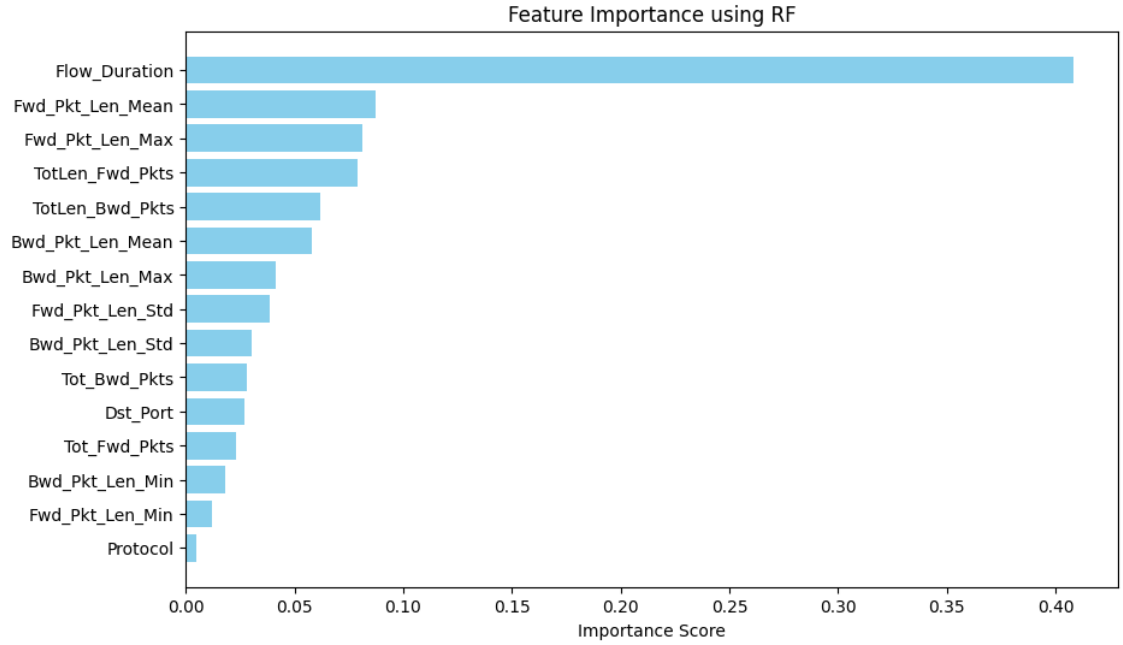


Figure 20

KNN Feature Importance Scores Bar Chart for Data set Features

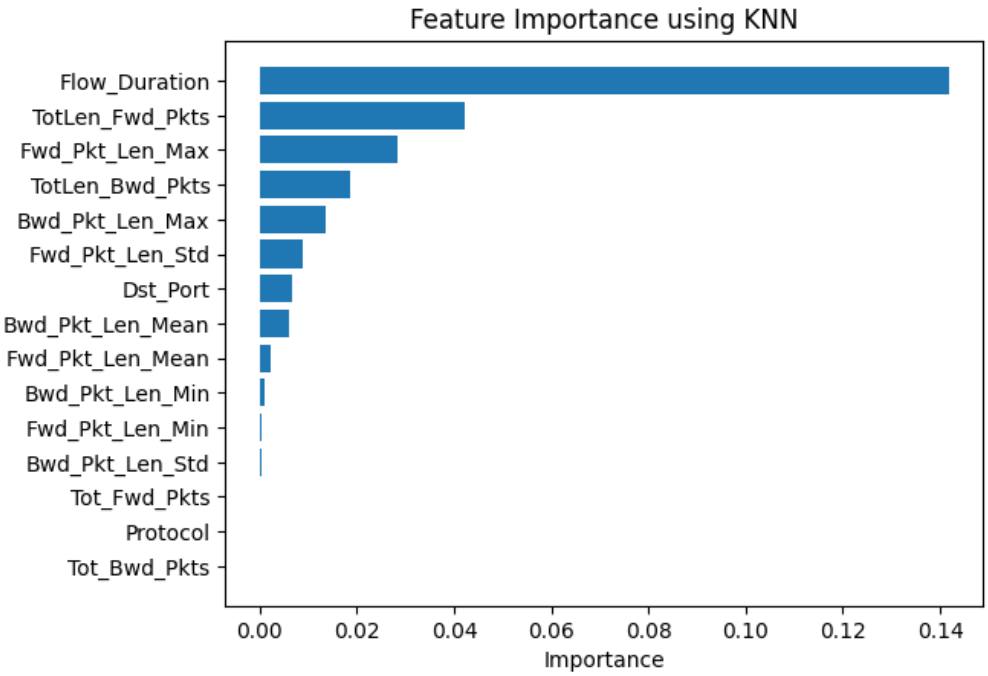
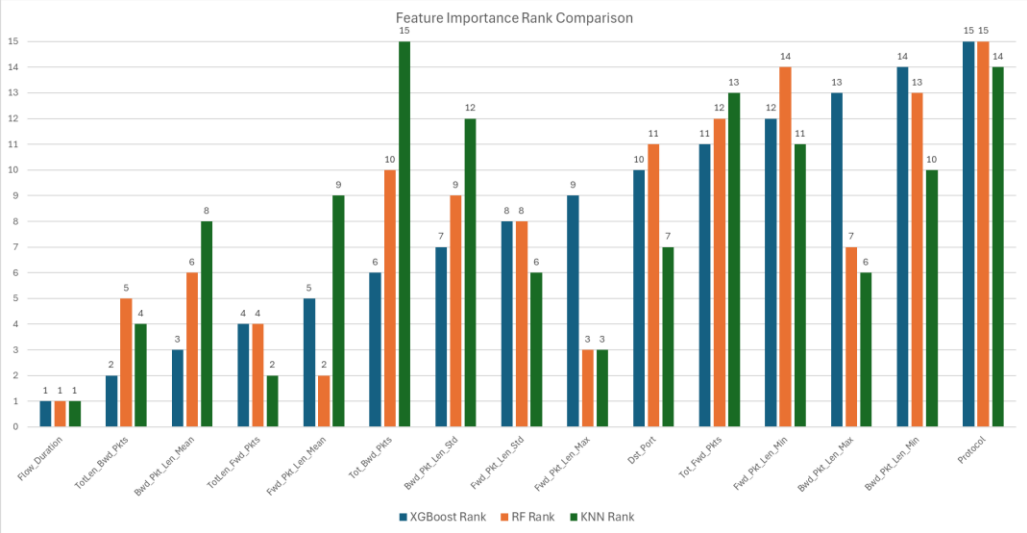


Figure 21 shows the comparison of ranks for each of the top fifteen features as evaluated by each of the three (3) ML models being examined during the praxis.

Figure 21

Feature Importance Rank Comparison between XGBoost, RF and KNN

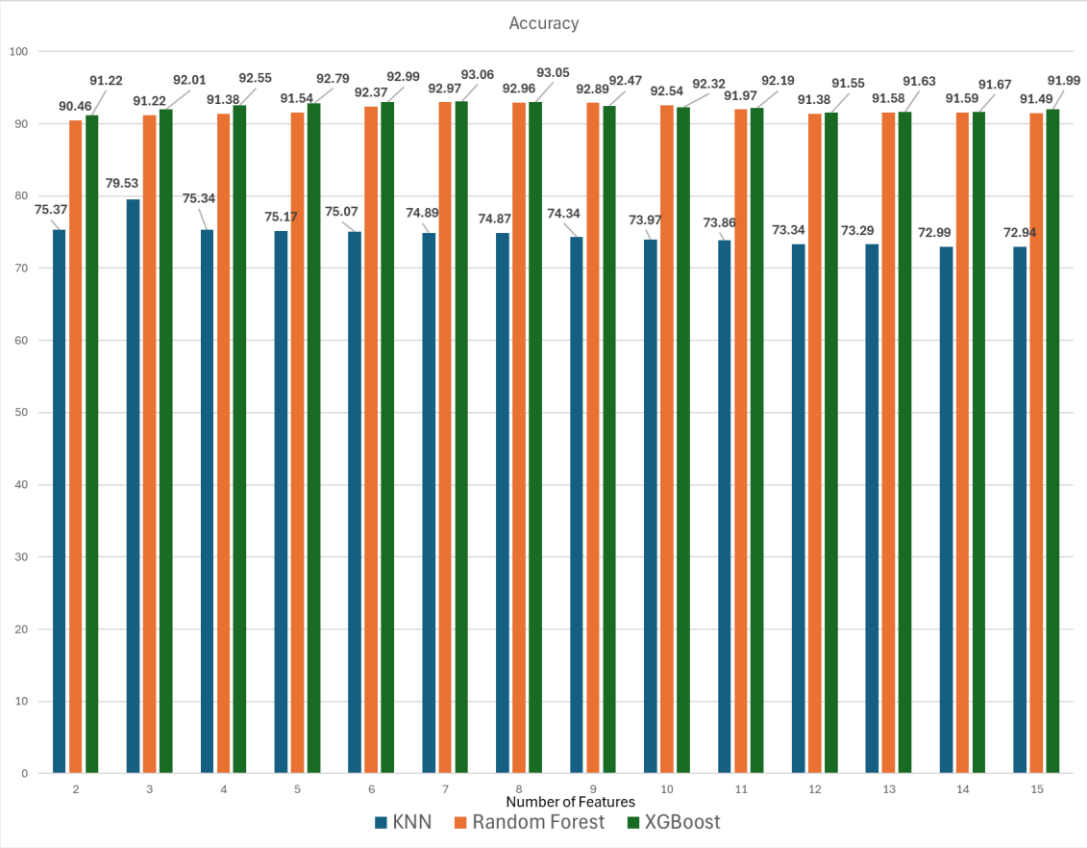


The next step in feature selection process is to identify the minimum number of features that can be used to train the model. The top fifteen data set features obtained by using the inbuilt feature importance scoring methods of the RF, XGBoost, and KNN algorithms were used to train the ML models. An iterative approach was followed to identify the optimum combination of features that produced desired accuracy. The models were first run using top two (2) features in terms of the importance score and incrementing by one (1) for every subsequent run resulting in a total of fourteen runs for the ML models being evaluated during this praxis. The accuracy against number of features used for each run is plotted and summarized in the graph shown in Figure 20. It is evident from the graph that RF and XGBoost based ML models hit their peak performance with an accuracy of 92.97% and 93.06% respectively at seven (7) features

while the KNN based model showed highest accuracy of 79.53% with three (3) features. Figure 22 shows a bar chart comparison of accuracy for each ML model plotted against number of features.

Figure 22

Accuracy Comparison plotted against Number of Features



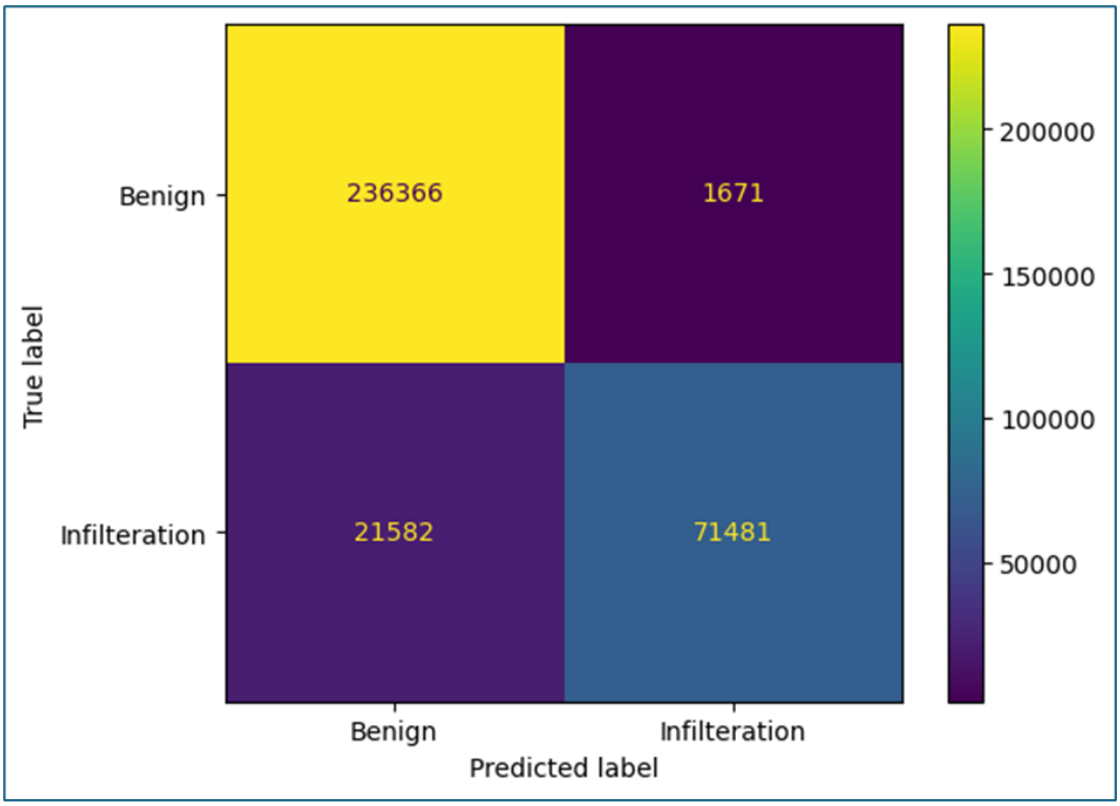
4.3.5 Training of RF, XGBoost, and KNN ML Models

The ML models were trained after performing data preprocessing activities like hyperparameter tuning and feature selection. A data set containing 331,100 records was used for training RF, XGBoost, and KNN models.

RF was the first ML model trained which produced an accuracy of 92.97%, precision value of 97.71%, recall value of 88.05%, and an F1 score of 90.66% as shown in Figure 23 below.

Figure 23

Confusion Matrix for the Trained RF Model

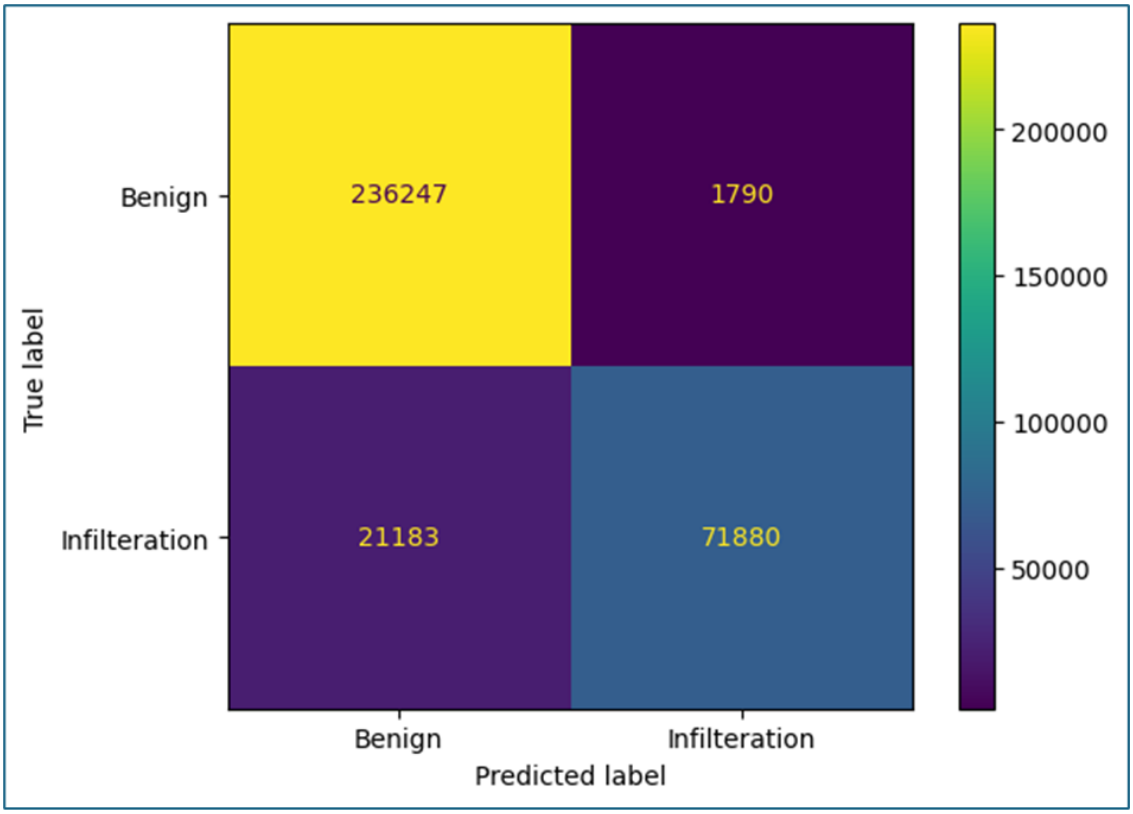


Accuracy: 0.9297704620960435
Precision: 0.9771571522309711
Recall: 0.8805363326862174
F1 Score: 0.9066102062989685

XGBoost was the second ML model that was trained which produced an accuracy of 93.06%, precision value of 97.57%, recall value of 88.24%, and an F1 score of 90.79%. Figure 24 below shows the confusion matrix for XGBoost.

Figure 24

Confusion Matrix for the Trained XGBoost Model

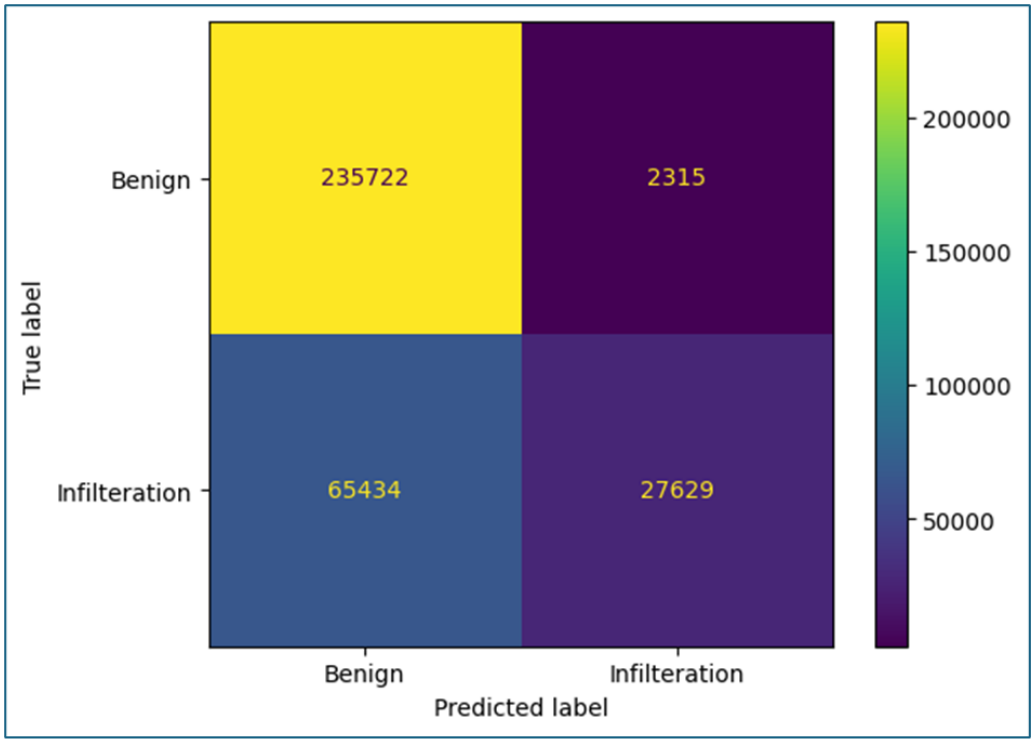


Accuracy: 0.9306161280579885
Precision: 0.9757024569024026
Recall: 0.8824300806460095
F1 Score: 0.9079252401512212

KNN was the third ML model that was trained which produced an accuracy of 79.53%, precision value of 66.09%, recall value of 64.35%, and an F1 score of 66.17%. Figure 25 below shows the confusion matrix for KNN model.

Figure 25

Confusion Matrix for Trained KNN Model



Accuracy: 0.7953820598006645
Precision: 0.66090708305635
Recall: 0.6435797635481414
F1 Score: 0.661788790197557

4.4 Machine Learning Model Development

This section discusses ML model development and testing results based on data preprocessing analysis, such as choosing the best set of parameters from the hyperparameter tuning process.

4.4.1 Machine Learning Model Hyperparameter Tuning

This praxis followed the hyperparameter tuning for preparing the ML models examined. To perform an exhaustive test of various combinations of hyperparameters and to identify the ideal combination that performs best, the '*GridSearchCV*()' method, which

is part of the *Sklearn* Python library was used. Table 5 outlines the optimal hyperparameter values chosen for evaluating each of the three ML models being investigated.

Table 5

Hyperparameter Tuning Results for RF, XGBoost, and KNN

ML Model Name	Hyperparameter	Set of Values	Selected Values
Random Forest	n_estimators	20, 50, 100, 150, 200	100
	max_features	auto, sqrt, log2	auto
	max_samples	0.5, 0.75, 1	0.5
	max_depth	5, 7, 9, 11, 13, 15, 17	9
	criterion	gini, entropy	gini
Extreme Gradient Boosting	n_estimators	20, 50, 100, 150, 200	200
	max_depth	3, 6, 9, 12, 15	9
	learning_rate	0.01, 0.05, 0.1	0.1
	reg_alpha	0, 0.1, 0.5, 1	0.5
	reg_lambda	0, 0.1, 0.5, 1	1
k-Nearest Neighbors	n_neighbors	2, 3, 4, 5, 7, 9	3
	weights	uniform, distance	uniform
	algorithm	auto, ball_tree, kd_tree, brute	auto
	leaf_size	10, 20, 30, 40, 50	30
	p	float, default=2	2
	metric	str, callable	str
	metric_params	dict, default=None	default=None
	n_jobs	1, -1	-1

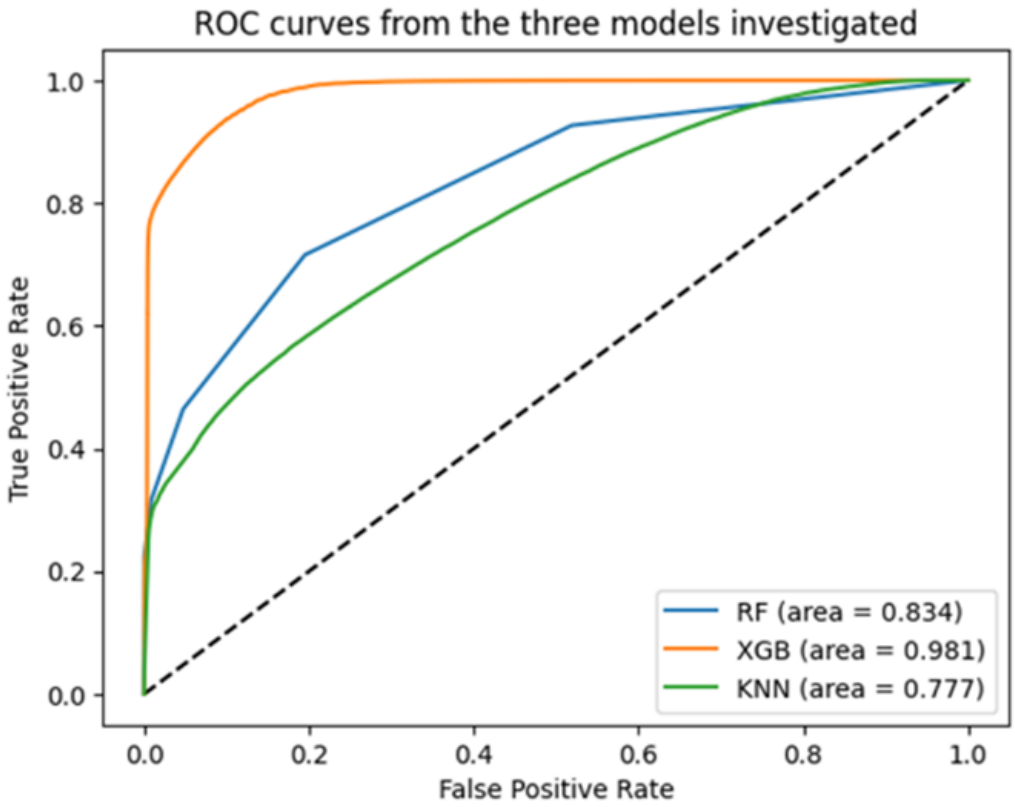
Unlike RF and XGBoost, KNN is not an ensemble ML model; rather, it is a supervised learner that memorizes from the training data set inputs and produces outcomes to learn the function. When test data is provided it predicts the outcome based on the learned function.

4.4.2 Performance Testing

Performance measurement for the RF and XGBoost ML models being evaluated in this praxis was done using the top seven (7) features of the testing data set that were identified from the feature selection process and the KNN ML model was evaluated using the top three (3) features. These models were configured using the hyperparameters that were determined to be optimal during the tuning step. From the results obtained, it is evident that the XGBoost model outranked RF and KNN in terms of accuracy, recall, F1 score, False Positive Rate (FPR), and Area Under the Curve of the Receiver Operator Characteristic (AUC-ROC) metrics. The precision for the RF model, 99.30% was only 0.07% higher than the precision value of the XGBoost ML model which was at 99.23%. The remainder of this section focuses on the test results of the three ML models investigated. The AUC-ROC curves for the three models investigated are plotted using Python's *sklearn* library's *AUC-ROC* curve function. Figure 26 below shows the AUC-ROC curves for the three ML Models examined.

Figure 26

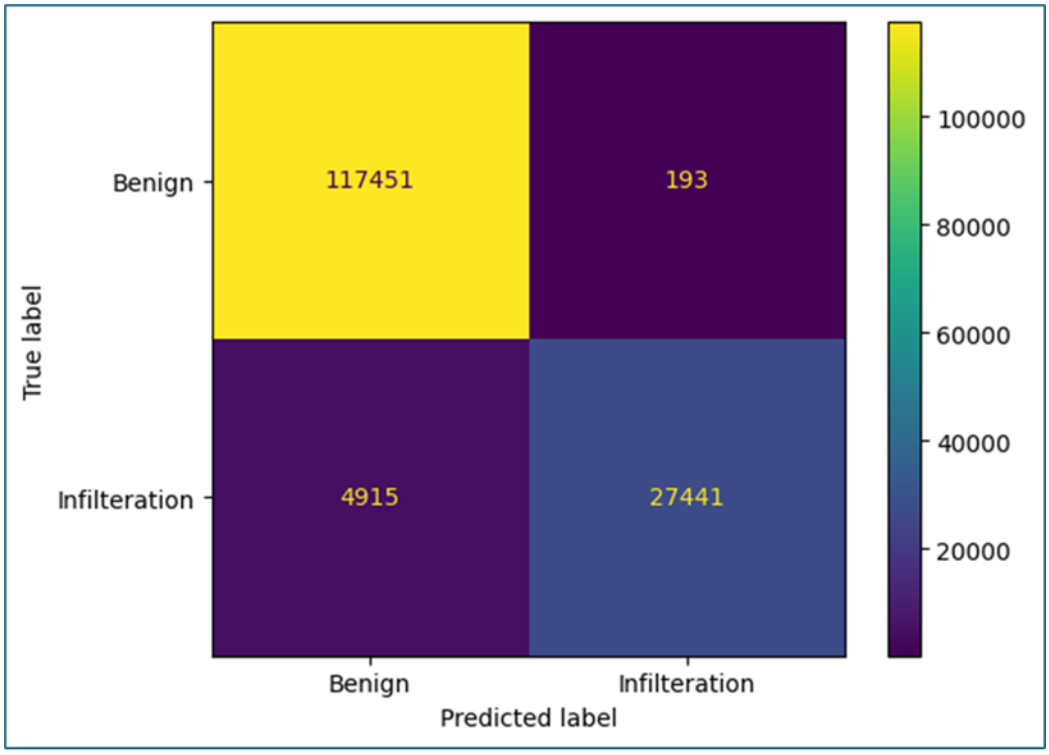
AUC-ROC curves for KNN, RF, and XGBoost



The confusion matrices in Figures 27, 28 and 29 below show the performance results of RF, XGBoost and KNN models, respectively. The RF and XGBoost ML models were evaluated using top seven (7) features and by randomly selecting 150,000 records from a test data set comprising of total of 613,071 records whereas the KNN model was testing using the top three (3) features and by using 150,000 records that were randomly selected from the same test data set with a total of 613,071 records.

Figure 27

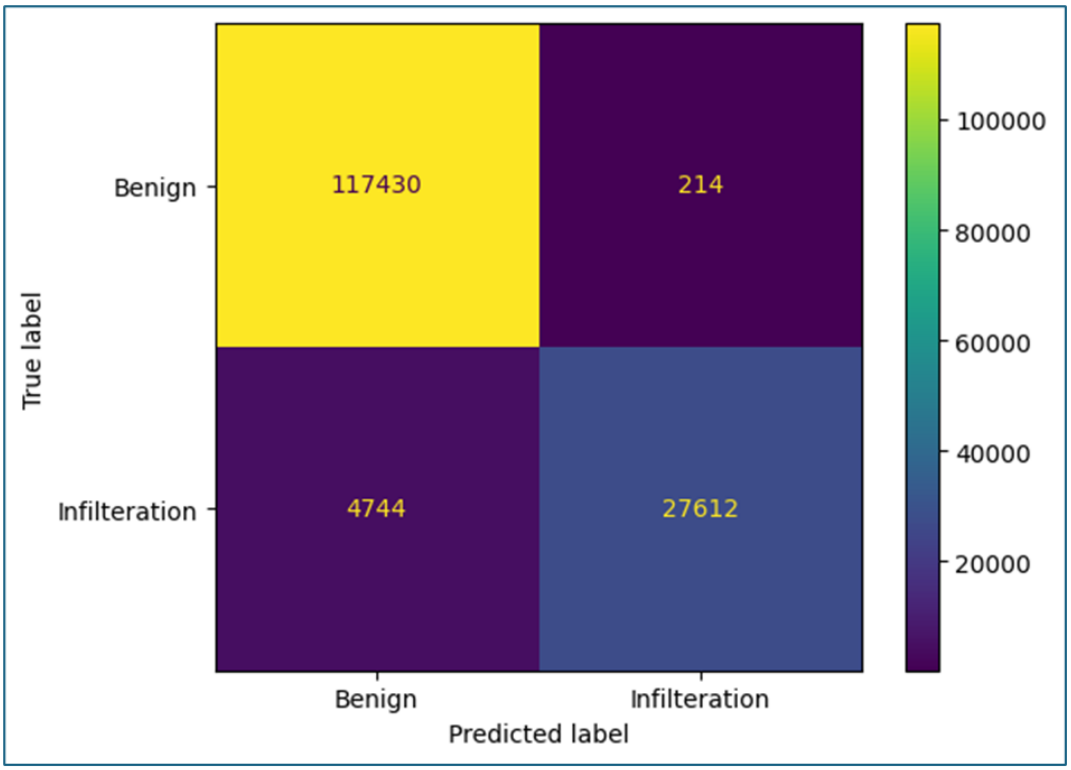
Confusion Matrix Obtained from Tested RF Model



Accuracy: 0.9659466666666666
Precision: 0.9930158500398061
Recall: 0.9232278186717093
F1 Score: 0.9467850144239212

Figure 28

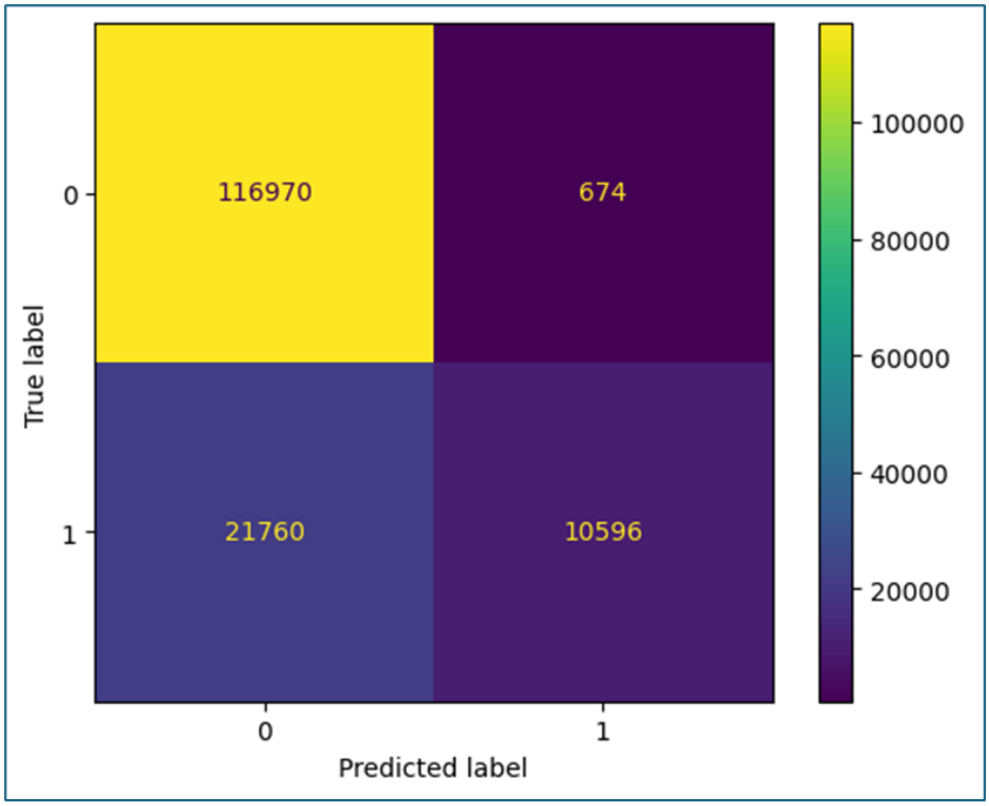
Confusion Matrix Obtained from Tested XGBoost Model



Accuracy: 0.9669466666666666
Precision: 0.9923093509667218
Recall: 0.9257810437896457
F1 Score: 0.9484712759834257

Figure 29

Confusion Matrix Obtained from Tested KNN Model



Accuracy: 0.85044
Precision: 0.66090708358925
Recall: 0.6608763082012465
F1 Score: 0.69913010359961

The confusion matrix for the XGBoost model shown in Figure 28 illustrates performance metrics which are stronger when compared with the metrics shown in confusion matrices for RF and KNN in Figures 27 and 29, respectively.

The XGBoost model was further evaluated using a multi-class label data set with random sample of 314,573 records from the test data set containing 613,071 records to

verify its capability to manage multiple label classes as well. Table 6 describes the mapping of the categorical values for the multi-class Label feature of the data set.

Table 6

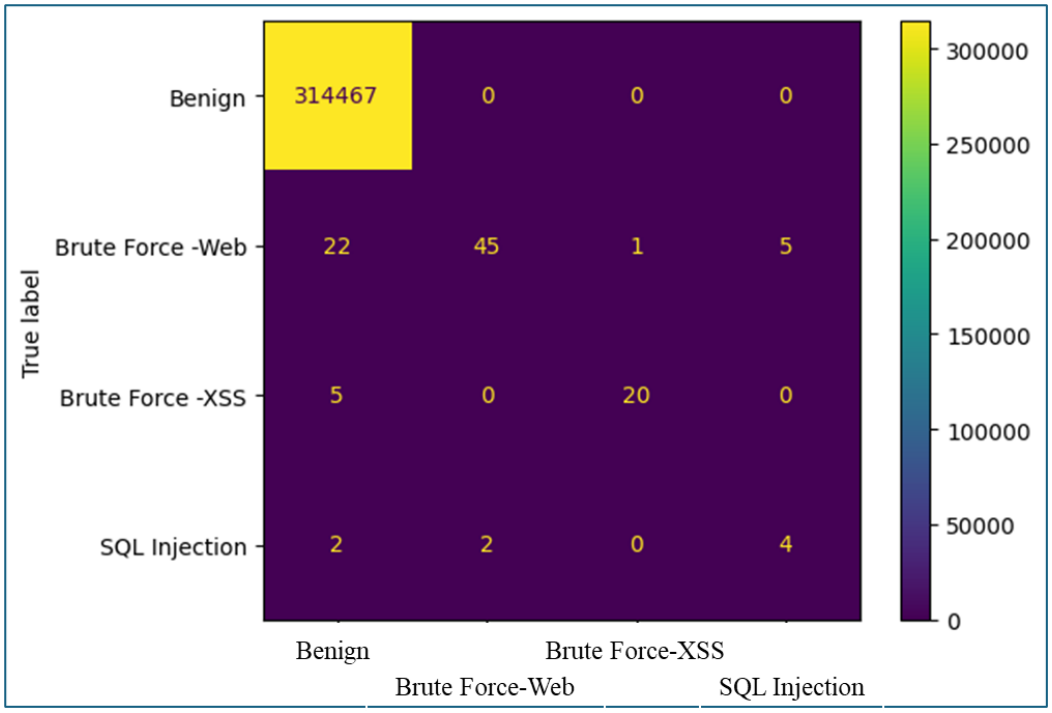
Data Encoding Map for Multi-Class Label Feature

Categorical Attack Type	Numerical Value Assigned
Benign	0
Brute Force – Web	1
Brute Force – XSS	2
SQL Injection	3

The confusion matrix shown in Figure 30 captures the results of this test run. The model produced an accuracy of 99.98%, precision of 72.94%, recall of 72.91%, and F1 score of 77.25%. The model achieved accuracy greater than 99% because of multiple training iterations using larger data sets.

Figure 30

Confusion Matrix for XGBoost with Multiclass Data set



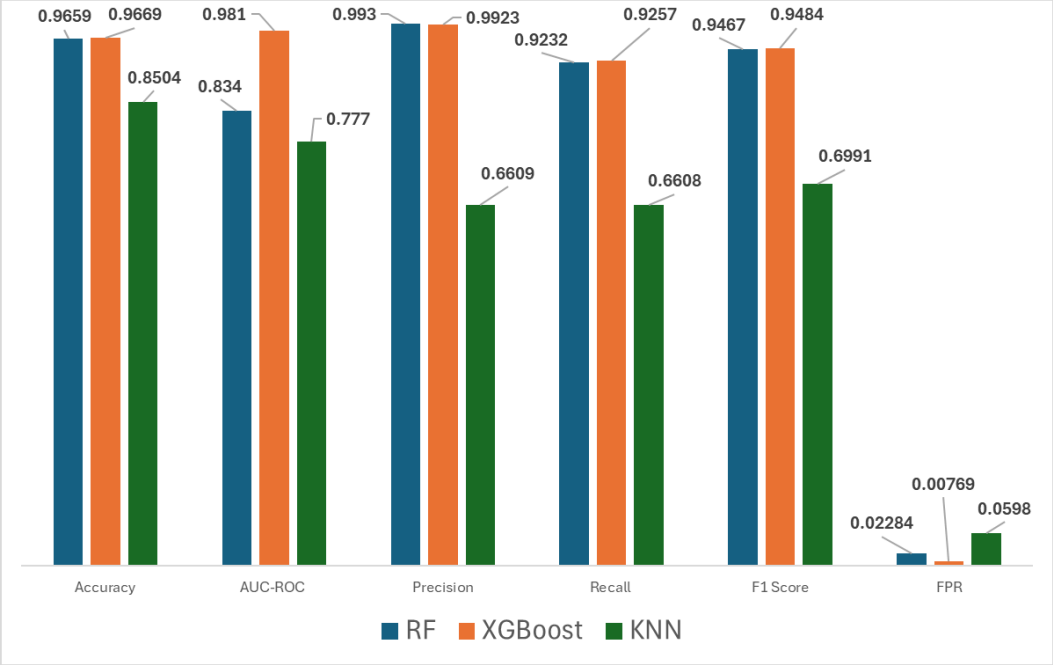
Accuracy: 0.9998823802424238
Precision: 0.72949867103946
Recall: 0.7291095890410959
F1 Score: 0.7725268362611876

4.4.3 Machine Learning Model Performance Testing Results

The goal of praxis, which was to develop an ML-based IDS that detects and classifies cyberattacks on corporate networks, was achieved. Figure 31 shows the performance metrics for the three ML models evaluated.

Figure 31

Bar Chart of Performance Test Results of the ML Models



The accuracy, AUC-ROC, recall and F1 score values for XGBoost outranked RF and KNN ML models. Recall shows true positives predicted correctly out of all positives in the data set. Precision is a metric for the correctness of positive predictions. The classification time metrics (seconds) were captured for the best ML model, XGBoost, which could identify and classify a cyberattack faster than the other two models. Classification time metrics for simulation using the XGBoost ML model are summarized in Table 7.

Table 7

XGBoost ML Model Classification Times (In Seconds)

Classification Time Metric	Cyberattack	No Attack
Minimum	0.000227	0.000049
Maximum	2.932548	0.386009
Average	0.166891	0.042933
Standard Deviation	0.153668	0.069287
Variance	0.023747	0.005518

When the ML-based IDS was deployed on a corporate network, it was to ensure that the time delay needed to notify the user to take appropriate security measures was as short as possible. Detection speed is key to an IDS. A slight margin of milliseconds can make an enormous difference in response time, thus enabling a successful cyberattack to be carried out. The acceptable latency within a corporate network is one hundred milliseconds (Mitchell & Chen, 2014). Wahl (2016) has shown a direct relationship between an IDS's latency period and the number of cyberattacks on an organization. If the XGBoost ML model were to be deployed on-premises, the total system latency would be 142.93 milliseconds.

Chapter 5: Discussion and Conclusions

5.1 Discussion

The central objective of this praxis was to create a machine learning (ML)-based intrusion detection system (IDS) capable of identifying cyberattacks on a corporate network. To structure and guide the research three research questions were established. The proposed ML-based IDS was implemented using simulated network traffic data. The study sought to determine whether an ML-based model could successfully classify malicious network data from benign traffic, which data set features were most relevant for the classification and evaluate which ML model performed the best overall.

5.1.1 Hypothesis 1

As the primary objective of this praxis was to create an ML-based IDS, the first research question examined whether an ML-based IDS is suitable for identifying and classifying intrusion attempts into a corporate network.

RQ1: Can a machine learning model accurately detect and classify a cyberattack on a corporate network?

This question was answered by Hypothesis 1 of this praxis, which claims that an ML-based model can be built to accurately detect and classify cyberattacks against normal traffic in a corporate network.

H1: A machine-learning-based model can accurately detect and classify cyberattack attempts on the corporate network.

During the performance testing phase in this praxis, metrics that were collected helped determine whether it is feasible to build an ML-based model and use it to detect

and classify cyberattacks on corporate networks accurately. Table 7 from Chapter 4 has the summarized results of ML model performance metrics.

RF, XGBoost, and KNN based models demonstrated a classification accuracy greater than 85% and a false positive rate (FPR) under 6%. These metrics confirmed the assertion made in hypothesis 1 that a machine-learning-based model can accurately detect and classify cyberattack attempts on a corporate network.

5.1.2 Hypothesis 2

The second research question in this praxis was to find out whether a model trained using the top seven features of the data set at a minimum could classify cyberattacks with an accuracy of at least 80%.

RQ2: Does an ML model trained using the top seven features of the data set detect cyberattacks with an accuracy of at least 80%?

Hypothesis 2 of this praxis responded to RQ2 by claiming that an ML model trained solely on the top seven features of the data set would detect a cyberattack with an accuracy of at least 80%.

H2: A machine learning model can be built to detect a potential cyberattack solely based on the top seven features with an accuracy of at least 80%.

The inbuilt functions that generate the feature importance scores available within the three (3) ML models, RF, XGBoost, and KNN evaluated during this praxis were used to identify features of the data set that have the most significant impact on the outcome of the classification results. The results from the Feature Importance Score demonstrated that features which contained the values for 1) flow duration, 2) total length of the forward flowing packets, 3) total length of the backward flowing

packets, 4) mean value of the forward packet length, 5) maximum value of the forward packet length, 6) mean value of the backward packet length, and 7) value of the total number of backward packets were sufficient to classify normal network traffic against infiltration with an accuracy of at least 80%. The impact of the number of features selected on accuracy is shown in Figure 18 of Chapter 4. The accuracy of RF and XGBoost models peaked at seven features and stayed approximately around the same percentage. The accuracy of the KNN model reached its peak with three (3) features and then significantly dropped.

5.1.3 Hypothesis 3

The third research question elaborated upon RQ1 and determined which was the best ML model out of the three models investigated.

RQ3: Which of the ML algorithms RF, KNN, or XGBoost can most accurately detect a potential cyberattack on a corporate network?

Hypothesis 3 of this praxis claimed that an ML-based model using XGBoost algorithm would most accurately detect cyberattacks on a corporate network.

H3: An ML-based model developed using XGBoost will most accurately detect cyberattacks on a corporate network.

Figure 31 from Chapter 4 has the summary of performance metrics, determined which of the three examined ML models satisfied the claim made in this hypothesis. The XGBoost model produced an accuracy of 96.69%, precision value of 99.23%, recall value of 92.57%, and an F1 score of 94.84% which substantiated the claim made in H3 of this praxis. Table 8 below summarizes the hypotheses test results.

Table 8

Summary of Hypotheses Testing

Hypothesis	Description	Test Result
H 1	A machine-learning-based model can accurately detect and classify cyberattack attempts on the corporate network	Verified
H 2	A machine learning model can be built to detect a potential cyberattack solely based on the top seven features with an accuracy of at least 80%.	Verified
H 3	An ML-based model developed using XGBoost will most accurately detect cyberattacks.	Verified

5.2 Conclusion

The three hypotheses postulated in this praxis were proven with sufficient research, testing and analysis performed. Hypothesis 1 proposed that a machine learning model is capable of accurately identifying potential cyberattacks on corporate networks. The analysis revealed that all three ML-based models investigated during this praxis were effective in identifying malicious network traffic with varying levels of accuracy and the values of other metrics that were considered for model performance. Hypothesis 2 asserted that an ML model that was developed only using the top seven features of the data set could effectively detect cyberattacks with an accuracy of at least 80%. This praxis verified that all the three ML models examined had an accuracy which is greater than 85%. The final hypothesis stated that XGBoost is the ML model that can classify network traffic with highest accuracy among the three models examined. Praxis identified that XGBoost has the highest accuracy and outperformed the other two models in terms of recall F1 score, and FPR.

Praxis confirmed that RF, XGBoost, and KNN ML-based IDS could accurately detect and classify cyberattacks on a corporate network. The protocol, destination port, total number of packets flowing backward, total number of packets flowing forward, duration of the flow, total length of packets flowing backward, and total length of packets flowing forward, features, which were the top seven features of the data set, could detect and classify cyberattacks with an accuracy greater than 85%.

5.3 Research Contributions

This praxis evaluated ML-based IDS systems and achieved accuracy, AUC-ROC, Precision, Recall, F1 Score, and FPR results that were on par with those captured previously by researchers investigating other ML models and provided an IDS solution that is applicable both for an on-premises network as well as cloud. This praxis used the feature importance score functionality of RF, XGBoost, and KNN algorithms to obtain the top features in ranked order that are essential for model building and classification by comparison of metrics obtained from performance testing of RF, XGBoost, and KNN ML models. The model demonstrated its capability for successfully testing multiple class label data sets. The data set used during this praxis is available on the Canadian Institute of Cybersecurity (CIC) website and the Python code developed for the ML model is available upon request from the author.

5.4 Recommendations for Future Work

The ML model developed in this praxis is capable of classifying binary labeled data sets as well as data sets labeled with multiple classes. The ‘Label’ feature in the data sets used for training and testing the three (3) ML models consisted of two (2) classifications which categorize the network traffic as either benign or infiltration traffic. The second

test data set used for evaluating the multi-class classification capability of the XGBoost model included four classifications namely: Benign, Brute Force – Web, Brute Force – XSS, and SQL Injection. The test data sets used in this praxis did not include DDoS attacks, malware communication, phishing attempts, port scanning, botnets and ransomware attack attempts which are the other major types of attack vectors adopted by cyber criminals. Present-day cyber attackers are equipped with the same tools and technologies that cybersecurity professionals utilize in their day-to-day operations. The cyber attackers are on a continuous mission to develop newer and novel methods to be employed during their cyberattack campaigns.

The machine learning model created during this praxis was evaluated using network traffic data that was not generated by an actual corporate network but by the Canadian Institute for Cybersecurity at the University of New Brunswick (CICUNB) Google LLC (CSE-CIC-IDS2018) (Amazon Web Services [AWS], 2018). Future researchers interested in furthering this ML-based IDS may deploy this model on an actual corporate network for measuring its performance.

References

Al-Rimy, B.A.M., Maarof, A.S., & Shaid, S.Z.M. (2018). Ransomware threat success factors, taxonomy, and countermeasures: A survey and research. *Computers & Security*, *74*, 144–166. https://doi.org/10.1016/j.cose.2018.01.001

Amazon Web Services (AWS). (2018). *A realistic cyber defense data set (CSE-CIC-IDS2018)*. Opendata.aws. Retreived [2/2/2024] from https://registry.opendata.aws/cse-cic-ids2018/

Andrade C. Z Scores, Standard Scores, and Composite Test Scores Explained. Indian J Psychol Med. 2021 Nov;43(6):555-557. doi: 10.1177/02537176211046525. Epub 2021 Oct 10. PMID: 35210687; PMCID: PMC8826187.

Aydin, Z. E., & Ozturk, Z. K. (2021, March). Performance analysis of XGBoost classifier with missing data. In 1st Int. Conf. Comput. Mach. Intell., no. Istanbul, Turkey.

Bakro, M.K., Albrah, R.R., Ashraf, A., Ahmed, Z., Shameem, M.N., Abdelsalm, M.N. (2023). An improved design for a cloud intrusion detection system using hybrid features selection approach with ml classifier. *IEEE Access, 11*, 64228–64247. http://doi.org/10.1109/ACCESS.2023.3289405

Balabanau, A. (2021). *How to recognize and successfully resist fileless malware threats*. IEEE.org. https://www.computer.org/publications/tech-news/trends/resist-fileless-malware-threats

Basu A., Warzel D., Eftekhari A., Kirby J.S., Freymann J., Knable J., Sharma A., Jacobs P. Call for Data Standardization: Lessons Learned and Recommendations in an Imaging Study. JCO Clin Cancer Inform. 2019 Nov;3:1-11. doi: 10.1200/CCI.19.00056. PMID: 31834820; PMCID: PMC6939740.

Brown J.B. Classifiers and their Metrics Quantified. Mol Inform. 2018 Jan;37(1-2):1700127. doi: 10.1002/minf.201700127. Epub 2018 Jan 23. PMID: 29360259; PMCID: PMC5838539.

Carmona, P., Climent, F., Momparler, A., (2019) Predicting failure in the U.S. banking sector: An extreme gradient boosting approach, International Review of Economics & Finance, Volume 61, 2019, Pages 304-323, ISSN 1059-0560, https://doi.org/10.1016/j.iref.2018.03.008.

Chakkaravarthy, S.S., Sangeetha, D., Cruz, M.V., Vaidehi, V., & Raman, B. (2020). Design of intrusion detection honeypot using social leopard algorithm to detect iot ransomware attacks. IEEE Access, 8, 169944–169956. https://doi.org/10.1109/ACCESS.2020.3023764

Chakraborty, S., Pandey, S.K., Maity, S., Dey, L., (2023) Detection and Classification of Novel Attacks and Anomaly in IoT Network using Rule based Deep Learning Model. https://arxiv.org/pdf/2308.00005

Chauhan, P.S., &. Kshetri, N. (2023). Ransomware as a service kit: A novel cybercrime strategy to monetize victims' data. *Computer*, *56*(10), 102–106. 10.1109/MC.2023.3298072 https://doi.org/10.1109/MC.2023.3298072

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794 San Francisco California USA August 13 - 17, 2016.

Chowdhury, R.N., Chowdhury, M.M., Chowdhury, S., Islam, M.R., Ayub, M.A., Chowdhury, A., Kalpoma, K.A. (2020). Parameter optimization and performance analysis of state-of-the-art machine learning techniques for intrusion detection system (IDS). *23rd International Conference on Computer and Information Technology* (ICCIT), Dhaka, Bangladesh, 2020, pp. 1–6. http://doi.org/10.1109/ICCIT51783.2020.9392683

CrowdStrike. (2024). *Threat Hunting Report*. Go.crowdstrike.com. https://go.crowdstrike.com/rs/281-OBQ-266/images/CrowdStrike2024ThreatHuntingReport.pdf?version=0

Damien, W.F., Komninos, N., & Chen, T. (2020). A study on the evolution of ransomware detection using machine learning and deep learning techniques. *IoT*, *1*(2), 551. https://doi.org/10.3390/iot1020030

Deloitte. (2023). *Global cyber executive briefing on insurance*. Deloitte.com. https://www2.deloitte.com/tw/en/pages/risk/articles/insurance.html

Elrawy, M.F., Ali, I.A., & Hamed, H.F.A. (2018). Intrusion detection systems for IoT-based smart environments: a survey. *Journal of Cloud Computing, 7*(1), 1–20. https://doi.org/10.1186/s13677-018-0123-6

Elreedy, D., & Atiya, A.F., A Comprehensive Analysis of Synthetic Minority Oversampling Technique (SMOTE) for handling class imbalance, Information Sciences, Volume 505, 2019, Pages 32-64, ISSN 0020-0255, https://doi.org/10.1016/j.ins.2019.07.070.

Fernando, D.W., Komninos, N., & Chen, T. (2020). A Study on the Evolution of Ransomware Detection Using Machine Learning and Deep Learning Techniques. IoT, 1(2), 551-604. https://doi.org/10.3390/iot1020030

Gazzan, M., Sheldon, F.T. (2023). Opportunities for early detection and prediction of ransomware attacks against industrial control systems. *Future Internet*, *15*(4), 144. https://doi.org/10.3390/fi15040144

Neprash, H.T., McGlave, C.C., Cross, D.A., Virnig, B.A., Puskarich, M.A., Huling, J.D., Rozenshtein, A.Z., Nikpay, S.S. (2022). Trends in ransomware attacks on US hospitals, clinics, and other health care delivery organizations, 2016–2021. JAMA Health Forum, 3(12), e224873. https://jamanetwork.com/journals/jama-health-forum/fullarticle/2799961

Han, Y., Wu, J., Zhai, B., Pan, Y., Huang, G., Wu, L., & Zeng, W. (2019). Coupling a bat algorithm with XGBoost to estimate reference evapotranspiration in the arid and semiarid regions of China. *Advances in Meteorology*, *2019*, 1–16. DOI 10.1155/2019/9575782.

IBM Security Report, 2023 https://www.ibm.com/reports/data-breach

Jadhav, A.D., & Vidyullatha, P. (2021). Highly accurate and efficient two phase-intrusion detection system (TP-IDS) using distributed processing of HADOOP and machine learning techniques. *Journal of Big Data*, *8*(1) https://doi.org/10.1186/s40537-021-00521-y

Kasongo, S.M., Development and evaluation of a deep learning based intrusion detection model for wireless networks (2020), https://hdl.handle.net/10210/418298

Khan, S.A., Ahmad, Z., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2020). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies, 32(1). https://doi.org/10.1002/ett.4150

Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J. (2019). Survey of intrusion detection systems: Techniques, data sets and challenges. *Cybersecurity*, *2*(1). https://cybersecurity.springeropen.com/articles/10.1186/s42400-019-0038-7

L'Heureux, A., Grolinger, K., Elyamany H.F., & Capretz, M.A.M. (2017) "Machine Learning With Big Data: Challenges and Approaches," in IEEE Access, vol. 5, pp. 7776-7797, 2017, doi: 10.1109/ACCESS.2017.2696365.

Latha, S., & Prakash, S.J. (2017). A survey on network attacks and intrusion detection systems. *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*. IEEE, 2017, pp. 1–7. doi: 10.1109/ICACCS.2017.8014614

Lee, TH., Ullah, A., Wang, R. (2020). Bootstrap Aggregating and Random Forest. In: Fuleky, P. (eds) Macroeconomic Forecasting in the Era of Big Data. Advanced Studies in Theoretical and Applied Econometrics, vol 52. Springer, Cham. https://doi.org/10.1007/978-3-030-31150-6_13

Li, M., Hengyang S., Yanjun H., Hong C.. 2024. Shapley value: From cooperative game to explainable artificial intelligence. Autonomous Intelligent Systems 4, (1) (12): 2, https://www.proquest.com/scholarly-journals/shapley-value-cooperative-game-explainable/docview/2924116820/se-2

Liu, H., Zhong, C., Alnusair, A., Sheikh R.I. FAIXID: A Framework for Enhancing AI Explainability of Intrusion Detection Results Using Data Cleaning Techniques. J Netw Syst Manage 29, 40 (2021). https://doi.org/10.1007/s10922-021-09606-8

Liyang W., Dantong N., Xinjie Z., Xiaoya W., Mengzhen H., Huilian C., A Comparative Analysis of Novel Deep Learning and Ensemble Learning Models to Predict the Allergenicity of Food Proteins (2021), https://doi.org/10.1101/2021.03.10.434710

Maxwell, P., Alhajjar, E., & Bastian, N.D. (2019, December). Intelligent feature engineering for cybersecurity. *2019 IEEE International Conference on Big Data*, (pp. 5005–5011). IEEE.

Microsoft Digital Defense Report (MDDR). (2023). *Microsoft digital defense report 2023*, Microsoft.com. https://www.microsoft.com/en-us/security/security-insider/microsoft-digital-defense-report-2023

Mijalkovic, J., & Spognardi, A. Reducing the False Negative Rate in Deep Learning Based Network Intrusion Detection Systems. Algorithms 2022, 15, 258. https://doi.org/10.3390/a15080258

Nargesian, F., Samulowitz, H., Khurana, U., Khalil, E. B., & Turaga, D. S. (2017). Learning feature engineering for classification. *IJCAI*, *17*, 2529–2535.

National Institure of Standards and Technology – NIST (2023), Information Technology Laboratory, Computer Security Resource Center, cyber resiliency, https://csrc.nist.gov/glossary/term/cyber_resiliency

Nopour, R., Shanbehzadeh, M., & Kazemi-Arpanahi, H. (2021). Developing a clinical decision support system based on the fuzzy logic and decision tree to predict colorectal cancer. Medical Journal of the Islamic Republic of Iran, 35, 44–44. https://doi.org/10.47176/mjiri.35.44

O'Kane, P., Sezer, S., & Carlin, D. (2018), Evolution of ransomware. *IET Network, 7*, 321–327. https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/iet-net.2017.0207

Ongun, T. (2023). *Resilient machine learning methods for cyber-attack detection* [Doctoral dissertation, Northeastern University]. ProQuest Dissertations & Theses (30418436). http://www.proquest.com/dissertations-theses/resilient-machine-learning-methods-cyber-attack/docview/2799322242/se-2

Raju, V.N.G., Lakshmi, K.P., Jain, V.M., Kalidindi, A. and Padma, V. "Study the Influence of Normalization/Transformation process on the Accuracy of Supervised Classification," 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 2020, pp. 729-735, doi: 10.1109/ICSSIT48917.2020.9214160.

Rawass, J. (2019). Cybersecurity Strategies to Protect Information Systems in Small Financial Institutions. ProQuest Dissertations & Theses. https://www.proquest.com/docview/2272840761/abstract/?pq-origsite=primo&_oafollow=false

Mitchell, R., & Chen, I-R. 2014. A survey of intrusion detection techniques for cyber-physical systems. ACM Comput. Surv. 46, 4, Article 55 (April 2014), 29 pages. https://doi.org/10.1145/2542049

Sahoo, K., Samal, A.K., Pramanik, J., Subhndu, K.P. (2019). Exploratory data analysis using python. *International Journal of Innovative Technology and Exploring Engineering, 8*(12), 4727–4735. https://doi.org/10.35940/ijitee.l3591.1081219

Savage, K., Coogan, P., Lau. H. (2015). *The evolution of ransomware*, Symantec. https://docs.broadcom.com/docs/the-evolution-of-ransomware-15-en

Selifanov, V.V., Maksudov, M.M., Doroshenko, I.E. & Titov, D.N. (2022). Methodology for the synthesis of acceptable options for organizational functional structure of the security management system of a significant object of critical information infrastructure. *IEEE 23rd International Conference of Young Professionals in Electron Devices and Materials* (EDM). Altai, Russian Federation, 2022, pp. 642–645, https://doi.org/10.1109/EDM55285.2022.9855036

Sharafaldin, I., Lashkari, A.H., & Ghorbani, A.A. (2018). Toward generating a new intrusion detection data set and intrusion traffic characterization. *4th International Conference on Information Systems Security and Privacy* (ICISSP). Portugal, January 2018.

Sheridan, R.P., Wang, W.M., Liaw, A., Ma, J., Gifford, E.M. (2016). Extreme gradient boosting as a method for quantitative structure–Activity relationships. *Journal of Chemical Information and Modeling, 56*(12), 2353–2360. https://pubs.acs.org/doi/10.1021/acs.jcim.6b00591

Singh, J., Singh, J., Singh, G., and Kaur, N. "Exploratory Data Analysis for Interpreting Model Prediction using Python," 2022 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON), Bangalore, India, 2022, pp. 1-6, doi: 10.1109/SMARTGENCON56628.2022.10083533.

Sonicwall (2023). *2023 Sonicwall cyber threat report*. Sonicwall.com https://www.sonicwall.com/2023-cyber-threat-report/

Yadav, S.M., & Kalpana, R. (2019). Data preprocessing for intrusion detection system using encoding and normalization approaches. *11th International Conference on Advanced Computing (ICoAC)*, Chennai, India, 2019, pp. 265–269, http://doi.org/10.1109/ICoAC48765.2019.246851

Stevens, F.R., Gaughan, A.E., Linard, C., & Tatem, A.J. (2015). Disaggregating census data for population mapping using random forests with remotely sensed and ancillary data. https://doi.org/10.1371/journal.pone.0107042

Su, J., Liu, J., Wang, X., & Wang, W. (2019). Detecting Android locker-ransomware on Chinese social networks. *IEEE Access*, *7*, 20381–20393. https://doi.org/10.1109/ACCESS.2018.2888568

Sun, X., Liu, P., & Singhal, A. (2018). Toward cyberresiliency in the context of cloud computing [resilient security]. *IEEE Security & Privacy*, *16*(6), 71–75, https://doi.org/10.1109/MSEC.2018.2882122

Suangli, S., Fahmi, F., and Zamzami, E.M. "Performance Analysis of Support Vector Machine and Xgboost Classifier Algorithms in Predicting Data Heart Disease," 2023 29th International Conference on Telecommunications (ICT), Toba, Indonesia, 2023, pp. 1-6, doi: 10.1109/ICT60153.2023.10374048.

Suramwar, M.V., & Bansode, S.M. (2015). A Survey on different types of Intrusion Detection Systems. *International Journal of Computer Applications, 122*(16), 34–38. https://doi.org/10.5120/21788-5097

Tan, Z.-F., Li, H.-H., Ju, L.-W., & Tan, Q.-K. (2018). Joint scheduling optimization of virtual power plants and equitable profit distribution using shapely value theory. *Mathematical Problems in Engineering, 2018,* Article 3810492. https://doi.org/10.1155/2018/3810492

Tait, K.-A., Khan, J.S., Alqahtani, F., Shah, A.A., Ali Khan, F., Rehman, M.U., Boulila, W., & Ahmad, J. (2021). Intrusion detection using machine learning techniques: An experimental comparison. *2021 International Congress of Advanced Technology and Engineering (ICOTEN)*. 4–5 July 2021, IEEE. 10.1109/ICOTEN52080.2021.9493543

Tharwat, A. (2020). Classification assessment methods. Applied Computing and Informatics, 17(1), 168-192. https://www.emerald.com/insight/content/doi/10.1016/j.aci.2018.08.003

Upadhyay, D., Manero, J., Zaman, M., and Sampalli, S. (2021) "Gradient Boosting Feature Selection With Machine Learning Classifiers for Intrusion Detection on Power Grids," in IEEE Transactions on Network and Service Management, vol. 18, no. 1, pp. 1104-1116, March 2021, doi: 10.1109/TNSM.2020.3032618

Venturini, M., Freda, F., Miotto, E., Giaretta, A., & Conti, M. (2023). Differential area analysis for ransomware: Attacks, countermeasures, and limitations. *ArXiv, 2023*, Article 17351. https://doi.org/10.48550/arxiv.2303.17351

Venu, K., Natesan, P., Krishnakumar, B., Rajasekar, V., Sasipriya, N., & Swathi, V. (2022). EEG signal classification for motor imagery tasks. *2022 International Conference on Computer Communication and Informatics* (ICCCI)

Verma, A., Ranga, V. (2020). Machine learning based intrusion detection systems for IoT applications., *111*, 2287–2310 https://doi.org/10.1007/s11277-019-06986-8

Vinayakumar, R., Alazab, K.P. Soman, P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep learning approach for intelligent intrusion detection system. *IEEE Access, 7*, 41525–41550. https://doi.org/10.1109/ACCESS.2019.2895334

Wahl, R.S. (2016). Latency in intrusion detection systems (IDS) and cyberattacks: A quantitative comparative study. https://www.semanticscholar.org/paper/Latency-in-intrusion-detection-systems-(IDS)-and-A-Wahl/6c892d1c66706dff8c5ed074d3d4ef99d9eda00b

Wang, Y., Wu, X., & Chen, H. (2016). An intrusion detection method for wireless sensor network based on mathematical morphology. Security and Communication Networks, 9(15), 2744–2751. https://doi.org/10.1002/sec.1181

Wang J., & Zhou S. CS-GA-XGBoost-Based Model for a Radio-Frequency Power Amplifier under Different Temperatures. Micromachines. 2023; 14(9):1673. https://doi.org/10.3390/mi14091673

Wolpert, D.H., & Macready, W.G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*. vol. 1, no. 1, pp. 67-82, April 1997, doi: 10.1109/4235.585893

World Economic Forum (2024) 2023 was a big year for cybercrime – here's how we can make our systems safer https://www.weforum.org/agenda/2024/01/cybersecurity-cybercrime-system-safety/

Yang, L., Abdallah, S. On hyperparameter optimization of machine learning algorithms: Theory and practice, Neurocomputing, Volume 415, 2020, Pages 295-316, ISSN 0925-2312, https://doi.org/10.1016/j.neucom.2020.07.061

Yeruva, A.R., Chaturvedi, P., Rao, A.L.N., Dimril, S.C., Shekar C., and Yirga, B., "Anomaly Detection System using ML Classification Algorithm for Network Security," 2022 5th International Conference on Contemporary Computing and Informatics (IC3I), Uttar Pradesh, India, 2022, pp. 1416-1422, doi: 10.1109/IC3I56241.2022.10072303.

Zammani, M., Razali, R., & Singh, D. (2021). Organisational information security management maturity model. *International Journal of Advanced Computer Science and Application*s, http://doi.org/10.14569/IJACSA.2021.0120974

Zhang Z. Introduction to machine learning: k-nearest neighbors. Ann Transl Med. 2016 Jun;4(11):218. doi: 10.21037/atm.2016.03.37. PMID: 27386492; PMCID: PMC4916348.

Zhou, Z.H. (2009). Ensemble Learning. In: Li, S.Z., Jain, A. (eds) Encyclopedia of Biometrics. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-73003-5_293