Arab American University

Faculty of Graduate Studies

Effect of Feature Selection and Machine Learning on IDS Performance

By

Ahmad Taha Ahmad Hamarshe

Supervisor

Dr. Mohammad M. N. Hamarsheh

This thesis was submitted in partial fulfillment of the requirements for the Master`s degree in Cybersecurity

July/ 2024

Thesis Approval

Effect of Feature Selection and Machine Learning on IDS Performance

By

Ahmad Taha Ahmad Hamarshe

This thesis was defended successfully on 10/07/2024 and approved by:

Committee members	Signature
1. Dr. Mohammad M. Hamarsheh: Supervisor	
2. Dr. Ahmad M. Hasasneh: Internal Examiner	
3. Dr. Saeed Salah: External Examiner	

Declaration

Ahmad Taha Ahmad Hamarshe with a university ID 202112750 declares that this thesis, "Effect of Feature Selection and Machine Learning on IDS Performance" is his own original work and that all informational and materials sources have been appropriately acknowledged.

I confirm that the material, facts, and concepts utilized in this thesis have all been properly cited and recognized in accordance with Arab American University citation guidelines.

I provide permission for this thesis to be archived and made accessible by Arab American University for research and educational purposes. Any additional uses or copies of this thesis must have my permission.

Name of the student: Ahmad Taha Ahmad Hamarshe.

University ID: 202112750

Signature: 

Date: 25/01/2025

Dedication

To my beloved parents,

The first teachers in my life, who taught me perseverance and hard work, and have been my greatest support. My dear father, your kindness, love, and unwavering faith gave me strength. My dearest mother, your prayers and guidance illuminated my path. This thesis is the fruit of your sacrifices.

To my dear brother,

My closest friend and first supporter, your encouragement and optimism were my lifeline in my toughest moments. I am grateful to have you by my side.

To my beloved sisters,

The source of love and warmth, always pushing me forward and lightening my burdens with silent care. Your presence made this journey more beautiful.

To my dear friends,

You are my second family. Thank you for your support, patience, and encouragement. This achievement would not be the same without you.

Acknowledgment

First and foremost, I extend my deepest gratitude to my supervisor, Dr. Mohammed Hamarsheh, whose invaluable guidance, insightful suggestions, and unwavering patience have been instrumental throughout my research journey. His expertise and dedication have left a profound impact on the quality and direction of this work, and I am truly fortunate to have had his support.

I would also like to express my heartfelt appreciation to the Department of Cyber Security at the Arab American University. Their continuous support, encouragement, and resources have provided me with a strong foundation to accomplish this work.

A special thanks to my friends and colleagues at the Arab American University, who have been a source of inspiration and motivation. Your companionship and encouragement during the challenges of this journey have made it both memorable and rewarding.

To everyone who has contributed to my academic and personal growth, whether through guidance, encouragement, or support, I offer my sincere thanks. This accomplishment reflects your belief in me.

Abstract

With the increasing sophistication of cyber-attacks and their evolving nature alongside advancements in network infrastructures, Intrusion Detection Systems (IDS) are facing growing challenges. Machine Learning offers a promising approach to efficiently analyze the diverse datasets generated by network traffic. This research investigates the impact of feature selection on enhancing the accuracy and efficiency of IDS. By applying our feature selection model on the CICIDS2017 dataset, we identified the top 10 most relevant features that significantly improve the performance of multiple Machine Learning models. In addition to evaluating our own feature selection methodology, we also applied the feature selection models used in previous studies on our system. The results demonstrate that our approach to feature selection still outperforms these previous models in terms of both accuracy and computational efficiency. Specifically, Random Forest achieved an accuracy of 96.1%, Naive Bayes reached 82.5%, and both AdaBoost and K-Nearest Neighbors (KNN) surpassed 98% accuracy. While K-Nearest Neighbors (KNN) demonstrated excellent accuracy, it required considerably longer computational time compared to the other models. This research emphasizes the role of feature selection in optimizing IDS performance, demonstrating how our approach in selecting the most relevant 10 features enhance detection accuracy while maintaining efficient processing times. Our findings confirm that the feature selection methodology employed in this thesis provides a clear advantage over prior models, improving both detection accuracy and real-time applicability of IDS.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

This chapter explores the growing importance of communication over the Internet and its impact on individuals and organizations. The section aims to highlight the increasing reliance on online platforms, emphasizing the motivation behind this study.

1.1 Motivation

Every day, millions of individuals and hundreds of thousands of organizations engage in communication over the Internet. The number of Internet users has surpassed 5 billion, indicating the pervasive influence and importance of the Internet in modern society (*Title in English: The Effect of Different Features and Algorithms on the Performance of the IDS Comparison and Analysis*, no date; Internet World Stats, 2024). As the Internet becomes more integral to personal, professional, and governmental functions, the need for robust cybersecurity measures has never been more critical. The estimated global cost of cybercrime is expected to soar to $23.84 trillion by 2027, a significant increase from $8.44 trillion in 2022, as projected by Statista. Notably, 2023 witnessed several significant cyberattacks, one of which targeted the US State Department (World Economic Forum, 2024). As shown in Figure (1), the rapid increase in the number of Internet users demonstrates the continuous expansion of the digital landscape.

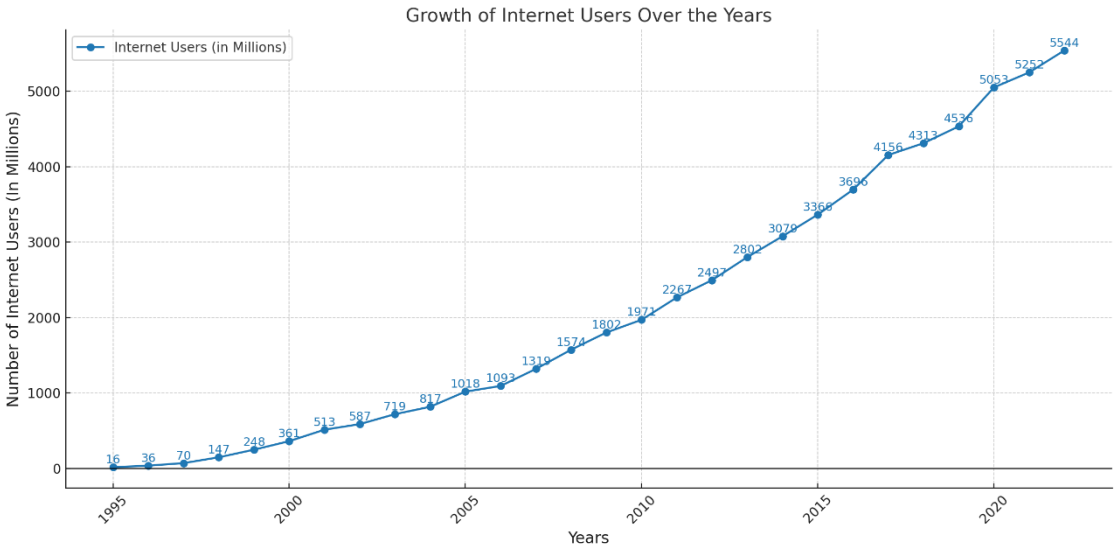

Figure 1. Internet growth (1995-2022) (Internet World Stats, 2024)

According to a report by Cybersecurity Ventures, cybercrime is predicted to inflict damages totaling $10.5 trillion annually by 2025, making it the third-largest economy in the world after the US and China (Steve Morgan, 2020). The rise in the Internet usage is accompanied by an increase in cyber-attacks, targeting sensitive information, disrupting services, and causing significant financial losses. In fact, studies show that a business falls victim to a ransomware attack every 11 seconds (Steve Morgan, 2019).

To safeguard information security, two primary methods are employed for attack detection: signature-based identification and anomaly-based detection.

Signature-based techniques rely on pre-established databases to identify attacks. These systems compare incoming data against known attack patterns (signatures) to detect threats. While generally effective, these databases require continual updates to remain relevant and to accommodate new attack patterns. However, they remain susceptible to zero-day attacks—previously unseen threats not cataloged in the database. Consequently, such attacks evade detection. Despite their limitations, signature-based systems are widely used due to their simplicity and effectiveness against known threats (Leung and

Leckie, 2005).

On the other hand, anomaly-based detection scrutinizes network flow to identify unusual behaviors. This approach has demonstrated efficiency in detecting novel attacks, including zero-day threats. Anomaly-based methods establish a baseline of normal network behavior and flag deviations from this norm as potential threats. This makes them particularly useful in identifying previously unknown attacks that signature-based methods might miss (Leung and Leckie, 2005).

In the contemporary digital landscape, a substantial proportion of internet traffic is encrypted using SSL/TLS (Secure Sockets Layer/Transport Layer Security) protocols, a trend that continues to escalate (Sharafaldin *et al.*, 2018). The pervasive encryption of internet traffic presents a formidable challenge for traditional signature-based detection mechanisms, which rely on inspecting packet contents for identifying malicious activity. However, emerging anomaly-based approaches offer a promising solution by scrutinizing network data based on intrinsic characteristics such as packet size, connection duration, and packet frequency. Unlike signature-based methods, anomaly-based detection does not necessitate decrypting encrypted traffic, enabling effective analysis of encrypted protocols. Consequently, anomaly-based detection methodologies are gaining prominence in the battle against network attacks, owing to their ability to operate effectively in encrypted environments without compromising privacy or security (Bhuyan, Bhattacharyya and Kalita, 2013a).

Recent advancements in Machine Learning (ML) have opened new avenues for enhancing the performance of Intrusion Detection Systems (IDS). Machine Learning algorithms can process vast amounts of data, learn from it, and improve detection

accuracy over time. They can identify complex patterns and relationships in data that traditional methods might overlook. Integrating ML with IDS can potentially offer faster and more accurate detection of network anomalies, thus providing a robust defense against evolving cyber threats (Buczak and Guven, 2015; Sarker *et al.*, 2020).

Furthermore, the use of Machine Learning in IDS has proven to reduce false positive rates by up to 90% in certain applications, providing more reliable and actionable insights for cybersecurity professionals (Shah and Issac, 2018).According to a study by Accenture (*68% of business leaders feel that their cybersecurity risks are increasing* , no date), 68% of business leaders feel their cybersecurity risks are increasing. This is further compounded by the fact that it takes an average of 280 days to identify and contain a breach, costing businesses on average $3.86 million (*What is the cost of a data breach in 2020? - IBM Z and LinuxONE Community*, no date). The pressing need for improved IDS solutions is clear.

This thesis investigates the performance of Intrusion Detection Systems (IDS) by analyzing the impact of various features and algorithms on key metrics, including execution time, accuracy, and efficiency. The goal of this research is to provide insights into optimizing IDS performance by comparing the effectiveness of different feature-algorithm combinations. Through rigorous analysis and experimentation, this study aims to identify the most efficient and reliable features that can enhance IDS performance. Consequently, this research contributes to the development of more robust, adaptable security systems, capable of safeguarding against both known and unknown threats.

1.2 Research Questions and Significance

This thesis investigates the performance of Intrusion Detection Systems (IDS) by

analyzing the impact of various features and algorithms on key metrics, including execution time, accuracy, and efficiency. The goal of this research is to provide insights into optimizing IDS performance by comparing the effectiveness of different feature-algorithm combinations. Through rigorous analysis and experimentation, this study aims to identify the most efficient and reliable features that can enhance IDS performance. Consequently, this research contributes to the development of more robust, adaptable security systems, capable of safeguarding against both known and unknown threats.

The significance of this thesis lies in its potential to improve the effectiveness of IDS by leveraging advanced feature selection methods and Machine Learning techniques. By optimizing the performance of IDS, this study aims to provide a more reliable defense mechanism against cyber threats, thereby enhancing the overall security posture of organizations and individuals alike.

1.3 Scope of Study

The scope of this thesis encompasses the following aspirations:

- Investigating a range of Machine Learning algorithms suitable for the identification of network anomalies.
- Focusing on key metrics such as execution time, accuracy, and efficiency to evaluate IDS performance.
- Comparing different Feature-algorithm combinations to identify the most efficient and reliable options.
- Aiming to optimize IDS performance by providing insights into feature-algorithm combinations that enhance effectiveness.

- Contributing to the development of more robust and adaptable security systems capable of safeguarding against known and unknown threats
- Making a meaningful contribution to the academic discourse by achieving results consistent with previous studies in the realm of network anomaly detection.
- Evaluating the suitability of selecting common feature techniques in improving the performance of IDSs.

1.4 Thesis Outline

After the introductory chapter, Chapter Two introduces concepts and background aims to provide foundational information crucial for comprehending the study, including details on datasets, anomaly and attack classifications, in-depth descriptions of attacks, and Machine Learning algorithms employed. Chapter Three provides an overview of relevant literature, detailing similar studies conducted in the field. Chapter Four explains the methodology and it is divided into two sections. The first section is about, the tools and methods, outlines the software and hardware utilized in the study, as well as detailing the methods employed for performance evaluation. The second subsection, Implementation, encompasses the following steps: data preprocessing, partitioning of data into training and testing sets, implementation of feature selection techniques, and application of Machine Learning algorithms. Chapter Five is titled as the Results and Discussion, presents the findings obtained during the implementation phase, along with an in-depth analysis of their cause-and-effect relationships and exploration of alternative approaches. Within this chapter, Evaluation scrutinizes the methodology and results, providing critical assessment and potential limitations of the study's approach. Conclusions and Future

Work in Chapter six states the conclusions drawn from the study's findings and outlines avenues for future research based on the results.

Chapter 2: Background

This chapter covers the basic concepts related to Intrusion Detection Systems (IDS), feature selection, the dataset used, and machine learning algorithms. It provides the necessary background to understand the approach and methods used in this research.

2.1 Intrusion Detection Systems (IDS)

Intrusion Detection Systems (IDS) play a crucial role in safeguarding computer networks and systems from malicious activities. These systems monitor network traffic or system activities for suspicious behavior that may indicate an intrusion attempt. IDS can be categorized into two primary types: signature-based IDS and anomaly-based IDS.

Signature-based IDS relies on a pre-defined database of attack patterns or "signatures." When a new packet or behavior matches a known signature, the system flags it as a potential threat. However, this method is limited by its inability to detect novel attacks or zero-day vulnerabilities, which are not yet cataloged in the signature database. In contrast, anomaly-based IDS does not depend on prior knowledge of attacks. Instead, it establishes a baseline for normal network behavior and flags deviations from this baseline as potential threats. This makes anomaly-based systems more effective in identifying previously unknown threats, though they can sometimes generate higher false positives due to legitimate behavior that deviates from the norm (Axelsson, 2000) . IDS is critical in environments where confidentiality, integrity, and availability of information are of utmost importance. Organizations use IDS to complement firewalls, encryption, and other security measures. With the increasing sophistication of cyberattacks, modern IDS often integrate Machine Learning techniques to improve their detection capabilities. Machine Learning enables IDS to learn from data, improve detection accuracy, and reduce false

positive rates over time, making them more adaptive to evolving threats (Sommer and Paxson, 2010).

2.2 Feature Selection

Feature selection is a fundamental step in building efficient IDS, especially when dealing with large datasets that contain numerous features. The goal of feature selection is to identify and retain the most relevant features that contribute to accurate detection while discarding redundant or irrelevant ones. Effective feature selection enhances the IDS's performance by improving its accuracy, reducing computational costs, and minimizing the risk of overfitting (Guyon and De, 2003).

There are three primary methods of feature selection: filter methods, wrapper methods, and embedded methods.

2.2.1 Filter methods

Rank features based on statistical metrics such as correlation or mutual information. These methods are fast and independent of Machine Learning algorithms, but they may not capture feature dependencies effectively.

2.2.2 Wrapper methods

Evaluate subsets of features by training and testing the model multiple times using different combinations. While more accurate, wrapper methods are computationally expensive, especially with large datasets.

2.2.3 Embedded methods

perform feature selection during the training process of the Machine Learning model. These methods, such as Lasso Regression or Decision Trees, tend to strike a balance

between speed and accuracy by embedding feature selection within the model-building process (Chandrashekar and Sahin, 2014).

2.3 Dataset

In the realm of network anomaly detection using Machine Learning methodologies and access to substantial volumes of both malicious and benign network traffic is imperative for effective training and testing procedures. However, due to privacy concerns, the utilization of actual network data in the public domain is often infeasible. To address this challenge, numerous datasets have been developed and are continually being curated. This section provides an overview of several prominent datasets, followed by a comparative evaluation to determine the most suitable option for utilization in the implementation phase.

2.3.1 DARPA 98

This dataset (Haines *et al.*, 2001), created by MIT Lincoln Laboratory with DARPA funding, aims to serve as a training and testing ground for Intrusion Detection Systems (IDS). It mimics the local computer network of the United States Air Force, emulating various activities like File Transfer Protocol (FTP) file transfers, web browsing, email exchanges, and Instant Relay Chat (IRC) messaging. Alongside regular network traffic, it includes 38 attack scenarios categorized into types such as Denial of Service (DoS), User to Remote (U2R), Probe, and Remote to Local (R2L) (Thomas, Sharma and Balakrishnan, 2008).

While criticized for its departure from real-world network behavior, outdatedness, and lack of false positive scenarios (Gharib *et al.*, 2016), the DARPA98 dataset remains influential as a precursor to widely-used datasets like KDD Cup 99 and NSL-KDD.

2.3.2 KDD99

The University of California, Irvine developed this dataset (University of California, 1999) specifically for Intrusion Detection Systems participating in The Third International Knowledge Discovery and Data Mining Tools Competition (The KDD Cup '99). It comprises data packets from the DARPA98 dataset, featuring 21 derived properties obtained through a feature extraction process tailored for Machine Learning applications. The dataset is split into two sections: a training set containing 4.9 million instances and a test set containing 311,029 instances. KDD99 encompasses 38 distinct attack types, with 14 attacks unique to the test section, representing unknown attack scenarios. This design allows for the evaluation of the detection of unknown attacks in the test set.

Compared to the DARPA99 dataset, KDD99 has gained preference in numerous studies due to its enhanced suitability for Machine Learning techniques, featuring a refined feature system and well-structured training and test partitions. While KDD99 served as a notable alternative to DARPA98, its extensive repetitions adversely affected research outcomes and Machine Learning algorithm performance. Moreover, due to its large size, researchers attempted to utilize only parts of it, but random selection failed to capture dataset nuances (Özgür and Erdem, 2016).

2.3.3 NSL-KDD

In 2009, Tavallaee et al. (Tavallaee *et al.*, 2009) introduced NSL-KDD to address KDD99 issues. This dataset rectified errors and redundancies present in KDD99, offering four distinct parts categorized into training and testing subsets (KDDTrain+ , KDDTrain+_20Percent, KDDTest+, KDDTest-21). These subsets contain both normal and attack data, providing a comprehensive range of scenarios for testing various machine

learning models. Each subset differs in size and purpose: KDDTrain+ contains a larger training set, while KDDTrain+_20Percent is a reduced version, which is often used for more efficient experiments. The test sets, KDDTest+ and KDDTest-21, are used to evaluate the performance of models on unseen data.

2.3.4 CICIDS2017

The CICIDS2017 (Intrusion Detection Evaluation Dataset), developed by the Canadian Institute for Cybersecurity at the University of New Brunswick, offers several advantages over other datasets (Gharib *et al.*, 2016; Sharafaldin *et al.*, 2018), (Sharafaldin, Lashkari and Ghorbani, 2018). It provides real-world data obtained from a testbed consisting of actual computers, featuring diverse operating systems including Mac, Windows, and Linux on both attacker and victim machines. Moreover, the dataset includes labeled data, essential for applying Machine Learning methods, with 85 extracted features available (Refer to Appendix A for the feature list) for analysis. Both raw data (pcap files) and processed data (CSV files) are provided, offering flexibility in analysis. Additionally, the dataset draws from the 2016 McAfee security report to include a wide and current array of attacks, encompassing a rich variety of protocols including Hyper Text Transfer Protocol (HTTPS) in addition to FTP, HTTP, Secure Shell (SSH) (SSH), and email protocols. To provide a comprehensive understanding of the attack scenarios simulated within the CICIDS2017 dataset (Sharafaldin, Lashkari and Ghorbani, 2018), a detailed Tables (1,2) are presented below:

Table 1. Attacks simulated in the CICIDS2017 dataset

Attack	Description
Brute Force FTP	An attack where adversaries attempt to gain unauthorized access to an FTP server by systematically guessing usernames and passwords.

Brute Force SSH	Similar to Brute Force FTP, this attack targets SSH servers, aiming to gain unauthorized access through iterative login attempts.
Denial of Service (DoS)	DoS attacks aim to disrupt the availability of network services by overwhelming target systems with a deluge of malicious traffic.
Heartbleed	Heartbleed exploits a vulnerability in OpenSSL, allowing attackers to read sensitive information from the memory of the target server.
Web Attack	Generic term for various attacks targeting web applications, such as SQL Injection, Cross-Site Scripting (XSS), and directory traversal.
Infiltration	Infiltration attacks involve unauthorized access to network systems or resources, often through exploitation of vulnerabilities in network protocols or applications.
Botnet	Botnets comprise networks of compromised devices (bots) controlled by attackers, often used to execute coordinated attacks or distribute malware.
Distributed Denial of Service (DDoS)	DDoS attacks involve coordinated efforts to flood target systems or networks with overwhelming volumes of traffic, rendering them inaccessible to legitimate users.

Table 2. CICIDS2017 dataset files description

File Name	Day	Attacks
Monday-WorkingHours. pcap_ISCX.csv	Monday	Benign (Normal Activity)
Tuesday-WorkingHours. pcap_ISCX.csv	Tuesday	Benign, FTP-Patator, SSH-Patator
Wednesday-workingHours.pcap_ISCX.csv	Wednesday	Benign, DoS GoldenEye, DoS Hulk, DoS Slowhttptest, DoS slowloris, Heartbleed
Thursday-WorkingHours-Morning-WebAttacks. Pcap_ISCX.csv	Thursday	Benign, Web Attack - Brute Force, Web Attack – Sql Injection, Web Attack – XSS
Thursday-WorkingHours-Afternoon Infilteration.pcap_ISCX.csv	Thursday	Benign, Infiltration
Friday-WorkingHours-Morning.pcap_ISCX.csv	Friday	Benign, Bot
Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv	Friday	Benign, PortScan
Friday-WorkingHours-Afternoon-DoS.pcap_ISCX.csv	Friday	Benign, DDoS

However, CICIDS2017 also presents some drawbacks (Gharib *et al.*, 2016; Sharafaldin

et al., 2018). The raw and processed data files are notably large (47.9 GB and 1147.3 MB respectively), requiring substantial storage and processing resources. Unlike previous datasets such as KDD99 and NSL-KDD, CICIDS2017 does not provide separate files for training and testing, necessitating users to create these sections independently. Additionally, as a relatively new dataset, CICIDS2017 has not undergone extensive study and may contain minor inaccuracies, which are addressed in the data cleansing section.

After a thorough comparison process, the decision has been made to select the CICIDS2017 processed data (CSV files) as the dataset for use in the implementation phase. Key factors contributing to this preference include the dataset's up-to-date nature and its extensive coverage of protocols and attack types. Furthermore, given the limited number of studies conducted with this dataset so far, utilizing it presents an opportunity for making a potentially significant contribution to the literature.

2.4 Machine Learning

Machine learning is an interdisciplinary field that combines scientific principles and artistic creativity to enable computers to learn from data (Géron, 2022). This process involves training models on a dataset (training set) and evaluating their performance on a separate dataset (test set), thereby reducing the need for human intervention in problem-solving. Machine Learning is particularly valuable in domains where conventional approaches fall short, such as analyzing large, complex datasets, solving intricate problems beyond the capabilities of traditional methods, and adapting to evolving environments with minimal manual updates. Despite its advantages, not all Machine Learning techniques are suitable for every problem. For instance, deep learning, a subset of Machine Learning, requires vast amounts of data and computational resources, which may not be feasible with smaller datasets like CICIDS2017. Deep learning's complexity

and need for extensive datasets make it less suitable for scenarios with limited data. In such cases, other Machine Learning algorithms, such as supervised, unsupervised, semi-supervised, and reinforcement learning, may be more appropriate choices depending on the nature and size of the dataset.

Given these constraints, the choice between traditional Machine Learning and deep learning should be guided by the specific requirements and limitations of the task at hand. For applications where data is limited or computational resources are constrained, traditional Machine Learning methods may offer a more practical and efficient solution. Conversely, for tasks that can benefit from the high-level feature extraction and pattern recognition capabilities of deep learning, and have access to extensive datasets and computational power, deep learning may be the preferred approach. Machine Learning algorithms are typically divided into four main categories, distinguished by the labeling of their training data and the degree of supervision involved. These categories include supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning (Géron, 2022).

- **Supervised learning:** In this approach, the training data is accurately classified and labeled. For instance, all data flows in the dataset are tagged with information about their nature, such as being normal or harmful. During the test prediction phase, these tags are matched against the algorithm's findings, and its success rate is determined. The method's performance is high, but supervised learning is expensive due to the need for external services, like manual tagging, for labeling. Decision Trees, K-Nearest Neighbours, and Random Forests (RF) are examples of algorithms used here (Géron, 2022).

- **Unsupervised learning:** This method does not involve a labeling process. Instead, the algorithm categorizes the data into clusters based on different attributes and examines their relationships. It is widely applied in fields like anomaly detection, relationship learning, and dimensionality reduction. The cost is lower because it eliminates the need for outsourced expertise such as labeling (Géron, 2022).

- **Semi-supervised learning:** This hybrid approach merges supervised and unsupervised learning methods. Typically, a small fraction of the data is labeled, while the remainder is not. It offers the high performance of supervised learning with the cost-effectiveness of unsupervised learning (Géron, 2022).

- **Reinforcement Learning:** Fundamentally different from the other three, this method rewards the algorithm for correct decisions and penalizes it for errors during training, allowing it to develop its own set of rules (Géron, 2022).

In this thesis, we will utilize supervised learning techniques to leverage a meticulously annotated dataset. Our objective is to attain superior performance benefits while avoiding excessive costs. The Machine Learning algorithms employed in the implementation stage encompass Naive Bayes (NB), Random Forest (RF), AdaBoost, and K Nearest Neighbours (KNN). NB and RF were selected because they are commonly used in the studies that will be compared, and they have demonstrated strong performance in feature selection tasks. K Nearest Neighbours (KNN) was chosen due to its simplicity and effectiveness in classification tasks, especially when dealing with high-dimensional data

where other models may struggle. AdaBoost, on the other hand, was selected for its ability to improve the performance of weak models by focusing on misclassified instances, making it particularly effective in cases where the data is noisy or contains outliers. These algorithms were selected not only based on their individual strengths but also for their relevance to the models used in the studies being compared.

2.4.1 Naive Bayes

The NB algorithm simplifies Machine Learning by introducing the independence condition to Bayes' theorem (Kotsiantis, Zaharakis and Pintelas, 2007). Operating on the assumption of strong independence, the NB classifier (NB) is a straightforward yet highly efficient probabilistic classification approach (Mukherjee and Sharma, 2012a). NB is favored for its simplicity and effectiveness in classification tasks, ranking among the top choices in terms of accuracy and computational efficiency (Ting, Ip and Tsang, 2011). NB forms a probabilistic network comprising a parent node representing the unobserved state and multiple child nodes representing observed states. While this approach assumes independence among the child nodes, this assumption often does not hold true in practice, potentially leading to lower accuracy compared to other Machine Learning methods (Kotsiantis, Zaharakis and Pintelas, 2007). Nonetheless, NB remains popular due to its brief training period and low computational cost.

2.4.2 Random Forest

RF Classifier utilizes decision trees in a technique where a "forest" is constructed by aggregating numerous decision tree structures generated in various ways (Breiman, 2001). RF produces N decision trees from the training dataset, employing random resampling of the training set for each tree. Consequently, a set of N distinct decision trees

is obtained, each differing from the others. Subsequently, a voting mechanism combines new estimates derived from the N trees, with the highest-rated value being designated as the final outcome. RF offers several advantages (Kostas, 2018):

1. It demonstrates effectiveness with extensive and intricate datasets.

2. Unlike decision trees, RF remarkably mitigates this issue.

3. Its versatility enables application across various Machine Learning problems.

4. The algorithm adeptly handles missing values within datasets by substituting them with internally generated values.

5. Additionally, RF incorporates the assessment and utilization of variable importance during classification, making it suitable for feature selection in Machine Learning tasks.

However, RF's complexity arises from its composition of numerous decision trees. Understanding its functionality can be challenging, constituting a notable disadvantage.

2.4.3 AdaBoost

AdaBoost (Adaptive Boosting), classified as a boosting method, represents a Machine Learning algorithm designed to enhance classification performance (Schapire, 2003). The fundamental operational concept of Boosting algorithms entails initially dividing data into groups using rudimentary rules. Upon each iteration of the algorithm, new rules are incorporated into these initial rules, resulting in the acquisition of numerous weak and underperforming rules referred to as "base rules". Through successive iterations, these weak rules are amalgamated into a single rule of significantly greater strength and efficacy. Throughout this process, the algorithm assigns a weighting coefficient to each

weak rule, with the highest coefficient attributed to the rule exhibiting the lowest error rate. These weight values play a pivotal role in the selection of final rules, prioritizing those with higher scores (Schapire, 2003). AdaBoost algorithm presents several advantages (Kostas, 2018):

1. It obviates the need for variable transformation for implementation.

2. It effectively operates on numerous weak rules.

3. The occurrence of overfitting in AdaBoost is notably infrequent.

4. Additionally, AdaBoost exhibits proficiency in handling missing values within datasets.

Conversely, it is noteworthy to mention some disadvantages, including susceptibility to noise and outliers, and comparatively lower predictive values when juxtaposed with other algorithms.

2.4.4 K Nearest Neighbour

KNN (K Nearest Neighbour), recognized as a sample-based technique, stands out as one of the widely employed Machine Learning algorithms due to its straightforward and efficient structure. This approach operates on the premise that instances within a dataset exhibit proximity to instances sharing similar characteristics in another dataset (Kotsiantis, Zaharakis and Pintelas, 2007). In practical terms, KNN discerns the classification of novel data points by leveraging the known class types within the training dataset. This classification process entails scrutinizing the nearest neighbors of the novel sample lacking specific classifications (Kotsiantis, Zaharakis and Pintelas, 2007; Kramer and Kramer, 2013). The following table (3) compares the supervised learning techniques

utilized in this thesis:

Table 3. Supervised learning techniques

Algorithm	Principle of Working	Model Parameters	Advantages	Applications
Naive Bayes	Calculates probabilities based on the assumption of feature independence, using Bayes' theorem to classify data	Probability estimates for each class, prior probabilities based on training data	Simple and fast for classification tasks; performs well with high-dimensional data	Spam detection, sentiment analysis, document categorization
Random Forest	Builds multiple decision trees and merges their predictions for better accuracy	Number of trees in the forest, maximum depth of each tree.	Reduces overfitting compared to single decision trees; handles missing values effectively	Feature selection, risk assessment, medical diagnosis
AdaBoost	Combines multiple weak classifiers to form a strong classifier through weighted voting	Number of weak classifiers, learning rate for updating weights	Reduces bias and variance; robust to outliers	Face detection, text classification, customer churn prediction
K Nearest Neighbour	Classifies a new data point based on the majority class among its k-nearest neighbors in the training dataset	Number of neighbors (k), distance metric used for measuring similarity	Simple to implement and understand; effective for small datasets	Recommendation systems, image classification, anomaly detection

Chapter 3: Chapter Three: Literature Review

This chapter reviews relevant studies and research related to intrusion detection systems, feature selection methods, and machine learning techniques. It highlights key findings, methodologies, and gaps in the existing literature that this research aims to address.

3.1 Introduction

In this chapter, we categorize and review the literature on Machine Learning approaches for intrusion detection based on three prominent datasets: KDD99, NSL-KDD, and CICIDS2017. This categorization allows for a structured examination of various studies, highlighting the methods, results, and contributions specific to each dataset. By exploring the performance and methodologies of previous research, we aim to provide a comprehensive overview of the advancements and challenges in the field of anomaly detection in computer networks. Each section will delve into the Machine Learning algorithms utilized, the specifics of the datasets, and the performance metrics that were achieved, offering a detailed and nuanced understanding of the current state of the art.

3.2 DARPA 98 and KDD99

In a study conducted by Chebrolu et al. (Chebrolu, Abraham and Thomas, 2005), investigated intrusion detection using the DARPA dataset, initially applying Principal Component Analysis (PCA) and Independent Component Analysis (ICA) for data compression, but these methods did not yield adequate results. This prompted the use of more advanced techniques like Markov blanket model-based feature selection and decision tree analysis. The researchers then applied a hybrid model combining Bayesian Networks (BN) and Classification and Regression Trees (CART), which led to impressive results. This ensemble approach achieved 100% accuracy for detecting Normal, Probe,

and Denial of Service (DoS) attacks, while User to Root (U2R) and Remote to Local (R2L) attacks had detection accuracies of 84% and 99.47%, respectively. These results suggest that hybrid approaches can be particularly effective in addressing the varying complexities of different attack types.

Similarly, the work by (Khan, Awad and Thuraisingham, 2007) builds upon the challenges of intrusion detection, particularly in handling the less successful U2R and R2L attack categories. In their 2007 study, Support Vector Machines (SVM) were applied to the DARPA 98 dataset, yielding high accuracy for Normal (98%), Probe (88%), and DoS (84%) attacks but falling short with U2R and R2L (0% and 18%, respectively). To address these limitations, the integration of the Dynamically Growing Self-Organizing Tree (DGSOT) algorithm significantly improved performance, particularly for U2R (23%) and R2L (43%) attacks. This study highlights how combining SVM with supplementary algorithms can enhance detection rates, especially for harder-to-detect anomalies. The SVM + DGSOT method outperformed other approaches in terms of accuracy and efficiency, further underscoring the importance of leveraging ensemble or hybrid techniques, as seen in Chebrolu et al.'s work.

The importance of feature selection, explored in the study by Wang (Wang *et al.*, 2015), ties back to Chebrolu et al.'s use of data reduction techniques and ensemble models. Instead of focusing on all 41 features in the KDD99 dataset, [39] identified the top 10 features using Information Gain (IG) and wrapper methods with Bayesian Networks (BN) and Decision Trees (C4.5). These reduced feature sets achieved detection rates of 99.8% with a false positive rate of 0.34%. This study reinforces the idea that careful feature selection can maintain or improve detection accuracy while enhancing system efficiency, echoing the earlier findings that reduced datasets can still produce high detection rates.

The focus on Distributed Denial of Service (DDoS) detection in (Wang *et al.*, 2015) extends the application of feature selection to real-world environments, a practical advancement over earlier studies that primarily focused on laboratory datasets.

Building on the theme of feature selection, (Peng *et al.*, 2018) introduced the FACO (Ant Colony Optimization-based Feature Selection) algorithm, leveraging ant colony optimization principles to reduce redundancy and avoid local optima in feature selection. Tested on the KDD CUP99 dataset, FACO achieved a classification accuracy of 98% using a curated subset of features, further confirming the effectiveness of optimized feature selection in intrusion detection tasks. This aligns with (Wang *et al.*, 2015)'s emphasis on the importance of reducing the number of features for greater efficiency, and it further extends the potential of advanced optimization techniques like Ant Colony Optimization for real-world intrusion detection scenarios.

Finally, the study by Tao (Tao, Sun and Sun, 2018) proposed a novel approach by combining the strengths of SVM and Genetic Algorithms (GA) for intrusion detection. This research introduced the Fuzzy Weighting Preprocessing - Support Vector Machines - Genetic Algorithms (FWP-SVM-GA), which optimizes feature selection, weight adjustment, and parameter tuning. The use of GA to optimize SVM parameters, combined with a fitness function designed to minimize error rates and maximize true positive rates, resulted in faster convergence and better detection rates. This study is particularly relevant in light of the earlier works that highlighted the limitations of SVM when used in isolation for complex attack types like U2R and R2L. By integrating GA, (Tao, Sun and Sun, 2018) demonstrates the potential for overcoming these challenges, providing improved results across various performance metrics, including reduced false positives and shorter training times. This further supports the evolving trend toward combining multiple algorithms to

enhance intrusion detection system performance, as initially proposed by Chebrolu et al. and expanded upon in subsequent studies.

3.3 NSL-KDD

The study by Mukherjee (Mukherjee and Sharma, 2012b) serves as an essential starting point, aiming to enhance IDS accuracy by reducing the input feature set through the Feature Vitality Based Reduction Method (FVBRM). Compared to traditional feature selection methods such as Correlation-based Feature Selection (CFS), IG, and Gain Ratio (GR), FVBRM showed superior results, achieving an impressive classification accuracy of 97.78%. Although this method utilized a larger feature set, its effectiveness in classifying various attack types, especially DoS and probe, was remarkable. This indicates that FVBRM's approach of retaining more features proved to be efficient in boosting classification performance. Connecting this to study by Popoola (Popoola and Adewumi, 2017), which adopted a different approach to improving intrusion detection accuracy, it employed Discretized Differential Evolution (DDE) alongside the C4.5 Machine Learning algorithm. By focusing on optimal feature selection, this method reduced the feature set and led to significant improvements in detection accuracy. The findings revealed that 16 critical features were sufficient to achieve high classification accuracy with minimal error rates, including a notable accuracy rate of 99.92% on the training dataset. This efficiency in feature selection was also reflected in the reduced training and testing times. When compared to FVBRM, which retained a larger feature set, DDE proved that fewer but more optimally selected features could still deliver comparable or even superior results in some instances, especially when considering processing time which adopted a different approach to improving intrusion detection accuracy, it employed Discretized Differential Evolution (DDE) alongside the C4.5 Machine

Learning algorithm. By focusing on optimal feature selection, this method reduced the feature set and led to significant improvements in detection accuracy. The findings revealed that 16 critical features were sufficient to achieve high classification accuracy with minimal error rates, including a notable accuracy rate of 99.92% on the training dataset. This efficiency in feature selection was also reflected in the reduced training and testing times. When compared to FVBRM, which retained a larger feature set, DDE proved that fewer but more optimally selected features could still deliver comparable or even superior results in some instances, especially when considering processing time.

The integration of hybrid feature selection methods was explored in study by H. Dong (Dong, Shui and Zhang, 2021), where a combination of IG and PCA was applied in conjunction with RF. This approach addressed the challenges of high-dimensional data in industrial control networks. With classification accuracies of 99.84% and 99.80% for feature subsets from NSL-KDD and CICIDS2017 datasets, respectively, the IG-PCA-RF model clearly outperformed traditional classifiers like DT and SVM. Furthermore, benchmarking against contemporary methods, such as DE-ELM and Cosine-PIO, demonstrated that hybrid approaches could achieve superior classification accuracy. When comparing this with DDE, both methods focus on optimizing feature selection but through different techniques DDE leveraging evolutionary algorithms and IG-PCA-RF combining statistical and dimensionality reduction techniques. The study by Belavagi (Belavagi and Muniyal, 2016), utilized all 42 features from the NSL-KDD dataset and evaluated the performance of several Machine Learning algorithms, including Logistic Regression, Gaussian NB, SVM, and RF. RF emerged as the most effective, achieving a 99% accuracy rate. This comprehensive analysis highlights the strength of RF in IDS applications, especially when working with the full feature set. In contrast to studies that

focused on reducing the feature set, this study demonstrates that, under the right algorithm, retaining all features can yield impressive results.

On the other hand, study by Yin (Yin *et al.*, 2017), shifted the focus towards deep learning by proposing a Recurrent Neural Network-based IDS (RNN-IDS). This study achieved notable detection rates for attack types like DoS and probe, outperforming traditional Machine Learning methods like J48 and RF. The performance of RNN-IDS underscores the potential of deep learning approaches in IDS applications, particularly in multiclass classification tasks. When comparing this with the previously discussed methods, RNN-IDS stands out in its ability to model complex data patterns, but it also introduces more computational complexity compared to the simpler feature selection techniques like FVBRM and DDE.

In study by Ambusaidi (Ambusaidi *et al.*, 2016), the issue of redundant and irrelevant features in network traffic classification was addressed by introducing the Flexible Mutual Information Feature Selection (FMIFS) algorithm. FMIFS was designed to enhance the accuracy and efficiency of feature selection by eliminating the need for a redundancy parameter, thus streamlining the process. When integrated into the Least Square SVM-based Intrusion Detection System (LSSVM-IDS), this approach yielded remarkable performance improvements. Across multiple datasets, including KDD Cup 99, NSL-KDD, and Kyoto 2006+, LSSVM-IDS paired with FMIFS achieved superior results compared to traditional methods. For instance, on the NSL-KDD dataset, the system attained a 99.91% accuracy rate, a detection rate of 98.76%, and a false positive rate of just 0.28%. This study's contributions are particularly notable when contrasted with other techniques discussed earlier. Study (Dong, Shui and Zhang, 2021), for example, employed a hybrid feature selection method combining IG and Principal

Component Analysis (PCA), which focused on reducing data dimensionality and redundancy, alongside boosting classification accuracy. While FMIFS prioritizes relevance by simplifying feature selection and reducing computational costs, the IG-PCA approach from study (Dong, Shui and Zhang, 2021) addresses dimensionality challenges more directly. Both methods, although different in their focus, have significantly advanced IDS capabilities by refining how features are selected and processed, offering distinct paths to improve the detection and classification of network intrusions. Furthermore, the performance metrics of FMIFS, particularly its low false positive rates and high detection accuracy, align closely with the objectives of reducing computational overhead without compromising on precision. This balance is critical for real-world deployment, especially in environments with high-dimensional data like industrial control systems or large-scale enterprise networks.

However, as study (Ambusaidi *et al.*, 2016) suggests, while FMIFS demonstrates great promise, future research could refine the search strategy to address challenges posed by unbalanced data distributions. This focus on optimization parallels discussions from study (Mukherjee and Sharma, 2012b), where the Feature Vitality Based Reduction Method (FVBRM) sought to maximize classification accuracy across different attack types while dealing with large feature sets. Both studies underscore the ongoing need to refine feature selection techniques to handle the complexities of modern Intrusion Detection Systems effectively.

3.4 CICIDS2017

As the field of intrusion detection continues to advance, the CICIDS2017 dataset has emerged as a crucial benchmark for evaluating the effectiveness of various Intrusion Detection Systems (IDS). Several studies have focused on leveraging this dataset to

develop and test innovative techniques aimed at improving detection accuracy and efficiency. In study by Yulianto [48], the researchers aimed to enhance AdaBoost-based Intrusion Detection Systems (IDS) by integrating multiple advanced techniques, including the Synthetic Minority Oversampling Technique (SMOTE), Principal Component Analysis (PCA), and Ensemble Feature Selection (EFS), applied on the CICIDS2017 dataset. The motivation behind this integration was to address the imbalance commonly found in Intrusion Detection datasets, where minority class samples (representing attack data) are underrepresented compared to majority class samples (representing normal traffic). The proposed system begins by applying SMOTE to balance the dataset by oversampling the minority class samples. This is a crucial step to avoid model bias toward the majority class and ensure more accurate detection of intrusion attempts. Following this, PCA is employed to reduce the dimensionality of the dataset, simplifying the dataset while retaining the most significant features. Feature selection is then performed using the EFS technique, which calculates the IG from each feature and ranks them. The final optimal set of features is chosen from this ranking to be used in the AdaBoost classifier. Through experimental evaluations, this combined system—SMOTE, PCA, EFS, and AdaBoost—achieved remarkable results. Notably, the system demonstrated an Area Under the Receiver Operating Characteristic curve (AUROC) of 92%. Additionally, it achieved an accuracy of 81.83%, precision of 81.83%, a perfect recall of 100%, and an F1 Score of 90.01%. These metrics highlight the system's effectiveness in not only identifying a large proportion of true positive instances (recall) but also maintaining precision and overall balanced performance. The study's outcomes are significant when compared to previous approaches, demonstrating improvements across multiple performance metrics. This highlights the importance of combining robust

Machine Learning techniques like AdaBoost with advanced data preprocessing methods such as SMOTE and PCA. The research emphasizes the growing need for high-performance IDS models capable of handling increasingly complex and diverse network behaviors, as seen in modern network environments like those represented by the CICIDS2017 dataset.

Building on this foundation of addressing data imbalance and feature selection, study by Jamadar (Jamadar, 2018) introduces a novel approach leveraging the decision tree algorithm to construct a Network Intrusion Detection System (NIDS). Here, the focus shifts toward anomaly detection, where the system utilizes Recursive Feature Elimination (RFE) after encoding categorical features from the CICIDS2017 dataset. The methodology emphasizes the importance of using unseen data for model validation, and the decision tree model developed in this study achieved a remarkable 99% accuracy on the test dataset, with a True Positive Rate (TPR) of 99.9% and a False Positive Rate (FPR) of just 0.1%. This further reinforces the argument that feature selection and model optimization are critical to the detection of network intrusions. Unlike the previous study, which used multiple techniques (SMOTE, PCA, EFS), this study focused on the decision tree as a single classifier, achieving impressive results with minimal complexity.

In a complementary study by Kurniabudi (Kurniabudi *et al.*, 2021), the focus broadens to tackle high-class imbalance within the CICIDS-2017 dataset using the IG feature selection technique. This study also employs the RF classifier, known for its robustness in handling multi-class datasets. The experimental results demonstrate that by using 22 to 28 selected features, the classifier achieves an accuracy of 99.84% on the training data and 99.83% on the test dataset. This emphasizes the critical role of feature selection, as the reduced set of features still maintains high classification performance. The study

introduces two approaches aimed at extracting relevant features crucial for detecting attacks within high-dimensional, multi-class, and imbalanced datasets. Additionally, it successfully highlights how different classifiers can handle imbalanced datasets when supported by robust feature extraction techniques. Further experimentation with the IG technique is warranted to fully optimize feature selection, underscoring the study's superiority over existing methods in terms of accuracy, True Positive Rate, False Positive Rate, Precision, and Receiver Operating Characteristic metrics.

In study by Reis (Reis, Maia and Praça, 2019), the dynamic evolution of network infrastructures and the escalating sophistication of cyberattacks underscore the need for robust Intrusion Detection Systems (IDS). Leveraging Machine Learning techniques, particularly feature selection and ensemble methods, the research focuses on the recent CICIDS2017 dataset to develop effective intrusion detection models. One of the key challenges addressed in this study is the need to balance high detection accuracy with efficient execution time. By employing permutation importance, the study efficiently distills the dataset's original 69 features into a concise set of 10, significantly enhancing model execution time without compromising detection performance. This method not only reduces the computational load but also facilitates faster intrusion detection processes, a critical factor in real-time network environments. Notably, the evaluation using the RF algorithm on both the reduced and full feature sets (10 vs. 69 features) reveals that the optimized dataset sustains high detection performance. This approach highlights a common challenge in IDS research: balancing feature reduction with maintaining or even improving detection rates. Metrics such as precision, recall, F1-score, false negatives, and false positives were calculated for various attack categories. The results indicate that feature selection can significantly impact detection efficiency, as

demonstrated by the RF algorithm achieving 93% precision, 91% recall, and a 91.9% F1-score with the reduced feature set. This underscores the potential of feature selection techniques in refining IDS models for real-time detection scenarios. Additionally, the study examines false positive and false negative rates, crucial for any IDS, showing minimal false positives and primarily false negatives. For instance, with 10 features, the false negative rate (FNR) is 0.09 compared to 0.081 with 69 features, while the false positive rate (FPR) remains zero. When considering attacks confused with benign traffic, the FNR decreases to 0.043 with 10 features and 0.036 with 69 features, showing the effectiveness of the reduced feature set in real-world scenarios. While attacks like DDoS, Heartbleed, Infiltration, and SSH Patator are well-detected, the Web Bruteforce attack poses a greater challenge, with a 0.554 FNR, underscoring the complexity of detecting more subtle or obfuscated attacks.

In study by Stiawan (Stiawan *et al.*, 2020), the focus lies on enhancing traffic anomaly detection accuracy while reducing computational complexity goals that closely align with those in study (Reis, Maia and Praça, 2019). The study employs IG as the primary feature selection technique to rank and group features based on minimum weight values. This method, similar to the permutation importance approach in (Reis, Maia and Praça, 2019), aims to extract the most relevant features from the CICIDS2017 dataset while minimizing computational overhead. Here, too, the challenge of balancing detection accuracy with execution time is evident. The study compares the performance of several classifier algorithms, including RF, Bayes Net (BN), Random Tree (RT), NB, and J48, revealing that the number of selected features significantly influences both detection accuracy and execution time. Notably, RF achieves the highest accuracy of 99.86% using 22 relevant features, though with longer execution times compared to J48, which attains a slightly

higher accuracy of 99.87% using 52 relevant features. This finding mirrors the trade-off observed in study (Reis, Maia and Praça, 2019), where a reduced feature set leads to faster execution but requires careful consideration to avoid sacrificing detection accuracy. The superiority of RF in achieving high accuracy with fewer features aligns with the conclusions drawn in study (Reis, Maia and Praça, 2019), where feature selection proved critical to improving model efficiency. However, study (Stiawan *et al.*, 2020) also highlights the potential benefits of exploring different classifiers, as Bayes Net (BN) demonstrated proficiency in detecting all traffic types with specific feature subsets, though with slightly lower accuracy. The study further emphasizes the importance of reducing false positive rates, particularly for BN, which benefits significantly from effective feature selection.

Study by Jany (Jany Shabu *et al.*, 2021) investigates a new framework for Anomaly-based Intrusion Detection Systems (ADS) that aims to improve the performance of existing intrusion detection methods. Despite the application of various supervised and unsupervised Machine Learning techniques, achieving high detection accuracy remains a challenge due to limited public datasets and the variable performance of different classifiers in detecting specific types of attacks. To address these challenges, this study compares multiple feature selection methods and classification decision algorithms, integrating them into a unified framework with automatic parameter adjustment. This approach enhances the robustness of both feature selection and ensemble classification, reducing reliance on manual experimentation for optimal parameter settings. The experimental analysis uses the NSL-KDD dataset and the more recent CICIDS2017 dataset, demonstrating the framework's efficiency in terms of accuracy and a significantly reduced False Positive Rate (FPR). The results show that automatic feature selection

combined with ensemble classifiers can markedly improve the effectiveness of Intrusion Detection Systems (IDS). The system's reduced FPR makes it particularly effective for anomaly detection, enhancing its practical utility in real-time network environments. When comparing this study with previous works such as (Reis, Maia and Praça, 2019), it is clear that (Jany Shabu *et al.*, 2021) focuses on enhancing the robustness of IDS frameworks through automatic feature selection and classifier parameter optimization, whereas studies (Reis, Maia and Praça, 2019) concentrated on optimizing detection accuracy by selecting a reduced number of features using different methods like permutation importance and IG, respectively. However, none of the previous studies explicitly addressed the challenge of automatic parameter adjustment, which (Jany Shabu *et al.*, 2021) incorporates, making it an important contribution that bridges a gap in the current IDS landscape.

In study by Mhawi (Mhawi, Aldallal and Hassan, 2022), a novel approach to enhance Intrusion Detection Systems (IDSs) is presented, focusing on the integration of advanced feature selection techniques within a hybrid ensemble learning framework. This study addresses the challenges faced in prior research (Reis, Maia and Praça, 2019), where high dimensionality and computational costs hindered the effectiveness of anomaly detection. The proposed methodology employs a hybrid of Correlation Feature Selection coupled with Forest Penalized Attributes (CFS–FPA), significantly improving feature relevance and reducing redundancy. The study's experiments utilize the CICIDS2017 dataset, akin to those in studies (Reis, Maia and Praça, 2019), which also focus on optimizing detection accuracy while minimizing false alarm rates. Notably, the proposed model demonstrates an impressive accuracy of 99.73% with a false-negative rate of 0.123, showcasing its efficacy in managing the balance between high detection rates and low error rates. This

aligns with the findings of study (Stiawan *et al.*, 2020), where the emphasis was placed on selecting relevant features to enhance detection performance while considering execution time. Furthermore, the comparison of multiple classifiers, including SVM, RF, NB, and KNN, echoes the classifier diversity discussed in study (Stiawan *et al.*, 2020). In both studies, the importance of selecting the right features becomes apparent, particularly when dealing with complex datasets like CICIDS2017. The findings suggest that employing an ensemble learning approach, as seen in study (Mhawi, Aldallal and Hassan, 2022), not only increases accuracy but also enhances the overall robustness of the IDS.

In this research, our evaluation and improvements will draw upon the methodologies outlined in studies (Reis, Maia and Praça, 2019), enabling us to advance our investigation. By leveraging their use of feature selection techniques and classifier optimization on the CICIDS2017 dataset, we aim to refine our own approach to achieving a balance between detection accuracy and computational efficiency. Specifically, we will incorporate the use of permutation importance and IG, as explored in these studies, to identify the most relevant features, which can streamline the intrusion detection process. Through detailed experimentation, we will seek to optimize the feature set that offers the highest accuracy for detecting attacks, while also minimizing the processing time and overhead required. Additionally, our work will include a rigorous validation process that not only compares our findings with those of prior studies but also delves into execution efficiency, evaluating the trade-offs between accuracy and speed. This comparative analysis will help in identifying a feature subset that is consistent across multiple extraction techniques, improving the interpretability and robustness of IDS. This comprehensive approach will pave the way for future research, setting the groundwork for more efficient and reliable intrusion detection frameworks.

Chapter 4: Chapter Four: Methodology

This chapter outlines the methodology used in this thesis. It explains the approaches and techniques applied to data collection, feature selection, model training, and evaluation. The chapter also justifies the choice of algorithms and optimization methods, providing a clear overview of the steps taken to achieve the research objectives.

4.1 Tools and Methods

4.1.1 Software Platform

Python (*3.12.3 Documentation*, 2024), Python, an object-oriented programming language known for its simplicity and flexibility, garners attention due to its intuitive syntax and dynamic nature. It facilitates both coding and analysis tasks seamlessly. Furthermore, Python boasts extensive documentation available through various channels such as books, online platforms, and community forums. Additionally, its compatibility with numerous libraries tailored for Machine Learning applications makes it a preferred choice, particularly Python 3.6 for its advantageous features.

Scikit-learn (*scikit-learn: machine learning in Python — scikit-learn 1.4.2 documentation*, no date), commonly referred to as sklearn, stands as a robust Machine Learning library designed to enhance Python's inherent simplicity and adaptability. Offering a rich suite of resources for Machine Learning and statistical analysis, it simplifies intricate tasks. Its user-friendly interface and detailed documentation make it a preferred choice among novices and experienced professionals in the field of data science. The seamless integration with Python's ecosystem and diverse algorithm support streamlines the creation and implementation of Machine Learning models. Furthermore, its commitment to clear, coherent code resonates with Python's principles, cementing its

position as a favorite among developers.

Pandas (‘pandas - Python Data Analysis Library’, no date), a library, within the Python ecosystem transforms the way data is managed and analyzed through its data structures and robust tools. With a foundation on NumPy Pandas provides user data structures like DataFrames and Series that simplify tasks such as cleaning, transforming and exploring data. Its smooth integration with other Python libraries like Matplotlib and scikit learn streamlines data analysis workflows. Additionally, Pandas broad range of features for handling missing data, time series and database like operations make it essential for professionals working with data in industries. From importing data to creating visualizations Pandas enables users to extract insights from their data effectively solidifying its position as a tool, in the field of data science.

Matplotlib (‘Matplotlib — Visualization with Python’, no date), a component of the Python data visualization toolkit allows users to effortlessly craft quality visual representations. With an array of plotting features ranging from line graphs to intricate 3D displays Matplotlib offers the versatility and personalization necessary to effectively communicate insights. Its seamless integration, with Jupyter Notebook and interactive interfaces facilitates data exploration and presentation. Moreover, Matplotlib’s comprehensive documentation and strong community backing ensure accessibility for users of varying expertise levels fostering an atmosphere for sharing knowledge and techniques. Whether designing visuals for reports or dynamic plots for applications Matplotlib remains a preferred tool, among data professionals, scholars and teachers alike propelling advancements and discoveries through visualization process.

NumPy (‘NumPy - ’, no date), the fundamental package for numerical computing in Python, serves as the backbone for scientific computing and data analysis tasks. Its powerful array objects and versatile functions enable efficient manipulation and computation of large datasets with ease. NumPy's array-oriented computing paradigm facilitates vectorized operations, optimizing performance and scalability for numerical tasks. Moreover, its seamless integration with other Python libraries, such as Pandas and Matplotlib, streamlines data analysis workflows, enabling users to explore, visualize, and analyze data seamlessly. With its robust functionality for mathematical operations, linear algebra, Fourier transforms, and random number generation, NumPy empowers users across various domains, from academia to industry, to tackle complex computational challenges and drive innovation forward.

4.1.2 Hardware Platform

A vital consideration when assessing Machine Learning algorithms is their execution time. However, it's crucial to acknowledge that execution time can fluctuate based on the performance of the specific computer utilized. To provide transparency, the technical specifications of the computer employed in the application are disclosed. The technical attributes of the computer during the implementation phase encompass:

Central Processing Unit: 11th Gen Intel(R) Core(TM) i5-11400H @ 2.70GHz 2.69 GHz

Random Access Memory: 32.0 GB (31.7 GB usable)

System Type: 64-bit operating system, x64-based processor

Operation System: Windows 11 Home

Graphic Processing Unit: NVIDIA GeForce RTX 3050 TI

4.1.3 Performance Evaluation Methods

The appraisal of the discoveries in this inquire about is based on four key measurements: accuracy, recall, precision and F-measure. These measurements are bounded between 0 and 1, where a esteem closer to 1 demonstrates superior execution, whereas a esteem nearing 0 means lower performance.

Accuracy is characterized as the extent of accurately classified information to the add up to dataset (Bhuyan, Bhattacharyya and Kalita, 2013b).

$$\text{Accuracy} = \frac{TN+TP}{FP+TN+TP+FN} \tag{4.1}$$

Recall, known as Affectability, speaks to the extent of information precisely recognized as an assault out of all assault occasions (Bhuyan, Bhattacharyya and Kalita, 2013b).

$$\text{Recall} = \frac{TP}{Tp+FN} \tag{4.2}$$

Precision measures the extent of precisely classified assault information among all information classified as assaults (Bhuyan, Bhattacharyya and Kalita, 2013b).

$$\text{Precision} = \frac{TP}{Fp+TP} \tag{4.3}$$

F-measure, F-score or F1-score, signifies the consonant cruel of review and accuracy. It serves as a comprehensive pointer of generally victory (Bhuyan, Bhattacharyya and Kalita, 2013b). Thus, in this examination, specific consideration will be paid to the F1 measure.

$$\text{F-measure} = \frac{2}{\frac{1}{Recall}+\frac{1}{Precision}} \tag{4.4}$$

The assessment of these four measurements depends on the following:

- True Positive (TP): Occasions where assault information is accurately distinguished as attacks.
- False Positive (FP): Occurrences where benign data is erroneously classified as attacks (Type-1 Error).
- False Negative (FN): Occurrences where attack data is wrongly classified as benign (Type-2 Error).
- True Negative (TN): Occasions where benign data is accurately recognized as benign.

These metrics are visually represented through a Confusion Matrix, as depicted in Figure (2), which provides a clearer understanding of their relationships and significance.

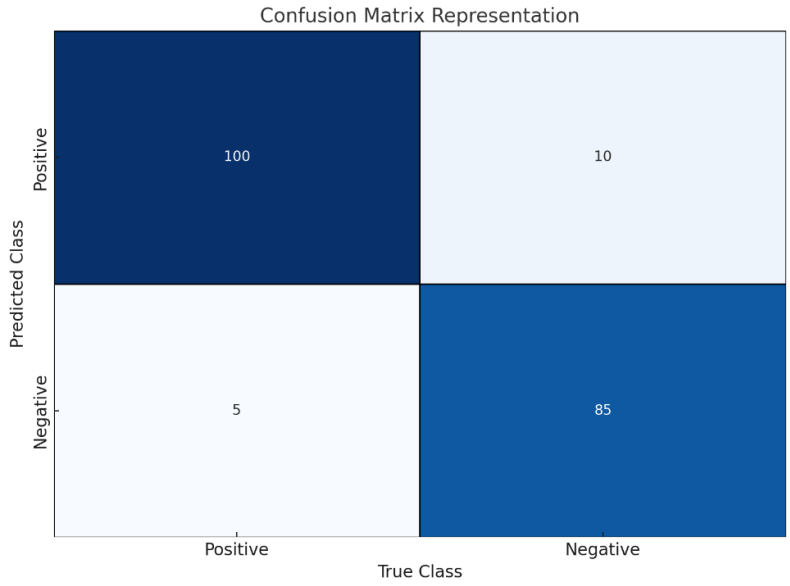

Figure 2. Confusion Matrix representation of classification results

Additionally, while not considered a success criterion, the processing time is included in

this evaluation list due to its significance in algorithm selection.

4.2 Implementation

This section encompasses a series of steps involving preprocessing and practical application aimed at anomaly detection using Machine Learning techniques. Initially, the data undergoes a cleaning process to rectify errors and defects. Subsequently, the dataset is partitioned into two segments: training and testing. Following these steps, the selection of features required by the algorithms is determined. The section concludes with the execution of Machine Learning algorithms. A detailed depiction of this implementation process is provided in Figure (3).

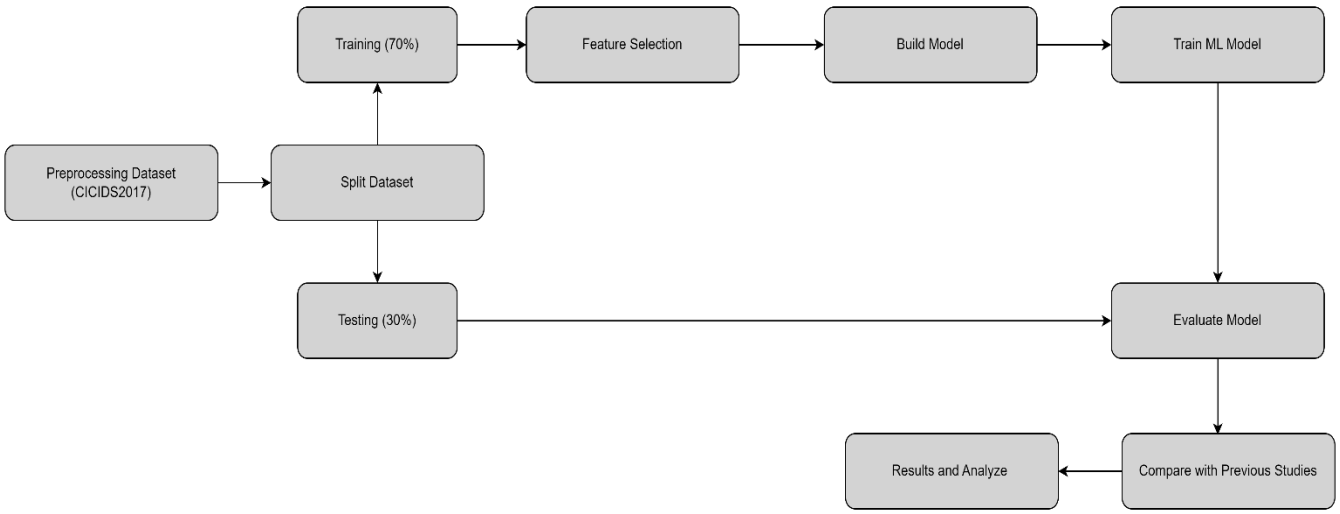

Figure 3. Methodology process

4.2.1 Data Cleaning

To enhance the utility of the dataset for practical applications, it may be necessary to implement certain modifications. In this section, we address several issues present in the

CICIDS2017 dataset and propose corresponding corrections. The initial dataset comprises 2,830,743 stream records (Sharafaldin, Lashkari and Ghorbani, 2018), as outlined in Table (4).

Table 4. Attacks records in the CICIDS2017 dataset

Label Name	Number
Benign	2,273,097
DoS Hulk	231,073
PortScan	158,930
DDoS	128,027
DoS GoldenEye	10,293
FTP-Patator	7,938
SSH-Patator	5,897
DoS Slowloris	5,796
DoS Slowhttptest	5,499
Bot	1,966
Web Attack – Brute Force	1,507
Web Attack – XSS	652
Infiltration	36
Web Attack – SQL Injection	21
Heartbleed	11

We identified a duplication issue with the Fwd Header Length feature within the dataset. This particular feature, crucial for defining the forward direction data flow in terms of total bytes used, was found to be duplicated, appearing in both the 35^{th} and 56^{th} columns. To address this anomaly, we resolved to eliminate the redundant occurrence located in column 56.

Then convert properties that include categorical and string values — such as source IP address, destination IP address, stream ID and timestamp into numeric data suitable for Machine Learning algorithms. This conversion can be done using Sklearn's LabelEncoder() by applying this method, It will be assigned different numerical values, which can be used directly in Machine Learning processes, ranging from 0 to n-1, making it more suitable for analysis. In any case, in spite of the 'Label' tag being a categorical

include, no adjustments were made to it. Despite the fact that the "Label" attribute qualifies as a categorical feature, it has been left unaltered. This decision stems from the necessity to preserve the original categories throughout the processing stage. Retaining these categories facilitates the classification of attack types in varied manifestations and enables experimentation with diverse analytical methods.

In the 'Label' feature, the character '–' used to differentiate web attack subtypes (such as Brute Force, XSS, SQL Injection) must be replaced with the character '-' because utf-8, the default encoding for the Pandas library, does not recognize the former character. If not replaced, this issue will cause the Pandas library to malfunction.

Furthermore, the 'Flow Bytes/s' and 'Flow Packets/s' features include the values 'Infinity' and 'NaN' alongside numerical values. These should be converted to -1 and 0, respectively, to ensure they are suitable for Machine Learning algorithms.

4.2.2 Splitting Data into Training and Testing

During the Machine Learning process, data is essential for enabling learning to occur. This necessity is met through the use of datasets. Besides the data required for training, test data is crucial for evaluating the performance of the algorithm and determining its effectiveness. The algorithm learns from the training data and applies the acquired knowledge to the test data, with the test results indicating the Machine Learning algorithm's performance.

However, the CICIDS2017 dataset does not come with predefined training and test sets; instead, it consists of a single, unpartitioned dataset. Consequently, it is necessary to split the data into training and test sets. This is accomplished using the Sklearn library. The commonly preferred split is 70% training data and 30% test data, a ratio also adopted in

this application. The command used for splitting randomly selects data when creating the partitions, a process known as cross-validation. To ensure robustness in the results, the training and test data creation was repeated 10 times. The final results are the arithmetic mean of these repeated operations.

4.2.3 Feature Selection

In this section, the features in the dataset are evaluated to identify which ones are significant for defining specific attacks. Evaluating and selecting the most relevant features is crucial for improving the performance of the Machine Learning algorithms. A comprehensive list of these features, along with detailed descriptions, can be found in Appendix A. This section will implement an approach using four different feature selection methods (Gini Importance, Permutation Importance, Information Gain, Random Forest), based on previous research mentioned in the literature review.

The key differences between these four feature selection methods, including their performance, computation complexity, model specificity, and interpretability, are summarized in Table (5) to provide a clearer understanding of their characteristics and usage.

Table 5. Differences between feature selection methods

Feature Selection Method	Performance	Model Specificity	Computation Complexity	Interpretability
Gini Importance	High	Specific	Medium	Medium
Permutation Importance	High	General	High	Low
Information Gain	Medium	Specific	Low	High
Random Forest	Very High	Specific	High	Medium

The approach focuses on selecting the top 10 features identified as important across all feature selection methods to ensure consistency and facilitate comparison with previous studies. By choosing 10 features, we align our model's feature selection with the standard commonly used in similar research, such as studies (Reis, Maia and Praça, 2019), which also utilized a set of 10 features. This unified approach enables a more direct comparison of our results with those studies, allowing us to evaluate the model's performance on a similar feature scale. After applying several feature selection algorithms, each method yields a ranked list of important features, from which we select the top 10 features commonly identified across all methods. Focusing on this specific number provides a balanced and robust feature subset, reflecting the consensus among methods without overburdening the model with unnecessary complexity. To avoid bias from a single algorithm, features are chosen based on agreement across methods rather than relying solely on feature weights, which can vary significantly across selection techniques. The selection process begins by identifying the most important features as ranked by each algorithm. If a feature, such as 'Bwd Packet Length Std,' is deemed critical by multiple algorithms, it is selected only once, and the next ranked feature from the remaining algorithms is considered to maintain diversity and ensure each algorithm's input is valued. This approach ensures that our final feature set represents a comprehensive, top 10 subset considered important by all algorithms, reinforcing the robustness and generalizability of our model.

The feature selection methods are as follows:

4.2.3.1 **Gini Importance** (Nembrini, König and Wright, 2018)
Gini importance, also known as mean decrease in impurity, is a metric used in decision trees to determine the significance of features. It measures how effectively each feature

reduces uncertainty or impurity in the data when making splits. Features that contribute to larger reductions in impurity across more splits are considered more important. This method is commonly used in tree-based algorithms like RF and gradient boosting. The process begins with the inclusion of necessary libraries, such as 'DecisionTreeClassifier' for constructing a decision tree classifier, and 'matplotlib.pyplot' for visualization. Subsequently, a DecisionTreeClassifier model is trained using the dataset. Following model training, the feature importances are computed using the respective method. The top 15 features with the highest importance scores are then identified and extracted. Finally, Figure (4) represents these important features.

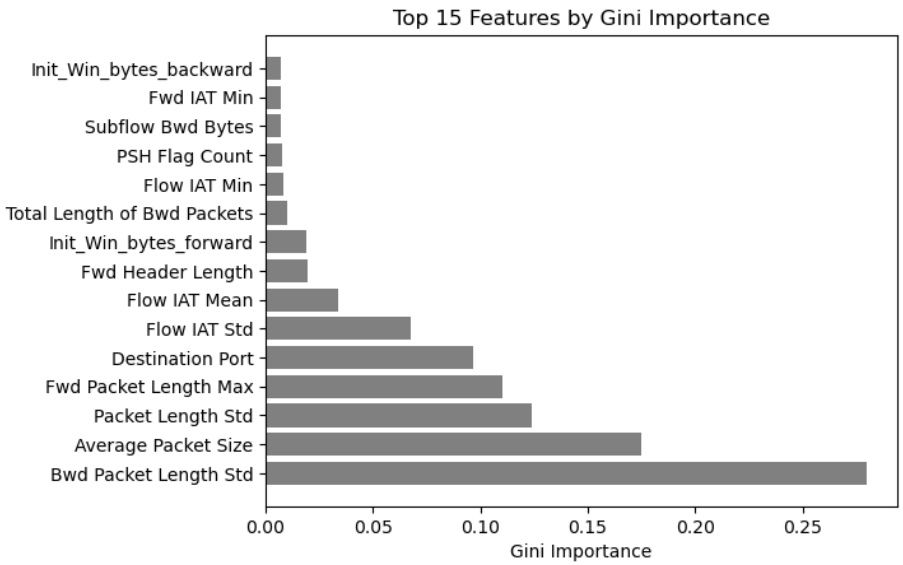

Figure 4. Top 15 features by gini importance

To elaborate on the most important features identified:

Bwd Packet Length Std: Represents the standard deviation of the lengths of packets traveling in the backward direction. A high standard deviation indicates significant variation in packet sizes, an important indicator of network behavior and potential anomalies.

Average Packet Size: Calculates the average size of packets in the network traffic, providing insights into the typical packet size being transmitted. This can affect various network performance metrics and help in detecting unusual traffic patterns.

Packet Length Std: Measures the standard deviation of packet lengths for packets traveling in both directions, helping to understand the variability in packet sizes within the network.

4.2.3.2 **Permutation Importance** (Altmann *et al.*, 2010)

Permutation importance evaluates the significance of a feature by measuring the impact on model performance when the feature's values are randomly shuffled. If shuffling a feature's values significantly decreases the model's accuracy, the feature is deemed important. This method is versatile as it can be applied to any trained model and uses the test set to assess the effect of shuffling, treating the model as a black box. The process used the 'permutation_importance()' function from 'sklearn.inspection' to calculate the importance of the permutation. The top 15 features with the highest average permutation importance scores are then selected, and their corresponding names are extracted from the columns of the dataset. Figure (5) represents these important features.

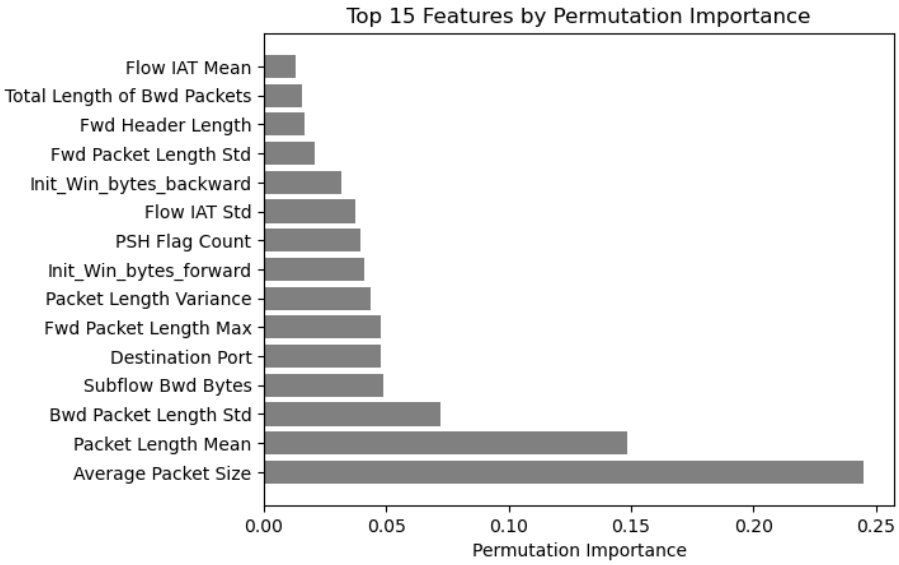

Figure 5. Top 15 features by premutation importance

The most important features identified:

Average Packet Size: Its high permutation importance indicates that altering this feature's values significantly impacts the model's performance, underscoring its predictive value.

Packet Length Mean: Represents the mean length of packets in the network traffic. A significant decrease in model accuracy upon shuffling this feature highlights its importance in maintaining the integrity of the model's predictions.

Bwd Packet Length Std: Indicates significant variation in packet sizes, and its importance is reflected in the substantial drop in accuracy when this feature is shuffled.

Comparison with previous results: The feature Average Packet Size appears prominently in both Gini Importance and Permutation Importance, reinforcing its critical role in the model's performance. Similarly, Bwd Packet Length Std is significant in both methods, highlighting its importance in understanding network behavior.

4.2.3.3 **Information Gain** (Kent, 1983)

Information Gain is a widely used filter-based feature selection technique. It ranks attributes by calculating their entropy, a measure of uncertainty, to determine how much information each feature provides about a specific class. This method helps reduce noise from irrelevant features and identifies the most informative features. In this method, the mutual information is calculated using 'mutual_info_classif()' from 'sklearn.feature_selection', and then the top 15 features that display the highest mutual information scores are selected. These pivotal features were then extracted based on their respective indicators. Figure (6) represents these important features according to the information acquisition.

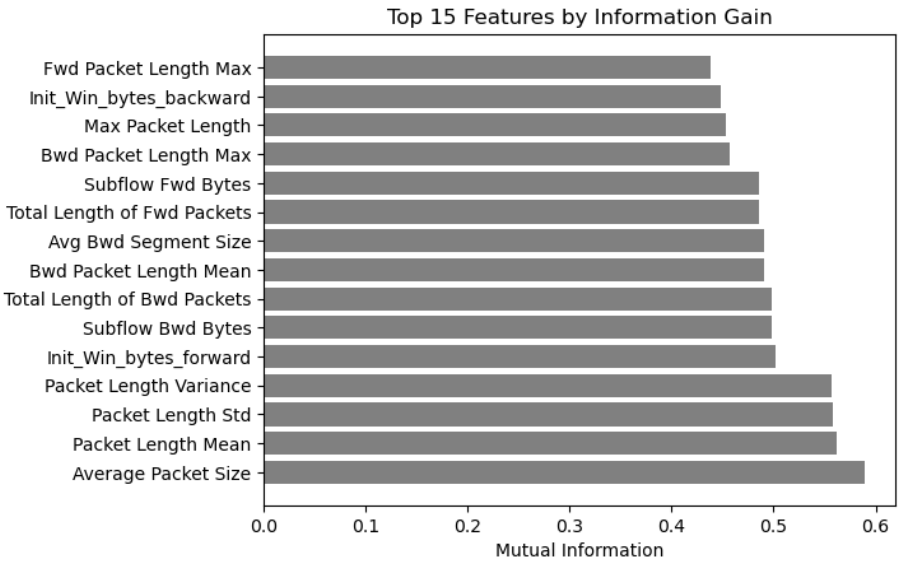

Figure 6. Top 15 features by information gain

The most important features identified:

Average Packet Size: Indicates a significant amount of information about the specific class being analyzed.

Packet Length Mean: Plays a significant role in providing valuable information

about the specific class under consideration.

Packet Length Std: Highlights its importance in capturing essential information about the specific class.

Comparison with previous results: The Average Packet Size feature is consistently important across Gini Importance, Permutation Importance, and IG methods, demonstrating its robust predictive power. Additionally, Packet Length Std and Packet Length Mean are identified as crucial features in multiple methods, underscoring their utility in network analysis.

4.2.3.4 **Random Forest (RF)** (Kursa, 2014)

Random Forest is a technique that combines multiple decision trees into a single model. It involves building prediction trees through bootstrap sampling and using random subsets of predictors for decision-making. The final prediction is made by aggregating the results of each decision tree through a majority vote. RF is known for its ensemble classifier approach, with each decision tree in the ensemble randomly selecting attributes for separation. The process begins by training a RandomForestClassifier, and then the feature importance is obtained through the `feature_importances_` attribute of the trained classifier, resulting in the relative importance scores of each feature. The top 15 features with the highest importance scores were then identified and extracted based on their respective indicators. Figure (7) represents these important features according to the RF.

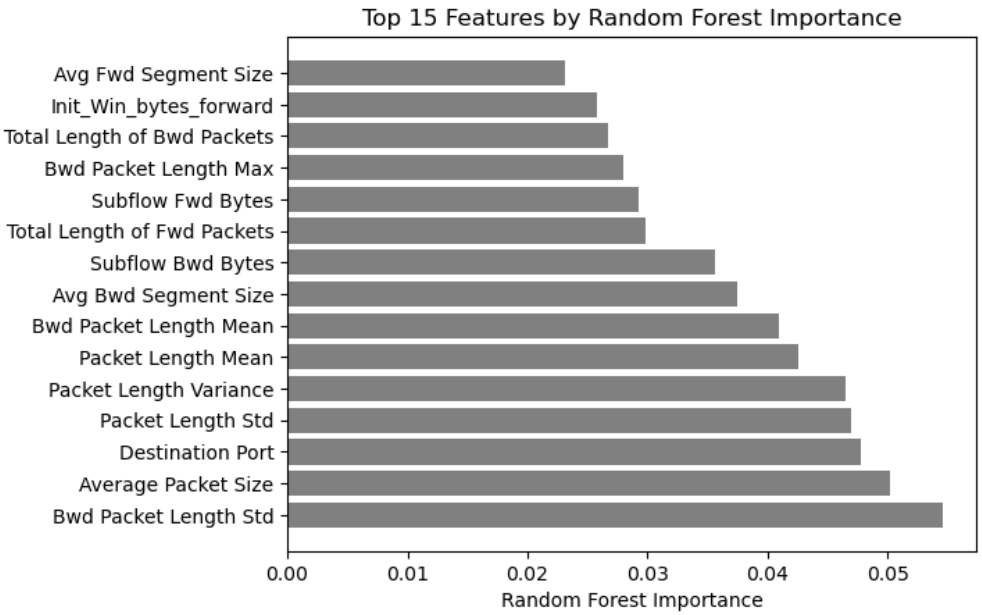

Figure 7. Top 15 features by random forest

The most important features identified:

Bwd Packet Length Std: Indicates significant variation in packet sizes, crucial for making accurate predictions.

Average Packet Size: Demonstrates its significant contribution to the RF model's predictive power.

Destination Port: Plays a crucial role in understanding network communication patterns and identifying potential vulnerabilities.

Comparison with previous results: The Bwd Packet Length Std and Average Packet Size features are consistently rated as highly important across all four methods, indicating their substantial influence on model performance. The inclusion of Destination Port in the RF importance list highlights an additional perspective on network analysis that complements the other methods.

4.2.3.5 Consensus-Based Feature Selection

The final feature set comprises the top 10 features that are consistently ranked highly by all methods, ensuring high importance and optimal performance when applied. This consensus-based approach enhances the robustness of the selected features and prevents model bias toward features identified by a single algorithm. For example, top features extracted from each method might include ‘Bwd Packet Length Std’ and ‘Average Packet Size’ as highly significant features. Consequently, these features are selected based on agreement across algorithms, ensuring a balanced representation. The feature selection approach and the final top 10 features identified from the dataset are presented in Table (6). This standardized selection allows for consistent comparison with prior studies while optimizing both detection accuracy and computational efficiency.

Table 6. Top 10 features ranked by our model

Number of Features	Features
Top 10 Features	'Bwd Packet Length Std', 'Average Packet Size', 'Packet Length Mean', 'Init_Win_bytes_forward', 'Packet Length Std', 'Packet Length Variance', 'Bwd Packet Length Mean', 'Destination Port', 'Fwd Packet Length Max', 'Subflow Bwd Bytes'

4.2.4 Implementation of Machine Learning Algorithms

In this section, we implement and evaluate four selected Machine Learning algorithms—NB, RF, AdaBoost, and K-Nearest Neighbors (KNN)—to assess the effectiveness of our feature selection approach. The primary objective is to determine whether our feature selection method outperforms those used in previous studies by comparing it directly with techniques from studies (Reis, Maia and Praça, 2019) and (Stiawan *et al.*, 2020), which employed different feature selection methodologies. To ensure a fair and comprehensive evaluation, we address potential differences in experimental setups and datasets used in prior research. Therefore, we apply the same 10 features selected by our approach

alongside the 10 features derived from the feature selection techniques in studies (Reis, Maia and Praça, 2019) and (Stiawan *et al.*, 2020). This process enables a rigorous comparison, respecting the original methods while evaluating the performance of our feature selection technique under standardized testing conditions.

Following the training and testing of the Machine Learning models using each of these three sets of 10 features, we provide a thorough analysis of the results. Key performance metrics, including accuracy, F1 score, and computational efficiency, will be compared across these feature sets to determine the optimal feature selection approach. This comparative analysis aims to clarify whether our method enhances IDS performance by optimizing feature selection under identical testing conditions. The detailed results and analysis of these experiments will be presented in Chapter Five. This upcoming section will offer insights into the relative effectiveness of each feature selection approach, providing an in-depth comparison to identify the best-performing system for feature selection.

Chapter 5: Experimental Results and Discussion

This chapter presents and analyzes the experimental results obtained using selected feature sets and Machine Learning algorithms on the CICIDS2017 dataset. The primary objective is to evaluate the performance of the intrusion detection system (IDS) based on metrics such as accuracy, F1 score, precision, recall, and computational efficiency, including training and testing times. To assess the impact of feature selection, models trained on the top 10 most relevant features are compared, providing insight into the effectiveness of the feature selection method in enhancing IDS performance.

5.1 Dataset Distribution and Attack Class Analysis

In this section, we present the distribution of the dataset used for evaluation, specifically focusing on the proportions of benign (normal) and attack samples. As shown in Figure (8), the dataset is highly imbalanced, with benign samples accounting for 80.3% of the total data, while attack samples constitute 19.7%. The imbalance is clearly visible in the chart, highlighting the dominance of benign traffic in the dataset.

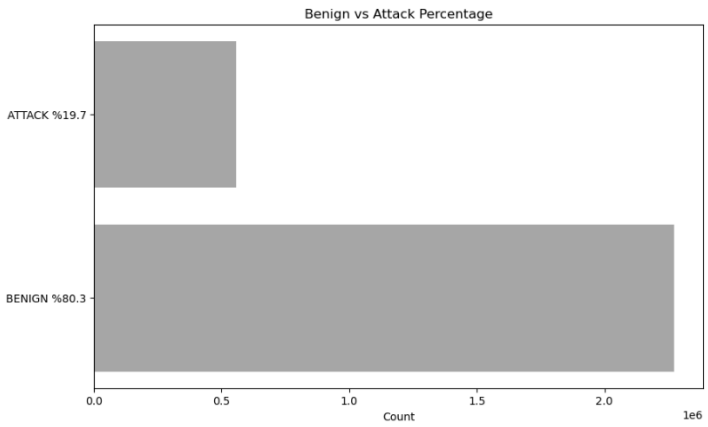

Figure 8. Proportion of benign and attack samples

It is important to note that this imbalance in the dataset reflects the real-world scenario,

where benign traffic typically outweighs malicious traffic. While this imbalance might affect model performance, it is essential for accurately simulating real-world conditions where the goal is to detect relatively rare attack patterns among a majority of benign activity. The focus here is not on balancing the dataset but on testing our feature selection and classification methods in the same context as previous studies to ensure consistency in results. To further analyze the dataset, we present the distribution of attack types in Figure (9), where the number of attacks for each attack type is shown. This figure clearly illustrates the significant differences in the frequency of each attack type, with some attacks occurring much more frequently than others.

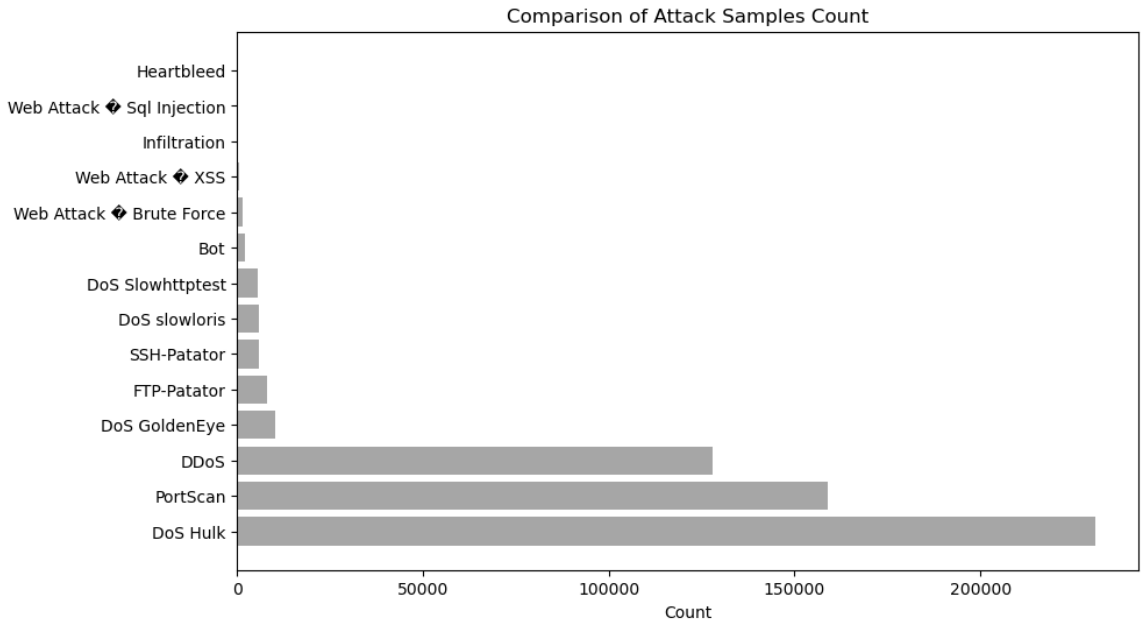

Figure 9. Number of attacks per attack type

It is important to clarify that no balancing techniques, such as oversampling or under sampling, were applied to the attack types in the dataset. This decision was made deliberately, as the studies we are comparing against did not perform such balancing either. The goal is to ensure that our results are comparable and not influenced by any preprocessing steps that could skew the outcomes. By keeping the dataset as it is, we

preserve the integrity of the experimental conditions used in the original studies and ensure that any differences in results are due to the feature selection and Machine Learning techniques, not the data preprocessing steps.

5.2 Features Selected

The top 10 features selected for analysis based on the filtering model utilized in our study are presented in Table (7), along with a description for each feature. This table provides insight into the characteristics of each feature and highlights its contribution to enhancing the Intrusion Detection System (IDS) process. These features were chosen based on their relevance and effectiveness in detecting various types of intrusions within the CICIDS2017 dataset.

Table 7. Descriptions of the top 10 selected features

Feature	Description
Bwd Packet Length Std	The standard deviation of packet lengths in the backward direction, useful for detecting anomalies.
Average Packet Size	The average size of packets in a session, indicating the overall data flow per connection.
Packet Length Mean	The mean length of packets, helps in identifying unusual packet sizes.
Init_Win_bytes_forward	Initial window size in bytes for packets sent forward, indicating network behavior per session.
Packet Length Std	Standard deviation of packet lengths, indicating variation and potential abnormal patterns.
Packet Length Variance	Variance of packet lengths in communication, useful for recognizing irregular data patterns.
Bwd Packet Length Mean	Mean packet length in the backward direction, aids in distinguishing legitimate from anomalous flows.
Destination Port	The port number to which packets are sent, useful for identifying the type of service in use.
Fwd Packet Length Max	Maximum packet length in the forward direction, helps detect large data transfers or abnormal packets.
Subflow Bwd Bytes	Total bytes in the backward subflow, useful for analyzing data exchange direction and volume.

Additionally, Table (8) presents the top 10 features selected in previous studies, namely studies (Reis, Maia and Praça, 2019) and (Stiawan *et al.*, 2020), which our research

compares against. These studies represent different approaches to feature selection, and by including their feature sets in this comparison, we aim to evaluate the effectiveness of our chosen features against those previously used in the literature.

Table 8. Top 10 features for previous studies

Study	Features
(Reis, Maia and Praça, 2019)	'Packet Length Std', 'Total Length of Bwd Packets', 'Subflow Bwd Bytes', 'Destination Port', 'Packet Length Variance', 'Bwd Packet Length Mean', 'Avg Bwd Segment Size', 'Bwd Packet Length Max', 'Init_Win_bytes_backward', 'Total Length of Fwd Packets'
(Stiawan *et al.*, 2020)	'Fwd IAT Min', 'Init_Win_bytes_forward', 'Destination Port', 'Init_Win_bytes_backward', 'Flow IAT Min', 'Bwd Packet Length Min', 'Subflow Fwd Bytes', 'Total Fwd Packets', 'Total Length of Bwd Packets', 'Bwd Packet Length Mean'

To assess the performance of the selected feature sets, we apply Machine Learning algorithms, including NB, RF Classifier, AdaBoost, and K-Nearest Neighbors (KNN), to both the features selected by our approach and the features from the comparison studies. This will allow us to evaluate how our feature selection strategy impacts the IDS performance compared to the established methods. The results of these evaluations are critical for understanding the practical benefits of our approach in improving IDS accuracy and efficiency.

5.3 Results and Analysis

The performance of the Machine Learning models on the top 10 common features is summarized in Table (9), which combines results from our model, previous studies, and these studies re-evaluated using our model's feature selection approach. The table demonstrates that our model consistently outperforms alternative methods, exhibiting superior accuracy and more efficient processing times. Specifically, NB achieved a balanced accuracy of 82.5% and an F1-score of 73%, with precision and recall values of 72% and 70%, respectively, highlighting its reliability for moderately balanced performance. RF proved to be more robust, with an accuracy of 96.1% and an F1-score

of 92%, along with precision and recall scores of 97% and 89%, reflecting its high dependability in distinguishing benign from malicious samples. Notably, AdaBoost and K-Nearest Neighbors (KNN) models delivered outstanding results, both surpassing 98% in accuracy, precision, recall, and F1-score, although KNN required significantly longer computational time than other algorithms. The Time (Sec) metric, as presented in the table, represents the total elapsed time each algorithm required to complete the entire intrusion detection process, from start to finish, capturing the computational demand of each model. This metric is essential for evaluating real-time applicability, where rapid detection is crucial. Collectively, these findings affirm that our model's approach to feature selection optimizes both detection performance and processing speed, demonstrating clear advantages over previous studies and confirming the effectiveness of our chosen feature selection methodology.

Table 9. Top 10 feature model performance comparison

Research	Feature Set	ML Algorithm	Accuracy	Precision	Recall	F1-Score	Time (Sec)
Study (Stiawan *et al.*, 2020)	Top 10 Features	Random Forest	99.81%	99%	99%	N/A	1908
		Naive Bayes	43.58%	91%	43%	N/A	11
Study (Reis, Maia and Praça, 2019)		Random Forest	N/A	93%	92%	91%	N/A
Our Study		Naive Bayes	84.56%	72%	70%	73%	2.30
		Random Forest	96.10%	97%	89%	92%	11.81
		AdaBoost	98.56%	98%	99%	98%	100.50
		K-Nearest Neighbour	99.38%	99%	99%	99%	252.20
Feature Set of Study (Stiawan *et al.*, 2020) **Applied to Our Model**		Random Forest	94.30%	95%	90%	92%	26.20
		Naive Bayes	75.80%	70%	66%	68%	5.70

Feature Set of Study (Reis, Maia and Praça, 2019) **Applied to Our Model**		Random Forest	93.75%	94%	91%	92%	19.50

In this table:

- N/A is used to indicate that the specific metric is not available for that entry.
- NaN is used to indicate that the specific numerical value is not available or undefined.

Figure (10) illustrates a detailed comparison of the RF model's performance across different studies in terms of accuracy and computational time. The models include results from a Study (Reis, Maia and Praça, 2019), Study (Stiawan *et al.*, 2020), and Our Study, each evaluated based on accuracy percentage and processing time (in seconds). Our model exhibits the highest accuracy at 96.10%, outperforming both Study (Stiawan *et al.*, 2020) at 94.30% and the Study (Reis, Maia and Praça, 2019) at 93.75%. Additionally, our study demonstrates a significantly lower processing time of 11.8 seconds, highlighting its computational efficiency compared to Study (Stiawan *et al.*, 2020) (26 seconds) and the Study (Reis, Maia and Praça, 2019) (19.5 seconds). The comparison underscores the effectiveness of our feature selection approach, achieving superior accuracy while reducing computational demands, which is critical for real-time applications in intrusion detection systems.

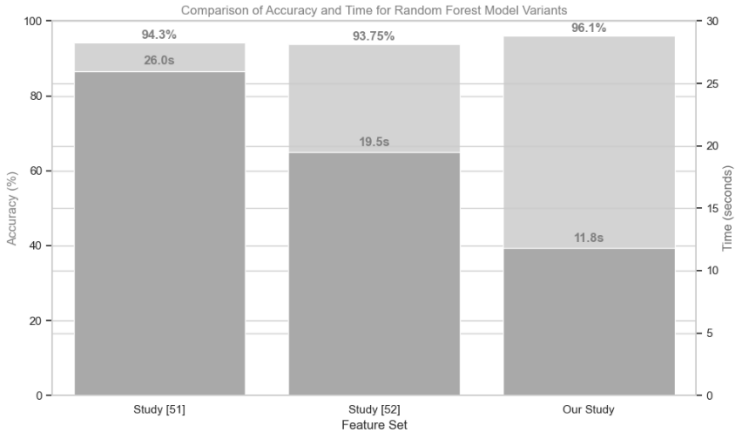

Figure 10. Comparison of accuracy and time for RF model

Based on Figure (11-12), the K-Nearest Neighbors (KNN) algorithm achieves the highest median F-measure score, approximately 0.99, among the four algorithms. Its box plot reveals a very narrow interquartile range (IQR), indicating that KNN's F-measure scores are tightly clustered around the median, reflecting highly consistent performance on this dataset. The RF algorithm follows with a median F-measure score of about 0.92. Although this score is slightly lower than KNN's, it is still high. However, RF's IQR is wider than that of KNN, suggesting a broader spread in F-measure scores and slightly more variability in performance. AdaBoost achieves a median F-measure score around 0.98, which, while high, is associated with a broader IQR than KNN's, suggesting more variability in the scores. Lastly, NB has the lowest median F-measure score, approximately 0.73. Its box plot shows the widest range of scores among all four models, indicating the greatest performance variability.

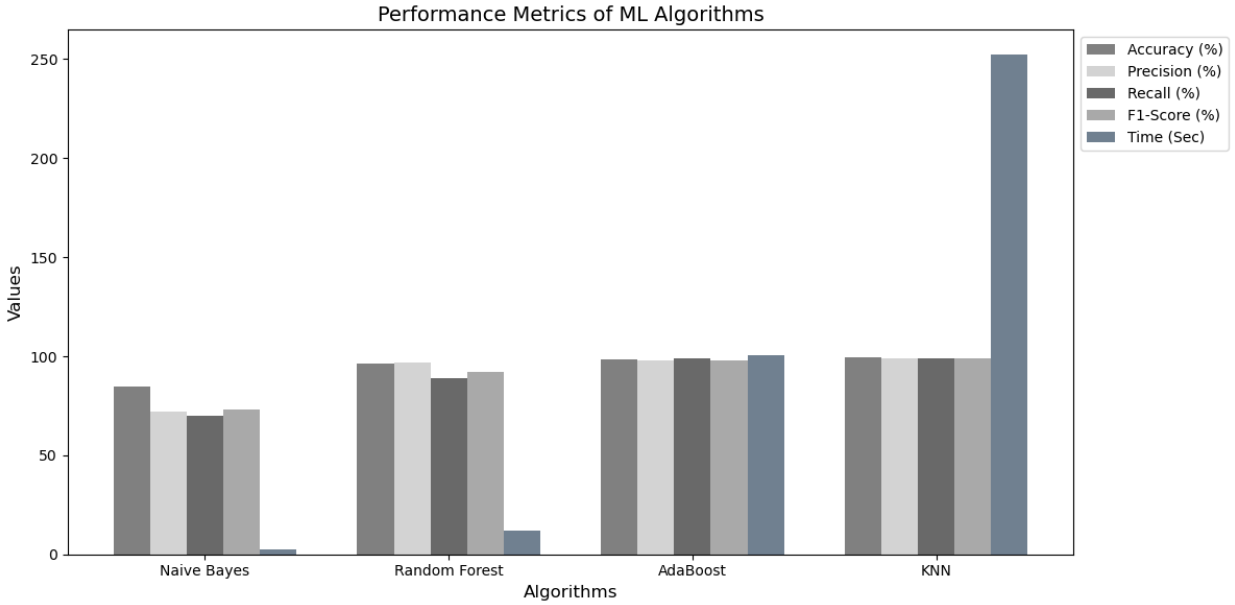

Figure 11. Comparison of ML algorithms based performance metrics

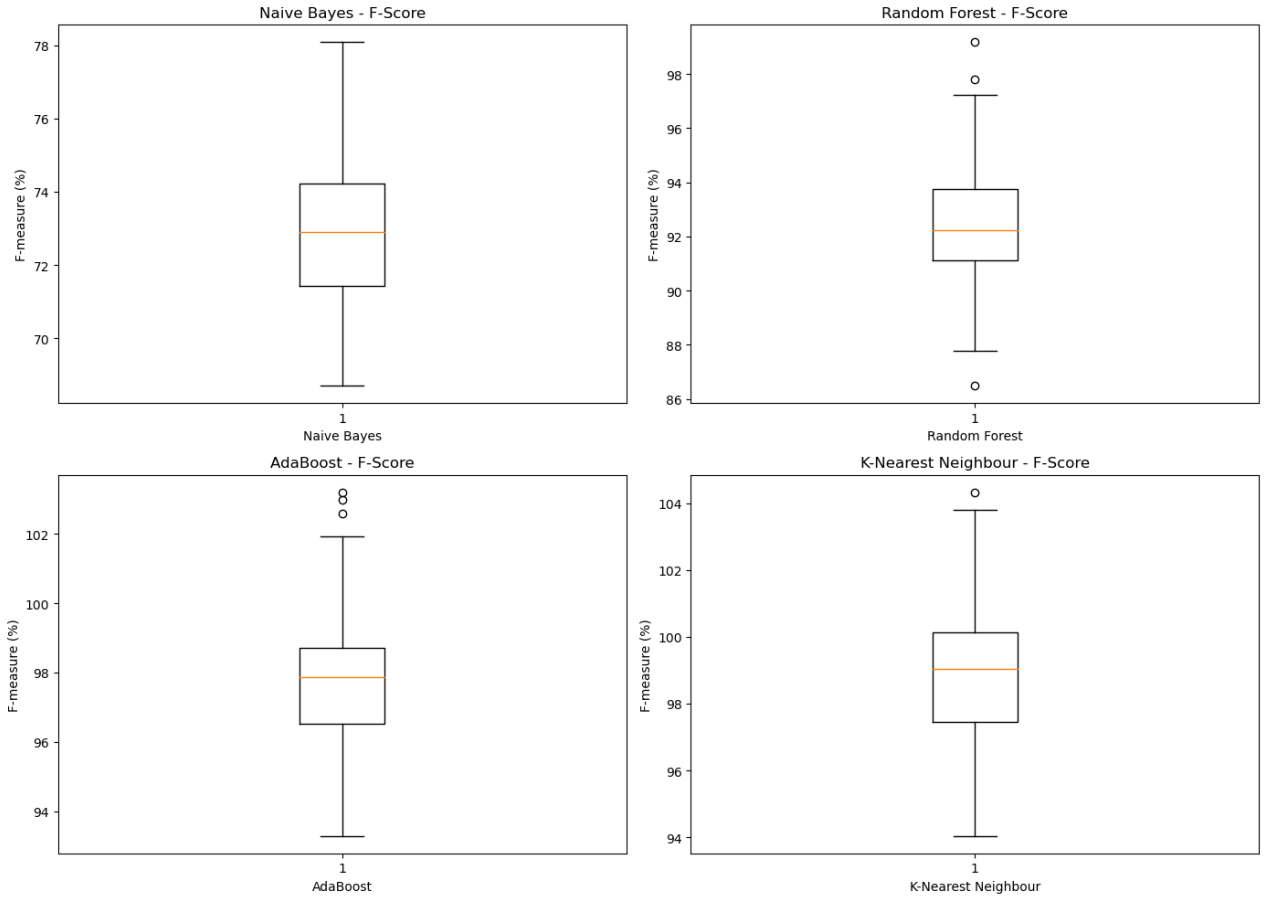

Figure 12. Comparison of F-measure for models

5.4 Evaluation

In this section, we evaluate the performance of our Intrusion Detection System (IDS) by comparing the results of our experiments with those from two previous studies, specifically focusing on feature selection and Machine Learning algorithms. Despite incorporating feature sets from these studies into our model, our approach, which utilizes the top 10 features, demonstrates superior performance in terms of both accuracy and computational efficiency.

Study (Stiawan *et al.*, 2020) reported a strong performance with its RF classifier, achieving an accuracy of 99.81%, along with precision and recall both at 99%. However, the computational time for this model was significantly high, taking 1908 seconds, which presents challenges for real-time applications. Additionally, NB showed a much lower accuracy of 43.58%, but with a very short computational time of only 11 seconds. While NB offers fast processing, it is less effective in terms of accuracy. Study (Reis, Maia and Praça, 2019) evaluated a larger feature set of 69 features, reporting an accuracy of 93.75% with precision at 94%, recall at 91%, and F1-score at 92% for their RF model. Although these results are respectable, they still fall short when compared to the performance of our model. Furthermore, while the computational time for Study (Reis, Maia and Praça, 2019) was not provided, it is likely that the extensive feature set of 69 features required more processing time, which could be detrimental to real-time IDS systems. In contrast, our study, using only the top 10 features selected through our feature selection method, achieved remarkable results. The RF model reached an accuracy of 96.10%, with precision at 97%, recall at 89%, and F1-score at 92%. Our model not only surpassed the accuracy and precision of Study (Stiawan *et al.*, 2020)'s RF, but it also outperformed it in computational efficiency, taking only 11.81 seconds compared to the 1908 seconds

required by Study (Stiawan *et al.*, 2020). Additionally, our AdaBoost model achieved an accuracy of 98.56%, with precision and recall both at 98%, and an F1-score of 98%. Although the computational time was higher at 100.50 seconds, it remains an efficient option for many practical applications. The K-Nearest Neighbour (KNN) model showed the highest accuracy of 99.38%, with precision and recall both at 99%, and an F1-score of 99%. However, KNN's computational time of 252.20 seconds makes it less suitable for real-time IDS applications. Our NB model, with a computational time of 2.30 seconds, achieved an accuracy of 84.56%. While this is the fastest model, it does not perform as well as other algorithms in terms of detection accuracy. Even after applying feature sets from Study (Stiawan *et al.*, 2020), based on IG, our top 10 feature selection model still outperforms their results. The RF model from Study (Stiawan *et al.*, 2020) achieved 99.81% accuracy but required a long processing time of 45.20 seconds, whereas our RF model, using just the top 10 features, achieved 96.10% accuracy and only required 11.81 seconds for processing. This highlights that our feature selection methodology is not only effective in maintaining high accuracy but also optimized for faster computational performance, making it more suitable for real-time IDS systems. By reducing the feature set and optimizing the feature selection process, we have achieved a balance between high detection accuracy and fast processing times, which is critical for deploying Intrusion Detection Systems in real-world environments.

Chapter 6: Conclusions and Future Works

This chapter summarizes the key findings of the research, discusses the implications of the results, and highlights the contributions to the field of intrusion detection systems. It also outlines potential directions for future research, including areas for improvement and further exploration in feature selection and machine learning techniques for IDS.

6.1 Conclusion

This research embarked on a meticulous journey to identify the most critical features for Machine Learning models in the context of Intrusion Detection Systems (IDS). By leveraging multiple feature selection algorithms, we identified the top 10 features that consistently emerged as significant across all methods. This approach enabled a comprehensive evaluation of various Machine Learning models, providing deep insights into the trade-offs between model complexity, accuracy, and computational efficiency. The analysis of the top 10 features, including 'Bwd Packet Length Std', 'Average Packet Size', 'Packet Length Mean', and others, revealed substantial performance improvements in Machine Learning models. NB achieved an accuracy of 84.56% with remarkable computational efficiency, processing data in just a few seconds. RF stood out with an accuracy of 96.10%, precision at 97%, recall at 89%, and an F1-score of 92%, establishing it as a robust model for IDS. Additionally, AdaBoost and KNN models demonstrated exceptional performance with accuracy, precision, recall, and F1-scores all at 98% or higher, though KNN exhibited significantly longer computational times. Even when applying feature sets extracted from the previous studies under the same conditions as those used for our optimized feature set, the results did not surpass the performance of our model. Specifically, the accuracy from Study [52] reached only 94.30%, and Study [51] achieved 93.75%, with processing times of 26.20 seconds and 19.50 seconds,

respectively. This comparison clearly shows that our feature selection methodology outperforms previous approaches, achieving better accuracy while maintaining significantly lower computational times. RF, when applied with the top 10 features, achieved an accuracy of 96.10%, with precision at 97%, recall at 89%, and an F1-score of 92%. This performance was accompanied by a remarkable reduction in computational time (11.81 seconds), in stark contrast to the over 1900 seconds required by previous models to achieve similar accuracy. These findings underscore the importance of balancing model complexity, accuracy, and computational efficiency in IDS. The high performance achieved with the top 10 features demonstrates the effectiveness of our feature selection methodology. The impressive results from RF, AdaBoost, and KNN models highlight the trade-offs between computational time and accuracy, emphasizing the need for efficient, real-time models in IDS deployments where quick detection is critical.

6.2 Future Works

In future research, it would be beneficial to explore the integration of deep learning techniques, such as convolutional or recurrent neural networks, to improve Intrusion Detection Systems (IDS) in handling complex and evolving cyber threats. Additionally, evaluating the model with real-world network traffic, rather than relying solely on international datasets, could provide valuable insights into its performance in diverse environments. These directions hold great potential for advancing IDS and improving network security.

References

3.12.3 Documentation (2024). Available at: https://docs.python.org/3/ (Accessed: 2 May 2024).

68% of business leaders feel that their cybersecurity risks are increasing (no date). Available at: https://truefort.com/2023-cybersecurity-statistics/ (Accessed: 28 May 2024).

Altmann, A. *et al.* (2010) 'Permutation importance: a corrected feature importance measure', *Bioinformatics*, 26(10), pp. 1340–1347.

Ambusaidi, M.A. *et al.* (2016) 'Building an intrusion detection system using a filter-based feature selection algorithm', *IEEE transactions on computers*, 65(10), pp. 2986–2998.

Axelsson, S. (2000) *Intrusion Detection Systems: A Survey and Taxonomy*.

Belavagi, M.C. and Muniyal, B. (2016) 'Performance evaluation of supervised machine learning algorithms for intrusion detection', *Procedia Computer Science*, 89, pp. 117–123.

Bhuyan, M.H., Bhattacharyya, D.K. and Kalita, J.K. (2013a) 'Network anomaly detection: methods, systems and tools', *Ieee communications surveys & tutorials*, 16(1), pp. 303–336.

Bhuyan, M.H., Bhattacharyya, D.K. and Kalita, J.K. (2013b) 'Network anomaly detection: methods, systems and tools', *Ieee communications surveys & tutorials*, 16(1), pp. 303–336.

Breiman, L. (2001) 'Random forests', *Machine learning*, 45, pp. 5–32.

Buczak, A.L. and Guven, E. (2015) 'A survey of data mining and machine learning methods for cyber security intrusion detection', *IEEE Communications surveys & tutorials*, 18(2), pp. 1153–1176.

Chandrashekar, G. and Sahin, F. (2014) 'A survey on feature selection methods', *Computers & Electrical Engineering*, 40(1), pp. 16–28. Available at: https://doi.org/10.1016/J.COMPELECENG.2013.11.024.

Chebrolu, S., Abraham, A. and Thomas, J.P. (2005) 'Feature deduction and ensemble design of intrusion detection systems', *Computers & security*, 24(4), pp. 295–307.

Dong, R.-H., Shui, Y.-L. and Zhang, Q.-Y. (2021) 'Intrusion detection model based on feature selection and random forest', *International Journal of Network Security*, 23(6), pp. 985–996.

Géron, A. (2022) *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. ‘ O’Reilly Media, Inc.’

Gharib, A. *et al.* (2016) ‘An evaluation framework for intrusion detection dataset’, in *2016 International conference on information science and security (ICISS)*. IEEE, pp. 1–6.

Guyon, I. and De, A.M. (2003) *An Introduction to Variable and Feature Selection André Elisseeff*, *Journal of Machine Learning Research*.

Haines, J.W. *et al.* (2001) ‘DARPA intrusion detection evaluation: Design and procedures’, *Lincoln Laboratory, Massachusetts Institute of Technology* [Preprint].

Internet World Stats (2024) *INTERNET GROWTH STATISTICS*. Available at: https://www.internetworldstats.com/emarketing.htm (Accessed: 28 April 2024).

Jamadar, R.A. (2018) ‘Network intrusion detection system using machine learning’, *Indian Journal of Science and Technology*, 7(48), pp. 1–6.

Jany Shabu, S. *et al.* (2021) ‘Automatic feature selection and ensemble classifier for intrusion detection’, *iopscience.iop.orgC Lin, A Li, R JiangJournal of Physics: Conference Series, 2021•iopscience.iop.org*, 1856, p. 12067. Available at: https://doi.org/10.1088/1742-6596/1856/1/012067.

Kent, J.T. (1983) ‘Information gain and a general measure of correlation’, *Biometrika*, 70(1), pp. 163–173.

Khan, L., Awad, M. and Thuraisingham, B. (2007) ‘A new intrusion detection system using support vector machines and hierarchical clustering’, *The VLDB journal*, 16, pp. 507–521.

Kostas, K. (2018) ‘Anomaly detection in networks using machine learning’, *Research Proposal*, 23, p. 343.

Kotsiantis, S.B., Zaharakis, I. and Pintelas, P. (2007) ‘Supervised machine learning: A review of classification techniques’, *Emerging artificial intelligence applications in computer engineering*, 160(1), pp. 3–24.

Kramer, O. and Kramer, O. (2013) ‘K-nearest neighbors’, *Dimensionality reduction with unsupervised nearest neighbors*, pp. 13–23.

Kurniabudi, K. *et al.* (2021) ‘Important features of CICIDS-2017 Dataset for anomaly detection in high dimension and imbalanced class dataset’, *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)*, 9(2), pp. 498–511.

Kursa, M.B. (2014) ‘Robustness of Random Forest-based gene selection methods’, *BMC bioinformatics*, 15, pp. 1–8.

Leung, K. and Leckie, C. (2005) ‘Unsupervised anomaly detection in network intrusion detection using clusters’, in *Proceedings of the Twenty-eighth Australasian conference on Computer Science-Volume 38*, pp. 333–342.

‘Matplotlib — Visualization with Python’ (no date). Available at: https://matplotlib.org/ (Accessed: 2 May 2024).

Mhawi, D.N., Aldallal, A. and Hassan, S. (2022) ‘Advanced Feature-Selection-Based Hybrid Ensemble Learning Algorithms for Network Intrusion Detection Systems’, *Symmetry 2022, Vol. 14, Page 1461*, 14(7), p. 1461. Available at: https://doi.org/10.3390/SYM14071461.

Mukherjee, S. and Sharma, N. (2012a) ‘Intrusion detection using naive Bayes classifier with feature reduction’, *Procedia Technology*, 4, pp. 119–128.

Mukherjee, S. and Sharma, N. (2012b) ‘Intrusion detection using naive Bayes classifier with feature reduction’, *Procedia Technology*, 4, pp. 119–128.

Nembrini, S., König, I.R. and Wright, M.N. (2018) ‘The revival of the Gini importance?’, *Bioinformatics*, 34(21), pp. 3711–3718.

‘NumPy - ’ (no date). Available at: https://numpy.org/ (Accessed: 2 May 2024).

Özgür, A. and Erdem, H. (2016) ‘A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015’.

‘pandas - Python Data Analysis Library’ (no date). Available at: https://pandas.pydata.org/ (Accessed: 2 May 2024).

Peng, H. *et al.* (2018) ‘An improved feature selection algorithm based on ant colony optimization’, *Ieee Access*, 6, pp. 69203–69209.

Popoola, E. and Adewumi, A.O. (2017) ‘Efficient Feature Selection Technique for Network Intrusion Detection System Using Discrete Differential Evolution and Decision.’, *Int. J. Netw. Secur.*, 19(5), pp. 660–669.

Reis, B., Maia, E. and Praça, I. (2019) ‘Selection and performance analysis of CICIDS2017 features importance’, in *International Symposium on Foundations and Practice of Security*. Springer, pp. 56–71.

Sarker, I.H. *et al.* (2020) ‘Cybersecurity data science: an overview from machine learning perspective’, *Journal of Big data*, 7, pp. 1–29.

Schapire, R.E. (2003) ‘The boosting approach to machine learning: An overview’, *Nonlinear estimation and classification*, pp. 149–171.

scikit-learn: machine learning in Python — scikit-learn 1.4.2 documentation (no date). Available at: https://scikit-learn.org/stable/ (Accessed: 2 May 2024).

Shah, S.A.R. and Issac, B. (2018) 'Performance comparison of intrusion detection systems and application of machine learning to Snort system', *Future Generation Computer Systems*, 80, pp. 157–170. Available at: https://doi.org/10.1016/j.future.2017.10.016.

Sharafaldin, I. *et al.* (2018) 'Towards a reliable intrusion detection benchmark dataset', *Software Networking*, 2018(1), pp. 177–200.

Sharafaldin, I., Lashkari, A.H. and Ghorbani, A.A. (2018) 'Toward generating a new intrusion detection dataset and intrusion traffic characterization.', *ICISSp*, 1, pp. 108–116.

Sommer, R. and Paxson, V. (2010) 'Outside the closed world: On using machine learning for network intrusion detection', *Proceedings - IEEE Symposium on Security and Privacy*, pp. 305–316. Available at: https://doi.org/10.1109/SP.2010.25.

Steve Morgan (2019) 'Global Ransomware Damage Costs Predicted To Reach $20 Billion (USD) By 2021', 21 October. Available at: https://cybersecurityventures.com/global-ransomware-damage-costs-predicted-to-reach-20-billion-usd-by-2021/ (Accessed: 16 August 2024).

Steve Morgan (2020) 'Cybercrime To Cost The World $10.5 Trillion Annually By 2025', 13 November. Available at: https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016/ (Accessed: 16 August 2024).

Stiawan, D. *et al.* (2020) 'CICIDS-2017 dataset feature analysis with information gain for anomaly detection', *IEEE Access*, 8, pp. 132911–132921.

Tao, P., Sun, Zhe and Sun, Zhixin (2018) 'An improved intrusion detection algorithm based on GA and SVM', *Ieee Access*, 6, pp. 13624–13631.

Tavallaee, M. *et al.* (2009) 'A detailed analysis of the KDD CUP 99 data set', in *2009 IEEE symposium on computational intelligence for security and defense applications*. Ieee, pp. 1–6.

Thomas, C., Sharma, V. and Balakrishnan, N. (2008) 'Usefulness of DARPA dataset for intrusion detection system evaluation', in *Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security 2008*. SPIE, pp. 164–171.

Ting, S.L., Ip, W.H. and Tsang, A.H.C. (2011) 'Is Naive Bayes a good classifier for document classification', *International Journal of Software Engineering and Its Applications*, 5(3), pp. 37–46.

Title in English: The Effect of Different Features and Algorithms on the Performance of the IDS Comparison and Analysis (no date).

University of California, I. (1999) *KDD Cup 1999 Data*. Available at: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. (Accessed: 28 April 2024).

Wang, W. *et al.* (2015) ‘Constructing important features from massive network traffic for lightweight intrusion detection’, *IET Information Security*, 9(6), pp. 374–379.

What is the cost of a data breach in 2020? - IBM Z and LinuxONE Community (no date). Available at: https://community.ibm.com/community/user/ibmz-and-linuxone/blogs/erin-zhang1/2020/11/11/cost-of-a-data-breach?communityKey=406e5630-08ab-45a7-8592-d1c960f86311 (Accessed: 28 May 2024).

World Economic Forum (2024) *2023 was a big year for cybercrime – here’s how we can make our systems safer | World Economic Forum*. Available at: https://www.weforum.org/agenda/2024/01/cybersecurity-cybercrime-system-safety/#:~:text=The%20global%20cost%20of%20cybercrime,on%20the%20US%20State%20Department. (Accessed: 15 August 2024).

Yin, C. *et al.* (2017) ‘A deep learning approach for intrusion detection using recurrent neural networks’, *Ieee Access*, 5, pp. 21954–21961.

Appendices

Appendix A: The List of The Features (Sharafaldin *et al.*, 2018)

The table below provides a comprehensive list of the 79 features included in the CICIDS2017 dataset. These features represent various network traffic attributes designed to aid in the detection of network intrusions.

Features #	Featues Name	Features #	Featues Name	Features #	Featues Name
1	Destination Port	30	Bwd IAT Min	59	Fwd Avg Bulk Rate
2	Flow Duration	31	Fwd PSH Flags	60	Bwd Avg Bytes/Bulk
3	Total Fwd Packets	32	Bwd PSH Flags	61	Bwd Avg Packets/Bulk
4	Total Backward Packets	33	Fwd URG Flags	62	Bwd Avg Bulk Rate
5	Total Length of Fwd Packets	34	Bwd URG Flags	63	Subflow Fwd Packets
6	Total Length of Bwd Packets	35	Fwd Header Length	64	Subflow Fwd Bytes
7	Fwd Packet Length Max	36	Bwd Header Length	65	Subflow Bwd Packets
8	Fwd Packet Length Min	37	Fwd Packets/s	66	Subflow Bwd Bytes
9	Fwd Packet Length Mean	38	Bwd Packets/s	67	Init_win_bytes_forward
10	Fwd Packet Length Std	39	Min Packet Length	68	Init_win_bytes_backward
11	Bwd Packet Length Max	40	Max Packet Length	69	act_data_pkt_fwd
12	Bwd Packet Length Min	41	Packet Length Mean	70	min_seg_size_forward
13	Bwd Packet Length Mean	42	Packet Length Std	71	Active Mean
14	Bwd Packet Length Std	43	Packet Length Variance	72	Active Std
15	Flow Bytes/s	44	FIN Flag Count	73	Active Max
16	Flow Packets/s	45	SYN Flag Count	74	Active Min
17	Flow IAT Mean	46	RST Flag Count	75	Idle Mean
18	Flow IAT Std	47	PSH Flag Count	76	Idle Std
19	Flow IAT Max	48	ACK Flag Count	77	Idle Max
20	Flow IAT Min	49	URG Flag Count	78	Idle Min

21	Fwd IAT Total	50	CWE Flag Count	79	Label
22	Fwd IAT Mean	51	ECE Flag Count		
23	Fwd IAT Std	52	Down/Up Ratio		
24	Fwd IAT Max	53	Average Packet Size		
25	Fwd IAT Min	54	Avg Fwd Segment Size		
26	Bwd IAT Total	55	Avg Bwd Segment Size		
27	Bwd IAT Mean	56	Fwd Header Length.1		
28	Bwd IAT Std	57	Fwd Avg Bytes/Bulk		
29	Bwd IAT Max	58	Fwd Avg Packets/Bulk		

الملخص

مع تزايد تعقيد الهجمات الإلكترونية وطبيعتها المتطورة جنبًا إلى جنب مع التقدم في البنية التحتية للشبكة، تواجه أنظمة اكتشاف التسلل (IDS) تحديات متزايدة. يُعد التعلم الآلي نهجًا واعدًا لتحليل مجموعات البيانات المتنوعة الناتجة عن حركة مرور الشبكة بكفاءة. يبحث هذا البحث في تأثير اختيار الميزات على تحسين دقة وكفاءة أنظمة اكتشاف التسلل. من خلال تطبيق نموذج اختيار الميزات الخاص بنا على مجموعة بياناتCICIDS2017 ، حددنا أهم 10 ميزات ذات صلة تؤثر بشكل كبير على تحسين أداء نماذج التعلم الآلي المتعددة. بالإضافة إلى تقييم منهجية اختيار الميزات الخاصة بنا، قمنا أيضًا بتطبيق نماذج اختيار الميزات المستخدمة في الدراسات السابقة على نموذجنا. توضح النتائج أن نهجنا في اختيار الميزات لا يزال يتفوق على هذه النماذج السابقة من حيث الدقة والكفاءة الحسابية. على وجه الخصوص، حققت خوارزمية Random Forest دقة بنسبة 96.1٪، بينما حققت Naive Bayes دقة بلغت 82.5٪، وتجاوزت كل من خوارزميات AdaBoostو K-Nearest Neighbors (KNN)دقة 98٪. على الرغم من أن KNN أظهرت دقة ممتازة، إلا أنها تطلبت وقتًا حسابيًا أطول بكثير مقارنة بالنماذج الأخرى. يؤكد هذا البحث على دور اختيار الميزات في تحسين أداءIDS ، ويظهر كيف يعزز نهجنا في اختيار أفضل 10 ميزات دقة الكشف مع الحفاظ على أوقات معالجة فعّالة. تؤكد نتائجنا أن منهجية اختيار الميزات المستخدمة في هذه الدراسة توفر ميزة واضحة على النماذج السابقة، مما يحسن كل من دقة الكشف وقابلية تطبيق IDS في الوقت الفعلي.