REPUBLIC OF TÜRKİYE

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Information Technologies

DETECTING DENIAL OF SERVICE ATTACK OF NETWORK TRAFFIC BY BUILD ACCURATE INTRUSION DETECTION SYSTEM BASED ON MACHINE LEARNING ALGORITHM

Sabreen ALMOHAMEDAWI

Master's Thesis

Supervisor

Asst. Prof. Dr. Abdullahi Abdu Ibrahim

Istanbul, 2022

DETECTING DENIAL OF SERVICE ATTACK OF NETWORK TRAFFIC BY BUILD ACCURATE INTRUSION DETECTION SYSTEM BASED ON MACHINE LEARNING ALGORITHM

Sabreen ALMOHAMEDAWI

Information Technologies

Master's Thesis

ALTINBAŞ UNIVERSITY

2022

The thesis titled DETECTING DENIAL OF SERVICE ATTACK OF NETWORK TRAFFIC BY BUILD ACCURATE INTRUSION DETECTION SYSTEM BASED ON MACHINE LEARNING ALGORITHM prepared by SABREEN ALMOHAMEDAWI and submitted on 7/12/2022 has been **accepted unanimously** for the degree of Master of Science in Information Technology.

Asst. Prof. Dr. Abdullahi Abdu Ibrahim

Thesis Defence Committee Members:

Asst. Prof. Dr. Abdullahi Abdu Ibrahim	Department of Computer Engineering, Altınbaş University	_____
Asst. Prof. Dr. AYÇA KURNAZ	Department of Software Engineering , Altınbaş University	_____
Asst. Prof. Dr. Serdar KARGIN	Department of Biomedical Engineering, Beykent University	_____

I hereby declare that this thesis meets all format and submission requirements of a Master's thesis.

Submission date of the thesis to Institute of Graduate Studies: / /

I hereby declare that all information/data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Sabreen ALMOHAMEDAWI

Signature

DEDICATION

I dedicate this humble research to my dear parents who have done everything in their power since my childhood to push me forward for educational attainment, also my beloved wife and children who have devoted their efforts to creating the appropriate environment and creating an encouraging atmosphere to continue research and analysis, also to every researcher looking forward to collecting information, which I tried as much as possible to collect as much as possible how much can be related to the topic of research.

ABSTRACT

DETECTING DENIAL OF SERVICE ATTACK OF NETWORK TRAFFIC BY BUILD ACCURATE INTRUSION DETECTION SYSTEM BASED ON MACHINE LEARNING ALGORITHM

Sabreen ALmohamedawi

M.Sc, Information Technologies, Altınbaş University,

Supervisor: Asst. Prof. Dr. Abdullahi Abdu Ibrahim

Date: December/2022

Pages: 63

Today, the creation of more effective intrusion detection systems has become crucial due to the rise in computer malware. Ensure the availability of the system is an important component of information security and the most important requirement of any network. Recently the Machine Learning algorithm (ML) has been used to improve intrusion detection over the network. It is currently necessary to release an updated version of these systems. The presented work aimed to build a reliable and accurate IDS based on ML to classify and prevent distributed denial of service attacks to protect any system working on the network from temporary or complete system failure. The proposed ML models, including (decision tree, random forest, logistic regression, support vector machine, and multi-layer neural network) were trained and evaluated using the cic-ids-2018 dataset. Furthermore, principal component analysis (PCA) was used to reduce the dimensionality of the dataset. According to the classification results, the proposed multi-layer neural network model has optimal performance at an accuracy of 99.9992%.

Keywords: Machine Learning, Intrusion Detection System, Distributed Denial of Service Attacks Classification, Services Availability

TABLE OF CONTENTS

Pages

ABSTRACT ..... **vi**

LIST OF TABLES ..... **x**

LIST OF FIGURES ..... **xi**

ABBREVIATIONS ..... **xiii**

1. INTRODUCTION **1**

1.1 OVERVIEW 1

1.2 AIM OF THIS WORK 2

1.3 PROBLEM STATEMENT 2

1.4 LITERATURE SURVEY 3

1.5 THESIS LAYOUT 4

2. THE THEORETICAL BACKGROUND OF IDS **6**

2.1 INTRODUCTION 6

2.2 WHAT IS THE IDS 6

2.3 TYPE OF IDS IN THE TERMS OF AVAILABILITY 6

2.3.1 Signature-Based-Detection 6

2.3.2 Anomaly based detection 7

2.4 MALWARE ATTACK AND COMMON TYPES 7

2.4.1 The Common Types of Malware 7

2.4.2 Denial of Service Attack (DOS) 9

2.4.3 Distributed Denials of service Attacks (DDoS) 10

2.5 MACHINE LEARNING FOR ATTACK DETECTION AND CLASSIFICATION 11

2.6 CASE STUDY ML TECHNIQUES FOR CLASSIFICATION ATTACKS 14

2.6.1 Decision Tree algorithm (DT) 14

2.6.2 Random Forest (RF) 15
2.6.3 Support Vectors Machine (SVM) 16
2.6.4 Linear Regression (LR) 17
2.6.5 Deep Learning (DL) 18
2.6.6 Neural Network (NN) 18
2.7 FEATURE SELECTION TECHNIQUES 21
2.8 DATASET DESCRIPTION 22
3. THE METHODOLOGY OF THE PROPOSED DDOS-IDS 23
3.1 INTRODUCTION 23
3.2 THE METHODOLOGY OF THE PROPOSED IDS 23
3.3 PREPROCESSING STEPS 24
3.3.1 Understanding The Dataset 24
3.3.2 Handling with Missing Values and Outlier 25
3.3.3 Features Scaler 25
3.3.4 Preparing Dataset and Splitting Procedure 25
3.4 CASE STUDY TRAINING, TESTING THE PROPOSED IDS USING ML ALGORITHMS 26
3.4.1 Training, Testing Using DT 26
3.4.2 Training, Testing Using RF 27
3.4.3 Training, Testing Using LR 28
3.4.4 Training, Testing Using SVM 29
3.4.5 Training, Testing Using ML-ANN 30
3.5 APPLY FEATURE SELECTION METHOD PCA 31
4. THE RESULTS 33
4.1 INTRODUCTION 33
4.2 IMPLEMENTATION ISSUES 33

4.3 PREPROCESSING RESULTS 33
4.3.1 Statistical Summary of the Dataset 33
4.3.2 Features Scaler Results 34
4.4 PROPOSED IDS CLASSIFICATION RESULTS 35
4.4.1 Classification Results of DT-Model 35
4.4.2 Classification Results of RF-Model 38
4.4.3 Classification Results of LR-Model 40
4.4.4 Classification Results of SVM-Model 42
4.4.4 Classification Results of MLP-NN-Model 44
4.5 EXPERIMENTAL RESULTS OF APPLYING PCA TO THE ALL MODELS 46
5. THE DISCUSSION AND CONCLUSIONS 50
5.1 RESULTS DISCUSSION 50
5.2 CONCLUSIONS 50
REFERENCES 52

LIST OF TABLES

Pages

Table 4.1: Statistical summary of data observations 34

Table 4.2: Statistical summary of training set after standard scaler 35

Table 4.3: Testing classification report of DT-model 37

Table 4.4: Testing classification report of RF-model 39

Table 4.5: Testing classification report of LR-model 41

Table 4.6: Testing classification report of SVM-model 43

Table 4.7: Testing classification report of MLP-NN -model 45

Table 4.8: The PCA output variance results. 46

Table 4.9: The comparison results of ML-model before PCA 48

Table 4.10: The comparison results of ML-model after PCA 48

Table 4. 11: The comparison results of the proposed IDS and other related works 49

LIST OF FIGURES

Pages

Figure 1. 1: Traditional network with IDS 2

Figure 2. 1: The architecture of DDoS 10

Figure 2. 2: The process of DDoS 11

Figure 2. 3: Machine learning approach 12

Figure 2. 4: Supervised learning approach 13

Figure 2. 5: Unsupervised approach 13

Figure 2. 6: Semi-supervise learning approach 14

Figure 2. 7: DT algorithm 15

Figure 2. 8: Random Forest algorithm (RF) 16

Figure 2. 9: SVM algorithm 17

Figure 2. 10: LR algorithm 18

Figure 2. 11: Neural Network (NN) 19

Figure 2. 12: ReLU activation function shape 20

Figure 2. 13: Tanh activation function shape 20

Figure 2. 14: Sigmoid activation function 21

Figure 3. 1: The block-diagram procedure of the proposed IDS 24

Figure 4. 1: CM of testing DT model 36

Figure 4. 2 label distribution of DT-model between actual and predicted class 37

Figure 4. 3: CM of testing RF model 38

Figure 4. 4: Label distribution of DT-model between actual and predicted class 39

Figure 4. 5: CM of testing LR model 40

Figure 4. 6: Label distribution of LR-model between actual and predicted class 41

Figure 4. 7: CM of testing SVM model 42

Figure 4. 8: Label distribution of SVM-model between actual and predicted class 43

Figure 4. 9: CM of testing MLP-NN model 44

Figure 4. 10: Label distribution of MLP-NN -model between actual and predicted class 45

ABBREVIATIONS

Adam	:	ADAptive optimization Method
AF	:	Activation Function
ANN	:	Artificial Neural Network
DDOS	:	Distributed Denial Of Service
DL	:	Deep Leaning
DOS	:	Denial Of Service
DT	:	Decision Tree
GA	:	Genetic Algorithm
HIDS	:	Host-Based Intrusion Detection System
IDS	:	Intrusion Detection System
KDD	:	Knowledge Discovery and Data Mining
LR	:	Logistic Regression
LSTM	:	Long Short Term Method
ML	:	Machine Learning
ML-ANN	:	Multi-Layer Artificial Neural Network
MLP	:	Multi-Layer Perceptron
PCA	:	Principal Component Analysis
Relu	:	Rectified Linear Unit
RF	:	Random Forest
SVM	:	Support Vectors Machine
UNSF	:	United Nations Security Force

1. INTRODUCTION

1.1 OVERVIEW

In today's world, almost everyone has access to a computer, and network-based technology is rapidly evolving. As a result, network security has become a critical part, if not necessary, component of any computer system. A security attack or intrusion can be defined as any threat or unintentional attempt to destroy the availability, integrity, or confidentiality of any information resource or the information itself. Such these threats could be limited by using an Intrusion Detection System (IDS) [1].

The availability is a major part of the information security, and it’s the most fundamental requirement for any network. The network would cease to exist if its connection ports were unreachable or if its data routing and forwarding mechanisms were malfunctioning. Therefore, Availability means that despite denial-of-service attacks, the network must always be accessible [2]. The proliferation of malware presents a serious problem for IDS architects to solve it. Since the creators of unknown and obfuscated malware typically employ multiple evasion techniques for information concealment in order to thwart detection by an IDS, spotting such threats can be a difficult task. The sophistication of malicious attacks has increased. Additionally, there has been an increase in security threats like zero-day attacks that target people who use the internet. Therefore, computer security has become paramount as the use of information technology has permeated every aspect of our lives. The result is that the zero-day attacks have had a significant impact on many nations, including Australia and the US [3].

An IDS's goal is to quickly identify various malware types because a traditional firewall is unable to do so. The creation of more effective IDSs has become crucial due to the rise in computer malware. An updated version of these systems is currently required because machine learning has been used to enhance intrusion detection over the past few decades, Figure 1.1 shows a traditional network with IDS [4].

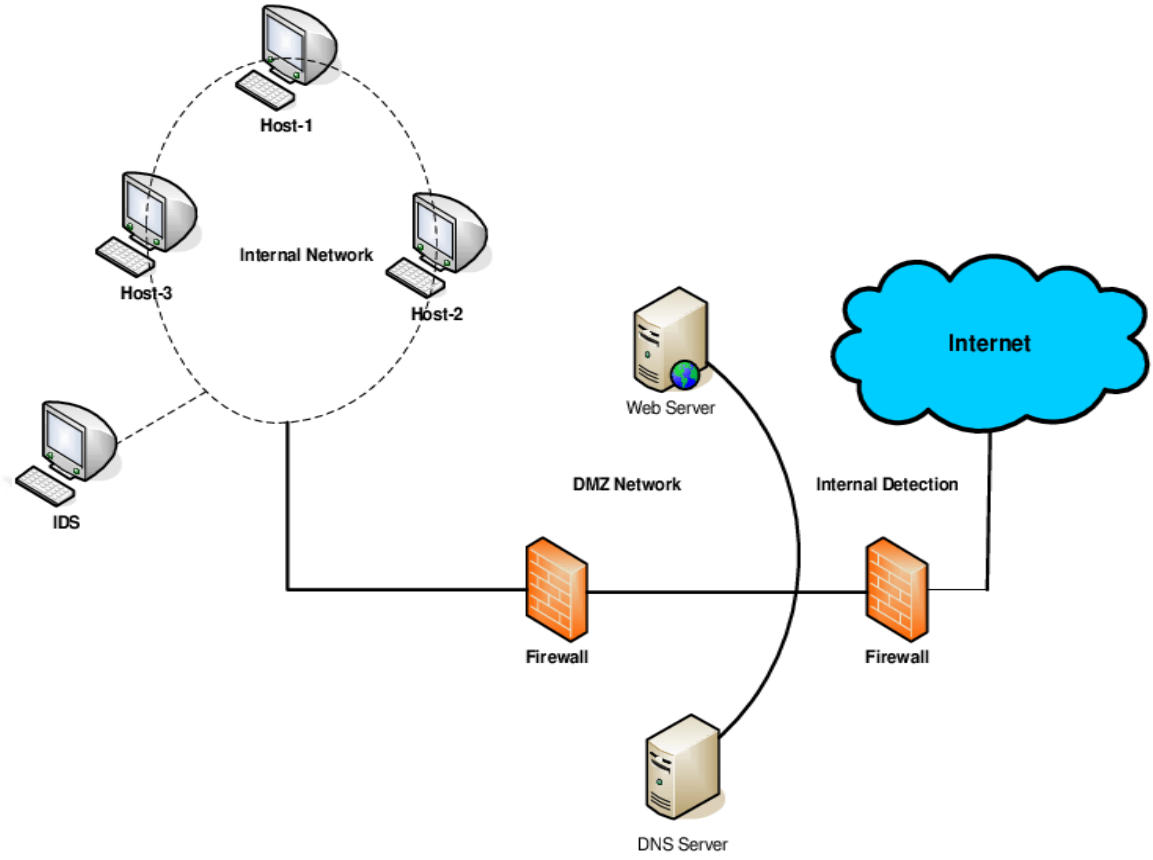

Figure 1. 1: Traditional network with IDS

1.2 AIM OF THIS WORK

This work aim to build a reliable intrusion detection system with higher detection rate based on machine learning algorithm. The proposed IDS use to insure the availability of the network and services by detect malwares from inter to the system such as Dos attacks and its similar types. Such kind of these security systems have the ability to learn from the experience by preventing these attacks before threat happened. IDS also useful and effective to increase the power of the security of the system behind the firewall which is lead to insure the availability of the services at run time.

1.3 PROBLEM STATEMENT

The problem of this work could be summarized as following points.

a. Handle with dataset problems by understand data statistic and preprocessing steps.
b. Selecting the best feature using feature selection methods.

c. Design reliable model based on classification of proposed dataset by training it on various attacks and normal one.
d. Improving the detection rate with best classification results.

1.4 LITERATURE SURVEY

A several related works was listed in this section; each one has different methodology about Anomaly-based intrusion detection and attacks classification. In the last year IDS have a reputation for having a high accuracy and high detection rate. Furthermore, deploying and training them incurs a sizable computational cost.

a. In [5] Kotpalliwar, et.al. (2015) they utilize SVM for to classify attacks in KDD99 dataset and propose ids system work on single computer. Also, they achieved validation accuracy 89.8%.
b. (Basant Subba , et.al. (2016)) [6] the authors utilized benchmark NSL-KDD 99 dataset with proposed model trains on deferent ML- algorithm were included Support vectors machine (SVM), Artificial Neural Network (A-NN), and others methods. The overall average accuracy achieved at 98.6%.
c. In [7] An, et al.(2018) authors introduce unsupervised assassination analyses of IDS on Distributed Denial-Of-Services attack (DDoS). Through this study they verify a higher scoring utilization rate of promoted these attacks.
d. In [8] Tama, et.al.(2019) they proposed IDs model due to hybrid feature selection and ensembles of tow-levels classifiers, for reducing the dimensionality of the training set on (NSL-KDD and UNSW-NB15). Also, utilized three optimization methods which is includes Particle Swarm, Ant Colony, and genetic algorithm (GA), and the result of classification is 85.8% accuracy on the NSL-KDD dataset.
e. He, et al. (2019) [9] For conducting anomaly detection, the authors suggested combining method of utilizing LSTM and multi-models deep auto encoder. On three different datasets from 1999 to 2017, this innovative approach was tested and achieved accuracy scores for multi labels at 80%, 86%, and 98.6%.
f. In this review [10] (2020) These are last several datasets that include new attack categories and network attack attributes. This article describes recent improvements in IDS datasets which can be used by a wide range of research societies as a mission statement for creating efficient and appropriate ML and data mining-based IDS.

g. Khammassi, et al. in [11] (2020) the author proposed wrapper feature selection method on UNSW-N15 and KDD99 datasets were implemented with GA and LR. The results show accuracy at 80% on unsw-15 dataset with 42 features and 99.9% on KDD99 dataset with 19 selected features.
h. In [12] Saranya, et al. (2020) a number of machine learning models' performances were evaluated against the KDD-99 dataset, and Random forest (RF) outperformed other methods like SVMs, Naive Bayes, and Logistic Regression with an accuracy score of 99.81 %.
i. In [13] Kasongo, et al. (2020) the researcher utilizes the UNSW-NB15 dataset and analyzed through several ML methods, this study refers to apply features selection techniques to proposed models were trained and tested to improving the accuracy score of binary and multi-class classification. The XGboost- based feature selection method improves the accuracy from 88% to 90.8% in DT model.
j. (Nuno Oliveira, et.al (2021)) [14] researcher utilized the MLP and LSTM on CIDS-01 dataset to build an accurate malicious classification model on sequential viewpoint. This study shows the LSTM was highly accuracy in sequential information pattern and achieved 99.96% classification accuracy.

1.5 THESIS LAYOUT

The presents study includes five chapters. the description summary related to the other four chapters will be provided as follows:

Chapter Two: Theoretical Background of The IDS

This chapter refers to the basic background of common methods for IDS and common attacks. Also, ML and its methods in anomaly-detection based and classification tasks. Finally, feature selection technique and how it important to reduce the dimensionality of data.

Chapter Three: The Proposed IDS Methodology

The stages and steps of the proposed IDS classification system are covered in this chapter. The main subjects of this chapter can be specified in three-part, each part related to a specific task, which is include pre-processing on the proposed dataset, then selecting the best model by training several models using different ML algorithms and ML-ANN, and finally apply

feature selection technique using (PCA) to reduce the dimensionality and improve the detection rate.

Chapter Four: Results

The testing results of implementing the proposed IDS system are shows in this chapter. It includes tables and figures to illustrates the evaluation result.

Chapter Five: Discussions and Conclusions

This chapter concern with a list of conclusions and some suggestions for future work that are related to the proposed techniques. Also includes discussion about income results.

2. THE THEORETICAL BACKGROUND OF IDS

2.1 INTRODUCTION

The theoretical background and concepts about thesis topic are covered in this chapter, it also includes details about methods of ML algorithms that related to this topic. IDS definition, how it works, its types and last interested anomaly detection research about network security also describe in details. Furthermore, this chapter includes information about attacks and its common type in the term of availability of the system. Finally, contains information of the proposed dataset in order to start the procedure of designing the proposed IDS in next chapter (3).

2.2 WHAT IS THE IDS

The term "intrusion" refers to a group of connected malicious acts carried out by an internal or external invader in an effort to breach the targeted system. Monitoring computer systems, network traffic, and analysing activity are all part of intrusion detection, which entails looking for potential system invasions. IDS is a set of tools and methods used for this goal [15]. In general, the majority of IDS offer standard functionality to protect network security. Data from observed actions are first gathered by an IDS. It provides thorough event-related data logging and correlates events from many sources. The detection engine, which uses various approaches and associated techniques depending on the circumstance, is the heart of an IDS. Additionally, preventative skills can be offered. The system in question is referred to as an intrusion detection and prevention system (IDPS) [16].

2.3 TYPE OF IDS IN THE TERMS OF AVAILABILITY

The most widely utilized approaches for intrusion detection are anomaly and signature based detection. In order to improves the performance of IDS model, they are frequently employed in combination, either integrated or independently.

2.3.1 Signature-Based-Detection

Due to the utilization of information integrated from previous intrusions and vulnerabilities, “signature based detection” is also refers to misuse or knowledge based detection. Because such their patterns are unknown, this method is insufficient to identify unknown intrusions

and known intrusion variants. Another issue is keeping the knowledge updated because it is a laborious and time-consuming process [17].

2.3.2 Anomaly Based Detection

Anomaly can be defining as Any departure from typical behaviour is considered an anomaly. The process of comparing typical behaviour to observed events in order to locate significant deviations is known as anomaly-based detection, also known as behaviour-based detection [15].

Anomaly detection techniques can be divided into three categories based on the target system's "behavioural" model and type of process:

a. Statistical-based.
b. knowledge-based.
c. Machine Learning based.

2.4 MALWARE ATTACK AND COMMON TYPES

As information technology has grown and adapted in recent years, malware has become a threat to large networks. Malicious software may be faced every day as long as the devices are connected to the internet. Additionally, despite the fact that end-user awareness of malware is growing, it is still not enough.

Malware is unwanted, harmful software that has been created with the intention of harming the end users or attack the system. Many different types of malware, for examples trojans, backdoors, viruses, crypto lockers, spyware, and ransomware, may fall under this category. Malware can be categorized based on its features and objectives. According to the types of behaviour, it can be petitioned into four categories: propagations, infections, persistence's, and payloads. The most popular ways that malicious software attacks are through propagation behaviour, which describes the malware spreads when there is communication over the Internet[18].

2.4.1 The Common Types of Malware

Although there are many different kinds of malware [18], the following are the most prevalent today:

a. Spyware**:** is intended to gather information from the user covertly, without the user's knowledge or observation. When a computer is compromised, the user's every action is recorded, including logins, file operations, and sensitive information like saved usernames, passwords, and credit card numbers.
b. Adware**:** These malicious programs display advertisements while a user is browsing the web. Although it is not malicious software on its own, it can be transformed into a more dangerous form when used in conjunction with certain spyware to steal user data and track the activity of the system.
c. Virus**:** Malicious software like this, which can hide in any kind of code or document, can be very problematic, as it can either delete important system documents or completely freeze the computer. Email, USB drives, and physical media like CDs and DVDs are common vectors for its spread.
d. Bot**:** The infected system can be controlled by the attacker or a certain task can be performed without the user's knowledge or consent thanks to this malware. In large-scale attacks, bot malware is typically used to take advantage of the system's computing power**.**
e. Bug: Software bugs are incorrect errors that occur. And even with not being considered malicious software in and of themselves, they are employed to grant the attacker currently a wide**.**
f. Rootkit**:** Its purpose is to allow access to the system from afar without the user's knowledge. A rootkit can successfully run and perform a wide range of actions on the system, including file uploads, program installations, system file modifications, and the deactivation of security applications.
g. Trojan**:** This type of malicious software typically gains access to a system through email or the web without the user's knowledge. After entering the system, the malware conceals itself (for example, by imitating an image file). Trojans, like rootkits, have a wide range of capabilities once they have gained access to a system.
h. Worm**:** By utilizing vulnerabilities in the system, worms spread across networks. Its goals could be to use up network resources or overload web servers to cause a denial of service.

i. Crypto malware**:** Crypto malware, as the name suggests, is malicious software that encrypts data in order to render it unusable. There are two main types of malware in this category: lockers and encoders. The most prevalent form is malicious software, which, once installed on a computer, locks down all data in the system. The user needs a unique password key to regain access to the files, and the attacker demands payment before sending the password key.

2.4.2 Denial of Service Attack (DOS)

The internet is widely used today. Through a variety of networks, systems are linked to one another. A network is any collection of n computers, routers, servers, etc. A network may contain both valid and unauthorized users, or hackers. A hacker is someone who illegally accesses and uses another person's data. A hacker may employ a number of techniques for this. These techniques are divided into two categories: active attacks and passive attacks. While in active attacks hackers manipulate data and stop users from acting as they wish, passive attacks involve the hacker simply observing the system and gathering as much information as possible without actually affecting it. In the world of cybercrime today, DOS attacks are among the most well-known and dangerous. Active attacks known as denial-of-service attacks (DOS) cause servers and networks to crash due to an overwhelming number of requests or packets. Given the current size of the network, there may be a sizable population of users waiting to join. [19]**.**

A DOS attack is any kind of attack on a network that prevents a server from serving its clients. In order to slow down a server, an attacker may try to flood it with invalid data packets, use a spoofed IP address, or send an overwhelming number of requests. [20].

Network-based flooding attacks are difficult to distinguish from rapid spikes in normal activity or flash events using DOS detection techniques such as activity profiling, change point detection, and based methods signal analysis. Despite encouraging outcomes from preliminary testing, no detector has yet completely resolved the detection problem. Most likely, the best outcomes will be achieved through a combination of methods led by experienced network administrators. [21]. Ping of Death, TCP SYN Flood, and Distributed-DOS are the three main varieties of DOS. A Ping of Death attack is when one host floods another with so many ping requests (ICMP Echo Requests) that the target host either goes offline or is too busy processing ICMP Echo replies to attend to its clients. Sending a TCP-

connection with the wrong return address during the TCP three-way handshake is what the TCP SYN Flood attack relies on. [20].

2.4.3 Distributed Denials of Service Attacks (DDoS)

DDoS attacks, which overwhelm a system with requests, have become more sophisticated in recent years. This is especially true for distributed denial of service (DDoS) attacks, where a single assault can compromise hundreds or thousands of targets. Figure (2.1) depicts the architecture of a zombie attack, which is designed to disrupt legitimate users' access to services by overwhelming the network with requests, while Figure (2.2) depicts the framework of a distributed denial-of-service attack.

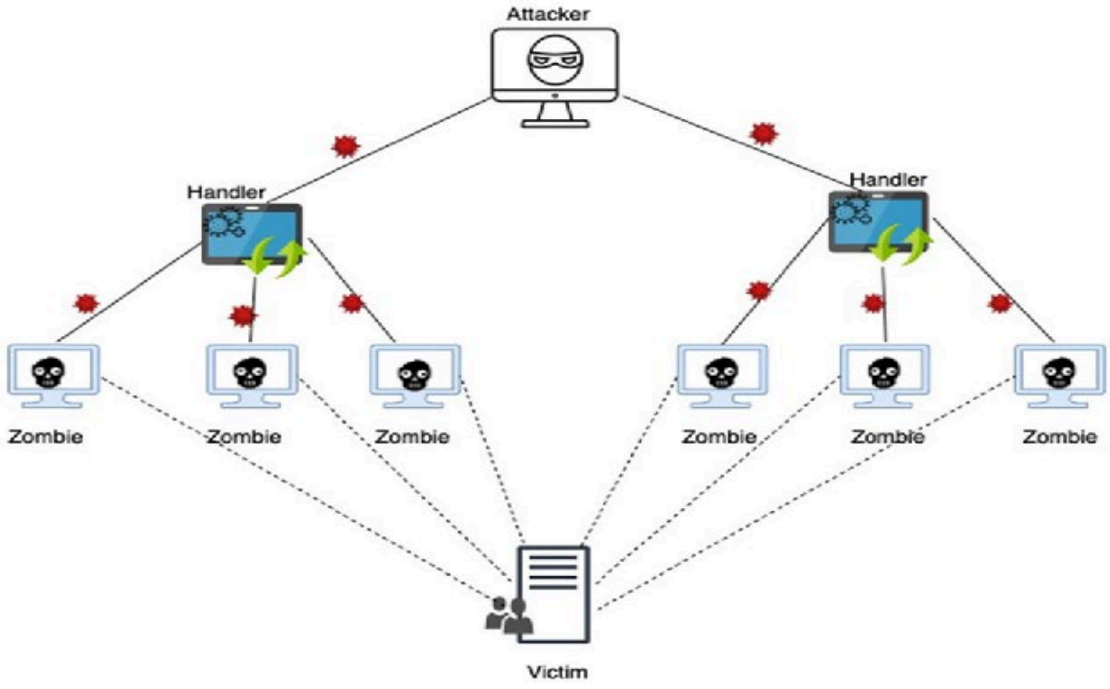

Figure 2. 1: The architecture of DDoS

The master DDoS is a malicious software program that attackers install in an attempt to seize control of a collection of compromised machines within the same network. Distributed denial of service (DDoS) attacks pose an early danger to service providers. A DDoS attack is designed to disrupt and deny services to authorized users by bombarding the target with a large number of malicious requests. [22]. A DDoS attack includes flooding a network with requests in an effort to destroy its capacity or computing resources. It endangers cloud services and makes it difficult to respond to users. Blackmail, attack capability

demonstration, destruction of property, political conflicts, hacktivism, business rivalry, dissuasion from data leakage and other data theft activities, destruction of property, and blackmail are some of the primary reasons behind DDoS attacks[23].

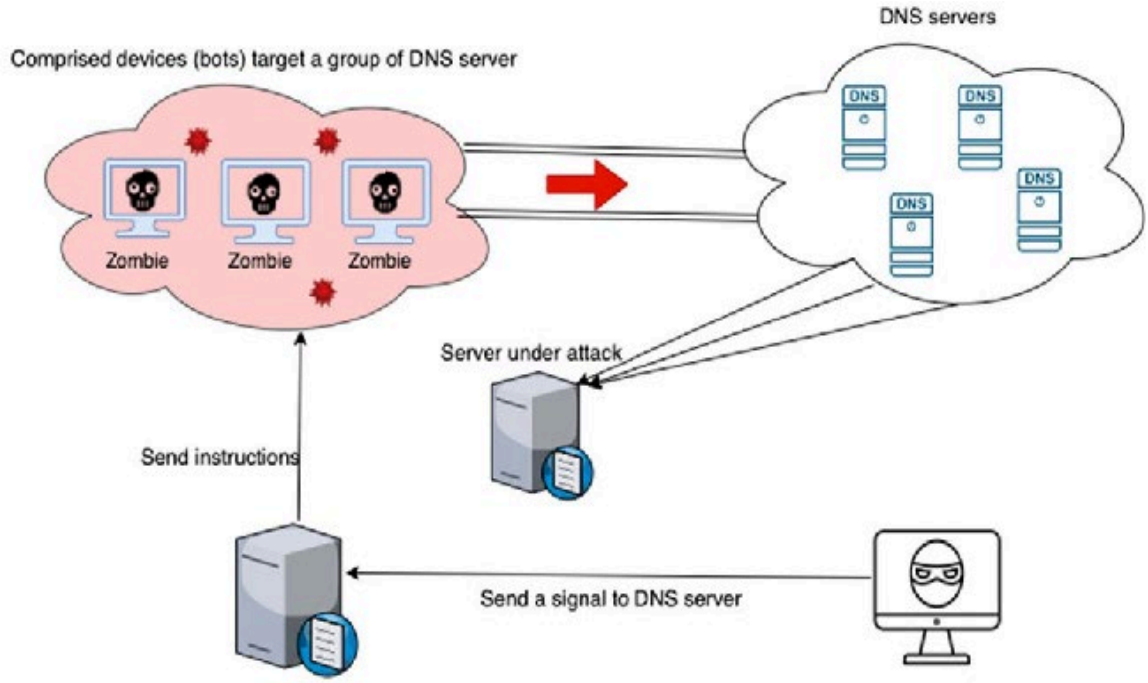

Figure 2. 2: The process of DDoS

In recent years, ML techniques have been used to prevent DDoS attacks. In fact, by employing ML techniques, numerous defense systems have been transformed into intelligent, DDoS-resistant systems. It is believed that DDoS attacks can be detected using ML techniques. These methods acquire attack patterns in order to identify attacks before network resources are exhausted. Modern defense systems use ML techniques in addition to other detection models such as intrusion detection systems (IDS) and host-based intrusion detection systems (HIDS) to effectively counter complex cyberattacks like DDoS attacks [22].

2.5 MACHINE LEARNING FOR ATTACK DETECTION AND CLASSIFICATION

ML studies algorithms that improve with use and computerize exercises. The machine is well-maintained and runs smoothly. Artificial intelligence allows computers to learn without being programmed. [24].

Machine learning is a branch of computer science based on pattern recognition and AI learning theory. It examines how to build and research data-driven algorithms. These algorithms create a model from sample inputs to make data-driven predictions or decisions. A multinomial classifier problem known as the intrusion detection model can categorize network events as either normal or attack events, including DOS, Probe, U2R, and R2L [25], see figure (2.3).

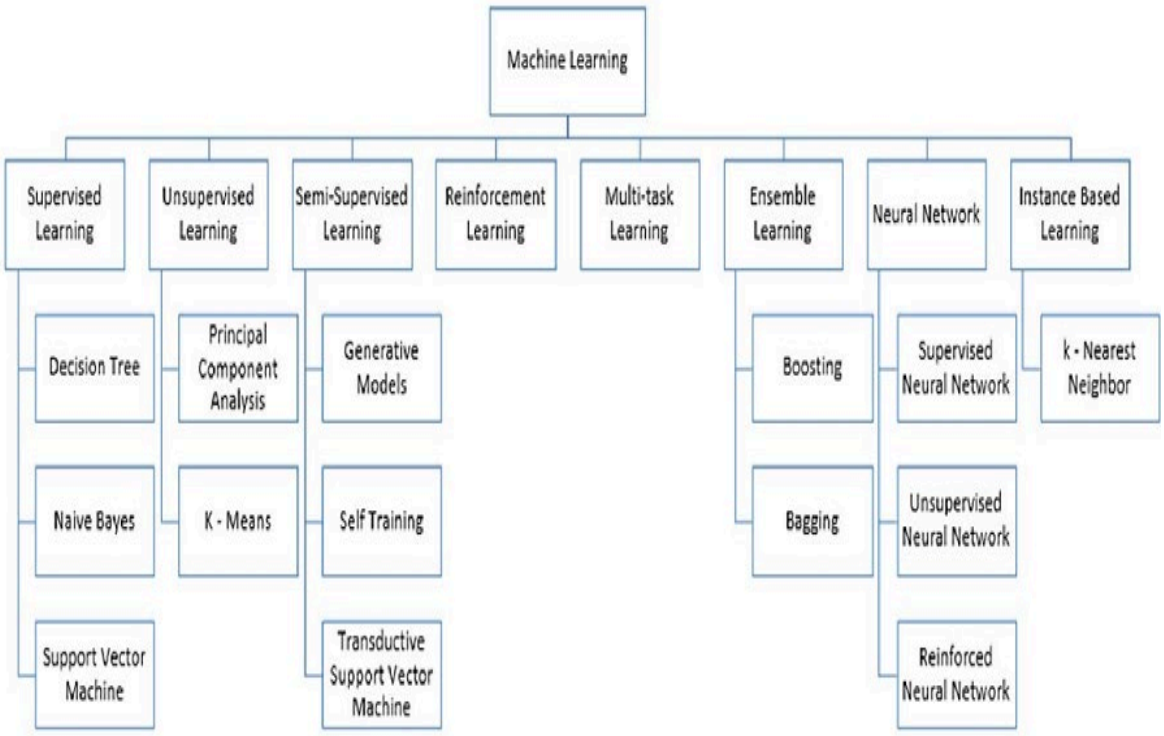

Figure 2. 3: Machine learning approach [26]

In general, machine learning methods are categorized as supervised, unsupervised and semisupervise learning depending on the presence or absence of labelled data and the actual prediction that is being made from the dataset.

a. Supervised Learning: With the help of labeled training samples, supervised learning allows the program to predict similar unlabeled samples. Prediction, Knowledge extraction, and Compression tasks are included, figure (2.4) shows the supervised learning approach.

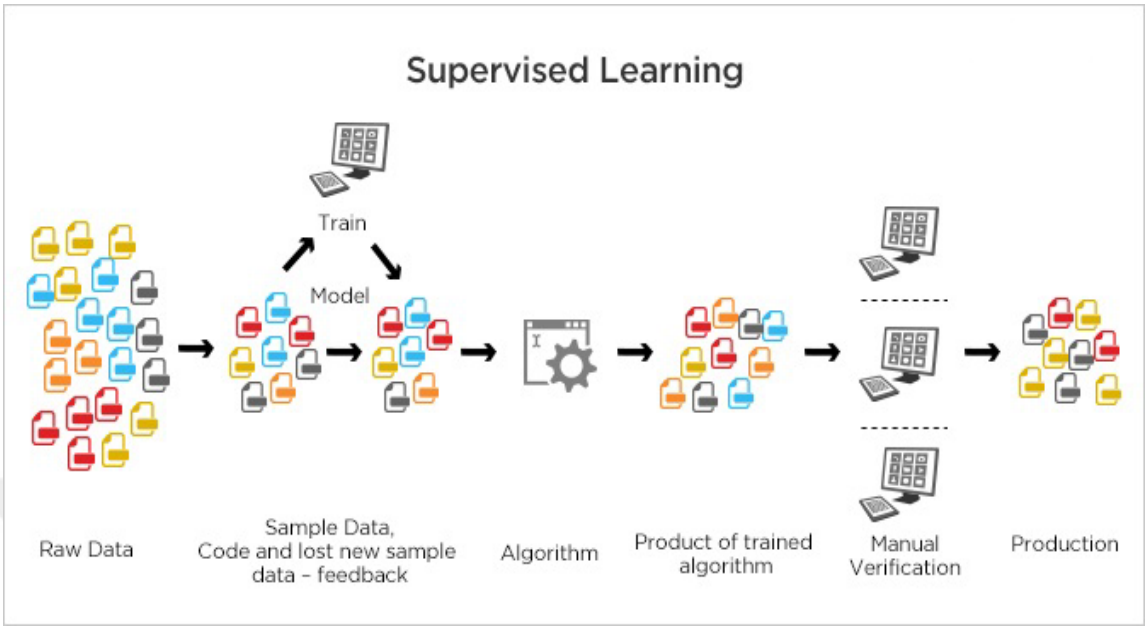

Figure 2. 4: Supervised learning approach [27]

b. Unsupervised learning: Unsupervised learning uses density estimation without training samples. Unsupervised learning assumes that grouping or clustering similar data can reveal the data's hidden structure. Works like pattern recognition and outlier detection are among them [24], see figure (2.5)

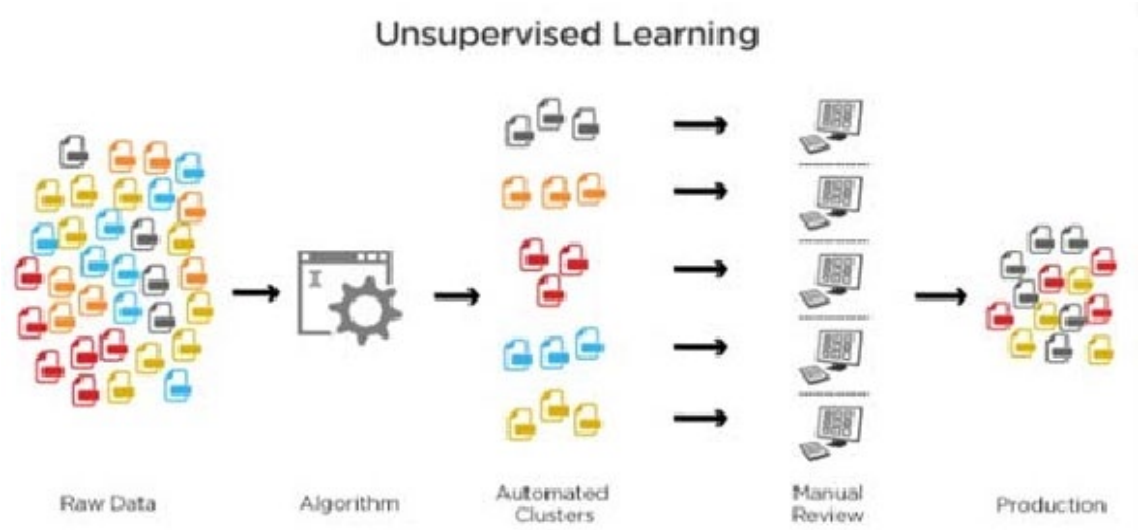

Figure 2. 5: Unsupervised approach **[27]**

c. Semi-supervised machine learning combines unsupervised and supervised methods. In machine learning and data mining, where labeling data is difficult, it may be useful. Figure (2.6) shows this approach.

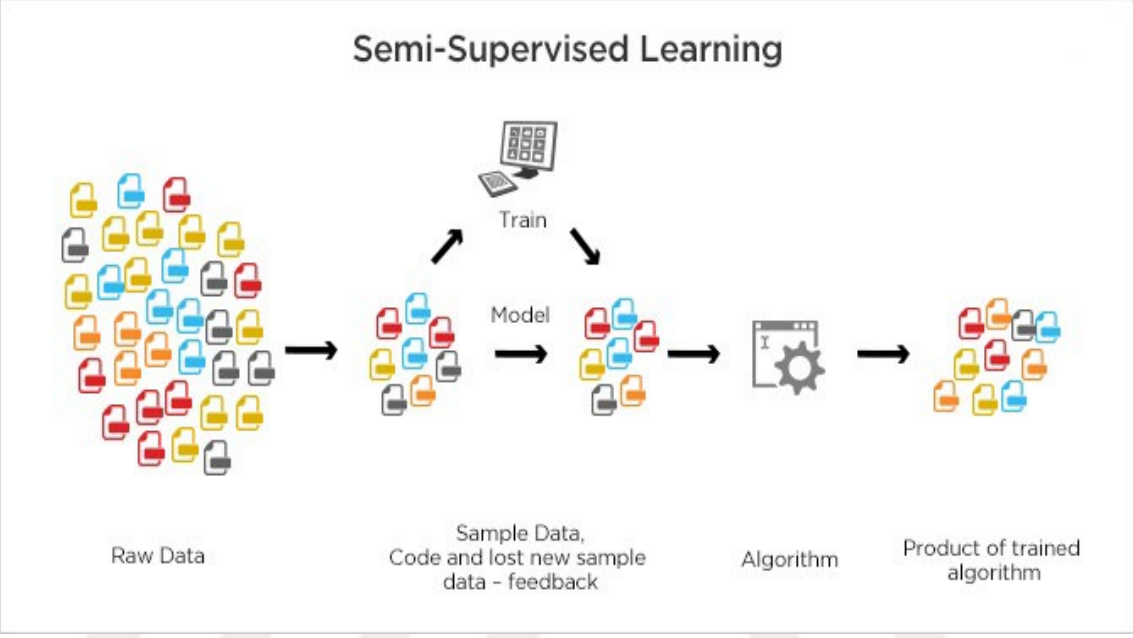

Figure 2. 6: Semi-supervise learning approach [27]

2.6 CASE STUDY ML TECHNIQUES FOR CLASSIFICATION ATTACKS

There are several techniques recently utilized for classifying deferent types of attacks. This section related to presenting the basic information about the classification algorithm that used to build the proposed model.

2.6.1 Decision Tree Algorithm (DT)

It is an algorithm that can be utilized in a variety of settings and domains. DT is a useful algorithm for finding solutions to problems involving classification and regression. The very name of this tool gives the impression that it employs a flowchart that is shaped like a tree in order to present the results of a series of feature-based splits as the basis for its predictions. The final leaves, which are located directly above the root node, are the ones that make the decision, see figure (2.7)

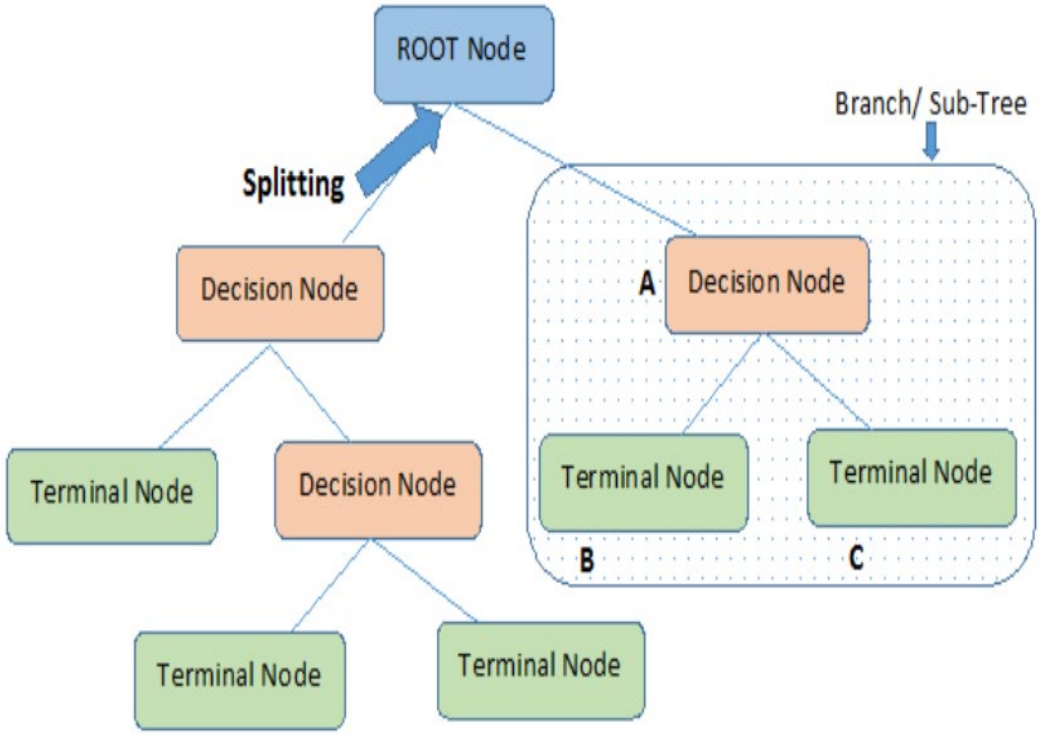

Figure 2. 7: DT algorithm

This method good for small dataset but, in large dataset the uses of DT caused overfitting problems. However, DT pruning with overfitting issues and it could be limited by using tree pruning technique it will be effected with small dataset. Another algorithm works with same strategy of DT but in utilize multiple trees and split the task call Random forest (RF) [28].

2.6.2 Random Forest (RF)

RF is a supervised learning algorithms; it's a collection of multiple DTs that work by randomly splitting the training dataset using the divide-and-conquer method. Also this algorithm can be used for classification and regression, each tree related to an independent random sample. For classification problem, the final results of decision can be received by take mean of all trees and votes the most particular class, figure (2.8) shows the procedure of RF.

How dose RF work:

a. One, choose the samples at random from the collected data.
b. Two, for each sample, build a decision tree and examine its results.

c. Provide a vote for each expected result.

d. As the final prediction, choose the result with the most votes.

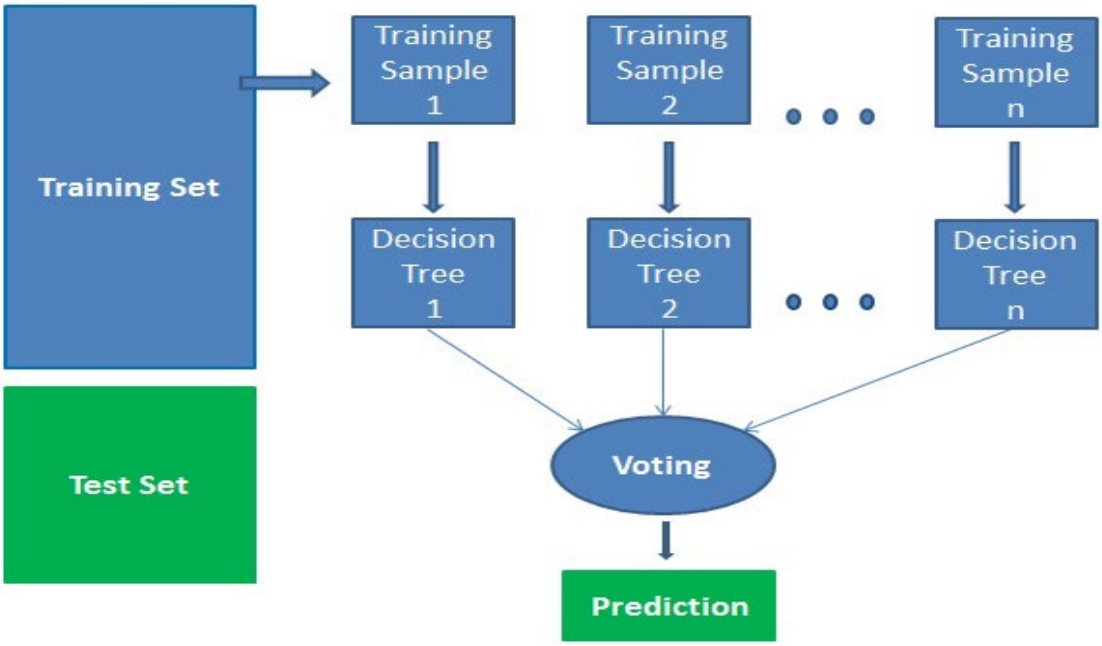

Figure 2. 8: Random Forest algorithm (RF)

the efficiency of this algorithm is a highly accurate and reliable method more than single DT because of the large number of decision trees involved and that can help in overfitting issue [29].

2.6.3 Support Vectors Machine (SVM)

is one of the most powerful ML algorithm and it used commonly to solve pattern recognition problems and variety of classification and regressions tasks. Its work by splitting the data samples using hyperplane, the best hyperplane can fit the training set by computing the maximum distance between two support vectors which is called “maximum margin” [30], figure (2.9) shows SVM algorithm.

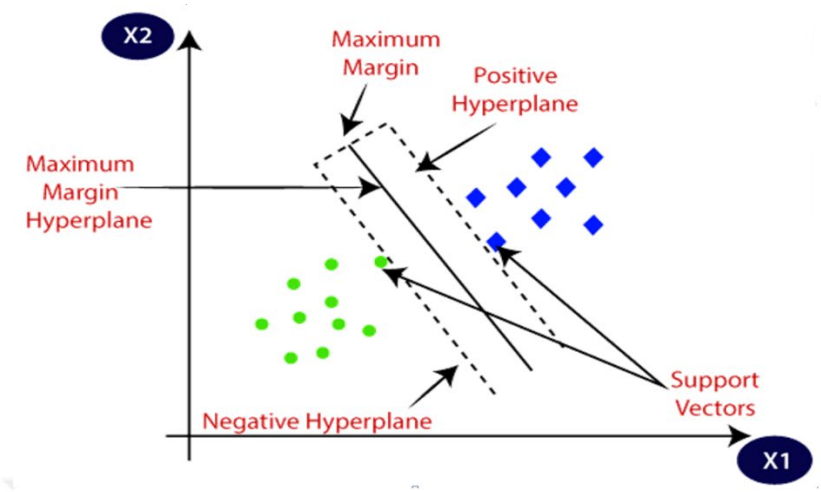

Figure 2. 9: SVM algorithm [31]

2.6.4 Linear Regression (LR)

Linear regression is among the simplest regression analysis methods**.** Although it is the simplest and direct regression model, it is also the most popular and widely used in real-world applications. We only have one independent variable and one dependent variable in this case. It is assumed that the dependent variable, which has a real value, has a normal distribution, the equation of linear regression is:

$$y = a + b * x \tag{2.1}$$

y is the dependent variable, and x is the independent variable. The regression line's intercept is presented by (a , and the slope by the letter b) [28], figure (2.10) shows LR algorithm

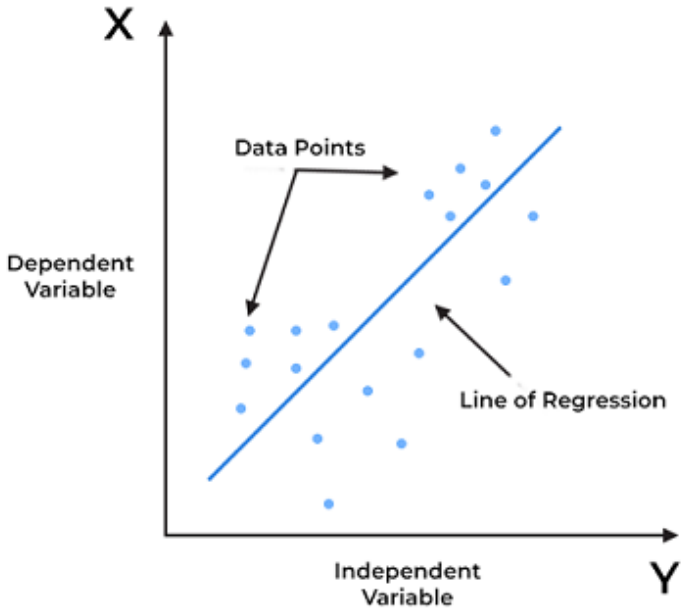

Figure 2. 10: LR algorithm [32]

2.6.5 Deep Learning (DL)

DL is a recent machine learning area that has recently gained popularity. The training process for DL algorithms allows for the automatic learning of features, which is much more efficient than manually extracting these features. Instead of creating a set of guidelines and algorithms to extract features from the unprocessed data [33]. Deep network can represent functions of increasing the process complexity, and solve the large data by adding more layers and more nodes within each layer. DL can be solves big data and large scale tasks with accurate and speed results [34].

2.6.6 Neural Network (NN)

Basic components of neural networks resemble neurons in some ways. These units are connected to one another through connections, the strength of which can be changed as a result of an algorithm or learning process. To assess its level of activation, each of these units independently integrates (in parallel) the data provided by its synapses. The unit response

then depends linearly or nonlinearly on the activation of the unit [35], figure (2.11) shows NN.

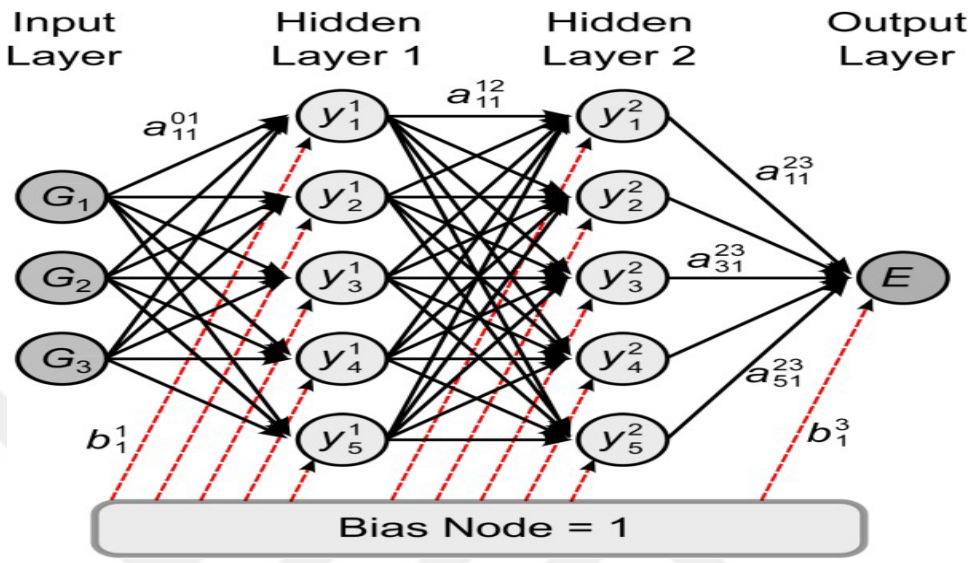

Figure 2. 11: Neural Network (NN) [36]

Typically, the NN consist of three main layers (input-hidden-output) [37]:

a. Input Layer: this is an important section where the input layer needs accurate data from the training phase for the voltage stability assessment. It is a set of input training features in the form of one-dimension or one-vector, for example (X1 - Xi). For each input X are multiplying by trainable parameter called (weight) to produce a dot product (X1*W1) or weighted connection which is the input of the next layer (hidden layer).
b. Hidden layer: it is a set of activation functions (AF) and depend on the tasks its either linear or non-linear. The AFs are determining the particular neurons to be fire or not i.e. (active or not active). Furthermore, there are several types of AF utilizing based on the given dataset and task which are includes (rectify linear unit Rleu and its types, hyperbolic Tanh and Sigmoid etc.), which is describe us below:
    a. Rleu: It's the most common AF used in ANN, this function could be defined as:

$$\boldsymbol{f}(Z) = \boldsymbol{max}(0 , Z) \tag{2.2}$$

where Z is the input of AF. ReLU is work by set the negative value equal to zero and keeps only the positive value, see Figure (2.12).

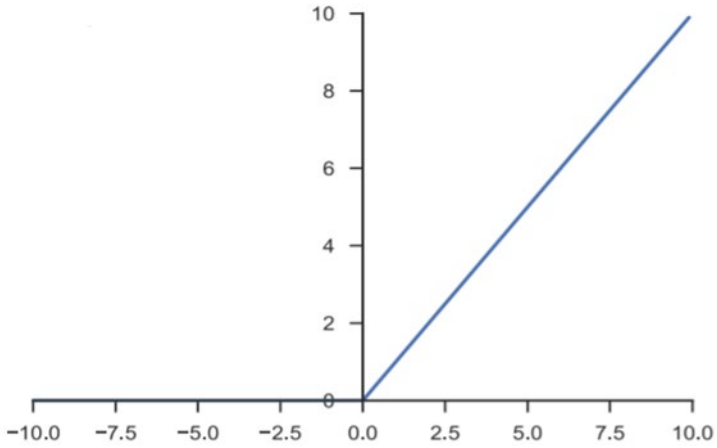

Figure 2. 12: ReLU activation function shape [38]

b. Tanh: activation utilizes a similar type of S-shaped nonlinearity, but rather than scaling from 0 to 1, the tanh output ranges from −1 to 1. As expected, tanh(z) is used. The following relationship between the output y and the input z, as shown in Figure (2.13)

this function defined as:

$$\tanh(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}} \qquad \dots \qquad (2.3)$$

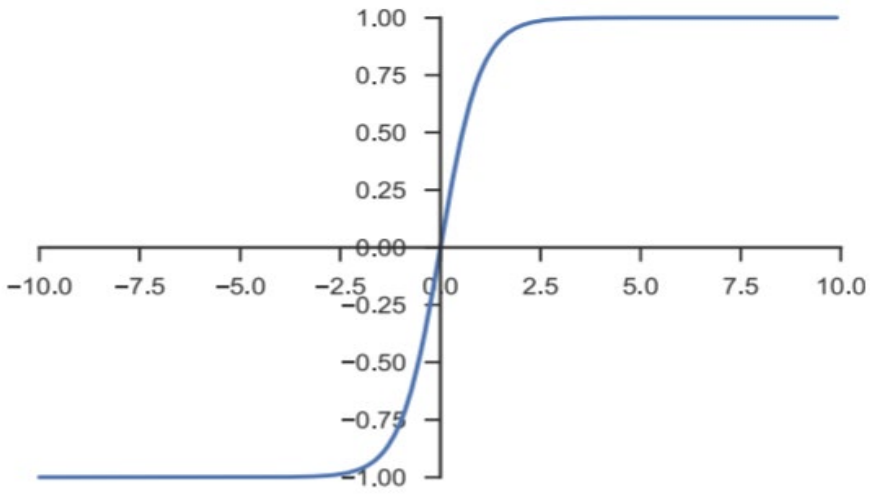

Figure 2. 13: Tanh activation function shape [39]

c. Sigmoid: activation function, which it defined as:

$$f(z) = \frac{1}{1+e^{-z}} \tag{2.4}$$

This indicates that if the logit is small, that means that the logistic neuron output is so close to 0, otherwise the logit is very large, and means that the logistic neuron output is closest to 1, in the between those limits, the neuron implies an S-shape, as shown in the figure (2.14)

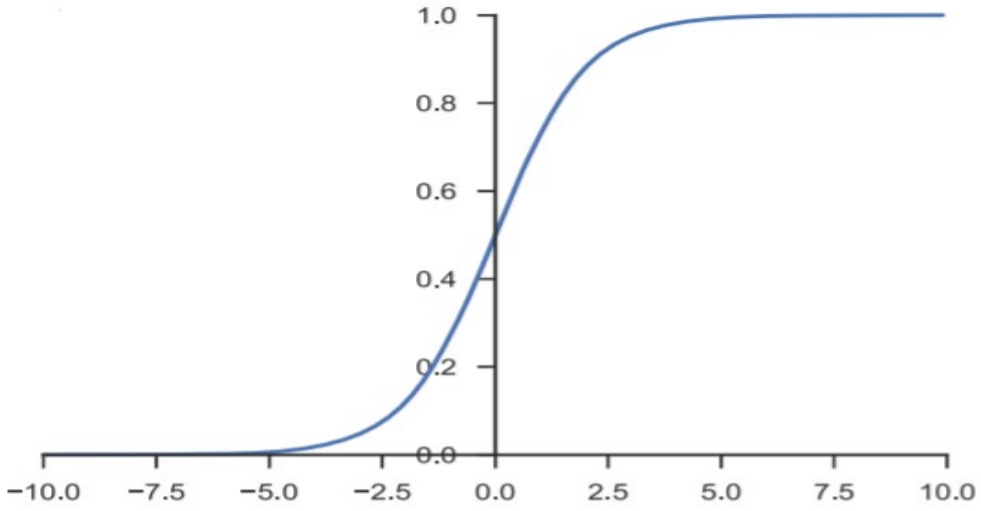

Figure 2. 14: Sigmoid activation function [40]

c. Output Layer: In this case, there is just one neuron in the output layer. This layer's goal is to forecast the maximum load value based on information gleaned from the training phase.

2.7 FEATURE SELECTION TECHNIQUES

In many areas of computer science, including computer visions and image classification, patterns recognition, and ML, the feature selection methods has been extensively used. The dimensionality of high dimensional features can be reduced by feature selection methods. The benefits of feature selection include: First, it typically reduces the high dimensions of the features and selects or indicates about robust features. Second, it typically increases the method's accuracy score [41].

2.8 DATASET DESCRIPTION

The Canadian Institute for Cyber Security (CIC) has released CSE-CIC-IDS2018 [42], a new and comprehensive intrusion detection dataset built on Amazon Web Services (AWS) in 2018. It was amassed with the intention of facilitating actual attacks. This dataset is enhancement versions of the CSE-CICIDS2017 dataset, and includes the required specifications for the attack dataset and extends protection against many common threats.

3. THE METHODOLOGY OF THE PROPOSED DDOS-IDS

3.1 INTRODUCTION

This chapter related to design and implementation of the proposed system and includes three stages: the first one related to preparing the dataset by preprocessing step. This stage contains sequence of methods for further understanding the dataset and its problems. The second stage includes utilizing the ML algorithms that described in details at chapter two to start train the proposed model, each algorithm that used at ones and trained on all features. The third stage related with applying feature selection method presented by using (PCA) and retrain all proposed models to see how many features are robust and which model have best classification accuracy.

3.2 THE METHODOLOGY OF THE PROPOSED IDS

In order to start building the proposed IDS model that can classify and distinguish between each class in the CSE-CIC-IDS-2018 dataset, let's numerate the research problem and then solve it one by one. The first problem is to understand the data and its contents. The dataset contains 79 features and 1 label (classes) and is formatted in a CSV file. Each row in the dataset represents an attack or benign label with specific features. The classes include (benign, DDOS attack-HOIC, DDOS attack-HOIC-UDP) class names and over 1048575 rows. The second problem is determining the best data splitting method. It is either utilizing the Hold-out or k-fold cross-validation method to generate the training and testing sets. After that, the third problem is determining which ML algorithms can fit the dataset with accurate classification results. The training set is used to fit the proposed ML models and the testing set is utilized for evaluating of model performance. The proposed ML algorithms to fit our dataset include (D-Tree, R-forest, Logistic-Regression, SVM, and ML-ANN). After comparing the results of each model, it is clear this dataset has a lot of features and maybe needs to the features selection method. In order to handle of this issue, the PCA can determine the high variance of the given features. After that, it needs to retrain all the ML models by utilizing the PCA method and comparing the results, figure (3.1) illustrates the diagram of the IDS procedure.

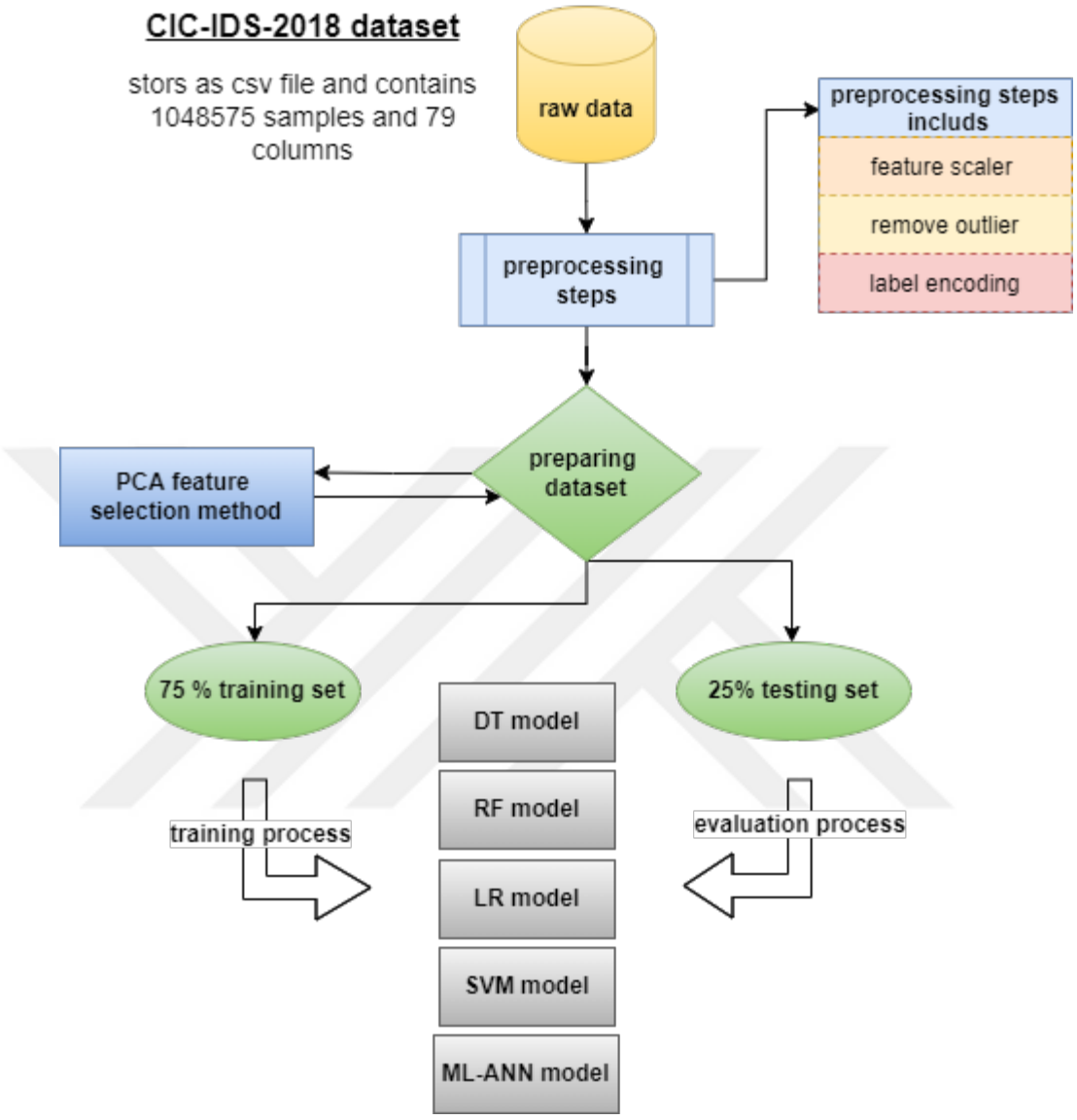

Figure 3.1: The block-diagram procedure of the proposed IDS

3.3 PREPROCESSING STEPS

This section contains several pre-processing steps that applied on the dataset includes understanding the data, missing values and outlier handling, features scaler and data splitting.

3.3.1 Understanding The Dataset

The dataset CSE-CIC-IDS-2018 on AWS contains 79 columns (78 features, 1 class label) and approximate 1048575 rows. Using Python 3.7 program language to read and load the dataset with csv file by utilizing Pandas library the classes names and distributed as follow:

Label	Sample count
Benign	360833
DDOS attack-HOIC	686012
DDOS attack-LOIC-UDP	1730

With shape at (1048575, 79), where 1048575 presents the number of rows and 79 presents number of columns.

3.3.2 Handling With Missing Values and Outlier

The proposed dataset does not have any missing values and All features are in the same data type which are integer or float 64bit and 32bit, except one column named (time-stamp) have time and date values. To address this issue, this column must be removed since it takes into account outliers from other columns and has an impact on how the ML algorithm is trained. The features were reduced by one after the time-stamp column was eliminated.

3.3.3 Features Scaler

Most likely, the attributes in our dataset have different scales, but we can't give the ML algorithm those data, so rescaling is necessary. Attributes are ensured to be scaled equally by data rescaling. Typically, attributes are rescaled to fall between 0 and 1.

With the aid of the StandardScaler class of the scikit-learn Python library, we can standardize the features to (mean = 0 and SD = 1).

3.3.4 Preparing Dataset and Splitting Procedure

When there are enough observations to produce reasonable results, the validation set approach is a very straightforward and popular method. Basically, a model is built on a train set, then its accuracy is checked on a validation set, using the data that we have divided into train set and validation set (or holdout set). And the resulting accuracy from the validation set is an estimate of the actual test data (unseen data).

And by utilizing the train-test-split class in python Sikit-learn library, the dataset was partitioned into (X-train, Y-train and X-test, Y-test) where X is input features and Y is class label. Typically, the dataset was divided into 75% training set that used to train the proposed models and 25% testing set are used for evaluating models performance.

3.4 CASE STUDY TRAINING, TESTING THE PROPOSED IDS USING ML ALGORITHMS

This section involved training and evaluating the proposed ML algorithms to fit the dataset on 75% of all samples and evaluating them on another 25% of samples. Also, it includes algorithms that are written to describe the procedure of our work in a form of a combination between the python code and description sentences.

3.4.1 Training, Testing Using DT

In order to start fitting our dataset on the first ML algorithm DT, sikit-learn library in python is used to create a DT classifier. The classifier was fitted via the x-train and y-train parameters, which are already prepared in previous section dataset splitting. The x-train parameter contains 75% from all data which are equals to 786431 rows and 78 columns: where 78 consist of (77 features, 1 class labels). The procedure of DT classifier illustrated in algorithm (3.1).

Algorithm 3.1: Train, Test DT classifier

Input	Features
Output	Classification results
Let	Train data: training set 0.75 % from all samples
	Features: (x1), . . ., (x78)
	Test data: 0.25 % from all samples
	classes: 0, 1,2 encoded class label
	max feature: maximum number of features
	min node split: minimum node split in tree

Begin	1:	Read from directory dataset.csv file
	2:	Data split into two sets (75% train, 25%test)
	3:	Set X-train, X-test = features from train dataset
	4:	Set Y-train, Y-test = classes
		Training of DT model
	5:	function Create classifier = DT (criteria = “gini-index”, Random state = 0, max feature = 0.5, node split = 2, tree depth = 10)
	6:	return classifier
	7:	end function
	8:	classifier. Fit (X-train)
		Testing of DT model
	9:	y-predict = classifier. Predict (X-test)
	10:	results = confusion matrix (y-predict, Y-test)
End		

3.4.2 Training, Testing Using RF

The same Sikit-learn library in Python is used to create the RF classifier in class named "ensample.RandomForestClassifier". It also utilizes the same splitting procedure with 75% as the training set and 25% to evaluate model performance. The hyperparameters of the RF classifier are estimated as follows:

(number_tree_estimators=30, criterion=”gini-index”, random_state=0, max_features=0.5, max_depth=10). The procedure of RF classifier illustrated in algorithm (3.2).

Algorithm 3.2: Train, Test RF classifier

Input	Features
Output	Classification results
Let	Train data: training set 0.75 % from all samples
	Features: (x1), . . ., (x78)
	Test data: 0.25 % from all samples
	classes: 0, 1,2 encoded class label
	max feature: maximum number of features

	min node split: minimum node split in tree
Begin	1: Read from directory dataset.csv file
	2: Data split into two sets (75% train, 25%test)
	3: Set X-train, X-test = features from train dataset
	4: Set Y-train, Y-test = classes
	Training of RF model
	5: Function Create classifier = Random Forest (number of Tree=30, criterion = ”gini-index”, random_state = 0,max_features = 0.5, max_depth = 10)
	6: return classifier
	7: End Function
	8: classifier.fit(Xtrain)
	Testing of RF model
	9: y-predict = classifier. predict (X-test)
	10: results = confusion matrix(y-predict, Y-test)
End	

3.4.3 Training, Testing Using LR

The multi-class logistic regression (M-LR) was used to create a model that classifies the given dataset, which includes (0, 1, 2) encoded classes. Basically, it’s also provided by python's Sikit-learn in the linear model class (Logistic-Regression). The LR classifier created and fitted the training set (x-train, y-train) with max iteration = 500, learning rate = 0.01, loss function = log-loss, and sigmoid activation function. Then the LR trained model was evaluated on (X-test, Y-test), which represents 25% and equals to 262144 samples. The learning process of LR is displayed in the algorithm (3.3).

Algorithm 3.3: Train, Test LR classifier

Input	Features
Output	Classification results
Let	Train data: training set 0.75 % from all samples
	Features: (x1), . . ., (x78)
	Test data: 0.25 % from all samples

	classes: 0, 1,2 encoded class label
Begin	1: Read from directory dataset.csv file
	2: Data split into two sets (75% train, 25%test)
	3: Set X-train, X-test = features from train dataset
	4: Set Y-train, Y-test = classes
	Training of LR model
	5: function Create classifier = Logistic-Regression (max iteration = 200, learning rate = 0.0001, loss function = log-loss, optimizer='gradient descent')
	6: return classifier
	7: end function
	8: classifier. Fit (X-train)
	Testing of LR model
	9: y-predict = classifier. predict (X-test)
	10: results = confusion matrix(y-predict, Y-test)
End	

3.4.4 Training, Testing Using SVM

In SVM, the best class separation depends on the maximum margin between support vectors to create the hyperplane. In our case, we need to identify the best decision boundary that aids in classifying the data points. The sikit-learn library sports the SVM classifier via a SVC class. The first step after the dataset is split and prepared is to create an instance of that class (SVC), then fit the training set (Xtrain, Ytrain). After fitting is completed, the next step is to evaluate the model performance using (x-test, y-test), finally compute the prediction and the classification metrics. Algorithm (3.4) shows the training and testing of SVM classifier.

Algorithm 3.4: Train, Test SVM classifier

Input	Features
Output	Classification results
Let	Train data: training set 0.75 % from all samples
	Features: (x1), . . ., (x78)
	Test data: 0.25 % from all samples

	classes: 0, 1,2 encoded class label
Begin	1: Read from directory dataset.csv file
	2: Data split into two sets (75% train, 25%test)
	3: Set X-train, X-test = features from train dataset
	4: Set Y-train, Y-test = classes
	Training of SVM model
	5: function Create classifier = SVC (kernel='linear')
	6: return classifier
	7: end function
	8: classifier. Fit (X-train)
	Testing of SVM model
	9: y-predict = classifier. predict (X-test)
	10: results = confusion matrix(y-predict, Y-test)
End	

3.4.5 Training, Testing Using ML-ANN

This experiment involved to train the proposed IDS model using Multi-Layer Artificial Neural Network (ML-ANN). The proposed network consists of 77 units as input layer followed by activation function Relu. Also, the network contains two hidden layer each one at 100 units and Softmax output layer. The optimizer used "Adam" with learning rate (α) = 0.00001 and the iteration = 300, early stop regularization technique also used to prevent overfitting problem. This model trained using 75 % as a training set (X-train, Y-train) and evaluated using testing set (X-test, Y-test). Algorithm (3.5) shows the process of the proposed network.

Algorithm 3.5: Train, Test ML-ANN classifier

Input	Features
Output	Classification results
Let	Train data: training set 0.75 % from all samples
	Features: (x1), . . ., (x78)

Test data: 0.25 % from all samples

classes: 0, 1,2 encoded class label

Begin

1: Read from directory dataset.csv file

2: Data split into two sets (75% train, 25%test)

3: Set Xtrain, Xtest = features from train dataset

4: Set Ytrain, Ytest = classes

Training of ML-ANN model

5: function Create classifier = MLP Classifier (

optimizer = ”Adam”, alpha = “1e-5”, hidden layer sizes= (200, 3), iteration = 300, early stopping = True)

6: return classifier

7: end function

8: classifier. Fit(X-train)

Testing of ML-ANN model

9: y-predict = classifier. predict (X-test)

10: results = confusion matrix(y-predict, Y-test)

End

3.5 APPLY FEATURE SELECTION METHOD PCA

In order to improve the performance of our proposed models and increase the detection rate that is presented by classification accuracy, the feature selection method PCA was proposed to do this job. This method works by computing the Egan values and Egan vectors via the covariance matrix and sorting the results from highest to lowest variances. It’s also an indication of the robust features that reached the higher classification accuracy. The experiment was done by applying PCA using sklearn. decomposition class in the sikit-learn python library with 15 components. That means we select the most robust (15) features instead of using all features (77). Finally, the selected features are fitted and retrained to the all proposed models, Algorithm 3.6 shows the implementation of PCA features section method.

Algorithm 3.6: Train, Test All proposed model with PCA

Input	Features
Output	Classification results
Let	Train data: training set 0.75 % from all samples Features: (x1), . . ., (x78) Test data: 0.25 % from all samples classes: 0, 1,2 encoded class label
Begin	1: Read from directory dataset.csv file 2: Data split into two sets (75% train, 25%test) 3: Set X-train, X-test = features from train dataset 4: Set Y-train, Y-test = classes 5: PCA-features = PCA (number of components = 15) 6: X-train = PCA. Transform (X-train) 7: X-test = PCA. Transform (X-test) Retraining all ML proposed model 8: Function Create classifier = (recreate all classifiers with usage of PCA and 15 highest features) 9: return classifier

Algorithm 3.6: Train, Test All proposed model with PCA "Algorithm continued"

	10: end function 11: classifier. Fit(X-train) Retesting all ML proposed model 12: y-predict = classifier. predict (X-test) 13: results = confusion matrix(y-predict, Y-test)
End	

4. THE RESULTS

4.1 INTRODUCTION

This chapter includes the results of the previous experiments, the results presented in figures and tables. Also, its contains comparison between the outcome classification results of the proposed methodology of IDS. Each presented tables have a short discussion and reasons behand obtained it. In addition, the implementation issues of our work also presents in this chapter.

4.2 IMPLEMENTATION ISSUES

All experiments were done using PC with hardware issues includes (CPU core I7 eighth generation, RAM 16G and 2G NVidia GPU). The software issues are python 3.7 program language and the main libraries includes (pandas, sikit-learn for ML implementation, seaborn, motplotlib, glob.. etc.)

4.3 PREPROCESSING RESULTS

Preprocessing steps consist of two main parts included dataset understanding via the statistic and features scaler which are presents as follows.

4.3.1 Statistical Summary of the Dataset

For better understanding and data representation, this section involved to shows the results of preprocessing via the statistical summary of the features and its correlation. Table (4.1) shows the four rows as short represent of the statistical summary of data observations.

Table 4.1: Statistical summary of data observations

Features name (number of columns) / **methods**	**Dst Port (0)**	**Protocol (1)**	**…**	**Idle Min (77)**
Count features	1.048575e+06	1.048575e+06	…	1.048575e+06
Mean features	1.958764e+04	6.037952e+00	…	1.633481e+04
Min features	0.000000e+00	0.000000e+00	…	0.000000e+00
Max features	6.553400e+04	1.700000e+01	…	1.060000e+08

According to statistics results that shows in table 4.1, its clearly have a variation in min values and the max values of the features. This issues can be limits by using features scaler.

4.3.2 Features Scaler Results

In order to limit the variation of the features values where presented in table (4.1), the standard scaler of preprocessing techniques that used to normalize the features values into a form of mean = 0 and standard deviation = 1. This technique is very useful to increase the detection rate as well as improving of the proposed model classification results. Table (4.2) shows the results of standard scaler preprocessing technique after computing the statistical summary.

Table 4.2: Statistical summary of training set after standard scaler

Features name (number of columns) / **Methods**	**Dst Port (0)**	**Protocol (1)**	**…**	**Idle Min (77)**
Count features	7.864310e+05	7.864310e+05	…	7.864310e+05
Mean features	-3.070103e-17	-7.122133e-16	…	9.757832e-19
Min features	**-7.203216e-01**	-9.319977e+00	…	-2.728046e-02
Max features	**1.689289e+00**	1.692177e+01	…	1.748310e+02

4.4 PROPOSED IDS CLASSIFICATION RESULTS

This section related to show the evaluation results of the proposed ML algorithm that used to build the IDS. To evaluates all experiments, confusion matrix and the classification metrics including accuracy, error rate, precision, recall are used for evaluating models performance.

4.4.1 Classification Results of DT-Model

The DT model was trained using training set which is contains 75% from all samples in dataset, also this model evaluated on testing set included 25% of samples. The evaluation results on testing set are shows as follows: Figure (4.1) illustrate the testing confusion matrix (CM) with dimensions 3*3 and presents the true and predicted for each class (0, 1, 2).

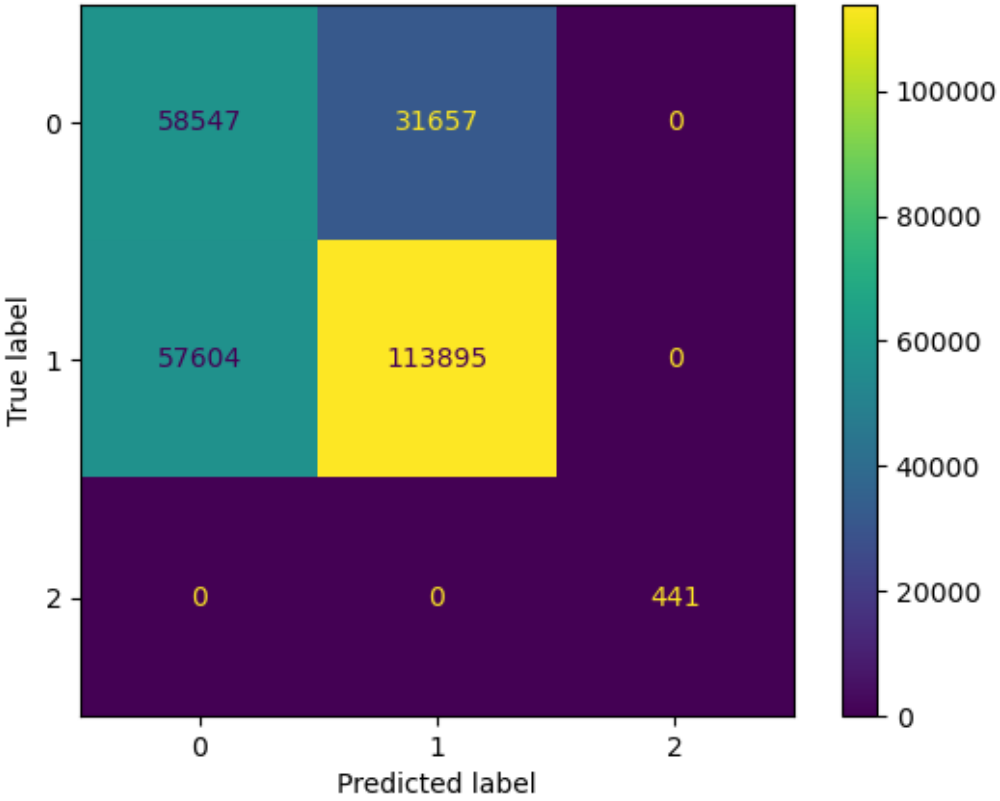

Figure 4.1: CM of testing DT model

Furthermore, figure (4.2) shows the labels density distribution and performance of the DT-model between prediction and actual values, and Table (4.3) shows the classification report of the DT-model.

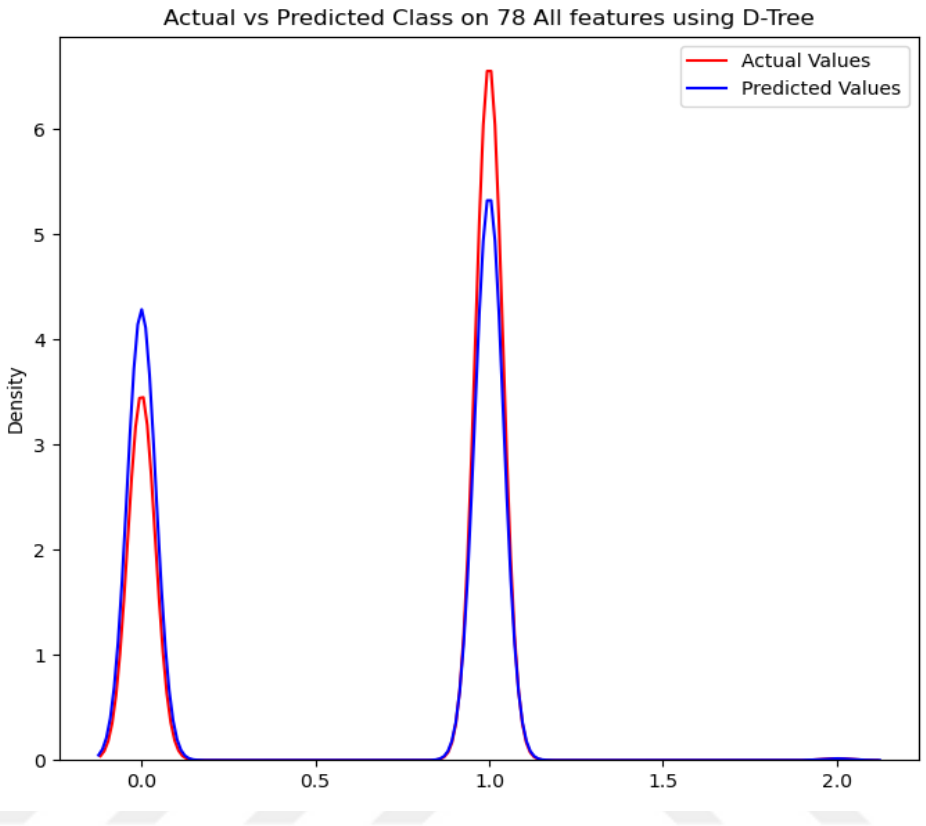

Figure 4.2 label distribution of DT-model between actual and predicted class

Table 4.3: Testing classification report of DT-model

Class	**Precision**	**Recall**	**F1-score**	**Supported samples**
0	0.5041	0.6491	0.5674	90204
1	0.7825	0.6641	0.7185	171499
2	1	1	1	441
Average	**0.7622**	**0.7711**	**0.7620**	**262144**
Accuracy = 0.6595			**MSE** = 0.3405	

According to the classification results were mentioned in table 4.3, the DT-model was misclassified in the class (0, 1) and the average accuracy achieved at 65.95% with high MSE. That because the number of observation (78 features) seems to be very large to fit all features and it needs to apply feature selection method. Let's see the performance of RF proposed model in the next section. The results should be better in RF model because it trains a multi-tree to fit the dataset and then take the average of the accuracy instead of train a single tree in DT-model.

4.4.2 Classification Results of RF-Model

The RF proposed model was evaluated on testing set which are included 25% of samples. The evaluation results on testing set are shows at Figure (4.3) contains the testing confusion matrix (CM) with dimensions 3*3 and presents the true and predicted for each class (0, 1, 2).

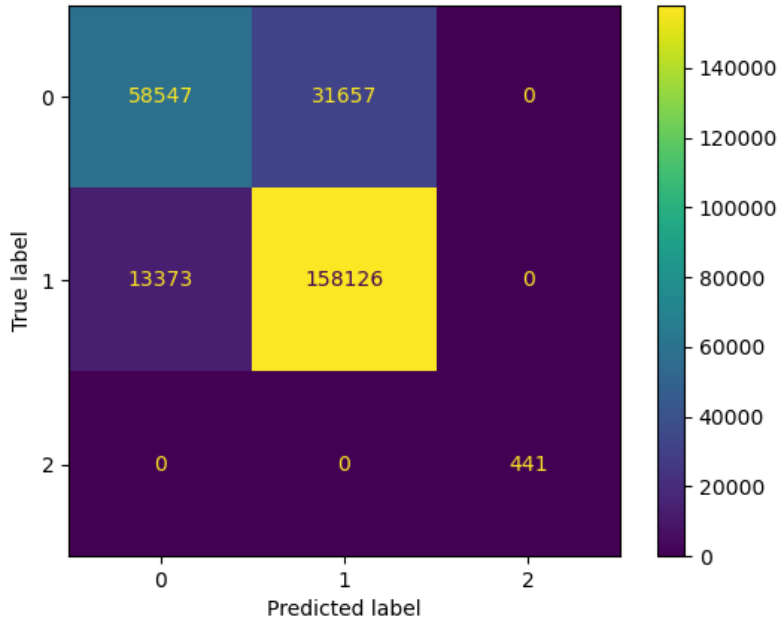

Figure 4.3: CM of testing RF model

In addition, Figure (4.4) shows the labels density distribution and performance of the RF-model between prediction and actual values. Table (4.4) shows the classification report of the RF-model.

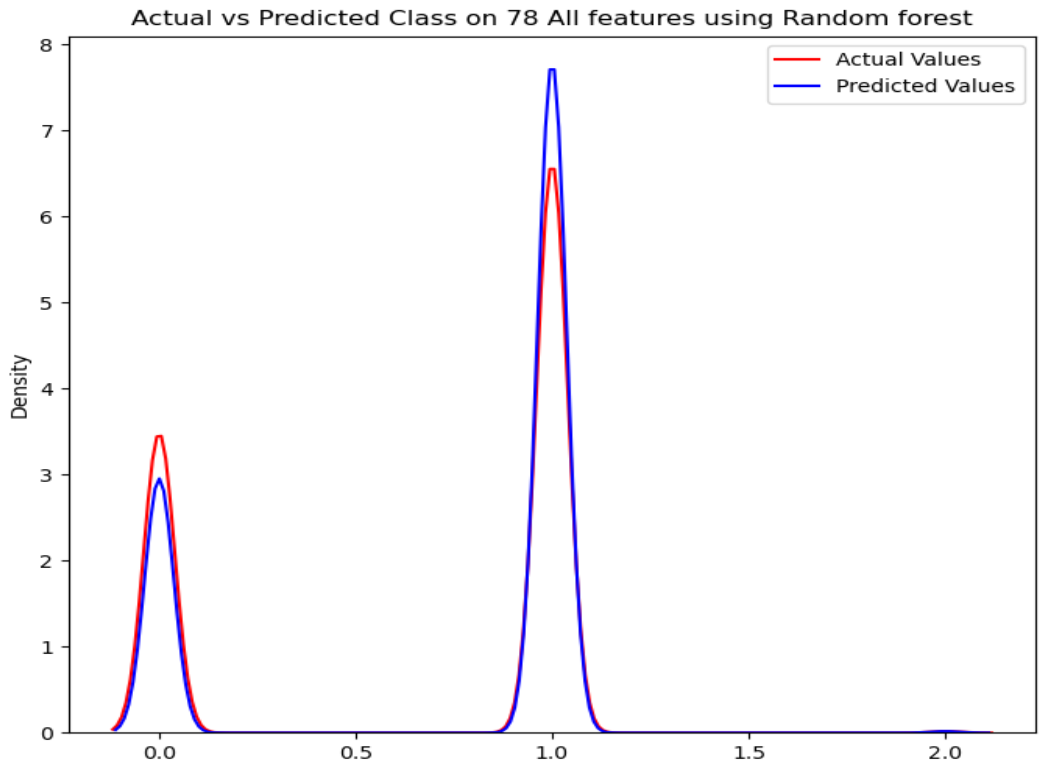

Figure 4.4: Label distribution of DT-model between actual and predicted class

Table 4.4: Testing classification report of RF-model

Class	**Precision**	**Recall**	**F1-score**	**Supported samples**
0	0. 8141	0. 6491	0. 7222	90204
1	0. 8332	0. 9220	0. 8754	171499
2	1	1	1	441
Average	**0. 8824**	**0. 8570**	**0. 8754**	**262144**
Accuracy = 0.8282				
MSE = 0.1717				

From the results that shows in table (4.4), the RF-model improves the classification accuracy to 82.82% but, it's still not the optimal results and we need more accurate detection rate of the proposed IDS.

4.4.3 Classification Results of LR-Model

The LR proposed model was evaluated on testing set which are included 25% of samples. The evaluation results on testing set are shows at Figure (4.5) contains the testing confusion matrix (CM) with dimensions 3*3 and presents the true and predicted for each class (0, 1, 2).

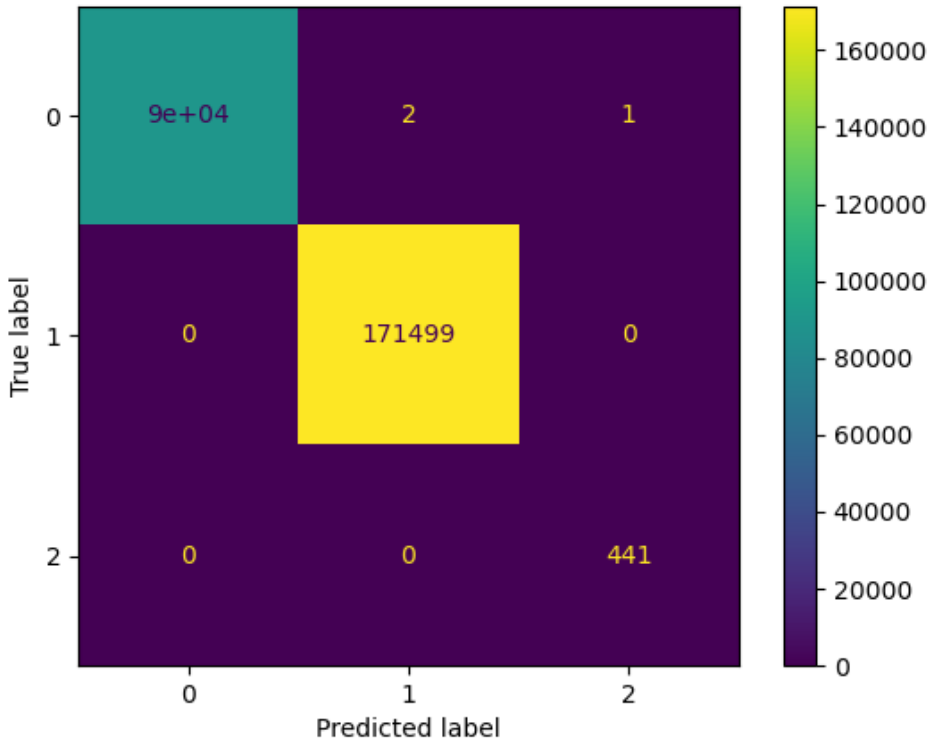

Figure 4. 5: CM of testing LR model

Moreover, Figure (4.6) shows the labels density distribution and performance of the LR-model between prediction and actual values. Table (4.5) shows the classification report of the LR-model.

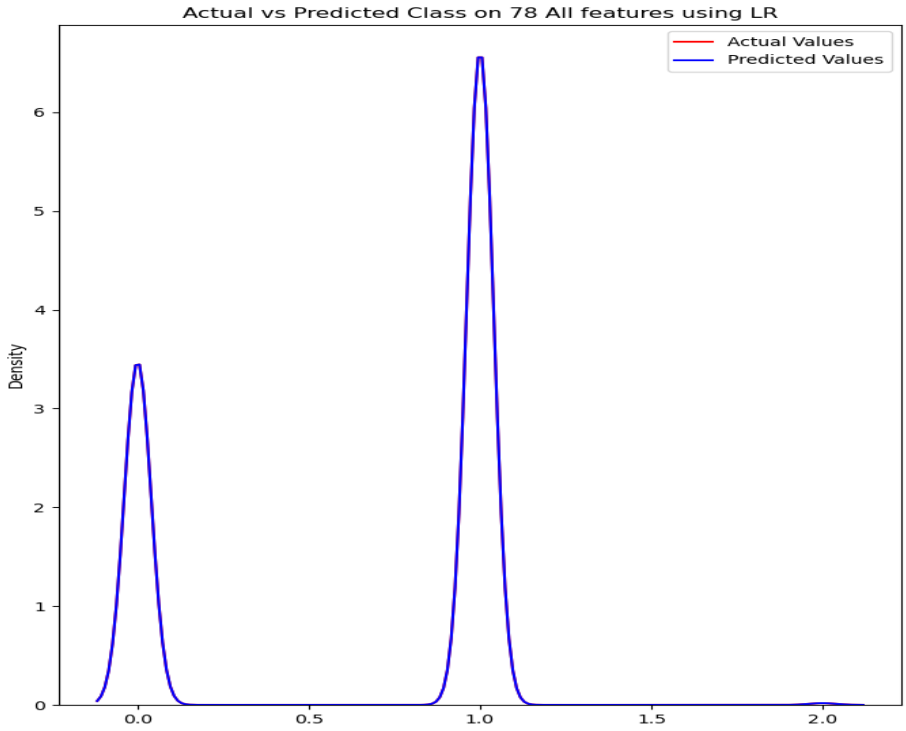

Figure 4. 6: Label distribution of LR-model between actual and predicted class

Table 4.5: Testing classification report of LR-model

Class	***Precision***	***Recall***	***F1-score***	***Supported samples***
0	1	1	1	90204
1	1	1	1	171499
2	0.9977	1	0.9989	441
Avg	**0. 9992**	**1**	**0. 9996**	**262144**
Accuracy = 0.99998				
MSE = 0.00023				

4.4.4 Classification Results of SVM-Model

The SVM proposed model was evaluated on testing set which are included 25% of samples. The evaluation results on testing set are shows at Figure (4.7) contains the testing confusion matrix (CM) with dimensions 3*3 and presents the true and predicted for each class (0, 1, 2).

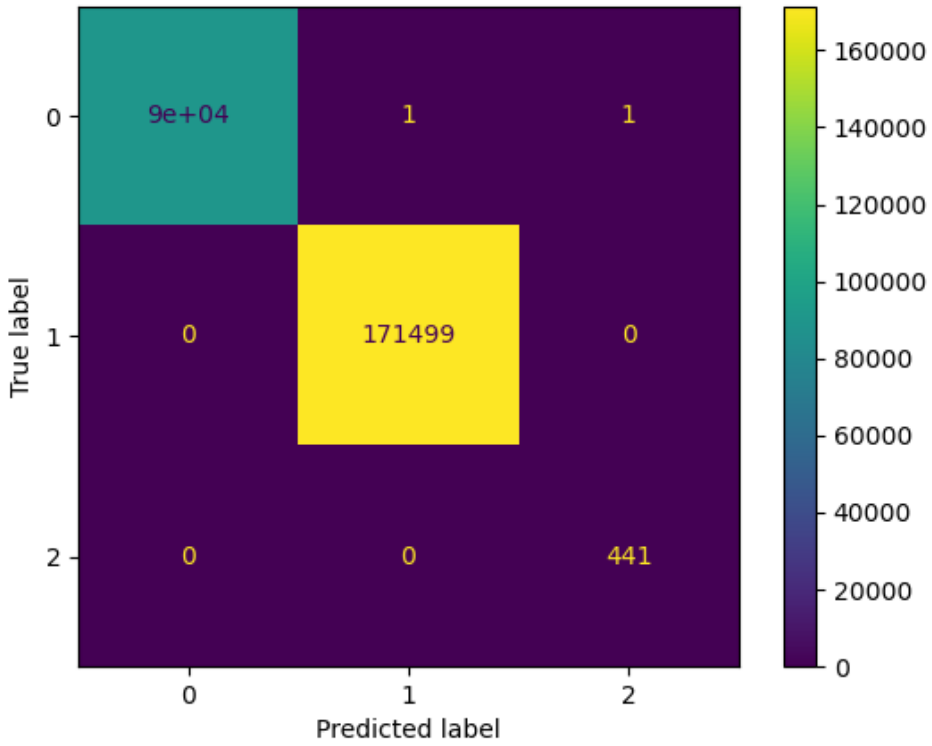

Figure 4.7: CM of testing SVM model

Moreover, Figure (4.8) shows the labels density distribution and performance of the SVM-model between prediction and actual values. Table (4.6) shows the classification report of the SVM-model.

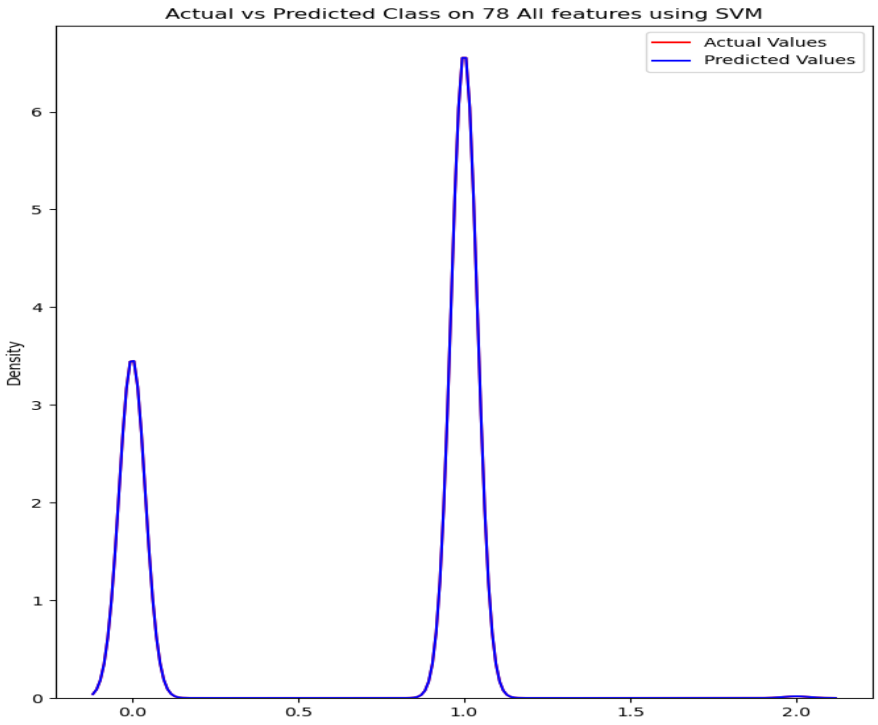

Figure 4.8: Label distribution of SVM-model between actual and predicted class

Table 4.6: Testing classification report of SVM-model

Class	***Precision***	***Recall***	***F1-score***	***Supported samples***
0	1	1	1	90204
1	1	1	1	171499
2	0. 9977	1	0.9989	441
Average	**0. 9992**	**1**	**0. 9996**	**262144**
Accuracy = 0.99999				
MSE = 0.000019				

4.4.5 Classification Results of MLP-NN-Model

The MLP-NN proposed model was also evaluated on testing set which are included 25% of samples. The evaluation results on testing set are shows at Figure (4.9) contains the testing confusion matrix (CM) with dimensions 3*3 and presents the true and predicted for each class (0, 1, 2).

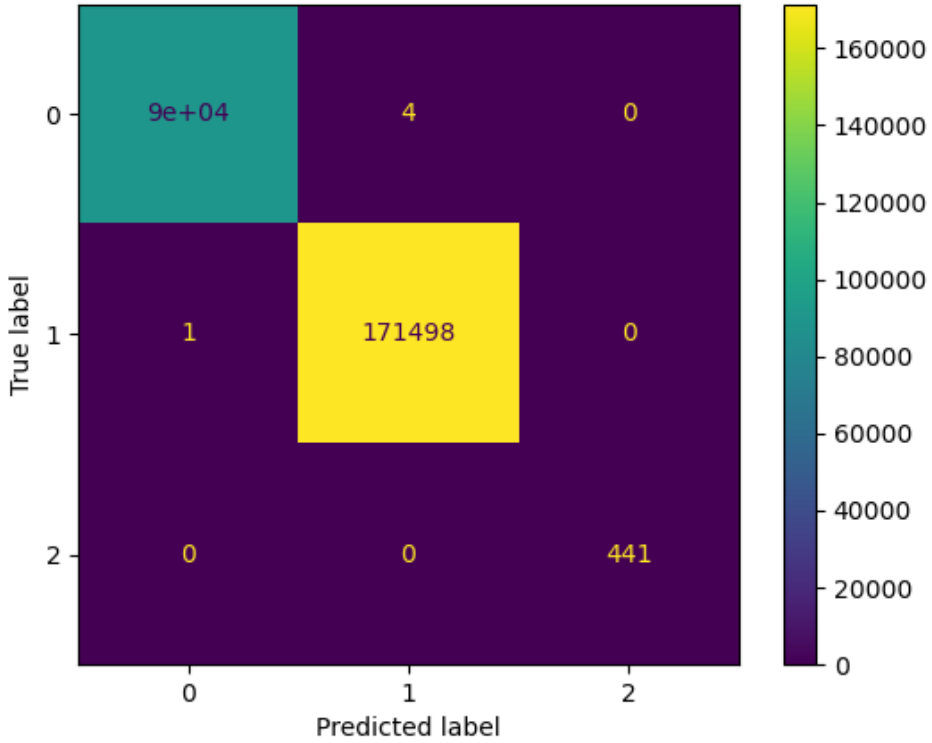

Figure 4. 9: CM of testing MLP-NN model

Moreover, Figure (4.10) shows the labels density distribution and performance of the MLP-NN -model between prediction and actual values. Table (4.7) shows the classification report of the MLP-NN -model.

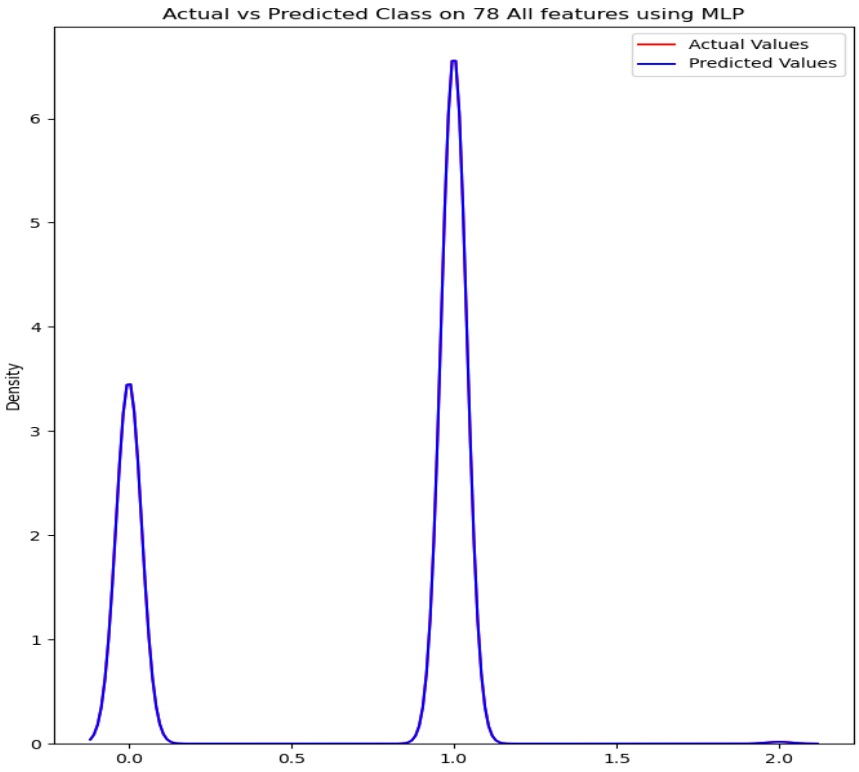

Figure 4. 10: Label distribution of MLP-NN -model between actual and predicted class

Table 4.7: Testing classification report of MLP-NN -model

Class	***Precision***	***Recall***	***F1-score***	***Supported samples***
0	0.9999	0.9999	0.9999	90204
1	0.9999	0.9999	0.9999	171499
2	1	1	1	441
Average	**0.9999**	**0.9999**	**0.9999**	**262144**
Accuracy = 0.99998				
MSE = 0.000019				

4.5 EXPERIMENTAL RESULTS OF APPLYING PCA TO THE ALL MODELS

This experiment involved retraining the proposed ML models by applying the PCA feature selection method. As mentioned earlier, this method works by computing Egan values and an Egan vector with a covariance matrix for all features, and then sorting the features according to highest variance to lowest variance. It's a very powerful method to identify the robust features. The output of the PCA is illustrated in table (4.8), which is the sorted features according to highest variance.

Table 4.8: The PCA output variance results.

Number of features component	PCA Variance values	Number of features component	PCA Variance values
1	**3.24814350e-01**	22	4.74011355e-04
2	**2.09720689e-01**	23	3.62316606e-04
3	**1.08312813e-01**	24	2.93828295e-04
4	**7.18553917e-02**	25	2.65310164e-04
5	**5.86990076e-02**	26	1.62043045e-04
6	**4.45226107e-02**	27	1.47510786e-04
7	**2.97387519e-02**	28	1.21875736e-04
8	**2.76632665e-02**	29	1.10811704e-04
9	**2.39571375e-02**	30	1.09751894e-04

Table 4.8: The PCA output variance results "Table continued"

10	**1.75332157e-02**	31	9.68858431e-05
11	**1.47702502e-02**	32	6.58714074e-05
12	**1.47061227e-02**	33	5.60422694e-05
13	**1.32074744e-02**	34	4.67980500e-05
14	**1.08842327e-02**	35	3.83008649e-05
15	**9.59272758e-03**	36	2.93568219e-05
16	7.28700519e-03	37	1.84979868e-05
17	3.40671053e-03	38	1.33563187e-05
18	2.71322876e-03	39	8.25890949e-06
19	2.35216019e-03	40	6.50386870e-06
20	1.17911601e-03	…	…
21	6.44670675e-04	78	1.31804761e-33

From the PCA results that are shown in table (4.8), the results refer to the robust features in the dataset, and it is clear we don't need all 78 features to reach the optimal detection rate. Furthermore, the first sorted 15 features from the table above could be more reliable and accurate classification results even with the week models (DT, RF). Let's present the comparison results between each experiment. Table (4.9) shows the comparison for all ML

proposed models before applying the PCA, and Table (4.10) illustrates the comparison after applying the PCA.

Table 4.9: The comparison results of ML-model before PCA

ML- proposed models for IDS	**Testing Accuracy %**	**Testing MSE**
DT	65.95	0.3405
RF	82.82	0.1717
LR	99.998	0.00023
SVM	99.999	0.000019
MLP-NN	99.998	0.000019

Table 4.10: The comparison results of ML-model after PCA

ML- proposed models for IDS	**Testing Accuracy %**	**Testing MSE**
DT	99.9950	0.000095
RF	99.9984	0.000015
LR	99.9927	0.000095
SVM	99.9935	0.000076
MLP-NN	**99.9992**	**0.000007**

According to the comparison results mentioned in both (table 4.9 and 4.10), this experiment improved the classification accuracy of week models (DT, RF). Furthermore, the strong models (LR, SVM, MLP-NN) still have the same strong classification performance after applying PCA (15 features) and also the size is small then used (78 features). Table (4.11) illustrates the comparison results of the proposed IDS based on MLP-NN and other related works.

Table 4. 11: The comparison results of the proposed IDS and other related works

Methods for IDS	**Dataset**	**Results of the Accuracy %**
In [5] used SVM	KDD99	89.8
In [6] used ANN	KDD99	99.8
In [8] used Particle Swarm, Ant Colony, and genetic algorithm (GA)	(NSL-KDD and UNSW-NB15)	85.8
In [12] SVMs, Naive Bayes, LR and RF	KDD99	99.81
Proposed IDS based on MLP-NN	**CSE-CIC-IDS2018**	**99.9992**

5. THE DISCUSSION AND CONCLUSIONS

5.1 RESULTS DISCUSSION

Our methodology is to build a reliable and accurate IDS based on ML to classify and prevent DDoS attacks to protect any system working on the network from temporary or complete system failure. In order to start building the IDS model, the SCIC-2018 dataset was used, as it contains the attack to be classified, which is distributed denial of service. The five models were proposed using machine learning algorithms, which included (DT, RF, LR, SVM, MLP-NN), each one trained on 75 % of the dataset samples and tested on another 25%.

From the testing results of the proposed ML models, the DT was not good for this task, and because of the large number of features (78), this model is going through the overfitting problem. Furthermore, the RF model performance was better than DT in classification accuracy, but it's still not enough and the results need improvement. The accuracy of the RF model reached 82% better than the DT model, as mentioned in the comparison results table (4.9). Also, the other proposed models, including LR, SVM, and MLP-NN, have very good performance in testing classification. This experiment was done to fit all 78 features.

The next experiment is to apply the PCA feature selection method to better model performance improvement. The PCA results illustrated in table (4.8) where sorted the features from the highest to the lowest variance. The PCA results are also indicated to selecting 15 features instead of 78. We can take less than 15, but we need the proposed model to be more reliable. Therefore, the 15 features were selected and retrained for all proposed ML models. According to the comparison results of applying the PCA method were mentioned in table (4.10), both DT and RF performances were increased to reach the highest accuracy at 99.99%. Furthermore, the LR, SVM, and MLP-NN models still have very good classification results at accuracy approximates 99.99% on 15 selected features with model size less than used all 78 features and without PCA. In addition, the proposed MLP-NN model has optimal performance at accuracy 99.9992% and lowest MSE 0.000007 in the comparison result where illustrated in table (4.10).

5.2 CONCLUSIONS

The set of conclusions are mentioned on this section as points, its obtained from this work.

a. The statistical summary was useful to better understand the dataset and present its problems. It also refers to the variation of feature values.
b. The standard scaler preprocessing technique improved the model computations in training and testing set by limiting the variation of features.
c. Training several ML models on the same dataset was good for selecting the best one.
d. The PCA feature selection method improves the classification results and prevents the overfitting problem in the DT model.
e. The MLP-NN model achieved a magnificent testing performance even when utilizing all the features in the dataset.

REFERENCES

[1] S. K. Wagh, V. K. Pachghare, and S. R. J. I. J. o. C. A. Kolhe, "Survey on intrusion detection system using machine learning techniques," vol. 78, no. 16, 2013.

[2] N. Shah, S. J. I. J. o. E. Valiveti, and C. S. Engineering, "Intrusion detection systems for the availability attacks in ad-hoc networks," vol. 1, no. 03, pp. 1850-1857, 2012.

[3] A. Khraisat, I. Gondal, P. Vamplew, and J. J. C. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," vol. 2, no. 1, pp. 1-22, 2019.

[4] H. M. A. Alsafi, S. S. J. J. o. E. T. i. C. Basamh, and I. Sciences, "A review of intrusion detection system schemes in wireless sensor network," vol. 4, no. 9, pp. 688-697, 2013.

[5] M. V. Kotpalliwar and R. Wajgi, "Classification of attacks using support vector machine (svm) on kddcup'99 ids database," in *2015 Fifth International Conference on Communication Systems and Network Technologies*, 2015, pp. 987-990: IEEE.

[6] B. Subba, S. Biswas, and S. Karmakar, "A neural network based system for intrusion detection and attack classification," in *2016 Twenty Second National Conference on Communication (NCC)*, 2016, pp. 1-6: IEEE.

[7] X. An, J. Su, X. Lü, F. J. E. J. o. W. C. Lin, and Networking, "Hypergraph clustering model-based association analysis of DDOS attacks in fog computing intrusion detection system," vol. 2018, no. 1, pp. 1-9, 2018.

[8] B. A. Tama, M. Comuzzi, and K.-H. J. I. a. Rhee, "TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system," vol. 7, pp. 94497-94507, 2019.

[9] H. He, X. Sun, H. He, G. Zhao, L. He, and J. J. I. A. Ren, "A novel multimodal-sequential approach based on multi-view features for network intrusion detection," vol. 7, pp. 183207-183221, 2019.

[10] A. Thakkar and R. J. P. C. S. Lohiya, "A review of the advancement in intrusion detection datasets," vol. 167, pp. 636-645, 2020.

[11] C. Khammassi and S. J. C. N. Krichen, "A NSGA2-LR wrapper approach for feature selection in network intrusion detection," vol. 172, p. 107183, 2020.

[12] T. Saranya, S. Sridevi, C. Deisy, T. D. Chung, and M. A. J. P. C. S. Khan, "Performance analysis of machine learning algorithms in intrusion detection system: a review," vol. 171, pp. 1251-1260, 2020.

[13] S. M. Kasongo and Y. J. J. o. B. D. Sun, "Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset," vol. 7, no. 1, pp. 1-20, 2020.

[14] N. Oliveira, I. Praça, E. Maia, and O. J. A. S. Sousa, "Intelligent cyber attack detection and classification for network-based intrusion detection systems," vol. 11, no. 4, p. 1674, 2021.

[15] K. Scarfone and P. J. N. s. p. Mell, "Guide to intrusion detection and prevention systems (idps)," vol. 800, no. 2007, p. 94, 2007.

[16] A. Aldweesh, A. Derhab, and A. Z. J. K.-B. S. Emam, "Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues," vol. 189, p. 105124, 2020.

[17] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, K.-Y. J. J. o. N. Tung, and C. Applications, "Intrusion detection system: A comprehensive review," vol. 36, no. 1, pp. 16-24, 2013.

[18] I. J. C. F. Kara and Security, "A basic malware analysis method," vol. 2019, no. 6, pp. 11-19, 2019.

[19] S. S. Priya, M. Sivaram, D. Yuvaraj, and A. Jayanthiladevi, "Machine learning based DDoS detection," in *2020 International Conference on Emerging Smart Computing and Informatics (ESCI)*, 2020, pp. 234-237: IEEE.

[20] K. M. Elleithy, D. Blagovic, W. K. Cheng, and P. Sideleau, "Denial of service attack techniques: analysis, implementation and comparison," 2005.

[21] G. Carl, G. Kesidis, R. R. Brooks, and S. J. I. I. c. Rai, "Denial-of-service attack-detection techniques," vol. 10, no. 1, pp. 82-89, 2006.

[22] A. J. I. A. Aljuhani, "Machine learning approaches for combating distributed denial of service attacks in modern networking environments," vol. 9, pp. 42236-42264, 2021.

[23] A. Bhardwaj, V. Mangat, R. Vig, S. Halder, and M. J. C. S. R. Conti, "Distributed denial of service attacks in cloud: State-of-the-art of scientific and commercial solutions," vol. 39, p. 100332, 2021.

[24] Y. Hamid, M. Sugumaran, and L. Journaux, "Machine learning techniques for intrusion detection: a comparative analysis," in *Proceedings of the International Conference on Informatics and Analytics*, 2016, pp. 1-6.

[25] Y. Hamid, M. Sugumaran, V. J. B. J. o. A. S. Balasaraswathi, and Technology, "Ids using machine learning-current state of art and future directions," vol. 15, no. 3, 2016.

[26] B. J. I. J. o. S. Mahesh and R. . "Machine learning algorithms-a review," vol. 9, pp. 381-386, 2020.

[27] P. G. S. (2022). *What Is Machine Learning and How Does It Work?* Available: https://www.simplilearn.com/what-is-machine-learning-and-why-it-matters-article

[28] D. Sarkar, R. Bali, and T. J. A. P.-S. G. T. B. R.-W. I. S. B. A. Sharma, "Practical machine learning with python," 2018.

[29] A. Sarica, A. Cerasa, and A. J. F. i. a. n. Quattrone, "Random forest algorithm for the classification of neuroimaging data in Alzheimer's disease: a systematic review," vol. 9, p. 329, 2017.

[30] A. J. I. J. o. E. T. Pradhan and A. Engineering, "Support vector machine-a survey," vol. 2, no. 8, pp. 82-85, 2012.

[31] skilltohire. (2020). *Support Vector Machine*. Available: https://medium.com/@skilltohire/support-vector-machines-4d28a427ebd

[32] V. Kanade. (2022). *What Is Linear Regression? Types, Equation, Examples, and Best Practices for 2022*. Available: https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-linear-regression/

[33] M. A. Jabbar and A. M. J. I. J. o. S. Radhi, "Diagnosis of Malaria Infected Blood Cell Digital Images using Deep Convolutional Neural Networks," pp. 380-396, 2022.

[34] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[35] H. J. J. o. B. S. Abdi, "A neural network primer," vol. 2, no. 03, pp. 247-281, 1994.

[36] J. J. I. J. o. Q. C. Behler, "Constructing high-dimensional neural network potentials: a tutorial review," vol. 115, no. 16, pp. 1032-1050, 2015.

[37] A. M. Alzakkar, I. Valeev, N. Mestnikov, and E. Nurullin, "The artificial power system networks stability control using the technology of neural network," in *E3S Web of Conferences*, 2019, vol. 124, p. 05002: EDP Sciences.

[38] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, 2013, vol. 30, no. 1, p. 3: Citeseer.

[39] B. Xu, R. Huang, and M. J. a. p. a. Li, "Revise saturated activation functions," 2016.

[40] M. D. Zeiler and R. J. a. p. a. Fergus, "Stochastic pooling for regularization of deep convolutional neural networks," 2013.

[41] F. Song, Z. Guo, and D. Mei, "Feature selection using principal component analysis," in *2010 international conference on system science, engineering design and manufacturing informatization*, 2010, vol. 1, pp. 27-30: IEEE.

[42] I. Sharafaldin, A. H. Lashkari, and A. A. J. I. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," vol. 1, pp. 108-116, 2018.