A Machine Learning Based Approach For DDos Attack Detection

by

Tasmiah Tahrim
19101101
MD Asif Sharan
23341133
Meshaq Monsur
18201134
MD Abid Hasan
18101692
Tanzina Binte Azad
20201217

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
September 2023

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.

2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.

3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.

4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Tasmiah Tahrim

Tasmiah Tahrim

19101101

Meshaq

Meshaq Monsur

18201134

Abid

MD Abid Hasan

18101692

Sharan

MD Asif Sharan

23341133

Tanzina Binte Azad

Tanzina Binte Azad

20201217

Approval

The thesis/project titled "A Machine Learning Based Approach For DDos Attack Detection" submitted by

1. Tasmiah Tahrim (19101101)
2. Meshaq Monsur (18201134)
3. MD Asif Sharan (23341133)
4. MD Abid Hasan (18101692)
5. Tanzina Binte Azad (20201217)

Summer 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science and Engineering on September 25, 2023.

Examining Committee:

Supervisor:
(Member)

Arif Shakil

Lecturer

Department of Computer Science and Engineering

BRAC University

Co-Supervisor:
(Member)

Faisal

Md Faisal Ahmed

Lecturer

Department of Computer Science and Engineering

BRAC University

Program Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam

Associate Professor

Department of Computer Science and Engineering

BRAC University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD

Chairperson and Associate Professor

Department of Computer Science and Engineering

BRAC University

Abstract

The modern era saw the rise of technologies in almost every sector. Computers are gradually becoming faster and smaller, also allowing people to utilize them almost everywhere. As computer technology has become an important part to simplify our life at work, the security of computer networks is one of the hardest challenges for the technology experts to overcome. Network security is a must because it protects private information from online attacks and upholds the dependability of the network. In this study, after reviewing a few previous papers and research works, we decided to work on the detection process of DDoS that can be used on the web or server security. Working on the datasets (CICDDoS2019) to merge them and create a new taxonomy for detecting DDoS attacks was our primary step. Then, the data were generated for the two types of attack which are Reflection based and Exploitation based to reduce the time consumption. Thirdly, using the generated dataset, some Machine Learning based models and classifiers have been implemented on important features that have the most contributions. For getting a better accuracy rate, Random Forest, Naive Bayes, Decision Tree and XGBoost model were applied. Finally, we get a better accuracy rate with these models to detect the attack in a reduced amount of time.

Keywords: DDoS Attack, Cyber-security, Machine learning, Random Forest Classifier, HTTP based Protocol, Transport Layer, Application Layer.

Dedication

We dedicate our thesis to our parents, family, and, most importantly, our respected faculties.

Acknowledgement

All praise to Almighty Allah for whom we have completed it. Then Arif Shakil, and our Co-supervisor, Md Faisal Ahmed, for their support and advices regarding our work. They both helped us whenever we needed help. Moreover, we sincerely thank all our respective faculties, companions, and staff for all their support. We are forever indebted to our parents for their continuous support and prayers. Only because of them did we get this amazing opportunity to learn and complete our undergraduate degree. With their kind support and prayer, we are now almost done with our graduation.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

AI	Artificial Intelligence
DDOS	Distributed Denial of Service
DNS	Domain Name System
IT	Information Technology
ML	Machine Learning
NTP	Network Time Protocol
SNMP	Simple Network Management Protocol

Chapter 1

Introduction

Internet security nowadays is one of the biggest concerns, particularly with the everyday demand for IT services. One of the most popular network attacks is DDoS. DDoS attacks can occur when a malevolent person targets a website or server in order to deny services by flooding it with unwanted data. This leads legitimate users to experience service delays. Denial of Service (DoS) attacks occur because of a single source, but Distributed Denial of Service (DDoS) attacks occur when the attack originates from a large number of sources, such as a Botnet, which remotely manages the devices for malevolent purposes.

A number of studies have been found where some new taxonomies are proposed about DDoS attacks. It is very important to get acknowledged to new attacks and come up with new counters. So, new attacks have been analyzed that can be carried out using TCP/UDP based protocols at the application layer.[21]

DDoS mitigation is a big challenge nowadays as the resources are limited. Mitigation capabilities, capacity, and scalability vary from network to network. The actual mitigation activities must take into account all of these variables and more, such as preferences for the amount of security filters used on routers, whether NETCONF or Flowspec should be utilized, and so on. It is now feasible to detect DDoS activity early and adopt quick, targeted, and optimal mitigation actions to stop such attacks using AI/ML based algorithms. So that, users can protect their networks from malicious DDoS attacks, keep services functioning, and keep people online by combining big data analytics and AI/ML into all phases of a complete DDoS defense plan.

Exploitation Based Attacks: The packets are sent by the attackers to reflect servers with the target victim IP address as the source IP address in an effort to attack the target with reply packets. These attacks can also be carried out utilizing transport layer protocols, such as TCP and UDP, through application layer protocols. Attacks based on UDP and TCP exploitation techniques include SYN flood and UDP flood, respectively. By providing with several UDP packets to the remote host, UDP flood attack is fully launched.[1]

Reflection Based Attacks: In this attack, attackers mostly try to hide any evidence regarding their involvement. To accomplish this, they try to alter the way to detect them through internet services operated by default so that the services successfully conceal the real attacker. The vast number of Domain Name System (DNS), Network Time Protocol (NTP), and Simple Network Management (SNMP) servers are services that are frequently utilized in these kinds of assaults. One of the main draws for attackers to use a DDoS approach is because of this. Due to the fact that most servers don't keep thorough records of the services that have utilized them, internet services can make it more challenging for defenders to identify the attack's origin.[1]

1.1 Motivation

Today's era is the modern era of AI rising where various AI based technologies are taking it over. Artificial Intelligence (AI) is being used in every other aspect of life let it be small or huge. It's revolutionizing the IT industry and the Cyber Security sector is no different. Cyber Security related software such as Anti-viruses, Malware detection and protection are becoming AI dependent. As the viruses and attacking malwares are also being dominated by AI, it would be a sure job to take concern about how AI controlled DDos attacks can be prevented with machine learning. For example, when a gaming server is down due to several reasons (hardware failure, power outage, software issue, network problem), various cyber attacks can occur during that time. Most of these attacks are based on AI software. So to prevent these, ML in the world of cyber protection is a necessity. Attackers frequently alter their expertise approaches and so escape the current detection measures. As the attacking approaches keep changing, we need to keep up with those by developing more accurate prevention systems using Machine Learning. This certain area is still not explored yet as much as the other ones. For this reason, it holds great opportunities for research and development.

1.2 Research Problem

According to past records, the first DDoS attack took place in 1974. The hackers launched their attack on this system in 1990, when IRC (Internet Relay Chat) started to gain popularity as an online communication tool. Additionally, a SYN flood, a technique that has come to be regarded as a standard DDoS assault, took one of the first internet service providers offline for a number of days in 1996. After that, in 1999, hackers launched a significant attack with the intention of breaching the University of Minnesota's computer network. Over the subsequent years, the onslaught spread. Cisco projects that they will have quadrupled from the 7.9 million reported in by 2023.

DDoS attacks can happen through a variety of protocols which includes TCP, UDP, ICMP, and HTTP, at the network, transport, and application layers. The attacks can be detected and defended in many ways. A DDoS attack employs many IP addresses or workstations, often including thousands of compromised hosts. Now-a-days, hackers are executing more denial DDoS attacks than ever before, setting new targets, and developing new botnets, as the online threat landscape continues to grow at a rapid pace. So, the tempting current scenario necessitates stronger security and privacy safeguards. Mainly, DDoS attack detection is the process of differentiating the attacks from normal network traffic.[3]

1.3 Research Objectives

This research aims at developing a DDoS detection approach based on the Machine Learning model for networking security of Application Layer and we can say that it will be an efficient way to detect the attacks in real-time. The packets we get from that OSI model are TCP, HTTP, SYN, UDP, UDP-Lag, ICMP, NTP, DNS and some others. We gathered some of these features that have been extracted to a dataset that we used for our research work.
The following goals we have aimed to reach:
1. Gain a thorough understanding of DDoS attacks and how they occur.
2. Making a better understanding of all the specifications and how to categorize the data.
3. Generating the categorization of data and how to choose the model.
4. Analyzing the models and developing a better model and improving the runtime.
5. Further research about how to improve the model.

1.4 Thesis structure

Chapter 1 - Introduction: motivation, research problem research objectives.
Chapter 2 - Background and background information about algorithms.
Chapter 3 - Proposed methodology: data preprocessing, feature selection processes and used models.
Chapter 4 - Dataset: Data description, data analysis and visualization.
Chapter 5 - Experimentation: workflow environment and evaluation.
Chapter 6 - Result Analysis: Each model's evaluation metrics, curve and data visualization and comparative analysis discussion.
Chapter 7 - Conclusion and Future works.

Chapter 2

Background

Background

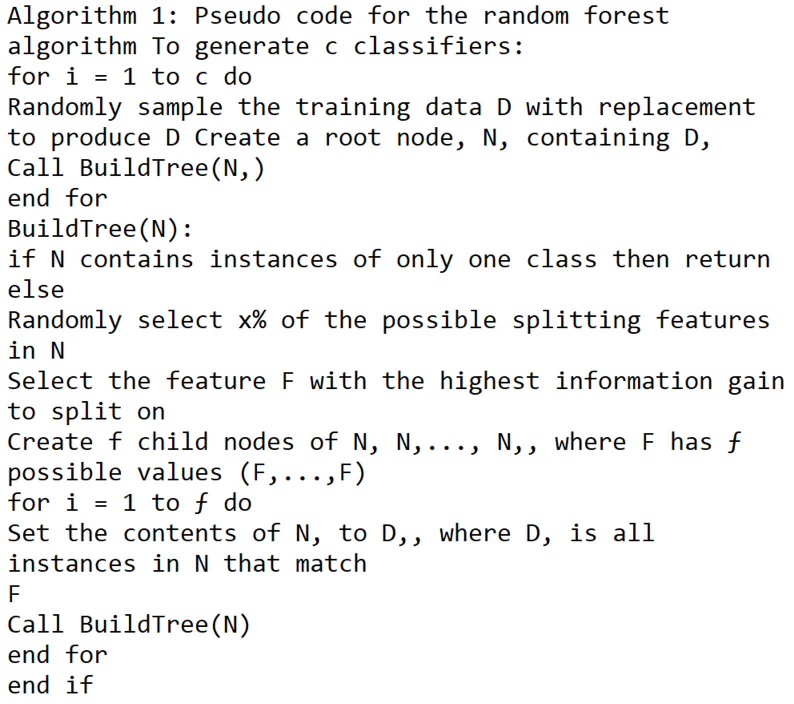
One of the most popular network attacks is DDoS. A DDoS attack happens when a malevolent person targets a website or server in order to deny services by flooding it with unwanted data. This leads legitimate users to experience service delays. Denial of Service (DoS) attacks occur when the attack originates from a single source, but Distributed Denial of Service (DDoS) attacks occur when the assault originates from a large number of sources, such as a Botnet, which remotely manages the devices for malevolent purposes. To identify DDoS attacks, a collection of eight supervised machine learning methods is chosen, and the optimal model in terms of accuracy, precision, recall, and false alarm rate is determined. A common benchmark dataset CIC-IDS2017 is utilized for training and assessing experimental findings. The preprocessing stage includes K-Fold cross validation. The eight models are then trained and evaluated using K-Fold cross validation to see which is the best at detecting DDoS attacks at the early stage. We assessed the trained models with the parameters Accuracy, Precision, Recall, and FAR during the testing phase. We discovered that Random Forest is the best model among eight when all parameters are considered. It achieved 99.88 percent accuracy, 99.88 percent precision, 100 percent recall, and a false alarm rate of 0.05 percent to identify DDoS attacks as soon as possible.

2.1 Algorithms

Some Machine Learning algorithms have been applied and run in this project. The algorithms have been chosen according to our data. The models that have been followed which are Random Forest Algorithm, Naive Bayes Algorithm, Decision Tree Classifier, Gradient Boost Algorithm.

2.1.1 Random Forest Classifier

Random Forest is a type of supervised machine learning algorithms that are commonly used in classification and regression related problems. Also, it is a tree based probabilistic classifier. Each decision tree consists of a number of internal nodes and leaves. The internal node uses the selected function to determine how to split the dataset into two separate sets with similar answers. Build a decision tree based on different samples and use the majority vote for averaging for classification and regression.[10] The emphasis is on optimizing the split for each node rather than considering the impact of the split on the whole tree. Random forests are suitable for situations where there are large data sets and interpretability. Random forests improve bagging because they de-correlate trees by introducing a split of features into random subsets. This means that each time the tree is split, the model only considers a small subset of the features instead of all the features of the model.[12]

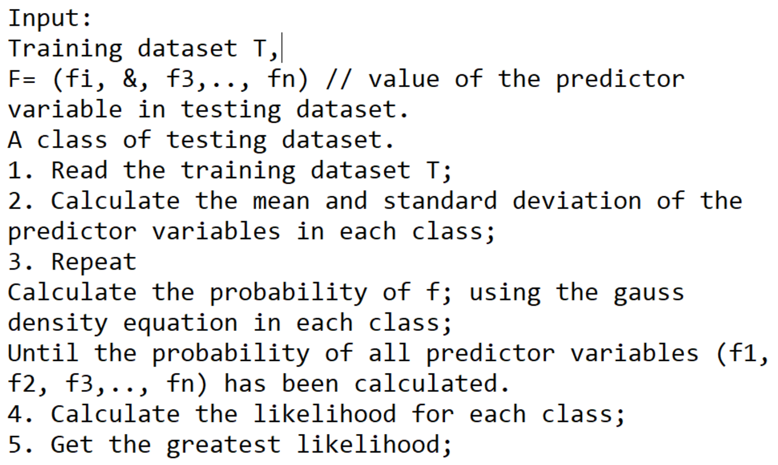


```
Algorithm 1: Pseudo code for the random forest
algorithm To generate c classifiers:
for i = 1 to c do
Randomly sample the training data D with replacement
to produce D Create a root node, N, containing D,
Call BuildTree(N,)
end for
BuildTree(N):
if N contains instances of only one class then return
else
Randomly select x% of the possible splitting features
in N
Select the feature F with the highest information gain
to split on
Create f child nodes of N, N,..., N,, where F has ƒ
possible values (F,...,F)
for i = 1 to ƒ do
Set the contents of N, to D,, where D, is all
instances in N that match
F
Call BuildTree(N)
end for
end if
```

Figure 2.1: Pseudo code for Random Forest Algorithm

2.1.2 Naive Bayes Classifier

Naive Bayes algorithm is another algorithm of supervised learning which is based on the Bayes' Theorem. Generally this algorithm is used to solve classification based problems in machine learning. The main advantage is it handles both continuous and numerical data. [6] If the feature independence assumption can be made correct correct, it performs better than other models and requires much less training data. It scales well with the number of predictors and data points. Another point is, this classifier can work fast and saves a lot of time. and it can be used for real-time predictions. It does not respond to irrelevant features. If the feature independence assumption is correct, it performs better than other models and requires much less training data.[15]

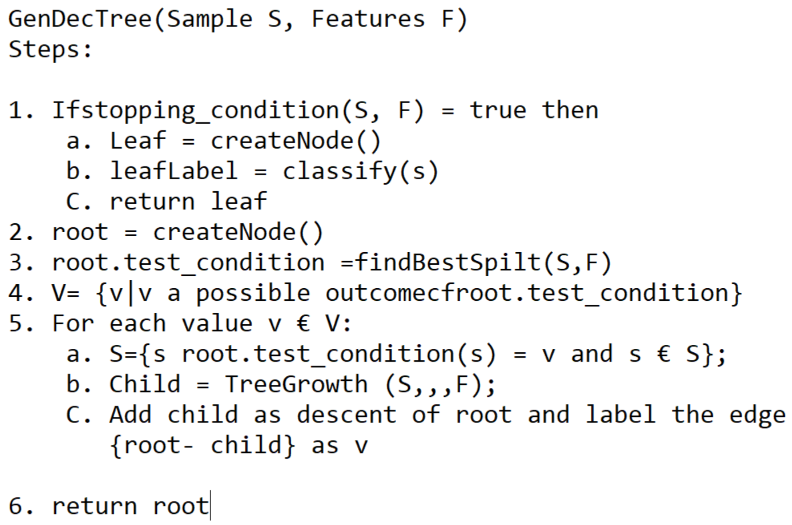


```
Input:
Training dataset T,
F= (fi, &, f3,.., fn) // value of the predictor
variable in testing dataset.
A class of testing dataset.
1. Read the training dataset T;
2. Calculate the mean and standard deviation of the
predictor variables in each class;
3. Repeat
Calculate the probability of f; using the gauss
density equation in each class;
Until the probability of all predictor variables (f1,
f2, f3,.., fn) has been calculated.
4. Calculate the likelihood for each class;
5. Get the greatest likelihood;
```

Figure 2.2: Pseudo code for Naive Bayes Algorithm

2.1.3 Decision Tree

It is a tree-like structure which helps in making decisions through a series of questions until an output is reached. It is a very popular method for non-parametric, supervised learning. The decision tree can be started with a set of questions and proceed further by answering the questions. Each answer can lead to another question. In this way decision tree helps to predict the final value. There are some important terminologies of the decision tree. The sample is represented by the core node at first, then it is partitioned into two or more homogenous sets. Next, splitting is the division of a node into two or more sub-nodes. Following that, a sub-node is regarded as a decision node when it divides into more sub-nodes. The leaf node doesn't divide after that. Finally, there is the process of pruning for removing sub-nodes from a decision node. Though the decision tree has a big disadvantage. Overfitting is a common problem with single decision tree models specially when there are too many nodes in a tree. One of the way to make up this issue which is setting a max depth of the tree. It can drastically lower the overfitting chance.[8]



```
GenDecTree(Sample S, Features F)
Steps:

1. Ifstopping_condition(S, F) = true then
    a. Leaf = createNode()
    b. leafLabel = classify(s)
    C. return leaf
2. root = createNode()
3. root.test_condition =findBestSpilt(S,F)
4. V= {v|v a possible outcomecfroot.test_condition}
5. For each value v € V:
    a. S={s root.test_condition(s) = v and s € S};
    b. Child = TreeGrowth (S,,,F);
    C. Add child as descent of root and label the edge
       {root- child} as v

6. return root
```

Figure 2.3: Pseudo code for Decision Tree Classifier

2.1.4 Gradient Boost

XGBoost is another powerful machine-learning algorithm which is based on Gradient Boost that is best for comprehending data and making wiser decisions. It is basically an application of gradient-boosting decision trees. The functional gradient approach selects a function that points in the direction of a negative gradient in order to minimize a loss function. [9] The gradient boosting classifier is used to combine numerous poor learning models into a potent predicting model. It has been used by researchers and data scientists from all around the world to improve their machine-learning models. A group of decision trees and gradient boosting are combined in the machine learning algorithm XGBoost to provide predictions. This algorithm is widely employed in data science and has excelled in a number of machine learning challenges. It improves on the Gradient Boosting framework by introducing some precise approximation methods and has improved forecasting power and performance.[13]

Chapter 3

Proposed Methodology

The implementations started with the datasets that we have to merge for preparation and then the steps of Machine Learning have been applied. At first, we set the batch size according to our data as we have a huge amount of data. Then, we took and described the datasets separately and merged them to make a new datatable. Next, we set data for preprocessing. Next, relatable and important features have been selected that actually we needed as we got a huge number of options. Also, the train-test process has been followed before featuring and label encoding. For building our main model, we used some algorithms and classifiers for two different types of attacks, one is Reflection based and the other one is Exploitation based. For exploitation based attacks, random forest classification was used which is a decision-tree based model and Naive Bayes classifier and SVM algorithm for testing the detection accuracy rate. Again, for reflection based attacks, we applied random forest classifiers and decision trees.

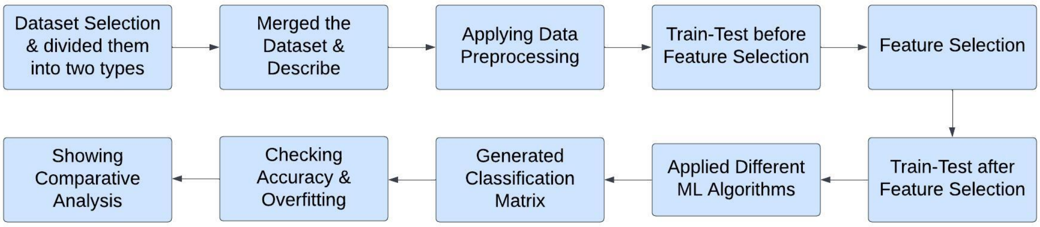

Dataset Selection & divided them into two types
Merged the Dataset & Describe
Applying Data Preprocessing
Train-Test before Feature Selection
Feature Selection
Train-Test after Feature Selection
Applied Different ML Algorithms
Generated Classification Matrix
Checking Accuracy & Overfitting
Showing Comparative Analysis

Figure 3.1: Workflow of our proposed methodology

3.1 Batch Size

Batch size refers to the amount used in one iteration in machine learning. It is one of the most important hyperparameters in machine learning for ensuring the best performance of a mode. It can increase the effectiveness of fast data processing, and decrease the requirement for human engagement. [11] Batch size is needed when there arelarge volumes of discrete data and multiple datasets that can be handled through batch processing. There are some possibilities for taking the batch size. In batch mode, the iteration and epoch values are similar since the batch size is equal to the complete dataset.[14] Typically, the batch size is set as a power of two, ranging from 16 to 512. Generally, the batch size is to be divisible by 8 and as large as feasible while still fitting on the GPU memory. Also, the range of batch size must be zero to infinity. More specifically, the batch size should be equal or more than one and less than or equal to the number of samples in a training dataset. As Tensorflow has been used, the batch size is 32 by default. But the batch size has to be taken 100000 as the work is on robust data and the range for one iteration starts with 100000. Some variables, including total training time, training time per epoch, and model quality are impacted by batch size.[4]

3.2 Data Preprocessing

Data preparation is the first step of machine learning model which is basically the process of turning unsorted raw data into a suitable form which is usable for any machine learningmodel. The categorization process or encoding data is the main part of data preprocessing so that a machine can read it easily. The fundamental need for a model to produce precise and accurate predictions is that the algorithm should be able to recognize the input's features in a short amount of time. Most real-world machine learning datasets are highly sensitive to missing, inconsistent, and noisy data because of their different origins. The use of data mining techniques would not give useful findings since they were unable to correctly identify patterns in this noisy data. For instance, duplicate or missing numbers can give a false impression of the data's overall statistics. That's why data preprocessing is essential for raising a wider level of data quality.[18]

3.2.1 Standardization

In order to create machine learning models that are precise and efficient, feature engineering is an essential step. Scaling, normalizing, and standardization, or the process of altering data to make it more appropriate for modeling, are the most common and basic components of feature engineering. These methods mainly help to increase model performance, lessen the effect of outliers, and ensure that the data is scaled equally.[8] The method of feature scaling includes converting the values of features or variables in a dataset to a comparable scale. We follow this step to make sure that each feature contributes equally to the model and to stop features with higher values from predominating it. When analyzing datasets containing features that have varying ranges, feature scaling is crucial. In our project, we follow Standardization. Standardization is followed because data must be scaled to a corresponding standard normal distribution. A distribution with a mean of 0 and a standard deviation of 1 is known as a standard normal distribution. After the mean has been removed, standardization is calculated by dividing by the standard deviation. [16] Normalization, which includes dividing a vector by its length, converts data into a range between 0 and 1. Standardization is a great tool to utilize when the data has a normal distribution. It can be used in a machine learning process when the distribution of the data is assumed. For our project, we chose to use Feature Importance for standardizing our dataset because our proposed model is based on Random Forest classifier which is a decision tree based classifier. Again, Random Forest and Extra Trees classifiers are basically used for this type of feature selection method.

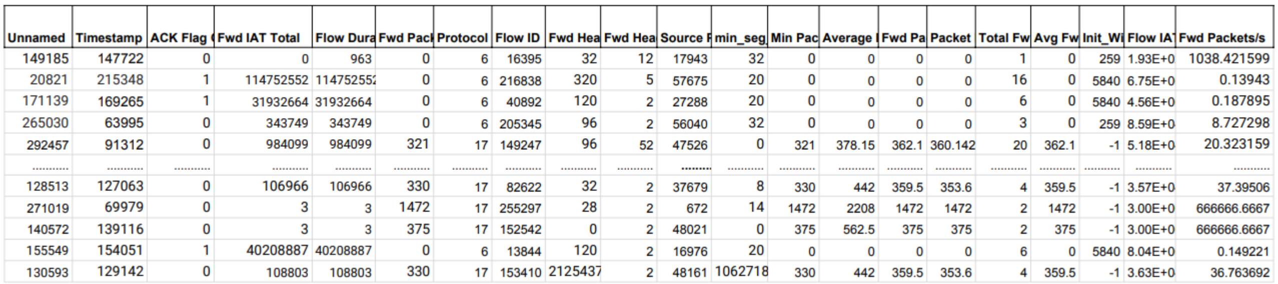
3.3 Feature Selection

Feature selection is a technique for lowering the input variable to a model because it is necessary to select only relevant data and eliminate distortion from the data. It is the procedure of automatically selecting relevant features for a machine learning model according to the kind of problem. To lower the computational expense of modeling and, in some situations, to boost model performance, it is preferable to minimize the number of input variables. It is possible to categorize supervised approaches into wrapper, filter, and intrinsic. There are two core forms of feature selection that are supervised and unsupervised. The correlation or dependence between input variables that can be filtered to identify the most pertinent features is scored using statistical measures in filter-based feature selection approaches. We chose Feature Importance which is under supervised filter methods for our project.

3.3.1 Analysis of Features

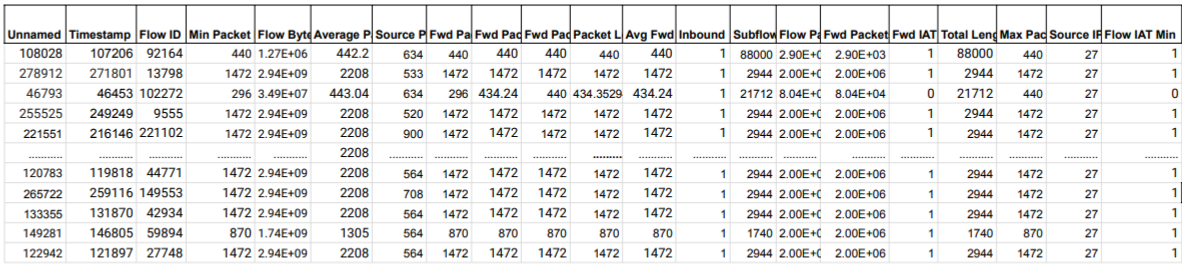
In machine learning, removing the target variable from data-X (i.e., the 'label' column in this case) is an usual approach. The dataset's other variables can be used to create predictions about the target variable, which is why we do this. The model will have access to the actual values that it is attempting to predict if the target variable was left in data-X, that could result in overfitting and erroneous predictions. So, the target variables' true values are effectively hidden from the model during training by removing it from data-X. Instead of just memorizing the proper values for each data point, this forces the model to discover patterns in the other variables that predict the target variable. As a result, our model may perform better in terms of generality and be more accurate when applied to fresh, unfiltered information.
By applying filter method, the best 20 features were selected for the exploitation type which are 'Timestamp', 'ACK-Flag-Count', 'Fwd IAT Total', 'Flow-Duration', 'Fwd Packet Length Min', 'Protocol', 'Flow ID', 'Fwd Header Length.1', 'Fwd Header Length', 'Source Port', 'min-seg-size-forward', 'Min Packet Length', 'Average Packet Size', 'Fwd-Packet-Length-Mean', 'Packet Length Mean', 'Total Fwd Packets', 'Avg Fwd Segment Size', 'Init-Win-bytes-forward', 'Flow IAT Mean', 'Fwd Packets/s'. Same as, we chose another 20 features for the Reflection type which are

‘Timestamp’, ‘Flow ID’, ‘Min Packet Length’, ‘Flow Bytes/s’, ‘Average Packet Size’, ‘Source Port’, ‘Fwd Packet Length Min’, ‘Fwd Packet Length Mean’, ‘Fwd Packet Length Max’, ‘Packet Length Mean’, ‘Avg Fwd Segment Size’, ‘Inbound’, ‘Subflow Fwd Bytes’, ‘Flow Packets/s’, ‘Fwd Packets/s’, ‘Fwd IAT Min’, ‘Total Length of Fwd Packets’, ‘Max Packet Length’, ‘Source IP’ and ‘Flow IAT Min’.



Unnamed	Timestamp	ACK Flag (	Fwd IAT Total	Flow Dura	Fwd Pack	Protocol	Flow ID	Fwd Hea	Fwd Hea	Source P	min_seg	Min Pac	Average I	Fwd Pa	Packet	Total Fw	Avg Fw	Init_Wi	Flow IAT	Fwd Packets/s
149185	147722	0	0	963	0	6	16395	32	12	17943	32	0	0	0	0	1	0	259	1.93E+0	1038.421599
20821	215348	1	114752552	11475255	0	6	216838	320	5	57675	20	0	0	0	0	16	0	5840	6.75E+0	0.13943
171139	169265	1	31932664	31932664	0	6	40892	120	2	27288	20	0	0	0	0	6	0	5840	4.56E+0	0.187895
265030	63995	0	343749	343749	0	6	205345	96	2	56040	32	0	0	0	0	3	0	259	8.59E+0	8.727298
292457	91312	0	984099	984099	321	17	149247	96	52	47526	0	321	378.15	362.1	360.142	20	362.1	-1	5.18E+0	20.323159
....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....
128513	127063	0	106966	106966	330	17	82622	32	2	37679	8	330	442	359.5	353.6	4	359.5	-1	3.57E+0	37.39506
271019	69979	0	3	3	1472	17	255297	28	2	672	14	1472	2208	1472	1472	2	1472	-1	3.00E+0	666666.6667
140572	139116	0	3	3	375	17	152542	0	2	48021	0	375	562.5	375	375	2	375	-1	3.00E+0	666666.6667
155549	154051	1	40208887	40208887	0	6	13844	120	2	16976	20	0	0	0	0	6	0	5840	8.04E+0	0.149221
130593	129142	0	108803	108803	330	17	153410	2125437	2	48161	1062718	330	442	359.5	353.6	4	359.5	-1	3.63E+0	36.763692

Figure 3.2: DataTable after Feature Selection



Unnamed	Timestamp	Flow ID	Min Packet	Flow Byte	Average P	Source P	Fwd Pa	Fwd Pac	Fwd Pac	Packet L	Avg Fwd	Inbound	Subflow	Flow Pa	Fwd Packet	Fwd IAT	Total Leng	Max Pac	Source IP	Flow IAT Min
108028	107206	92164	440	1.27E+06	442.2	634	440	440	440	440	440	1	88000	2.90E+0	2.90E+03	1	88000	440	27	1
278912	271801	13798	1472	2.94E+09	2208	533	1472	1472	1472	1472	1472	1	2944	2.00E+0	2.00E+06	1	2944	1472	27	1
46793	46453	102272	296	3.49E+07	443.04	634	296	434.24	440	434.3529	434.24	1	21712	8.04E+0	8.04E+04	0	21712	440	27	0
255525	249249	9555	1472	2.94E+09	2208	520	1472	1472	1472	1472	1472	1	2944	2.00E+0	2.00E+06	1	2944	1472	27	1
221551	216146	221102	1472	2.94E+09	2208	900	1472	1472	1472	1472	1472	1	2944	2.00E+0	2.00E+06	1	2944	1472	27	1
....	....	....	....	....	2208	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....
120783	119818	44771	1472	2.94E+09	2208	564	1472	1472	1472	1472	1472	1	2944	2.00E+0	2.00E+06	1	2944	1472	27	1
265722	259116	149553	1472	2.94E+09	2208	708	1472	1472	1472	1472	1472	1	2944	2.00E+0	2.00E+06	1	2944	1472	27	1
133355	131870	42934	1472	2.94E+09	2208	564	1472	1472	1472	1472	1472	1	2944	2.00E+0	2.00E+06	1	2944	1472	27	1
149281	146805	59894	870	1.74E+09	1305	564	870	870	870	870	870	1	1740	2.00E+0	2.00E+06	1	1740	870	27	1
122942	121897	27748	1472	2.94E+09	2208	564	1472	1472	1472	1472	1472	1	2944	2.00E+0	2.00E+06	1	2944	1472	27	1

Figure 3.3: DataTable after Feature Selection

So, in summary, dropping the target variable from data-X and selecting the best features is a standard practice in machine learning to prevent overfitting and improve the generalization performance of our model.

3.3.2 Model Description

Some Machine Learning algorithms have been followed and run for this project. The algorithms have been chosen according to the type of data. Also, it has been tried to minimize the number of models to take for our project. Most importantly, we have selected the models based on their efficiency, lowest runtime. Random Forest Algorithm, Naive Bayes Algorithm, Decision Tree Classifier, Gradient Boost Algorithm have been implemented for the two types of attacks.

3.3.3 Random Forest Classifier

As it has been studied by us that there are several advantages in using random forest algorithms and the most important ones is that it lowers the possibility of overfitting along with reduces the amount of training time. Again, gives decently high level of precision. By approximating missing data, the Random Forest algorithm efficiently can be run in big databases and generates extremely accurate predictions. tree-based models are comparatively effective from the linear models to outliers and also it does not require the normalization of the variables. At first, for the train-test segment, we divided the dataset into train and test data in a ratio of 60:40 instead of 80:20 because it was checked that if we take the ratio of 60:40 it gives a better outcome for balancing the accuracy. Random Forest classifiers have been used for both exploitation based and Reflection based attacks. Then, the models have been run through the train-test process both before and after feature selection. After featuring 20, the test-size was 0.40, and the random-state was 42. The number of trees in the forest was 100. We applied the algorithm on the best 20 features and also checked the overfitting by applying K-Fold cross validation.

3.3.4 Naive Bayes Classifier

Another machine learning method that works for categorization issues is the Naive Bayes algorithm. As it is a probabilistic classifier, it makes predictions based on the probability that a given event will occur. At first, Naive Bayes classifiers are applied on our selected features for exploitation based attacks and took the train-test ratio 60:40 instead of 80:20. Then, the overfitting has been checked by applying the K-fold cross validation process. Finally, both predictions result in the accuracy rate we got from our model.

3.3.5 Decision Tree Classifier

Supervised Machine Learning techniques like decision trees require continuously segmenting the data based on a particular parameter. In this project, another basic machine learning algorithms like Decision Tree has been implemented. The ratio of train-test parts was 60 and 40 respectively. Then, the confusion matrix have been shown for the model where we stored the accuracy rate. Also, the overfitting was checked by implementing K-Fold cross validation.

3.3.6 Gradient Boost Classifier

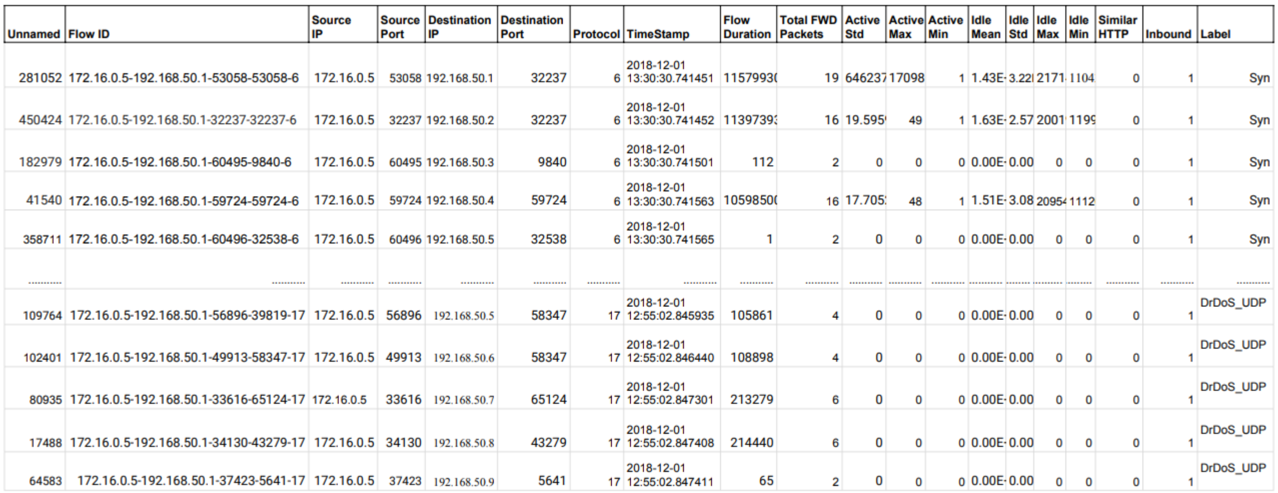
Gradient boosting is one type of boosting strategy which helps to build a very strong model by iterative learning from each of the weak learners. The algorithm may appear complex at first, but in the majority of cases, just one standard configuration has been utilized for classification and one for regression. To create predictions, we have to use a logistic function first to translate the log(odds) into a probability. So, we applied the algorithm on our project for Reflection based attacks. Here, we took the train-test ratio into 60:40 for balancing the train and test part.

Chapter 4

Dataset

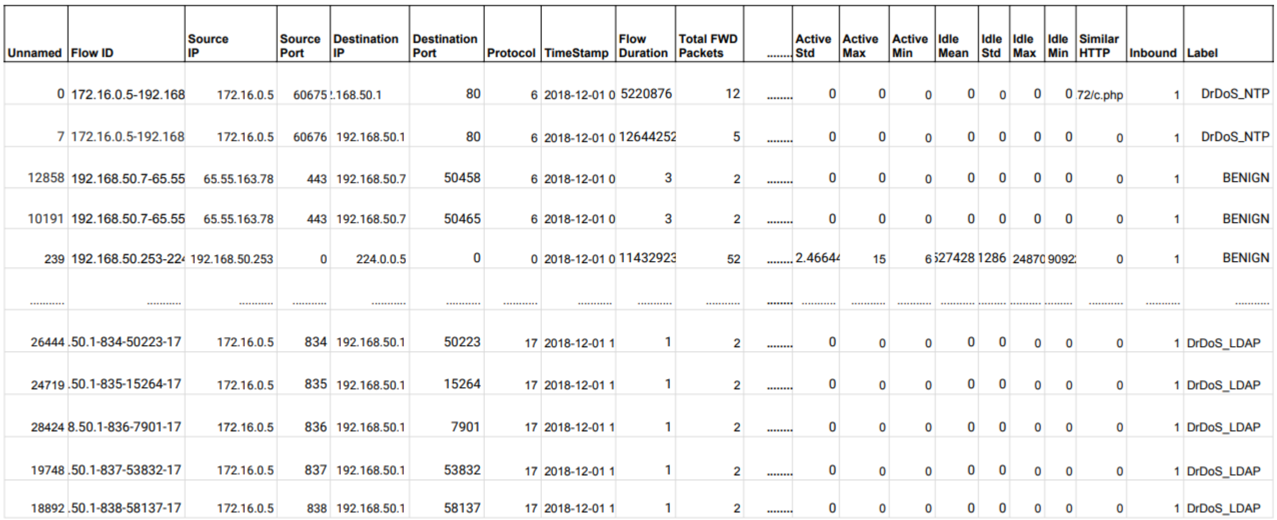
4.1 Data Description

For this research, multiple datasets have been taken and merged to build up a new dataset and worked on that. The main dataset was DDoS Evaluation Dataset (CIC-DDoS2019). Also, the benign and most recent common data for the attacks, which are closely similar to the actual real-world data (PCAPs), are contained in the main dataset (CICDoS2019). There are more than ten single datasets of various types of packets like SYN, UDP, UDPLag, ICMP, LDAP, NTP, DNS, SNMP, PortMap, NetBIOS, MSSQL in the main resource. But, we took six dataset of them, which are Syn.csv, UDPLag.csv, DrDoS-UDP.csv, DrDoS-NTP.csv, DrDoS-DNS.csv and DrDoS-LDAP.csv. [1] Then, to build a new taxonomy for detecting the attacks, we separated them under two types that are Exploitation based attacks and Reflection based attacks. There are a number of survey studies that have proposed taxonomies with respect to DDoS attacks. But our main target was to find new attacks and develop a new method that can differentiate the two types of attacks like Reflection based or Exploitation based. Hence, we have analyzed new attacks that can be carried out using HTTP and TCP/UDP based protocols at the application layer.[1]



Unnamed	Flow ID	Source IP	Source Port	Destination IP	Destination Port	Protocol	TimeStamp	Flow Duration	Total FWD Packets	Active Std	Active Max	Active Min	Idle Mean	Idle Std	Idle Max	Idle Min	Similar HTTP	Inbound	Label
281052	172.16.0.5-192.168.50.1-53058-53058-6	172.16.0.5	53058	192.168.50.1	32237	6	2018-12-01 13:30:30.741451	11579930	19	646237	17098	1	1.43E-	3.22	2171	1104	0	1	Syn
450424	172.16.0.5-192.168.50.1-32237-32237-6	172.16.0.5	32237	192.168.50.2	32237	6	2018-12-01 13:30:30.741452	11397393	16	19.595	49	1	1.63E-	2.57	2001	1199	0	1	Syn
182979	172.16.0.5-192.168.50.1-60495-9840-6	172.16.0.5	60495	192.168.50.3	9840	6	2018-12-01 13:30:30.741501	112	2	0	0	0	0.00E-	0.00	0	0	0	1	Syn
41540	172.16.0.5-192.168.50.1-59724-59724-6	172.16.0.5	59724	192.168.50.4	59724	6	2018-12-01 13:30:30.741563	10598500	16	17.705	48	1	1.51E-	3.08	20954	1112	0	1	Syn
358711	172.16.0.5-192.168.50.1-60496-32538-6	172.16.0.5	60496	192.168.50.5	32538	6	2018-12-01 13:30:30.741565	1	2	0	0	0	0.00E-	0.00	0	0	0	1	Syn
....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....
109764	172.16.0.5-192.168.50.1-56896-39819-17	172.16.0.5	56896	192.168.50.5	58347	17	2018-12-01 12:55:02.845935	105861	4	0	0	0	0.00E-	0.00	0	0	0	1	DrDoS_UDP
102401	172.16.0.5-192.168.50.1-49913-58347-17	172.16.0.5	49913	192.168.50.6	58347	17	2018-12-01 12:55:02.846440	108898	4	0	0	0	0.00E-	0.00	0	0	0	1	DrDoS_UDP
80935	172.16.0.5-192.168.50.1-33616-65124-17	172.16.0.5	33616	192.168.50.7	65124	17	2018-12-01 12:55:02.847301	213279	6	0	0	0	0.00E-	0.00	0	0	0	1	DrDoS_UDP
17488	172.16.0.5-192.168.50.1-34130-43279-17	172.16.0.5	34130	192.168.50.8	43279	17	2018-12-01 12:55:02.847408	214440	6	0	0	0	0.00E-	0.00	0	0	0	1	DrDoS_UDP
64583	172.16.0.5-192.168.50.1-37423-5641-17	172.16.0.5	37423	192.168.50.9	5641	17	2018-12-01 12:55:02.847411	65	2	0	0	0	0.00E-	0.00	0	0	0	1	DrDoS_UDP

Figure 4.1: Dataset Representation 1 (Exploitation Type)



Unnamed	Flow ID	Source IP	Source Port	Destination IP	Destination Port	Protocol	TimeStamp	Flow Duration	Total FWD Packets	....	Active Std	Active Max	Active Min	Idle Mean	Idle Std	Idle Max	Idle Min	Similar HTTP	Inbound	Label
0	172.16.0.5-192.168	172.16.0.5	60675	.168.50.1	80	6	2018-12-01 0	5220876	12	....	0	0	0	0	0	0	0	.72/c.php	1	DrDoS_NTP
7	172.16.0.5-192.168	172.16.0.5	60676	192.168.50.1	80	6	2018-12-01 0	12644252	5	....	0	0	0	0	0	0	0	0	1	DrDoS_NTP
12858	192.168.50.7-65.55	65.55.163.78	443	192.168.50.7	50458	6	2018-12-01 0	3	2	....	0	0	0	0	0	0	0	0	1	BENIGN
10191	192.168.50.7-65.55	65.55.163.78	443	192.168.50.7	50465	6	2018-12-01 0	3	2	....	0	0	0	0	0	0	0	0	1	BENIGN
239	192.168.50.253-224	192.168.50.253	0	224.0.0.5	0	0	2018-12-01 0	11432923	52	....	2.46644	15	6	527428	1286	24870	9092	0	1	BENIGN
....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....
26444	.50.1-834-50223-17	172.16.0.5	834	192.168.50.1	50223	17	2018-12-01 1	1	2	....	0	0	0	0	0	0	0	0	1	DrDoS_LDAP
24719	.50.1-835-15264-17	172.16.0.5	835	192.168.50.1	15264	17	2018-12-01 1	1	2	....	0	0	0	0	0	0	0	0	1	DrDoS_LDAP
28424	8.50.1-836-7901-17	172.16.0.5	836	192.168.50.1	7901	17	2018-12-01 1	1	2	....	0	0	0	0	0	0	0	0	1	DrDoS_LDAP
19748	.50.1-837-53832-17	172.16.0.5	837	192.168.50.1	53832	17	2018-12-01 1	1	2	....	0	0	0	0	0	0	0	0	1	DrDoS_LDAP
18892	.50.1-838-58137-17	172.16.0.5	838	192.168.50.1	58137	17	2018-12-01 1	1	2	....	0	0	0	0	0	0	0	0	1	DrDoS_LDAP

Figure 4.2: Dataset Representation 2 (Reflection Type)

Overall, the generated two final dataset showed four and three different categories of data respectively for Exploitation and Reflection type. The categories for type exploitation were DrDoS-UDP, Syn, UDP-lag and WebDDoS. On the other hand, the categories of data we got for type reflection were DrDoS-DNS, DrDoS-LDAP and DrDoS-NTP.

SYN

This happens when the standard TCP (three-way-handshake) requests are sent as a flood of SYN packets from a single system by the attacker. The main issue in the

TCP connection process, where a SYN request has to be sent to establish a TCP connection with a host and that must be matched with a SYN-ACK reply from that host. Mainly, the request has to be followed by the ACK response. That is how the request packets are exploited and cause a SYN flood.[7] In other words, the attack takes advantage of a known flaw in the TCP requests that requires a host to respond. The requester sends numerous SYN requests in a SYN flood situation, but either ignores the host's SYN-ACK response or sends the SYN queries from a fake IP address. In either case, the host system keeps waiting for a response to each request. This SYN attack is a common known Protocol type attack under Exploitation based.

UDP

The User Datagram Protocol (UDP) flood attack is another known DDoS attack. The idea of the attack is to saturate some specific ports on the selected computer. This directs the host to check for the application waiting on that port. When the hosts don't find any, they keep looking for the application to respond with an unreachable destination packet of ICMP. [16] The majority of operating systems restrict the ICMP response packet in part to prevent the flood that demands an ICMP reply. Any mitigation that takes place at the server level will not be sufficient enough if the UDP packets have a large amount of volume to cover the targeted server's firewall. UDP flood attack is also under the Exploitation based DDoS attack.

UDPLag

The idea of UDP-Lag is an attempt that is made to disconnect the link between client and server. This technique is mostly employed to defeat other players in online gaming by slowing down their movement. A hardware switch known as a lag switch, and also a network-running application that uses the bandwidth of other users are two ways to carry out this attack. UDP-Lag is counted as another Exploitation type attack in DDoS. [5]

DrDoS-NTP

In an NTP amplification attack, the attacker utilizes Network time Protocol (NTP) servers that are kept open to the public users to overload a targeted server with UDP traffic. This allows anyone who has the access to easily reach the server. This situation can execute a devastating high-bandwidth, high-volume DDoS attack. the attacker utilizes Network Time Protocol (NTP) servers that are open to the public to overload a targeted server with UDP traffic. This suggests that anyone who has access to a list of NTP servers that are reachable may easily execute a devastating high-bandwidth, high-volume DDoS attack. Because of this, any attacker who acquires a list of open NTP servers may frequently launch a damaging high-bandwidth, high-volume attack. So, NTP DDOS attack has been classified under reflection type.[7]

DrDoS-DNS

The Domain Name System (DNS) Internet protocol is exploited via the DNS Distributed Reflection Denial of Service (DrDoS) method. Hackers or malicious actors will spoof, or pretend to be, the primary target's IP address before sending application requests to a list of vulnerable DNS servers. Each DNS server is deceived into responding to the spoof IP address of the hacker's main target when it gets the forged request. Thus, the victim DNS servers will unintentionally bombard the main target with unwanted answers. So, DNS attack has been considered under reflection based attacks. [1]

DrDoS-LDAP

LDAP DDoS attack refers when a server that is openly accessible by the public users is infected by the attacker to send modest queries. That results in huge responses which are reflected to a target server. So, LDAP DDOS attacks have been classified under reflection based attacks. [1]

4.2 Data Analysis

Exploitation Type

At the beginning, the Syn, UDPLag and UDP packets are detected under Exploitation type. The raw data of 80 columns were sorted. Also, numerical data like int64 type and float64 type were found. By applying filter method, we got the best 20 features from our data that are 'Timestamp', 'ACK Flag Count', 'Fwd IAT Total', 'Flow Duration', 'Fwd Packet Length Min', 'Protocol', 'Flow ID', 'Fwd Header Length.1', 'Fwd Header Length', 'Source Port', 'min-seg-size-forward', 'Min Packet Length', 'Average Packet Size', 'Fwd Packet Length Mean', 'Packet Length Mean', 'Total Fwd Packets', 'Avg Fwd Segment Size', 'Init-Win-bytes-forward', 'Flow IAT

Mean', 'Fwd Packets/s' and the data-type is object.[1]

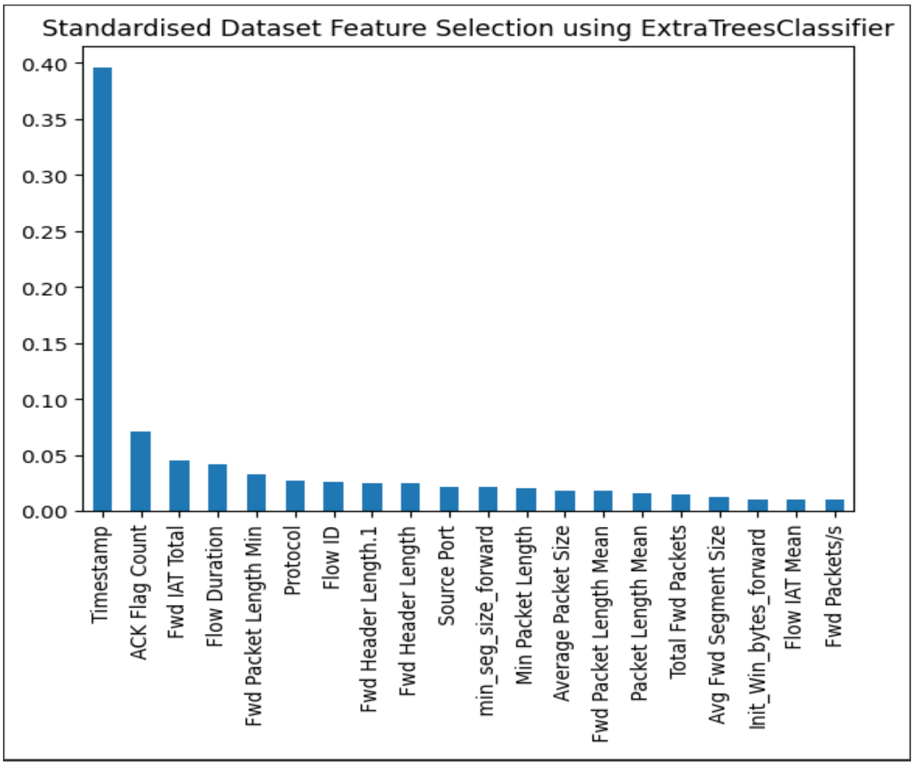


Figure 4.3: Selection of Best 20 Features (Exploitation Type)

Reflection Type

By doing the same process, the NTP, DNS and LDAP packets have been detected under Reflection type. We got the same attributes from these data. Applying data preprocessing, we got data of length 292600 and got the same data-type. Also, we dropped the eight column 'Label', 'SimillarHTTP', 'Frwd Packet Length Std', 'Bawd Packet Length Max', 'Bawd Packet Length Min', 'Bawd Packet Length Mean', 'Bawd Packet Length Std', 'Unnamed: 0' as well and got the rest 80 column. We got the same numerical data like int64 type and float64 type. After following feature selection, we selected the best 20 features from our data that are the same as well and the data-type is object.[1]

For both types, the top most five features have been analyzed which are Timestamp, ACK Flag Count, Flow ID and Flow Duration.
Timestamp: A timestamp comprises a series of characters or encoded data that signifies the date, time, and occasionally additional pertinent information about a specific event or moment. Its purpose is to document the occurrence time of an

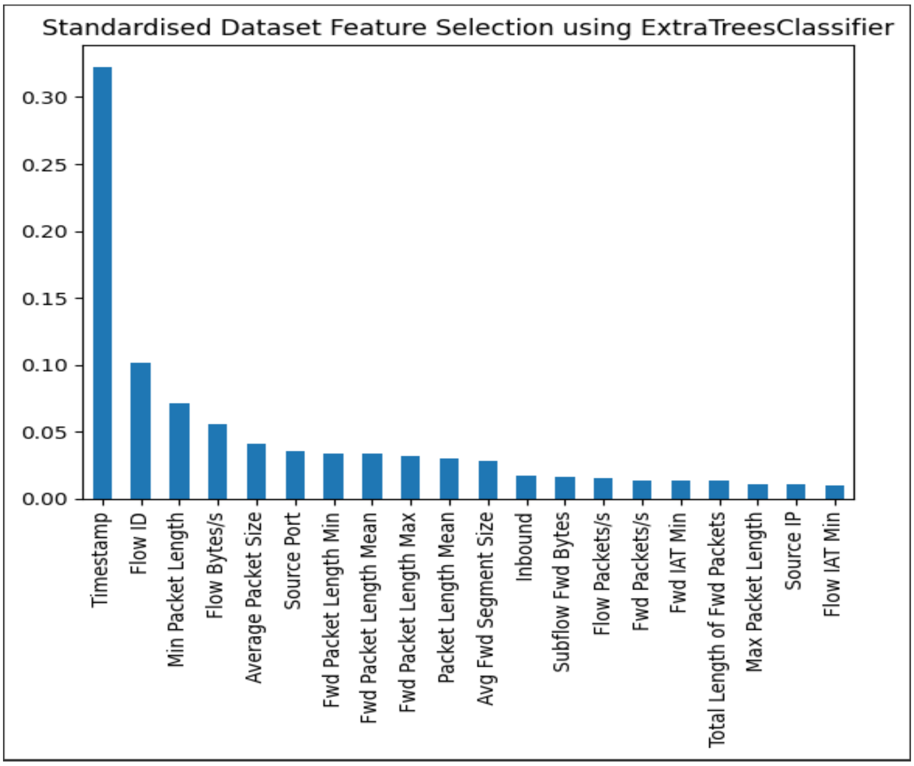


Figure 4.4: Selection of Best 20 Features (Reflection Type)

event, typically presented in a format easily comprehensible by both humans and computers.

ACK Flag Count: Once a connection has been made between the host and the client, information is sent in both directions using ACK or PUSH ACK packets until the session is terminated. An ACK flood victim server will accept fake ACK packets that are not associated with any of the sessions listed in the server's connection list.

Flow ID: The Flow IDS may work overtime to find malicious traffic in the event of a DDoS attack. IDSs that employ flow-based traffic representation group packets with similar properties together into a single record. The definition of the flows and the format features are determined by the targeted assaults used for detection. IPv4 addresses, port numbers, and the protocol being used are typically utilized to create the flows in IPv4 flow-based IDSs.

Forward Packet Length Mean: The 'Forward Packet Length Mean' denotes the average dimension of data packets transmitted in the forward direction within a network. This metric is employed in network analysis to grasp the usual size of data packets sent from the source to the destination.

Flow Duration: Flow Duration pertains to the duration, often quantified in seconds or an applicable unit, over which a sustained flow of data or activity takes place within a network or system. This metric offers valuable information about the time-related aspects of data transmission or process execution, aiding in pattern analysis, anomaly detection, and efficient resource allocation.

Chapter 5

Experimentation

The main workflow of our implementations is represented below:

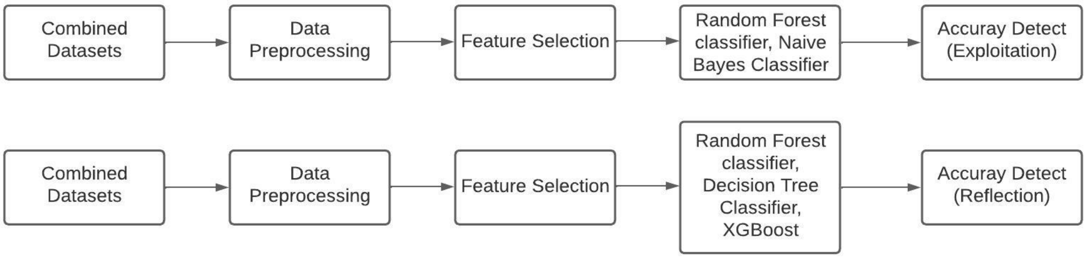

Combined Datasets
Data Preprocessing
Feature Selection
Random Forest classifier, Naive Bayes Classifier
Accuray Detect (Exploitation)
Combined Datasets
Data Preprocessing
Feature Selection
Random Forest classifier, Decision Tree Classifier, XGBoost
Accuray Detect (Reflection)

Figure 5.1: Workflow

To begin with, Google Colab has been used for the code implementation. The main dataset used for our research is the DDoS Evaluation Dataset (CIC-DDoS2019) which was divided into two types and there are three separate datasets under each type. The first three datasets are taken for working on exploitation based attacks which are Syn.csv, UDPLag.csv, DrDoS-UDP.csv. The other three datasets for reflection based attacks are NTP.csv, DrDoS-DNS.csv' and DrDoS-LDAP.csv At first, some necessary libraries were imported from scikit-learn. Then, the batch size has been set as 100000 and as the work has to be done with robust data. Colab doesn't permit users to run this huge data properly without setting the batch size because some runtime errors have been faced. Then, the datasets were described separately. After merging them, the new dataset was formed. After that, data preprocessing was applied. Then, feature selection was performed by applying the filter method and feature importance. Then, the train-test processes were implemented both before and after feature selection. After that, the most important part of our work was implementing the models. Random Forest classifier and Naive bayes classifier were the main priority for getting the required accuracy rate as they are probabilistic classifiers. For type reflection, we applied Random Forest classifier, Decision Tree classifier and Gradient Boost classifier to detect accuracy. Next, the classification report, confusion matrix were shown for each model used to visualize the data. Lastly, the K-fold cross validation was performed to check the overfitting issue.

The accuracy score of all the model and their training time that have been represented in the following table:

Attack Types	Algorithms	Testing Accuracy	Training Time
Exploitation Based	Random Forest	0.9964	27s
Exploitation Based	Naive Bayes	0.4853	0.57
Reflection Based	Random Forest	0.99748	27s
Reflection Based	Decision Tree	0.9919	54s
Reflection Based	XGBoost	0.9907	34s

Table 5.1: Comparison of Accuracy and Training Time among the four models (When Batch size 100000)

Attack Types	Algorithms	Testing Accuracy	Training Time
Exploitation Based	Random Forest	0.9989	37s
Exploitation Based	Naive Bayes	0.5843	0.54s
Reflection Based	Random Forest	0.9999	27s
Reflection Based	Decision Tree	0.9974	57s
Reflection Based	XGBoost	0.9964	34s

Table 5.2: Comparison of Accuracy and Training Time among the four models (When Batch size 200000)

So, according to the table it can be shown that we got the highest accuracy score from the random forest model and also the runtime is lower than any other model.

Chapter 6

Result Analysis

To begin with, the Machine Learning models such as Random Forest, Naive Bayes, Decision Tree and Gradient Boost have been proposed in this project. Then, we observed the existing papers and research work to compare the result with ours. We analyzed some of the other ML based approaches like Support Vector Machine(SVM), LSTM. We showed the accuracy in the confusion matrix and classification report for each algorithm. Also, the heatmap was represented for each model. Then, the Accuracy, Precision, Recall, and F1-score were generated as evaluation metrics in our implementation.

Accuracy: The information that compares the percentage of accurate predictions made by a model to all predictions made is known as the accuracy score in machine learning. By dividing the total number of predictions by the total number of accurate estimates, it can be calculated.

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)} \tag{6.1}$$

Precision: Precision is an important measure of performance for models since it indicates how well they forecast the future. The entire number of accurate positive predictions is divided by the total number of genuine positives to determine precision. [21]

$$Precision = \frac{TP}{(TP + FP)} \tag{6.2}$$

Recall: It is the percentage of all relevant models that were successfully retrieved. However, knowledge and the level of relevance are both related to precision and memory. [3]

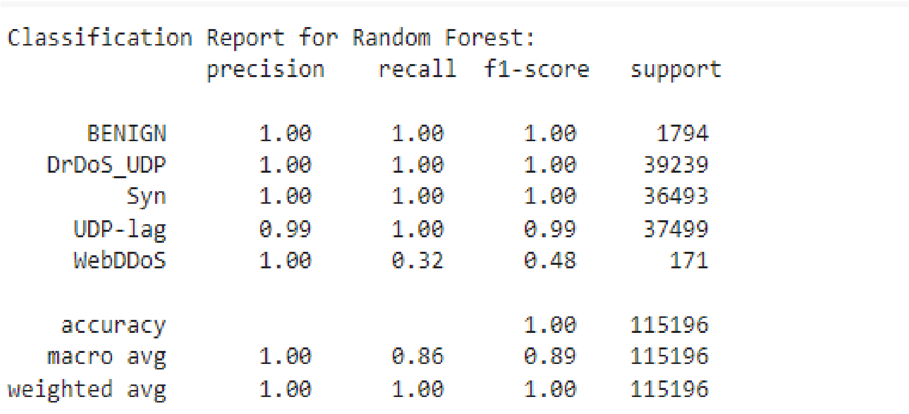
$$Recall = \frac{TP}{(TP + FN)} \tag{6.3}$$

F1-score: It is a classification report based on machine learning that evaluates a model's accuracy. It combines the recall and accuracy ratings of a model. The accuracy statistic measures how often an entire set of data was correctly predicted by a model. A system's accuracy and recall values are in concert averaged to provide an F-score. The F1-score increases as accuracy and recall increase. [3]

$$F1score = \frac{(2 * precision * recall)}{(precision + recall)} \tag{6.4}$$

6.1 Random Forest (Exploitation type)

In this proposed Random Forest model, the accuracy we got 99.96 percent or 0.9964, the precisions for DrDoS-UDP, Syn, UDP-lag, WebDDoS are 1.00, 1.00, 0.99 and 1.00 respectively. Also, the precision of Benign is 1.00.
The Classification report of the proposed model is given below :

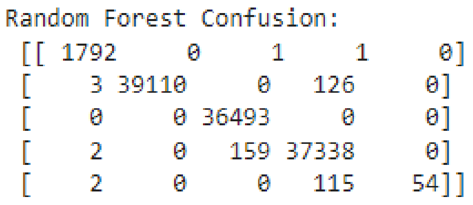


```
Classification Report for Random Forest:
              precision    recall  f1-score   support

      BENIGN       1.00      1.00      1.00      1794
   DrDoS_UDP       1.00      1.00      1.00     39239
         Syn       1.00      1.00      1.00     36493
     UDP-lag       0.99      1.00      0.99     37499
     WebDDoS       1.00      0.32      0.48       171

    accuracy                           1.00    115196
   macro avg       1.00      0.86      0.89    115196
weighted avg       1.00      1.00      1.00    115196
```

Figure 6.1: Classification report

Classification report is basically the performance evaluation metric in machine learning. The precision, recall, F1 Score, and support of our trained classification model are displayed in the figure using random forest algorithm. The accuracy of predictions made by the classification report shows how many of them comes true and how many comes false.[17]

The Confusion Matrix of the proposed model is given below :

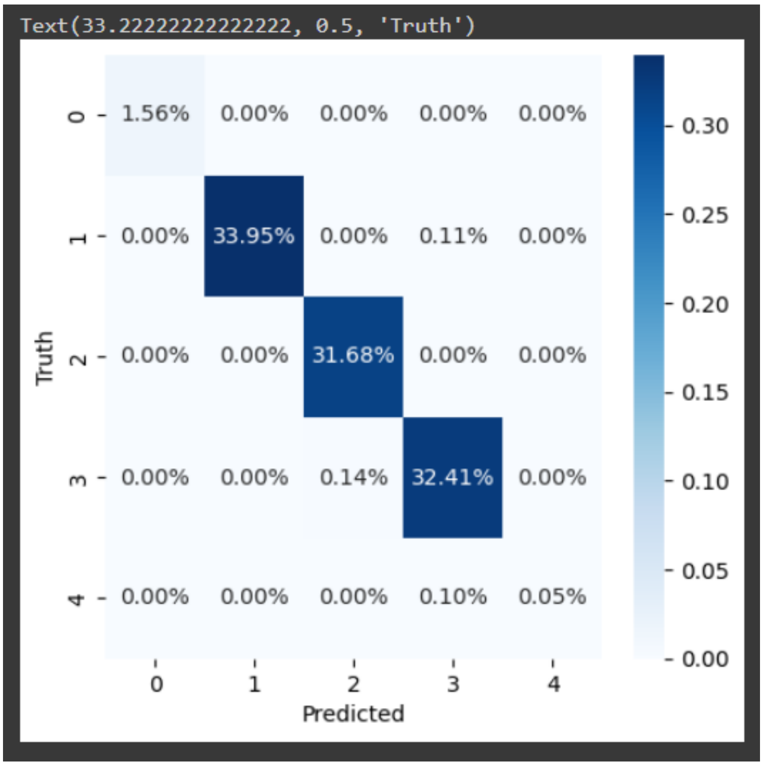


```
Random Forest Confusion:
 [[ 1792     0     1     1     0]
 [    3 39110     0   126     0]
 [    0     0 36493     0     0]
 [    2     0   159 37338     0]
 [    2     0     0   115    54]]
```

Figure 6.2: Confusion Matrix

Confusion matrix is basically shown as the prediction summary. It displays the number of accurate and wrong predictions made for each class and also helps to focus on the classes that models mistake for other classes.[20]

The Heat Map of the proposed model is given below :



```
Text(33.22222222222222, 0.5, 'Truth')
```

Figure 6.3: Heat Map

Heat Maps are basically the graphic representations of data that make use of color-coding schemes. Heat Maps focus on the part of data that matter the most by visualizing the volume of locations of a dataset. The color of each cell indicate the degree and direction of the association. Also, the stronger correlations are denoted by the darker colors.

The Classification Probabilities of the proposed model is given below :

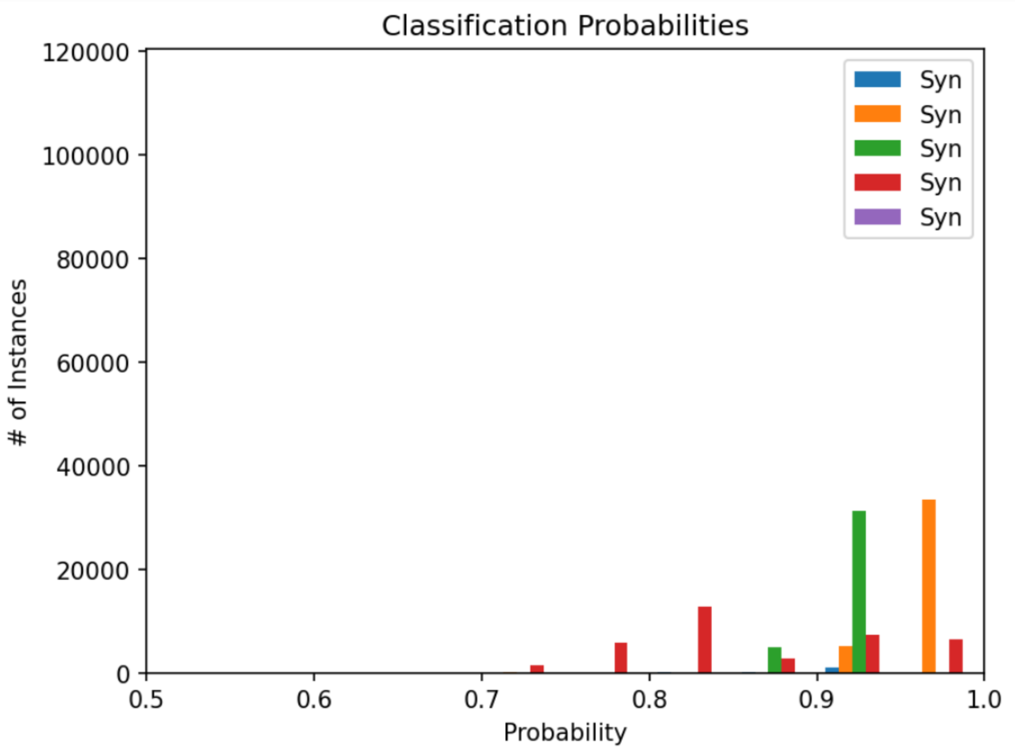


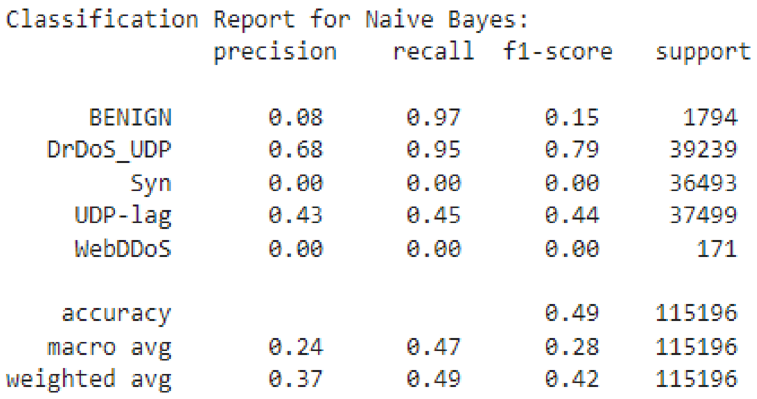
Figure 6.4: Classification Probabilities

The classification probabilities in machine learning predicts the probability distribution over based on the collection of classes given in the observation instead of producing the most likely class. As our research is based on classification, and probabilistic classifiers such as Random Forest, Naive Bayes, Decision Trees have been used, so the classification probabilities are also shown.

6.2 Naive Bayes (Exploitation type)

In our proposed Naive Bayes model, we got the accuracy 48.53DrDoS-UDP, Syn, UDP-lag, WebDDoS are 0.68, 0.00, 0.43 and 0.00 respectively. Also, the precision of Benign is 0.08.

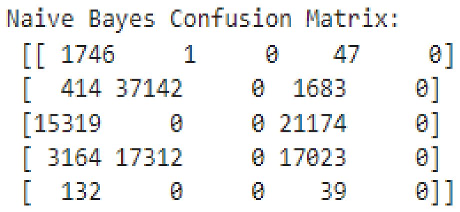
The Classification report of the proposed model is given below :



```
Classification Report for Naive Bayes:
                precision    recall  f1-score   support

      BENIGN       0.08      0.97      0.15      1794
   DrDoS_UDP       0.68      0.95      0.79     39239
         Syn       0.00      0.00      0.00     36493
     UDP-lag       0.43      0.45      0.44     37499
     WebDDoS       0.00      0.00      0.00       171

    accuracy                           0.49    115196
   macro avg       0.24      0.47      0.28    115196
weighted avg       0.37      0.49      0.42    115196
```

Figure 6.5: Classification report

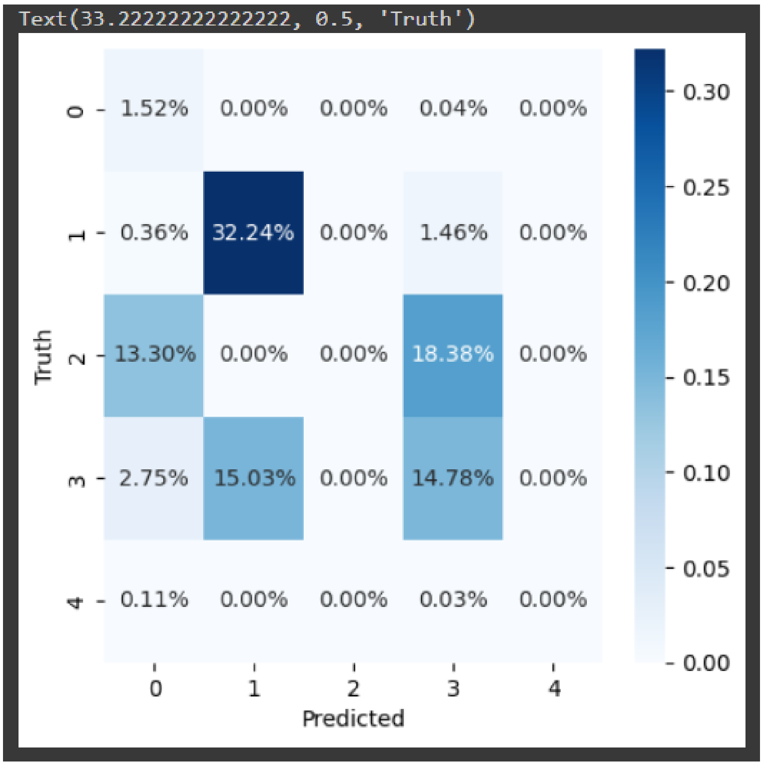
The Confusion Matrix of the proposed model is given below :



```
Naive Bayes Confusion Matrix:
 [[ 1746     1     0    47     0]
 [  414 37142     0  1683     0]
 [15319     0     0 21174     0]
 [ 3164 17312     0 17023     0]
 [  132     0     0    39     0]]
```

Figure 6.6: Confusion Matrix

The Heat Map of the proposed model is given below :



```
Text(33.22222222222222, 0.5, 'Truth')
```

Figure 6.7: Heat Map

The Classification Probabilities of the proposed model is given below :

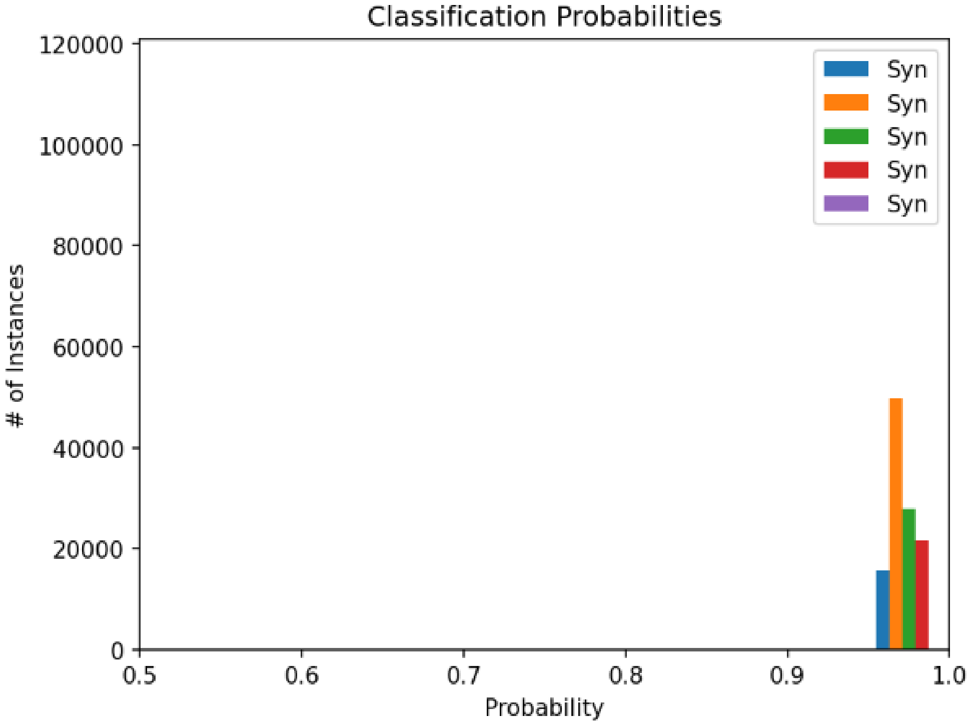


Figure 6.8: Classification Probabilities

The ROC Curve of the proposed model is given below :

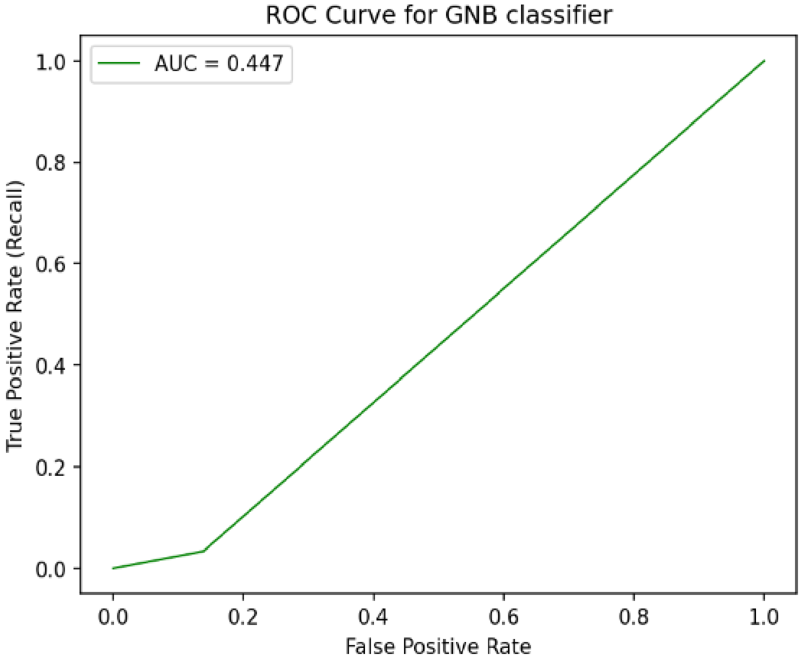
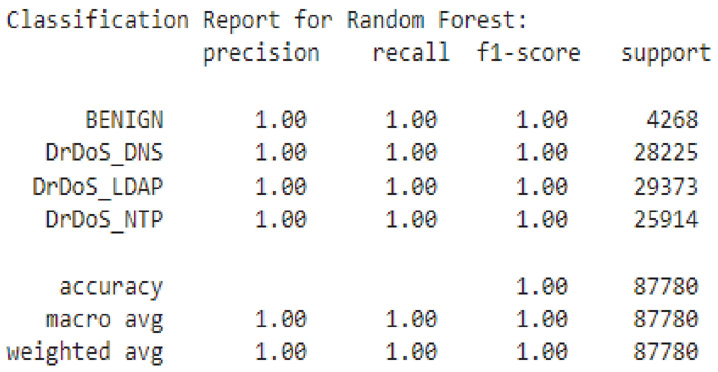


Figure 6.9: ROC Curve

6.3 Random Forest (Reflection type)

In our proposed Random Forest model, we got the accuracy 0.9974, the precisions for DrDoS-DNS, DrDoS-LDAP, DrDoS-NTP and Benign are 1.00, 1.00, 1.00 and 1.00 respectively.

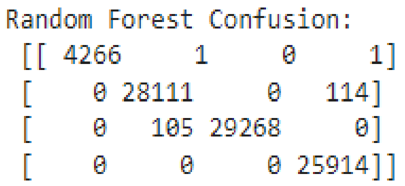
The Classification report of the proposed model is given below :



```
Classification Report for Random Forest:
              precision    recall  f1-score   support

      BENIGN       1.00      1.00      1.00      4268
   DrDoS_DNS       1.00      1.00      1.00     28225
  DrDoS_LDAP       1.00      1.00      1.00     29373
   DrDoS_NTP       1.00      1.00      1.00     25914

    accuracy                           1.00     87780
   macro avg       1.00      1.00      1.00     87780
weighted avg       1.00      1.00      1.00     87780
```

Figure 6.10: Classification report

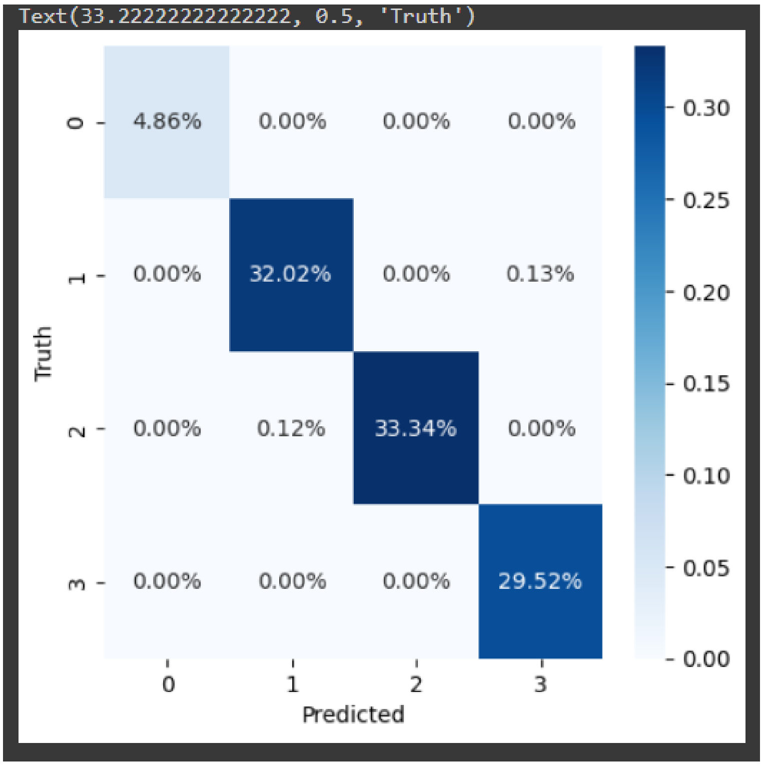
The Confusion Matrix of the proposed model is given below :



```
Random Forest Confusion:
 [[ 4266     1     0     1]
 [    0 28111     0   114]
 [    0   105 29268     0]
 [    0     0     0 25914]]
```

Figure 6.11: Confusion Matrix

The Heat Map of the proposed model is given below :

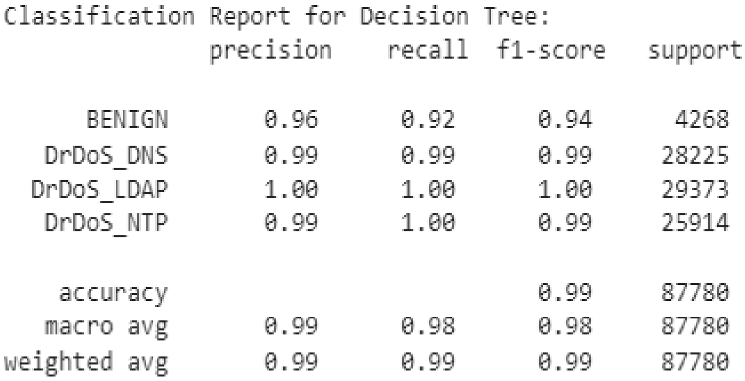


```
Text(33.22222222222222, 0.5, 'Truth')
```

Figure 6.12: Heat Map

6.4 Decision Tree (Reflection type)

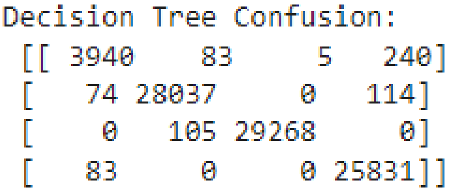
In our proposed Decision Tree based model, we got the accuracy 0.9919, the precisions for DrDoS-DNS, DrDoS-LDAP, DrDoS-NTP and Benign are 0.99, 1.00, 0.99 and 0.96 respectively.
The Classification report of the proposed model is given below :



```
Classification Report for Decision Tree:
              precision    recall  f1-score   support

      BENIGN       0.96      0.92      0.94      4268
   DrDoS_DNS       0.99      0.99      0.99     28225
  DrDoS_LDAP       1.00      1.00      1.00     29373
   DrDoS_NTP       0.99      1.00      0.99     25914

    accuracy                           0.99     87780
   macro avg       0.99      0.98      0.98     87780
weighted avg       0.99      0.99      0.99     87780
```

Figure 6.13: Classification report

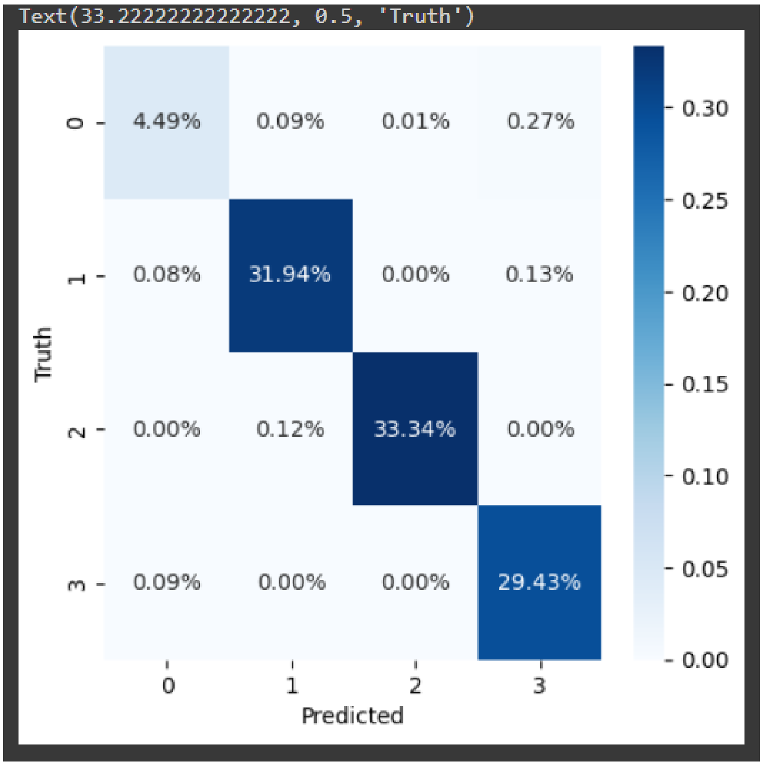
The Confusion Matrix of the proposed model is given below :



```
Decision Tree Confusion:
 [[ 3940    83     5   240]
 [   74 28037     0   114]
 [    0   105 29268     0]
 [   83     0     0 25831]]
```

Figure 6.14: Confusion Matrix

The Heat Map of the proposed model is given below :

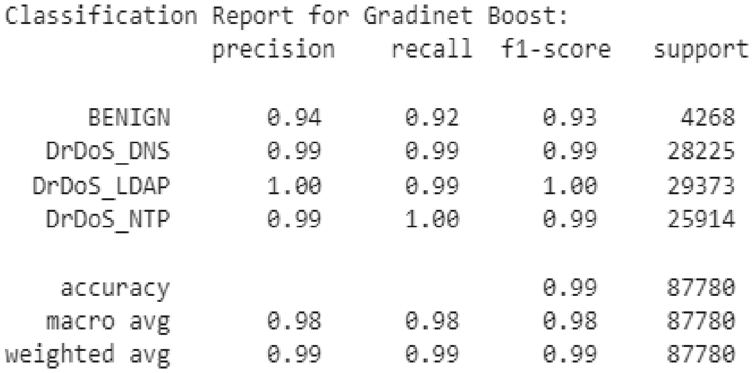


```
Text(33.22222222222222, 0.5, 'Truth')
```

Figure 6.15: Heat Map

6.5 Gradient Boost (Reflection type)

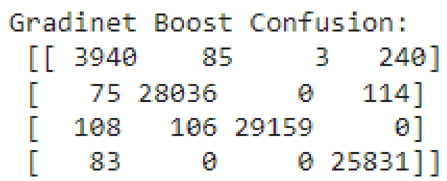
In our proposed Decision Tree based model, we got the accuracy 0.9919, the precisions for DrDoS-DNS, DrDoS-LDAP, DrDoS-NTP and Benign are 0.99, 1.00, 0.99 and 0.96 respectively.
The Classification report of the proposed model is given below :



```
Classification Report for Gradinet Boost:
              precision    recall  f1-score   support

      BENIGN       0.94      0.92      0.93      4268
  DrDoS_DNS       0.99      0.99      0.99     28225
 DrDoS_LDAP       1.00      0.99      1.00     29373
  DrDoS_NTP       0.99      1.00      0.99     25914

    accuracy                           0.99     87780
   macro avg       0.98      0.98      0.98     87780
weighted avg       0.99      0.99      0.99     87780
```

Figure 6.16: Classification report

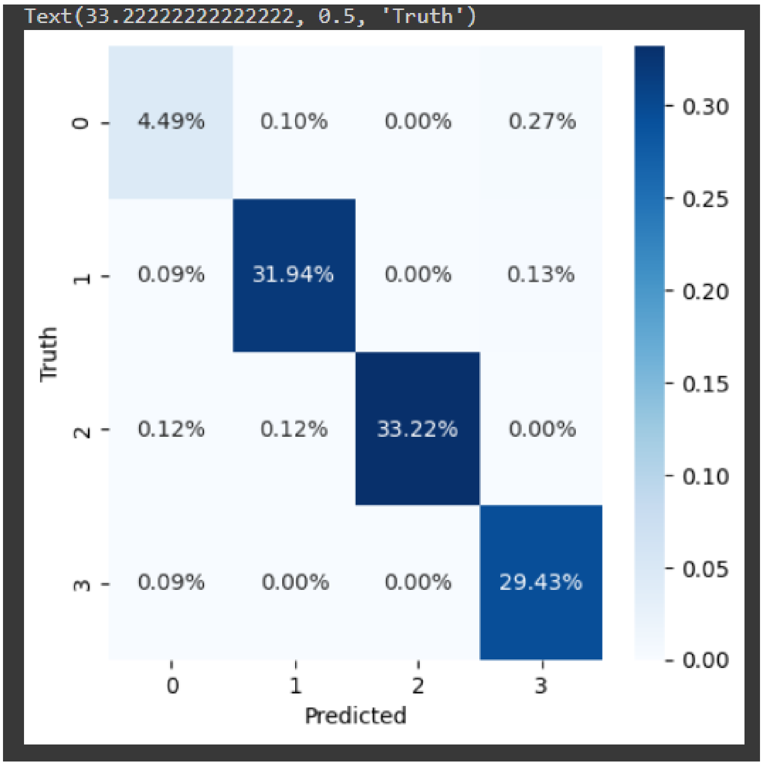
The Confusion Matrix of the proposed model is given below :



```
Gradinet Boost Confusion:
 [[ 3940    85     3   240]
 [   75 28036     0   114]
 [  108   106 29159     0]
 [   83     0     0 25831]]
```

Figure 6.17: Confusion Matrix

The Heat Map of the proposed model is given below :



```
Text(33.22222222222222, 0.5, 'Truth')
```

Figure 6.18: Heat Map

6.6 Explainable AI Features Implementation

Explainable AI is a collection of tools and frameworks that are natively linked with a number of Google products and services to assist you in comprehending and interpreting predictions provided by your machine learning models. With it, you may debug models, enhance their performance, and aid in the behavioral understanding of others. Explainable AI refers to the capacity to comprehend a model's output in terms of the features of the input data, the method employed, and the context in which the model was trained. It enables humans to evaluate and comprehend the outcomes produced by ML models. Explainable AI uses a variety of strategies to shed light on how AI systems make decisions. Utilizing model-independent techniques, such as feature importance analysis and rule extraction, is one strategy. In our project, the feature importance techniques have been implemented.

6.6.1 Type Exploitation

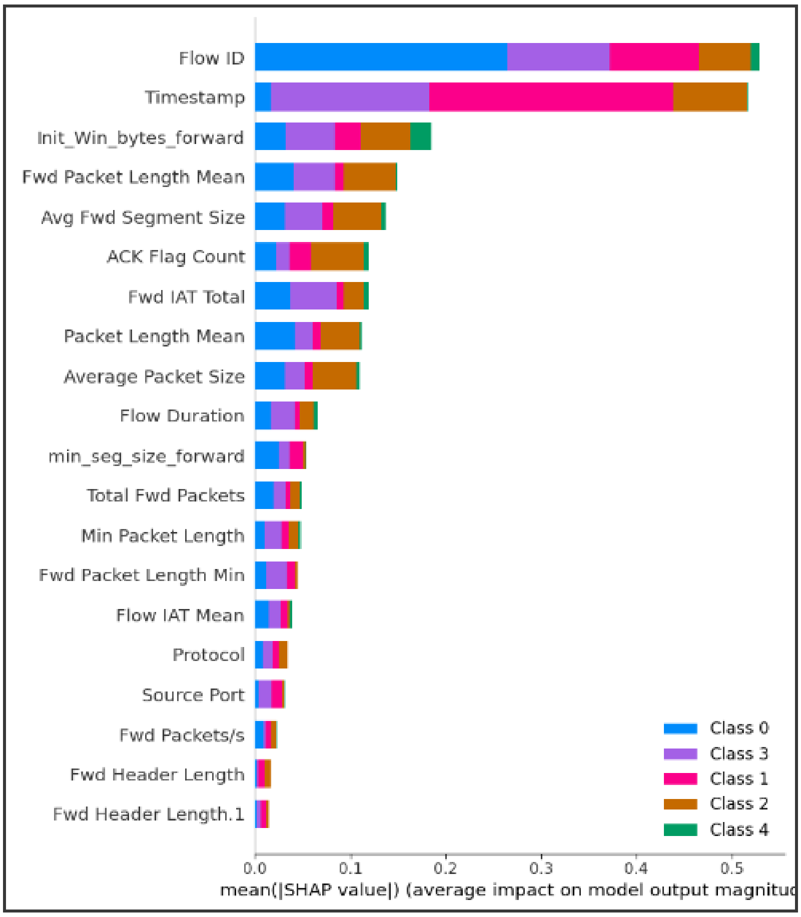


Figure 6.19: Explainable AI features after K-fold cross validation.

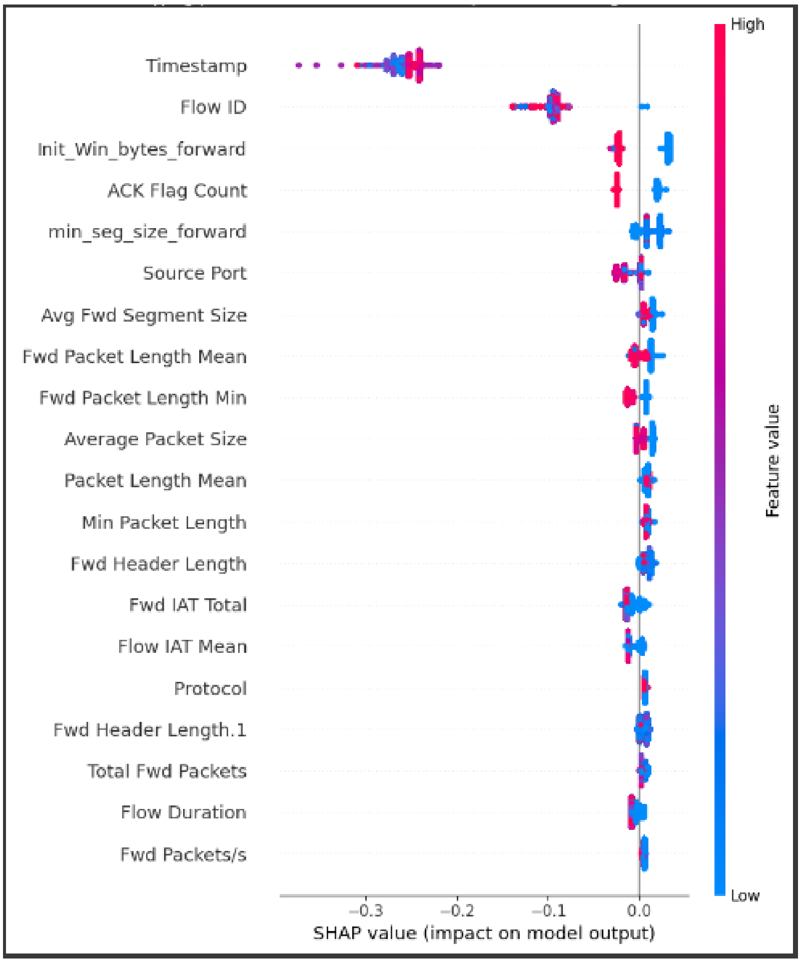


Figure 6.20: Explainable AI features after K-fold cross validation.

Here, the explainable features that we further get from K-fold cross validation, which are Timestamp, Init-Win-Bytes-Forward, Fwd Packet Length Mean, Ack Flag Count, Source Port under type type exploitation. So, here the features are quite different from the first selected features.

6.6.2 Type Reflection

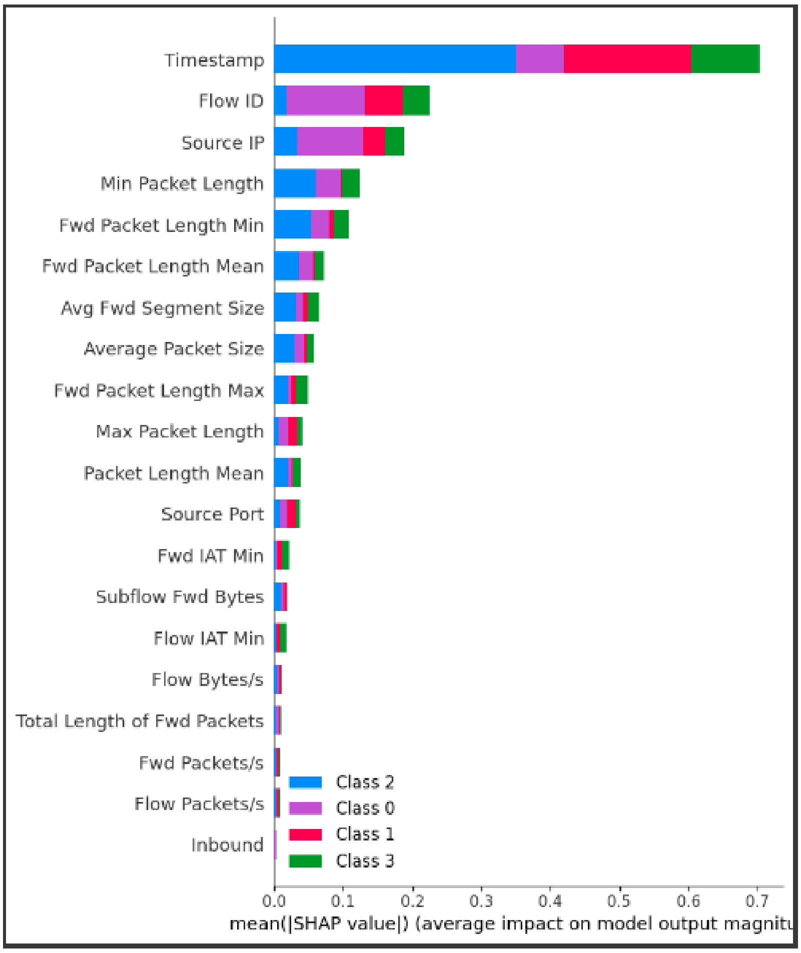
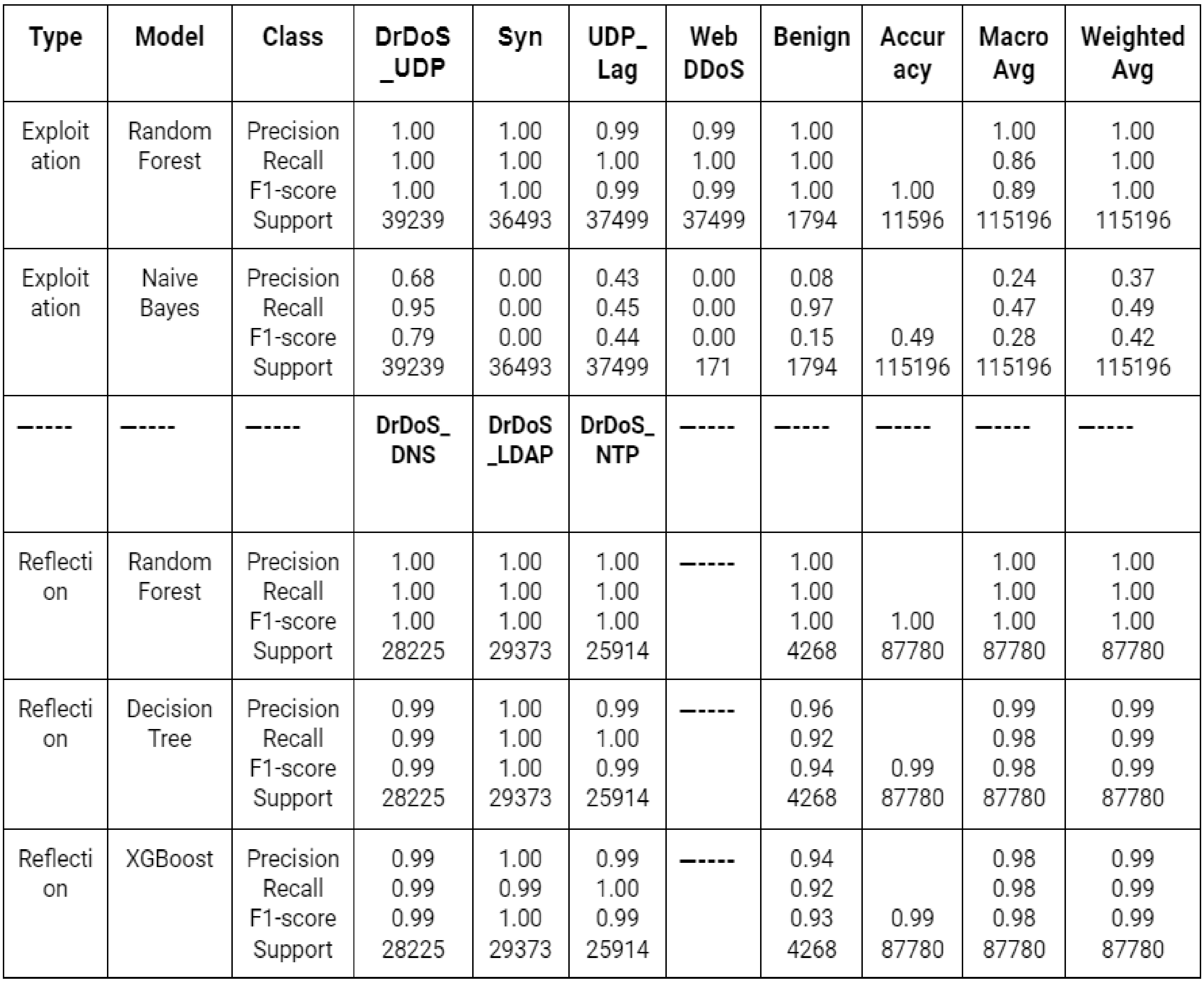


Figure 6.21: Explainable AI features after K-fold cross validation.

The features Timestamp, Flow ID, Source IP, Min Packet Length, Fwd Packet Length Min are the outcomes under type reflection. So the best 20 features have been selected which are same as the previous features.

6.7 Combined Analysis

The combined categorization report for all the models employed is presented in the table below. It includes accuracy, support values, precision, recall, the f1-score, the macro average, and the weighted average across all classes. We could deduce from this table that the Random Forest model for both type attack detection, produces the highest scores when compared to the others.



Type	Model	Class	DrDoS_UDP	Syn	UDP_Lag	Web DDoS	Benign	Accuracy	Macro Avg	Weighted Avg
Exploitation	Random Forest	Precision	1.00	1.00	0.99	0.99	1.00		1.00	1.00
		Recall	1.00	1.00	1.00	1.00	1.00		0.86	1.00
		F1-score	1.00	1.00	0.99	0.99	1.00	1.00	0.89	1.00
		Support	39239	36493	37499	37499	1794	11596	115196	115196
Exploitation	Naive Bayes	Precision	0.68	0.00	0.43	0.00	0.08		0.24	0.37
		Recall	0.95	0.00	0.45	0.00	0.97		0.47	0.49
		F1-score	0.79	0.00	0.44	0.00	0.15	0.49	0.28	0.42
		Support	39239	36493	37499	171	1794	115196	115196	115196
-----	-----	-----	DrDoS_DNS	DrDoS_LDAP	DrDoS_NTP	-----	-----	-----	-----	-----
Reflection	Random Forest	Precision	1.00	1.00	1.00	-----	1.00		1.00	1.00
		Recall	1.00	1.00	1.00		1.00		1.00	1.00
		F1-score	1.00	1.00	1.00		1.00	1.00	1.00	1.00
		Support	28225	29373	25914		4268	87780	87780	87780
Reflection	Decision Tree	Precision	0.99	1.00	0.99	-----	0.96		0.99	0.99
		Recall	0.99	1.00	1.00		0.92		0.98	0.99
		F1-score	0.99	1.00	0.99		0.94	0.99	0.98	0.99
		Support	28225	29373	25914		4268	87780	87780	87780
Reflection	XGBoost	Precision	0.99	1.00	0.99	-----	0.94		0.98	0.99
		Recall	0.99	0.99	1.00		0.92		0.98	0.99
		F1-score	0.99	1.00	0.99		0.93	0.99	0.98	0.99
		Support	28225	29373	25914		4268	87780	87780	87780

Figure 6.22: Combined Analysis of all Models.

6.8 Comparative Analysis

The accuracy score have been compared before and after applying K-Fold Cross validation.

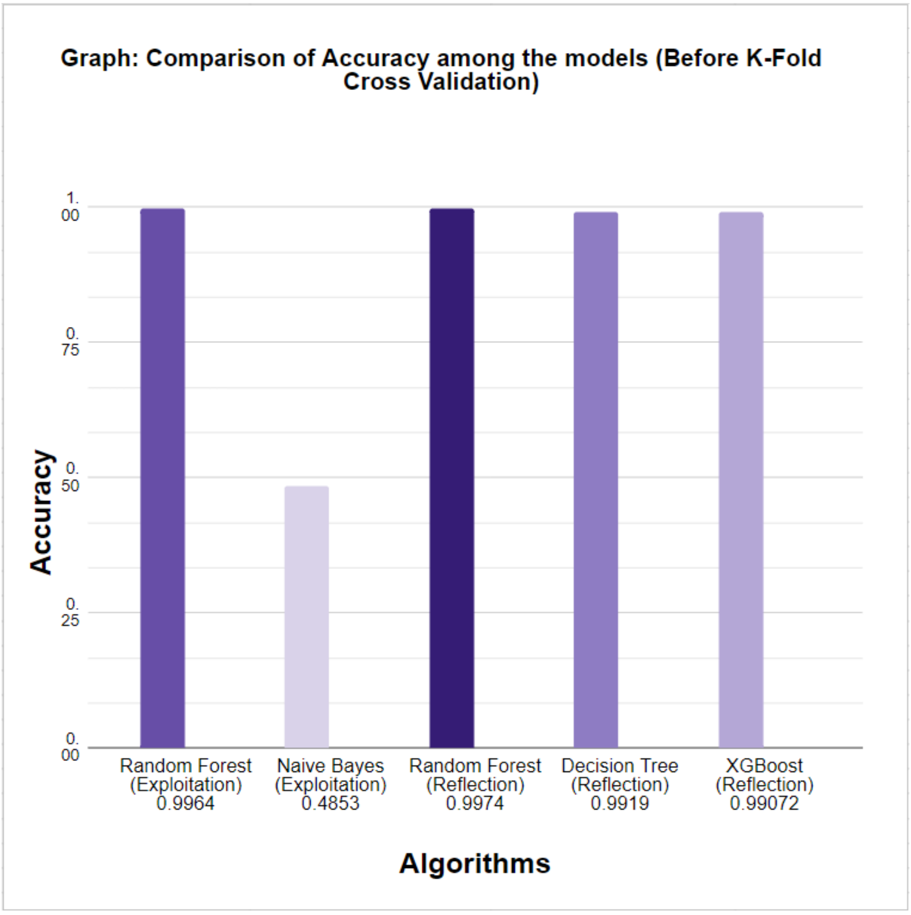


Figure 6.23: Accuracy before K-fold cross validation.

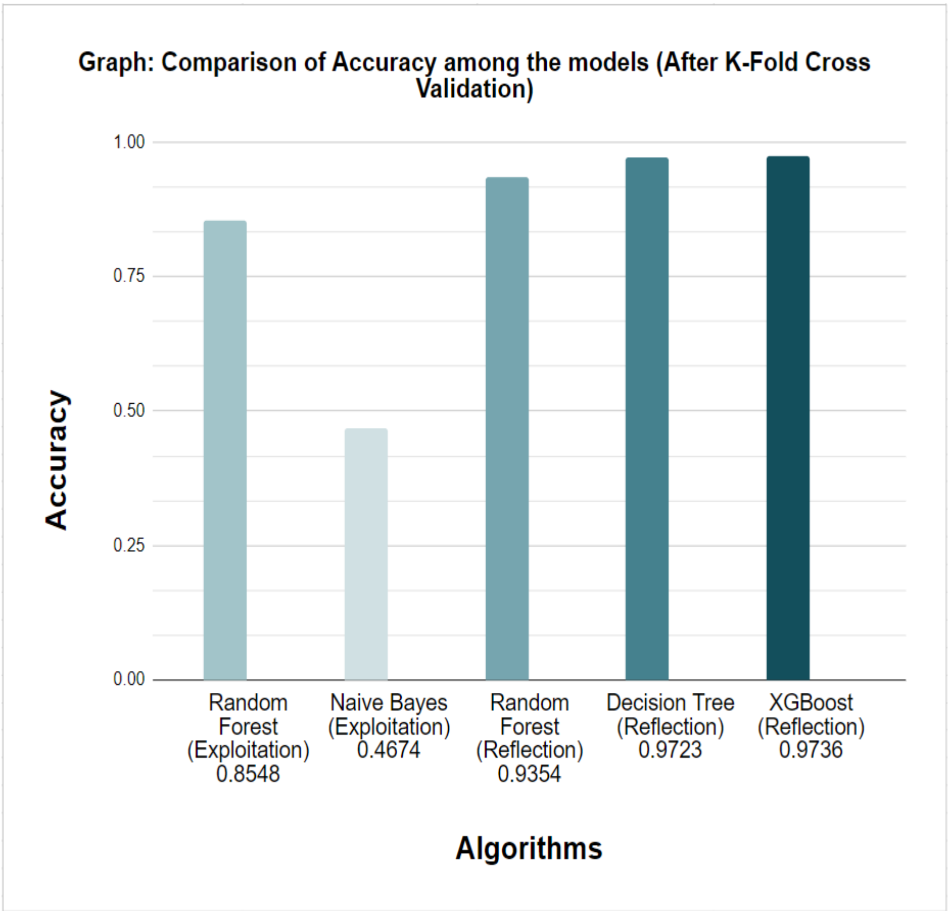


Figure 6.24: Accuracy after K-fold cross validation.

Here, it can be shown that, from the first model (Random Forest) applied in exploitation based attacks, a higher difference in the accuracy rate before and after k-fold cross validation have been noticed. The accuracy before k-folding was 0.9989, and after k-folding, it was 0.8548. So, it was further decided to choose multiple values of folding for k. We took 5, 10, 14, 15, 17, 21 as the folding values. Finally, the result came out that, by increasing the value of k, we can reduce the difference between the accuracy rate as well as the problem of overfitting.

Table 6.1: The Cross-Validation Scores and Number of CV Scores used in Average (Random Forest).

Averrage CV Scores	Number of CV Scores used in Average
0.7707	5
0.8549	10
0.9158	14
0.9359	17
0.941	21

According to the result, after applying K-Fold cross validation in the Random Forest model for type Exploitation, the Average Cross-Validation score is 0.8548 and the number of CV used in Average is 10. Again, for type Reflection, the Average CV score has come out 0.9354 in the Random Forest model. Here, the value of k was taken 10 as it is considered as an ideal value. But, from the first case of random forest model in type exploitation, it was noticed the difference of accuracy before and after cross validation which is a bit higher than the other cases. So, it was decided to take multiple folding values for splits and test the accuracy. So, five values have been taken for the folding values which are 5, 10, 14, 17 and 21. The rising of the accuracy rate regarding the CV scores are shown in the table.

Table 6.2: Comparison of Accuracy with other papers based on the same Datasets.

Authors	Year	Models	Accuracy
Arpit Kumar Jain[2]	2021	Naive Bayes Random Forest	99.353726 99.747371
Ruikui Ma[11]	2023	Random Forest	99.93
Devrim Akgun[6]	2022	CNN, LSTM	99.30
Chin-Shiuh Shieh[19]	2023	BI-LSTM, GMM	up to 94
Ours	2023	Random Forest	99.89, 99.748

At last, the most important part is the comparion of our testing accuracy with some other papers' work based on the same dataset. The better accuracy value getting from this work has been shown clearly in the table. It have been found that the proposed work have given the best result for detecting the attack with the highest accuracy.

6.9 Discussion

Firstly, the datasets have been chosen wisely. Secondly, the most important part was model selection for our experimentation. Also the models have been selected based on the data, by following the steps of machine learning. From this work, the results show that the Random Forest algorithm can give us the best accuracy. The obtained accuracy is 0.9989 for type exploitation, and accuracy 0.9974 for type reflection. Also, the overfitting has been checked in our implementation. For type Exploitation, after applying K-Fold cross validation in the Random Forest model, according to the result, the Average Cross-Validation score is 0.8548873224764749 and the number of CV used in Average is 10. Again, for type Reflection, the Average Cross-Validation score has come out 0.9354955570745045 in the Random Forest model. Here, first we have chosen the ideal value of k-folds as 10 is considered as an ideal value. Also, five values have been taken for the folding values. But, from the first case of random forest model in type exploitation, the difference of accuracy before and after cross validation was noticed which is a bit higher than the other cases. So, it was decided to take multiple folding values for splits and testing the accuracy. After that, a very important point came out that, if we the value of k is decreased, the accuracy got less, but if it is increased, the value of k, the accuracy got higher from the previous one. So, it can be said that there was no overfitting in our accuracy rate, solve the issue can be solved by increasing the folding value of k. Secondly, the other most important result of this work is the less amount of runtime compared to the existing works. As a total of six datasets have been selected and worked on by dividing into two types. Each type contains three of them. So that they can be run parallely and that gives a better training time. Also, some facts have been analyzed that, if the datasets were taken separately and run one by one, it would take a long time to run the datasets. From a previous existing work with the same dataset where the researchers took ten datasets together and also worked on the single one, the same processes were followed and got the run time for Random Forest algorithm was 1.7 minutes, where our proposed model gives a better runtime. The runtime for the same algorithm we have got is only 27 seconds. Also, a comparatively better accuracy has appeared which is the highest using the mentioned datasets. So, the primary goal of this project have been fulfilled.

Chapter 7

Conclusion

This paper emphasizes a distinct explanation of DDOS attack detection through ML strategie. As this attack is the most dangerous attack on the computer networking system, so here the work shows that how it can be detected because there are so many possible ways this attack can cause harm to multiple sources. The presented models acquired in this paper with the findings from six individual datasets. Machine learning probabilistic models have been tested to detect the DDoS attack from different sources in this work. However, the machine learning models are slightly more efficient than the mathematical models. But a real time simulation was not be presented. It can serve as a limitation of the models. Also a multithreading method can be used to decrease the runtime of detecting this attack. In future, this work will be extend with the implementation of multithreading method and also the focus will be on real-time detection with proposing new models.

Bibliography

[1] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, Funchal, Madeira, Portugal: SCITEPRESS - Science and Technology Publications, 2018.

[2] B. Arpit Kumar Jaina Himanshu Dhawan, "Ddos detection using machine learning ensemble," *Turkish Journal of Computer and Mathematics Education*, vol. 12, no. 12, pp. 1647–1655, 2021.

[3] Admin, *Precision: Definition, precision vs accuracy, recall, formula and example*, Jul. 2022. [Online]. Available: https://byjus.com/maths/precision/#:~:text=Precision%20is%20the%20amount%20of,pi%E2%80%9D%2C%20i.e%2C%203.142857143..

[4] J. Brownlee, *Difference between a batch and an epoch in a neural network*, Aug. 2022. [Online]. Available: https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/.

[5] K. B. Dasari and N. Devarakonda, "Tcp/udp-based exploitation ddos attacks detection using ai classification algorithms with common uncorrelated feature subset selected by pearson, spearman and kendall correlation methods," *Revue d'Intelligence Artificielle*, vol. 36, no. 1, pp. 61–71, 2022. DOI: 10.18280/ria.360107.

[6] U. C. Devrim Akgun Selman Hizal, "A new ddos attacks intrusion detection model based on deep learning for cybersecurity," *Elsevier*, vol. 118, no. 102748, 2022. DOI: 10.1016/j.cose.2022.102748.

[7] A. A. Bahashwan, M. Anbar, S. Manickam, T. A. Al-Amiedy, M. A. Aladaileh, and I. H. Hasbullah, "A systematic literature review on machine learning and deep learning approaches for detecting ddos attacks in software-defined networking," *Sensors*, vol. 23, no. 9, p. 4441, 2023. DOI: 10.3390/s23094441.

[8] A. Bhandari, *Feature engineering: Scaling, normalization, and standardization (updated 2023)*, Jul. 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/.

[9] guest$_b$*log*, *Introduction to xgboost algorithm in machine learning*, Mar. 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/.

[10] S. E. R, *Understand random forest algorithms with examples (updated 2023)*, Jul. 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/.

[11] X. C. Ruikui Ma and R. Zhai, "A ddos attack detection method based on natural selection of features and models," *Electronics*, vol. 12, no. 1059, pp. 1–32, 2023. DOI: 10.3390/electronics12041059.

[12] Simplilearn, *Random forest algorithm*, Feb. 2023. [Online]. Available: https://www.simplilearn.com/tutorials/machine-learning-tutorial/random-forest-algorithm.

[13] Simplilearn, *What is xgboost? an introduction to xgboost algorithm in machine learning: Simplilearn*, Jun. 2023. [Online]. Available: https://www.simplilearn.com/what-is-xgboost-algorithm-in-machine-learning-article.

[14] W. b. E. Zvornicanin, *Relation between learning rate and batch size*, Mar. 2023. [Online]. Available: https://www.baeldung.com/cs/learning-rate-batch-size.

[15] [Online]. Available: https://www.javatpoint.com/machine-learning-naive-bayes-classifier#:~:text=Na%C3%AFve%20Bayes%20Classifier%20is%20one, the%20probability%20of%20an%20object..

[16] [Online]. Available: https://www.kdnuggets.com/2020/04/data-transformation-standardization-normalization.html.

[17] J. Brownlee, *A Gentle Introduction to Probability Metrics for Imbalanced Classification - MachineLearningMastery.com — machinelearningmastery.com*, https://machinelearningmastery.com/probability-metrics-for-imbalanced-classification/, [Accessed 25-09-2023].

[18] *Data Preprocessing in Machine learning - Javatpoint — javatpoint.com*, https://www.javatpoint.com/data-preprocessing-machine-learning, [Accessed 25-09-2023].

[19] *Detection of Unknown DDoS Attacks with Deep Learning and Gaussian Mixture Model — mdpi.com*, https://www.mdpi.com/2076-3417/11/11/5213, [Accessed 25-09-2023].

[20] *ML — Matrix plots in Seaborn - GeeksforGeeks — geeksforgeeks.org*, https://www.geeksforgeeks.org/ml-matrix-plots-in-seaborn/, [Accessed 25-09-2023].

[21] *TCP/UDP-Based Exploitation DDoS Attacks Detection Using AI Classification Algorithms with Common Uncorrelated Feature Subset Selected by Pearson, Spearman and Kendall Correlation Methods — IIETA — iieta.org*, https://www.iieta.org/journals/ria/paper/10.18280/ria.360107, [Accessed 27-09-2023].