National College of Ireland

# Detecting Distributed Denial of Service attack using ensemble learning

MSc Research Project
Cybersecurity

Atharva Muttepawar
Student ID: x20182601

School of Computing
National College of Ireland

Supervisor: Dr. Rohit Verma

**National College of Ireland**

**MSc Project Submission Sheet**

**School of Computing**

**Student Name:** **Atharva Muttepawar**

**Student ID:** **X20182601**

**Programme:** **MSc in Cyber Security** **Year:** **2021-2022**

**Module:** **MSc Internship**

**Supervisor:** **Dr. Rohit Verma**

**Submission Due Date:** **16/12/2021**

**Project Title:** **Detecting Distributed Denial of Service attack using ensemble learning**

**Word Count: 4393** **Page count: 27**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

**Signature:** **Atharva Muttepawar**

**Date:** **16/12/2021**

**PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST**

| | |
|---|---|
| Attach a completed copy of this sheet to each project (including multiple copies) | □ |
| **Attach a Moodle submission receipt of the online project submission,** to each project (including multiple copies). | □ |
| **You must ensure that you retain a HARD COPY of the project**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. | □ |

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

| **Office Use Only** | |
|---|---|
| Signature: | |
| Date: | |
| Penalty Applied (if applicable): | |

# Detecting Distributed Denial of Service Attack using Ensemble Learning

Atharva Muttepawar
X20182601

**Abstract**

The Distributed Denial of Service (DDoS) attack is now one of the most used types of attack. As technology evolves, new tools and methods of attacks occur in the picture. As a result of this distributed denial of service attack, detection technologies should improve. This paper is created to show how advanced DDoS can be detected using a machine learning algorithm that can be operated on any hardware. We achieved better accuracy of DDoS attacks using these machine learning techniques. This paper will detect DDoS attacks accurately using four different algorithms and one ensemble technique using the staking method. This detector can detect User Datagram Protocol (UDP) flood, Internet Control Message Protocol (ICMP) flood, Transmission Control Protocol (TCP) flood, and many other forms of DDoS. Previous detectors could identify a limited number of DDoS types or required the usage of many features. Some of the detectors only function with specified procedures. With no predefined protocols, this detector will identify a wide range of DDoS types.

# 1 Introduction

Nowadays, most of the hardware is connected to the internet. The network can be considered various computers, routers, and servers connected. Network security has developed as a significant area of research in computer security. There are both permitted and prohibited users on the network. Not allowed users to access the network are considered hackers and steal or get critical information in a prohibited manner. Hackers usually use two methods which are the active and passive methods. In an active attack, the hacker directly interacts with the victim and can also modify the victim's resources to gain access, while in a passive attack hacker neither interact with the victim nor modifies the victim's resources (What is a DDoS attack?, 2021). The most common technique is to perform DDoS attack. Active attacks include DDoS attacks as shown in the figure 1 in which is occur due to the request and flooding packets form multiple devices or botnets. As a result, servers, systems, and networks fail. Because there are different types of xf-service attacks, identifying them becomes increasingly difficult. Denial of service attacks includes ICMP floods, Synchronize (SYN) floods, Internet Protocol (IP) packet floods, and others. It is needed to find a feasible solution to this problem. Confidentiality, integrity, and availability are essential safety pillars to consider.

Is ensemble machine learning algorithm can used to detect Distributed-Denial-of-Service Attacks accurately and precisely?

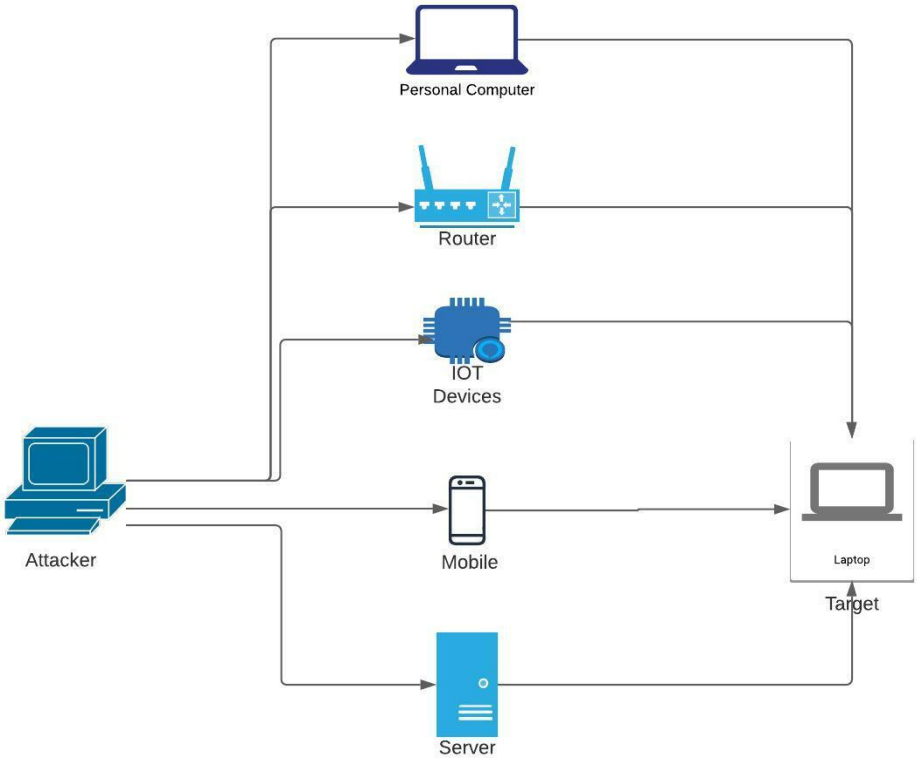



Figure 1: DDoS Attack

This paper uses the stacked ensemble learning technique for DDoS attacks using Naïve Bayes and gradient boosting algorithms. The recommended method will add up features of both naïve Bayes and gradient boosting classifiers. Gradient boosting consists of gradient descent and boosting. GBM includes risk modelling, regressing, resistance regression, K-class classification (Understanding Gradient Boosting Machines, 2021). The Bayes Theorem is used to describe a collection of classification algorithms known as the Naive Bayes. It's a group of algorithms that all have the same premise: every pair of features is classified as independent of one another. (Naive Bayes Classifier, 2021).

# 2 Ensemble Learning using Stacked Method

Ensemble learning is divided into three types bagging, stacking, boosting as shown in figure 2. The voting method is alternatively used for bagging and boosting. Similar models are chosen to predict the class of test in the bagging method. Firstly, similar chosen model's results are documented. In the last, the class assumed with a more significant number of models is given to the dataset. During the training phase, the models are intensively trained for misclassified data. Lastly, the model with the best accuracy will be chosen as a classifier for the dataset. Stacking ensemble learning is effective because of its common framework, which also helps to mix multiple ensemble methods. The stacked first step is base learning; base learners are trained with a training dataset, making a new dataset for meta learners. The second step is training the meta learner with the newly created dataset. Trained meta learning is used to classify the testing set. The central part is that stacking selects the best base learner (Chaudhry, Aniol and Shegos, 2020). So that rather than selecting a single base learner, it selects multiple base learners for the training data set. Meta learned classification is used as the final classifier; this is the difference between stacking and other ensemble learning technique.

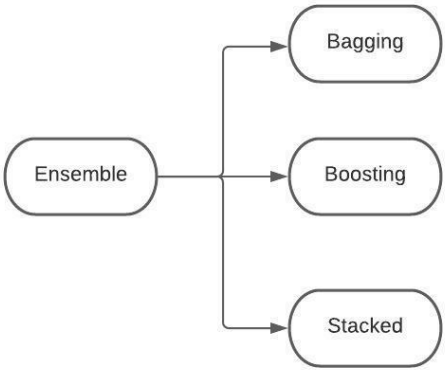



Figure 2: Types of ensemble learning

This paper has used gradient boosting and naïve Bayes as base learners. Gradient boosting is an upgraded and improved algorithm of AdaBoost. An predictive and additive model is used in the AdaBoost machine learning. Prediction is made using a step by step forward stagewise

manner. The upgraded version of AdaBoost is gradient boost, which instead of accumulating data points, introduces a new learner at each iteration on an existing weak learner (Understanding Gradient Boosting Machines, 2021). A probabilistic machine learning model called a Naive Bayes classifier is utilized to accomplish classification tasks. The Bayes theorem lies at the heart of the classifier. (Naive Bayes Classifier, 2021). Bayes Theorem is as shown below figure 3.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Figure 3: Bayes Theorem

Given that B has come up, we can calculate the similar as of A occurring using the Bayes theorem. In this case, B represents the proof, and A represents the hypothesis. In this situation, the features are supposed to be independent. The existence of one trait does not affect the present so that it is considered naïve (Naive Bayes Classifier, 2021).

# 3 Related Work

Journal DDoS attack evolution by Nazario, 2008, had briefly explained the Distributed Denial of Service attack and its types. Necessary bandwidth to happen DDoS attack and strategy evolution is described. Early-stage DDoS attack mitigation strategies were presented, which helped me move with machine learning techniques to detect the DDoS traffic flow and analyse packets.

From the following website DDoS Attack Types & Mitigation Methods | Imperva, 2021, I learned how DDoS attacks are differentiated as volume-based attacks, protocol attacks, application-layer attacks. Some widespread types of DDoS attacks like UDP flood, ICMP (ping) flood, SYN flood, slowloris, ping of death, HTTP flood, zero-day attacks. This site also explained the motivation behind the DDoS attacks like ideology, business feuds, boredom, extortion, cyber warfare.

For many years distributed denial of service attacks have been growing intensely because it was effective and can cause damage. In a previous paper, researcher Jelena Mirkovic has Presented two taxonomies for distributed denial of service attacks. The researcher proves that attack has two taxonomies for classifying attacks and mitigation techniques. Thus, to provide a better understanding to every researcher. The researcher tells how DDOS attacks start and how can we mitigate them using machine learning techniques (A taxonomy of DDoS attack and DDoS defence mechanisms, 2021).

Additionally, the researcher in this paper focuses on the DDoS problem and its attempts. As we focus on the form, it tells us that DDoS attacks can be mitigated using different techniques. Still, the audience should know when we can prevent, detect and respond to the

DDoS attack. We need to start using ensemble machine learning techniques (A Survey of Defence Mechanisms Against DDoS Flooding Attacks, 2021).

The different researchers have focused on mitigating DDoS attacks using machine learning in the past year. Here researchers show that if DDoS happens on network confidentiality, integrity, availability of network should not affect. The researcher uses a test theory showing a ping of death attack and mitigating it using machine learning techniques such as the Random Forest algorithm (Elizondo and Matthews, 2008).

So, the main aim of DDoS attacks is network conjunction is different tools. The researcher proved that a stacked ensemble learning model could be used in the wireless network for Detection purposes. Here researcher combines multiple machine learning techniques to detect the DDOs attack. The researcher illustrates an algorithm such as SVM, Random Forest, CART, ANN, and other machine learning methods. He states that DDOS attacks can be effectively and efficiently used to mitigate DDO attacks (Chaudhry, Aniol and Shegos, 2020).

In the previous paper, Lima Filho et al., 2019 implemented DDoS detection using machine learning techniques using random forest algorithm, Decision tree algorithm, Logistic regression, SDG, Adaboost. The used system can effectively achieve much better accuracy. I used a similar approach with an upgraded ensemble learning algorithm to achieve the best accuracy.

Prasad, V and Amarnath, 2019 also used machine learning techniques to detect DDoS attacks in the following paper. They compared the Stochastic gradient boosting algorithm predictions with other algorithms like KNN, decision tree, naïve Bayes, and random forest. They also explained the confusion matrix using the ROC plot and compared them.

In ‘A stacked ensemble learning model for intrusion detection in wireless network’, Rajadurai and Gandhi, 2020, used machine learning techniques in intrusion detection. They introduced stacked ensemble learning using random forest and gradient boosting algorithm as a base learner and combined their characteristics to get better output. This encouraged me to use the stacked ensemble learning technique to detect DDoS using Naïve Bayes and Gradient boosting as a base learner and get the best accuracy.

In this paper, Maglaris, 2021 implemented the Detection of DDoS attacks using a Multilayer Perceptron (MLP) classifier. They also explained how MLP could detect DDoS attacks and combine them with an Artificial Neural network (ANN). This gave me the idea to implement DDoS detection using MLP and compare the accuracy with my proposed model with the same data set.

# 4 Research Methodology

My prior research from the previous part aided me in performing and completing this technique and using this research style. I referenced and used the CIC DoS dataset (2016), CICDS (2017), and CSE-CIC-IDS (2018)-AWS datasets in my research (DDoS Dataset, 2021). I found 84 characteristics for both DDoS and benign packets in this dataset. After manually encoding the data, a random forest approach was used to determine the correlation between the characteristics and the most relevant ones. These were employed to locate more results after obtaining relevant and vital qualities. A brief description of the proposed model will follow.

## 4.1 Dataset selection

Typically, the attacker installs botnets, and the DDoS attack is carried out using them on numerous firms, corporations, and enterprises in order to disclose or conceal logs, any proof or pieces of evidence of the attack due to reputation, data security, and privacy. To carry out this study project, DDoS attack spoofing tools such as Hping 3, SDBot, and others were examined to create and learn a data set (DDoS Dataset, 2021).

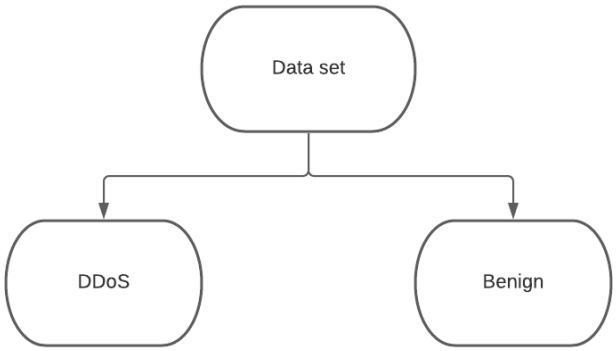



Figure 4: Unbalanced Dataset

Total number of record in data set is listed in below table:

| **Data Set** | |
|---|---|
| **DDoS** | **8356925** |
| **Benign** | **4437701** |
| **Total** | **12794626** |

Table 1: Unbalanced Data set

In the public domain latest DDoS dataset is found rarely. The used dataset was extracted DDoS packets from various IDS data sets, which are CIC DoS dataset (2016), CICDS (2017), CSE-CIC-IDS (2018)-AWS. Dataset is divided into two parts, i.e. DDOS packets and benign as shown in figure 4. The dataset has a total size of 6,79,47,44,782 bytes, or 6.32 GB. The data collection had a total of 1,27,94,626 data points, with DDoS and benign packets

accounting for 83,56,925 and 44,37,701 data points, respectively. Table 1 shows the unbalanced data set distribution. The following dataset contains 84 different features of a single packet for both DDoS and benign (DdoS Dataset, 2021).

## 4.2 Pre-processing of data

During the preceding phase, i.e., data set selection, we obtained a final dataset with proper DdoS and benign packets; however, the data was too large and unbalanced to execute. As a result, I needed to make it data-limited and balanced to work on it with the most incredible accuracy possible. However, certain DDoS and benign packets are chosen and concatenated together. So, for both DDoS and benign, I chose 99,999 data points each as shown in figure 5 and figure 6. At last total number of data points were 1,99,998 as shown in figure 7, which was manageable to execute. Table 2 and figure 8 shows the selected balanced data set distribution.

```
data_Dos = pd.read_csv('/content/drive/My Drive/DDOS_Attack/Data/final_dataset.csv',nrows=99999)
col = data_Dos.columns.tolist()
data_Dos.head(5)
```

| | Unnamed: 0 | Flow ID | Src IP | Src Port | Dst IP | Dst Port | Protocol | Timestamp | Flow Duration | Tot Fwd Pkts | Tot Bwd Pkts | TotLen Fwd Pkts | TotLen Bwd Pkts | Fwd Pkt Len Max | Fwd Pkt Len Min | Fwd Pkt Len Mean | Fwd Pkt Len Std | Bwd Pkt Len Max | Bwd Pkt Len Min | Bwd Pkt Len Mean | Bw Le |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 624 | 192.168.4.118-203.73.24.75-4504-80-6 | 192.168.4.118 | 4504 | 203.73.24.75 | 80 | 6 | 12/06/2010 08:34:32 AM | 3974862 | 29 | 44 | 86.0 | 59811.0 | 86.0 | 0.0 | 2.965517 | 15.969799 | 1460.0 | 0.0 | 1359.340909 | 372.0 |
| 1 | 625 | 192.168.4.118-203.73.24.75-4504-80-6 | 192.168.4.118 | 4504 | 203.73.24.75 | 80 | 6 | 12/06/2010 08:34:36 AM | 63 | 1 | 1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.000000 | 0.0 |
| 2 | 626 | 192.168.4.118-203.73.24.75-4505-80-6 | 192.168.4.118 | 4505 | 203.73.24.75 | 80 | 6 | 12/06/2010 08:34:36 AM | 476078 | 2 | 6 | 86.0 | 3037.0 | 86.0 | 0.0 | 43.000000 | 60.811183 | 1460.0 | 0.0 | 506.166667 | 740.2 |
| 3 | 627 | 192.168.4.118-203.73.24.75-4505-80-6 | 192.168.4.118 | 4505 | 203.73.24.75 | 80 | 6 | 12/06/2010 08:34:37 AM | 151 | 2 | 1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.000000 | 0.0 |
| 4 | 628 | 192.168.4.118-203.73.24.75-4506-80-6 | 192.168.4.118 | 4506 | 203.73.24.75 | 80 | 6 | 12/06/2010 08:34:37 AM | 472507 | 2 | 5 | 73.0 | 1050.0 | 73.0 | 0.0 | 36.500000 | 51.618795 | 1050.0 | 0.0 | 210.000000 | 469.5 |

5 rows × 85 columns

Figure 5: DDoS packets

```
data_Bening = pd.read_csv('/content/drive/My Drive/DDOS_Attack/Data/final_dataset.csv',skiprows=1284626,nrows=99999)
data_Bening.columns = col
data_Bening.head(5)
```

| | Unnamed: 0 | Flow ID | Src IP | Src Port | Dst IP | Dst Port | Protocol | Timestamp | Flow Duration | Tot Fwd Pkts | Tot Bwd Pkts | TotLen Fwd Pkts | TotLen Bwd Pkts | Fwd Pkt Len Max | Fwd Pkt Len Min | Fwd Pkt Len Mean | Fwd Pkt Len Std | Bwd Pkt Len Max | Bwd Pkt Len Min | Bwd F Len Me |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 4581673 | 172.31.67.126-123.223.160.41-3389-65229-6 | 123.223.160.41 | 65229 | 172.31.67.126 | 3389 | 6 | 20/02/2018 10:26:12 | 3287793 | 9 | 7 | 1148.0 | 1581.0 | 677.0 | 0.0 | 127.555556 | 218.691056 | 1173.0 | 0.0 | 225.8571 |
| 1 | 5948750 | 172.31.69.9-31.13.38.218-445-56272-6 | 31.13.38.218 | 56272 | 172.31.69.9 | 445 | 6 | 20/02/2018 01:13:33 | 431632 | 5 | 4 | 286.0 | 170.0 | 103.0 | 0.0 | 57.200000 | 52.485236 | 131.0 | 0.0 | 42.5000 |
| 2 | 2554297 | 172.31.0.2-172.31.64.7-53-52573-17 | 172.31.64.7 | 52573 | 172.31.0.2 | 53 | 17 | 20/02/2018 02:15:02 | 12906 | 1 | 1 | 46.0 | 131.0 | 46.0 | 46.0 | 46.000000 | 0.000000 | 131.0 | 131.0 | 131.0000 |
| 3 | 3690756 | 172.31.66.29-54.88.25.241-51513-443-6 | 172.31.66.29 | 51513 | 54.88.25.241 | 443 | 6 | 20/02/2018 11:40:22 | 113994432 | 8 | 5 | 864.0 | 1039.0 | 288.0 | 0.0 | 108.000000 | 149.054161 | 325.0 | 32.0 | 207.8000 |
| 4 | 5740622 | 172.31.0.2-172.31.69.23-53-51779-17 | 172.31.69.23 | 51779 | 172.31.0.2 | 53 | 17 | 20/02/2018 10:21:00 | 1788 | 1 | 1 | 43.0 | 99.0 | 43.0 | 43.0 | 43.000000 | 0.000000 | 99.0 | 99.0 | 99.0000 |

5 rows × 85 columns

Figure 6: Benign packets

```
[ ] df_final = pd.concat([data_Dos, data_Bening])
    df_final.head(5)
```

| | Unnamed: 0 | Flow ID | Src IP | Src Port | Dst IP | Dst Port | Protocol | Timestamp | Flow Duration | Tot Fwd Pkts | Tot Bwd Pkts | TotLen Fwd Pkts | TotLen Bwd Pkts | Fwd Pkt Len Max | Fwd Pkt Len Min | Fwd Pkt Len Mean | Fwd Pkt Len Std | Bwd Pkt Len Max | Bwd Pkt Len Min | Bwd Pkt Len Mean | Bw Le |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 624 | 192.168.4.118-203.73.24.75-4504-80-6 | 192.168.4.118 | 4504 | 203.73.24.75 | 80 | 6 | 12/06/2010 08:34:32 AM | 3974862 | 29 | 44 | 86.0 | 59811.0 | 86.0 | 0.0 | 2.965517 | 15.969799 | 1460.0 | 0.0 | 1359.340909 | 372.0 |
| 1 | 625 | 192.168.4.118-203.73.24.75-4504-80-6 | 192.168.4.118 | 4504 | 203.73.24.75 | 80 | 6 | 12/06/2010 08:34:36 AM | 63 | 1 | 1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.000000 | 0.0 |
| 2 | 626 | 192.168.4.118-203.73.24.75-4505-80-6 | 192.168.4.118 | 4505 | 203.73.24.75 | 80 | 6 | 12/06/2010 08:34:36 AM | 476078 | 2 | 6 | 86.0 | 3037.0 | 86.0 | 0.0 | 43.000000 | 60.811183 | 1460.0 | 0.0 | 506.166667 | 740.2 |
| 3 | 627 | 192.168.4.118-203.73.24.75-4505-80-6 | 192.168.4.118 | 4505 | 203.73.24.75 | 80 | 6 | 12/06/2010 08:34:37 AM | 151 | 2 | 1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.000000 | 0.0 |
| 4 | 628 | 192.168.4.118-203.73.24.75-4506-80-6 | 192.168.4.118 | 4506 | 203.73.24.75 | 80 | 6 | 12/06/2010 08:34:37 AM | 472507 | 2 | 5 | 73.0 | 1050.0 | 73.0 | 0.0 | 36.500000 | 51.618795 | 1050.0 | 0.0 | 210.000000 | 469.5 |

5 rows × 85 columns

Figure 7: Final Dataset

Total number of records in data set after selection is listed in below table:

| **Data Set** | |
|---|---|
| **DdoS** | 99999 |
| **Benign** | 99999 |
| **Total** | 199998 |

Table 2: Selected Balanced Data set

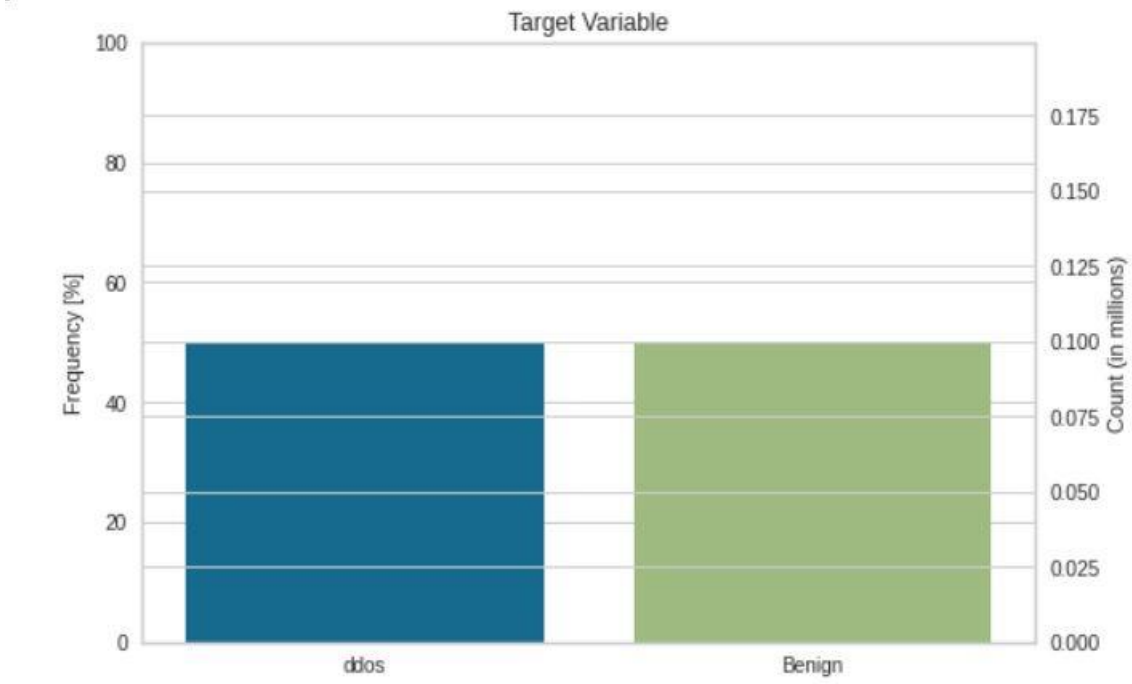



Figure 8: Balanced dataset

The accomplished data set is the final data set, and a CSV file with both DDoS and benign traffic has been created. None the less, the data set contained infinite values and NaN (not a number) values. Even if specific values are missing more than 50% and values that are missing less than 5% must be eliminated. Python code resolved this problem by removing the columns with such data types.

## 4.3 Feature selection

Random forest algorithms are one of the most well-known algorithms. Random forest algorithms are popular because they give better predictability, interpretability, reduced overfitting, and performance. The tree decision's interpretability is enhanced by the ease with which each variable's significance may be calculated. It is simple to calculate the contribution of each variable to the decision (Feature Selection Using Random Forest, 2021).

Random forest feature selection techniques come under the embedded methods category. These random forest or embedded methods have the qualities of wrapper and filter methods. Some built-in feature selection techniques are implemented by these algorithms. Embedded methods have benefits that are we can gain high accuracy, generalizes better, and is interpretable.

I started with this random forest technique and found specific 15 features among 84 which was used to gain better accuracy. In the future, I'll try using a hybrid method category for feature selection which may give some better features.

In embedded methods, one of the categories is feature selection using random forest. Combining wrapper methods and qualities of filter comes under embedded method. Algorithms with built-in feature selection techniques are used to implement them. Embedded approaches have several advantages some of them are:

- High accuracy
- Generalize better
- Interpretable

Around 400 to 1200 decision trees are present in random forests; each one comprises a random extraction of the dataset's observations and a random extraction of the features. Because not every tree sees all of the characteristics or data, the trees are de-correlated and hence less prone to over-fitting. Each tree additionally has a series of yes-no questions depending on a single or several attributes. The three separates the dataset into two buckets at every node, each containing observations that are more comparable to one another and dissimilar from those in another bucket (Feature Selection Using Random Forest, 2021). As a result, the value of each character is determined by how "pure" each bucket is.

Fifteen different features were selected by random forest algorithm for best accuracy are listed below:

1. feature Init Fwd Win Byts (0.173607)
2. feature Subflow Fwd Byts (0.112483)
3. feature Subflow Bwd Pkts (0.109908)
4. feature Src Port (0.103255)
5. feature Fwd URG Flags (0.044095)
6. feature Bwd URG Flags (0.039288)

7. feature Bwd Header Len (0.028766)
8. feature Src IP (0.026100)
9. feature Fwd Pkt Len Mean (0.024996)
10. feature Fwd Header Len (0.021836)
11. feature FIN Flag Cnt (0.021018)
12. feature Fwd Seg Size Avg (0.019901)
13. feature Pkt Len Std (0.019754)
14. feature Bwd Pkt Len Std (0.015620)
15. feature Dst IP (0.013747)
`

# 5 Design Specification

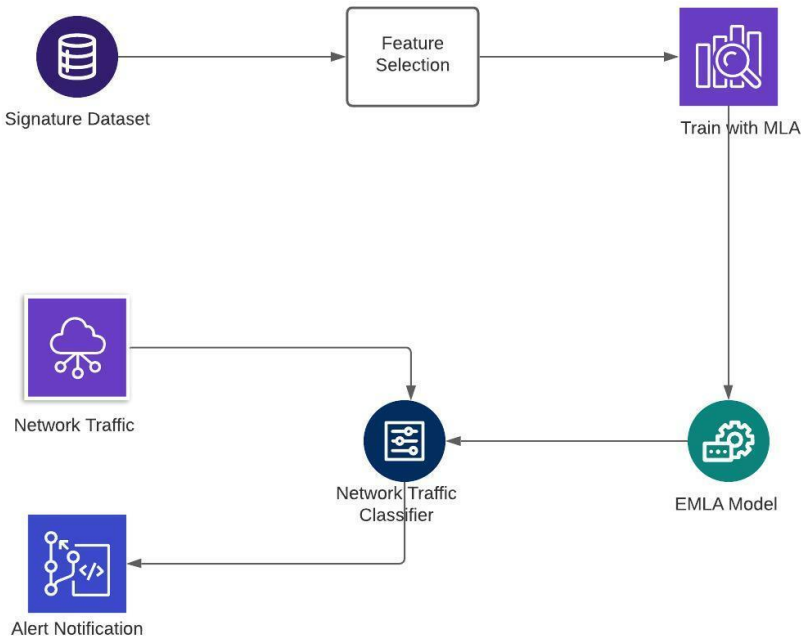



Figure 9: Model Overview

This section describes the model's operation and method. In this part, mechanisms for identifying data and traffic are thoroughly detailed based on their type. The research model may accept input in various formats; however, I experimented with using the CSV file for importing data, which is then utilized for pre-processing, feature selection, and feature selection by applying random forest. Selected features are then encoded on a different set of freshly formed data sets. This freshly created data set is utilized for training the model and testing the various classification algorithms.

The above figure 9 explains the complete structure of the research model. The selected data set has net traffic captured in a PCAP file and transformed into a CSV file. Before splitting the data set into sub-sets, it is subjected to feature extraction against the target variable because all attack groups have the same network traffic properties (Lima Filho et al., 2019). Encoded target features are used for the training model.

The multiple classifiers system, another name for the stacked ensemble learning, uses multiple sets of classifiers as base learners and builds or creates new training data set to classify unknown data. In below architecture diagram, it has explained basic flow diagram for the project. In the stacked ensemble learning method, firstly, it selects multiple base learners for example A1, A2, A3,….An, trains them using the training dataset, and makes same number of learners for example L1, L2, L3,….Ln. Output is made by combining these learners and creating a new dataset used as a input for this meta or second-level classifier (DDoS Dataset, 2021).

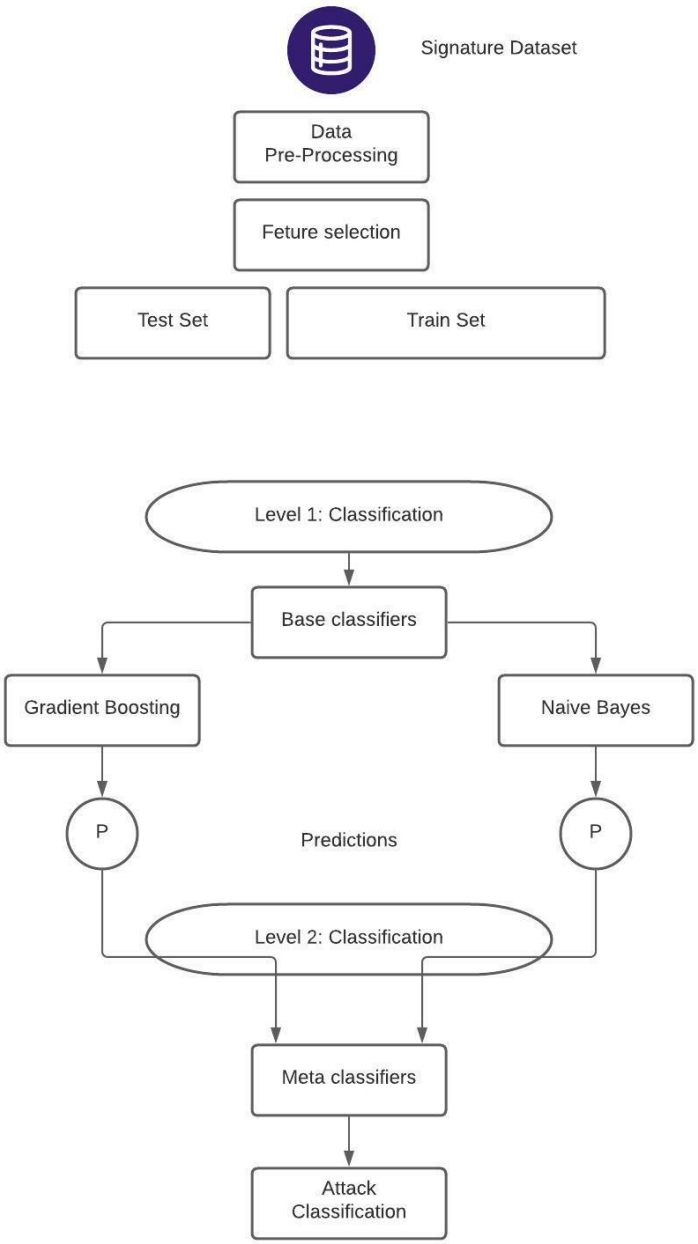



Figure 10: Architecture Diagram

There are two kinds of base learners, i.e., heterogeneous, and homogeneous. Base learners who are not under the same type are called heterogeneous otherwise homogeneous. The meta

or second-level classifier takes input from base classifiers and trains the input. At the same time, training process on the input data, the second level classifier look for the errors of base learners output and changes it to give the best output (Chaudhry, Aniol and Shegos, 2020). This identification and changes repeat multiple times to more accurate output. Figure 10 shows the basic architecture how meta classifier works.

In this research model accuracies of various algorithms are compared. Accuracies of Logical regression, MLP, Naïve Bayes, Gradient Boosting, and stacking ensemble methods are compared. Data is trained and tested using each algorithm, and accuracy is achieved.

# 6 Implementation

This section will explain steps taken to make our proposed model implemented. All the software and hardware utilized, and the coding framework is explained in depth.

## 6.1 Hardware

This model was built using a Lenovo laptop with the following specifications:

- CPU: Intel 10th Gen i5 Processor with 2.50 GHZ
- RAM: 32 Gb DDR4
- Storage: 512 Gb SSD
- GPU: NVidia GEFORCE GTX 4Gb

## 6.2 Software

This model was built on Windows 10 (64 bits) operating system. Even below software apps were also used:

- Google Drive to store data set
- Google Colab is used as a development environment
- Python 3 is used to code this model
- Libraries – pandas, NumPy, pickle, sklearn, GaussianNB, LogisticRegression, AdaBoostClassifier, RandomForestClassifier, DecisionTreeClassifier, plt, preprocessing, roc_curve, accuracy_score, confusion_matrix, roc_auc_score, GradientBoostingClassifier.

## 6.3 Data files

Basically, two files were used to implement this project:

- final_dataset.csv: This is a data set file which was imported to Google Colab to use while implementation.
- project.ipynb: This is a main file which includes

# 7 Evaluation

To find the best accuracy of the proposed model, I have performed tests using various algorithms which will use DDoS and benign data set. Results are documented and compared below.

## 7.1 Experiment using Logistic Regression algorithm

The data set was trained and tested using Logistic Regression algorithm to verify the accuracy. 1,99,998 data points were used, and it was balanced data set. In this data set there were both DDoS and benign packets. Using Logistic Regression, we achieved accuracy of 81.53% as output in shown in figure 11, and with precision of 83.90%.

```
Logistic Regression:Accuracy :  81.53624097897709
```

Figure 11: Accuracy

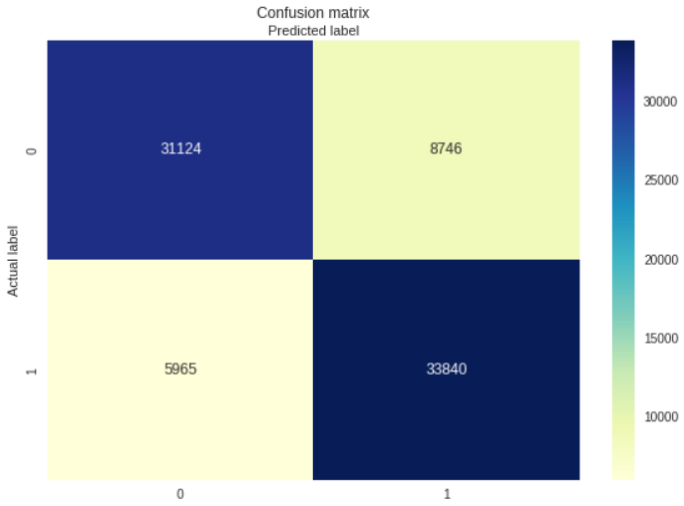



Figure 12: Confussion Matrix

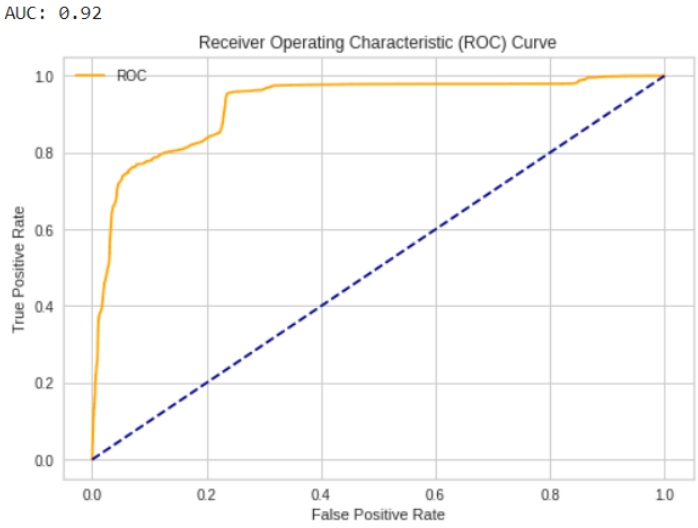
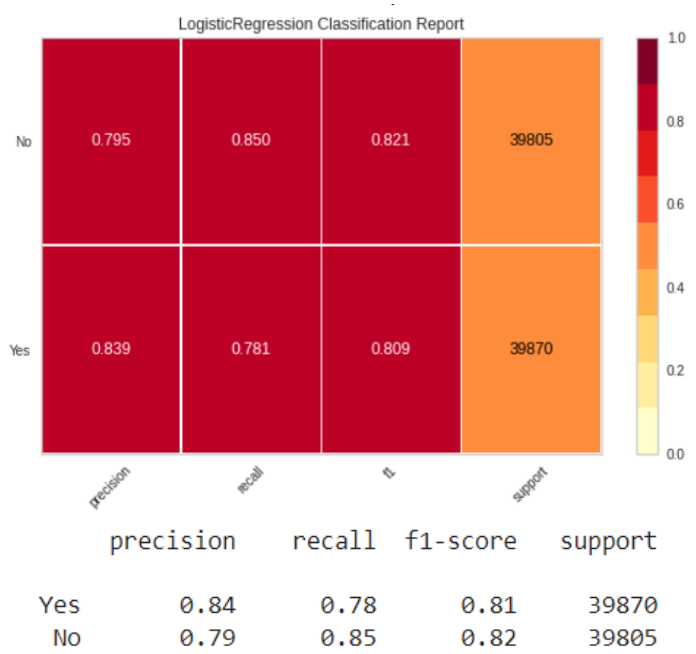



Figure 13: ROC-AUC Curve




| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Yes | 0.84 | 0.78 | 0.81 | 39870 |
| No | 0.79 | 0.85 | 0.82 | 39805 |

Figure 14: Classification Report

Another performance metric that indicates the diagnostic capabilities of binary classifiers is the Receiver Operating Characteristic Curve. The figure 12 shows confusion matrix and figure 13 and figure 14 shows AUC-ROC curve and classification report respectively for logistic regression algorithm.

## 7.2 Experiment using MLP algorithm

In this case data set was trained and tested with MLP. MLP is multilayer perceptron and also called NLP natural language processing. Using MLP I achieved accuracy of 98.83% as output is shown in figure 15 and precision of 99%

```
MLP:Accuracy :  98.83526827737684
```

Figure 15: Accuracy

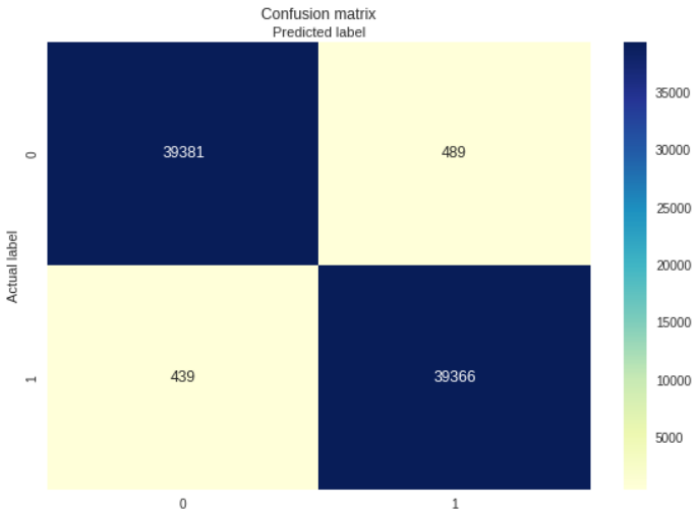



Figure 16: Confusion Matrix

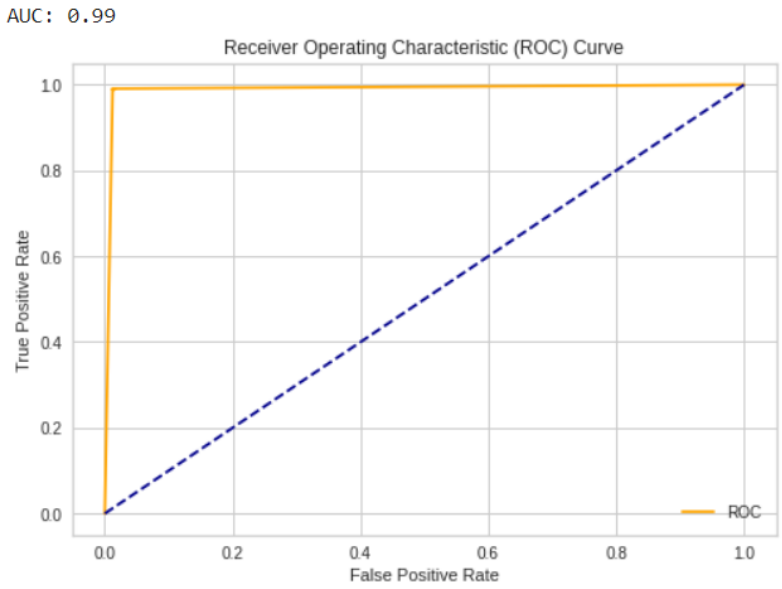
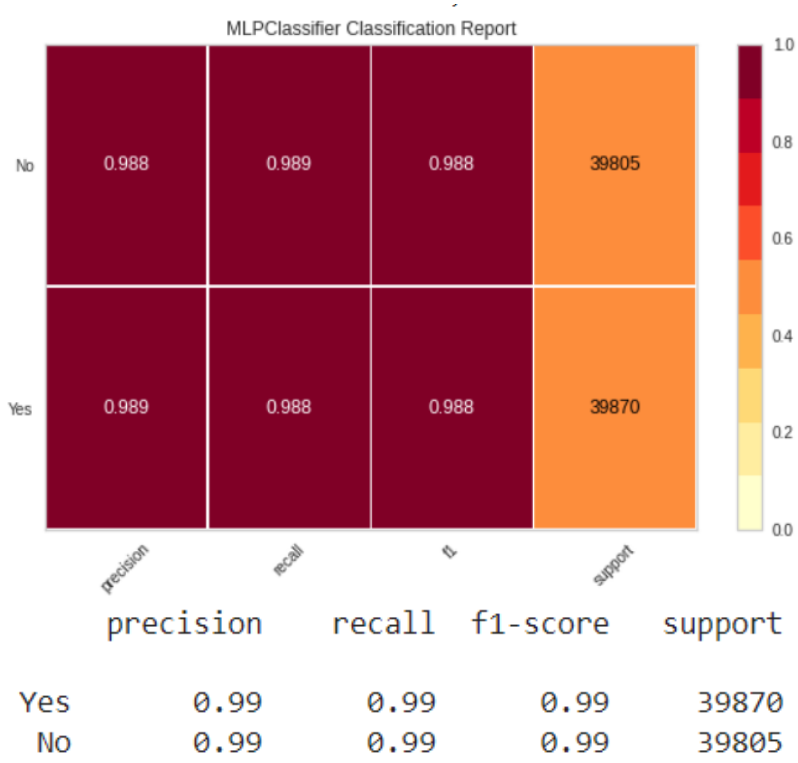



Figure 17: ROC-AUC Curve




| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Yes | 0.99 | 0.99 | 0.99 | 39870 |
| No | 0.99 | 0.99 | 0.99 | 39805 |

Figure 18: Classification Report

The figure 16 shows confusion matrix and figure 17 and figure 18 shows AUC-ROC curve and classification report respectively for multilayer preceptor algorithm.

## 7.3 Experiment using Naïve Bayes algorithm

In this case we trained and tested out data set using Naïve Bayes algorithm and achieved accuracy and precision. Achieved accuracy was 73.98% as output in shown in figure 19 and precision was 90%.

```
Naive_Bayes:Accuracy :  73.98431126451209
```

Figure 19: Accuracy

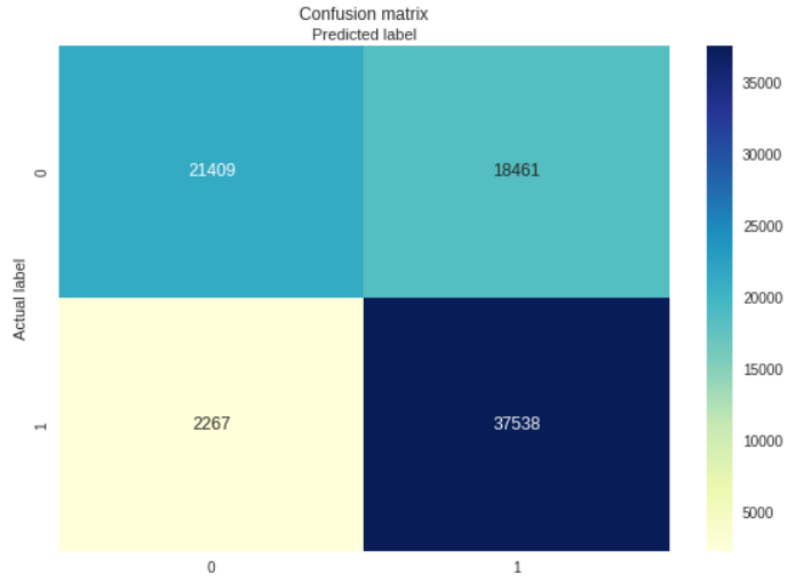



Figure 20: Confusion Matrix

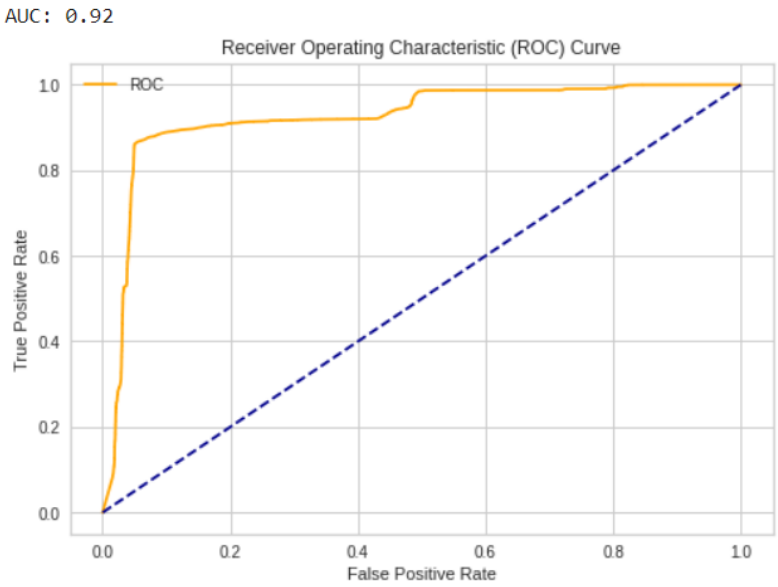
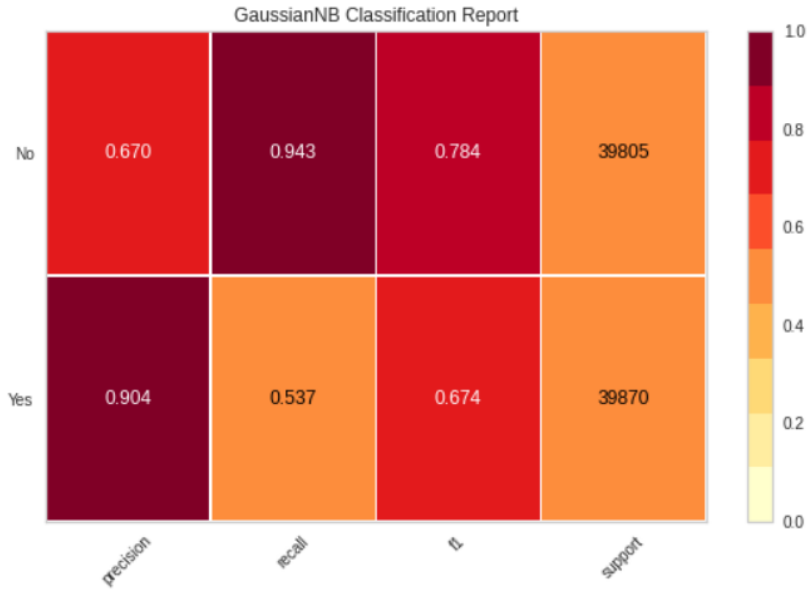



Figure 21: AUC-ROC Curve




| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Yes | 0.90 | 0.54 | 0.67 | 39870 |
| No | 0.67 | 0.94 | 0.78 | 39805 |

Figure 22: Classification Report

The figure 20 shows confusion matrix and figure 21 and figure 22 shows AUC-ROC curve and classification report respectively for naïve bayes algorithm.

## 7.4 Experiment using Gradient Boosting algorithm

In this case we trained and tested out data set using Naïve Bayes algorithm and achieved accuracy and precision. Achieved accuracy was 95.74% as output in shown in figure 23 and precision was 100%.

```
GradientBoosting:Accuracy :  95.74772513335425
```

Figure 23: Accuracy

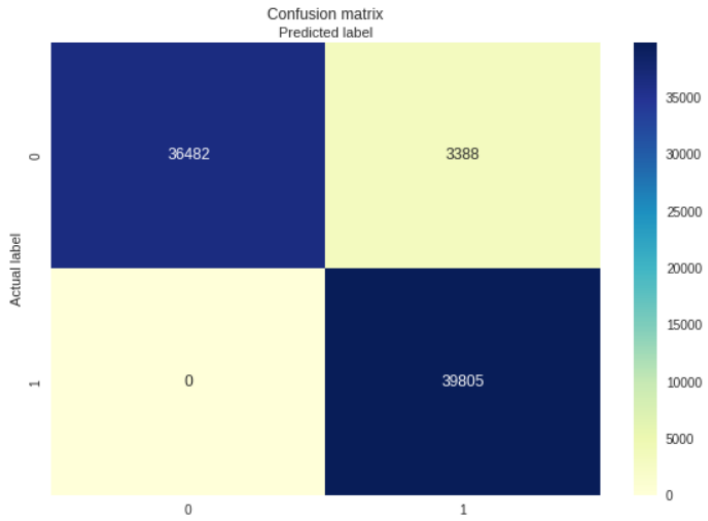



Figure 24: Confusion Matrix

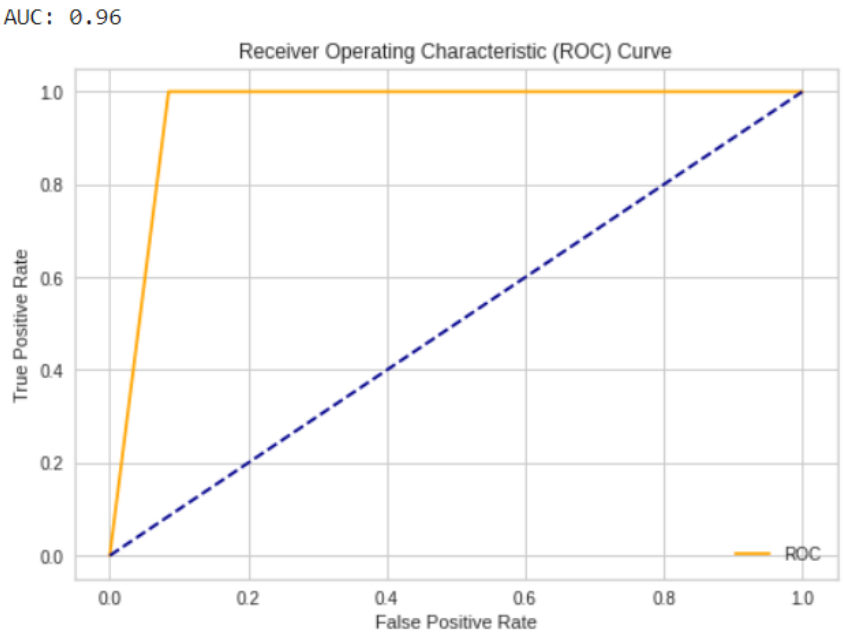
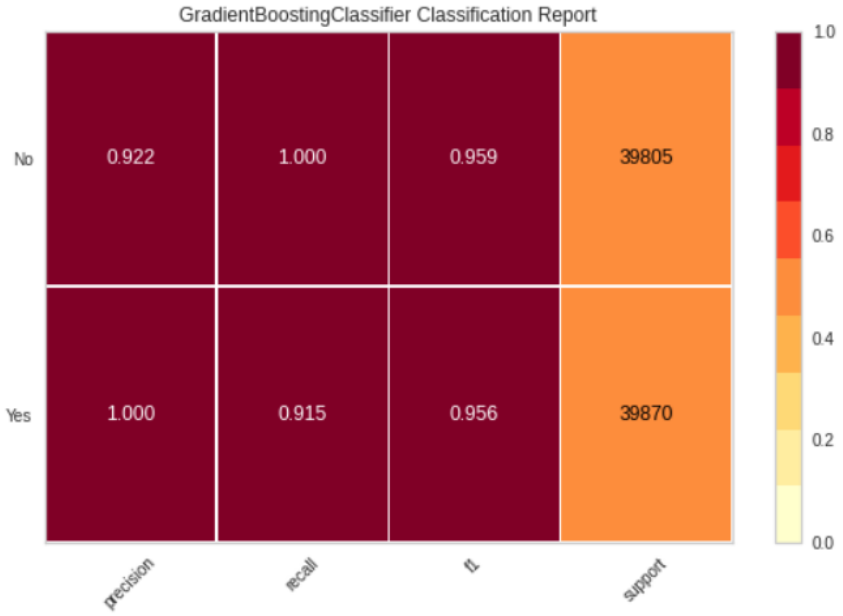



Figure 25: AUC-ROC Curve




| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Yes | 1.00 | 0.92 | 0.96 | 39870 |
| No | 0.92 | 1.00 | 0.96 | 39805 |

Figure 26: Classification Report

The figure 24 shows confusion matrix and figure 25 and figure 26 shows AUC-ROC curve and classification report respectively for gradient boosting algorithm.

## 7.5 Using Stacked Ensemble Learning

In this case accuracy is calculated using meta classifier, i.e. using stacked ensemble learning. Achieved accuracy and precision was 99.83% as output in shown in figure 27 and 100% respectively. Below results are shown:

```
Stacking:Accuracy :  99.83809224976467
```

Figure 27: Accuracy

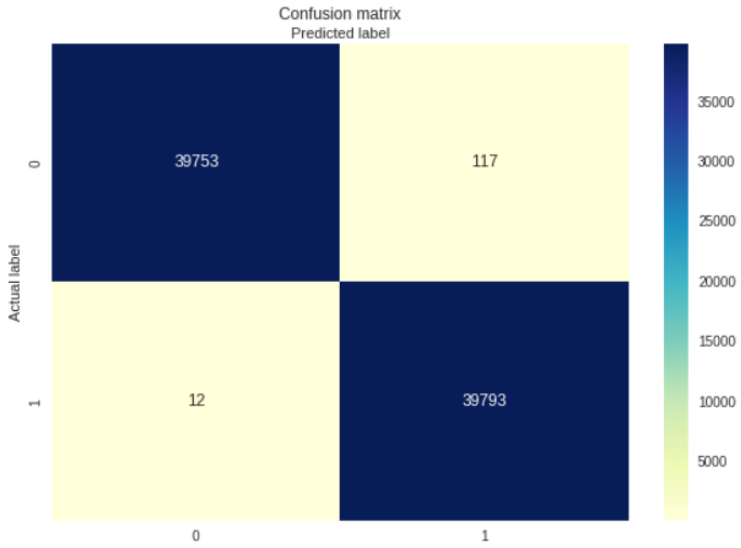



Figure 28: Confusion Matrix

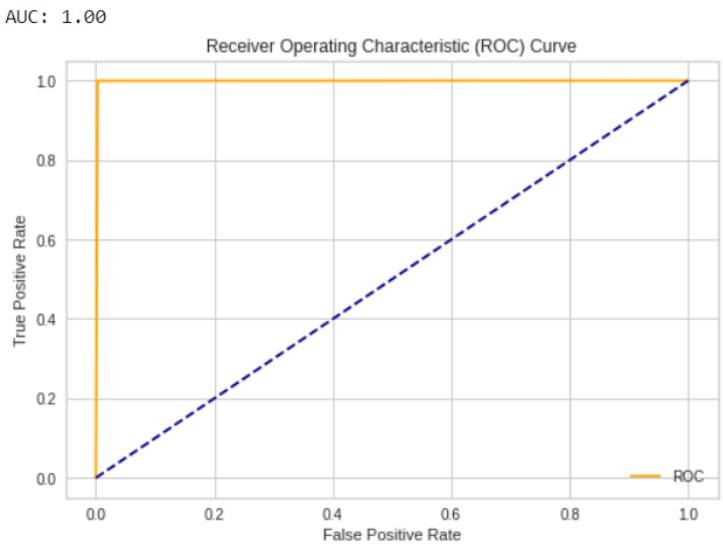
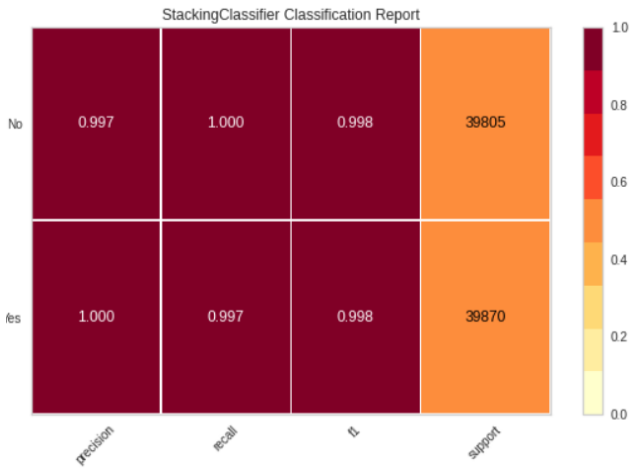



Figure 29: AUC-ROC Curve




| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Yes | 1.00 | 1.00 | 1.00 | 39870 |
| No | 1.00 | 1.00 | 1.00 | 39805 |

Figure 30: Classification Report

The figure 28 shows confusion matrix and figure 29 and figure 30 shows AUC-ROC curve and classification report respectively for gradient boosting algorithm.

## 7.6 Discussion

In the above sections, I have trained and tested the data set using various algorithms and found the accuracy and precision of detecting DDoS attack. Totally I have used four algorithms, and one stacked ensemble learned technique and found out that the accuracy and precision of meta classifier, i.e. stacked ensemble technique, is more accurate and precise. The below bar chart in figure 31 shows a comparison between accuracies and precisions of all the machine learning techniques.

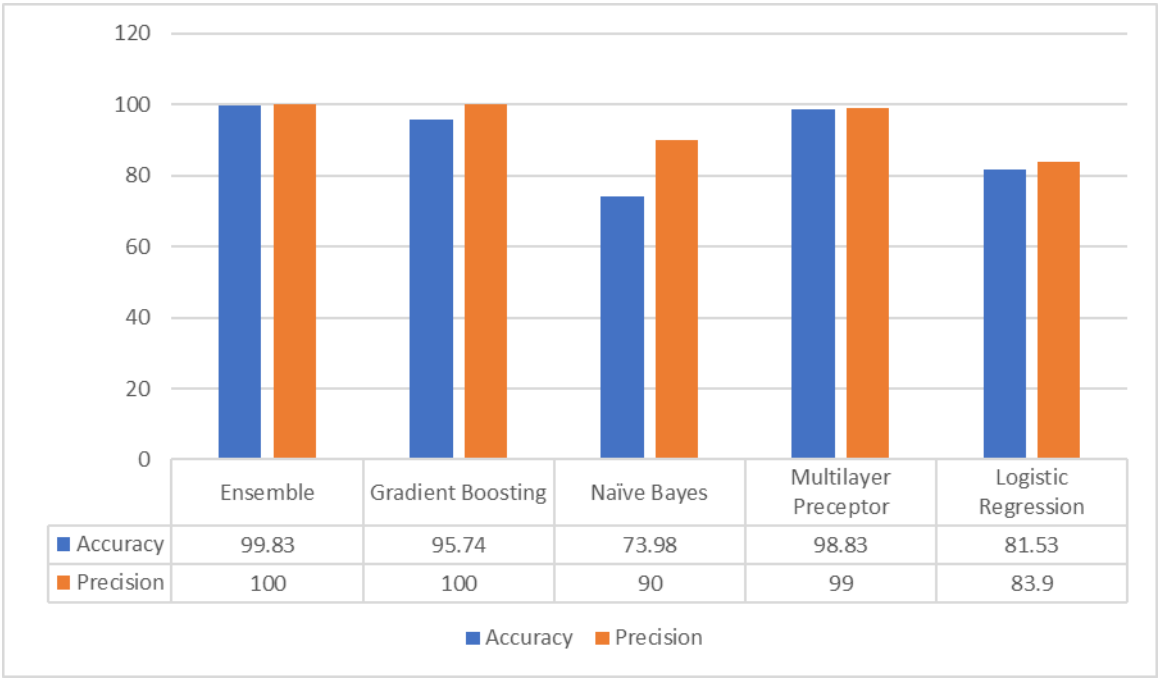



Figure 31: Accuracy and precision comparision

# 8 Conclusion and Future Work

Presently and in the future, more and more smart devices and IoT devices will be connected to the internet. It has become necessary to differentiate DDoS packets and benign in the network, and various DDoS attacks can be avoided. Machine learning techniques can be used to detect DDoS attacks. My proposed and research project explains that the stacked ensemble machine learning method finds the difference between DDoS traffic at the network layer is more accurately and precisely. The proposed research project also states that the accuracy is not affected by the size of the data set or the number of packets. Lastly, I conclude that my proposed research model is more accurate and precise than other machine learning algorithms like logistic regression, MLP, naïve Bayes and gradient boosting to detect DDoS.

In the future, a meta classifier or stacked ensemble learning classifier can also be used to detect IDS or IPS. A decision tree, as well as an exploration of probabilistic, non-probabilistic, and rule induction-based classification algorithms, can be combined as an ensemble for significantly improved performance.

# References


[1] Us.norton.com. 2021. *What is a DDoS attack?*. [online] Available at: <https://us.norton.com/internetsecurity-emerging-threats-what-is-a-ddos-attack-30sectech-by-norton.html> [Accessed 13 December 2021].

[2] Chaudhry, A., Aniol, H. and Shegos, C., 2020. https://www.consultant360.com/article/consultant360/congenital-hypothyroidism-due-thyroid-agenesis. *Consultant*, [Accessed 13 December 2021].

[3] Kaggle.com. 2021. *DDoS Dataset*. [online] Available at: <https://www.kaggle.com/devendra416/ddos-datasets> [Accessed 13 December 2021].

[4] Lima Filho, F., Silveira, F., de Medeiros Brito Junior, A., Vargas-Solar, G. and Silveira, L., 2019. Smart Detection: An Online Approach for DoS/DDoS Attack Detection Using Machine Learning. *Security and Communication Networks*, 2019, pp.1-15 [Accessed 13 December 2021].

[5] Medium. 2021. *Feature Selection Using Random forest*. [online] Available at: <https://towardsdatascience.com/feature-selection-using-random-forest-26d7b747597f> [Accessed 13 December 2021].

[6] Medium. 2021. *Understanding Gradient Boosting Machines*. [online] Available at: <https://towardsdatascience.com/understanding-gradient-boosting-machines-9be756fe76ab> [Accessed 13 December 2021].

[7] Medium. 2021. *Naive Bayes Classifier*. [online] Available at: <https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c> [Accessed 13 December 2021].

[8] Sci-hub.mksa.top. 2021. *A taxonomy of DDoS attack and DDoS defense mechanisms*. [online] Available at: <https://sci-hub.mksa.top/10.1145/997150.997156> [Accessed 13 December 2021].

[9] Ieeexplore.ieee.org. 2021. *A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks*. [online] Available at: <https://ieeexplore.ieee.org/abstract/document/6489876> [Accessed 13 December 2021].

[10] Elizondo, D. and Matthews, S., 2008. Recent Patents on Computational Intelligence. *Recent Patents on Computer Sciencee*, 1(2), pp.110-117 [Accessed 14 December 2021].

[11] Medium. 2021. *Understanding Gradient Boosting Machines*. [online] Available at: <https://towardsdatascience.com/understanding-gradient-boosting-machines-9be756fe76ab> [Accessed 14 December 2021].

[12] Lima Filho, F., Silveira, F., de Medeiros Brito Junior, A., Vargas-Solar, G. and Silveira, L., 2019. Smart Detection: An Online Approach for DoS/DDoS Attack Detection Using Machine Learning. *Security and Communication Networks*, 2019, pp.1-15 [Accessed 14 December 2021].

[13] Prasad, M., V, P. and Amarnath, C., 2019. Machine Learning DDoS Detection Using Stochastic Gradient Boosting. *International Journal of Computer Sciences and Engineering*, 7(4), pp.157-166 [Accessed 14 December 2021].

[14] Nazario, J., 2008. DDoS attack evolution. *Network Security*, 2008(7), pp.7-10 [Accessed 15 December 2021].

[15] Rajadurai, H. and Gandhi, U., 2020. A stacked ensemble learning model for intrusion detection in wireless network. *Neural Computing and Applications*, [Accessed 15 December 2021].

[16] Maglaris, V., 2021. *Detecting DDoS attacks using a multilayer Perceptron classifier*. [online] Academia.edu. Available at: <https://www.academia.edu/21212337/Detecting_DDoS_attacks_using_a_multilayer_Perceptron_classifier> [Accessed 15 December 2021].

[17] Learning Center. 2021. *DDoS Attack Types & Mitigation Methods | Imperva*. [online] Available at: <https://www.imperva.com/learn/ddos/ddos-attacks/> [Accessed 15 December 2021].