DDoS ATTACK PREDICTION SYSTEM

A PROJECT REPORT

Submitted by

B.DEVIKA	**822719104011**
S.DHARSHINI	**822719104014**
A.KARTHIKAYINI	**822719104026**
N.PAVITHRA	**822719104037**

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

DEPARTMENT OF

COMPUTER SCIENCE AND ENGINEERING

GOVERNMENT COLLEGE OF ENGINEERING, THANJAVUR

ANNA UNIVERSITY :: CHENNAI - 600 025

MAY 2023

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report "**DDoS ATTACK PREDICTION SYSTEM**" is bonafide work of "**B.DEVIKA (822719104011), S.DHARSHINI (822719104014), A.KARTHIKAYINI (822719104026) and N.PAVITHRA (822719104037)",** who carried out the project work under my supervision.

SIGNATURE
Mr.K.MANOJ KUMAR, M.E.,
HEAD OF THE DEPARTMENT
Associate Professor
Department of CSE
Government College of Engineering
Sengipatti,
Thanjavur - 613402

SIGNATURE
Dr.R.MANIKANDAN, M.E.Ph.D.,
Assistant Professor
Department of CSE
Government College of Engineering
Sengipatti,
Thanjavur - 613402

Submitted for CS8811 – Project Work viva–voce Examination held on ……

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

DDoS attack is crucial for network security. DDoS attacks are a type of cyber attack that can cause significant damage to businesses and organizations, resulting in financial losses, reputational damage, and legal liabilities. These attacks involve overwhelming a network or a website with a massive amount of traffic, making it impossible for legitimate users to access the network or website. Predicting DDoS attacks is essential for ensuring business continuity and maintaining the trust of customers and stakeholders. With the increasing frequency and complexity of cyber attacks, predictive models that can effectively detect and prevent DDoS attacks are essential for organizations to protect their assets and maintain their operations. These attacks are increasing day by day and have become more sophisticated. So, it has become difficult to detect these attacks and secure online services. The system presents a technique for predicting the ddos attack using Random Forest algorithm on the UNSW_NB_15 dataset. The UNSW_NB_15 dataset is a publicly available dataset that contains different types of network traffic, including normal and attack traffic. The proposed model uses a feature selection technique to select relevant features from the dataset, and these features are used to train the Random Forest model. The model's effectiveness is evaluated using different performance metrics, including accuracy, precision and recall. This study contributes to the development of effective DDoS attack prediction models and highlights the effectiveness of the Random Forest algorithm in predicting such attack.

ACKNOWLEDGEMENT

This project work is dedicated to Almighty God blessing with inspirational parents, teachers and good friends.

We are extremely thankful with no words of formal nature to the dynamic principal of our college **Dr.M.NATARAJ, M.E., Ph.D and Dr.S.JAYABAL M.E., Ph.D,** for providing all the necessary facilities to complete our work.

We would like to thank **Mr.K.MANOJKUMAR, M.E.**, Associate Professor, Head of Computer Science and Engineering department for his valuable guidance and encouragement in completing this project work.

We would like to express our sincere gratitude and heartfelt thanks to our well-wisher and our project coordinator **Dr.G.MAHALAKSHMI, M.E, Ph.D.,** Assistant Professor for her valuable guidance and her constant efforts to make this project successful.

We extend our gratitude to our guide, **Dr.R.MANIKANDAN, M.E.Ph.D.,** Assistant Professor for his valuable ideas and suggestions, which have been very helpful in the project.

We are grateful to all the faculty members of the Computer Science and Engineering Department, for their support.

We also like to express our sincere thanks and gratitude to our parents and friends for their continuous encouragement and support.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF ABBREVIATION

DDoS	-	Distributed Denial of Service
DoS	-	Denial of Service
UDP	-	User Datagram Protocol
TCP	-	Transmission Control Protocol
DNS	-	Domain Name System
HTTP	-	Hyper Text Transfer Protocol

CHAPTER – 1

INTRODUCTION

1.1 ABOUT MACHINE LEARNING

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the usage of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. It allows systems to learn from their failures and develop without having to be explicitly designed. It mainly aims at developing computer programs that can access data and utilize it to learn for themselves.

Like the human brain profits knowledge and understanding, machine learning depends on input, such as training data to understand entities, domains and the connections. The process of machine learning starts with learning or data, such as examples, direct experience or instruction. It seeks for patterns in data so it can later make inferences based on the examples provided. The primary aim of machine learning is to allow computers to learn independently without human intervention or assistance and adjust actions accordingly. Machine learning is critical since it provides insight into customer behaviour and business patterns, as well as assisting in the development of new products. Many of today's leading companies, such as Facebook, Google and Uber, make machine learning a core part of their operations. Machine learning has become an essential competitive differentiator for many companies.

1.2 FOUNDATIONS OF MACHINE LEARNING

It affords computers the potential to learn without being explicitly instructed," said Arthur Samuel, an American pioneer in the fields of computer games and artificial intelligence, in 1959. "A computer model is said to learn from experience E with reference to some task T and some performance 2 measure P, if its performance on T, as measured by P, rises with experience E," according to Tom Mitchell's definition from 1997. The phrase "machine learning" is currently trending. Learning algorithms have been successfully deployed in a variety of applications, including

- Spam detection, for example, requires text or document classification.
- Morphological analysis, part-of-speech tagging, statistical parsing, and named-entity recognition are examples of natural language processing.

- Speaker verification, voice recognition, and speech synthesis.
- OCR stands for optical character recognition.

1.3 GOAL OF DDoS ATTACK PREDICTION SYSTEM

Distributed Denial of Service attack (DDoS) is one of the most dangerous attack in the field of network security. It is an attack which coordinated stream of request is launched against from many locations at a same time. These attacks are increasing day by day and have become more sophisticated. So, it has become difficult to detect these attacks and secure online services. Many of the organizations like github, aws get affected due to the DDoS attack. So, there is a need to detect the DDoS attack. Hence, proposed a system to predict these attacks using random forest algorithm in machine learning.

1.4 ARCHITECTURE OF DDoS ATTACK PREDICTION SYSTEM

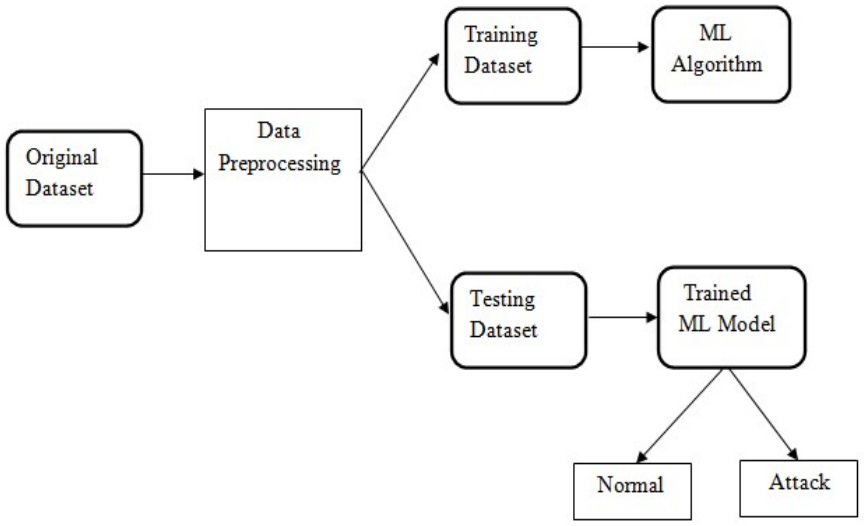


Fig. No. 1.4.1 Architecture of DDoS attack prediction system

CHAPTER – 2

LITERATURE SURVEY

2.1 LITERATURE SURVEY

Arun Nagaraja et al.,[1] proposed a hybrid model deep learning model for intrusion detection. They combined two deep learning models for the classification of CNN+ LSTM from the RNN model. The dataset was used in this work is KDD. They found an 85.14% average accuracy for the proposed. They used k mean cluster model for feature similarity detection and naïve Bayes model used for classification.

Ashutosh Nath Rimal and Dr.Raja Praveen [2] have proposed a DDoS attack detection system using ML algorithms and packet analysis in a smart way. In this case the classification algorithm being used are Naïve Bayes and SVM algorithm out of which SVM was found to give maximum accuracy. This system has achieved with a maximum accuracy of 99.68%, if the recommended combination of feature selection and classification algorithm is chosen. The user is left with the choice for both feature selection and classification algorithm.

Bakker, J. et [3] This paper has shown how statistical classification can be deployed using SDN to detect DDoS attacks. Three classifiers were selected in a off-line environment to be integrated with nmeta2. These were then evaluated on a physical network testbed by replaying a DDoS attack scenario. While statistical classification can be deployed using SDN to classify traffic, careful consideration must be made to pick classifiers that result in the smaller possible packet processing overhead.

Guiomar et al [4] has implemented a Network Intrusion Detection System (NIDS) using OPNET simulation. This was based on misuse detection and network traffic was imported using an ACE module into OPNET. A NMAP port scanner was used to simulate a flood attack, and the proposed IDS was tested in a controlled environment; the result was satisfactory with around 93% accuracy rate.

Jing Wu, et al [5] have suggested a method for detecting DDoS Attacks in Software-Defined Networks through Feature Selection Methods and Machine Learning Models. In this study, The

SDN-based detection systems developed for DDoS attacks were analyzed by using machine learning systems. In the first proposed approach, by analyzing flow data, algorithms with 98.3% accuracy ensure the detection of attacks without discriminating the type of traffic. With 97.7% sensitivity, KNN algorithms can perform this control by facilitating the charge of the controller.

Khuphiran, P et al [6] the application of machine learning algorithm for the problem of DDoS attack detection has been addressed. Two algorithms, Support Vector Machine (SVM) and Deep Feed Forward (DFF) have been evaluated to demonstrate the feasibility of applying these algorithms. The experiments have been conducted to compare the performance of these algorithms. It has been found that DFF can classify the data with a higher accuracy.

Kimmi kumari and M.Mrunalini [7] have proposed a system for detecting DDoS attack by using various machine learning algorithms. The major goal of this paper's work is to distinguish between normal and attacks scenarios by analyzing the throughput of the data packets respectively.

Larriva-Novo et al.[8] proposed two benchmark datasets, especially UGR16 and UNSW-NB15, and the most used dataset KDD99 were used for evaluation. The pre-processing strategy is evaluated based on scalar and standardization capabilities. These pre-processing models are applied through various attribute arrangements. These attributes depend on the classification of the four sets of highlights: basic associated highlights, content quality, fact attributes, and finally the creation of highlights based on traffic and traffic quality based on associated titles Collection. The goal of this inspection is to evaluate this arrangement by using different information pre-processing methods to obtain the most accurate model.

Liu et al [9] have suggested a method for detecting DDoS attack using deep learning techniques. To increase the validity and effectiveness of feature extraction, a convolutional neural network (CNN) modeling approach for intrusion detection was applied. The convolution kernel was chosen and convolved with the data to extract local correlation. The new approach can raise classification accuracy for jobs involving intrusion detection and recognition.

Maede Zolanvari et al.[10] proposed a recurrent neural network model for classification intrusion detection. They compared other deep learning models with RNN. Finally, they found RNN is the best model for intrusion detection by using the KDD dataset. It was a multiple classification problem. They used advanced deep learning LSTM for multiple classification

problems. They found good results with 89% average accuracy for the proposed work.

Mouhammd Alkasassbeh, Ahmad B.A Hassanat, Ghazi AI-Naymat, Mohammad Almseidin [11] have suggested a method for detecting DDoS attack using Data Mining Techniques. In this model, the DDoS attack can be detected using Intrusion Detection System (IDS) technique with a maximum accuracy of 89%.This system mainly concentrated on network layer and application layer.

Nisha Ahuja A et al [12] have proposed a system Automated DDOS attack detection in software defined networking. This model is to classify normal traffic from DDOS attack traffic using the machine learning technique. The important contribution of this paper is to identify the novel features for DDoS attack detections. New features are saved in the CSV file to create the dataset and machine learning algorithms are trained on the created SDN dataset. Several previous works on DDoS detection have either used a non-SDN dataset or the research data has not been made public. Software Defined Networking (SDN) is the networking architecture defined by the software program. In SDN, the network traffic is managed by software, which leads the traffic between hosts.

Norouzian et al [13] presented a most effective classification technique for detecting and classifying DDoS attacks into two groups normal or threat. They proposed a new approach to IDS based on a Multilayer Perceptron Neural Network to detect and classify data into 6 groups. They implemented their MLP design with two hidden layers of neurons and achieved 90.78% accuracy rate.

Sana Ullah Jan et al.[14] proposed a PSO-Xgboost model because it is higher than the overall classification accuracy alternative models, e.g. Xgboost, Random-Forest, Bagging, and Adaboost. First, establish a classification model based on Xgboost, and then use the adaptive search PSO optimal structure Xgboost. NSL-KDD, reference dataset used for the proposed model evaluation. Our results show that, PSO-Xgboost model of precision, recall, and macro-average average accuracy, especially in determining the U2R and R2L attacks. This work also provides an experimental basis for the application group NIDS in intelligence.

Saini et al [15] proposed a machine learning approach that detects DDoS attacks and classifies the type of DDoS attack. It contains various types of modern DDoS attacks like HTTP-flood,

SIDDoS, smurf and UDP-Flood. Also, the dataset consists of 27 features and 5 different classes. For detecting the DDoS attack a machine learning tool WEKA was used. Four machine learning algorithms like Random Forest, MLP, Naïve Bayes, and J48 were implemented. Among the four algorithms, J48 gave the best results compared to other classifiers.

Somani, G, Gaur, M.S, Sanghi, D, Conti, M, Buyya, R [16] have proposed a detailed taxonomy on DDoS detection in cloud computing. This system detect and analyze the DDoS attack in cloud computing using Dampster- Shafer Theory (DST) operation in 3-valued logic and fault tree analysis for each VM- based IDS. In this work, they focused on only a particular environment that is cloud computing and context-aware mechanism is needed.

Wani et al [17] discussed DDoS attacks, their impact and the losses incurred by these attacks. They provide a comprehensive and systematic analysis of block chain based DDoS mitigation techniques. The mitigation techniques presented in paper are divided into various types such as Software Defined Networking (SDN), Artificial Intelligence, Block chain and Collaborative platforms.

Warrender et al.[18] have proposed several intrusion detection methods based upon system call trace data. This system tested a method that utilizes sliding windows to determine a database of normal sequences to form a database for testing against test instances then used a similar method to compare windows in the test instances against the database and classify instances according to those in the normal sequence database.

Xianwei Gao et al.,[19] proposed a comparative work for network traffic classification. They used machine learning classifiers for intrusion detection. The dataset is taken is CICIDS and KDD from the UCI repository. They found support vector machine SVM one of the best algorithms as compare to othersThey used the KDD dataset from an online repository. These models are Dtree, R-forest, and KNN classifiers. In this study, the authors found that Dtree and ensemble models are good for classification results. The overall accuracy of the proposed work is 85%.

Zheng et al [20] implemented a Hierarchical Intrusion Detection (HIDE) system which can detect attacks using preprocessing statistical values and a Neural Network. They tested five classifiers: Perceptron, Back propagation (BP), Perceptron-back propagation-hybrid (PBH), Fuzzy ARTMAP, and Radial-based Function. They stated that BP and PBH achieved the highest accuracy

rate for detecting and classifying network attacks.

2.2 EXISTING SYSTEM

From the above literature survey discussion, the existing system using SVM algorithm to predict the DDoS attack. The existing system have some further improvements. The details of the improvements are listed below.

a. One of the disadvantage of SVM is that it may not perform well with very large datasets or datasets with high degree of noise or overlapping classes.
b. The potential disadvantage of SVM is that it may have a higher false positive rate.

2.3 PROPOSED SYSTEM

The proposed system is expected to meet the following requirements

a. Random forest can overcome the issue faced in existing system by combining multiple decision trees and using voting system to classify network traffic.
b. Random forest is a decision tree-based ensemble method that can handle large dataset with high dimensionality and noise. It can also handle missing values and outliers, and is less sensitive to hyper parameters than SVM.
c. So, this system using the random forest algorithm for predicting the ddos attack by extracting some important features in the dataset.

2.4 FEASIBILITY STUDY

The Feasibility of the project is analysed in this phase, and a business proposal is put forth with a very general plan for the project and some cost estimates. Feasibility study is necessary to determine if creating a new or improved system is friendly with the cost, benefits, operation, technology and time. Following feasibility study is given as below:

2.4.1 Technical Feasibility

This study was carried out to check the technical feasibility, which is one of the first studies that must be conducted after the project has been identified. Technical feasibility study

includes the hardware and software devices. The required technologies include machine learning algorithm which can be implemented using python and some required libraries, and UNSW_NB_15 dataset. Hardware requirements are all satisfied as we plan to train data on jupyter notebook. We plan to use Anaconda Navigator for training purposes. Any system developed must not have a high demand on the available technical resources.

2.4.2 Economic Feasibility

The purpose of economic feasibility is to determine the positive economic benefits that include quantification and identification. The system is economically feasible due to the availability of all requirements such as collection of data and further can be implemented if possible in different operating system to enhance security.

2.4.3 Operational Feasibility

Operational Feasibility is a measure of how well a proposed system work out for the goal and takes advantage of the opportunities identified during scope definition. The following points were considered for the project's operational feasibility:

- The libraries are all available in sklearn and can be implemented using python.
- The datasets are all public and no corporate issues generated.

CHAPTER - 3

SYSTEM REQUIREMENTS

3.1 HARDWARE REQUIREMENTS

Pre-processor	:	Intel Core i7
RAM	:	4 GB
Hard Disk	:	60 GB
Keyboard	:	Standard Keyboard
Monitor	:	15-inch color monitor

3.2 SOFTWARE REQUIREMENTS

Operating System	:	Windows 11
Language	:	Python
Front End	:	HTML, CSS
Back End	:	JavaScript
Packages	:	Numpy, Pandas, Sklearn, Matplotlip

3.3 SOFTWARE SPECIFICATION

3.3.1 Language

Python is a general-purpose, high-level, interpreted programming language. Its design philosophy prioritizes code readability by employing a large amount of indentation. Python is garbage-collected and dynamically typed. It supports a variety of programming paradigms, including structured programming, object oriented programming, and functional programming.

Guido van Rossum started working on Python in the late 1980s as a replacement for the ABC programming language, and Python 0.9.0 was launch in 1991. List comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support were all included in Python 2.0, which was published in 2000. Python 3.0, introduced in 2008, was a significant update that

was not fully backwards compatible with previous versions. Python 2 was deprecated in 2020, with version 2.7.18.

3.3.2 Features of Python

3.3.2.1 Simple

Python is a simple language to learn compared to other programming languages like C, C#, Javascript, and Java. Python is a simple language to code with, and anyone can learn the basics in a few hours or days. It's also a language that's easy to learn for programmers.

3.3.2.2 Free and Open Source

Python is a coding language that is freely available on the website and can be downloaded. The software is also available to the entire public because it is open source. As a result, it is free to download, use, and share it.

3.3.2.3 Object Oriented Language

Object-Oriented programming is one of Python's most important features. Python supports object-oriented programming and its concepts like classes, encapsulation, and etc...

3.3.2.4 High Level Language

Python is a programming language that is designed to be used at a high level. While writing Python apps, it is not required to remember the system architecture or manage memory.

3.3.3 Python Packages

3.3.3.1 Numpy

Numpy is a Python library that adds support for huge, multi-dimensional arrays and matrices, to work on these arrays, as well as a huge number of highlevel mathematical functions. Numeric, Numpy's forerunner, was built by Jim Hugunin with help from a number of other people. Numpy was created in 2005 by Travis Oliphant, who substantially modified Numeric and combined features from the competition Numarray. Numpy is a huge open-source project with

many contributors. Numpy is a financially supported NumFOCUS project.

3.3.3.2 Pandas

Pandas is a Python package that lets users to work with massive amounts of data. It includes tools for data purification, exploration, and modification as well as data analysis. Wes McKinney came up with the name "Pandas" in 2008, which refers to both "Panel Data" and "Python Data Analysis."

3.3.3.3 Random

The Random module in Python is a built-in module for generating random integers. These numbers are pseudo-random and do not represent actual randomness. This module can be used to generate random integers, generate a random value for a list or string, and much more.

3.3.3.4 Matplotlib.Pyplot

Matplotlib is a charting toolkit for Python that allows users to create static, animated, and interactive charts. Matplotlib is a Python library that may be used in Python scripts, the Python and IPython shells, and extra graphical user interface toolkits in the IPython shells.

Pyplot is a Matplotlib package that mimics the MATLAB graphical user interface. Matplotlib is intended to be as user-friendly as MATLAB, but with the added benefit of being open-source and free. Each pyplot function alters a figure in some way, such as adding a figure or changing the plotting area. charting certain lines in a plotting area, decorating the plot with labels, and so on. Some of the plots that may be used with Pyplot are Line Plot, Histogram, Scatter, 3D Plot, Image, Contour, and Polar.

3.3.3.5 Sklearn

Scikit-learn (Sklearn) is the most useful and robust for machine learning in python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in python. This library, which is largely written in python, is built upon Numpy. Scipy and matplotlib.

3.3.4 Installation

3.3.4.1 Anaconda

Anaconda is a Python and R programming language distribution for scientific computing that aims to make package management easier and makes deployment easier (data science, machine learning applications, large-scale data processing, predictive analytics, and so on). Data-science packages for Windows, Linux, and macOS are included in the release. Anaconda, Inc., created by Peter Wang and Travis Oliphant in 2012, is responsible for its development and maintenance. Other Anaconda, Inc. products include Anaconda Distribution and Anaconda Individual Edition. Whereas Anaconda Team Edition and Anaconda Enterprise Edition, both of which are not free, are other Anaconda, Inc. products.

3.3.4.2 Jupyter

Jupyter Notebook is an open source web tool for creating and sharing documents with live code, equations, visualisations, and text. Project Jupyter is in charge of maintaining Jupyter Notebook. Jupyter Notebooks is a fork of the IPython project, which used to have its own IPython Notebook project. Julia, Python, and R are the three primary programming languages that Jupyter supports. Jupyter provides the IPython kernel, which allows users to write Python programs, although there are currently over 100 other kernels available.

3.3.5 Random Forest Algorithm

A random forest algorithm is a combination of the decision tree. It is very fast compared to other classifiers. Now after feature scaling the next step is the machine learning classification model. In our proposed work, the random forest classification algorithm is used.

The Random forest, which is one of the most popular and powerful machine learning classification algorithm. It can apply for both classification and regression problems. It is based on ensemble learning, which integrates multiple classifiers to solve a complex issue and increases the model's performance.

Random Forest is a classifier that contains several decision trees on various subsets of a given dataset and takes the average to enhance the predicted accuracy of that dataset. Instead of relying on a single decision tree, the random forest collects the result from each tree and expects the final output based on the majority votes of prediction.

3.3.6 Working of Random Forest Algorithm

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

3.3.6.1 Classification Result

In this phase, it create a combination of decision tree by selecting those important features to predict the DDoS attack. Those features are total duration, source bytes, destination bytes, source time to live, destination time to live. Because if the total duration is less than or equal to one second, source bytes is less than or equal to thousand, then the destination bytes are less than or equal to five hundred, source time to live is less than or equal to two hundred and fifty four and the destination time to live is less than or equal to two hundred and fifty two then it predict the category is attacked. Otherwise, the category is normal.

3.3.6.2 Decision Tree

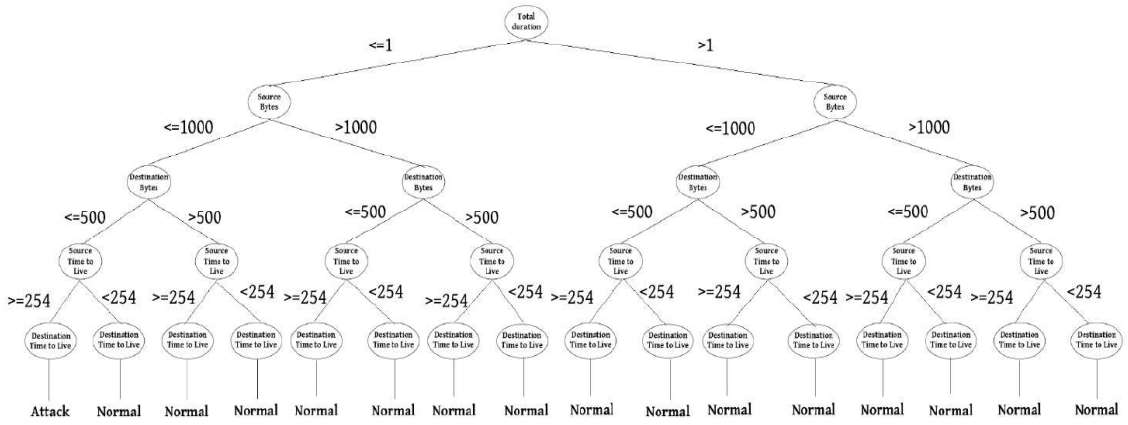
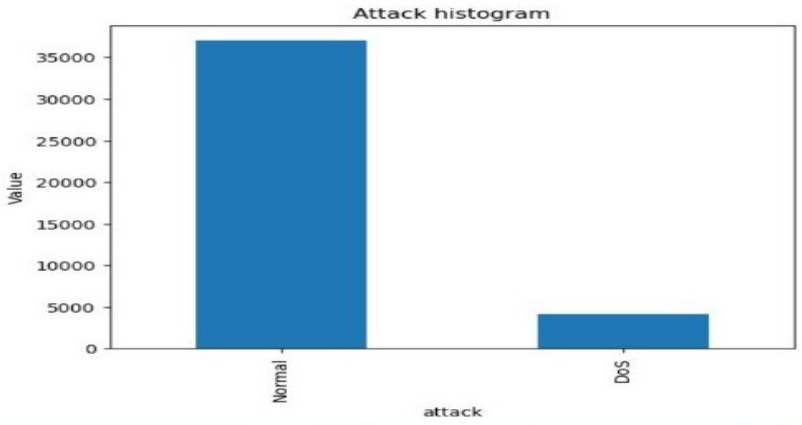


Fig. No. 3.3.6.2.1 Decision Tree for DDoS attack prediction system

3.3.6.3 Confusion Matrix

The confusion matrix denotes the overall number of actual and predicted labels for a particular algorithm. It is used to calculate the accuracy of the representation, just like arranging true and prescient marks. The below figure represents the confusion matrix for UNSW_NB_15 dataset.

```
Out[26]: Normal    37000
         DoS        4089
         Name: attack_cat, dtype: int64
```




```
#prediction
predictions=model.predict(X_test)
print(predictions)
print(accuracy_score(y_test,predictions)*100)
print("Confusion Matrix:",confusion_matrix(y_test,predictions))
print("Precison :",precision_score(y_test,predictions,average='macro'))
print("Recall :",recall_score(y_test,predictions,average='macro'))
print("F1 Score :",f1_score(y_test,predictions,average='macro'))
```

```
['Normal' 'Normal' 'Normal' ... 'Normal' 'Normal' 'Normal']
97.7914334387929
Confusion Matrix: [[ 1445   283]
 [   80 14628]]
Precison : 0.964280853281382
Recall : 0.9153938175495322
F1 Score : 0.9380775255520706
```

```
In [ ]:
```

3.3.7 Algorithm Approach

- Pick M data points at random from the training set.
- Create decision trees for your chosen data points (Subsets).
- Each decision tree will produce a result. Analyze it.
- For classification and regression, accordingly, the final output is based on Majority Voting or Averaging, accordingly.

CHAPTER - 4

SYSTEM DESIGN

System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. It involves planning and designing the overall structure of a system, including its hardware and software components, to achieve specific goals and requirements. The system design process typically involves identifying the system's requirements, analyzing and defining its architecture, selecting appropriate hardware and software components, designing system interfaces, and creating a detailed plan for implementation, testing, and maintenance.

4.1 UML DIAGRAM

4.1.1 Work Flow Diagram

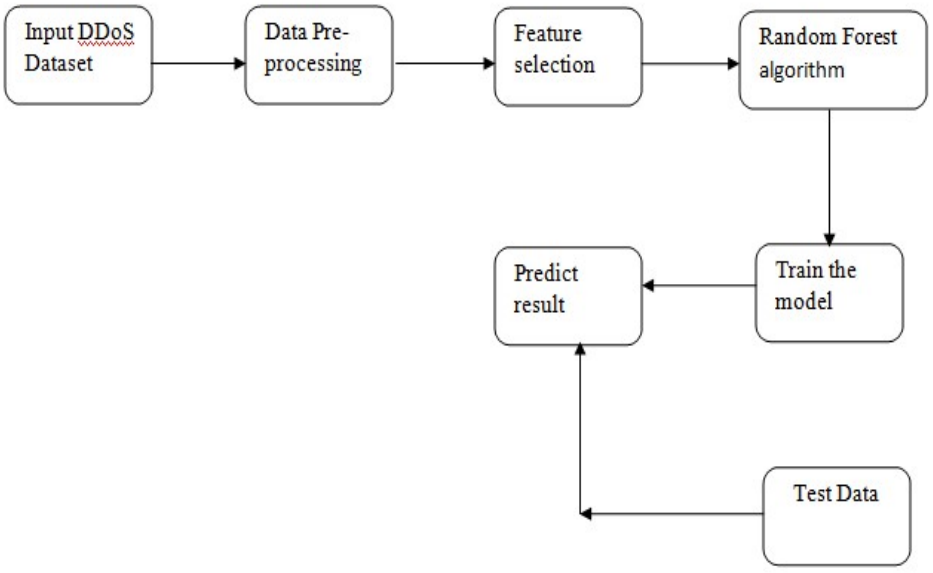


Fig. No. 4.1.1.1 Work Flow Diagram

.

4.1.2 Activity Diagram

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. So, the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent. Activity diagrams deals with all type of flow control by using different elements like fork, join.

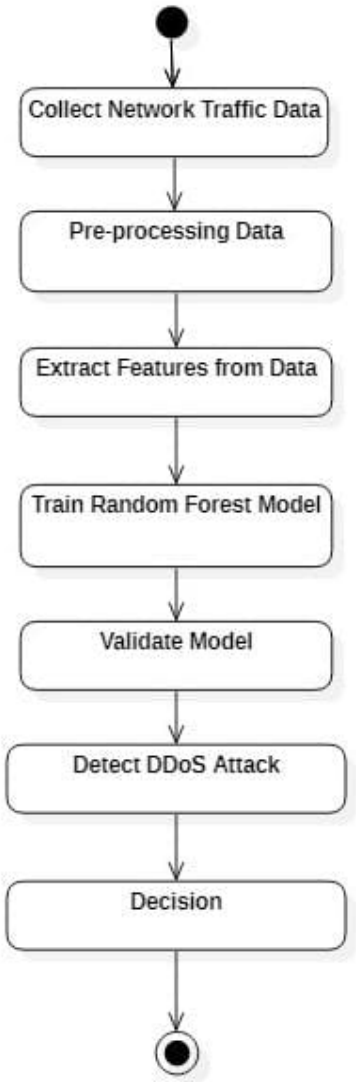


Fig. No. 4.1.2.1 Activity Diagram

4.1.3 Use Case Diagram

Use case diagrams are usually referred to as behaviour should provide some observable and valuable result to the actors or other stakeholders of the system. Each use case diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors).

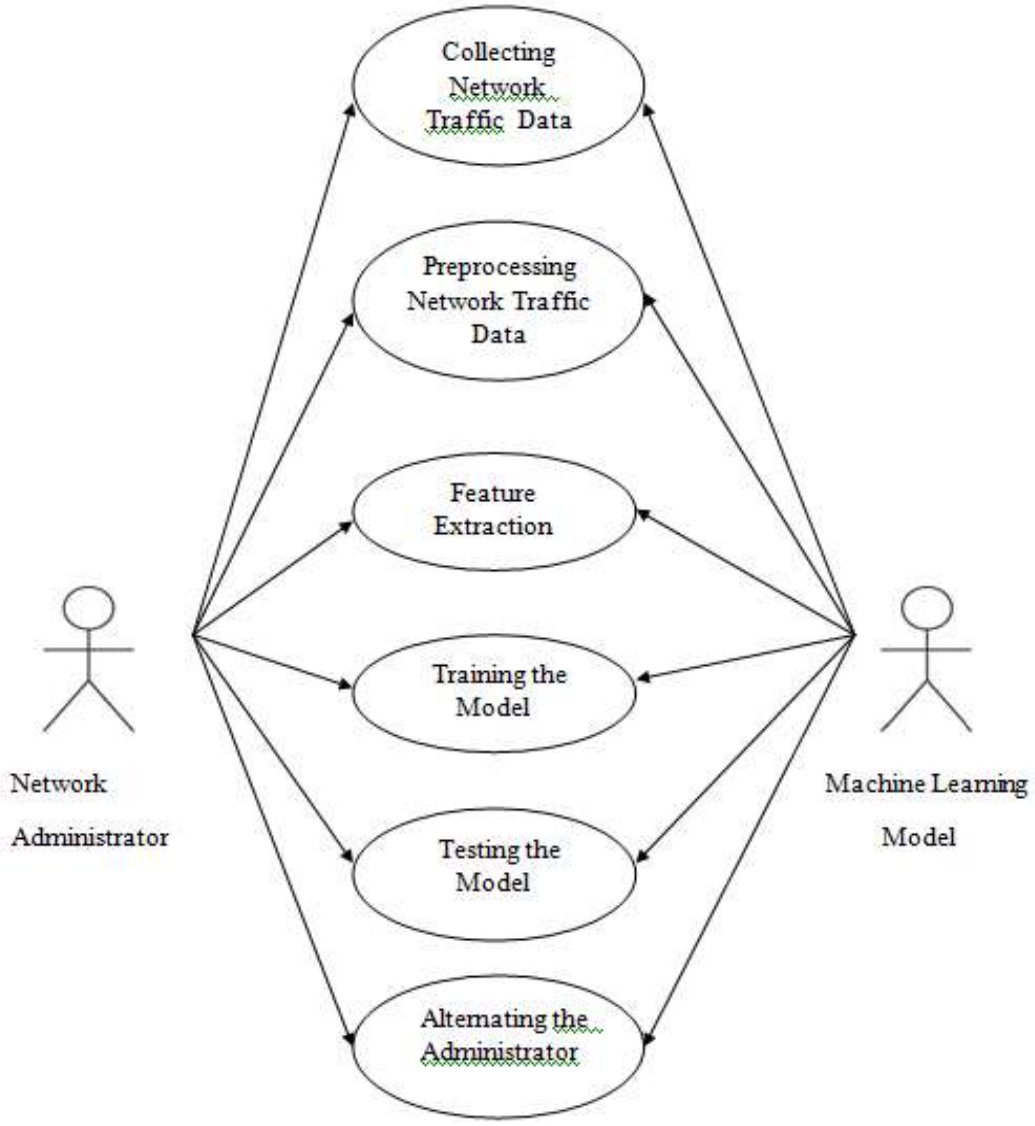


Fig. No. 4.1.3.1 Use Case Diagram

4.1.4 Sequence Diagram

A Sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart. A sequence diagram shows object interactions arranged in time sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

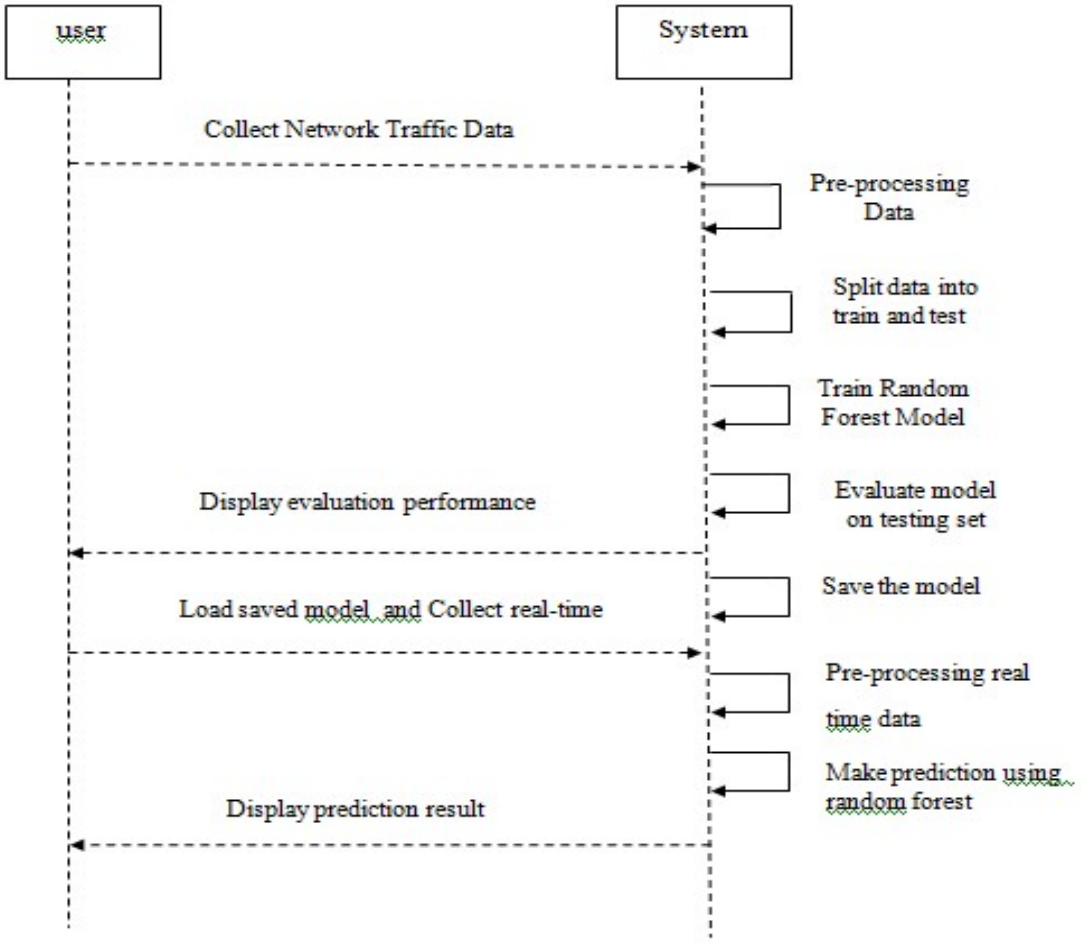


Fig. No. 4.1.4.1 Sequence Diagram

CHAPTER – 5

SYSTEM IMPLEMENTATION

5.1. MODULES

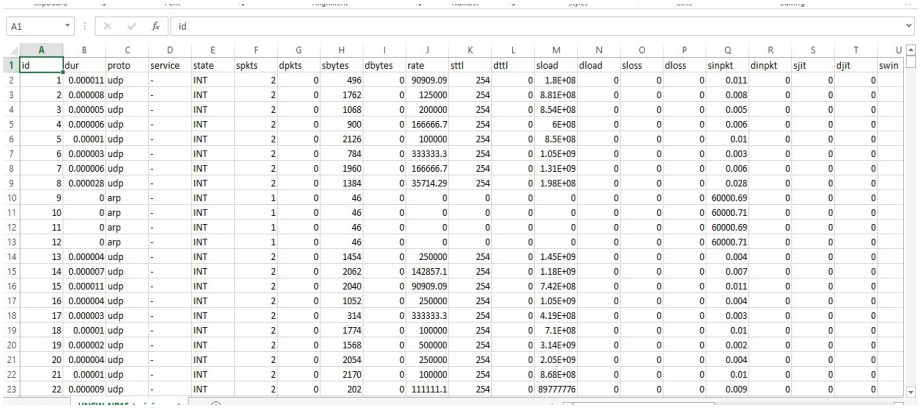
A module is a set of source files and builds options that allow you to break down your project into separate functional components. Our project can include up to five modules, each of which can rely on another module. Each module may be built, tested, and debugged individually. The details of the modules are given below.

1. Data Collection
2. Data Preprocessing
3. Data Visualization
4. Train the Model
    (i) Classification Result
    (ii) Confusion Matrix
5. Result Analysis

5.2 MODULES DESCRIPTION

5.2.1 Data Collection

A dataset is a collection of numbers or values that relate to a particular subject. In this project the UNSW_NB_15 dataset is used, that contain the different features about the DDoS attack including ID number, protocol, duration, attack_cat which presents the severity of the DDoS attack.



id	dur	proto	service	state	spkts	dpkts	sbytes	dbytes	rate	sttl	dttl	sload	dload	sloss	dloss	sinpkt	dinpkt	sjit	djit	swin
1	0.000011	udp	-	INT	2	0	496	0	90909.09	254	0	1.8E+08	0	0	0	0.011	0	0	0	
2	0.000008	udp	-	INT	2	0	1762	0	125000	254	0	8.81E+08	0	0	0	0.008	0	0	0	
3	0.000005	udp	-	INT	2	0	1068	0	200000	254	0	8.54E+08	0	0	0	0.005	0	0	0	
4	0.000006	udp	-	INT	2	0	900	0	166666.7	254	0	6E+08	0	0	0	0.006	0	0	0	
5	0.00001	udp	-	INT	2	0	2126	0	100000	254	0	8.5E+08	0	0	0	0.01	0	0	0	
6	0.000003	udp	-	INT	2	0	784	0	333333.3	254	0	1.05E+09	0	0	0	0.003	0	0	0	
7	0.000006	udp	-	INT	2	0	1960	0	166666.7	254	0	1.31E+09	0	0	0	0.006	0	0	0	
8	0.000028	udp	-	INT	2	0	1384	0	35714.29	254	0	1.98E+08	0	0	0	0.028	0	0	0	
9	0	arp	-	INT	1	0	46	0	0	0	0	0	0	0	0	60000.69	0	0	0	
10	0	arp	-	INT	1	0	46	0	0	0	0	0	0	0	0	60000.71	0	0	0	
11	0	arp	-	INT	1	0	46	0	0	0	0	0	0	0	0	60000.69	0	0	0	
12	0	arp	-	INT	1	0	46	0	0	0	0	0	0	0	0	60000.71	0	0	0	
13	0.000004	udp	-	INT	2	0	1454	0	250000	254	0	1.45E+09	0	0	0	0.004	0	0	0	
14	0.000007	udp	-	INT	2	0	2062	0	142857.1	254	0	1.18E+09	0	0	0	0.007	0	0	0	
15	0.000011	udp	-	INT	2	0	2040	0	90909.09	254	0	7.42E+08	0	0	0	0.011	0	0	0	
16	0.000004	udp	-	INT	2	0	1052	0	250000	254	0	1.05E+09	0	0	0	0.004	0	0	0	
17	0.000003	udp	-	INT	2	0	314	0	333333.3	254	0	4.19E+08	0	0	0	0.003	0	0	0	
18	0.00001	udp	-	INT	2	0	1774	0	100000	254	0	7.1E+08	0	0	0	0.01	0	0	0	
19	0.000002	udp	-	INT	2	0	1568	0	500000	254	0	3.14E+09	0	0	0	0.002	0	0	0	
20	0.000004	udp	-	INT	2	0	2054	0	250000	254	0	2.05E+09	0	0	0	0.004	0	0	0	
21	0.00001	udp	-	INT	2	0	2170	0	100000	254	0	8.68E+08	0	0	0	0.01	0	0	0	
22	0.000009	udp	-	INT	2	0	202	0	111111.1	254	0	89777776	0	0	0	0.009	0	0	0	

5.2.2 Data Preprocessing

It is very important and time-consuming part of data analysis. This step is used to clean the information from irrelevant data and convert it to quality information using some statistical techniques to clean data and replace those values which are not important in our experimental analysis. This module consists of three steps. First one is to check whether the missing value is present or not. The second one is to convert the categorical values into numerical values such that UDP as 77 and TCP as 111. And the third step is to replace the ' – ' of service attribute into corresponding values such that for 77 (UDP) it replace the attribute as 0 (DNS) and for 111 (TCP) it replace (5) HTTP. Because the DNS request are generally very small and fit well within UDP protocol and the HTTP requests are work using TCP because it verifies that data is delivered across the network accurately and in the proper sequence.

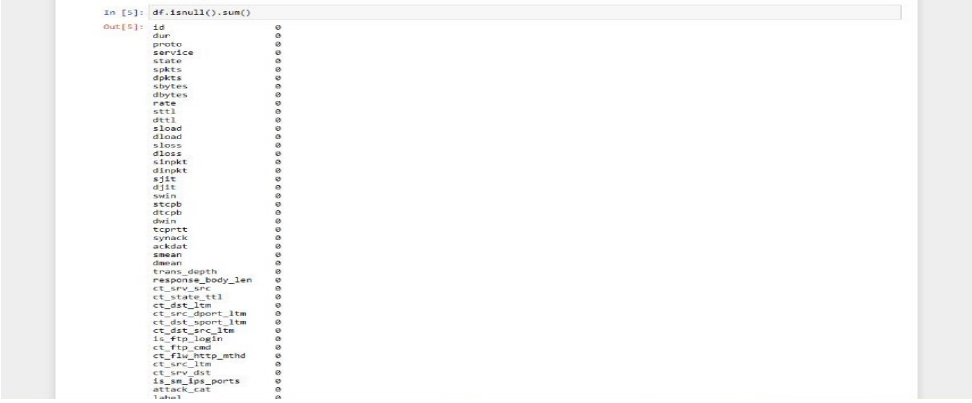


Fig. No. 5.2.2.1 Checking Missing Values

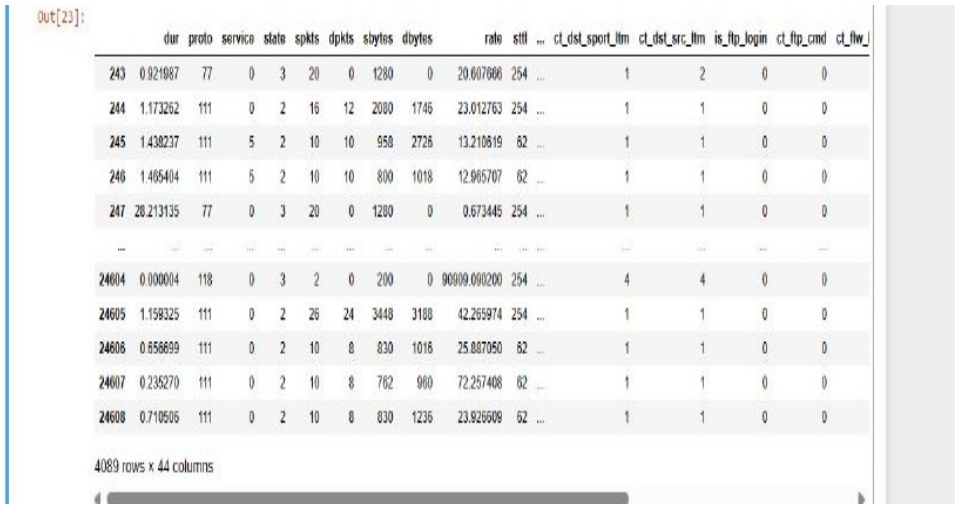


Fig. No. 5.2.2.2 Convert Categorical into Numerical Values

5.2.3 Data Visualization

The present of data where the information will understandable in the form of image or diagram. It is important to understand easily the information. This is the initial step where we are selecting our target for the proposed algorithm. Also, this step is used for selecting the test class. This step is very important to understand data in a much better way. Through this method is able to select our target class for classification. The visualization of showed total number of Normal = 37,000, Generic = 18871, Exploits = 11,132, Fuzzers = 6,062, DoS = 4,089, Reconnaissance = 3,496, Analysis = 677, Backdoor = 583, Shellcode = 378, and Worms = 44 attacks.

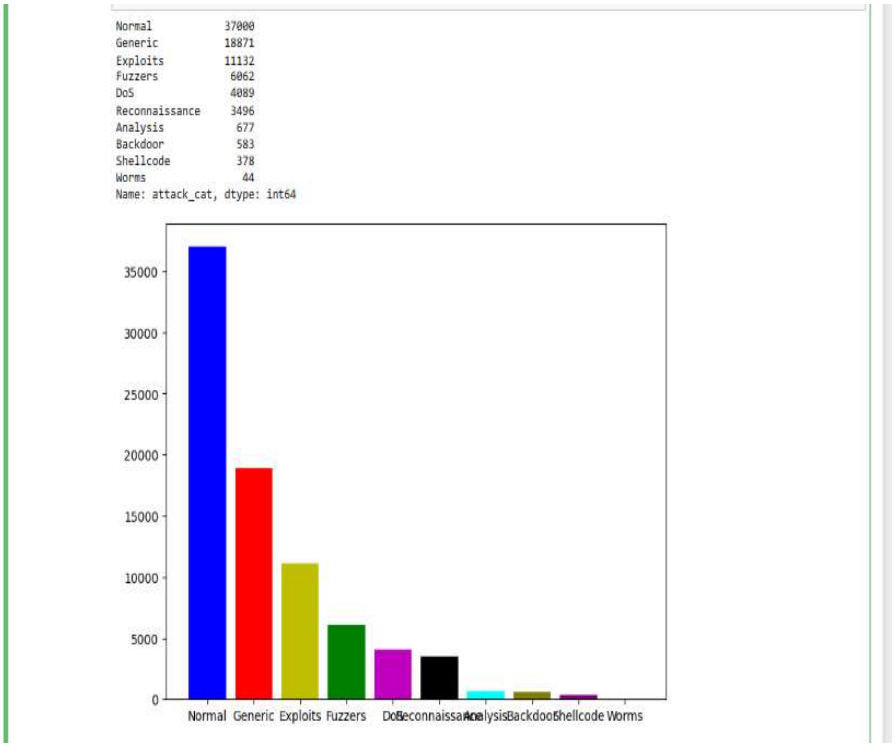


Fig. No. 5.2.3.1 Data Visualization

5.2.4 Train the Model

In this module, it train the training datasets using Random forest algorithm. It consists of two steps. They are classification result and confusion matrix. And finally connecting the web page to the trained model. In classification result it create a combination of decision tree by selecting those important features to predict the DDoS attack.

FEATURES	ATTACK	NORMAL
Total Duration	<=1 sec	>1 sec
Source Bytes	<=1000 bytes	>1000 bytes
Destination Bytes	<=500 bytes	>500 bytes
Source Time to Live	<=254 bytes	>254 bytes
Destination Time to Live	<=252 bytes	>252 bytes

Fig.No.5.2.4.1 Feature Selection

In confusion matrix denotes the overall number of actual and predicted labels for a particular algorithm. It is used to calculate the accuracy of the representation, just like arranging true and prescient marks.

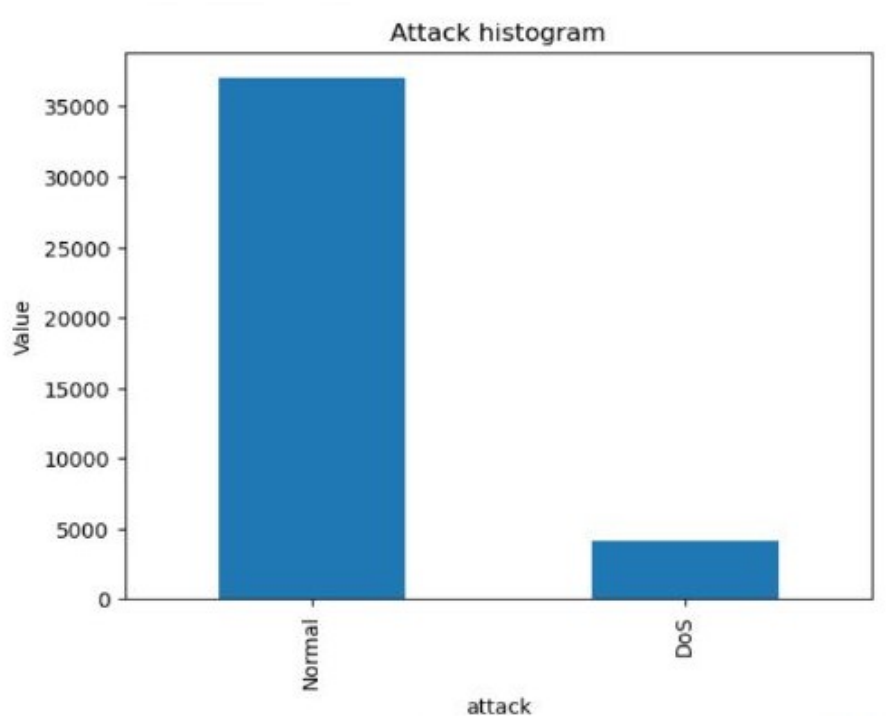


Fig. No. 5.2.4.2 Confusion Matrix

5.2.5 Result Analysis

In this module, first create a web page by extracting some important features from the dataset and then connecting the web page to the trained model using python flask.

Steps for connecting the web page to the flask:

1) Create an instance of Flask: The following statement creates an instance of the Flask application and assigns it to the variable app.

```
app = Flask(__name__)
```

2) Define Routes: A route maps a URL to a Python function that will handle requests to that URL.

```
@app.route('/')
```

3) Run the Application: Run the Flask application using the following code:

```
if __name__ == '__main__':
app.run()
```

4) Connect templates to Flask: To connect the HTML templates to Flask, and create a folder called "templates" in the root directory of the Flask application. In this folder, create a new file for each web page. In each file, include the necessary HTML code, and use Flask's templating engine to add dynamic content.

5) Render templates in Flask: To render the HTML templates in Flask, modify the route functions to return the HTML templates instead of plain text. using Flask's render template function to do this.

```
@app.route('/')
def home():
return render template('index.html')
```

CHAPTER – 6

SYSTEM TESTING

6.1 INTRODUCTION

Testing is a method for determining the functionality of a program. Although there are other types of software testing, the two most common are dynamic and static testing. Static testing is an assessment that takes place when the program is running. Dynamic testing takes place while the software is operating. In a study of the program's code and associated documentation, dynamic and static methodologies are typically mixed. Testing is a standardized activity that may be planned and carried out in a methodical manner. Testing starts with individual modules and develops to complete computer-based system integration.

Management delegated a quality team to verify all relevant paperwork and test the software while entering data at law levels. Various forms of testing are performed during the development process. Each test type is designed to fulfill a distinct testing need.

1. Unit tests
2. Integration tests
3. Functional tests are the most popular types of testing used in the development process.

6.2 VARIOUS FORMS OF TESTING

6.2.1 Unit Testing

Unit testing is a software testing technique where individual units or components of a software application are tested in isolation from the rest of the system to ensure they work as intended. In this process, each unit is tested individually and in an automated manner to verify its behavior and functionality. The goal of unit testing is to identify defects or bugs in the code early in the development process, before they can cause problems in the larger system. Unit tests typically involve testing the smallest possible parts of the code, such as individual functions or methods, to ensure they work correctly in all possible scenarios.

6.2.2 Integration Testing

Integration testing is a software testing technique that focuses on testing the interactions between different components or modules of a software application. The objective of integration testing is to verify that the integrated components work correctly and as expected when they are combined to form a larger system. Integration testing can be done in different ways, such as top-down, bottom-up, or a combination of both. In top-down integration testing, the higher-level modules are tested first, and then the lower-level modules are gradually integrated and tested. In contrast, in bottom-up integration testing, the lower-level modules are tested first, and then the higher-level modules are gradually integrated and tested. A combination of both approaches, known as the sandwich approach, can also be used.

6.2.3 Functional Testing

The primary goal of functional testing is to test the behavior of the system against the functional requirements specified in the software requirements document. During functional testing, the software application is tested to ensure that it performs all the functions that it is intended to perform, and that it does not perform any unintended or unexpected functions. Functional testing is usually performed using a black-box testing approach, where the tester does not have knowledge of the internal workings of the software application. The main objective of functional testing is to identify defects or errors in the software application's functionality and to ensure that the application meets the end-user's requirements and expectations.

Functional testing consists of following components:

Valid Input : identified classes of valid input must be accepted.
Invalid Input : identified classes of invalid input must be rejected.
Functions : identified functions must be exercised.
Output : identified classes of application outputs must be exercised.
Systems/Procedure : interfacing systems or procedures must be invoked.

6.2.4 Test Strategy and Approach

Test strategy and test approach are two essential components of software testing that define the overall testing process and approach to be taken for a software application. Test strategy refers to the overall plan or roadmap for testing a software application. It outlines the objectives, scope, and resources needed for testing, as well as the testing methods, tools, and techniques to be used.

Test strategy also includes the identification of the testing environment, test data requirements, and the roles and responsibilities of the testing team.

6.2.4.1 Objectives for Testing

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.
- Features to be tested

6.2.5 Test Results

Test results refer to the output of the testing process that provides information about the quality and reliability of a software application. Test results provide valuable feedback to the development team about the software's behavior and performance under different conditions and scenarios. Test results play a crucial role in the software testing process, providing valuable feedback to the development team about the software's quality and performance. Accurate and reliable test results are essential to making informed decisions about the software's readiness for release or deployment.

CHAPTER – 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

The system predict the DDoS attacks using the Random Forest Algorithm. In this system, it extract some important features from the UNSW_NB_15 dataset for predicting the DDoS attack. Based on the analysis, it can be concluded that the random forest algorithm is an effective approach for predicting DDoS attacks. The model achieved a high accuracy rate, indicating that it can be effectively distinguish between normal traffic and malicious traffic. This project can achieved the accuracy of 97%. Additionally, feature importance analysis revealed that certain features, such as total duration, source bytes, destination bytes, source time to live and destination time to live were more significant in determining whether traffic was malicious or not. This information can be useful in identifying potential attacks and taking proactive measures to prevent them. Finally, the results suggest that the random forest algorithm is a promising method for predicting DDoS attack and can provide valuable insights into network security.

7.2 FUTURE ENHANCEMENT

The system is implemented with the help of Random Forest Algorithm. In future, the system can work for some more attacks. Extend its prediction accuracy and complexity by combining any other machine learning algorithm to predict the DDoS attack.

APPENDIX 1

CODING

test1.ipynb

```
import pandas as pd

import numpy as np

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.model_selection import train_test_split

# Load the dataset

df = pd.read_csv('./UNSW_NB15_training-set.csv')

import smote_variants as sv

import pandas  as pd

import matplotlib.pyplot as plt

import numpy as np

import imbalanced_databases as imbd

from sklearn import metrics

from sklearn.datasets import load_wine

from sklearn.metrics import roc_curve, auc, roc_auc_score

from sklearn.metrics import confusion_matrix

from sklearn import metrics

from scikitplot.metrics import plot_roc_curve

from imblearn.over_sampling import SMOTE
```

```
%matplotlib inline

from sklearn.model_selection import train_test_split

df.shape

df.attack_cat.unique()

df.isnull().sum()

df.dur.value_counts()

df.dur.describe()

df.spkts.value_counts()[:10]

df.dpkts.value_counts()[:10]

df.sbytes.value_counts()

df.sbytes.describe()

df.rate.describe()

# List of attack categories to drop

attack_categories_to_drop = ['Reconnaissance', 'Backdoor', 'Exploits', 'Analysis', 'Fuzzers',
'Worms', 'Shellcode', 'Generic']

df = df[~df['attack_cat'].isin(attack_categories_to_drop) | df['attack_cat'].isin(['DoS',
'Normal'])]

# Reset the index

df.reset_index(drop=True, inplace=True)

df.head()

df.attack_cat.value_counts()

# Drop irrelevant columns

df = df.drop(['id'], axis=1)

# Replace missing values with NaN
```

```
df.replace(' ', np.nan, inplace=True)

# Convert categorical columns to numeric

df['proto'] = df['proto'].astype('category').cat.codes

df['service'] = df['service'].astype('category').cat.codes

df['state'] = df['state'].astype('category').cat.codes

df.head()

# Fill missing values with appropriate values

df['rate'].fillna(df['rate'].median(), inplace=True)

df['dttl'].fillna(df['dttl'].median(), inplace=True)

df['sload'].fillna(df['sload'].median(), inplace=True)

df['dload'].fillna(df['dload'].median(), inplace=True)

df['dpkts'].fillna(df['dpkts'].median(), inplace=True)

df['spkts'].fillna(df['spkts'].median(), inplace=True)

df.head()

df.to_csv('ddos_P.csv')

!copy ddos_P.csv "D:\po"

df[df['label']==1]

pd.value_counts(df['attack_cat']).plot.bar()

plt.title('Attack histogram')

plt.xlabel('attack')

plt.ylabel('Value')

df['attack_cat'].value_counts()

df[df['proto']==77]
```

model.ipynb

```
import pandas as pd

from sklearn.preprocessing import LabelEncoder

from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import train_test_split

from sklearn import neighbors

from sklearn.metrics import accuracy_score,precision_score,recall_score,f1_score

from sklearn.metrics import confusion_matrix

import joblib

pd.set_option('display.max_rows', None)

pd.set_option('display.max_columns', None)

import warnings

warnings.filterwarnings("ignore")

df = pd.read_csv("ddos_P.csv")

X = df[['dur','sbytes','dbytes','sttl','dttl']]

encoder = LabelEncoder()

y = df[['attack_cat']]

X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=.4)

model=RandomForestClassifier(n_estimators=5, bootstrap=True,
criterion='entropy',max_depth=20,random_state=23)

model.fit(X_train, y_train)

joblib.dump(model, 'model.pkl')

#prediction
```

```
predictions=model.predict(X_test)

print(predictions)

print(accuracy_score(y_test,predictions)*100)

print("Confusion Matrix:",confusion_matrix(y_test,predictions))

print("Precison :",precision_score(y_test,predictions,average='macro'))

print("Recall :",recall_score(y_test,predictions,average='macro'))

print("F1 Score :",f1_score(y_test,predictions,average='macro'))
```

app.py

```
import pandas as pd

from flask import Flask, request, jsonify, render_template

import joblib

app = Flask(__name__)

model = joblib.load(open('model.pkl', 'rb'))

@app.route('/')

def home():

    return render_template('index.html')

@app.route('/predict',methods=['POST'])

def predict():

    '''

    For rendering results on HTML GUI

    '''

    int_features = [x for x in request.form.values()]
```

```
    final_features = pd.DataFrame(int_features).T
    final_features.columns = ['dur','sbytes','dbytes','sttl','dttl']
    print('df',final_features)
    prediction = model.predict(final_features)
    output = prediction[0]
    return render_template('index.html',prediction_text='Category is {}'.format(output))
if __name__ == "__main__":
    app.run()
```

templates / index.html

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <!-- Bootstrap CSS -->
    <link rel="stylesheet"href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css
      /bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6
       cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
    <title>DDoS attack prediction</title>
  </head>
<body>
 <div class="login container text-center" style="margin-top: 100px;">
```

```
<h1 style="margin-bottom: 20px;">Category of DDoS attack</h1>

  <!-- Main Input For Receiving Query to our ML -->

   <form action="{{ url_for('predict')}}" method="post">

<div class="form-group">

  <label for="avg_dur">Total Duration</label>

   <input type="number" name="avg_dur" placeholder="duration" required="required"
step="0.000001" />

</div>

  <div class="form-group">

  <label for="source_byte">Source Bytes</label>

 <input type="number" name="source_byte" placeholder="Source byte"
required="required"/>

</div>

<div class="form-group">

  <label for="Destination_byte">Destination Bytes</label>

    <input type="number" name="Destination_byte" placeholder="Destination byte"
required="required"/>

</div>

<div class="form-group">

  <label for="sttl">Source Time to Live</label>

    <input type="number" name="sttl" placeholder="Stll" required="required"
step="0.000001"/>

</div>

<div class="form-group">
```

```
    <label for="dttl">Destination Time to Live</label>

      <input type="number" name="dttl" placeholder="dttl" required="required"
step="0.000001"/>

  </div>

  <button type="submit" class="btn btn-dark btn-default">Predict</button>

</form>

    <form  action="{{ url_for('predict')}}"method="post">

    </form>

   <br>

   <br>

   {{ prediction_text }}

 </div>

    <!-- Optional JavaScript -->

    <!-- jQuery first, then Popper.js, then Bootstrap JS -->

    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous"></script>

    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x

      0xIM+B07jRM" crossorigin="anonymous"></script>

  </body>
```

```
</html>
```

static / css / style.css

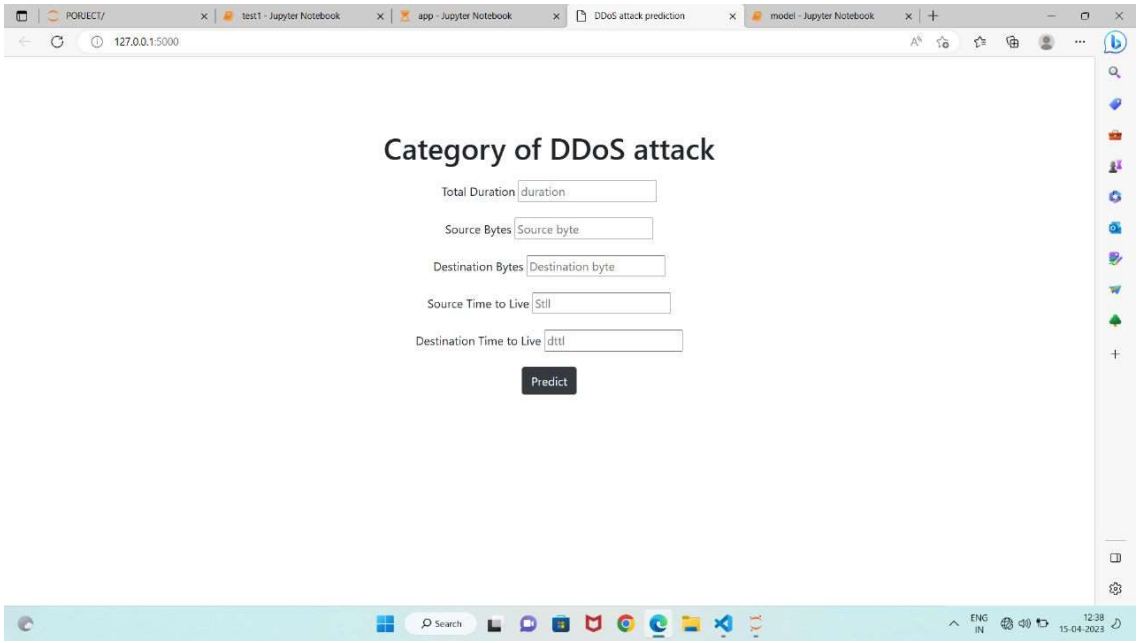
```
html
{
width: 100%; height:100%;
overflow:hidden;
 }
body {
        width: 100%;
        height:100%;
        font-family: 'Helvetica';
        background: #000;
        color: #fff;
        font-size: 24px;
        text-align:center;
        letter-spacing:1.4px;
}
.login {
        position: absolute;
        top: 40%;
        left: 50%;
        margin: -150px 0 0 -150px;
        width:400px;
```

```
        height:400px;
}
.login h1 {
 color: #fff;
 text-shadow: 0 0 10px rgba(0,0,0,0.3);
 letter-spacing:1px;
text-align:center;
}
input {
        width: 100%;
        margin-bottom: 10px;
        background: rgba(0,0,0,0.3);
        border: none;
        outline: none;
        padding: 10px;
        font-size: 13px;
        color: #fff;
        text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
        border: 1px solid rgba(0,0,0,0.3);
        border-radius: 4px;
        box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px
        rgba(255,255,255,0.2);
        -webkit-transition: box-shadow .5s ease;
```

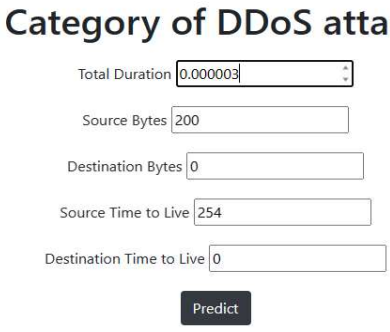
```
        -moz-transition: box-shadow .5s ease;
        -o-transition: box-shadow .5s ease;
        -ms-transition: box-shadow .5s ease;
        transition: box-shadow .5s ease;
}
input:focus
 {
 box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px rgba(255,255,255,0.2);
 }
```

APPENDIX 2

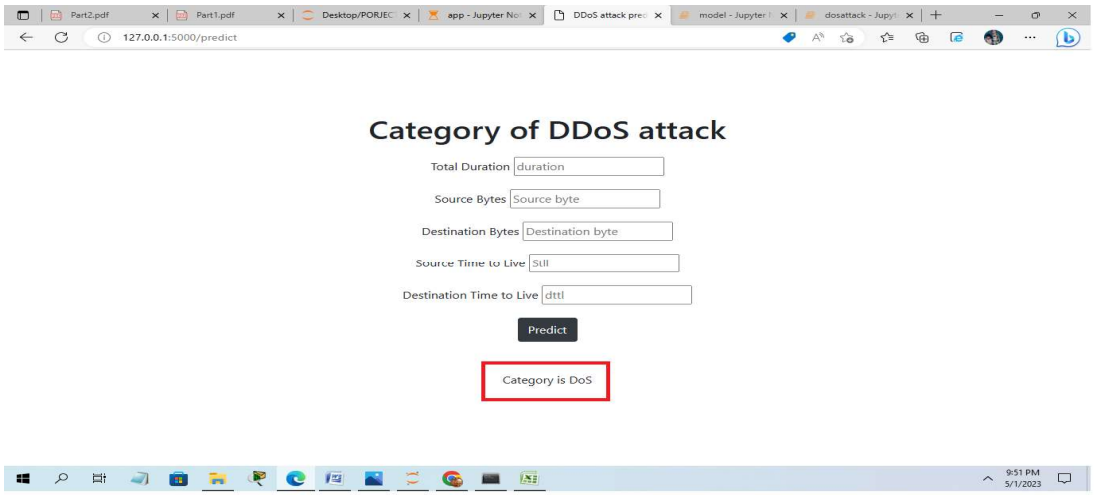
SAMPLE SCREENSHOTS



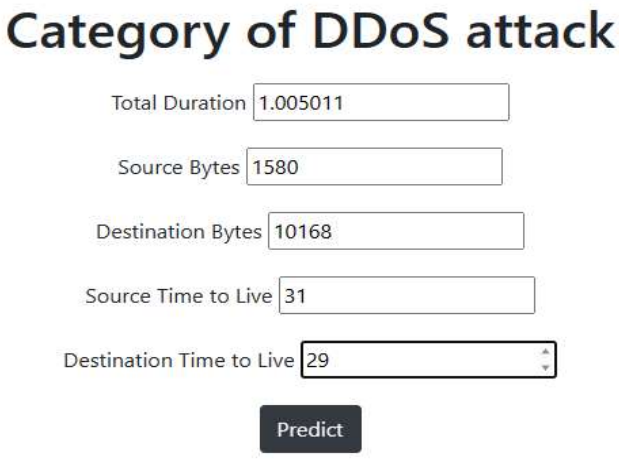
Web Page



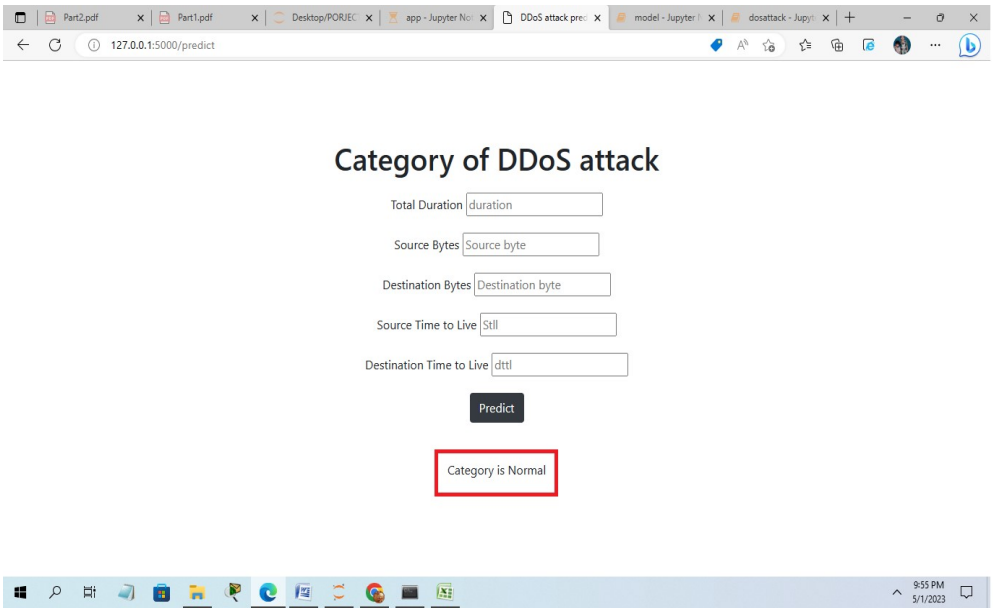
Give Input for predict the attack



Predict the Category is DoS



Give Input for Predict the Normal



Predict the Category is Normal

REFERENCES

[1]Arun Nagaraja et al.,[1] "Hybrid model deep learning model for intrusion detection".

[2] Ashutosh Nath Rimal and Dr.Raja Praveen., "DDoS attack Detection System" using SVM and Naïve Bayes.

[3] Bakker, J. et "DDoS Attack Detection System" using Software Defined Networking (SDN).

[4] Guiomar et al., "Network Intrusion Detection System (NIDS)" using OPNET simulation.

[5] Jing Wu, et al., "DDoS Attack Detection System" using KNN algorithm.

[6] Khuphiran, P et al., "DDoS Attack Detection System" using Support Vector Machine (SVM) and Deep Feed Forward (DFF).

[7] Kimmi kumari and M.Mrunalini., "DDoS Attack Detection System" using various machine learning algorithm.

[8] Larriva-Novo et al.' "proposed two benchmark datasets, especially UGR16 and UNSW-NB15, and the most used dataset KDD99 were used for evaluation".

[9] Liu et al., "DDoS Attack Detection System in deep learning" using CNN algorithm.

[10] Maede Zolanvari et al., "recurrent neural network model for classification intrusion detection".

[11]Mouhammd Alkasassbeh, Ahmad B.A Hassanat, Ghazi AI-Naymat, Mohammad Almseidin., "DDoS Attack Detection System in Data Mining" using Intrusion Detection System.

[12]Nisha Ahuja A et al., "Automated DDOS attack detection in software defined networking".

[13] Norouzian et al., "Classification Technique for Detecting and Classifying DDoS Attacks" in Multilayer perceptron neural network.

[14]Sana Ullah Jan et al.' "PSO-Xgboost model because it is higher than the overall classification accuracy alternative models".

[15] Saini et al., “Detecting DDoS attacks and classifies the type of DDoS attack” using Naïve Bayes algorithm.

[16] Somani, G, Gaur, M.S, Sanghi, D, Conti, M, Buyya, R., “DDoS Attack Detcection System in cloud computing” using Dampster-Shafer Theory.

[17] Wani et al., “Detecting the impact of DDoS attack” using mitigation techniques.

[18] Warrender et al., “Intrusion Detection System” based system call trace data.

[19] Xianwei Gao et al., “comparative work for network traffic classification”.

[20] Zheng et al., “DDoS Attack Detection System” using Hierarchical Intrusion Detection (HIDE) system.