PREDICTION OF INTRUSION ATTACKS USING MACHINE LEARNING APPROACH

A Capstone Project report submitted

in partial fulfilment of requirement for the award of degree

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

by

M.SAIGANESH	**(2003A53011)**
M.KOUSHIK	**(2003A53010)**
P.SHASHI PREETHAM	**(2003A53001)**
T.ARAVIND	**(2003A53014)**

Under the guidance of

Mr. Dr . Eranki L. N. Kiran Kumar

Associate Professor, School of CS&AI.

SR University, Ananthsagar,Warangal,Telagnana-506371

SR University

Ananthasagar, Warangal.

CERTIFICATE

This is to certify that this project entitled **"Prediction of Intrusion Attacks Using Machine Learning Approach**" is the bona fide work carried out by **M. Sai Ganesh, M. Koushik, P. Shashi Preetham, T. Aravind** as a Capstone Project phase-1 for the partial fulfilment to award the degree **BACHELOR OF TECHNOLOGY** in **School of Computer Science and Artificial Intelligence** during the academic year 2023-2024 under our guidance and Supervision.

Dr . Eranki. L. N. Kiran Kumar	**Dr . M. Sheshikala**
Associate Professor,	Associate Professor &
SR University	Head, School of CS&AI,
Anathasagar, Warangal	SR University
	Ananthasagar, Warangal.

Reviewer-1	**Reviewer-2**
Name:	Name:
Designation:	Designation:
Signature:	Signature:

ACKNOWLEDGEMENT

We owe an enormous debt of gratitude to our Capstone project phase-1 guide **Mr. Dr . Eranki L. N. Kiran Kumar, Associate Professor** as well as Head of the School of CS&AI , **Dr . M. Sheshikala, Professor** for guiding us from the beginning through the end of the Capstone Project Phase-1 with their intellectual advices and insightful suggestions. We truly value their consistent feedback on our progress, which was always constructive and encouraging and ultimately drove us to the right direction.

We express our thanks to project co-ordinators **Mr. Sallauddin Md, Asst. Prof., and R. Ashok Asst. Prof.** for their encouragement and support.

Finally, we express our thanks to all the teaching and non-teaching staff of the department for their suggestions and timely support.

ABSTRACT

Technologies advances have improved the systems at the same time also new forms of attacks are noticed in cyber physical systems. Some of them are critical to detect and require through analysis to understand the level of severity, damage caused to the systems. In this paper, we presented different forms of attacks, their severity and possible recommendations have been suggested. We have validated the cyber physical dataset on KDD99 to predict the intrusion attacks using 5 different type of Classifiers. Our results show Logistic Regression and SVM classifiers seem to perform better than others with 90% accuracy. Researchers will find it useful for attack analysis and prepare the recommendations to identify the causality and provide the inputs for predictive analytic. Further, we plan to extend these results to latest intrusion attack system as out future scope of the work.

Keywords: Intrusion detection, DDOS Attack , KDD 1999 Data set,Logistic Regression

Table Of Contents

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

The significance of intrusion detection systems (IDS) in mitigating security risks in computer networks is explained in the text. It makes a distinction between anomaly-based intrusion detection systems (IDS), which find departures from typical network activity, and misuse-based IDS, which finds established assault patterns [5]. Among the difficulties IDS faces is effectively distinguishing between normal and aberrant activity, which calls for the application of data mining techniques like categorization [2]. Studies have shown network attacks such as service denial, attack related to U2R and others are considered to be crucial, as they increase complexity of these attacks. The KDDCUP'99 dataset is the main subject of the analysis for intrusion detection. Most of noted intrusion attacks mitigate security functionalities of the computer networks, targeting information loss or ransom efforts [1]. Open source Knowledge Discovery and Data Mining Tools competition dataset (KDD) is used for the study. Prediction of attack pattern is very complex as they keep changing to infiltrate the systems. There are roughly 4,900,000 separate connections in the dataset. vectors, each of which is identified as having 41 characteristics either typical or an attack, with a single, targeted attack.

Since KDDcup'99 dataset has been extensively utilized to assess anomaly detection techniques. It originates from the U.S.Defense research evaluation program, which collected around 4 terabytes of attack history tcpdump helps to analyze the network. About 4.9 million connection vectors make up the training set; each is classified as either normal or an assault that falls into one of several categories, including DDoS, root attacks, remote login, port probe attacks. Real world attacks have been incorporated into both test and train datasets with specific assault kinds for better analysis. There are fourteen more attack types in the test data and twenty four training attack types in the dataset. The characteristics of KDD'99 are divided into three categories: basic, traffic (same host and same service), and content. The traffic category addresses different aspects of TCP/IP connections, while the content category contains features for identifying particular attack types like R2L and U2R. The diversity in attack types and feature categories contributes to its efficacy in evaluating intrusion detection systems [4]. It is known that the common intrusion detection dataset, KDDCUP9923, has duplicate records, which could distort the outcomes of machine learning algorithms. In order to tackle this issue, supervised machine learning techniques are evaluated using the NSLdataset, which is an enhanced variant of KDDCUP99. Along with 42 features, NSL-KDD24 adds four simulated attacks5. These attacks mainly target the excessive bandwidth such as neptune and others. Similarly others such as root attacks, in which the attacker gains root access using scripts like Perl to gather network information prior to an attack. Other root attacks, include vulnerabilities to obtain local access remotely comprising the network [1]. By reducing redundancy concerns, this dataset seeks to provide a more equitable assessment of machine learning techniques to identify intrusions [3].

2.RELATED WORK

In [3], Considering cyberattacks are occurring more frequently and causing more harm, there is a greater need than ever for intrusion detection systems (IDS) and cybersecurity. The cyber world has advanced significantly due to the growing usage of internet-based services, which makes cybersecurity essential for online operations. Systems for detecting intrusions on networks and hosts (NIDS/HIDS) are the foundation of cybersecurity infrastructure. Scientists frequently use datasets such as NSL-KDD for testing and developing algorithms. This paper uses Python Spyder for simulation to offer a neural network-based method for threat prediction in intrusion detection systems.

In [6] ,The goal of this work was to use Nearest Neighbour Distance Variance (NNDV), a binary class predictor, to detect malicious activity in network traffic. The KDD CUP 99 dataset was used to assess the NNDV classifier, which makes use of variance in object distance. The popular K Nearest Neighbour (KNN) classifier was compared to the prediction accuracy of NNDV, taking into account its binary nature. The outcomes showed that NNDV outperformed KNN in several situations and performed similarly to KNN in others. Furthermore, studies using normalised data produced more accurate NNDV results. To assist in determining the parameters of the algorithm, a variety of cross-validation techniques were also evaluated, such as 2-fold, 5-fold, 10-fold, and leave-one-out.

In [11],The growing threat of IoT-based attacks and the drawbacks of current defence are the subjects of this research, particularly in complex network settings like Software-Defined Networks (SDN) that integrate traditional and IoT protocols. The study presents a Long Short-Term Memory (LSTM) based method for SDNsupported IoT network intrusion detection. With an amazing accuracy of 97.1%, the model's thorough evaluation shows how successful it is at correctly identifying and classifying network threats. The study also makes use of a variety of visualisation approaches to learn more about the properties of the datasets and feature embeddings. Network threats, deep learning, SDN, IoT, intrusion detection, and LSTM are some of the key concepts.

In [4], Real-time threat identification and prevention made possible by AI and ML have completely changed cybersecurity. Sophisticated algorithms enhance cybersecurity defences by analysing data, identifying patterns, and minimising false positives. These technologies also make it easier to create thorough security frameworks that protect sensitive data and important assets from constantly changing threats, improving overall organisational security.

In [5], In this research, a hierarchical approach to machine learning and knowledge-based intrusion detection systems (IDS) is presented. The system predicts attack classes and types and delivers multi-stage hierarchical predictions that distinguish regular connections from attacks. To explore attack taxonomies and choose suitable prediction models, it makes use of knowledge models. The system's performance is evaluated using the KDD 99 dataset, and it presents a viable method for improving network security by integrating machine learning and knowledge modelling with intrusion detection systems.

In [6], The increasing demand for internet use and the growing concerns about
network security are the subjects of this study. It centres on Intrusion Detection Systems (IDS) as a line of defence against ransomware, DDoS, and botnet attacks, among other types of network attacks. On the UNSW-NB15 dataset, three machine learning algorithms—Naïve Bayes, Support Vector Machine, and K-nearest neigh boring — are used to improve intrusion detection accuracy and speed up processing. The goal of the research is to determine which algorithm is best for identifying patterns of suspicious network activity. Confusion matrices and performance metrics like Precision, Recall, and F1-score will be used to help select the best algorithm for predicting and analysing future intrusion behaviour.

In [8], This paper investigates how to tackle the problems caused by large, complex and dynamic intrusion behaviour in network-based traffic data by applying machine learning and data mining techniques. It draws attention to how machine learning can support network security professionals in their work, thereby enhancing information efficiency and data quality. Four supervised learning algorithms are implemented in the study using the WEKA environment: C4.5 Decision tree Classifier (J48), Instance-Based Learning (IBK), Naive Bayes (NB), and Multilayer Perceptron (MLP). The prediction of attack risks is made possible by these algorithms, which are trained using data from the Knowledge Discovery Databases (KDD) for Intrusion Detection. The accuracy of the classification models is evaluated using 10-fold Cross Validation, which supports their practical application in network security and web server environments.

In [2], Research into creating efficient attack detection and prediction systems has increased as a result of the growing threat posed by intrusion attacks on international computer networks. Even with the availability of intrusion detection tools, forecasting network intrusion events is still a difficult field of study. Attack prediction has traditionally relied heavily on statistical techniques, but in recent times, data-intensive regression tasks have shown promise for both shallow and deep learning methods. This work first investigates the application of shallow learning to intrusion prediction, with a particular focus on determining the probability that a malevolent source will launch another attack in the near future. In this context, it highlights the shortcomings of shallow learning and highlights the applicability of deep learning, especially recurrent neural network-based methods, for network alert prediction tasks. The study assesses both strategies.

In [9], Online communication between suppliers and customers has grown in popularity in response to the COVID-19 pandemic. Precise and efficient algorithms are required to improve this communication's efficacy and security. This study presents an intrusion detection system designed specifically for Apache web servers, using the machine learning algorithm Naive Bayes for training. With an impressive cross-validation accuracy of 98.6%, the system demonstrates its potential for reliable and secure online interactions.

In [10], Strong network intrusion detection systems (NIDS) are becoming more and more necessary as a result of the increasing scope of cyberattacks. This paper presents an ensemble machine learning based network intrusion detection system (NIDS) model that can block unknown threats in addition to known attacks. Through the combination of four different machine learning techniques and extensive data processing methods, the model improves the accuracy rate of the intrusion detection system considerably. The effectiveness of this NIDS system over conventional single-machine learning techniques is demonstrated by the simulation analysis, underscoring its significance in contemporary cybersecurity.

3.PROBLEM STATEMENT

The integrity and confidentiality of sensitive data are seriously threatened by the growing sophistication of intrusion attacks in the quickly changing field of cybersecurity. Because these attacks are so dynamic, traditional intrusion detection systems frequently find it difficult to keep up. Thus, investigating and putting into practice cutting-edge machine learning techniques for intrusion attack prediction is imperative. The goal of this project is to create a strong and effective machine learning model that can recognize and anticipate different kinds of intrusion attacks with accuracy, acting as a preventative defense mechanism to protect important systems and data.

4.DATASET CONSTRUCTION

Current research study uses KDD dataset that particularly contains attack scenarios that target both the cyber and physical components of a system in order to address cyber-physical attacks. In addition to network traffic, these datasets may involve the manipulation of sensors, actuators, and control systems. Common pattern of cyber-attacks involve data hack or manipulation of both digital and physical components. The KDD Cup 1999 dataset does not specifically model attacks on cyber-physical systems, even though its primary focus is network intrusion detection. A variety of attack scenarios, including denial of services, phishing of access authorization both at root and local access assaults, are included in the dataset. Based on KDDCUP Dataset we have created our own dataset with possible recommendations. We shall now about recommendations in detail While there are many different kinds of counterattacks in the KDD dataset, Features provided in the dataset include network intrusion detection levels and does not specifically model cyber-physical attacks. However, the dataset encompasses a number of attack categories within the context of network invasions [4]. Among the assault types identified, some of them are discussed below:

• Service Denials: The goal of this kind of attack is to prevent users from accessing a network or service. You can find examples of DoS attacks—attackers attempting to overwhelm the targeted system with a deluge of traffic.

• Investigation (or Watchfulness): An unauthorized user trying to learn more about a system or network is known as a probing attack. To find potential vulnerabilities, these operations could involve port scanning or other reconnaissance methods.

• User-to-Root (U2R): U2R attacks begin with ordinary user access, and the attacker attempts to employ security flaws to obtain administrative or root privileges. This can entail abusing the system's flaws to increase their privileges.

• Remote-to-Local (R2L) attacks: R2L attacks gain remote access of the target system locally from a distance. These assaults frequently go at weak points in the system that can provide illegal access.

• Normative (or benign): The collection also contains examples of typical network behaviour. It is essential to comprehend what regular behaviour is in order to spot anomalies and identify potentially dangerous activity. It is also called Normal.

Once identification of attack type has been analysed, one can look for possible recommendations and preventive measures based on the type of attacks. Studies also show that as attacks keep changing their pattern of infiltration to target systems, physical access to networks and application also need to be improved to control the vulnerabilities of the existing systems [6].

We can further analyze the attack into categories as discussed below.

• Normal Traffic Recommendation: Since normal traffic is what you want to identify as a baseline, it's important to understand the typical patterns in your network. Employ anomaly detection methods like clustering or statistical analysis to identify deviations from the norm.

Precautions: Ensure that your training data for normal traffic is representative and up-to-date. Also, consider using additional network monitoring tools to enhance your understanding of normal network behaviour.

• Denial of Service (DoS) Attacks: Recommendation: Develop a robust DoS attack detection algorithm that can quickly identify and mitigate these attacks. This may involve monitoring for unusually high traffic or analysing packet patterns. Precautions: Be cautious of false positives, as legitimate traffic spikes can sometimes mimic DoS attacks. Continuously update your detection methods to adapt to evolving attack techniques.

• Probe Attacks: Recommendation: Employ network intrusion detection systems (NIDS) to monitor and detect probe attacks. These attacks typically involve scanning and probing activities, which can be detected by monitoring for unusual port scanning patterns.

Precautions: Ensure that you have adequate logging and monitoring in place to capture and analyze probe activities. Consider setting up alerting mechanisms to respond promptly to probe attacks.

• User-to-Root (U2R) Attacks: Recommendation: Develop machine learning models that can identify unauthorized user escalation at tempts. Feature engineering and behavioural analysis can help detect unusual user actions.

Precautions: Be aware that U2R attacks can be subtle and hard to detect. Continuously update your detection models to account for new attack vectors.

• Remote-to-Local (R2L) Attacks: Recommendation: Implement a system that monitors for unauthorized remote access attempts. This may involve analysing login failures, authentication logs, and patterns of access.

Precautions:R2L attacks can be challenging to detect, as they may appear as legitimate access attempts. Combine signature-based detection with anomaly detection to improve accuracy.

4.1 Features Extraction and Attack Analysis

One important observation is that the test data differs from the training data in terms of probability distribution. Studies also show novel attacks are really just repackaged versions of popular ones, and you can occasionally tell which ones are new simply by examining their signature. As shown in Figure 1 and Table 1 of the KDD dataset. We have further grouped KDD'99 features into three categories is possible:

1) Fundamental features: Attacks based on TCP/IP port comprises which usually delay in implicit detection of these attack.

2) Based on how they are calculated in connection to a window interval, traffic-related features are classified as follows:

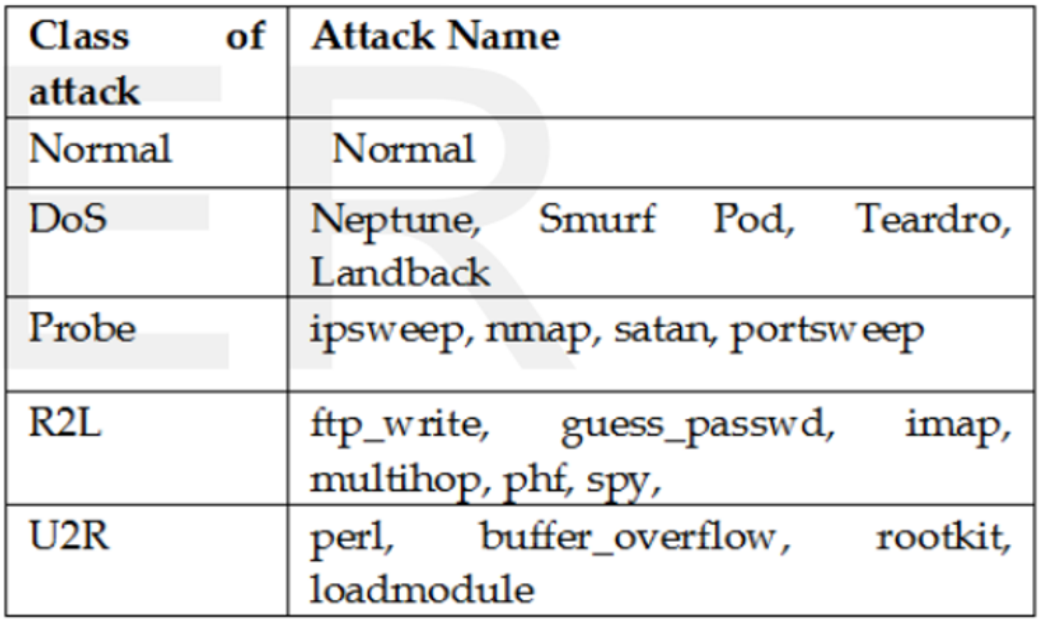
a) Identical hosts properties where connections are established with identical destination host as the active session. Computed statistics would reveal the session behaviour necessary.

b) Identical services option only considers connections that were established in the last two seconds and offer the same service as the one you are now utilizing.

c) The two types of ”traffic” characteristics that were previously addressed are time-based attributes. However, certain sluggish probing attacks require more time to scan the ports and review the logs. One such assault requires one minute.

d) Apart from these, connection time attacks are also evident in some studies [6].

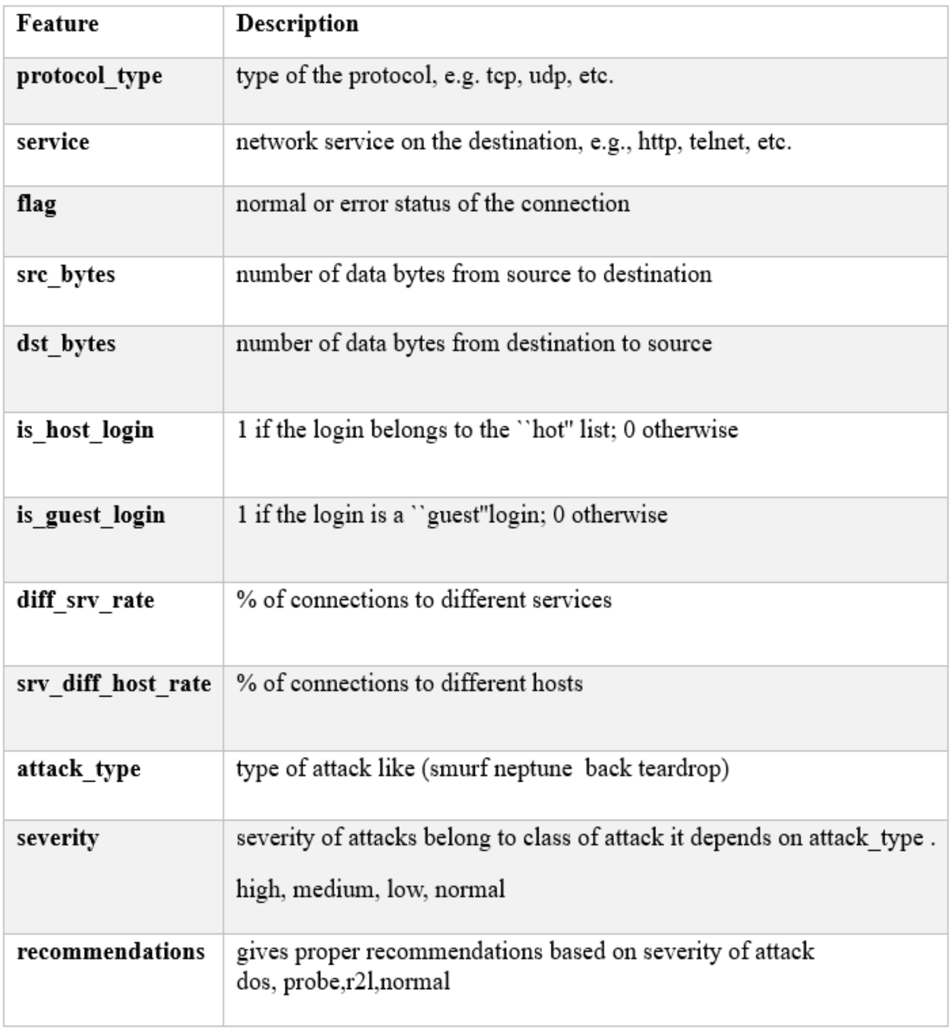
e) Attacks related to content of the target systems examine the data area for anomalous activity. These features are referred to as content characteristics [7].



Class of attack	Attack Name
Normal	Normal
DoS	Neptune, Smurf Pod, Teardro, Landback
Probe	ipsweep, nmap, satan, portsweep
R2L	ftp_write, guess_passwd, imap, multihop, phf, spy,
U2R	perl, buffer_overflow, rootkit, loadmodule

Fig.1.Class of Attack and Attack type.

The above Figure contains different types of attacks. each attack is different from one another. but few of the attacks fall under same class. We will be extracting only few features from the official dataset to build the model and the features which we selected based on the key features of the dataset.



Feature	Description
protocol_type	type of the protocol, e.g. tcp, udp, etc.
service	network service on the destination, e.g., http, telnet, etc.
flag	normal or error status of the connection
src_bytes	number of data bytes from source to destination
dst_bytes	number of data bytes from destination to source
is_host_login	1 if the login belongs to the ``hot'' list; 0 otherwise
is_guest_login	1 if the login is a ``guest''login; 0 otherwise
diff_srv_rate	% of connections to different services
srv_diff_host_rate	% of connections to different hosts
attack_type	type of attack like (smurf neptune back teardrop)
severity	severity of attacks belong to class of attack it depends on attack_type . high, medium, low, normal
recommendations	gives proper recommendations based on severity of attack dos, probe,r2l,normal

Table.1 Dataset Description

5. PROPOSED SOLUTION

We propose a state of the art intrusion attack prediction model using the KDD dataset and its features to predict the attacks. We attempt to answer the research questions using the prediction model. Out implementation comprise of two phases and each phase has specific set of tasks to perform and analyze the features of the dataset [1].

5.1 Phase-1

• Collection of KDDCUP dataset from the official website.

• Preprocess the data set .

• Add the extra columns for the dataset.

5.2 Phase-2

• Developing the model takes places in this phase .

• Trying different algorithms in order to get proper accuracy of model. We have used hybrid model approach.

• We are taking inputs from the user and we will be predicting the attack first, I mean attack type column from the dataset.

• From 1st column to 9th column is used to predict 10th column in the dataset. With the data from 1st column to 10th column we are going to predict the 11th column .

• From the 1st column in the dataset to 11th column we are going to predict the recommendations. So we are doing 3 predictions and we have developed 3 pickle files for integration to develop the web application as shown in Figure 5.

5.3 Phase-3

• Django is used for integration of pickle files and user interface.

• High level web framework Django is written in the Python programming language and is highly secure. It adheres to the MVC architectural pattern (model view controller).Models, views, templates, URL routing, admin interface, and other features are some of the main ones. Django is a popular choice for Python web development because of its ease of use, versatility, and scalability.

5.4 Flow-Diagram of Model

• Generate user login page for the application.

• Page to obtain input from the user.

• Validate and authorize user login with submit button.

• All the inputs are given to a Machine Learning Model and if it able to predict the attack then it will move into next block otherwise it moves to end of flow chart and detect as Normal .

• If the attack is detected then it should able to predict the attack belong to which class and provide proper recommendations as shown in Figure 4.

6.DIAGRAMS

6.1UML DIAGRAMS

a) Use-Case Diagram:

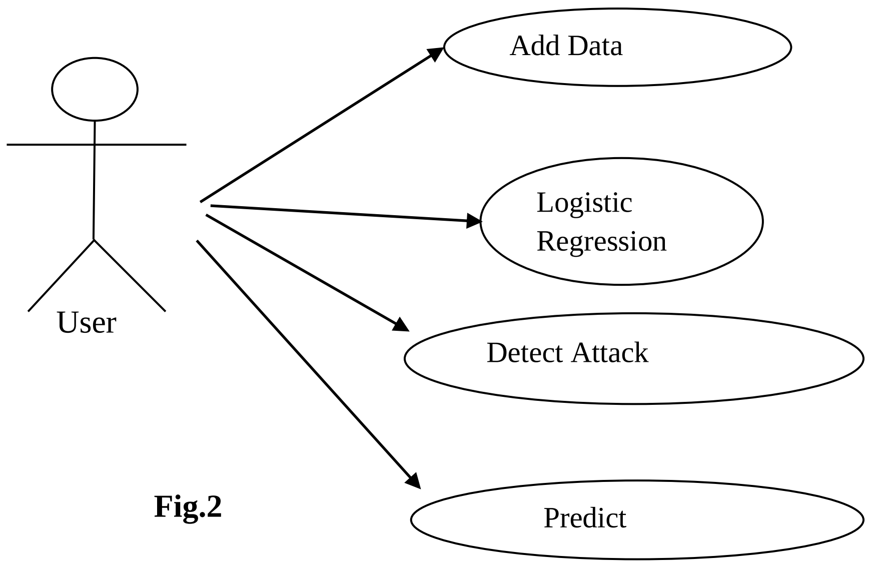


Fig.2

b) Sequence Diagram:

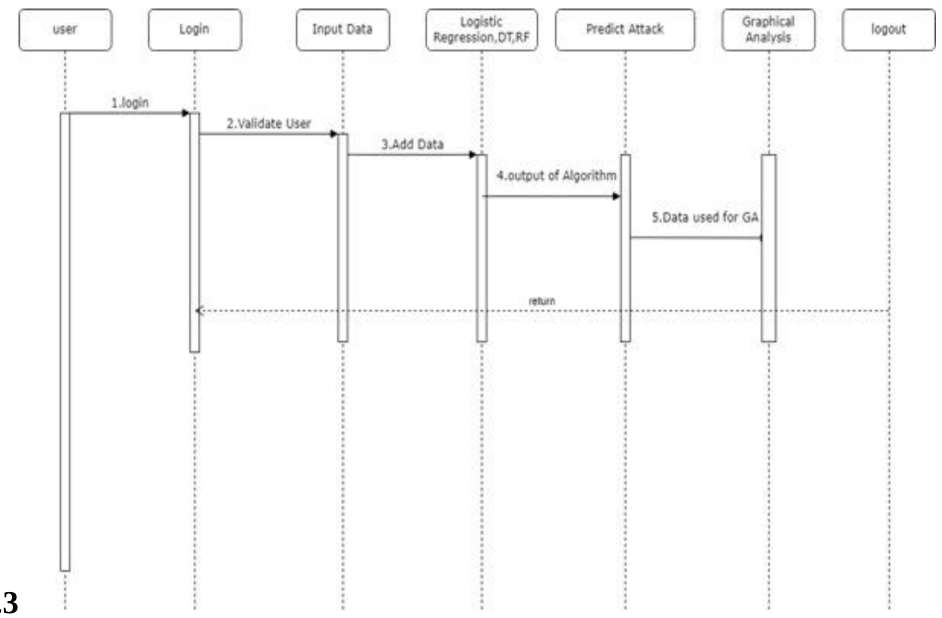


Fig.3

6.2FLOW CHART

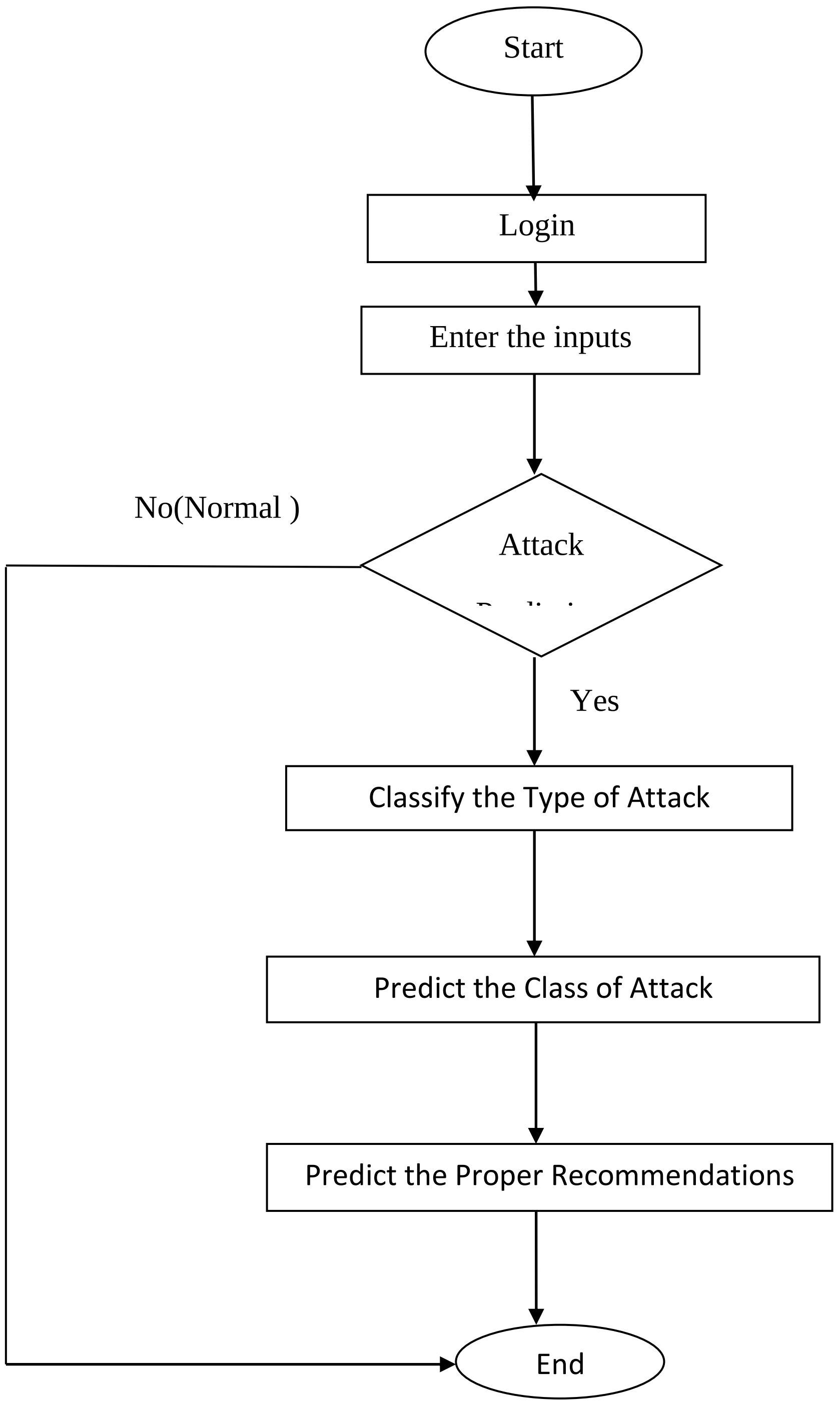
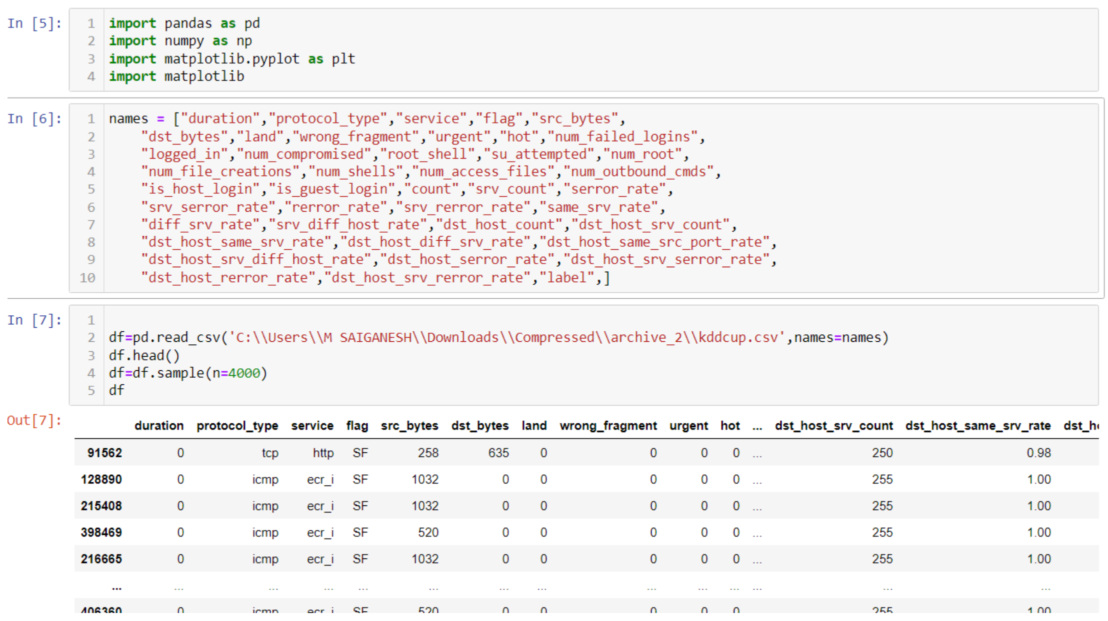

Start
Login
Enter the inputs
No(Normal)
Attack
Yes
Classify the Type of Attack
Predict the Class of Attack
Predict the Proper Recommendations
End

Fig.4

7.IMPLEMENTATION

Step 1: Collection of dataset

For our study, we are using the KDD (knowledge discovery in databases) 99 dataset. It has 4898431 rows and 42 columns.

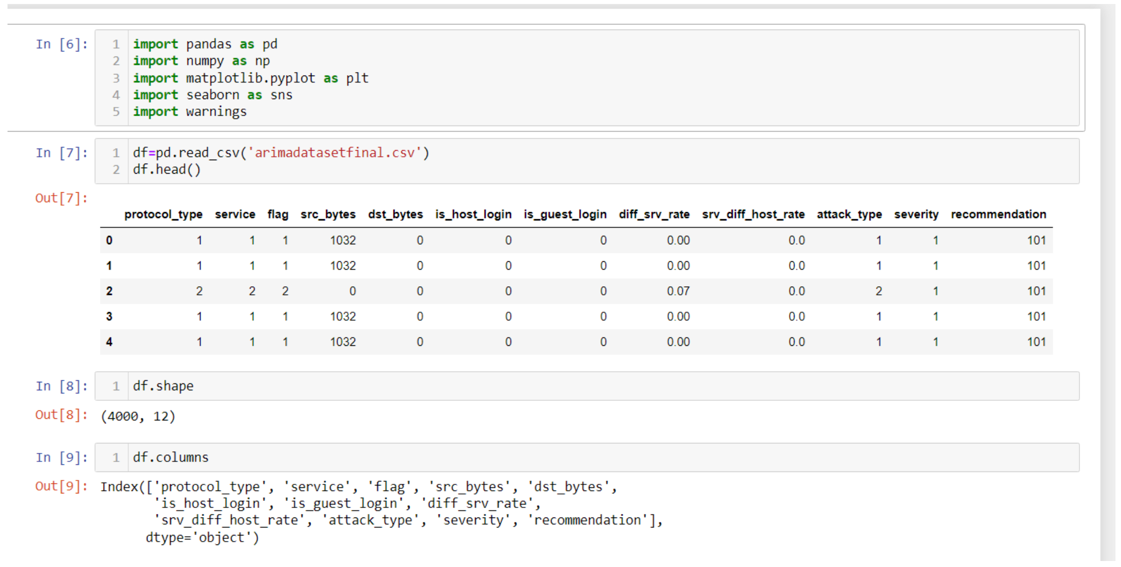


```
In [5]:  import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         import matplotlib

In [6]:  names = ["duration","protocol_type","service","flag","src_bytes",
             "dst_bytes","land","wrong_fragment","urgent","hot","num_failed_logins",
             "logged_in","num_compromised","root_shell","su_attempted","num_root",
             "num_file_creations","num_shells","num_access_files","num_outbound_cmds",
             "is_host_login","is_guest_login","count","srv_count","serror_rate",
             "srv_serror_rate","rerror_rate","srv_rerror_rate","same_srv_rate",
             "diff_srv_rate","srv_diff_host_rate","dst_host_count","dst_host_srv_count",
             "dst_host_same_srv_rate","dst_host_diff_srv_rate","dst_host_same_src_port_rate",
             "dst_host_srv_diff_host_rate","dst_host_serror_rate","dst_host_srv_serror_rate",
             "dst_host_rerror_rate","dst_host_srv_rerror_rate","label",]

In [7]:  
         df=pd.read_csv('C:\\Users\\M SAIGANESH\\Downloads\\Compressed\\archive_2\\kddcup.csv',names=names)
         df.head()
         df=df.sample(n=4000)
         df

Out[7]:
```

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_srv_count	dst_host_same_srv_rate	dst_h
91562	0	tcp	http	SF	258	635	0	0	0	0	...	250	0.98	
128890	0	icmp	ecr_i	SF	1032	0	0	0	0	0	...	255	1.00	
215408	0	icmp	ecr_i	SF	1032	0	0	0	0	0	...	255	1.00	
398469	0	icmp	ecr_i	SF	520	0	0	0	0	0	...	255	1.00	
216665	0	icmp	ecr_i	SF	1032	0	0	0	0	0	...	255	1.00	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
406360	0	icmp	ecr_i	SF	520	0	0	0	0	0		255	1.00	

Step 2: Data preprocessing

The KDD dataset has been pre-processed in this case. After preparing the KDD99 dataset, we had 4000 rows and 12 columns. Three columns have been added: assault kind, severity, and recommendations. Convert all the data into numerical values in the dataset for prediction. checked for correlation matrix.



```
In [6]:  import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         import seaborn as sns
         import warnings

In [7]:  df=pd.read_csv('arimadatasetfinal.csv')
         df.head()

Out[7]:
```

	protocol_type	service	flag	src_bytes	dst_bytes	is_host_login	is_guest_login	diff_srv_rate	srv_diff_host_rate	attack_type	severity	recommendation
0	1	1	1	1032	0	0	0	0.00	0.0	1	1	101
1	1	1	1	1032	0	0	0	0.00	0.0	1	1	101
2	2	2	2	0	0	0	0	0.07	0.0	2	1	101
3	1	1	1	1032	0	0	0	0.00	0.0	1	1	101
4	1	1	1	1032	0	0	0	0.00	0.0	1	1	101

```
In [8]:  df.shape

Out[8]:  (4000, 12)

In [9]:  df.columns

Out[9]:  Index(['protocol_type', 'service', 'flag', 'src_bytes', 'dst_bytes',
                'is_host_login', 'is_guest_login', 'diff_srv_rate',
                'srv_diff_host_rate', 'attack_type', 'severity', 'recommendation'],
               dtype='object')
```

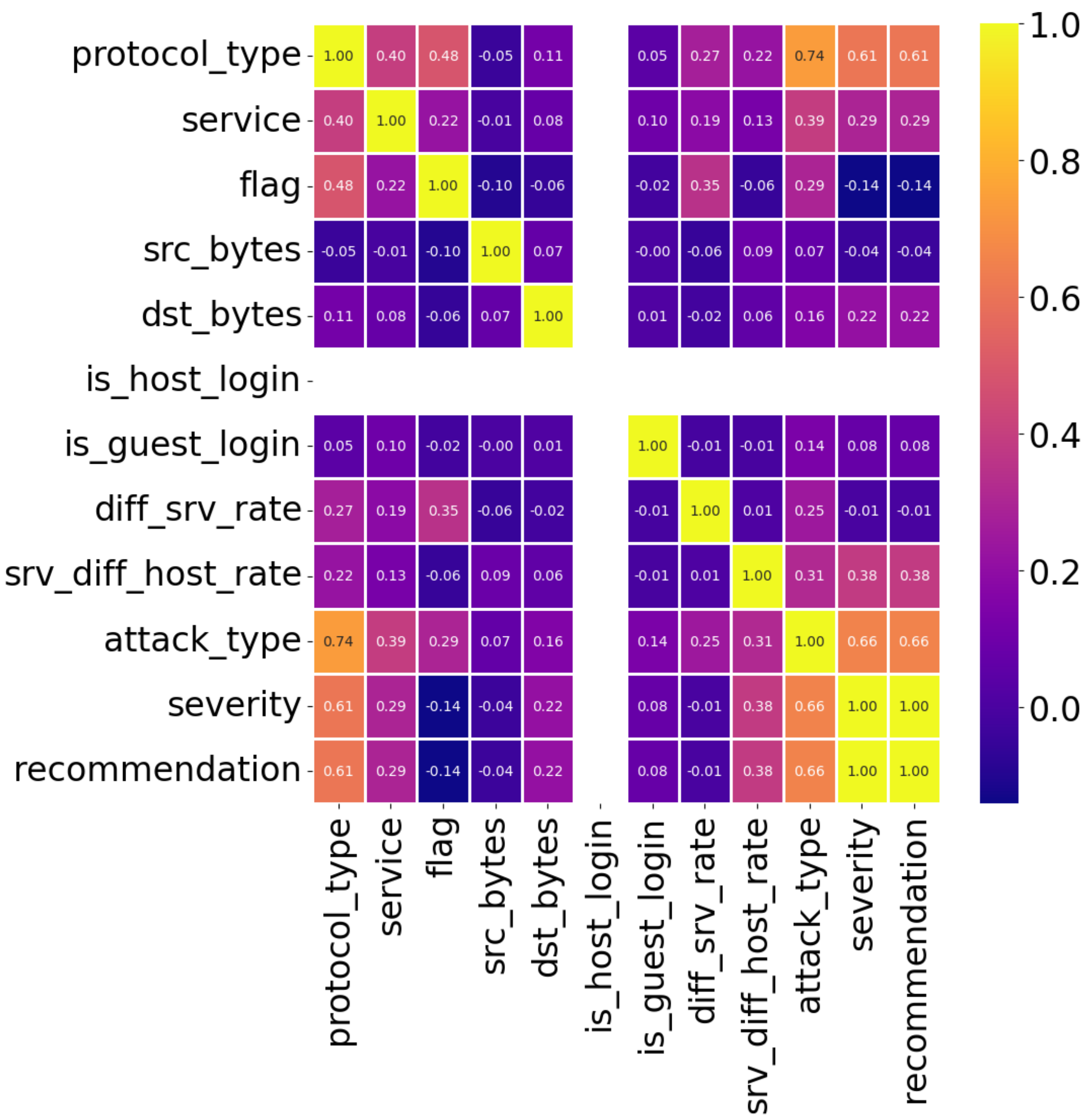


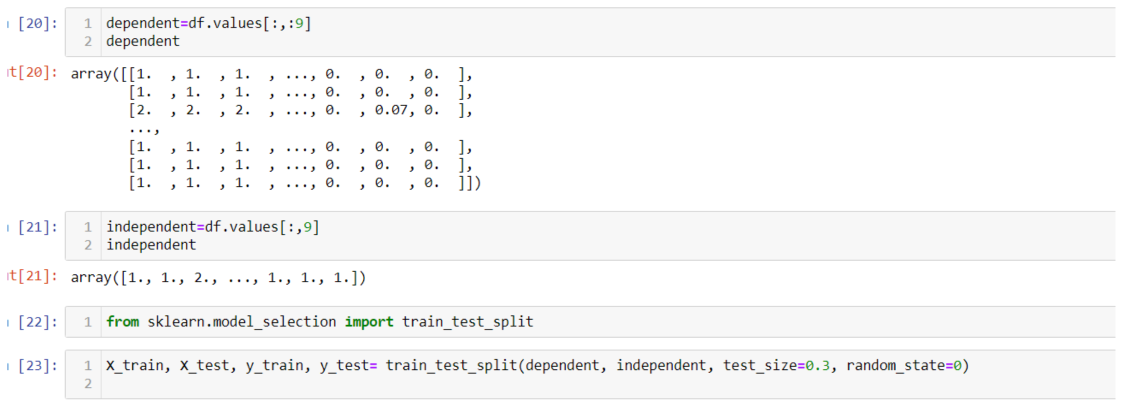
Fig.5

Step 3: Splitting of Data

Here we have splitted 4000 rows, of which 2800 are for training and 1200 are for testing.

Step 4: Implementation of algorithm

Here we apply a classification algorithm for our dataset. Here we tested support vector machine (SVM), Random Forest, Decision Tree and logistic regression algorithms with our dataset. In which we choose the logistic regression algorithm which will fit for our model due to its accuracy of 97%.



```
[20]: dependent=df.values[:,:9]
      dependent
```

```
[20]: array([[1.  , 1.  , 1.  , ..., 0.  , 0.  , 0.  ],
             [1.  , 1.  , 1.  , ..., 0.  , 0.  , 0.  ],
             [2.  , 2.  , 2.  , ..., 0.  , 0.07, 0.  ],
             ...,
             [1.  , 1.  , 1.  , ..., 0.  , 0.  , 0.  ],
             [1.  , 1.  , 1.  , ..., 0.  , 0.  , 0.  ],
             [1.  , 1.  , 1.  , ..., 0.  , 0.  , 0.  ]])
```

```
[21]: independent=df.values[:,9]
      independent
```

```
[21]: array([1., 1., 2., ..., 1., 1., 1.])
```

```
[22]: from sklearn.model_selection import train_test_split
```

```
[23]: X_train, X_test, y_train, y_test= train_test_split(dependent, independent, test_size=0.3, random_state=0)
```

7.1 Algorithms And Techniques

7.1.1Random Forest

• Random forest uses multiple decision tress to ensemble the classification process using cross validation and branching factor.

• Algorithm uses tasks that include regression and classification as a result of its high scalability, robustness, feature importance, versatility, and accuracy.

• The Random Forest algorithm's implementation is justified by the following arguments: It involves less training time than other algorithms. It predicts output with high accuracy and performs well even with a large dataset.

It can continue to be accurate even in cases where a significant portion of the data is absent. Mathematical form of decision tree model, which is a component of a Random Forest, can be represented as follows Yˆ = Tree(X; θ) (1) Considering the factor that Yˆ is the predicted output, Tree represents the decision tree model, X is the input feature vector, θ represents the parameters or structure of the decision tree.

7.1.2 Logistic Regression

• Logistic regression determines model best fit by describing non-linear distribution of data. To determine the likelihood occurrence of attacks.

• This algorithm is applied to determine the dependency among the variables and their effect on prediction analysis.

• Implementation factors applied in the model are shown in Equation 2.

P(Logreg) = 1/(1 + e−Z) Where Z = α0 + α1X1 + α2X2 + . . . + αnXn (2)

7.1.3 Linear Regression

• Using linear regression analysis, which predicts the gradient slope fit to determine the loss.

• The variable you want to predict, or the independent variable you are using to predict the other variable's value, is the dependent variable.

• Features considered as independent and dependent variable are applied by the algorithm to predict future events. We can analyze the linear regression using the Equation 3 $Y = \beta 0 + \beta 1X + E$ (3) While Y is the output to be predicted, given the X as slope, E as error or bias factor of the regression.

7.1.4.Support Vector Machine

• Support Vector classifier has a broad range of applications, including linear or nonlinear classification, regression, outlier detection, handwriting recognition, text and image classification, anomaly detection, face detection, spam detection, and gene expression analysis.

• Benefits of SVM perform well in multidimensional scenarios. Its memory is efficient because it uses vectors which determine the decision boundaries of the data. Different kernel functions and custom kernels can be specified for the decision functions.

• The SVM algorithm looks for the best line, or decision boundary, that can separate n-dimensional space into classes. Known as a hyperplane, this optimal decision boundary helps to improve accuracy. The equation for a linear support vector machine as shown in Equation 4 $f(x) = \text{sign}(w \cdot x + b)$ (4) We can analyze using the following terms. f(x) provides decision function, w is the weight vector, x is the input feature vector, b is bias term.

7.1.5 Decision Tree

• Decision Tree classifier supports both regression and classification tasks.

• Its hierarchical tree structure is composed of the internal nodes, leaf nodes, branches, and root node.

• For example, a decision tree is a flowchart that helps someone choose clothes based on the weather.

• Decision Tree classifier can be implemented using the Equation 5 A decision tree can be represented by a set of decision rules: If $F1 \le \alpha 1$ and $F2 \le \alpha 2$ then Predict class A Else if $F1 > \alpha 1$ and $F3 \le \alpha 3$ then Predict class B Else Predict class C (5) F1, F2, F3, . . . are the input features, α1, α2, α3, . . . are the decision thresholds.

7.1.6 Gaussian Naive Bayes

• Gaussian Naive Bayes is a probabilistic classification algorithm that relies on using the Bayes theorem under strong independence assumptions.

• An algorithm for probabilistic machine learning is called Gaussian. Numerous applications for classification use Naive Bayes. Naive Bayes is widely applied to document classification, spam filtering, and prediction.

• The supervised learning algorithm Gaussian Naive Bayes is applied to classification tasks. The Gaussian Naive Bayes classifier predict a x belonging to class Ci as follows

$$P(C_i|\mathbf{x}) = \frac{1}{Z}P(C_i)\prod_{j=1}^{n}\frac{1}{\sqrt{2\pi\sigma_{ij}^2}}\exp\left(-\frac{(x_j-\mu_{ij})^2}{2\sigma_{ij}^2}\right)$$

Given P(Ci |x) as posterior probability of class Ci , Z is normalized factor for µij and σ 2 ij mean and variance of features.

7.1.7 Gradient Boost

• Regression and classification tasks are performed by machine learning techniques like gradient boosting.

• It offers a prediction model in the form of an ensemble of weak prediction models, or models that make relatively few assumptions about the data and typically take the form of simple decision trees.

• One application of gradient boosting is in learning to rank. Yandex and Yahoo, two commercial web search engines, use gradient boosting variants in their machine learning algorithms.

The update equation for a regression problem using Gradient Boosting is given by:

$$F(x) = F_{\text{prev}}(x) + \eta \cdot h(x;\theta) \qquad (7)$$

i.e.,

$F(x)$ is the final boosted model,

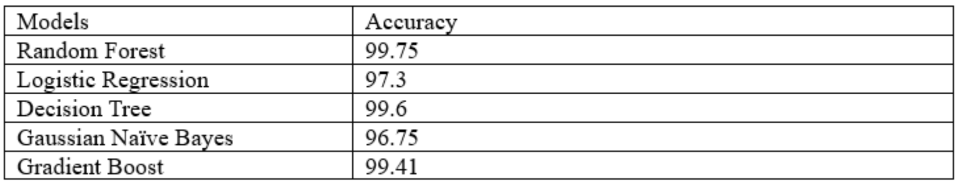
$F_{\text{prev}}(x)$ is the model from the previous iteration,

η is the learning rate,

$h(x;\theta)$ is the weak learner (e.g.,DT) with parameters θ.

Step 5: Training and testing of data

Here we train and test our data using logistic regression algorithm where 3800 rows are for training and 200 rows are for testing. There are two types of values in our data set from column 1 to 9 are independent values and other three columns are dependent values they are the outcomes of their previous column values. There are 4 variables after dividing the data into training and testing they are X_train ,X_test which are independent training and testing data , Y_train and Y_test are dependent training and testing data.



Models	Accuracy
Random Forest	99.75
Logistic Regression	97.3
Decision Tree	99.6
Gaussian Naïve Bayes	96.75
Gradient Boost	99.41

Table.2 Accuracy on Various Algorithms

We have used the State Of The Art Model for prediction of Attack type, Class of Attack, Recommendations.

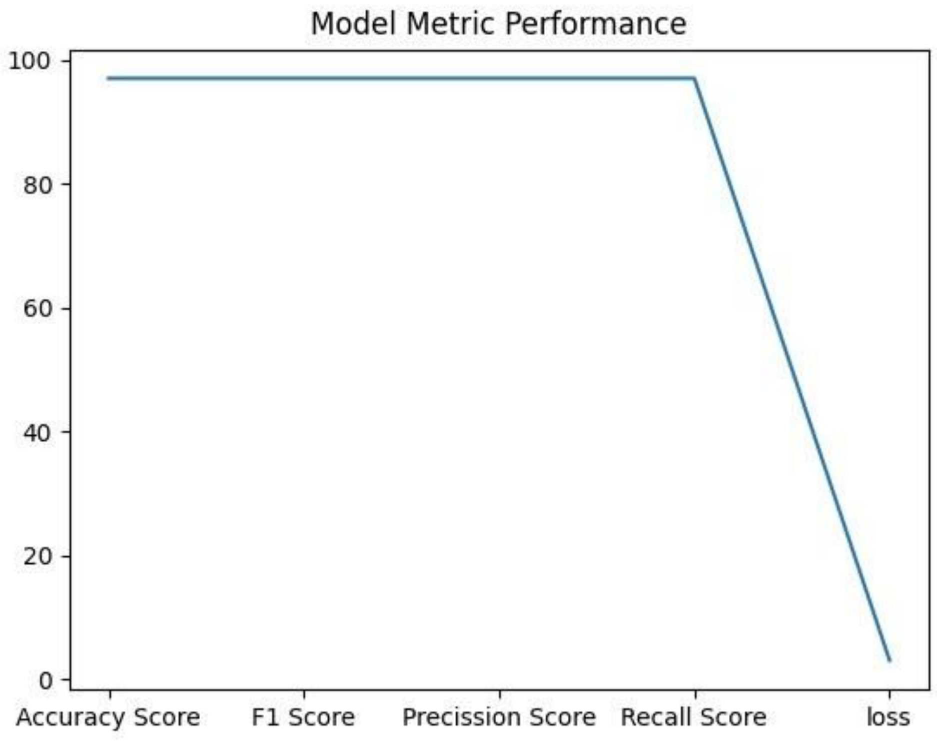


Fig.6 Model Metric Test Data

Step 6: Implementation of Django framework

Here we implement Django framework with our model which provides a user friendly interface for our users.

We have developed few Web Pages for the project.

7.2 Source-code:

Index.html :

```
<!DOCTYPE html>
{% load static %}
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        body{
        background: url("{% static 'images1.jpg'%}") ;

 background-size:cover;
        }

form {
  position: relative;
  width: 250px;
  margin-top: 150px ;
  margin-left:500px;
  background: rgba(130,130,130,.8);
  padding: 20px 22px;
  border: 1px solid;
  border-top-color: rgba(255,255,255,.4);
  border-left-color: rgba(255,255,255,.4);
  border-bottom-color: rgba(60,60,60,.4);
  border-right-color: rgba(60,60,60,.4);
}

form input, form button {
  width: 212px;
  border: 1px solid;
  border-bottom-color: rgba(255,255,255,.5);
  border-right-color: rgba(60,60,60,.35);
  border-top-color: rgba(60,60,60,.35);
  border-left-color: rgba(80,80,80,.45);
  background-color: rgba(0,0,0,.2);
  background-repeat: no-repeat;
  padding: 8px 24px 8px 10px;
  font: bold .875em/1.25em "Open Sans Condensed", sans-serif;
  letter-spacing: .075em;
  color: #fff;
  text-shadow: 0 1px 0 rgba(0,0,0,.1);
  text-color:blue;
  margin-bottom: 19px;
}

form input:focus { background-color: rgba(0,0,0,.4); }

form input.email {
  background-image:
url(data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAAA8AAAAMCAY
AAAC9QufkAAAAGXRFWHRTb2Z0d2FyZQBBZG9iZSBJbWFnZVJlYWR5ccllP
AAAAyJpVFh0WE1MOmNvbS5hZG9iZS54bXAAAAAAADw/eHBhY2tldCBiZW
dpbj0i77u/IiBpZD0iVzVNME1wQ2VoaUh6cmVTek5UY3prYzlkIj8+IDx4OnhtcG1l
dGEgeG1sbnM6eD0iYWRvYmU6bnM6bWV0YS8iIHg6eG1wdGs9IkFkb2JlIFhNU
CBDb3JlIDUuMC1jMDYxIDY0LjE0MDk0OSwgMjAxMC8xMi8wNy0xMDo1Nzo
wMSAgICAgICAgIj4gPHJkZjpSREYgeG1sbnM6cmRmPSJodHRwOi8vd3d3LnczL
m9yZy8xOTk5LzAyLzIyLXJkZi1zeW50YXgtbnMjIj4gPHJkZjpEZXNjcmlwdGlvbi
ByZGY6YWJvdXQ9IiIgeG1sbnM6eG1wPSJodHRwOi8vbnMuYWRvYmUuY29tL
3hhcC8xLjAvIiB4bWxuczp4bXBNTT0iaHR0cDovL25zLmFkb2JlLmNvbS94YXAv
MS4wL21tLyIgeG1sbnM6c3RSZWY9Imh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLz
EuMC9zVHlwZS9SZXNvdXJjZVJlZiMiIHhtcDpDcmVhdG9yVG9vbD0iQWRvYm
UgUGhvdG9zaG9wIENTNS4xIFdpbmRvd3MiIHhtcE1NOkluc3RhbmNlSUQ9Inhtc
C5paWQ6M0YwNDIzMTQ3QzIzMTFFMjg3Q0VFQzhDNTgxMTRCRTQiIHhtcE1
NOkRvY3VtZW50SUQ9InhtcC5kaWQ6M0YwNDIzMTU3QzIzMTFFMjg3Q0VFQ
zhDNTgxMTRCRTQiPiA8eG1wTU06RGVyaXZlZEZyb20gc3RSZWY6aW5zdGFu
Y2VJRD0ieG1wLmlpZDozRjA0MjMxMjdDMjMxMUUyODdDRUVDOEM1ODE
```

```
xNEJFNCIgc3RSZWY6ZG9jdW1lbnRJRD0ieG1wLmRpZDozRjA0MjMxMzdDMj
MxMUUyODdDRUVDOEM1ODExNEJFNCIvPiA8L3JkZjpEZXNjcmlwdGlvbj4gP
C9yZGY6UkRGPiA8L3g6eG1wbWV0YT4gPD94cGFja2V0IGVuZD0iciI/PsOChsg
AAADUSURBVHjaYvz///9JBgYGMwbSwSkGoOafQPwKiAOBmIEIHAXED0H6
QJwPQGwAxE+AOJOAxnwgvgfEKiB9MM0gWg6IbwNxIw6NXUB8HogloHwUz
SAsBAoDIJ4DxMxQMRA9H4gPADE/kloMzSCsBcR/gHgj1LAt0HBRR1P3gQktB
A2AeBcQZwHxCyB+AsT3gTgFKq6FohrJZnssoW6AxPaDBqoZurP9oBrtCYS2ExA
/h9JgzX+gAsZExrMZVP0fmGZ1IjWiBCoL0NsXgPgGGcnzLECAAQD5y8iZ2Z69I
wAAAABJRU5ErkJggg==);
  background-position: 220px 10px;
}

form input.pass {
  background-image:
url(data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAAA0AAAAQCAY
AAADNo/U5AAAAGXRFWHRTb2Z0d2FyZQBBZG9iZSBJbWFnZVJlYWR5ccll
PAAAAyJpVFh0WE1MOmNvbS5hZG9iZS54bXAAAAAAADw/eHBhY2tldCBiZ
Wdpbj0i77u/IiBpZD0iVzVNME1wQ2VoaUh6cmVTek5UY3prYzlkIj8+IDx4Onhtc
G1ldGEgeG1sbnM6eD0iYWRvYmU6bnM6bWV0YS8iIHg6eG1wdGs9IkFkb2JlIFh
NUCBDb3JlIDUuMC1jMDYxIDY0LjE0MDk0OSwgMjAxMC8xMi8wNy0xMDo1
NzowMSAgICAgICAgIj4gPHJkZjpSREYgeG1sbnM6cmRmPSJodHRwOi8vd3d3Ln
czLm9yZy8xOTk5LzAyLzIyLXJkZi1zeW50YXgtbnMjIj4gPHJkZjpEZXNjcmlwdGl
vbiByZGY6YWJvdXQ9IiIgeG1sbnM6eG1wPSJodHRwOi8vbnMuYWRvYmUuY29
tL3hhcC8xLjAvIiB4bWxuczp4bXBNTT0iaHR0cDovL25zLmFkb2JlLmNvbS94YX
AvMS4wL21tLyIgeG1sbnM6c3RSZWY9Imh0dHA6Ly9ucy5hZG9iZS5jb20veGFw
LzEuMC9zVHlwZS9SZXNvdXJjZVJlZiMiIHhtcDpDcmVhdG9yVG9vbD0iQWRv
YmUgUGhvdG9zaG9wIENTNS4xIFdpbmRvd3MiIHhtcE1NOkluc3RhbmNlSUQ9I
nhtcC5paWQ6NTVFMDg1QzU3QzIzMTFFMjgwQThGODZFM0EwQUZFQ0YiIH
htcE1NOkRvY3VtZW50SUQ9InhtcC5kaWQ6NTVFMDg1QzY3QzIzMTFFMjgwQ
ThGODZFM0EwQUZFQ0YiPiA8eG1wTU06RGVyaXZlZEZyb20gc3RSZWY6aW5
zdGFuY2VJRD0ieG1wLmlpZDo1NUUwODVDMzdDMjMxMUUyODBBOEY4Nk
UzQTBBRkVDRiIgc3RSZWY6ZG9jdW1lbnRJRD0ieG1wLmRpZDo1NUUwODV
DNDdDMjMxMUUyODBBOEY4NkUzQTBBRkVDRiIvPiA8L3JkZjpEZXNjcmlw
dGlvbj4gPC9yZGY6UkRGPiA8L3g6eG1wbWV0YT4gPD94cGFja2V0IGVuZD0ici
```

```
I/Pv2NSIIAAADYSURBVHjanJAxCsJAEEXXaBMQtvIMqTxDKjtPELC1svMoOY
M2WlqIhVcQFMVgG7ATAoIggfGPjrLIrBo/vCzZ+Z+dGUNExiECI7Clhw5gAtqur
8YfUQxm4AzGIAMRSIAFXbC8OyUdghwsgH173cp9Lr5XqAeOSsANcj3h/8BpbQ
4Ko6uQOvtMQy6noG4+iz3XZ4iHbIEQ9L8EeUlN3t5etvSrMg6RqajAc78BQ7BTq6
QrllV3tKLvpZOclyrt/TWTlTP0zVQqba/BAKyUWsmh1BPUxL70JsAABHkyyK1uo
cIAAAAASUVORK5CYII=);
  background-position: 223px 8px
}

::-webkit-input-placeholder { color:#ccc; text-transform: uppercase; }
::-moz-placeholder { color: #ccc; text-transform: uppercase; }
:-ms-input-placeholder { color: #ccc; text-transform: uppercase; }

form button[type=submit] {
  width: 248px;
  margin-bottom: 50;
  color:   ;
  letter-spacing: .05em;
  text-shadow: 0 1px 0 #133d3e;
  text-transform: uppercase;
  background:  #1503A5 ;
  border-top-color: #9fb5b5;
  border-left-color: #608586;
  border-bottom-color: #1b4849;
  border-right-color: #1e4d4e;
  cursor: pointer;
}
form button[type=button] {
  width: 248px;
  margin-bottom: 50;
  color:#050404   ;
  letter-spacing: .05em;
  text-shadow: 0 1px 0 #133d3e;
  text-transform: uppercase;
  background:#1503A5 ;
  border-top-color: #9fb5b5;
  border-left-color: #608586;
  border-bottom-color: #1b4849;
  border-right-color: #1e4d4e;
  cursor: pointer;
}


.container{
            position:relative;
            width:400px;
            height:300px;
            border-radius:5px;
            border:;
            overflow:hidden;
            margin-top:-270px;
            margin-left:900px;

        }

{
border-style:none;
height:110px;
width:1330px;
background:rgba(255, 0, 0,0.5);
margin-top:26px;
}
.titlecs{
border-style:solid;
height:100px;
width:1300px;
background:rgba(178,34,34,.8);
margin-left:20px;
```

```
}



    </style>
</head>
<body >
<div class="titlecs">
<h1 style="color:yellow; font-family:;margin-left:50px">
 PREDICTION OF INTRUSION ATTACKS USING MACHINE LEARNING
APPROACH</h1>
</div>

<form method="post">

    {% csrf_token %}

  <label></label>
  <input type="text" name="username"  placeholder="userid" class="email"
style="color:rgb(1, 146, 1)">

  <label ></label>
  <input type="password" name="password"  placeholder="password" class="pass">

  <button type="submit" style="color:"> LOGIN HERE</button>

</form>
<div class="image"></div>
<div class="image1"></div>



<script>
(function(){
        var imgLen = document.getElementById('imgGallary');
        var images = imgLen.getElementsByTagName('img');
        var counter = 1;

        if(counter <= images.length){
            setInterval(function(){
                images[0].src = images[counter].src;
                console.log(images[counter].src);
                counter++;

                if(counter === images.length){
                    counter = 1;
                }
            },4000);
        }
})();
</script>
<div class="userpageimage"></div>
</body>
</html>
```

Design.html

```
<!DOCTYPE html>
{% load static %}
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
    body{
    background: url("{% static 'image.jpg' %}");
 background-size: cover;
    }
        .menu table{
          width:100%;
```

```
         text-align:center;

          }
          .menu table td:hover{
          background:rgba(0,0,255,0.3);

          }
          .menu table td{
          background:#8A0829;

         }
          .menu table,.menu table th,.menu table td {
    border: ;
    border-collapse: collapse;

}
.menu table th,.menu table  td {
    padding: 15px;

    }


          .topic h1{
          color:black;
          padding:2px;
          text-align:center;
          border-style:none;
          height:100px;
          width:1330px;
          float:left;


          }
          .mainholder{
```

```
          position:relative;
          top:50px;
          left:50px;
          z-index:999;
          float:left;
          }



    </style>
</head>
<body>
<div class="background-image">
    <div class="topic"><h1 style="color:orange;margin-top:10px;margin-
left:30px;border-style:none;width:1300px;height:40px;border-
color:black;background:;">
        PREDICTION OF INTRUSION ATTACKS USING MACHINE LEARNING
APPROACH </h1></div>
<div class="menu">
        <table>
            <tr>
                <td><a  style="color:yellow;text-decoration: none;" href="{% url 'userpage'
%}">PREDICTION</a></td>
                <td><a  style="color:yellow;text-decoration: none;" href="{% url
'chart_page' %}"> GRAPHICAL ANALYSIS </a></td>

                <td><a  style="color:yellow;text-decoration: none;" href="{% url 'index'
%}">LOGOUT</a></td>



            </tr>
        </table>
    </div>
</div>
```

```
</div>
<div class="marqee">

</div>
<div class="mainholder">
{% block userblock %}
{% endblock %}
</div>

</body>
</html>
```

Userpage.html:

```
{% extends 'design.html' %}
{% block userblock %}
{% load static %}

<style>
   .userpage{


   margin-top:10px;
   margin-left:90px;
   padding:10px;
   width:500px;
}

.userpage table{
   width:20em;
   text-align:center;
   //border-collapse:collapse;
   border-spacing:1px;
   background:;
}
```

```
.userpage table tr th{
   color:;
}
.userpage table tr th{
background:
padding:20px;
}
.userpage table tr td{
background:    #8B0000;
padding:10px;
}
.userpage table tr:hover td{
background:r);
}
.userpageimage{
border-style:solid;
width:500px;
height:380px;
margin-top:-270px;
margin-left:650px;
background: url("{% static 'images1.jpg' %}");
 background-size: 100%100%;

}
.userpage table label
{
   color: white;
}
.userpage table select
{
   width: 120px;
   height: 30px;
}
```

```
.userpage table input[type=text]
{
   width: 120px;
   height: 30px;
}
</style>
<div class="userpage">
<form method="post">
   {% csrf_token %}
   <table>
      <tr>
         <td><label>Protocol Type</label></td>
      <td><select name="protocol">
         <option value="1">ICMP</option>
         <option value="2">TCP</option>
         <option value="3">UDP</option>
      </select></td>
      </tr>
      <tr>
         <td><label>Service</label></td>
         <td><select name="service">
            <option value="1">ecr_i</option>
            <option value="2">private</option>
            <option value="3">http</option>
            <option value="4">smtp</option>
            <option value="5">other</option>
            <option value="6">ftp_data</option>
            <option value="7">domain_u</option>
            <option value="8">ftp</option>
            <option value="9">eco_i</option>
            <option value="10">telnet</option>
            <option value="11">auth</option>
            <option value="12">urp_i</option>
            <option value="13">ntp_u</option>
```

```
        <option value="14">finger</option>
        <option value="15">discard</option>
        <option value="16">Z39_50</option>
        <option value="17">echo</option>
        <option value="18">whois</option>
        <option value="19">nntp</option>
        <option value="20">remote_job</option>
        <option value="21">uucp</option>
        <option value="22">shell</option>
        <option value="23">domain</option>
        <option value="24">http_443</option>
        <option value="25">imap4</option>
        <option value="26">exec</option>
        <option value="27">ctf</option>
        <option value="28">time</option>
        <option value="29">sunrpc</option>
        <option value="30">klogin</option>
        <option value="31">kshell</option>
        <option value="32">netbios_ns</option>
        <option value="33">hostnames</option>
        <option value="34">pm_dump</option>
        <option value="35">nnsp</option>
        <option value="36">supdup</option>
        <option value="37">login</option>
        <option value="38">IRC</option>
        <option value="39">sql_net</option>
        <option value="40">vmnet</option>
        <option value="41">ldap</option>
        <option value="42">iso_tsap</option>
    </select></td>
</tr>
<tr>
    <td><label>Flag</label></td>
   <td><select name="flag">
```

```
            <option value="1">sf</option>
            <option value="2">s0</option>
            <option value="3">REJ</option>
            <option value="4">RSTR</option>
            <option value="5">RSTO</option>
          </select> </td>
        </tr>
        <tr>
          <td><label>Src Bytes</label></td>
          <td><input type="text" placeholder="0-60000" name="src_bytes"
required></td>
        </tr>
        <tr>
          <td><label>Dst Bytes</label></td>
          <td><input type="text" placeholder="0-300000" name="dst_bytes"
required></td>
        </tr>
        <tr>
          <td><label>Is Host Login</label></td>
          <td><select name="is_host_login">
            <option value="0">No</option>
            <option value="1">Yes</option>
          </select></td>
        </tr>
        <tr>
          <td><label>Is Guest Login</label></td>
          <td><select name="is_guest_login">
            <option value="0">No</option>
            <option value="1">Yes</option>
          </select></td>
        </tr>
        <tr>
          <td><label>Diff Srv Rate</label></td>
          <td><input type="text" placeholder="0-1" name="diff_srv_rate"
```

```
required></td>
        </tr>
        <tr>
            <td><label>srv_diff_host_rate</label></td>
            <td><input type="text" placeholder="0-300000" name="srv_diff_host_rate"
required></td>
        </tr>
        <tr>
            <td colspan="2"><input type="submit" value="Submit"></td>
        </tr>
    </table>
</form>
    </div>

{% endblock %}
```

Prediction Result.html:

```
{% extends 'design.html' %}
{% block userblock %}
{% load static %}

<style>
    .userpage{

    margin-top:10px;
    margin-left:90px;
    padding:10px;
    width:500px;
}

.userpage table{
    width:20em;
```

```
   text-align:center;
   //border-collapse:collapse;
   border-spacing:1px;
   background:;
}

.userpage table tr th{
   color:;
}
.userpage table tr th{
background:
padding:20px;
}
.userpage table tr td{
background:    #8B0000;
padding:10px;
}
.userpage table tr:hover td{
background:r);
}
.userpageimage{
border-style:solid;
width:500px;
height:380px;
margin-top:-370px;
margin-left:650px;
background: url("{% static 'images1.jpg' %}");
 background-size: 100%100%;


}
.userpage table label
{
   color: white;
}
```

```
.userpage table select
{
   width: 120px;
   height: 30px;
}
.userpage table input[type=text]
{
   width: 120px;
   height: 30px;
}
.userpage h1,h2,h3,h4
{
   color: white;
}
</style>
<div class="userpage">
<center><h1>Prediction Result</h1></center>
<h2>Attack Type</h2>
<h2>{{ attack_type }}</h2>
<h2>Class of Attack</h2>
<h3>{{ attack }}</h3>
<h3>Recommendation</h3>
<h4>{{ recommend }}</h4>
<h3>Precautions</h3>
<h4>{{ precautions }}</h4>

   </div>
{% endblock %}
```

Chart_page.html:

```
{% extends 'design.html' %}
{% block userblock %}
{% load static %}
```

```
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
<style>
   .userpage{

   margin-top:10px;
   margin-left:90px;
   padding:10px;
   width:500px;
}

.userpage table{
   width:20em;
   text-align:center;
   //border-collapse:collapse;
   border-spacing:1px;
   background:;
}

.userpage table tr th{
   color:;
}
.userpage table tr th{
background:
padding:20px;
}
.userpage table tr td{
background:    #8B0000;
padding:10px;
}
.userpage table tr:hover td{
background:r);
}
.userpageimage{
border-style:solid;
```

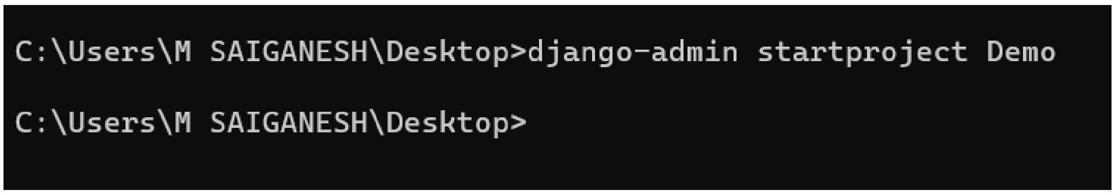
```
width:500px;
height:380px;
margin-top:-270px;
margin-left:650px;
background: url("{% static 'images1.jpg' %}");
 background-size: 100%100%;

}
.userpage table label
{
   color: white;
}
.userpage table select
{
   width: 120px;
   height: 30px;
}
.userpage table input[type=text]
{
   width: 120px;
   height: 30px;
}
</style>
<div class="userpage">
<h3 style="color:white">Confussion Matrix</h3>
<canvas id="confusionMatrix"></canvas>

   <script>
      document.addEventListener('DOMContentLoaded', function() {
         var ctx = document.getElementById('confusionMatrix').getContext('2d');
         var data = {
            labels: ['True Negative', 'False Positive', 'False Negative', 'True Positive'],
            datasets: [{
               data: {{ confusion_matrix|safe }},
               backgroundColor: [
                   'rgba(255, 99, 132, 1)',
                   'rgba(54, 162, 235, 1)',
                   'rgba(255, 206, 86, 1)',
                   'rgba(75, 192, 192, 1)',
               ],
               borderColor: [
                   'rgba(255, 99, 132, 1)',
                   'rgba(54, 162, 235, 1)',
                   'rgba(255, 206, 86, 1)',
                   'rgba(75, 192, 192, 1)',
               ],
               borderWidth: 1
            }]
         };

         var options = {
            responsive: true,
            scales: {
               y: {
                   beginAtZero: true
               }
            }
         };

         new Chart(ctx, {
            type: 'bar',
            data: data,
            options: options
         });
      });
   </script>
   <h3 style="color: white">Model Test Metrics:</h3>
<img style="float: left;" src="data:image/png;base64,{{ image_data }}">
```

```
    </div>

{% endblock %}
```

7.3Django:

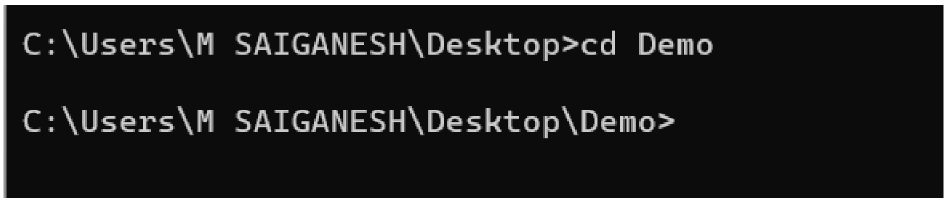
We need to install Django frame work in our system.
Command: pip install Django

By typing the above command Django is installed in local system,to create a project we need to type the below command.it creates the folder with Demo on its file location.
Command: django-admin startproject projectname



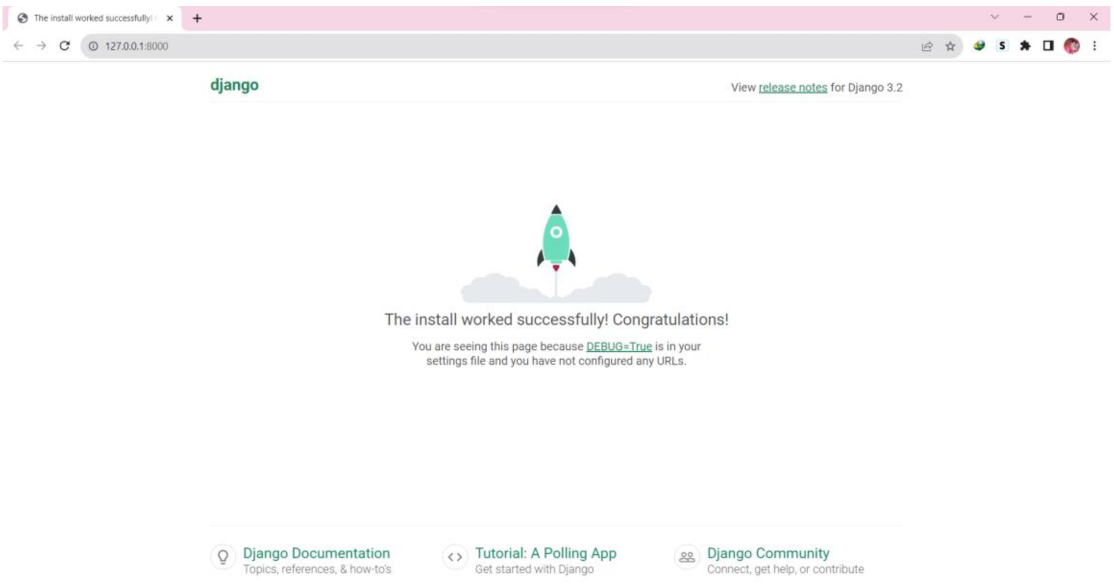
```
C:\Users\M SAIGANESH\Desktop>django-admin startproject Demo

C:\Users\M SAIGANESH\Desktop>
```

Navigate to Project by following the command: cd projectname



```
C:\Users\M SAIGANESH\Desktop>cd Demo

C:\Users\M SAIGANESH\Desktop\Demo>
```

Run the Development Server: python manage.py run server
This will start the server, and you can view your Django project by navigating to **http://127.0.0.1:8000/** in your web browser.


The install worked successfully!
127.0.0.1:8000
django
View release notes for Django 3.2

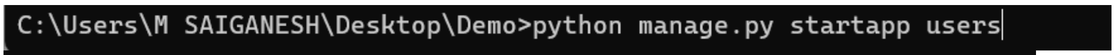
The install worked successfully! Congratulations!

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

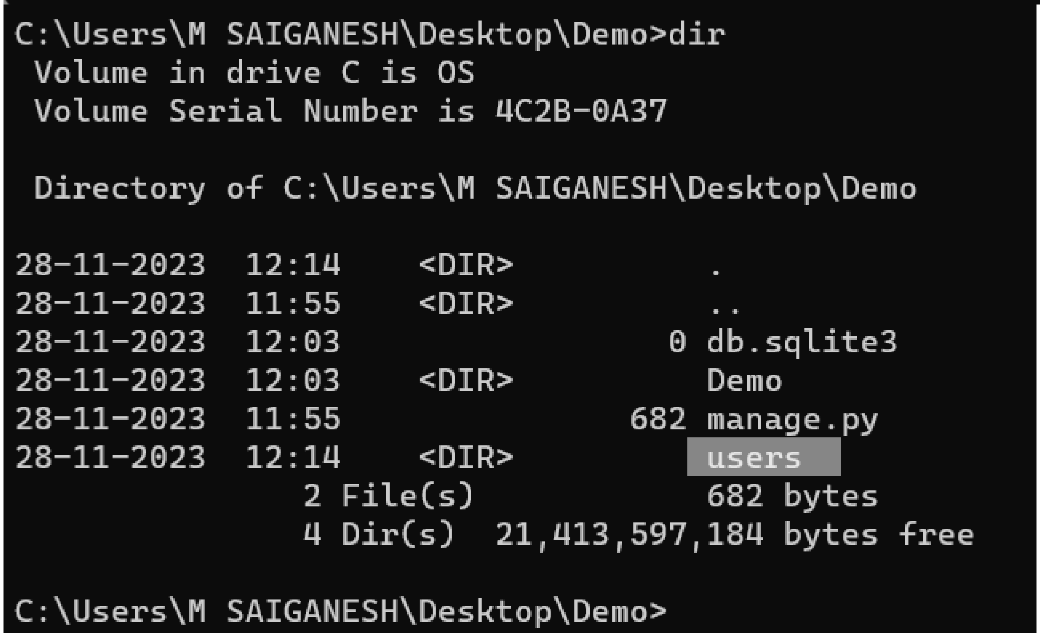
Django Documentation
Topics, references, & how-to's
Tutorial: A Polling App
Get started with Django
Django Community
Connect, get help, or contribute

We have created Project in Django to use it clearly then we need to create a App.

We are creating App by following the command: python manage.py startapp users.

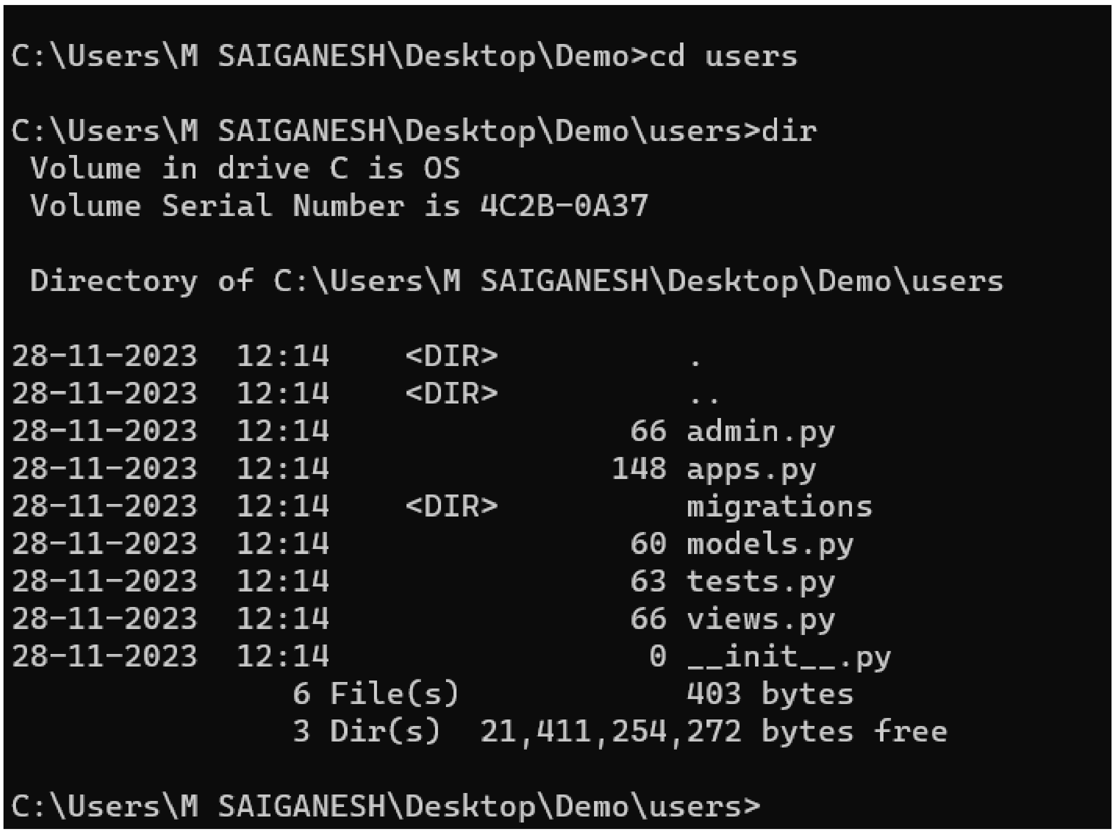


```
C:\Users\M SAIGANESH\Desktop\Demo>python manage.py startapp users
```



```
C:\Users\M SAIGANESH\Desktop\Demo>dir
 Volume in drive C is OS
 Volume Serial Number is 4C2B-0A37

 Directory of C:\Users\M SAIGANESH\Desktop\Demo

28-11-2023  12:14    <DIR>          .
28-11-2023  11:55    <DIR>          ..
28-11-2023  12:03                 0 db.sqlite3
28-11-2023  12:03    <DIR>          Demo
28-11-2023  11:55               682 manage.py
28-11-2023  12:14    <DIR>          users
               2 File(s)            682 bytes
               4 Dir(s)  21,413,597,184 bytes free

C:\Users\M SAIGANESH\Desktop\Demo>
```

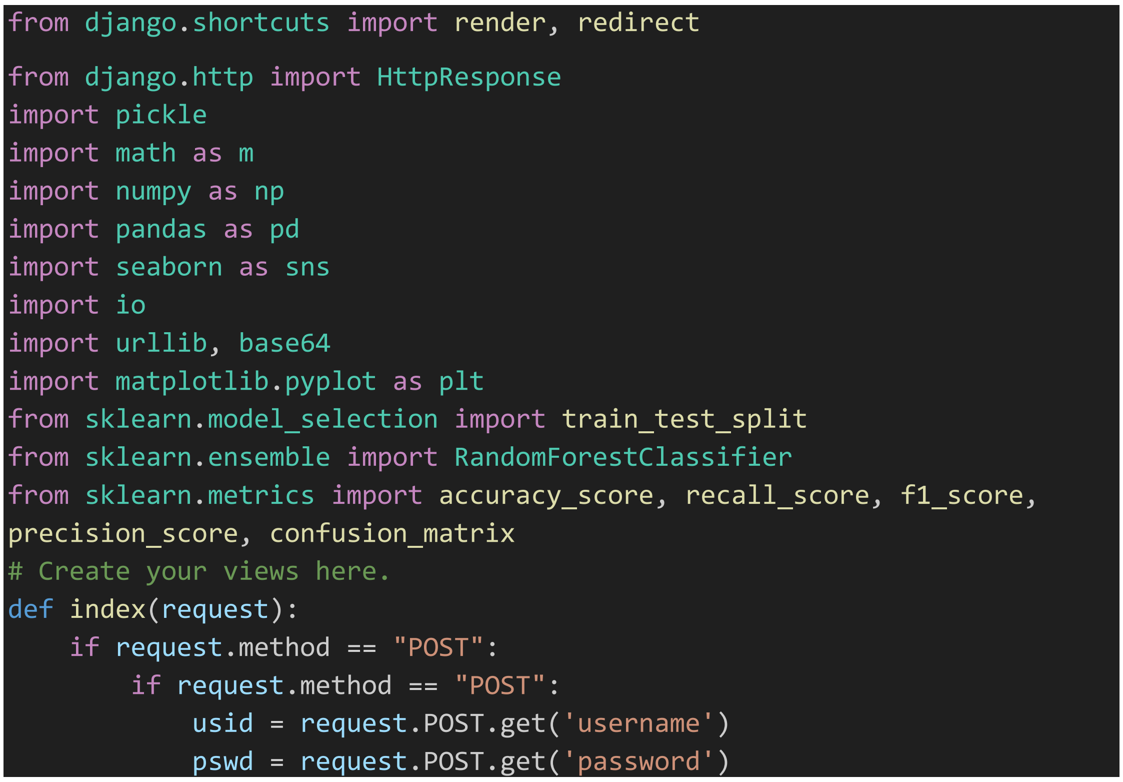
It has created an app with users as app name, it contains few python files.



```
C:\Users\M SAIGANESH\Desktop\Demo>cd users

C:\Users\M SAIGANESH\Desktop\Demo\users>dir
 Volume in drive C is OS
 Volume Serial Number is 4C2B-0A37

 Directory of C:\Users\M SAIGANESH\Desktop\Demo\users

28-11-2023  12:14    <DIR>          .
28-11-2023  12:14    <DIR>          ..
28-11-2023  12:14                66 admin.py
28-11-2023  12:14               148 apps.py
28-11-2023  12:14    <DIR>          migrations
28-11-2023  12:14                60 models.py
28-11-2023  12:14                63 tests.py
28-11-2023  12:14                66 views.py
28-11-2023  12:14                 0 __init__.py
               6 File(s)            403 bytes
               3 Dir(s)  21,411,254,272 bytes free

C:\Users\M SAIGANESH\Desktop\Demo\users>
```

In Django framework main logic is written in views.py file in app folder .

Views.py



```python
from django.shortcuts import render, redirect

from django.http import HttpResponse
import pickle
import math as m
import numpy as np
import pandas as pd
import seaborn as sns
import io
import urllib, base64
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, recall_score, f1_score,
precision_score, confusion_matrix
# Create your views here.
def index(request):
    if request.method == "POST":
        if request.method == "POST":
            usid = request.POST.get('username')
            pswd = request.POST.get('password')
            if usid == 'admin' and pswd == 'admin':
                return redirect('userpage')
    return render(request,'index.html')

def userpage(request):
    if request.method=="POST":
        if request.method=="POST":
            protocol_type=request.POST.get('protocol')
            service=request.POST.get('service')
            flag=request.POST.get('flag')
            src_bytes=request.POST.get('src_bytes')
            dst_bytes=request.POST.get('dst_bytes')
            is_host_login=request.POST.get('is_host_login')
            is_guest_login=request.POST.get('is_guest_login')
            diff_srv_rate=request.POST.get('diff_srv_rate')
            srv_diff_host_rate=request.POST.get('srv_diff_host_rate')

            model=pickle.load(open("model/attack_prediction.pkl","rb"))
            model1 = pickle.load(open("model/severity_prediction.pkl",
"rb"))
            model2 =
pickle.load(open("model/recommendation_prediction.pkl", "rb"))
            inputs=[protocol_type, service, flag, src_bytes, dst_bytes,
is_host_login, is_guest_login, diff_srv_rate, srv_diff_host_rate]
            arr=np.array(inputs,dtype='float64')
            arr=arr.reshape(1,-1)
            pred=round(float(model.predict(arr)))#attack prediction
            inputs1 = [protocol_type, service, flag, src_bytes,
dst_bytes, is_host_login, is_guest_login, diff_srv_rate,
                       srv_diff_host_rate, pred]
            arr1=np.array(inputs1, dtype='float64')
            arr1=arr1.reshape(1,-1)
            pred1=round(float(model1.predict(arr1)))#severity
prediction
            #pred1=m.ceil(float(model1.predict(arr1)))#severity
prediction
            inputs2 = [protocol_type, service, flag, src_bytes,
dst_bytes, is_host_login, is_guest_login, diff_srv_rate,
                       srv_diff_host_rate, pred, pred1]
            arr2 = np.array(inputs2, dtype='float64')
            arr2 = arr2.reshape(1, -1)
            pred2 = round(float(model2.predict(arr2)))#recommendations
prediction
           # pred2 =
m.ceil(float(model2.predict(arr2)))#recommendations prediction


            print(pred,pred1,pred2)
            if(pred==1 or pred==2 or pred==4 or pred==9 or pred ==10):
                pred1=1
                pred2=101
            if(pred==3):
                pred1=4
                pred2=104
            if(pred==6 or pred==11 or pred==12):
                pred1=3
                pred2=103
            if(pred==5 or pred==7 or pred==8):
                pred1=2
                pred2=102



            if(pred==1 and pred2==101 and pred1==1):
                attack_type="smurf"
            elif(pred==2 and pred2==101 and pred1==1):
                attack_type="neptune"
            elif (pred == 3 and pred2 == 104 and pred1 == 4):
                attack_type = "normal"
            elif (pred == 4 and pred2 == 101 and pred1 == 1):
                attack_type = "back"
            elif (pred == 5 and pred2 == 102 and pred1 == 1):
                attack_type = "satan"
            elif (pred == 6 and pred2 == 103 and pred1 == 3):
                attack_type = "warezclient"
            elif (pred == 7 and pred2 == 102 and pred1 == 2):
                attack_type = "portsweep"
            elif (pred == 8 and pred2 == 102 and pred1 == 2):
                attack_type = "ipsweep"
            elif (pred == 9 and pred2 == 101 and pred1 == 1):
                attack_type = "teardrop"
            elif (pred == 10 and pred2 == 101 and pred1 == 1):
                attack_type = "pod"
            elif (pred == 11 and pred2 == 103 and pred1 == 2):
                attack_type = "guess_passwd"
            elif (pred == 12 and pred2 == 103 and pred1 == 2):
                attack_type="imap"
            else:
                attack_type="some intrusion attack"
            print(pred,pred1,pred2)
            context1 = {'attack_type': attack_type}
            if(pred2==101):
                attack="""Denial of Service (DoS) Attacks:"""
                recommend="""Develop a robust DoS attack detection
algorithm that can quickly identify and mitigate these attacks. This
may involve monitoring for unusually high traffic or analyzing packet
patterns."""
                precautions="""Be cautious of false positives, as
legitimate traffic spikes can sometimes mimic DoS attacks. Continuously
update your detection methods to adapt to evolving attack
techniques."""
            elif(pred2==102):
                attack=""" Probe Attacks:"""
                recommend="""Employ network intrusion detection systems
(NIDS) to monitor and detect probe attacks. These attacks typically
involve scanning and probing activities, which can be detected by
monitoring for unusual port scanning patterns."""
                precautions="""Ensure that you have adequate logging
and monitoring in place to capture and analyze probe activities.
Consider setting up alerting mechanisms to respond promptly to probe
attacks."""
            elif(pred2==103):
                attack="""Remote-to-Local (R2L) Attacks:"""
                recommend="""Implement a system that monitors for
unauthorized remote access attempts. This may involve analyzing login
failures, authentication logs, and patterns of access."""
                precautions="""R2L attacks can be challenging to
detect, as they may appear as legitimate access attempts. Combine
signature-based detection with anomaly detection to improve
accuracy."""
            #elif(pred2==104):
            else:
                attack="""Normal Traffic:"""
                recommend="""Since normal traffic is what you want to
identify as a baseline, it's important to understand the typical
patterns in your network. Employ anomaly detection methods like
clustering or statistical analysis to identify deviations from the
norm."""
                precautions="""Ensure that your training data for
normal traffic is representative and up-to-date. Also, consider using
additional network monitoring tools to enhance your understanding of
normal network behavior."""
            #else:
             #   attack="""**User-to-Root (U2R) Attacks:**"""
              #  recommend=""" Recommendation:** Develop machine
learning models that can identify unauthorized user escalation
```

```
attempts. Feature engineering and behavioral analysis can help detect
unusual user actions."""
               # precautions="""Be aware that U2R attacks can be subtle
and hard to detect. Continuously update your detection models to
account for new attack vectors."""

            context2 = {'attack': attack}
            context3 = {'recommend': recommend}
            context4 = {'precautions': precautions}
            return render(request, "predictionresult.html",
{**context1, **context2, **context3, **context4})

    return render(request,'userpage.html')

def predictresult(request):
    return render(request,"predictionresult.html")
def model_metrics(request):
    df=pd.read_csv('dataset/arimadatasetfinal.csv')
    x=df.values[:,:9]
    y=df.values[:,9]
    y.reshape(-1,1)
    X_train, X_test, y_train, y_test =
train_test_split(x,y,test_size=0.2)
    model=pickle.load(open("model/attack_prediction.pkl","rb"))
    pred=model.predict(X_test)
    acc=accuracy_score(pred,y_test)*100

fscore=f1_score(pred,y_test,pos_label='positive',average='micro')*100

pscore=precision_score(pred,y_test,pos_label='positive',average='micro'
)*100

rescore=recall_score(pred,y_test,pos_label='positive',average='micro')*
100
    score_names=["Accuracy Score","F1 Score","Precission Score","Recall
Score","loss"]
    scores=[acc,fscore,pscore,rescore,100-fscore]
    cm = confusion_matrix(y_test, pred)
    #scores=[acc,fscore,pscore,rescore]
    scores=[acc,fscore,pscore,rescore,100-fscore]
    print(scores)
    plt.plot(score_names,scores)
    plt.title('Model Metric Performance')
    buffer = io.BytesIO()
    plt.savefig(buffer, format='png')
    buffer.seek(0)
    image_data = base64.b64encode(buffer.getvalue()).decode()
    context1 = {'image_data': image_data}
    context2 = {'confusion_matrix': cm.tolist()}
    return render(request, 'chart_page.html', {**context1, **context2})
```

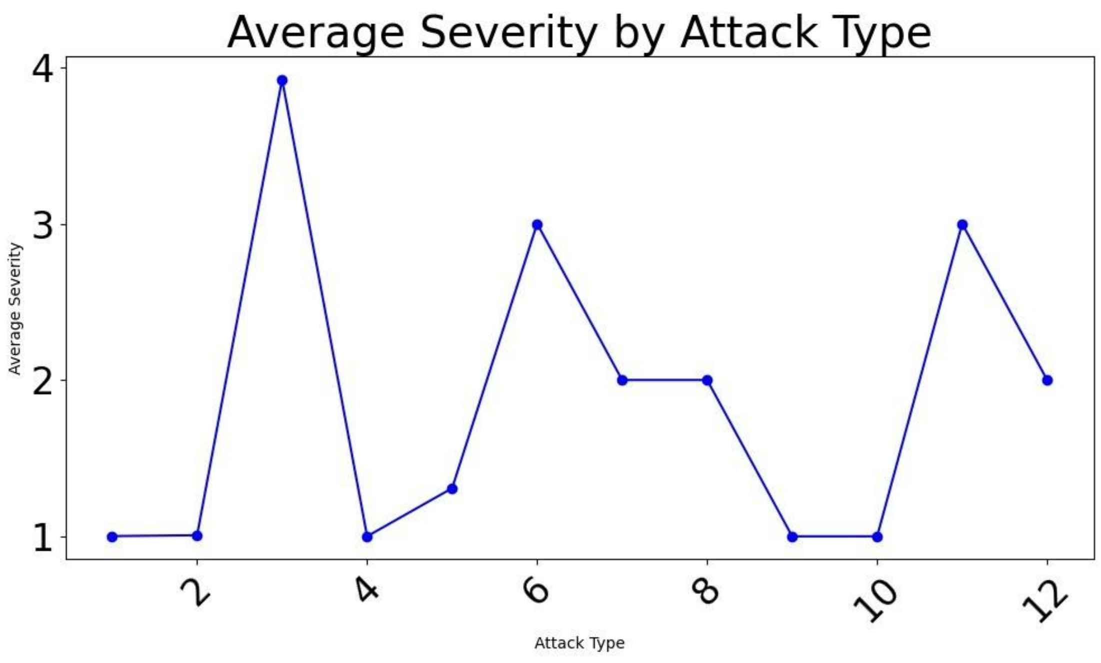


Fig.7 Attack type vs Severity plot

• In Figure 7 we can observe X-axis and Y-axis. Attack types on the X -axis in the data set and on Y-axis we can see severity of attack.

• Severity of Attacks is classified as High[1], Medium[2] ,Low[3] ,Normal[4].

• Attack Types are classified as smurf[1],,Neptune[2],,back[4] ,teardrop[9],, pod[10], normal[3],satan[5] ,portsweep[7], ,ipsweep[8], warezclient[6], guess[11], imap[12].

8.RESULTS

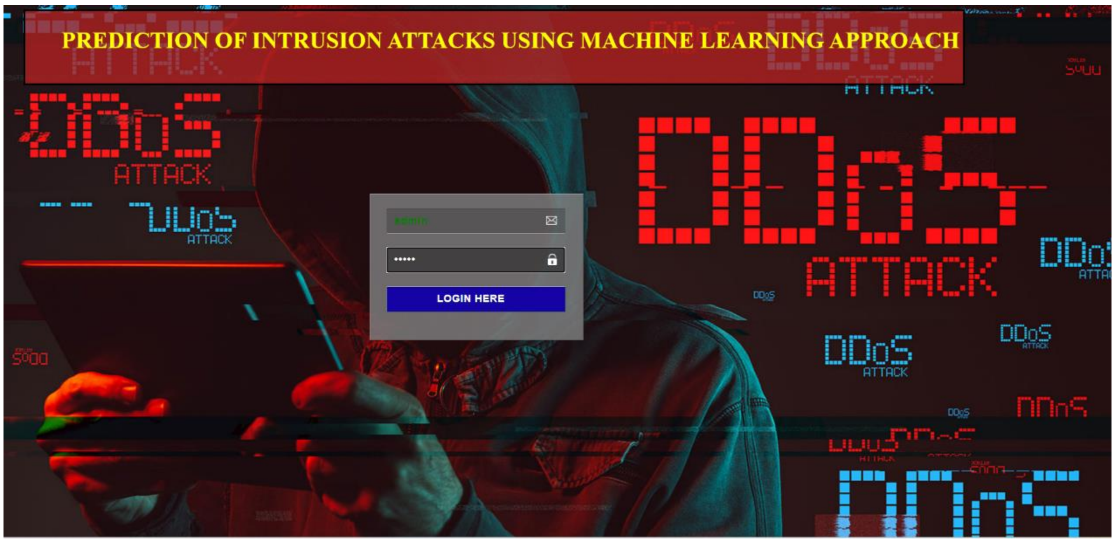
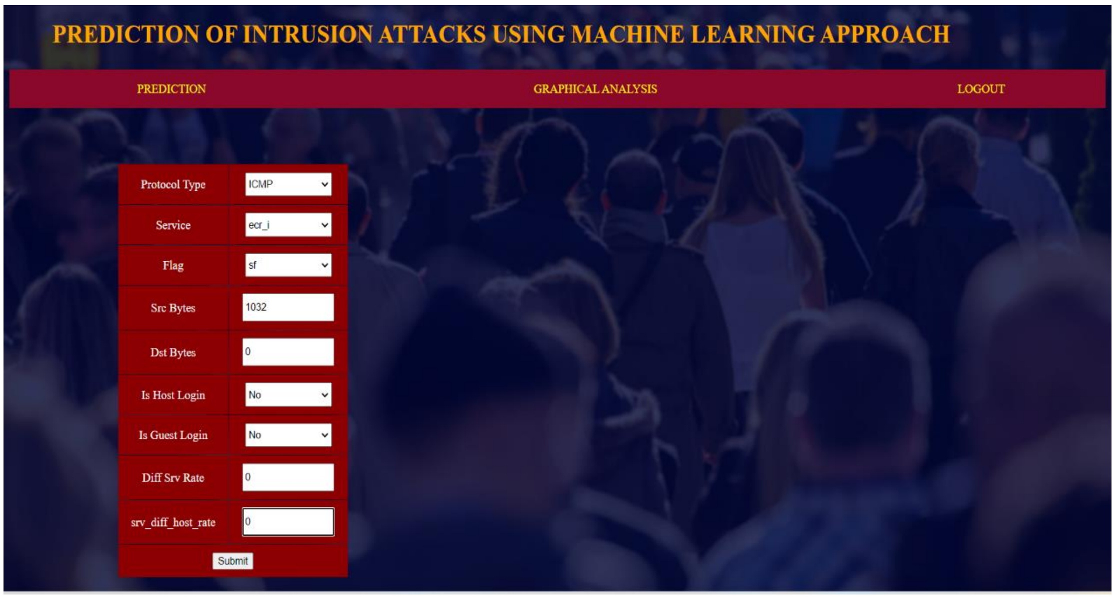

PREDICTION OF INTRUSION ATTACKS USING MACHINE LEARNING APPROACH
DDoS ATTACK
LOGIN HERE

Fig.8.1

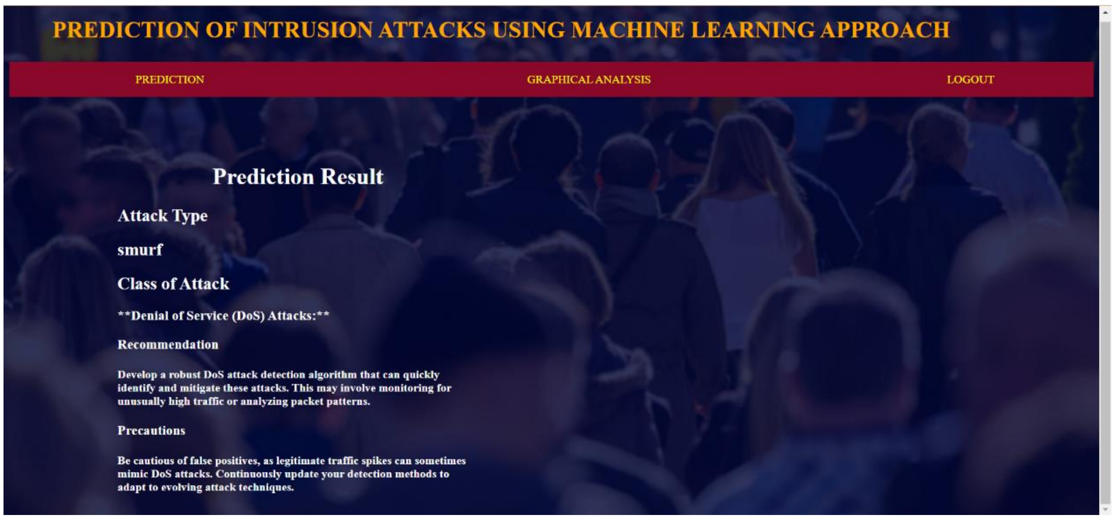

PREDICTION OF INTRUSION ATTACKS USING MACHINE LEARNING APPROACH

PREDICTION GRAPHICAL ANALYSIS LOGOUT

Protocol Type	ICMP
Service	ecr_i
Flag	sf
Src Bytes	1032
Dst Bytes	0
Is Host Login	No
Is Guest Login	No
Diff Srv Rate	0
srv_diff_host_rate	0

Submit

Fig.8.2


PREDICTION OF INTRUSION ATTACKS USING MACHINE LEARNING APPROACH

PREDICTION GRAPHICAL ANALYSIS LOGOUT

Prediction Result

Attack Type

smurf

Class of Attack

Denial of Service (DoS) Attacks:

Recommendation

Develop a robust DoS attack detection algorithm that can quickly identify and mitigate these attacks. This may involve monitoring for unusually high traffic or analyzing packet patterns.

Precautions

Be cautious of false positives, as legitimate traffic spikes can sometimes mimic DoS attacks. Continuously update your detection methods to adapt to evolving attack techniques.

Fig.8.3

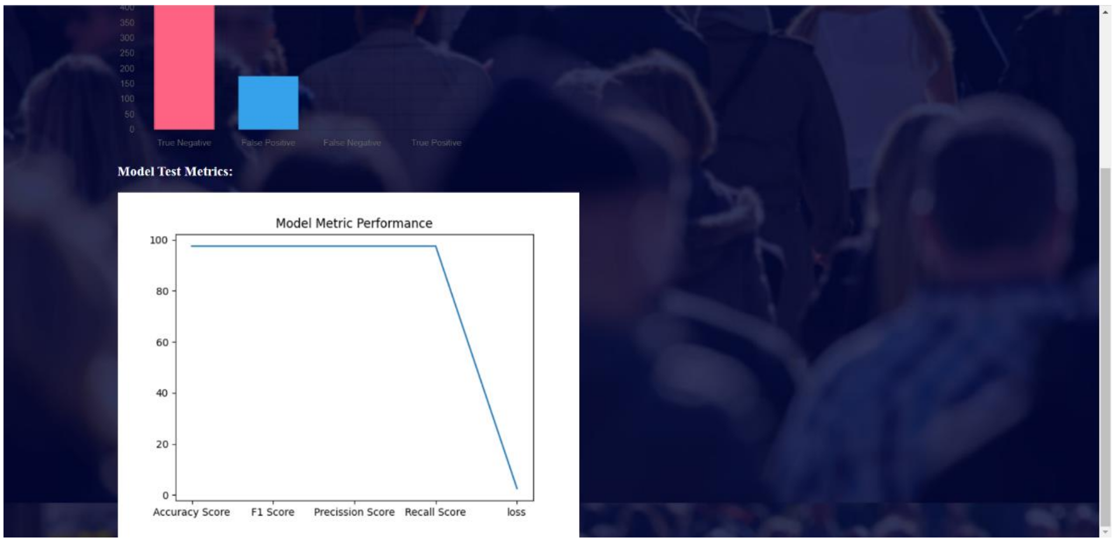


Fig.8.4

• When user gives the 9 inputs to the model and clicks on submit button, page gets redirected to results page,we can see the outputs as

• Attack Type identified in the dataset.

• Class of Attack as defined categorical in the earlier sections of the report.

• Recommendations and Precautions From input data we can find the accuracy score, F1 score, Precision Score, Recall Score and Loss also be calculated as shown in Figure 6.

9.LEARNING OUTCOMES

Technical Skills: Learn about machine learning algorithms and Django. We discovered that the MVT design of the Django framework, which distinguishes it by writing the frontend and backend code in different templates, is one of its primary features. Other features include object-relation mapping, an automated admin interface, url routing, and more. Django assisted us in creating an interface that is easy to use for our project. In addition, we now know about the machine learning techniques for decision tree classification, logistic regression, support vector machines, and random forests, which we will be using in our project to make predictions.

Problem-Solving: We acquired problem-solving skills by addressing the specific challenges while expanding our data set with new columns and creating a Django web interface with independent and dependent variables.

Collaboration: Working in a varied team including web developers, engineers, and experts in the subject of cyber security will help you build your collaboration abilities. To achieve project goals, cultivate strong communication and teamwork abilities.

User – Centered Design: Develop a grasp of design principles by thinking like an intrusive or hacker. Provide user-friendly features and interfaces.

Adaptability: Develop your ability to adapt by being open to feedback and ready to make project adjustments in response to user testing and real-world feedback. Gain the flexibility to alter course and make the necessary changes to improve the usability and effectiveness of the solution.

Empathy: Discover the traits that systems encounter when they are attacked in order to elicit empathy. Make use of this information to create strategies that significantly improve their level of digital security.

Organize, arrange, and total the venture inside the apportioned budget and timeline to sharpen your venture administration aptitudes. Get it how to distribute assets and part up work successfully

When thinking about innovation, it is important to take into account the moral implications regarding consent, information security, and protection, particularly for those who have less knowledge in the security field.

Presentation and Communication: Develop your ability to communicate intricate, creative ideas to audiences that are both non-technical and specialized.

10.CONCLUSION AND FUTURE SCOPE

We have also identified attacks which are highly occurred in the cyber physical systems and there damaged to the systems. We have applied the state of art model(SOTA/Hybrid Model) using KDD dataset to validate our research questions. From our analysis we found that Gradient Boost and SVM model perform better with 90% accuracy predicting attacks. In addition to this, explanation with possible recommendations to address these attacks, our results are in line with existing state of cyber physical models. In Future we expect researchers to predict the attacks possible with respective to time based and content based attacks.

11.REFERENCES

[1] Agarwal, A., Sharma, P., Alshehri, M., Mohamed, A.A., Alfarraj, O.: Classification model for accuracy and intrusion detection using machine learning approach. PeerJ Computer Science 7, e437 (2021)

[2] Ansari, M.S., Bartos, V., Lee, B.: Shallow and deep learning approaches for network intrusion alert prediction. Procedia Computer Science 171, 644–653 (2020)

[3] Jain, J.K., Waoo, A.A.: An artificial neural network technique for prediction of cyber-attack using intrusion detection system. Journal of Artificial Intelligence, Machine Learning and Neural Network (JAIMLNN) ISSN: 2799-1172 3(02), 33–42 (2023)

[4] Muneer, S.M., Alvi, M.B., Farrakh, A.: Cyber security event detection using machine learning technique. International Journal of Computational and Innovative Sciences 2(2), 42–46 (2023)

[5] Sarnovsky, M., Paralic, J.: Hierarchical intrusion detection using machine learning

and knowledge model. Symmetry 12(2), 203 (2020)

[6] Sharma, K.G., Singh, Y.: Predicting intrusion in a network traffic using variance of neighboring object's distance. International Journal of Com puter Network and Information Security 13(2), 73 (2023)

[7] Ullah, M.U., Hassan, A., Asif, M., Farooq, M., Saleem, M.: Intelligent intrusion detection system for apache web server empowered with ma chine learning approaches. International Journal of Computational and Innovative Sciences 1(1), 21–27 (2022).

[8] MeeraGandhi, G. (2010). Machine learning approach for attack prediction and classification using supervised learning algorithms. *Int. J. Comput. Sci. Commun*, *1*(2), 247-250.

[9] Ullah, M. U., Hassan, A., Asif, M., Farooq, M. S., & Saleem, M. (2022). Intelligent Intrusion Detection System for Apache Web Server Empowered with Machine Learning Approaches. *International Journal of Computational and Innovative Sciences*, *1*(1), 21-27.

[10] Sathya, R. (2022). Ensemble Machine Learning Techniques for Attack Prediction in NIDS Environment. *Iraqi Journal For Computer Science and Mathematics*, *3*(2), 78-82.

[11] Chaganti, R., Suliman, W., Ravi, V., & Dua, A. (2023). Deep learning approach for SDN-enabled intrusion detection system in IoT networks. *Information*, *14*(1), 41.