An optimised machine learning approach for detecting a network attack in an Intrusion detection system

ADITYA YADAV

Under the Supervision of

PANKAJ KUMAR KESERWANI

PROJECT REPORT

Submitted to

NATIONAL INSTITUTE OF TECHNOLOGY SIKKIM

for the award of the Degree of

Bachelor of Technology

in

Computer Science and Engineering

June 2021

Certificate by Supervisors

This is to certify that the project report entitled **“An optimised machine learning approach for detecting a network attack in an Intrusion detection system”** is being submitted by Mr. Aditya Yadav (Roll No. B170071CS), a student in the Department of Computer Science and Engineering, National Institute of Technology Sikkim, for the award of the degree of Bachelor of Technology (B.Tech). It is an original work carried out by him under my supervision and guidance. The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma.

Mr. Pankaj Kumar Keserwani

<u>Certificate by Student</u>

I hereby declare that the work presented in the report entitled **"An optimised machine learning approach for detecting a network attack in an Intrusion detection system"** is a bonafide record of the work done by me under the supervision of Asst. Prof. Pankaj Kumar Keserwani, Department of Computer Science and Engineering, and that no part thereof has been presented for the award of any other degree.

- I have followed the guidelines provided by the institute in writing the report.
- The report does not contain any classified information.
- Whenever I have used materials from other sources, I have given due credits to those by citing them in the text of the report and giving the details in the references.
- Whenever I have quoted written materials from other sources, I have put them under quotation marks and given due credits to the sources by citing them in the text of the report and giving their details in the references.

Dated: 28-06-2021 Mr. Aditya Yadav
Place: NIT SIKKIM Roll No.: B170071CS

Acknowledgment

I wholeheartedly thank our Project Guide Asst. Prof. Pankaj Kumar Keserwani, Department of Computer Science and Engineering, National Institute of Technology Sikkim for his extreme support and guidance throughout the project work.

My special thanks to our head of department Dr. Pratyay Kuila, Department of Computer Science and Engineering, National Institute of Technology Sikkim, who allowed me to carry out this project work and provided his guidance whenever necessary.

I am grateful to my parents who gave me persistent encouragement and inspiration throughout my being.

Finally, I would like to thank everyone who has directly or indirectly helped me in the successful completion of this project and throughout.

Aditya Yadav

Contents

Page no.

List of Tables

List of Equations

List of figures **Page no.**

Chapter 1
Introduction

1.1. Abstract

With increasing day-to-day emerging technologies, IoT, cloud computing, artificial intelligence, edge computing, quantum computing the network security is degraded because of the continuous increase in the number of cyber-attacks. However, at the same time, security has become a major concern due to increased attack scenarios and varied threats to the genuine user community. A recent trend in the number of increasing cyber-attacks is shown in Fig1. The traditional solutions are inefficient and ineffective in providing desired protection against continuously advancing cyberattacks such as Denial of Service (DoS) attacks, IP flooding, malware, illegal access, altering of data, forgery, interrupting the normal behavior of the system and so on. Varied cyberattacks are not only resulting in massive financial losses but have other serious consequences. An intrusion detection system is proposed based on chi-square feature selection algorithm and tuned hyper-parameters using randomized search cv in random forest algorithm to classify the attack category with a reduced set of features by eliminating less relevant features using proposed feature selection methods to detect malicious attacks in computer networks by experimenting on the CICIDS 2017 dataset [1]. Most researchers stated that by choosing the optimal segment of features for an intrusion detection system, it is quite possible to slowly improve the accuracy of attack detection and model performance. In the present era where there is the rapid emergence of the latest technologies like Cloud Computing, IoT, etc. leading to a continuous increase in network traffic logs, so there is an urgent need for intrusion detection systems should be improved by collecting and analyzing the information of incoming network traffic data. There are some very massive popularly known datasets in which full features set are not contributing to the network logs traffic, thus to turn down and choose the only optimal variety of features could improve the performance, precision, speed, and correctness of the intrusion detection system. For this sci-kit learn library was utilized in this study. With this approach, optimal features were known within the dataset, and therefore the accuracy rate is enhanced. The outcomes are studied showing the concept that selected features improve the classifier performance.

1.2. Introduction

The recent increase in day-to-day usage of the number of network devices, advancement in existing and new technologies and services leading to a steady increase in cyber-attacks by hackers performing malicious attacks to modify or steal the data from victim systems. To overcome these all cybersecurity network-related attacks there is an increasing demand for protective measures like Intrusion Detection System (IDS) which is one of the essential elements of network security that helps to detect the attacks by analyzing network traffic packets or operating system network logs. In Fig1.showing continuous increase in cyber-attacks count over past years and this is increased about 200%. Therefore current scenario needs faster development in the field of cybersecurity, especially in network intrusion detection and prevention systems.

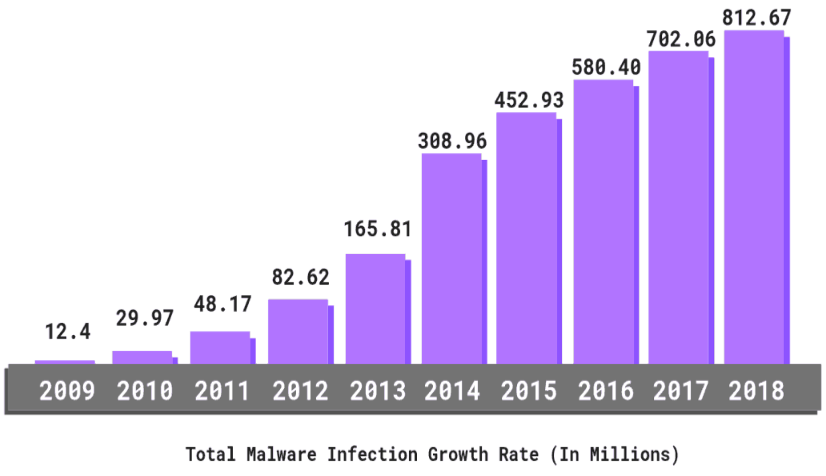


Figure 1 Cyber-attacks growth in recent years

To mitigate the risk of cyber-attacks IDS is one effective solution. Intrusion detection is a process of detecting and preventing intrusions activity on any network or system. It is one of the significant ways of preventing the computer system from intrusions. In general intrusion detection systems are software applications or hardware security programs that verify that anything occurring in the system over a network is malicious or normal in behavior [1]. There are five types of intrusion detection systems: 1) Host-based intrusion detection system 2) Network-based intrusion detection system 3) Protocol-based intrusion detection system 4) Application protocol-based intrusion detection system 5) Hybrid intrusion detection system. Each of the above-mentioned intrusion detection systems has a varied procedure in detection and providing security to the data, and each of them has its pros and cons [2]. Initiating alert to organizations when any malicious or intrusive activity is detected within network or system devices so network and host-based intrusion detection system play a vital role in cybersecurity. One of the most common problems of these intrusion detection systems is handling the large amounts of attack alerts and the way how to handle them. Another major problem that arises due to the recent emerging of new technologies is the upcoming issues of “Big Data”, “IoT”, “Fog & Edge Computing”, “Cloud Computing” is their large network data logs. These emerging technologies produce a large amount of data that is multidimensional with numerous features is needed to be reduced to get an efficient intrusion detection system [3]. It is now very surged need to implement these enormous data to be considered not because of the increasing number of tuples, but also for the number of features in each tuple that may lead to generate more false positives so redundancy of dataset classification will increase.

For any intrusion detection system, selecting a subset of features to reduce the set of non-contributing features in classification is called the pillar of an intrusion detection system. The efficiency and accuracy of a dataset have a major dependency on the performance of the classifier model on that dataset. If the dataset split into training components is less redundant and has more accurate logs then the testing component will have more improvised model performance [4]. Therefore, it is one of the most inevitable steps to get better accuracy performance choosing a less redundant dataset for testing and training components of the system model, and gathering the network logs dataset is a big quandary as a dataset that is publicly available for use is very diverted from real scenario. There are some well-known available datasets on network attacks mostly used by the researchers are such as NSL KDD, UNSW NB-15, CICIDS 2017 datasets.

In our implementing CICDS dataset, there are a set total of 79 features. To make our dataset fit for applying machine learning classification algorithms for gaining high accuracy dataset needs to be first pre-processed, cleaning of redundant data, encoding of categorical features, scaling of features with very huge values, feature reduction should be performed first to increase the efficiency and get training time reduced.

Machine learning approaches helped us to analyze learn from very huge datasets and make correct predictions overtime on continuous learning from each outcome feature has now reduced human intervention to a greater extent in the real-time data science world.

To get the proper intrusion detection system it involves feature elimination and classification of attack category for enhancing the performance of the model to get better metric value on accuracy, precision [7].

The rest of the study is organized as follows. In Chapter II, the related survey of works done by various researchers is discussed briefly. In Chapter III, the proposed model is discussed and analyzed in-depth for algorithms implemented from dataset selection to performance evaluation. Different phases of intrusion detection systems are described in detail with a detailed analysis of the performance evaluation of our proposed model. In Chapter IV Conclusion and future work are briefly mentioned.

1.3. Background Study & Motivation

Motivation to perform a detailed study in network security topic of intrusion detection systems is:

1) Continuously developing IoT networks, cloud computing have become an increasingly valuable target of malicious attacks due to the increased amount of valuable user data they contain. In response, network intrusion detection systems have been developed to detect suspicious network activity with more accuracy.

2) Traditional network security solutions may not be directly applicable due to the differences in the advancement of IoT structure and behavior in recent times.

3) Healthy class discussions with our project supervisor about recent issues, challenges, and solutions by machine learning and deep learning methods to solve the problem arouse interests in the field and motivated to do some contribution.

4) During this project, I have read many research and journal papers related to intrusion detection systems on various methods to solve this problem each having its pros and cons led me to work in this field more efficiently.

Detection of intruders is the detection and prevention of intrusive activity on any network or system. These intrusion attacks may lead to serious damage to networks and systems. The intrusion detection system keeps an eye on network systems to alert the admin if any malicious or intrusive activity is noticed. It is a radar-like software application continuously searching for fraud or intrusive activity or policy violations.

Types of Intrusion Detection System:

The intrusion detection system is of 5 types:

1. **Network Intrusion Detection System:**

   A network intrusion detection system is a hardware or software-based system that detects malicious traffic on the network. In a network intrusion detection system, there are some points on the network which inspect the congestion from incoming packets to all network devices. It checks the packets for if there is an attack by the pre-determined or known existing database. If it matches to attack an alert is sent to the administrator [8].

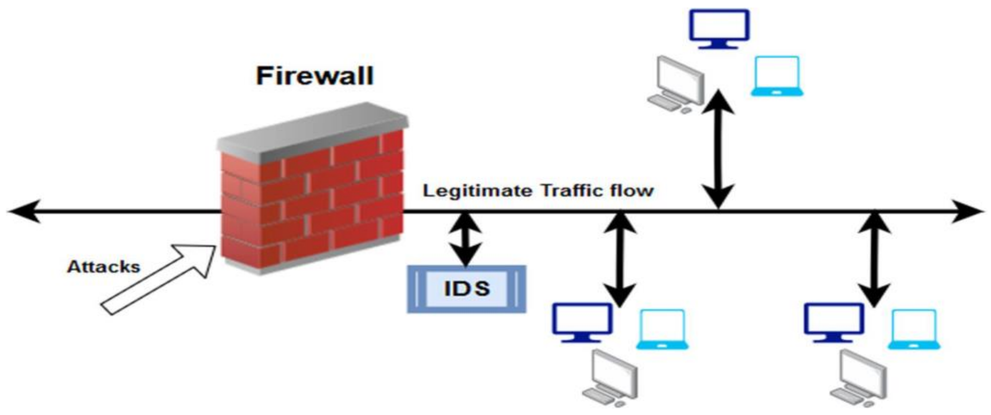


Figure 2 Network Intrusion Detection System

2. Host Intrusion Detection System:

A host-based intrusion detection system is an application that inspects for any malicious activity on a network and reports to the administrator if any suspicious activity is detected by operating through data on individual computers on the whole network system. Host-based IDS analyses the network by capturing screenshots of individual hosts information logs [9].

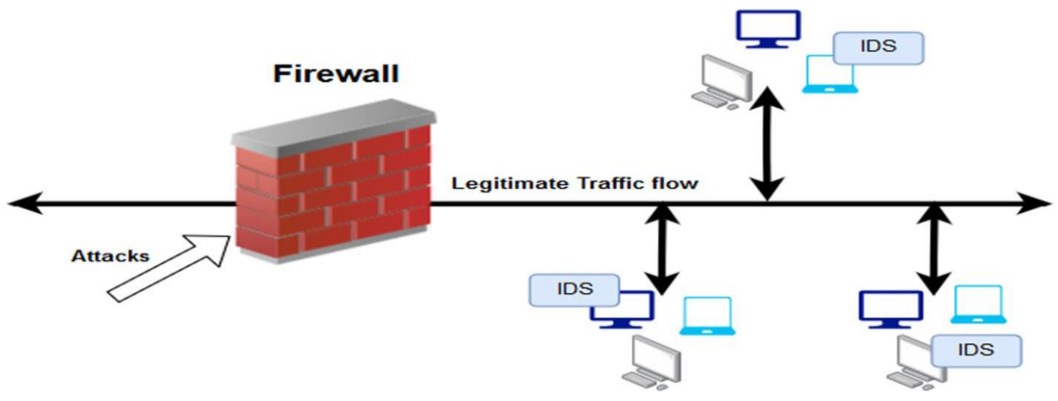


Figure 3 Host-based Intrusion Detection system

3. Protocol-based Intrusion Detection System:

A protocol-based intrusion detection system is installed on the front end of the webserver which performs its operation by analyzing the protocol system used between the host network device and the webserver. Protocol-based IDS keeps an eye on the front end of a web server HTTP, HTTPS protocol stream [10].

4. Application Protocol-based Intrusion Detection System:

Application Protocol-based Intrusion Detection System also monitors and analyses the dynamic network behavior from server installed application. But it typically performs its operation from the application protocol of a group of servers [11].

5. **Hybrid Intrusion Detection System :**

   If any intrusion detection system has two or more machine learning approaches are combined to perform as a single algorithm for classification then it is called a hybrid intrusion detection system. In this individual system, data is mixed with the data on the network system to get full network logs of the system [12].

Detection methods of IDS:

1. **Signature-based Method:**

   A signature-based intrusion detection system is based on detecting the attack or any malicious activity by matching the known attack pattern from existing data of pre-defined attacks. These attack patterns which are known in the database are called signatures. When any network behavior is matched with these known signatures then the intrusion detection system issues an alert alarm to the administrator that a malicious attack is happening over a network [13].

2. **Anomaly-based Method:**

   An anomaly-based intrusion detection system is based on detecting any unknown attack or malicious activity which is not known or its behavior is not in the existing database of the intrusion detection system. In an anomaly-based intrusion detection system, machine learning models are trained based on existing attack behavior and identifies for the attacks based on that available information and predictions [14].

Comparison of IDS with Firewalls:

Intrusion detection system it monitors and issues only alert if any malicious activity is detected however in firewalls it blocks any unauthorized access and also disables users to access any malicious content on the system. The major difference is that firewall can protect if any intrusive activity is happening inside the system but an intrusion detection system can alert only over a network.

- A firewall does not restrict or alerts for permitted network traffic areas while an intrusion detection system keeps an eye over the complete network.

- To function a firewall no human intervention is required but for an intrusion detection system, an administrator is required to handle the attacks when alerted by the system.

Machine learning solves many different types of problems; we can group these problems into the following tasks:

- Classification outputs a predicted label from some predefined set.
- Regression outputs a predicted real-valued numerical result.
- Clustering groups objects such that objects within the same cluster have similar properties compared to that of other clusters. There may or may not be a predefined number of clusters to fit the training data to.

Both classification and clustering are types of approaches which are suitable for this project. Classification aims to predict a category and assign the labels benign or attack. While clustering aims to divide the data into groups that are similar to each other. Here, we cluster the data into normal and anomalous cases.

1.4. Problem Statement

For high-dimensional data spaces, the machine learning-based intrusion detection systems have decreased their performance and a very large increase in execution time of the model as compared to low dimensional data spaces in which it has higher accuracy and efficiency in detecting network attack. Therefore it is an urgent need for implementing an appropriate feature reduction method with proper classification methods but some of the features which have a lesser dependency on target outcome variable can be eliminated so that it does not possess a great impact on the classification process and results in reducing time. There are many network intrusions dataset that works upon any network attacks-related datasets like UNSW-NB15, NSL KDD, CICIDS, etc. which includes many irrelevant features or features having least dependency on outcome variable so it drastically decreasing the rate of intrusion detection and increasing false alarm rate. Here the aim is to propose a relevant feature extraction technique to obtain more accurate and decrease false alarm rate. Apply a proper classification algorithm to train a model which detects the attacks in a network by following network log data.

Compare with other feature extraction and classifier algorithms of previously done researches with your proposed method.

1.5. Objectives

1. To identify the most significant features extraction mechanism of Intrusion detection on any network attacks dataset.

    - ✓ To achieve this objective a statistical-based feature selection algorithm chi-square is applied which selects the k best features having a top feature scores (chi-square).

2. Comparing Machine learning-based approaches with the reduced feature set and all features by measuring respective performance metric score.

    - ✓ To achieve this objective after feature extraction classification on all attack labels, binary classification and multi-class classification is performed on the dataset. Linear support vector machine classifier, Naïve Bayes classifier, Random Forest classifier, Decision tree classifier is applied and compared the performance of each model.

3. Propose a method with better accuracy and performance scores on testing set than standard machine learning approaches.

    - ➢ To propose a better classification algorithm a randomized Random forest classifier is proposed with different hyper-parameters selected by randomized search cv algorithm.

4. Obtain better accuracy and decreased false alarm rate with strongest and reduced features by Chi-square algorithm proposed.

    - ✓ To achieve this objective by our proposed trained model and used feature selection algorithm has shown accuracy of 99.98% in overall attack categories and has decreased the false alarm rate which results in a better intrusion detection system approach based on chi-square feature selection and proposed randomized random forest classifier.

1.6. Summary of Chapter

In this chapter, a discussion to introduction of Intrusion detection is done. It is a process of detecting and preventing intrusions activity on any network or system. The intrusion detection system is of five types NIDS, HIDS, Hybrid IDS, Protocol IDS, Application protocol IDS. There are two types of detection methods of IDS signature-based and anomaly-based detection. Signature-based detection can detect only pre-known attacks in the database while an anomaly-based intrusion detection system is capable of handling the unknown attacks on basis of information or trend learn from the huge network attacks log data. Due to continuous growth in network devices, IoT, cloud computing data files, and information security a primary-concern. In some recent reports of increasing cyber-attacks continuously as shown in Figure 1, there is a need for improvement in this field. Apart from seeing these incidents news in-class discussion with our project supervisor motivated me to give some contribution in this field. Machine Learning (ML), Deep Learning (DL), and nature-inspired approaches are applied to design and develop more efficient and intelligent security solutions to fight against fraudsters or attackers. Cloud computing has become popular among the business community due to its cost effectiveness and other advantages. Security is also one of the paramounts in the cloud environment, and hence efforts are made to propose enhanced NIDS. My problem is to find the best minimal subset of features to get better accuracy on the classification of attacks category. For this problem some objectives are defined to achieve that are selecting relevant features subset, applying proper classification model, comparing the metric score such as accuracy precision-recall f score with that of previously done researches. To perform this work CICIDS 2017 dataset is chosen for our works. CICIDS 2017 dataset has 80 features and 12 types of attacks mainly listed into 7 categories DOS, Probe, DDoS, Brute force, Benign, Botnet, Web attack. Each attack category has some attacks from the dataset which is properly listed in Table @ for the CICIDS dataset. I have discussed the relevant background knowledge required for this dissertation, my success criteria and project requirements, how I planned to implement the project and my starting point.

Chapter 2

Literature Survey

2.1. Related Tools & Technologies

Technology and tools used for this study are:

1. **Python 3.8.5:** Python is one of the most powerful dynamic programming languages. Python is also called an interpreted scripting language. The programming language in our implementation of machine learning algorithms is python's 3.8.5 version.

2. **Anaconda Software:** Anaconda is software used for machine learning, deep learning, data science for python and R programming languages. It has a collection of individual software packages named Jupyter Notebook, Spyder, Anaconda navigator, etc.

3. **Jupyter Notebook:** Jupyter Notebook is an open-source software also known as a python notebook included in Anaconda software which is capable of creating python notebooks mostly used for data science machine learning, deep learning. Jupyter notebook has filed with the extension ".ipynb".

4. **Machine Learning:** Machine Learning is the field of study in which machines are trained on basis of known patterns and are made able to think like humans. Machine learning is now used up in almost all fields to reduce human intervention and a better way of prediction and analysis.

5. **Python Libraries:** Some most common python libraries used in machine learning are used such as Pandas, numpy, sci-kit learn, matplotlib, etc.

2.2. Related Works

2.3. Summary

Chapter 3

Proposed Work and Implementation

3.1 Proposed Work

In this section, a framework to solve the problem statement a work is proposed to achieve all the goals and objectives is depicted in following flow-diagram Figure @ and brief later.

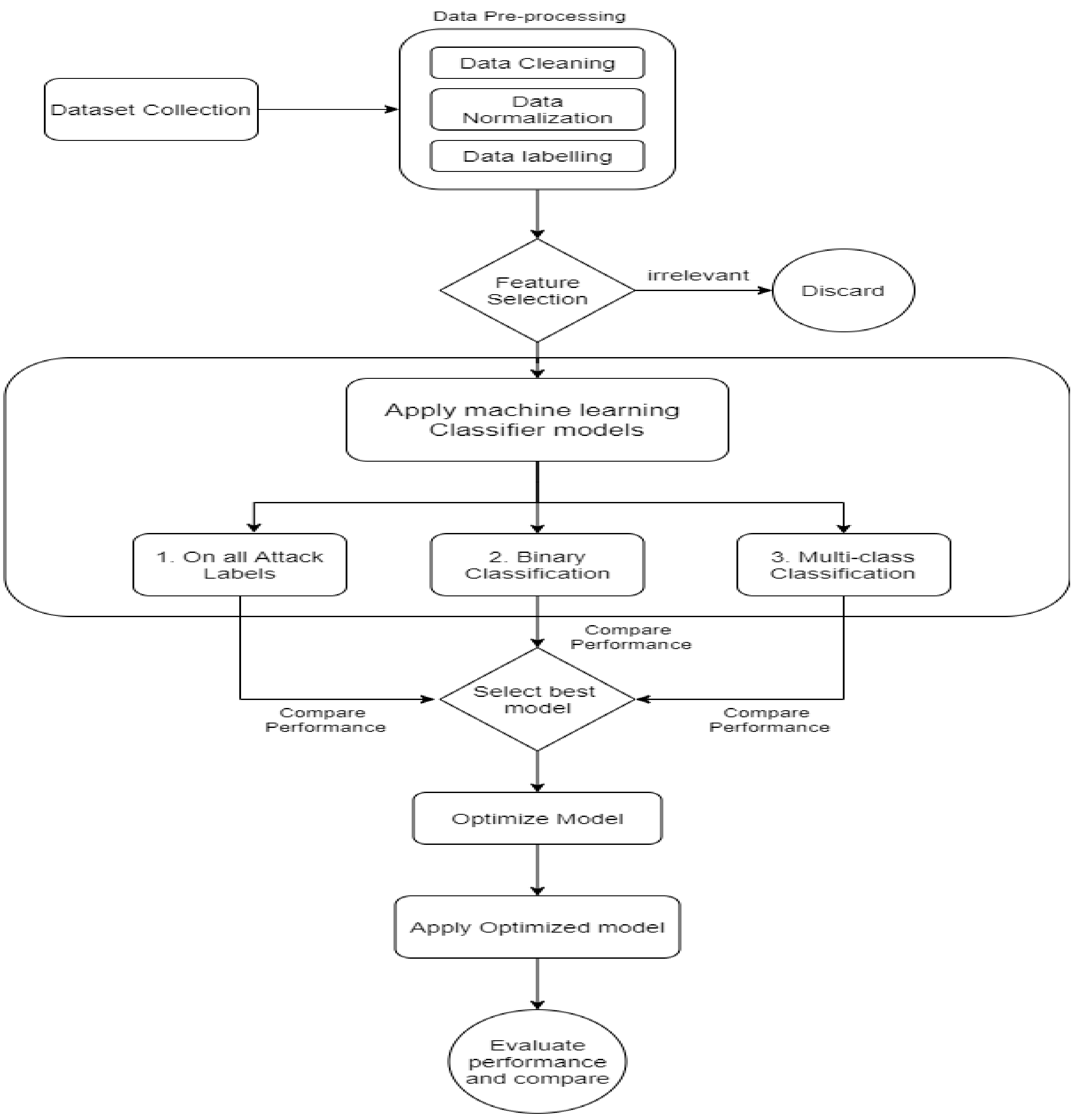


Figure 4 Proposed Work Flow Chart

3.1.1 Data Collection and cleaning

The first process to build a network intrusion detection system is to collect a dataset to experiment. Collected data may have some missing values, infinite values, duplicate columns, redundant attributes, etc. these all are termed as noise in the dataset so, the dataset is required to be cleaned for such noises if present any with suitable methods like replacing NaN values with mean, median values or dropping the columns redundant. In this, data cleaning step all inconsistencies in the dataset are removed to get a cleaned dataset. Data cleaning is performed within the Python environment.

3.1.2 Data Pre-processing

Data Pre-processing is another important step to improve the quality of data which gives an enhancement in results of our experimented proposed approach. Data Pre-processing involves analyzing the dataset attributes, removing and selecting proper attributes, labeling outcome attributes for better visualization, data normalization, encoding, feature scaling, splitting into training and testing set, etc.

Data Normalization

Normalization of a feature or attributes in data is a step to control the upper and lower limits of numerical values in data to a smaller range generally in [-1,1] or [0,1] without losing the actual information values. Normalization is required to perform because generally, high variances in different attributes of data may induce some problems while applying classification models. Normalization helps in the generation of new values within the specified range without affecting the general distribution and results. For example, if attributes have values -10, 234, -657, 987, 213 then to normalize this data, first check for the largest digit value and then divide each by the largest number of that digits for this example largest is 987 so divide by 1000 each then, normalized data will be -0.01, 0.234, -0.657, 0.987, 0.213. This method is called decimal standard scaling.

Data Encoding

In most of the datasets, some categorical features have their values in form of strings that are difficult to interpret by machines. To get better classification relevancy of these features should be included so these types of data features are first converted into numerical data types by several encoding methods. There are various encoding techniques such as one-hot encoder, label encoder, binary encoding, etc. are available. One-Hot-Encoding is a binary representation in which all are zero except one whose value is set to 1. This method of encoding variables is used to get binary features from the input of all categorical features. If any categorical feature has 'n' distinct values then one hot encoding makes 'n' columns with binary datatype for each distinct value of features.

Attack Labelling

Attack labelling is a step to be performed in any network intrusion detection dataset to perform experiment as label attribute present in dataset has many attack classes. To perform binary classification labelling is to be done as two category one is normal and other is attack all the attack classes except normal i.e. benign in our dataset are clumped in a single label called attack. For performing multi-class classification labelling is done in groups. For example: a highly benchmarked dataset NSL KDD has

22 attacks categorized in 4 different attack categories similarly CICIDS 2017 dataset has 12 attack classes categorized into 7 attack categories.

Splitting the dataset

The CICIDS 2017 dataset is splitted in two parts one is for training set and other is testing set. I have splitted dataset in ration 7:3 in training set with respect to testing set respectively. While splitting the dataset it is stratified with y attribute that is our target variable so that data is splitted proportionaly in all attack categories.

3.1.3 Feature Extraction

Feature Selection is the process of selecting the features which show the greatest contribution in finding the correct class prediction to target variable outcome. If there are some irrelevant features with no or least dependency on the target variable then it decreases the accuracy of classification.
For example, if there are 10 features with their different ratio of contribution to target variable outcome as 0.92, 0.2, 0.11, 0.84, 0.97, 0.05, 0.79, 0.123, 0.976, 0.01 so the feature with very low contribution can be removed from the dataset for classification here only 5 features can be taken into consideration which has their dependency score above 0.7, as it has multiple benefits for improvement in model performance. The use of an optimized feature selection technique reduces data size, training time, and detection time and safeguards against overfitting, leading to overall better performance enhancement of the network intrusion detection system.
There are many feature extraction algorithms like chi-square, recursive feature elimination, Pearson's correlation coefficient, ANOVA F-test, etc. In this work, chi-square method in select k best feature technique for extracting relevant features is used.

3.1.4 Classification algorithms

There are many standard supervised and unsupervised machine learning algorithms. Implemented at some of the machine learning algorithms and evaluated its performance score on a proportion of the dataset reserved for testing.

Support Vector Machine

A SVM uses an optimal hyperplane for classification, which is a line for 2D space, a plane for 3D space, and so on. In other words, the SVM maximizes the margins between classes during its training phase by creating the hyperplanes.

Decision Tree

This classifier follows the flowchart-like structure to build a tree from top to down. The internal node in the tree is equivalent to a "test" on an attribute, and the branch is equivalent to the outcome for the test. The leaf node is equivalent to a class label or decision as an outcome after computing all attributes.

Naïve Bayes

This classifier is based on the Bayes' Theorem, where many algorithms are used from a family by sharing a common principle. Here, each pair of classified features are independent of each other. This classifier is simple and effective. It can build an ML model and predict quickly. It is also known as a probabilistic classifier as it predicts using probability. In supervised learning, every classifier has to undergo the training phase and testing phase. The dataset is prepared after optimization of the feature set, which selects the relevant set. The data of selected features is divided into two subsets - training and testing.

KNN

k-NN is a supervised learning approach that is simple and gets trained fairly well on large training sets. It follows the assumption that data points with similar characteristics reflect similar outcomes. k-NN method predicts an unknown data sample based on majority neighbors among k neighbors. It is robust to noisy training data and is effective in case of a large number of training examples. It requires more time to compute the distance from each instance to all training data samples. K-NN classifies using a majority voting on nearby points, so if we assume that similar types of attacks feature vectors are nearby to each other, it should produce a good performance.

K-means clustering

K means cluster is an unsupervised clustering algorithm that groups objects based on their feature values into K disjoint clusters (where K is a positive integer specifying the number of clusters). It clusters data based on the Euclidean distance between the data, classifying objects into the same cluster if they have similar feature values.

Random Forest

RF makes multiple decision trees during its training from the dataset given as input to work upon the unknown data. For unknown input, each decision tree votes for a specific class as per its training, and finally, the majority vote decides the class of unknown data. It is able to work upon the massive and multidimensional dataset effectively that makes it appropriate for an IoT network.

Optimization of best performed model

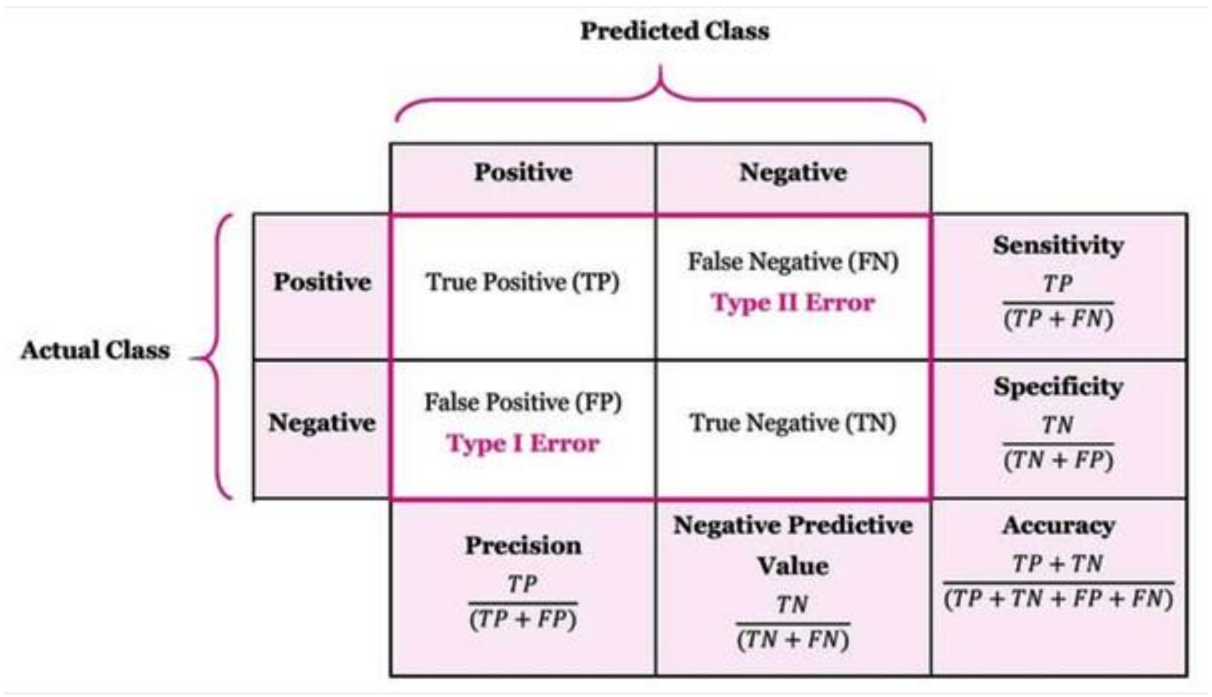
Among all above machine learning classifier algorithms one of the best performed model is to be optimized in a method so that it aims to achieve a improvement in accuracy of classification of network attacks in an intrusion detection systems.

After optimization of model again apply the classification on proposed algorithm and compare it from standard machine learning algorithms earlier applied for its performance.

3.1.5 Performance evaluation

The test dataset is now used to predict our model and for evaluation, on basis of various performance metric values such as the accuracy score, precision, recall, f-1 score, and a confusion matrix. A confusion matrix shows the number of correct and incorrect predictions made by the model against the actual outcomes (target value) in the data.

True positives are when the prediction of a result attacks and in reality, it is also an attack that means positive predicted for positive. True negatives are when the prediction of result is normal and in reality, it is normal means negative is predicted for negative. False positives are when the prediction of a result is normal but in reality, it is attack means negative predicted for positive. False negatives are when the prediction of result is attack but in reality, it is normal. All above performance metric bases are calculated from the confusion matrix.



		Predicted Class		
		Positive	**Negative**	
Actual Class	**Positive**	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

Figure 5 Performance evaluation by Confusion Matrix

- Accuracy is defined as the percentage of true predictions calculated by dividing true predictions by total predictions.
- Precision is defined as the fractions of that are attack and predicted true with all attacks in reality.
- The recall is defined as the fractions of prediction that attack and in reality, is also an attack with total attack predicted.
- F1 Score is calculated from the values of Precision and Recall. F1 score, in general, has more importance in the metric score for comparison. F1 score is calculated by finding the harmonic mean of precision and recall.

Overfitting is another prominent issue in machine learning; it occurs when your model does not generalize well to new data. It can appear when you refine your model too much that it also has learnt all of the noise in the dataset, meaning that although the performances seem to be making improvements, it performs worse when you come to test it on new data.

3.2 Implementation & Results

Implementation of proposed work is done on CICIDS 2017 dataset read using panda's python library in jupyter notebook. After the dataset is loaded in the jupyter notebook the process flow of an implementation of the proposed approach is shown by the figure @ below:

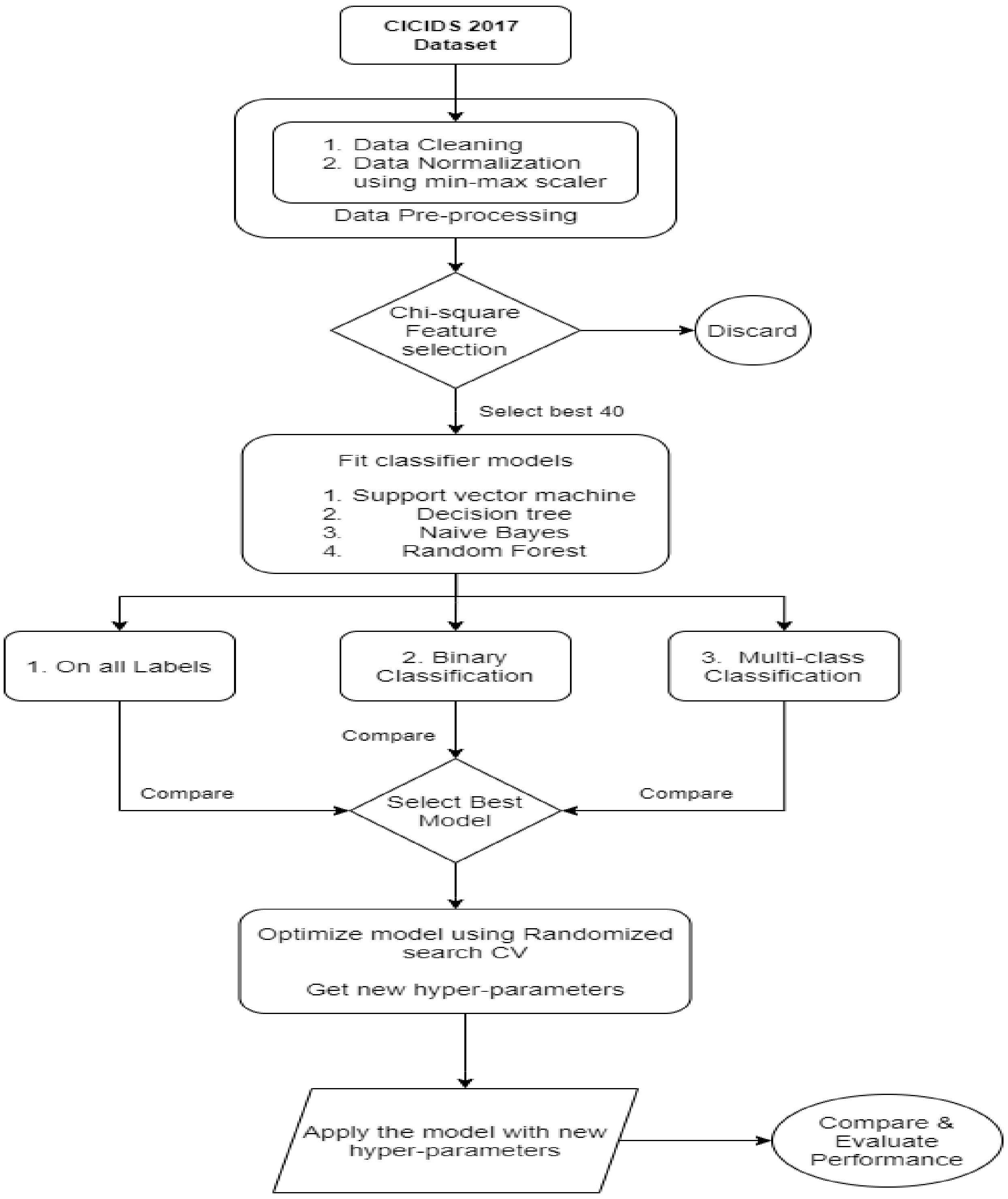


Figure 6 Flow chart of implemented work

Dataset Collection & Analyzing

To make a network intrusion detection system a benchmark network data should be used to prove the effectiveness and superior performance of our work. In this work, for detecting the network attack a network intrusion detection systems dataset CICIDS 2017 is used.

Analyzing CICIDS-2017 dataset

This dataset contains the realistic background of network traffic events generated by twenty-five users. The profiles of users were detailed, including specific purpose protocols such as Secure Shell (SSH), HTTP, HTTPS, email, and FTP. This dataset covers different types of common attack scenarios. It is publicly accessible in Packet Capture (PCAP) format and CSV files format. There are eight CSV files to perform our work all these files were clumped into a single csv file cicids.csv.
The CICIDS-2017 was created with the actual traces of benign and fourteen types of attacks from the network traffic data, where 2,830,108 records are available in the dataset. 2,358,036 records belong to the benign traffic, and 471,454 records belong to attack records. The CICIDS-2017 dataset contains 84 features, where the last column represents the traffic status or target attribute. This dataset is one of the popular datasets which includes new attacks as its uniqueness. Table @ depicts the distribution of fifteen class labels in the CICIDS2017 dataset.

Table 1 All attack label counts in CICIDS 2017 dataset

Attack labels in CICIDS 2017 dataset	Attack Count
Benign	2,273,097
DoS Hulk	231,073
Port Scan	158,930
DDoS	128,027
DoS Golden Eye	10,293
FTP Patator	7,938
SSH Patator	5,987
DoS Slowloris	5,796
DoS Slowhttptest	5,499
Bot	1,966
Web attack: Brute Force	1,507
Web attack: XSS	652
Infiltration	36
Web attack: SQL Injection	21
Heartbleed	11

These above 15 categories of attacks are classified further into 7 different attack categories as shown in below Table @ is:

Table 2 Grouped Attack categories of all attack labels

Attack Categories	**All Attack labels**
Benign	Benign
DoS	DoS Golden eye, DoS Slowloris, DoS Slowhttpptest, DoS Hulk
Probe	Port scan
DDoS	DDoS
Brute force	FTP patator, SSH Patator
Botnet	Bot
Web attack	Web attack: Brute Force, Web attack: XSS

Different types of attacks in the CICIDS-2017 dataset are briefed below as:

1. DoS Hulk: In this HULK tool is used by the attacker to carry out DoS attack on a web server.

2. Port Scan: In this, an attacker attempts to collect information such as OS and active service on the targeted machine by hitting packets of changing destination ports.

3. DDoS: In this, the attacker performs a distributed DoS attack on the victim machine through multiple machines under the control of a server machine.

4. DoS GoldenEye: In this GoldenEye tool is used by the attacker to carry out DoS attack.

5. FTP Patator: In this the attacker performs brute force attack using FTP Patator for guessing File Transfer Protocol (FTP) login password.

6. SSH Patator: In this the attacker perform the brute force attack using SSH Patator to guess Secure Shell (SSH) login Password.

7. DoS Slow Loris: In this, the Slow Loris tool is used by the attacker to carry out a attack.

8. DoS Slow HTTP Test: In this HTTP request is exploited by the attacker to conduct HTTP flooding on a server.

9. Botnet: In this trojans are rooted by the attacker to take control of victim machines for making a network to conduct desired attacks remotely. The hampered machine is known as Bot.

10. Web Attack: Brute Force: In this trial and error approach is adopted by the attacker to get privilege of user personal data like password, Personal Identification Number (PIN), etc.

11. Web Attack: XSS: In this, the attacker targets the trusted websites to carry out the attack by sending malicious scripts.

12. Infiltration: In this infiltration, techniques are used by the attacker to infiltrate and get unlawful access inside the system within a network.

13. Web Attack: SQL Injection: In this, the attacker tries to insert SQL queries in an entry field on the data-driven applications to obtain or update the confidential data.

14. Heart Bleed: In this OpenSSL protocol is exploited by the attacker to root malicious data in OpenSSL memory to obtain unauthorized access to the confidential information.

Dataset Cleaning & Pre-processing

After this loading and analyzing dataset it is checked that dataset has any Null, Nan values so that it should be replace with mean, median, etc. In this work, Dataset is very large so that it may contain redundant and duplicate data which has no contribution toward the targeted outcome. Removal of these redundancies is data cleaning, In our work, CICIDS 2017 dataset has Fwd_header_length column repeated twice hence its redundant so one is removed. Another there are 6 attributes which have single unique values which also has no contribution to the outcome. Hence this is also removed, now dataset is cleaned, For preprocessing it involves Data normalization and Data encoding , Data splitting are performed.

Data Normalization

The normalization of feature or attribute is done to limit the numerical values of data in a range (usually 0 -1) without affecting range differences of actual values or loosing the information. For example, if the first column values range from 0 to 1, and the other column values range from 10,000 to 10,00,000, the variance in the two columns can lead to problems in modeling and analysis. Normalization helps in the generation of new values within the specified range without affecting the general distribution and results. Eq. (1) is used to normalize the values of various attributes present in the datasets.

$$x' = a + \frac{(x - \min(x))(b - a)}{\max(x) - \min(x)}$$

Equation 1 Min-max scaler equation

Above equation represents the way of doing min-max scaling of numerical features where, X = feature value , a = lower boundary range of feature, b = higher boundary of feature value, Min(x) and Max(x) denotes the minimum and maximum value of all tuples in that feature attribute respectively.

Data Encoding

Dataset is checked for if any categorical data type is present or not if present then before proceeding too classifier algorithms it must be encoded. Here in this work CICIDS 2017 dataset ha no categorical data type hence there is no requirement for applying Machine learning classifier algorithms.

Feature Selection

Extracting the appropriate network feature set is referred to as feature selection and is a challenging task. The use of an optimized feature selection technique reduces data size, training time, and detection time and safeguards against overfitting, leading to overall performance enhancement of the NIDS system. Various optimization techniques can be used for feature selection but for feature selection in this work on CICIDS 2017 dataset chi-square scoring function in select k best algorithm is used.

Chi-square Feature selection method

A chi-square test is used in statistics to test the independence of two events. Chi-square test is used for categorical features in a dataset. We calculate Chi-square between each feature and the target and select the desired number of features with best Chi-square scores. It determines if the association between two categorical variables of the sample would reflect their real association in the population.

Chi square scores is calculated as:

$$X^2 = \frac{(Observed\ frequency - Expected\ frequency)^2}{Expected\ frequency}$$

Equation 2 Chi-square equation

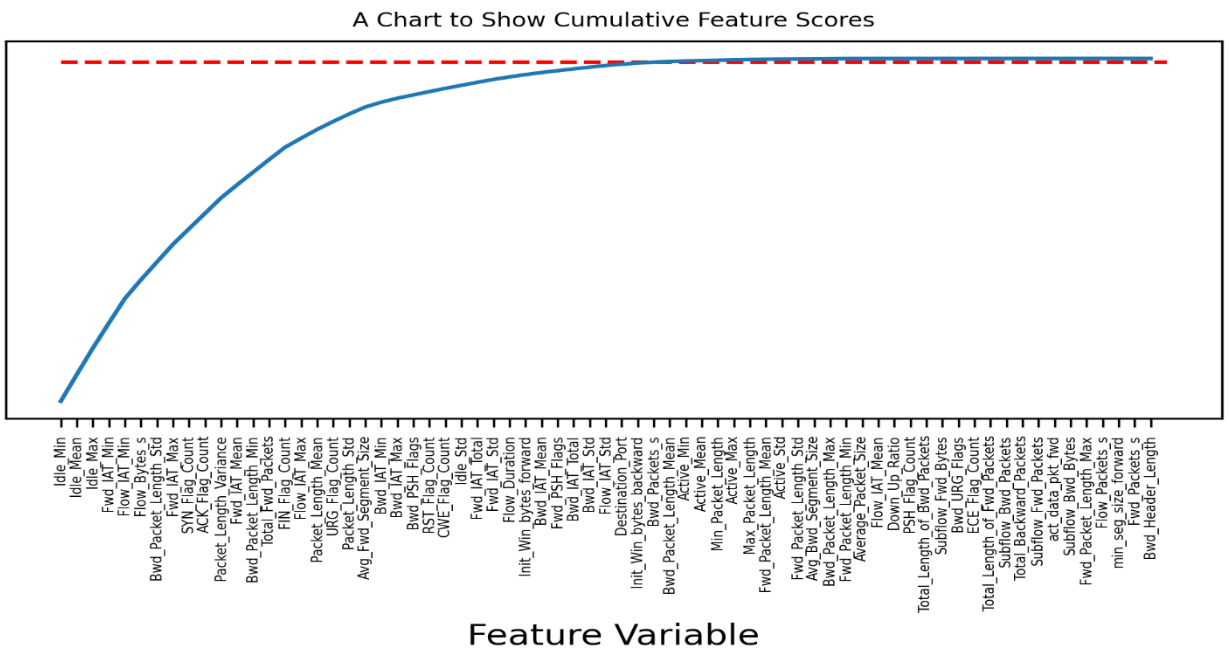
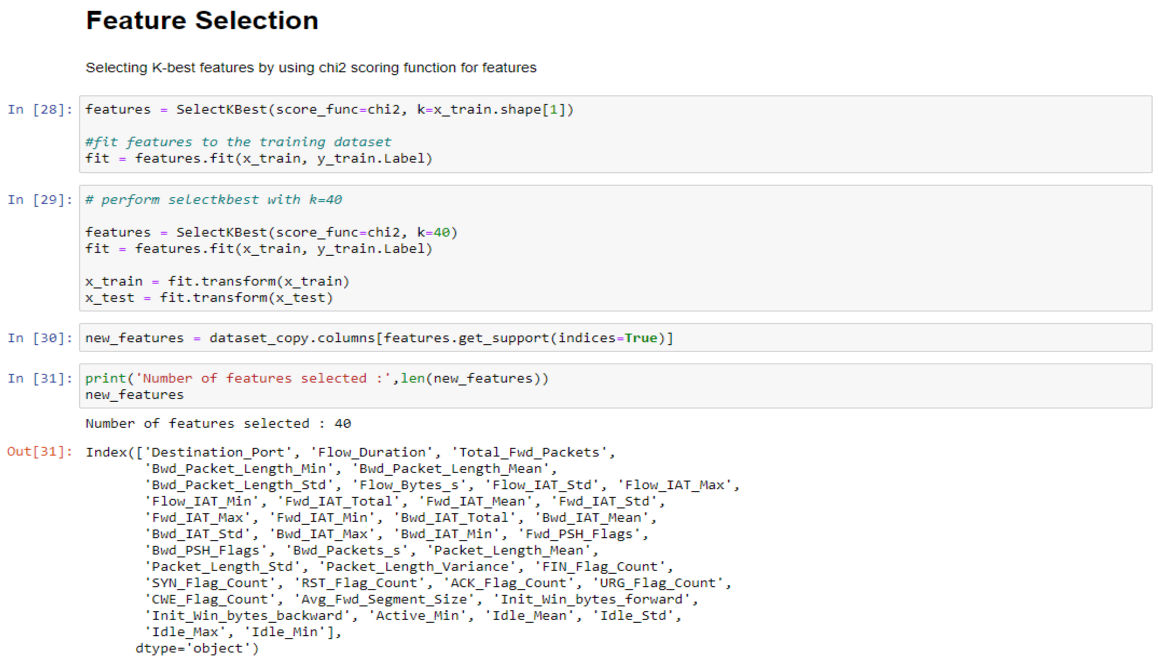


Figure 7 Cummulative feature scores using chi-square

In our work, after calculating feature scores by plotting a graph in Figure @ showing cumulative feature importances of all the features it is observed that only approx. 40 features out of 80 are relevant which can give almost 99% of information so it is best to select only those features which also reduces the training time of model. By Select K best method using chi-square as feature scoring parameter 40 most relevant features were selected for building model and evaluation by above mentioned method and feature selected are shown in snapshot of code below:



Feature Selection

Selecting K-best features by using chi2 scoring function for features

```
In [28]: features = SelectKBest(score_func=chi2, k=x_train.shape[1])

         #fit features to the training dataset
         fit = features.fit(x_train, y_train.Label)
```

```
In [29]: # perform selectkbest with k=40

         features = SelectKBest(score_func=chi2, k=40)
         fit = features.fit(x_train, y_train.Label)

         x_train = fit.transform(x_train)
         x_test = fit.transform(x_test)
```

```
In [30]: new_features = dataset_copy.columns[features.get_support(indices=True)]
```

```
In [31]: print('Number of features selected :',len(new_features))
         new_features

         Number of features selected : 40

Out[31]: Index(['Destination_Port', 'Flow_Duration', 'Total_Fwd_Packets',
                'Bwd_Packet_Length_Min', 'Bwd_Packet_Length_Mean',
                'Bwd_Packet_Length_Std', 'Flow_Bytes_s', 'Flow_IAT_Std', 'Flow_IAT_Max',
                'Flow_IAT_Min', 'Fwd_IAT_Total', 'Fwd_IAT_Mean', 'Fwd_IAT_Std',
                'Fwd_IAT_Max', 'Fwd_IAT_Min', 'Bwd_IAT_Total', 'Bwd_IAT_Mean',
                'Bwd_IAT_Std', 'Bwd_IAT_Max', 'Bwd_IAT_Min', 'Fwd_PSH_Flags',
                'Bwd_PSH_Flags', 'Bwd_Packets_s', 'Packet_Length_Mean',
                'Packet_Length_Std', 'Packet_Length_Variance', 'FIN_Flag_Count',
                'SYN_Flag_Count', 'RST_Flag_Count', 'ACK_Flag_Count', 'URG_Flag_Count',
                'CWE_Flag_Count', 'Avg_Fwd_Segment_Size', 'Init_Win_bytes_forward',
                'Init_Win_bytes_backward', 'Active_Min', 'Idle_Mean', 'Idle_Std',
                'Idle_Max', 'Idle_Min'],
               dtype='object')
```

Applying Classification algorithms

Now on preprocessed data feature selection is also applied and is reduced to 40 attributes one by one machine learning algorithm is applied and analyzed the results.

For each machine learning algorithm it is applied in 3 different categories :

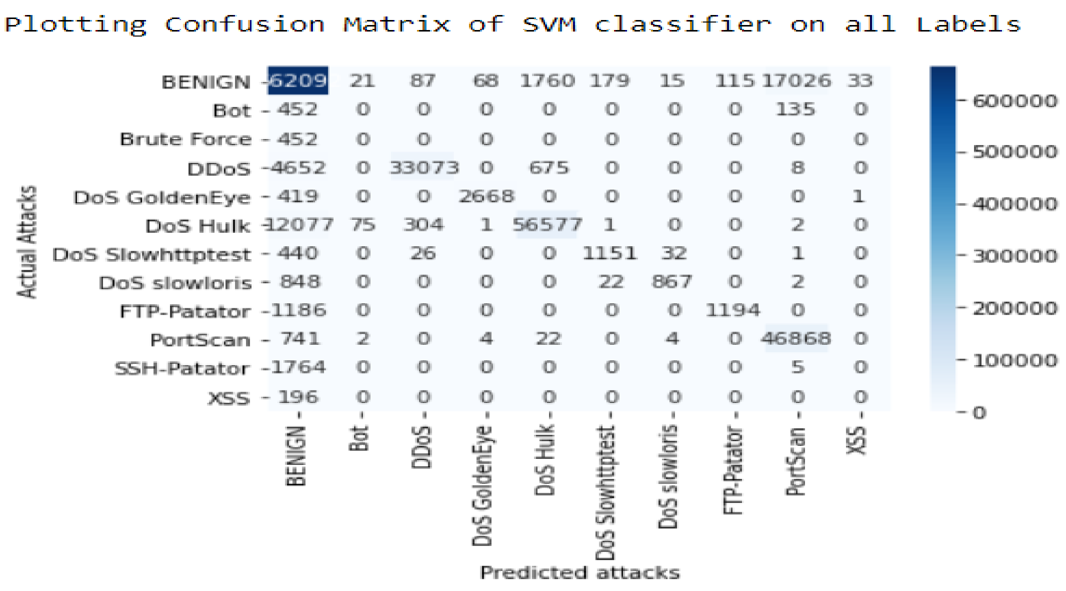
- For all Attack labels : In this category there are total 12 attack labels.
- Binary Classification : In this category there is only two label normal and attack.
- Multi-class classification : Here 7 different attack categories are there.

1. Support vector machine classifier (SVM)

Support Vector machine classifier is applied first with kernel parameter as linear.

a) On all Labels

On all Label model training time is approx. 462 seconds for SVM classifier while testing time is 0.144 seconds. SVM model showed an accuracy of 94.83%. Confusion matrix for SVM on all Labels is shown below:



	BENIGN	Bot	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	FTP-Patator	PortScan	XSS
BENIGN	6209	21	87	68	1760	179	15	115	17026	33
Bot	452	0	0	0	0	0	0	0	135	0
Brute Force	452	0	0	0	0	0	0	0	0	0
DDoS	4652	0	33073	0	675	0	0	0	8	0
DoS GoldenEye	419	0	0	2668	0	0	0	0	0	1
DoS Hulk	12077	75	304	1	56577	1	0	0	2	0
DoS Slowhttptest	440	0	26	0	0	1151	32	0	1	0
DoS slowloris	848	0	0	0	0	22	867	0	2	0
FTP-Patator	1186	0	0	0	0	0	0	1194	0	0
PortScan	741	2	0	4	22	0	4	0	46868	0
SSH-Patator	1764	0	0	0	0	0	0	0	5	0
XSS	196	0	0	0	0	0	0	0	0	0

Figure 8 Confusion matrix plot for SVM classifier on all labels

b) Binary Classification

In Binary Classification model training time was 158 seconds and testing time being 0.72 seconds. In this model showed an accuracy of 92.72%. Confusion matrix for SVM on binary label is shown below:

```
Plotting Confusion Matrix of SVM classifier on binary Labels
```

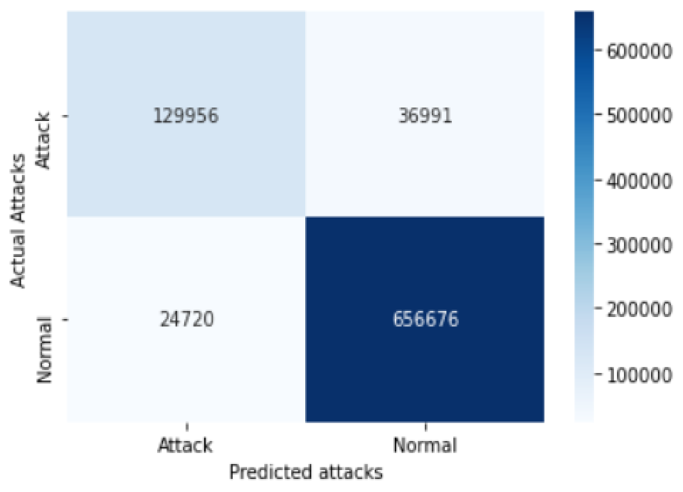


Figure 9 Confusion matrix plot of SVM on Binary classification

c) Multi-class classification

In Multi-class Classification model training time was 340 seconds and testing time being 0.89 seconds. In this model showed an accuracy of 94.43%. Confusion matrix for SVM on multi-class label is shown below:

```
Plotting Confusion Matrix of SVM classifier on multi-class Labels
```

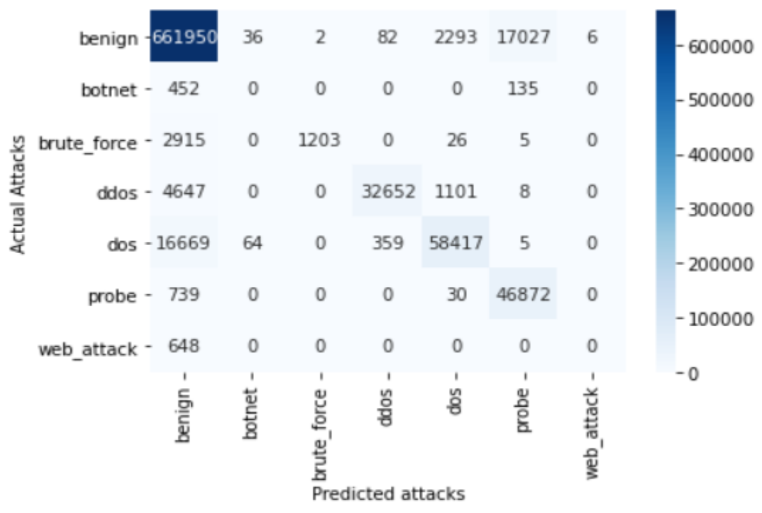


Figure 10 Confusion matrix plot of SVM on multi-class classification

2. Decision Tree

a) On all Labels

On all Label model training time is approx. 112.453 seconds for Decision Tree classifier while testing time is 0.239 seconds. Decision tree model showed an accuracy of 99.97%. Confusion matrix for Decision tree classifier on all Labels is shown below:

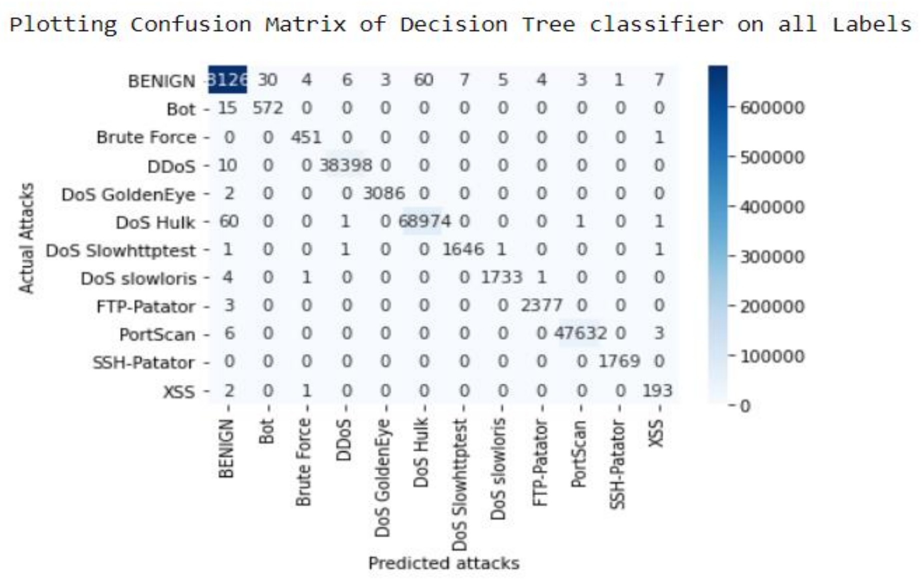


Figure 11 Confusion matrix plot of Decision tree on all labels

b) Binary Classification

In Binary Classification model training time was 113 seconds and testing time being 0.19 seconds. In this model showed an accuracy of 99.97%. Confusion matrix for Decision tree classifier on binary label is shown below:

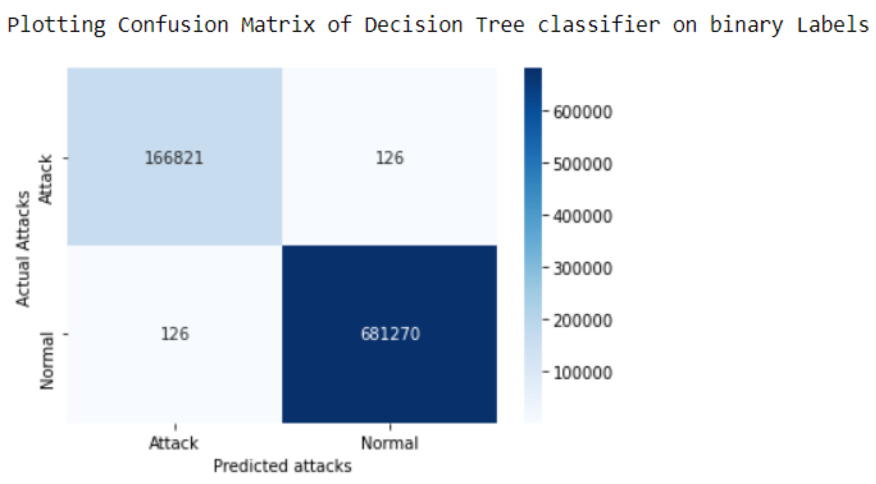


Figure 12 Confusion matrix plot of Decision tree on binary classification

c) Multi-class classification

In Multi-class Classification model training time was 113.71 seconds and testing time being 0.22 seconds. In this model showed an accuracy of 99.97%. Confusion matrix for Decision tree classifier on multi-class label is shown below:

```
Plotting Confusion Matrix of Decision Tree classifier on multi-class Labels
```

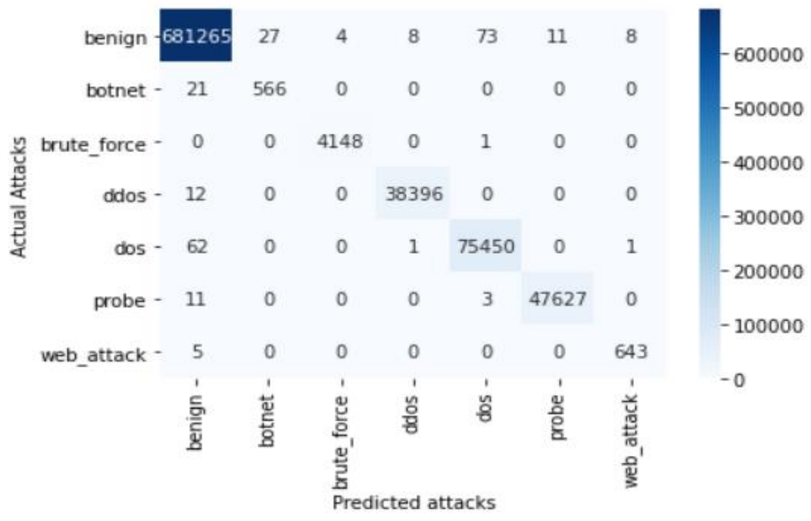


Figure 13 Confusion matrix plot of Decision tree on multi-class classification

3. Naïve Bayes

a) On all Labels

On all Label model training time is approx. 462 seconds for Naïve bayes classifier while testing time is 0.144 seconds. Naïve bayes model showed an accuracy of 94.83%. Confusion matrix for Naïve bayes on all Labels is shown below:

```
Plotting Confusion Matrix of Naive Bayes classifier on all Labels
```

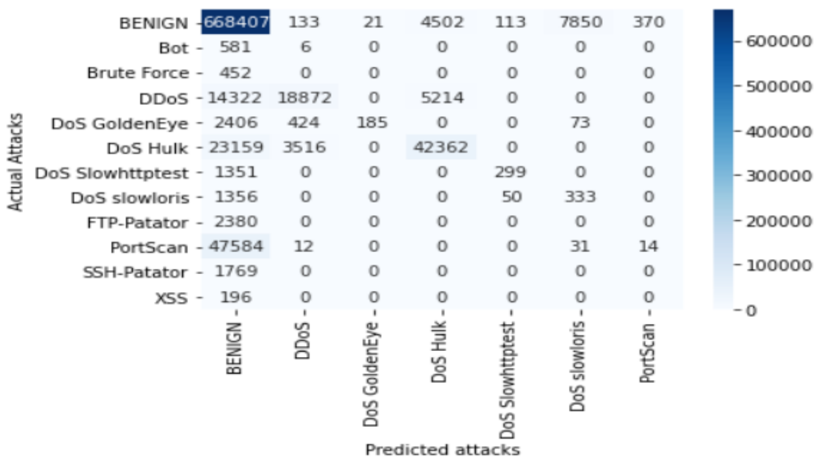


Figure 14 Confusion matrix plot of Naive Bayes on all labels

b) Binary Classification

In Binary Classification model training time was 158 seconds and testing time being 0.72 seconds. In this model showed an accuracy of 92.72%. Confusion matrix for Naïve bayes on binary label is shown below:

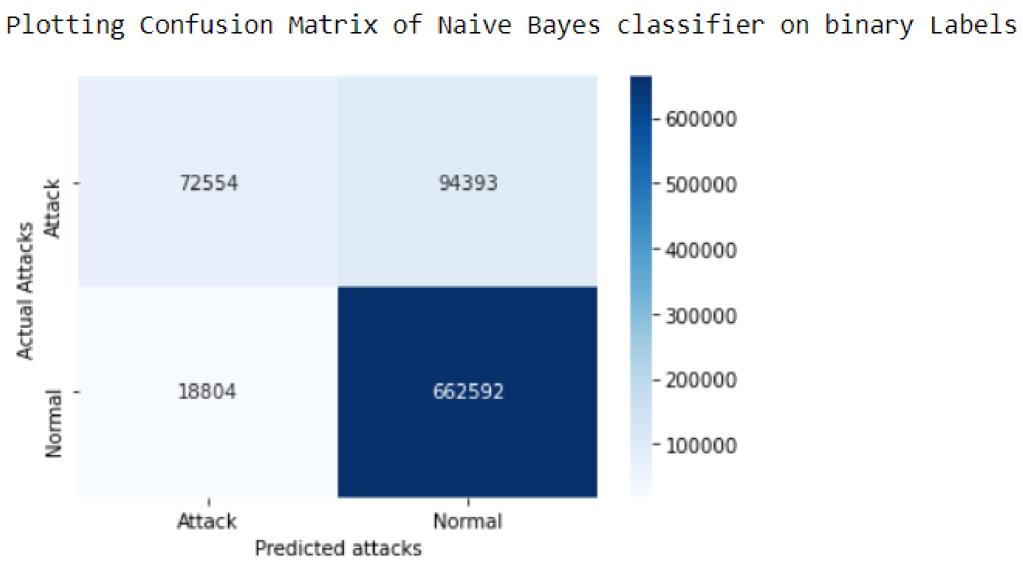
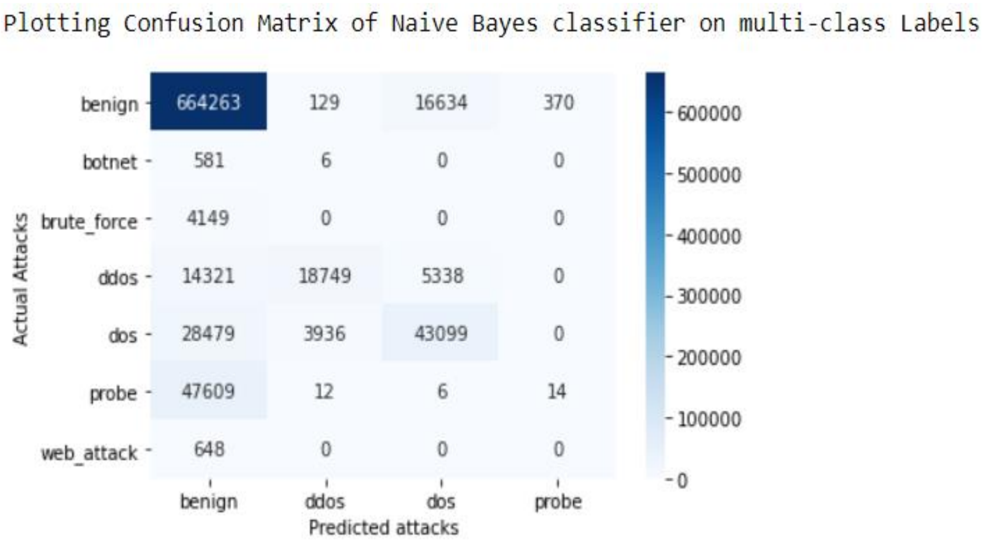


Figure 15 Confusion matrix plot of Naive Bayes on binary classification

c) Multi-class classification

In Multi-class Classification model training time was 340 seconds and testing time being 0.89 seconds. In this model showed an accuracy of 94.43%. Confusion matrix for Naïve bayes on multi-class label is shown below:



	benign	ddos	dos	probe
benign	664263	129	16634	370
botnet	581	6	0	0
brute_force	4149	0	0	0
ddos	14321	18749	5338	0
dos	28479	3936	43099	0
probe	47609	12	6	14
web_attack	648	0	0	0

Figure 16 Confusion matrix plot of Naive Bayes on multi-class labels

4. Random Forest Classifier

a) On all Labels

On all Label model training time is approx. 462 seconds for Random Forest classifier while testing time is 0.144 seconds. Random Forest model showed an accuracy of 94.83%. Confusion matrix for Random Forest on all Labels is shown below:

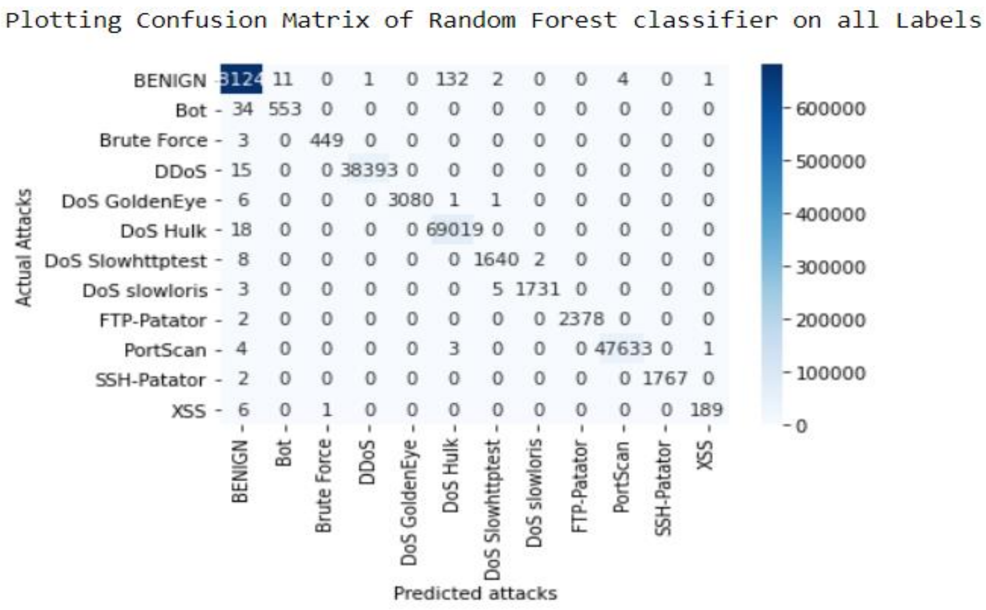


Figure 17 Confusion matrix plot of random Forest on all labels

b) Binary Classification

In Binary Classification model training time was 158 seconds and testing time being 0.72 seconds. In this model showed an accuracy of 92.72%. Confusion matrix for Random Forest on binary label is shown below:

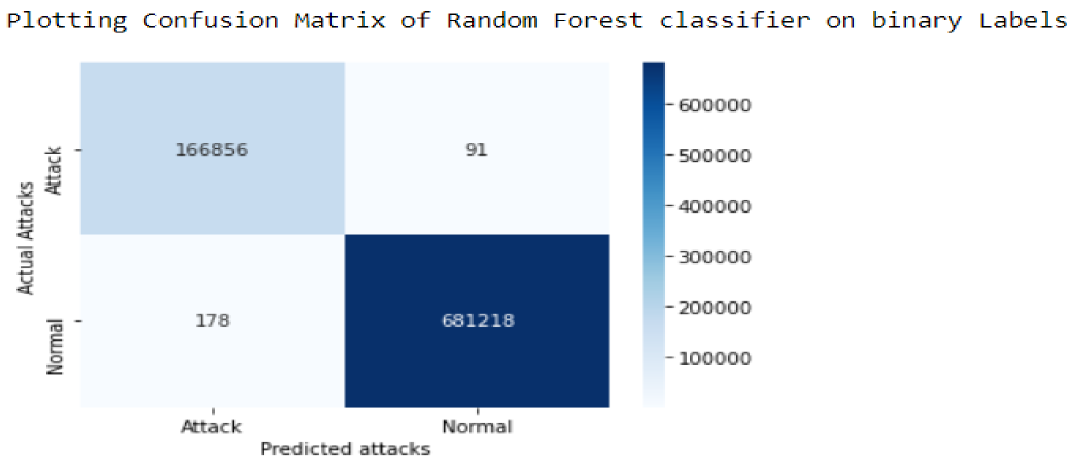


Figure 18 Confusion matrix plot of random Forest on binary labels

c) Multi-class classification

In Multi-class Classification model training time was 340 seconds and testing time being 0.89 seconds. In this model showed an accuracy of 94.43%. Confusion matrix for Random Forest on multi-class label is shown below:

```
Plotting Confusion Matrix of Random Forest classifier on multi-class Labels
```

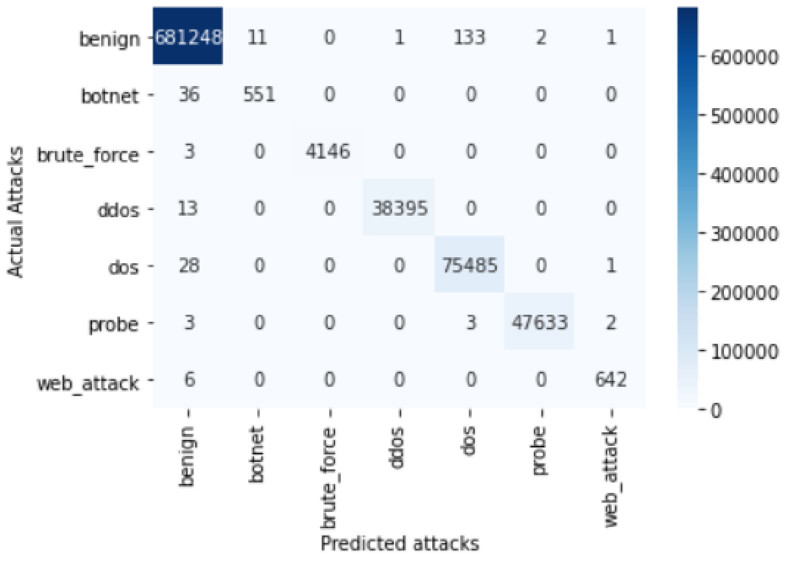


Figure 19 Confusion matrix plot of random Forest on multi-class labels

Comparing Performance scores of all models

Comparison of performance scores of all machine learning models is shown in below Table @.

Table 3 Performance score comparison of all ML models

	SVM	**Decision Tree**	**Naïve Bayes**	**Random Forest**	
On all labels	94.83% 61.03% 51.64% 54.94%	99.97% 98.84% 99.56% 99.19%	86.10% 34.43% 20.99% 22.51%	99.96% 99.66% 99.02% 99.34%	1. Accuracy 2. Precision 3. Recall 4. F1-score
Binary Classification	92.72% 89.34% 87.10% 88.16%	99.97% 99.95% 99.95% 99.95%	86.65% 83.47% 70.34% 74.15%	99.96% 99.94% 99.95% 99.94%	1. Accuracy 2. Precision 3. Recall 4. F1-score
Multi-class Classification	94.43% 66.04% 55.27% 57.41%	99.97% 99.11% 99.35% 99.23%	85.59% 34.19% 29.05% 30.68%	99.71% 99.60% 99.96% 99.27%	1. Accuracy 2. Precision 3. Recall 4. F1-score

On analyzing all the implemented standard machine learning model it is clearly visible that Random Forest, Decision tree out-performed the classification with very high accuracy and low false alarm rate.

Machine learning models also take a collection of parameters as input, these values are known as hyperparameters and control the behavior of the model (such as the depth of the tree in decision trees and random forests). We can tune these hyperparameters to and the values which yield the optimal results for our problem. For our model, a random forest, listed are the parameters which we will tune, and a description of them according to the scikit-learn library.

To approach towards a more scalable and better classifier method I have optimized the best standard performed model that is random forest classification by tuning the hyper -parameters to get a little improvement in the accuracy.

Optimizing Random Forest Classifier

Random forest algorithm accepts several parameters these are called Hyper-parameters. Hyperparameters is like the settings of an algorithm that can be adjusted to optimize performance, just as we might turn the knobs of an AM radio to get a clear signal (or your parents might have!). While model parameters are learned during training — such as the slope and intercept in a linear regression — hyperparameters must be set by the data scientist before training.

In the case of a random forest, hyperparameters include the number of decision trees in the forest and the number of features considered by each tree when splitting a node. (The parameters of a random forest are the variables and thresholds used to split each node learned during training). Scikit-Learn implements a set of sensible default hyperparameters for all models, but these are not guaranteed to be optimal for a problem.

Random forest algorithm has following hyper-parameters :

- n estimators: The number of trees in the forest.
- max depth: The maximum depth of the tree.
- min samples split: The minimum number of samples required to split a decision node.
- min samples leaf: The minimum number of samples required to be a leaf node.
- max features: The number of features to consider when looking for the best split.
- bootstrap: Whether bootstrap samples are used when building trees. If false, the whole dataset is used to build each tree.

The best hyperparameters are usually impossible to determine ahead of time, and tuning a model is where machine learning turns from a science into trial-and-error based engineering.

First, we consider the number of estimators parameter. This controls the number of decision trees that we evaluate in our random forest tress. Figure @ shows the F1 scores for different values of the n estimators parameter for the random forest on the validation dataset. The highest peak is when n estimators=800. However, Figure @ shows why we do not choose this as the value for the number of estimators. Time taken for classifiation increases linearly with the number of estimators, so setting n estimators=800 would incur a greater time consumption.

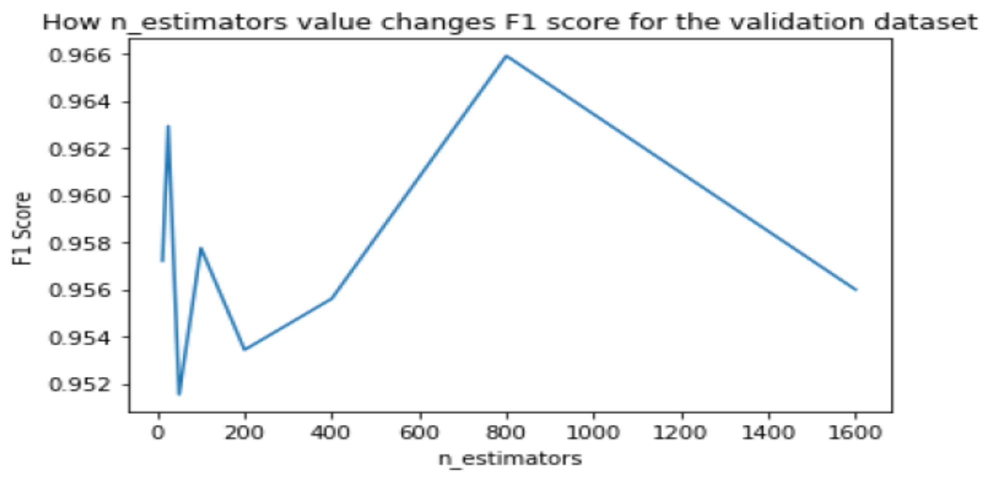


Figure 20 F1-score v/s n-estimators plot for optimizing

Although n_estimators=800 is visible was highest, but that many number of estimators is not realistic in terms of classificaiton times. We go with the second peak - n_estimators = 25 and reduces the time signinificantly as shown in Figure @.

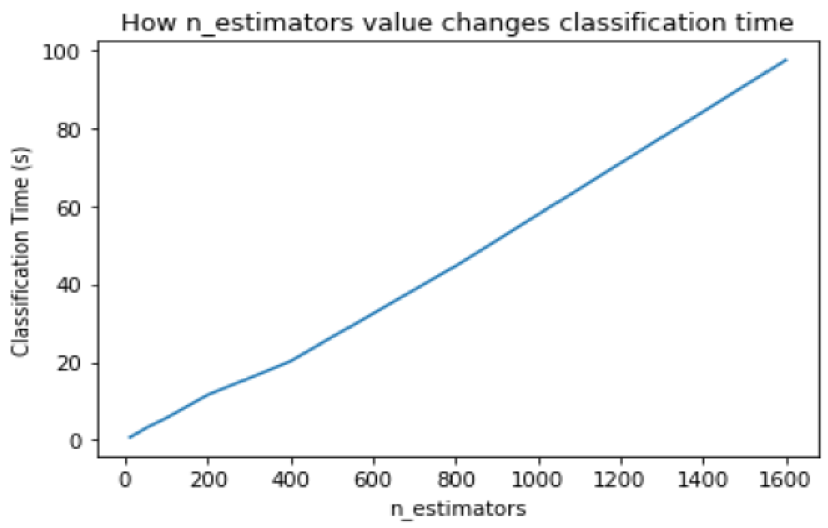
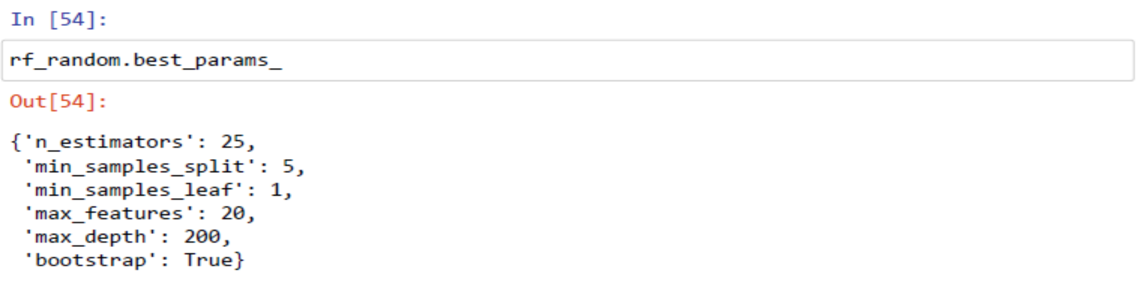


Figure 21 n-estimators v/s classification time for optimizing

To optimise the remaining hyperparameters, we perform a randomised search. This search strategy creates a grid of different values for each hyper-parameter. It then performs training and testing on random combinations of these hyperparameter values, providing each combination with a score. We use RandomizedSearchCV from the scikit-learn library to implement parameter optimisation, and then call best params to return the parameter setting that gave the best results on the hold out data. This provides us with our final parameter results:



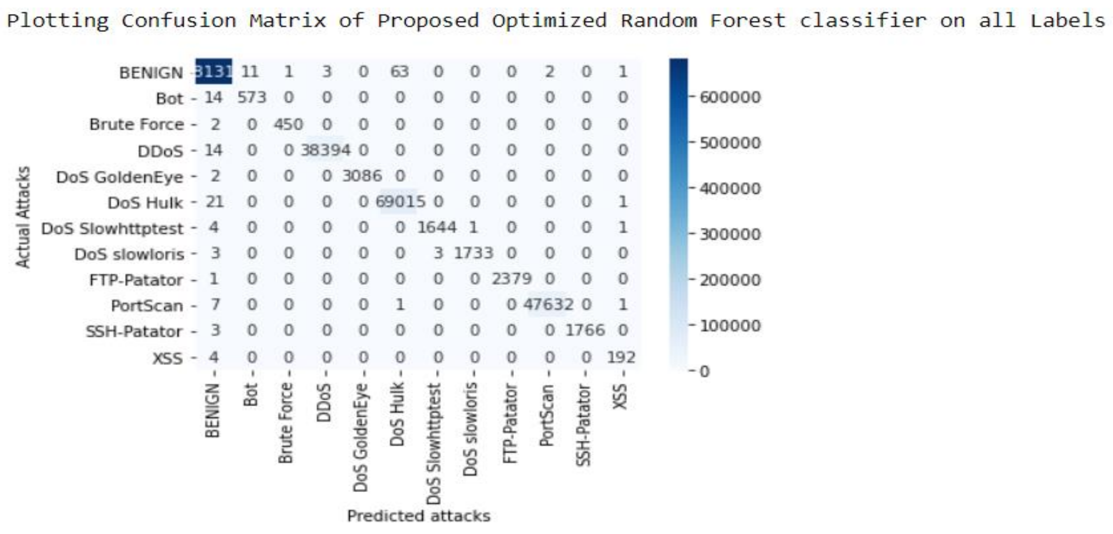
```
In [54]:
rf_random.best_params_
Out[54]:
{'n_estimators': 25,
 'min_samples_split': 5,
 'min_samples_leaf': 1,
 'max_features': 20,
 'max_depth': 200,
 'bootstrap': True}
```

Now after getting all the Hyper-parameters of Randomized random forest approach the proposed algorithm is applied on all 3 categories of classification and results were compared and studied.

5. Proposed Optimized Random Forest Classifier

a) On all Labels

On all Label model training time is approx. 462 seconds for optimized random forest classifier while testing time is 0.144 seconds. SVM model showed an accuracy of 94.83%. Confusion matrix for optimized random forest on all Labels is shown below:



Plotting Confusion Matrix of Proposed Optimized Random Forest classifier on all Labels

Actual Attacks \ Predicted attacks	BENIGN	Bot	Brute Force	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	FTP-Patator	PortScan	SSH-Patator	XSS
BENIGN	3131	11	1	3	0	63	0	0	0	2	0	1
Bot	14	573	0	0	0	0	0	0	0	0	0	0
Brute Force	2	0	450	0	0	0	0	0	0	0	0	0
DDoS	14	0	0	38394	0	0	0	0	0	0	0	0
DoS GoldenEye	2	0	0	0	3086	0	0	0	0	0	0	0
DoS Hulk	21	0	0	0	0	69015	0	0	0	0	0	1
DoS Slowhttptest	4	0	0	0	0	0	1644	1	0	0	0	1
DoS slowloris	3	0	0	0	0	0	3	1733	0	0	0	0
FTP-Patator	1	0	0	0	0	0	0	0	2379	0	0	0
PortScan	7	0	0	0	0	1	0	0	0	47632	0	1
SSH-Patator	3	0	0	0	0	0	0	0	0	0	1766	0
XSS	4	0	0	0	0	0	0	0	0	0	0	192

Figure 22 Confusion matrix plot for proposed model on all labels

b) Binary Classification

In Binary Classification model training time was 158 seconds and testing time being 0.72 seconds. In this model showed an accuracy of 92.72%. Confusion matrix for optimized random forest on binary label is shown below:

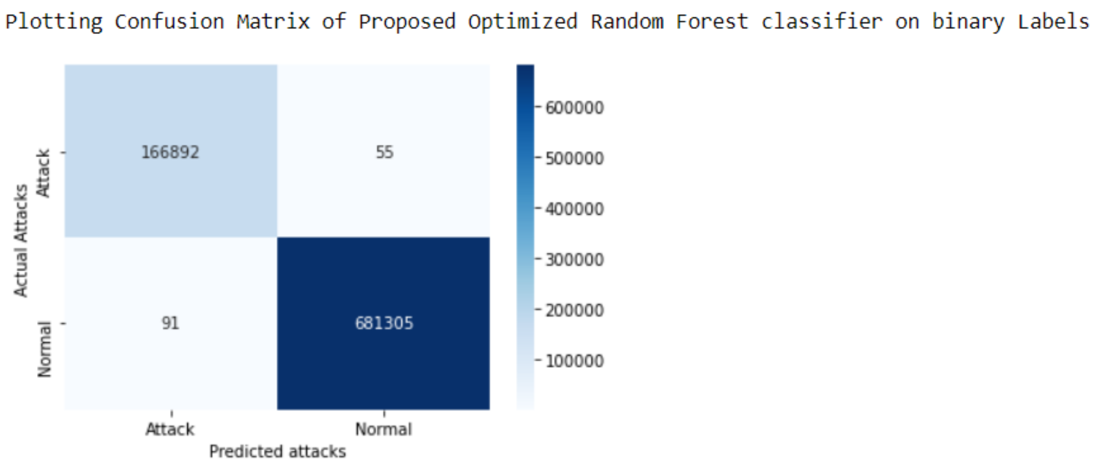
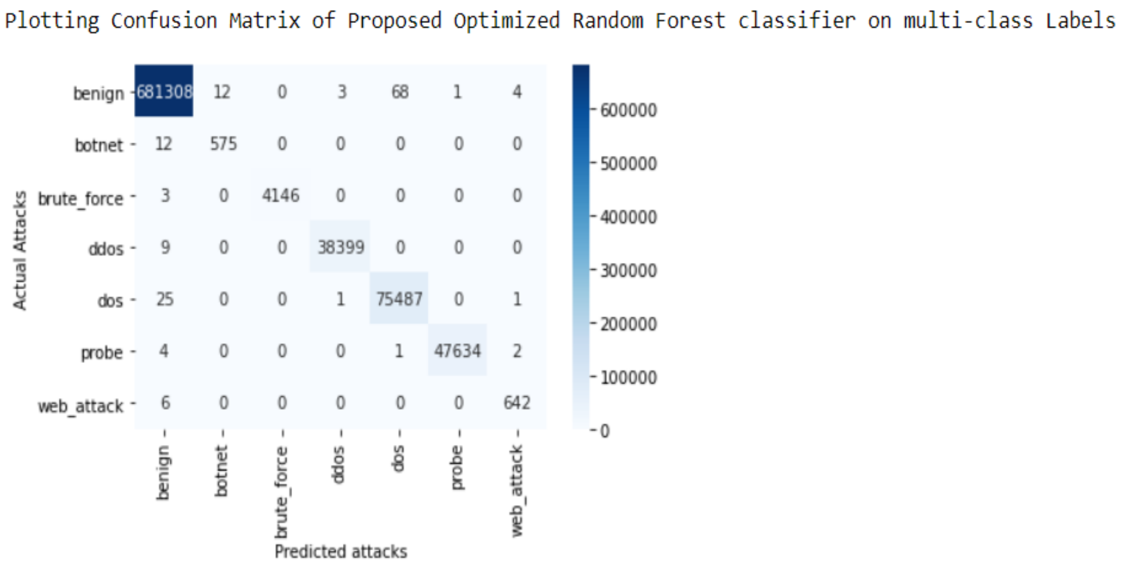


Figure 23 Confusion matrix plot of proposed model on binary labels

c) Multi-class classification

In Multi-class Classification model training time was 340 seconds and testing time being 0.89 seconds. In this model showed an accuracy of 94.43%. Confusion matrix for optimized random forest on multi-class label is shown below:



	benign	botnet	brute_force	ddos	dos	probe	web_attack
benign	681308	12	0	3	68	1	4
botnet	12	575	0	0	0	0	0
brute_force	3	0	4146	0	0	0	0
ddos	9	0	0	38399	0	0	0
dos	25	0	0	1	75487	0	1
probe	4	0	0	0	1	47634	2
web_attack	6	0	0	0	0	0	642

Figure 24 Confusion matrix plot of proposed model on multi-class labels

Analyzing performance score of proposed model:

The proposed randomized random forest algorithm with tuned hyper-parameters has attained performance scores shown in below Table @.

Table 4 Proposed model Performance scores

Performance scores of Proposed Model (Randomized random Forest)				
	Accuracy	**Precision**	**Recall**	**F1-score**
On all Labels	99.98%	99.62%	99.50%	99.56%
Binary Classifier	99.98%	99.96%	99.97%	99.97%
Multi-class Classifier	99.98%	99.53%	99.55%	99.54%

On analyzing the results of our proposed algorithm, I can successfully state that this model has outperformed all the previous models and achieve a mile stone accuracy in each category of attack for binary as well as multi -class classification.

3.3 Key Contributions

- Understanding of network attacks and its detection, different types of intrusion detection systems, feature optimization techniques, machine learning based classification techniques.

- Implemented machine learning algorithms to detect network attack in CICIDS 2017 dataset and a new randomized random forest feature selection algorithm is proposed. This algorithm is a tuned hyper-parameter of random forest algorithm.

- Comparison of the proposed GWO-CSA-FS-DSAE NIDS model with other similar models and is found that the proposed NIDS model has achieved the highest performance for binary as well as multiclass classification.

3.4 Comparison of proposed model and previous studies

A detailed comparison of proposed randomized random forest classifier is done with previous studies in below Table @.

Work	Dataset	Method	Type	Classifier	Accuracy
Hongpo Zhang et al (2020)	CICIDS 2017	SGM	Multi-class	RF MLP CNN	93.08% 99.60% 99.85%
Cengiz Colak et al (2017)	CICIDS 2017	SMOTE BMCD	Binary Multi-class	GLM Boost Logit Boost RF MLP NB	82.47% 96.95% 99.32% 94.82% 75.35%
Proposed Model	**CICIDS 2017**	**Chi-square**	**Binary** **Multiclass**	**Randomized random forest**	**99.98%** **99.98%**

3.5 Summary

In this chapter, proposed work and its implementation is discuused in detail.

Chapter 4

Conclusion and Future Works

1.1. Conclusions

The proposed decision tree classifier algorithm helps the network administrator to check the incoming and outgoing information or data over a network is original or showing any intrusive behavior from the known attacks if found separates it and releases an alert to the administrator. For feature selection, the proposed method is a hybrid of ANOV F-TEST univariate feature selection and recursive feature elimination technique that finally selects 13 most relevant subset of feature for each attack category and then by building a decision tree classifier model for classification of attacks in DoS, Probe, R2L, U2R, Normal categories. In this work recursive feature elimination continuously selects a subset of features and builds the model repeatedly till all features in the dataset are over. Evaluation of the performance of our classification model is successfully done on different metric score values which result in 99% approx. accuracy for all categories of attacks. So, this reduced subset of features extracted from the proposed method proved that accuracy is improved, and the best 13 features for each category result in good accuracy with a minimal time of model training.

1.2. Future works

This work illustrated the importance of using an optimal subset of features with a suitable classification algorithm for designing an intrusion detection system. On evaluation, the advantage of the proposed method is that reducing the features to best contributors improves the model performance by eliminating redundant features. Therefore, feature selection could be improved with different classification techniques. The proposed method in this study may have a higher execution time which could be reduced in the future. The feature selection algorithms with various machine learning algorithms should give better accuracy in lesser execution time on all classifier algorithms. Concepts of Deep learning could be involved like creating neural hidden layers of features and classification approach to have lesser execution time.

REFERENCES

[1] Tiwari, Mohit & Kumar, Raj & Bharti, Akash & Kishan, Jai. (2017). INTRUSION DETECTION SYSTEM. International Journal of Technical Research and Applications. 5. 2320-8163.

[2] Othman, Suad & Alsohybe, Nabeel & Ba-Alwi, Fadl & Zahary, Ammar. (2018). Survey on Intrusion Detection System Types. 7. 444-462.

[3] Elrawy, M., Awad, A. & Hamed, H. Intrusion detection systems for IoT-based smart environments: a survey. J Cloud Comp 7, 21 (2018). https://doi.org/10.1186/s13677-018-0123-6

[4] H. M. Imran, A. B. Abdullah, M. Hussain, S. Palaniappan, I. Ahmad. Intrusions Detection based on Optimum Features Subset and Efficient Dataset Selection. International Journal of Engineering and Innovative Technology (IJEIT). Vol. 2, Issue 6, December 2012, pp. 265-270.

[5] KDD Dataset details accessed https://kdd.ics.uci.edu/databases/kddcup99/task.html

[6] Preeti Aggarwal, Sudhir Kumar Sharma, "Analysis of KDD Dataset Attributes - Class wise for Intrusion Detection", Procedia Computer Science, Volume 57,2015, Pages 842-851, ISSN 1877-0509,https://doi.org/10.1016/j.procs.2015.07.490.

[7] Nguyen, G., Dlugolinsky, S., Bobák, M. et al. Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey. Artif Intell Rev 52, 77–124 (2019). https://doi.org/10.1007/s10462-018-09679-z

[8] Tiwari, Mohit & Kumar, Raj & Bharti, Akash & Kishan, Jai. (2017). INTRUSION DETECTION SYSTEM. International Journal of Technical Research and Applications. 5. 2320-8163.

[9] Othman, Suad & Alsohybe, Nabeel & Ba-Alwi, Fadl & Zahary, Ammar. (2018). Survey on Intrusion Detection System Types. 7. 444-462.

[10] Yu, Kun-Ming & Wu, Ming-Feng & Wong, Wai-Tak. (2008). Protocol-based classification for intrusion detection. 3.

[11] K. V. Rajkumar, V. Vaidehi, S. Pradeep, N. Srinivasan, and M. Vanishree, "Application Level IDS using Protocol Analysis," 2007 International Conference on Signal Processing, Communications and Networking, 2007, pp. 355-359, DOI: 10.1109/ICSCN.2007.350762.

[12] Garg, Akash & Maheshwari, Prachi. (2016). A hybrid intrusion detection system: A review. 1-5. 10.1109/ISCO.2016.7726909.

[13] Khraisat A, Gondal I, Vamplew P (2018) An anomaly intrusion detection system using C5 decision tree classifier. In: Trends and applications in knowledge discovery and data mining. Springer International Publishing, Cham, pp 149–155

[14] Khraisat, A., Gondal, I., Vamplew, P. et al. Survey of intrusion detection systems: techniques, datasets, and challenges. Cybersecurity 2, 20 (2019). https://doi.org/10.1186/s42400-019-0038-7

[15] Ashoor, Asmaa & Gore, Sharad. (2011). Difference between Intrusion Detection System (IDS) and Intrusion Prevention System (IPS). 196. 10.1007/978-3-642-22540-6_48.

[16] KDD Dataset details accessed https://kdd.ics.uci.edu/databases/kddcup99/task.html

[17] Dataset KDD https://github.com/defcom17/NSL_KDD

[18] Rish, Irina. (2001). An Empirical Study of the Naïve Bayes Classifier. IJCAI 2001 Work Empir Methods Artif Intell. 3.

[19] Patel, Harsh & Prajapati, Purvi. (2018). Study and Analysis of Decision Tree-Based Classification Algorithms. International Journal of Computer Sciences and Engineering. 6. 74-78. 10.26438/IGCSE/v6i10.7478.

[20] Wang, Lishan. (2019). Research and Implementation of Machine Learning Classifier Based on KNN. IOP Conference Series: Materials Science and Engineering. 677. 052038. 10.1088/1757-899X/677/5/052038.

[21] S. Ghosh, A. Dasgupta, and A. Swetapadma, "A Study on Support Vector Machine based Linear and Non-Linear Pattern Classification," 2019 International Conference on Intelligent Sustainable Systems (ICISS), 2019, pp. 24-28, DOI: 10.1109/ISS1.2019.8908018.

[22] Qiong Ren, Hui Cheng, and Hai Han, "Research on machine learning framework based on random forest algorithm", AIP Conference Proceedings 1820, 080020 (2017) https://doi.org/10.1063/1.4977376

[23] A. Alazab, M. Hobbs, J. Abawajy and M. Alazab, "Using feature selection for intrusion detection system," 2012 International Symposium on Communications and Information Technologies (ISCIT), 2012, pp. 296-301, DOI: 10.1109/ISCIT.2012.6380910.

[24] B. Abolhasanzadeh, "Nonlinear dimensionality reduction for intrusion detection using auto-encoder bottleneck features," 2015 7th Conference on Information and Knowledge Technology (IKT), 2015, pp. 1-5, DOI: 10.1109/IKT.2015.7288799.

[25] Aslahi-Shahri BM, Rahmani R, Chizari M, Maralani A, Eslami M, Golkar MJ, Ebrahim A (2015) A hybrid method consisting of GA and SVM for the intrusion detection system. Neural Comput Appl.doi:10.1007/s00521-015-1964-2

[26] J. Hussain, S. Lalmuanawma, and L. Chhakchhuak, "A two-stage hybrid classification technique for network intrusion detection system," International Journal of Computational Intelligence Systems, vol. 9, no. 5, pp. 863–875, 2016.

[27] Hosseinzadeh Aghdam, Mehdi & Kabiri, Peyman. (2016). Feature Selection for Intrusion Detection System Using Ant Colony Optimization. International Journal of Network Security. 18. 420-432.

[28] Kim, J. et al. "Method of intrusion detection using deep neural network." 2017 IEEE International Conference on Big Data and Smart Computing (BigComp) (2017): 313-316.

[29] S. Omar, H. H. Jebur, and S. Benqdara, "An adaptive intrusion detection model based on machine learning techniques," International Journal of Computer Applications, vol. 70, no. 7, pp. 1–5, 2017.

[30] Khraisat, A., Gondal, I., Vamplew, P. et al. Survey of intrusion detection systems: techniques, datasets, and challenges. Cybersecurity 2, 20 (2019). https://doi.org/10.1186/s42400-019-00387

[31] P.-F. Marteau, "Sequence covering for efficient host-based intrusion detection," IEEE Transactions on Information Forensics and Security, vol. 14, no. 4, pp. 994–1006, 2019.

[32] J. Gu, L. Wang, H. Wang, and S. Wang, "A novel approach to intrusion detection using SVM ensemble with feature augmentation," Computers & Security, vol. 86, pp. 53–62, 2019.

[33] T.H. Divyasree, K.K. Sherly, A network intrusion detection system based on ensemble CVM using efficient feature selection approach, Procedia Comput. Sci. 143 (2018) 442–449.

[34] M. Jin, Z. Xu, R. Li, and D. Wu, "Fuzzy ARTMAP ensemble-based decision making and application," Mathematical Problems in Engineering, vol. 2013, Article ID 124263, 7 pages,2013.

[35] S. Mohammadi and A. Namadchian, "A new deep learning approach for anomaly base IDS using a memetic classifier," International Journal of Computers Communications & Control, vol. 12, no. 5, pp. 677–688, 2017.

[36] Y. Jia, M. Wang, and Y. Wang, "Network intrusion detection algorithm based on deep neural network," IET Information Security, vol. 13, no. 1, pp. 48–53, 2019.

[37] A. A. Aburomman and M. B. Ibne Reaz, "A novel SVM-kNNPSO ensemble method for an intrusion detection system," Applied Soft Computing, vol. 38, pp. 360–372, 2016.

[38] A. A. Aburomman and M. B. I. Reaz, "Ensemble of binary SVM classifiers based on PCA and LDA feature extraction for intrusion detection," in Proceedings of the 2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), pp. 636–640, Xi'an, China, October 2016.

[39] Mirsky Y, Doitshman T, Elovici Y, Shabtai A (2018)"Kitsune: an ensemble of autoencoders for online network intrusion detection," arXiv preprint arXiv:1802.09089

[40] M. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita. Network Anomaly Detection: Methods, Systems, and Tools, IEEE communications surveys & tutorials, vol. 16, No. 1, first quarter 2014, pp. 303-336.

[41] Li X, Chen W, Zhang Q, Wu L. Building an auto-encoder intrusion detection system based on random forest feature selection. Comput Secure. 2020;95:101851.

[42] Zhao F, Zhang H, Peng J, Zhuang X, Na S-G. A semi-self-taught network intrusion detection system. Neural Comput Appl. 2020;32:17169–79.

[43] G. Kaur, V. Saxena, J.P. Gupta, “Detection of TCP targeted high bandwidth attacks using self-similarity”, Journal of King Saud University - Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2017.05.004, 2017.

[44] B. Neethu, “Adaptive intrusion detection using machine learning,” IJCSNS International Journal of Computer Science and Network Security, vol. 13, no. 3, pp. 118–124, 2013.

[45] Kanimozhi V, Jacob TP. Calibration of various optimized machine learning classifiers in network intrusion detection system on the realistic cyber dataset CSE-CIC-IDS2018 using cloud computing. Int J Eng Appl Sci Technol.2019;4(6):2143–455.

[46] P. Aggarwal and S. K. Sharma, “Analysis of KDD dataset attributes - class-wise for intrusion detection,” in Proceedings of the 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015), Ghaziabad, India, MAR 2015.

[47] P.Amudha, S.Karthik, S.Sivakumari, “Intrusion Detection Based on Core Vector Machine and Ensemble Classification Methods”, 2015 International Conference on Soft-Computing and Network Security (ICSNS -2015), Feb. 25 – 27, 2015, Coimbatore, INDIA.

[48] Welikala, R.A, Fraz, MM, Dehmeshki, J, Hoppe, A, Tah, V, Mann, S, Williamson, TH & Barman, SA (2015) Genetic Algorithm Based Feature Selection combined with dual classification for the

Automated Detection of proliferative Diabetic Retinopathy‟, Computerized Medical Imaging and Graphics, vol. 43, pp.64-77

[49] Wenjuan Lian, Guoqing Nie, Bin Jia, Dandan Shi, Qi Fan, and Yongquan Liang “An Intrusion Detection Method Based on Decision Tree-Recursive Feature Elimination in Ensemble Learning” https://doi.org/10.1155/2020/2835023

[50] I. M. Akashdeep, I. Manzoor, and N. Kumar, “A feature reduced intrusion detection system using ANN classifier,” Expert Systems with Applications, vol. 88, pp. 249–257, 2017.