AN INTRUSION DETECTION SYSTEM (IDS)

PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT

FOR THE AWARD OF THE DEGREE OF

MASTER OF COMPUTER APPLICATIONS

2017-2020

SUBMITTED BY

MOHD IMRAN

Roll No- 10535

CENTRAL UNIVERSITY OF HARYANA

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

UNDER THE SUPERVISION OF

External Supervisor

Mr. Amit Singh
Scientist
Indian Computer Emergency Response Team
Ministry of Electronics and Information technology
Government of India.

Internal Supervisor

Mr. Dr. Suraj Arya
Assistant Professor
Dept. of Computer Science
Central University of Haryana

ACKNOWLEDGEMENT

The bliss & euphoria that accompany the successful completion of any task would be incomplete without the expression of the appreciation of simple virtues to the people who made it possible. Therefore, with reverence, veneration and honors I express my heartiest gratitude and respectful regards to my mentor **Mr. Amit Singh, Scientist, CERT-IN,** for his valuable guidance, constructive criticism and consistent enthusiastic interest during the course of investigation, development and writing of report that led this work to its successful completion. He always shows me the right path whenever I need his help.

I acknowledge all those, whose guidance and encouragement has made this project successful. I Owen a huge debt of thanks to a large number of people without whom none of this would have been possible. I am very much grateful to the University and **Mr. Suraj Arya**, Assistant Professor, without his valuable attention, help and care; it would not have been possible for me to complete this project.

MOHD IMRAN

DECLARATION

I, MOHD IMRAN, student of MCA 6th semester, at Department of Computer Science of Central University of Haryana, Mahendragarh, hereby declare that the Six-month training/internship on **“Intrusion Detection System”** submitted to Department of Computer Science & Information Technology of **Central University of Haryana** is the original work conducted by me.

The information and data given in the report is authentic to the best of my knowledge.

This Six-month training/internship report is not being submitting to any other University or Institution for award of any other Degree, Diploma and Fellowship.

MOHD IMRAN

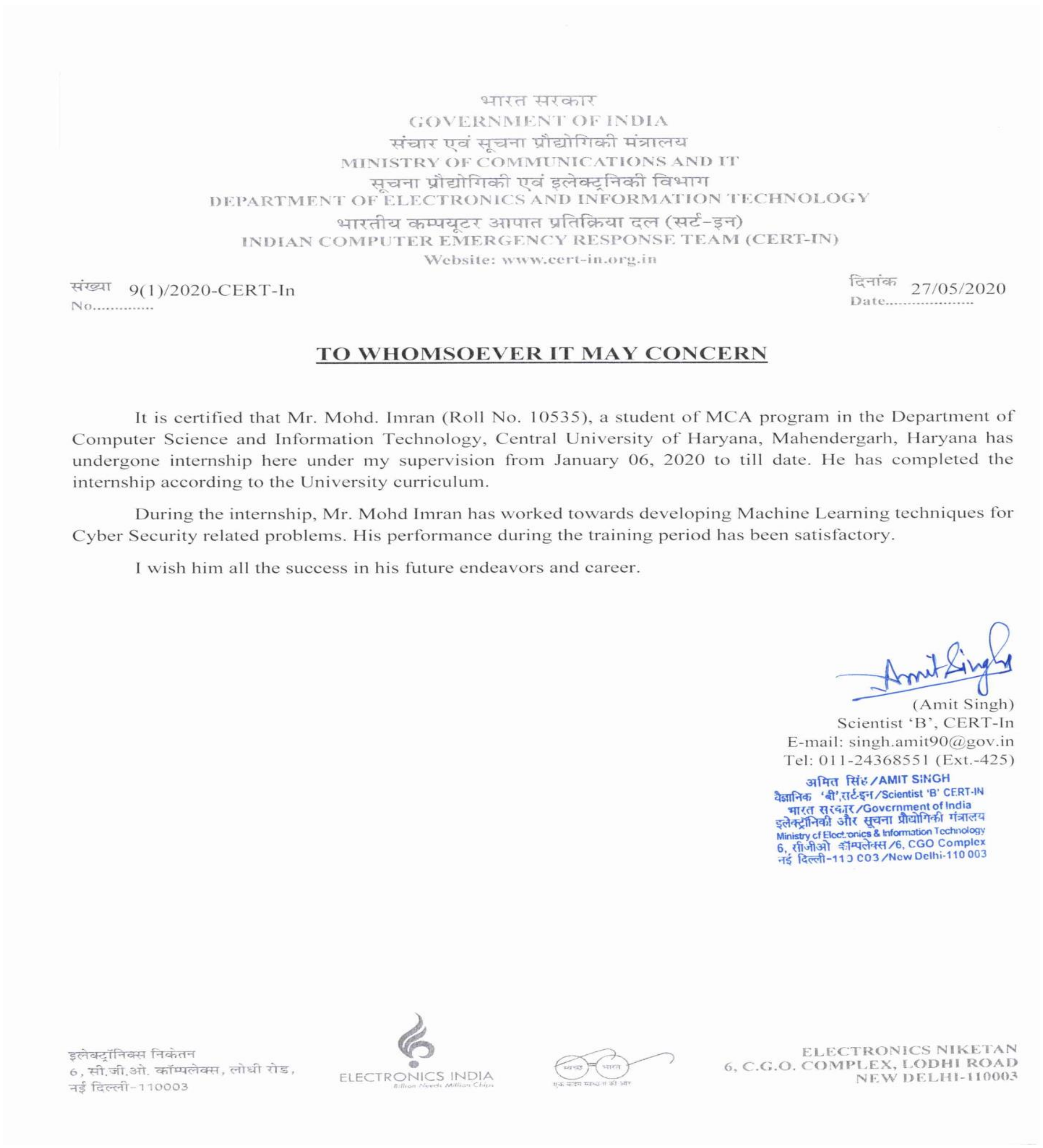
Roll No. 10535

MCA 6th Semester

Central University of Haryana

Department of Computer Science and Information Technology

CERTIFICATE BY ORGANIZATION



भारत सरकार
GOVERNMENT OF INDIA
संचार एवं सूचना प्रौद्योगिकी मंत्रालय
MINISTRY OF COMMUNICATIONS AND IT
सूचना प्रौद्योगिकी एवं इलेक्ट्रनिकी विभाग
DEPARTMENT OF ELECTRONICS AND INFORMATION TECHNOLOGY
भारतीय कम्पयूटर आपात प्रतिक्रिया दल (सर्ट-इन)
INDIAN COMPUTER EMERGENCY RESPONSE TEAM (CERT-IN)
Website: www.cert-in.org.in

संख्या No. 9(1)/2020-CERT-In

दिनांक Date 27/05/2020

TO WHOMSOEVER IT MAY CONCERN

It is certified that Mr. Mohd. Imran (Roll No. 10535), a student of MCA program in the Department of Computer Science and Information Technology, Central University of Haryana, Mahendergarh, Haryana has undergone internship here under my supervision from January 06, 2020 to till date. He has completed the internship according to the University curriculum.

During the internship, Mr. Mohd Imran has worked towards developing Machine Learning techniques for Cyber Security related problems. His performance during the training period has been satisfactory.

I wish him all the success in his future endeavors and career.

Amit Singh

(Amit Singh)
Scientist 'B', CERT-In
E-mail: singh.amit90@gov.in
Tel: 011-24368551 (Ext.-425)

अमित सिंह/AMIT SINGH
वैज्ञानिक 'बी',सर्ट-इन/Scientist 'B' CERT-IN
भारत सरकार/Government of India
इलेक्ट्रॉनिकी और सूचना प्रौद्योगिकी मंत्रालय
Ministry of Electronics & Information Technology
6, सीजीओ कॉम्पलेक्स/6, CGO Complex
नई दिल्ली-110 003/New Delhi-110 003

इलेक्ट्रॉनिक्स निकेतन
6, सी.जी.ओ. कॉम्पलेक्स, लोधी रोड,
नई दिल्ली-110003

ELECTRONICS INDIA

ELECTRONICS NIKETAN
6, C.G.O. COMPLEX, LODHI ROAD
NEW DELHI-110003

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Project Introduction-:

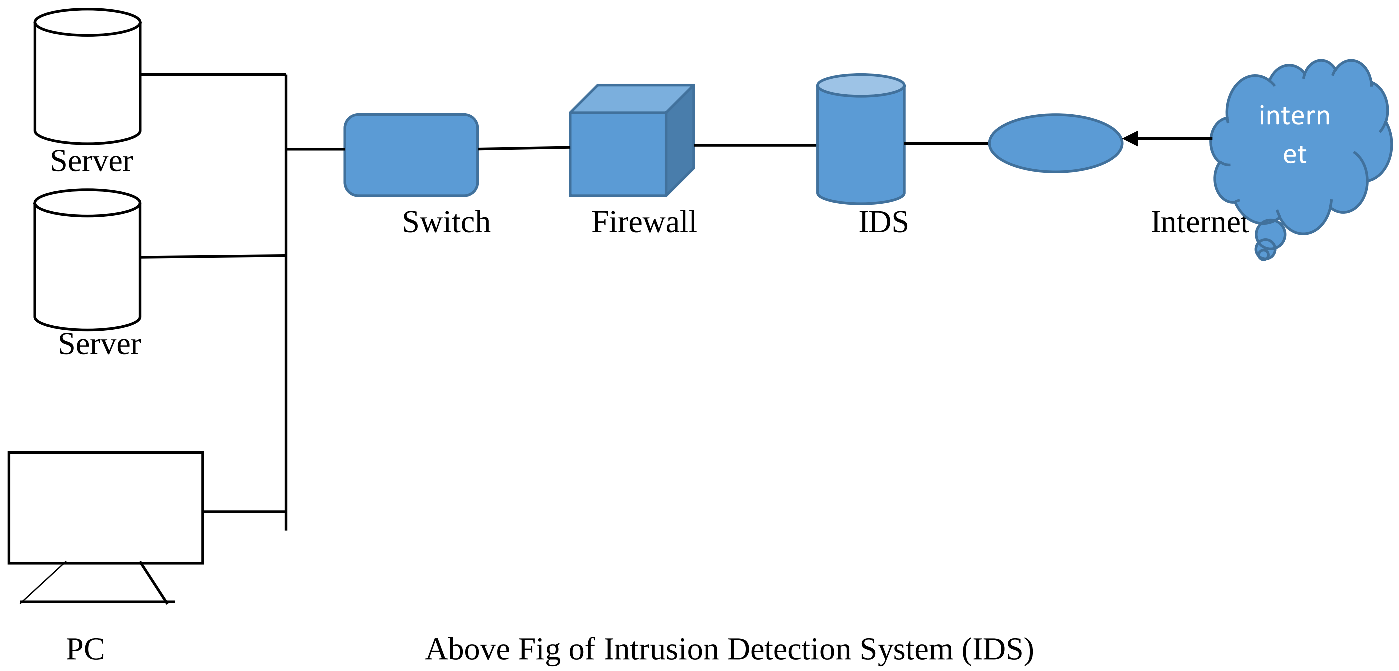
The Intrusion detection systems (IDS) are defined as efficient security tools which are used for improving the security of the communicating and the information systems as they primarily focus on detecting malicious network traffics. An IDS is seen to be very similar to many processes like firewalls, antivirus software and can access the control schemes. The IDS is classified depending on detection as the signature detection and anomaly detection systems, the system identify the traffic pattern or the application data as malicious and this requires an updated database for storing all the new attack signatures, whereas the anomaly detection system compare all activities against the normal defined behavior.

The main objective of the IDS system is detecting and them raising an alarm if the network is attacked. The best IDS process detects the new or more malicious attacks within a short time period and carries out the necessary actions The currently used IDS system do not show 100% accuracy. Many of machine learning techniques have been used for helping in the detection of the network attacks, improving the accuracy detection rate and developing effective classification and clustering models for distinguishing between normal and abnormal behavior packet. The procedure of detecting the intrusion accurately from the complete network traffic is classified as the classification problem. The IDS system are classified as per the detection methods used for identifying all the malicious attacks misuse or a signature detection technique identifies signature or patterns present in the existing attacks within the network traffic.

This misuse detection system requires an updated database for storing the new attack signatures. But new attacks are not detected until the system gets trained. The anomaly detection uses an approach based on detection of the traffic anomalies by identifying the behavior. Hence this show that IDS can handle new attacks, it is unable to detect or identify a particular attack pattern.

Several researches have used the Data mining for improving the IDS by offering an external intrusion detection that identifies the presence of any existing boundaries within a normal network activity. This helps in distinguishing between a normal and abnormal activity.

The classification models help in identifying any malicious attacks, improving the accuracy detection rate and decreasing false alarms. Many Machine learning algorithms have been proposed earlier for generating an efficient IDS like the Random Forest Classifier, Logistic Regression, Support Vector Machine (SVM), Decision Tree, K-Nearest Neighbors Classifier, Naïve Bayes Classifier, XGBoost Classifier, AdaBoost Classifier etc. All these algorithms are integrated with the different models for distinguishing between the malicious attacks and determining a normal behavior for detecting unknown malicious attacks.



Above Fig of Intrusion Detection System (IDS)

1.2 Classification of Intrusion Detection System:

IDS are classified into 5 types:

- 1. **Network Intrusion Detection System (NIDS):** Network intrusion detection systems (NIDS) are set up at a planned point within the network to examine traffic from all devices on the network. It performs an observation of passing traffic on the entire subnet and matches the traffic that is passed on the subnets to the collection of known attacks. Once an attack is identified or abnormal behavior is observed, the alert can be sent to the administrator. An example of an NIDS is installing it on the subnet where firewalls are located in order to see if someone is trying crack the firewall.

- 2. **Host Intrusion Detection System (HIDS) :** Host intrusion detection systems (HIDS) run on independent hosts or devices on the network. A HIDS monitors the incoming and outgoing packets from the device only and will alert the administrator if suspicious or malicious activity is detected. It takes a snapshot of existing system files and compares it with the previous snapshot. If the analytical system files were edited or deleted, an alert is sent to the administrator to investigate. An example of HIDS usage can be seen on mission critical machines, which are not expected to change their layout.
- 3. **Protocol-based Intrusion Detection System (PIDS):** Protocol-based intrusion detection system (PIDS) comprises of a system or agent that would consistently resides at the front end of a server, controlling and interpreting the protocol between a user/device and the server. It is trying to secure the web server by regularly monitoring the HTTPS protocol stream and accept the related HTTP protocol. As HTTPS is un-encrypted and before instantly entering its web presentation layer then this system would need to reside in this interface, between to use the HTTPS.
- 4. **Application Protocol-based Intrusion Detection System (APIDS):** Application Protocol-based Intrusion Detection System (APIDS) is a system or agent that generally resides within a group of servers. It identifies the intrusions by monitoring and interpreting the communication on application specific protocols. For example, this would monitor the SQL protocol explicit to the middleware as it transacts with the database in the web server.
- 5. **Hybrid Intrusion Detection System:** Hybrid intrusion detection system is made by the combination of two or more approaches of the intrusion detection system. In the hybrid intrusion detection system, host agent or system data is combined with network information to develop a complete view of the network system. Hybrid intrusion detection system is more effective in comparison to the other intrusion detection system. Prelude is an example of Hybrid IDS.

1.3 Detection Method of IDS-:

- 1. **Signature-based Method:** Signature-based IDS detects the attacks on the basis of the specific patterns such as number of bytes or number of 1's or number of 0's in the network traffic. It also detects on the basis of the already known malicious instruction sequence that is used by the malware. The detected patterns in the IDS are known as signatures.

  Signature-based IDS can easily detect the attacks whose pattern (signature) already exists in system but it is quite difficult to detect the new malware attacks as their pattern (signature) is not known.
- 2. **Anomaly-based Method:** Anomaly-based IDS was introduced to detect the unknown malware attacks as new malware are developed rapidly. In anomaly-based IDS there is use of machine learning to create a trustful activity model and anything coming is compared with that model and it is declared suspicious if it is not found in model. Machine learning based method has a better generalized property in comparison to signature-based IDS as these models can be trained according to the applications and hardware configurations.

1.4 How do Intrusion Detection System Work:

Intrusion detection systems are used to detect anomalies with the aim of catching hackers before they do real damage to a network. They can be either network- or host-based. A host-based intrusion detection system is installed on the client computer, while a network-based intrusion detection system resides on the network.

Intrusion detection systems work by either looking for signatures of known attacks or deviations from normal activity. These deviations or anomalies are pushed up the stack and examined at the protocol and application layer. They can effectively detect events such as Christmas tree scans and domain name system (DNS) poisonings.

1.5 Problem Description:

Many Intrusion Detection System (IDS) has been proposed in the current decade to evaluate the effectiveness of the IDS Canadian Institute of Cybersecurity presented a state of art data set named CICIDS 2018, consisting of latest threats and features. The dataset draws attentions of many researches as it represent threats which were not addressed by the older datasets. While undertaking an experimental research on CICIDS2018, it has been found that the data set has few major shortcomings. These issues are sufficient enough to biased the detection engine of any typical IDS. This paper explores the detailed characteristics of CICIDS2018 Data set and outlines, issues inherent to it. Finally, it also presents a combined data set by eliminating such issues for better classification and detection of any future intrusion detection engine.

My project work in this internship driven by the following motivations:

(a) There is no existing framework to determine the cut-off ranks for the features ranked by filter measures. Thus, researchers resort to the most convenient way, which is by arbitrarily selecting a cut-off rank (e.g., top-10, top-20, etc.). Obviously, such an arbitrary selection is not an optimized technique because it is prone to overstating or understating the cut-off rank. Hence, we are motivated to explore on the optimizations of feature dimensionality.

(b) Different machine learning datasets vary in characteristics such as feature type, number of features, etc. Through our preliminary experiments, we found that the effectiveness of the existing feature selection.

(c) In Intrusion Detection system (IDS), there is no common way where classifier is more predictable. When the objective is to maximize detection accuracy. Here eleven classifiers are testing with different feature subset combinations derived from various filter measures rankings.

1.6 Proposed approach:

Here in this study, the authors have proposed an enhanced Machine Learning algorithm which has been developed based on the Decision tree machine learning algorithm for enhancing the intrusion detection accuracy and IDS performance. One of the major problems in the construction of the decision tree involves the split value of the node, wherein the split value is the condition for dividing the data into two more subset. The ist split is known as the root node, while the rest of the splits are known as the root node, (also called as the terminal or the decision nodes), every internal is seen to split the space into Two more subspaces depending on the discrete function for the input attribute values. The split values provide an effective method for building the decision tree. For obtaining a smaller and more effective decision tree, the split must be based on the maximal gain. This proposed algorithm introduced a novel approach for the selection of the spilt values, estimation of the IG and the GR for constructing the decision tree. Here the attribute having the maximal normalised IG is utilized and the algorithm is seen to recur using small subset. The split procedure stops when all the examples in the subset are seen to use belong to one class. In the authors have tried to use the standard deviation coefficient as a significant factor for improving the machine learning algorithm.

The standard deviation (SD) can be defined as the number used for measuring how much the group is spread from the mean (average) value or from the expected value. The SD measures the data dispersement, which describe the spread of the data based on the mean value. Furthermore, SD describes the class distribution. If the attribute has a low SD describes the class distribution. If the attribute has a low SD value, it shows that the data value is closer to the value of the mean and has a simple distribution; where's if the attribute shows that the data value is more widespread from the mean and is highly randomized. Furthermore, the entropy and SD are different parameters however in many cases (not all) the entropy is seen to be dependent on the SD of distribution. The IG and entropy values along with SD help in deciding on which attributes the data is split while constructing the tree.

1.7 Related Work-:

Many Studies published earlier have focused on investigating the effect of applying the machine learning algorithms for enhancing the accuracy of the intrusion detection system, detection security and algorithms. Several IDS use a single algorithm system which classifies data as either anomalies or normal, but using one classifying system is unable to provide a precise detection system which detects and reports intrusions with a low rate of false alarms. Which needed different classifiers to be implemented, would improve the detection and make it very genuine, thereby improving the result quality. In this paper, the researchers have applied a two class classification strategy which is based on the K fold cross validation process, which would increase the rate of intrusion detection and also decrease the rate of false alarms.

This sections highlights the important outcomes of various published studies and discusses the limitations and strengths of the systems used by them. A rapid advancement in the field of information technology has introduced many machine learning techniques that can be implemented in the IDS. The researchers concluded that several methods could be applied that used various classifier. For example, some approaches decreased the variance and included boosting and bagging, whereas some decreased the bias. Some other methods, like cascading helped in developing new attributes. In such attribute every Classifier could handle a specific data set, whereas the rest handled by other classifier observed within the whole ensemble. Moreover, the authors also consider many ensemble methods depended on the voting system as the were considered to be sample processes that could generate the desired outcome. Many studies have stated that the hybrid method is popularly amongst the techniques for combining the classifier. This system is very reliable because it is able to correct all the errors produced by the other classifier thus improving the performance of the classifier. In this study I have noted that using the Machine learning algorithm Logistic Regression showed a low detection rate of 94.74%. Support Vector Classifier showed a low detection rate of 95.16%. and Stochastic Gradient Descent (SGD) Classifier showed a low detection rate of 81.95%.

In another study aim to identify the false positive rate by developing a system that used the machine learning algorithm for processes for decreasing the false positives. Their technique combined the Random Forest Classifier, Xgboost Classifier, Decision tree Classifier and K-Nearest Neighbors Classifier; SVM Classifier which would improve the efficiency and the accuracy of their IDS.

The SVM was used as the new binary model for classifying if the traffic was an attack or not after identifying the abnormal attacks.

Comprised of transferring the attacking traffic using a random forest. In this step, the probable attacks were transferred using the Decision tree. After integrating the Random Forest algorithm and also excluding any irrelevant Labels. The Decision Tree was seen to be a versatile system which could detect true positive or also detect if the alarm was new to their system. Also, this system raised a true positive alarm if it detected a leaf. But if no leaf was detected then the system would tag the alarm as new.

The Decision tree and the Random forest collectively for detecting the previously undetected attacks. This system was very effective as it showed an overall accuracy of K-Nearest Neighbors Classifier is 99.78% and a relatively lower false positive rate of 1.57 % Thought the system was seen to very efficient, the authors suggested that further research was to be carried out for the implementation of this model on many other network systems.

Several IDS classified the data using the anomaly detection or the misuse detection. Every approach had some advantages and some limitations. It must be mentioned that no IDS can modify all the problems within any system. The best detection system is one which provides a more acceptable security level, which is achieved by improving the efficiency of the detection of the bigger intrusion detects. The author have suggested that a perfect IDS process can provide a more accurate confidence report for the results seen which is an important measure for any IDS. As per many review articles, using the decision tree provided promising insights while improving the IDS performance.

Some threats faced by the virtualized systems are common threats that are encountered by any system. As they effect all the systems, and include the DoS attacks and the denial of service attacks. However, other threats or vulnerabilities are more specific for the virtual machines. Additionally, some virtual machine vulnerabilities occur due to a vulnerability which is seen in one of the virtual machine systems. They could also harm the systems in some cases. This is possible as a majority of the Virtual machines use similar physical hardware systems. For improving the attacks detection accuracy in the network systems, many researchers have focused on designing and developing some machine learning algorithm which are integrated within the algorithms identified the major abnormal attack within their network traffic, called as the denial of service (DoS) attack. The Dos attacks comprised of many forms of attacks like the Teardrop, Smruf, Land, Neptune and Back. The authors investigated the accuracy of the algorithms which were used for detecting the Dos attacks. All these algorithms which were used for detecting the DoS attacks. All these algorithms were used in different systematic techniques and included Logistic Regression, Random Forest, Support Vector Classifier, Stochastic Gradient Descent (SGD) Classifier, Xgboost Classifier, Decision tree Classifier and K-Nearest Neighbors Classifier Committee.

1.8 About organization:

CERT-In is a functional organization of Ministry of Electronics and Information Technology, Government of India, with the objective of securing Indian cyber space. CERT-In provides incident prevention and response services as well as Security quality management services.

- **a) Vision:**

  Proactive Contribution in Securing India's cyberspace.

- **b) Mission:**

  To enhance the security of India's Communications and Information Infrastructure through proactive action and effective collaboration.

- **c) Objectives:**
- Preventing cyber-attacks agent against the country's cyberspace.
- Responding to cyber-attacks minimizing damage, and recovery time reducing ‘national vulnerability to cyber-attacks.
- Enhancing security awareness among common citizens.
- **d) Function/Activities (allocation of Business Rules):**

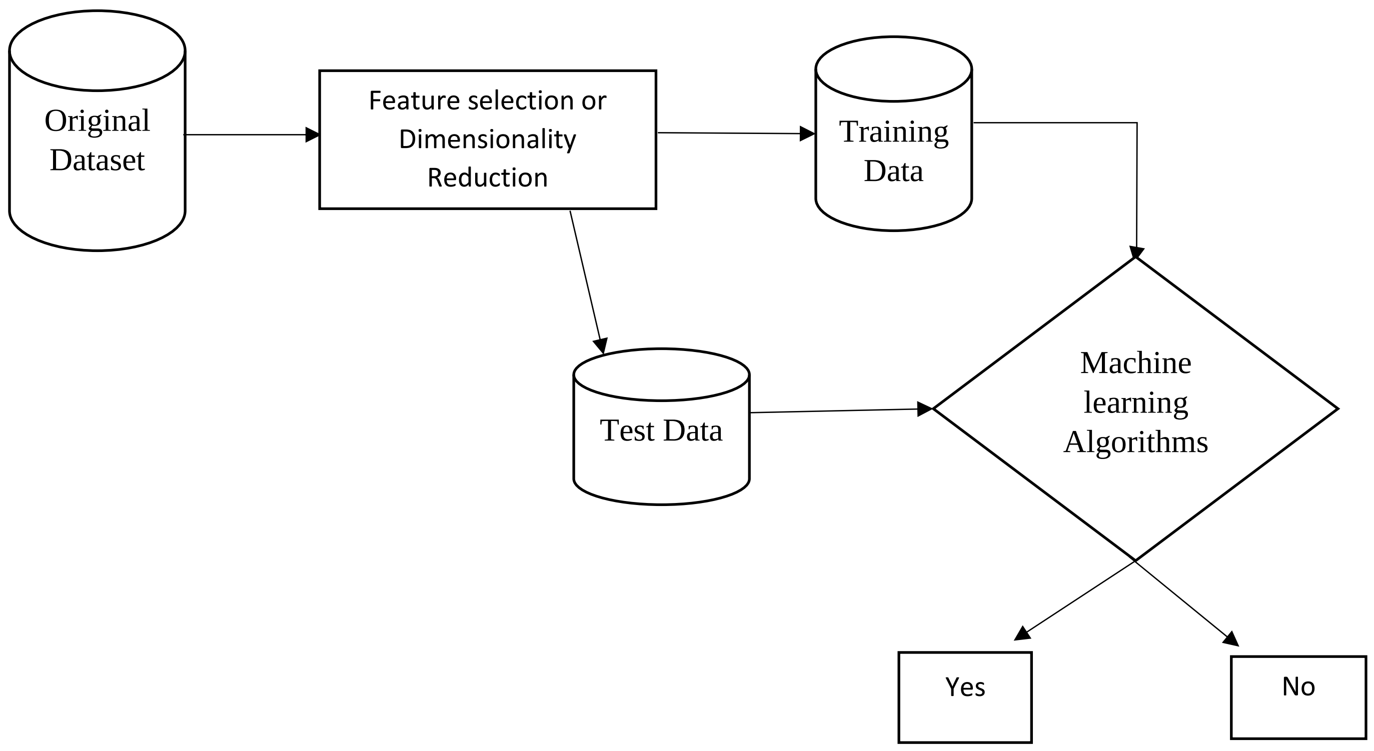
  The Information Technology (Amendment) Act 2008, designated CERT-In to serve as the national agency to perform the following functions in the area of cyber security.

- Collection, angels and dissemination of information on cyber incidents.
- Forecast and alerts of cyber security incidents.
- Emergency measures for handling cyber security incident.
- Coordination of cyber incident response activities
- .Issue Guidelines, advisories, vulnerability notes and white papers relating to information security practices, prevention, response and reporting of cyber incidents.
- Such other functions relating to cyber security.

2. SYSTEM ANALYSIS

2.1 System Flow Charts-:

`Block Diagram Classification Based on Intrusion Detection System (IDS)


Original Dataset
Feature selection or Dimensionality Reduction
Training Data
Test Data
Machine learning Algorithms
Yes
No

2.2 Hardware requirement:

I have implemented the improved Machine Learning algorithm on the computer system Environment, show by Table1

Table1

System Environment	Version
Processor	Intel(R) Celeron(R) CPU N3050 @ 1.60Hz 1.60GHz
Memory RAM	4.00 GB

System Type	64-bit Operating system, x64-based processor

2.3 Software Requirement:

I have used python language and some API for machine learning algorithm in IDS Dataset showed by Table2-

Table2

API for Machine Learning	Version
Scikit-Learn	'0.22.2.post1'
pandas	'1.0.3'
numpy	'1.18.2'
Seaborn	'0.10.0'
Matplotlib	'3.2.1'

3. Overviews of Python concepts

3.1 Python-:

Python is a clear and powerful object-oriented programming language. Python is an interpreted, high level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, python design philosophy emphasize code readability with its notable use of significant whitespace. Its language constructs and object oriented approach aims to help programmers write clear, logical code for small and large scale projects.

Python is easy to learn yet powerful and versatile scripting language, which makes it attractive for application development.

Python's syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development.

Python supports multiple programming pattern, including object oriented imperative and functional or procedural programming styles.

Python is not intended to work in a particular area, such as web programming. That is why it is known as multipurpose programming language because it can be used with web, enterprise 3D CAD, etc.

Python breaks barriers between different computers, chips and operating systems. Python is high level, interpreted, interactive and object oriented scripting language use punctuation, and it has fewer syntactical constructions then other languages.

Punctuation, and it has fewer syntactical constructions then other languages.

3.2 Features of python:

- 1. **Interpreted Language:** Python is an interpreted language i.e. interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.
- 2. **Cross-Platform Language:** Python can run equally on different platforms such as windows, Linux, UNIX, and machines etc. So we can say that python is a portable language.
- 3. **Free and Open Source:** Python language is freely available at official web address. The source- code is also available. Therefore it is open source.
- 4. **Extensible:** It implies that other language such as C/C++ can be used to compile the code and thus it can be used further in our python code.
- 5. **Large Standard Library:** Python has a large and broad library and provides rich set of module and functions for rapid application development.
- 6. **Easy to learn:** Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- 7. **Easy to read:** Python code is more clearly defined and visible to the eyes.
- 8. **Easy to maintain:** Python Source code easy to maintain.
- 9. **Portable:** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- 10. **Database**: python provides interfaces to all major commercial database.
- 11. **GUI Programming Support:** Graphical User Interfaces can be developed using python.

- 12**. Scalable:** Python provides a better structure and support for large programs.

4. Introduction to machine learning

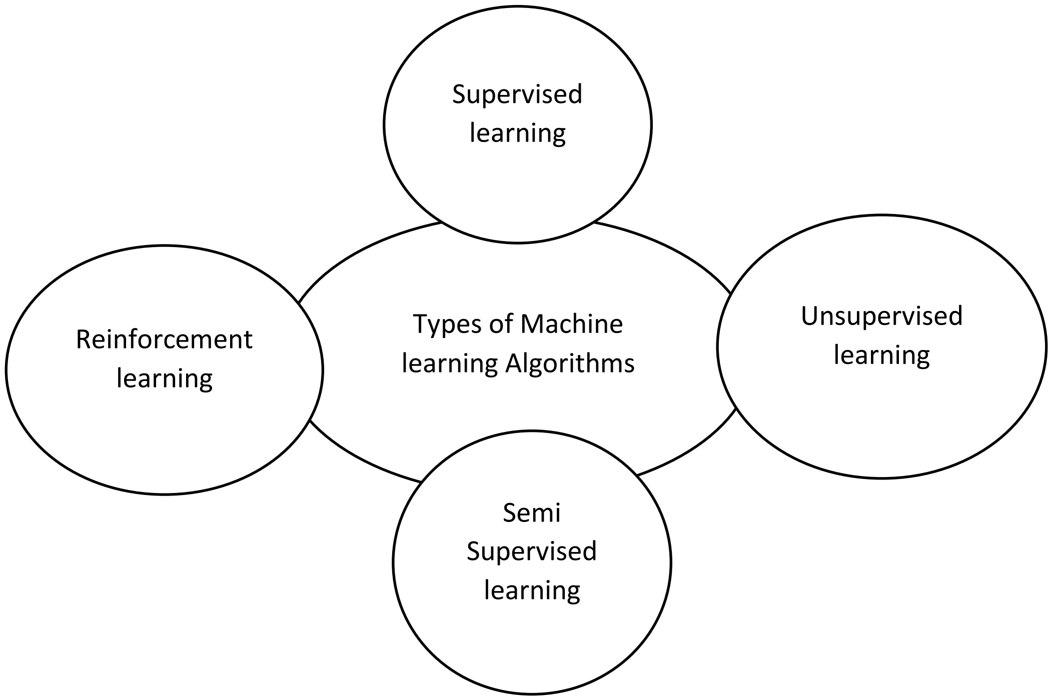
Machine Learning is a subfields of Artificial Intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people.

Although Machine Learning is a fields within computer Science, it differ from traditional computational approaches. In traditional computing, algorithms are sets of explicitly programmed instructions used by computers to calculate or problem solve. Machine Learning algorithms are sets explicitly programmed instructions used by computers to calculate or problem solve. Machine learning algorithms instead allow for computers to train on data inputs and use statistical analysis in order to output values that fall within a specific range. Because of this, Machine learning facilities computers in building models from sample data in order to automate decision-making processes based on data inputs.

Any technology user today has benefitted from machine learning. Facial recognition technology allows social media platforms to help users tag and share photos of friends. Optical character recognition (OCR) technology converts images of text into movable type. Recommendation engines, powered by machine learnings, suggest what movies or television shows to watch next based on user preferences. Self-driving cars that relay on machine learning to navigate may soon available to consumers.

Machine learning is a continuously developing field. Because of this there are some Consideration to keep in minds as you work with machine learning methodologies, or analyze the impact of machine learning processes.

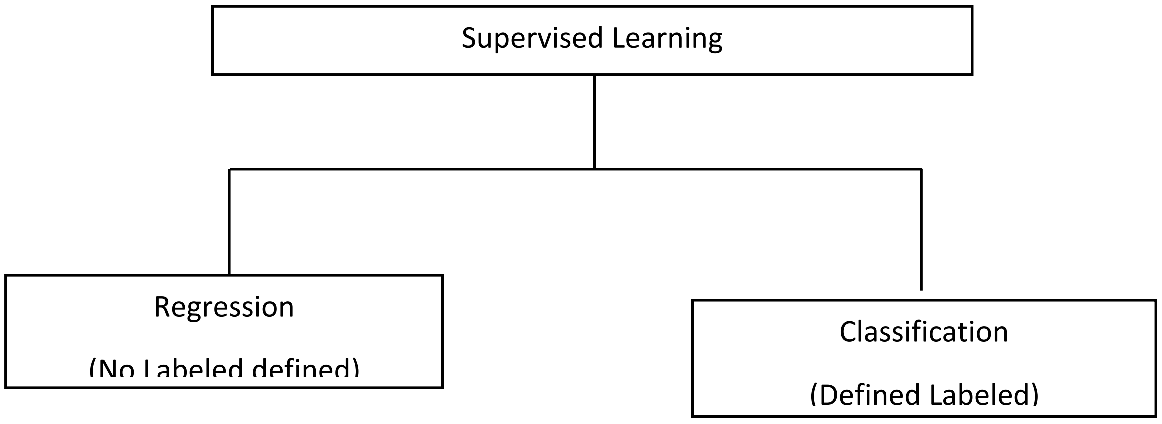
➢ 4.1 Types of Machine Learning-:



Supervised Learning-:

Supervised learning as the name indicates the presence of a supervisor as a teacher. Basically supervised learning is a learning in which we teach or train the machine using data which is well labeled that means some data is already tagged with the correct answer. After that, the machine is provided with a new set of examples (data) so that supervised learning algorithm analysis the training data (set of training examples) and produces a correct outcome from labeled data.

Supervised learning classified into two categories of algorithms:



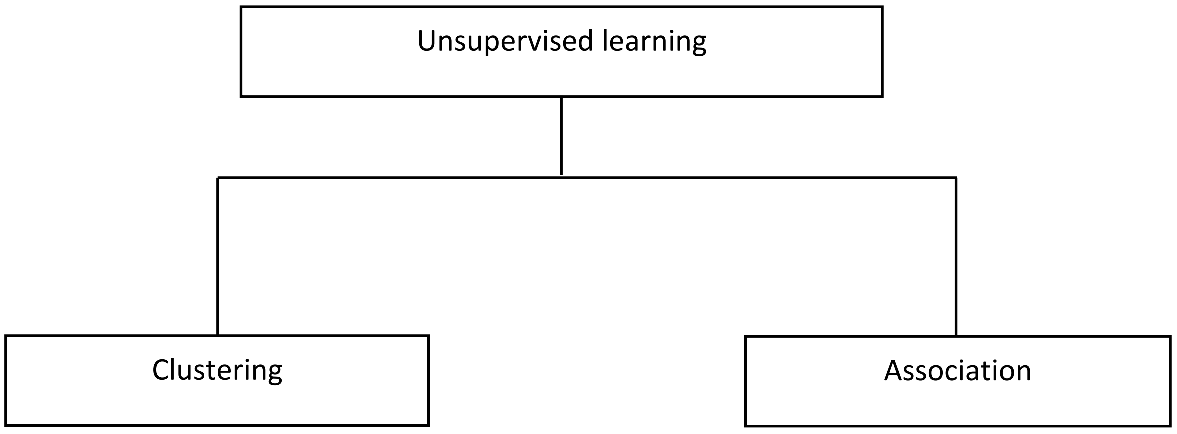
- **Regression-:** A regression problem is when the output variable is a real value. Such as a “Dollars” or “weight”.
- **Classification-:** A classification problem is when the output variable is category. Such as a “red” or “blue” or “disease” or “no disease”.

Unsupervised Learning-:

Unsupervised learning is the training of machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and difference without any prior training of data.

Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore machine is restricted to find the hidden structure in unlabeled data by our-self.

Unsupervised learning classified into two categories of algorithms:



- **Clustering-:** Clustering is a method of grouping the objects into cluster such that objects with most similarities remains into a group and has less or no similarities with the objects of another group. Cluster analysis finds the commonalities between the data objects and categories them as per the presence and absence of those commonalities.

- **Association-:** An association rule is an unsupervised learning method which is used for finding the relationship between variables in the large database. It determines the set of items that occurs together in the data set. Association rule makes marketing strategy more effective. Such as people who buy X item (Suppose a bread) are also tend to purchase Y (Butter/Jam) item. A typical example of Association rule is market Basket Analysis.

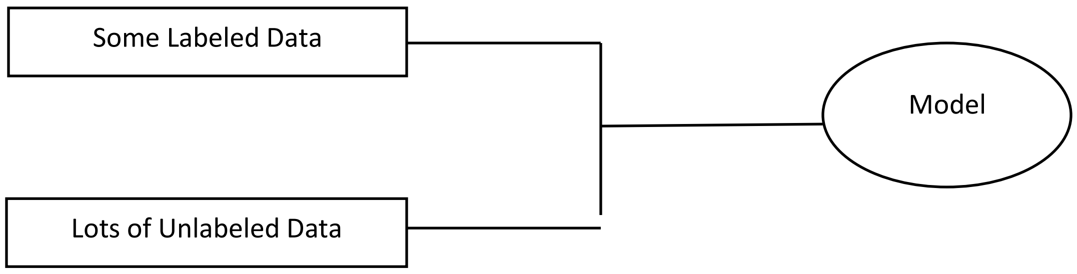
Semi-supervised learning-:

Semi-supervised learning is an approach to machine learning that combines a small amount of labeled data with a large amount of unlabeled data during training. Semi-supervised learning falls between unsupervised learning (with no labeled training data) and supervised learning (with only labeled training data).

Unlabeled data, when used in conjunction with a small amount of labeled data, can produce considerable improvement in learning accuracy. The acquisition of labeled data for a learning problem often requires a skilled human agent (e.g. to transcribe an audio segment) or a physical experiments (e.g. determining the 3D structure of a protein or determine whether there is oil at a particular location). The cost associated with the labeling process thus may render large, fully labeled training sets infeasible, whereas acquisition of unlabeled data is relatively inexpensive. In such

Situations, semi-supervised learning can be of great practical value. Semi-supervised learning is also of theoretical interest in machine learning and as a model for human learning.

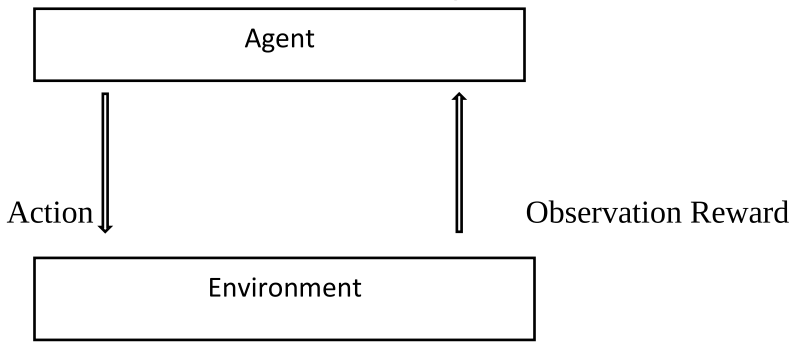
Semi supervised learning-



Reinforcement learning-:

Reinforcement learning is an area of machine learning. It is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behavior or path it should take in a specific situation. Reinforcement learning differs from the supervised learning in a way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of training dataset, it is bound to learn from its experience.

Reinforcement Learning



5. Data Preprocessing in Machine learning:

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

5.1 Why do we need Data Preprocessing:

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

5.2 It involves below steps:

- 1. Getting the dataset
- 2. Importing libraries
- 3. Importing datasets
- 4. Finding Missing Data
- 5. Encoding Categorical Data
- 6. Splitting dataset into training and test set
- 7. Feature scaling

Getting the Data set:

To create a machine learning model, the first thing we required is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a proper format is known as the dataset**.**

Dataset may be of different formats for different purposes, such as, if we want to create a machine learning model for business purpose, then dataset will be different with the dataset required for a liver patient. So each dataset is different from another dataset. To use the dataset in our code, we usually put it into a CSV file**.** However, sometimes, we may also need to use an HTML or xlsx file.

What is CSV file: CSV stands for "Comma-Separated Values" files; it is a file format which allows us to save the tabular data, such as spreadsheets. It is useful for huge datasets and can use these datasets in programs.

Importing libraries:

In order to perform data preprocessing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs. There are specific libraries that we will use for data preprocessing, which are:

```
Import pandas as pd
Import numpy as np
Import sklearn
Import seaborn as sns
Import matplotlib.pyplot as plt
```

Numpy: Numpy Python library is used for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in Python. It also supports to add large, multidimensional arrays and matrices. So, in Python, we can import it as:

```
Import numpy as np
```

Matplotlib: The second library is matplotlib**,** which is a Python 2D plotting library, and with this library, we need to import a sub-library pyplot**.** This library is used to plot any type of charts in Python for the code. It will be imported as below:

```
Import matplotlib.pyplot as plt
```

Pandas: The last library is the Pandas library, which is one of the most famous Python libraries and used for importing and managing the datasets. It is an open-source data manipulation and analysis library. It will be imported as below:

```
Import pandas as pd
```

Seaborn: statistical data visualization. Seaborn is a Python visualization library based on matplotlib. It provides a high-level interface for drawing attractive statistical graphics.

```
Import seaborn as sns
```

Scikit-learn: Scikit-learn is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.

Please note that sklearn is used to build machine learning models. It should not be used for reading the data, manipulating and summarizing it. There are better libraries for that (e.g. NumPy, Pandas etc.).

Import sklearn

Classification: Identifying which category an object belongs to.

Regression: Predicting a continuous-valued attribute associated with an object.

Clustering: Automatic grouping of similar objects into sets.

Dimensionality reduction: Reducing the number of random variables to consider.

Importing datasets: Now we need to import the datasets which we have collected for our machine learning project. But before importing a dataset, we need to set the current directory as a working directory. To set a working directory in Colab IDE, we need to follow the below steps:

- 1. Save your Python file in the directory which contains dataset.
- 2. Go to File explorer option in Colab IDE, and select the required directory.
- 3. Click on Control, Enter button or run option to execute the file.

Finding Missing Data:

The next step of data preprocessing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

Ways to handle missing data:

There are mainly two ways to handle missing data, which are:

By deleting the particular row: The first way is used to commonly deal with null values. In this way, we just delete the specific row or column which consists of null values. But this way is not so efficient and removing data may lead to loss of information which will not give the accurate output.

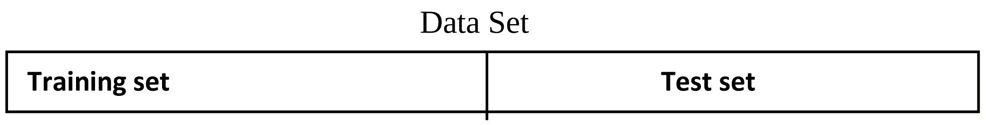
By calculating the mean: In this way, we will calculate the mean of that column or row which contains any missing value and will put it on the place of missing value. This strategy is useful for the features which have numeric data such as age, salary, year, etc. Here, we will use this approach.

Encoding Categorical Data: Categorical data is data which has some categories such as, in our dataset.

Splitting dataset into training and test set: In machine learning data preprocessing, we divide our dataset into a training set and test set. This is one of the crucial steps of data preprocessing as by doing this, we can enhance the performance of our machine learning model.

Suppose, if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:



Training Set: A subset of dataset to train the machine learning model, and we already know the output.

Test set: A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

For splitting the dataset, we will use the below lines of code:

```
from sklearn.model_selection import train_test_split
```

training_x, test_x, training_y, test_y = train_test_split(X_over, y_over, test_size=0.3, random_st
ate=0)

Explanation:

In the above code, the first line is used for splitting arrays of the dataset into random train and test subsets.

In the second line, we have used four variables for our output that are

- **training_x:** features for the training data.
- **test_x:** features for testing data
- **training_y:** Dependent variables for training data
- **test_y:** Independent variable for testing data

In train_test_split() function**,** we have passed four parameters in which first two are for arrays of data, and **test_size** is for specifying the size of the test set. The test_size maybe , .3, or .2, which tells the dividing ratio of training and testing sets.

The last parameter random_state is used to set a seed for a random generator so that you always get the same result, and the most used value for this is 42.

6. Feature scaling:

Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. It is performed during the data pre-processing to handle highly varying magnitudes or values or units. If feature scaling is not done, then a machine learning algorithm tends to weigh greater values, higher and consider smaller values as the lower values, regardless of the unit of the values.

6.1 Techniques to perform Feature Scaling:

Consider the two most important ones:

- **Min-Max Normalization:** This technique re-scales a feature or observation value with distribution value between 0 and 1.

$$X_{new} = \frac{X_i - \min(X)}{\max(x) - \min(X)}$$

- **Standardization:** It is a very effective technique which re-scales a feature value so that it has distribution with 0 mean value and variance equals to 1.

$$X_{new} = \frac{X_i - X_{mean}}{\text{Standard Deviation}}$$

7. Feature selection-:

One of the best way I use to learn machine learning is by benchmark myself against the best data scientists in competitions. It gives you lot of insight into how you perform against the best on a label playing field.

Initially, I used to believe that machine learning is going to be all about algorithms – know which one to apply when and you will come on the top. When I got there, I realized that was not the case – winners were using the same algorithms which a lot of other people were using.

Next, I thought surely these people would have better / superior machines. I discovered that is not the case. I saw competitions being won using a MacBook Air, which is not the best computational machine. Over time, I realized that there are 2 things which distinguish winners from others in most of the cases: Feature Creation and Feature Selection.

In other words, it boils down to creating variables which capture hidden business insights and then making the right choices about which variable to choose for your predictive models, Sadly or thankful, both these skills require a ton of practice. There is also some art involved in creating new features – some people have a knack of finding trends where other people struggle.

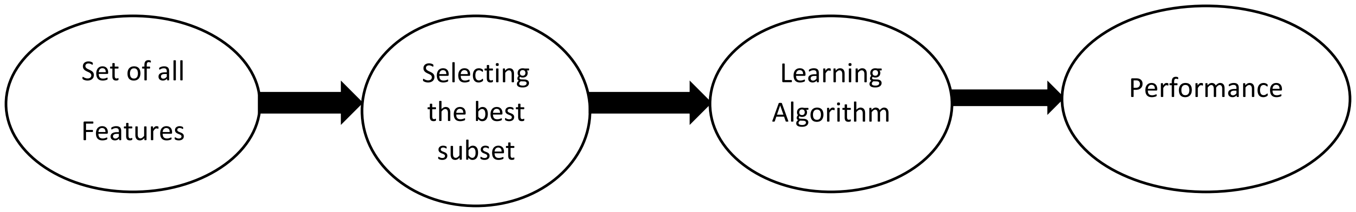
7.1Importance of feature selection:

Top reasons to use feature selection are:

- It enables the machine learning algorithm train faster.
- It reduces the complexity of a model and makes it easier to interpret.
- It improves the accuracy of a model if the right subset is chosen.
- It reducing overfitting.

7.2 Types of Feature Selection:

- **2. Filter Methods:**



Filter methods basically are generally as a preprocessing step. The selection of features is independent of any machine learning algorithms. Instead, feature are selected on the basis of their scores in various statistical tests for their correlation with the outcome variable. The correlation is a subjective term here. For basic guidance, you can refer to the following table for defining correlation co-efficient

Feature\Response	Continuous	Categorical
Continuous	Pearson's Correlation	LDA
Categorical	Anova	Chi-Square

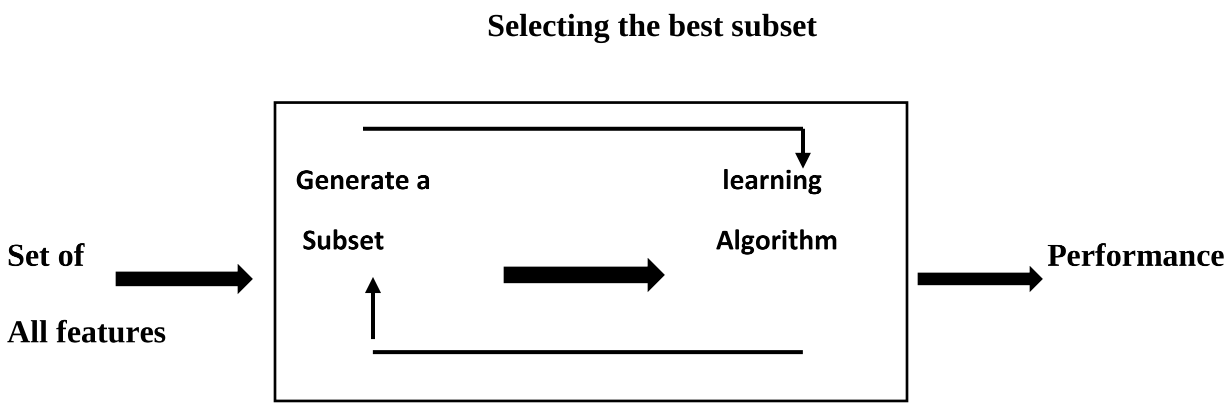
.

- **Pearson's Correlation:** It is used as a measure for quantifying linear dependence between two continuous variables X and Y. its value varies from -1 to +1. Pearson's correlation is given as:

$$\text{Chi}(x, y) = \frac{\text{cov}(x,y)}{\text{Sigmax, sigmay}}$$

- **LDA(Linear Discriminant Analysis):** Linear discernment analysis is used to find a linear combination of feature that characterizes or separates two or more classes (or levels) of a categorical variables.
- **ANOVA:** ANOVA Stands for Analysis of variance. It is similar to LDA except for the fact that it is operated using one or more categorical independent features and one continuous dependent features. It provides a statistical test of whether the means of several groups are equal or not.

➢ 3. Wrapper methods:

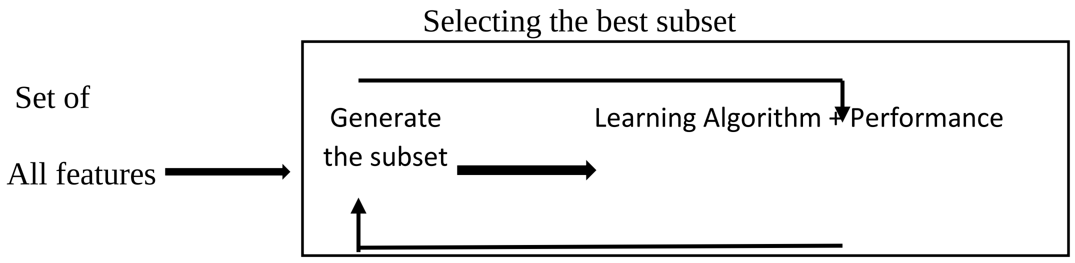


In wrapper methods, we try to use a subset of feature and train a model using them. Based on the inferences that we draw from the previous model, we decide to add or remove features from your subset. The problem is essentially reduced to a search problem. These methods are usually computationally very expensive.

Some common examples of wrapper methods are forward feature selection, backward feature elimination, recursive feature elimination etc.

- ➢ **Forward Selection:** Forward selection is an iterative method in which we start with having no features in the model. In each iteration, we keep adding the features which best improves our model till an addition of a new variable does not improve the performance of the model.
- ➢ **Backward Elimination:** In backward elimination, we start with all the features and removes the least significant feature at each iteration which improves the performance of the model. We repeat this until no improvement is observed on removal of features.
- ➢ **Recursive Feature elimination:** It is a greedy optimization algorithm which aims to find the best performance feature subset. It repeatedly creates models and keeps aside the best or the worst performing feature at each iteration. It constructs the next model with the left features until all the features are exhausted. It then ranks the features based on the order of their elimination.

- **4. Embedded methods:**



Embedded methods combine the qualities of filter and wrapper methods. It implemented by algorithms that have their own built-in feature selection methods.

Some of the most popular examples of these methods are LASSO and RIDGE regression which have inbuilt penalization functions to reduce overfitting.

- Lasso regression perform L1 regularization which adds penalty equivalent to absolute value of the magnitude of coefficients.
- Ridge regression perform L2 regularization which adds penalty equivalent to square of the magnitude of coefficients.

7.3 Dimensionality Reduction-:

Feature selection is different from dimensionality reduction. Both methods seek to reduce the number of attributes in the dataset, but a dimensionality reduction method do so by creating new combinations of attributes, whereas feature selection methods include and exclude attributes present in the data without changing them. Examples of dimensionality reduction methods include Principal Component Analysis, Singular Value Decomposition and Sammon’s mapping.

- **Principle Component Analysis (PCA):** Principal Component Analysis (or PCA) uses linear algebra to transform the dataset into a compressed form. Principal component analysis (PCA) is a statistical procedure that uses an orthogonal_transformation to convert a set of observations of possibly correlated variables (entities each of which takes on various numerical values) into a set of values of linearly_uncorrelated variables called principal components.

Generally, this is called a data reduction technique. A property of PCA is that you can choose the number of dimensions or principal component in the transformed result.

7.4 Chi-square Test for feature selection

Feature selection is also known as attribute selection is a process of extracting the most relevant features from the dataset and then applying machine learning algorithms for the better performance of the model. A large number of irrelevant features increases the training time exponentially and increase the risk of overfitting.

7.5 Chi-square Test for Feature Extraction:

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 score is given by:

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

Where –

Observed frequency = No. of observations of class

Expected frequency = No. of expected observations of class if there was no relationship between the feature and the target.

Python Implementation of Chi-Square feature selection:

```
from sklearn.feature_selection import SelectKBest

from sklearn.feature_selection import chi2
X_update=SelectKBest(score_func=chi2,k=60).fit_transform(X_new,real_y)
```

8. Introduction to Resampling methods

8.1 Imbalanced Data Distribution: While reading about Machine Learning and Data Science we often come across a term called Imbalanced Class Distribution**,** generally happens when observations in one of the classes are much higher or lower than any other classes. As Machine Learning algorithms tend to increase accuracy by reducing the error, they do not consider the class distribution. This problem is prevalent in examples such as Fraud Detection, Anomaly Detection, Facial recognition etc.

Two common methods of Resampling are –

- Cross Validation
- Bootstrapping

8.2 Cross Validation:

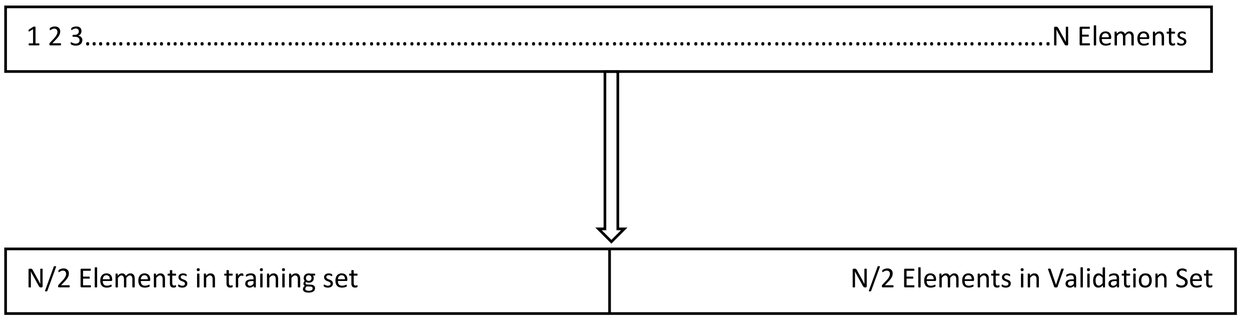
Cross-Validation is used to estimate the test error associated with a model to evaluate its performance.

8.3 Bootstrapping:

Bootstrap is a powerful statistical tool used to quantify the uncertainty of a given model. However, the real power of bootstrap is that it could get applied to a wide range of models where the variability is hard to obtain or not output automatically.

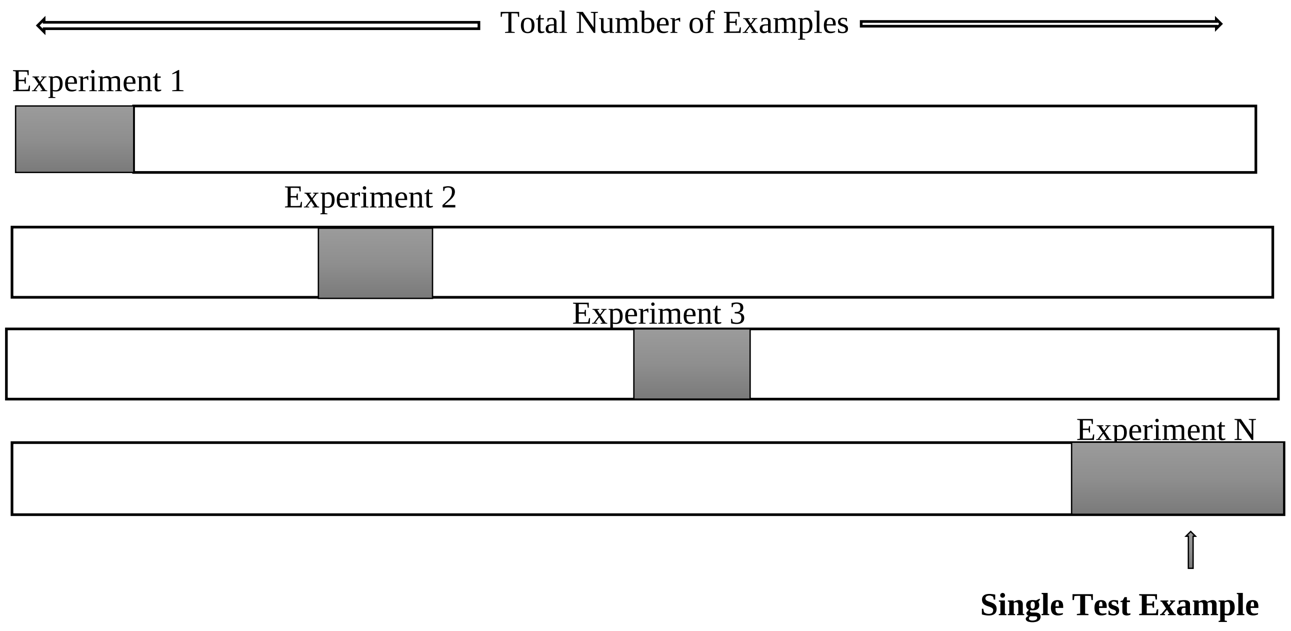
8.3 Validation set approach:

This is the most basic approach. It simply involves randomly dividing the dataset into two parts: first a training set and second a validation set or hold-out set. The model is fit on the training set and the fitted model is used to make predictions on the validation set.


1 2 3……..N Elements
N/2 Elements in training set
N/2 Elements in Validation Set

8.4 Leave-one-out-cross-validation:

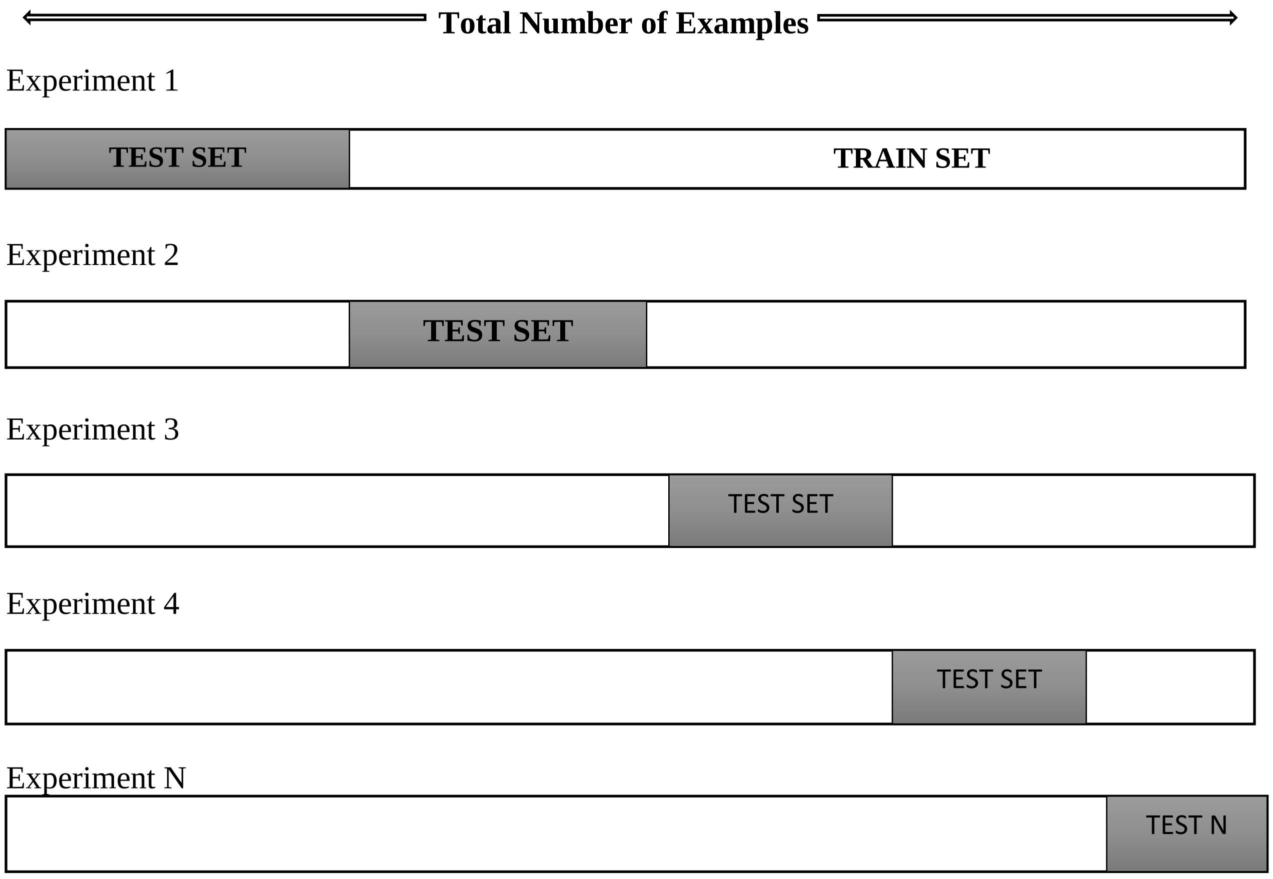
LOOCV is a better option than the validation set approach. Instead of splitting the entire dataset into two halves only one observation is used for validation and the rest is used to fit the model.



8.5 K-fold cross-validation:

This approach involves randomly dividing the set of observations into k folds of nearly equal size. The first fold is treated as a validation set and the model is fit on the remaining folds. The procedure is then repeated k times, where a different group each time is treated as the validation set.

N Fold Cross Validation Approach



9. Handling Approach:

9.1 Random Over-sampling:

It aims to balance class distribution by randomly increasing minority class examples by replicating them.

9.2 Random Under-Sampling:

It aims to balance class distribution by randomly eliminating majority class.

9.3 SMOTE (Synthetic Minority Oversampling Technique):

SMOTE (Synthetic Minority Oversampling Technique) synthesises new minority instances between existing minority instances. It randomly picks up the minority class and calculates the K-nearest neighbour for that particular point. Finally, the synthetic points are added between the neighbours and the chosen spot.

Balanced the Data set Using Random Sampling with (SMOTE) in Python Implementation (Code)

```
from imblearn.over_sampling import RandomOverSampler
oversample = RandomOverSampler()
X_over, y_over = oversample.fit_sample(X_update, real_y)
```

10. Hyper parameter tuning

A Machine Learning model is defined as a mathematical model with a number of parameters that need to be learned from the data. By training a model with existing data, we are able to fit the model parameters.
However, there is another kind of parameters, known as hyper parameters that cannot be directly learned from the regular training process. They are usually fixed before the actual training process begins. These parameters express important properties of the model such as its complexity or how fast it should learn.

Some examples of model hyper parameters include:

- The penalty in Logistic Regression Classifier i.e. L1 or L2 regularization
- The learning rate for training a neural network.
- The C and sigma hyper parameters for support vector machines.
- The k in k-nearest neighbors.

The aim of this article is to explore various strategies to tune hyper parameter for Machine learning model.

Models can have many hyper parameters and finding the best combination of parameters can be treated as a search problem. Two best strategies for

Hyper parameter tuning are:

- 1. **GridSearchCV**
- 2. **RandomizedSearchCV**

GridSearchCV:

In GridSearchCV approach, machine learning model is evaluated for a range of hyper parameter values. This approach is called GridSearchCV, because it searches for best set of hyper parameters from a grid of hyper parameters values.

RandomizedSearchCV:

RandomizedSearchCV solves the drawbacks of GridSearchCV, as it goes through only a fixed number of hyper parameter settings. It moves within the grid in random fashion to find the best set hyper parameters. This approach reduces unnecessary computation.

11. Machine Learning Classifier

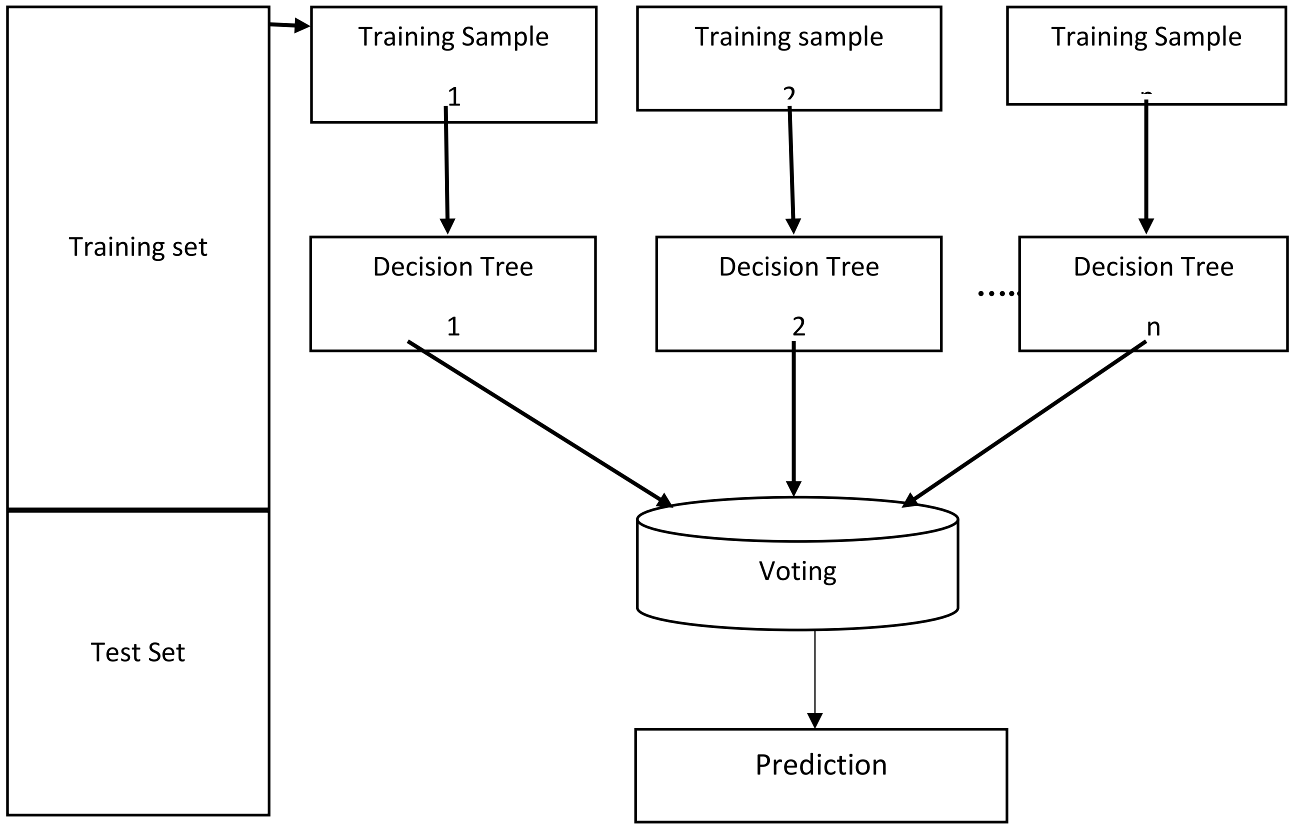
A classification model attempts to draw some conclusion from observed values. Given one or more inputs, a classification model will try to predict the value of one or more outcomes. The Intrusion Detection system, prediction models in this study are built using nine different classification algorithms, including Multinomial Naive Bayes (MNB), Gaussian Naive Bayes (GNB), K Nearest Neighbors (KNN), AdaBoost, Random forest, XGBoost, Support Vector Machine (SVM), Decision tree and Logistic Regression (LR).

Random Forest Classifier-:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests**,** "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The below diagram explains the working of the Random Forest algorithm:



Decision Tree Classifier-:

Decision Tree is a supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node**.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

The decisions or the test are performed on the basis of features of the given dataset.

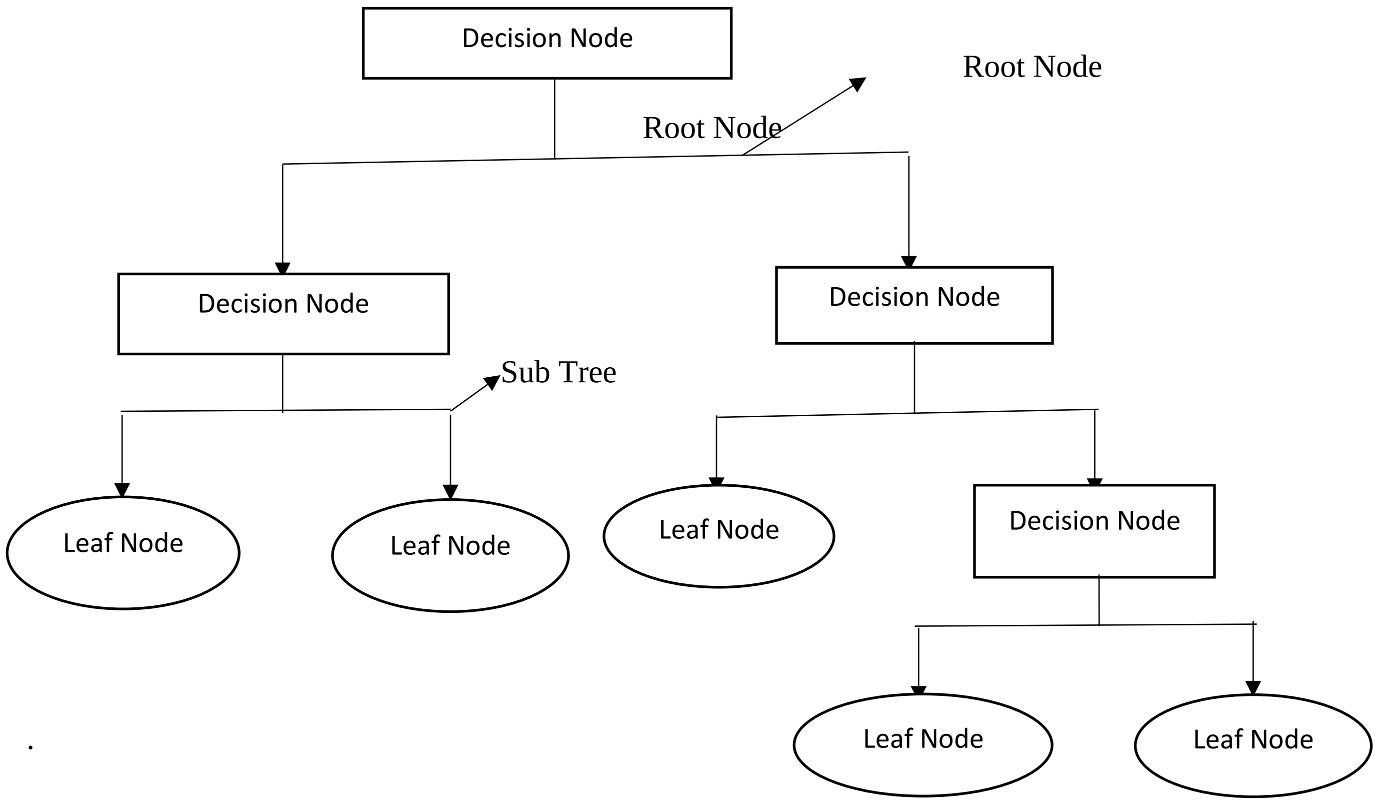
It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.

A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

Below diagram explains the general structure of a decision tree:



K-Nearest Neighbor Classifiers-:

K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique.

K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.

K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.

K-NN is a non-parametric algorithm**,** which means it does not make any assumption on underlying data.

It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data..

Support Vector Classifier (SVM) -:

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

Support Vectors:

Support vectors are the data points, which are closest to the hyperplane. These points will define the separating line better by calculating margins. These points are more relevant to the construction of the classifier.

Hyperplane:

A hyperplane is a decision plane which separates between a set of objects having different class memberships.

Margin:

A margin is a gap between the two lines on the closest class points. This is calculated as the perpendicular distance from the line to support vectors or closest points. If the margin is larger in between the classes, then it is considered a good margin, a smaller margin is a bad margin.

AdaBoost Classifier-:

Ada-boost or Adaptive Boosting is one of ensemble boosting classifier proposed by Yoav Freund and Robert Schapire in 1996. It combines multiple classifiers to increase the accuracy of classifiers. AdaBoost is an iterative ensemble method. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier. The basic concept behind Adaboost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observations. Any machine learning algorithm can be used as base classifier if it accepts weights on the training set. Adaboost should meet two conditions:

- 1. The classifier should be trained interactively on various weighed training examples.
- 2. In each iteration, it tries to provide an excellent fit for these examples by minimizing training error.

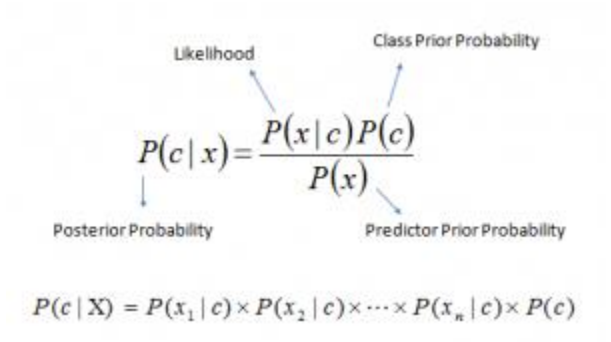
Naïve Bayes Classifier-:

It is a classification_technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naive'.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:



$$P(c \mid x) = \frac{P(x \mid c)P(c)}{P(x)}$$

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Above,

- P(c|x) is the posterior probability of class (c, target) given predictor (x, attributes).
- P(c) is the prior probability of class.
- P(x|c) is the likelihood which is the probability of predictor given class.
- P(x) is the prior probability of predictor.

How to build a basic model using Naïve Bayes in Python:

- 1. **Gaussian:** It is used in classification and it assumes that features follow a normal distribution.

- 2. **Multinomial:** It is used for discrete counts. For example, let’s say, we have a text classification problem. Here we can consider Bernoulli trials which is one step further and instead of “word occurring in the document”, we have “count how often word occurs in the document”, you can think of it as “number of times outcome number x is observed over the n trials”.
- 3. **Bernoulli:** The binomial model is useful if your feature vectors are binary (i.e. zeros and ones). One application would be text classification with ‘bag of words’ model where the 1s & 0s are “word occurs in the document” and “word does not occur in the document” respectively.

Logistic Regression-:

Logistic regression is a statistical method for predicting binary classes. The outcome or target variable is dichotomous in nature. Dichotomous means there are only two possible classes. For example, it can be used for cancer detection problems. It computes the probability of an event occurrence.

It is a special case of linear regression where the target variable is categorical in nature. It uses a log of odds as the dependent variable. Logistic Regression predicts the probability of occurrence of a binary event utilizing a logit function.

Linear Regression Equation:

$$y = \beta 0 + \beta 1X1 + \beta 2X2 + \ldots\ldots + \beta nXn$$

Where, y is dependent variable and x1, x2 ... and Xn are explanatory variables.

Sigmoid Function:

$$p = 1/1 + e^{-y}$$

Apply sigmoid function on linear regression:

$$p = 1/1 + e^{-(\beta 0 + \beta 1X1 + \beta 2X2 \ldots\ldots \beta nXn)}$$

Types of Logistic Regression:

- **Binary Logistic Regression:** The target variable has only two possible outcomes such as Spam or Not Spam, Cancer or No Cancer.
- **Multinomial Logistic Regression:** The target variable has three or more nominal categories such as predicting the type of Wine.
- **Ordinal Logistic Regression:** the target variable has three or more ordinal categories such as restaurant or product rating from 1 to 5.

Stochosting Gradient Boosting Classifier (SGD)-:

Gradient Descent-:

Before explaining Stochastic Gradient Descent (SGD), let's first describe what Gradient Descent is. Gradient Descent is a popular optimization technique in Machine Learning and Deep Learning, and it can be used with most, if not all, of the learning algorithms. A gradient is the slope of a function. It measures the degree of change of a variable in response to the changes of another variable. Mathematically, Gradient Descent is a convex function whose output is the partial derivative of a set of parameters of its inputs. The greater the gradient, the steeper the slope.

Starting from an initial value, Gradient Descent is run iteratively to find the optimal values of the parameters to find the minimum possible value of the given cost function.

Types of Gradient Descent:

Typically, there are three types of Gradient Descent:

- 1. Batch Gradient Descent
- 2. Stochastic Gradient Descent
- 3. Mini-batch Gradient Descent

Batch Gradient Descent: This is a type of gradient descent which processes all the training examples for each iteration of gradient descent. But if the number of training examples is large, then batch gradient descent is computationally very expensive. Hence if the number of training examples is large, then batch gradient descent is not preferred. Instead, we prefer to use stochastic gradient descent or mini-batch gradient descent.

Stochastic Gradient Descent: This is a type of gradient descent which pro-cesses 1 training example per iteration. Hence, the parameters are being updated even after one iteration in which only a single example has been processed. Hence this is quite faster than batch gradient descent. But again, when the number of training examples is large, even then it processes only one example which can be additional overhead for the system as the number of iterations will be quite large.

Mini Batch gradient descent: This is a type of gradient descent which works faster than both batch gradient descent and stochastic gradient descent. Here b examples where $b<m$ are processed per iteration. So even if the number of training examples is large, it is processed in batches of b training examples in one go. Thus, it works for larger training examples and that too with lesser number of iterations.

Stochosting Gradient Boosting Classifier (SGD)-:

The word 'stochastic'means a system or a process that is linked with a random probability. Hence, in Stochastic Gradient Descent, a few samples are selected randomly instead of the whole data set for each iteration. In Gradient Descent, there is a term called "batch" which denotes the total number of samples from a dataset that is used for calculating the gradient for each iteration. In typical Gradient Descent optimization, like Batch Gradient Descent, the batch is taken to be the whole dataset. Although, using the whole dataset is really useful for getting to the minima in a less noisy and less random manner, but the problem arises when our datasets gets big.

Gradient Boosting is an ensemble learner**.** This means it will create a final model based on a collection of individual models. The predictive power of these individual models is weak and prone to overfitting but combining many such weak models in an ensemble will lead to an overall much improved result. In Gradient Boosting machines, the most common type of weak model used is decision trees - another parallel to Random Forests.

XG-Boost Classifier-:

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. In prediction problems involving unstructured data (images, text, etc.)

artificial neural networks tend to outperform all other algorithms or frameworks. However, when it comes to small-to-medium structured/tabular data, decision tree based algorithms are considered best-in-class right now.

How to build an intuition for XGBoost:

Decision trees, in their simplest form, are easy-to-visualize and fairly interpretable algorithms but building intuition for the next-generation of tree-based algorithms can be a bit tricky. See below for a simple analogy to better understand the evolution of tree-based algorithms.

Each step of the evolution of tree-based algorithms can be viewed as a version of the interview process.

- 1. **Decision Tree**: Every hiring manager has a set of criteria such as education level, number of years of experience, interview performance. A decision tree is analogous to a hiring manager interviewing candidates based on his or her own criteria.
- 2. **Bagging**: Now imagine instead of a single interviewer, now there is an interview panel where each interviewer has a vote. Bagging or bootstrap aggregating involves combining inputs from all interviewers for the final decision through a democratic voting process.
- 3. **Random Forest**: It is a bagging-based algorithm with a key difference wherein only a subset of features is selected at random. In other words, every interviewer will only test the interviewee on certain randomly selected qualifications (e.g. a technical interview for testing programming skills and a behavioral interview for evaluating non-technical skills).
- 4. **Boosting**: This is an alternative approach where each interviewer alters the evaluation criteria based on feedback from the previous interviewer. This 'boosts' the efficiency of the interview process by deploying a more dynamic evaluation process.
- 5. **Gradient Boosting**: A special case of boosting where errors are minimized by gradient descent algorithm e.g. the strategy consulting firms leverage by using case interviews to weed out less qualified candidates.
- 6. **XGBoost**: Think of XGBoost as gradient boosting on 'steroids' (well it is called 'Extreme Gradient Boosting' for a reason!). It is a perfect combination of software and

hardware optimization techniques to yield superior results using less computing resources in the shortest amount of time.

12. Data set description

In this data set I have consider Flow ID is not important. Some important attribute in this data sets like, Source IP, Source Port, Destination IP, Destination Port, Protocol, Timestamp, Flow Duration, Label. All attribute I have converted into integer (numeric) form Show in below table.

In this paper the total IDS data set is 3119345 rows and 85 columns. In this data sets data set some attribute like, Flow ID, Source IP, Source Port, Destination IP, Destination Port, Protocol, Timestamp, Flow Duration, Label etc.

The evaluation data sets play a vital role in the validation of any IDS approach, by allowing us to assess the proposed methods capability in detecting intrusive behavior. The data sets used for network packets analysis in commercial products are not easily available due to privacy issues.

Attribute or features of Intrusion Detection System (IDS)		
Flow ID	**Source IP**	**Source Port**
Destination IP	**Destination Port**	**Timestamp**
Flow Duration	**Total Fwd Packets**	**Total Backward Packets**
Total Length of Fwd Packets	**Total Length of Bwd Packets**	**Fwd Packet Length Max**
Fwd Packet Length Min	**Fwd Packet Length Mean**	**Fwd Packet Length Std**
Bwd Packet Length Max	**Bwd Packet Length Min**	**Bwd Packet Length Mean**
Bwd Packet Length Std	**Flow Bytes/s**	**Flow Packets/s**
Flow IAT Mean	**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	**Fwd URG Flags**	**Bwd URG Flags**
Fwd Header Length	**Bwd Header Length**	**Fwd Packets/s**
Bwd Packets/s	**Min Packet Length**	**Max Packet Length**
Packet Length Mean	**Packet Length Std**	**Packet Length Variance**
FIN Flag Count	**SYN Flag Count**	**RST Flag Count**
PSH Flag Count	**ACK Flag Count**	**URG Flag Count**
CWE Flag Count	**ECE Flag Count**	**Down/Up Ratio**
Average Packet Size	**Avg Fwd Segment Size**	**Avg Bwd Segment Size**
Fwd Header Length.1	**Fwd Avg Bytes/Bulk**	**Fwd Avg Packets/Bulk**
Fwd Avg Bulk Rate	**Bwd Avg Bytes/Bulk**	**Subflow Fwd Packets**
Bwd Avg Packets/Bulk	**Bwd Avg Bulk Rate**	**Subflow Fwd Bytes**
Subflow Bwd Packets	**Subflow Bwd Bytes**	**Init_Win_bytes_forward**
Init_Win_bytes_backward	**act_data_pkt_fwd**	**min_seg_size_forward**
Active Mean	**Active Std**	**Active Max**
Active Min	**Idle Mean**	**Idle Std**
Idle Max	**Idle Min**	**Label**

➢ Flow ID:

Flowid is leading in continuous process solutions.

From process development services to convert batch processes in to continuous processes, to the engineering and construction of skids based on flow technology. These continuous solutions are offered on a laboratory-, pilot- and production scale.

With Flowid’s proprietary SpinPro technology, a unique continuous flow reactor is provided to control fast exothermic multiphase reactions, and performing precipitation and emulsification applications.

- **Source IP**: Source IP address - the IP packet field containing the IP address of the workstation from which it came.
- **Source port:** A source port is a software project based on the source code of a game engine that allows the game to be played on operating systems or computing_platforms with which the game was not originally compatible.
- **Destination IP:** Destination IP address - the IP packet field containing the IP address of the workstation to which it is addressed.
- **Destination Ports:** Destination Ports Are Server Applications.

  Destination ports may be "well-known ports" (0-1023) for the major Internet applications, such as Web and email. For example, all port 80 packets (HTTP packets) are directed to and processed by a Web server.
- **Protocol:** It is a digital language through which we communicate with others on the Internet. Protocol meaning is that it a set of mutually accepted and implemented rules at both ends of the communications channel for the proper exchange of information. By adopting these rules, two devices can communicate with each other and can interchange information. We can't even think of using the Internet without Protocols. Each protocol is defined in different terms and different use with unique name. Message travel from sender to reciever via a medium (The medium is the physical path over which a message travels) using a protocol.
- **Timestamp:** A timestamp is a sequence of characters or encoded information identifying when a certain event occurred, usually giving date and time of day, sometimes accurate to a small fraction of a second.

  The most commonly used concurrency protocol is the timestamp based protocol Lock-based protocols manage the order between the conflicting pairs among transactions at the time of execution, whereas timestamp-based protocols start working as soon as a transaction is created.
- **The Label Column is my target attribute.**

In this data set target attribute is 15 types of attacks BENIGN, DDoS, Port Scan, Botnet, Infiltration, Web Attack \x96 Brute Force, Web Attacks \x96 XSS, Web Attack \x96 Sql Injection, FTP-Patator, SSH-Patator, DoS slowloris, DoS Slowhttptest, DoS Hulk, DoS GoldenEye, Heartbleed etc.

Show in table my target attribute-

BENIGN	DDoS	Port Scan
Botnet	Infiltration	Web Attack \x96 Brute Force
Web Attacks \x96 XSS	Web Attack \x96 Sql Injection	FTP-Patator
SSH-Patator	DoS slowloris	DoS Slowhttptest
DoS Hulk	DoS GoldenEye	Heartbleed

In this paper the total IDS data set is 3119345 rows and 85 columns.

I am working from 15000 data sets and 84 columns. In this data set I have applied feature selection, hyper parameter tuning and after the training data set is number of rows 10500 and columns 83 and testing data set is number of rows 4500 and columns 84

12.1 Overview of IDS attacks

1) BENIGN:

Benign is a normal attacks for Intrusion Detection System.

2) DDoS attacks:

A distributed denial-of-service (DDoS) attack is a malicious attempt to disrupt normal traffic of a targeted server, service or network by overwhelming the target or its surrounding infrastructure with a flood of Internet traffic. DDoS attacks achieve effectiveness by utilizing multiple compromised computer systems as sources of attack traffic. Exploited machines can include computers and other networked resources such as IoT devices. From a high level, a DDoS attack is like a traffic jam clogging up with highway, preventing regular traffic from arriving at its desired destination.

3) Port Scan:

To launch a port scan attack, hackers take advantage of a tool like Nmap to sort through the available hosts on your network. A port scan will return one of three potential classifications for identified ports:

Open: Target host is listening on the port and the service used in the scan is being used.

Closed: Packet requests are received but the service isn't listening on the port.

Filtered: Packet request is sent but there's no reply, indicating a firewall has filtered the request packet.

Mapping ports in this way gives attackers insight into the weak points of your network. Every open port indicates the potential for a vulnerable system that attackers can exploit to gain a foothold in your network or launch denial of service campaigns.

4) Botnet:

A botnet refers to a group of computers which have been infected by malware and have come under the control of a malicious actor. The term botnet is a portmanteau from the words robot and network and each infected device is called a bot. Botnets can be designed to accomplish illegal or malicious tasks including sending spam, stealing data, ransomware, fraudulently clicking on ads or distributed denial-of-service (DDoS) attacks.

While some malware, such as ransomware, will have a direct impact on the owner of the device, DDoS botnet malware can have different levels of visibility; some malware is designed to take total control of a device, while other malware runs silently as a background process while waiting silently for instructions from the attacker or "bot herder."

Self-propagating botnets recruit additional bots through a variety of different channels. Pathways for infection include the exploitation of website vulnerabilities, Trojan horse malware, and cracking weak authentication to gain remote access. Once access has been obtained, all of these methods for infection result in the installation of malware on the target device, allowing remote control by the operator of the botnet. Once a device is infected, it may attempt to self-propagate the botnet malware by recruiting other hardware devices in the surrounding network.

While it's infeasible to pinpoint the exact numbers of bots in a particular botnet, estimations for total number of bots in a sophisticated botnet have ranged in size from a few thousand to greater than a million.

5) **Infiltration:**

Privilege escalation normally occurs deep into an attack. This means that the attacker will have already done reconnaissance and successfully compromised a system, thereby gaining entry. After this, the attacker will have traversed the compromised system through lateral movement and identified all the systems and devices of interest. In this phase, the attacker wants to have a strong grip on the system. The attacker may have compromised a low-level account and will, therefore, be looking for an account with higher privileges, in order to study the system further or get ready to give the final blow. Privilege escalation is not a simple phase, as it will at times require the attacker to use a combination of skills.

6) **Web Attack \x96 Brute Force:**

A brute force attack is an attempt to crack a password or username or find a hidden web page, or find the key used to encrypt a message, using a trial and error approach and hoping, eventually, to guess correctly. This is an old attack method, but it's still effective and popular with hackers.

Depending on the length and complexity of the password, cracking it can take anywhere from a few seconds to many years. In fact, IBM reports that some hackers target the same systems every day for months and sometimes even years.

7) **Web Attacks \x96 XSS:**

Cross-site scripting attacks, often abbreviated as XSS, are a type of attack in which malicious scripts are injected into websites and web applications and run on an end user’s platform. XSS attacks are a common and widespread type of attack, using unsanitized or unvalidated user inputs, aimed at the generated output.

The XSS attack does not have to choose a specific target; the attacker simply exploits the vulnerability of the application or site, taking advantage of anyone unlucky enough to trigger an attack. Using XSS attacks, a web application or web site becomes the vector of delivering malicious scripts to the browsers of several victims.

XSS attacks can exploit vulnerabilities in several software environments, including VBScript, Flash, ActiveX, and JavaScript; XSS attacks most often use JavaScript due to the integrated nature of JavaScript in most browsers. This ability to exploit commonly used platforms makes XSS attacks one of the most common security vulnerabilities.

8) **Web Attack \x96 Sql Injection:**

A SQL injection attack consists of insertion or "injection" of a SQL query via the input data from the client to the application. A successful SQL injection exploit can read sensitive data from the database, modify database data (Insert/Update/Delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system and in some cases issue commands to the operating system. SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to effect the execution of predefined SQL commands. – OWASP.

9) FTP-Patator:

Patator is a multi-purpose brute-forcer, with a modular design and a flexible usage.

Patator was written out of frustration from using Hydra, Medusa, Ncrack, Metasploit modules and Nmap NSE scripts for password guessing attacks. I opted for a different approach in order to not create yet another brute-forcing tool and avoid repeating the same shortcomings. Patator is a multi-threaded tool written in Python that strives to be more reliable and flexible than his fellow predecessors.

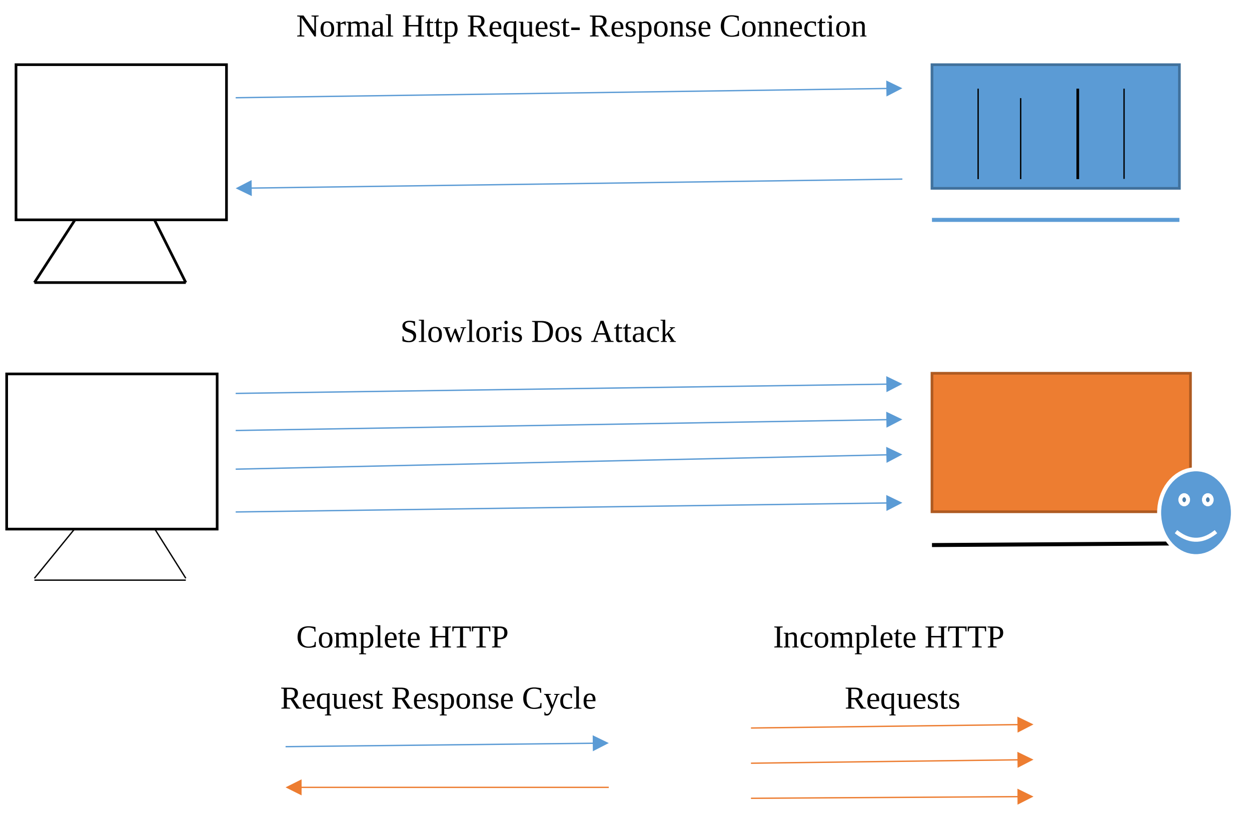
10) SSH-Patator:

SSH, or Secure Shell, is similar to SSL in that they're both PKI based and both form encrypted communication tunnels. But whereas SSL is designed for the transmission of information, SSH is designed to execute commands. You generally see SSH when you want to log in to some part of a network remotely.

SSH uses port 22 and also requires client authentication. After all, the ability to run commands requires a certain level of permission, so, obviously, you need to confirm the identity of the individual trying to log in.

11) DoS slowloris:

Slowloris is a denial-of-service attack program which allows an attacker to overwhelm a targeted server by opening and maintaining many simultaneous HTTP connections between the attacker and the target.



12) DoS Slowhttptest:

SlowHTTPTest is a highly configurable tool that simulates some Application Layer Denial of Service attacks. It works on majority of Linux platforms, OSX and Cygwin – a Unix-like environment and command-line interface for Microsoft Windows.

It implements most common low-bandwidth Application Layer DoS attacks, such as slowloris, Slow HTTP POST, Slow Read attack (based on TCP persist timer exploit) by draining concurrent connections pool, as well as Apache Range Header attack by causing very significant memory and CPU usage on the server.

Slowloris and Slow HTTP POST DoS attacks rely on the fact that the HTTP protocol, by design, requires requests to be completely received by the server before they are processed. If an HTTP request is not complete, or if the transfer rate is very low, the server keeps its resources busy waiting for the rest of the data. If the server keeps too many resources busy, this creates a denial of service. This tool is sending partial HTTP requests, trying to get denial of service from target HTTP server.

13) **DoS Hulk:**

HULK is a Denial of Service (DoS) tool used to attack web servers by generating volumes of unique and obfuscated traffic.

HULK's generated traffic also bypasses caching engines and hits the server's direct resource pool.

14) **DoS GoldenEye:**

GoldenEye is a HTTP DoS Test Tool. This tool can be used to test if a site is susceptible to Deny of Service (DoS) attacks. Is possible to open several parallel connections against a URL to check if the web server can be compromised.
The program tests the security in networks and uses 'HTTP Keep Alive + No Cache' as attack vector.

15) **Heartbleed:**

The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. This weakness allows stealing the information protected, under normal conditions, by the SSL/TLS encryption used to secure the Internet. SSL/TLS provides communication security and privacy over the Internet for applications such as web, email, instant messaging (IM) and some virtual private networks (VPNs).
The Heartbleed bug allows anyone on the Internet to read the memory of the systems protected by the vulnerable versions of the OpenSSL software. This compromises the secret keys used to identify the service providers and to encrypt the traffic, the names and passwords of the users and

the actual content. This allows attackers to eavesdrop on communications, steal data directly from the services and users and to impersonate services and users.

The IDS dataset can be divided into two class classification probelom-

12.2 Binary Class Classification : Binary or binomial classification is the task of classifying the elements of a given set into two groups (predicting which group each one belongs to) on the basis of a classification rule.

Contexts requiring a decision as to whether or not an item has some qualitative property, some specified characteristic, or some typical binary classification include:

IDS testing to determine if a attacks has certain attack or not – the classification property is the presence of the IDS.

12.3 Binary Classification with Implementation of Python:

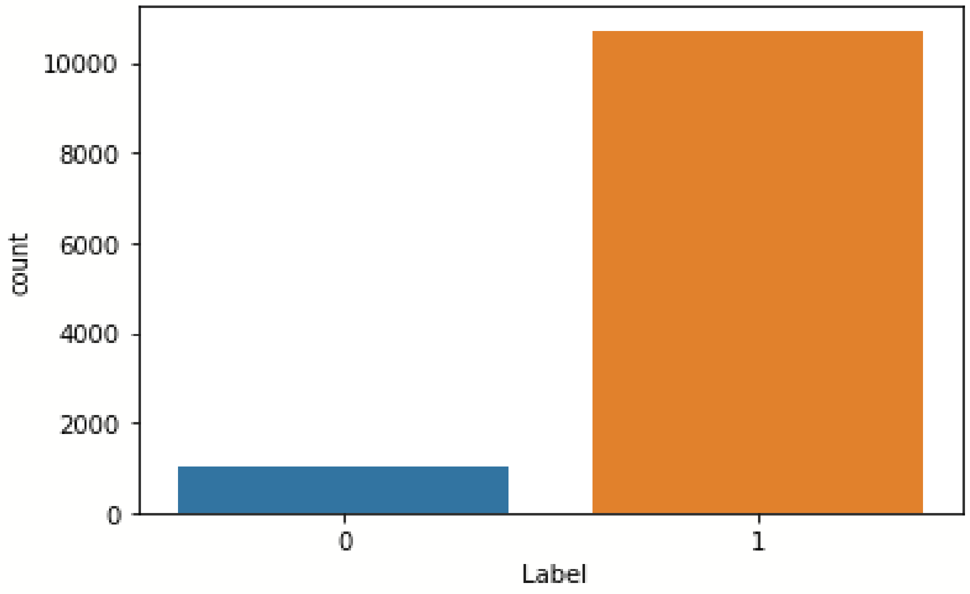
```
dic1 = {'BENIGN':0,'DDoS':1,'PortScan':1, 'Bot':1,'Infiltration':1,'Web Attack \x96 Brute Force':
1,'Web Attack \x96 XSS':1,
        'Web Attack \x96 Sql Injection':1,'FTP-Patator':1,'SSH-
Patator':1,'DoS slowloris':1,'DoS Slowhttptest':1,
         'DoS Hulk':1,'DoS GoldenEye':1, 'Heartbleed':1}


df1.replace({' Label':dic1},inplace=True)
df1[' Label'].unique()
```

OUTPUT:

```
array([0, 1])
```

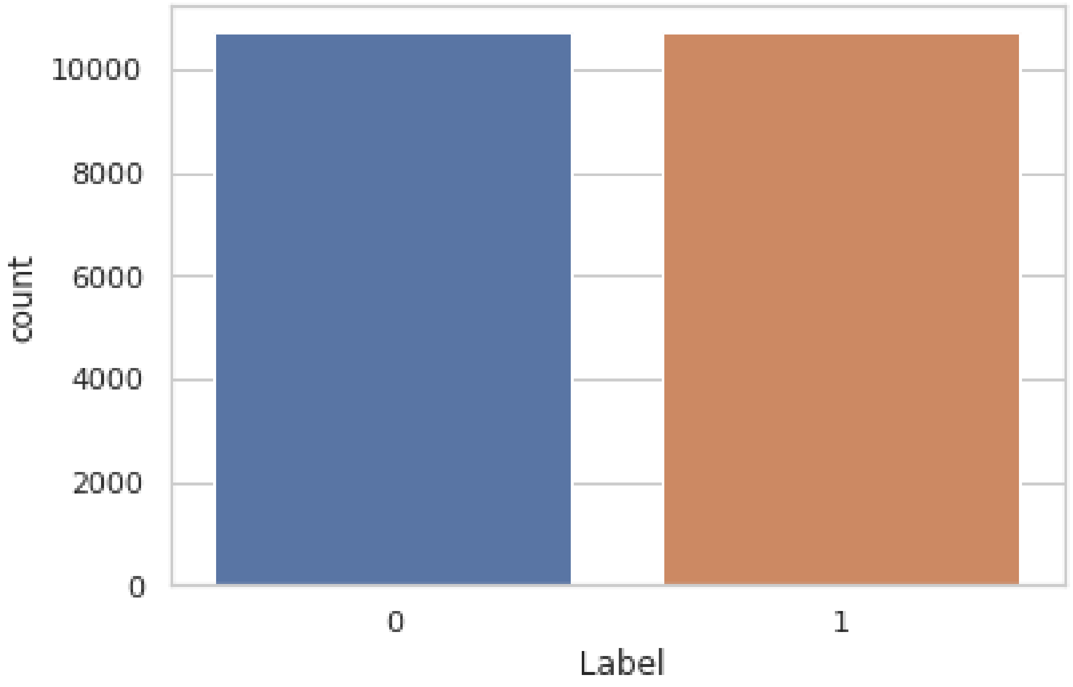
12.4 Binary Class With Imbalanced data Visulization :

```
sns.countplot(x=' Label',data=df1)
```



12.5 Binary Classification With Balanced the Data Visualization:

```
sns.countplot(x=' Label',data=df3)
```



12.6 Multiclass classification

In machine learning, multiclass or multinomial classification is the problem of classifying instances into one of three or more classes (classifying instances into one of two classes is called binary classification).

While many classification algorithms (notably multinomial logistic regression) naturally permit the use of more than two classes, some are by nature binary algorithms; these can, however, be turned into multinomial classifiers by a variety of strategies.

Multiclass classification should not be confused with multi-label classification, where multiple labels are to be predicted for each instance.

In this data set I have applying the machine learning algorithm K-Nearest Neighbors Classifier showed better accuracy, detection rate and performance using hyper parameter tuning in comparison to the other Classifier techniques.

12.7 Implementation of multiclass Classification problem with Python:

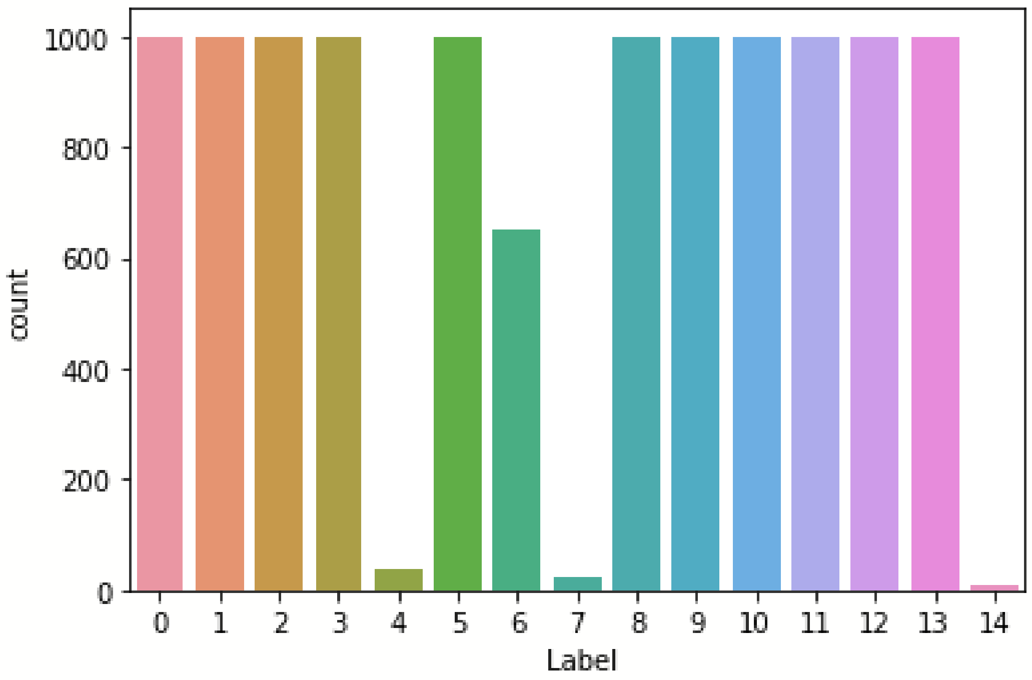
```
dict1 = {'BENIGN':0,'DDoS':1,'PortScan':2, 'Bot':3,'Infiltration':4,'Web Attack \x96 Brute Force':
5,'Web Attack \x96 XSS':6,
        'Web Attack \x96 Sql Injection':7,'FTP-Patator':8,'SSH-
Patator':9,'DoS slowloris':10,'DoS Slowhttptest':11,
         'DoS Hulk':12,'DoS GoldenEye':13, 'Heartbleed':14}
df.replace({' Label':dict1},inplace=True)
```

Output:

```
df[' Label'].unique()
array([ 0,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14,  1])
```

12.7 Multiclassification Problem with Imbalanced Data Visualization:

```
sns.countplot(x=' Label',data=df)
```



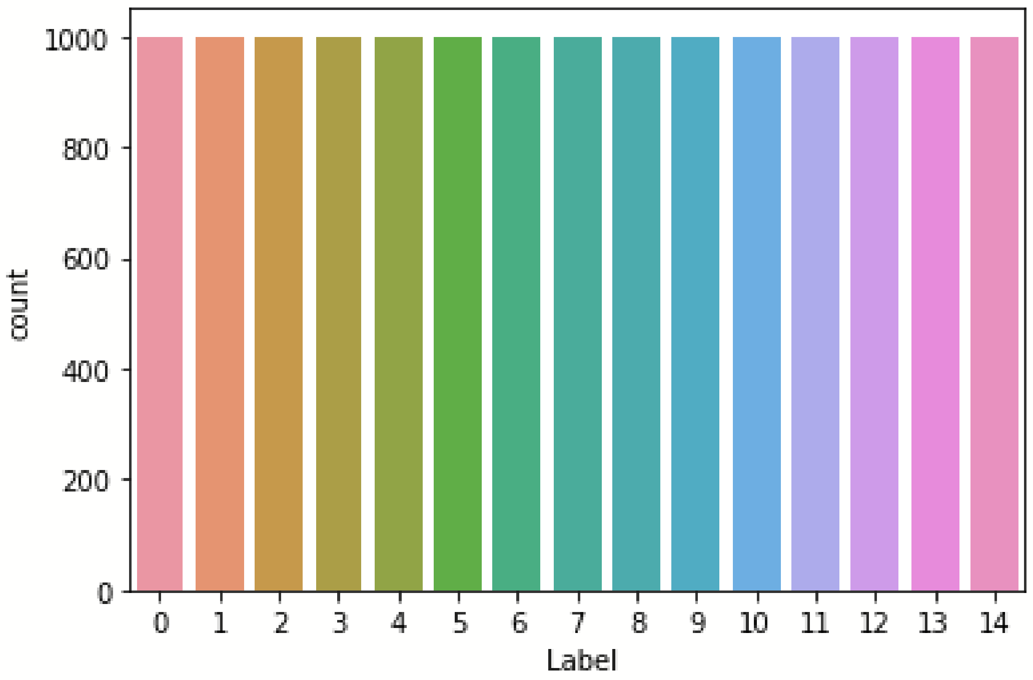
12.8 Multiclassification problem with Balanced the Data:

df2[' Label'].value_counts()

Output:

```
7       1000
14      1000
6       1000
13      1000
5       1000
12      1000
4       1000
11      1000
3       1000
10      1000
2       1000
9       1000
1       1000
8       1000
0       1000
```

sns.countplot(x=' Label',data=df2)



13. Confusion matrix for a two-class problem

	Actual value		
Predicted value		**Positive (1)**	**Negative(0)**
	Positive (1)	(TP)True positive	(FP) False positive
	Negative (0)	(FN) False Negative	(TN)True Negative

There are four important terms:

True Positives: The cases in which we predicted YES and the actual output was YES.

True Negatives: The cases in which we predicted NO and the actual output was NO.

False Positives: The cases in which we predicted YES and the actual output was NO.

False Negatives: The cases in which we predicted NO and the actual output was YES.

False Positive Rate and True Positive Rate both have values in the range [0, 1].

Accuracy: Classification Accuracy is what we usually mean, when we use the term accuracy. It is the ratio of number of correct predictions to the total number of input samples.

$$Accuracy = \frac{TruePositives + FalseNegatives}{TotalNumberofSamples}$$

It works well only if there are equal number of samples belonging to each class.

Precision:

It is the number of correct positive results divided by the number of positive results predicted by the classifier.

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

Recall:

It is the number of correct positive results divided by the number of ***all*** relevant samples (all samples that should have been identified as positive).

$$Precision = \frac{TruePositives}{TruePositives + FalseNegatives}$$

Area under Curve: Area under Curve (AUC) is one of the most widely used metrics for evaluation. It is used for binary classification problem. AUC of a classifier is equal to the probability that the classifier will rank a randomly chosen positive example higher than a randomly chosen negative example.

Receiver Operating Characteristic (ROC) Curve:

The Receiver Operating Characteristic Curve, better known as the **ROC Curve,** is an excellent method for measuring the performance of a Classification model. The **True Positive Rate (TPR)** is plot against **False Positive Rate (FPR)** for the probabilities of the classifier predictions. Then, the area under the plot is calculated.

F1 score:

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it’s better to look at both Precision and Recall.

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

14. Results and Analysis

14. Results analysis for Binary Classifier Models:

Accuracy = TP+TN / (TP+TN+FP+FN)

Detection Rate = TP / (TP+FN)

False alarm rate = FP / (TN+FP)

Where TP=True Positive, TN=True negative, FP= False positive, FN=False Negative

The true positive (TP) and true negative (TN) are correct classifications. A false positive (FP) occurs when the outcome is incorrectly predicted as yes (or positive) when it is actually no (negative). A false negative (FN) occurs when the outcome is incorrectly predicted as negative when it is actually positive.

In this section I have analysed the Machine Learning algorithm for evaluating the proposed algorithm and compared their results with those obtained using standard algorithm using similar values and parameters. This comparison was carried out using parameter like the accuracy of the technique in detecting the attacks which affected the training and the testing IDS data set. Furthermore I have tested various different classifier like Random Forest Classifier, Logistic Regression, SVM Classifier, Decision Tree Classifier, K-Nearest Neighbors Classifier, Naïve Bayes Classifier, Two Naïve Bayes Classifier, Multinomial Naïve Bayes Classifier XGBoost Classifier, SGD Classifier, AdaBoost Classifier, in addition to their proposed algorithm using IDS Dataset. The compared their results to all the published studies between 2013-2017 with the help of the feature selection techniques. Also they determined the accuracy and the robustness of their approach using various parameters. Their results proved that their proposed enhanced XGBoost Classifier algorithm showed better accuracy and performance in comparison to the other techniques. Also, the XGBoost Classifier algorithm showed a better performance of such the other algorithms.

All Classifier Showed scores, Precision, Recall, F1_score and Accuracy in Table 1

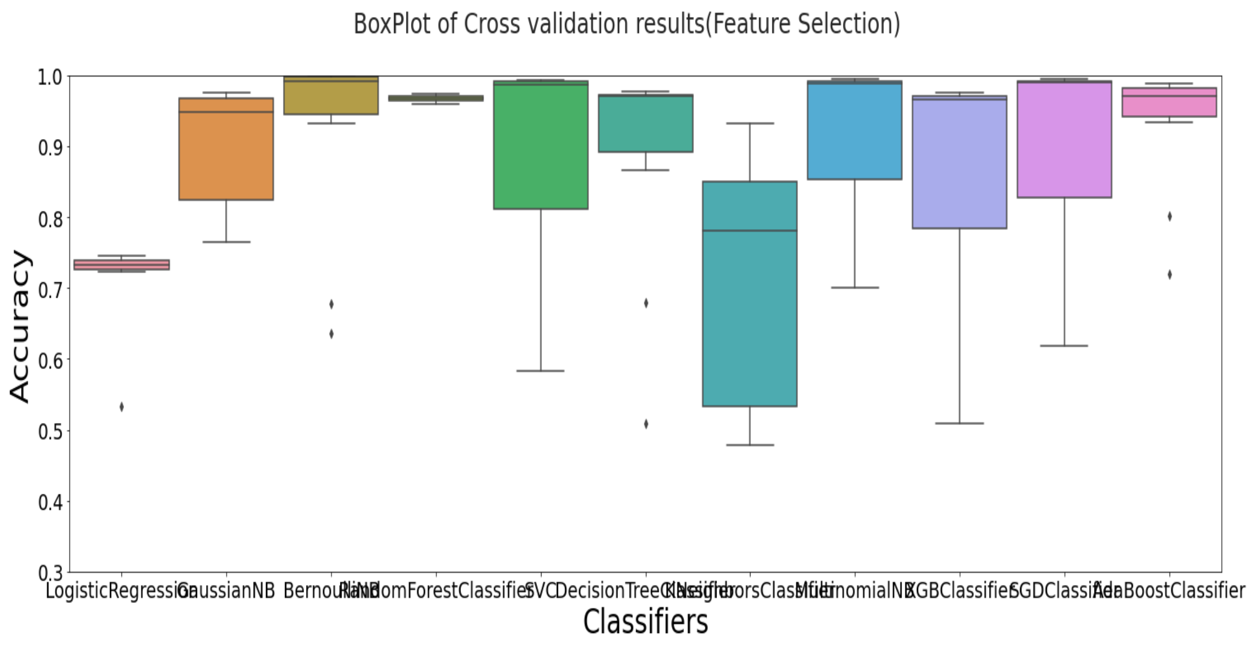
14.1 Cross validation mean, Score, Precision, Recall, Accuracy for Binary Classifier Models:

Table-1

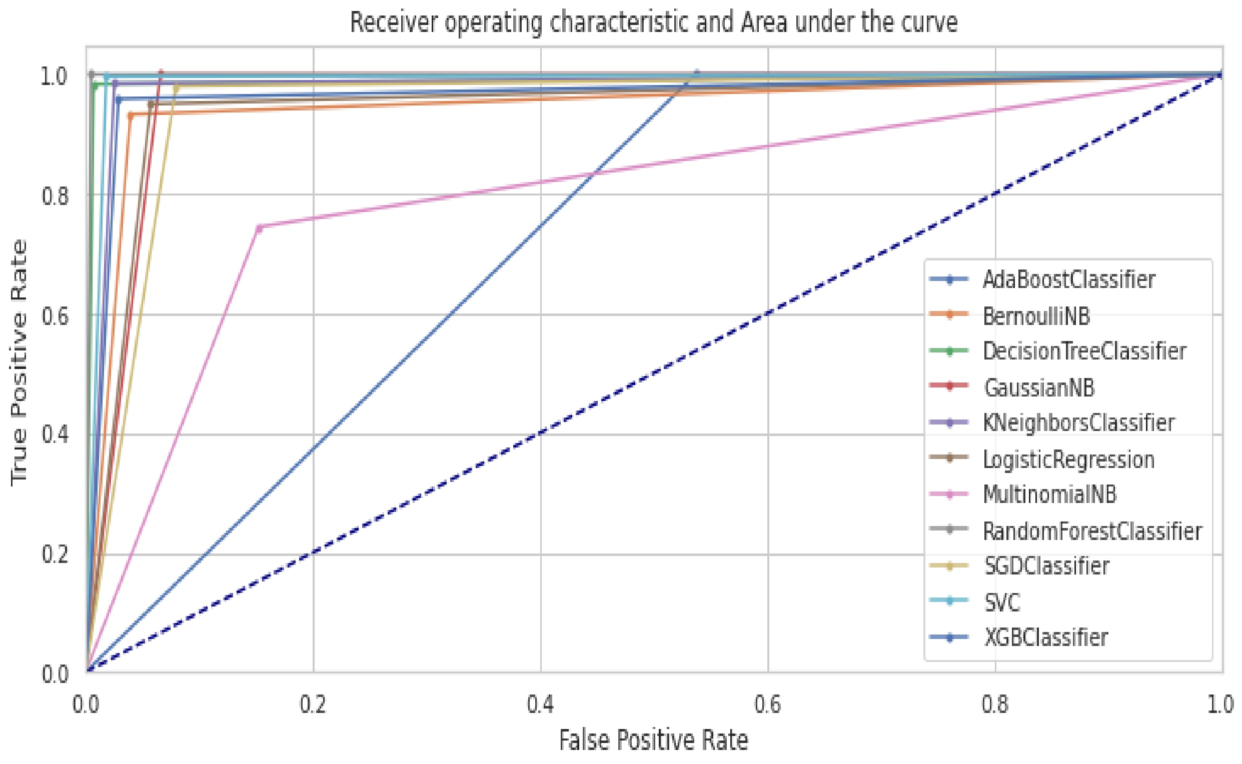
Binary Classifier Models

	scores	Precision	Recall	f1_score	Accuracy (%)
AdaBoostClassifier	0.729167	0.729167	0.729167	0.729167	72.916667
BernoulliNB	0.946984	0.946984	0.946984	0.946984	94.698383
DecisionTreeClassifier	0.988340	0.988340	0.988340	0.988340	98.833955
GaussianNB	0.966729	0.966729	0.966729	0.966729	96.672886
KNeighborsClassifier	0.979633	0.979633	0.979633	0.979633	97.963308
LogisticRegression	0.946362	0.946362	0.946362	0.946362	94.636194
MultinomialNB	0.796642	0.796642	0.796642	0.796642	79.664179
RandomForestClassifier	0.997668	0.997668	0.997668	0.997668	99.766791
SGDClassifier	0.950093	0.950093	0.950093	0.950093	95.009328
SVC	0.989117	0.989117	0.989117	0.989117	98.911692
XGBClassifier	0.965174	0.965174	0.965174	0.965174	96.517413

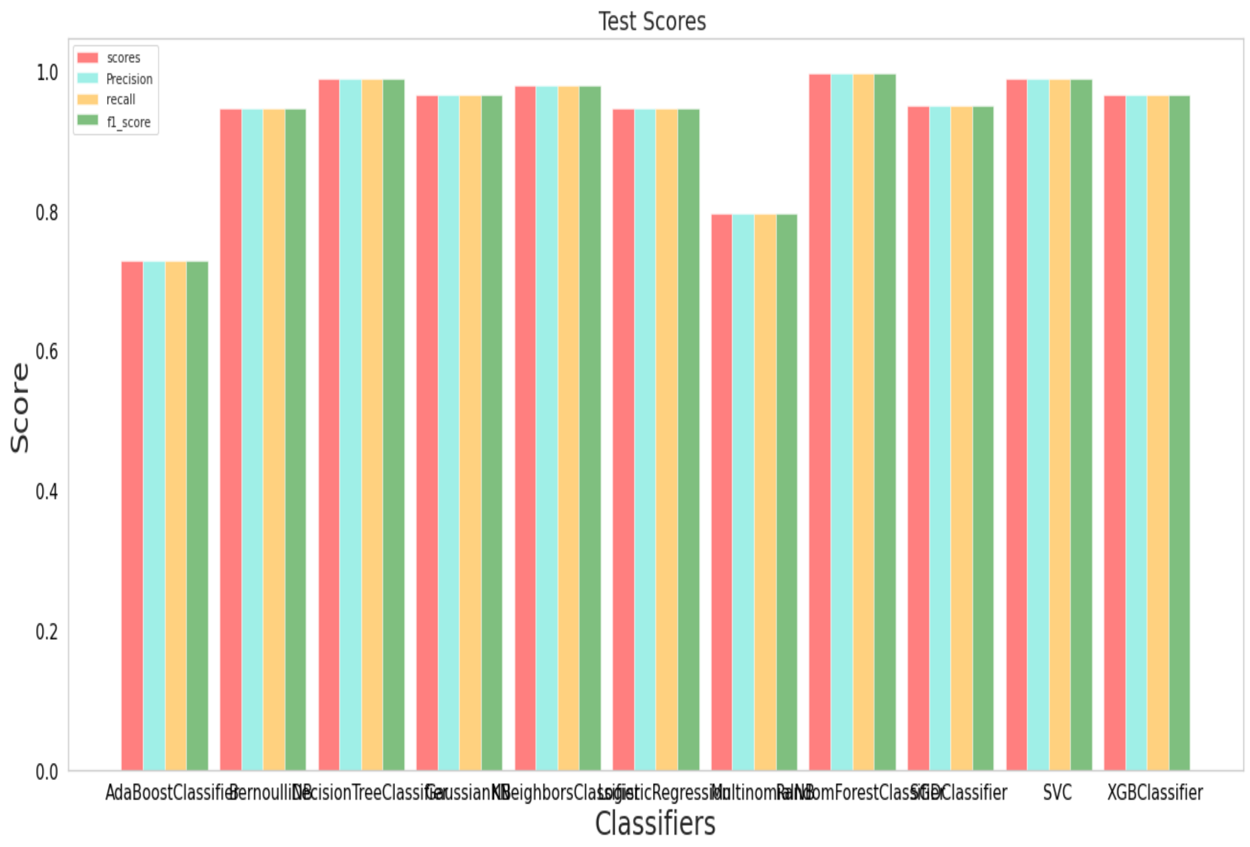
14.3 Visualization Box Plot for Binary Classifier Models:



14.5 Visualization ROC and AUC curve for Binary Classifier Models:



14.7 Visualization for Bar plots Binary Classifier Models:



15. Results Analysis for Multiple class classifier Models

Accuracy = TP+TN / (TP+TN+FP+FN)

Detection Rate = TP / (TP+FN)

False alarm rate = FP / (TN+FP)

Where TP=True Positive, TN=True negative, FP= False positive, FN=False Negative

The true positive (TP) and true negative (TN) are correct classifications. A false positive (FP) occurs when the outcome is incorrectly predicted as yes (or positive) when it is actually no (negative). A false negative (FN) occurs when the outcome is incorrectly predicted as negative when it is actually positive.

In this section I have analysed the Machine Learning algorithm for evaluating the proposed algorithm and compared their results with those obtained using standard algorithm using similar values and parameters. This comparison was carried out using parameter like the accuracy of the technique in detecting the attacks which affected the training and the testing IDS data set. Furthermore I have tested various different classifier like Random Forest Classifier, Logistic Regression, SVM Classifier, Decision Tree Classifier, K-Nearest Neighbors Classifier, Naïve Bayes Classifier, Two Naïve Bayes Classifier, Multinomial Naïve Bayes Classifier XGBoost Classifier, SGD Classifier, AdaBoost Classifier, in addition to their proposed algorithm using IDS Dataset. The compared their results to all the published studies between 2013-2017 with the help of the feature selection techniques. Also they determined the accuracy and the robustness of their approach using various parameters. Their results proved that their proposed enhanced XGBoost Classifier algorithm showed better accuracy and performance in comparison to the other techniques. Also, the XGBoost Classifier algorithm showed a better performance of such the other algorithms.

All Classifier Showed scores, Precision, Recall, F1_score and Accuracy in Table 2

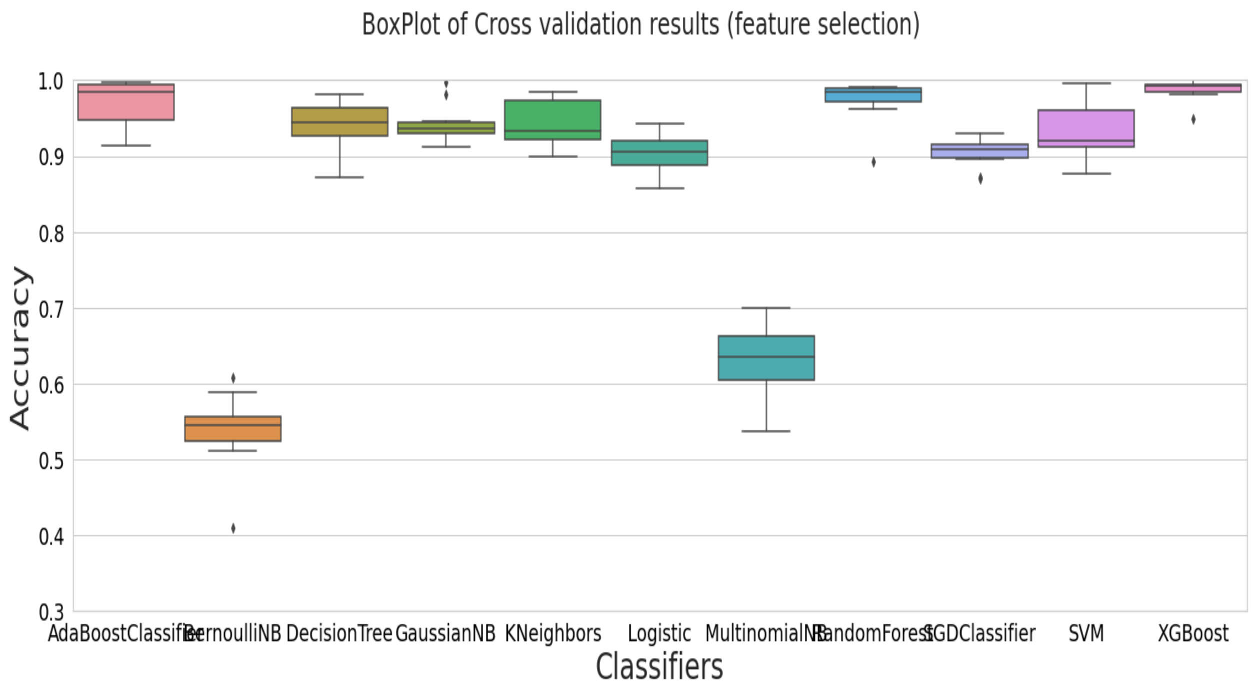
15.2 Cross validation mean, Score, Precision, Recall, Accuracy for Multiclass Classifier Models in Table-2

Table-2

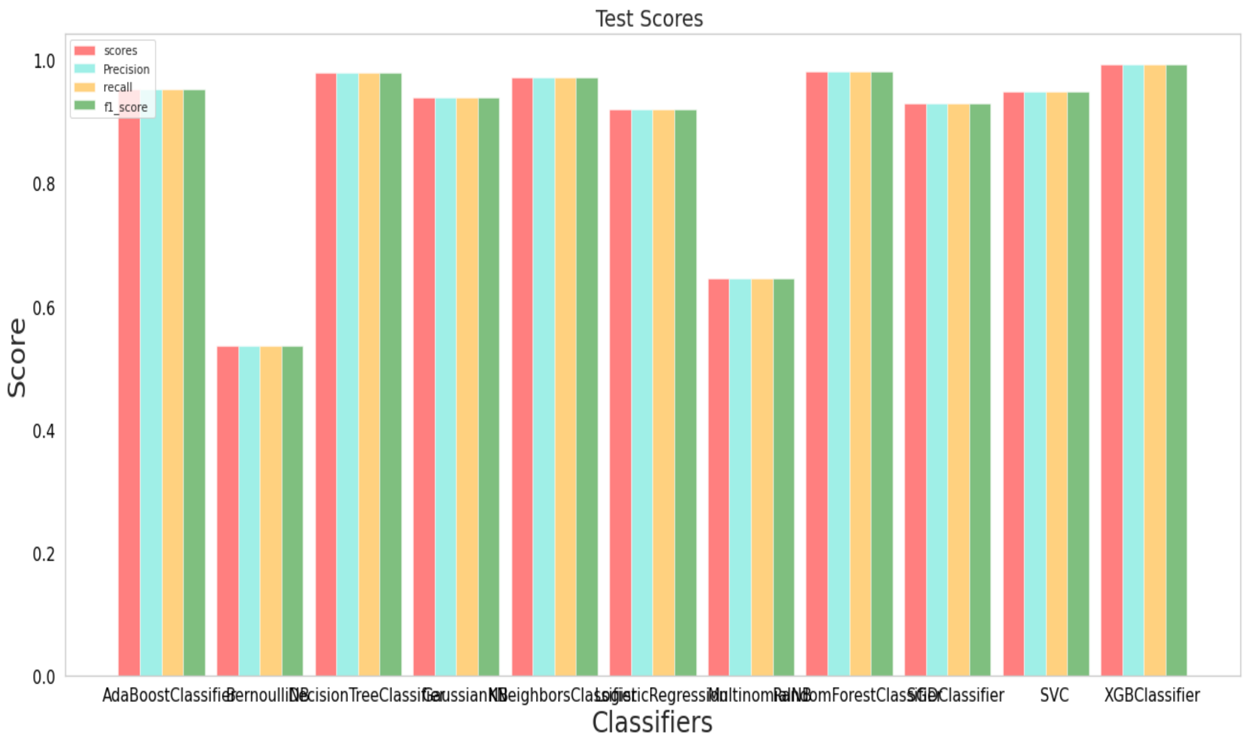
Multiclass classifier Models

	scores	Precision	Recall	f1_score	Accuracy (%)
AdaBoostClassifier	0.952667	0.952667	0.952667	0.952667	95.266667
BernoulliNB	0.537556	0.537556	0.537556	0.537556	53.755556
DecisionTreeClassifier	0.980444	0.980444	0.980444	0.980444	98.044444
GaussianNB	0.940000	0.940000	0.940000	0.940000	94.000000
KNeighborsClassifier	0.972222	0.972222	0.972222	0.972222	97.222222
LogisticRegression	0.921111	0.921111	0.921111	0.921111	92.111111
MultinomialNB	0.645778	0.645778	0.645778	0.645778	64.577778
RandomForestClassifier	0.983111	0.983111	0.983111	0.983111	98.311111
SGDClassifier	0.930667	0.930667	0.930667	0.930667	93.066667
SVC	0.950222	0.950222	0.950222	0.950222	95.022222
XGBClassifier	0.994667	0.994667	0.994667	0.994667	99.466667

15.4 Visualization Box Plot for Multiclass Classifier Models:



15.7 Visualization for Bar plots Multiclass Classifier Models:



16. Conclusion:

Finally, the conclusion of the study and the summary of the results obtained from all the experiments carried out using the machine learning algorithm for anomaly detection have been
Presented in this section. Hence, intrusion detection is a very important feature in the network security. Furthermore, the misuse detection methods are unable to detect the unknown attacks; hence, anomaly detection needs to be used for identifying such attacks, the Machine learning technique is applied in the anomaly-based detection techniques for improving the intrusion detection accuracy rate. We have developed and proposed the Enhanced Machine learning Classification Algorithm for the intrusion and anomaly detection. We also have compared the results of this algorithm with other machine learning approaches and it was seen that the proposed algorithm showed a better performance.