

**NETWORK INTRUSION DETECTION USING  
SUPERVISED MACHINE LEARNING  
TECHNIQUE WITH FEATURE SELECTION**

**A PROJECT REPORT**

*Submitted in partial fulfillment of the  
requirements for the award of the degree of*

**BACHELOR OF TECHNOLOGY  
in  
COMPUTER SCIENCE & ENGINEERING**

*Submitted by*

**Y. EASHITHA**  
**Roll No : 204N1A0560**

**B. RAMA KRISHNA**  
**Roll No : 204N1A0530**

**D. ABHILASHA**  
**Roll No : 204N1A0515**

**P. SUKEERTHI**  
**Roll No : 204N1A0546**

**Under the Guidance of**  
**G. RAJESH**  
**Assistant Professor**



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING  
VISVODAYA ENGINEERING COLLEGE  
KAVALI-524 201, NELLORE DISTRICT, A.P.**

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY,  
ANANTAPUR, A.P., INDIA**

**MAY 2024**

## **BONAFIDE CERTIFICATE**

Certified that this project report “**NETWORK INTRUSION DETECTION USING SUPERVISED MACHINE LEARNING TECHNIQUE WITH FEATURE SELECTION**” is the Bonafide work done by “**Y.EASHITHA (204N1A0560), B.RAMA KRISHNA (204N1A0530), D.ABHILASHA (204N1A0515), P.SUKEERTHI (204N1A0546)**”, who carried out the project under my guidance during the year 2023-2024, towards partial fulfillment of the requirements of the Degree of Bachelor of Technology in Computer Science & Engineering from Jawaharlal Nehru Technological University, Anantapur. The results embodied in this report have not been submitted to any other University for the award of any degree.

**G. RAJESH**

Assistant Professor

**PROJECT SUPERVISOR**

DEPARTMENT OF CSE

**P.ESWARAIAH**

Associate Professor

**HEAD OF THE DEPARTMENT**

DEPARTMENT OF CSE

External Viva voce conducted on -----

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

**VISVODAYA ENGINEERING COLLEGE**  
**KAVALI**

**CERTIFICATE OF AUTHENTICATION**

We solemnly declare that this project report “**NETWORK INTRUSION DETECTION USING SUPERVISED MACHINE LEARNING TECHNIQUE WITH FEATURE SELECTION**” is the bonafide work done purely by us, carried out under the supervision of **G. RAJESH**, Assistant Professor towards partial fulfillment of the requirements of the Degree of **BACHELOR OF TECHNOLOGY** in **Computer Science & Engineering** from Jawaharlal Nehru Technological University, Anantapur during the year 2023-2024.

It is further certified that this work has not been submitted, either in part or in full, to any other department of the Jawaharlal Nehru Technological University, or any other University, institution or elsewhere, or for publication in any form.

DATE:

SIGNATURE OF THE STUDENTS

*Y. EASHITHA (204N1A0560)*

*B. RAMA KRISHNA (204N1A0530)*

*D. ABHILASHA (204N1A0515)*

*P. SUKEERTHI (204N1A0546)*

## ACKNOWLEDGEMENT

We express heartfelt gratitude towards our institution, **VISVODAYA ENGINEERING COLLEGE, KAVALI** for giving an opportunity for the successful completion of our degree.

We express great sense of gratitude and indebtedness to our Honorable Chairman **Mr. D. Vidyadhara Kumar Reddy** and Academic Director **Dr. D. Prathyusha Reddi** for promoting excellent academic environment in the course.

It gives us immense pleasure to express a gratitude to our respected Principal **Dr. Dattatreya Sarma**, who gave us an opportunity in completing this course.

We deeply express our profound gratitude and wholehearted thanks to our beloved Head of the Department **P. ESWARAIAH**, Assoc. Professor & HOD, Department of Computer Science and Engineering, who provided us with necessary facilities, guidance and endless encouragement which helped us a lot in completing the project with in time.

We express our great sense of gratitude and indebtedness to our respected guide **Mr.G.RAJESH**, Assistant Professor for his inspiring guidance and his encouragement throughout the course and project.

We also express our gratitude to our lecturers and lab coordinators, non-teaching staff and friends who directly or indirectly helped us in completing the project successfully and providing us with suitable suggestions and guidance throughout.

## **ABSTRACT**

A novel supervised machine learning system is developed to classify network traffic whether it is malicious or benign. To find the best model considering detection success rate, combination of supervised learning algorithm and feature selection method have been used. Through this study, it is found that Artificial Neural Network (ANN) based machine learning with wrapper feature selection outperform support vector machine (SVM) technique while classifying network traffic. To evaluate the performance, NSL-KDD dataset is used to classify network traffic using SVM and ANN supervised machine learning techniques. Comparative study shows that the proposed model is efficient than other existing models with respect to intrusion detection success rate.

## **TABLE OF CONTENTS**

<b>CHAPTER NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
<b>1.</b>	<b>INTRODUCTION</b>	<b>1-4</b>
	1.1 Problem Statement	1
	1.2 About the Project	1
	1.3 Objectives of the Project	2
	1.4 Existing system	3
	1.4.1 Drawbacks of Existing system	3
	1.5 Proposed system	3
	1.5.1 Advantages of Proposed system	4
<b>2.</b>	<b>LITERATURE SURVEY</b>	<b>5-8</b>
<b>3.</b>	<b>SYSTEM ANALYSIS</b>	<b>9-13</b>
	3.1 Introduction	9
	3.2 Feasibility Study	11
	3.2.1 Economical Feasibility	11
	3.2.2 Technical Feasibility	12
	3.2.3 Social Feasibility	12
	3.3 System Specifications	13
	3.3.1 Software Requirements	13
	3.3.2 Hardware Requirements	13
<b>4.</b>	<b>SYSTEM DESIGN</b>	<b>14-38</b>
	4.1 System Architecture	14
	4.2 Module description	14
	4.3 Algorithms used in this project	15
	4.3.1 Support Vector Machine	15
	4.3.2 Artificial Neural Network	21
	4.4 Detailed Design	28

<b>5.</b>	<b>IMPLEMENTATION</b>	<b>39-72</b>
	5.1 What is Python	39
	5.1.1 Advantages of Python	40
	5.1.2 Disadvantages of Python	42
	5.2 History of Python	43
	5.3 What is Machine Learning	44
	5.3.1 Categories of Machine Learning	45
	5.3.2 Need of Machine Learning	46
	5.3.3 Challenges in Machines Learning	47
	5.3.4 Applications of Machines Learning	48
	5.3.5 How to Start Learning Machine Learning?	49
	5.3.6 Advantages of Machine learning	52
	5.3.7 Disadvantages of Machine Learning	52
	5.4 Python Development Steps	53
	5.5 Purpose	54
	5.6 Python for Machine Learning	55
	5.7 Modules Used in Project	56
	5.8 Installation of python in windows and mac	57
	5.9 Coding	64
<b>6.</b>	<b>TEST RESULTS</b>	<b>73-76</b>
	6.1 Unit Testing	75
	6.2 Integration Testing	75
	6.3 Acceptance Testing	76
<b>7.</b>	<b>RESULTS AND DISCUSSIONS</b>	<b>77-82</b>
<b>8.</b>	<b>CONCLUSION &amp; FUTURE WORK</b>	<b>83</b>
<b>9.</b>	<b>REFERENCES</b>	<b>84</b>

## **LIST OF FIGURES**

<b>S.NO</b>	<b>FIGURE</b>	<b>PAGE NO</b>
4.1	System Architecture	14
4.3.2	Architecture of ANN	23
4.3.2	Working of ANN	26
5.3	Machine Learning	44
5.5	Use case diagram	31
5.5	Class diagram	32
5.5	Sequence Diagram	33
5.5	Data Flow Diagram	34
5.5	Component Diagram	35
5.5	Activity Diagram	36
5.5	State Chart Diagram	37
5.5	Flow Chart Diagram	38



## **LIST OF ABBREVIATIONS**

<b>S.NO</b>	<b>ABBREVIATION</b>
1.	ML-Machine Learning
2.	SVM-Support Vector Machine
3.	ANN-Artificial Neural Network
4.	UML-Unified Modeling Language
5.	DFD-Data Flow Diagram

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 PROBLEM STATEMENT:**

In this project we are evaluating performance of two supervised machine learning algorithms such as SVM (Support Vector Machine) and ANN (Artificial Neural Networks). Machine learning algorithms will be used to detect whether request data contains normal or attack (anomaly) signatures. Now-a-days all services are available on internet and malicious users can attack client or server machines through this internet and to avoid such attack request IDS (Network Intrusion Detection System) will be used, IDS will monitor request data and then check if it contains normal or attack signatures, if contains attack signatures then request will be dropped.

IDS will be trained with all possible attack's signatures with machine learning algorithms and then generate train model, whenever new request signatures arrived then this model applied on new request to determine whether it contains normal or attack signatures. In this paper we are evaluating performance of two machine learning algorithms such as SVM and ANN and through experiment we conclude that ANN outperform existing SVM in terms of accuracy.

### **1.2 ABOUT THE PROJECT:**

In the modern computer world, use of the internet is increasing day by day. However, the increasing use of the internet creates some security issues. These days, such new type of security attacks occurs every day and it is not easy to detect and prevent those attacks effectively. One common method of attack involves sending large amount of request to site or server and server will be unable to handle such huge requests and site will be offline for many days. This type of attack is called distributed denial of service (DDOS) attack, which act as a major security threat to internet services and most critical attack for cyber security world.

Detection and prevention of Distributed Denial of Service Attack (DDOS) becomes a crucial process for the commercial organizations that uses the internet. Different approaches have been adopted to process traffic information collected by

a monitoring station (Routers and Servers) to distinguish malicious traffic such as DDOS attack from normal traffic in Intrusion Detection Systems (IDS). In general, Machine learning techniques can be designed and implemented with the intrusion systems to protect the organizations from malicious traffic. Specifically, supervised clustering techniques allow to effectively distinguish the normal traffic from malicious traffic with good accuracy. In this paper, machine learning algorithms are used to detect DDoS attacks collected from “KDD cup 99 Dataset”, pre-processing and feature selection technique is used on the dataset to enhance the performance of the classifiers and reduce the detection time.

The classification algorithms such as C4.5 decision tree and Navie Bayes is applied on the training dataset and the implementation of the algorithm is done using spyder tool. The performance comparison of algorithms is shown using confusion matrix and it is found that C4.5 decision is more efficient in detection of DDOS attack. The proposed method can be used as DDOS defense system.

### **1.3 OBJECTIVE:**

In supervised learning, learning data comes with labels or desired outputs and the objective is to find a general rule that maps inputs to outputs. This kind of learning data is called labeled data. The learned Rule is then used to label new data with unknown outputs. It involves building a machine learning model based that is based on labeled samples. Reducing the impact of attacks; and secondly the evaluation of the system IDS. Indeed, in one hand the IDSs collect network traffic information from some sources present in the network or the computer system and then use these data to enhance the systems safety.

In the other hand, the evaluation of IDS is a critical task. In fact, its important to note the difference between evaluating the effectiveness of an entire system and evaluating the characteristics of the system components.

In this project, we present an approach for IDS evaluating based on measuring the performance of its components. First of all, in order to implement the IDS SNORT components safely we have proposed a hardware platform.

Then we have tested it by using a generator of traffics and attacks based on Linux KALI (Backtrack) and Metasploite 3 Framework. The obtained results show that the IDS performance is closely related to the characteristics of these components.

#### **1.4 EXISTING SYSTEM:**

Studying the field of intrusion detection first started in 1980 and the first such model was published in 1987. For the last few decades, though huge commercial investments and substantial research were done, intrusion detection technology is still immature and hence not effective.

While network IDS that works based on signature have seen commercial success and widespread adoption by the technology-based organization throughout the globe, anomaly-based network IDS have not gained success in the same scale. Due to that reason in the field of IDS, currently anomaly-based detection is a major focus area of research and development. And before going to any wide scale deployment of anomaly-based intrusion detection system, key issues remain to be solved. But the literature today is limited when it comes to compare on how intrusion detection performs when using supervised machine learning techniques.

##### **1.4.1 Disadvantages of existing system:**

- Low prediction
- Low accuracy

#### **1.5 PROPOSED SYSTEM:**

The promise and the contribution machine learning did till today are fascinating. There are many real-life applications we are using today offered by machine learning. It seems that machine learning will rule the world in coming days. Hence, we came out into a hypothesis that the challenge of identifying new attacks or zero-day attacks facing by the technology enabled organizations today can be overcome using machine learning techniques.

Here we developed a supervised machine learning model that can classify unseen network traffic based on what is learnt from the seen traffic. We used both SVM and ANN learning algorithm to find the best classifier with higher accuracy.

### **1.5.1 Advantages of proposed system:**

- The new proposal was innovative as Hidden Naïve bayes which shows more advantage than traditional naïve bayes.
- which analyzes the large volume of network data and considers the complex properties of attack behaviors to improve the performance of detection speed and detection accuracy.

## **CHAPTER2**

### **LITERATURE SURVEY**

**Title:** NETWORK INTRUSION DETECTION USING SUPERVISED MACHINE LEARNING TECHNIQUE WITH FEATURE SELECTION.

Recently, Internet has become a part and parcel of daily life. The current internet-based information processing systems are prone to different kinds of threats which lead to various types of damages resulting in significant losses. Therefore, the importance of information security is evolving quickly. The most basic goal of information security is to develop defensive information systems which are secure from unauthorized access, use, disclosure, disruption, modification, or destruction. Moreover, information security minimizes the risks related to the three main security goals namely confidentiality, integrity, and availability.

Various systems have been designed in the past to identify and block the Internet-based attacks. The most important systems among them are intrusion detection systems (IDS) since they resist external attacks effectively. Moreover, IDSs provide a wall of defense which overcomes the attack of computer systems on the Internet. IDS could be used to detect different types of attacks on network communications and computer system usage where the traditional firewall cannot perform well. Intrusion detection is based on an assumption that the behaviors of intruders differ from a legal user. Generally, IDSs are broadly classified into two categories namely anomaly and misuse detection systems based on their detection approaches.

Anomaly intrusion detection determines whether deviation from the established normal usage patterns can be flagged as intrusions. On the other hand, misuse detection systems detect the violations of permissions effectively. Intrusion detection systems can be built by using intelligent agents and classification techniques. Most IDSs work in two phases namely preprocessing phase and intrusion detection phase. The intrusions identified by the IDSs can be prevented effectively by developing an intrusion prevention system.

This project mainly provides a survey on intelligent techniques proposed for developing IDSs. In addition, it explains about new IDS which has been developed using two proposed algorithms namely intelligent rule-based attribute selection

algorithm and intelligent rule-based enhanced multiclass support vector machine (IREMSVM). Intelligent IDSs are the ones considered to be intelligent computer programs situated in either a host or a network which analyzes the environment and acts flexibly to achieve higher detection accuracy. These programs compute the actions to be performed on the environment both by learning the environment and by firing rules of inference.

Intelligent IDSs are capable of decision making and constraint checking. In most intelligent systems, either rules are fired or agents are used for decision making. Moreover, a set of static agents or a set of mobile and static agents have been used to achieve a single goal. Intelligent intrusion detection systems have been developed by proposing intelligent techniques for preprocessing and effective classification.

Such IDSs have provided better detection rate in comparison with the other approaches.

**“A macro-social exploratory analysis of the rate of interstate cyber-victimization”:**

**Authors: H. Song, M. J. Lynch, and J. K. Cochran**

This study examines whether macro-level opportunity indicators affect cyber-theft victimization. Based on the arguments from criminal opportunity theory, exposure to risk is measured by state-level patterns of internet access (where users access the internet). Other structural characteristics of states were measured to determine if variation in social structure impacted cyber-victimization across states. The current study found that structural conditions such as unemployment and non-urban population are associated with where users access the internet. Also, this study found that the proportion of users who access the internet only at home was positively associated with state-level counts of cyber-theft victimization. The theoretical implications of these findings are discussed.

**“Incremental anomaly-based intrusion detection system using limited labeled data”:**

**Authors: P. Alaei and F. Noorbehbahani**

With the proliferation of the internet and increased global access to online media, cybercrime is also occurring at an increasing rate. Currently, both personal users and companies are vulnerable to cybercrime. A number of tools including firewalls and Intrusion Detection Systems (IDS) can be used as defense mechanisms. A firewall acts

as a checkpoint which allows packets to pass through according to predetermined conditions.

In extreme cases, it may even disconnect all network traffic. An IDS, on the other hand, automates the monitoring process in computer networks. The streaming nature of data in computer networks poses a significant challenge in building IDS. In this project, a method is proposed to overcome this problem by performing online classification on datasets. In doing so, an incremental naive Bayesian classifier is employed. Furthermore, active learning enables solving the problem using a small set of labeled data points which are often very expensive to acquire. The proposed method includes two groups of actions i.e. offline and online. The former involves data preprocessing while the latter introduces the NADAL online method. The proposed method is compared to the incremental naive Bayesian classifier using the NSL-KDD standard dataset.

There are three advantages with the proposed method: (1) overcoming the streaming data challenge; (2) reducing the high cost associated with instance labeling; and (3) improved accuracy and Kappa compared to the incremental naive Bayesian approach. Thus, the method is well-suited to IDS applications.

### **“Modelling and implementation approach to evaluate the intrusion detection system”**

**Authors: M. Saber, S. Chadli, M. Emharraf, and I. El Farissi**

Intrusions detection systems (IDSs) are systems that try to detect attacks as they occur or when they were over. Research in this area had two objectives: first, reducing the impact of attacks; and secondly the evaluation of the system IDS. Indeed, in one hand the IDSs collect network traffic information from some sources present in the network or the computer system and then use these data to enhance the systems safety. In the other hand, the evaluation of IDS is a critical task.

In this project, we present an approach for IDS evaluating based on measuring the performance of its components. First of all, in order to implement the IDS SNORT components safely we have proposed a hardware platform based on embedded systems. Then we have tested it by using a generator of traffics and attacks based on Linux KALI (Backtrack) and Metasploit 3 Framework. The obtained results show that the IDS performance is closely related to the characteristics of these components.



## **“Machine Learning Techniques for Intrusion Detection”**

**Authors: M. Zamani and M. Movahedi**

An Intrusion Detection System (IDS) is a software that monitors a single or a network of computers for malicious activities (attacks) that are aimed at stealing or censoring information or corrupting network protocols. Most techniques used in today's IDS are not able to deal with the dynamic and complex nature of cyber-attacks on computer networks. Hence, efficient adaptive methods like various techniques of machine learning can result in higher detection rates, lower false alarm rates and reasonable computation and communication costs.

In this project, we study several such schemes and compare their performance. We divide the schemes into methods based on classical artificial intelligence (AI) and methods based on computational intelligence (CI). We explain how various characteristics of CI techniques can be used to build efficient IDS.

## **“Intrusion Detection System and Intrusion Prevention System”**

**Authors: N. Chakraborty**

Intrusions in computing environment are a very common undesired malicious activity that is going on since the inception of computing resources. A number of security measures have taken place for the last three decades, but as Technology has grown up, so as the security threats. With the whole world depending on computers, being directly or indirectly, it is a very important issue to prevent the malicious activities and threats that can hamper the computing infrastructures.

Intrusion Detection System (IDS) and Intrusion Prevention System (IPS) are the standard measures to secure computing resources mostly in a network. They are deployed in a network for assuring an intrusion free computing environment. In this project, we discuss two technologies in details, their functionality, their performances and their effectiveness to stop the malicious activity over a computer network.

## **CHAPTER 3**

### **SYSTEM ANALYSIS**

#### **3.1 INTRODUCTION**

Network intrusion detection is a critical component of cybersecurity, aimed at safeguarding digital assets and infrastructure from malicious activities. With the increasing complexity and sophistication of cyber threats, traditional rule-based approaches to intrusion detection are often insufficient. As a result, there is a growing interest in leveraging machine learning techniques to enhance the detection of anomalous behaviors within network traffic.

This project focuses on the design and analysis of a system for network intrusion detection using supervised machine learning techniques, augmented with feature selection methodologies. By employing supervised learning, the system learns from labeled examples of normal and intrusive network activities, enabling it to identify and classify potential threats with greater accuracy and efficiency.

Designing a system for network intrusion detection using supervised machine learning techniques with feature selection involves several steps. Here's an outline of the process:

##### **1. Problem Definition:**

- Define the objectives: Determine what types of network intrusions you want to detect (e.g., DoS attacks, port scanning, unauthorized access).
- Identify the dataset: Collect or obtain a dataset that contains labeled examples of normal and intrusive network activities.

##### **2. Data Preprocessing:**

- Data Cleaning: Handle missing values, outliers, and inconsistencies in the dataset.
- Data Transformation: Encode categorical variables, normalize numerical features, and perform other transformations as needed.
- Feature Engineering: Create new features from the existing ones if necessary, which may help improve model performance.

### 3. Feature Selection:

- Identify relevant features: Use techniques like correlation analysis, mutual information, or domain knowledge to select features that are most relevant to the problem.
- Dimensionality Reduction: Apply techniques like Principal Component Analysis (PCA) or feature importance ranking to reduce the number of features while preserving the most relevant information.

### 4. Model Selection:

- a. Choose appropriate supervised machine learning algorithms for classification, such as:
  - i. Decision Trees
  - ii. Random Forest
  - iii. Support Vector Machines (SVM)
  - iv. Neural Networks
  - v. Gradient Boosting Machines (GBM)
  - vi. Naive Bayes
- b. Experiment with different algorithms and evaluate their performance using appropriate metrics (e.g., accuracy, precision, recall, F1-score).

### 5. Model Training:

- a. Split the dataset into training and testing sets (and possibly a validation set).
- b. Train the selected models using the training data.
- c. Tune hyperparameters using techniques like grid search or random search to optimize model performance.

### 6. Model Evaluation:

- a. Evaluate the trained models on the testing dataset using the chosen evaluation metrics.
- b. Perform cross-validation to ensure the robustness of the results.

- c. Analyze the confusion matrix to understand the model's performance in detecting different types of intrusions.

#### 7. Deployment:

- a. Once satisfied with the model performance, deploy the model in a production environment.
- b. Monitor the model's performance over time and retrain/update it as needed to adapt to evolving threats.

#### 8. Continuous Improvement:

- c. Regularly update the system with new data and retrain the model to keep it effective against emerging threats.
- d. Explore advanced techniques or ensemble methods to further improve detection accuracy and robustness.

### 3.2 FEASIBILITY STUDY

A feasibility study is an assessment conducted during the early stages of a project to evaluate its viability and determine whether it is worth pursuing. It involves analyzing various factors, such as technical, financial, operational, legal, and ethical considerations, to understand the potential risks, benefits, and constraints associated with the project. The primary purpose of a feasibility study is to provide decision-makers with the information they need to make informed choices about whether to proceed with the project or not.

Three key considerations in feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

#### 3.2.1 Economical Feasibility

- **Cost of Data Acquisition:** Estimate the cost associated with acquiring labeled datasets, whether from public repositories, commercial vendors, or data collection efforts.
- **Hardware and Software Costs:** Evaluate the cost of hardware infrastructure for model training and software licenses for machine learning libraries or tools.

- **Maintenance and Updates:** Estimate ongoing costs for maintaining and updating the system, including data storage, software updates, and personnel expenses.

### 3.2.2 Technical Feasibility

- **Data Availability:** Assess the availability of labeled datasets containing network traffic data for training and testing purposes. Explore public repositories, commercial sources, or proprietary data sources.
- **Feature Selection Techniques:** Evaluate the feasibility of implementing feature selection algorithms, considering computational resources and expertise required for implementation.
- **Model Training and Evaluation:** Determine if the hardware infrastructure is sufficient for training and evaluating machine learning models efficiently. Consider factors like processing power, memory, and storage requirements.

### 3.2.3 Social Feasibility

Social feasibility for network intrusion detection refers to the assessment of how acceptable, ethical, and beneficial the implementation of such a system is within a given social context.

- **Privacy Concerns:** Evaluate how the implementation of network intrusion detection may impact individuals' privacy rights. Consider whether the monitoring of network traffic and detection of potential intrusions could lead to the collection of sensitive information about users or employees.
- **Transparency and Accountability:** Assess whether the implementation of network intrusion detection systems is transparent to stakeholders, including employees, customers, and the general public. Ensure that there are clear policies and procedures in place for the collection, use, and protection of data, as well as mechanisms for accountability in case of misuse or abuse.
- **Ethical Implications:** Consider the ethical implications of monitoring and analyzing network traffic for the purpose of intrusion detection.

Evaluate whether the benefits of increased cybersecurity outweigh the potential risks of infringing on individuals' rights to privacy and autonomy.

- **Impact on Trust and Relationships:** Examine how the implementation of network intrusion detection systems may impact trust and relationships within the organization and with external stakeholders. Communicate openly and transparently about the reasons for implementing such systems and how they will be used to protect sensitive information and ensure the security of the network.

### 3.3 SYSTEM SPECIFICATION

#### 3.3.1 Software Requirements

Functional requirements for a secure cloud storage service are straightforward:

- 1.The service should be able to store the user's data.
- 2.The data should be accessible through any devices connected to the Internet.
- 3.Data should be shareable with other users.
- 4.The service should support SSO.
- 5.The service should be interoperable with other cloud storage services, enabling data migration from one CSP to another.

- **Operating System:** Windows.
- **Coding Language:** Python 3.7.

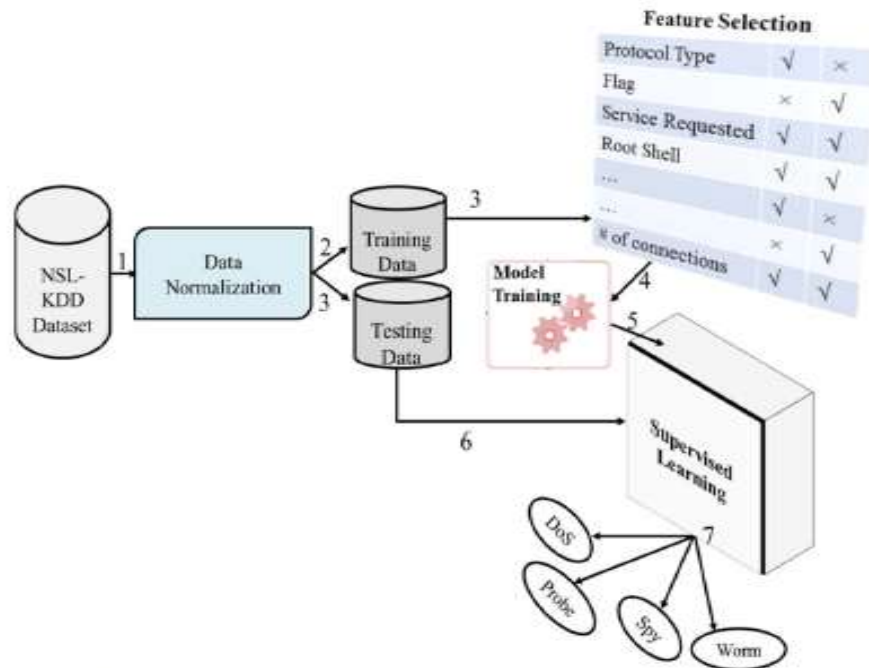
#### 3.3.2 Hardware Requirements:

- **Processor** - Pentium –III.
- **Speed** – 2.4 GHz.
- **RAM** - 512 MB (min).
- **Hard Disk** - 20 GB.
- **Floppy Drive** - 1.44 MB.
- **Key Board** - Standard Keyboard.
- **Monitor** – 15 VGA Colour.

## CHAPTER 4

### SYSTEM DESIGN

#### 4.1 SYSTEM ARCHITECTURE



**Fig 1: Proposed supervised machine learning classifier system**

#### 4.2 MODULE DESCRIPTION

Feature selection is an important part in machine learning to reduce data dimensionality and extensive research carried out for a reliable feature selection method. For feature selection filter method and wrapper method have been used.

In filter method, features are selected on the basis of their scores in various statistical tests that measure the relevance of features by their correlation with dependent variable or outcome variable. Wrapper method finds a subset of features by measuring the usefulness of a subset of feature with the dependent variable.

Hence filter methods are independent of any machine learning algorithm whereas in wrapper method the best feature subset selected depends on the machine learning algorithm used to train the model.

In wrapper method a subset evaluator uses all possible subsets and then uses a classification algorithm to convince classifiers from the features in each subset. The classifier considers the subset of feature with which the classification algorithm performs the best. To find the subset, the evaluator uses different search techniques like depth first search, random search, breadth first search or hybrid search.

The filter method uses an attribute evaluator along with a ranker to rank all the features in the dataset. Here one feature is omitted at a time that has lower ranks and then sees the predictive accuracy of the classification algorithm. Weights or rank put by the ranker algorithms are different than those by the classification algorithm.

Wrapper method is useful for machine learning test whereas filter method is suitable for data mining test because data mining has thousands of millions of features.

#### **4.3 ALGORITHMS USED IN THIS PROJECT: -**

##### **4.3.1. Support Vector Machine: -**

“Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes.

Support Vectors are simply the co-ordinates of individual observation. The SVM classifier is a frontier which best segregates the two classes (hyper-plane/ line).

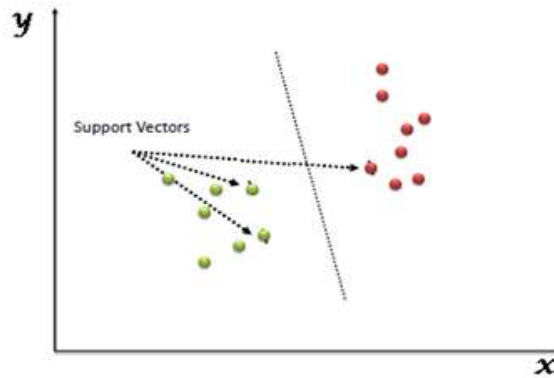
Support Vector Machine (SVM) is a powerful machine learning algorithm used for linear or nonlinear classification, regression, and even outlier detection tasks. SVMs can be used for a variety of tasks, such as text classification, image classification, spam detection, handwriting identification, gene expression analysis, face detection, and anomaly detection. SVMs are adaptable and efficient in a variety of applications because they can manage high-dimensional data and nonlinear relationships.



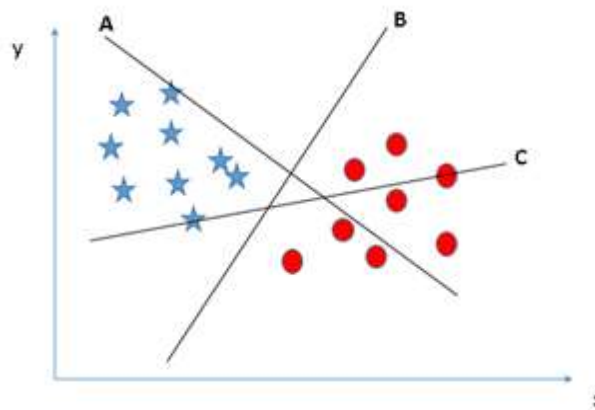
## How does it work?

Above, we got accustomed to the process of segregating the two classes with a hyper-plane. Now the burning question is “How can we identify the right hyper-plane?”.

Let's understand:

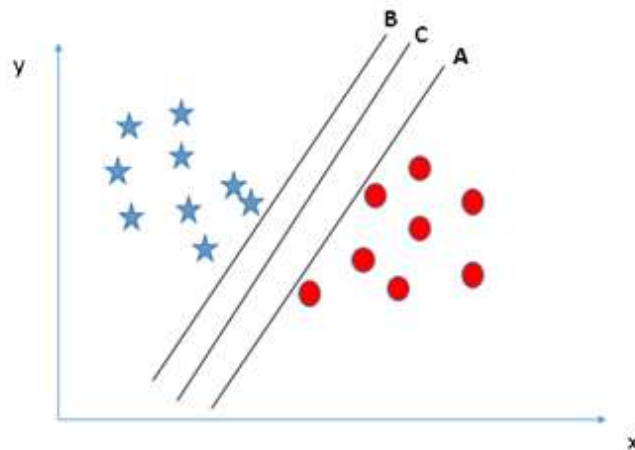


- **Identify the right hyper-plane (Scenario-1):** Here, we have three hyper-planes (A, B and C). Now, identify the right hyper-plane to classify star and circle.

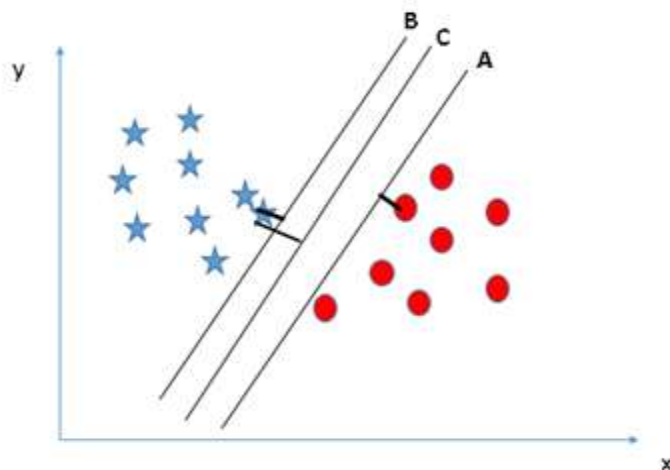


You need to remember a thumb rule to identify the right hyper-plane: “Select the hyper-plane which segregates the two classes better”. In this scenario, hyper-plane “B” has excellently performed this job.

- **Identify the right hyper-plane (Scenario-2):** Here, we have three hyper-planes (A, B and C) and all are segregating the classes well. Now, How can we identify the right hyper-plane?

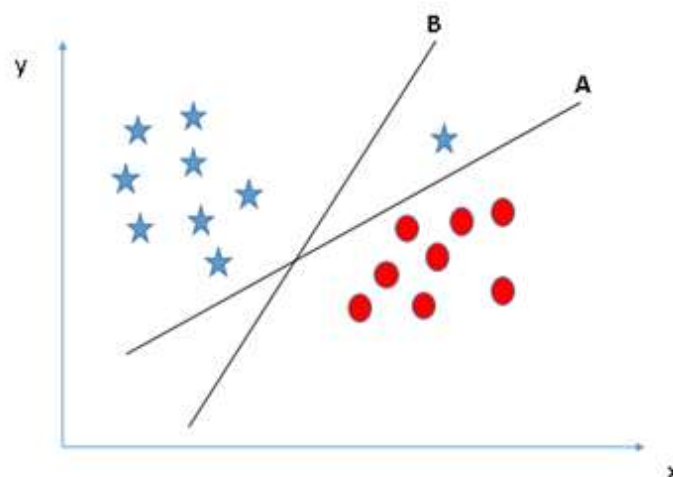


Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as **Margin**.



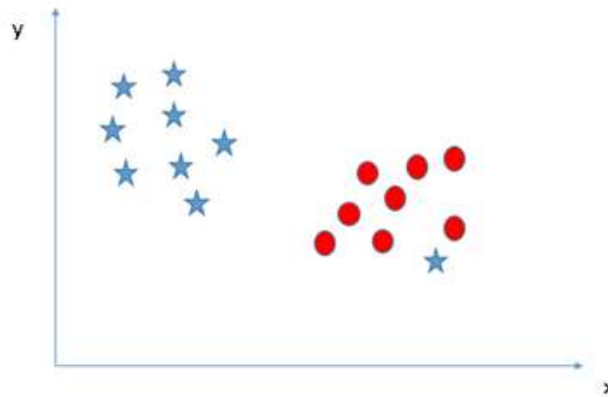
Look at the above figure, you can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C. Another lightning reason for selecting the hyper-plane with higher margin is robustness. If we select a hyper-plane having low margin then there is high chance of miss-classification.

- **Identify the right hyper-plane (Scenario-3):**

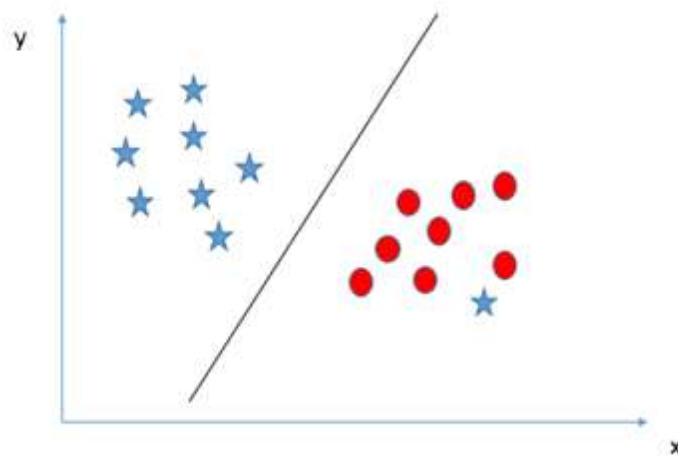


Some of you may have selected the hyper-plane **B** as it has higher margin compared to **A**. But here is the catch, SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is **A**.

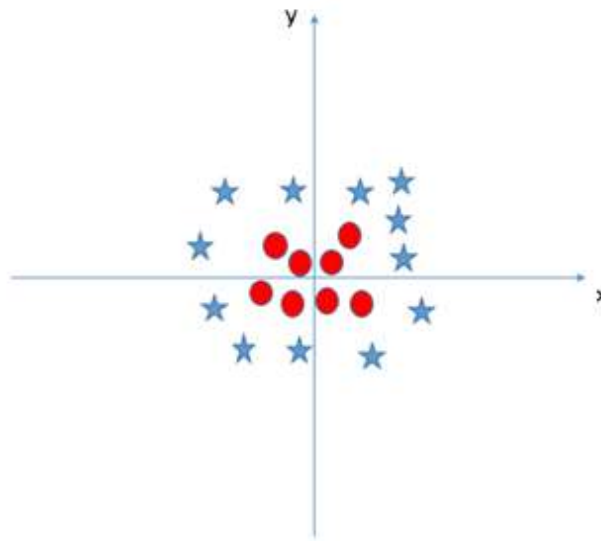
- **Can we classify two classes (Scenario-4)?** : Below, I am unable to segregate the two classes using a straight line, as one of the stars lies in the territory of other(circle) class as an outlier.



As I have already mentioned, one star at other end is like an outlier for star class. The SVM algorithm has a feature to ignore outliers and find the hyper-plane that has the maximum margin. Hence, we can say, SVM classification is robust to outliers.

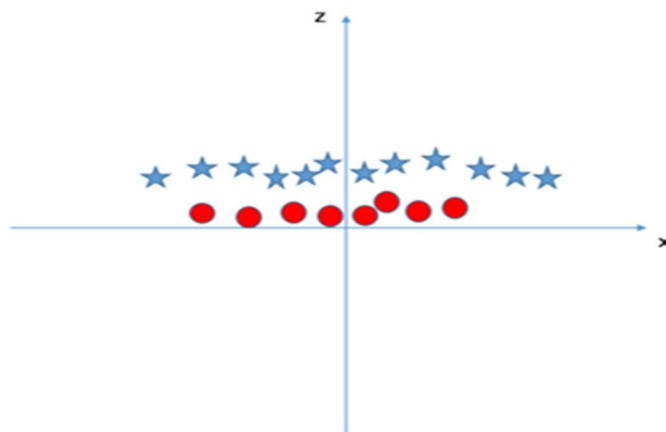


• **Find the hyper-plane to segregate to classes (Scenario-5):** In the scenario below, we can't have linear hyper-plane between the two classes, so how does SVM classify these two classes? Till now, we have only looked at the linear hyper-plane.



---

SVM can solve this problem. Easily! It solves this problem by introducing additional feature. Here we will add a new feature  $z=x^2+y^2$ . Now, let's plot the data points on axis x and z.

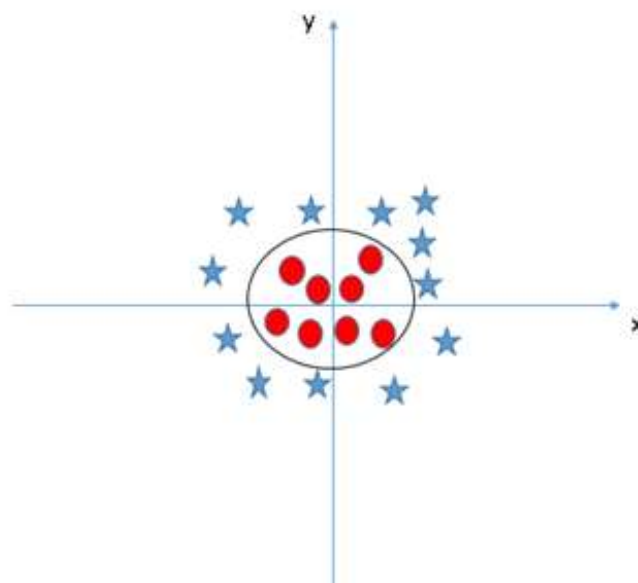


All values for z would be positive always because z is the squared sum of both x and y.

In the original plot, red circles appear close to the origin of x and y axes, leading to lower value of z and star relatively away from the origin result to higher value of z.

In the SVM classifier, it is easy to have a linear hyper-plane between these two classes. But another burning question which arises is, should we need to add this feature manually to have a hyper-plane. Now, the SVM algorithm has a technique called the kernel trick. The SVM kernel is a function that takes low dimensional input space and transforms it to a higher dimensional space i.e. it converts not separable problem to separable problem. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then finds out the process to separate the data based on the labels or outputs you've defined.

When we look at the hyper-plane in original input space it looks like a circle:



#### 4.3.2. Artificial Neural Network:

The term "Artificial neural network" refers to a biologically inspired sub-field of artificial intelligence modeled after the brain. An Artificial neural network is usually a computational network based on biological neural networks that construct the structure of the human brain.

Similar to a human brain has neurons interconnected to each other, artificial neural networks also have neurons that are linked to each other in various layers of the networks. These neurons are known as nodes. Artificial neural network tutorial covers all the aspects related to the artificial neural network.

Artificial neural networks are one of the main tools used in machine learning. As the “neural” part of their name suggests, they are brain-inspired systems which are intended to replicate the way that we humans learn. Neural networks consist of input and output layers, as well as (in most cases) a hidden layer consisting of units that transform the input into something that the output layer can use. They are excellent tools for finding patterns which are far too complex or numerous for a human programmer to extract and teach the machine to recognize.

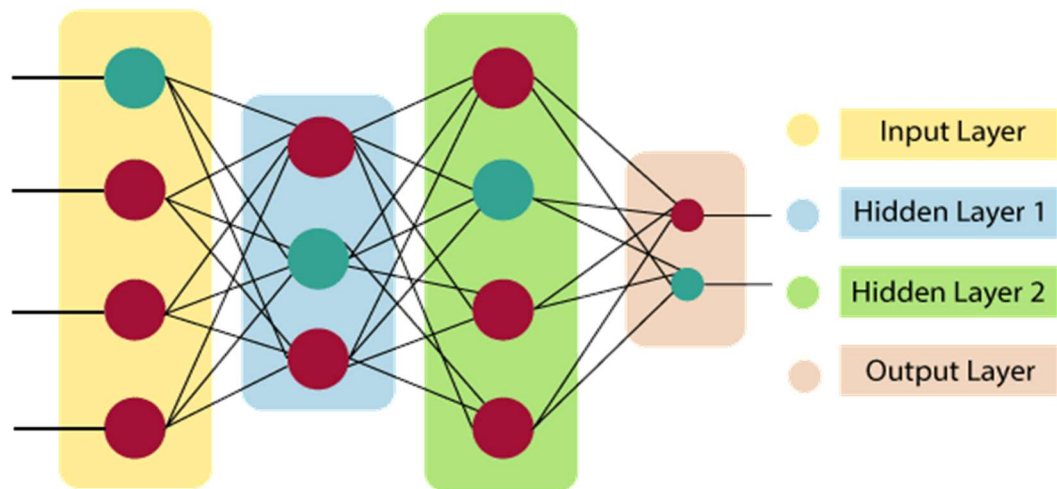
In Computer Science, we model this process by creating “networks” on a computer using matrices. These networks can be understood as abstraction of neurons without all the biological complexities taken into account. To keep things simple, we will just model a simple NN, with two layers capable of solving linear classification problem.

#### **Architecture of Artificial Neural Network: -**

To understand the concept of the architecture of an artificial neural network, we have to understand what a neural network consists of. In order to define a neural network that consists of a large number of artificial neurons, which are termed units arranged in a sequence of layers. Let's us look at various types of layers available in an artificial neural network.

Artificial Neural Network primarily consists of three layers:

1. InputLayer
2. HiddenLayer
3. OutputLayer



### Input Layer:

The input layer accepts inputs in several different formats provided by the programmer.

### Hidden Layer:

The hidden layer presents in-between input and output layers. It performs all the calculations to find hidden features and patterns.

### Output Layer:

The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.

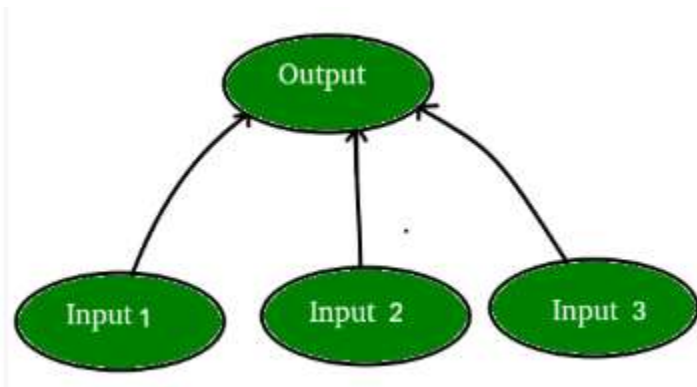
The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias. This computation is represented in the form of a transfer function.

$$\sum_{i=1}^n W_i * X_i + b$$

It determines weighted total is passed as an input to an activation function to produce the output. Activation functions choose whether a node should fire or not.



Only those who are fired make it to the output layer. There are distinctive activation functions available that can be applied upon the sort of task we are performing.



Let's say we have a problem where we want to predict output given a set of inputs and outputs as training example like so:

Input 1	Input 2	Input 3	Output
0	1	1	1
1	0	0	0
1	0	1	1

Fig 2: Training Examples

Now we want to predict the output the following set of inputs:

1	0	1	?
---	---	---	---

Fig 3: Test Example

Note that the output is directly related to third column i.e. the values of input 3 is what the output is in every training example in fig. 2. So, for the test example output value should be 1.

The training process consists of the following steps:

### 1. Forward Propagation:

Take the inputs, multiply by the weights (just use random numbers as weights)

Let  $Y = W_i I_i = W_1 I_1 + W_2 I_2 + W_3 I_3$

Pass the result through a sigmoid formula to calculate the neuron's output.

The Sigmoid function is used to normalize the result between 0 and 1:

$$1 / (1 + e^{-y})$$

## 2. Back Propagation

Calculate the error i.e. the difference between the actual output and the expected output. Depending on the error, adjust the weights by multiplying the error with the input and again with the gradient of the Sigmoid curve:

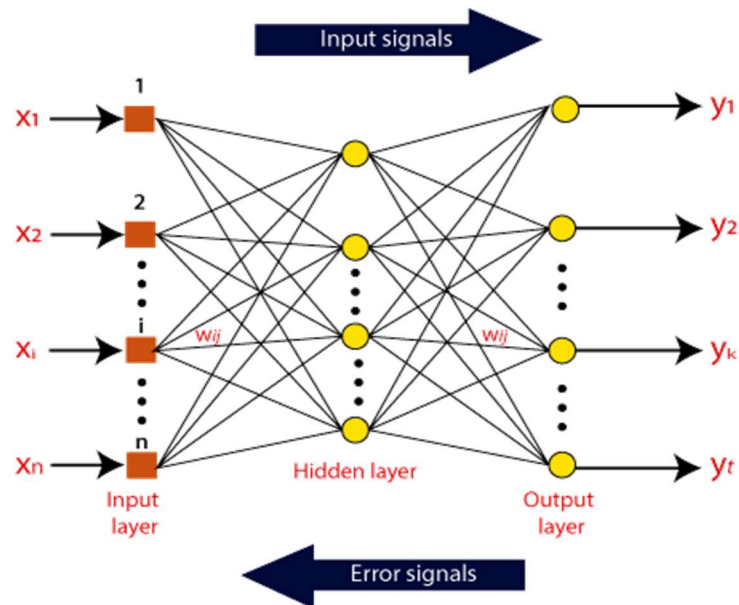
Weight  $\pm$  Error Input Output (1-Output) , here Output (1-Output) is derivative of sigmoid curve.

### How Does Artificial Neural Networks Works?

In this, each arrow represents a connection between two neurons. Also, they used to indicate the pathway for the flow of information. As it was noticed that each connection has a weight, an integer number. That used to control the signal between the two neurons.

If the output is good that was generated by the network, then we don't require to adjust the weights. Although, if poor output generated. Then surely system will alter the weight to improve results.

Artificial Neural Network can be best represented as a weighted directed graph, where the artificial neurons form the nodes. The association between the neurons outputs and neuron inputs can be viewed as the directed edges with weights. The Artificial Neural Network receives the input signal from the external source in the form of a pattern and image in the form of a vector. These inputs are then mathematically assigned by the notations  $x(n)$  for every  $n$  number of inputs.



Afterward, each of the input is multiplied by its corresponding weights (these weights are the details utilized by the artificial neural networks to solve a specific problem). In general terms, these weights normally represent the strength of the interconnection between neurons inside the artificial neural network. All the weighted inputs are summarized inside the computing unit.

If the weighted sum is equal to zero, then bias is added to make the output non-zero or something else to scale up to the system's response. Bias has the same input, and weight equals to 1. Here the total of weighted inputs can be in the range of 0 to positive infinity. Here, to keep the response in the limits of the desired value, a certain maximum value is benchmarked, and the total of weighted inputs is passed through the activation function.

The activation function refers to the set of transfer functions used to achieve the desired output. There is a different kind of the activation function, but primarily either linear or non-linear sets of functions. Some of the commonly used sets of activation functions are the Binary, linear, and Tan hyperbolic sigmoidal activation functions.

**Binary:** In binary activation function, the output is either a one or a 0. Here, to accomplish this, there is a threshold value set up. If the net weighted input of neurons is more than 1, then the final output of the activation function is returned as one or else the output is returned as 0.

### **Sigmoidal Hyperbolic:**

The Sigmoidal Hyperbola function is generally seen as an "S" shaped curve. Here the tan hyperbolic function is used to approximate output from the actual net input. The function is defined as:

$$F(x) = (1/1 + \exp(-x))$$

### **Advantages of Artificial Neural Network: -**

#### **1.Parallel processing capability:**

Artificial neural networks have a numerical value that can perform more than one task simultaneously.

#### **2.Storing data on the entire network:**

Data that is used in traditional programming is stored on the whole network, not on a database. The disappearance of a couple of pieces of data in one place doesn't prevent the network from working.

#### **3.Capability to work with incomplete knowledge:**

After ANN training, the information may produce output even with inadequate data. The loss of performance here relies upon the significance of missing data.

#### **4.Having a memory distribution:**

For ANN is to be able to adapt, it is important to determine the examples and to encourage the network according to the desired output by demonstrating these examples to the network. The succession

of the network is directly proportional to the chosen instances, and if the event can't appear to the network in all its aspects, it can produce false output.

### **5. Having fault tolerance:**

Extortion of one or more cells of ANN does not prohibit it from generating output, and this feature makes the network fault-tolerance.

### **Disadvantages of Artificial Neural Network:**

#### **1. Assurance of proper network structure:**

There is no particular guideline for determining the structure of artificial neural networks. The appropriate network structure is accomplished through experience, trial, and error.

#### **2. Unrecognized behaviour of the network:**

It is the most significant issue of ANN. When ANN produces a testing solution, it does not provide insight concerning why and how. It decreases trust in the network.

#### **3. Hardware dependence:**

Artificial neural networks need processors with parallel processing power, as per their structure. Therefore, the realization of the equipment is dependent.

#### **4. Difficulty of showing the issue to the network:**

ANNs can work with numerical data. Problems must be converted into numerical values before being introduced to ANN. The presentation mechanism to be resolved here will directly impact the performance of the network. It relies on the user's abilities.

#### **5. The duration of the network is unknown:**

The network is reduced to a specific value of the error, and this value does not give us optimum results.

## **4.4 DETAILED DESIGN**

UML is an acronym that stands for **Unified Modeling Language**. Simply put, UML is a modern approach to modeling and documenting software. In fact, it's one of the most popular business process modeling techniques.

It is based on diagrammatic representations of software components. As the old proverb says: “a picture is worth a thousand words”. By using visual representations, we are able to better understand possible flaws or errors in software or business processes.

UML was created as a result of the chaos revolving around software development and documentation. In the 1990s, there were several different ways to represent and document software systems. The need arose for a more unified way to visually represent those systems and as a result, in 1994-1996, the UML was developed by three software engineers working at Rational Software. It was later adopted as the standard in 1997 and has remained the standard ever since, receiving only a few updates.

### **GOALS:**

The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of OO tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns and components.
- Integrate best practices.

### **UML DIAGRAMS**

Unified Modeling Language is a language available to perform modeling of software. A model is simplification of reality. A model provides the blue print of the system, model encompasses detailed plans.

## **Building blocks of the UML**

The vocabulary of the UML encompasses three kinds of building blocks.

1. Things.
2. Relationships.
3. Diagrams.

### **Things in the UML**

Things are the abstractions that are first-class citizen in model. There are four kinds of things in the UML.

1. Structure things.
2. Behavioral things.
3. Grouping things.
4. Annotation things.

These things are the basic object-oriented building blocks of the UML. You use them to write well-formed models.

### **Relationships in the UML**

Things can be connected to logically or physically with the help of relationship in object-oriented modeling. These are four kinds of relationships in the UML.

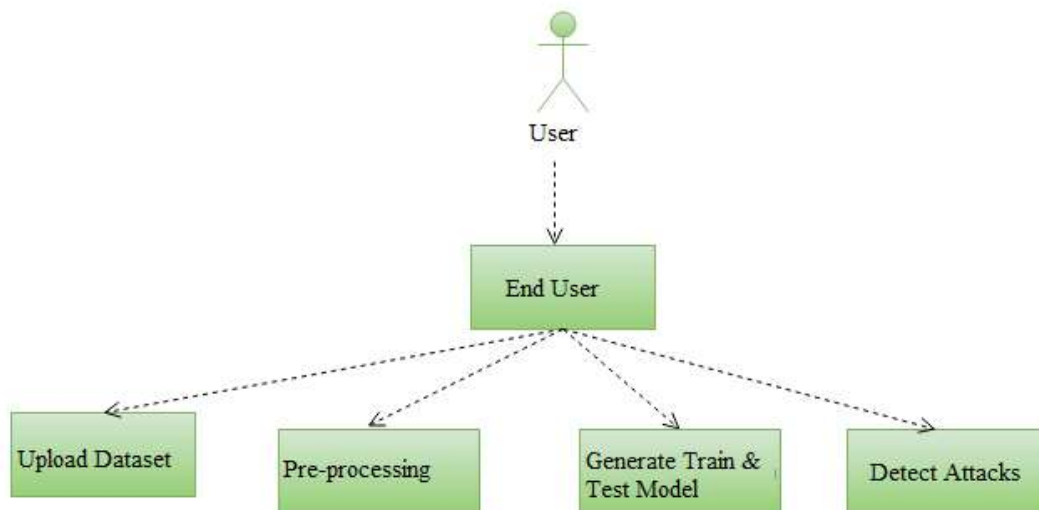
1. Dependency
2. Association.
3. Generalization.
4. Realization.

### i. Use Case Diagram:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.

Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



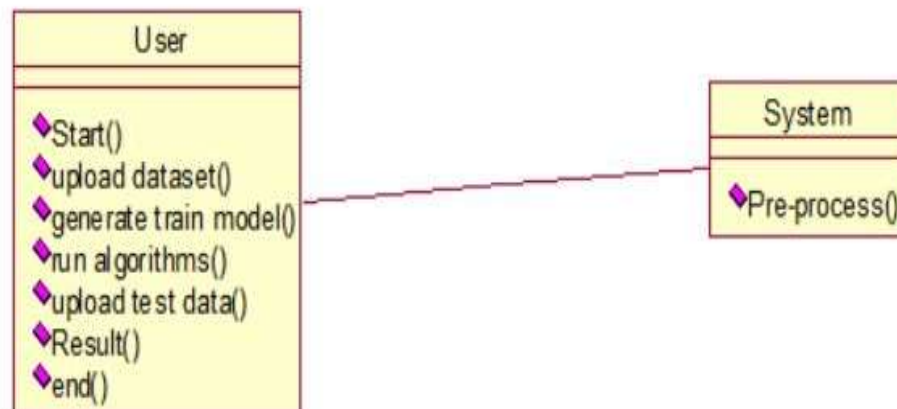


## ii. Class Diagram:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

The class diagram provides a visual representation of the classes in a system and their associations, aggregations, compositions, and inheritance relationships.

These are widely used in software development to design and visualize the structure of object-oriented systems, helping developers understand the system's architecture, relationships, and behavior.

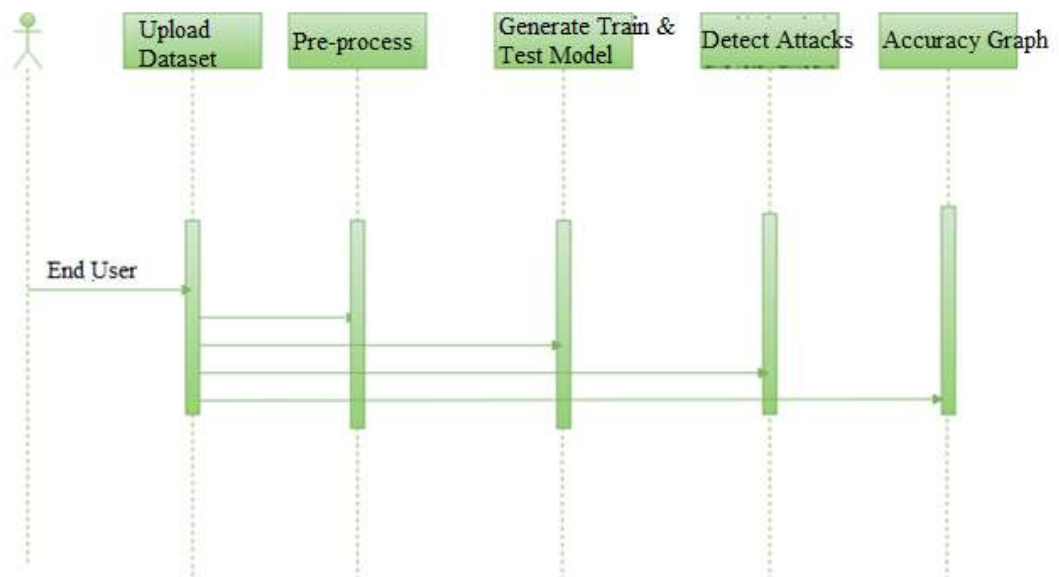


### iii. Sequence Diagram:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order.

It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

It represents the sequence of activities which are going through the project.



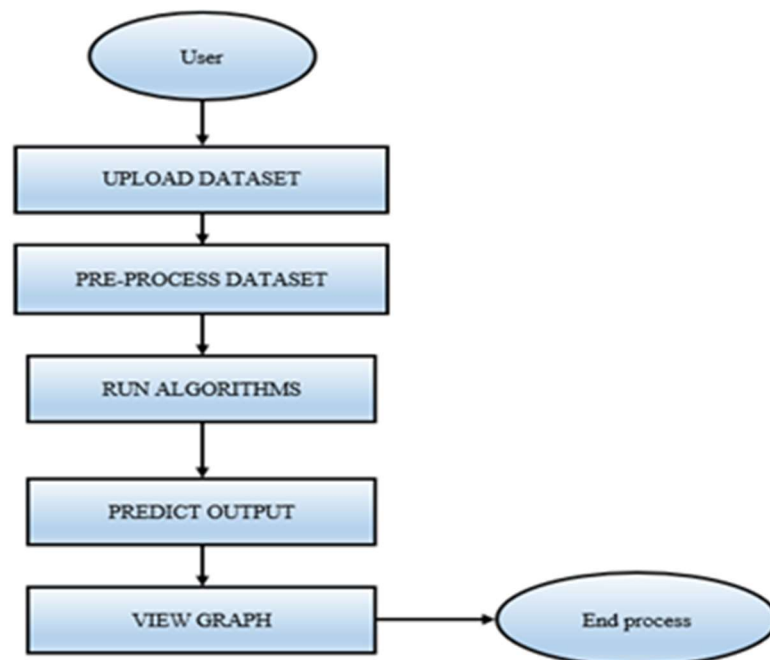
#### iv.Data Flow Diagram:

The Data flow diagram is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

The DFD is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

It is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.



#### v. Component Diagram: -

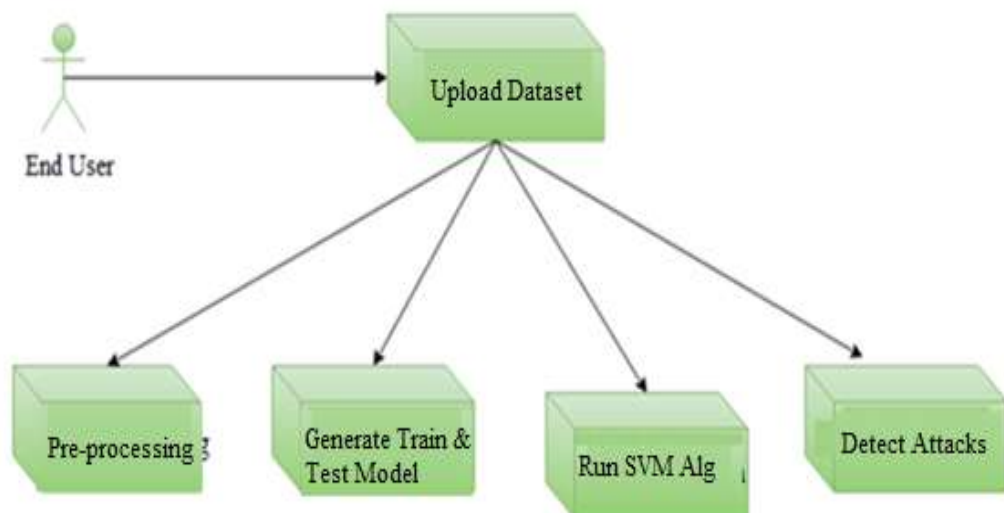
Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities.

Thus from that point of view, component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files, etc.

Component diagrams can also be described as a static implementation view of a system. Static implementation represents the organization of the components at a particular moment.

A single component diagram cannot represent the entire system but a collection of diagrams is used to represent the whole.

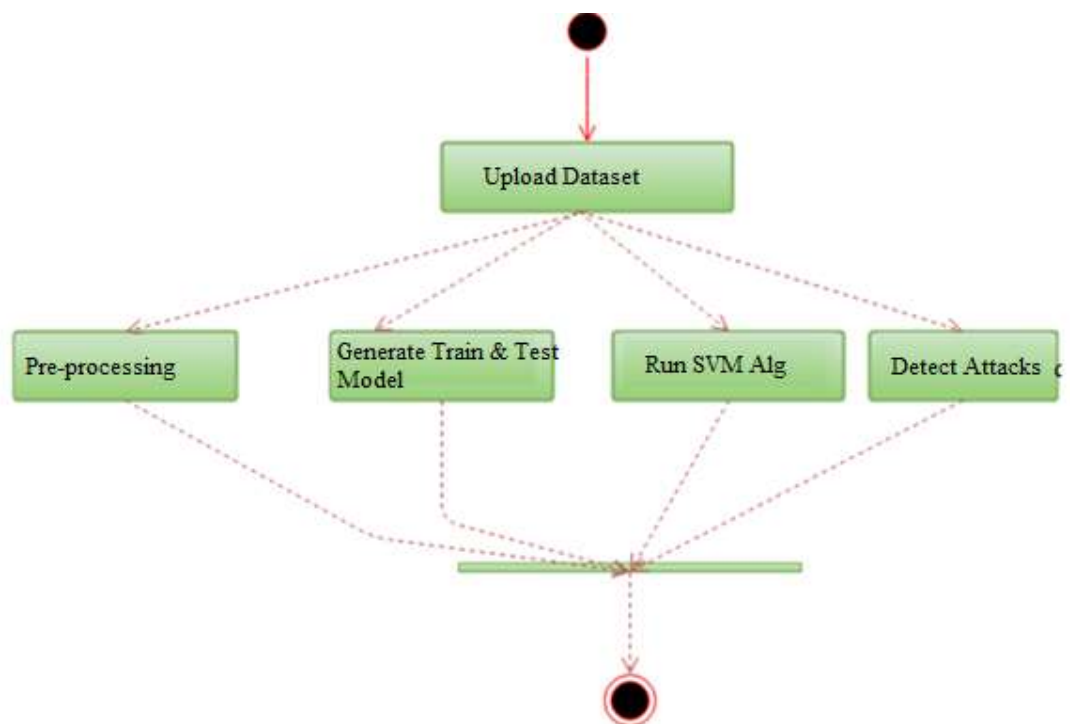
UML Component diagrams are used in modeling the physical aspects of object-oriented systems that are used for visualizing, specifying, and documenting component-based systems and also for constructing executable systems through forward and reverse engineering. Component diagrams are essentially class diagrams that focus on a system's components that often used to model the static implementation view of a system.



#### vi. Activity Diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency.

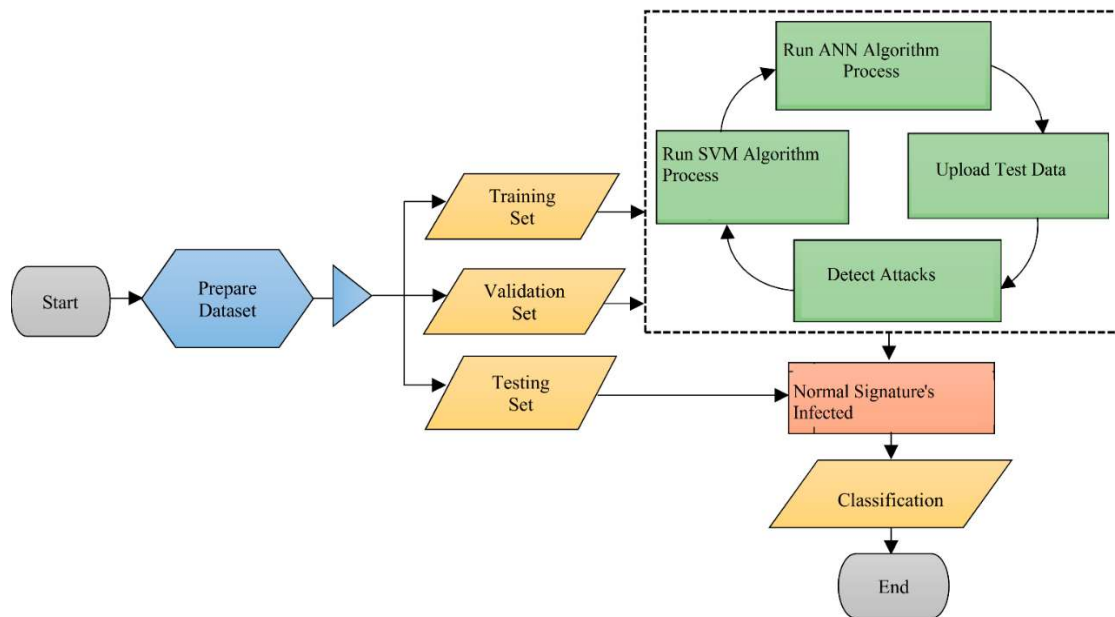
In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



### Vii. State Chart Diagram: -

Statechart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. Statechart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

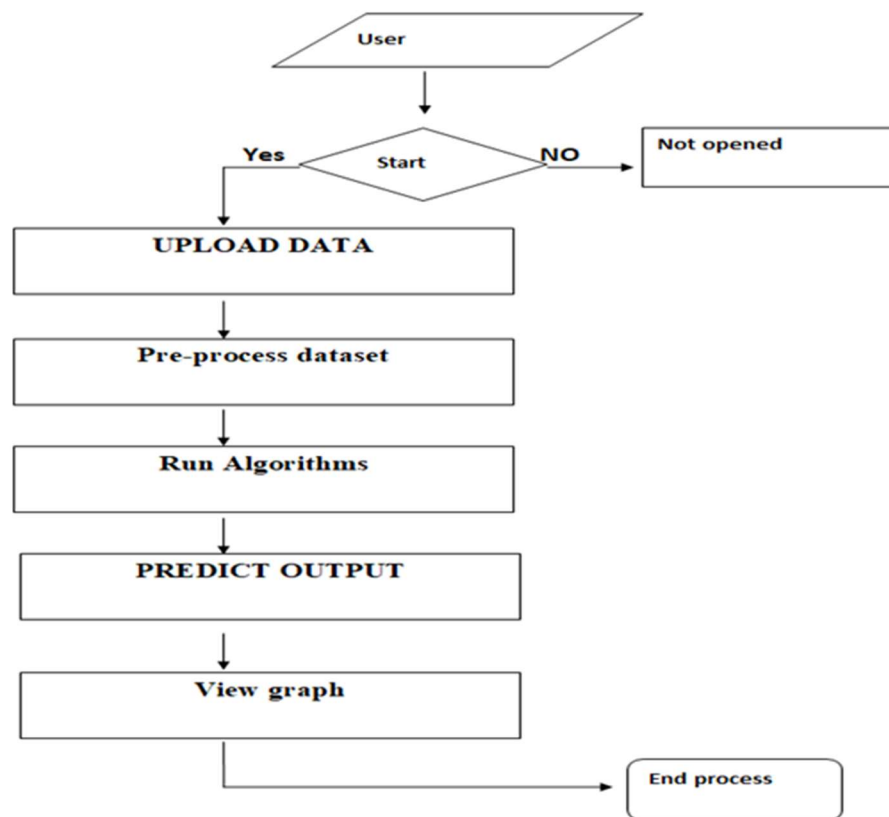
Statechart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of Statechart diagram is to model lifetime of an object from creation to termination.



#### viii.Flow Chart Diagram :-

A flowchart is simply a graphical representation of steps. It shows steps in sequential order and is widely used in presenting the flow of algorithms, workflow or processes.

Typically, a flowchart shows the steps as boxes of various kinds, and their order by connecting them with arrows.



## **CHAPTER 5**

### **IMPLEMENTATION**

#### **5.1 WHAT IS PYTHON: -**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance.

Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Python is currently the most widely used multi-purpose, high-level programming language.

Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.

Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber... etc.

The biggest strength of Python is huge collection of standard libraries which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like OpenCV, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test framework.



### 5.1.1 Advantages of Python: -

#### 1. Extensive Libraries

Python downloads with an extensive library and it contains code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

#### 2. Extensible

As we have seen earlier, Python can be **extended to other languages**. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

#### 3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add **scripting capabilities** to our code in the other language.

#### 4. Improved Productivity

The language's simplicity and extensive libraries render programmers **more productive** than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

#### 5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet of Things. This is a way to connect the language with the real world.

#### 6. Simple and Easy

When working with Java, you may have to create a class to print '**Hello World**'. But in Python, just a print statement will do. It is also quite **easy to learn, understand, and code**. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

## 7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and **indentation is mandatory**. These further aids the readability of the code.

## 8. Object-Oriented

This language supports both the **procedural** and **object-oriented** programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the **encapsulation of data** and functions into one.

## 9. Free and Open-Source

Python is **freely available**. But not only can you **download Python** for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

## 10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to **code only once**, and you can run it anywhere. This is called **Write Once Run Anywhere (WORA)**. However, you need to be careful enough not to include any system-dependent features.

## 11. Interpreted Language

Python is an interpreted language. Since statements are executed one by one, **debugging is easier** than in compiled languages.

## 12. Less Coding

Many of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

### 13. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 GitHub annual survey showed us that Python has overtaken Java in the most popular programming language category.

### 14. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and **machine learning**, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

#### 5.1.2 Disadvantages of Python

##### 1. Speed Limitations

We know that Python code is executed line by line. But since Python is interpreted, it often results in **slow execution**.

This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

##### 2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the **client-side**. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called **Carbannelle**. The reason it is not so famous despite the existence of Bryton is that it isn't that secure.

##### 3. Design Restrictions

We know that, Python is **dynamically-typed**. This means that you don't need to declare the type of variable while writing the code. It uses **duck-typing**. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can **raise run-time errors**.

#### 4. Underdeveloped Database Access Layers

Compared to more widely used technologies like **JDBC (Java Database Connectivity)** and **ODBC (Open Database Connectivity)**, Python's database access layers are a bit underdeveloped.

#### 5. Simple

Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

#### 5.2 HISTORY OF PYTHON: -

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde & Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system.

In an interview with Bill Venners<sup>1</sup>, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it." Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems.

So, I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

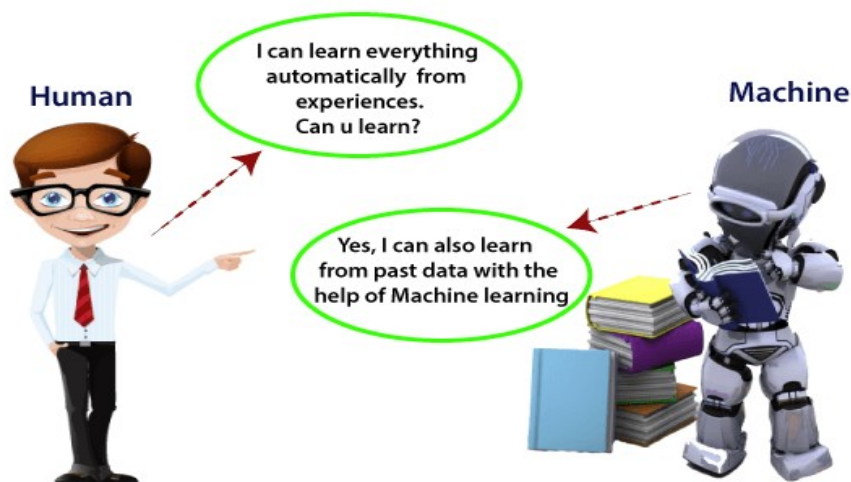
### 5.3 WHAT IS MACHINE LEARNING: -

Machine learning is categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building models of data.

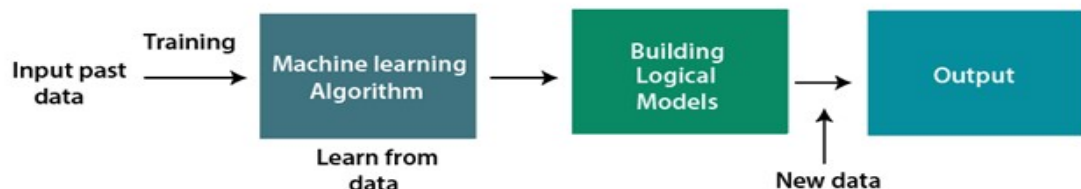
A subset of artificial intelligence known as machine learning focuses primarily on the creation of algorithms that enable a computer to independently learn from data and previous experiences. Arthur Samuel first used the term "machine learning" in 1959. It could be summarized as follows:

Without being explicitly programmed, machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things.

Machine learning algorithms create a mathematical model that, without being explicitly programmed, aids in making predictions or decisions with the assistance of sample historical data, or training data. For the purpose of developing predictive models, machine learning brings together statistics and computer science. Algorithms that learn from historical data are either constructed or utilized in machine learning. The performance will rise in proportion to the quantity of information we provide.



Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models tunable parameters that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively.



### 5.3.1 Categories of Machine Learning: -

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

#### 1. Supervised Learning

Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into classification tasks and regression tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities.

#### 2. Unsupervised Learning

Unsupervised learning involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself."

These models include tasks such as clustering and dimensionality reduction. Clustering algorithms identify distinct groups of data, while dimensionality reduction algorithms search for more succinct representations of the data.

### **5.3.2 Need of Machine Learning: -**

The demand for machine learning is steadily rising. Because it is able to perform tasks that are too complex for a person to directly implement, machine learning is required. Humans are constrained by our inability to manually access vast amounts of data; as a result, we require computer systems, which is where machine learning comes in to simplify our lives.

By providing them with a large amount of data and allowing them to automatically explore the data, build models, and predict the required output, we can train machine learning algorithms. The cost function can be used to determine the amount of data and the machine learning algorithm's performance. We can save both time and money by using machine learning.

The significance of AI can be handily perceived by its utilization's cases. Presently, AI is utilized in self-driving vehicles, digital misrepresentation identification, face acknowledgment, and companion idea by Facebook, and so on. Different top organizations, for example, Netflix and Amazon have constructed AI models that are utilizing an immense measure of information to examine the client interest and suggest item likewise.

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven't surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, "to make decisions, based on data, with efficiency and scale".

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programming logic, in the problems that cannot be programmed inherently. The fact is that we can't do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

Following are some key points which show the importance of Machine Learning:

- Rapid increment in the production of data
- Solving complex problems, which are difficult for a human
- Decision making in various sector including finance
- Finding hidden patterns and extracting useful information from data.

### **5.3.3 Challenges in Machines Learning: -**

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are –

- **Quality of data** – Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.
- **Time-Consuming task** – Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.
- **Lack of specialist persons** – As ML technology is still in its infancy stage, availability of expert resources is a tough job.
- **No clear objective for formulating business problems** – Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.



- **Issue of overfitting & underfitting** – If the model is overfitting or underfitting, it cannot be represented well for the problem.
- **Curse of dimensionality** – Another challenge ML model faces is too many features of data points. This can be a real hindrance.
- **Difficulty in deployment** – Complexity of the ML model makes it quite difficult to be deployed in real life.

#### **5.3.4 Applications of Machines Learning: -**

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach.

Following are some real-world applications of ML –

- Emotion analysis
- Sentiment analysis
- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping

#### **5.3.5 How to Start Learning Machine Learning?**

Arthur Samuel coined the term “**Machine Learning**” in 1959 and defined it as a “**Field of study that gives computers the capability to learn without being explicitly programmed**”.

And that was the beginning of Machine Learning! In modern times, Machine Learning is one of the most popular (if not the most!) career choices. According to Indeed, Machine Learning Engineer is the Best Job of 2019 with a 344% growth and an average base salary of **\$146,085** per year.

But there is still a lot of doubt about what exactly is Machine Learning and how to start learning the machine learning?

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer.

### **Step 1 – Understand the Prerequisites**

In case you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don’t know these, never fear! You don’t need a Ph.D. degree in these topics to get started but you do need a basic understanding.

#### **(a) Learn Linear Algebra and Multivariate Calculus**

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you are more focused on application heavy machine learning, then you will not be that heavily focused on math’s as there are many common libraries available. But if you want to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

## (b) Learn Statistics

Data plays a huge role in Machine Learning. In fact, around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data.

Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking is also a very important part of ML which deals with various concepts like Conditional Probability, Priors, and Posteriors, Maximum Likelihood, etc.

## (c) Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is Python! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as Keras, TensorFlow, Scikit-learn, etc.

So if you want to learn ML, it's best if you learn Python! You can do that using various online resources and courses such as **Fork Python** available Free on Geeks for Geeks.

## Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to actually learning ML. It's best to start with the basics and then move on to the more complicated stuff. Some of the basic concepts in ML are:

### (a) Terminologies of Machine Learning

- **Model** – A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.
- **Feature** – A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.
- **Target (Label)** – A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.
- **Training** – The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.
- **Prediction** – Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

### (b) Types of Machine Learning

- **Supervised Learning** – This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.
- **Unsupervised Learning** – This involves using unlabeled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.
- **Semi-supervised Learning** – This involves using unlabeled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.
- **Reinforcement Learning** – This involves learning optimal actions through trial and error. So, the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

### **5.3.6 Advantages of Machine learning: -**

#### **1. Easily identifies trends and patterns -**

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

#### **2. No human intervention needed (automation)**

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus softwares, they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

#### **3. Continuous Improvement**

As ML algorithms gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

#### **4. Handling multi-dimensional and multi-variety data**

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

#### **5. Wide Applications**

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

### **5.3.7 Disadvantages of Machine Learning: -**

#### **1. Data Acquisition**

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

## **2. Time and Resources**

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

## **3. Interpretation of Results**

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

## **4. High error-susceptibility**

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

## **5.4 PYTHON DEVELOPMENT STEPS: -**

Guido Van Rossum published the first version of Python code (version 0.9.0) at Altisource's in February 1991. This release included already exception handling, functions, and the core data types of lists, dict, str and others.

It was also an object-oriented and had a module-based system. Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced.

This release included list comprehensions, a full garbage collector and it was supporting Unicode. Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x.

The emphasis in Python 3 had been on the removal of duplicate programming

constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it." Some changes in Python 7.3:

- Print is now a function
- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g. a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e. int. long is int as well.
- The division of two integers returns a float instead of an integer. "/" can be used to have the "old" behaviour.
- Text Vs. Data Instead of Unicode Vs. 8-bit

## **5.5 PURPOSE: -**

Python is commonly used for developing websites and software, task automation, data analysis, and data visualization. Since it's relatively easy to learn, Python has been adopted by many non-programmers, such as accountants and scientists, for a variety of everyday tasks, like organizing finances.

It can automate tedious tasks, including checking information in databases, data visualizations, financial analysis and much, much more. Learning Python will allow you to save time throughout your life, and it has the bonus of being one of the easier programming languages to learn!

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

## **Python**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management.

It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python is Interpreted – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

Python is Interactive – you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels.

All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background.

## **5.6 PYTHON FOR MACHINE LEARNING:**

Machine learning (ML) is at the core of the majority of data science tasks. It represents a field of artificial intelligence (AI) concerned with using algorithms to enable machines to learn patterns and trends from historical data to make predictions on unknown data.

Using ML techniques, we can create models that can accurately predict the customer churn rate of the company, estimate the risk of a person having a certain disease, identify the optimal positioning of taxi vehicles, etc.

With Python, we can build an ML model using as few as three lines of code (see an example of such a model for predicting fraudulent bank transactions). Even though behind those few lines of code, there are complex processes and calculations, Python ML libraries do most of the work under the hood, which significantly



facilitates the user's task. The most common libraries are scikit-learn, Keras, TensorFlow, and PyTorch.

## **5.7 MODULES USED IN PROJECT: -**

### **TensorFlow**

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

### **NumPy**

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

### **Pandas**

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem.

Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze.

Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

### **Matplotlib**

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with Python. For the power user, you have full control of line styles, font properties, axes properties etc., via an object-oriented interface or via a set of functions familiar to MATLAB users.

### **Scikit – learn**

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

## **5.8 INSTALL PYTHON STEP-BY-STEP IN WINDOWS AND MAC:**

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

### How to Install Python on Windows and Mac:

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

**Note:** The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a **Windows 64-bit operating system**. So, the steps below are to install python version 3.7.4 on Windows 7 device or to install Python. The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to understand better.

Download the Correct version into the system we need to follow these steps:

**Step 1:** Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: <https://www.python.org>










Now, check for the latest and the correct version for your operating system.

**Step 2:** Click on the Download Tab.



**Step 3:** You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version.

Here, we are downloading the most recent python version for windows 3.7.4.

Looking for a specific release?			
Python releases by version number:			
Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	 Download	<a href="#">Release Notes</a>
Python 3.6.9	July 2, 2019	 Download	<a href="#">Release Notes</a>
Python 3.7.3	March 25, 2019	 Download	<a href="#">Release Notes</a>
Python 3.4.10	March 18, 2019	 Download	<a href="#">Release Notes</a>
Python 3.5.7	March 18, 2019	 Download	<a href="#">Release Notes</a>
Python 2.7.16	March 4, 2019	 Download	<a href="#">Release Notes</a>
Python 2.7.2	Dec. 24, 2018	 Download	<a href="#">Release Notes</a>

**Step 4:** Scroll down the page until you find the Files option.

**Step 5:** Here you see a different version of python along with the operating system.

Files					
Version	Operating System	Description	MD5 Sum	File Size	6PG
Clipped source tarball	Source Release		08111671e5b3db4aef70b0b010f09be	23017663	543
XZ compressed source tarball	Source Release		d33e4aee5097051c2ec45ee56b4803	17131432	543
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.5 and later	6a28b4fa75e3fa71a442c8abce08e6	34999436	543
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5db05c38217a457738f5e4e936243f	28982845	543
Windows help file	Windows		d63999573a2r0602ac50cade0b477cd2	8131763	543
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9600c3c3f0f0c3b0abed319e4d729a2	7804391	543
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	4702b4b0a770abed0c3043a503e563400	20480368	543
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28c51c008bd73a0e53a3b03032b4b02	1362904	543
Windows x86 embeddable zip file	Windows		9fa030d108+1079f0a9412357413b02	6741626	543
Windows x86 executable installer	Windows		33cc0029+2a5+46a306451e76394789	25683848	543
Windows x86 web-based installer	Windows		1b470cfa5d17d982c30983ea371887c	1324608	543

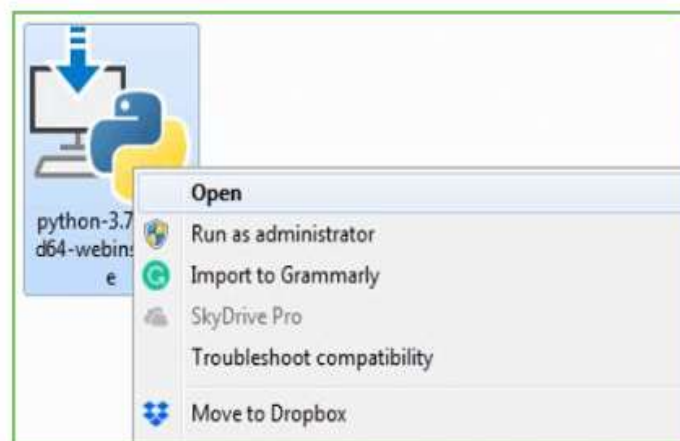
- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

**Note:** To know the changes or updates that are made in the version you can click on the Release Note Option.

## Installation of Python

**Step 1:** Go to Download and Open the downloaded python version to carry out the installation process.



**Step 2:** Before you click on Install Now, make sure to put a tick on Add Python 3.7 to PATH.



**Step 3:** Click on Install Now After the installation is successful. Click on Close.



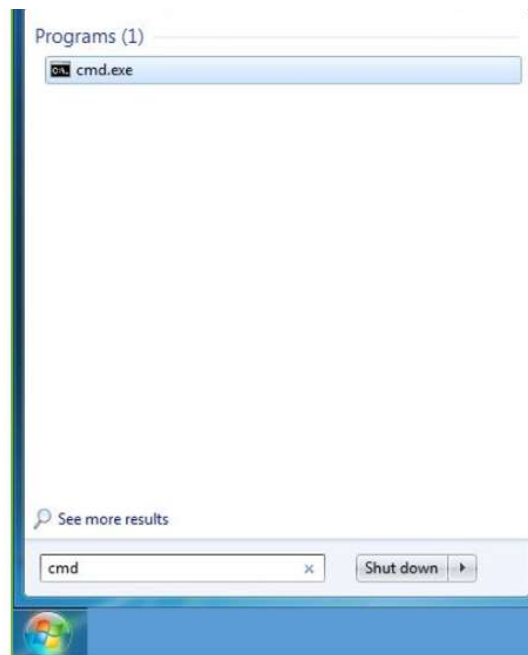
With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

**Note:** The installation process might take a couple of minutes.

### Verify the Python Installation

**Step 1:** Click on Start

**Step 2:** In the Windows Run Command, type “cmd”.



**Step 3:** Open the Command prompt option.

**Step 4:** Let us test whether the python is correctly installed. Type **python -V** and press Enter.

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\DELL>python -V
Python 3.7.4

C:\Users\DELL>_
```

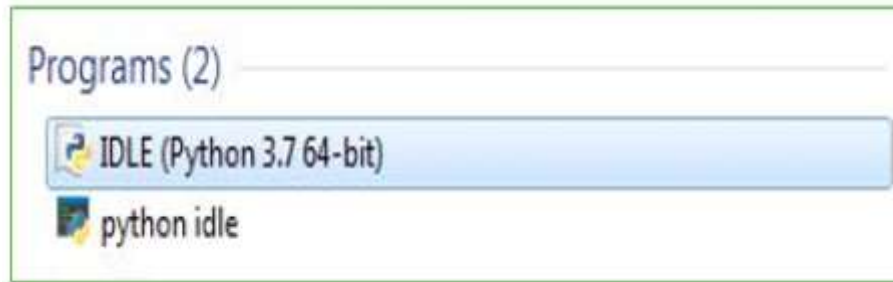
**Step 5:** You will get the answer as 3.7.4

**Note:** If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

**Check how the Python IDLE works:**

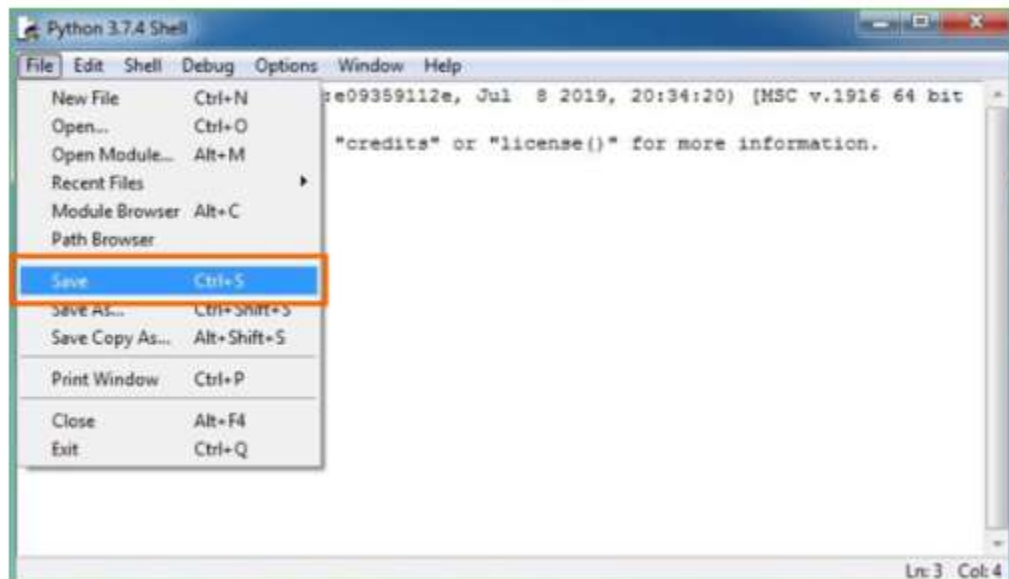
**Step 1:** Click on Start

**Step 2:** In the Windows Run command, type “python idle”.



**Step 3:** Click on IDLE (Python 3.7 64-bit) and launch the program

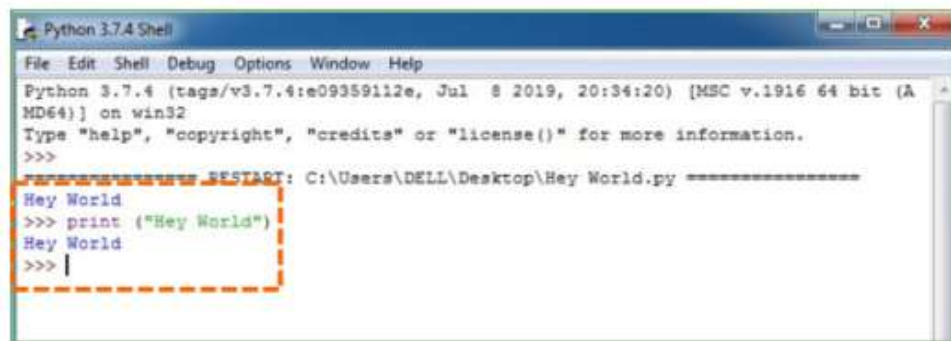
**Step 4:** To go ahead with working in IDLE you must first save the file. **Click on File**  
> **Click on Save**



**Step 5:** Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

**Step 6:** Now for e.g. enter print (“Hey World”) and Press Enter.





```
Python 3.7.4 Shell
File Edit Shell Debug Options Window Help
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 20:34:20) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> python C:\Users\DELL\Desktop\Hey World.py
Hey World
>>> print ("Hey World")
Hey World
>>> |
```

You will see that the command given is launched. With this, we are able to know about how to install Python.

**Note:** Unlike Java, Python doesn't need semicolons at the end of the statements otherwise it won't work.

## 5.9 Coding

```
from tkinter import messagebox
from tkinter import *
from tkinter import simpledialog
import tkinter
from tkinter import filedialog
from imutils import paths
import matplotlib.pyplot as plt
import numpy as np
from tkinter.filedialog import askopenfilename
import numpy as np
import pandas as pd
from sklearn import *
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
from sklearn.feature_selection import SelectFromModel
```

```

from sklearn.linear_model import Lasso

from sklearn.feature_selection import SelectKBest

from sklearn.feature_selection import chi2

from keras.models import Sequential

from keras.layers import Dense


main = tkinter.Tk()

main.title("Network Intrusion Detection")

main.geometry("1300x1200")


global filename

global labels

global columns

global balance_data

global data

global X, Y, X_train, X_test, y_train, y_test

global svm_acc, ann_acc, classifier


def isfloat(value):

    try:

        float(value)

        return True

    except ValueError:

        return False


def splitdataset(balance_data):

    X = balance_data.values[:, 0:38]

    Y = balance_data.values[:, 38]

    print(X)

```

```

print(Y)

X_train, X_test, y_train, y_test = train_test_split(
X, Y, test_size = 0.2, random_state = 0)

return X, Y, X_train, X_test, y_train, y_test

def upload():

    global filename

    text.delete('1.0', END)

    filename = askopenfilename(initialdir = "NSL-KDD-Dataset")

    pathlabel.config(text=filename)

    text.insert(END,"Dataset loaded\n\n")

def preprocess():

    global labels

    global columns

    global filename

    text.delete('1.0', END)

    columns =

["duration","protocol_type","service","flag","src_bytes","dst_bytes","land","wrong_fragment",
"urgent","hot","num_failed_logins","logged_in","num_compromised","root_shell","su_
attempted","num_root","num_file_creations","num_shells","num_access_files","num_outb
ound_cmds","is_host_login","is_guest_login","count","srv_count","error_rate","srv_error
_rate","error_rate","srv_error_rate","same_srv_rate","diff_srv_rate","srv_diff_host_rate","
dst_host_count","dst_host_srv_count","dst_host_same_srv_rate","dst_host_diff_srv_rate","
dst_host_same_src_port_rate","dst_host_srv_diff_host_rate","dst_host_error_rate","dst_ho
st_srv_error_rate","dst_host_error_rate","dst_host_srv_error_rate","label"]

    labels =

{"normal":0,"neptune":1,"warezclient":2,"ipsweep":3,"portsweep":4,"teardrop":5,"nmap":6,
"satan":7,"smurf":8,"pod":9,"back":10,"guess_passwd":11,"ftp_write":12,"multihop":13,"ro

```

```
otkit":14,"buffer_overflow":15,"imap":16,"warezmaster":17,"phf":18,"land":19,"loadmodule":20,"spy":21,"perl":22,"saint":23,"mscan":24,"apache2":25,"snmpgetattack":26,"processable":27,"httptunnel":28,"ps":29,"snmpguess":30,"mailbomb":31,"named":32,"sendmail":33,"xterm":34,"worm":35,"xlock":36,"xsnoop":37,"sqlattack":38,"udpstorm":39}
```

```
balance_data = pd.read_csv(filename)

dataset = ""

index = 0

cols = ""

for index, row in balance_data.iterrows():
```

```
    for i in range(0,42):
```

```
        if(isinstance(row[i],float)):
```

```
            dataset+=str(row[i])+','
```

```
        if index == 0:
```

```
            cols+=columns[i]+','
```

```
    if row[41] == 'normal':
```

```
        dataset+='0'
```

```
    if row[41] == 'anomaly':
```

```
        dataset+='1'
```

```
    if index == 0:
```

```
        cols+='Label'
```

```
    dataset+='\n'
```

```
    index = 1;
```

```
f = open("clean.txt", "w")
```

```
f.write(cols+"\n"+dataset)
```

```
f.close()
```

```
text.insert(END,"Removed non numeric characters from dataset and saved inside clean.txt\n\n")
```

```

text.insert(END,"Dataset Information\n\n")

text.insert(END,dataset+"\n\n")

def generateModel():

    text.delete('1.0', END)

    global X, Y, X_train, X_test, y_train, y_test

    global balance_data

    balance_data = pd.read_csv("clean.txt")

    X, Y, X_train, X_test, y_train, y_test = splitdataset(balance_data)

    text.insert(END,"Train & Test Model Generated\n\n")

    text.insert(END,"Total Dataset Size : "+str(len(balance_data))+"\n")

    text.insert(END,"Split Training Size : "+str(len(X_train))+"\n")

    text.insert(END,"Split Test Size : "+str(len(X_test))+"\n")

def prediction(X_test, cls):

    y_pred = cls.predict(X_test)

    for i in range(len(X_test)):

        print("X=%s, Predicted=%s" % (X_test[i], y_pred[i]))

    return y_pred

# Function to calculate accuracy

def cal_accuracy(y_test, y_pred, details):

    print('y test---',y_test.shape)

    print('y pred---',y_pred.shape)

    accuracy = accuracy_score(y_test,y_pred)*100

    text.insert(END,details+"\n\n")

    text.insert(END,"Accuracy : "+str(accuracy)+"\n\n")

    return accuracy

def runSVM():

    text.delete('1.0', END)

```

```

global svm_acc

global classifier

global X, Y, X_train, X_test, y_train, y_test

total = X_train.shape[1];

X_train1 = SelectKBest(chi2,15).fit_transform(X_train, y_train)

X_test1 = SelectKBest(chi2,15).fit_transform(X_test,y_test)

text.insert(END,"Total Features : "+str(total)+"\n")

text.insert(END,"Features set reduce after applying features selection concept : "+str((total
- X_train.shape[1]))+"\n\n")

cls = svm.SVC(kernel='rbf', class_weight='balanced', probability=True)

cls.fit(X_train, y_train)

text.insert(END,"Prediction Results\n\n")

prediction_data = prediction(X_test, cls)

svm_acc = cal_accuracy(y_test, prediction_data,'SVM Accuracy, Classification Report &
Confusion Matrix')

classifier = cls

def runANN():

    text.delete('1.0', END)

    global ann_acc,classifier1

    global X, Y, X_train, X_test, y_train, y_test

    total = X_train.shape[1];

    X_train = SelectKBest(chi2,38).fit_transform(X_train, y_train)

    X_test = SelectKBest(chi2,38).fit_transform(X_test,y_test)

    text.insert(END,"Total Features : "+str(total)+"\n")

    text.insert(END,"Features set reduce after applying features selection concept : "+str((total
- X_train.shape[1]))+"\n\n")

    model = Sequential()

    model.add(Dense(30, input_dim=38, activation='relu'))

```

```

model.add(Dense(38, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(X_train, y_train, epochs=100, batch_size=32)
_, ann_acc = model.evaluate(X_train, y_train)
ann_acc = ann_acc*100
text.insert(END, "ANN Accuracy : "+str(ann_acc)+"\n\n")
classifier1 = model

def detectAttack():
    text.delete('1.0', END)
    global X, Y, X_train, X_test, y_train, y_test
    filename = filedialog.askopenfilename(initialdir="NSL-KDD-Dataset")
    test = pd.read_csv(filename)
    text.insert(END, filename+" test file loaded\n");
    y_pred = classifier1.predict(test)
    print(y_pred)
    pred = y_pred.tolist()
    for i in range(len(test)):
        if 1.0 in pred[i]:
            text.insert(END, "X=%s, Predicted=%s" % (X_test[i], ' Infected. Detected Anamoly
Signatures')+"\n\n")
        else:
            text.insert(END, "X=%s, Predicted=%s" % (X_test[i], 'Normal Signatures')+"\n\n")

def graph():
    height = [svm_acc, ann_acc]
    bars = ('SVM Accuracy', 'ANN Accuracy')

```

```

y_pos = np.arange(len(bars))

plt.bar(y_pos, height)

plt.xticks(y_pos, bars)

plt.show()

font = ('times', 16, 'bold')

title = Label(main, text='Network Intrusion Detection using Supervised Machine Learning
Technique with Feature Selection')

title.config(bg='PaleGreen2', fg='Khaki4')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 14, 'bold')

upload = Button(main, text="Upload NSL KDD Dataset", command=upload)

upload.place(x=700,y=100)

upload.config(font=font1)


pathlabel = Label(main)

pathlabel.config(bg='DarkOrange1', fg='white')

pathlabel.config(font=font1)

pathlabel.place(x=700,y=150)


preprocess = Button(main, text="Preprocess Dataset", command=preprocess)

preprocess.place(x=700,y=200)

preprocess.config(font=font1)


model = Button(main, text="Generate Training Model", command=generateModel)

model.place(x=700,y=250)

model.config(font=font1)

```



```

runsvm = Button(main, text="Run SVM Algorithm", command=runSVM)
runsvm.place(x=700,y=300)
runsvm.config(font=font1)

annButton = Button(main, text="Run ANN Algorithm", command=runANN)
annButton.place(x=700,y=350)
annButton.config(font=font1)

attackButton = Button(main, text="Upload Test Data & Detect Attack",
command=detectAttack)
attackButton.place(x=700,y=400)
attackButton.config(font=font1)

graphButton = Button(main, text="Accuracy Graph", command=graph)
graphButton.place(x=700,y=450)
graphButton.config(font=font1)

font1 = ('times', 12, 'bold')
text=Text(main,height=30,width=80)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=10,y=100)
text.config(font=font1)

main.config(bg='PeachPuff2')
main.mainloop()

```

## **CHAPTER 6**

### **TEST RESULTS**

The main purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

#### **TYPES OF TESTS**

##### **Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application

.it is done after the completion of an individual unit before integration.

This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

##### **Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields.

Integration tests demonstrate that although the components were individually satisfactory, as shown by successful unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

## **Functional testing**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

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

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

## **System Testing**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

## **White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

## **Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most

other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot –see into it. The test provides inputs and responds to outputs without considering how the software works.

### **6.1 Unit Testing:**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

#### **Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

#### **Test objectives**

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

#### **Features to be tested**

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

### **6.2 Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

### **6.3 Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

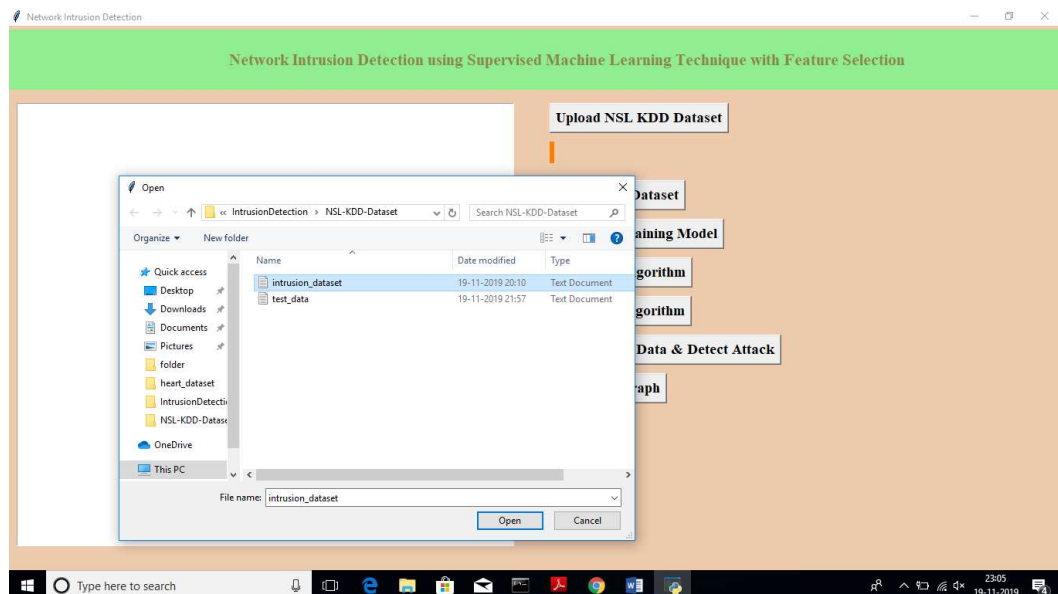
## CHAPTER 7

### RESULTS AND DISCUSSIONS

**HOME PAGE:**



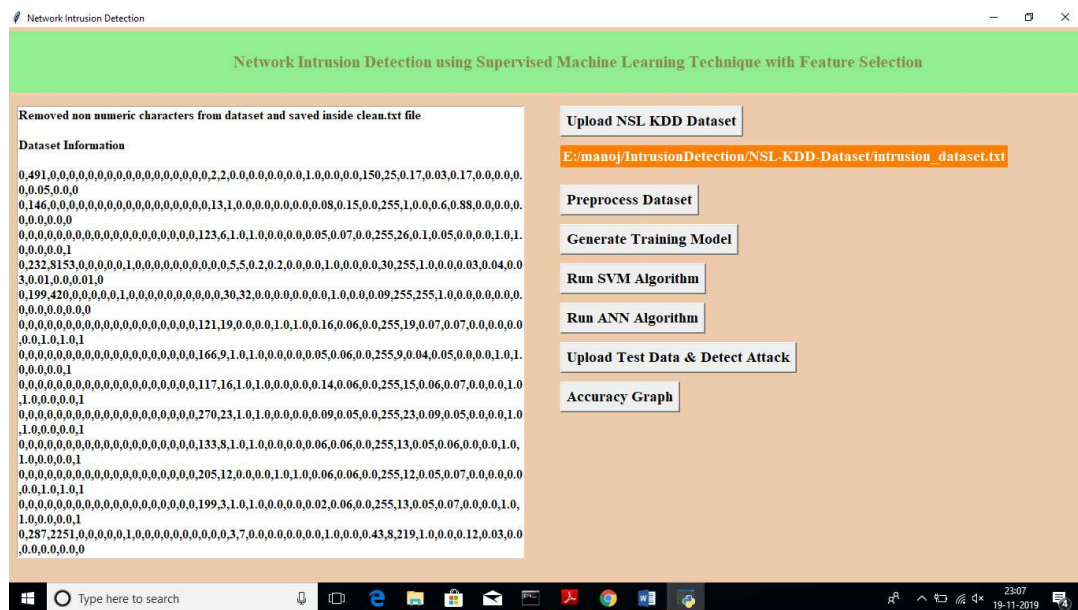
In above screen click on 'Upload NSL KDD Dataset' button and upload dataset.



In above screen I am uploading ‘intrusion\_dataset.txt’ file, after uploading dataset will get below screen.



Now click on ‘Pre-process Dataset’ button to clean dataset to remove string values from dataset and to convert attack names to numeric values.



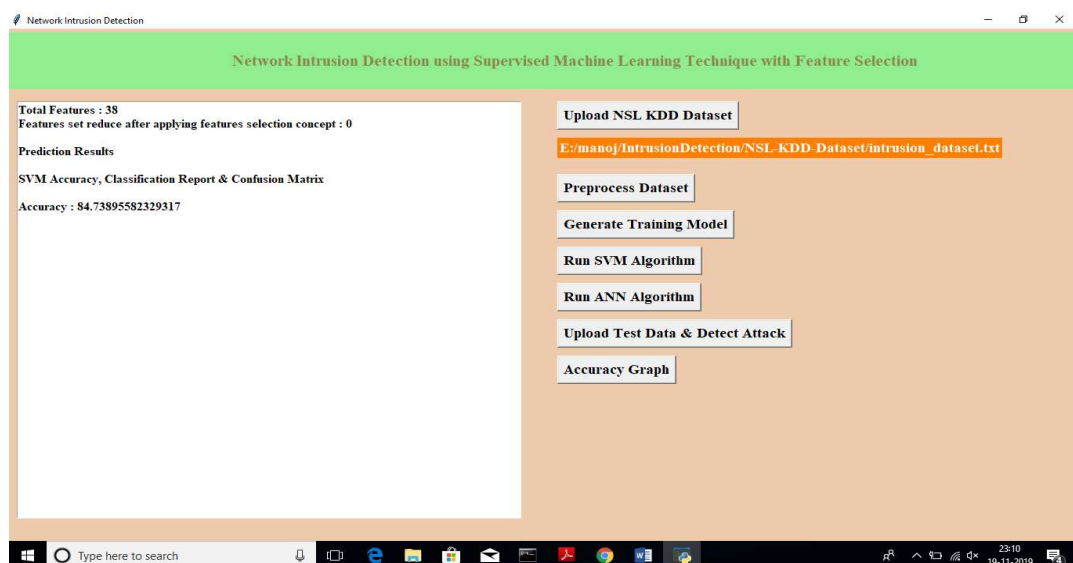
After pre-processing all string values removed and convert string attack names to numeric values such as normal signature contains id 0 and anomaly attack contains signature id 1.

Now click on 'Generate Training Model' to split train and test data to generate model for prediction using SVM and ANN.



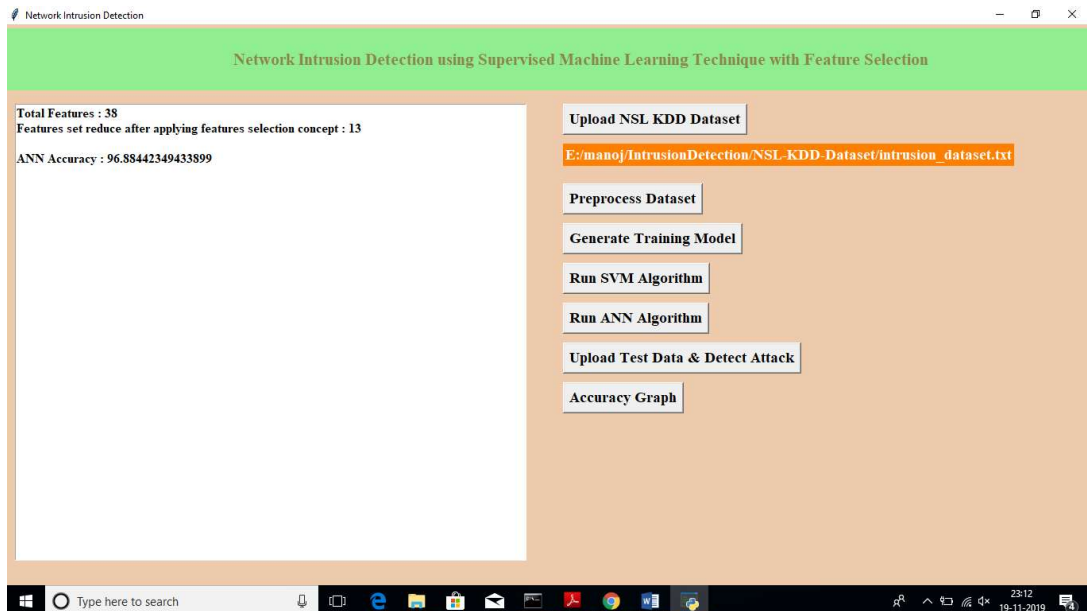
In above screen we can see dataset contains total 1244 records and 995 used for training and 249 used for testing.

Now click on 'Run SVM Algorithm' to generate SVM model and calculate its model accuracy.



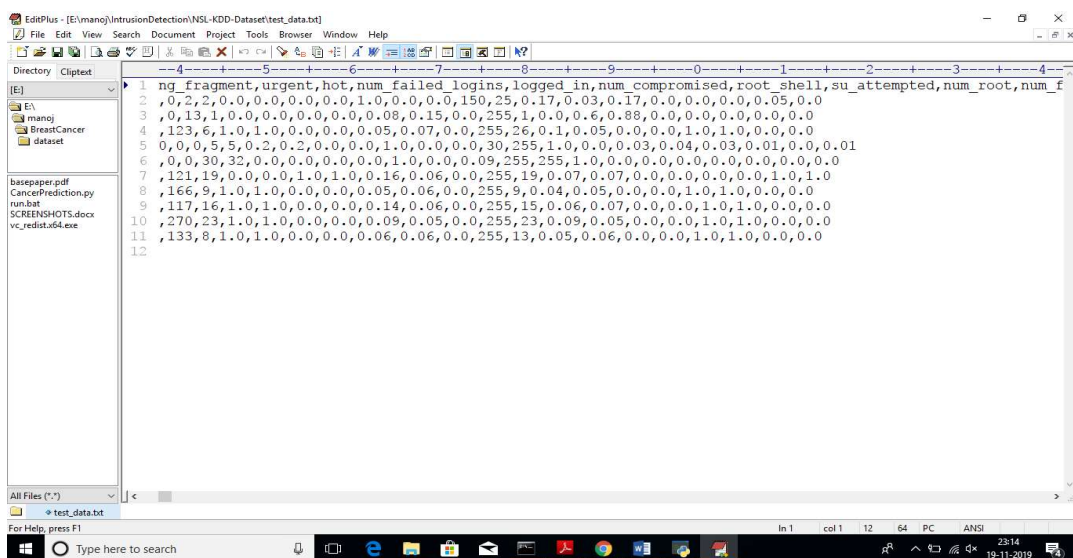


In above screen we can see with SVM we got 84.73% accuracy, now click on 'Run ANN Algorithm' to calculate ANN accuracy.

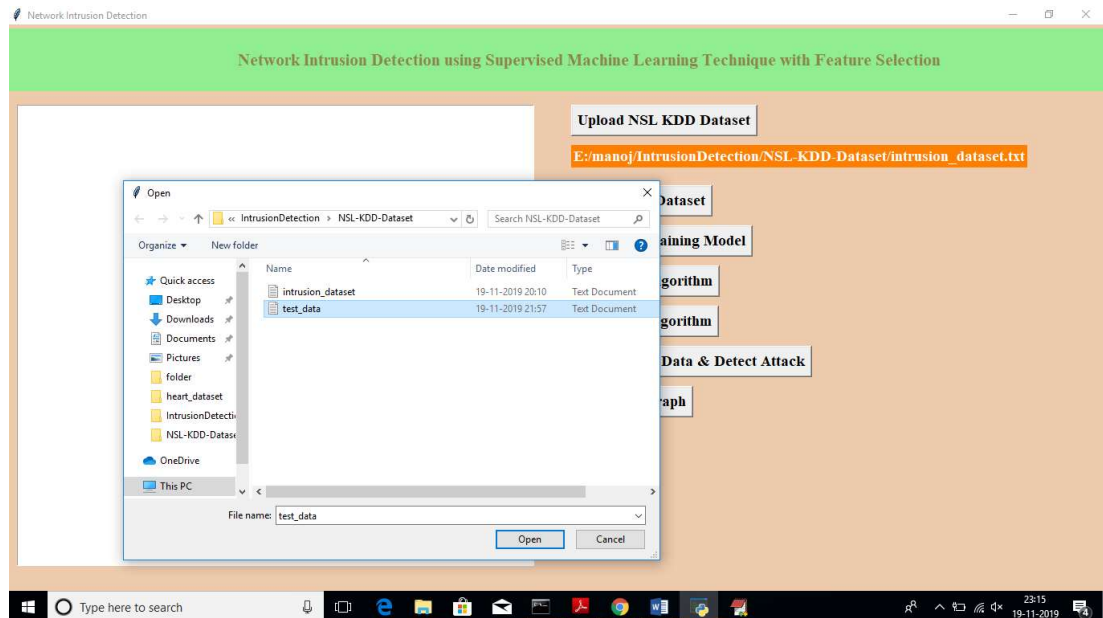


In above screen we got 96.88% accuracy, now we will click on 'Upload Test Data & Detect Attack' button to upload test data and to predict whether test data is normal or contains attack.

All test data has no class either 0 or 1 and application will predict and give us result. See below some records from test data.



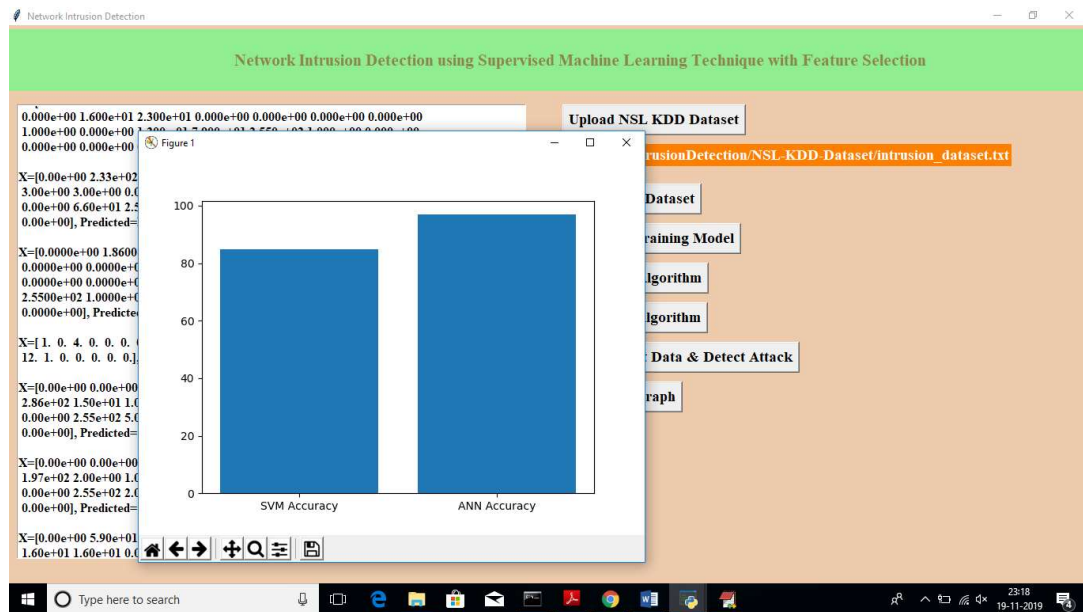
In above test data we don't have either '0' or '1' and application will detect and give us result.



In above screen I am uploading 'test data' file which contains test record, after prediction will get below results.



In above screen for each test data, we got predicted results as 'Normal Signatures' or 'infected' record for each test record. Now click on 'Accuracy Graph' button to see SVM and ANN accuracy comparison in graph format.



From above graph we can see ANN got better accuracy compare to SVM, in above graph x-axis contains algorithm name and y-axis represents accuracy of that algorithms.

## **CHAPTER 8**

### **CONCLUSION & FUTURE WORK**

#### **Conclusion**

In this project, we have presented different machine learning models using different machine learning algorithms and different feature selection methods to find a best model. The analysis of the result shows that the model built using ANN and wrapper feature selection outperformed all other models in classifying network traffic correctly with detection rate of 94.02%.

We believe that these findings will contribute to research further in the domain of building a detection system that can detect known attacks as well as novel attacks. The intrusion detection system exist today can only detect known attacks. Detecting new attacks or zero-day attack still remains a research topic due to the high false positive rate of the existing systems.

#### **Future Work**

Research further in the domain of building a detection system that can detect known attacks as well as novel attacks. The intrusion detection system exist today can only detect known attacks. Detecting new attacks or zero-day attack still remains a research topic due to the high false positive rate of the existing systems.

## **CHAPTER 9**

### **REFERENCES**

- H. Song, M. J. Lynch, and J. K. Cochran, “A macro-social exploratory analysis of the rate of interstate cyber-victimization,” *American Journal of Criminal Justice*, vol. 41, no. 3, pp. 583–601, 2016.
- P. Alaei and F. Noorbehbahani, “Incremental anomaly-based intrusion detection system using limited labeled data,” in *Web Research (ICWR), 2017 3th International Conference on*, 2017, pp. 178–184.
- M. Saber, S. Chadli, M. Emharraf, and I. El Farissi, “Modeling and implementation approach to evaluate the intrusion detection system,” in *International Conference on Networked Systems*, 2015, pp. 513–517.
- M. Tavallaee, N. Stakhanova, and A. A. Ghorbani, “Toward credible evaluation of anomaly-based intrusion-detection methods,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 5, pp. 516–524, 2010.
- A. S. Ashoor and S. Gore, “Importance of intrusion detection system (IDS),” *International Journal of Scientific and Engineering Research*, vol. 2, no. 1, pp. 1–4, 2011.
- M. Zamani and M. Movahedi, “Machine learning techniques for intrusion detection,” *arXiv preprint arXiv:1312.2177*, 2013.
- N. Chakraborty, “Intrusion detection system and intrusion prevention system: A comparative study,” *International Journal of Computing and Business Research (IJCBR) ISSN (Online)*, pp. 2229–6166, 201