A Mini Project Report

**On**

“**MACHINE LEARNING ALGORITHMS-BASED PREDICTION OF BOTNET ATTACK FOR IOT DEVICES**”

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

**Bachelor of Technology In**

**Computer Science & Cyber Security**

**By**

**B. JEEVAN NAIK - 21R21A6208**

**K. SUJITH REDDY-21R21A6229**

**M. KIRAN -21R21A6233**

**P. SAHASRA-21R21A6242**

**S. HARI PRIYA – 21R21A6247**

**SAHIL JADHAV-21R21A6248**

Under the guidance of

**Ms. Tejaswini**

**Assistant Professor**



**2023-2024**

**Department of Computer Science & Cyber Security**

i



## Department of Computer Science & Cyber Security

**CERTIFICATE**

This is to certify that the project entitled “**MACHINE LEARNING ALGORITHMS-BASED PREDICTION OF BOTNET ATTACK FOR IOT DEVICES**” has been Submitted by

**B. Jeevan Naik(21R21A6208), K. Sujith Reddy(21R21A6229), M. Kiran**

**Narasimha(21R21A6233), P. Sahasra(21R21A6242), S. Haripriya(21R21a6247), Sahil Jadhav(21R21A6248)** in partial fulfillment of the requirements for the award of degree of Bachelor of Technology in Computer Science and Engineering(Cyber Security) from

MLR Institute of Technology, Hyderabad. The results embodied in this project have not been submitted to any other University or Institution for the award of any degree or diploma.

**Internal Guide Head of the Department**

**External Examiner**

ii



## Department of Computer Science & Cyber Security

**DECLARATION**

We hereby declare that the project entitled “ Machine Learning -Based Prediction of Botnet attacks in Iot devices” is the work done during the period from August 2023 to January 2024 and is submitted in partial fulfillment of the requirements for the award of degree of Bachelor of Technology in Computer Science and Cyber Security from MLR Institute of Technology, Hyderabad.

#### B. Jeevan 21R21A6208

**K. Sujith Reddy 21R21A6229**

#### M. Kiran 21R21A6233

#### P. Sahasra 21R21A6242

#### S. Haripriya 21R21A6247

#### Sahil Jadhav 21R21A6248

#### 

iii



## Department of Computer Science & Cyber Security

**ACKNOWLEDGEMENT**

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that we now have the opportunity to express our gratitude for all of them.

First of all we would like to express our deep gratitude towards our Coordinator, **Ms. Tejaswini,** for her support in the completion of our dissertation. We wish to express our sincere thanks to **Dr. M.Chiranjeevi,** HOD, Dept. of CSC and also principal,

**Dr. K. SRINIVAS RAO**, for providing the facilities to complete the dissertation.

We would like to thank all our faculty and friends for their help and constructive criticism during the project period. Finally, we are very much indebted to our parents for their moral support and encouragement to achieve our goals.

#### B. Jeevan Naik 21R21A6208

**K. Sujith Reddy 21R21A6229**

#### M. Kiran 21R21A6233

#### P. Sahasra 21R21A6242

#### S. Haripriya 21R21A6247

#### Sahil Jadhav 21R21A6248

iv



**Department of Computer Science & Cyber Security**

# ABSTRACT

With the increasing use of smart devices, there's a growing risk of cyber threats, particularly from botnet attacks. In this study, we propose a solution that uses machine learning to predict and prevent such attacks on Internet of Things (IoT) devices. We collect data on how these devices communicate, their system logs, and network traffic. By teaching machine learning models to recognize patterns in normal and malicious behavior, our system can identify potential threats in real-time. Through testing with diverse datasets, our approach proves effective in accurately spotting and preventing botnet attacks on IoT devices. This research aims to strengthen IoT security and ensure the safe evolution of smart technologies. This research contributes to the ongoing efforts to enhance IoT security by providing a proactive defense mechanism against botnet threats. The integration of machine learning-based prediction systems into IoT environments can aid in fortifying these systems, mitigating potential risks, and ensuring the continued growth and adoption of IoT technologies.

v

## INDEX

|  |  |
| --- | --- |
| **TITLE** | **PAGENO.** |
| **CHAPTER 1 INTRODUCTION** | **1-2** |
| 1.1 About The Project |  |
| 1.2 Challenging Issues |  |
| 1.3 Our Contributions |  |
| **CHAPTER 2 SYSTEM ANALYSIS** | **3-8** |
| 2.1 Need for The System |  |
| 2.2 Existing System Disadvantages |  |
| 2.2.1 Random Forest Algorithm |  |
| 2.2.2 K-Nearest Neighbor(KNN) |  |
| 2.3 Proposed System Advantages |  |
| 2.3.1 Pre-Processing |  |
| 2.3.2 Splitting The Dataset Into The Training Set And Test Set |  |
| 2.3.3 Support Vector Machine Algorithm |  |
| **CHAPTER 3 SYSTEM STUDY** | **09-10** |
| 3.1 Economic Feasibility |  |
| 3.2 Technical Feasibility |  |
| 3.3 Social Feasibility |  |
| **CHAPTER 4**  **SYSTEM REQUIREMENTS AND SPECIFICATIONS** | **11-16** |
| 4.1 Functional Requirements |  |
| 4.2 Non-Functional Requirements |  |

|  |  |  |
| --- | --- | --- |
| * 1. Hardware Requirements   Software Requirements |  |  |
| **CHAPTER 5**  **TECHNICAL DISCRIPTION**   * 1. Technology Use   2. Domain Description   **CHAPTER 6**   1. **SYSTEM ARCHITECTURE**    1. UML Diagrams       1. Use case Diagram       2. Activity Diagram       3. Component Diagram       4. Sequence Diagram       5. Class Based Modelling       6. Collaboration Diagram   **CHAPTER 7 CODING**  **CHAPTER 8 SYSTEM TESTING**  **CHAPTER 9 SCREENSHOTS**   * 1. Pycharm Application   2. Uploading Dataset   3. Inserting Dataset      1. Selecting File | **17-32**  **33-38**    **39-41**  **42-43**  **44-50** |  |

|  |  |
| --- | --- |
| 9.3.2 Selecting Dataset   * 1. Jupyter Notebook   2. Non-Malicious Dataset   3. Malicious Dataset   4. Description Of Non-Malicious Dataset   5. Description Of Malicious Dataset   6. Adding Label To The Dataset   7. Combining Non-Malicious Dataset And Malicious Dataset   8. Normalization   9. Prediction And Accuracy Score   10. SVM Confusion Matrix   11. RF Confusion Matrix   **CHAPTER 10 CONCLUSION**  **CHAPTER 11**  **FUTURE SCOPE** | **51**  **52** |
| **CHAPTER 12**  **REFERENCES** | **53-54** |

**Chapter 1**

**Introduction**

* 1. **About The Project**

The general idea of the Internet of Things (IoT) is to allow for communication between human-to-thing or thing-to-thing(s). Things denote sensors or devices, whilst human or an object is an entity that can request or deliver a service [1]. The interconnection amongst the entities is always complex. IoT is broadly acceptable and implemented in various domains, such as healthcare, smart home, and agriculture. However, IoT has a resource constraint and heterogeneous environments, such as low computational power and memory. These constraints create problems in providing and implementing a security solution in IoT devices. These constraints further escalate the existing challenges for IoT environment. Therefore, various kinds of attacks are possible due to the vulnerability of IoT devices.

Many studies are trying to protect against these botnet attacks on the IoT environment. However, there are many gaps still existing to develop an effective detection mechanism. An intrusion detection system (IDS) is one of the efficient ways to deal with attacks. However, the traditional IDSs are often not able to be deployed for the IoT environments due to the resource constraint problem of these devices. The complex cryptographic mechanisms cannot be embedded in many IoT devices either for the same reason. There are mainly two kinds of IDSs: the anomaly and misuse approaches. The misuse-based, also called the signature-based, approach, is based on the attacks’ signatures, and they can also be found in most public IDSs, specifically Suricata [4]. Formally, the attacker can easily circumvent the signature-based approaches, and these mechanisms cannot guarantee to detect the unknown attacks and the variances of known attacks. The anomaly-based systems are based on normal data and can support to identify the unknown attacks. However, the different nature of IoT devices is being faced with the difficulty of collecting common normal data. The machine learning-based detection can guarantee detection of not only the known attacks and their variances. Therefore, we proposed a machine learning-based botnet attack detection architecture. We also adopted a feature selection method to reduce the demand for processing resources for performing the detection system on resource constraint devices. The experiment results indicate that the detection accuracy of our proposed system is high enough to detect the botnet attacks. Moreover, it can support the extension for detecting the new distinct kinds of attacks.

1

* 1. **Challenging Issues**

The traditional attack detection systems cannot be competently relocated in the IoT environments because of the different nature of such devices, and the diverse architecture of the underlying network methodologies with the conventional network. Additionally, the possible attacks can be distinct from the attacks that are found on the traditional network devices. The heavyweight encryption methods cannot be deployed on these resource constraint devices. On the other side, the IoT devices become very cheap to set up for personal usages, like in small business and smart home appliances. The attackers were launching the attacks to the victim nodes after infecting the botnets on these devices. They can also circumvent formal rule-based detection systems. Although the machine learning-based system can detect the variances of the many kinds of attacks, the new distinct kinds of attacks can be launched sometimes. Additionally, the complex processing of ML classifiers is a challenge to implement the lightweight attack detection system on the resource constraint devices.

* 1. **Our Contributions**

In this study, our main contributions are as follows.

1. A botnet attacks detection framework with sequential architecture based on machine learning (ML) algorithms is proposed for dealing with attacks in IoT environments.
2. A correlated-feature selection approach is adopted for reducing the irrelevant features, which makes the system lightweight.

(3) In our proposal, classifiers based on different ML algorithms may be applied in different attack detection sub-engines, which leads to better detection performance and shorter processing times and a lightweight implementation.

2

**Chapter 2**

**System Analysis**

**2.1. Need For the System**

The field of cybersecurity is always a challenging task for researchers. As a result, cybercriminals constantly research new approaches to identify weaknesses and use them for nefarious and illegal purposes. Malware spreading technique is now growing with new and innovative manners. The malware is then used to carry out further attacks like data exfiltration and denial of service attacks utilizing or on compromised machines

Internet of Things (IoT) services and applications have significantly increased due to their functionality and ease of use. Companies have started to develop a variety of Internet of Things (IoT)-based products, ranging from modest personal gadgets like a smartwatch to an entire network of smart grid, smart mining, smart manufacturing, and autonomous driverless vehicles. The overwhelming quantity and ubiquitous presence have enticed potential hackers for data theft and cyberattacks. One of the most significant issues with the Internet of Things is security. This study's main objective is to suggest a novel machine learning algorithm-based model for detecting and thwarting botnet attacks on IoT networks

**2.2. Existing System Disadvantages**

**2.2.1 Random Forest Algorithm**

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

3

**Assumptions for Random Forest**

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

• There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.

• The predictions from each tree must have very low correlations.

Below are some points that explain why we should use the Random Forest algorithm

* + - It takes less training time as compared to other algorithms.
    - It predicts output with high accuracy, even for the large dataset it runs efficiently.
    - It can also maintain accuracy when a large proportion of data is missing.

**Disadvantages of random forest**

* + - Increased accuracy requires more trees.
    - More trees slow down model.
    - Can't describe relationships within data.

**2.2.2 K-Nearest Neighbor (KNN)**

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique. K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.

It stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into

4

a well suite category by using K- NN algorithm. It can be used for Regression as well as for Classification but mostly it is used for the Classification problems. It is a non-parametric algorithm, which means it does not make any assumption on underlying data. It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

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

**Why do we need a K-NN Algorithm?**

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:

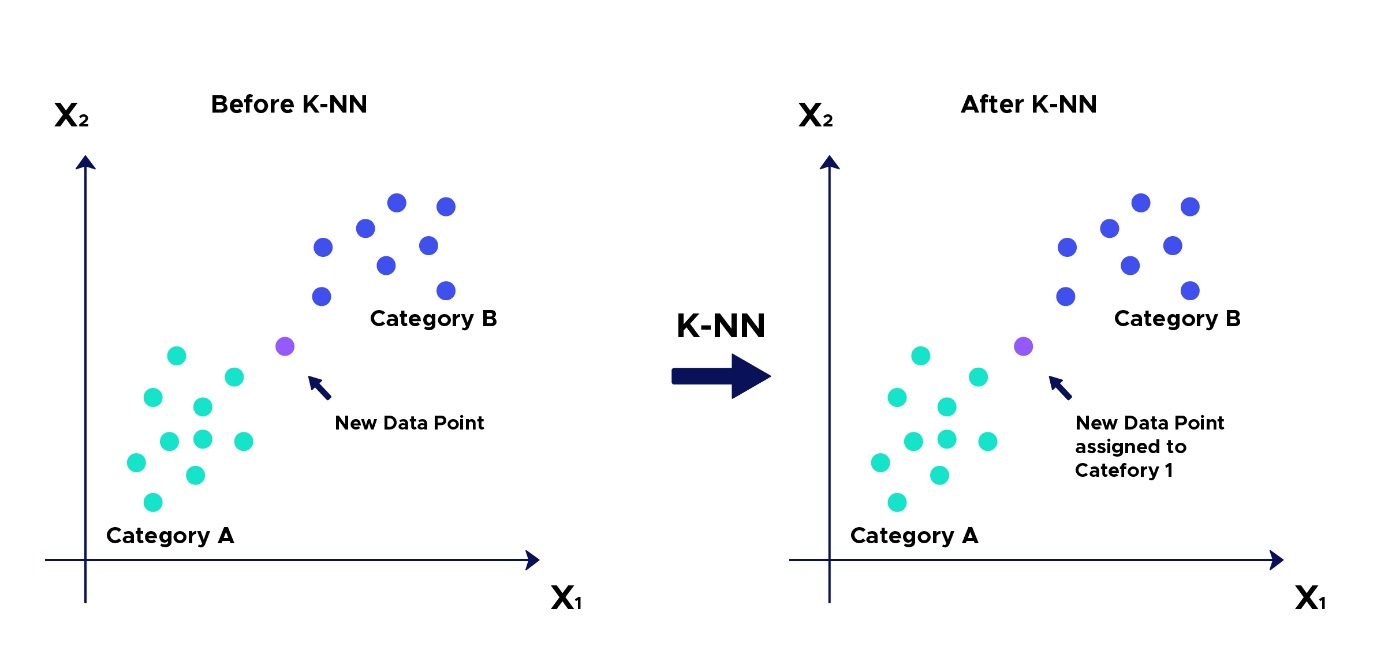


Fig. 3.4: KNN on dataset.

**How to select the value of K in the K-NN Algorithm?**

Below are some points to remember while selecting the value of K in the K-NN algorithm:

* + There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.

5

* + A very low value for K such as K=1 or K=2, can be noisy and lead to the effects of outliers in the model
  + Large values for K are good, but it may find some difficulties.

**Disadvantages of KNN Algorithm**

* + Always needs to determine the value of K which may be complex some time.
  + The computation cost is high because of calculating the distance between the data points for all the training samples.

**2.3. Proposed System Advantages**

**2.3.1 Pre-processing**

Data pre-processing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When creating a project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way.So, for this, we use data pre-processing task.

**Why do we need Data Pre-processing?**

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

* + Getting the dataset
  + Importing libraries
  + Importing datasets
  + Finding Missing Data
  + Encoding Categorical Data
  + Splitting dataset into training and test set
  + Feature scaling

6

**2.3.2 Splitting the Dataset into the Training set and Test set**

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

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

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



Fig. 4.2: Dataset splitting.

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

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

**2.3.3 Support Vector Machine Algorithm**

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

7

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.

**Advantages of SVM**

* Support vector machine works comparably well when there is an understandable margin of dissociation between classes.
* It is more productive in high-dimensional spaces.
* It is effective in instances where the number of dimensions is larger than the number of specimens.
* Support vector machine is comparably memory systematic. Support Vector Machine (SVM) is a powerful supervised machine learning algorithm with several advantages.

Some of the main advantages of SVM include:

* Handling high-dimensional data: SVMs are effective in handling high-dimensional data, which is common in many applications such as image and text classification.

8

**CHAPTER 3**

**SYSTEM STUDY**

**3.1 Economic Feasibility**

Economic feasibility refers to the ability of a project or business venture to generate enough revenue to cover its costs and provide a reasonable return on investment. It involves analyzing the costs and benefits of a project, including the costs of materials, labor, and equipment, as well as the projected revenue from sales or other sources of income. Economic feasibility is an important consideration when determining whether a project or venture should be undertaken, and it is often used in conjunction with other types of feasibility analysis, such as technical feasibility and operational feasibility. Economic feasibility is a kind of cost-benefit analysis of the examined project, which assesses whether it is possible to implement it. This term means the assessment and analysis of a project's potential to support the decision-making process by objectively and rationally identifying its strengths, weaknesses, opportunities and risks associated with it, the resources that will be needed to implement the project, and an assessment of its chances of success. It consists of market analysis, economic analysis, technical and strategic analysis.

**3.2 Technical Feasibility**

Technical feasibility is a standard practice for companies to conduct feasibility studies before commencing work on a project. Businesses undertake a technical feasibility study to assess the practicality and viability of a product or service before launching it. Whether you are working as a product engineer, product designer or team manager, there may be plenty of situations in your career where you are required to prepare a technical feasibility study. In this article, we discuss what is technical feasibility, explain how to conduct one and share tips on writing a feasibility study report. A technical feasibility study helps organisations determine whether they have the technical resources to convert the idea into a fully functional and profitable working system. It helps in troubleshooting the project before commencing work. The study identifies potential challenges and uncovers ways to flowchart for how products and services evolve before they reach the overcome them. It also helps in long-term planning, as it can serve as a market.

9

**3.3 Social Feasibility**

The effect that a proposed project may have on the social system in the project environment is addressed in the social feasibility. It may happen that a particular category of employees may be short or not available as a result of ambient social structure.

The influence on the social status of the participants by the project should be evaluated in order to guarantee compatibility. It must be identified that employees in particular industries may have specific status symbols within the society.

10

**CHAPTER 4**

**SYSTEM REQUIREMENTS AND SPECIFICATION**

**4.1 Functional Requirements**

**Output Design**

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provides a permanent copy of the results for later consultation. The various types of outputs in general are:

* + External Outputs, whose destination is outside the organization.
  + Internal Outputs whose destination is within organization, and they are the user’s main interface with the computer.
  + Operational outputs whose use is purely within the computer department.
  + Interface outputs, which involve the user in communicating directly.

**Output Definition**

The outputs should be defined in terms of the following points:

* Type of the output
* Content of the output
* Format of the output
* Location of the output
* Frequency of the output
* Volume of the output
* Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

11

**Input Design**

Input design is a part of overall system design. The main objective during the input design is as given below:

* To produce a cost-effective method of input.
* To achieve the highest possible level of accuracy.
* To ensure that the input is acceptable and understood by the user.

**Input Stages**

The main input stages can be listed as below:

* Data recording
* Data transcription
* Data conversion
* Data verification
* Data control   
  Data transmission
* Data validation
* Data correction

**Input Types**

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

* External inputs, which are prime inputs for the system.
* Internal inputs, which are user communications with the system.
* Operational, which are computer department’s communications to the system?
* Interactive, which are inputs entered during a dialogue.

**Input Media**

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to:

12

* Type of input
* Flexibility of format
* Speed
* Accuracy
* Verification methods
* Rejection rates
* Ease of correction
* Storage and handling requirements
* Security
* Easy to use
* Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive.

As Input data is to be the directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

**Error Avoidance**

At this stage care is to be taken to ensure that input data remains accurate form the stage at which it is recorded up to the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

**Error Detection**

Even though every effort is make to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data.

13

**Data Validation**

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the user to commit errors. The system will not accept invalid data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary. The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with popup menus.

**User Interface Design**

It is essential to consult the system users and discuss their needs while designing the user interface:

**User Interface Systems Can Be Broadly Classified As**:

* User initiated interface the user is in charge, controlling the progress of the user/computer dialogue. In the computer-initiated interface, the computer selects the next stage in the interaction.
* Computer initiated interfaces

In the computer-initiated interfaces the computer guides the progress of the user/computer dialogue. Information is displayed and the user response of the computer takes action or displays further information.

User Initiated Interfaces

User initiated interfaces fall into two approximate classes:

* Command driven interfaces: In this type of interface the user inputs commands or queries which are interpreted by the computer.
* Forms oriented interface: The user calls up an image of the form to his/her screen and fills in the form. The forms-oriented interface is chosen because it is the best choice.

**Computer-Initiated Interfaces**

The following computer – initiated interfaces were used:

* The menu system for the user is presented with a list of alternatives and the user chooses one; of alternatives.

14

* Questions – answer type dialog system where the computer asks question and takes action based on the basis of the users reply.

Right from the start the system is going to be menu driven, the opening menu displays the available options. Choosing one option gives another popup menu with more options. In this way every option leads the users to data entry form where the user can key in the data.

**Error Message Design**

The design of error messages is an important part of the user interface design. As user is bound to commit some errors or other while designing a system the system should be designed to be helpful by providing the user with information regarding the error he/she has committed.

This application must be able to produce output at different modules for different inputs.

Performance Requirements

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

* The system should be able to interface with the existing system • The system should be accurate
* The system should be better than the existing system
* The existing system is completely dependent on the user to perform all the duties.

15

**4.2 Non Functional Requirements**

The following are the non-functional requirements of the system:

* The system shall be a web-based application that can provide all the functions over the internet.
* The system shall deliver messages in the same order it gets sent out.
* The system shall guarantee the delivery. And shall notice sender if message is not delivered successfully.
* The system shall be scalable and robust

**4.3 Hardware Requirements**

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

* Operating system : Windows, Linux
* Processor : minimum intel i3
* Ram : minimum 4 GB
* Hard disk : minimum 250GB

**4.4 Software Requirements**

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation.

The appropriation of requirements and implementation constraints gives the general overview of the project in regard to what the areas of strength and deficit are and how to tackle them.

* Python IDLE 3.7 version (or)
* Anaconda 3.7 (or)
* Jupiter

16

**CHAPTER 5**

**TECHNICAL DISCRIPTION**

**5.1 Technology Use**

**Machine Learning**

**What is Machine Learning**

Before we look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often 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.

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, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

**Categories of Machine Learning**

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised 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. We will see examples of both types of supervised learning in the following section.

17

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. We will see examples of both types of unsupervised learning in the following section.

**Need for Machine Learning**

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 have not 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 programing 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.

**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 –

18

1. 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.
2. Time-Consuming task − Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.
3. Lack of specialist persons − As ML technology is still in its infancy stage, availability of expert resources is a tough job.
4. 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.
5. Issue of overfitting & underfitting − If the model is overfitting or underfitting, it cannot be represented well for the problem.
6. Curse of dimensionality − Another challenge ML model faces is too many features of data points. This can be a real hindrance.
7. Difficulty in deployment − Complexity of the ML model makes it quite difficult to be deployed in real life.

**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

19

* Fraud detection
* Fraud prevention
* Recommendation of products to customer in online shopping

**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 it? So, this article deals with the Basics of Machine Learning and also the path you can follow to eventually become a full-fledged Machine Learning Engineer. Now let’s get started!!!

**How to start learning ML?**

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer. Of course, you can always modify the steps according to your needs to reach your desired end-goal!

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

20

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

maths 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. So it is no surprise that you need to learn it!!! 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 GeeksforGeeks.

Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to actually learning ML (Which is the fun part!!!) 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:

1. **Terminologies of Machine Learning**

21

* 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).

1. **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 unlabelled 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 unlabelled 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.

**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.

22

1. **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.

1. **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 multivariety, and they can do this in dynamic or uncertain environments.

1. **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.

**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.

23

**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.

* 1. **Domain Description**

**What is Python?**

Below are some facts about Python.

* 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 library 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 frameworks

Multimedia

Advantages of Python

Let’s see how Python dominates over other languages.

24

**1. Extensive Libraries**

Python downloads with an extensive library and it contain 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.

1. **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.

1. **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.

1. **Readable**

25

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**

Like we said earlier, 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**

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

**Advantages of Python Over Other Languages**

**1. Less Coding**

Almost all 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.

**2. Affordable**

26

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.

**3. 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.

**Disadvantages of Python**

So far, we’ve seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let’s now see the downsides of choosing Python over another language.

**1. Speed Limitations**

We have seen 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 clientside. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called Carbonnelle.

The reason it is not so famous despite the existence of Brython is that it isn’t that secure.

**3. Design Restrictions**

As you know, 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.

27

**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. Consequently, it is less often applied in huge enterprises.

**5. Simple**

No, we’re not kidding. 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.

This was all about the Advantages and Disadvantages of Python Programming Language.

**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 Venners1, 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 en 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."

28

**Python Development Steps**

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module 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

29

**Purpose**

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 - without breaking.

**Modules Used in Project**

30

**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 datatypes 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.

31

**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 IPython. 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. 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 - without breaking.

32

**CHAPTER 6**

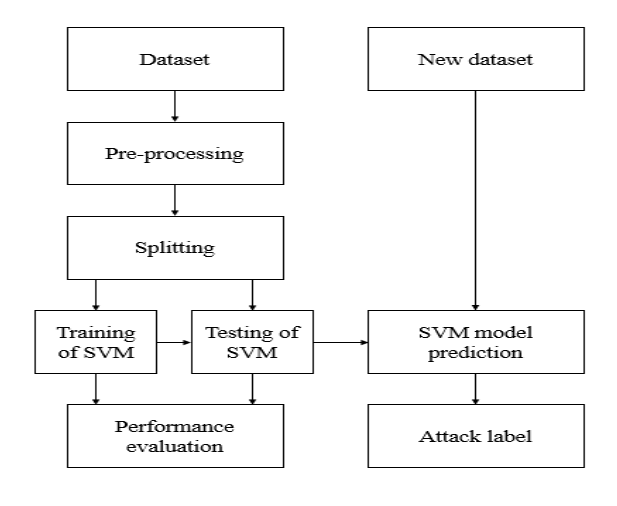
**6.1 SYSTEM ARCHITECTURE**

Fig. 6.1: Block diagram of proposed system.

33

**6.1.1UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

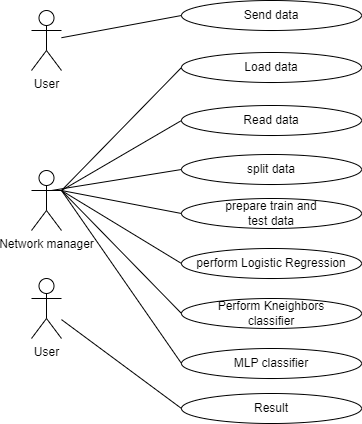
**GOALS:** The Primary goals in the design of the UML are as follows:

* 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.

**6.1.2 Use Case Diagram**

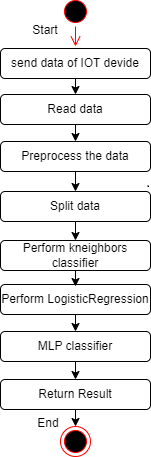
A use case diagram in the Unified Modeling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies 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.

34



### **6.1.3 Activity diagram**

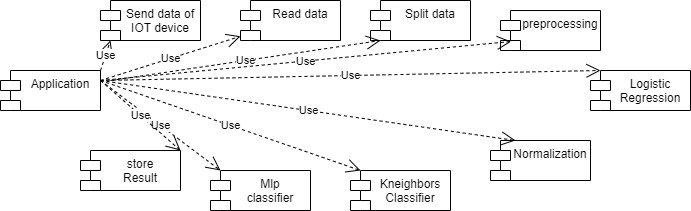
Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.



35

### **6.1.4 Component diagram**

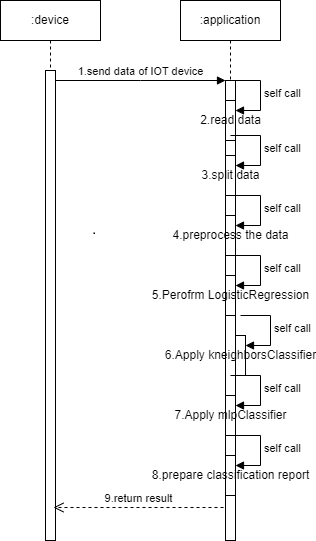
Component diagram describes the organization and wiring of the physical components in a system.



### **6.1.5 Sequence Diagram**

Represent the objects participating in the interaction horizontally and time vertically. A Use Case is a kind of behavioral classifier that represents a declaration of an offered behavior. Each use case specifies some behavior, possibly including variants that the subject can perform in collaboration with one or more actors. Use cases define the offered behavior of the subject without reference to its internal structure. These behaviors, involving interactions between the actor and the subject, may result in changes to the state of the subject and communications with its environment. A use case can include possible variations of its basic behavior, including exceptional behavior and error handling.

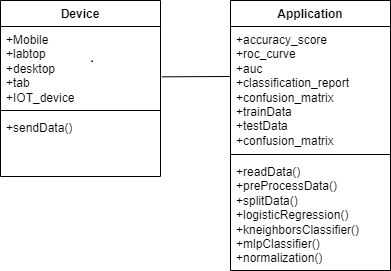
36



### **6.1.6 Class Diagram**

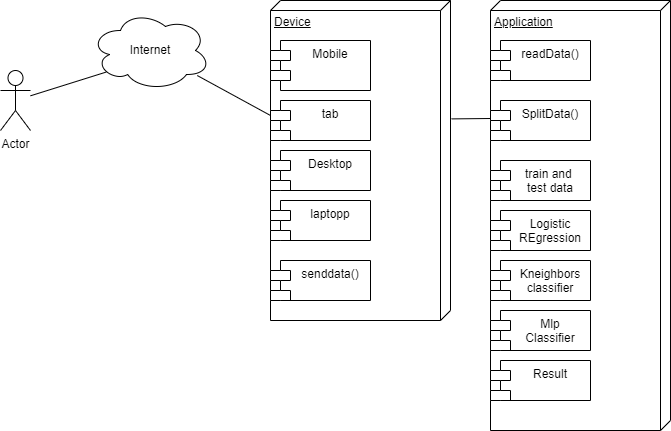
The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an "is-a" or "has-a" relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed "methods" of the class. Apart from this, each class may have certain "attributes" that uniquely identify the class.

37



### **6.1.7 Deployment diagram**

The deployment diagram visualizes the physical hardware on which the software will be deployed.



38

# CHAPTER 7 CODING

### **Importing** **all** **the** **required** **libraries**

**import** pandas **as** pd **import** numpy **as** np **import** itertools

**import** matplotlib.pyplot **as** plt

**import** vpython **as** vs

**from** sklearn.preprocessing **import** MinMaxScaler

**from** IPython.display **import** display

**from** sklearn.model\_selection **import** train\_test\_split **from** sklearn.linear\_model **import** LogisticRegression **from** sklearn.svm **import** SVC

**from** sklearn.ensemble **import** RandomForestClassifier **from** sklearn.neighbors **import** KNeighborsClassifier **from** sklearn.neural\_network **import** MLPClassifier

**from** sklearn.metrics **import** accuracy\_score,roc\_curve,auc,classification\_report,confusion\_matrix

**Loading dataset**

pureData **=** pd.read\_csv("C:\Users\sabba\OneDrive\Desktop\Python\benign\_traffic.csv")

print("Non-malicious Datasets") display(pureData.head())

maliciousDataset **=** pd.read\_csv("C:\Users\sabba\OneDrive\Desktop\Python\junk.csv")

print("Malicious Dataset") display(maliciousDataset.head())

### **Description** **of** **dataset**

print("There are %d records with %d features in non-malicious dataset"**%**(pureData.shape[0],pure Data.shape[1]))

display(pureData.describe())

print("There are %d records with %d features in malicious dataset"**%**(maliciousDataset.shape[0], maliciousDataset.shape[1]))

display(maliciousDataset.describe())

**Adding label to the dataset**

print("Adding output column in the datasets with all 0 in pureData dataset and all 1 in maliciousData set dataset")

pureData["output"] **=** 0

maliciousDataset["output"] **=** 1

### **Combining** **Non-malicious and Malicious datasets**

dataset **=** pd.concat([pureData, maliciousDataset], axis**=**0)

print("There are %d records with %d features in combined dataset"**%**(dataset.shape[0],dataset.sha pe[1]))

39

**Splitting input and output**

Output **=** dataset.output

Input **=** dataset.loc[:,"MI\_dir\_L5\_weight":"HpHp\_L0.01\_pcc"] print("Output shape :-",Output.shape)

print("Intput shape :-",Input.shape)

### **Preprocessing**

Output**=**np.array(Output).flatten()

**Normalization**

print("Calculation Z-score normalization which converts all indicators to a common scale with an av erage of zero and standard deviation of one. The average of zero means that it avoids introducing ag gregation distortions stemming from differences in indicators' means.") datasetNormalised**=**(dataset**-**dataset.mean())**/**(dataset.std()) datasetNormalised\_array**=**np.array(datasetNormalised)

print("Data after normalisation") display(datasetNormalised.head())

### **Split dataset (80-20)**

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(datasetNormalised\_array, Output, test\_size **=** 0.2, rand om\_state **=** 3)

print ("Training set has {} samples.".format(X\_train.shape[0])) print ("Testing set has {} samples.".format(X\_test.shape[0]))

**Model definition**

SVM **=** SVC(probability**=True**)

RF **=** RandomForestClassifier(n\_estimators**=**1500)

### **Model training and prediction**

**def** train\_predict(model, X\_train, y\_train, X\_test, y\_test): model.fit(X\_train, y\_train)

y\_pred **=** model.predict(X\_test)

acc\_test **=** accuracy\_score(y\_test,y\_pred) report **=** classification\_report(y\_test,y\_pred)

**return** acc\_test,report

**Prediction and accuracy score**

SVM\_acc,SVM\_report **=** train\_predict(SVM,X\_train,y\_train,X\_test,y\_test) print("The accuracy score of SVM is %f"**%**SVM\_acc) print("Classification report :-")

print(SVM\_report)

RF\_acc,RF\_report **=** train\_predict(RF,X\_train,y\_train,X\_test,y\_test) print("The accuracy score of Random Forest Classifier is %f"**%**RF\_acc) print("Classification report :-")

print(RF\_report)

40

### **Confusion matrix**

**def** plot\_confusion\_matrix(cm, classes,normalize**=False**,title**=**'Confusion matrix',cmap**=**plt.cm.Blues)

:

plt.imshow(cm, interpolation='nearest', cmap=cmap)

plt.title(title)

plt.colorbar()

tick\_marks = np.arange(len(classes))

plt.xticks(tick\_marks, classes)

plt.yticks(tick\_marks, classes)

thresh = cm.max() / 2.

for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):

plt.text(j, i, cm[i, j],

horizontalalignment="center",

color="white" if cm[i, j] > thresh else "black")

plt.tight\_layout()

plt.ylabel('True label')

plt.xlabel('Predicted label')

def plot\_confusion\_matrix(cm, classes, normalize=False, title='Confusion matrix', cmap=plt.cm.Blues):

plt.imshow(cm, interpolation='nearest', cmap=cmap)

plt.title(title)

plt.colorbar()

tick\_marks = np.arange(len(classes))

plt.xticks(tick\_marks, classes)

plt.yticks(tick\_marks, classes)

thresh = cm.max() / 2.

for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):

plt.text(j, i, cm[i, j],

horizontalalignment="center",

color="white" if cm[i, j] > thresh else "black")

plt.tight\_layout()

plt.ylabel('True label')

plt.xlabel('Predicted label')

# Assuming y\_test and X\_test are defined

cnf\_matrix = confusion\_matrix(y\_test, RF.predict(X\_test))

np.set\_printoptions(precision=2)

plt.figure()

plot\_confusion\_matrix(cnf\_matrix, classes=[1, 0], title='RF Confusion matrix')

plt.show()

41

# CHAPTER 8

# SYSTEM TESTING

The 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, sub assemblies, 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. Itis 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 satisfaction, as shown by successfully 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 test:**

42

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

### **System Test:**

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.

# 

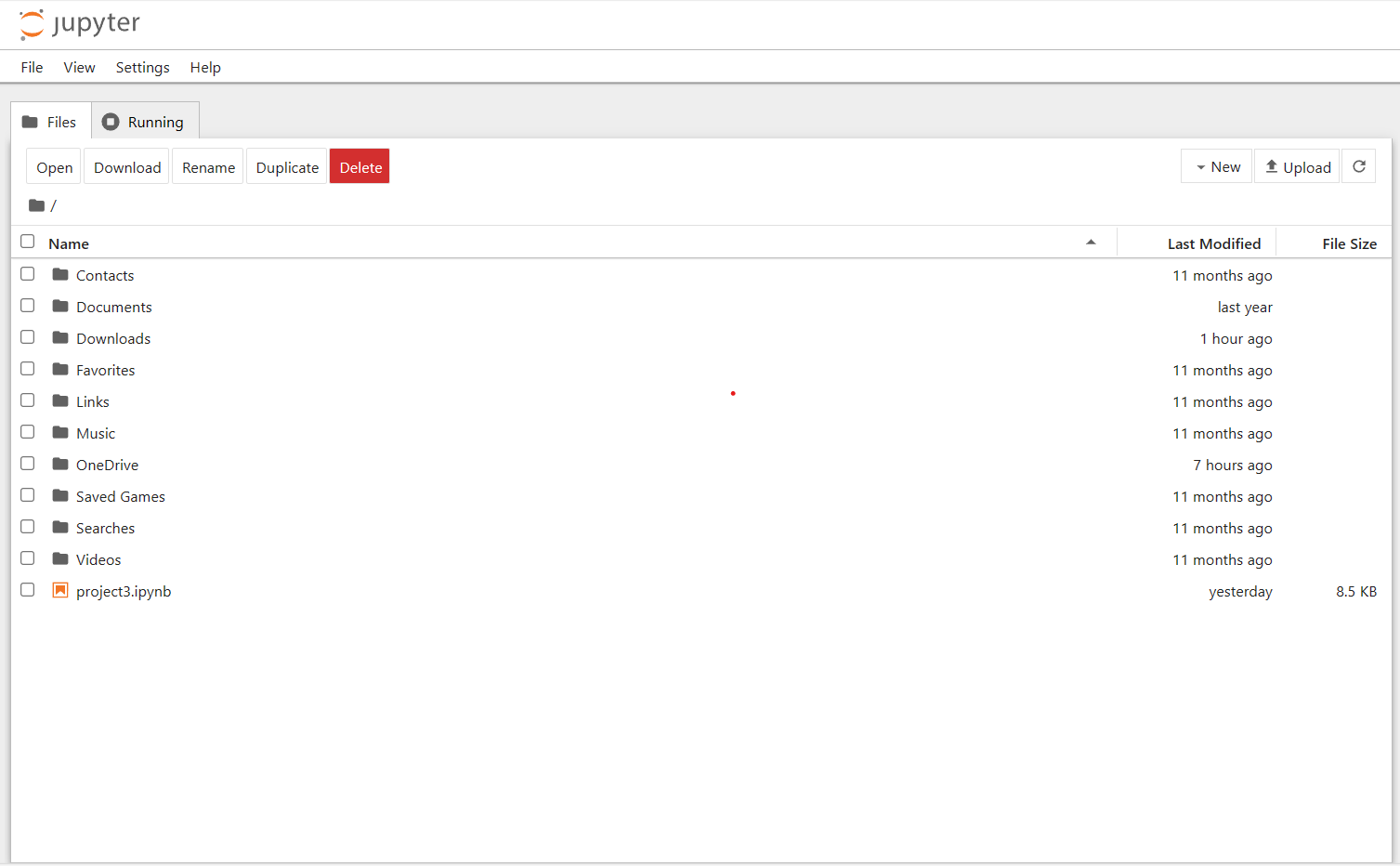
43

# CHAPTER 9

# Screenshots

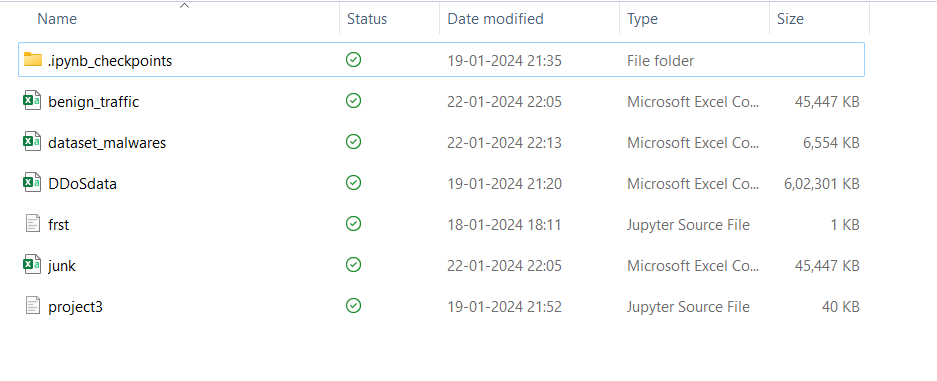
# 

**9.1 Open Terminal**

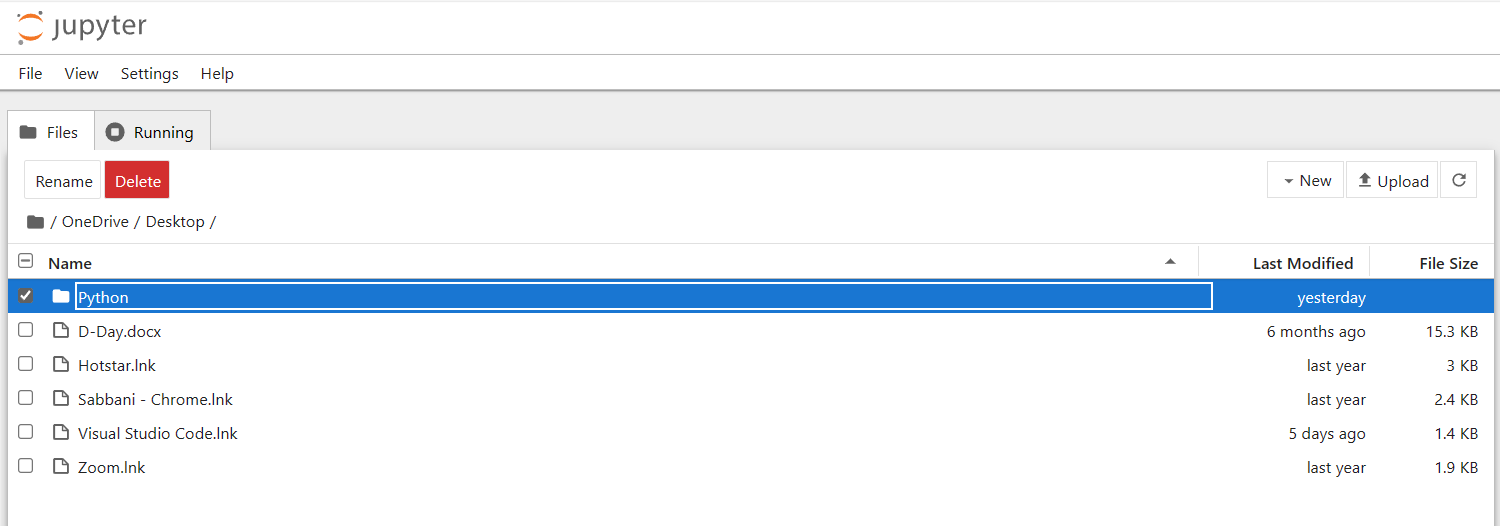


**9.2 Jupyter Notebook Software**

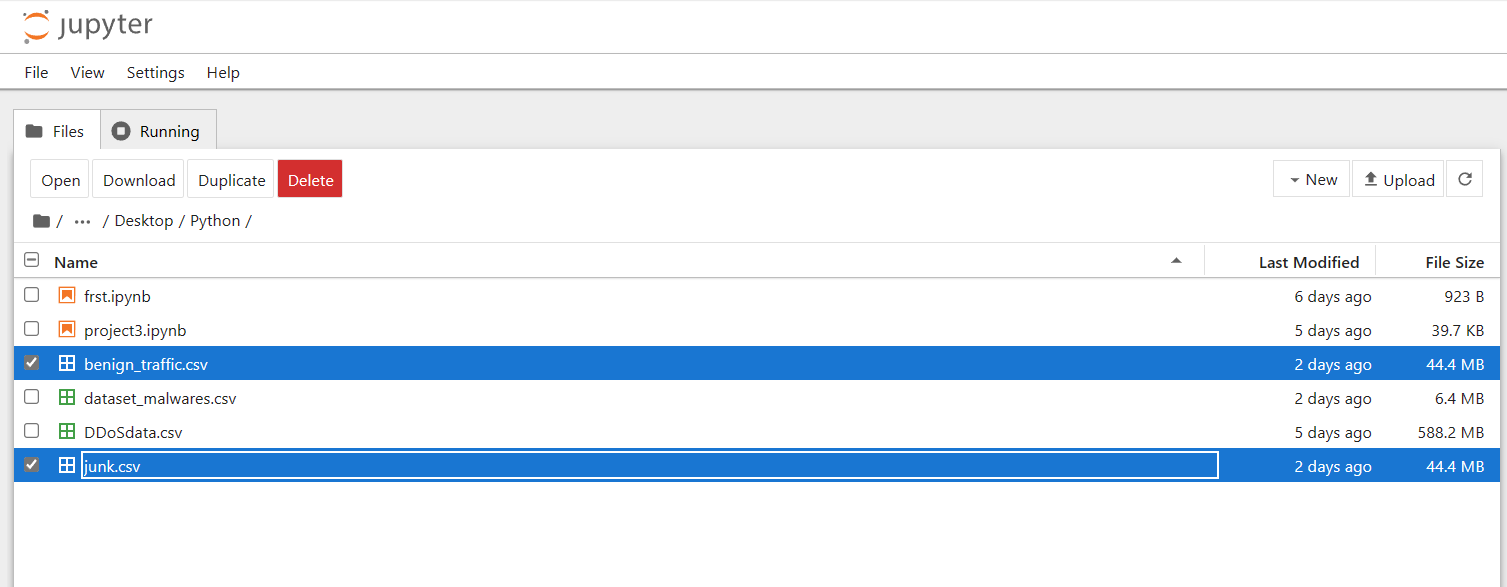
44



**9.3 Uploading Datasets**

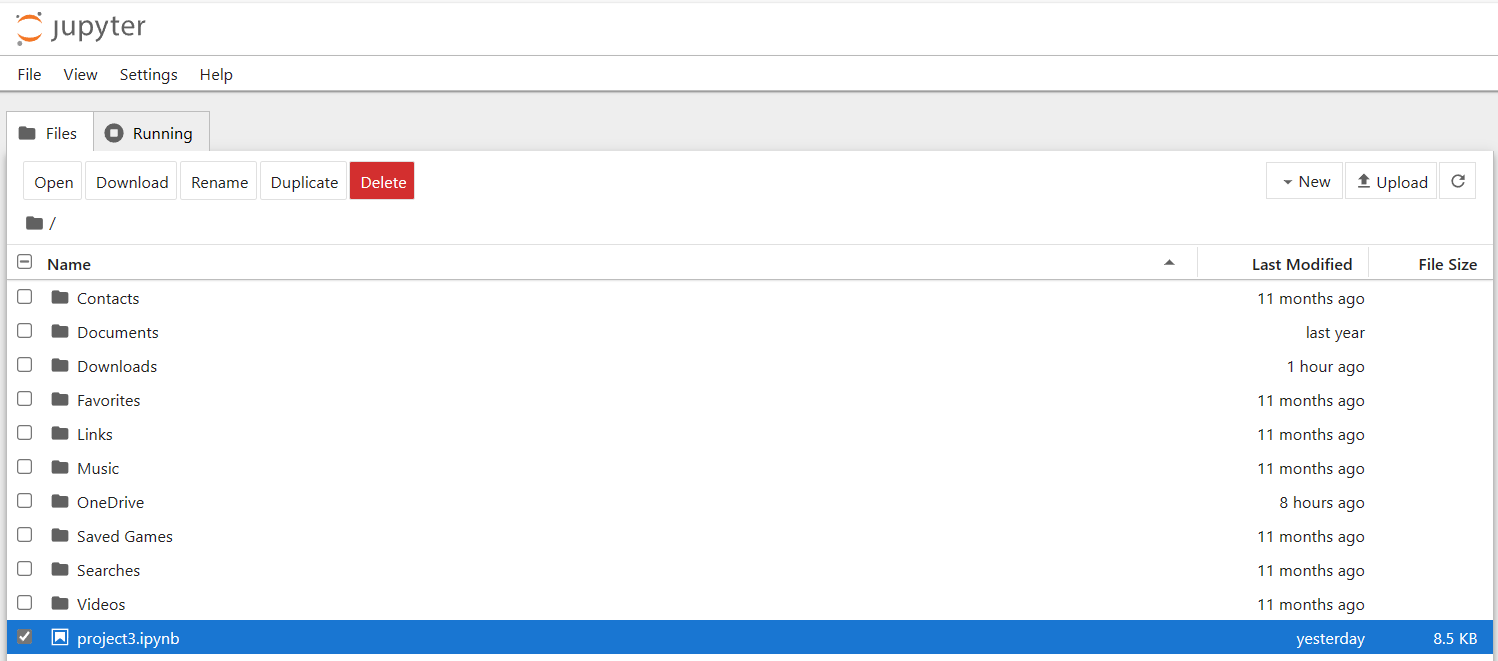


**9.3.1 Selecting File**

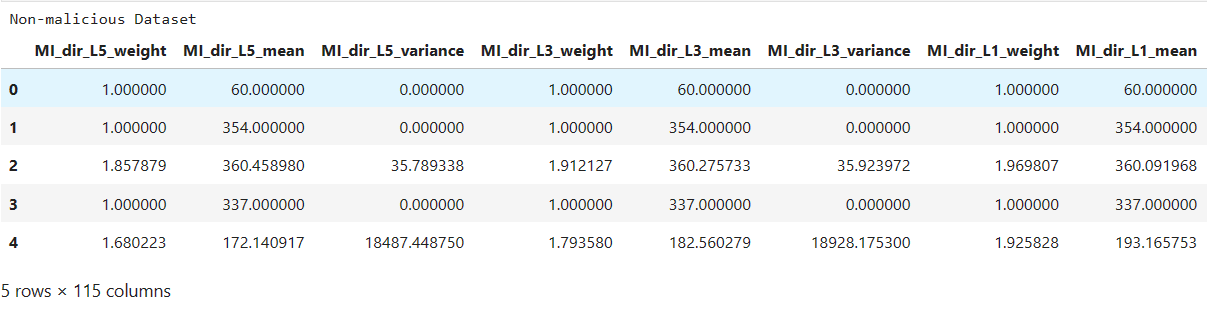
****

**9.3.2 Selecting Datasets**

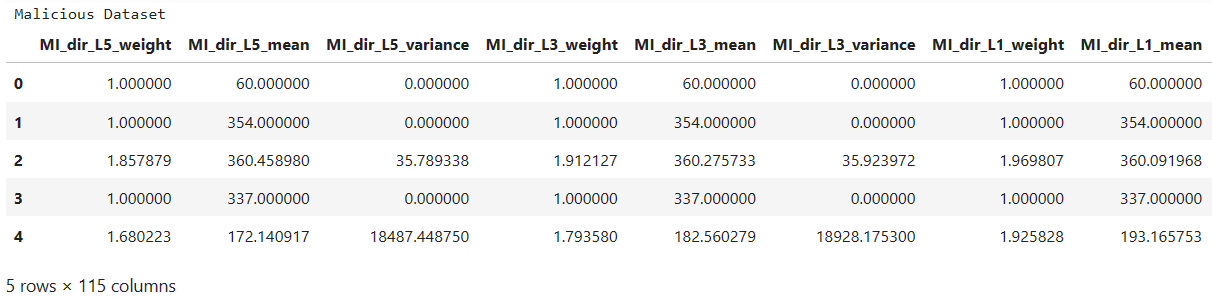
45



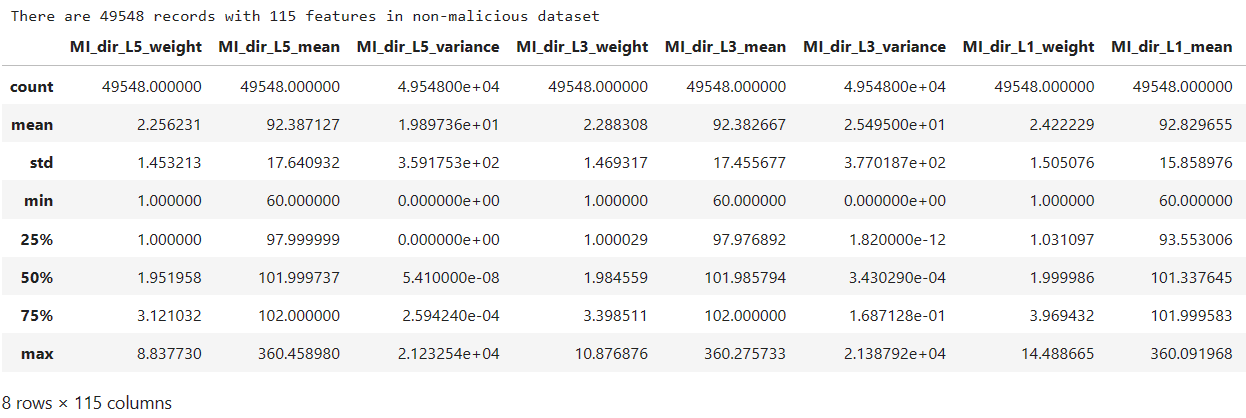
**9.4 Jupyter Notebook**

****

**9.5 Non-Malicious Dataset**

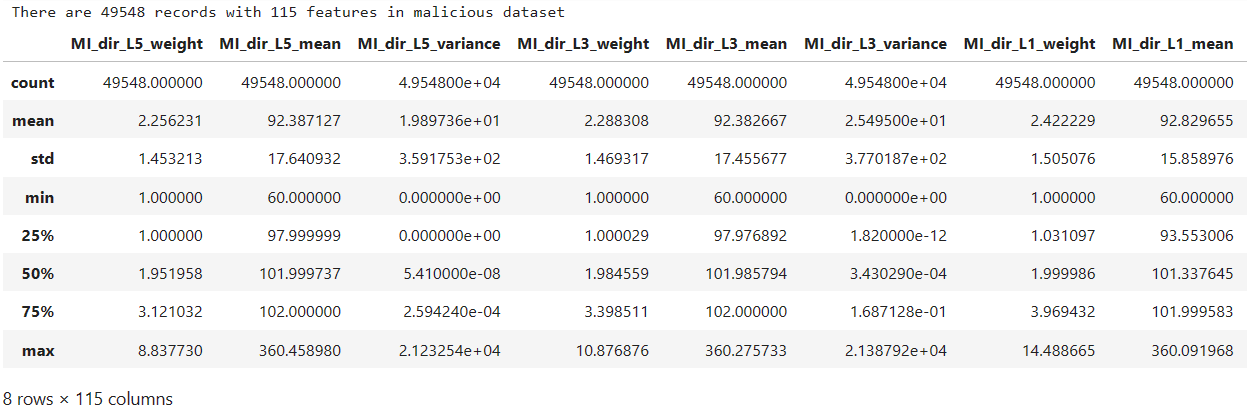


**9.6 Malicious Dataset**

****

**9.7 Description of Non-malicious Dataset**

46

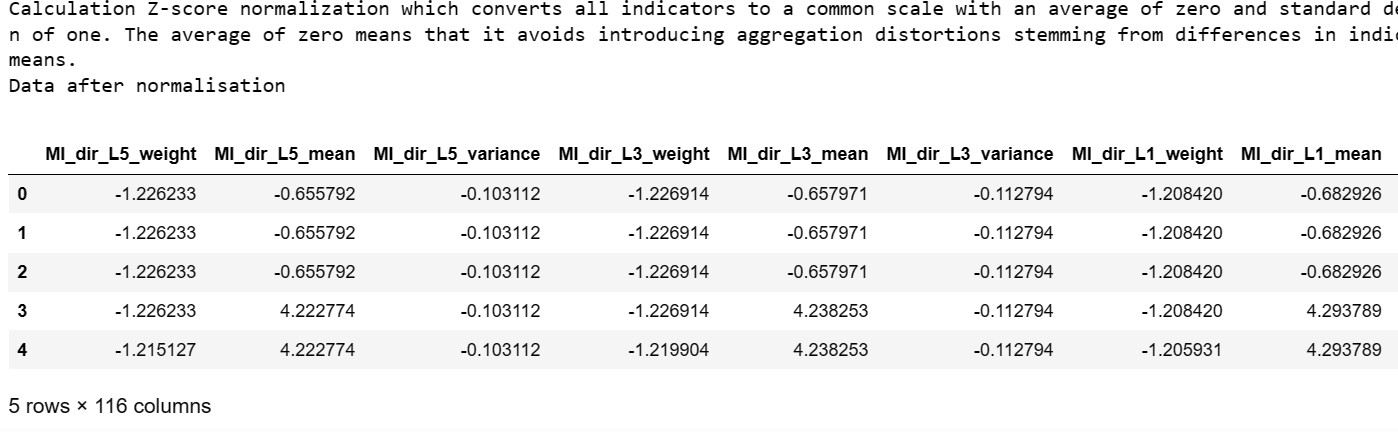


**9.8 Description of malicious dataset**

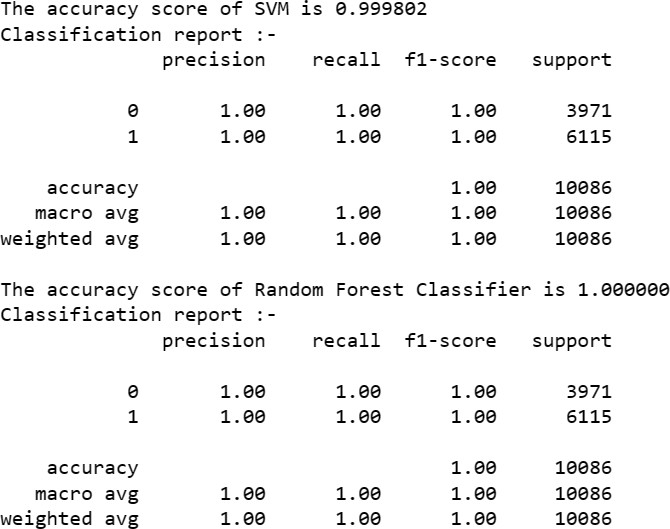
****

**9.9 Adding label to the dataset**

****

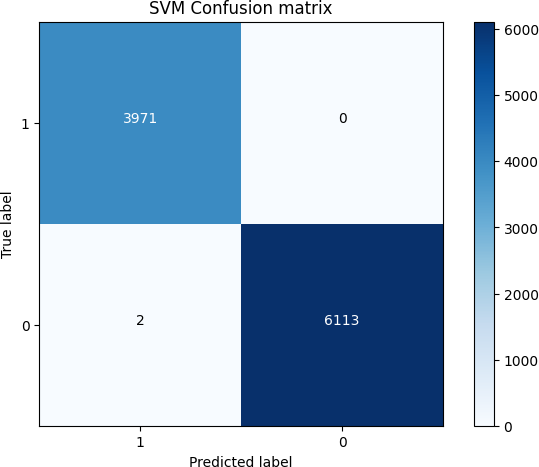
**9.10 Combining non-malicious and malicious dataset**

**9.11 Normalization**

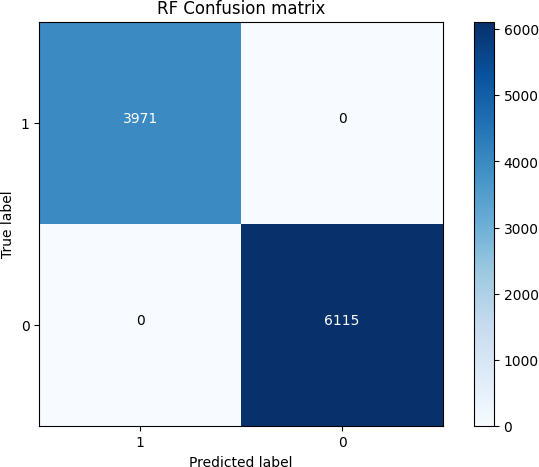


**9.12 Prediction of Accuracy Score**

47



**9.13 SVM Confusion Matrix**



**9.14 RF Confusion Matrix**

48

# Results and Discussion

Earlier experimental studies on the detection of IoT botnets or IoT traffic anomalies typically relied on emulated or simulated data. In contrary, this dataset enables empirical evaluation with \*real\* traffic data, gathered from nine commercial IoT devices infected by authentic botnets from two families in an isolated network. It facilitates the examination of Mirai and BASHLITE, two of the most common IoT-based botnets, which have already demonstrated their harmful capabilities.

Number of Attributes: 115 independent features in each file, plus a class label to be derived from the respective filename (e.g., "benign" or "TCP attack").

Attribute Information:

--The following describes each of the feature’s headers:

--Stream aggregation:

H: ("Source IP" in N-BaIoT paper) Stats summarizing the recent traffic from this packet's host (IP)

MI: ("Source MAC-IP" in N-BaIoT paper) Stats summarizing the recent traffic from this packet's host (IP + MAC)

HH: ("Channel" in N-BaIoT paper) Stats summarizing the recent traffic going from this packet's host (IP) to the packet's destination host.

HH\_jit: ("Channel jitter" in N-BaIoT paper) Stats summarizing the jitter of the traffic going from this packet's host (IP) to the packet's destination host.

HpHp: ("Socket" in N-BaIoT paper) Stats summarizing the recent traffic going from this packet's host+port (IP) to the packet's destination host+port.

-- Timeframe (The decay factor Lambda used in the damped window):

-- How much recent history of the stream is capture in these statistics

-- L5, L3, L1, L0.1 and L0.01

49

-- The statistics extracted from the packet stream:

weight: The weight of the stream (can be viewed as the number of items observed in recent history)

mean: ...

std: ...

radius: The root squared sum of the two streams' variances. magnitude: The root squared sum of the two streams' means. cov: An approximated covariance between two streams.

pcc: An approximated correlation coefficient between two streams.

50

# CHAPTER 10

**Conclusion**

Cyber-attacks involving botnets are multi-stage attacks and primarily occur in IoT environments; they begin with scanning activity and conclude with distributed denial of service (DDoS). Most existing studies concern detecting botnet attacks after IoT devices become compromised and start performing DDoS attacks. Furthermore, most machine learning-based botnet detection models are limited to a specific dataset on which they are trained. Consequently, these solutions do not perform well on other datasets due to the diversity of attack patterns. In this work, real traffic data is used for experimentation. EDA (Exploratory Data Analysis) is the statistical analysis phase through which the whole dataset is analyzed. The model will be able to be trained on a large data set in the future. ResNet50 and LSTM models, deep learning models can also be used in run-time Botnet detection. Besides being integrated with front-end web applications, the research' model can also be used with back-end web applications.

51

# CHAPTER 11

**Future Scope**

In recent years, the Internet has enabled access to widespread remote services in the distributed computing environment; however, integrity of data transmission in the distributed computing platform is hindered by a number of security issues. For instance, the botnet phenomenon is a prominent threat to Internet security, including the threat of malicious codes. The botnet phenomenon supports a wide range of criminal activities, including distributed denial of service (DDoS) attacks, click fraud, phishing, malware distribution, spam emails, and building machines for illegitimate exchange of information/materials. Therefore, it is imperative to design and develop a robust mechanism for improving the botnet detection, analysis, and removal process.

52

# CHAPTER 12

**References**

* + - 1. S. Dange and M. Chatterjee, "Iot botnet: The largest threat to the iot network" in Data Communication and Networks, Cham, Switzerland:Springer, pp. 137-157, 2020.
      2. J. Ceron, K. Steding-Jessen, C. Hoepers, L. Granville and C. Margi, "Improving IoT botnet investigation using an adaptive network layer", Sensors, vol. 19, no. 3, pp. 727, Feb. 2019.
      3. Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, et al., "N- baiot-network-based detection of iot botnet attacks using deep autoencoders", IEEE Pervas. Comput., vol. 17, no. 3, pp. 12-22, 2018.
      4. Shah, S.A.R.; Issac, B. Performance comparison of intrusion detection systems and application of machine learning to Snort system. Futur. Gener. Comput. Syst. 2018, 80, 157–170.
      5. Soe YN, Feng Y, Santosa PI, Hartanto R, Sakurai K. Machine Learning-Based IoT-Botnet Attack Detection with Sequential Architecture. Sensors. 2020; 20(16):4372. https://doi.org/10.3390/s20164372
      6. I. Ali et al., "Systematic Literature Review on IoT-Based Botnet Attack," in IEEE Access, vol. 8, pp. 212220-212232, 2020, doi: 10.1109/ACCESS.2020.3039985.
      7. Irfan, I. M. Wildani and I. N. Yulita, "Classifying botnet attack on Internet of Things device using random forest", IOP Conf. Ser. Earth Environ. Sci., vol. 248, Apr. 2019.
      8. Shah, T., Venkatesan, S. (2019). A Method to Secure IoT Devices Against Botnet Attacks. In: Issarny, V., Palanisamy, B., Zhang, LJ. (eds) Internet of Things – ICIOT 2019. ICIOT 2019. Lecture Notes in Computer Science(), vol 11519. Springer, Cham. https://doi.org/10.1007/978-3-030-23357-0\_3
      9. C. Tzagkarakis, N. Petroulakis and S. Ioannidis, "Botnet Attack Detection at the IoT Edge Based on Sparse Representation," 2019 Global IoT Summit (GIoTS), Aarhus, Denmark, 2019, pp. 1-6, doi: 10.1109/GIOTS.2019.8766388.
      10. Y. Meidan et al., "N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders," in IEEE Pervasive Computing, vol. 17, no. 3, pp. 12-22, Jul.-Sep. 2018, doi: 10.1109/MPRV.2018.03367731.

53

* + - 1. S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh and O. Jogunola, "Federated Deep Learning for Zero-Day Botnet Attack Detection in IoT-Edge Devices," in IEEE Internet of Things Journal, vol. 9, no. 5, pp. 3930-3944, 1 March1, 2022, doi: 10.1109/JIOT.2021.3100755.
      2. F. Hussain et al., "A Two-Fold Machine Learning Approach to Prevent and Detect IoT Botnet Attacks," in IEEE Access, vol. 9, pp. 163412-163430, 2021, doi: 10.1109/ACCESS.2021.3131014.
      3. Abu Al-Haija Q, Al-Dala’ien M. ELBA-IoT: An Ensemble Learning Model for Botnet Attack Detection in IoT Networks. Journal of Sensor and Actuator Networks. 2022; 11(1):18. https://doi.org/10.3390/jsan11010018
      4. Alharbi A, Alosaimi W, Alyami H, Rauf HT, Damaševičius R. Botnet Attack Detection Using Local Global Best Bat Algorithm for Industrial Internet of Things. Electronics. 2021; 10(11):1341. https://doi.org/10.3390/electronics10111341
      5. Ahmed, A.A., Jabbar, W.A., Sadiq, A.S. et al. Deep learning-based classification model for botnet attack detection. J Ambient Intell Human Comput 13, 3457–3466 (2022). https://doi.org/10.1007/s12652-020-01848-9

54