ABSTRACT

Phishing attacks are one of the most prevalent and least defended security threats today. PhishingAttacks are a primary mode of entry for attackers into an organization. Phishing is a type ofcybersecurity attack that involves stealing personal information such as passwords, credit card numbers,etc. To avoid phishing scams, we have used Machine learning techniques to detect Phishing Websites.A successful phishing attempt leads to unauthorized access to sensitive information and systems. Anapproach is proposed which uses Natural Language Processing (NLP) techniques to analyze text anddetect inappropriate statements that are indicative of phishing attacks. NLP offers a wholesome solutionfor this problem as it is capable of analyzing the textual content to perform intelligent recognition andperforming semantic analysis of text to detect malicious intent. The approach also makes use of DeepLearning frameworks with Neural Network. The Phishector (Phishing Detector) takes input as an emailand gives the output as the accuracy with which it is predicting whether an email is phished or legitimate.We have worked upon dataset: that contain the two columns and many rows columns that that containthe url and other one has the Good or bad. Also build our model using various Machine Learningtechniques such as Logistic Regression, Multinomial NB. As per our observance, Logistic Regressionand Multinomial NB Trees outperform for almost same accuracacy.

CHAPTER 1
INTRODUCTION

1.1 Overview

Detecting phishing attacks is a critical challenge in contemporary cybersecurity, as thesemalicious activities continue to exploit evolving tactics to deceive users and compromisesensitive information. Traditional defense mechanisms, such as email filtering, often fall shortin effectively identifying and thwarting sophisticated phishing attempts. To address this issue,the objective is to develop a machine learning solution capable of autonomously detecting andmitigating phishing attacks in real-time. The scope of the solution revolves around analyzingdiverse email attributes, content structures, and sender characteristics to discern patternsindicative of phishing behavior. The complexity of the problem lies in the dynamic and everchanging nature of phishing tactics, necessitating a solution that can adapt and learncontinuously. The proposed approach involves the implementation of machine learningalgorithms that learn from diverse datasets encompassing both legitimate and phishing emails.The system aims to achieve real-time detection, swiftly identifying phishing attempts as theyoccur, thus minimizing the risk of user compromise. Scalability is a key consideration,ensuring the solution can handle a high volume of incoming emails and adapt to the dynamiclandscape of phishing attacks. The performance of the system will be evaluated based onprecision, recall, and false positive rates, aiming for the development of a robust and efficientphishing detection system.

Figure 1.1.1 Types of Phishing Attack

1.2 Purpose of the system

The purpose of the detecting phishing attack system is to fortify cybersecurity defenses against the pervasive threat of deceptive online practices. By leveraging advanced machine learning algorithms, the system aims to autonomously identify and thwart phishing attacks in real-time. This proactive approach mitigates the risks associated with user compromise and the unauthorized disclosure of sensitive information. The system focuses on analyzing diverse email attributes, content structures, and sender characteristics to discern patterns indicative of phishing behavior, adapting to the dynamic nature of evolving tactics. Its primary goal is to enhance the accuracy and efficiency of phishing detection, providing a robust defense mechanism against sophisticated and continually changing attack methods. The system contributes to a secure digital environment by swiftly identifying and neutralizing phishing attempts,

safeguarding users from falling victim to fraudulent activities. Scalability is a key

consideration, ensuring the system can handle a high volume of incoming emails and remain resilient to the evolving landscape of phishing attacks. The ongoing learning process of the machine learning algorithms further strengthens the system's ability to adapt and respond effectively to emerging threats. Ultimately, the purpose is to establish a reliable and advanced defense mechanism, reducing the susceptibility of individuals and organizations to the detrimental impact of phishing attacks on cybersecurity.

CHAPTER 2

LITERATURE SURVEY

In this paper, we will discuss previous studies in terms of their methods, datasets, contributions, and results.

S. Arvind Anwekar, V. Agrawal [19]: In this study, the authors focused on extracting features from URLs, in addition to other features such as the age of the SSL certificate and the universal resource locator of the anchor, IFRAME, and website rank. They collected URLs of phishing websites from PhishTank and URLs of benign websites from the Alexa website. Using a combination of the random forest (RF), decision tree(DT), and support vector machine (SVM), contributed to improving the detection mechanism for phishing websites and achieved a high noticeable detection accuracy of 97.14%, with a low rate of false positives at 3.14%. The results also showed that the classifier's performance improves with more training data.

N. Choudhary b, K. Jain, S. Jain [20]: This study emphasizes the significance of only using attributes from the URL. Both the Kaggle and Phishtank websites make it easy to get the dataset used in this study. The researchers used a hybrid approach that com- bined Principal Component Analysis (PCA) with Support Vector Machine (SVM) and Random Forest algorithms to reduce the dataset's dimensionality while keeping all im- portant data, and it produced a higher accuracy rate of 96.8% compared to other tech- niques investigated.

A. Lakshmanarao, P. Surya, M Bala Krishna [21]: This thesis collected a dataset of phishing websites from the UCI repository and used various Machine learning tech- niques, including decision trees, AdaBoost, support vector machines (SVM), and ran- dom forests, to analyze selected features (such as web traffic, port, URL length, IP address, and URL_of_Anchor). The most effective model for detecting phishing web- sites was chosen, and two priority-based algorithms (PA1 and PA2) were proposed. The team utilized a new fusion classifier in conjunction with these algorithms and at- tained an accuracy rate of 97%. when compared to previous works in phishing website detection

L. Tang, Q. Mahmoud [22]: The proposed approach in the current study uses URLs collected from a variety of platforms, including Kaggle, Phish Storm, Phish Tank, and ISCX-UR, to identify phishing websites. The researchers made a big contribution since they created a browser plug-in that can quickly recognize phishing risks and offer warnings. Various datasets and machine learning techniques were investigated, and the proposed RNN-GRU model outperformed SVM, Random Forest (RF), and Logistic Regression with a maximum accuracy rate of 99.18%. On the other hand, the suggested method is not always accurate in identifying if short links are phishing risks.

A. Kulkarni & L. Brown[23]: A machine learning system was created to categorize websites based on URLs from the University of California, Irvine Machine Learning Repository. Four classifiers were used: SVM, decision tree, Naive Bayesian, and neural network. The outcome of experiments utilizing the model developed with the support of a training set of data demonstrates that the classifiers were able to successfully differentiate authentic websites from fake ones with an accuracy rate of over 90%. Limitations include a small dataset and all features being

discrete, which may not be suitable for some classifiers.

Tyagi; J. Shad; S. Sharma; S. Gaur Gagandeep Kaur [24]: The research taken into account focuses on the use of various machine learning algorithms to identify if a website is legitimate or a phishing site based on a URL. This study's most important contribution is the creation of the Generalized Linear Model (GLM), a brand-new model. This model combines the results of two various methods. With a 98.4% accuracy rate, the Random Forest and GLM combination produced the best results for detecting phishing websites.

M. Karabatak and T. Mustafa [25]: The objective of this research is to assess the effectiveness of classification algorithms on a condensed dataset of phishing websites obtained from the UCI Machine Learning Repository. The paper investigates how data mining and feature selection algorithms affect reduced datasets through experiments and analysis, finally selecting the methods that perform the best in terms of classification. According to the results, some classification strategies improve performance while others have the opposite impact. Ineffective classifiers for condensed phishing datasets included Lazy, BayesNet, SGD Multilayer Perceptron, PART, JRip, J48, RandomTree, and RandomForest. However, it was discovered that KStar, LMT, ID3, and R.F.Clas- sifier were efficient. Lazy produced the highest classification accuracy rate of 97.58% on the compressed 27-feature dataset, whereas KStar performed at its best on the same dataset.

W. Fadheel, M. Abusharkh, and I. Abdel-Qader [27]: The present study utilized da- tasets from the UCI machine learning repository, including Domain, HTML, Address Bar, and URLs, the main contribution was conducting a comparative analysis of the impact of feature selection on detecting phishing websites. The KMO test was applied in the study to evaluate the dataset using (LR) and (SVM) classification algorithms.

The test was conducted based on a correlation matrix to analyze the performance. Re- sults showed that LR with the KMO test achieved an accuracy of 91.68%, while SVM with the KMO test yielded an accuracy of 93.59%

CHAPTER 3

EXISTING SYSTEM

Several existing systems focus on phishing attack detection, employing a range of techniques and technologies to enhance cybersecurity. Some notable examples include:

1. Google Safe Browsing:
   Provides a database of known phishing websites, warning users before accessing potentially harmful links.
2. Microsoft Defender SmartScreen:
   Integrated into Microsoft Edge and Internet Explorer, it identifies and blocks known phishing sites.
3. PhishTank:
   A community-driven platform that collects and shares information about phishing attacks, offering a database for detection.
4. Cisco Email Security:
   Utilizes advanced threat intelligence and machine learning to identify and block phishing emails.
5. Symantec Email Security:
   Employs email filtering and analysis to detect and block phishing attempts in realtime.
6. FireEye Email Security:
   Utilizes a multi-vector approach, combining signature-based detection and machine learning to identify phishing threats.
7. Phishing Intelligence Engine (PIE):
   Developed by Agari, it uses artificial intelligence to detect and prevent phishing attacks by analyzing email behavior.
8. Proofpoint Email Protection:
   Employs a comprehensive approach to block phishing emails, combining email filtering, URL analysis, and threat intelligence.
9. Cofense PhishMe:
   Focuses on human-driven detection, training users to recognize and report phishing threats effectively
10. IronScales:
    Utilizes machine learning and collective intelligence to automatically detect and mitigate phishing attacks.
11. Area 1 Security:
    Employs predictive analytics and preemptive threat intelligence to block phishing attacks across various channels.
12. Sophos Phish Threat:
    Provides simulated phishing attacks to train users and offers real time protection against actual threats

13.Barracuda Sentinel:

Uses artificial intelligence to analyze communication patterns and detect phishing attempts within emails.

14.Web of Trust (WOT):

A browser extension that crowdsources website reputation data, warning users about potentially malicious sites.

15.F5 Silverline Shape Defense:

Combines behavioral analysis, machine learning, and threat intelligence to protect against phishing attacks and fraudulent activities.

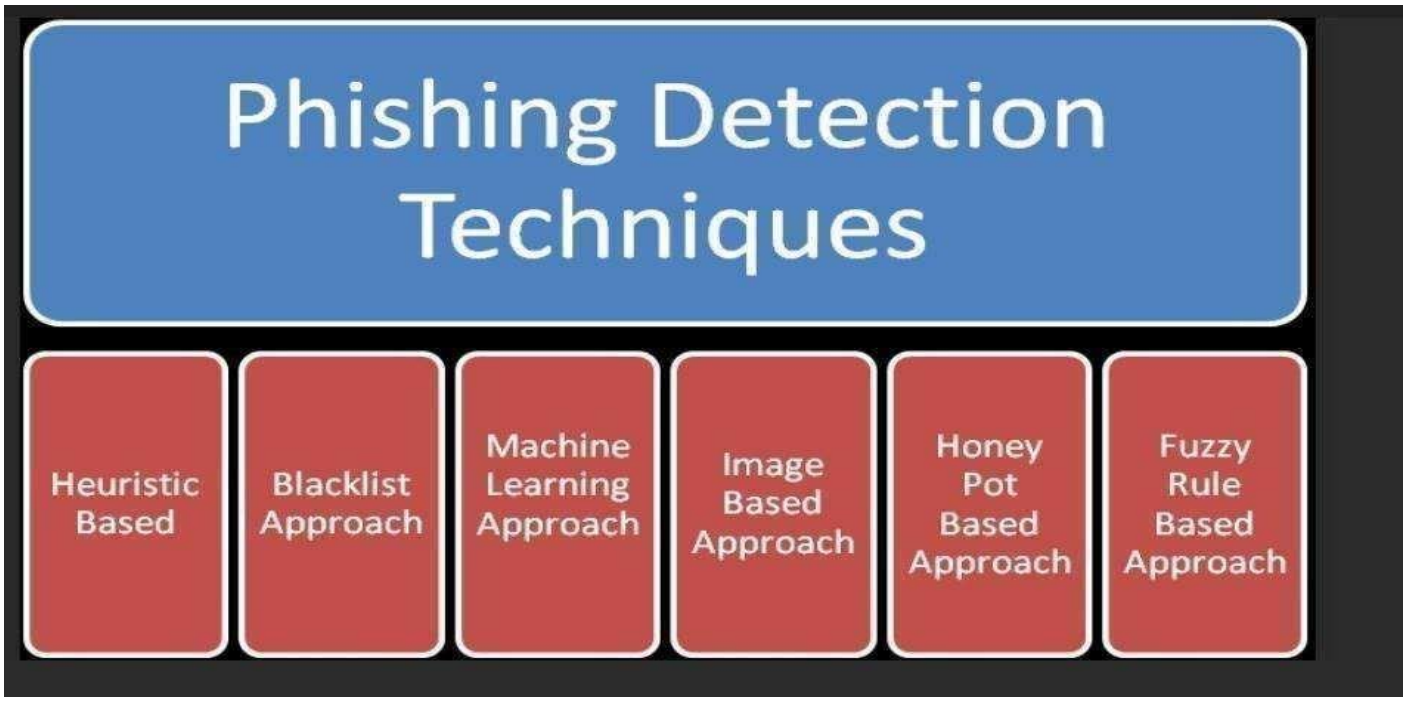
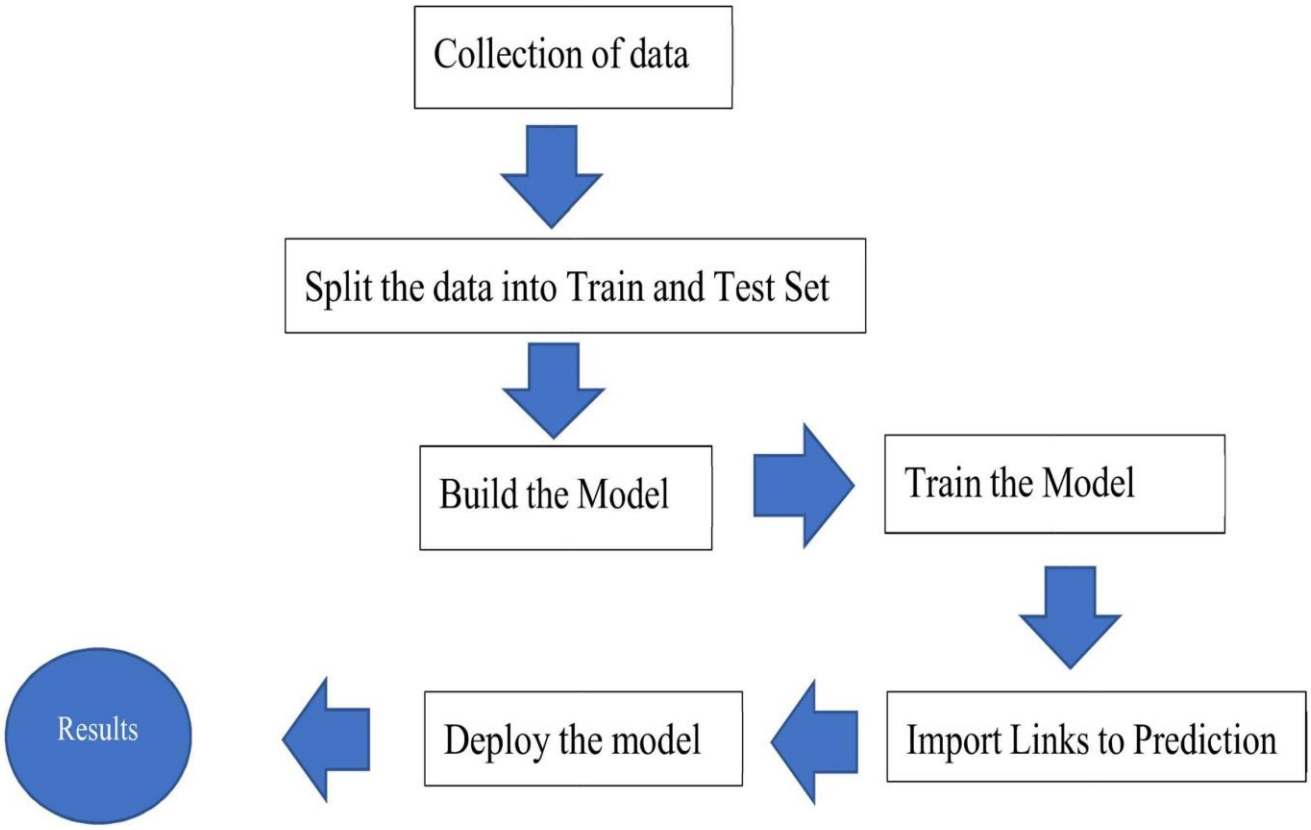


Figure 3.1.1 Phishing Detection Techniques

CHAPTER 4
PROPOSED SYSTEM

4.1. Proposed System

Figure 4.1.1 Proposed System



1. Problem Definition:

   Define the problem clearly:

   Detect phishing attacks based on URL features using machine learning.

2. Dataset Overview:

   Dataset Source:

   Specify the source ofthe dataset (e.g., collected from online sources, generated synthetically).

   Dataset Characteristics:

   Understand the distribution ofgood and bad URLs in thedataset

3. Data Preprocessing:

   Feature Engineering:

   Extract relevant features such as domain length, path length, the presence ofspecial characters, and others.

Explore domain-specific features that might be indicative ofphishing.

Label Encoding: Convert categorical labels (good/bad) into numerical format.

4. Dataset Splitting:

   Training and Testing Split:

   Split the dataset into training (80%) and testing (20%) sets
5. Feature Scaling:

   Normalization:

   Normalize numerical features to ensure they are on a similar scale.
6. Model Selection:

   Logistic Regression:

   Train a logistic regression model for its simplicity and interpretability. Optimize hyperparameters using techniques like grid search or randomized search.

   Multinomial Naive Bayes:

   Train a Multinomial Naive Bayes model for its effectiveness with text-based features.
7. Ensemble Learning:

   Voting Classifier:

   Combine predictions from logistic regression and Multinomial Naive Bayes using a voting classifier.

   Experiment with different ensemble methods (e.g., soft voting, hard voting) to find the most effective combination.
8. Model Evaluation:

   Metrics:

   Evaluate models using metrics like accuracy, precision, recall, F1 score, and ROC- AUC. Consider the trade-offs between false positives and false negatives based on the application'requirements.

Algorithms used

1 Logistic Regression

Logistic regression is used for binary classification where we uses sigmoid function that takes input as independent variables and produces a probability value between 0 and 1.

For example, we have two classes Class 0 and Class 1 if the value of the logistic function for an input is greater than 0.5 (threshold value) then it belongs to Class 1 otherwise it belongs to Class 0. It's referred to as regression because it is the extension of linear regression but is mainly used for classification problems.

Key Points:

- Logistic regression predicts the output of a categorical dependent variable. Therefore, the outcome must be a categorical or discrete value.

- It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).

Logistic Function – Sigmoid Function

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.

- It maps any real value into another value within a range of 0 and 1. The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form.
  The S-form curve is called the Sigmoid function or the logistic function.

- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Types of Logistic Regression

On the basis of the categories, Logistic Regression can be classified into three types:

1. Binomial: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.

2. Multinomial: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"

3. Ordinal: In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Assumptions of Logistic Regression

We will explore the assumptions of logistic regression as understanding these assumptions is important to ensure that we are using appropriate application of the model. The assumption include:

1. Independent observations: Each observation is independent of the other. meaning there is no correlation between any input variables.

2. Binary dependent variables: It takes the assumption that the dependent variable must be binary or dichotomous, meaning it can take only two values. For more than two categories SoftMax functions are used.

3. Linearity relationship between independent variables and log odds: The relationship between the independent variables and the log odds of the dependent variable should be linear.

4. No outliers: There should be no outliers in the dataset.

5. Large sample size: The sample size is sufficiently large

Terminologies involved in Logistic Regression

Here are some common terms involved in logistic regression:

- Independent variables: The input characteristics or predictor factors applied to the dependent variable's predictions.
- Dependent variable: The target variable in a logistic regression model, which we are trying to predict.
- Logistic function: The formula used to represent how the independent and dependent variables relate to one another. The logistic function transforms the input variables into a probability value between 0 and 1, which represents the likelihood of the dependent variable being 1 or 0.

- Odds: It is the ratio of something occurring to something not occurring. it is different from probability as the probability is the ratio of something occurring to everything that could possibly occur.

- Log-odds: The log-odds, also known as the logit function, is the natural logarithm of the odds. In logistic regression, the log odds of the dependent variable are modeled as a linear combination of the independent variables and the intercept.

- Coefficient: The logistic regression model's estimated parameters, show how the independent and dependent variables relate to one another.

- Intercept: A constant term in the logistic regression model, which represents the log odds when all independent variables are equal to zero.

- Maximum likelihood estimation: The method used to estimate the coefficients of the logistic regression model, which maximizes the likelihood of observing the data given the model.

How does Logistic Regression work?

The logistic regression model transforms the linear regression function continuous value output into categorical value output using a sigmoid function, which maps any realvalued set of independent variables input into a value between 0 and 1. This function is known as the logistic function.

Let the independent input features be:

X=[x11 …x1mx21 …x2m ⋮⋱⋮ xn1 …xnm]X= x11 x21

⋮xn1 ……⋱ …x1mx2m⋮ xnm

and the dependent variable is Y having only binary value i.e. 0 or 1. Y={0 if Class

11 if Class 2Y={01 if Class1 if Class2 then, apply the multi-linear function to

the input variables X.

$z=(\sum i=1nwixi)+b$

Here xixi is the ith observation of X, wi=[w1,w2,w3,⋯ ,wm]wi=[w1,w2,w3,⋯,wm] is the weights or Coefficient, and b is the bias term also known as intercept. simply this can be represented as the dot product of weight and bias.

$z=w\cdot X+bz=w\cdot X+b$

2.MultinomialNB

Multinomial Naive Bayes (MNB) is a popular machine learning algorithm for text classification problems in Natural Language Processing (NLP). It is particularly useful for problems that involve text data with discrete features such as word frequency counts. MNB works on the principle of Bayes theorem and assumes that the features are conditionally independent given the class variable.

Here are the steps for applying Multinomial Naive Bayes to NLP problems:

Preprocessing the text data: The text data needs to be preprocessed before applying the algorithm. This involves steps such as tokenization, stop-word removal, stemming, and lemmatization.

Feature extraction: The text data needs to be converted into a feature vector format that can be used as input to the MNB algorithm. The most common method of feature

extraction is to use a bag-of-words model, where each document is represented by a vector of word frequency counts.

Splitting the data: The data needs to be split into training and testing sets. The training set is used to train the MNB model, while the testing set is used to evaluate its performance.

Training the MNB model: The MNB model is trained on the training set by estimating the probabilities of each feature given each class. This involves calculating the prior probabilities of each class and the likelihood of each feature given each class.

Evaluating the performance of the model: The performance of the model is evaluated using metrics such as accuracy, precision, recall, and F1-score on the testing set.

Using the model to make predictions: Once the model is trained, it can be used to make predictions on new text data. The text data is preprocessed and transformed into the feature vector format, which is then input to the trained model to obtain the predicted class label.

MNB is a simple and efficient algorithm that works well for many NLP problems such as sentiment analysis, spam detection, and topic classification. However, it has some limitations, such as the assumption of independence between features, which may not hold true in some cases. Therefore, it is important to carefully evaluate the performance of the model before using it in a real-world application.

Naive Bayes Classifier Algorithm is a family of probabilistic algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of a feature.
Bayes theorem calculates probability P(c|x) where c is the class of the possible outcomes and x is the given instance which has to be classified, representing some certain features. P(c|x) = P(x|c) * P(c) / P(x)
Naive Bayes are mostly used in natural language processing (NLP) problems. Naive Bayes predict the tag of a text. They calculate the probability of each tag for a given text and then output the tag with the highest one.

Feature Engineering:

The important part is to find the features from the data to make machine learning algorithms works.

In this case, we have text. We need to convert this text into numbers that we can do calculations on. We use word frequencies. That is treating every document as a set of the words it contains. Our features will be the counts of each of these words. In our case, we have P(positive | overall liked the movie), by using this theorem:

P(positive | overall liked the movie) = P(overall liked the movie | positive) * P(positive) / P(overall liked the movie)
Since for our classifier we have to find out which tag has a bigger probability, we can discard the divisor which is the same for both tags,
P(overall liked the movie | positive)* P(positive) with P(overall liked the movie | negative) * P(negative)
There's a problem though: "overall liked the movie" doesn't appear in our training dataset, so the probability is zero. Here, we assume the 'naive' condition that every word in a sentence is independent of the other ones. This means that now we look at individual words.
We can write this as:
P(overall liked the movie) = P(overall) * P(liked) * P(the) * P(movie) The next step is just applying the Bayes

theorem:-

P(overall liked the movie| positive) = P(overall | positive) * P(liked | positive) * P(the
| positive) * P(movie | positive)

And now, these individual words actually show up several times in our training data, and we can calculate them!
Calculating probabilities:
First, we calculate the a priori probability of each tag: for a given sentence in our training data, the probability that it is positive P(positive) is 3/5. Then, P(negative) is 2/5.
Then, calculating P(overall | positive) means counting how many times the word "overall" appears in positive texts (1) divided by the total number of words in

positive (17). Therefore, P(overall | positive) = 1/17, P(liked/positive) = 1/17, P(the/positive) = 2/17, P(movie/positive) = 3/17.

If probability comes out to be zero then By using Laplace smoothing: we add 1 to every count so it's never zero. To balance this, we add the number of possible words to the divisor, so the division will never be greater than 1. In our case, the total possible words count are 21.

Applying smoothing, The results are:

Word	P(word | positive)	P(word | negative)
overall	1 + 1/17 + 21	0 + 1/7 + 21
liked	1 + 1/17 + 21	0 + 1/7 + 21
the	2 + 1/17 + 21	0 + 1/7 + 21
movie	3 + 1/17 + 21	1 + 1/7 + 21

Now we just multiply all the probabilities, and see who is bigger:
P(overall | positive) * P(liked | positive) * P(the | positive) * P(movie | positive) * P(positive) = 1.38 * 10^{-5} = 0.0000138

P(overall | negative) * P(liked | negative) * P(the | negative) * P(movie | negative) * P(negative) = 0.13 * 10^{-5} = 0.0000013

Our classifier gives “overall liked the movie” the positive tag.

CHAPTER 5
ER DIAGRAM

- ER model stands for an Entity-Relationship model. It is a high-level data model. This model is used to define the data elements and relationship for a specified system.
- It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.
- In ER modeling, the database structure is portrayed as a diagram called an entityrelationship diagram.

1. Entity:

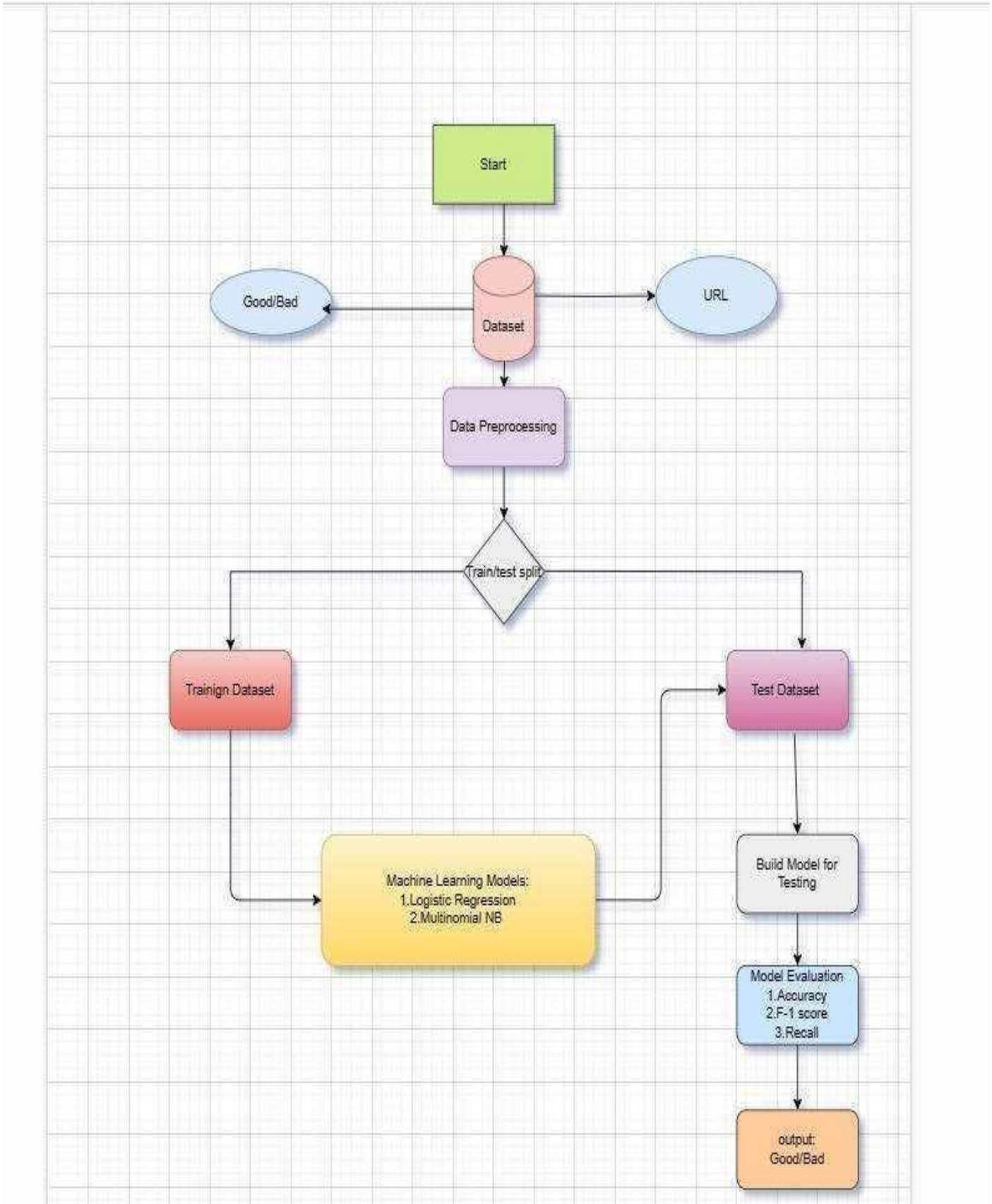
   An entity may be any object, class, person or place. In the ER diagram, an entity can be represented as rectangles.

2. Attribute

   The attribute is used to describe the property of an entity. Eclipse is used to represent an attribute.

3. Relationship

   A relationship is used to describe the relation between entities. Diamond or rhombus is used to represent the relationship.


Start
Dataset
Good/Bad
URL
Data Preprocessing
Train/test split
Trainign Dataset
Test Dataset
Machine Learning Models:
1.Logistic Regression
2.Multinomial NB
Build Model for Testing
Model Evaluation
1.Accuracy
2.F-1 score
3.Recall
output:
Good/Bad

CHAPTER 6
MACHINE LEARNING

What is Machine Learning

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.Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. Understanding the problem setting in machine learning is essential to using these tools effectively.

6.1.Categories of Machine Leaning

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 labels 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. 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. 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. 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 datadriven decisions taken by machines, particularly to automate the process. These datadriven decisions can be used, instead of using programming logic, in problems that cannot be programmed inherently. The fact is that we can’t do without human intelligence, but another 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.

6.2.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 –

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 s ll in its infancy stage, the availability of expert resources is a tough job.
4.No clear objective for formulating business problems − Having no clear objec ve 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 overfi ng or underfi ng, it cannot be represented well for the problem.

6.3. 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 realworld complex problems that cannot be solved with the traditional approach. Following are some realworld applications of ML.
Emotion analysis
Sentiment analysis
Error detection and prevention
Weather forecasting and prediction
Stock market analysis and forecasting Speech recognition
Customer segmentation
Object recognition
Fraud detection and prevention

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. 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 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. 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,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.Many Python libraries are specifically useful for Artificial Intelligence and Machine Learning such as Keras, TensorFlow, Scikit-learn, etc. 21 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:

(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 unlabelled data and then finding 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.

6.4. 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.

6.5. 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.

CHAPTER 7

SOFTWEAR REQUIREMENS

1. For Data Preprocessing:

7.1.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 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 frameworks Multimedia

Advantages of Python

Let's see how Python dominates over other languages.

1.Extensive Libraries :

Python downloads with an extensive library and it contains code for various purposes like regular expressions documentation generation, unittesting, 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.

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 systemdependent 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.

7.1.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 manypeople suggest learning Python to beginners.

2. 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.

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.

7.2.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 client-side. 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.

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.

7.3.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, Guidovan 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."

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 lists, 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.

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.

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.PythonPython is an interpreted high-level programming language for generalpurpose 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. 7.4. Modules Used in Project

NumPy

NumPy is a general-purpose array-processing package. It provides a highperformance 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 multidimensional 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 highperformance 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 wide range of fields including academic and commercial domains including finance,economics,Statistics, analytics, etc.

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 Python is an interpreted high-level programming language for generalpurpose 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 run me 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.

Installing Jupyter Notebook on Windows

Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. Uses include data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

Jupyter Notebook can be installed by using either of the two ways described below:

Using Anaconda:

Install Python and Jupyter using the Anaconda Distribution, which includes Python, the Jupyter Notebook, and other commonly used packages for scientific computing and data science. To install Anaconda, go through How to install Anaconda on windows? and follow the instructions provided.

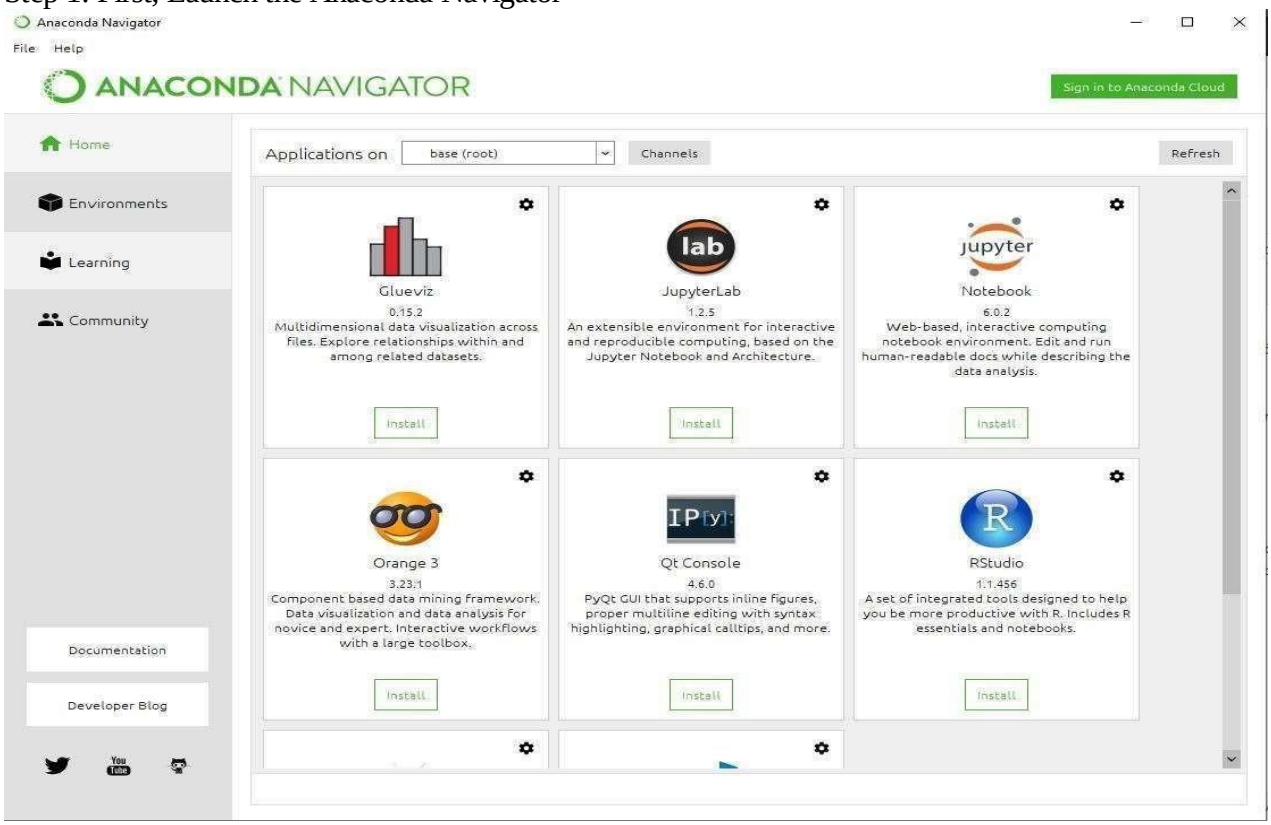
Using PIP:

Install Jupyter using the PIP package manager used to install and manage software packages/libraries written in Python. To install pip, go through How to install PIP on Windows? and follow the instructions provided.

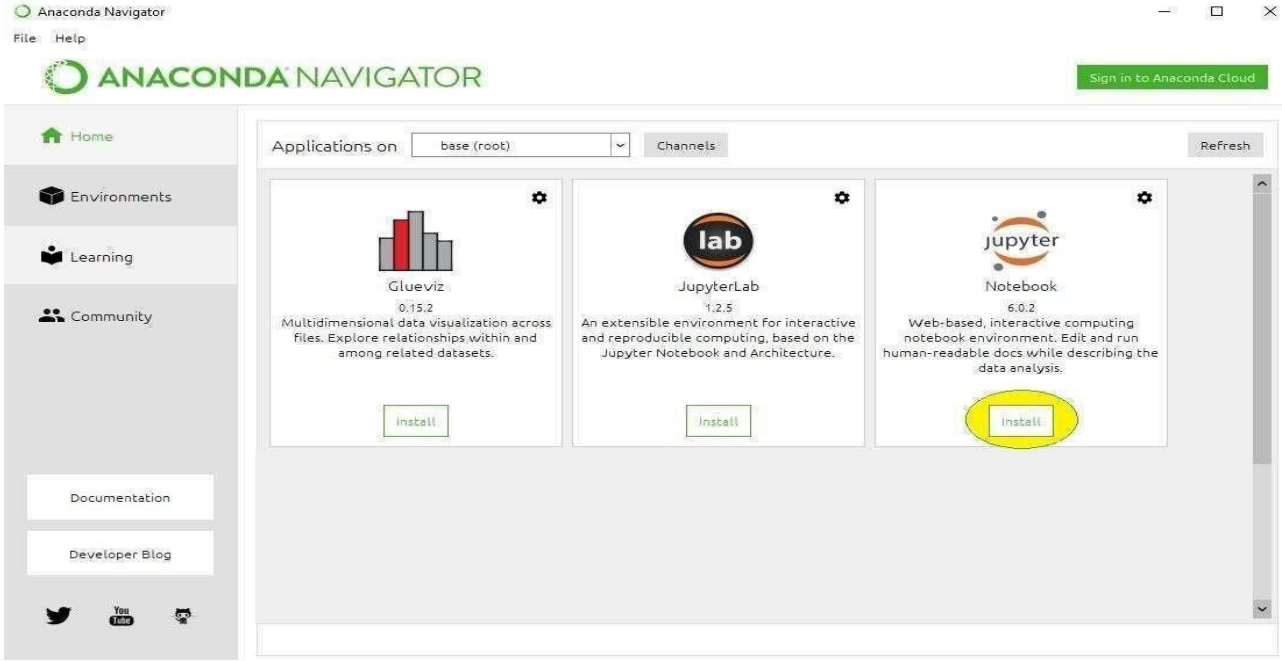
Installing Jupyter Notebook using Anaconda

Anaconda is an open-source software that contains Jupyter, spyder, etc that are used for large data processing, data analytics, heavy scientific computing. Anaconda works for R and python programming language. Spyder(sub-application of Anaconda) is used for python. Opencv for python

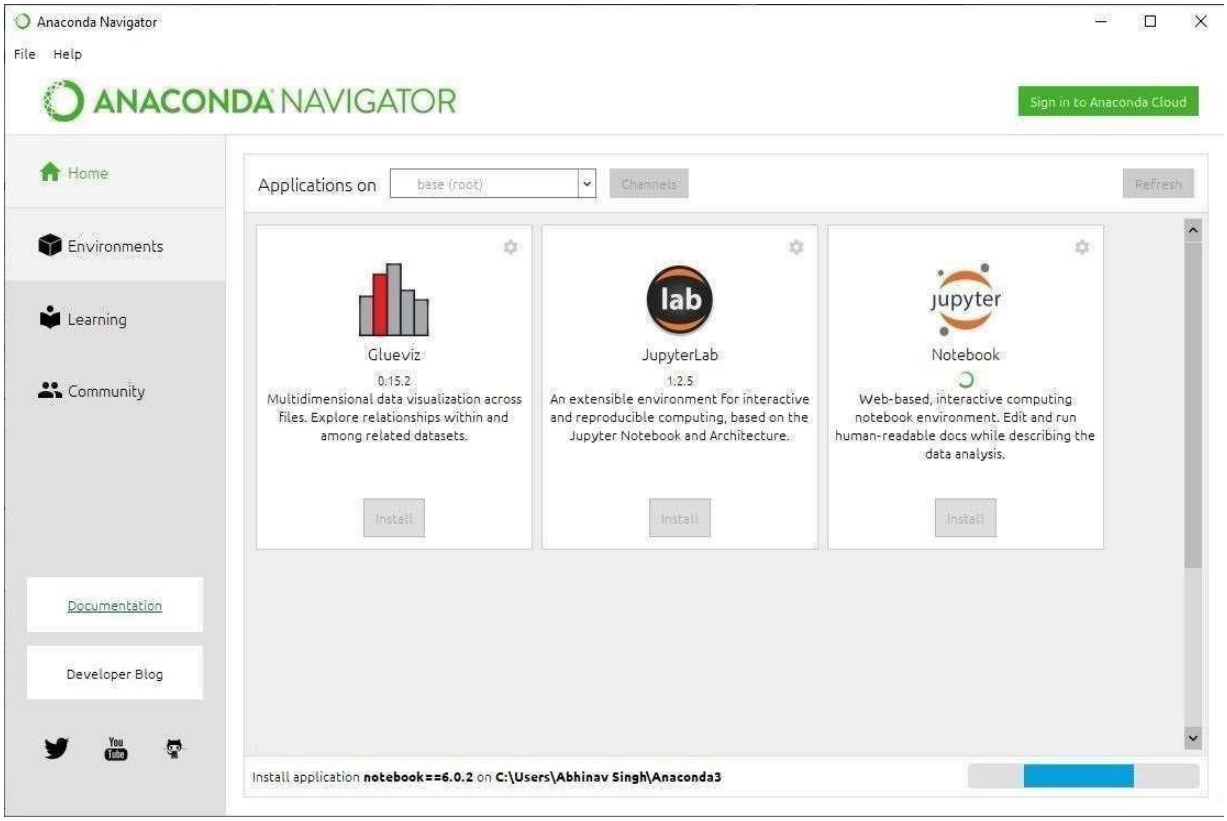
will work in spyder. Package versions are managed by the package management system called conda.
To install Jupyter using Anaconda, just go through the following instructions:
Step 1: First, Launch the Anaconda Navigator



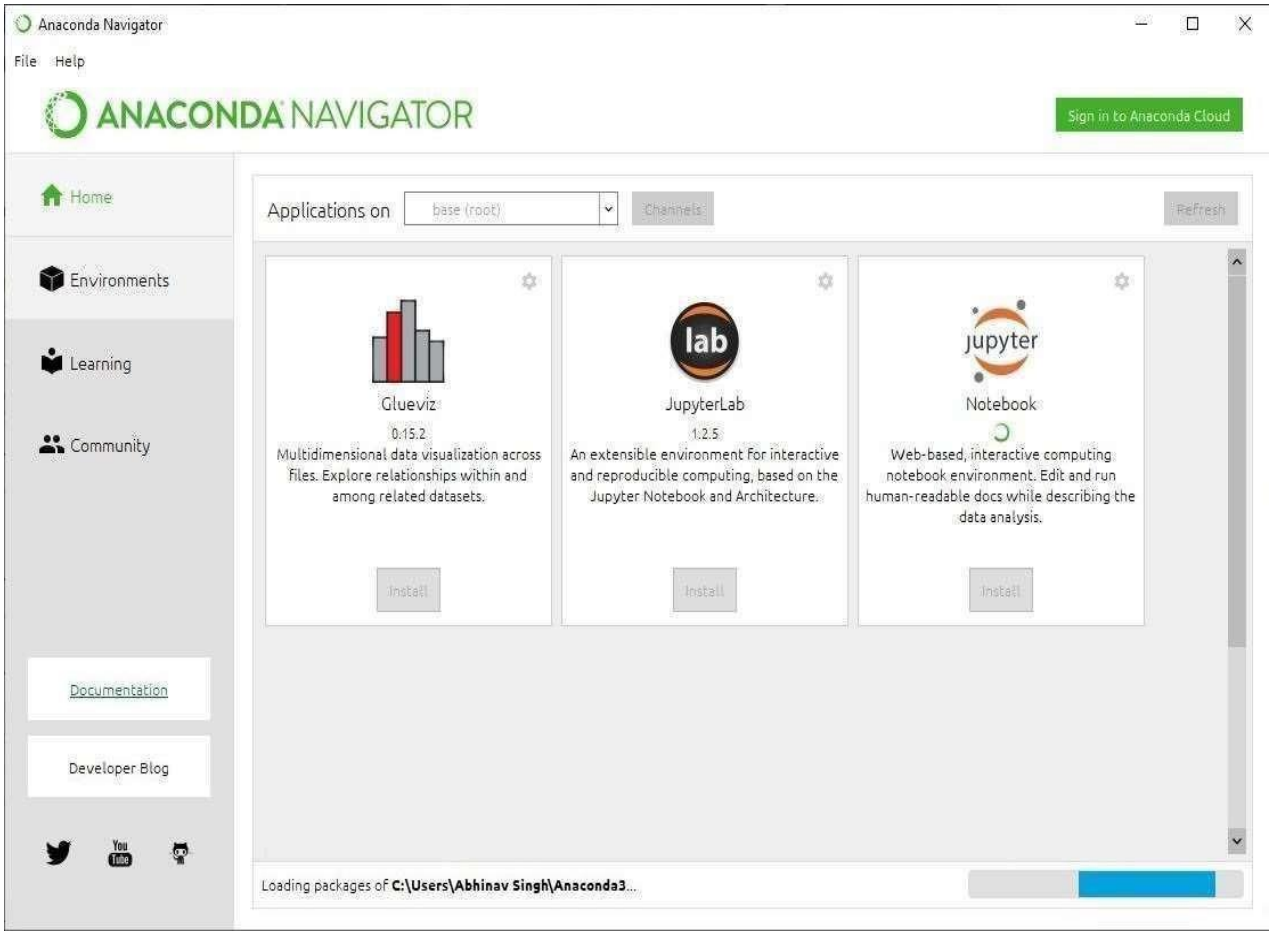
Step 2: Click on the Install Jupyter Notebook Button



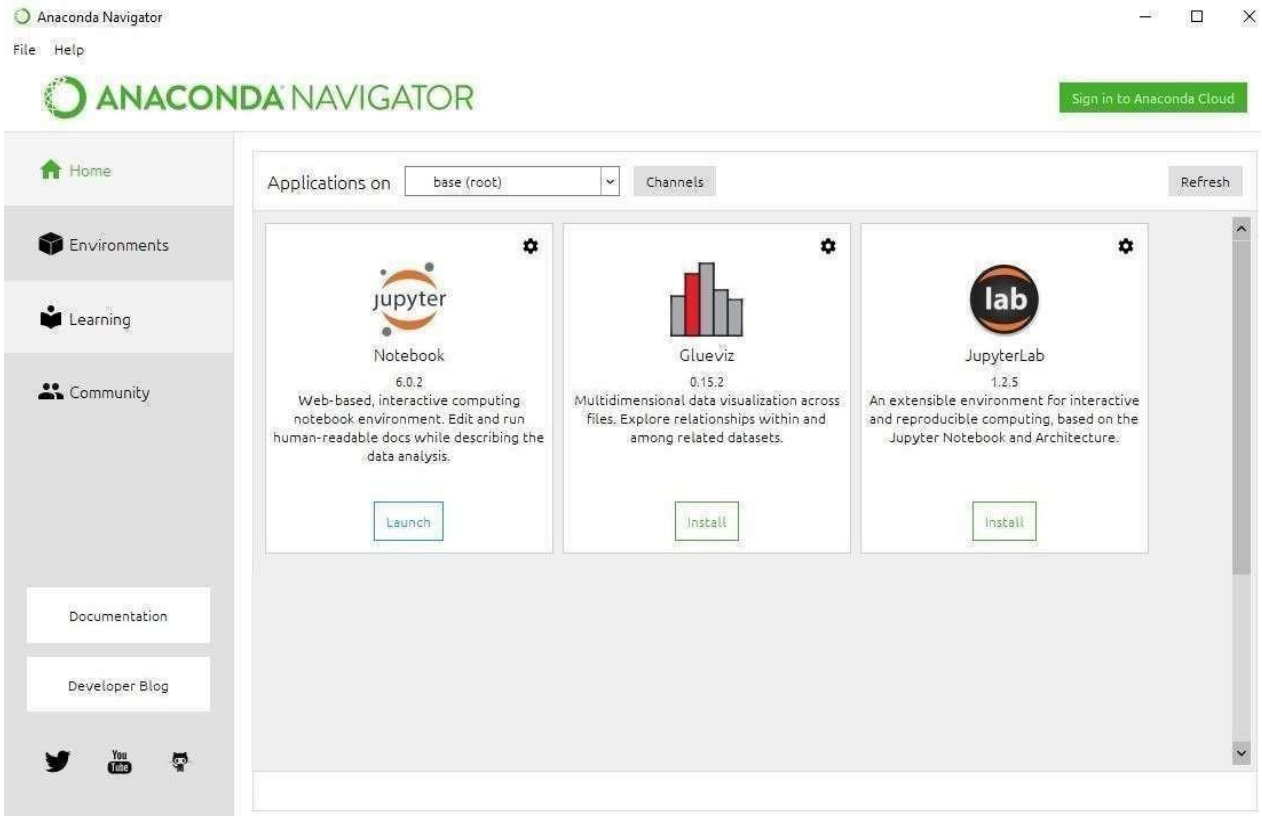
The installation process is begin to Start!



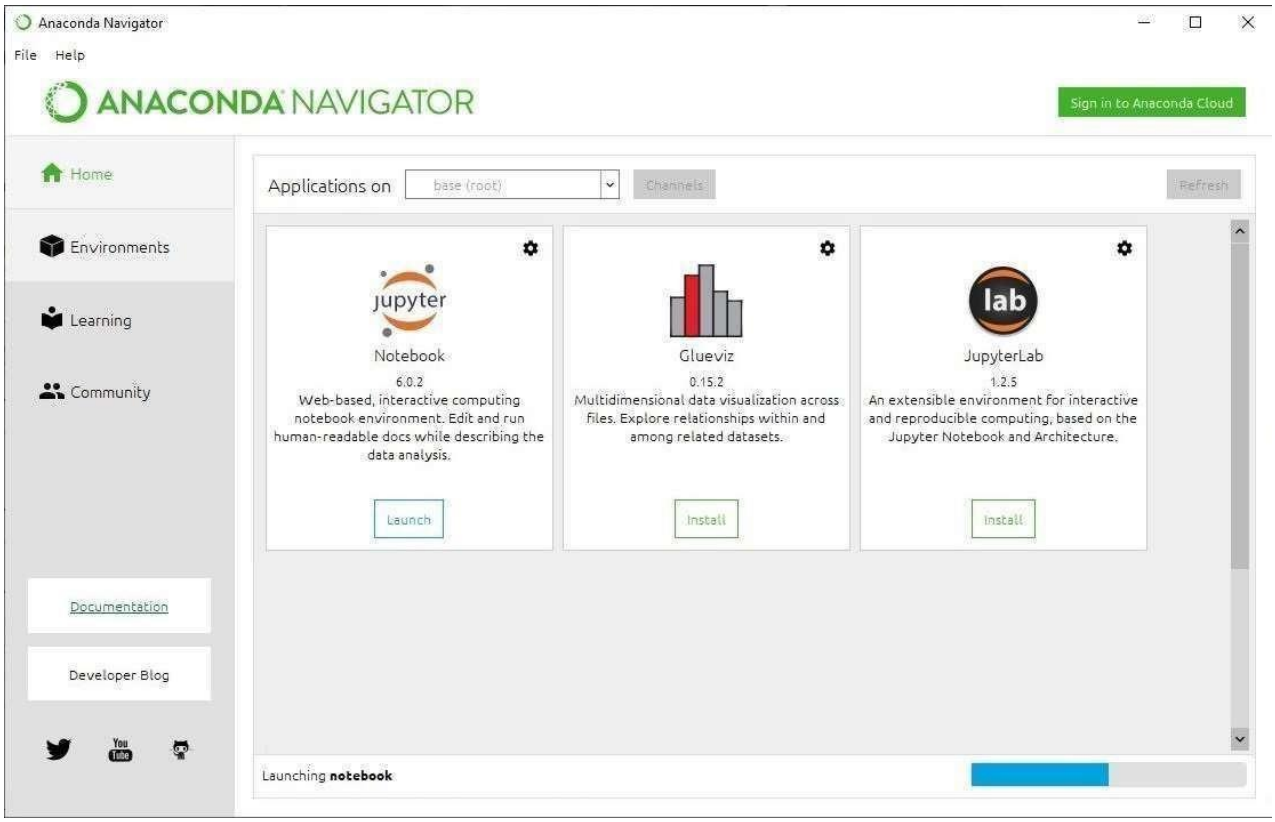
Loading Packages:

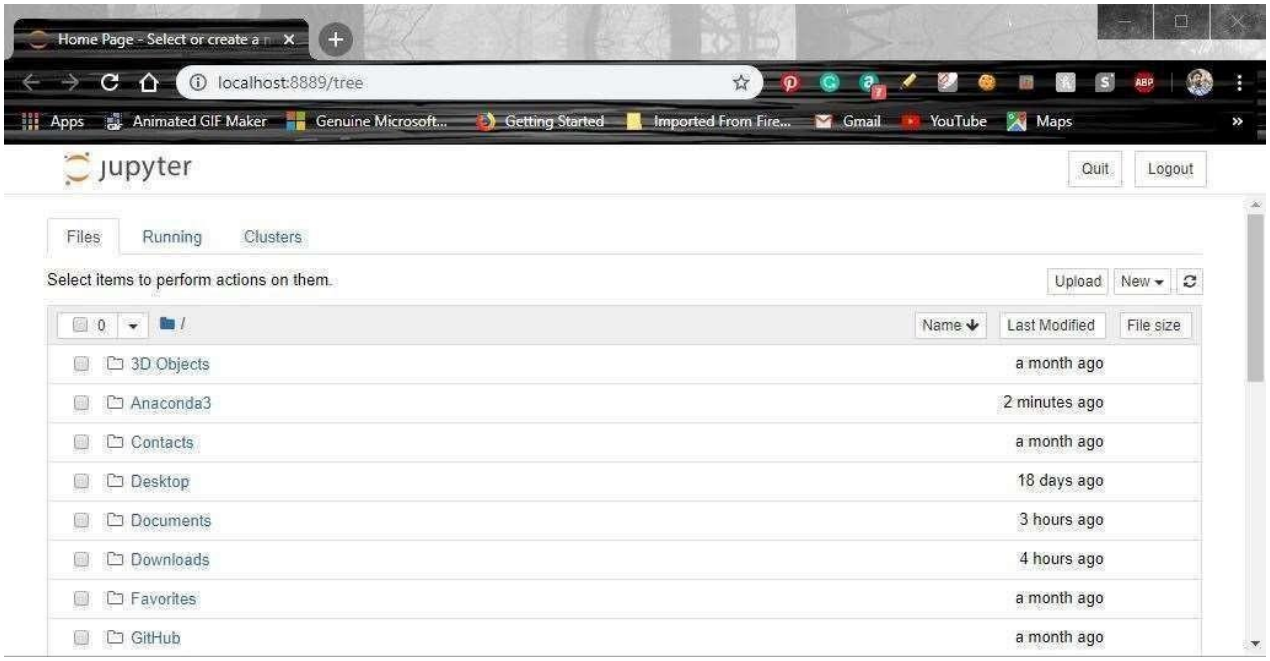


Finished Installation:



Step 3: Now, click on Launch button to Launch the Jupyter.





CHAPTER 8

IMPLEMENTATION

Objective:

The goal of this implementation is to detect phishing attacks based on a dataset containing URLs labeled as either "good" or "bad." We will employ logistic regression and multinomial Naïve Bayes algorithms, utilizing the CountVectorizer method for feature extraction.

Steps:

Import Necessary Libraries:

Import essential Python libraries, including pandas for data manipulation, scikitlearn for machine learning, and associated modules.

```
In [1]: import pandas as pd # use for data manipulation and analysis
        import numpy as np # use for multi-dimensional array and matrix

        import seaborn as sns # use for high-level interface for drawing attractive and informative statistical graphics
        import matplotlib.pyplot as plt # It provides an object-oriented API for embedding plots into applications
        %matplotlib inline
        # It sets the backend of matplotlib to the 'inline' backend:


        from sklearn.linear_model import LogisticRegression # algo use to predict good or bad
        from sklearn.naive_bayes import MultinomialNB # nlp algo use to predict good or bad



        import time
        import plotly.express as px
        from sklearn.model_selection import train_test_split # spliting the data between feature and target
        from sklearn.metrics import classification_report # gives whole report about metrics (e.g, recall,precision,f1_score,c_m)
        from sklearn.metrics import confusion_matrix # gives info about actual and predict
        from nltk.tokenize import RegexpTokenizer # regexp tokenizers use to split words from text
        from nltk.stem.snowball import SnowballStemmer # stemmes words
        from sklearn.feature_extraction.text import CountVectorizer # create sparse matrix of words using regexptokenizes
        from sklearn.pipeline import make_pipeline # use for combining all prerocessors techniuqes and algos


        import networkx as nx # for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

        import pickle# use to dump model

        import warnings # ignores pink warnings
        warnings.filterwarnings('ignore')
```

Figure 4.1.1 Libraries

Load Dataset:

Load the dataset into a pandas DataFrame. Ensure that the dataset includes columns for URLs and corresponding labels (e.g., "good" or "bad").

POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an Excel file format. Don't show again

A7 smilesvoegol.servebbs.org/voegol.php

	A	B	C
1	URL	Label	
2	nobell.it/70ffb52d079109dca5664cce6f317373782/login.SkyPe.com/en/cgi-bin/verification/login/70ffb52d079109dca5664cce6f317373/index.php?cmd=_	bad	
3	www.dghjdgf.com/paypal.co.uk/cycgi-bin/webscrcmd=_home-customer&nav=1/loading.php	bad	
4	serviciosbys.com/paypal.cgi.bin.get-into.herf.secure.dispatch35463256rzr321654641dsf654321874/href/href/href/secure/center/update/limit/seccure/4d	bad	
5	mail.printakid.com/www.online.americanexpress.com/index.html	bad	
6	thewhiskeydregs.com/wp-content/themes/widescreen/includes/temp/promocoessmiles/?84784787824HDJNDJDSJSHD//2724782784/	bad	
7	smilesvoegol.servebbs.org/voegol.php	bad	
8	premierpaymentprocessing.com/includes/boleto-2via-07-2012.php	bad	
9	myxxxcollection.com/v1/js/jih321/bpd.com.do/do/l.popular.php	bad	
10	super1000.info/docs	bad	
11	horizonsgallery.com/js/bin/ssl1/_id/www.paypal.com/fr/cgi-bin/webscr/cmd=_registration-run/login.php?cmd=_login-run&dispatch=1471c4bdb044a	bad	
12	phlebolog.com.ua/libraries/joomla/results.php	bad	
13	docs.google.com/spreadsheet/viewform?formkey=dE5rVEdSV2pBdkpSRy11V3o2eDdwbnc6MQ	bad	
14	www.coincoele.com.br/Scripts/smiles/?pt-br/Paginas/default.aspx	bad	
15	www.henkdeinumboomkwekerij.nl/language/pdf_fonts/smiles.php	bad	
16	perfectsolutionofall.net/wp-content/themes/twentyten/wiresource/	bad	
17	lingshc.com/old_aol.1.3/?Login=&Lis=10&LigertID=1993745&us=1	bad	
18	anonymeidentity.net/remax./remax.htm	bad	
19	dutchweb.gtphost.com/zimbra/exch/owa/uleth/index.html	bad	
20	www.avedeoiro.com/site/plugins/chase/	bad	
21	asladconcentration.com/paplkuk1/webscrcmd=_home-customer&nav=1/	bad	
22	www.regaranch.info/grafika/file/2012/atualizacao/www.itau.com.br/	bad	
23	optimistic-pessimism.com/aoluserupdatealert.info.htm	bad	
24	mercadolivre.com.br.premiosfidelidade2012.com.br/confirmar/	bad	

Figure 4.1.2 Dataset

```
In [2]: %cd "C:\Users\DELL\Downloads\archive (8)"
        C:\Users\DELL\Downloads\archive (8)

In [3]: phish_data = pd.read_csv('phishing_site_urls.csv')
```

Figure 4.1.3 Loading Dataset

Split Dataset:

Divide the dataset into training and testing sets using the train_test_split function from scikit- learn. This ensures that the model is trained on a subset of the data and evaluated on a separate, unseen subset.

Training and Testing

```
In [45]: trainX, testX, trainY, testY = train_test_split(feature, phish_data.Label)
```

Figure 4.1.4 Training and Testing Feature

Extraction - CountVectorizer:

Utilize the CountVectorizer method from scikit-learn to convert URL strings into numerical features. This step involves tokenizing the URLs and representing them as vectors of term frequencies.

CountVectorizer ¶

```
In [42]: cv = CountVectorizer() #CountVectorizer is a scikit-learn tool for converting a collection of text documents into a matrix of tok

In [43]: feature = cv.fit_transform(phish_data.text_sent) #each column represents the count of a unique word in the entire document collec
```

Logistic Regression Model:

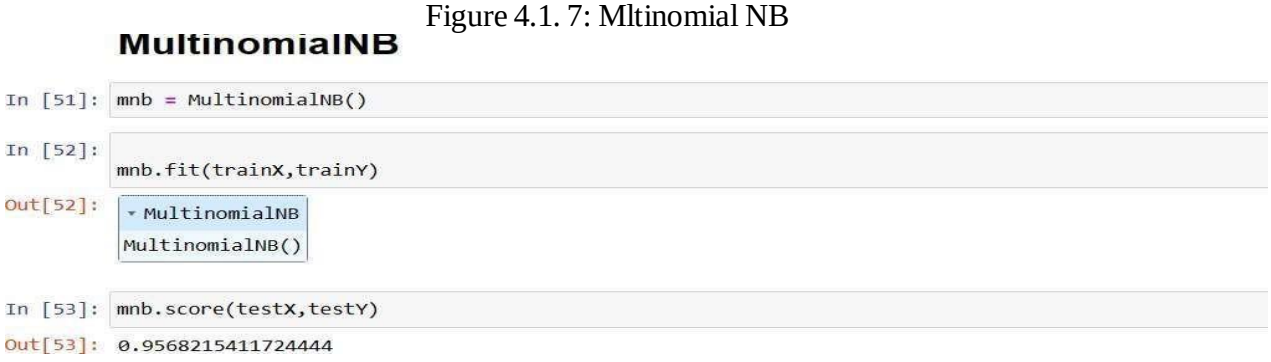
Train a logistic regression model on the training set using the LogisticRegression class from scikit-learn. This model learns to classify URLs as extracted using CountVectorizer.

Figure 4.1.6 Logistic Regression

Multinomial Naive Bayes Model:

Train a multinomial Naive Bayes model on the same training set using the MultinomialNB class from scikit-learn. This model is suitable for working with discrete data, making it suitable for text classification tasks.

Figure 4.1. 7: Mltinomial NB



Assess the performance of the trained models on the testing set. Calculate metrics such as accuracy, precision, recall, and F1 score using functions provided by scikit-learn's metrics module.

```
Training Accuracy : 0.9742044469902357
Testing Accuracy : 0.9568215411724444

CLASSIFICATION REPORT

              precision    recall  f1-score   support

         Bad       0.91      0.94      0.92     38311
        Good       0.97      0.97      0.97     99026

    accuracy                           0.96    137337
   macro avg       0.94      0.95      0.95    137337
weighted avg       0.96      0.96      0.96    137337

CONFUSION MATRIX
```

5]: <Axes: >

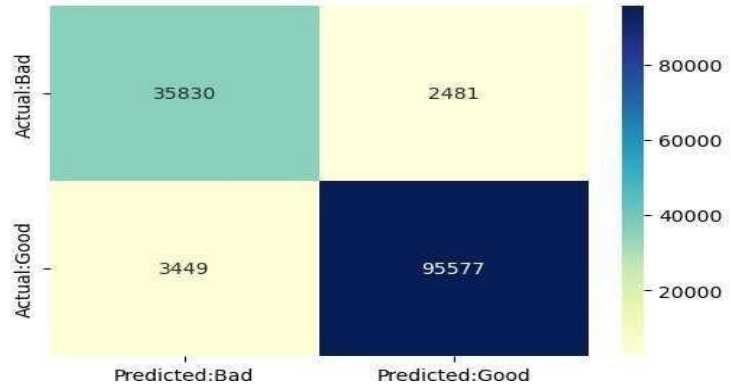


Figure 4.1.8: Classification Report

Note: Ensure that the dataset used for training and testing is representative and appropriately labeled. Fine-tune parameters and explore additional techniques for model optimization based on the dataset's characteristics.This implementation provides a foundation for detecting phishing attacks using machine learning techniques. Further customization and optimization may be necessary based on the specific requirements of the dataset and the desired level of model performance.

CHAPTER 9

PROJECT CODE

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.linear_model import
LogisticRegression from
sklearn.naive_bayes import
MultinomialNB import time
import plotly.express as px
from sklearn.model_selection import
train_test_split from sklearn.metrics
import classification_report from
sklearn.metrics import confusion_matrix
from nltk. tokenize import RegexpTokenizer
from nltk.stem.snowball import SnowballStemmer
from sklearn.feature_extraction.text import Count
Vectorizer from sklearn.pipeline import
make_pipeline
import networkx as nx
import
pickle
import
warning
s
warnings.filterwarnings('ignore') phish_data=pd.read_csv('phishing_site_urls.csv'
phish_data.head()
phish_data.tail()
phish_data.info()phish_data.isnull().sum()
fig=px.bar(label_counts, x=label_counts.index, y=label_counts.Label)
fig.show() tokenizer = Regexp Tokenizer(r[A-Za-z]+') phish_data.URL[0]
tokenizer.tokenize(phish_data. URL[0]) print('Training Accuracy:',Ir.score(trainX,trainY))
print('Testing Accuracy:',Ir.score(testX,testY))
con_mat=pd.DataFrame(confusion _matrix(lr.predict(testX), testY), columns=['Predicted: Bad',
'Predicted:Good'], index= ['Actual: Bad', 'Actual: Good']) print('\nCLASSIFICATION
REPORT\n') print(classification_report(Ir.predict(testX), testY, target_names
=['Bad','Good']))
print("\nCONFUSION MATRIX')
plt.figure(figsize=(6,4)) sns.heatmap(con_mat, annot =True,fmt='d',cmap="YIGnBu")
pickle.dump(pipeline_Is.open('phishing.pkl','wb')) loaded_model
```

```
=pickle.load(open('phishing pkl', 'rb')) result =loaded_model.score(testX,testY)
print(result) predict_bad['yeniik.com.tr/wp-
admin/js/login.alibaba.com/login.jsp.php','fazan- pacir.rs/temp/libraries/ipad',
'tubemoviez.exe', 'svision-
online.de/mgfi/administrator/components/com_babackup/classes/fx29id1.txt')
predict_good['youtube.com/''] loaded_model =pickle.load(open('phishing.pkl',
'rb')) #predict_bad vectorizers.transform(predict_bad) #predict_good
=vectorizer
.transform(predict_good) result loaded_model.predict(predict_bad) result2 =
loaded_model.predict(predict_good) print(result) print("*"*30) print(result2)
```

CHAPTER 10
RESULT

```
predict_bad = ['yeniik.com.tr/wp-admin/js/login.alibaba.com/login.jsp.php','fazan-pacir.rs/temp/librari
predict_good = ['youtube.com/']
loaded_model = pickle.load(open('phishing.pkl', 'rb'))
#predict_bad = vectorizers.transform(predict_bad)
# predict_good = vectorizer.transform(predict_good)
result = loaded_model.predict(predict_bad)
result2 = loaded_model.predict(predict_good)
print(result)
print("*"*30)
print(result2)
```

```
['bad' 'bad' 'bad' 'bad']
******************************
['good']
```

Figure 5.1: output of our project

Algorithms	Training Accuracy	Testing Accuracy
Logistic Regression	97.81%	96.31%
Multinomial NB	97.42%	95.68%

classifica on Report (Logis c Regression):

	Precision	recall	F1-score	Support
Bad	0.90	0.96	0.93	36,792
Good	0.99	0.96	0.97	100,545

Classification Report (Multinomial Naive Bayes):

	precision	recall	F1-score	Support
Bad	0.91	0.94	0.92	38,311
Good	0.97	0.97	0.97	99,026

Both models demonstrate strong performance in detecting phishing attacks, with high accuracy and precision.

Logistic Regression exhibits slightly better performance on the testing set, achieving a 96.31% accuracy.

Multinomial Naive Bayes also performs well, with a testing accuracy of 95.68%

CHAPTER 11

CONCLUSION

In conclusion, the phishing attack detection project has demonstrated the effectiveness of employing machine learning models for identifying malicious URLs. The logistic regression and multinomial Naive Bayes algorithms, coupled with the CountVectorizer method, exhibited robust performance with high accuracy and precision in distinguishing between phishing and legitimate URLs. The project's success highlights the potential of machine learning in bolstering cybersecurity efforts. Ongoing monitoring, continuous model evaluation with real-world data, and exploration of advanced feature engineering techniques are recommended for maintaining and enhancing the models' capabilities over time. Overall, this project provides a valuable foundation for phishing attack detection, contributing to the proactive defense against evolving cybersecurity threats.

CHAPTER 12

REFERENCES

[1] F. Salahdine and N, Kaabouch, “Social Engineering Attacks: ASurvey,” Future Internet J,, 11, 89, pp. 1-17, 2019.

[2] R. Mohammad, F. Thabtah, and L. McCluskey, “Intelligent rule-basedphishing websites classification,” IET Inf. Secur., pp. 153–160, 2014.

[3] F. Salahdine and N. Kaabouch, “Security threats, detection, andcountermeasures for physical layer in cognitive radio networks: Asurvey,” Physical Commun. J., 2020.

[4] J. He and Y. Zhu, “Social engineering/phishing,” Encycl. Soc. Netw. Anal.Min., pp. 1777–1783, 2014.

[5] M. Moghimi and A. Varjani, “New rule-based phishing detection method,”Expert Syst. Appl., vol. 53, pp. 231–242, 2016.

[6] B. Gupta, N. Arachchilage, and K. Psannis, “Defending against phishingattacks: Taxonomy of methods, current issues and future directions,”Telecommun. Syst. 67, 247–267, 2018.

[7] J. Hong, T. Kim, and S. Kim, “Phishing URL detection with lexicalfeatures and blacklisted domains,” Adaptive Auton. Secur. Cyber Syst.,pp. 253-267, 2020.

[8] Y. Huang, Q. Yang, J. Qin, W. Wen, “Phishing URL Detection via CNNand AttentionBased Hierarchical RNN,” IEEE Int. Conf. Trust, Security,Privacy Comput. Commun., pp. 112-119, 2019.

[9] Moghimi M, Varjani AY. New rule-based phishing detection method.Expert systems with applications., 1;53:231-42, 2016.

[10] G. Ramesh, I. Krishnamurthi, and K. Kumar, “An efficacious method fordetecting phishing webpages through target domain identification,”Decision Support Systems, vol. 61, no. ,pp. 12–22, 2014.

[11] Y. Suga, “SSL/TLS servers status survey about enabling forward secrecy,”Int. Conf. Network-Based Information Systems, pp. 501–505, 2014.

[12] A. Albarqi, E. Alzaid, F. Ghamdi, S. Asiri, and J. Kar, “Public keyinfrastructure: A survey,” J. Inf. Secur., vol. 06, no. 01, pp. 31–37, 2015.

[13] S. Krishnamurthy and A. Ve, “Information retrieval models: Trends andtechniques,” Web Semant. Textual Vis. Inf. Retr., pp. 17–42, 2017.

[14] A. Kharraz, W. Robertson, and E. Kirda, “Surveylance: Automaticallydetecting online survey scams,” IEEE Symp. Secur. Privacy, pp. 723–739, 2018.

[15] Y. Reddy and N. Varma, “Review on supervised learning techniques,”Emerg. Res. Data Eng. Syst. Comput. Commun. J., pp. 577-587, 2020.

[16] C. Bircano and N. Arıca, “A comparison of activation functions in artificialneural networks,” Signal Proc. Commun. App. Conf., pp. 1-4, 2018.

[17] Y. Arjoune, F. Salahdine, Md. Islam, E. Ghribi, and N. Kaabouch, “Anovel jamming attacks detection approach based on machine learning forwireless communication,” Int. Conf. Inf. Netw., pp. 1–6, 2020.