A Project Report on

Phishing Website Detection using Machine Learning

submitted in partial fulfillment for the award of

Bachelor of Technology

in

Computer Science & Engineering

by

G. Varun (Y20ACS455) **E. Venkatesh (Y20ACS444)**

CH. Naveen (Y20ACS427) **B. Karthik (L21ACS402)**

Under the guidance of
Prof. M V Pavan Kumar

Department of Computer Science and Engineering
Bapatla Engineering College
(Autonomous)
(Affiliated to Acharya Nagarjuna University)
BAPATLA – 522 102, Andhra Pradesh, INDIA

2023-2024

Department of
Computer Science & Engineering

CERTIFICATE

This is to certify that the project report entitled **Guidelines for the Preparation of Project Thesis Department of CSE** that is being submitted by G.Varun (Y20ACS455), E.Venkatesh (Y20ACS444), CH.Naveen (Y20ACS427), B.Karthik (L21ACS402) in partial fulfillment for the award of the Degree of Bachelor of Technology in Computer Science & Engineering to the Acharya Nagarjuna University is a record of bonafide work carried out by them under our guidance and supervision.

Date:

Signature of the Guide	**Signature of the HOD**
M V Pavan Kumar	**M. Rajesh Babu**
Assistant Professor	**Prof. & Head**

DECLARATION

We declare that this project work is composed by ourselves, that the work contained herein is our own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

G.Varun (Y20ACS455)
E.Venkatesh (Y20ACS444)
CH.Naveen (Y20ACS427)
B.Karthik(L21ACS402)

Acknowledgement

We sincerely thank the following distinguished personalities who have given their advice and support for successful completion of the work.

We are deeply indebted to our most respected guide Mr. **M V Pavan Kumar, Assistant Professor,** Department of CSE, for his/her valuable and inspiring guidance, comments, suggestions and encouragement.

We extend our sincere thanks to **M. Rajesh Babu** , Assistant Prof. & Head of the Dept. for extending his cooperation and providing the required resources.

We would like to thank our beloved Principal **Dr. Nazeer Shaik** for providing the online resources and other facilities to carry out this work.

We would like to express our sincere thanks to our project coordinator **Dr. N. Sudhakar,** Prof. Dept. of CSE for his helpful suggestions in presenting this document.

We extend our sincere thanks to all other teaching faculty and non-teaching staff of the department, who helped directly or indirectly for their cooperation and encouragement.

G.Varun (Y20ACS455)
E.Venkatesh (Y20ACS444)
CH.Naveen (Y20ACS427)
B.Karthik(L21ACS402)

Abstract

Malicious uniform resource locator (URL), i.e., Malicious websites are one of the most common cybersecurity threats. They host gratuitous content (spam, malware, inappropriate ads, spoofing, etc.) and tempt unwary users to become victims of scams (financial loss, private information disclosure, malware installation, extortion, fake shopping site, unexpected prize etc.) and cause loss of billions of rupees each year. The visit to these sites can be driven by email, advertisements, web search or links from other websites. In each case, the user must click on the malicious URL. The rising cases of phishing, spamming and malware has generated an urgent need for a reliable solution which can classify and identify the malicious URLs. Traditional classification techniques like blacklisting and signature matching approach are challenged because of huge data volume, patterns and technology changing over time, along with complicated relationship among features. Here we address the detection of malicious URLs as a binary classification problem and evaluate the performance of several well-known machine learning classifiers. We adopted a public dataset from Kaggle to train the model. The best performing classifier was used to detect malicious URLs. It was found to give better results.

Keywords: Malicious URL, Machine learning, Phishing, Blacklisting, Signature Matching, Spamming, Malware, Spoofing.

Table of Contents

List of Tables

List of figures

1 Introduction

Phishing Uniform Resource Locator (URL) host unsolicited information and attackers use malicious URLs as one of a primary tool to carry out cyber-attacks. E-mail and social media resources such as Facebook, Twitter, WhatsApp, Orkut, etc. are the most commonly used applications to spread malicious URLs. They host unsolicited information on the web page. Whenever an unsuspecting user visits that website unknowingly through the URL, the host may get compromised, making them victims of various types of frauds including malware installation, data, and identity theft. Every year, malicious URLs have been causing billions of dollars' worth of losses. These factors force the development of efficient techniques to detect malicious URLs promptly and give an alert to the network administrator. Most of the commercial products exist in markets are based on the blacklisting method [1]. This method relies on a database that contains a list of malicious URLs. The blacklists are continually updated by the anti-virus group through scanning and crowdsourcing solutions. The blacklisting method can be used to detect the malicious URLs which are already present in the database. But they completely fail to detect the variants of the existing malicious URLs or entirely new malicious URLs. In recent days, cyber attackers follow mutation techniques to generate several variants of existing malware. To cope with this, machine learning techniques are employed. In recent days, the most used approach is applying domain knowledge to extract lexical features of URL, followed by applying machine learning models.

1.1 Problem Statement

URL has been used and misused a lot to exploit the vulnerability of the user. This paper focusses on classification of any URL as benign or malicious. Furthermore, it compares the results of the multiple machine learning classification techniques such as Gradient Boost, Logistic Regression (LR), Random Forest (RF), Naïve Bayes (NB), and K-Nearest Neighbors (KNN), Support Vector Classifier, Decision Tree, XGBoost. The best performing classifier is used to detect malicious websites [2].

1.2 What is a Malicious URL?

In simple words, a malicious URL is a clickable link that directs users to a malicious or otherwise fraudulent web page or website. As the name suggests, nothing good can ever come out of a malicious URL. That's because the goal of creating these bad site pages is typically for a nefarious purpose such as to carry out a political agenda, steal personal or company data, or make a quick buck. For example, cybercriminals may create malicious URLs to:

1. Carry out phishing attacks to gain access to users' personal information to carry out identity theft or other types of fraud.

2. Gain access to users' login credentials to gain access to their personal or professional accounts.

3. Trick users into downloading malicious software that cybercriminals can use to spy on victims or take over their devices.

4. Get into victims' computers to encrypt their files for a ransomware attack.

1.3 Machine Learning

Machine learning is a growing technology which enables computers to learn automatically from past data. Machine learning uses various algorithms for building mathematical models and making predictions using historical data or information. Currently, it is being used for various tasks such as image recognition, speech recognition, email filtering, Facebook auto-tagging, recommender system, and many more. The below Figure 1.1 shows the working of ML Algorithm. At a broad level, machine learning can be classified into three types:

1. Supervised learning

2. Unsupervised learning

3. Reinforcement learning

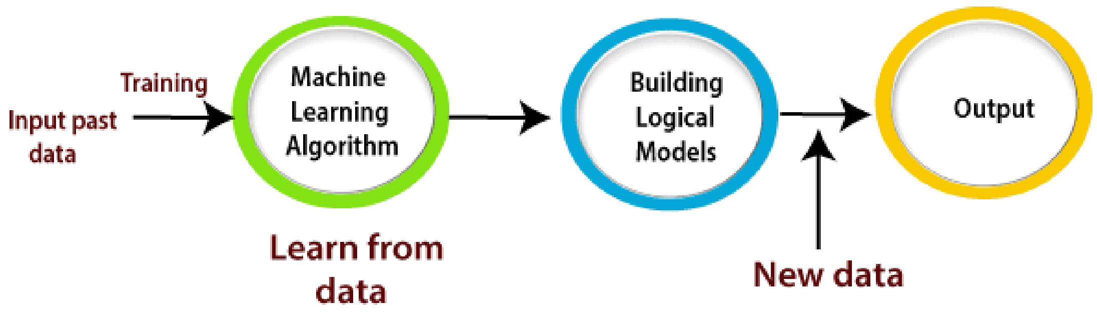


Figure 1.1Working of ML Algorithm

1.3.1 Supervised Learning Approach

Supervised learning is a type of machine learning method in which we provide sample labeled data to the machine learning system in order to train it, and on that basis, it predicts the output. The system creates a model using labeled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not. The goal of supervised learning is to map input data with the output data. The supervised learning is based on supervision, and it is the same as when a student learns things in the supervision of the teacher.

1.3.2 Unsupervised learning

Unsupervised learning, on the other hand, uses unlabeled data. The reason for using unlabeled instances can be that we do not know the correct label, or we try to find a better label. This type of learning has a broad range of uses. Amongst them are clustering, dimensionality reduction, outlier detection, association rule mining. An example of a clustering task is customer grouping. We would like to identify groups of people with similar purchasing patterns, but we might not have identified these patterns yet. Similarly, we might not know how many there are beforehand. Some of the well-known unsupervised algorithms are K-Means and Principal component analysis (PCA).

1.3.3 Reinforcement learning

Reinforcement learning uses an agent to learn a policy by receiving rewards or penalties. An agent is a system that observes and interacts with the environment it has been deployed to. Based on the actions it takes, it receives rewards or penalties. The goal is to maximize rewards by learning the best strategy, a policy.

2 Literature Survey

2.1 Feature Rich Model for Predicting Spam Emails

Authors: Tran, K. N., Alazab, M., & Broadhurst, R

Spam emails are increasingly include malicious URLs & attachments. Malicious URLs & attachments aim towards spread software that can jeopardize a computer's security. Additionally, these dangerous attachments make an effort towards hide their contents from virus scanners that are typically employed through email services towards check considering such dangers. Malicious URLs give an additional layer about cover, as email content attempts towards persuade recipient towards click on a URL that directs them towards a malicious website or downloads a harmful file. In this study, we report our preliminary research on identifying kind about spam email most likely towards contain these extremely risky spam emails, based on two real-world data sets. We demonstrate that these variables may accurately predict dangerous URLs & attachments within an area under precious recall curve (AUC-PR) about up towards 95.2 percent. Our efforts can lessen need considering URL blacklists & virus scanners, which frequently don't update as quickly as dangerous information they're trying towards catch. Such techniques could decrease numerous resources currently required towards identify dangerous information.

2.2 Malicious Spam Emails Developments

Authors:Alazab,M,Layton,R.,Broadhurst.

Internet is a decentralized system that allows considering quick communication, has a wide audience, & gives anonymity-a quality that makes it very useful considering carrying out criminal actions. Cybercrime has swiftly changed from a relatively low volume crime towards a widespread, large volume crime in tandem among growth about Internet. dissemination about spam emails, where email's content tries towards persuade recipient towards click a URL leading towards a malicious Web site or download a malicious attachment, is a common illustration about this type about crime. In this study, we report our preliminary research on identifying kind about spam email. Overwhelming amount about spam that is sent out every day makes it difficult considering analysts towards provide intelligence on spam activities; as a result, information they do gather only represents a small portion about overall picture.

Our exploratory research examines utility about utilizing authorship-based models considering this purpose while prior studies have looked at automating parts about these analyses using topic-based models, i.e. classifying email clusters into groups among comparable topics. In initial stage, we used an authorship-based clustering method towards group a collection about spam emails. We examined such clusters using a variety about linguistic, structural, & syntactic criteria in subsequent step. In previous authorship study, this issue about high purity among low recall has been present. Although this is a shortcoming about our study as well, clusters themselves are still helpful considering automated analysis because they minimize amount about work required. Our second phase uncovered pertinent data about organization that can be used in future studies considering a deeper examination about such groups, such as tracing further connections behind spamming activities.

2.3 An Analysis about Groups Engaged in Cyber Crime

Authors: Broadhurst, R., Grabosky, P., Alazab, M., Bouhours, B., & Chon, S,

This essay investigates characteristics about cybercriminal organizations. Concept & breadth about cybercrime are briefly discussed, as well as theoretical & empirical difficulties in addressing what is known about cyber criminals & likely contribution about organized crime organizations. Study provides examples about well-known cases that demonstrate motivations about typical offenders, including state actors, as well as individual & collective behavior. Using typology proposed through McGuire, various forms about criminal organization & cybercrime are described (2012). There are obviously many different organizational structures involved in cybercrime. There seems towards be a need considering leadership, organization, & specialization in business- or profit-oriented operations, particularly cybercrime committed through governmental actors. Contrarily, protest activity is typically less planned & has a weak or non-existent chain about command.

2.4 Zero-Day Malware Detection Based on Supervised Learning Algorithms

Authors: Alazab, M., Venkatraman, S., Watters, P., & Alazab, M

Code obfuscation techniques are used towards construct zero-day or unidentified malware through altering parent code towards make offspring copies that have same functionality but different signatures. Current methods described in literature are insufficiently accurate & effective towards identify zero-day malware. We have presented & assessed a novel approach in this work that makes use about a variety about data mining techniques towards quickly & accurately identify & categorize zero-day malware based on frequency about Windows API calls. This study uses a fully automated tool created specifically considering this study towards carry out various experimental investigations & evaluations, & it describes methodology used considering collection about large data sets towards train classifiers. It also analyses performance results about different data mining algorithms adopted considering study.

We are able towards analyze & discuss benefits about one data mining technique over other considering successfully detecting zero-day malware through performance outcomes about these algorithms from our experimental research.

Learning process considering data mining system used in this study involves examining behavior about both harmful & benign algorithms in big datasets. We used a number about reliable classifiers, including Naive Bayes (NB) Algorithm, K-Nearest Neighbor (KNN) Algorithm, Sequential Minimal Optimization (SMO) Algorithm among four different kernels (SMO-Normalized PolyKernel, SMO - PolyKernel, SMO Puk, & SMO Radial Basis Function (RBF)), and Backpropagation Ne Eds. Reproduction considering academic & nonprofit reasons is allowed as long as this text is present Overall, automated data mining approach used in this work has produced results among low false positive (FP) rates about less than 0.025 & high true positive (TP) rates about more than 98.5 percent, both about which have not previously been seen in literature. This indicates that our unique technique is a significant advancement in identifying zero-day malware because it is significantly higher than necessary commercial acceptance level. This article also suggests future study directions considering examining various obfuscations that now influence IT industry.

3 System Analysis

System analysis is a review of a technological system, like a software package, for troubleshooting, development, or improvement purposes. With a systems analysis, considering the goals of the system is important for solving problems and creating efficiencies. From there, dividing a system into components can make it easier to perform individual analyses that influence the complete system. In this chapter, we discuss about the existing system, proposed system, system requirements and the classification techniques.

3.1 Existing System

A malicious website, often known as a malicious URL, is main platform considering hosting unsolicited content. It is crucial towards quickly identify dangerous URLs. Blacklisting [1], regular expression [3], & signature matching techniques [4] have all been employed in earlier investigations. These methods are utterly useless considering identifying new URLs, malicious URL variants, or URLs that have never been seen before. Through suggesting a machine learning-based solution, this problem can be minimized.

3.1.1 Disadvantages about Existing System

1. Ineffective
2. Less accuracy models

3.2 Proposed System

URL has been used and misused a lot to exploit the vulnerability of the user. This paper focusses on classification of any URL as benign or malicious. Furthermore, it compares the results of the multiple machine learning classification techniques such Gradient Boost, Logistic Regression (LR), Random Forest (RF), Naïve Bayes (NB), and K-Nearest Neighbors (KNN), Support Vector Classifier, Decision Tree, XGBoost. The best performing classifier is used to detect malicious websites. The proposed framework has 6 stages:

1. Data Collection: A labelled dataset of malicious and benign websites is collected from the Kaggle repository.

2. Data Pre-processing: Pre-processing includes extraction of additional features, normalisation, encoding of categorical values, standardisation of values and handling of missing data.

3. Model Training: The model is trained using different machine learning techniques such as Logistic Regression, Multinomial Naïve Bayes, XGBoost and K- Nearest Neighbours (KNN) on 80% of the data.

4. Model Testing and Optimisation: Trained model is tested on the remaining 20 % of the data. Hyper parameters are tuned to increase accuracy, precision, and recall.

5. Model Comparison: The machine learning classification techniques are compared based on evaluation metrics.

6. Model Prediction: We make predictions using the algorithm with the most accuracy.

3.2.1 Advantages about Proposed System

1. Effective
2. High performance

3.3 System Requirements

System requirements are the configuration that a system must have in order for a hardware or software application to run smoothly and efficiently. Failure to meet these requirements can result in installation problems or performance problems. The former may prevent a device or application from getting installed, whereas the latter may cause a product to malfunction or perform below expectation or even to hang or crash.

3.3.1 Software Requirements

1. **Technologies used:** Python, Machine Learning
2. **Programming Languages:** Python - sklearn, flask, GradientBoostClassifier
3. **Operating System:** Windows (64bit operating system)
4. **IDE:** Jupyter Notebook, VS Code
5. **Version:** Python 3.8

3.3.2 Hardware Requirements

1. **Processor:** Any Dual core processor
2. **RAM :** 4GB (min)
3. **Hard Disk :** 128 GB (min)

3.4 Classification Techniques

We proposed different algorithms namely Logistic Regression, KNN, Naive Bayes Random Forest algorithms [6], Gradient Boost Classifier, Support Vector Machine, and Decision Tree.These algorithms will give different accuracies based on the dataset are listed here in the Table 3.1 Algorithm and Accuracy. It shows how accurate the algorithms works and we identified that Gradient Boost algorithm works more accurately.

Table 3.1 Algorithm and Accuracy

ML Models	**Accuracy**
Gradient Boosting Classifier	0.974
Random Forest	0.971
Support Vector Machine	0.964
Decision Tree	0.958
K-Nearest Neighbors	0.956
Logistic Regression	0.934
Naive Bayes Classifier	0.605
XGBoost Classifier	0.549

3.4.1 Logistic Regression

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

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.

Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.

In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1). The curve from the logistic function indicates the likelihood of something. The sigmoid function or logistic 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.

Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets. Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The following Figure 3.1 demonstrates the graph of logistic regression.

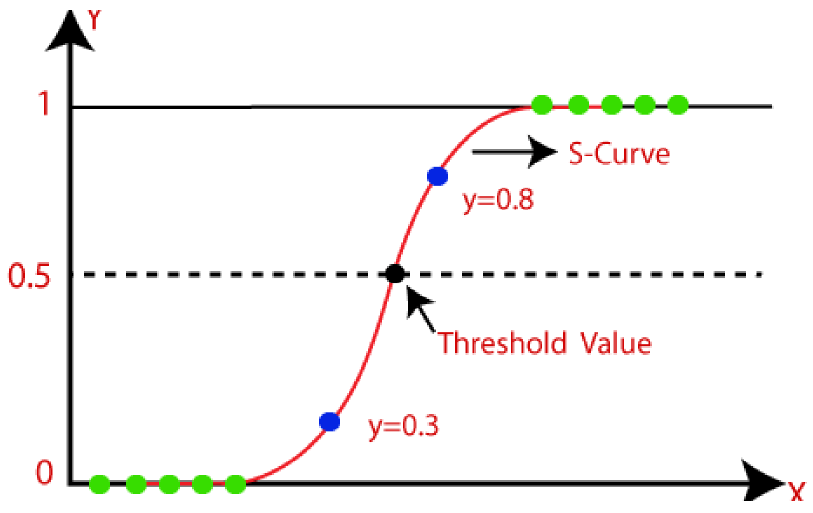


Figure 3.1 Logistic Regression Graph

3.4.2 Naïve Bayes

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems. It is mainly used in text classification that includes a high-dimensional training dataset.

Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions. It is a probabilistic classifier, which means it predicts based on the probability of an object. Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

3.4.3 Random Forest

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. The below Figure 3.2 depicts the working of random forest algorithm.

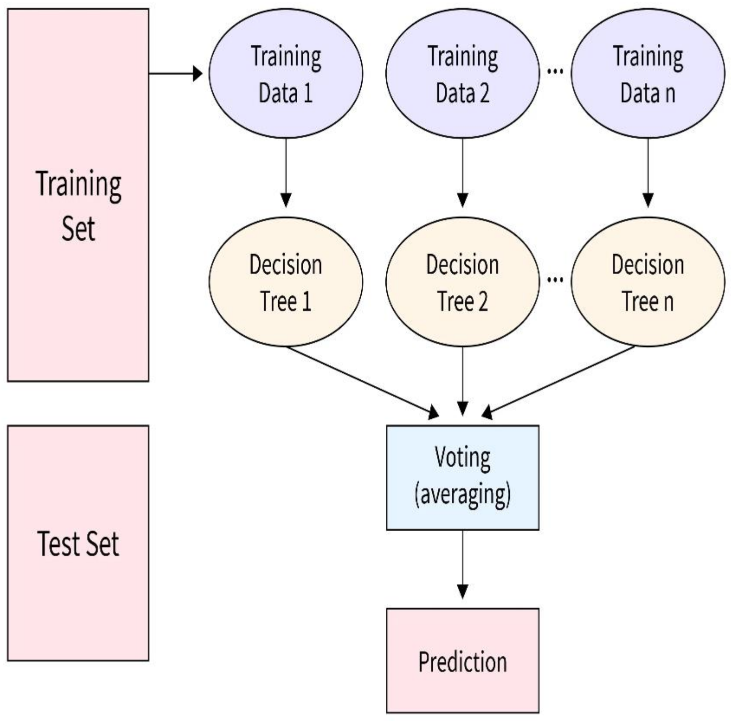


Figure 3.2 Working of Random Forest

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase. The working process can be explained in the below steps and diagram:

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points.

Step-3: Choose the number N for decision trees that you want to build.

Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

3.4.4 XGBoost

XGBoost is one of the gradient boosting algorithms; known for its effectiveness in predictive modelling, mostly with structured or tabular data. It builds trees sequentially to correct errors from prior trees while trying to optimize the specified loss function. Regularization techniques are applied to avoid overfitting, like shrinkage (learning rate) and tree pruning. It is provided with efficient and parallel scalable multiprocessor support in multi-threading and distributed frameworks like Apache Spark. Starting from now, XGBoost is hugely respected as a top algorithm in machine learning competitions and also shows high efficiency in various real-world tasks. XGBoost handles regression, classification, multiple-class classification, and ranking tasks; it has built-in support for missing data and feature importance analysis.

3.4.5 Support Vector Machine

Support Vector Machine (SVM) is a very good model for supervised learning in classification or regression tasks. SVM tries to find a hyperplane best separating data into different classes or making predictions of continuous outcomes. SVM works by mapping data into a high-dimensional feature space so that it finds an optimal hyperplane with a maximum margin that maximizes the distance from the closest data points between classes. For kernel function, it is through it that the SVM has the capability of handling linearly separable data or data that is not separable linearly. SVMs are quite robust against the overfitting problem of a high-dimensional space and perform very well when the dataset used has a complex decision boundary. For these reasons, due to their flexibility and good theoretical foundation, they find applications in many areas, from image classification to text categorization, up to bioinformatics.

3.4.6 Decision Tree

Decision Tree is the state-of-the-art, very flexible, and intuitive method to solve both classification and regression tasks. It divides the dataset into subsets based on features recursively, splitting the data best by a chosen criterion like Gini impurity or information gain. The process builds a structure that can be compared to a tree, where each internal node represents a decision on an attribute and each leaf node represents the value. The biggest pro of decision trees is their high interpretability and visualizability, which helps users understand the relevance of features and provides insight into how models make decisions. It is prone to overfitting, especially when it is pitted against complex datasets or deep trees, which could be mitigated by the use of pruning techniques or ensemble methods like Random Forests. However, that limitation aside, Decision Trees are the most used method in many domains, just for their properties: simplicity, flexibility, and the possibility of managing both numeric and categoric data.

3.4.7 K-Nearest Neighbors

K-Nearest Neighbors (KNN) is one of the very basic yet powerful algorithms used for both classification and regression tasks in supervised learning. It belongs to the method of classification for points of new input data in the feature space with respect to the class label or value of the k-closest data points. KNN works based on the principle of similarity and metrics of distance between elements, such as Euclidean distance. It is very simple to understand and easy to implement, hence befitting a very wide array of applications. One of the major weaknesses is the performance of KNN, which is very sensitive to the choice of K and the distance metric. It could be time-consuming due to the computationally expensive nature, especially when dealing with large datasets. KNN provides good performance if the regular nature of the decision boundaries does not matter, so it is applicable to a recommendation system, anomaly detection, or pattern recognition.

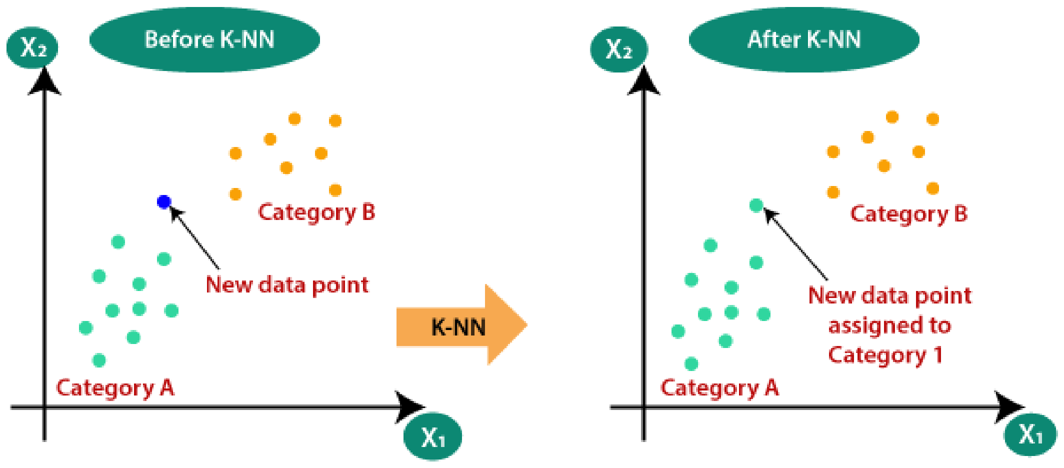


Figure 3.3 KNN Algorithm

3.4.8 Gradient Boost Classification

Gradient Boosting Classification is an ensemble learning method that combines the predictions of several base estimators, usually decision trees, to improve predictive accuracy. It works by sequentially fitting new models to the residuals of the previous models, thereby reducing the errors in prediction at each step. Gradient boosting is robust to outliers and noisy data and can handle a variety of data types. However, it tends to be computationally expensive and may require careful tuning of hyperparameters to prevent overfitting. Despite these challenges, gradient boosting often achieves state-of-the-art performance on many classification tasks and is widely used in both industry and academia.

4 Dataset Description

The Dataset we used in our project contains 32 columns with 30 optimized features of URL. It contains three different labels where 1 means legitimate, 0 means suspicious and-1 means phishing [7]. The Description of the features are as follows: The Dataset we used in our project contains 32 columns with 30 optimized features of URL. It contains three different labels where 1 means legitimate, 0 means suspicious and-1 means phishing [7].

The data set borrowed from https://www.kaggle.com/eswarchandt/phishing-website-detector .A collection of website URLs for 11000+ websites. Each sample has 30 website parameters and a class label identifying it as a phishing website or not (1 or -1).The overview of this dataset is, it has 11054 samples with 32 features. Download the dataset from the link provided. The Description of the features are as follows:

1.Using IP:

This is the column to check whether access to the website is done through the IP address, not the domain. Websites viewed through IP do not have reputation or branding, which is always associated with domain names. The website may also look unprofessional to build trust. Websites that use IP addresses may not be easily accessed and seem therefore less established and credible than those that have domain names.

2.LongURL:

The LongURL column signifies the length of the website's URL. A long URL would probably verify that the site has a complex structure, or it might point at a dynamic content-generation system. Long URLs, indeed, affect the user experience, as it is harder to remember, type by mouth, or enter manually. Very long URLs could even be seen in a negative light by search engines and adversely affect the SEO outcome. Managing long URLs effectively is crucial for maintaining website usability and search engine visibility.

3.ShortURL:

This column denotes whether the website employs shortened URLs. The usual use for shortened URLs is to take very long links and make them short enough for general usage in sharing on social media or in very limited spaces, like emails or messages with a limited amount of characters. While shortening does improve the former two points, it can obscure the final destination and raise potential concerns about security. Shortened URLs may, in one way or the other, thwart even SEO efforts since they hide keyword-rich URLs that would have been indexed for search placement.

4.Symbol@:

The Symbol@ column indicates the presence of the "@" symbol in the website's URL. The "@" character is very common in a URL when it relates to an email address, and therefore, its presence can give a clue to the existence of email-based functions or, at times, the user authentication mechanism. However, its occurrence in a different origin URL would suggest something quite exceptional or even hint at an insecurity. In a URL, the "@" symbol is very important to scrutinize because it can allow understanding the structure, functioning, and possibly even the security posture of the website.

5.Redirecting//:

This column flags whether the website utilizes double forward slashes in its URL redirection process. This method of URL redirection generally has double forward slashes in the protocol-relative URLs, under which the sites will be able to maintain consistency in the protocol while doing redirection between HTTP and HTTPS version sites. However, an incorrect or misconfigured use of double forward slashes in redirections exposes general problems, ranging from broken links or duplicate content to even more serious security issues. It is an important check to verify that there are no double forward slashes within the URL redirections, which could compromise website integrity and user experience.

6. PrefixSuffix-:

The PrefixSuffix- column indicates the presence of prefixes or suffixes in the website's URL. These prefixes or suffixes could include additional strings appended to the domain name, potentially altering the perception or interpretation of the website's identity. Such modifications might be employed for branding purposes, geographical targeting, or to differentiate between versions of the website. However, excessive or arbitrary use of prefixes or suffixes could lead to confusion among users and may affect search engine indexing and ranking.

7.Subdomains:

This column reflects the number of subdomains associated with the website's domain name. Subdomains are subdivisions of the main domain, allowing organizations to create separate web addresses for specific sections or functionalities of their website. Common examples include "blog.domain.com" or "shop.domain.com." The presence of multiple subdomains can indicate a diverse website structure or the implementation of specialized services. Managing subdomains effectively is crucial for maintaining website organization, accessibility, and SEO performance.

8.HTTPS:

The HTTPS column indicates whether the website uses HTTPS encryption for secure communication over the internet. HTTPS encrypts data exchanged between the user's browser and the website, providing confidentiality and integrity protection. Websites employing HTTPS demonstrate a commitment to security and user privacy, as sensitive information such as login credentials, payment details, and personal data are safeguarded against interception or tampering. The adoption of HTTPS is also favored by search engines, potentially enhancing the website's visibility and trustworthiness in search results.

9.DomainRegLen:

DomainRegLen represents the length of time that the website's domain name has been registered. The registration length is an important factor in assessing the website's credibility, stability, and long-term commitment to its online presence. Longer domain registration periods are often associated with established businesses or organizations, indicating reliability and trustworthiness. Conversely, short registration periods may raise concerns about the website's legitimacy or transient nature. Monitoring domain registration lengths is essential for evaluating website longevity and mitigating risks associated with domain expiration or ownership changes.

10.Favicon:

The Favicon column indicates the presence of a favicon, a small icon associated with the website displayed in the browser tab or next to the URL in bookmarks. Favicons serve as visual identifiers, enhancing brand recognition and user experience. Websites with favicons appear more polished and professional, contributing to a positive first impression. Additionally, favicons can improve website navigation by enabling users to distinguish between multiple open tabs or bookmarks at a glance. Ensuring the presence and consistency of favicons across web pages reinforces branding efforts and reinforces user engagement.

11.NonStdPort:

This column indicates whether the URL uses a non-standard port number. Standard ports for web traffic are:

HTTP: Port 80

HTTPS: Port 443

If the port number in the URL is anything other than 80 (for HTTP) or 443 (for HTTPS), this column will likely be flagged as "Yes" or contain a value indicating a non-standard port. Phishing attempts may sometimes use non-standard ports to bypass security measures.

12.HTTPSDomainURL:

This column likely focuses on whether the domain itself uses HTTPS encryption. This is different from the standard HTTPS protocol, which refers to the entire secure connection. Here, HTTPSDomainURL might indicate if the domain name itself explicitly mentions HTTPS within the text, which can be a tactic used in phishing attempts to appear legitimate.

13.RequestURL:

This column most likely refers to the complete URL requested by the browser when loading a webpage. It includes all the parts of the URL, such as protocol (http:// or https://), domain name, path, and query string. Analyzing request URLs can help identify suspicious patterns or hidden elements within a URL used for phishing.

14.AnchorURL:

This column likely refers to the URLs associated with anchor tags (a.k.a. links) embedded within the webpage itself. These are the URLs users would be directed to if they click on a link on the webpage. By examining anchor URLs, the data set can identify inconsistencies or redirects that might be red flags for phishing attempts.

15.LinksInScriptTags:

This column indicates whether the webpage contains links embedded within its script tags. Script tags are used to add dynamic functionality to webpages. While legitimate websites can use them for various purposes, some phishing attempts might embed malicious links within scripts to redirect users or inject harmful code. The presence of links in script tags can be a potential indicator of phishing activity.

16.ServerFormHandler:

Server-side forms: These are standard web forms where users enter information (like login credentials or personal details) that gets submitted to the website's server for processing. The presence of server forms itself isn't necessarily suspicious. Phishing websites might use forms to steal user information. Analyzing this feature alongside others (like the website's legitimacy) can help identify red flags.

17.InfoEmail:

Contact information: Legitimate websites often display contact information, including email addresses.Phishing attempts might display fake contact information, including email addresses, to appear more trustworthy. The data set likely analyzes this alongside other factors to assess legitimacy.

18.AbnormalURL:

Standard URLs: URLs typically follow a format with a protocol (http:// or https://), domain name, path, and sometimes a query string. Phishing URLs might contain unusual characters, excessive subdomains, or other irregularities to bypass security checks or appear more complex than they are. This column likely flags URLs that exhibit such abnormalities.

19.WebsiteForwarding:

Legitimate forwarding: Some websites might forward users for various reasons, like redirects based on location or temporary maintenance. Phishing attempts might use website forwarding to take users to a malicious website designed to steal information. This column helps identify potential redirects that could be part of a phishing scheme.

20.StatusBarCust:

Status bar: The browser's status bar typically displays information like the current URL and loading progress.Some phishing attempts might try to customize the status bar to display fake messages or manipulate how information is presented. This column can be a clue for identifying such attempts.

21.DisableRightClick:

Right-click menu: Right-clicking typically brings up a context menu with options to inspect the element, save the page, or copy links. Phishing websites might disable right-click functionality to prevent users from easily accessing these options. This could hinder users from inspecting the website's source code or saving the page for later verification, making it harder to identify the attempt as fraudulent.

22.UsingPopupWindow:

Pop-ups: Legitimate websites might use pop-ups for various purposes, like login modals or advertisements. Phishing attempts might use aggressive pop-ups to display fake messages, urgency tactics, or attempt to steal information through forms within the pop-up. This column helps identify websites that heavily rely on pop-ups, which can be a suspicious sign.

23.IframeRedirection:

Iframes: Iframes are HTML elements that embed another webpage within the current one. Phishing attempts might use iframes to redirect users to malicious content or hide the actual source of the content being displayed. This column helps identify websites that use iframes in a way that could be suspicious.

24.AgeofDomain:

Established websites: Legitimate websites typically have domain names registered for a longer period, indicating a more established presence. Phishing attempts might use newly registered domain names to appear more legitimate or avoid detection. While not a definitive indicator, a very young domain age can be a red flag.

25.DNSRecording:

Domain Name System (DNS). The DNS translates domain names into IP addresses for web browsers.DNS records: Legitimate websites typically have valid DNS records that allow browsers to locate them. Phishing attempts might use domain names that don't have proper DNS records, making them harder to track or taking advantage of typosquatting (similar-sounding domain names). The absence of a DNS record can be a potential indicator of a suspicious website.

26.Website Traffic:

This feature measures the volume of visitors a website receives over a specific period. It provides insights into the popularity and reach of a website, indicating its level of engagement and effectiveness in attracting users. Analyzing website traffic helps in understanding audience behavior, identifying trends, and optimizing marketing strategies to enhance visibility and conversions.

27.PageRank:

PageRank is an algorithm used by search engines, particularly Google, to assess the importance of web pages based on the quantity and quality of links pointing to them. It assigns a numerical value to each page, influencing its ranking in search engine results. Higher PageRank indicates greater authority and relevance, which can improve a page's visibility and organic traffic.

28.GoogleIndex:

This feature refers to the collection of web pages that Google has discovered and stored in its database for retrieval in search results. It signifies the extent to which a website's pages are accessible and recognized by Google's crawlers. Being indexed by Google is crucial for visibility, as it enables web pages to appear in search results when users query relevant keywords.

29.LinksPointingToPage:

Links pointing to a page represent the number of other web pages that reference and direct users to a specific page on a website. These inbound links are essential for search engine optimization (SEO) as they contribute to a page's authority, relevance, and ranking in search results. Analyzing the quantity and quality of links pointing to a page helps in assessing its online visibility and influence within its niche.

30.StatsReport:

A Stats Report is a comprehensive summary of various metrics and data related to website performance and online presence. It typically includes information on website traffic, user demographics, engagement metrics, conversion rates, and other key performance indicators (KPIs). Stats reports are valuable for evaluating the effectiveness of marketing efforts, identifying areas for improvement, and making informed decisions to optimize online strategies.

5 Design

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.

5.1 UML Diagrams

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. 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. The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.

2. Provide extendibility and specialization mechanisms to extend the core concepts.

3. Be independent of programming languages and development process.

4. Provide a formal basis for understanding the modeling language.

5. Encourage the growth of OO tools market.

6. Support higher level development concepts such as collaborations, frameworks, patterns and components.

7. Integrate best practices.

5.1.1 Use Case Diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The below Figure 5.1 shows the use case diagram.

It is essential to analyze the whole system before starting with drawing a use case diagram, and then the system's functionalities are found. And once every single functionality is identified, they are then transformed into the use cases to be used in the use case diagram.

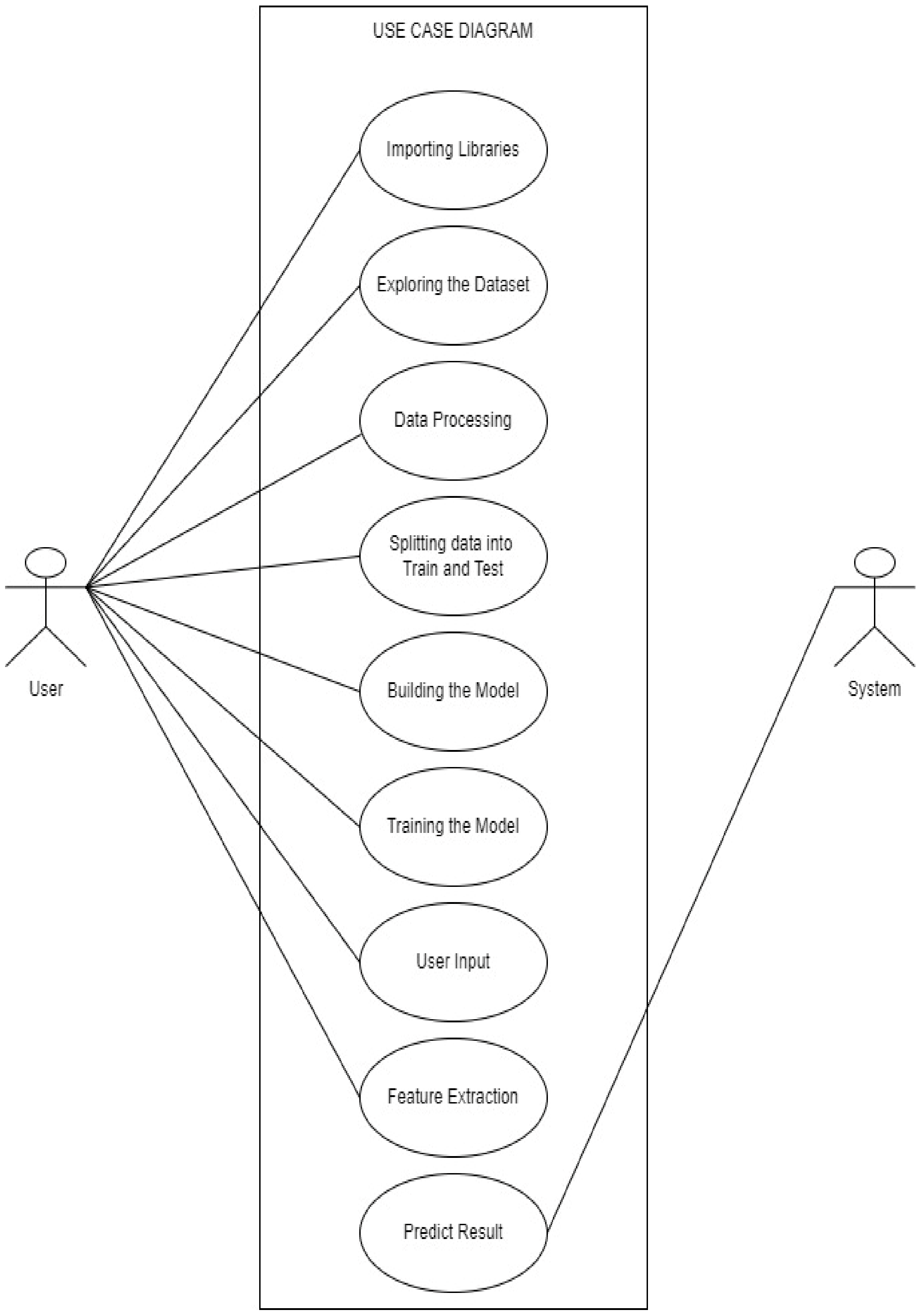


Figure 5.1UseCase Diagram

5.1.2 Flowchart Diagram

A flowchart is a visual representation of the sequence of steps and decisions needed to perform a process. Each step in the sequence is noted within a diagram shape. Steps are linked by connecting lines and directional arrows. This allows anyone to view the flowchart and logically follow the process from beginning to end. The below Figure 5.2 shows the flowchart diagram. A flowchart is a powerful business tool. With proper design and construction, it communicates the steps in a process very effectively and efficiently.

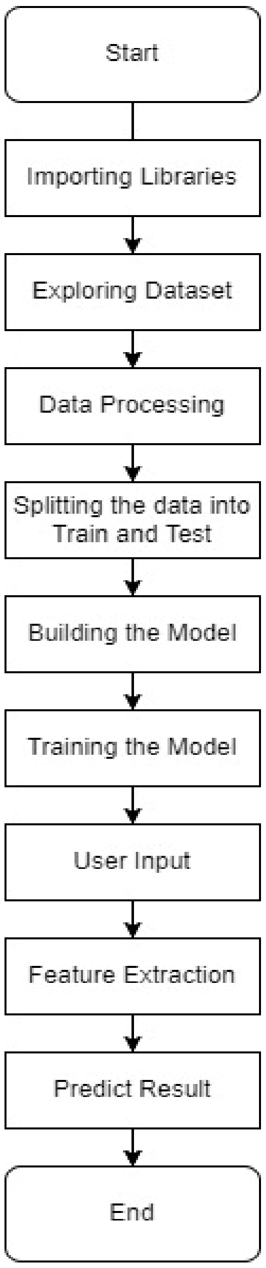


Figure 5.2 Flow Chart

5.1.3 Collaboration Diagram

A collaboration diagram groups together the interactions between different objects. The interactions are listed as numbered interactions that help to trace the sequence of the interactions. The collaboration diagram helps to identify all the possible interactions that each object has with other objects. The below Figure 5.3 shows the collaboration diagram.

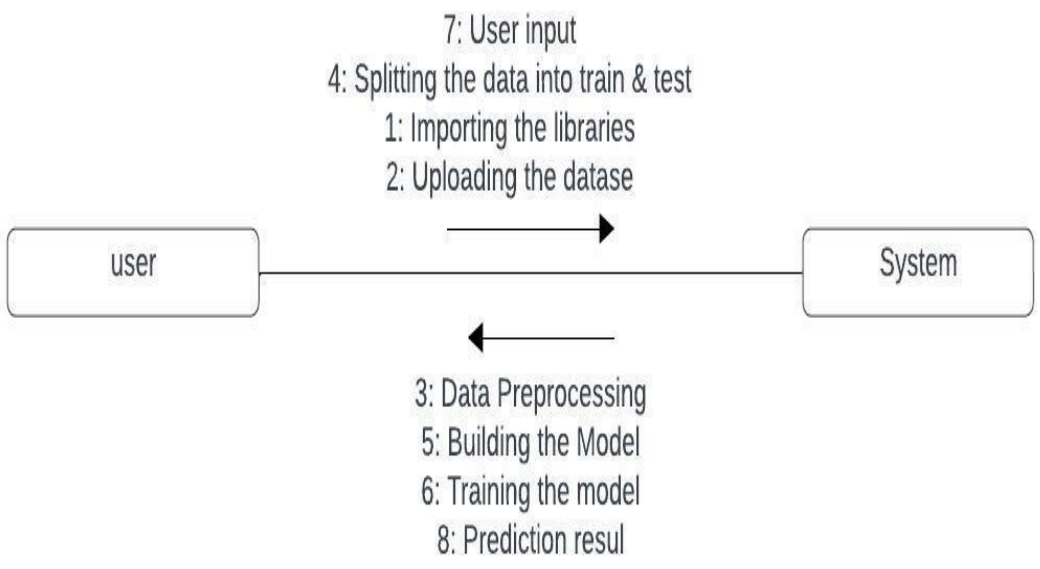


Figure 5.3Collaboration Diagram

6 Implementation

In this phase the designs are translated into code. Computer programs are written using a conventional programming language or an application generator. Programming tools like Compilers, Interpreters, and Debuggers are used to generate the code. Different high level programming languages like C, C++, Pascal, Python, Java, Net are used for coding. With respect to the type of application, the right programming language is chosen.

6.1 Libraries/API's used

1. **Pandas:** Pandas is an open-source library that is made mainly for working with relational or labeled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. This library is built on top of the NumPy library. Pandas is fast and it has high performance & productivity for users.
2. **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. Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data.In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.
3. **Matplotlib:** Matplotlib is easy to use and an amazing visualizing library in Python. It is built on NumPy arrays and designed to work with the broader SciPy stack and consists of several plots like line, bar, scatter, histogram, etc.

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

4. **Scikit learn:** Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, and clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.
5. **OS:** It is possible to automatically perform many operating system tasks. The OS module in Python provides functions for creating and removing a directory (folder), fetching its contents, changing and identifying the current directory, etc. You first need to import the os module to interact with the underlying operating system. So, import it using the import os statement before using its functions.
6. **Flask:** Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular Python web application frameworks.Flask offers suggestions, but doesn't enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are

many extensions provided by the community that make adding new functionality easy.

7. **Warnings:** The warnings module is part of Python's standard library and provides utilities for issuing warning messages to alert users about potential issues, such as deprecated features or runtime conditions that may cause unexpected behavior. It allows developers to control the display and handling of warnings during program execution.
8. **Pickle:** The pickle module is used for serializing and deserializing Python objects into a byte stream. It enables the conversion of complex data structures, such as objects and arrays, into a format that can be stored in files or transmitted over networks. Pickle is commonly used for saving trained machine learning models to disk and loading them back into memory for inference.

6.2 Modules

The following are the Modules in our project:

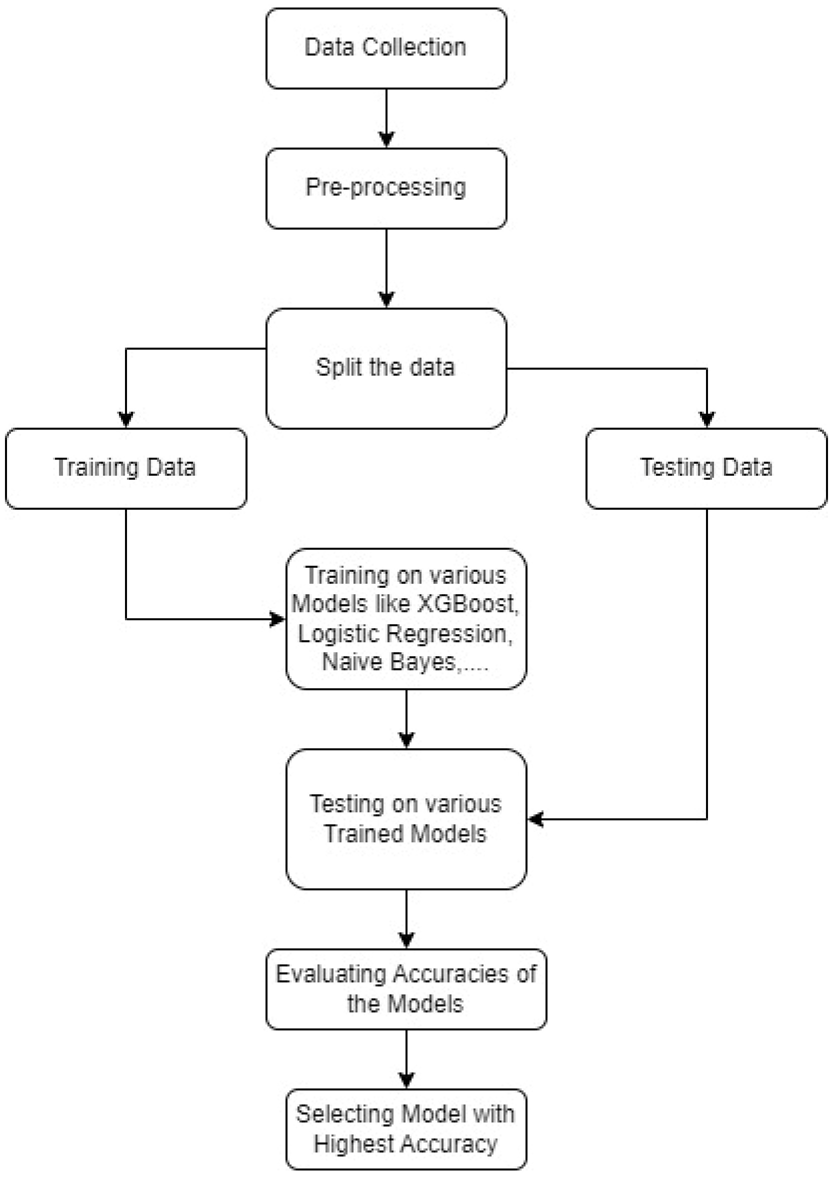

Data Collection
Pre-processing
Split the data
Training Data
Testing Data
Training on various Models like XGBoost, Logistic Regression, Naive Bayes,....
Testing on various Trained Models
Evaluating Accuracies of the Models
Selecting Model with Highest Accuracy

Figure 6.1 Design

1.Data Collection:

Gather the dataset containing examples of URLs along with their labels indicating whether they are safe or phishing.

2.Pre-Processing:

Clean and preprocess the data to prepare it for modeling. This may involve tasks such as handling missing values, encoding categorical variables, and scaling numerical features.

3.Split the Data:

Divide the dataset into training and testing sets. The training set is used to train the machine learning models, while the testing set is used to evaluate their performance.

4.Training Data:

Train the machine learning models using the training data. This involves feeding the input features (e.g., extracted features from URLs) and their corresponding labels into the models and adjusting their parameters to minimize a chosen loss function.

5.Teating Data:

After training the models, use the testing data to assess their performance on unseen examples. This helps to gauge how well the models generalize to new data.

6.Training on various Models:

Experiment with different machine learning algorithms such as Gradient Boost, XGBoost, Logistic Regression, and Naive Bayes. Train each model on the training data and evaluate their performance on the testing data.

7.Testing on Various Trained Models:

After training, use each trained model to make predictions on the testing data and compare their performance.

8.Evaluating Accuracies if the Models:

Calculate evaluation metrics such as accuracy, precision, recall, F1-score, and ROC-AUC to assess the performance of each model. These metrics provide insights into different aspects of model performance.

9.Selecting Models with Highest Accuracy:

Choose the model with the highest accuracy or the best overall performance based on the evaluation metrics. Consider other factors such as computational efficiency, interpretability, and domain-specific requirements when selecting the final model.

10.Predict the result for user Input with Model:

Now the model will takes user input and uptain the features from the input and predict the results and display the results.

6.3 Input

The input refers to the URL (Uniform Resource Locator) provided by the user. It is in the Figure 6.2 A URL is a web address that specifies the location of a resource (such as a webpage, image, or file) on the internet. It consists of several components, including the protocol (such as HTTP or HTTPS), domain name (e.g., www.example.com), path, and optional query parameters.

URL: The URL provided by the user is typically entered into a web application or system interface.

Purpose: The purpose of collecting the URL as input is to determine whether it is safe or potentially malicious (phishing).

Format: URLs follow a specific format, starting with the protocol (e.g., "http://" or "https://"), followed by the domain name and optional path and query parameters.

Content: The content of the URL could vary widely, ranging from legitimate websites to phishing pages, malicious downloads, or other types of online resources.

Security Considerations: Since URLs can be used to access various resources on the internet, it's essential to validate and verify them to ensure users' safety and prevent security threats such as phishing attacks, malware distribution, or data breaches.

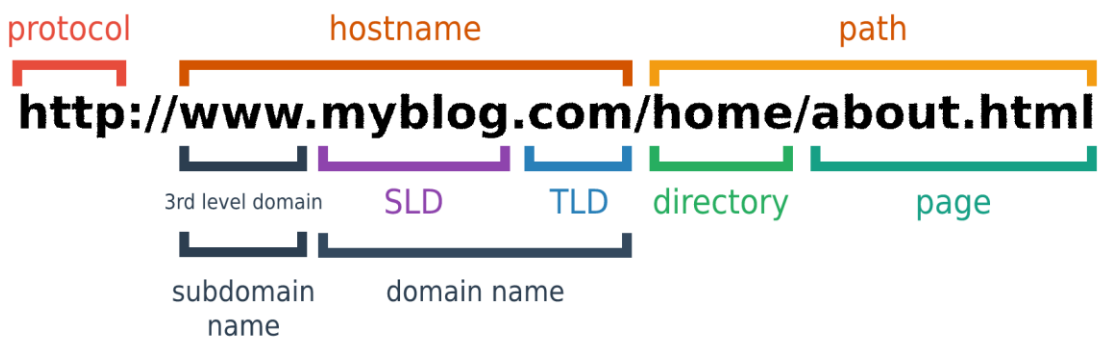


Figure 6.2 Url Format

6.4 Feature Extraction

The feature extraction process for URL classification involves analyzing several key characteristics to identify potential security risks. Features such as 'UsingIP' determine whether the URL employs an IP address instead of a domain name, while 'HTTPS' checks for secure communication protocols. 'LongURL' and 'ShortURL' assess the length and potential use of URL shortening services. Suspicious patterns like 'Symbol@' or 'Redirecting//' are flagged, along with anomalies like 'AbnormalURL' or 'NonStdPort'. Additionally, indicators of phishing attempts such as 'InfoEmail' or 'AnchorURL' are scrutinized. Features like 'PageRank' and 'WebsiteTraffic' provide insights into the webpage's authority and popularity. By extracting and analyzing these features, a comprehensive understanding of the URL's characteristics is obtained, aiding in the detection of potentially harmful URLs and safeguarding users against online threats.Every feature have its won importance and it was displayed in Figure 6.3

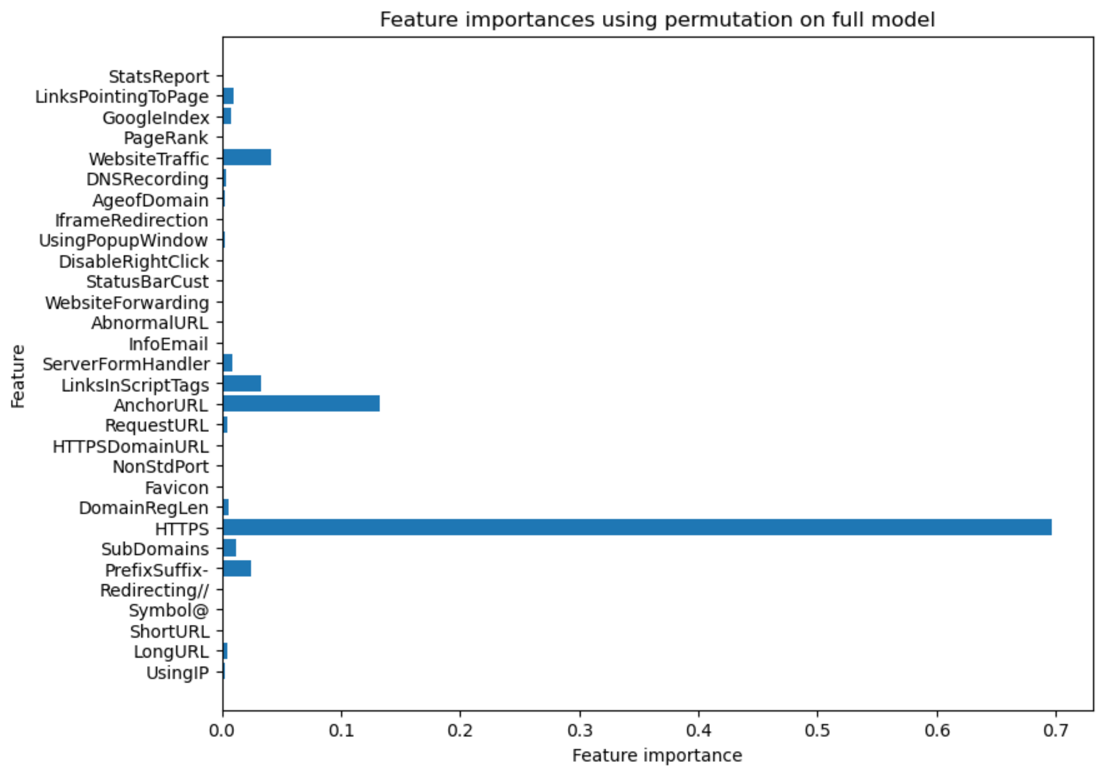


Figure 6.3Feature importance.

This table neatly arranges 30 URL features into three columns for efficient presentation. Each feature plays a crucial role in assessing the security risks associated with URLs. The layout facilitates quick identification and comparison of these features. Despite the compact design, the table maintains clarity and readability, aiding in the analysis of URL characteristics. Its organized structure enhances usability and accessibility for users reviewing the features.

Table 6.1 URL Features

Features		
UsingIP	LongURL	ShortURL
Symbol@	Redirecting//	PrefixSuffix-
SubDomains	HTTPS	DomainRegLen
Favicon	NonStdPort	HTTPSDomainURL
RequestURL	AnchorURL	LinksInScriptTags
ServerFormHandler	InfoEmail	AbnormalURL
WebsiteForwarding	StatusBarCust	DisableRightClick
UsingPopupWindow	IframeRedirection	AgeofDomain
DNSRecording	WebsiteTraffic	PageRank
GoogleIndex	LinksPointingToPage	StatsReport

7 Results

Input URL provided by the user:

The user enters a URL into the input field and clicks the "Check here" button.

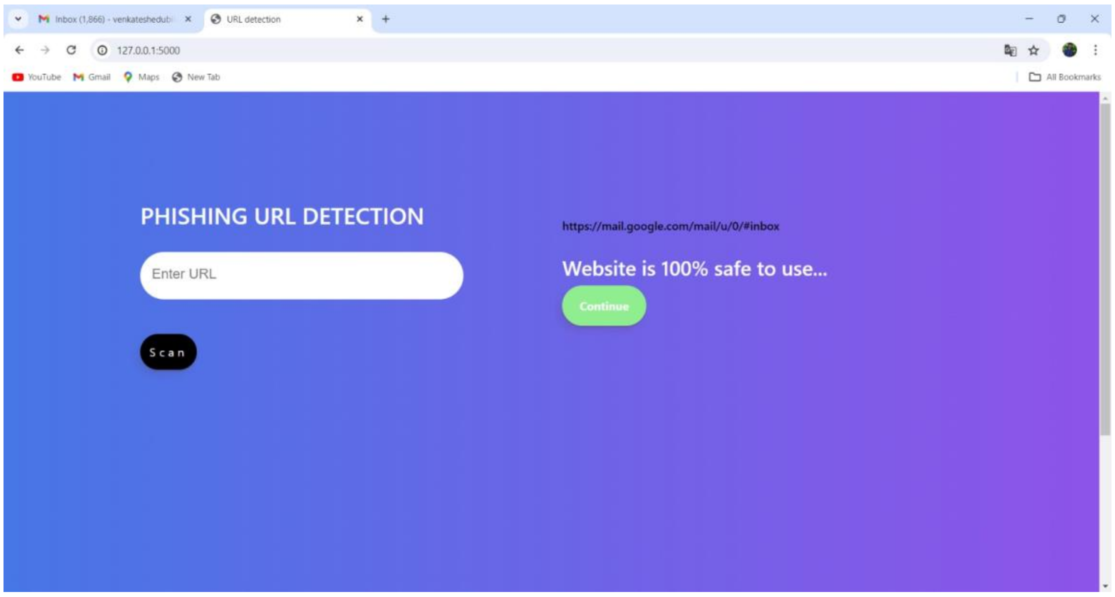
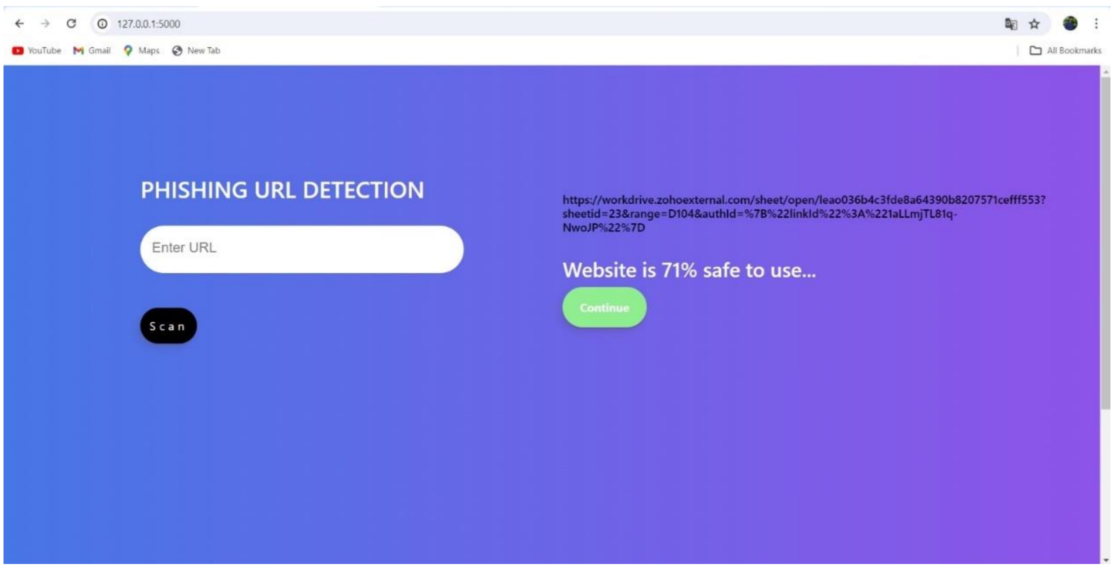

PHISHING URL DETECTION
Enter URL
Scan
https://mail.google.com/mail/u/0/#inbox
Website is 100% safe to use...
Continue

Figure 7.1User Interface for entering the Url

Safe URL prediction:

If the machine learning model predicts the URL to be safe (probability >= 0.50), the output will display "Website is [probability]% safe to use..." along with a "Continue" button.

Example output: "Website is 90% safe to use..."



PHISHING URL DETECTION

https://workdrive.zohoexternal.com/sheet/open/leao036b4c3fde8a64390b8207571cefff553?sheetid=23&range=D104&authId=%7B%22linkId%22%3A%221aLLmjTL81q-NwoJP%22%7D

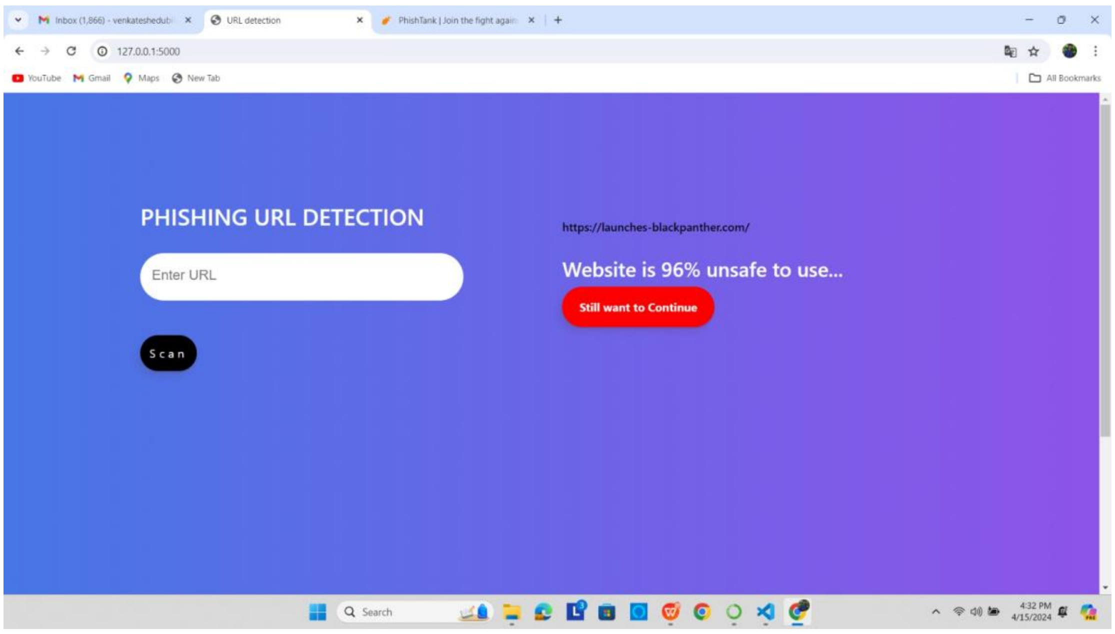
Website is 71% safe to use...

Figure 7.2 Safe URL Prediction

Unsafe URL prediction:

If the machine learning model predicts the URL to be unsafe (probability < 0.50), the output will display "Website is [probability]% unsafe to use..." along with a "Still want to Continue" button.

Example output: "Website is 40% unsafe to use..."



PHISHING URL DETECTION

https://launches-blackpanther.com/

Website is 96% unsafe to use...

Figure 7.3 Unsafe URL Prediction

Rendering the URL:

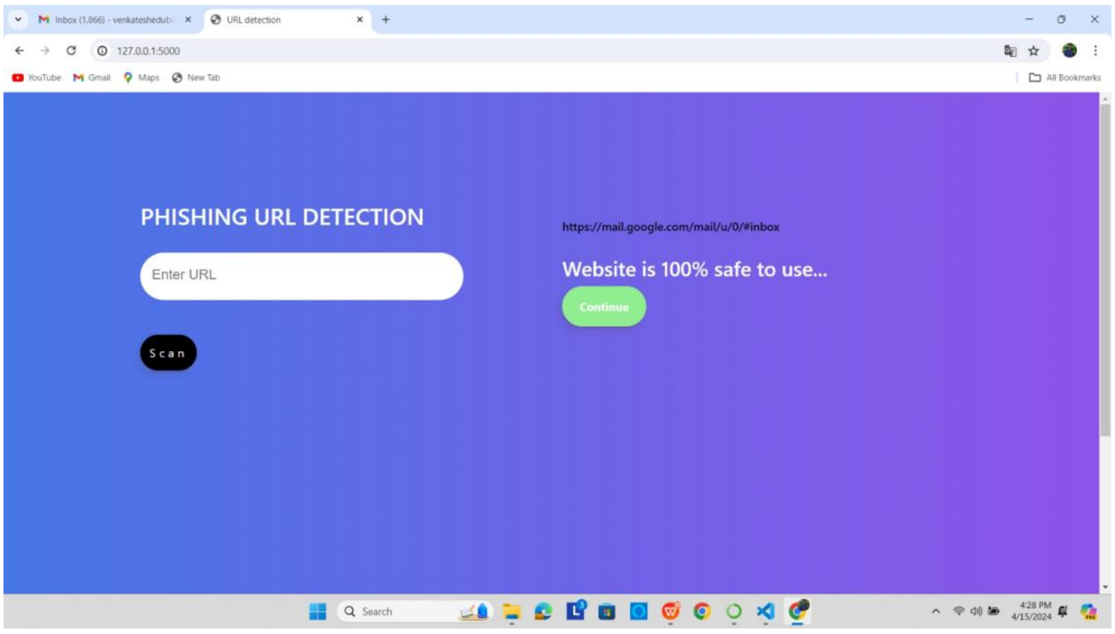
The URL provided by the user is rendered as a hyperlink below the prediction result.

Button visibility:

Depending on the prediction result, either the "Continue" button or the "Still want to Continue" button will be displayed.

If the URL is predicted to be safe, the "Continue" button will be shown.

If the URL is predicted to be unsafe, the "Still want to Continue" button will be shown.



https://mail.google.com/mail/u/0/#inbox

Website is 100% safe to use...

Figure 7.4 Overall output of Phishing Website Detection

8 Conclusion and Future Work

This chapter concludes the thesis. It briefly relates the motivation and provides the conclusion on the work done in this thesis. Finally, the direction of future research is discussed.

8.1 Conclusion

The visualization of the URL dataset indicates promising results, with all models demonstrating high prediction accuracy. Notably, Gradient Boosting achieved the highest accuracy among the models examined. To further enhance the performance of the Gradient Boosting model on OpenPhish data, it could be beneficial to train it on a more balanced dataset, with an equal representation of malicious and benign websites. This analysis underscores the efficacy of employing a Gradient Boosting model trained on selected features to effectively detect malicious URLs and predict potential phishing attacks..

8.2 Future Scope

By including auxiliary modules like registration services, website content, network reputation, file paths, & registry keys, DeepURLDetect (DUD) model can be strengthened. One about key directions considering future work can be considered towards this.

9 References

[1] R. Dhanalaksmi and C. Chellappan, "Detecting Malicious URLs in E-mail - An Implementation," AASRI Procedia, vol. IV, no. 4, pp. 125-131, 2013.

[2]"OpenPhish," [Online]. Available: https://openphish.com/. [Accessed 20 10 2022).

[3] Y. Fuqiang, "Malicious URL Detection Algorithm based on BM Pattern Matching," International Journal of Security and Its Applications, vol. IX, no. 4, pp. 33-44, 2015.

[4]K. Nirmal, B. Janet and R. Kumar, "Phishing - the threat that still exists," ICCCT. vol. III, pp. 139-143, 2015.

[5]F. Vanhoenshoven, G. Napoles and R. Falcon, "Detecting Malicious URLs using machine learning techniques," IEEE SSCI, pp. 1-8, 2016.

[6]S. Doyen, L. Chenghao and C. H. Steven, "Malicious URL Detection using Machine Learning: A Survey," vol. III, no. 21, pp. 30-45, 2019.

[7]K. Akash, "Kaggle Repository," 2018. [Online]. https://www.kaggle.com/datasets/akashkr/phishing-website- dataset?resource-download. [Accessed 15 10 2022]. Available

[8]V. Rakesh and D. Avisha, "What's in a URL: Fast Feature Extraction," ACM ISBN, vol. III, 2017.