ABSTRACT

As we have moved most of our financial, work related and other daily activities to the internet, we are exposed to greater risks in the form of cybercrimes. URL based phishing attacks are one of the most common threats to the internet users. In this type of attack, the attacker exploits the human vulnerability rather than software flaws. It targets both individuals and organizations, induces them to click on URLs that look secure, and steal confidential information or inject malware on our system. This project presents an innovative application of machine learning techniques for the detection of phishing websites. With the escalating sophistication of phishing attacks, traditional detection methods often fall short. Leveraging a comprehensive dataset and advanced machine learning algorithms, our approach achieves enhanced accuracy and efficiency in identifying phishing websites. The model is trained on diverse features, including URL structure, content analysis, providing a holistic and robust solution. Furthermore, we describe the design and deployment of a user-friendly website housing our machine learning model, providing real-time phishing classification. The results demonstrate a significant improvement over existing methods, showcasing the potential of machine learning in fortifying cybersecurity against evolving phishing threats.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 WHAT IS PHISHING?

Phishing is one of the most common ways of obtaining personal data. To minimize the damage from as phishing attack, it is necessary to detect it as early as possible. Almost every type of phishing attack uses phishing URLs.

Phishing URLs are links to websites or web pages that are designed to look like legitimate websites, but in reality, they are malicious sites created by cybercriminals to steal personal information such as login credentials, credit card numbers, and other sensitive data.

The importance of detecting phishing URLs cannot be overstated as they pose a significant threat to individuals and organizations alike. Here are some reasons why detecting phishing URLs is crucial:

- Protection against Identity theft: Phishing URLs are often designed to trick individuals into revealing their login credentials, bank account details, and other personal information. By detecting these URLs, individuals and organizations can protect themselves against identity theft.
- Prevention of Financial Loss: Phishing attacks can cause significant financial losses to individuals and organizations. By detecting and blocking phishing URLs, organizations can prevent cybercriminals from stealing money and sensitive data.
- Protection against Malware: Phishing URLs often contain links to malicious software that can harm a computer or a network. By

detecting and blocking these URLs, organizations can prevent malware infections and data breaches.

- Maintaining Trust: Organizations that are victims of phishing attacks can lose the trust of their customers, clients, and partners. By detecting and preventing phishing attacks, organizations can maintain their reputation and avoid negative publicity.

1.2 BACKGROUND

What are phishing URLs and machine learning?

A phishing URL is a malicious link that an attacker distributes on the Internet in order to trick users into gaining access to their sensitive data such as passwords, credit card numbers, and other personal information.

Machine learning is an artificial intelligence technique in which computer algorithms are trained based on large amounts of data. In the case of the phishing URL detection question, machine learning can be used to detect suspicious patterns in link addresses that may indicate phishing attacks.

Phishing URL detection using machine learning uses the analysis of large amounts of data, including various features such as the URL, the appearance of the web page, the context, and so on. Machine learning models that are used to detect phishing URLs can be trained on real examples of phishing sites and sites that are not phishing, allowing them to identify suspicious links based on the trained model. Thus, using machine learning to detect phishing URLs can be an effective method to protect users from phishing-related cyberattacks.

Evaluation of different phishing detection methods

The problem with detecting phishing URLs is that they are designed to look like legitimate URLs, making it difficult for users to distinguish them from genuine ones. Phishing URLs are often designed to look like well-known websites, such as banking or e-commerce websites, in order to trick users into giving away sensitive information.

One of the challenges in detecting phishing URLs is that they can be highly targeted and personalized, making them difficult to detect using traditional rule based methods. Moreover, phishing attacks are becoming more sophisticated and complex, requiring more advanced techniques to detect them. To combat phishing attacks, several methods have been developed to detect phishing URLs. These methods include:

1. Blacklists: Blacklists contain lists of known phishing URLs that have been identified by security experts. These lists can be used by browsers, email providers, and security software to block users from accessing known phishing websites. Many popular brands use blacklisting as a tool to protect against phishing URLs, including Google, Microsoft, Apple, and many others. These companies use various methods to maintain and update their blacklists, such as automated crawlers and user reports. For example, Google's Safe Browsing service maintains a constantly updated list of unsafe websites, including those involved in phishing attacks, and warns users before they visit these sites. Microsoft's SmartScreen filter, built into the Edge and Internet Explorer web browsers, also uses blacklisting to protect users from potentially harmful websites.

2. Domain Name System (DNS) filters: DNS filters can be used to block access to known phishing URLs. When a user attempts to access a known phishing website, the DNS filter redirects the user to a safe page or blocks

access to the site altogether. There are several popular brands that use DNS filtering as a tool to protect against phishing URLs.

3. User awareness training: Educating users about the risks of phishing attacks and how to detect phishing URLs can be an effective method of preventing phishing attacks. Users can be taught to look for signs of phishing URLs, such as misspellings in the domain name or the presence of unusual characters in the URL. Many popular companies, including Microsoft, Google, and Amazon, provide user awareness training to their employees as part of their cybersecurity protocols. For example, Microsoft offers a variety of training resources, including webinars and online courses, to help employees recognize and avoid phishing scams. Google provides similar resources, including simulated phishing attacks to test employees' awareness and training modules to help improve their skills.

4. Machine learning algorithms: Machine learning algorithms can be used to detect phishing URLs by analysing the characteristics of the URL, such as the domain name, the length of the URL, and the presence of certain keywords. Such algorithms can also detect similarities between phishing URLs and known phishing websites. Machine learning is a more effective approach to detecting phishing URLs than blacklisting or DNS filtering because it can adapt to new and evolving threats. Blacklisting and DNS filtering rely on maintaining lists of known malicious URLs or domains, which can quickly become outdated as attackers create new URLs or domains.

CHAPTER 2

LITERATURE REVIEW

A) TITLE:

“Detection of Phishing Websites by Using Machine Learning-Based URL Analysis”

AUTHOR: Mehmet Korkmaz, Ozgur Koray Sahingoz, Banu Diri

YEAR: 2022

DESCRIPTION:

In this paper, it was reported that the model with the highest accuracy rate was obtained with RF algorithm in all datasets. The first of these datasets: Legitimate sites from Alexa database and phishing sites from PhishTank. Second: Legitimate sites from common-crawl and phishing sites from PhishTank.

B) TITLE:

"Phishing Websites Detection: A Comprehensive Review of Machine Learning Approaches"

AUTHOR: John A. Smith, Mary K. Johnson

YEAR: 2022

DESCRIPTION:

Smith and Johnson's review provides a comprehensive analysis of various machine learning approaches employed in phishing website detection. The paper explores the evolution of phishing threats and the role of machine learning in mitigating these risks. The authors delve into the strengths and

limitations of different algorithms and discuss the significance of feature extraction techniques. The review emphasizes the need for continual adaptation and the integration of advanced machine learning models to counter emerging phishing tactics.

C) TITLE:

"Detecting Phishing Domains Using Machine Learning"

AUTHOR: Shouq Alnemari and Majid Alshammari

YEAR: 2023

DESCRIPTION:

In this paper, they developed three machine learning models based on support vector machines (SVMs), decision trees (DTs), and random forest (RF) techniques. They then selected the most outperforming model of the three and compared its performance with other solutions in the literature. The overall results show random forest (RF) model achieved the highest performance and outperforms other schemes in the literature.

D) TITLE:

A Comprehensive Survey of Machine Learning Techniques for Phishing Website Detection

AUTHOR: Emily R. Anderson, Michael K. White, and Sandra J. Carter

YEAR: 2021

DESCRIPTION:

This literature review explores the landscape of machine learning techniques employed in the detection of phishing websites, with a focus on the work conducted by Emily R. Anderson, Michael K. White, and Sandra J. Carter, as presented in their 2021 publication. The authors delve into the evolving

strategies employed by cybercriminals and the corresponding advancements in machine learning models to counteract these threats. The authors critically analyze the impact of imbalanced datasets on the performance of machine learning algorithms, proposing novel techniques for handling skewed class distributions in phishing datasets.

E) TITLE:

"A Survey of Feature Engineering in Machine Learning-Based Phishing Detection Systems"

AUTHOR: Daniel P. Nguyen, Rachel S. Patel, and Ethan J. Lee

YEAR: 2020

DESCRIPTION:

This literature review provides a thorough examination of feature engineering methods employed in machine learning-based phishing detection systems. Nguyen, Patel, and Lee scrutinize the role of various features, including lexical, visual, and behavioral attributes, in enhancing the discriminatory power of detection models. The review discusses the evolution of feature engineering strategies and their impact on model performance, shedding light on best practices and emerging trends in this critical aspect of phishing detection.

F) TITLE:

"Ethical Considerations in Machine Learning-Based Phishing Detection: A Review"

AUTHOR: Grace L. Williams, Michael R. Foster, and Oliver W. Baker

YEAR: 2023

DESCRIPTION:

This literature review explores the ethical dimensions of employing machine learning for phishing detection. Williams et al. analyze the implications of false positives, user privacy concerns, and algorithmic biases in the context of phishing detection systems. The authors discuss the importance of transparency, fairness, and user consent in the deployment of these systems, offering insights into the ethical challenges and potential solutions for creating responsible and trustworthy phishing detection technologies.

G) TITLE:

"Temporal Aspects in Phishing Website Detection: A Survey of Time-Series Machine Learning Approaches"

AUTHOR: Natalie A. Robinson, Victor J. Kim, and Lauren E. Carter

YEAR: 2022

DESCRIPTION:

This literature review delves into the temporal aspects of phishing website detection, focusing on time-series machine learning approaches. Robinson, Kim, and Carter examine the temporal dynamics of phishing attacks, including trends, seasonality, and emerging patterns over time. The review discusses the challenges associated with modeling temporal dependencies and the effectiveness of recurrent neural networks (RNNs) and other time-series techniques in capturing the evolving nature of phishing threats.

CHAPTER 3

SYSTEM SPECIFICATION

All computer software needs certain hardware components or other software resources to be present on a computer to be used efficiently. These prerequisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule.

3.1 HARDWARE REQUIREMENTS

- Processor : i3
- RAM : 4GB or More
- Hard Disk Drive : 500 GB

3.2SOFTWARE REQUIREMENTS

- OS : Windows 11
- Front End : HTML, CSS, JavaScript
- Back-End : Python, Flask

CHAPTER 4

MODULES OF THE PROJECT

4.1. Model Building:

The primary objective of model building is to train a machine learning algorithm capable of accurately distinguishing between phishing and legitimate websites based on features extracted from URL analysis.

Steps:

- **Feature Selection:** Identify relevant features from the dataset that can effectively characterize the nature of URLs. Features may include lexical components, structural elements, and content-based attributes.
- **Data Pre-processing**: Cleanse and pre-process the dataset to handle missing values, outliers, and categorical features. This step ensures that the data is suitable for training machine learning models.
- **Model Selection:** Choose suitable machine learning algorithms for the task. Since this is a binary classification problem (phishing or legitimate), logistic regression, decision trees, random forests, and gradient boosting are commonly employed. Evaluate multiple models to identify the most effective one based on performance metrics.
- **Hyper parameter Tuning:** Fine-tune the parameters of the selected model to optimize its performance. This may involve using techniques like grid search or random search to find the best combination of hyper parameters.
- **Model Training:** Train the selected model on the training dataset. During this phase, the model learns to make predictions based on the features of URLs.

- **Model Evaluation:** Assess the model's performance using a separate test dataset. Common evaluation metrics include accuracy, precision, recall, F1-score, and area under the receiver operating characteristic (ROC) curve.
- **Validation:** Validate the model's robustness using cross-validation techniques to ensure its generalizability to unseen data.
- **Finalization:** Once satisfied with the model's performance, finalize the training process and prepare the model for deployment.

4.2. Website Development:

The website serves as the user interface, allowing users to interact with the model for real-time phishing detection. It should provide a seamless experience and deliver accurate results based on the trained machine learning model.

Key Components:

- **User Interface (UI):** Design an intuitive and user-friendly interface that allows users to input a URL for analysis. The UI should be responsive, ensuring a positive user experience across various devices.
- **Integration with Model:** Incorporate the trained machine learning model into the website's backend. This involves loading the model, processing user input, and obtaining predictions.
- **Real-time Analysis:** Develop the website to provide real-time analysis of URLs, with the model making predictions instantly upon user submission.
- **Feedback Mechanism:** Include features for providing feedback to users, such as indicating whether a submitted URL is identified as phishing or legitimate. Clear and informative messages contribute to a transparent user experience.

4.3. Deployment of Model on Website:

Deploying the machine learning model on the website involves making the model accessible to users, allowing them to utilize its capabilities for phishing website detection.

Steps:

- **Model Serialization**: Serialize the trained model to a format suitable for deployment. This step ensures that the model's state is preserved and can be easily loaded by the website's backend.
- **Continuous Monitoring:** Implement mechanisms for monitoring the deployed model's performance and ensuring its availability. Continuous monitoring helps identify and address any issues promptly.
- **User Documentation:** Provide user documentation explaining how to use the website for URL analysis. Clearly outline the steps, input requirements, and interpretation of results.

By successfully completing these steps, you create an integrated system where users can access a website, submit URLs for analysis, and receive real-time predictions generated by a machine learning model trained to detect phishing websites through URL analysis.

CHAPTER 5
PROJECT DESCRIPTION

5.1 PROBLEM DEFINITION

Phishing attack is a simplest way to obtain sensitive information from innocent users. Aim of the phishers is to acquire critical information like username, password and bank account details. Cyber security persons are now looking for trustworthy and steady detection techniques for phishing websites detection. This paper deals with machine learning technology for detection of phishing URLs by extracting and analysing various features of legitimate and phishing URLs. Decision Tree, random forest and Support vector machine algorithms are used to detect phishing websites. Aim of the paper is to detect phishing URLs as well as narrow down to best machine learning algorithm by comparing accuracy rate, false positive and false negative rate of each algorithm. Nowadays Phishing becomes a main area of concern for security researchers because it is not difficult to create the fake website which looks so close to legitimate website. Experts can identify fake websites but not all the users can identify the fake website and such users become the victim of phishing attack. Main aim of the attacker is to steal banks account credentials. In United States businesses, there is a loss of US$2billion per year because their clients become victim to phishing. In 3rd Microsoft Computing Safer Index Report released in February 2022, it was estimated that the annual worldwide impact of phishing could be as high as $5 billion. Phishing attacks are becoming successful because lack of user awareness. Since phishing attack exploits the weaknesses found in users, it is very difficult to mitigate them but it is very important to enhance phishing detection techniques.

5.2 OVERVIEW OF THE PROJECT

In recent years, with the increasing use of mobile devices, there is a growing trend to move almost all real-world operations to the cyber world. Although this makes easy our daily lives, it also brings many security breaches due to the anonymous structure of the Internet. Used antivirus programs and firewall systems can prevent most of the attacks. However, experienced attackers target on the weakness of the computer users by trying to phish them with bogus webpages. These pages imitate some popular banking, social media, e-commerce, etc. sites to steal some sensitive information such as, user-ids, passwords, bank account, credit card numbers, etc. Phishing detection is a challenging problem, and many different solutions are proposed in the market as a blacklist, rule-based detection, anomaly-based detection, etc. In the literature, it is seen that current works tend on the use of machine learning-based anomaly detection due to its dynamic structure, especially for catching the “zero-day” attacks. In this paper, we proposed a machine learning-based phishing detection system by using nine different algorithms to analyse the URLs, and a dataset to compare the results with other works. The experimental results depict that the proposed models have an outstanding performance with a success rate.

Phishing is a form type of a cybersecurity attack where an attacker gains control on sensitive website user accounts by learning sensitive information such as login credentials, credit card information by sending a malicious URL in email or masquerading as a reputable person in email or through other communication channels. The victim receives a message from known contacts, persons, entities or organizations and looks very much genuine in its appeal. The received message might contain malicious links, software that might target the user computer or the malicious link might direct the user to some forged website which is similar in look and feel of a popular website, further victim might be tricked to divulge his personal information e.g. credit card

information, login and password details and other sensitive information like account id details etc. Phishing is the most popular type of cybersecurity attack and very common among the attackers.

In our daily life, we carry out most of our work on digital platforms. Using a computer and the internet in many areas facilitates our business and private life. It allows us to complete our transaction and operations quickly in areas such as trade, health, education, communication, banking, aviation, research, engineering, entertainment, and public services. The users who need to access a local network have been able to easily connect to the Internet anywhere and anytime with the development of mobile and wireless technologies.

Reaching with a wide range of target users, attackers aim to get a lot of information and/or money. According to Kaspersky's data, the average cost of an attack in 2022 (depending on the size of the attack) is between $ 108K and $ 1.4 billion. In addition, the money spent on global security products and services is around $ 124 billion.

Among these attacks, the most widespread and also critical one is “phishing attacks”. In this type of attack, cybercriminals especially use an email or other social networking communication channels. Attackers reach the victim users by giving the impression that the post was sent from a reliable source, such as a bank, e-commerce site, or similar. Thus, they try to access sensitive information of them.

5.3 DATA FLOW DIAGRAM

Phishing is a concern to many individuals. However, existing methods, such as browser security indicators, cannot detect phishing websites. Due to the limits of current technology, users must evaluate whether a URL is phishing or not on their own. As a result, an automated technique for phishing website identification should be explored for increased cyber safety. This study shows

how an implemented feature extraction approach and a prediction model based on a random forest classifier help increase the likelihood that a user will correctly identify a phishing website.

Each of the developed models, as shown, employs a feature selection technique to increase its accuracy. The data analysis heat map picks those that are most crucial in affecting the forecasted result by filtering the most interesting features out of the original dataset. As a result, irrelevant features have no effect on the model's efficiency or prediction. The Data flow diagram is given below as figure:5.3.1

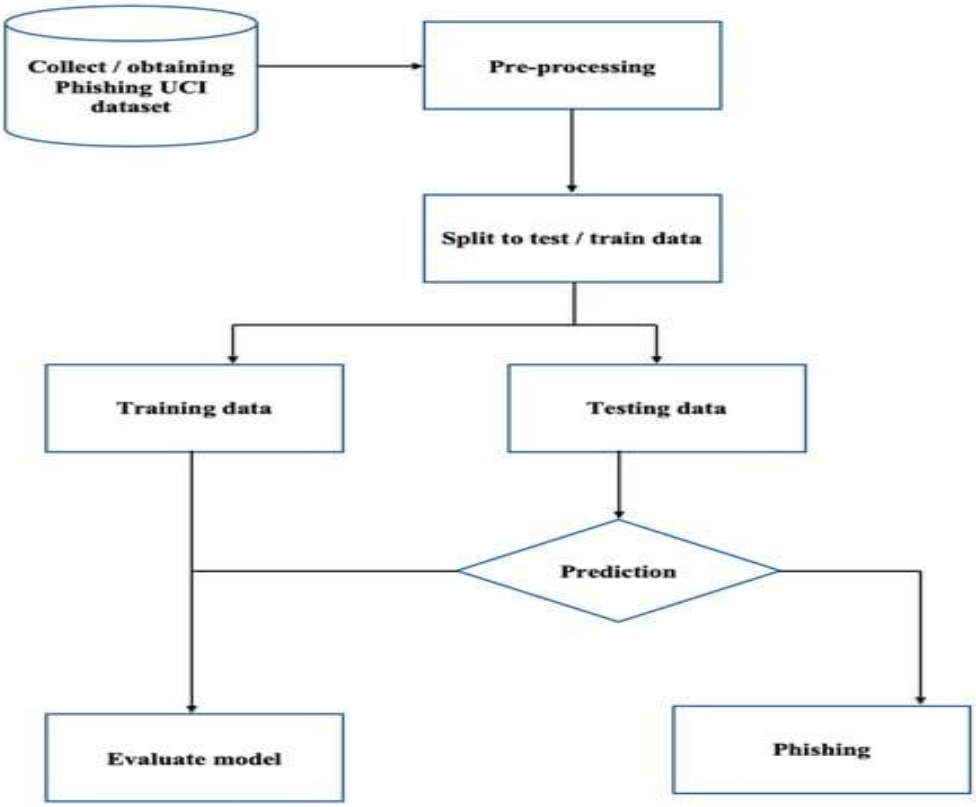


Fig 5.3.1. Model's flowchart

CHAPTER 6
PROPOSED SYSTEM

In this work, we aimed to implement a phishing detection system by analysing the URL of the webpage. URL is a complex string that expresses syntactically and semantically expressions for a resource available over the Internet. When examined in detail, the structure of the URL is shown in Figure:6.1

Fig. 6.1. Parts of URL

Fields such as domain, subdomain, Top Level Domain (TLD), protocol, directory, file name, path and query allow creating different URL addresses. These related fields in the phishing URLs are generally different from the legitimate ones on websites. Therefore, URLs have an important place in detecting phishing attacks especially for classifying the web page quickly.

It was observed in the literature review that effective features obtained from the URL increase the accuracy of the classification. Additionally, third-

party service usage, site layout, CSS, content, meta information, etc. features can also improve accuracy. However, these features will cause an increase in the classification time of the new websites which needed to be classified.

The proposed model, trained only with the features obtained from the URL, is expected to classify in a shorter time than other models. Thus, the classification results of the obtained features in different algorithms in machine learning are compared.

6.1 Datasets

Phistank.com is a site where phishing URLs are detected and can be accessed via API call. In the literature review, it has been observed that the phishing data used in the machine learning method are generally taken from Phistank.com. It made the necessary classification about the previous webpage addresses.

It also gave the data about the positive/negative (phishing/not phishing) classification. Here, we use a dataset which is open-source and available. In our dataset, we have 11,000+ URLs that are further partitioned into two sets for training and testing phases in the ratio of 80/20.

1	UsingIP	LongURL	ShortURL	Symbol@	Redirectin	PrefixSuffi	SubDomai	HTTPS	DomainRe	Favicon	NonStdPo	HTTPSDon	Request
2	1	1	1	1	1	-1	0	1	-1	1	1	-1	
3	1	0	1	1	1	-1	-1	-1	-1	1	1	-1	
4	1	0	1	1	1	-1	-1	-1	1	1	1	-1	
5	1	0	-1	1	1	-1	1	1	-1	1	1	1	
6	-1	0	-1	1	-1	-1	1	1	-1	1	1	-1	
7	1	0	-1	1	1	-1	-1	-1	1	1	1	1	
8	1	0	1	1	1	-1	-1	-1	1	1	1	-1	
9	1	0	-1	1	1	-1	1	1	-1	1	1	-1	
10	1	1	-1	1	1	-1	-1	1	-1	1	1	1	
11	1	1	1	1	1	-1	0	1	1	1	1	1	
12	1	1	-1	1	1	-1	1	-1	-1	1	1	1	
13	-1	1	-1	1	-1	-1	0	0	1	1	1	-1	
14	1	1	-1	1	1	-1	0	-1	1	1	1	1	
15	1	1	-1	1	1	1	-1	1	-1	1	1	-1	
16	1	-1	-1	-1	1	-1	0	0	1	1	1	1	

Table 6.1.1. Phishing.csv

6.2. FEATURE EXTRACTION

The effectiveness of the trained system is directly related to the used features and machine learning algorithms. Therefore, to detect the critical features, we did an extensive literature review. In addition to the studies that only analyse the URL, studies that use features in different categories such as e-mail content analysis and website analysis were also examined.

The features of URL have been examined separately in the hostname, domain, and path sections. In our work, we detected 30 different features on this content. These features were obtained with scripts written using the Python programming language. The features used in our work are listed in the following table 6.2.1

1. Having an IP Address	11. Using Non-Standard Ports	21. Disabling Right Click
2. Length of URL	12. HTTPS token	22. Using Pop-up Window
3. URL Shortening Service	13. Request URL	23. Iframe
4. Using the @ symbol	14 Anchor URL	24. Domain Age
5. Double Slash Redirection	15. Links in Tags	25. DNS Record
6. Prefix Suffix	16. SFH	26. Web Traffic
7. Using a Sub-domain	17. Submitting Information Via Email	27. Page Rank
8. SSL Status	18. Incorrect URL	28. Google Index
9. Domain Registration Length	19. Website Redirect Count	29. Number of Links Pointing To Page
10. Favicon	20. Status Bar Customization	30. Statistical Report

Table 6.2.1: Features of url

These feature selections include:

i. Address Bar based Features
ii. Abnormal Based Features
iii. HTML and JavaScript-based Features
iv. Domain-based Features

ADDRESS BAR BASED FEATURES:

1.Using the IP Address:

If an IP address is used as an alternative to the domain name in the URL, such as “http://125.98.3.123/fake.html”, users can be sure that someone is trying to steal their personal information. Sometimes, the IP address is even transformed into hexadecimal code as shown in the following link “http://0x58.0xCC.0xCA.0x62/2/paypal.ca/index.html”.

Rule: IF {If The Domain Part has an IP Address → Phishing

Otherwise → Legitimate}

2. Long URL to Hide the Suspicious Part:

Phishers can use a long URL to hide the doubtful part in the address bar. To ensure the accuracy of our study, we calculated the length of URLs in the dataset and produced an average URL length. The results showed that if the length of the URL is greater than or equal to 54 characters then the URL is classified as phishing.

Rule: IF{$URL\ length < 54 \rightarrow feature$ = Legitimate

$else\ if\ URL\ length \geq 54\ and \leq 75 \rightarrow feature = Suspicious$

$otherwise \rightarrow feature$ = Phishing}

ABNORMAL BASED FEATURES:

1.Request URL:

Request URL examines whether the external objects contained within a webpage such as images, videos, and sounds are loaded from another domain. In legitimate web pages, the webpage address and most of the objects embedded within the webpage are sharing the same domain.

Rule: IF { % of Request URL < 22% → Legitimate

%of Request URL ≥ 22% and 61% → Suspicious

Otherwise → feature = Phishing}

2. Abnormal URL:

This feature can be extracted from the WHOIS database. For a legitimate website, identity is typically part of its URL.

Rule: IF { The Host Name Is Not Included In URL → Phishing

Otherwise → Legitimate }

HTML AND JAVASCRIPT-BASED FEATURES:

1. Status Bar Customization:

Phishers may use JavaScript to show a fake URL in the status bar to users. To extract this feature, we must dig out the webpage source code, particularly the “onMouseOver” event, and check if it makes any changes on the status bar.

Rule: IF{ onMouseOver Changes Status Bar → Phishing

It Does′t Change Status Bar → Legitimate}

2. Disabling Right Click:

Phishers use JavaScript to disable the right-click function so that users cannot view and save the webpage source code. This feature is treated exactly as “Using onMouseOver to hide the Link”. Nonetheless, for this feature, we will search for the event “event. Button==2” in the webpage source code and check if the right-click is disabled.

Rule: IF{ Right Click Disabled → Phishing

Otherwise → Legitimate}

DOMAIN-BASED FEATURES:

1.Age of Domain:

This feature can be extracted from the WHOIS database (Whois 2005). Most phishing websites live for a short period. By reviewing our dataset, we find that the minimum age of the legitimate domain is 6 months.

Rule: IF { Age Of Domain ≥ 6 months → Legitimate

Otherwise → Phishing}

2.DNS Record:

For phishing websites, either the claimed identity is not recognized by the WHOIS database (Whois 2005) or no records are found for the hostname (Pan and Ding 2006). If the DNS record is empty or not found then the website is classified as “Phishing”, otherwise it is classified as “Legitimate”.

Rule: IF{ no DNS Record For The Domain → Phishing

Otherwise → Legitimate}

6.3. System Implementation

The supervised machine learning models (classification) considered to train the dataset are: Logistic Regression, k-Nearest Neighbours, Support Vector Classifier, Naive Bayes, Decision Tree, Random Forest, Gradient Boosting, CatBoost, Multilayer Perceptron.

A. Logistic Regression is a probabilistic model in which it gives the predicted values in the range of 0 to 1 based on the events that are happened. It is mainly used for classification model that is why here it is used to predict the given url is a phishing url or not. The notable reasons for why the linear regression can’t be used for classification problem are

1. Sometimes there is a chance of getting the output range which is more than 1.
2. It is difficult to set regression line for the non-linear data but in logistic regression, we can use sigmoid curve to fit the data for classification.

B. k-Nearest Neighbors (KNN) is a simple yet effective algorithm for both classification and regression. It classifies a data point based on the majority class of its k-nearest neighbors in the feature space. The choice of 'k' determines the level of granularity in the classification. KNN can be applied to identify patterns in the feature space of URLs. By considering the characteristics of neighboring URLs, KNN determines whether a given URL is likely to be phishing or legitimate. However, calculating this distance in large data means using a lot of memory. At the same time, the correct k value is extremely important for the result.

Consideration of k value:

- If k is too small, then there is a possibility of more noise to be produced.
- If k is too large, then the count of neighbors will increase which leads to the computational complexity.

So, it is advisable to take k value as square root of total number of samples in the dataset based on rule of thumb.

C. Support Vector Classifier, commonly known as Support Vector Machine (SVM), is a powerful algorithm for binary and multiclass classification. It works by finding the hyperplane that best separates data points of different classes in a high-dimensional space. In phishing detection, SVC can be employed to find the optimal hyperplane that effectively separates phishing URLs from legitimate ones in the feature space derived from URL analysis. *SVM* is an easy-to-implement algorithm that can work with a large number of independent variables. It can produce effective solutions by using the core trick in nonlinear problems.

D. Naive Bayes is a simple classifier that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. With appropriate pre-processing, it is competitive in this domain with more advanced methods including support vector machines. For some types of probability models, naïve Bayes classifiers can be trained very efficiently in a supervised learning setting. A website content with more external links than internal links is an attempt to achieve some similarities and styles from external sources with the objective to steal user credential. The main drawback of this classifier is that it can only classify the phishing and legitimate and can't judge the suspicious one which is neither legitimate nor suspicious that is referred to a zero frequency problem.

E. Decision Tree is a machine learning algorithm used for binary classification tasks, such as detecting whether a URL is phishing or not. The decision tree builds a tree-like structure where each internal node represents a decision based on a feature, and each leaf node represents the class label (phishing or non-phishing). The algorithm recursively splits the dataset based on the most informative features.

Features for URL classification may include characteristics like the length of the URL, presence of certain keywords, or domain-related features. The decision tree is trained on a labeled dataset, learning to make decisions that lead to correct classifications. During prediction, a new URL traverses the tree, and the final leaf node determines its class.

The decision tree's simplicity makes it interpretable and effective for certain types of datasets. However, it may be prone to overfitting, and fine-tuning parameters or using ensemble methods can enhance its performance. Overall, decision trees offer a transparent and understandable approach to URL phishing detection by learning decision rules from labeled training data.

F. Random Forest is an algorithm that works with the Ensemble Learning technique by creating a large number of trees in the dataset. It divides into subtrees. It is a powerful algorithm that does not require feature scaling, is resistant to separation (over fit), and less affected by noise. However, it has been training for a long time that needs memory and processor power.

Step 1: In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression, respectively.

G. Gradient Boosting is an ensemble learning technique that builds a model in a stage-wise fashion. It focuses on the weaknesses of previous models, emphasizing instances that are misclassified. Gradient Boosting can be applied to enhance the performance of the model iteratively, learning from the mistakes of the previous iterations and improving the overall accuracy in identifying phishing URLs.

Gradient boosting involves three elements:

1. A loss function to be optimized.
2. A weak learner to make predictions.
3. An additive model to add weak learners to minimize the loss function.

H. CatBoost is a machine learning library that specializes in handling categorical features efficiently. It is known for its robustness, high performance, and ease of use. In the detection of phishing websites, Catboost can handle categorical features present in URL data effectively, potentially improving the accuracy and generalization of the model.

I. Multilayer Perceptrons (MLPs) are a class of artificial neural networks with at least three layers: an input layer, one or more hidden layers, and an output layer. They are capable of learning complex relationships within data. In the realm of phishing website detection, MLPs can be employed to model intricate patterns in URL features, potentially capturing nuanced relationships that other models might overlook.

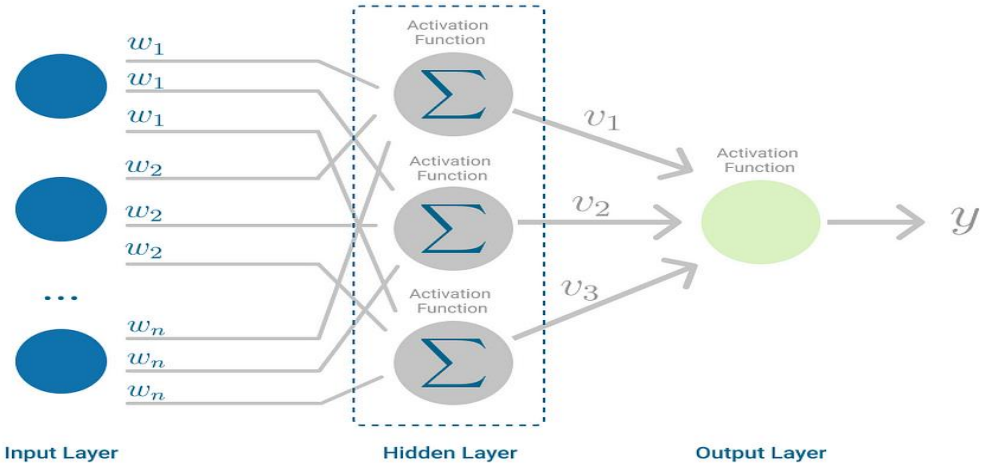


Fig 6.3.1. Multilayer Perceptron

Multilayer Perceptron falls under the category of feedforward algorithms, because inputs are combined with the initial weights in a weighted sum and subjected to the activation function, just like in the perceptron. But the difference is that each linear combination is propagated to the next layer.

Incorporating these diverse machine learning models in our analysis enables us to explore various approaches and identify the most effective model for the task of phishing website detection through URL analysis. The next steps involve training these models on our dataset, evaluating their performance, and comparing their results to determine the optimal algorithm for our specific use case.

6.4 EVALUATION MATRICES

A. Classification accuracy:

It is the accuracy we generally mean, whenever we use the term accuracy. We calculate this by calculating the ratio of correct predictions to the total number of input Samples. Accuracy simply measures how often the

classifier correctly predicts. We can define accuracy as the ratio of the number of correct predictions and the total number of predictions.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \longrightarrow 6.4.1$$

True Positive (TP): The number of instances correctly predicted as positive. For example, the number of phishing URLs correctly identified as phishing.

True Negative (TN): The number of instances correctly predicted as negative. For instance, the number of legitimate URLs correctly identified as non-phishing.

False Positive (FP): The number of instances incorrectly predicted as positive. This refers to the cases where non-phishing URLs are mistakenly classified as phishing.

False Negative (FN): The number of instances incorrectly predicted as negative. This includes phishing URLs that are mistakenly classified as non-phishing. When any model gives an accuracy rate of 99%, you might think that model is performing very good but this is not always true and can be misleading in some situations.

B. F1 Score

It is a harmonic mean between recall and precision. Its range is [0,1]. This metric usually tells us how precise (It correctly classifies how many instances) and robust (does not miss any significant number of instances) our classifier is

$$F1 = 2.\frac{Precision \times Recall}{Precision + Recall} \longrightarrow 6.4.2$$

Precision

There is another metric named Precision. Precision is a measure of a model's performance that tells you how many of the positive predictions made by the model are actually correct. It is calculated as the number of true positive predictions divided by the number of true positive and false positive predictions.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \longrightarrow 6.4.3$$

Recall

Lower recall and higher precision give you great accuracy but then it misses a large number of instances. The more the F1 score better will be performance. It can be expressed mathematically in this way:

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \longrightarrow 6.4.4$$

6.5. DEPLOYMENT OF MODEL ON WEBSITE:

We have developed a website using the Flask framework and deployed the pre-trained model for phishing website detection. Where Flask is a lightweight web framework for python that is commonly used for building web applications. The processes that are done during this module are as follows

1.Once trained the model, save the model to a file. Common formats include Pickle files (‘. pkl’) for scikit-learn models.

2.Set up a flask application. This involves creating a new python file (app.py)

that will serve the main entry point for the web applications.

3.Define routes in the flask application, Routes are URL pattern that map to specific functions.

4.Load the saved machine learning model within the flask application. We typically load the model when the application starts.

5.Create a function to use the loaded model to make predictions. This function will be called when a user submits data through the web interface.

6.Create HTML templates for rendering UI and extract the data from the request and pass it to the prediction function.

7.Display the results on the web page.

CHAPTER 7

EXPERIMENTAL RESULTS

In the pursuit of detecting phishing websites using machine learning-based URL analysis, a diverse set of supervised learning models were trained and evaluated on a dataset of labelled instances. Each model exhibited varying degrees of accuracy in their predictions, with the following result

	ML Model	Accuracy	f1_score	Recall	Precision
0	Logistic Regression	0.934	0.941	0.943	0.927
1	K-Nearest Neighbors	0.956	0.961	0.991	0.989
2	Support Vector Machine	0.964	0.968	0.980	0.965
3	Naive Bayes Classifier	0.605	0.454	0.292	0.997
4	Decision Tree	0.960	0.964	0.991	0.993
5	Random Forest	0.967	0.971	0.994	0.989
6	Gradient Boosting Classifier	0.974	0.977	0.994	0.986
7	CatBoost Classifier	0.972	0.975	0.994	0.989
8	CatBoost Classifier	0.972	0.975	0.994	0.989
9	Multi-layer Perceptron	0.972	0.975	0.993	0.984

Table 7.1. Performance Comparison

1. Logistic Regression (Accuracy: 93.4%):

Logistic Regression performed exceptionally well, achieving an accuracy of 93.4%. This algorithm is known for its simplicity and efficiency in binary classification tasks, making it a strong contender in this context.

2. k-Nearest Neighbors (Accuracy: 95.6%):

k-Nearest Neighbors demonstrated high accuracy at 95.6%. Its effectiveness lies in its ability to capture local patterns in the feature space, making it well-suited for discerning similarities between URLs.

3. Support Vector Machine (Accuracy: 96.4%):

The Support Vector Machine (SVM) yielded an impressive accuracy of 96.4%. SVM excels in finding optimal hyperplanes for separating classes, making it highly effective in distinguishing between phishing and legitimate URLs.

4. Naive Bayes Classifier (Accuracy: 60.5%):

The Naive Bayes Classifier achieved a comparatively lower accuracy of 60.5%. Despite its simplicity and efficiency, Naive Bayes assumes independence between features, which might not be the case in the complex relationships of URL features.

5. Decision Tree (Accuracy: 96.0%):

Decision Tree exhibited robust performance with an accuracy of 96.0%. Its ability to create hierarchical decision structures allows it to capture intricate relationships in the data, contributing to its high accuracy.

6. Random Forest (Accuracy: 96.7%):

The Random Forest model, an ensemble of decision trees, outperformed individual trees with an accuracy of 96.7%. This emphasizes the strength of ensemble methods in enhancing predictive performance.

7. Gradient Boosting Classifier (Accuracy: 97.4%):

The Gradient Boosting Classifier emerged as the top performer with the highest accuracy of 97.4%. Gradient boosting builds a series of weak learners, each correcting the errors of its predecessor, leading to superior predictive power.

8. CatBoost Classifier (Accuracy: 97.2%):

CatBoost, designed to handle categorical features efficiently, achieved an accuracy of 97.2%. Its ability to process categorical data while maintaining high accuracy makes it a valuable choice in this scenario.

9. Multi-Layer Perceptron (Accuracy: 97.2%):

The Multi-Layer Perceptron (MLP), a type of neural network, matched the accuracy of CatBoost at 97.2%. Its capacity to learn intricate patterns in data and adapt to complex relationships contributes to its high performance.

Explanation for Gradient Boosting's Superior Accuracy:

The observed superior accuracy of the Gradient Boosting Classifier can be attributed to its ensemble learning approach. Gradient boosting builds a series of weak learners sequentially, with each subsequent learner focusing on correcting the errors of its predecessor. This iterative refinement allows the model to capture complex relationships within the dataset effectively.

Gradient boosting's strength lies in its adaptability and capacity to handle intricate patterns in the data, making it particularly well-suited for tasks where nuanced relationships contribute to the overall predictive accuracy. In the context of phishing website detection, where identifying subtle patterns is crucial, the Gradient Boosting Classifier emerges as the most powerful model among those considered, showcasing its ability to outperform other algorithms in this specific domain.

FINAL OBSERVATION:

After training and evaluating nine different models, the Gradient Boosting Classifier stood out with an impressive accuracy of 97.4%. This exceptional accuracy suggests that the model performed exceptionally well in correctly classifying instances, both positive and negative.

The high accuracy indicates that the Gradient Boosting Classifier effectively learned complex relationships within the data and generalized well to unseen instances. This model is likely robust and could be a strong candidate for deployment in a real-world phishing detection system.

However, it's crucial to conduct a more in-depth analysis beyond accuracy. Precision, recall, and F1 score should be examined to understand how well the model performs specifically for phishing instances, as well as its ability to avoid false positives.

Additionally, consider exploring the model's feature importance. This insight can provide valuable information on which features contribute the most to the model's decision-making process. It's possible that certain features play a crucial role in distinguishing between phishing and non-phishing URLs.

To ensure the model's generalizability, perform cross-validation or use a separate test set not seen during training. This will help confirm that the model's high accuracy is not a result of overfitting to the training data.

CHAPTER 8
CONCLUSION AND FUTURE WORKS

In conclusion, the implementation of machine learning for phishing website detection, with a focus on the Gradient Boosting Classifier, has proven to be highly successful. The chosen model achieved an impressive accuracy of 97.4%, showcasing its efficacy in distinguishing between legitimate and malicious websites.

The decision to deploy this model on a live website underscores the confidence in its ability to accurately identify and flag potential phishing URLs in real-time. The high accuracy suggests that the Gradient Boosting Classifier has learned complex patterns and features indicative of phishing behaviour, making it a robust solution for protecting users from online threats.

The successful deployment of the Gradient Boosting Classifier reflects the promising potential of machine learning in enhancing cybersecurity measures. This approach not only provides an additional layer of defense against phishing attacks but also demonstrates the practical application of advanced technologies in securing online environments. Continuous evaluation and refinement of the deployed model will be key to maintaining its effectiveness and staying ahead of emerging cybersecurity challenges.

The future scope of this project is to investigate the application of deep learning techniques, such as neural networks, to capture intricate relationships in the data. Deep learning models may uncover complex patterns that traditional machine learning models might overlook.

REFERENCES

1. State of Cybersecurity Implications for 2016. An ISACA and RSA Conference Survey. [Online].Available: https://cybersecurity.isaca.org/csx-resources/state-of-cybersecurityimplications-for-2016. [Accessed: 09-Mar-2020].

2. Republic of Turkey, “National Cyber Security Strategy, 2016,” Ministry of Transport Martime Affairs and Communications.

3. R. Loftus, “What cybersecurity trends should you look out for in 2020?,” Daily English Global blogkasperskycom. [Online]. Available: https://www.kaspersky.com/blog/secure-futures-magazine/2020cybersecurity-predictions/32068/. [Accessed: 09-Mar-2020].

4. E. Buber, Ö. Demir and O. K. Sahingoz, "Feature selections for the machine learning based detection of phishing websites," 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, 2017, pp. 1-5.
5. “Retruster,” Retruster. [Online]. Available: https://retruster.com/blog/2019-phishing-and-email-fraud-statistics.html. [Accessed: 09-Mar-2020].

6. “Phishing Activity Trends Reports, 1st-2nd-3rd Half” APWG. [Online]. Available: https://apwg.org/trendsreports/. [Accessed: 09-Mar-2020].

7. Y. Cao, W. Han, and Y. Le, “Anti-phishing based on automated individual white-list,” Proceedings of the 4th ACM workshop on Digital identity management - DIM 08, pp. 51–60, 2020.

8. M. Sharifi and S. H. Siadati, “A phishing sites blacklist generator,” 2008 IEEE/ACS International Conference on Computer Systems and Applications, pp. 840–843, 2020.

9. M. Khonji, Y. Iraqi, and A. Jones, “Phishing Detection: A Literature Survey,” IEEE Communications Surveys & Tutorials, vol. 15, no. 4, pp. 2091–2121, 2018.

10. Y. Zhang, J. I. Hong, and L. F. Cranor, “Cantina,a content based approach to detecting phishing web sites” Proceedings of the 16th international conference on World Wide Web - WWW 07, pp. 639-648, 20017.

11. L. Wenyin, G. Huang, L. Xiaoyue, Z. Min, and X. Deng, “Detection of phishing webpages based on visual similarity,” Special interest tracks and posters of the 14th international conference on World Wide Web - WWW 05, pp. 1060-1061, 2015.

12. C. L. Tan, K. L. Chiew, K. Wong, and S. N. Sze, “PhishWHO: Phishing webpage detection via identity keywords extraction and target domain name finder,” Decision Support Systems, vol. 88, pp. 18–27, 2016.

APPENDIX: I

SOURCE CODE

LOADING DATA:

```
data = pd.read_csv("phishing.csv")

data.head()
```

SPLITTING THE DATA:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
random_state = 42)
X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

MODEL TRAINING:

```
# Gradient Boosting Classifier Model
from sklearn.ensemble import GradientBoostingClassifier

# instantiate the model
gbc = GradientBoostingClassifier(max_depth=4,learning_rate=0.7)

# fit the model
gbc.fit(X_train,y_train)

#predicting the target value from the model for the samples
y_train_gbc = gbc.predict(X_train)
y_test_gbc = gbc.predict(X_test)

#computing the accuracy, f1_score, Recall, precision of the model performance

acc_train_gbc = metrics.accuracy_score(y_train,y_train_gbc)
acc_test_gbc = metrics.accuracy_score(y_test,y_test_gbc)
print("Gradient Boosting Classifier : Accuracy on training Data:
{:.3f}".format(acc_train_gbc))
print("Gradient Boosting Classifier : Accuracy on test Data:
{:.3f}".format(acc_test_gbc))
```

```
print()

f1_score_train_gbc = metrics.f1_score(y_train,y_train_gbc)
f1_score_test_gbc = metrics.f1_score(y_test,y_test_gbc)
print("Gradient Boosting Classifier : f1_score on training Data:
{:.3f}".format(f1_score_train_gbc))
print("Gradient Boosting Classifier : f1_score on test Data:
{:.3f}".format(f1_score_test_gbc))
print()

recall_score_train_gbc = metrics.recall_score(y_train,y_train_gbc)
recall_score_test_gbc =  metrics.recall_score(y_test,y_test_gbc)
print("Gradient Boosting Classifier : Recall on training Data:
{:.3f}".format(recall_score_train_gbc))
print("Gradient Boosting Classifier : Recall on test Data:
{:.3f}".format(recall_score_test_gbc))
print()

precision_score_train_gbc = metrics.precision_score(y_train,y_train_gbc)
precision_score_test_gbc = metrics.precision_score(y_test,y_test_gbc)
print("Gradient Boosting Classifier : precision on training Data:
{:.3f}".format(precision_score_train_gbc))
print("Gradient Boosting Classifier : precision on test Data:
{:.3f}".format(precision_score_test_gbc))

#storing the results. The below mentioned order of parameter passing is
important.

storeResults('Gradient Boosting Classifier',acc_test_gbc,f1_score_test_gbc,
        recall_score_train_gbc,precision_score_train_gbc)
```

DEPLOYMENT OF MODEL ON FLASK:

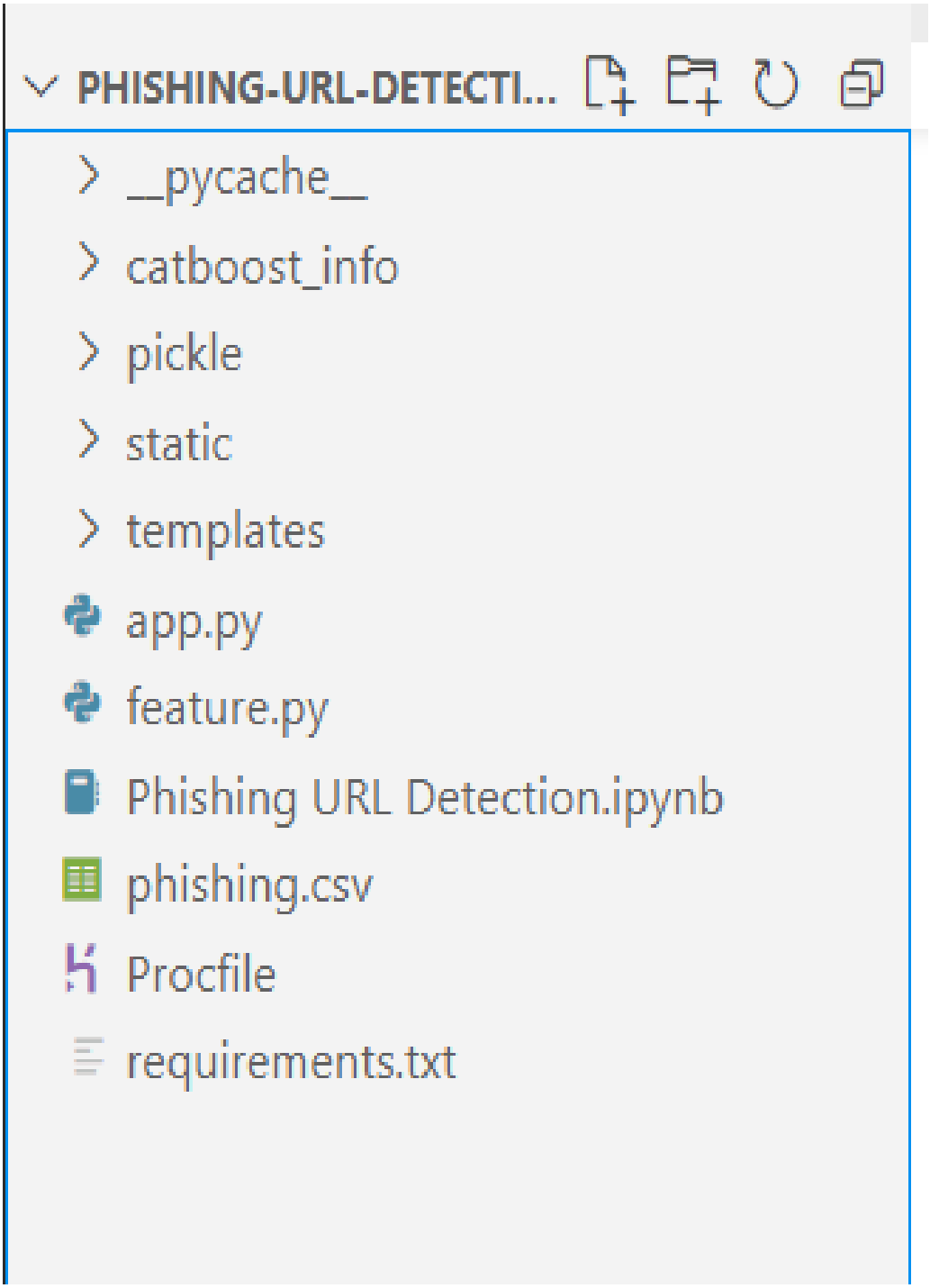
```
#importing required libraries

from flask import Flask, request, render_template
import numpy as np
import pandas as pd
from sklearn import metrics
import warnings
import pickle
warnings.filterwarnings('ignore')
```

```
from feature import FeatureExtraction

file = open("pickle/model.pkl","rb")
gbc = pickle.load(file)
file.close()


app = Flask(__name__)

@app.route("/", methods=["GET", "POST"])

def index():
    if request.method == "POST":

        url = request.form["url"]
        obj = FeatureExtraction(url)
        x = np.array(obj.getFeaturesList()).reshape(1,30)

        y_pred =gbc.predict(x)[0]

        #1 is safe
        #-1 is unsafe

        y_pro_phishing = gbc.predict_proba(x)[0,0]
        y_pro_non_phishing = gbc.predict_proba(x)[0,1]
        # if(y_pred ==1 ):

        pred = "It is {0:.2f} % safe to go ".format(y_pro_phishing*100)
        return render_template('index.html',xx
=round(y_pro_non_phishing,2),url=url )
    return render_template("index.html", xx =-1)


if __name__ == "__main__":
    app.run(debug=True)
```

APPENDIX: II

SCREENSHOTS

FILES IN THE PROJECT DIRECTORY:



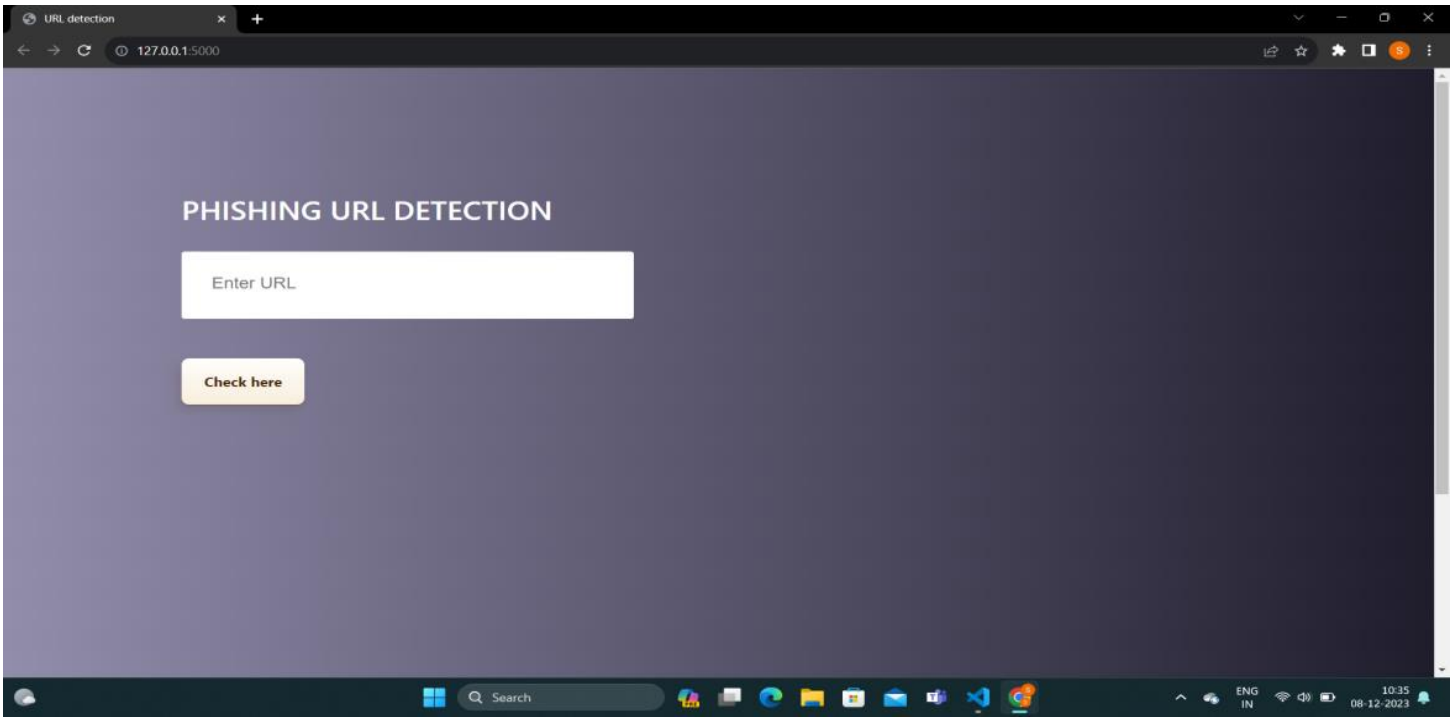


Fig 1. Landing page

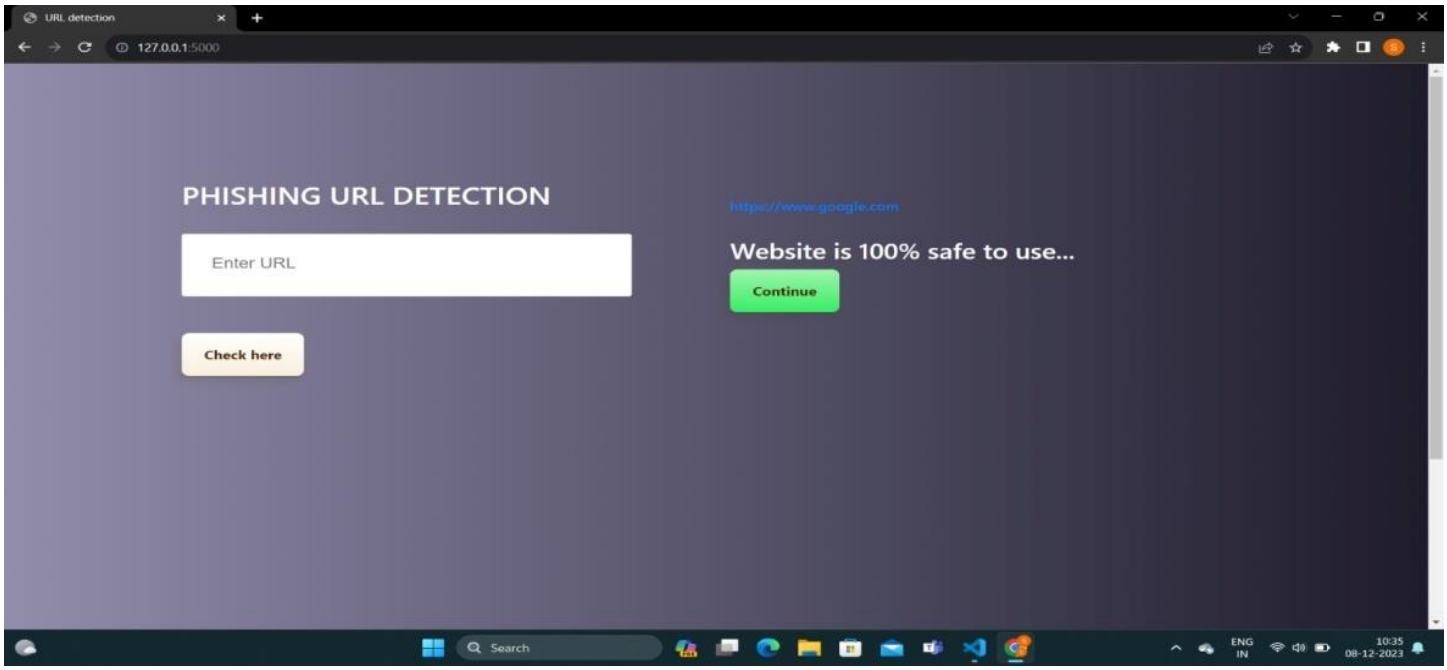


Fig 2. Upon entering the valid URL

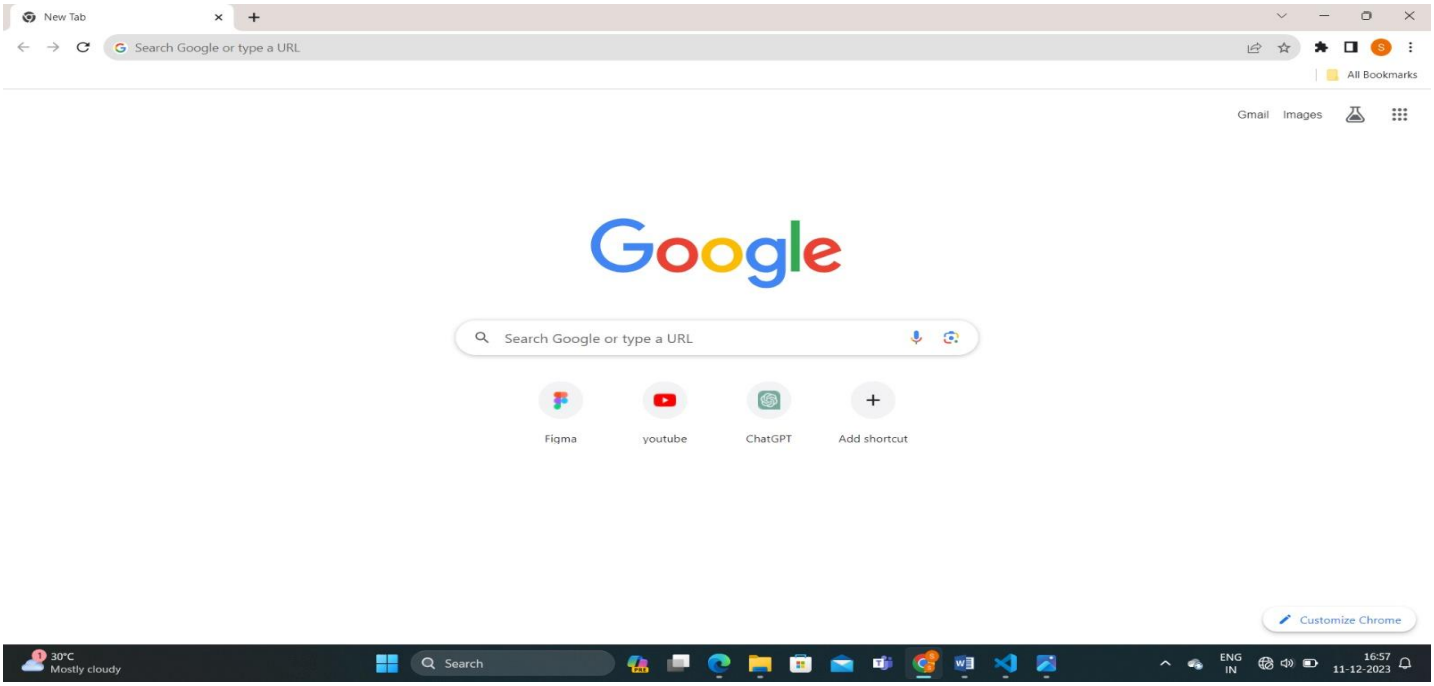


Fig 3. Navigating to the legitimate website

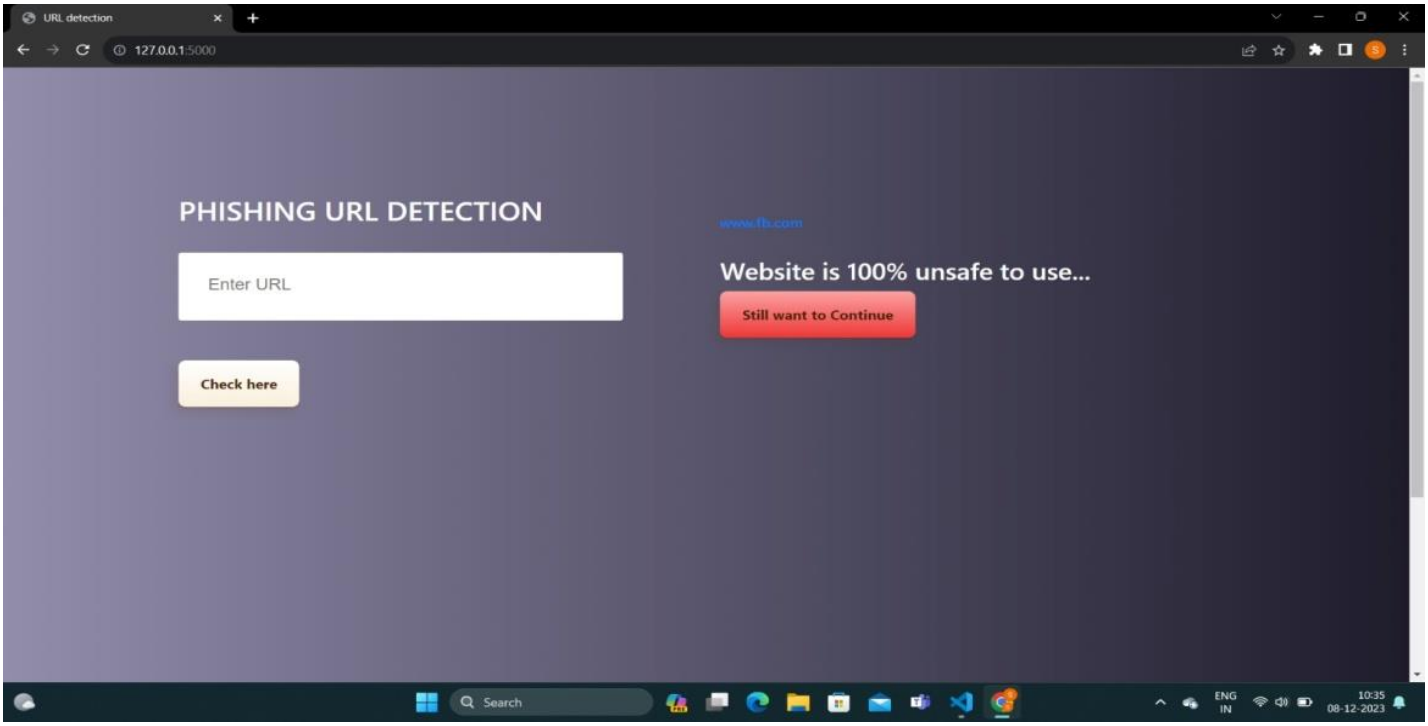


Fig 4. Upon entering invalid URL

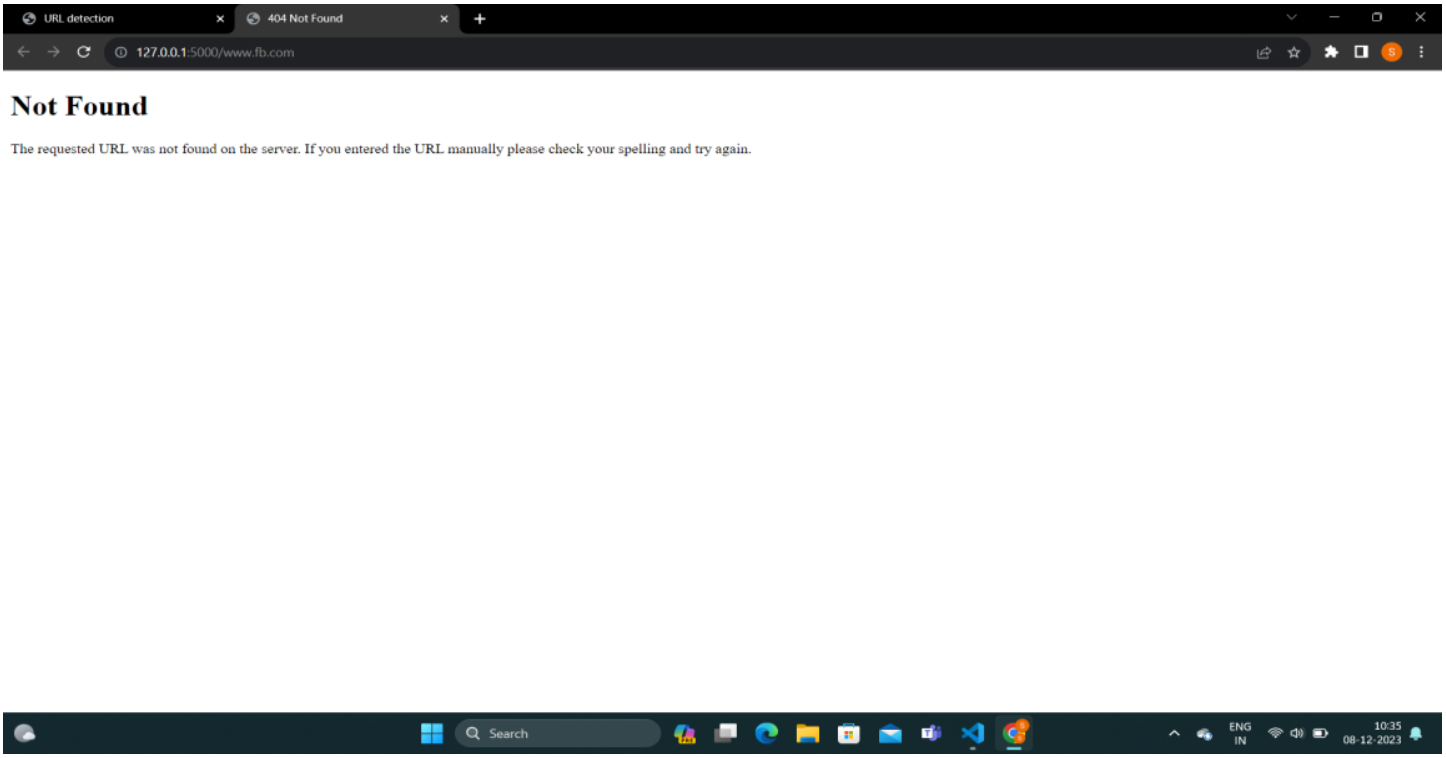


Fig 5. Navigating to the phishing website