PHISHING WEBSITE DETECTION USING MACHINE LEARNING

A PROJECT REPORT

Submitted by

MEENALOCHNE.S.(311019104048)

in partial fulfillment for the award of

the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

KCG COLLEGE OF TECHNOLOGY, KARAPAKKAM

ANNA UNIVERSITY: CHENNAI 600 025

MAY 2023

BONAFIDE CERTIFICATE

Certified that this project report titled **"PHISHING WEBSITE DETECTION USING MACHINE LEARNING"** is the bonafide work of **"MEENALOCHNE.S (311019104048)"** who carried out the project work under my supervision.

Dr. Cloudin S
HEAD OF THE DEPARTMENT
Professor
Dept. of CSE
KCG College of Technology
Karapakkam.

Dr.Kamatchi K S
SUPERVISOR
Assistant Professor
Dept. of CSE
KCG College of Technology
Karapakkam.

INTERNAL EXAMINER

EXTERNAL EXAMINER

ABSTRACT

Phishing attacks pose a significant threat to cybersecurity, and machine learning algorithms offer a promising approach to detect and prevent such attacks effectiveness of three machine learning algorithms, namely Logistic Regression, Support Vector Machine (SVM), and Random Forest, in detecting phishing websites as a means to address the significant threat posed by phishing attacks to cybersecurity.

Visualization techniques, such as word clouds, are also utilized to uncover key patterns and gain insights into the characteristics of phishing websites. Word clouds visually represent the most frequently occurring words in the dataset, helping researchers identify common terms or phrases associated with phishing attacks.

To facilitate serving the trained model and handling prediction requests, the FastAPI framework is employed. FastAPI is a high-performance web framework for building APIs, and it offers features that enable seamless integration of machine learning models. Additionally, modules like uvicorn and joblib are used to serve the trained model and ensure its availability for making predictions.

Overall, aim to deliver accurate predictions and valuable insights to enhance the detection of phishing websites. By comparing different machine learning algorithms, employing effective data preprocessing techniques, and utilizing appropriate evaluation metrics, the study seeks to contribute to the development of robust cybersecurity solutions.

ACKNOWLEDGEMENT

We thank the Almighty GOD for the abundant blessings showered on us. We extend our deepest love and gratitude to our dear parents who built up our career and backed us up in life.

We thank our management and our Principal **Dr. P Deiva Sundari** for the opportunities given to us for our career development.

We feel indebted to the Head of the Department **Dr. Cloudin S,** Professor, Department of Computer Science & Engineering, KCG College of Technology, for all his encouragement, which has sustained our labour and efforts.

We express our deepest gratitude to the internal guide **Dr Kamatchi K S,** Assistant Professor, Department of Computer Science & Engineering, KCG College of Technology, for her valuable guidance, ideas and support.

We extend our sincere thanks to the project coordinator **Mr. Rajesh George,** Associate Professor for his guidance and support.

We would like to thank all other faculty members of Department of Computer Science & Engineering, for their help and advice throughout our life in this campus.

Finally, we are thankful to all our friends and all others who encouraged us and helped us in does project.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

URL	Uniform Resource Locator
MCS	Functional Network Connectivity
SVC	Support Vector Classifier
CNN	Convolutional Neural Network
LR	Logistic Regression
ML	Machine Learning
KNN	K-means Clustering
AUC	Area Under the Curve
API	Application Programming Interface.

CHAPTER 1
INTRODUCTION

Cyber attacks have become increasingly sophisticated and widespread, posing a major threat to individuals and organizations alike. One type of cyber attack that has become particularly prevalent is phishing, which involves tricking the victim into providing sensitive information, such as login credentials and financial data, by disguising as a trustworthy entity. [5] Phishing attacks can cause significant harm, including identity theft, financial loss, and damage to an organization's reputation.One common way phishing attacks are carried out is through the use of phishing URLs, which are hyperlinks that lead to fake websites designed to steal information. These phishing URLs can be difficult to identify, as often resemble legitimate websites, making easy for victims to fall for the scam. This makes the detection and prevention of phishing URLs an important aspect of cybersecurity.

[7] To address this problem, the phishing URL detection project was undertaken to develop a solution for real-time phishing URL detection. The solution utilizes a combination of techniques, including URL blacklists, reputation analysis, machine learning, and human analysis, to identify and block phishing URLs in real-time. The goal of this project is to provide a robust and effective solution that can protect individuals and organizations from the devastating effects of phishing attacks.In this report, will discuss the methods used in the phishing URL detection project, the results of analysis, and the significance of these results. By providing a comprehensive overview of the phishing URL detection project, [6][2][8] paper will serve as a valuable resource for anyone interested in improving their cybersecurity and protecting themselves from phishing attacks.These days Phishing is a prevalent type of cyberattack where an attacker poses as a trustworthy entity to deceive a target into divulging sensitive information, such as credit card numbers, passwords, and personal data. These attacks can have severe consequences, including financial loss, identity theft, and reputational damage. Moreover, phishing attacks often serve as a conduit for more sophisticated cyberattacks, such as the dissemination of ransomware and malware. As phishing attacks become increasingly frequent and complex, to make is essential to have robust phishing monitoring systems in place to protect individuals and businesses.

[1]Phishing attacks can result in various security concerns beyond just financial loss, including identity theft, viruses, and other related issues. These attacks are becoming more challenging to detect and prevent, posing a risk to both individuals and organizations security and privacy. As cybercriminals become more sophisticated, traditional methods of detecting phishing attacks, such as filters and signatures, are becoming less effective. Hence, researchers have been investigating computer-based methods that utilize machine learning techniques to identify instances of phishing.

[2][4][5] paper examines the different strategies that researchers have developed to recognize and thwart phishing attacks using machine learning algorithms. To explore potential future improvements to enhance this process and contribute to the creation of better anti-phishing solutions. The goal is to highlight the need for strong phishing monitoring systems and to provide insights into how machine learning algorithms can help detect and prevent these attacks.

According to the 2021 FBI report, phishing has become the most prevalent online crime in the United States, with over 240,000 complaints and $54 million in losses. Additionally, recent incidents such as the cyberattack on AIIMS in India and the healthcare industry's primary targeting in India highlight the need for effective anti-phishing solutions. In this context, Kaspersky's anti-phishing technology thwarted over 500 million attempts to access fake websites in 2022, representing a significant increase from the previous year. The new report by Kaspersky titled "Spam and Phishing in 2022" provides various insights and analysis into these attacks.

1.1 OVERVIEW

Reviewing the origins, varieties, and effects of phishing assaults on people and organizations would be necessary. In order to do this, existing research on the techniques for detecting phishing URLs, such as URL blacklists, reputation analysis, machine learning, and human analysis, will be reviewed. This would entail looking through studies that have compared the performance of various phishing URL detection techniques. This would entail going through the state of the art in real-time phishing URL detection research as well as the difficulties in

putting into practice. In order to do this, would be necessary to evaluate the body of knowledge regarding the incorporation of phishing URL detection into a variety of systems, such as email filters, web browsers, endpoint security programmes, and network security programmes.The visual resemblance of the web page is evaluated in the visual similarity-based approach. Since users are unable to clearly distinguish small differences between fake and real web pages, is easier to detect them by utilizing image processing algorithms.

By utilizing a model that has been trained on a sizable dataset of well-known legitimate and phishing URLs, machine learning can categorize new URLs as either valid or phishing. This technique has the potential to be quite powerful, but needs a sizable and varied dataset to train the model, and the might not work against brand-new, untested phishing URLs. Lists of known phishing URLs that have been reported and verified are called URL blacklists. These blacklists can be used by the phishing URL detection system to recognise and restrict access to phishing URLs.

This is a straightforward and efficient solution, but in order to stay current with new phishing URLs, this needs to be updated frequently. Reputation analysis entails assessing a website's reputation in light of several characteristics, including its age. Reputation analysis involves determining a website's reputation based on a variety of variables, including the age of the domain, the volume of links referring to the site, and user evaluations. Reputation analysis can be used by the phishing URL detection system to find suspicious websites and assess whether or not the are likely to be phishing URLs.

1.2 OBJECTIVE

A system should be created that can quickly recognise phishing URLs and restrict people from viewing them. The system should be able to recognise legitimate URLs from phishing attempts in real-time and issue alerts or prevent access to known phishing URLs. Some of the project's specific goals could be a collection of well-known phishing and valid URLs, to create a machine learning model that can accurately categorize new URLs as either phishing or legitimate. To assess the model's effectiveness using metrics like accuracy, precision, recall, and F1 score and to progressively enhance the model's effectiveness over time.

To include the machine learning model, a phishing URL detection system that may be used to defend people against phishing assaults in real-world settings.to create a procedure for extracting useful data from each URL and using those features as inputs for a machine learning model. to investigate and contrast the effectiveness of several machine learning techniques for phishing URL identification, including decision trees, random forests, and neural networks. The project's ultimate goal is to develop a very effective method of identifying phishing URLs that can be used to shield people and businesses from the harm that phishing attempts can do . To investigate and contrast the effectiveness of several machine learning techniques for phishing URL identification, including decision trees, random forests, and neural networks. The project's ultimate goal is to develop a very effective method of identifying phishing URLs that can be used to shield people and businesses from the harm that phishing attempts can cause.

1.3 CHALLENGES IN THE DOMAIN

To create a useful solution, is necessary to address a number of issues in the field of phishing URL detection. Several of these difficulties include:

Evasion: In an effort to avoid being discovered, phishing attackers are continually improving their strategies and methods. Because of this, phishing URL detection systems must be extremely flexible and capable of spotting new and undiscovered phishing URLs.

Data Unbalance: Only a tiny fraction of URLs are phishing URLs, whereas the great majority of URLs are valid. Due to this, the dataset used to train the machine learning model becomes uneven, which might have a negative effect on the model's performance.

Limited Data Availability: Especially when dealing with fresh and unknown phishing URLs, can be challenging to compile a sizable and diversified dataset of legitimate and phishing URLs. This may reduce the machine learning model's capacity to precisely identify phishing URLs.

False Positives: Systems for detecting phishing URLs may yield false positives, in which case genuine URLs are mistakenly labeled as phishing URLs. Both the user experience and the system's reputation may suffer as a result.

Human Factor: Social engineering techniques are frequently used in phishing attacks to deceive people into clicking on phishing URLs. To challenging to fully automate phishing URL detection because of the human component, and systems must be created with the end user in mind.

This takes a multifaceted strategy to overcome these obstacles, integrating machine learning with other methods like URL blacklists, reputation research, and human analysis. It is possible to create a very effective phishing URL detection system that can shield consumers from the negative impacts of phishing attempts by creating a comprehensive solution that tackles these problems.

1.4 MOTIVATION

The increased threat posed by phishing assaults is what spurred the development of a machine learning project for phishing URL detection. Phishing is a type of cybercrime that includes convincing people to divulge private information, including passwords and financial information, by pretending to be a reliable organization. Phishing attacks are becoming more sophisticated and difficult to spot, making tougher for people and organizations to defend themselves against these dangers. Effective phishing URL detection systems that can instantly

A crucial instrument in the battle against cybercrime, the use of machine learning can considerably increase the efficacy and accuracy of phishing URL detection. A successful phishing URL detection project using machine learning can assist to improve the field of cybersecurity and knowledge of how to successfully detect and prevent cyberattacks in addition to shielding people and organizations from the negative impacts of phishing attempts. The overall goal of a machine-learning-based phishing URL detection project is to provide a highly effective solution that can shield people and businesses from the rising threat of phishing assaults while also making a significant contribution to the field of cybersecurity.

1.5 ORGANIZATION OF THE REPORT

This report is organized as follows: Chapter 1 consists of overview, objective and motivation about the project. Chapter 2 consists of the related works on the vehicle safety techniques, and other tools and methodology used to achieve it. Chapter 3 consists of the system analysis which defines the problem definition, various software components and use cases. Chapter 4 describes the system design which consists of the system architecture for the proposed system and module description for each module. Chapter 5 talks about the system implementation and the various algorithms. Chapter 6 describes the system testing which consists of different test cases for the various module . The Chapter 7 talks about the various results and discussion obtained from the various outputs. Chapter 8 consists of conclusion and the future work for the project. The Appendix 1 consists of the code and software implementations. The Appendix 2 consists of the technical paper presentation and the references.

CHAPTER 2

RELATED WORK

2.1 LITERATURE REVIEW

Several studies that use the aforementioned algorithms are evaluated and their findings are summed up in this section.

[1]Aniket Garje1 , Namrata Tanwani1 , Sammed Kandale1 , Twinkle Zope1 , Prof. Sandeep Gore2"Detecting Phishing Websites Using Machine Learning"© 2021 IJCRT | Volume 9, Issue 11 November 2021 | ISSN: 2320-2882 -This paper explores machine learning techniques like KNN, Naive Bayes, and Decision Tree for detecting phishing websites. Analyze accuracy, precision, recall, and ROC curve for different algorithms. Decision Trees show promising performance with high accuracy (0.94) and balanced recall and precision, making them a preferred choice.

[2]Arathi Krishna V*, Anusree A, Blessy Jose, Karthika Anilkumar, Ojus Thomas Lee "Machine learning to detect phishing using URL analysis A survey"- presented different techniques used to us in phishing detection as well as datasets and URL features used to train the machine learning models, a goal in this work is to review several machine learning techniques utilized for this purpose. To examine and analyze how various machine learning algorithms perform as well as the techniques employed to raise their accuracy measurements. The objective is to establish a survey resource for academics to learn about recent advancements in the industry and help develop phishing detection models that produce more reliable findings.Phishing; URL features; machine learning; phishing detection. Machine Learning: In this method, URLs are analyzed and classified as either phishing or authentic using machine learning algorithms. When paired with additional methods like URL blacklists and reputation monitoring, machine learning can be incredibly effective at spotting phishing URLs.

[3] Dipayan Sinha, Dr. Minal Moharir, Prof. Anitha Sandeep,“Phishing Website URL Detection using Machine Learning,”International Journal of Advanced Science and Technology,29(3):2495-2504, January (2020):Software can detect fraudulent websites using Logistic Regression, Decision Tree, Random Forest, Adaboost, Gradient Boosting, Gaussian NB, and Fuzzy pattern tree classifier. People use fake or real websites to gather information. Feature extraction involves identifying elements such as IP address, symbols such as "@" and dashes, URL length, dot quantity, and sub-domains within a web address. Website ranking, age, and authenticity. 80% for learning, 20% for verification. Logistic Regression has 96% accuracy and a 95% recall and F1 score.

[4] Sahingoz, O. K., Buber, E., Demir, O., & Diri, B. “MachineLearning-Based Phishing Detection from URLs,” ExpertSystems with Applications, 117:345-357, January 2019 ” -The dataset used in this study was self-constructed, with phishing websites sourced from PhishTank and legitimate URLs obtained from Yandex Search API. The primary objective was to detect words that resemble brand names and keywords composed of random characters. Various classification algorithms, including Naive Bayes, Random Forest, kNN(n=3), Adaboost, K-star, SMO, and Decision Tree, were employed, along with feature extraction techniques such as NLP-based features, Word Vectors, and Hybrid. The system demonstrated high accuracy during the testing phase.

[5]Salvi Siddhi Ravindra1, Shah Juhi Sanjay1, Shaikh Nausheenbanu Ahmed Gulzar1, Khodke Pallavi2 “Phishing Website Detection Based on URL”-presented different techniques used to us in phishing detection During to the increased use of the internet and other online platforms in the modern period, security has received a lot of attention. Every day, there are several cyberattacks, with website phishing being the most prevalent problem. The involves pretending to be a trustworthy website in order to deceive users and obtain their private information. In light of this issue, this article will present a potential fix to prevent such attacks by determining whether the provided URLs are genuine URLs or phishing URLs.

[6] S. Carolin Jeeva1* and Elijah Blessing Rajsingh”Intelligent phishing url detection using association rule mining” -Phishing is an online criminal act that occurs when a malicious web page impersonates a legitimate webpage so as to acquire sensitive information from the

user. Phishing attacks continue to pose a serious risk for web users and an annoying threat within the field of electronic commerce. This paper focuses on discerning the significant features that discriminate between legitimate and phishing URLs.

[7] Taizhen Wang et al"Phishing Detection Based on Machine Learning Algorithms: A Systematic Literature Review". (2021) These features are then subjected to associative rule mining—apriori and predictive apriori. The rules obtained are interpreted to emphasize the features that are more prevalent in phishing URLs. Ana- lyzing the knowledge accessible on phishing URL and considering confidence as an indicator, the features like transport layer security, unavailability of the top level domain in the URL and keyword within the path portion of the URL were found to be sensible indicators for phishing URL. In addition to this number of slashes in the URL, dots in the host portion of the URL and length of the URL are also the key factors for phishing URL.

[8]Yiming Liu et al."Phishing Detection Using Support Vector Machine and Logistic Regression" (2020) Machine learning-based system, with supervised learning being its specialization, and has provided datasets for 2000 phishing and 2000 legal URLs. Because of the performance and accuracy of the Logistics Regression Algorithm, as have taken into consideration. takes into account 9 features,Because of the performance and accuracy of the Logistics Regression Algorithm, has been taken into consideration. Determines whether a URL is safe to browse or a phishing URL by taking into account nine criteria.

2.2 EXISTING TECHNOLOGY

The performance assessment in this case is carried out for 100 and 500 data. The suggested MCS-DNN has a FNR of 6.21% (for 100 data), although when compared to the existing ones, their values are noticeably higher. The system performance suffers with higher FNR values. As a result, to believe that the suggested MCS-DNN performs at a high level. The proposed one also has a lower FPR value when looking at FOR value analysis. FPR for the MCS-DNN is 4.21%.

When compared to previous strategies, the suggested MCS-DNN produces a lower FPR of 4.32% and FNR of 4.98% for 500 data, as shown in Fig. 7d. Consequently, may be concluded that theTherefore, can be concluded that when compared to the current, the suggested MCS-DNN shows a significant improvement. This compared the suggested MCS-performance DNN's with that of the existing.

The system performs better due to the low FNR and FPR values. The existing ANFIS, KNN, ANN, and SVM, which are greater when compared to the suggested FPR, have respective FPRs of 22.12%, 17.23%, 10.45%, and 12.78%, despite the perception that the proposed achieves 4.21%. The suggested MCS-DNN has a FNR of only 5.78%, while the existing approaches have a greater FNR. For 100 data, this analysis is shown in Fig. 8c.When compared to the suggested MCS-DNN with 500 data, existing techniques have a higher FNR of 4.21% and FPR of 3.05%.

As a result, this concluded that the suggested MCS-DNN performs better than the ones that are currently in use. Similar to this, the effectiveness of the suggested and current techniques for the 200, 300, and 400 data sets is examined. The existing MCS-DNNs perform poorly on all 200, 300, and 400 data while the suggested one produces superior results. When cross-validation was utilised for the evaluation, the hybrid model that combines the DNN and BiLSTM algorithms delivered 98.79% accuracy, 0.9878 AUC, and 0.9881 F1-score on the Ebbu2017 phishing dataset.

The KNN algorithm-based model exhibited 99.21% accuracy, 0.9934 AUC, and 0.9941 F1-score on the Phishtank dataset.The DNN-LSTM hybrid model, which has a 98.62% accuracy in the Ebbu2017 dataset and a 98.98% accuracy in the PhishTank dataset, comes in second to the KNN model. The CNN-based classifier was the model that performed the worst on the Ebbu2017 dataset. The accuracy, AUC, and F1-Score of the CNN-based classification model were 93.25%, 0.9326, and 0.9339, respectively. On the other hand, the DNN-based classifier, which achieved 91.13% accuracy, was the lowest classifier for the PhishTank dataset. This article offers machine learning-based phishing detection.

Additionally, classifiers produced by machine learning algorithms recognise authentic phishing sites. With an accuracy of 95.66% and an extremely low false-positive rate, the

suggested technique utilised SVM. The suggested method can stop the harm done by phishing assaults and find new temporary phishing sites. The performance of the suggested machine learning-based solution is superior to earlier phishing detection technologies. To be beneficial to look at the effects of feature selection using other classification methods in the future.

In International Journal on Semantic Web and Information Systems, Volume 18 • Issue 1, the suggested a phishing detection model that successfully detects phishing output by the use of semantic features for word insertion. On Chinese websites, there are seven semanticized features and numerous statistics characteristics.

To acquire statistical aspects of web pages, eleven features were retrieved and divided into five categories.AdaBoost, Bagging, Random Forest, and SMO are utilized for the model learning and testing processes. The Anti-Phishing Alliance of China acquired valid URLs from phishing information and direct industrial web guides. The study demonstrates that the very effective phishing sites and the fusion model were only successfully constructed with semic features and were able to obtain the best output detection.

The researchers used machine learning and image surveillance to take a hybrid strategy. The requirement for a prior awareness (web history) of the website or an initial database for images is a key constraint to the detection of photo and visual phishers. The three sorts of accessibility—third-party applications, hyperlink features, and URL obscenity—were all used.The accuracy of the system improves to 99.55% when the time taken to identify an attack increases due to the utilization of resources by other parties.

2.3 INFERENCE OF LITERATURE REVIEW

Table 2.1 Inference of Literature Review

Title	Contribution
Machine learning to detect phishing using URL analysis A survey [1]	The objective is to establish a survey resource for academics to learn about recent advancements in the industry and help develop phishing detection models that produce more reliable findings.
Phishing Website Detection Based on URL[2]	To Present a potential fix to prevent such attacks by determining whether the provided URLs are genuine URLs or phishing URLs. Machine learning-based system, with supervised learning being its specialization, and has provided datasets for 2000 phishing and 2000 legal URLs
Intelligent phishing url detection using association rule mining [3]	The features like transport layer security, unavailability of the top level domain in the URL and keyword within the path portion of the URL were found to be sensible indicators for phishing URL. In addition to this number of slashes in the URL

2.4 EXTRACTION FROM LITERATURE REVIEW

Internet users are at serious risk from phishing. Web security faces a significant issue as a result of the rapid development and spread of phishing techniques. A wholly original content approach was suggested by Zhang et al.

[1] For locating phishing websites. The highest ranking keywords are discovered by looking at a webpage's source code and using TF-IDF. The highest ranking keywords are

discovered by looking at a webpage's source code and used .The collected keywords are entered into the Google search bar, and the URL's domain name is checked to see if it matches one of the top search results, at which point is deemed to be valid. The Google search engine is wholly dependent on this strategy. is an improved version that includes new features to produce better results, as proposed by Xiang et al.

[2] Is an improved version that has new features to produce better outcomes. The authors specifically mention the Document Object Model, as well as third-party and Google search engines that use machine learning to recognize phishing web pages. Due to the rise in cyberattacks, including phishing assaults on websites and SQL injection, etc. People's personal information is in jeopardy and at risk. Therefore, is necessary to safeguard the user's sensitive data, such as bank account and credit card information. And as a result of these attacks, a user can fall victim to phishing and lose login information.

[3] Is necessary to develop a system that would protect the user and data from these threats. Therefore, the proposed solution will prevent hackers from hacking. The user will be made informed of valid emails and phishing scams so they can protect themselves.

CHAPTER 3

SYSTEM ANALYSIS

3.1 PROBLEM DEFINITION

The primary objective of countering phishing attacks is to safeguard individuals and businesses from deceitful and malicious techniques employed to trick victims into divulging sensitive information or compromising their security. the aim is to prevent and mitigate the negative impacts of these cyber risks through preventive measures such as education, awareness, and cutting-edge technology. However, distinguishing between malicious and legitimate URLs for the purpose of detecting URL phishing poses a significant challenge. To assess the risk and potential harm posed by these URLs, to crucial to develop sophisticated algorithms and methodologies that can efficiently analyze and detect various properties and characteristics of the URLs, including their domain name, structure, and content.

The project's goal is to enhance user security by issuing timely email or pop-up alerts that warn users about potentially hazardous URLs and the risk of disclosing personal information. This will be achieved by implementing advanced detection methods capable of accurately identifying and categorizing unsafe URLs based on factors such as reputation, history, and content, and subsequently issuing appropriate warnings to users. The dataset has been completed, and a model utilizing Logistics Regression Algorithms has been created for predictive analysis after extensive testing and training through multiple iterations. The model has achieved a high level of accuracy in identifying and predicting outcomes based on numerous input factors, following thorough examination and refinement. Users are presented with the final output of the model, which provides insightful commentary and practical advice.

3.2 PROPOSED SYSTEM

Phishing detection systems have been proposed with different attributes to improve their effectiveness in recognizing and preventing phishing attacks. However, existing approaches in the literature tend to be specialized and focused on specific elements, such as attack methods, security environments, data collection systems, detection techniques, and delivery devices. None of these solutions have demonstrated an integrated strategy that covers various settings (e.g., mobile and desktop) and combines different aspects.

The lack of integration is primarily due to the complexities introduced by the dependencies and interdependencies between multiple detection methods. Each solution often has specific requirements and may face challenges when integrating with other third-party software, like antivirus programs. Removing a particular detection technique can become difficult when is intertwined with other components. Therefore, careful consideration of compatibility and integration is crucial to minimize such issues.

Another limitation is the difficulty of adapting a detection technique to different contexts, which can be exploited by threat actors. For instance, a detection technique designed for phishing may not be effective in identifying other threats like botnets. Flexibility and adaptability in detection techniques are necessary to address the evolving threat landscape and mitigate risks effectively.

Integrating diverse detection techniques into a unified tool is challenging due to the complexities involved in merging different targets, sources, and approaches. However, to essential to overcome these challenges in order to create a robust and comprehensive phishing detection system.

Creating an effective phishing detection system requires careful consideration of factors such as compatibility, interoperability, and data integration. Integrating multiple techniques can result in increased resource utilization and maintenance costs, to essential to assess the benefits and drawbacks before implementation.

1. Independence from platform and context: The strategy implements a proxy-based architecture capable of intercepting any network pattern and being deployed on both mobile and

desktop devices. It can effectively intercept and analyze traffic to and from the Internet in various contexts.

2. Flexibility, multimodality, and multiple functions: The framework extends detection methods like blacklists, heuristics, and machine learning to other malware detection contexts, ensuring adaptability. These methods can be enabled or disabled as needed, considering the presence of third-party software like antivirus clients.

3. Scalability: The method can handle diverse rates and types of connections required by user applications, ensuring it does not become a bottleneck for high-speed and low-latency network connections. Computationally intensive client tasks can be offloaded to a cloud platform, reducing workloads.

The ultimate goal of this research is real-time identification of phishing attacks. The client-side application consistently checks with the machine learning server to determine the safety of websites or apps.

3.3 SOFTWARE REQUIREMENTS

Development & Usage:

Programming Language: The project utilized Python 3.6.0 for machine learning. Python was chosen due to its versatility and extensive data processing capabilities, making it well-suited for handling large datasets and complex analysis in phishing detection projects.

Operating System: The project required Windows 8.1 or above as the operating system. This choice ensured compatibility with the necessary software and libraries used in the project, enabling efficient processing and analysis of large datasets.

IDE: The preferred Integrated Development Environment (IDE) for this project was Anaconda. iPython version 3.x was utilized for coding and analysis. Anaconda provides a user-friendly interface and efficient processing capabilities, making it suitable for the complex data processing and analysis tasks involved in phishing detection projects.

Client Operating System: FastAPI, a modern and high-performance web framework for

building APIs with Python 3.7+, was used for the client operating system. Visual Studio Code (VS Code), a popular code editor, provided a rich development environment for Python.

Dataset: The project utilized a CSV dataset, specifically the URL detection dataset. This dataset is a valuable resource for training machine learning models in phishing detection as it contains millions of verified phishing URLs. Researchers and security professionals can use this dataset, provided by Phish Tank, as a simulation tool for testing their phishing detection systems with real-world phishing data.

3.4 USE CASES

A use case diagram is a critical tool for capturing the dynamic behavior of a phishing detection system. The provides a simplified and graphical representation of the system's functionality and portrays the interactions between different users and the system.

In the context of phishing detection, a use case diagram typically includes use cases such as "user enters login credentials," "system checks for phishing indicators," and "system provides a warning message in case of a phishing attempt." These use cases capture the essential functionality of the system and visually depict how different elements of the system interact to detect and prevent phishing attacks.

Use case diagrams often coexist with textual use cases and other diagram forms, such as sequence diagrams and activity diagrams. Together, these technologies offer a thorough insight into the behavior of the system and help ensure that meets the needs of its users.

As mentioned in Figure 3.4 Use Case will demonstrate the flow of the Phishing Attack Detection project in real-time. The user enables the app on their phone or device, opens the browser, and visits a website. If the system detects phishing indicators, displays a warning message to the user, helping them avoid potential phishing attempts directly on the current page.

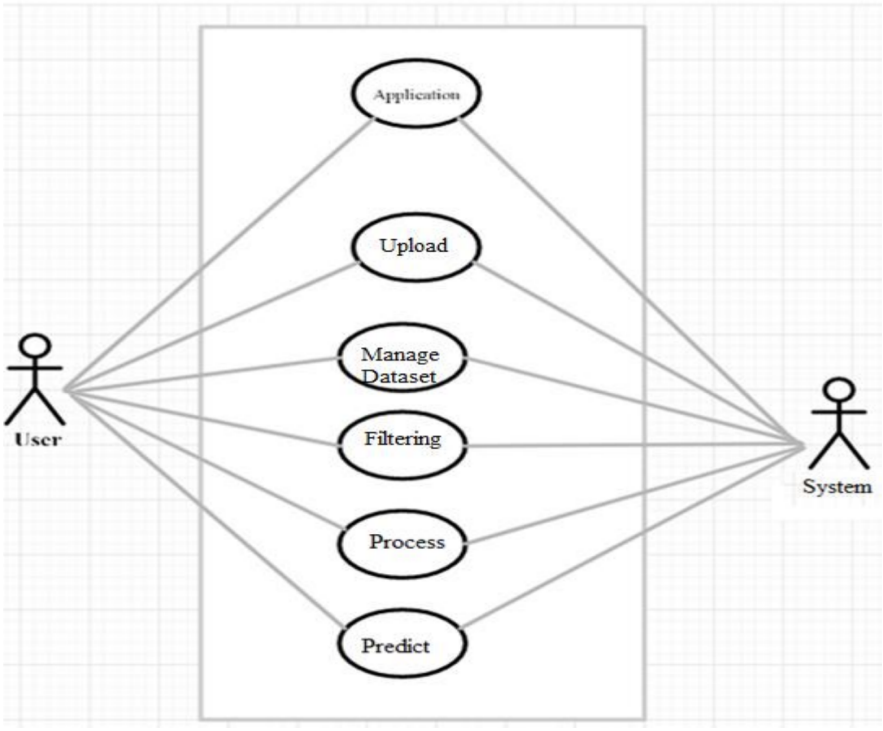

Figure 3.1 Uses Case Diagram

Real-time Phishing Detection: Leveraging machine learning, this system provides real-time detection of phishing attacks by analyzing the content and URLs of incoming emails, instant messages, or social media posts. By scrutinizing suspicious links and messages, the system can promptly notify users or restrict access to protect against phishing attempts.

Website Inspection: This system facilitates thorough website inspections to identify potential phishing threats, such as suspicious URLs or HTML tags. Website owners and administrators can utilize this information to take corrective actions and enhance user protection measures.

Email Filtering: Employing machine learning techniques, the system effectively filters out phishing emails from an organization's email system. By preventing employees from falling victim to phishing attacks, significantly reduces the risk of data breaches and associated consequences.

Fraud Detection: With its machine learning capabilities, the system enables the detection of fraudulent transactions, including instances involving stolen credit card information. By analyzing transaction data and recognizing patterns indicative of fraud, to enhance security measures and safeguards against financial losses.

Usage Scenario

The typical application scenario for a machine learning-based phishing detection system entails the following steps:

1. Dataset collection :

The dataset was downloaded from Kaggle and contains 549,346 entries. The has two columns, namely URLs and Categories. The URLs column displays all the predicted URLs, while the Categories column categorizes them as either good or bad. The dataset consists of a total of 549,346 URLs, with good URLs accounting for 72% and "bad" URLs accounting for 28%. "Good" URLs indicate that the URLs do not contain malicious content and are not phishing sites, while bad URLs indicate that the URLs contain malicious content and are phishing sites. There are no missing values in the dataset.

2. Feature extraction :

URL domain: The domain is the name of the website more often than not enlisted by the phisher, whereas the first space being phished is a portion of the way, the inquiry or the upper level space.

URL keywords: use 1-2 keywords in google what the page should show up for in search results .

Long domains: The URL is the space being phished but incorrectly spelled, with letters or words lost or included. The focus on brand can too be combined with other words to form an unregistered space.

URL IP address: a unique label assigned to websites and servers as well as digital

devices .

URL shortener:allow to reduce long links from Instagram, Facebook , linked in and more

3. Model Training:

Using the extracted features from both phishing and legitimate examples within the identified dataset, the system trains a machine learning model.

4. Real-time Detection

Whenever an email or website is accessed, the system analyzes its attributes and employs the trained model to predict whether it represents a phishing attack. If a phishing attack is detected, the system can either block access to the website or promptly notify the user.

SEQUENCE DIAGRAM

A machine learning-based phishing detection system serves as a valuable tool for organizations and individuals seeking robust protection against phishing attacks. By efficiently analyzing extensive volumes of data and identifying subtle patterns, it effectively prevents data breaches and safeguards sensitive information.

As mentioned in figure 3.5 System Sequence Diagram is a type of UML diagram that focuses on capturing the interactions between external actors and the system. It represents a specific scenario of a use case and shows the sequence of events or messages exchanged between the actors and the system. The primary purpose of an SSD is to illustrate the inputs and outputs of the system during a particular use case scenario. While cannot create visual diagrams, can help to understand the elements and relationships in an SSD.

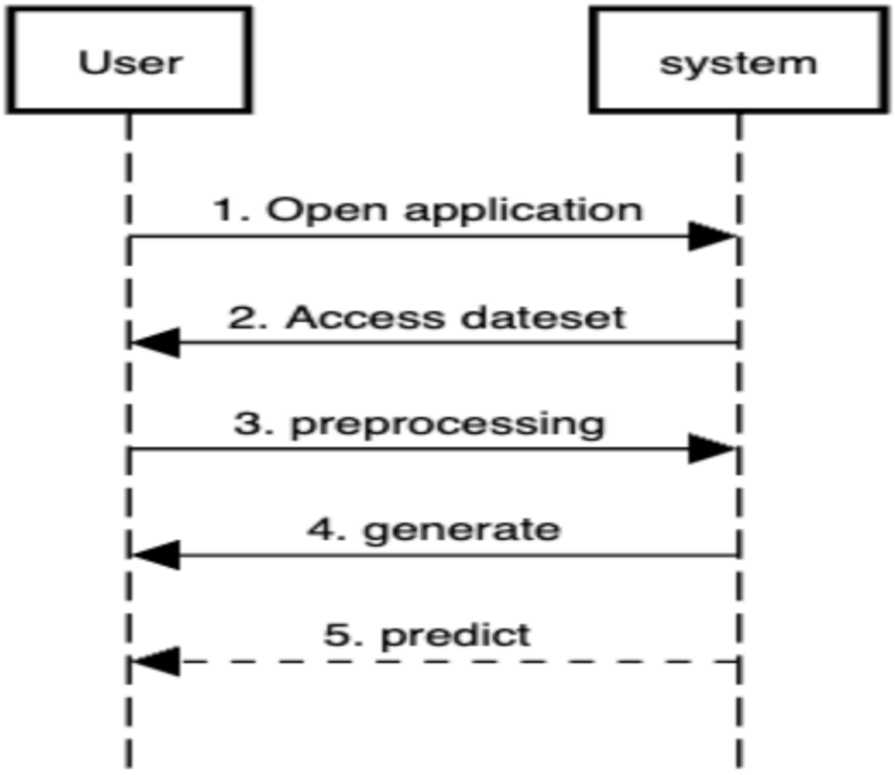

Figure 3.2 Sequence Diagram

CHAPTER 4

SYSTEM DESIGN

4.1 PROJECT DESCRIPTION

Phishing attacks present a substantial threat in cybersecurity as specifically target unsuspecting individuals, tricking them into divulging sensitive information. These attacks typically involve the creation of fraudulent websites or emails that closely resemble legitimate counterparts, making difficult for users to distinguish their authenticity. To combat this growing menace, and is crucial to develop robust and effective phishing detection mechanisms capable of accurately identifying and flagging potential phishing websites.

A machine learning algorithm based on Logistic Regression is proposed for utilization. URLs, which serve as website addresses, consist of various components such as the protocol, domain, subdomain, path, query parameters, and fragment identifier. By leveraging feature extraction techniques, To make possible to analyze these components and identify indicators of phishing. Examples of features may include the presence of IP addresses, symbols such as "@" and dashes, URL length, quantity of dots, and the presence of sub-domains within the web address. Additionally, factors such as website ranking, age, and authenticity can be taken into account to enhance the accuracy of the phishing detection system.

To train and evaluate the model, the dataset can be split into 80% for training and 20% for verification purposes. By utilizing the Logistic Regression algorithm, an impressive accuracy rate of 96% can be achieved, accompanied by a 95% recall and F1 score. This indicates that the model is effective in distinguishing between phishing and legitimate websites.

To make note that continuous monitoring and updating of the model are essential to ensure its effectiveness against evolving phishing techniques. Regular feedback collection from

users and the integration of new data into the training process will enable the system to adapt and improve its phishing detection capabilities over time.

4.2 MACHINE LEARNING-BASED ANALYSIS

Machine learning analysis is utilized for phishing detection, targeting fraudulent emails, websites, and digital communications. The dataset includes 549,346 entries with two columns: a label column indicating "Good" or "Bad" URLs (non-phishing and phishing), and no missing values. URLs are vectorized using CountVectorizer, extracting important words. Logistic regression is employed for classification, yielding a remarkable 98% accuracy rate in predicting phishing likelihood. This approach effectively safeguards users from phishing attacks.

4.1.1 Website Analysis

FastAPI is a modern, fast (high-performance), web framework for building APIs with Python 3.7+ based on standard Python type hints. VS Code is a popular code editor that provides a rich development environment for Python.The code imports the necessary modules (uvicorn, FastAPI, joblib), loads the machine learning model from the "phishing.pkl" file, and defines a route ("/predict/{features}") that expects a string parameter. When a GET request is made to this route with the specified URL path parameter, the predict function is executed.

4.1.2 Machine Learning Algorithm

Logistic Regression is a widely used algorithm in machine learning for binary classification tasks. Models the relationship between input features and a binary outcome by applying a logistic function. This function maps the input features to a probability score, which is then used to make a binary classification decision.

This algorithm finds applications in various domains, including spam detection, fraud detection, and medical diagnosis, among others. Its simplicity and interpretability make a popular choice for tasks where the goal is to predict one of two classes based on input features.

1. Binary Classification: Logistic Regression is primarily used for binary classification problems where the target variable has two classes. The can be extended to handle multi-class classification by using techniques like one-vs-rest or multinomial logistic regression.

2. Assumptions: Logistic Regression assumes that the relationship between the input features and the log-odds of the outcome is linear. It also assumes the absence of multicollinearity, independence of observations, and a sufficient number of observations.

3. Training and Inference: During training, logistic regression fits the model parameters by minimizing a loss function, typically the negative log-likelihood or cross-entropy loss. Once trained, the model can be used to predict the probability or class label for new instances.

4. Interpretability: Logistic Regression provides interpretability by estimating coefficients for each input feature. These coefficients indicate the direction and magnitude of the influence of each feature on the log-odds of the outcome.

5. Regularization: To prevent overfitting, logistic regression can be regularized using techniques like L1 or L2 regularization. Regularization helps control the complexity of the model and can improve generalization performance.

6. Performance Evaluation: Besides accuracy, Logistic Regression models can be evaluated using various metrics such as precision, recall, F1 score, and area under the ROC curve.

To evaluate and compare the performance of different models, they can record the scores in a dictionary. In this case, the logistic regression model achieved an accuracy of 98%, indicating a strong predictive capability for the given binary classification problem.

4.3 SYSTEM ARCHITECTURE

As mentioned in figure 4.3, In this architecture, the system consists of several components working together to detect phishing attempts:

1. User Interface: This component provides an interface for users to interact with the system, input URLs, and receive phishing detection results.

2. Phishing Detection Engine: This component is responsible for analyzing and determining whether a given URL is potentially a phishing attempt. It utilizes various techniques and algorithms to assess the characteristics of the URL and its associated content.

3. Fake ip address Metadata : This component preprocesses the input data to ensure it is in a suitable format for analysis. It may involve tasks such as data cleaning, normalization, and feature extraction.

4. Detection: This component employs machine learning algorithms to train and classify URLs as either legitimate or phishing. Techniques like supervised learning, feature engineering, and anomaly detection can be utilized to build an effective phishing detection model.

5. Reports : This component assesses the performance of the phishing detection system by using evaluation metrics such as accuracy, precision, recall, and F1-score. These metrics provide insights into the effectiveness and reliability of the system.

The overall architecture aims to provide a user-friendly interface for users to input URLs, while the backend components process and analyze the URLs using machine learning algorithms and evaluation metrics to determine whether they are phishing attempts or not.

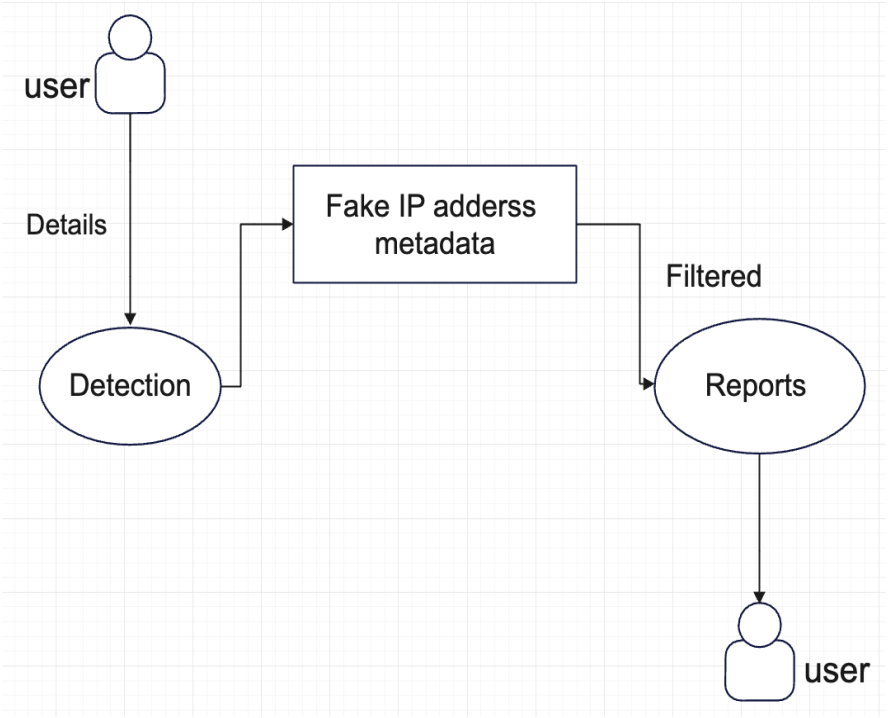

Figure 4.1 Architecture Diagram

4.4 MODULES DESCRIPTION

Here's a detailed description of each module:

1. ***User Interface Module:***Frameworks like Fast API for building the user interface.Json for designing and implementing the user interface components.
2. ***Phishing Detection Engine Module:***URL parsing and analysis libraries like urllib or urlparse for extracting relevant information from URLs.Feature extraction techniques to capture characteristics such as domain age, SSL certificate validity, and URL length.Regular expression libraries like re for pattern matching and detecting suspicious patterns in URLs.Phishing blacklists or reputation databases for comparing against known malicious URLs.Natural language processing libraries like NLTK or spaCy for analyzing textual content associated with URLs.
3. ***FastAPI Module:***FastAPI: A modern, fast, web framework for building APIs with Python. It provides easy-to-use decorators and functions for defining API routes, handling HTTP requests, and generating interactive API documentation.

CHAPTER 5

IMPLEMENTATION

5.1 INTRODUCTION

Phishing code in machine learning extensive data manipulation and analysis are conducted using powerful libraries such as Pandas and NumPy. These libraries enable efficient handling and processing of data, facilitating in-depth insights and exploration. The project's primary objective is classification, focusing on predicting whether a given input falls into the "good" or "bad" category. Evaluation metrics such as recall, precision, F1-score, and confusion matrix are utilized to assess the performance of the classification algorithm, providing a comprehensive understanding of its effectiveness.

To prepare the data for classification, various text preprocessing techniques are employed. This includes utilizing regular expression tokenizers to split words from the text and applying the Snowball Stemmer to obtain root words. These techniques aid in reducing the dimensionality of the data and capturing important features. Versatile visualization library, is utilized to create visually appealing and informative statistical graphics. These visualizations assist in the interpretation and visualization of the data, helping to uncover patterns and relationships. Additionally, word clouds are employed as a visualization technique to gain insights into the key aspects of the data, highlighting important words or patterns.

Automation tools such as the Chrome WebDriver are leveraged for web scraping and interacting with web applications. This allows the acquisition of data from online sources, expanding the scope and richness of the available data.

The chosen classification model for the project is Logistic Regression, known for its simplicity and interpretability. The model is integrated into a scikit-learn pipeline, along with the

CountVectorizer transformation, to streamline the data preprocessing and modeling process. By combining various tools, techniques, and models, the project aims to provide accurate predictions and valuable insights into the classification task at hand.

Overall, this project demonstrates a comprehensive approach to classification, encompassing data manipulation, preprocessing, visualization, and modeling. The integration of powerful libraries and automation tools enables efficient and effective analysis, allowing for accurate predictions and valuable insights.This is simple yet so effective. As I get an accuracy of 98%. That’s a very high value for a machine to be able to detect a malicious URL with. Want to test some links to see if the model gives good predictions.

5.2 IMPLEMENTATION DETAILS

Back-end development

The project involves extensive data manipulation and analysis using powerful libraries such as Pandas and NumPy. These libraries enable efficient handling and processing of data, allowing for in-depth insights and exploration. Matplotlib is utilized to create visually appealing and informative statistical graphics, aiding in the visualization and interpretation of the data.

The main objective of the project is classification, where the goal is to predict whether a given input falls into the "good" or "bad" category. Evaluation metrics such as recall, precision, F1-score, and confusion matrix are used to assess the performance of the classification algorithm, providing a comprehensive understanding of its effectiveness.

To prepare the data for classification, various text preprocessing techniques are employed. This includes using regular expression tokenizers to split words from the text, as well as employing the Snowball Stemmer to obtain root words. These techniques help in reducing the dimensionality of the data and capturing important features.

Visualization techniques, such as word clouds, are employed to gain insights into the key aspects of the data and highlight important words or patterns. Additionally, automation tools like the Chrome WebDriver are utilized for web scraping and interacting with web applications, enabling the acquisition of data from online sources.

The chosen classification model for the project is Logistic Regression, which is known for its simplicity and interpretability. The model is integrated into a scikit-learn pipeline, along with the CountVectorizer transformation, to streamline the data preprocessing and modeling process.

By combining various tools, techniques, and models, the project aims to provide accurate predictions and valuable insights into the classification task at hand.

5.3 ALGORITHM DESCRIPTION

The algorithm utilized in the project is Logistic Regression, a popular method for binary classification. Logistic Regression models the relationship between input features and a binary outcome using a logistic function. It is a supervised learning algorithm that predicts the probability of an input belonging to a certain class.

In the project, the Logistic Regression algorithm is used to predict whether a given input falls into the "good" or "bad" category. The algorithm takes the preprocessed input data, which has been transformed using techniques such as CountVectorizer and tokenization, as its input. The transformed features are then used to train the Logistic Regression model.

During the training process, the algorithm adjusts the model's parameters to minimize the difference between the predicted probabilities and the actual class labels. The optimization is typically performed using gradient descent or other optimization techniques.

Once the model is trained, it can be used to make predictions on new, unseen data. The algorithm takes the preprocessed input features and calculates the probability of the input

belonging to the "bad" class. If the probability exceeds a certain threshold, the input is classified as "bad"; otherwise, it is classified as "good".

The performance of the algorithm is evaluated using various metrics such as recall, precision, F1-score, and confusion matrix. These metrics provide insights into the algorithm's accuracy, precision, and ability to correctly classify the inputs.

Overall, the Logistic Regression algorithm is chosen for its simplicity, interpretability, and effectiveness in binary classification tasks. It plays a crucial role in the project by providing accurate predictions and aiding in the understanding and analysis of the data.

5.4 DEVELOPMENTAL TOOLS USED – DESCRIPTION

APIfast application to show the prediction

#Importing Modules
- `uvicorn`: The server that runs the FastAPI application.
- `FastAPI`: The main class used to create the API.
- `joblib`: Used to load the machine learning model from a .pkl file.
- `os`: Provides a way to interact with the operating system.

Loading the Machine Learning Model
- The code loads the machine learning model stored in the `phishing.pkl` file using `joblib.load()`.
- Make sure to provide the correct file path to your .pkl file.
- The loaded model is assigned to the `phish_model_ls` variable.

#Defining the Prediction Route
- The code defines a route `/predict/{features}` that expects a string parameter called `features`.

- When a GET request is made to this route with the specified URL path parameter, the `predict` function is executed.
- Inside the `predict` function, an empty list `X_predict` is created and the `features` parameter is appended to it.
- The loaded model `phish_model_ls` is used to predict the outcome using `phish_model_ls.predict(X_predict)`.
- Based on the prediction result, the code determines whether the site is classified as "bad" (phishing) or not and assigns the appropriate response message to the `result` variable.

Returning the Response
- Finally, the function returns a JSON response with the `result` message using `{"result": result}`.

#Running the Application
- The code block ensures that the FastAPI application is only run if the script is executed directly (not imported as a module).
- It starts the UNicorn server using `uvicorn.run()` and passes the `app` object.
- The server is set to listen on `http://127.0.0.1:8000`.
- The can make GET requests to `http://127.0.0.1:8000/predict/{features}`, replacing `{features}` with the desired URL to be classified as phishing or not.
- The response will include the result message based on the prediction.

Remember to adjust the file path to `phishing.pkl` file and test the application using appropriate URLs to see the prediction results.

CHAPTER 6

SYSTEM TESTING

6.1 TEST CASE TABLE

Table 6.1 Test Case Table

Module Name:	API fast application test
Test Title:	Enter the URLs randomly
Description	Test the predicted msg in above the webpage
Test Designed by	Meenalochne.S
Test Designed date	09.April.2023
Test Executed by	Meenalochne.S
Test Executed date	11.April.2023

6.2 TEST CASE RESULTS & ANALYSIS

Table 6.2 Test Case Result and Analysis

Step	Test Case	Test Data	Expected Result	Actual Result	Status (Pass/Fail)
1	(i) User Interface Module	URL : www.barclays.exe	Users should be able to detect the urls .	Users should be able to view whether the urls are phishing or not .	PASS
2	(ii) Phishing Detection Engine Module:	URL : www.youtube.com/			
3	(iii) FastAPI Module:	URL : www.hsbc-direct.com			
4	(vi) Feedback Module	Submit feedback about the application			

CHAPTER 7

RESULT AND DISCUSSION

7.1 PREDICTED OUTPUT

FastAPI 0.10 introduces support for OpenAPI 3 (OAS3), a specification for building and documenting RESTful APIs. With the inclusion of the "/openapi.json" endpoint, FastAPI allows developers to automatically generate comprehensive API documentation in a machine-readable format. This facilitates interoperability, simplifies integration with other tools, and enhances the overall developer experience.

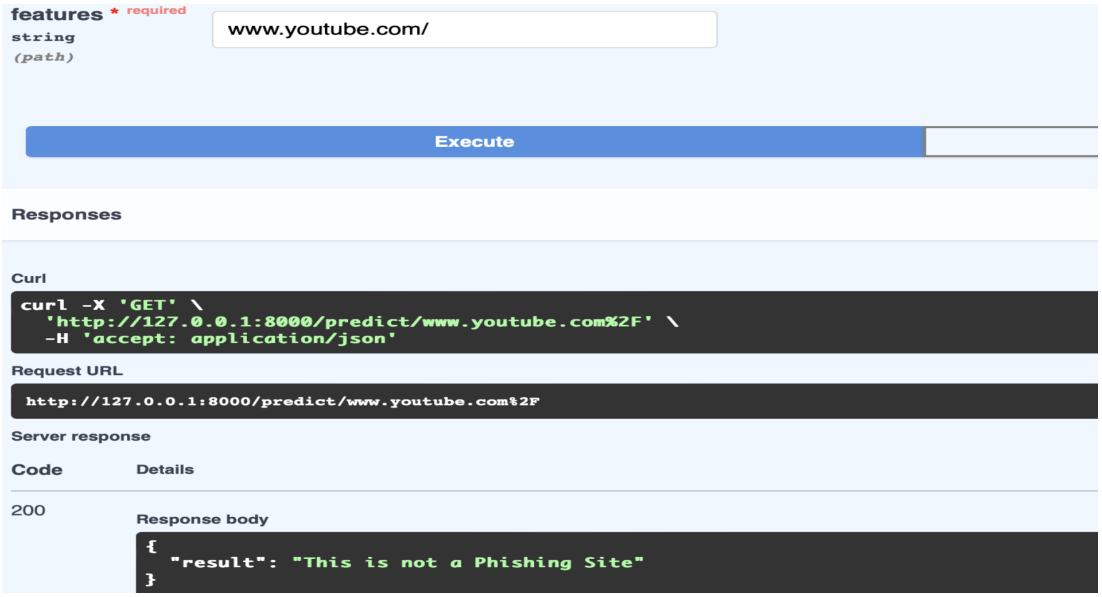

Figure 7.1 Legitimate Output

As mentioned in figure 7.1 To summarize Avoid clicking on the suspicious link or visiting the phishing site to prevent inadvertently disclosing personal information. Report the phishing site using the analysis tool's reporting option. This helps protect other users from falling victim to the scam.Notify the relevant organization about the fraudulent activity if the phishing attempt is related to a specific website or organization. They may have dedicated channels for reporting such incidents.Exercise caution when sharing personal information online, especially sensitive data like passwords, credit card details, or social security numbers. Only provide such information on trusted and secure websites.

* Good links => this are not phishing sites

1. www.youtube.com/
2. www.lloydstsbs.com
3. www.natwesti.com

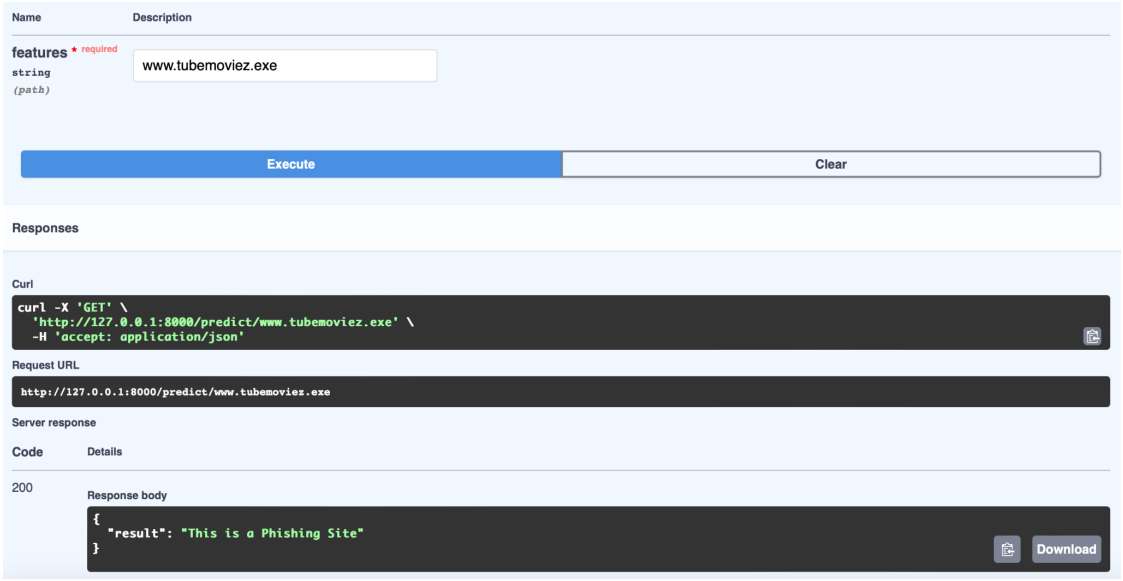

Figure 7.2 Phishing Output

As mentioned in figure 7.2 the sophistication of phishing attacks are on the rise, making it increasingly challenging for individuals and security systems to detect and prevent them.To address this issue, researchers have explored the use of machine learning techniques for phishing detection. Machine learning algorithms can analyze large amounts of data and learn to identify patterns and features that distinguish phishing emails and websites from legitimate ones.

* Bad links => this are phishing sites

1. www.barclays.exe
2. www.tubemoviez.exe
3. www.hsbc-direct.com

Overall, FastAPI 0.10's integration of OpenAPI 3 and the inclusion of the "/openapi.json" endpoint significantly enhance the developer experience by providing a standardized and machine-readable API documentation solution. This contributes to increased productivity, improved collaboration between frontend and backend teams, and seamless integration with other tools and frameworks in the API ecosystem.

7.2 DATASET

phishing_site_urls

URL	Label
nobell.it/70ffb52d079109dca5664cce6f317373782/login.SkyPe.com/en/cgi-bin/verification/login/70ffb52d079109dca5664cce6f317373/index.php?cmd=_profile-ach&outdated_page_tmpl=p/gen/failed-to-load&nav	bad
www.dghjdgf.com/paypal.co.uk/cycgi-bin/webscrcmd=_home-customer&nav=1/loading.php	bad
serviciosbys.com/paypal.cgi.bin.get-into.herf.secure.dispatch35463256rzr321654641dsf654321874/href/href/href/secure/center/update/limit/seccure/4d7a1ff5c55825a2e632a679c2fd5353/	bad
mail.printakid.com/www.online.americanexpress.com/index.html	bad
thewhiskeydregs.com/wp-content/themes/widescreen/includes/temp/promocoessmiles/?84784787824HDJNDJDSJSHD//2724782784/	bad
smilesvoegol.servebbs.org/voegol.php	bad
premierpaymentprocessing.com/includes/boleto-2via-07-2012.php	bad
myxxxcollection.com/v1/js/jih321/bpd.com.do/do/l.popular.php	bad
super1000.info/docs	bad
horizonsgallery.com/js/bin/ssl1/_id/www.paypal.com/fr/cgi-bin/webscr/cmd=_registration-run/login.php?cmd=_login-run&dispatch=1471c4bdb044ae2be9e2fc3ec514b88b1471c4bdb044ae2be9e2fc3ec514b	bad
phlebolog.com.ua/libraries/joomla/results.php	bad

Figure 7.3 Phishing Dataset

lets-go-away.co.uk/	good
lets-talk-computers.com/guests/diskeeper/dpan/	good
letsbookhotel.com/en/canada/landmark-hotels-near/circuit-gilles-villeneuve/hotels-accommodation/motels.aspx?rad=30	good
letsbookhotel.com/en/canada/montreal/hotel/marriott-chateau-champlain.aspx	good
letsbuyit.co.uk/books-music-films/books/subject-specific-literature/english-language-linguistics-books/	good
letsgo.com/2442-chihuahua_state-travel-guides-las_barrancas_del_cobre_copper_canyon-d	good
letsgoamerks.com/2010/06/01/three-awesome-words-benoit-groulx-resigns/	good
letsgoamerks.com/tag/benoit-groulx/	good
letsgopens.com/scripts/phpBB3/viewtopic.php?f=1&t=50037	good
letsgopens.com/scripts/phpBB3/viewtopic.php?f=6&t=46698	good
letsmove.gov/	good
letterboxing.org/BoxByRegion.asp?region=Central+Ohio	good
letterboxing.org/BoxByRegion.php?region=Central%20Ohio&order=new	good

Figure 7.4 Legitimate Dataset

Datasets from "phishing_site_urls" dataset comprises a substantial collection of 549,346 distinct entries, specifically curated to address the pervasive issue of phishing attacks. Phishing, recognized as a pernicious cybercrime, employs deceptive tactics such as fraudulent emails, websites, and various digital communication methods to exploit unsuspecting victims and extract sensitive information for malicious purposes.The dataset's structure includes two fundamental columns, which play a pivotal role in predicting and classifying the nature of URLs.

As mentioned in figure 7.3 The label column serves as a vital indicator, segregating entries into two distinct categories: "Good" and "Bad." The "Good" classification signifies URLs devoid of any malicious content, ensuring that the associated websites are legitimate and free from phishing attempts. Conversely, the "Bad" classification identifies URLs that pose a considerable threat, harboring dangerous content and confirming the presence of phishing activities within the corresponding websites.

A significant advantage of the "phishing_site_urls" dataset lies in its integrity . As mentioned in figure 7.4 it exhibits a complete absence of missing values. This attribute ensures the dataset's reliability and enables comprehensive analyses and modeling without data imputation or potential biases caused by missing entries.To further enhance the dataset's utility, various machine learning techniques can be employed. For instance, preprocessing techniques such as vectorization can be applied to the URLs, utilizing methods like CountVectorizer. By

transforming the URLs into a numerical representation, essential features and patterns can be extracted and utilized to improve the accuracy of phishing detection models.

One popular machine learning algorithm suitable for this classification task is logistic regression. This algorithm specializes in predicting the probability of binary outcomes, making it an ideal choice for distinguishing between "Good" and "Bad" URLs. By training the logistic regression model on the "phishing_site_urls" dataset, Estimate the likelihood of a URL being associated with phishing activities, providing valuable insights and robust predictions.Given its comprehensive nature and the absence of missing values, the "phishing_site_urls" dataset serves as a valuable resource for researchers and practitioners in the field of cybersecurity. Leveraging the dataset's rich information, researchers can develop sophisticated machine learning models that accurately detect and mitigate phishing attacks, bolstering user protection and fortifying defenses against these cyber threats.

7.3 PHISHING DETECTION

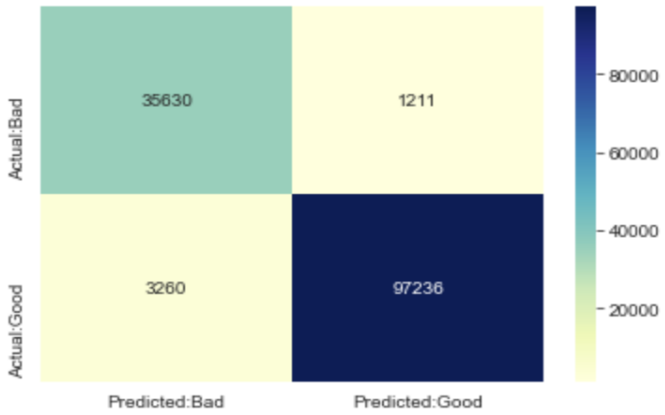

Figure 7.5 : Accuracy Percentage

As mentioned in figure 7.5 the dataset contains 549,346 distinct entries and two columns. The label column is a prediction column with two categories:

1. Good - indicating that the URL does not contain malicious content and that the website is not a phishing site.
2. Bad - indicating that the URL does contain dangerous content and that the website is a phishing site.

The dataset has no missing values.

After obtaining the data, the URLs are vectorized using CountVectorizer and important words are gathered using a tokenizer. This is because some words in URLs, such as "virus," ".exe," and ".dat," are more significant than others. The URLs are converted into a vector form.

Logistic regression, a machine learning classification algorithm, is used to predict the probability of a categorical dependent variable. The dependent variable in logistic regression is a binary variable with data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). In other words, the logistic regression model predicts P(Y=1) as a function of X.

To determine which model performs best, the will record the scores in a dictionary. The logistic regression model yields an accuracy of 98%.

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENTS

8.1 CONCLUSION

This project involves extensive data manipulation and analysis using powerful libraries such as Pandas and NumPy. The goal is to perform binary classification and predict whether a given input falls into the "good" or "bad" category. The Logistic Regression algorithm is chosen for its simplicity, interpretability, and effectiveness in binary classification tasks.The data is prepared for classification by employing various text preprocessing techniques, such as tokenization and stemming, to reduce dimensionality and capture important features. Visualization techniques, including word clouds, are used to gain insights into key aspects of the data.

The Logistic Regression model is trained using the preprocessed data, and its performance is evaluated using metrics such as recall, precision, F1-score, and confusion matrix. These metrics provide a comprehensive understanding of the model's effectiveness in accurately classifying inputs.To showcase the prediction, an APIfast application is implemented using FastAPI. The application loads the trained model and provides a route to make predictions on URL inputs. The application responds with a message indicating whether the provided URL is classified as phishing or not.Through the combination of various tools, techniques, and models, this project aims to provide accurate predictions and valuable insights in binary classification tasks, contributing to a better understanding of the data and its patterns.

8.2 DEVELOPMENT SUMMARY

The development summary outlines the key steps and components involved in setting up a FastAPI application for phishing website classification:

1. Module Import: The necessary modules are imported, including `unicorn` for running the server, `FastAPI` for creating the API, `joblib` for loading the machine learning model, and `os` for interacting with the operating system.

2. Loading the Machine Learning Model: The code loads the pre-trained machine learning model stored in the `phishing.pkl` file using the `joblib.load()` function. The model is assigned to the `phish_model_ls` variable.

3. Defining the Prediction Route: The code defines a route `/predict/{features}` that expects a string parameter called `features`. When a GET request is made to this route, the `predict` function is executed.

4. Prediction Function: Inside the `predict` function, an empty list `X_predict` is created, and the `features` parameter (URL to be classified) is appended to it. The loaded model `phish_model_ls` is then used to predict the outcome using `phish_model_ls.predict(X_predict)`.

5. Returning the Response: Based on the prediction result, the code determines whether the site is classified as "bad" (phishing) or not and assigns the appropriate response message to the `result` variable. The function returns a JSON response with the `result` message using `{"result": result}`.

6. Running the Application: The code block ensures that the FastAPI application is only run if the script is executed directly. It starts the UNicorn server using `uvicorn.run()` and passes the `app` object. The server listens on `http://127.0.0.1:8000`, and GET requests can be made to `http://127.0.0.1:8000/predict/{features}` to classify URLs as phishing or not. The response includes the result message based on the prediction.

8.3 FUTURE ENHANCEMENT

1. Enhancing Dataset Diversity: To further improve the performance of logistic regression models for phishing detection, it is important to consider a more diverse and

comprehensive dataset. Including different types of phishing attacks, languages, and demographics can help create models that are more robust and accurate in real-world scenarios.

2. Feature Engineering and Selection: Exploring advanced feature engineering techniques and selecting the most informative features can enhance the predictive power of logistic regression models. Techniques such as feature extraction, dimensionality reduction, and incorporating domain-specific knowledge can lead to more effective phishing detection.

3. Integration with Advanced Technologies: Integrating logistic regression models with advanced technologies such as deep learning and natural language processing can open new avenues for improving phishing detection. These techniques can enable models to capture more intricate patterns and nuances in phishing websites, leading to enhanced accuracy and performance.

4. Real-time Monitoring and Alerting: Developing systems that can continuously monitor and analyze website behavior in real-time can provide immediate alerts and warnings when potential phishing activities are detected. This can help users take preventive actions promptly, minimizing the risk of falling victim to phishing attacks.

5. User Education and Awareness: While technological advancements are crucial, user education and awareness play a vital role in combating phishing attacks. Future research can focus on developing effective strategies for educating users about phishing risks, safe online practices, and how to identify and report suspicious activities.

APPENDIX 1

SAMPLE CODE

CODING

PHISHING SITE URLs.ipymb

#Importing some useful libraries

```
import pandas as pd
import numpy as np
import seaborn as sns

import matplotlib.pyplot as plt
%matplotlib inline
import time
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from nltk.tokenize import RegexpTokenizer
from nltk.stem.snowball import SnowballStemmer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.pipeline import make_pipeline # use for combining all prerocessors techniuqes and
algos
from PIL import Image
```

```
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
from bs4 import BeautifulSoup
from selenium import webdriver
import networkx as nx import pickle
import warnings # ignores pink warnings
warnings.filterwarnings('ignore')

#Loading the main dataset

phish_data = pd.read_csv('phishing_site_urls.csv')
phish_data.head()
phish_data.info()
phish_data.isnull().sum()

#create a dataframe of classes counts

label_counts = pd.DataFrame(phish_data.Label.value_counts())
visualizing target_col
sns.set_style('darkgrid')
sns.barplot(label_counts.index,label_counts.Label)

#RegexpTokenizer

tokenizer = RegexpTokenizer(r'[A-Za-z]+')
phish_data.URL[0]
tokenizer.tokenize(phish_data.URL[0])
print('Getting words tokenized ...')
t0= time.perf_counter()
phish_data['text_tokenized'] = phish_data.URL.map(lambda t: tokenizer.tokenize(t)) # doing
with all rows
t1 = time.perf_counter() - t0
```

```
print('Time taken',t1 ,'sec')
phish_data.sample(5)

tokenizer = RegexpTokenizer(r'[A-Za-z]+')
```

#SnowballStemmer

```
stemmer = SnowballStemmer("english") # choose a language

print('Getting words stemmed ...')
t0= time.perf_counter()
phish_data['text_stemmed']  =  phish_data['text_tokenized'].map(lambda  l:  [stemmer.stem(word)
for word in l])
t1= time.perf_counter() - t0
print('Time taken',t1 ,'sec')

phish_data.sample(5)

print('Getting joiningwords ...')
t0= time.perf_counter()
phish_data['text_sent'] = phish_data['text_stemmed'].map(lambda l: ' '.join(l))
t1= time.perf_counter() - t0
print('Time taken',t1 ,'sec')

phish_data.sample(5)
```

#Visualization

#sliceing classes

```
bad_sites = phish_data[phish_data.Label == 'bad']
```

```
good_sites = phish_data[phish_data.Label == 'good']
bad_sites.head()
good_sites.head()
def plot_wordcloud(text, mask=None, max_words=400, max_font_size=120, figure_size=(24.0,16.0),
                   title = None, title_size=40, image_color=False):
    stopwords = set(STOPWORDS)
    more_stopwords = {'com','http'}
    stopwords = stopwords.union(more_stopwords)
    wordcloud = WordCloud(background_color='white',
                    stopwords = stopwords,
                    max_words = max_words,
                    max_font_size = max_font_size,
                    random_state = 42,
                    mask = mask)

    wordcloud.generate(text)

    plt.figure(figsize=figure_size)
    if image_color:
        image_colors = ImageColorGenerator(mask);
        plt.imshow(wordcloud.recolor(color_func=image_colors), interpolation="bilinear");
        plt.title(title, fontdict={'size': title_size,
                                   'verticalalignment': 'bottom'})
    else:
        plt.imshow(wordcloud);
        plt.title(title, fontdict={'size': title_size, 'color': 'green',
                                   'verticalalignment': 'bottom'})
    plt.axis('off');
    plt.tight_layout()
data = good_sites.text_sent
```

```
data.reset_index(drop=True, inplace=True)
common_text = str(data)
common_mask = np.array(Image.open('star.png'))
plot_wordcloud(common_text, common_mask, max_words=400, max_font_size=120,
        title = 'Most common words use in good urls', title_size=15)
data = bad_sites.text_sent
data.reset_index(drop=True, inplace=True)
common_text = str(data)
common_mask = np.array(Image.open('comment.png'))
plot_wordcloud(common_text, common_mask, max_words=400, max_font_size=120,
        title = 'Most common words use in bad urls', title_size=15)
```

#Chrome webdriver

```
browser = webdriver.Chrome(r"chromedriver.exe")

list_urls = ['https://www.ezeephones.com/','https://www.ezeephones.com/about-us'] #here to take phishing sites

links_with_text = []
```

#BeautifulSoup

```
for url in list_urls:
   browser.get(url)
   soup = BeautifulSoup(browser.page_source,"html.parser")
   for line in soup.find_all('a'):
      href = line.get('href')
      links_with_text.append([url, href])
```

#Turn the URL's into a Dataframe

```
df = pd.DataFrame(links_with_text, columns=["from", "to"])

df.head()

#Draw a graph

GA = nx.from_pandas_edgelist(df, source="from", target="to")
nx.draw(GA, with_labels=False)

#Creating Model

#CountVectorizer

#create cv object

cv = CountVectorizer()
feature = cv.fit_transform(phish_data.text_sent)
feature[:5].toarray()

trainX, testX, trainY, testY = train_test_split(feature, phish_data.Label)

### LogisticRegression

# create lr object

lr = LogisticRegression()
lr.fit(trainX,trainY)
lr.score(testX,testY)

Scores_ml = {}
```

```
Scores_ml['Logistic Regression'] = np.round(lr.score(testX,testY),2)
print('Training Accuracy :',lr.score(trainX,trainY))
print('Testing Accuracy :',lr.score(testX,testY))

con_mat = pd.DataFrame(confusion_matrix(lr.predict(testX), testY),
        columns = ['Predicted:Bad', 'Predicted:Good'],
        index = ['Actual:Bad', 'Actual:Good'])
print('\nCLASSIFICATION REPORT\n')

print(classification_report(lr.predict(testX), testY,
                    target_names =['Bad','Good']))
print('\nCONFUSION MATRIX')
plt.figure(figsize= (6,4))
sns.heatmap(con_mat, annot = True,fmt='d',cmap="YlGnBu")

acc = pd.DataFrame.from_dict(Scores_ml,orient = 'index',columns=['Accuracy'])

sns.set_style('darkgrid')

sns.barplot(acc.index,acc.Accuracy)
```

Logistic Regression is the best fit model, Now to make sklearn pipeline using Logistic Regression

```
pipeline_l=make_pipeline(CountVectorizer(tokenizer=
RegexpTokenizer(r'[A-Za-z]+').tokenize,stop_words='english'), LogisticRegression())
##(r'\b(?:http|ftp)s?://\S*\w|\w+|[^\w\s]+') ([a-zA-Z]+)([0-9]+) -- these tolenizers giving me low accuray

trainX, testX, trainY, testY = train_test_split(phish_data.URL, phish_data.Label)
```

```
pipeline_ls.fit(trainX,trainY)
pipeline_ls.score(testX,testY)

print('Training Accuracy :',pipeline_ls.score(trainX,trainY))
print('Testing Accuracy :',pipeline_ls.score(testX,testY))

con_mat = pd.DataFrame(confusion_matrix(pipeline_ls.predict(testX), testY),
        columns = ['Predicted:Bad', 'Predicted:Good'],
        index = ['Actual:Bad', 'Actual:Good'])

print('\nCLASSIFICATION REPORT\n')
print(classification_report(pipeline_ls.predict(testX), testY,
                    target_names =['Bad','Good']))
print('\nCONFUSION MATRIX')
plt.figure(figsize= (6,4))

sns.heatmap(con_mat, annot = True,fmt='d',cmap="YlGnBu")

pickle.dump(pipeline_ls,open('phishing.pkl','wb'))

loaded_model = pickle.load(open('phishing.pkl', 'rb'))

result = loaded_model.score(testX,testY)
print(result)
```

This is simple yet so effective and get an accuracy of 98%. That's a very high value for a machine to be able to detect a malicious URL with. Want to test some links to see if the model gives good predictions.

CODING (FRONT-END)

Prediction_app.py

```
import uvicorn
from fastapi import FastAPI
import joblib
import os
app = FastAPI()
#pkl
phish_model = open('/Users/meenalochnegmail.com/Desktop/final_model/phishing.pkl', 'rb')
phish_model_ls = joblib.load(phish_model)
# ML Aspect
@app.get('/predict/{features}')
async def predict(features: str):
    X_predict = []
    X_predict.append(features)
    y_Predict = phish_model_ls.predict(X_predict)
    if y_Predict == 'bad':
        result = "This is a Phishing Site"
    else:
        result = "This is not a Phishing Site"
    return {"result": result}
if __name__ == '__main__':
    uvicorn.run(app, host="127.0.0.1", port=8000)
```

APPENDIX 2
PAPER PRESENTATION

This submission presents a research paper titled "Phishing Website Detection using Machine Learning" from Nitte Meenakshi Institute of Technology, Bengaluru. The paper is submitted for the TRACK 1: NETWORKS category. The institute established an IEEE Student Branch in the year 2005 under the IEEE Bangalore Section.

Submission Summary

Hello,

The following submission has been created.

Track Name: TRACK 1: NETWORKS

Paper ID: 988

Paper Title: Phishing Website Detection using Machine Learning

Abstract:
Phishing attacks pose a significant threat to cybersecurity, and machine learning algorithms offer a promising approach to detect and prevent such attacks. This paper compares the effectiveness of Logistic Regression, Support Vector Machine, and Random Forest algorithms in detecting phishing websites. The study aims to identify the most accurate algorithm with minimal errors. Extensive data manipulation and analysis are conducted using libraries like Pandas and NumPy, while Matplotlib facilitates data visualization. Evaluation metrics such as recall, precision, F1-score, and confusion matrix provide a comprehensive assessment of the classification algorithms. Text preprocessing techniques, including tokenization and stemming, are employed to prepare the data. Visualization techniques, such as word clouds, help uncover key patterns. The chosen model, Logistic Regression, is integrated into a scikit-learn pipeline

with CountVectorizer for efficient data preprocessing and modeling. The project seeks to deliver accurate predictions and valuable insights to enhance phishing detection. Additionally, the FastAPI framework and modules like uvicorn and joblib are utilized for serving the trained model and facilitating prediction requests.

Keywords : machine learning , phishing urls .

Created on: Thu, 18 May 2023 09:10:57 GMT

Last Modified: Thu, 18 May 2023 09:10:57 GMT

Authors:

- meenalochne@gmail.com (Primary)

Secondary Subject Areas: Not Entered

Submission Files: Phishing Website Detection using Machine Learning (1).docx.pdf (879 Kb, Thu, 18 May 2023 09:10:05 GMT)

Submission Questions Response: Not Entered

REFERENCES

1. Aniket Garje1 , Namrata Tanwani1 , Sammed Kandale1 , Twinkle Zope1 , Prof. Sandeep Gore2 (2021)"Detecting Phishing Websites Using Machine Learning"International Journal of Creative Research Thoughts (IJCRT)

2. Arathi Krishna V, Anusree A, Blessy Jose, Karthika Anilkumar, Ojus Thomas Lee "Machine learning to detect phishing using URL analysis A survey"International Journal of Scientific Research and Engineering Development

3. Dipayan Sinha, Dr. Minal Moharir, Prof. Anitha Sandeep,"Phishing Website URL Detection using Machine Learning,"International Journal of Advanced Science and (2020),International Conference on Big Data Analytics and Computational Intelligence (ICBDAC)

4. Sahingoz, O. K., Buber, E., Demir, O., & Diri, B. "MachineLearning-Based Phishing Detection from URLs," ExpertSystems with Applications, (2019) "International Conference on Computing Methodologies and Communication (ICCMC).

5. Salvi Siddhi Ravindra1, Shah Juhi Sanjay1, Shaikh Nausheenbanu Ahmed Gulzar1, Khodke Pallavi2 "Phishing Website Detection Based on URL"International Research Journal of Engineering & Technology (IRJET).

6. S. Carolin Jeeva1* and Elijah Blessing Rajsingh"Intelligent phishing url detection using association rule mining"IEEE Systems Journal, Volume 11, Issue 2 .

7. Taizhen Wang et al"Phishing Detection Based on Machine Learning Algorithms: A Systematic Literature Review". (2021)International Journal of Innovative Research in Science, Engineering & Technology (IJIRSET), Volume 7, Issue 6

8. Yiming Liu et al."Phishing Detection Using Support Vector Machine and Logistic Regression" (2020),IEEE Systems Journal, Volume 19.