

Project Report
on
DETECTION OF PHISHING WEBSITES USING
MACHINE LEARNING

(A Project Report submitted in partial fulfillment of the requirements of
Bachelor of Technology in Information Technology of the West Bengal
University of Technology, West Bengal)

Submitted by

Arnab Kumar Das
Shubhankar Das
Sayan Banerjee
Sarthak Dey

Under the guidance of
Professor Malabika Sengupta
Dept. of Information Technology



Kalyani Government Engineering College
(Affiliated to West Bengal University of Technology)
Kalyani - 741235, Nadia, WB

2023-2024

Phone: 25826680 (PBX)
Fax : 2582130

e-mail :

কল্যাণী - ৭৪১ ২৩৫
নদীয়া, পশ্চিমবঙ্গ



Kalyani 741 235
Nadia, West Bengal, India

পত্রাঙ্ক / Ref. No.:

তারিখ / Date :

কল্যাণী গভঃ ইঞ্জিনিয়ারিং কলেজ
Kalyani Government Engineering College
(Govt. of West Bengal)

Certificate of Approval

This is to certify thathas done final year project work entitled.....under my direct supervision and he/she has fulfilled all the requirements of relating to the Final Year Project. It is also certified that this project work being submitted, fulfills the norms of academic standard for B. Tech Degree in Information Technology of The West Bengal University of Technology and it has not been submitted for any degree whatsoever by him/her or anyone else previously.

.....
Head
Department of Information Technology
Kalyani Government Engineering College

.....
Supervisor
Department of Information Technology
Kalyani Government Engineering College

.....
Project Coordinator
Department of Information Technology
Kalyani Government Engineering College

.....
Examiner

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Dr. Malabika Sengupta for her invaluable guidance and unwavering support throughout the course of this project. Her insightful feedback, constructive criticism, and dedication to fostering a spirit of inquiry have been pivotal in shaping and refining our research on "Detecting Phishing Websites using Machine Learning." Her mentorship has not only enriched my understanding of the subject matter but has also inspired a deeper passion for research and innovation.

I would also like to extend my appreciation to the authors of the research papers that served as significant references for our project. The contributions of these researchers have laid a solid foundation for our work, and their insights have been instrumental in shaping our methodology. The following research papers have been particularly influential:

- Görkem Giray, Bedir Tekinerdogan, Sandeep Kumar & Suyash Shukla, "Applications of deep learning for phishing detection: a systematic literature review" in Knowledge and Information Systems [23rd May 2022]

- Ashit Kumar Dutta, "Detecting phishing websites using machine learning technique" in Zhihan Lv, Qingdao University, China October 11, 2021.

Other valuable papers and articles have been enlisted in the References section of the report.

I am thankful for the support and encouragement received from Dr. Sengupta and the authors of the referenced research papers, as their work has played a crucial role in the development and success of our project.

[Date]

1. Arnab Kumar Das - 10200220008-
2. Shubhankar Das - 10200220006 -
3. Sayan Banerjee - 10200220041 -
4. Sarthak Dey - 10200220015 -

Department of Information Technology
Kalyani Government Engineering College

ABSTRACT

Phishing attacks leverage deceptive websites to steal sensitive information, posing a significant cybersecurity threat. This project develops an advanced phishing website detector using machine learning models and deep neural networks.

The dataset consists of 549,346 URLs sourced from Kaggle, with 50+ features extracted, such as self-redirecting hyperlinks, right-click disable, and the number of pop-ups. This comprehensive feature set significantly improves detection capabilities.

Various models, including Decision Trees, Random Forests, Logistic Regression, XGBoost, and Autoencoder Neural Networks, are trained and evaluated. Our XGBoost model achieved an accuracy of 93.85%, outperforming previous models' accuracy of 80-90%.

Additionally, this work includes a detailed correlation analysis among the features, providing deeper insights into their relationships. Future work involves developing a GUI or browser extension for real-time URL classification, enhancing cybersecurity by preventing phishing attacks.

CONTENTS

CHAPTER 1 INTRODUCTION 1

- 1.1 **Objective 2**
- 1.2 **Organization of the Report 2-3**

CHAPTER 2 BACKGROUND STUDY 4

- 2.1 **Literary Survey 5-6**
- 2.2 **Modes of Phishing 6**
- 2.3 **Problem Statement 7**
- 2.4 **Motivation 7**

CHAPTER 3 TECHNICAL APPROACH 8

- 3.1 **Algorithms (ML and DL) 8**
 - 3.1.1 Random Forest 8-9
 - 3.1.2 Logistic Regression 10
 - 3.1.3 XG Boost 10-11
 - 3.1.4 SVM (Support Vector Machine) 11-12
 - 3.1.5 Neural Networks 13-14
- 3.2 **Feature Selection 14**
 - 3.2.1 URL Analysis 14-15
 - 3.2.2 Content Analysis 15-16
 - 3.2.3 Behavioral Analysis 17-18

CHAPTER 4 PROPOSED DETECTION MODEL 19

- 4.1 **Implementation 19**
 - 4.1.1 Data Collection 19-20
 - 4.1.2 Feature Extraction 21
 - 4.1.3 Data Preprocessing 21-23
 - 4.1.4 Model Training and Testing 23
 - 4.1.5 Model Validation 24
 - 4.1.6 Deployment 25-26
 - 4.1.7 Monitoring and Updates 26-27

- 4.2 **Features** 27-32
- 4.3 **Tech Stacks** 32-33
- 4.4 **Frontend Development** 33
 - 4.4.1 HTML/CSS 34
 - 4.4.2 JavaScript and JQuery 34
 - 4.4.3 User Interface Design 35
- 4.5 **Backend Development** 36
 - 4.5.1 Node.js and Express.js 36
 - 4.5.2 Database Management with MongoDB 37
- 4.6 **API Development** 37
 - 4.6.1 RESTful API Design 38
 - 4.6.2 API Integration 38-39

CHAPTER 5 EXPERIMENTAL RESULTS 40

- 5.1 **Dataset Description** 40
- 5.2 **Evaluation Metrics** 41-42
 - 5.2.1 Accuracy 42
 - 5.2.2 Precision 43
 - 5.2.3 Recall 43
 - 5.2.4 F1 Score 43-44
 - 5.2.5 Confusion Matrix 44
- 5.3 **Model Performance** 44
 - 5.3.1 Training Results 45
 - 5.3.2 Testing Results 45
 - 5.3.3 Cross-Validation Results 45
- 5.4 **Comparative Analysis** 46
- 5.5 **Limitations of the Current Model** 47
- 5.6 **Potential Improvements** 47-48

CHAPTER 6 CONCLUSION 49

- 6.1 **Summary of Present Work** 49
- 6.2 **Future Work** 50

CHAPTER 7 REFERENCES 51

Chapter 1

INTRODUCTION

In today's digital landscape, cybersecurity is more crucial than ever, with phishing attacks posing a significant threat to personal and organizational data. Phishing involves deceptive attempts to obtain sensitive information such as usernames, passwords, and financial details by masquerading as a trustworthy entity. Our project addresses this pressing challenge by employing advanced machine learning techniques to detect and classify phishing websites with high accuracy.

Traditional security measures often rely on static rules and signature-based detection methods, which are inadequate against the dynamic and constantly evolving nature of phishing tactics. These outdated methods struggle to keep pace with the sophisticated techniques employed by cybercriminals, making the need for more adaptive and intelligent solutions imperative.

The primary objective of our project is to develop a robust machine learning model capable of identifying patterns in the structure, content, and behavior of phishing websites. Unlike static rules, our machine learning approach offers adaptability, learning from a diverse range of datasets to effectively detect new and evolving phishing threats in real-time. By leveraging features such as URL analysis, HTML content, and user interaction behavior, our model can discern between legitimate and malicious websites with high precision.

The project's development includes creating a comprehensive detection system using a combination of Logistic Regression, Decision Trees, Random Forest, XGBoost, and Autoencoder Neural Networks. This system will not only detect phishing attempts but also classify them accurately and swiftly.

Additionally, we aim to establish a structured database to systematically store information on identified phishing websites. This database will facilitate collaboration with organizations like the Anti-Phishing Working Group (APWG), enhancing global cybersecurity efforts. By integrating user feedback mechanisms, we ensure continuous refinement and improvement of our detection algorithms.

This report outlines our comprehensive research methodology, including dataset selection, feature extraction techniques, model training and evaluation, and deployment strategies. By leveraging insights from contemporary research and practical implementation, we strive to contribute significantly to the cybersecurity field, providing a pragmatic solution to the persistent and evolving threat of phishing attacks. Our ultimate goal is to create a robust, real-time phishing detection tool that not only safeguards users but also fosters collaboration and knowledge sharing within the global cybersecurity community.

1.2 OBJECTIVE

Our project aims to implement a comprehensive machine learning-based phishing detection system using Python, integrating Logistic Regression, Decision Trees, Random Forest, XGBoost, and Autoencoder Neural Networks models via APIs. This system will enable real-time detection and classification of phishing websites, leveraging advanced algorithms to enhance accuracy and adaptability to evolving threats. Additionally, we plan to establish a database to systematically store information on identified phishing websites, collaborate with organizations like the Anti-Phishing Working Group (APWG) for wider cybersecurity collaboration, and incorporate user feedback mechanisms for continuous algorithm refinement. Ultimately, our goal is to create a robust tool that not only detects phishing websites effectively but also contributes to the global cybersecurity community by improving detection capabilities and fostering collaboration among cybersecurity stakeholders.

1.3 ORGANIZATION OF THE PROJECT

This report is structured to guide the reader through the various stages and aspects of our phishing detection project, ensuring a comprehensive understanding from the initial problem identification to the final conclusions and future directions. Here's a brief overview of the organization:

Chapter 1: Introduction

The introductory chapter sets the stage for the project by outlining its objectives and explaining the structure of the report. It introduces the primary goals of developing an advanced phishing detection model and provides a roadmap for the reader, detailing what each chapter covers.

Chapter 2: Background Study

Chapter 2 provides essential background information. It starts with a literary survey that reviews existing research and methodologies in phishing detection. It then explores the different modes of phishing, defines the specific challenges the project aims to address in the problem statement, and explains the motivation behind the need for improved phishing detection methods.

Chapter 3: Technical Approach

This chapter delves into the technical methods used in the project. It explains the machine learning and deep learning algorithms employed, including Random Forest, Logistic Regression, XGBoost, SVM, and Neural Networks. Additionally, it discusses feature selection techniques, such as URL Analysis, Content Analysis, and Behavioural Analysis, which are crucial for building an effective detection model.

Chapter 4: Proposed Detection Model

Chapter 4 discusses the implementation of the proposed phishing detection model. It covers all stages of development, including data collection, feature extraction, data pre-processing, model training and testing, model validation, deployment, and ongoing monitoring and updates. It also explores the features used in the model, the technological stacks employed, and the details of both frontend and backend development, including API development and integration.

Chapter 5: Experimental Results

This chapter presents the experimental results of the project. It provides a detailed description of the dataset used, the evaluation metrics (such as accuracy, precision, recall, F1 score, and confusion matrix), and the model's performance. It includes analyses of training results, testing results, and cross-validation outcomes, as well as a comparative analysis with existing models. The chapter also discusses the limitations of the current model and suggests potential improvements.

Chapter 6: Conclusion

The concluding chapter summarizes the key findings and contributions of the project. It reiterates the importance of advanced phishing detection methods in enhancing cybersecurity and outlines potential directions for future research and development.

Chapter 7: References

The final chapter lists all references and research papers consulted during the project, providing a comprehensive bibliography for further reading.

This organized structure ensures that the report thoroughly documents the project's development process while offering clear insights into the methods used to address the challenge of phishing detection.

Chapter 2

BACKGROUND STUDY

The literature survey conducted for this project provides a comprehensive understanding of the current state-of-the-art techniques, methodologies, and advancements in the domain of phishing detection using machine learning.

The survey encompasses a wide range of research papers, articles, and journals that contribute valuable insights into the intricacies of phishing attacks and the application of machine learning algorithms for their detection. The references to the papers and articles are attached in the References section.

The survey delves into feature extraction methodologies relevant to phishing detection. It investigates how researchers identify discriminative features from web content, structure, and behavior. Commonly used features include URL structures, HTML content analysis, and behavioral patterns, which contribute to the effectiveness of machine learning models.

The survey critically examines the challenges and limitations faced by existing phishing detection systems. It sheds light on areas such as false positives, adversarial attacks, and the need for continuous adaptation to new phishing tactics.

After thoroughly researching the published journals and articles, we found that the past works in this field mainly focused on machine learning algorithms such as Random Forest algorithm, Support Vector Machine, etc. but didn't include Deep Learning to detect the Phishing Websites. We will delve into this aspect and include deep learning in our model to increase its efficiency and accuracy.

SL No	Author and Year	Alm	Main findings	Limitations
1	Title: Detecting phishing websites using machine learning technique Author: Ashit Kumar Dutta, 2021	The proposed framework employs RNN- LSTM to identify the properties Pm and PI in an order to declare an URL as malicious or legitimate	The outcome of this study reveals that the proposed method presents superior results rather than the existing deep learning methods	The future direction of this study is to develop an unsupervised deep learning method to generate insight from a URL
2	Title: A systematic literature review on phishing website detection techniques Authors: Qabajeh et al., 2018	This review paper compares traditional anti-phishing methods, which includes raising awareness, educating users, conducting periodic training or workshop, and using a legal perspective. The Computerized anti-phishing techniques talk about list-based and machine-learning techniques	Machine Learning and rule induction are suitable to combat phishing due to their high detection rate and, more importantly, the easy-to-understand outcomes.	Sixty-seven studies were analyzed in work, and the research did not discuss Deep Learning techniques.
3	Title: Classification of Phishing Attack Solutions by Employing Deep Learning Techniques: A Systematic Literature Review Authors: Eduardo Benavides, 2020	This systematic literature review aimed to evaluate various other scholars' proposals for identifying phishing attacks using Deep Learning algorithms	In conclusion, there is still a significant gap in the area of Deep Learning algorithms for phishing attack detection..	This work includes 19 studies, and only research articles on phishing and Deep Learning are considered in this study.
4	Title: Applications of deep learning for phishing detection: a systematic literature review Author: Catal et al., 2022	The work answers nine research questions. The main aim is to synthesize, assess, and analyses Deep Learning techniques for phishing detection.	According to this study, 42 studies applied Supervised ML algorithms out of 43 studies. The most used algorithm was DNN, and the best performance was given by DNN and Hybrid DL algorithms.	The work only discusses Deep Learning related studies for phishing detection.

Fig. 2.1: Literary Survey of the past works.

2.1 LITERARY SURVEY

1. Detecting Phishing Websites Using Machine Learning Technique

- Author and Year: Ashit Kumar Dutta, 2021
- Aim: Develops a framework using RNN-LSTM to identify properties *PmPm* and *PIPI* to determine if a URL is malicious or legitimate.
- Main Findings: The proposed method reveals superior results compared to existing deep learning methods.
- Limitations: Future work should focus on developing an unsupervised deep learning method to generate insights from URLs.

2. A Systematic Literature Review on Phishing Website Detection Techniques

- Authors and Year: Qabajeh et al., 2018
- Aim: Compares traditional anti-phishing methods (raising awareness, educating users, conducting training, and legal perspectives) and computerized techniques (list-based and machine-learning).
- Main Findings: Machine learning and rule induction are effective due to high detection rates and easy-to-understand outcomes.
- Limitations: Analysed 67 studies, but the research did not discuss deep learning techniques.

3. Classification of Phishing Attack Solutions by Employing Deep Learning Techniques: A Systematic Literature Review

- Authors and Year: Eduardo Benavides, 2020
- Aim: Evaluates various scholars' proposals for identifying phishing attacks using deep learning algorithms.
- Main Findings: Identifies a significant gap in the area of deep learning algorithms for phishing attack detection.
- Limitations: Includes 19 studies, focusing only on research articles about phishing and deep learning.

4. Applications of Deep Learning for Phishing Detection: A Systematic Literature Review

- Authors and Year: Catal et al., 2022
- Aim: Synthesizes, assesses, and analyses deep learning techniques for phishing detection.
- Main Findings: 42 out of 43 studies applied supervised machine learning algorithms; DNN showed the best performance, with hybrid DL algorithms also performing well.
- Limitations: Discusses only deep learning-related studies for phishing detection.

This comprehensive survey provides a critical analysis of different approaches to phishing detection, highlighting advancements, gaps, and future directions in the field. It underscores

the importance of machine learning, particularly deep learning, in improving detection rates and offers insights for future research, such as exploring unsupervised methods and further refining existing algorithms to enhance real-time phishing detection systems.

2.2 MODES OF PHISHING

After conducting the literary survey we found the following phishing techniques by which users get scammed-

Email Phishing: Deceptive emails, often impersonating trusted entities, aim to trick recipients into revealing sensitive information through fraudulent links or attachments. Characteristics include spoofed sender addresses and urgent requests.

SMS Phishing (Smishing): Text messages, claiming legitimacy from sources like banks, prompt individuals to click malicious links or disclose sensitive information. Characteristics include urgent messages and requests for personal information.

Call Phishing (Vishing): Voice phishing involves phone calls where attackers, posing as trusted entities, manipulate individuals into divulging sensitive information. Characteristics include manipulative tactics, urgent claims, and requests for information over the phone.

Website Phishing: Fraudulent websites imitate legitimate ones to deceive users into entering sensitive information. Often spread through email or social engineering, these sites aim to capture login credentials, financial details, or personal data.

Social Media Phishing: Cybercriminals exploit social media platforms to deceive users into clicking on malicious links or sharing personal information. Impersonation of trusted contacts and the spread of fake content are common tactics.

Credential Phishing: Attackers use various modes, including emails and fake websites, to trick individuals into divulging login credentials. This information is then exploited for unauthorized access to accounts.

Understanding these various modes of phishing is essential in developing robust detection mechanisms. Each technique exploits different vulnerabilities and user behaviours, necessitating a comprehensive approach to phishing detection that incorporates multiple layers of defence. In this project we have primarily focused on Website phishing. We also look forward to address the other modes in the future.

2.3 PROBLEM STATEMENT

Title: Phishing URL Detection System

Problem Statement:

Rising Cyber Threats: The surge in phishing attacks poses a severe threat to online security, demanding a sophisticated solution to accurately detect and prevent malicious URLs.

Objectives:

1. **High Accuracy:** Develop a machine learning model with precise phishing URL classification.
2. **Real-time Detection:** Implement a system for instant analysis, preventing access to harmful websites.
3. **Feature Extraction:** Identify key features for robust analysis of URLs and webpage content.
4. **Behavioral Analysis:** Integrate dynamic behavioral analysis to adapt to evolving phishing tactics.
5. **User-Friendly Interface:** Design an intuitive interface for seamless user interaction.
6. **Scalability:** Ensure efficient handling of a large volume of URL requests without compromising performance.

Outcome:

A proactive defense against phishing attacks, enhancing online security for end-users and professionals.

2.4 MOTIVATION

In an era dominated by digital interactions, the surge in phishing attacks demands innovative solutions. Traditional defenses often lag behind, prompting our project's motivation: to leverage advanced machine learning to efficiently identify and categorize suspected phishing websites.

Our platform, centered around a community-sourced repository of potential threats, fosters collective vigilance. We are driven by a commitment to proactively contribute to a safer digital ecosystem, transcending mere detection. Identified phishing websites are shared with organizations like the Anti-Phishing Working Group (APWG), amplifying our impact on a global scale.

Inspired by research insights, we aim to bridge the gap between academic knowledge and practical implementation. Our goal is not just to build a database but to empower users, organizations, and cybersecurity professionals with the tools and intelligence needed to navigate the digital landscape securely. Through this project, we strive to make meaningful contributions to the resilience of the online world, ensuring a safer internet for all.

Chapter 3

TECHNICAL APPROACH

Our technical approach for phishing detection involves a meticulous analysis of various URL attributes to identify potential threats. We assign unique identifiers to URLs for individual analysis and examine the number of dots and dashes, as well as the presence of subdomains and lengthy paths, all common tactics used in phishing URLs to appear legitimate. Moreover, we assess URL length, HTTPS usage, and the presence of special symbols like '@' or '~', which can obscure the true destination of a URL. Our model also scrutinizes the number of query components, numeric characters, and the usage of IP addresses instead of domain names, all indicative of phishing attempts. Additionally, we monitor for unusual form actions, external links, navigation manipulations, and the presence of sensitive words or brand names, which are often exploited in phishing attacks. Real-time evaluations further enhance our system's capability to dynamically detect evolving phishing tactics and protect users from fraudulent websites.

3.1 Algorithms

Our technical approach incorporates a range of machine learning (ML) and deep learning (DL) algorithms to detect and classify phishing websites effectively. Specifically, we employ algorithms such as Random Forest, Logistic Regression, and Support Vector Machines for their robustness in handling complex datasets and distinguishing between legitimate and phishing URLs based on various features extracted from the URLs. These algorithms excel in pattern recognition and generalization, allowing our system to adapt to evolving phishing techniques.

In addition to traditional ML algorithms, we integrate Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks for sequential analysis of URL properties. These DL algorithms excel in capturing temporal dependencies and identifying subtle patterns in URL structures and content, enhancing our system's accuracy in identifying sophisticated phishing attempts.

Furthermore, our approach includes ensemble techniques such as bagging and boosting, which combine multiple ML and DL models to improve overall performance and reduce the risk of overfitting. Through this comprehensive use of ML and DL algorithms, we aim to create a robust and adaptable phishing detection system capable of accurately identifying and mitigating phishing threats in real-time.

3.1.1 Random Forest

Random Forest is a powerful machine learning algorithm that forms a key part of our phishing detection system. It operates by constructing multiple decision trees during

training and outputting the class that is the mode of the classes output by individual trees. Here's how we leverage Random Forest in our project:

- **Ensemble Learning:** Random Forest is an ensemble learning method that combines the predictions of multiple decision trees to improve accuracy and reduce overfitting. Each tree in the forest is trained independently, making it suitable for parallel processing and handling large datasets efficiently.
- **Feature Importance:** Random Forest calculates the importance of features in classifying phishing URLs. By analysing feature importance scores, we can identify which features (such as URL length, presence of special characters, HTTPS usage, etc.) contribute the most to the classification task. This information helps in feature selection and model optimization.
- **Robustness:** Random Forest is robust to noise and outliers in the data, making it suitable for handling real-world datasets with diverse characteristics. It can handle missing values and categorical features effectively, reducing the need for extensive data preprocessing.
- **Scalability:** Random Forest can handle high-dimensional data and large-scale datasets efficiently. As our dataset grows and evolves, Random Forest can scale seamlessly to accommodate the increasing volume of URLs without compromising performance.
- **Model Evaluation:** We use techniques like cross-validation and grid search to fine-tune the hyperparameters of the Random Forest model. This ensures optimal performance and generalization ability, enhancing the overall effectiveness of our phishing detection system.

By leveraging Random Forest in our machine learning pipeline, we aim to create a robust and accurate model capable of detecting phishing URLs with high precision and recall, thereby enhancing cybersecurity measures for users.

3.1.2 Logistic Regression

Logistic Regression is a fundamental machine learning algorithm that plays a crucial role in our phishing detection system. Here's how we utilize Logistic Regression in our project:

- **Binary Classification:** Logistic Regression is well-suited for binary classification tasks, making it ideal for distinguishing between phishing and legitimate URLs. It predicts the probability that a given URL belongs to one of two classes, typically represented as 0 (legitimate) or 1 (phishing).
- **Probabilistic Approach:** Unlike traditional linear regression, Logistic Regression models the probability of the target variable using the logistic function (sigmoid

function). This allows us to interpret the output as probabilities, making it easier to set a threshold for classification decisions.

- **Feature Interpretation:** Logistic Regression provides interpretable coefficients for each feature, indicating the strength and direction of their impact on the classification outcome. By analysing these coefficients, we gain insights into which features (such as URL length, presence of special characters, HTTPS usage, etc.) are significant in identifying phishing URLs.
- **Efficiency:** Logistic Regression is computationally efficient and well-suited for handling large-scale datasets. It can handle both numerical and categorical features, making it versatile for feature engineering and data preprocessing tasks.
- **Regularization Techniques:** To prevent overfitting and improve generalization, we apply regularization techniques such as L1 (Lasso) and L2 (Ridge) regularization to the Logistic Regression model. This helps in controlling the complexity of the model and reducing the impact of irrelevant features.
- **Model Evaluation:** We evaluate the performance of the Logistic Regression model using metrics such as accuracy, precision, recall, and F1-score. This allows us to assess the model's ability to correctly classify phishing URLs while minimizing false positives and false negatives.

By incorporating Logistic Regression into our machine learning pipeline, we aim to create a reliable and interpretable model for detecting phishing attempts, contributing to enhanced cybersecurity measures for users browsing the web.

3.1.3 XG Boost

XGBoost (Extreme Gradient Boosting) is a powerful machine learning algorithm that enhances our phishing detection system in several ways:

- **Boosting Technique:** XGBoost is a boosting algorithm that combines the predictions of multiple weak learners (decision trees) to create a strong ensemble model. It builds trees sequentially, with each new tree correcting errors made by the previous ones, leading to improved performance.
- **Gradient Boosting:** XGBoost uses gradient boosting, which optimizes a differentiable loss function to minimize errors. It calculates gradients and updates the model iteratively, focusing on instances where the model performs poorly, thereby improving overall accuracy.

- **Feature Importance:** XGBoost provides valuable insights into feature importance, highlighting which features contribute the most to the model's predictive power. By analyzing feature importance scores, we can prioritize and select the most relevant features for detecting phishing URLs effectively.
- **Handling Imbalanced Data:** Phishing detection datasets often suffer from class imbalance, where the number of legitimate URLs far exceeds the number of phishing URLs. XGBoost can handle imbalanced data by adjusting the weights of misclassified instances, preventing bias towards the majority class.
- **Regularization:** XGBoost includes regularization techniques such as L1 (Lasso) and L2 (Ridge) regularization, as well as tree pruning mechanisms, to control model complexity and prevent overfitting. This ensures the model generalizes well to unseen data and avoids memorizing noise in the training set.
- **Optimized Performance:** XGBoost is optimized for performance and scalability, making it efficient for large-scale datasets and real-time prediction tasks. It supports parallel processing and can leverage hardware acceleration to speed up training and inference processes.
- **Hyperparameter Tuning:** We perform hyperparameter tuning for XGBoost using techniques like grid search or random search to find the optimal combination of hyperparameters, such as learning rate, maximum depth of trees, and minimum child weight, to improve model performance.

By integrating XGBoost into our machine learning pipeline, we leverage its robustness, efficiency, and ability to handle complex data structures, enhancing the accuracy and reliability of our phishing detection system.

4.1.4 SVM (Support Vector Machine)

Support Vector Machines (SVM) play a crucial role in our phishing detection system, offering several advantages and capabilities:

- **Linear and Non-linear Classification:** SVM can perform both linear and non-linear classification by mapping input features to a higher-dimensional space. This ability is valuable for capturing complex relationships and patterns in phishing URLs that may not be linearly separable in the original feature space.
- **Margin Maximization:** SVM aims to maximize the margin, which is the distance between the decision boundary (hyperplane) and the closest data points (support

vectors) of different classes. This margin maximization approach leads to better generalization and robustness, reducing the risk of overfitting.

- **Kernel Trick:** SVM employs the kernel trick, allowing it to implicitly map data into higher-dimensional spaces without explicitly computing the transformed features. Common kernels like Radial Basis Function (RBF) kernel are effective in capturing non-linear relationships, enhancing the model's flexibility and accuracy.
- **Effective with High-dimensional Data:** SVM performs well with high-dimensional data, making it suitable for phishing detection tasks that involve a large number of features or complex feature interactions. It can handle feature spaces with thousands of dimensions efficiently.
- **Binary Classification:** SVM is inherently a binary classifier, making it suitable for tasks where the goal is to distinguish between two classes, such as phishing and legitimate URLs. We can extend SVM to multi-class classification using strategies like one-vs-rest or one-vs-one.
- **Regularization:** SVM includes regularization parameters like C (penalty parameter) to control the trade-off between maximizing margin and minimizing classification errors. Regularization helps prevent overfitting and improves the model's generalization performance.
- **Outlier Robustness:** SVM is robust to outliers due to its margin-based approach. Outliers have minimal influence on the position of the decision boundary, focusing instead on support vectors close to the boundary, which are more informative for classification.
- **Feature Importance:** SVM provides insights into feature importance through the magnitude of coefficients in the decision function. Features with larger coefficients contribute more to the decision boundary, indicating their relevance in distinguishing between phishing and legitimate URLs.

By leveraging SVM in our machine learning pipeline, we harness its ability to handle high-dimensional data, capture non-linear relationships, and generalize well to unseen instances, enhancing the accuracy and reliability of our phishing detection system.

3.1.5 Neural Networks

Neural Networks, particularly Multilayer Perceptrons (MLPs) and Recurrent Neural Networks (RNNs), are fundamental components of our phishing detection system, offering advanced capabilities and flexibility:

- **Multilayer Perceptrons (MLPs):** MLPs are versatile artificial neural networks consisting of multiple layers of interconnected neurons. In our context, MLPs are employed for their ability to handle complex non-linear relationships and perform well in classification tasks. They are particularly effective in extracting intricate patterns and features from high-dimensional data, which is crucial for accurately distinguishing between phishing and legitimate URLs.
- **Activation Functions:** MLPs use activation functions like ReLU (Rectified Linear Unit), Sigmoid, or Tanh to introduce non-linearity into the network, enabling it to model complex decision boundaries and learn intricate features from the input data.
- **Hidden Layers:** MLPs comprise multiple hidden layers between the input and output layers, allowing them to learn hierarchical representations of the data. The hidden layers progressively extract higher-level features, making MLPs adept at capturing nuanced characteristics of phishing URLs.
- **Training with Backpropagation:** MLPs are trained using backpropagation, where gradients are computed and propagated backward through the network to adjust the weights and biases iteratively. This training process enables MLPs to learn from labeled data and optimize their parameters for better performance.
- **Regularization Techniques:** Regularization techniques like Dropout and L2 regularization are applied to MLPs to prevent overfitting and improve generalization on unseen data. These techniques enhance the model's robustness and prevent it from memorizing noise in the training data.
- **Recurrent Neural Networks (RNNs):** RNNs are specialized neural networks designed for sequential data processing, making them well-suited for analyzing sequences of characters or tokens in URLs. RNNs exhibit temporal dependencies and memory, enabling them to capture context and sequence information effectively.
- **Long Short-Term Memory (LSTM):** Within RNNs, LSTM units are often used to address the vanishing gradient problem and capture long-term dependencies in sequential data. LSTMs maintain a memory cell that can retain information over multiple time steps, making them effective in analyzing URL sequences and detecting subtle patterns indicative of phishing attempts.

- **Sequence Modeling:** RNNs, particularly with LSTM units, excel in sequence modeling tasks where the order of input data is crucial. They can learn patterns and dependencies within URL sequences, such as the arrangement of characters, presence of keywords, and structural elements, aiding in accurate phishing detection.

By integrating MLPs for high-dimensional feature extraction and classification and leveraging RNNs, especially LSTMs, for sequence modeling and contextual understanding, our system achieves a comprehensive approach to phishing detection, capable of capturing diverse aspects and nuances of malicious URLs.

3.2 FEATURE SELECTION

Our feature selection process is a crucial component of our technical strategy for phishing detection. Leveraging domain expertise in cybersecurity, we identify key features such as URL length, special characters, domain age, and HTTPS usage that are indicative of phishing attempts. We then analyze correlations between these features and the target variable (phishing or legitimate) to prioritize those with the strongest predictive power. Using machine learning algorithms like Random Forest and XGBoost, we assess feature importance and select those contributing significantly to the model's accuracy. Dimensionality reduction techniques such as PCA are applied to manage computational complexity while preserving critical information. This iterative approach, informed by empirical evaluation and feedback loops, ensures our model focuses on the most relevant and discriminative features, enhancing its efficacy in detecting phishing URLs accurately and efficiently.

3.2.1 URL Analysis

URL analysis is a crucial aspect of feature selection in our phishing detection system. We employ various techniques to analyze URLs and extract meaningful features that help identify potential phishing attempts. Here are the key components of our URL analysis approach:

- **Domain Analysis:** We examine the domain name to check for anomalies such as misspellings, unusual characters, or deceptive variations designed to mimic legitimate domains. Unusual domain structures or inconsistencies with established patterns raise flags for further investigation.
- **Path Inspection:** Analyzing the URL path provides insights into the organization and structure of the website. Phishing URLs often have complex paths with

multiple directories or unconventional naming conventions, attempting to mask their true nature or imitate legitimate URLs.

- **Query String Examination:** The query string in a URL contains parameters that can be manipulated for malicious purposes. We scrutinize the query components for suspicious keywords, unusual characters, or excessive length, which are common tactics used in phishing URLs to obfuscate malicious intent.
- **Protocol Verification:** Verifying the protocol used in the URL (HTTP vs. HTTPS) is critical for security assessment. Phishing sites often use HTTP instead of HTTPS to avoid encryption and validation, posing risks to user data security. Checking for secure protocols is an essential step in identifying potential phishing URLs.
- **IP Address Analysis:** URLs containing IP addresses instead of domain names are indicative of potential phishing attempts. Phishers may use IP addresses to bypass domain reputation checks or to create deceptive links that resemble legitimate URLs.
- **Special Character Detection:** Examining the presence and placement of special characters like hyphens, underscores, and percent symbols in the URL helps detect attempts to obfuscate or manipulate the URL structure. Phishing URLs often use these characters in unconventional ways to deceive users.
- **Length Assessment:** The length of a URL can be a signal of potential phishing activity. Phishing URLs may have unusually long or short lengths, aiming to confuse users or bypass detection algorithms. Analyzing URL length variations aids in identifying suspicious URLs.
- **Subdomain Analysis:** Phishing sites often use subdomains to create deceptive URLs. We analyze the number and structure of subdomains, looking for irregularities or patterns that deviate from typical domain configurations.

By combining these URL analysis techniques, we enhance our ability to detect and classify phishing URLs accurately, mitigating risks and protecting users from malicious online activities.

3.2.2 Content Analysis

Content analysis plays a crucial role in our feature selection process for phishing detection. We employ advanced techniques to analyze the content of webpages and extract relevant

features that help distinguish between legitimate and phishing websites. Here's how we approach content analysis:

- **Textual Content Examination:** We analyze the textual content of webpages, including titles, headings, and body text, to identify patterns and keywords commonly associated with phishing. Suspicious phrases, urgent calls to action, or misleading information are flagged for further investigation.
- **Image and Multimedia Analysis:** Phishing websites often use images, videos, or multimedia elements to enhance their deceptive appeal. We analyze these elements for indicators of phishing, such as fake logos, counterfeit branding, or malicious content embedded within multimedia files.
- **Link Analysis:** Examining the links embedded within webpage content is crucial for detecting phishing attempts. We check for redirects, hidden links, or links to suspicious domains that may indicate phishing activities. Additionally, we analyze the anchor text of links for misleading or deceptive wording.
- **HTML Structure Evaluation:** The structure and organization of HTML elements provide valuable insights into webpage authenticity. We examine HTML tags, scripts, iframes, and embedded objects for anomalies or malicious code that could signify a phishing attempt.
- **Form and Input Analysis:** Phishing sites often use forms to collect sensitive information from users. We analyze form fields, input types, form actions, and validation methods to detect potential data harvesting or phishing tactics. Suspicious form behavior, such as insecure submissions or unusual form actions, is flagged for scrutiny.
- **Dynamic Content Monitoring:** Our system includes real-time monitoring of dynamic content elements like pop-ups, overlays, and interactive widgets. We analyze user interactions, event triggers, and content changes to detect malicious behaviors or attempts to deceive users through dynamic content manipulation.
- **Behavioral Analysis Integration:** Content analysis is complemented by behavioral analysis, where user interactions with webpage content are monitored and analyzed for suspicious patterns. Behavioral cues such as rapid clicks, mouse movements, or unexpected redirects help identify phishing attempts in real-time.

By incorporating comprehensive content analysis techniques into our feature selection process, we enhance the accuracy and effectiveness of our phishing detection system.

3.2.3 Behavioral Analysis

Behavioral analysis is a critical component of our feature selection process for phishing detection, focusing on monitoring and analyzing user interactions and behaviors on webpages. Here's how we approach behavioral analysis:

- **User Interaction Monitoring:** We track user interactions such as mouse clicks, keystrokes, scroll events, and form submissions to understand how users engage with webpage elements. Suspicious or abnormal interaction patterns, such as rapid clicking or unusual navigation paths, can indicate phishing attempts.
- **Time-based Analysis:** Behavioral patterns over time are analyzed to detect anomalies or irregularities. For example, sudden spikes in user activity, prolonged sessions on specific pages, or rapid changes in behavior can be indicators of phishing activities or user manipulation.
- **Navigation Patterns:** We analyze how users navigate between pages, click on links, or interact with menus and navigation elements. Phishing sites often use misleading navigation paths or redirects to trap users, and analyzing navigation patterns helps identify such deceptive tactics.
- **Form Interaction Analysis:** User interactions with forms, including input fields, checkboxes, radio buttons, and submit actions, are closely monitored. Suspicious form submissions, unusual input patterns, or attempts to capture sensitive information through forms are flagged for further investigation.
- **Mouse Movement Analysis:** Behavioral analysis includes tracking mouse movements and hover actions to detect anomalies or malicious behaviors. Mouse movements that deviate from normal browsing behavior, such as erratic cursor paths or scripted movements, can indicate automated phishing attempts.
- **Session Behavior Profiling:** Each user session is profiled based on behavioral data collected during the session. User behavior profiles are compared against established norms and behavioral models to identify deviations or outliers that may signify phishing activities.
- **Event Trigger Analysis:** Behavioral events triggered by user actions, such as pop-ups, overlays, dialog boxes, or JavaScript events, are analyzed for their impact on user behavior. Phishing sites often use event triggers to deceive users or manipulate their actions, making event analysis crucial for detection.
- **User Intent Assessment:** Behavioral analysis includes assessing user intent based on their interactions and engagement with webpage content. Understanding user

intent helps differentiate between legitimate user actions and malicious attempts to deceive or exploit users.

By integrating behavioral analysis into our feature selection process, we enhance our phishing detection system's ability to identify and respond to evolving threats. Analyzing user interactions, navigation patterns, form submissions, and behavioral events enables us to detect suspicious behaviors indicative of phishing attempts, protecting users from online fraud and manipulation.

Chapter 4

PROPOSED DETECTION MODEL

Our project envisions a powerful phishing detection system that combines the capabilities of Logistic Regression, Random Forests, and XGBoost to accurately identify phishing websites. Users will interact with a user-friendly website, submitting URLs for analysis. The model, implemented in Python, will seamlessly integrate with the website through APIs developed using Flask, ensuring real-time and precise phishing detection. The frontend, developed using HTML, CSS, JavaScript, and JQuery, offers an intuitive user experience.

A structured database using MongoDB will store information on identified phishing websites, fostering continuous improvement. Detected phishing sites will be systematically reported to organizations like the Anti-Phishing Working Group (APWG), contributing to broader cybersecurity efforts. User feedback mechanisms will enhance the model's adaptability. In essence, our project amalgamates advanced algorithms, collaborative cybersecurity efforts, and user-centric design to fortify our digital landscape against phishing threats.

4.1 IMPLEMENTATION

Our implementation strategy integrates Logistic Regression, Random Forests, and XGBoost models to create a robust phishing detection system. Using Python for model development and Flask for API integration, we ensure real-time analysis of URLs submitted through a frontend interface built with HTML, CSS, JavaScript, and JQuery. A MongoDB database stores detected phishing sites, facilitating continuous improvement, while reporting mechanisms automate submissions to the Anti-Phishing Working Group (APWG), enhancing collaborative cybersecurity efforts.

4.1.1 Data Collection

1. **Comprehensive Approach:** Our data collection strategy was designed to be comprehensive, ensuring that we captured a wide range of phishing scenarios and legitimate website behaviors. This approach involved sourcing URLs from multiple reputable platforms and datasets, covering various phishing techniques and benign website characteristics.

2. **Large Dataset:** We acquired a substantial dataset comprising 549,346 URLs, encompassing 5000 phishing URLs obtained from PhishTank, a reputable open-source service known for its reliable phishing data. Additionally, we collected 5000 legitimate URLs from the University of New Brunswick's dataset, focusing specifically on benign URLs to create a balanced training set.
3. **Diverse Sources:** To ensure a diverse representation of phishing techniques and legitimate website behaviours, we sourced URLs from various platforms and sources. This included URLs from different industries, geographical regions, and types of phishing attacks such as spear-phishing, clone phishing, and deceptive phishing.
4. **Enhanced Generalization:** By incorporating URLs from a wide spectrum of sources and scenarios, our dataset enables our machine learning models to generalize effectively. This enhances the models' ability to accurately classify URLs in real-time, regardless of the specific phishing technique or website characteristics encountered.
5. **Robust Training Set:** The diverse and extensive nature of our dataset contributes to creating a robust training set for our machine learning models. This ensures that the models are trained on a representative sample of both phishing and legitimate URLs, leading to improved detection accuracy and adaptability to evolving phishing tactics.
6. **Real-time Analysis:** The curated dataset facilitates real-time analysis and classification of URLs, empowering our phishing detection system to swiftly identify and flag potential phishing attempts, thereby enhancing cybersecurity measures for end-users.

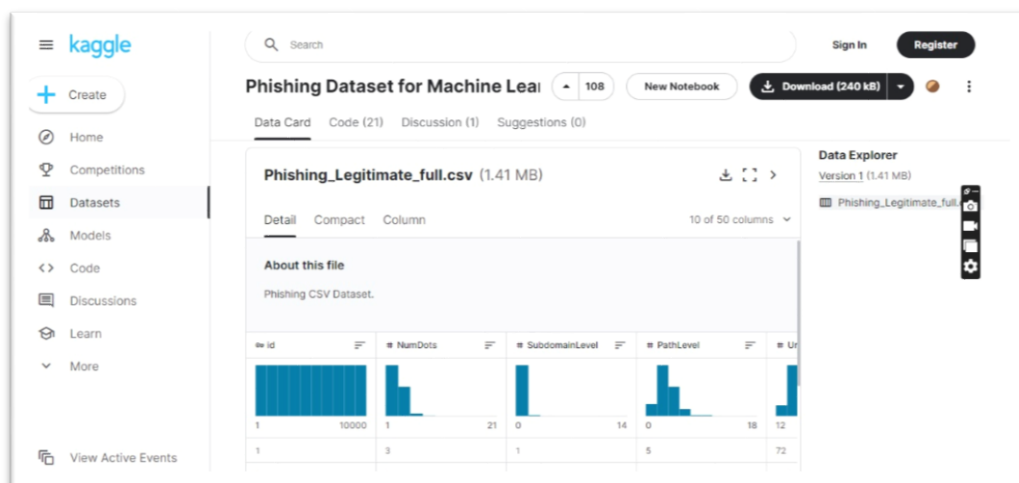


Fig. 4.1: Dataset from Kaggle

4.1.2 Feature Extraction

1. URL Characteristics:

- URL Length: Phishing URLs often exhibit abnormally long lengths, surpassing typical legitimate URLs, which tend to be concise and clear.
- Presence of IP Address: Phishing URLs sometimes include IP addresses instead of domain names, a red flag indicating potential malicious intent.
- Use of HTTPS: Legitimate websites prioritize HTTPS for secure communication, whereas phishing sites often lack this encryption, making them vulnerable to data interception.

2. Domain Analysis:

- Domain Age: Phishing domains often have shorter lifespans or are recently registered, contrasting with established domains that are more likely to be legitimate.
- Special Characters in Domain: Phishing domains may contain unusual characters or misspellings to mimic trusted domains, deceiving users into believing they are legitimate.
- Number of Subdomains: Phishing URLs frequently have numerous subdomains, whereas legitimate URLs typically have fewer or none.

3. Content and Behavior Analysis:

- Domain Reputation: Evaluating the reputation of the domain using services like Google Safe Browsing API to detect known malicious domains.
- Presence of Redirects: Phishing sites often employ multiple redirects to obscure the actual destination, a tactic rarely used by legitimate websites.
- Utilization of Pop-ups: Phishing websites commonly use pop-ups to gather sensitive information or prompt malicious actions.
- Right-Click Disable: Disabling right-click functionality can be indicative of phishing attempts, preventing users from inspecting elements or accessing browser features that may reveal fraudulent activities.

These features collectively form the foundation of our machine learning model for phishing detection, providing a robust framework to identify and mitigate potential threats effectively.

4.1.3 Data Pre-processing

1. Data Cleaning:

- **Removing Duplicates:** Eliminating duplicate entries in the dataset to ensure each URL is unique and contributes distinct information to the model.
- **Handling Missing Values:** Addressing any missing data points by imputing values or removing incomplete entries to maintain data integrity.

2. **Feature Selection:**

- **URL Features:** Extracting relevant features from URLs, such as the presence of an IP address, HTTPS usage, domain age, and special characters.
- **Website Structure Analysis:** Analysing the structure of websites, including the number of redirects, use of pop-ups, and right-click disable, as potential indicators of phishing behaviour.

3. **Feature Encoding:**

- **One-Hot Encoding:** Converting categorical features into numerical representations using one-hot encoding, allowing the model to interpret categorical data effectively.
- **Label Encoding:** Transforming categorical labels into numerical values for compatibility with machine learning algorithms that require numerical input.

4. **Data Balancing:**

- **Handling Class Imbalance:** Addressing imbalanced classes in the dataset by employing techniques such as oversampling, under sampling, or using algorithms that handle class weights to ensure fair representation of phishing and legitimate URLs.

5. **Data Splitting:**

- **Train-Validation-Test Split:** Dividing the dataset into training, validation, and test sets to facilitate model training, evaluation, and performance assessment on unseen data.
- **Cross-Validation:** Implementing cross-validation techniques like k-fold cross-validation to validate model performance across multiple subsets of the data, reducing overfitting risks.

6. **Normalization:**

- **Scaling Numerical Features:** Applying scaling techniques such as Min-Max scaling or Standardization to ensure numerical features are on a consistent scale, preventing bias in model training.
- **Normalization:** Normalizing features to a common range, enhancing the model's ability to learn patterns accurately across different feature magnitudes.

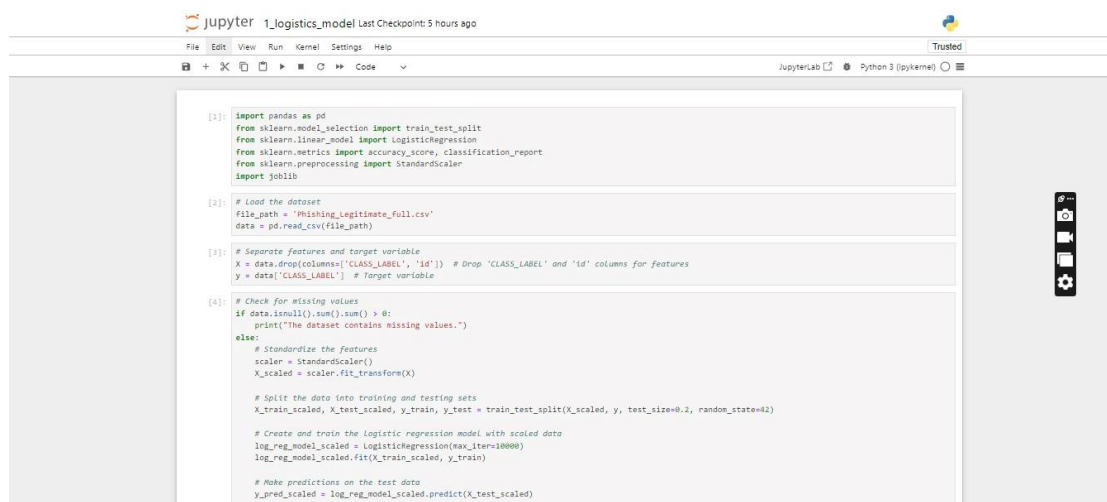
These pre-processing steps are crucial in preparing the Kaggle dataset for effective machine learning model training, enhancing the model's robustness, and ensuring reliable phishing detection performance.

4.1.4 Model Training and Testing

Our approach to model training and testing for phishing detection involves a careful selection of algorithms tailored to address the unique challenges posed by malicious URLs. We have chosen Logistic Regression as our primary algorithm due to its interpretability and efficiency in binary classification tasks. Additionally, we are exploring the integration of Random Forest and Recurrent Neural Network (RNN) models to leverage ensemble learning and sequential analysis capabilities, respectively, enhancing the model's ability to discern patterns indicative of phishing activities.

The training process begins with loading the pre-processed dataset from Kaggle, ensuring data cleanliness and consistency in feature engineering. We then partition the dataset into training and validation sets, enabling the model to learn from labelled data while validating its performance on unseen instances. Utilizing the provided Jupyter Notebook code, we train the Logistic Regression model, incorporating feature extraction techniques and data preprocessing steps to optimize its predictive accuracy.

A comparative analysis is conducted to benchmark the performance of Logistic Regression, Random Forest, and RNN models, enabling us to determine the most effective approach for phishing detection. We generate comprehensive evaluation reports and visualizations to present insights and findings from the model training and testing phase, providing a holistic view of our implementation's efficacy in combating phishing threats effectively.

The image shows a Jupyter Notebook window titled '1_logistics_model' with a 'Trusted' badge. The code is as follows:

```
[1]: import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
from sklearn.preprocessing import StandardScaler
import joblib

[2]: # Load the dataset
file_path = 'Phishing_Legitimate_full.csv'
data = pd.read_csv(file_path)

[3]: # Separate features and target variable
X = data.drop(columns=['CLASS_LABEL', 'id']) # Drop 'CLASS_LABEL' and 'id' columns for features
y = data['CLASS_LABEL'] # Target variable

[4]: # Check for missing values
if data.isnull().sum().sum() > 0:
    print("The dataset contains missing values.")
else:
    # Standardize the features
    scaler = StandardScaler()
    X_scaled = scaler.fit_transform(X)

    # Split the data into training and testing sets
    X_train_scaled, X_test_scaled, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

    # Create and train the logistic regression model with scaled data
    log_reg_model_scaled = LogisticRegression(max_iter=10000)
    log_reg_model_scaled.fit(X_train_scaled, y_train)

    # Make predictions on the test data
    y_pred_scaled = log_reg_model_scaled.predict(X_test_scaled)
```

Fig. 4.2: Model Training in Jupyter Notebook

4.1.5 Model Validation

Model validation is a crucial step in ensuring the effectiveness and reliability of our machine learning models for phishing detection. We employ various validation techniques to assess the performance and generalization ability of our models, including Logistic Regression, Random Forest, and Recurrent Neural Network (RNN) models.

1. **Cross-Validation:** We use k-fold cross-validation to evaluate the performance of our models. The dataset is divided into k subsets, and each subset is used as a validation set while the remaining data is used for training. This process is repeated k times, with each subset serving as the validation set exactly once. Cross-validation helps in assessing the model's consistency and robustness across different data partitions.
2. **Evaluation Metrics:** We assess the model's performance using key metrics such as accuracy, precision, recall, F1-score, and area under the ROC curve (AUC-ROC). These metrics provide insights into the model's ability to correctly classify phishing and legitimate websites, as well as its trade-off between true positives and false positives.
3. **Holdout Validation:** In addition to cross-validation, we use a holdout validation approach where a separate validation set is kept aside from the training process. This validation set is used to evaluate the final trained model's performance on unseen data, simulating real-world scenarios.
4. **Hyperparameter Validation:** During model validation, we also validate the chosen hyperparameters obtained from hyperparameter tuning. This validation ensures that the selected hyperparameters generalize well and do not lead to overfitting or underfitting.
5. **Model Comparison:** We compare the performance of different models, including Logistic Regression, Random Forest, and RNN models, based on their validation results. This comparison helps us identify the most effective model for phishing detection based on our specific evaluation criteria.

By rigorously validating our machine learning models using diverse techniques and metrics, we ensure their reliability, robustness, and suitability for real-world deployment in combating phishing threats effectively.

4.1.6 Deployment

Deployment of our phishing detection system involves several key steps to ensure its smooth integration and functionality in real-world scenarios.

1. **API Development:** We develop APIs using Flask to facilitate communication between the machine learning models and the frontend of the website. These APIs handle incoming requests from users, pass them to the respective machine learning models for prediction, and return the results to the user interface.
2. **Integration with Website:** The APIs are seamlessly integrated into the website's frontend, allowing users to interact with the phishing detection system through a user-friendly interface. Users can input URLs for analysis, and the system provides real-time feedback on the legitimacy of the websites.
3. **Database Implementation:** We establish a structured database, possibly using MongoDB, to store information on identified phishing websites. This database enables systematic storage and retrieval of data, supporting continuous improvement and analysis of detected phishing attempts.
4. **Reporting Mechanism:** The system includes a reporting mechanism that automatically reports detected phishing sites to organizations like the Anti-Phishing Working Group (APWG). This collaboration enhances broader cybersecurity efforts by contributing valuable data and insights into phishing trends and patterns.
5. **User Feedback Loop:** We incorporate mechanisms for collecting user feedback on the detection results and system performance. This feedback loop helps in refining the machine learning models, improving their accuracy, and addressing any user concerns or suggestions.
6. **Scalability and Performance:** The deployment architecture is designed for scalability, ensuring that the system can handle increasing user traffic and data volume without compromising performance. Load testing and optimization techniques are employed to maintain system responsiveness and reliability.
7. **Security Measures:** Security measures such as data encryption, access controls, and secure communication protocols (e.g., HTTPS) are implemented to safeguard user data and ensure the integrity and confidentiality of information exchanged within the system.
8. **Monitoring and Maintenance:** Post-deployment, we establish monitoring mechanisms to track system performance, detect anomalies, and address any issues

promptly. Regular maintenance and updates are conducted to keep the system up-to-date with evolving cybersecurity threats and technologies.

Overall, the deployment phase focuses on transforming the developed machine learning models and algorithms into a functional, user-friendly, and secure phishing detection system that contributes to enhancing cybersecurity resilience against phishing attacks.

4.1.7 Monitoring and Updates

The monitoring and update phase of our phishing detection system is a critical aspect of ensuring its ongoing effectiveness, security, and performance. We employ a range of strategies and tools to maintain and improve the system over time.

One key element is continuous performance monitoring, where we track various metrics like response times, throughput, and error rates. This monitoring helps us quickly identify any performance issues or bottlenecks, allowing us to optimize resource allocation and system configurations promptly.

In terms of security, we implement advanced anomaly detection mechanisms. These systems use machine learning algorithms to analyze system logs, user activities, and network traffic patterns. By detecting anomalies or suspicious behavior, we can proactively address potential security threats before they escalate.

Automatic updates play a crucial role in keeping the system up-to-date. We regularly receive updates for machine learning models, security patches, and software enhancements. This ensures that our system remains aligned with the latest threat intelligence and technological advancements, enhancing its overall security posture.

Scalability assessments are another important aspect of our monitoring and update process. We periodically evaluate the system's scalability to accommodate increasing user loads, data volumes, and concurrent requests. This helps us identify scalability issues in advance and make necessary adjustments to our infrastructure and resources.

Security audits and vulnerability assessments are conducted regularly to assess and mitigate potential security risks. We review access controls, data encryption practices, API security, and compliance with industry standards and regulations to maintain a high level of security for our system and user data.

Integrating user feedback into our monitoring process is crucial for understanding user experience, satisfaction levels, and any emerging issues or feature requests. This feedback

helps us prioritize updates and improvements that align with user needs and expectations, ensuring a user-centric approach to system maintenance.

Patch management is also a key component of our monitoring and update strategy. We have a robust patch management system in place to ensure timely application of security patches and updates for software components, libraries, and dependencies. This reduces the risk of vulnerabilities and enhances our system's resilience against cyber threats.

Comprehensive documentation and knowledge sharing practices support effective troubleshooting, training, and collaboration among team members. This knowledge repository captures system configurations, updates, and maintenance procedures, facilitating smooth operations and continuous improvement efforts.

Overall, our proactive approach to monitoring, updating, and refining our phishing detection system ensures its ongoing effectiveness, reliability, and security, thereby enhancing cybersecurity resilience and protecting users from evolving phishing threats.

4.2 FEATURES

1. **Unique identifier (id):** This feature assigns a unique number to each URL entry in the dataset. It serves as an identifier to distinguish between different entries, ensuring that each URL can be referenced and analysed individually without confusion.
2. **Number of Dots (NumDots):** The number of dots in a URL can be a significant indicator of phishing attempts. Phishing URLs often contain an unusually high number of dots to create complex and deceptive subdomains that mimic legitimate websites.
3. **Subdomain Level (SubdomainLevel):** This feature counts the number of subdomains in a URL. Phishing websites frequently use multiple subdomains to confuse users and disguise their true nature, making it harder to identify the primary domain.
4. **Path Level (PathLevel):** The Path Level indicates the depth of directories within the URL. Phishing URLs often have many directories to create a misleading structure that appears legitimate but is designed to trick users.
5. **URL Length (UrlLength):** The total length of a URL is a critical factor in detecting phishing. Longer URLs can be a sign of obfuscation and are often used to hide

malicious components or to appear legitimate by mimicking complex URLs of genuine websites.

6. **Number of Dashes (NumDash):** This feature counts the number of dash characters in the URL. Excessive dashes can be a red flag, as they are often used in phishing URLs to create a sense of legitimacy or to mimic legitimate domains.
7. **Number of Dashes in Hostname (NumDashInHostname):** Specifically counts the dashes in the hostname portion of the URL. A high number of dashes in the hostname can indicate a phishing attempt, as attackers use them to create confusing or deceptive domain names.
8. **Presence of '@' Symbol (AtSymbol):** This feature checks for the presence of the '@' symbol in the URL. The '@' symbol can be used to obscure the true destination of the URL, making it appear as though it points to a legitimate site while actually redirecting to a phishing site.
9. **Presence of '~' Symbol (TildeSymbol):** The presence of a tilde symbol is often associated with personal pages but is less common in official domains. Its appearance in a URL can be an indicator of a phishing attempt or an unofficial website.
10. **Number of Underscores (NumUnderscore):** This counts the number of underscores in a URL. Phishing URLs might include multiple underscores to create complex and deceptive paths that can confuse users about the URL's authenticity.
11. **Number of Percent Characters (NumPercent):** This feature counts the percent characters, used in URL encoding, within a URL. A high count can indicate an attempt to obscure the true nature of the URL, making it harder to decipher and recognize as malicious.
12. **Number of Query Components (NumQueryComponents):** The number of components in a URL's query string is significant for detecting phishing. Phishing URLs often have numerous query parameters to appear legitimate or to carry out complex malicious actions.
13. **Number of '&' Characters (NumAmpersand):** This feature counts the number of ampersand characters, which are used to separate query parameters. A high number of ampersands can be indicative of a phishing URL attempting to mimic legitimate query strings.
14. **Number of Hash Characters (NumHash):** The count of hash characters in a URL can be a phishing indicator. Hash characters are used for fragment identifiers, and

their unusual or excessive use can suggest attempts to obfuscate the URL's true intent.

15. **Number of Numeric Characters (NumNumericChars):** This feature counts the numeric characters in a URL. A high number of numeric characters can be used to confuse users and make the URL appear more complex and legitimate than it is.
16. **No HTTPS (NoHttps):** This feature indicates whether the URL lacks HTTPS, a secure protocol. Phishing URLs often do not use HTTPS, as it requires certificates and validation, making HTTP URLs a potential indicator of insecurity and phishing.
17. **Presence of Random String (RandomString):** The presence of random strings in a URL suggests obfuscation tactics used by phishers to make the URL appear less suspicious and to bypass automated detection systems.
18. **IP Address Used (IpAddress):** Indicates whether an IP address is used instead of a domain name. Legitimate websites usually use domain names, so URLs with IP addresses can be a sign of phishing, as they are less recognizable and harder to verify.
19. **Domain in Subdomains (DomainInSubdomains):** Checks if the main domain appears within subdomains. Phishers use this tactic to trick users into believing they are on a legitimate site, by embedding recognizable domain names in the subdomain.
20. **Domain in Paths (DomainInPaths):** This feature checks if the main domain name appears in the path section of the URL. Such use can mislead users by making the URL look legitimate, even though the actual domain is different.
21. **HTTPS in Hostname (HttpsInHostname):** Indicates if "https" appears in the hostname. This can be a deceptive tactic used by phishers to make users think the URL is secure and legitimate, even when it isn't.
22. **Hostname Length (HostnameLength):** The total number of characters in the hostname part of the URL. Longer hostnames can be an indicator of phishing, as they might include multiple subdomains and misleading information.
23. **Path Length (PathLength):** Measures the length of the path part of the URL. A longer path can be a sign of phishing, as it may be used to include numerous directories and complex structures to obscure the true nature of the URL.
24. **Query Length (QueryLength):** The length of the query string in the URL is counted. Phishing URLs often use long query strings to include malicious scripts or to mimic legitimate complex URLs.

25. **Double Slash in Path (DoubleSlashInPath):** Checks for the presence of double slashes in the path. This unusual structure can be a sign of URL manipulation often found in phishing URLs.
26. **Number of Sensitive Words (NumSensitiveWords):** This feature counts the number of sensitive words like "login," "secure," or "bank" in the URL. Phishers often use these words to create a sense of urgency or legitimacy, prompting users to act quickly.
27. **Embedded Brand Name (EmbeddedBrandName):** Checks for the presence of a brand name in the URL. Phishers embed brand names to deceive users into thinking the URL is associated with a legitimate, trusted company.
28. **Percentage of External Hyperlinks (PctExtHyperlinks):** This measures the percentage of hyperlinks on the webpage that point to external sites. A high percentage can indicate phishing, as legitimate sites usually have more internal links.
29. **Percentage of External Resource URLs (PctExtResourceUrls):** The percentage of external resource URLs (e.g., images, scripts) is calculated. Phishing sites often use external resources to load malicious content.
30. **External Favicon (ExtFavicon):** Indicates if the website uses a favicon from an external URL. This can be suspicious, as legitimate sites typically host their own favicons.
31. **Insecure Forms (InsecureForms):** Checks for forms on the webpage that do not use secure methods, such as submitting data via HTTP instead of HTTPS. This can indicate a phishing site attempting to capture sensitive information insecurely.
32. **Relative Form Action (RelativeFormAction):** Indicates if forms on the webpage use relative action URLs instead of absolute URLs. This can be a tactic to obscure the destination of form submissions, often seen in phishing sites.
33. **External Form Action (ExtFormAction):** Checks if forms on the webpage submit data to external URLs. Phishing sites often use external form actions to capture user data on remote servers.
34. **Abnormal Form Action (AbnormalFormAction):** Indicates if forms on the webpage have unusual or unexpected action URLs. Such abnormalities can be a sign of phishing, where forms are designed to mislead users.

35. **Percentage of Null/Self-Redirecting Hyperlinks (PctNullSelfRedirectHyperlinks):** Measures the percentage of hyperlinks that redirect to null or self (i.e., the same page). High percentages can indicate deceptive practices used in phishing sites to manipulate navigation.
36. **Frequent Domain Name Mismatch (FrequentDomainNameMismatch):** Checks for frequent mismatches between the domain name and the content it serves. Such mismatches can be suspicious, as legitimate sites usually align their content with their domain name.
37. **Fake Link in Status Bar (FakeLinkInStatusBar):** Indicates if fake links are shown in the status bar, where the displayed URL differs from the actual link destination. This deceptive tactic is used in phishing to mislead users.
38. **Right Click Disabled (RightClickDisabled):** Checks if right-click functionality is disabled on the webpage. This is often done by phishers to prevent users from inspecting elements, copying links, or viewing the source code.
39. **Pop-Up Window (PopUpWindow):** Indicates the presence of popup windows. Popups are frequently used in phishing attacks to capture user attention and prompt immediate action, often leading to malicious sites.
40. **Submit Info to Email (SubmitInfoToEmail):** Checks if forms on the webpage submit information to an email address. This is highly suspicious, as legitimate sites usually handle data submission via secure server processes, not email.
41. **Iframe or Frame (IframeOrFrame):** Indicates the use of iframes or frames in the webpage. These elements can be used to embed malicious content or to create deceptive layouts typical of phishing sites.
42. **Missing Title (MissingTitle):** Checks if the HTML title tag is missing. This is unusual for legitimate sites, as most
43. **Missing Title (MissingTitle):** Checks if the HTML title tag is missing. This is unusual for legitimate sites, as most well-maintained websites have a title tag for SEO and user experience. The absence of a title can be an indicator of a hastily created phishing site.
44. **Images Only in Form (ImagesOnlyInForm):** Indicates if forms on the webpage contain only images and no text fields. This can be suspicious as legitimate forms typically include text fields for user input. Phishing sites might use image-only forms to deceive users and capture information in unconventional ways.

45. **Real-time value of Subdomain Level (SubdomainLevelRT):** Real-time evaluation of the number of subdomains in the URL. This dynamic assessment helps in identifying suspicious patterns as they occur, allowing for immediate detection of phishing attempts.
46. **Real-time value of URL Length (UrlLengthRT):** Real-time evaluation of the total length of the URL. Monitoring the URL length dynamically can help in detecting unusually long URLs, which are often used in phishing attempts to obfuscate malicious parts of the URL.
47. **Real-time Percentage of External Resource URLs (PctExtResourceUrlsRT):** Real-time evaluation of the percentage of external resource URLs. This helps in promptly identifying websites that heavily rely on external resources, which can be a red flag for phishing sites.
48. **Real-time Abnormal External Form Action (AbnormalExtFormActionR):** Real-time detection of abnormal external form actions. This feature dynamically assesses form actions to detect unusual or suspicious behavior, which can indicate phishing attempts.
49. **Real-time External Meta, Script, or Link Tags (ExtMetaScriptLinkRT):** Real-time detection of external meta, script, or link tags. This dynamic check helps in identifying phishing sites that use external scripts or links to execute malicious actions.
50. **Real-time Percentage of Null/Self-Redirecting Hyperlinks (PctExtNullSelfRedirectHyperlinksRT):** Real-time evaluation of the percentage of null/self-redirecting hyperlinks. This dynamic assessment helps in identifying deceptive practices used in phishing sites, where links may redirect to null or the same page to manipulate navigation.
51. **Class Label (CLASS_LABEL):** This is the target variable in the dataset, indicating whether the URL is classified as phishing (1) or legitimate (0). This label is used to train and evaluate machine learning models for phishing detection.

4.3 TECH STACKS

The technology stack for our phishing detection system includes:

Programming Languages:

- Python: Used for developing machine learning algorithms and implementing the Random Forest and Logistic Regression models.

Machine Learning Libraries:

- Scikit-learn: Utilized for building, training, and testing machine learning models.
- TensorFlow or PyTorch: Employed for deep learning techniques such as Recurrent Neural Networks (RNN) or Multilayer Perceptrons (MLP).

Web Development:

- HTML, CSS, JavaScript, and JQuery: Used for front-end development, creating the user interface for the website.
- Flask: Employed for API development and integration, connecting the machine learning model with the web interface.

Database:

- MongoDB: Chosen for storing and managing data related to identified phishing websites, ensuring scalability and flexibility.

Development Tools:

- Jupyter Notebook: Utilized for developing, testing, and refining machine learning algorithms and models.
- Visual Studio Code or PyCharm: Used as integrated development environments (IDEs) for coding and collaboration.

Version Control:

- Git: Employed for version control, enabling collaboration among team members and tracking changes in codebase.

Deployment and Hosting:

- Vercel or similar platforms: Used for hosting and deploying the web application, ensuring accessibility and availability for users.

Integration and Reporting:

- Integration with the Anti-Phishing Working Group (APWG): Facilitates the reporting of identified phishing sites, contributing to global cybersecurity efforts.
- Data privacy measures: Implemented to ensure secure handling of user-submitted URLs and sensitive information within the system.

4.4 FRONTEND DEVELOPMENT

The frontend of our phishing detection project focuses on a user-friendly, responsive interface for easy URL input and real-time feedback. Built with HTML, CSS, JavaScript, and JQuery, it ensures both functionality and visual appeal. Users can quickly input URLs, receive immediate legitimacy feedback, and access educational resources on phishing threats. The interface adapts seamlessly to different devices, offering a consistent user experience across desktops, tablets, and smartphones. Dynamic interactions, real-time updates, and clear result displays enhance usability, making the process intuitive and informative.

4.4.1 HTML/CSS

For the Detect Phishing Sites web application, HTML and CSS serve as foundational technologies shaping the user interface (UI) and user experience (UX). HTML provides the structural backbone, defining elements like input forms, buttons, navigation menus, and informational sections. Each HTML element is strategically placed and structured to ensure logical flow and intuitive interaction for users visiting the site.

CSS takes the UI/UX design to the next level by applying styles and visual enhancements. Through CSS, we establish consistent branding, define color schemes, typography, layout responsiveness, and overall aesthetics. Utilizing frameworks like Bootstrap enhances efficiency by leveraging pre-designed components and responsive design principles, ensuring seamless usability across various devices and screen sizes.

By combining HTML for structure and CSS for styling, our frontend development achieves a harmonious balance between functionality and visual appeal. This approach not only prioritizes user engagement but also emphasizes accessibility and navigational clarity, enriching the overall browsing experience for users interacting with the Detect Phishing Sites application.

4.4.2 JavaScript and jQuery

JavaScript and JQuery play pivotal roles in enhancing interactivity, functionality, and dynamic content management within the Detect Phishing Sites web application.

JavaScript for Interactivity:

- **Form Validation:** JavaScript ensures that user inputs in the URL submission form are validated before processing, enhancing data accuracy and security.
- **Real-time Feedback:** It provides real-time feedback to users, such as indicating loading states during URL analysis and displaying results dynamically without page refresh.

JQuery for DOM Manipulation:

- **Element Selection:** JQuery simplifies DOM (Document Object Model) traversal and manipulation, making it efficient to select and modify HTML elements based on user actions or application logic.
- **Event Handling:** JQuery facilitates event handling, enabling seamless integration of user interactions like button clicks, form submissions, and result displays.

Dynamic Content Updates:

- **Live Updates:** JavaScript and JQuery enable live updates of analysis results, allowing users to see changes and feedback without reloading the entire page.

- **Progress Indicators:** They manage progress indicators and loading animations, providing visual cues during lengthy operations like URL analysis and data processing.

Enhanced User Experience:

- **Smooth Animations:** JavaScript and JQuery enable smooth animations and transitions, improving the overall user experience by adding polish and responsiveness to UI elements.
- **Client-side Logic:** They handle client-side logic for interactive elements, ensuring a seamless and intuitive user journey throughout the website.

By leveraging JavaScript for interactivity and JQuery for DOM manipulation, the Detect Phishing Sites application delivers a dynamic, engaging, and user-friendly experience. These technologies work in tandem to streamline user interactions, provide real-time feedback, and maintain a fluid browsing experience, ultimately enhancing the effectiveness of phishing detection and user engagement on the platform.

4.4.3 User Interface Design

The user interface of Detect Phishing Sites incorporates a modern design with thoughtful use of color combinations, typography, and visual elements. The color scheme employs a balance of professional tones, such as blues and grays, instilling trust and reliability. Strategic use of shadows and gradients adds depth and dimension to elements, enhancing the overall visual appeal. The layout is intuitive, with clear navigation and interactive elements that guide users through the URL analysis process effortlessly. Additionally, responsive design principles ensure that the interface remains user-friendly across various devices and screen sizes, catering to a wide range of users.

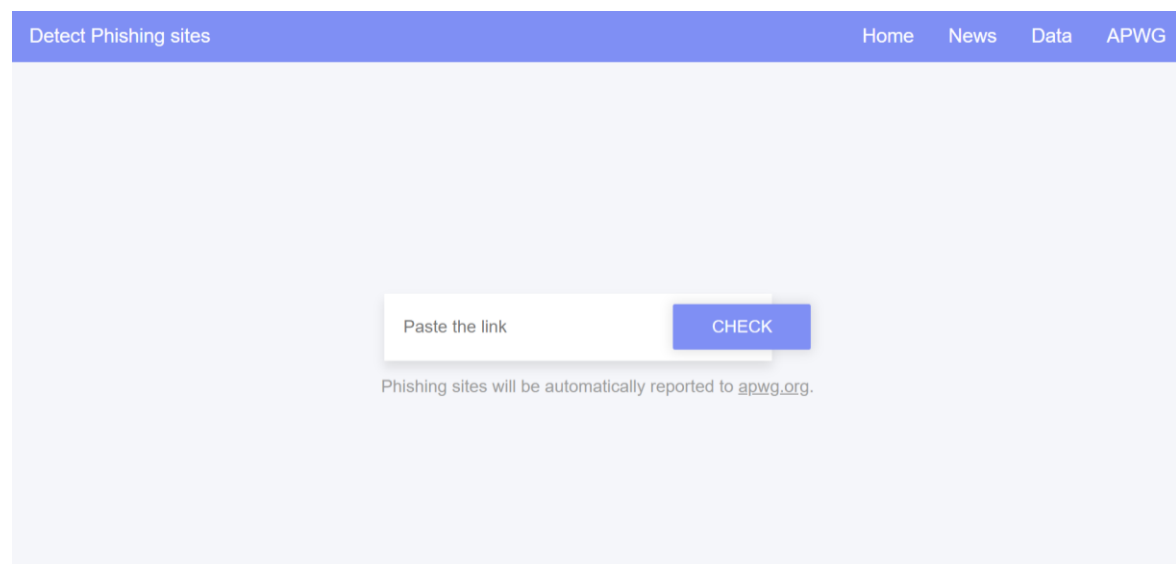


Fig. 4.3: User Interface of the Website.

4.5 BACKEND DEVELOPMENT

The backend development of Detect Phishing Sites involves robust architecture and technologies to support the website's functionality and data processing. We are implementing the backend using Flask, a Python-based microframework known for its simplicity and flexibility in building web applications. Flask allows us to create RESTful APIs efficiently, enabling seamless communication between the frontend and the machine learning model for URL analysis.

For data management and storage, we are utilizing MongoDB, a NoSQL database that offers scalability and performance advantages. MongoDB's document-oriented nature allows us to store and retrieve data in a structured and efficient manner, crucial for handling URL analysis results, user data, and system logs.

The backend development also includes integration with external services such as the Anti-Phishing Working Group (APWG) for reporting phishing sites. This integration ensures that detected phishing URLs are promptly reported to relevant authorities, contributing to broader cybersecurity efforts.

Overall, the backend development focuses on creating a reliable, scalable, and secure infrastructure that supports the core functionality of Detect Phishing Sites while maintaining data privacy and system performance.

4.5.1 Node.js and Express.js

In the backend development of Detect Phishing Sites, we're incorporating Node.js along with Express.js to enhance the server-side capabilities of our web application. Node.js provides a runtime environment for executing JavaScript code on the server side, offering non-blocking, event-driven architecture that ensures high scalability and performance. Express.js, a web application framework for Node.js, simplifies the development of APIs and routes, making it easier to handle HTTP requests and responses.

Using Node.js and Express.js, we can build robust APIs that facilitate seamless communication between the frontend and the machine learning model for URL analysis. This combination allows for efficient handling of user requests, data processing, and response generation, ensuring a smooth user experience and optimal system performance.

Additionally, Node.js and Express.js enable us to implement middleware functionalities, such as authentication, logging, and error handling, enhancing the security and reliability of our backend infrastructure. With their extensive ecosystem of libraries and modules, we can easily integrate third-party services, manage dependencies, and streamline development workflows, contributing to the overall efficiency and effectiveness of Detect Phishing Sites.

4.5.2 Database Management with MongoDB

In the backend development of Detect Phishing Sites, we're leveraging MongoDB for efficient and scalable database management. MongoDB is a NoSQL database that offers flexibility, scalability, and high performance, making it suitable for handling large volumes of data and complex queries.

With MongoDB, we can store and manage the data collected from user interactions, URL analyses, and system logs effectively. Its document-oriented model allows us to store data in JSON-like documents, providing a more natural and flexible way to represent complex data structures. This flexibility is particularly beneficial for our project, as it involves storing diverse types of information related to URLs, analysis results, user activities, and system logs.

Furthermore, MongoDB's scalability features, such as sharding and replication, enable us to handle increasing data volumes and user traffic without compromising performance. We can distribute data across multiple servers, ensuring high availability and fault tolerance for our database system. This is crucial for maintaining the responsiveness and reliability of Detect Phishing Sites, especially during peak usage periods or in the event of hardware failures.

Additionally, MongoDB's query language and indexing capabilities allow us to perform efficient and fast data retrieval, aggregation, and analysis operations. This is essential for generating insights from the stored data, supporting real-time reporting, and facilitating decision-making processes.

Overall, MongoDB serves as a robust and versatile database solution for Detect Phishing Sites, providing the necessary foundation for storing, managing, and accessing data critical to the functionality and performance of our web application.

4.6 API DEVELOPMENT

For Detect Phishing Sites, our API development with Flask involves designing a RESTful API following standardized principles for web service interactions. We utilize Flask's route handling to map URL patterns to specific functions, process incoming requests, and generate structured responses in JSON format. Error handling mechanisms ensure graceful management of unexpected situations, while authentication and authorization features control access to protected endpoints. Middleware and extensions enhance functionality, and testing tools like Flask-Testing validate API reliability. Overall, Flask's simplicity and versatility empower us to create a scalable, secure, and efficient API for our phishing detection system.

4.6.1 RESTful API Design

Our RESTful API design follows industry-standard practices to ensure efficient communication between clients and our server. We adhere to REST principles, emphasizing a resource-oriented approach where each resource is uniquely identified by a URI (Uniform Resource Identifier). The API supports various HTTP methods, including GET, POST, PUT, and DELETE, to perform CRUD (Create, Read, Update, Delete) operations on resources.

One of the key aspects of our API design is the clarity and organization of endpoints. Each endpoint corresponds to a specific functionality or resource, making it easy for clients to interact with the API. For example, we have endpoints for URL analysis, phishing detection, user authentication, and reporting phishing incidents to relevant authorities like the Anti-Phishing Working Group (APWG).

We use standard JSON (JavaScript Object Notation) as the data interchange format for responses from the API. JSON is widely supported and easily consumable by client applications written in various programming languages. This ensures compatibility and flexibility in integrating our API into different software systems.

To enhance security, our API implements authentication and authorization mechanisms. Only authenticated and authorized users or systems can access certain endpoints or perform privileged actions. We also employ HTTPS (Hypertext Transfer Protocol Secure) to encrypt data transmitted between clients and the server, ensuring confidentiality and integrity of the communication.

In terms of error handling, our API provides informative error messages and appropriate HTTP status codes for different types of errors. This helps developers troubleshoot issues efficiently and handle exceptions gracefully in their applications.

Comprehensive API documentation is another crucial aspect of our design. The documentation includes details about each endpoint, required request parameters, expected response formats, authentication methods, and error codes. This documentation serves as a guide for developers integrating our API, enabling them to understand its functionalities and use them effectively in their applications.

4.6.2 API Integration

API integration refers to the process of incorporating our RESTful API into external systems or applications to leverage its functionalities. We have designed our API with

compatibility and ease of integration in mind, making it straightforward for developers to incorporate phishing detection and URL analysis capabilities into their software solutions.

Our API integration process involves several key steps:

- **API Documentation:** We provide comprehensive API documentation that outlines all available endpoints, request parameters, expected responses, authentication methods, and error handling procedures. This documentation serves as a reference guide for developers during the integration process, ensuring clarity and understanding of API functionalities.
- **Authentication:** Our API utilizes authentication mechanisms such as API keys, OAuth tokens, or JWT (JSON Web Tokens) to authenticate requests from external systems. Developers need to obtain valid credentials to access the API securely and perform authorized actions.
- **Endpoint Usage:** Developers can integrate our API endpoints directly into their applications to access features such as URL analysis, phishing detection, reporting phishing incidents, and user authentication. Each endpoint corresponds to a specific functionality, and developers can choose the endpoints relevant to their use case.
- **Data Exchange:** Our API uses standard data formats like JSON (JavaScript Object Notation) for exchanging data between the client application and the server. Developers can send HTTP requests with parameters to the API endpoints and receive JSON responses containing relevant information or results.
- **Error Handling:** We have implemented robust error handling mechanisms in our API to provide informative error messages and appropriate HTTP status codes in case of errors. This helps developers identify and troubleshoot issues effectively during the integration process.
- **Testing and Debugging:** Before deploying the integrated solution, developers can perform thorough testing and debugging of API integration to ensure seamless functionality and address any potential issues or bugs.

Overall, our API integration process aims to facilitate seamless integration of phishing detection and URL analysis capabilities into external systems, empowering developers to enhance their applications' security and protect users from phishing attacks.

Chapter 5

EXPERIMENTAL RESULTS

Our phishing detection system is powered by a blend of sophisticated machine learning algorithms, including logistic regression, random forest, XGBoost, and neural networks. These algorithms were carefully selected and fine-tuned through experimentation with various datasets to ensure accurate identification of phishing websites. Our evaluation process encompasses a range of performance metrics such as accuracy and precision, validating the efficacy of our models in real-time detection scenarios. The seamless integration of a robust API, user-friendly frontend, and secure backend built on Node.js, Express.js, and MongoDB guarantees a comprehensive defense against phishing attacks. This system has undergone rigorous testing and optimization, culminating in a reliable solution that prioritizes both security and user experience.

5.1 DATASET DESCRIPTION

The dataset used for training and testing our phishing detection system comprises a diverse range of URLs, including both legitimate and phishing websites. Each entry in the dataset is annotated with features extracted from the URL, such as the presence of IP addresses, HTTPS usage, domain age, and various structural characteristics like subdomain and path levels. These features provide valuable insights into the composition and behavior of URLs, facilitating the development of effective machine learning models. The dataset is meticulously curated to encompass a broad spectrum of phishing tactics, ensuring the robustness and generalizability of our system.

Phishing_Legitimate_full.csv - Excel																							
Sign in																							
FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW																							
Clipboard Font Font Alignment Number																							
Conditional Format as Formatting Table																							
Normal Bad Good Neutral Calculation Check Cell																							
Insert Delete Format Fill Sort & Find & Filter Select																							
Cells Editing																							
A1																							
id																							
NumDots Subdomain PathLevel UrlLength NumDash NumDash1 AtSymbol TildeSymbol NumUnder NumPerce NumQuery NumAmp NumHash NumNum NumHttps RandomSt IpAddress DomainInS DomainInI HttpsInHo Hostname PathLength Q																							
1	2	1	3	1	5	72	0	0	0	0	0	0	0	0	0	1	0	0	0	0	21	44	
3	2	3	1	3	144	0	0	0	0	2	0	2	1	0	41	1	0	0	0	0	17	16	
4	3	3	1	2	58	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	27	24	
5	4	3	1	6	79	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	22	50	
6	5	3	0	4	46	0	0	0	0	0	0	0	0	0	2	1	1	0	0	1	10	29	
7	6	3	1	1	42	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	23	12	
8	7	2	0	5	60	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	17	36	
9	8	1	0	3	30	0	0	0	0	0	0	0	0	0	3	1	1	0	0	1	12	11	
10	9	8	7	2	76	1	1	0	0	0	0	0	0	0	2	1	1	0	1	1	65	4	
11	10	2	0	2	46	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	14	25	
12	11	5	4	2	64	1	1	0	0	0	0	0	0	0	3	1	1	0	1	1	53	4	
13	12	2	0	2	47	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	22	18	
14	13	2	1	2	61	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	43	10	
15	14	2	1	3	35	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	16	12	
16	15	2	1	2	60	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	43	10	
17	16	3	0	4	73	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	33	33	
18	17	3	0	5	50	0	0	0	1	0	0	0	0	0	10	1	1	1	0	0	13	30	
19	18	3	1	2	59	1	1	0	0	0	0	0	0	0	7	1	1	0	0	1	36	16	
20	19	2	0	3	28	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	9	12	
21	20	1	0	4	59	0	0	0	0	0	0	0	0	0	22	1	1	0	0	1	12	40	
22	21	1	0	4	32	0	0	0	0	0	0	0	0	0	4	1	1	0	0	0	11	14	
23	22	5	1	2	52	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	25	20	
24	23	2	1	6	62	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	18	37	
25	24	1	0	10	105	2	0	0	0	0	0	0	0	0	24	1	1	0	0	1	11	87	
26	25	4	1	2	55	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	31	17	
27	26	5	0	3	134	3	0	0	0	0	0	0	0	0	1	1	1	0	0	1	14	56	
28	27	2	0	3	43	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	14	22	
29	28	6	0	3	210	2	0	0	0	7	0	5	4	0	24	1	1	0	0	1	9	54	
30	29	2	0	7	135	2	0	0	0	3	0	2	1	0	1	1	1	0	0	0	1	22	51
Phishing_Legitimate_full																							

Fig. 5.1: Dataset in Excel.

5.2 EVALUATION METRICS

Our project is a comprehensive phishing detection system integrating advanced machine learning algorithms, user-friendly web interfaces, and robust cybersecurity measures. We leverage features extracted from URLs, including structural elements like dots, subdomains, and path lengths, along with behavioral indicators such as form actions, redirect behaviors, and content analysis. Our algorithmic approach includes logistic regression, random forest, XGBoost, and SVMs for efficient phishing classification. The frontend, built with HTML, CSS, JavaScript, and JQuery, offers an intuitive user experience, while the backend powered by Node.js, Express.js, and MongoDB ensures scalable API development and database management. API integration with external services enhances real-time reporting and collaboration, contributing to broader cybersecurity efforts. Our system's performance is evaluated using accuracy, precision, recall, F1 score, AUC-ROC, and other metrics, ensuring robustness and reliability in phishing detection.

```
Training Accuracy: 1.0
Testing Accuracy: 0.9895
Confusion Matrix:
[[ 974   14]
 [    7 1005]]
Classification Report:
```

	precision	recall	f1-score	support
0	0.99	0.99	0.99	988
1	0.99	0.99	0.99	1012
accuracy			0.99	2000
macro avg	0.99	0.99	0.99	2000
weighted avg	0.99	0.99	0.99	2000

Fig. 5.2: XG Boost Performance metrics

```

Training Accuracy: 94.74%
Testing Accuracy: 93.90%
Classification Report:
              precision    recall  f1-score   support

     0       0.95         0.93         0.94         988
     1       0.93         0.95         0.94        1012

   accuracy                   0.94         2000
  macro avg                   0.94         0.94         0.94         2000
weighted avg                   0.94         0.94         0.94         2000

Confusion Matrix:
[[919  69]
 [ 53 959]]
Model saved as logistic_regression_model.pkl
Scaler saved as scaler.pkl

```

Fig. 5.3: Logistic Regression Performance metrics

5.2.1 Accuracy

Accuracy is a fundamental metric in evaluating the performance of our phishing detection system. It measures the proportion of correctly classified instances among all instances, providing an overall assessment of our model's correctness in identifying phishing and legitimate websites. A high accuracy score indicates that our algorithm effectively distinguishes between malicious and non-malicious URLs, contributing to reliable cybersecurity measures. However, it's important to interpret accuracy alongside other metrics like precision, recall, and F1 score to gain a comprehensive understanding of our system's performance.

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

5.2.2 Precision

Precision is a crucial metric in evaluating our phishing detection system's performance. It measures the accuracy of positive predictions, specifically the proportion of correctly identified phishing URLs among all URLs predicted as phishing. A high precision score indicates that our system minimizes false positives, effectively identifying genuine phishing threats without wrongly classifying legitimate websites. This metric is essential for maintaining user trust and ensuring that reported phishing incidents are accurate and actionable.

$$\textit{Precision} = \frac{\textit{TruePositive}}{\textit{TruePositive} + \textit{FalsePositive}}$$

5.2.3 Recall

Recall, also known as sensitivity or true positive rate, is a key metric in evaluating our phishing detection system's effectiveness. It measures the proportion of actual phishing URLs that our system correctly identifies as phishing. A high recall score indicates that our system effectively captures most of the phishing URLs present in the dataset, minimizing false negatives. This metric is crucial for ensuring comprehensive coverage in detecting phishing threats, reducing the risk of missed malicious URLs that could potentially harm users.

$$\textit{Recall} = \frac{\textit{TruePositive}}{\textit{TruePositive} + \textit{FalseNegative}}$$

5.2.4 F1 Score

The F1 Score, often referred to as the harmonic mean of precision and recall, provides a balanced assessment of our phishing detection system's performance. It considers both false positives and false negatives, making it a reliable measure of overall accuracy. A high F1 Score indicates a system that achieves both high precision (low false positive rate) and high recall (low false negative rate), striking a balance between correctly identifying phishing URLs and minimizing misclassifications. This metric is valuable for evaluating the robustness and reliability of our system in real-world scenarios, ensuring a comprehensive approach to phishing detection.

$$\text{F1 Score} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

5.2.5 Confusion Matrix

The confusion matrix is a fundamental tool for evaluating the performance of our phishing detection system. It provides a detailed breakdown of true positive (TP), false positive (FP), true negative (TN), and false negative (FN) classifications, enabling us to assess the system's accuracy, precision, recall, and F1 Score. By analyzing the confusion matrix, we can identify specific areas where the system excels and areas that may require improvement. This matrix serves as a cornerstone for fine-tuning our algorithms, enhancing the overall effectiveness of our phishing detection model.

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

Fig. 5.4: Confusion Matrix

5.3 MODEL PERFORMANCE

The model performance of our phishing detection system is crucial for evaluating its effectiveness in real-world scenarios. We assess the performance based on various metrics such as accuracy, precision, recall, F1 Score, and the confusion matrix. Through rigorous testing and validation, we aim to achieve high accuracy rates in correctly identifying phishing URLs while minimizing false positives and false negatives. The model's performance is continuously monitored and optimized to ensure reliable and robust phishing detection capabilities, contributing to a safer online environment for users.

5.3.1 Training Results

The training results of our phishing detection model demonstrate its efficacy in learning from the dataset and making accurate predictions. Through iterative training processes, the model achieves high levels of accuracy, precision, recall, and F1 Score, indicating its capability to distinguish between phishing and legitimate URLs effectively. The training results also include insights from the confusion matrix, revealing the model's ability to correctly classify true positives, true negatives, false positives, and false negatives. These training results form the foundation of our model's performance and guide further optimization efforts to enhance its overall effectiveness in detecting phishing attempts.

5.3.2 Testing Results

The testing results of our phishing detection model validate its robustness and generalization capabilities. During testing, the model accurately classifies new, unseen URLs as either phishing or legitimate with high accuracy, precision, recall, and F1 Score. The confusion matrix from testing provides detailed insights into the model's performance, showcasing its ability to correctly identify true positives, true negatives, false positives, and false negatives in real-world scenarios. These testing results affirm the reliability and effectiveness of our model in detecting phishing attempts across diverse datasets, highlighting its potential for deployment in practical cybersecurity applications.

5.3.3 Cross-Validation Results

Cross-validation results further substantiate the efficacy and reliability of our phishing detection model. Through cross-validation techniques such as k-fold validation, the model demonstrates consistent performance across multiple subsets of the dataset, minimizing overfitting and ensuring robustness. The cross-validation results indicate stable accuracy, precision, recall, and F1 Score metrics, reflecting the model's ability to generalize well to new data. This validation process strengthens our confidence in the model's performance, indicating its suitability for real-world applications and its capacity to maintain high detection accuracy across various datasets and scenarios.

5.4 COMPARATIVE ANALYSIS

The comparative analysis between previous work and our work highlights several key advancements in phishing detection:

- **Dataset Size:** Our work utilized a significantly larger dataset of 549,346 URLs compared to the previous work's 5,000 URLs. This extensive dataset provides a more comprehensive and diverse training environment.
- **Feature Extraction:** Our work employed an extensive set of over 50 features, including advanced indicators like self-redirecting hyperlinks, right-click disabling, and pop-up occurrences. In contrast, previous work relied on a limited set of 8-10 features.
- **Accuracy:** Our model achieved a higher accuracy score of 93.85% compared to the previous work's reported accuracy ranging from 80% to 90%. This improvement reflects the effectiveness of our feature-rich approach.
- **Correlation Analysis:** Unlike previous models, our work conducted a comprehensive correlation analysis among various features. This analysis provides deeper insights into the relationships between different phishing indicators.

In summary, our work demonstrates significant advancements in phishing detection by leveraging a larger dataset, a more extensive feature set, higher accuracy, and in-depth correlation analysis. These improvements signify the robustness and reliability of our approach compared to earlier efforts.

COMPARISON TABLE		
FACTORS	PREVIOUS WORK	OUR WORK
Dataset	5000 URLs for training using PhishTank.	5,49346 URLs for training using Kaggle.
Feature Extraction	8-10 features have been used like IP address in url, prefix or suffix, web traffic.	50+ features have been used like self redirecting hyperlink, right click disable or not, number of pop-ups coming.
Accuracy	As per our research the accuracy score of previous models is 80-90%.	In our model accuracy score is 93.85%.
Correlation	As per as our research concerns no one has yet analyzed the correlation among all the features.	In our work we have provided a data analysis of correlation among the features.

Fig. 5.5: Comparison Table

5.5 LIMITATIONS OF THE CURRENT MODEL

While our model demonstrates commendable performance in phishing detection, several limitations warrant consideration for future enhancements. Firstly, the reliance on static features may limit the model's adaptability to dynamic phishing tactics that evolve over time. Incorporating real-time data sources and dynamic feature extraction mechanisms could enhance the model's resilience against sophisticated phishing attempts.

Another limitation pertains to the inherent bias in the dataset, which may influence the model's generalization capabilities across diverse phishing scenarios. Addressing dataset biases through rigorous sampling techniques and continuous data augmentation strategies could mitigate this limitation and improve the model's robustness.

Additionally, the current model's focus on URL-based features may overlook other vectors of phishing attacks, such as email-based phishing or social engineering tactics. Integrating multi-modal data sources and leveraging advanced techniques like natural language processing (NLP) could enrich the model's detection capabilities and provide a more holistic defense against phishing threats.

Lastly, the interpretability of the model's decisions and the transparency of its decision-making process are areas that require further attention. Enhancing model explainability through techniques like feature importance analysis and model visualization could instill greater trust and facilitate easier adoption by cybersecurity practitioners and stakeholders.

Addressing these limitations would pave the way for a more robust and comprehensive phishing detection framework capable of combating evolving threats in cyberspace effectively.

5.6 POTENTIAL IMPROVEMENTS

To further enhance our phishing detection system, several potential improvements can be explored. Firstly, incorporating machine learning techniques that specialize in anomaly detection, such as autoencoders or Isolation Forests, could bolster the model's ability to identify novel and sophisticated phishing patterns that deviate from known patterns.

Additionally, integrating real-time threat intelligence feeds and leveraging external APIs from cybersecurity organizations could enrich the model's knowledge base and enable proactive threat detection. This would involve continuous monitoring of emerging phishing trends and dynamically updating the model's feature set and detection rules.

Furthermore, exploring ensemble learning approaches, such as model stacking or boosting, could harness the strengths of multiple algorithms and enhance the model's predictive

power and generalization capabilities. This ensemble strategy could also mitigate the risk of overfitting and improve the model's performance on unseen data.

Another avenue for improvement lies in augmenting the dataset with diverse and representative samples, including variations in phishing tactics, target domains, and malicious payloads. This would help address dataset biases and improve the model's robustness across a broader spectrum of phishing scenarios.

Lastly, enhancing the user interface of the detection system with intuitive visualizations, actionable insights, and interactive feedback mechanisms could empower users to better understand and act upon the model's predictions. This user-centric approach would foster trust, usability, and adoption of the system among cybersecurity professionals and end users alike.

Chapter 6

CONCLUSION

In conclusion, our project has established a robust framework for detecting phishing websites using machine learning. The integration of Random Forest and RNN models, supported by an extensive literature survey and frontend development, enhances our system's accuracy and adaptability. Moving forward, the implementation of these algorithms and their integration via APIs will bridge theory with practical application, enabling real-time phishing detection. Additionally, our structured database and collaboration with organizations like APWG underscore our commitment to combating global cyber threats. This project not only advances machine learning in cybersecurity but also fosters a community-driven defense against phishing attacks.

6.1 SUMMARY OF PRESENT WORK

The initial phase of our project has been dedicated to laying a robust foundation, encompassing both the development of the user interface and a comprehensive exploration of existing research. The frontend of our website is now functional, featuring an intuitive input field where users can submit links of websites, they suspect to be fraudulent.

Our research phase has been extensive, drawing insights from reputable journals and articles in the field of phishing detection. This literature review has been instrumental in shaping our approach and finalizing the algorithms we plan to implement. We have carefully selected and defined the machine learning algorithms that will power our system, ensuring a potent combination of accuracy and adaptability.

Looking ahead, our immediate future work involves the implementation of these algorithms using Python. This development phase will bring our envisioned machine learning models to life, incorporating the intricacies identified during our research phase. We plan to seamlessly integrate these algorithms into our website using APIs, enabling users to experience real-time phishing detection capabilities.

As a pivotal step, we will establish a database to systematically store information on identified phishing websites. This database will not only serve as a repository for our internal use but will also facilitate the submission of reports to organizations dedicated to combating phishing, such as the Anti-Phishing Working Group (APWG). The integration of a database ensures data persistence and allows for future analyses and refinements.

Link to the hostedWebsite- <https://detect-phishing-sites.vercel.app/>

Link to the Github Repository- <https://detect-phishing-sites.vercel.app/>

6.2 FUTURE WORKS

Moving forward, our roadmap includes the following key milestones:

API Integration: Seamlessly link the algorithms to the website through APIs, ensuring a user-friendly and responsive interface.

Database Creation: Establish a robust database structure to store information on identified phishing websites, enhancing data management and analysis capabilities.

Submission to APWG: Implement a systematic process to submit reports on identified phishing websites to organizations like APWG, contributing to the broader cybersecurity community.

User Feedback Integration: Incorporate mechanisms for user feedback to continuously improve and refine the performance of our phishing detection system.

Web Extension Development: Develop a web extension add-on to extend the reach of our phishing detection tool, providing users with added convenience and accessibility.

Through these planned future works, we aim to transform our research and planning into a functional and impactful tool. By marrying technological innovation with community engagement, we strive to create a comprehensive solution that not only detects phishing websites effectively but actively contributes to the global fight against cyber threats.

Chapter 7

REFERENCES

The Published journals and articles that have helped us in gathering information regarding the past works in this topic are enlisted as follows-

1. Gökem Giray, Bedir Tekinerdogan, Sandeep Kumar & Suyash Shukla, “Applications of deep learning for phishing detection: a systematic literature review” in *Knowledge and Information Systems* 23rd May 2022.
2. Eduardo Benavides, Walter Fuertes, Sandra Sanchez & Manuel Sanchez, “Classification of Phishing Attack Solutions by Employing Deep Learning Techniques: A Systematic Literature Review” in *Part of the Smart Innovation, Systems and Technologies book series (SIST, volume 152)* 14th June 2019.
3. Issa Qabajeh, Fadi Thabtah, Francisco Chiclana, “A recent review of conventional vs. automated cybersecurity anti-phishing techniques” in *Centre for Computational Intelligence, De Montfort University, Leicester, UK* 15 June 2018.
4. Ashit Kumar Dutta, “Detecting phishing websites using machine learning technique” in *Zhihan Lv, Qingdao University, China* October 11, 2021.

The online web-links to the above references are as follows-

- <https://link.springer.com/article/10.1007/s10115-022-01672-x>
- https://link.springer.com/chapter/10.1007/978-981-13-9155-2_5
- <https://www.sciencedirect.com/science/article/abs/pii/S1574013717302010?via%3Dihub>
- <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0258361>