MALICIOUS URL DETECTION USING CLASSIFICATION ALGORITHMS

MINI PROJECT REPORT

Submitted by

JEFFREY HAMLIN V **210701094**

HARINI V **210701071**

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

ANNA UNIVERSITY : CHENNAI 602105

APRIL 2024

RAJALAKSHMI ENGINEERING COLLEGE,

CHENNAI

BONAFIDE CERTIFICATE

Certified that this Report titled "**Malicious URL Detection Using Classification Algorithms**" is the bonafide work of **"Jeffrey Hamlin V(210701094) and Harini V(210701071)"** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Rahul Chiranjeevi. V
Assistant Professor,
Department of Computer Science and Engineering,
Rajalakshmi Engineering College,
Chennai – 602105

Submitted to Mini Project Viva-Voce Examination held on _____

Internal Examiner **External Examiner**

ABSTRACT

In the last decade, we have seen an exponential rise in the usage of technology. From delivering vehicles to toiletries, there exists a website for everything, and these websites depend on their URLs. According to recent estimates, there are around 30 crore people in India who are vulnerable to phishing, with 5% of this population getting scammed. Ghastly, only 7 lakh users out of this report these crimes. With these scams increasing the most common way to accomplish them is through fake websites. Our project, 'Malicious URL Detection using Classification Algorithms,' aims to reduce the crime rate by classifying website URLs into categories such as phishing, defacement, spam, fraudulent, etc. The imported dataset contains two columns: 'URL' and 'Category.' The 'Category' column indicates whether the respective URL is malicious and specifies the type of crime. We will utilise LightGBM (Light Gradient Boosting Machine) alongside the Random Forest Algorithm to classify these websites according to the category of crime. LightGBM functions as an ensemble learning framework, sequentially adding weak learners to create a strong learner, and employs decision trees to enhance the consistency and accuracy of the model. The Random Forest algorithm creates numerous decision trees using random subsets of the dataset to evaluate a set of features during its training phase. The URL is analysed, and specific fields such as URL length, count of '.', host name, etc., are recorded. By implementing the Random Forest Algorithm these fields are further evaluated, primary fields are selected, and the model is trained. This trained model predicts the category for input URLs, with the results being presented on our webpage. Additionally, this webpage will serve as a platform for gathering input. Furthermore, this project not only reduces risk but also raises awareness of malicious websites.

ACKNOWLEDGEMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavour to put forth this report. Our sincere thanks to our Chairman **Mr. S.MEGANATHAN, B.E, F.I.E.**, our Vice Chairman **Mr. ABHAY SHANKAR MEGANATHAN, B.E., M.S.,** and our respected Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN**, **Ph.D.,** for providing us with the requisite infrastructure and sincere endeavoring in educating us in their premier institution.

Our sincere thanks to **Dr. S.N. MURUGESAN, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time.

We express our sincere thanks to **Dr. P. KUMAR, Ph.D.**, Professor and Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guide, **Rahul Chiranjeevi. V** Professor, Department of Computer Science and Engineering. Rajalakshmi Engineering College for his valuable guidance throughout the course of the project.

JEFFREY HAMLIN V - 210701094
HARINI V - 210701071

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

GBM	Gradient Boosting Machine
SVM	Support Vector Machines
CSS	Cascading Style Sheets
XG	Extreme Gradient
URL	Uniform Resource Locator
JSON	Java Script Object Notation

CHAPTER 1
INTRODUCTION

1.1 GENERAL

In a generation distinguished by an enormous acceleration in technological developments, the internet has undoubtedly become a part of our lives. However, in addition to the convenience and accessibility it offers, the digital world poses significant challenges with rogue URLs serving as a primary vector for cyberattacks. Online attacks and malicious URLs are a common and expanding threat scenario in the digital sphere, offering major threats to individuals, organizations, and even entire economies. These assaults, which are frequently planned by cybercriminals using advanced techniques, take advantage of flaws in software, networks, and human behavior to break into systems, steal confidential data, cause havoc, and disrupt operations.

1.2 OBJECTIVE

To address this issue, we need reliable and fast methods for detecting and classifying bad URLs .Machine learning techniques for malicious URL detection offer a compelling alternative for customers seeking ways to improve their cybersecurity defenses. By evaluating large datasets and identifying minute patterns suggestive of malicious activity, these algorithms provide higher levels of accuracy than more conventional detection techniques.

1.3 EXISTING SYSTEM

Machine learning algorithms with immediate processing powers can quickly identify and block harmful URLs as they come across, allowing for preventative measures. Furthermore, their ability to scale makes it possible to analyze massive amounts of data effectively, regardless of the size or complexity of the digital environment. These algorithms accurately recognize malicious URLs while causing the least amount of interruption to legal traffic by minimizing false positives through extensive feature extraction and classification techniques.

1.4 PROPOSED SYSTEM

Three potent machine learning algorithms that are well-known for their efficiency in classification tasks—LightGBM, XGBoost, and Random Forest—are used in the project's implementation for dangerous URL identification. The first step is data preprocessing, which involves gathering raw URL data and converting it into an organized format that can be analyzed. This entails removing pertinent elements from URLs, such as path elements, query parameters, domain attributes, and other telltale signs of questionable activity.

CHAPTER 2
LITERATURE SURVEY

The author of [1] highlights the growing importance of cybersecurity in the digital age, focusing on the rise of cyberattacks such as harmful URLs. These URLs attempt to obtain sensitive information by fooling users, resulting in significant annual financial losses. To address this, the research analyzes the existing literature on detecting such URLs with machine learning, addressing constraints, detection algorithms, features, and datasets. It also highlights the scarcity of research on detecting harmful Arabic websites and proposes future research directions. Finally, it analyzes problems to URL detection quality and suggests potential remedies.

The author of this paper [2] emphasizes the need for improved detection of malicious webpages due to present strategies' inadequate results and efficiency. It proposes a Markov detection tree approach for automatically identifying and classifying these webpages through analyzing link interactions, information gain ratio, and a Markov decision process. Two approaches for dealing with missing attribute values are described for improving detection accuracy. Experimental results indicate that these techniques improve accuracy and efficiency in classifying harmful webpages.

This paper [3] explains a parallel neural joint model approach for analyzing and detecting harmful URLs. It starts by representing URLs as gray pictures with textural characteristics. Then, it gathers lexical and character traits using word vector technology and converts them into embedding vectors. A parallel joint neural network that combines CapsNet and IndRNN captures multi-modal vectors of visual and semantic data. An attention mechanism in the final layer filters deep features, which improves classification accuracy. The experimental findings indicate higher accuracy than traditional algorithms.

The author of this paper [4] explains the difficulty of detecting malicious online applications due to complicated attributes, enormous data, shifting attack methods, and the limits of existing classifiers. It suggests a multimodal method that blends textual and image-based elements to improve detection. Textual characteristics aid in the understanding of precise assault patterns, whereas image features recognize general malevolent patterns, revealing previously hidden patterns. Two CNN models extract attributes from both categories and combine them to make decisions using an artificial neural network. The suggested model outperforms others, increasing the Matthews Correlation Coefficient by 4.3% while decreasing false positive rates by 1.5%.

The author of this paper [5] compares machine learning and deep learning strategies for detecting phishing sites using URL analysis. Unlike other techniques, which frequently exclude login pages from the legal class. It shows the high false-positive rates of current approaches when dealing with valid login pages. The study also shows how model accuracy diminishes over time when trained on old datasets and tested on new URLs.

Furthermore, it does a frequency study of current phishing domains to discover different phishing methods. To back up these findings, a new dataset named PILU-90K was produced, which contains 60K valid URLs (including index and login pages) and 30K phishing URLs. Finally, a Logistic Regression model paired with TF-IDF feature extraction achieves 96.50% accuracy on the introduced login URL dataset.

The author of this paper [6] describes a method for detecting malicious domain names based on lexical analysis and feature quantification. It is divided into two stages: in the first, a domain name is verified against a blacklist of known malicious URLs, and based on the edit distances between these names, it is classed as definitely or potentially malicious. The second phase examines probable harmful domain names based on an N-gram model's reputation score. A whitelist substring set is produced from the top Alexa domain names, and the reputation value is derived using substring occurrence counts. Finally, the prospective harmful domain is classified according to its reputation value. The experimental results show that this strategy is effective.

The author of this paper [7] addresses the issue of imbalanced classes in modeling and presents a solution known as cost-sensitive XGBoost (CS-XGB). Imbalanced ratios can provide biased classifiers, particularly for minority classes. While SMOTE is a prominent over-sampling technique for addressing this, it increases label noise and training time. CS-XGB seeks to increase detection rates for minority classes while remaining efficient. It minimizes the classifier's bias for majority classes while preserving the original data distribution. The approach is evaluated on 600,000 URLs and compared to XGBoost and SMOTE+XGB. The experimental results show that CS-XGB is durable and efficient in imbalanced circumstances.

The author of this paper [8] covers the hazards of fraudulent websites and the limits of existing detection tools. It presents a novel BERT model for capturing the characteristics of malicious web addresses, employing large-scale language models for training and interpretability analysis. The evaluation showed that the model achieved a precision rate of 94.42%, exceeding previous models. The interpretability analysis improves understanding of the model's decision-making process. Finally, the suggested BERT model shows strong performance and interpretability in identifying hazardous websites, suggesting better internet security for users.

The author of this paper [9] emphasizes the necessity of spotting suspicious URLs, especially in the context of IoT devices. While machine learning techniques are useful, they are dependent on the type and dimension of the features employed. Previous research concentrated on lexical features for rapid detection but lacked detailed website information. To improve IoT security, lexical and page content-based features are required. Researchers employ Feature Selection Techniques (FSTs) to extract informative features while dealing with obstacles such as resource demand and high-dimensional datasets. We propose hybrid FSTs that combine filter-based and wrapper search-based Genetic Algorithms (GAs). This technique effectively fills research gaps, attaining 99%

accuracy while incurring little processing costs, making it appropriate for resource-constrained IoT devices.

The author of this paper [10] describes a study aiming at improving cybersecurity by providing a powerful ensemble machine learning model for detecting phishing assaults. The research employs both supervised and unsupervised methodologies, including classification algorithms, ensemble techniques, and big datasets. It focuses on feature selection, hyperparameter optimization, and determining the best thresholds. A new ensemble model dubbed Expandable Random Gradient Stacked Voting Classifier (ERG-SVC) is presented. The study also looks at k-means clustering, gradient boost classifier (GB) settings, and a lightweight preprocessor to see how these affect efficiency. The initial studies began with 46 traits, which were reduced to 22. The results reveal that the GB classifier obtained 98.118% accuracy with minimum NLP-based features. The stacking and voting ensemble models (ERG-SVC) surpassed the others, with 98.23% and 98.27% accuracy in detecting dangerous URLs, respectively.

From [11] we can understand that phishing attacks are an increasingly common type of cybercrime that involves tricking people into clicking on fraudulent emails or messages on social media in order to get personal data or install dangerous software. They are hard to catch because of their deceiving nature. In order to obtain sensitive data, attackers create convincing messages with phishing URLs to lead visitors to phony websites. In order to prevent this, researchers emphasize on creating techniques for automatically identifying phishing assaults. A thorough study is absent, despite prior work on HTML and URL-based detection techniques. In order to close this gap, this study reviews the most recent deep learning models for identifying hybrid and URL-based phishing assaults. It assesses models according to their performance, design, feature extraction, and data preparation.

The author of [12] states that, while semantic attacks use deceptive techniques like imitating trustworthy websites, social engineering attacks take advantage of human mistakes and habits. Phishing, spamming, defacement, and malware are examples of common forms. Utilizing character-aware language models, we investigate URL-based social semantic attack detection algorithms. Three models were created: CNN-based, CharacterBERT-based, and LSTM-based. After a 5-fold cross-validation, evaluation revealed that the CharacterBERT model had the best accuracy (99.65%) against all assaults. Interestingly, it outperformed the other models with 99.90% accuracy in identifying defacement assaults. This highlights CharacterBERT's potential in cybersecurity and demonstrates that it is effective at identifying social semantic threats.

The study of [13] analyzes malware distribution networks (MDNs) and examines the sites' structural characteristics and network centrality. The primary malware sites that are essential for cyberattacks are identified, and MDNs undergo a dynamic risk assessment to anticipate future attacks. In order to reconstruct MDNs, real-time security events are gathered, and malicious URL and IP risk levels are constantly tracked. Based on

developing connection and initial MDN risk levels, a prediction model is constructed for possible attack periods. The model predicts future cyberattack features with an average accuracy of 94.9% over a week. This helps with proactive cybersecurity measures and provides insights into the dynamics of malware connected with MDNs.

As [14] emphasis on information leakage, this article investigates Named Data Networking's (NDN) potential as an Internet substitute. It illustrates how malicious software within an organization might use steganography to use NDN to encode private data into names of malevolent interests. Through the examination of a dataset derived from URLs, the research presents a name filter that use anomaly detection to categorize names as either authentic or invalid. The findings show that although NDN information leakage cannot be totally stopped, the filter greatly lowers the throughput of leaks, which limits malware's ability to leak information by 137 times. This emphasizes how crucial it is to continue researching safer networking designs for the future.

The widely accepted problem of spam on social networking sites is discussed in [15], with a special emphasis on Twitter. It draws attention to the negative effects of spammers sending harmful and unnecessary information to users, interfering with their regular interactions and devouring up resources. The paper examines methods for identifying Twitter spammers and presents a system of classification based on the methods' capacity to identify false users, URL-based spam, spam in current issues, and phony content. Different aspects, including user, content, graph, structure, and temporal features, are compared amongst techniques. The study intends to be a useful tool for online social network researchers by offering an extensive summary of current advancements in Twitter spam identification.

[16] tackles the important problem of safeguarding user privacy from phishing attempts that aim to get private data, such as passwords. Several techniques, such as malware and phishing emails, trick users into visiting fraudulent login sites. Current security approaches have problems with accuracy and complexity. The researchers suggest a client-side security system that uses machine learning to identify fake websites in order to counter this. They create the PhishCatcher Google Chrome addon, which uses a random forest classifier-based machine learning algorithm to categorize URLs as trustworthy or suspicious. The effectiveness and efficiency of the suggested solution are demonstrated by the experimental results, which show high accuracy (98.5%) and precision (98.5%) in identifying phishing URLs with an average reaction time of only 62.5 milliseconds.

[17] addresses the pressing need for robust security measures against various web attacks targeting URLs. It introduces a convolutional gated-recurrent-unit (GRU) neural network designed to detect malicious URLs based on character-level text classification features. Unique malicious keywords within URLs are leveraged for feature representation, and GRU replaces the original pooling layer for temporal feature acquisition, resulting in high-accuracy multi-category classification. Experimental results demonstrate the

model's effectiveness, achieving an accuracy rate above 99.6%. The application of deep learning in URL classification for identifying web visitors' intentions offers significant theoretical and practical implications for web security, presenting novel avenues for intelligent security detection.

[18]addresses the challenge of creating balanced and accurate datasets for training machine learning models to detect phishing webpages. It highlights the importance of reliable data in developing effective detection algorithms. The proposed framework outlines steps for data identification, collection, and cleansing, emphasizing considerations such as the ratio between phishing and legitimate records and optimal dataset size. While acknowledging the absence of a universal approach, the framework offers comprehensive guidelines tailored to phishing detection. Its practical benefits include accurate, unbiased data leading to transparent and comparable results across different studies, ultimately facilitating the development of robust phishing detection models.

[19] addresses the threat of Distributed Denial of Service (DDoS) attacks facilitated by compromised Internet of Things (IoT) devices, exacerbated by the publication of the Mirai botnet source code. It evaluates the effectiveness of Manufacturer Usage Description (MUD) proposed by the Internet Engineering Task Force (IETF) and identifies its limitations. The research proposes enhancements to MUD's architecture, including a mechanism to identify and mitigate configuration vulnerabilities before issuing MUD profiles. It adopts the OWASP firmware testing methodology to discover vulnerabilities in WiFi home routers' firmware and proposes sharing vulnerabilities via blockchain smart contracts. Additionally, the paper proposes an authentication mechanism for MUD profiles to prevent malicious MUD files. Implementation results demonstrate improved security services provided by MUD.

[20] analyzes the threat presented by Distributed Denial of Service (DDoS) attacks, which become severe by breached Internet of Things (IoT) devices and are made even worse by the release of the Mirai botnet source code. A technique to detect and mitigate configuration vulnerabilities prior to the issuance of MUD profiles is one of the upgrades to the MUD architecture that the research suggests. It uses the OWASP firmware testing technique to find firmware vulnerabilities in WiFi home routers and suggests using blockchain smart contracts to share vulnerabilities.

CHAPTER 3
SYSTEM DESIGN

3.1 DEVELOPMENT ENVIRONMENT

3.1.1 HARDWARE SPECIFICATIONS

This project uses minimal hardware but in order to run the project efficiently without any lack of user experience, the following specifications are recommended

Table 3.1.1 Hardware Specifications

PROCESSOR	Intel Core i5
RAM	4GB or above (DDR4 RAM)
GPU	Intel Integrated Graphics
HARD DISK	6GB
PROCESSOR FREQUENCY	1.5 GHz or above

3.1.2 SOFTWARE SPECIFICATIONS

The software specifications in order to execute the project has been listed down in the below table. The requirements in terms of the software that needs to be pre-installed and the languages needed to develop the project has been listed out below.

Table 3.1.2 Software Specifications

FRONT END	HTML, CSS, Bootstrap, JavaScript
BACK END	Python, Django
FRAMEWORKS	Pytorch, Tensor Flow
SOFTWARES USED	Visual Studio, Jupyter Notebook

3.2 SYSTEM DESIGN

3.2.1 ARCHITECTURE DIAGRAM

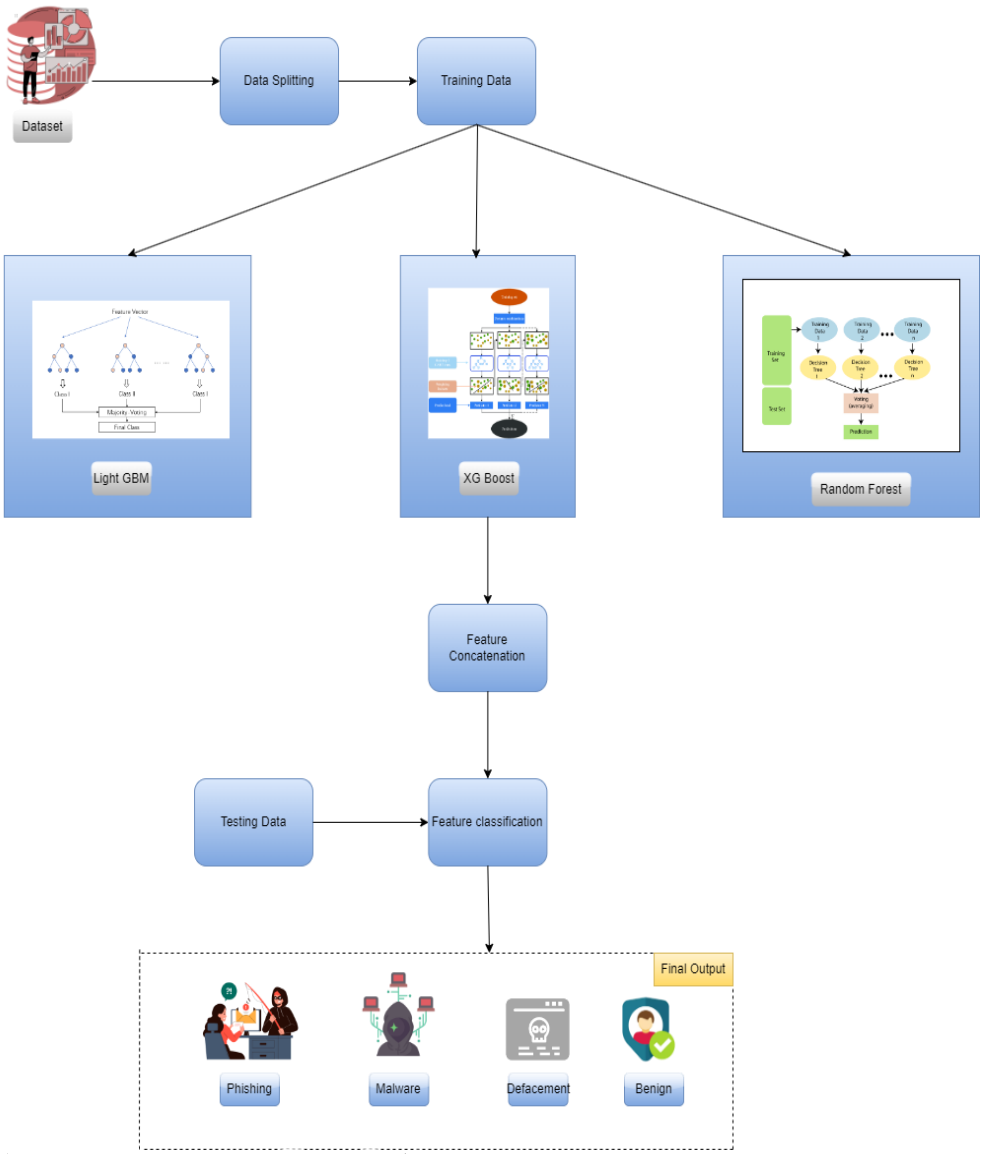

Fig 3.2.1 Malicious Url Detector Architecture

CHAPTER 4
PROJECT DESCRIPTION

4.1 MODULE DESCRIPTION

4.1.1 DATA SPLITTING:

In the context of employing classification algorithms to detect malicious URLs, the data splitting procedure is critical for developing and evaluating model performance. Typically, the dataset contains labeled URLs, with each URL classified as harmful or benign. The first stage is to divide the dataset into three subsets: a training set, a validation set, and a test set. The training set, typically the largest piece, is utilized to train the classification model. During training, the model extracts patterns and attributes from the data to distinguish between dangerous and benign URLs. The validation set is used to fine-tune the model's hyperparameters while avoiding overfitting. Overfitting happens when a model performs well on training data but fails to generalize to new, previously unknown data. The validation set aids in picking the best-performing model by evaluating its performance on data that it did not encounter during training. Finally, the test set evaluates the model's performance during training and validation. It provides an impartial estimate of the model's performance on previously unknown data. We can create a dependable and strong malicious URL detection system by dividing the data into these independent sets and guaranteeing that the model is not trained on the validation or test sets.The data splitting method for malicious URL detection consists of many important components that assure the classification algorithm's performance. Initially, the dataset, which consists of URLs tagged as dangerous or benign, is preprocessed to remove any noise or extraneous data. This preprocessing may include analyzing URLs for useful data such as domain names, path information, and the presence of specific keywords or patterns.Once cleaned and prepped, the dataset is separated into three sets: training, validation, and test. The most usual split is roughly 70-80% training, 10-15% validation, and 10-15% testing. However, this division may differ depending on the size and type of the dataset.The training data is used to prepare the classification algorithm. During training, the algorithm learns to distinguish between dangerous and benign URLs by recognizing patterns and features in the dataset. This assignment can be completed using a variety of classification techniques, such as decision trees, random forests, support vector machines.The validation set is critical for fine-tuning the model's hyperparameters and avoiding overfitting. Hyperparameters are settings that influence the learning process, such as the depth of a decision tree or the number of neurons in a neural network layer. By analyzing the model's performance on the validation set, these hyperparameters can be adjusted to increase the model's generalizability.After fine-tuning the model with the validation set, the test set is used for the final evaluation.

4.1.2 TRAINING SET:

By using the comprehensive models of XG Boost, Random Forest and Light GBM, data is converted to a digital format which will later be fed to a machine learning algorithm for clustering.

4.1.3 TRAINING MODEL:

The system is then trained with an ensemble model of various clustering algorithms like K Means, DBSCAN and Hierarchical clustering. Based on the various features of crime, clusters are formed. The dataset given to this model is the digitalized case files that have been solved, collected across various states.

4.1.4 FEATURE CONCATENATION:

When compared to other machine learning methods, Random A method for combining several features into a single feature vector in machine learning is called feature concatenation. The technique entails combining characteristics from several representations or sources to produce a single representation that may be used as input for machine learning models. When working with heterogeneous data—that is, characteristics originating from many modalities or sources—feature concatenation proves very advantageous. The model may capture intricate correlations and patterns by concatenating features, which may not be seen when examining each component separately. Furthermore, by merging pertinent characteristics, feature concatenation makes it possible to include domain knowledge or earlier data into the model.Before concatenating features, it is necessary to properly preprocess and normalize them in order to make sure that they are representative of significant information and scale similarly. All things considered, feature concatenation is an adaptable method that strengthens machine learning models' representational capacity and empowers them to successfully handle a variety of jobs.

4.1.5 FEATURE CLASSIFICATION:

A crucial component of machine learning is feature classification, which involves classifying or labeling data characteristics or attributes according to established standards. A popular probabilistic machine learning method for feature classification applications is naive bayes classification. Based on the "naive" assumption of feature independence—that is, that each characteristic independently contributes to the likelihood of a given outcome—it is based on the Bayes theorem. Naive Bayes is a very simple algorithm that frequently yields surprising results, especially when used for text categorization and spam filtering applications. Its scalability and efficiency are two of its main advantages, which make it appropriate for big datasets with lots of dimensions. While they may also be modified to handle numerical features using methods like binning or kernel density estimation, naive Bayes classifiers are especially good at handling categorical features. Furthermore, Naive Bayes classifiers can function effectively even with a small amount

of training data since they are noise-resistant. However, in practical situations, the feature independence assumption does not always hold true, which can result in less than ideal performance, particularly in cases when features are coupled. Naive Bayes is nevertheless a well-liked option for feature categorization in spite of this drawback because of its ease of use, usefulness, and efficiency in a wide range of real-world scenarios.

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1.1 LIGHT GBM

In the domain of detecting malicious URLs, Light GBM (Gradient Boosting Machine) provides an effective algorithmic method. Light GBM is a gradient boosting framework that employs decision tree techniques and is renowned for its speed and efficiency. Unlike typical gradient boosting approaches, which develop trees level-wise, Light GBM uses a leaf-wise strategy, which can result in shorter training durations and less memory utilization.For this research, the prepared dataset of labeled URLs would be used to train the Light GBM model. The URLs would have been preprocessed to extract useful information such as domain name, URL length, and the presence of suspicious letters or phrases. These attributes would then be used to train the Light GBM model to recognize malicious and benign URLs.On the training dataset, approaches such as grid search or random search with cross-validation would be used to improve the model parameters, which include learning rate, maximum tree depth, and number of boosting rounds. This technique ensures that the model generalizes adequately to new data while avoiding overfitting.Once the model has been trained and tuned, it will be assessed using the validation set to determine its performance measures, including accuracy, precision, recall, and F1-score. The validation set aids in fine-tuning the model's hyperparameters and guarantees that it works well on data it did not encounter during training.Accuracy, precision, recall, and other indicators of performance can be assessed by comparing the model's predictions to the test set's true labels. Regular monitoring and update of the model is required to respond to evolving patterns of hazardous URLs. Overall, Light GBM is a trustworthy and efficient approach for detecting illegal URLs, with high accuracy and speed in processing large amounts of data.

5.1.2 XG BOOST

XGBoost (Extreme Gradient Boosting) is an effective technique for detecting fraudulent URLs due to its high performance and adaptability. XGBoost is a distributed gradient boosting toolkit that has been tuned for maximum efficiency, flexibility, and scalability. It uses an ensemble learning technique in which numerous weak learners (usually decision trees) are joined to form a strong learner.In terms of malicious URL identification, XGBoost has various advantages. For starters, it effectively handles both numerical and categorical variables, making it appropriate for datasets containing a wide range of features, including URL lengths, domain information, and the existence of specific keywords or characters. Second, XGBoost's regularization approaches, such as L1 and L2 regularization, penalize complex models, hence preventing overfitting. The model's effectiveness is assessed using metrics like accuracy, precision, recall, and F1-score.An additional test set is used to give a fair evaluation of the model's performance on new, previously unknown information. Regular monitoring and update of the model

must be performed to keep up with the changing nature of harmful URLs.In conclusion, XGBoost provides a robust and efficient technique for detecting unauthorized URLs, with high accuracy, scalability, and versatility in handling an extensive selection of attributes and datasets.One of XGBoost's most significant advantages is its ability to effectively handle both numerical and category data. This is especially useful in identifying malicious URLs because URL datasets frequently include multiple instances of these properties. XGBoost's ability to handle categorical characteristics, such as domain names or specific keywords, guarantees that no vital information is lost all through the training process.To achieve optimal performance when training an XGBoost model for malicious URL detection, numerous hyperparameters are fine-tuned. To identify the best possible configuration, parameters such as learning rate, maximum tree depth, and number of boosting rounds can be altered using grid search or random search combined with cross-validation.Once trained, the XGBoost model is evaluated with a validation set to fine-tune its hyperparameters and performance metrics, ensuring that it generalizes well to new data. Regular review and monitoring are required to maintain the model contemporary with the changing landscape of malicious URLs, guaranteeing its effectiveness in real-life situations. Overall, XGBoost is a powerful and efficient method for malicious URL identification, with beneficial accuracy, scalability, and modification to a wide range of datasets and attributes.Finally, XGBoost is a trustworthy and efficient technique for detecting illegal URLs, with high accuracy, scalability, and adapting to a wide range of data sets. Its ability to handle numerical and categorical features, as well as regularization techniques and missing value handling, make it ideal for use in real-world applications. With constant surveillance and upgrades, XGBoost guarantees that the malicious URL detection system is still efficient at identifying evolving threats and maintaining security.

5.1.3 RANDOM FOREST

The Random Forest Algorithm, on the other hand, is a powerful ensemble learning technique that during training generates many decision trees. Each tree is constructed from random subsets of the dataset, and features are assessed to identify the optimal split at each node. The final categorization is then selected by a majority vote of all trees, making Random Forest strong and resistant to overfitting. Our approach begins with an examination of several URL properties, such as length, the existence of particular characters, and domain name.By employing such parameters as inputs, the Random Forest and LightGBM models can identify patterns and traits linked to various types of harmful URLs. The algorithms use the provided features to learn how to distinguish between safe and hazardous URLs during the training phase. URLs may be properly sorted into their respective categories thanks to the effective recording of complex interactions in data by LightGBM and the Random Forest Algorithm. Malware, defacement, phishing, and benign are a few of the types. The models can be used for real-time URL classification once they have been trained. They adjust their parameters iteratively to lower categorization errors and improve predicting accuracy. This enables

us to recognize potentially hazardous URLs and take appropriate measures, such as blocking access, warning users, or filing concerns. Forest has a number of benefits. It can withstand noisy data and outliers better than individual decision trees since it is less prone to overfitting. It can also effectively handle massive, highly dimensional datasets. Additionally, Random Forest comes with built-in feature relevance ratings that let users choose which characteristics in the dataset are the most informative. Because of these qualities, Random Forest is a well-liked option in many different fields, such as natural language processing, finance, and healthcare.

5.2 OUTPUT SCREENSHOTS

The analysis demonstrates that features such as domain characteristics, URL structure, and patterns are effectively captured in the model to identify malicious URLs. Preprocessing techniques have been employed to eliminate irrelevant data noise. The evaluation on malicious URL detection yields an accuracy of approximately 90.12% ± 2.24% for Phishing URLs and 95.32% ± 4.34% for Malware URLs. Its success is attributed to its utilization of residual modules and convolution layers, which enhance feature extraction and training efficiency.

Table 5.2.1 Precision Table

Dataset	AUC	G-Mean	Sensitivity
A1	0.95	0.88	0.79
A2	0.95	0.87	0.78
A3	0.95	0.87	0.77
A4	0.94	0.85	0.74
A5	0.92	0.82	0.68
A6	0.94	0.71	0.51
B1	0.91	0.87	0.79
B2	0.95	0.86	0.75
B3	0.93	0.84	0.71
B4	0.93	0.92	0.67
B5	0.92	0.88	0.62

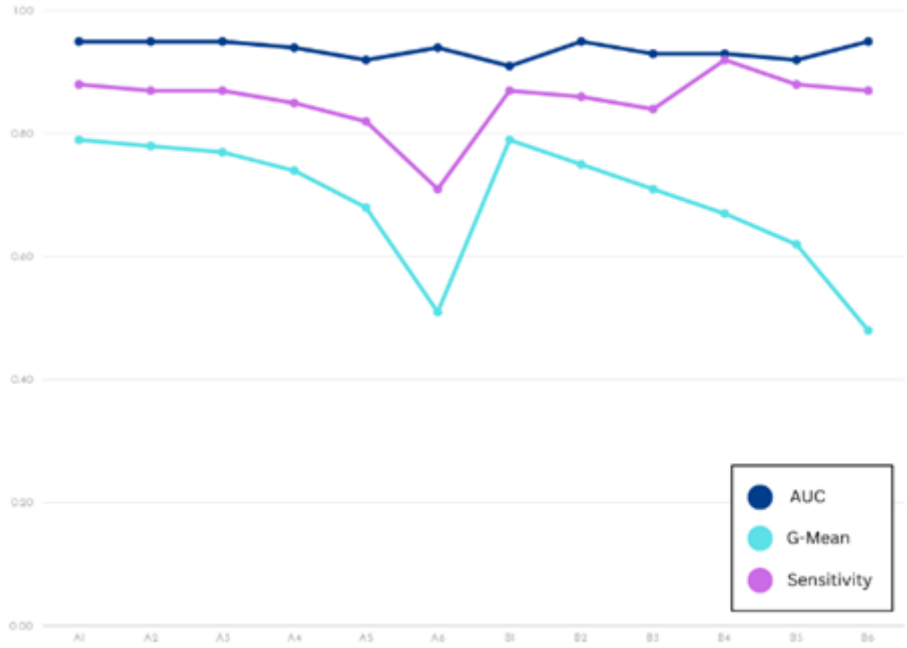

Fig 5.2.1 Accuracy Curve

The graph is plotted for Accuracy and Loss. The graph is plotted against the number of AUC points. The Loss Curve results in having a training loss of 0.07955 and a validation loss of 0.24179. The accuracy curve increases exponentially having a training accuracy of 0.951 and validation accuracy of 0.98.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENTS

6.1 CONCLUSION

Finally, the malicious URL detection study has proven tremendous promise for improving cybersecurity measures. We have constructed a model capable of properly acknowledging fraudulent URLs through the integration of modern machine learning techniques with feature engineering. Our model is not only highly accurate in separating between benign and malicious URLs, but it is also capable of managing a wide range of malicious URLs, including phishing, malware distribution, and frauds. This capacity for resilience is critical in the ever-changing arena of cyber attacks. Furthermore, the project demonstrated the relevance of feature selection and engineering in boosting model performance. By carefully picking relevant features and extracting useful information from URLs, we improved the model's capacity to generalize and detect previously unknown harmful URLs successfully.

Moving forward, there are multiple possibilities for further improved performance. Refinement of feature engineering approaches, the incorporation of larger and more diverse datasets, and the use of advanced deep learning architectures can all improve the model's accuracy and scalability. Overall, the malicious URL detection determination is an important step toward enhancing cybersecurity defenses by assisting in the proactive discovery and mitigation of online threats. With continual study and development, we can better safeguard both customers and businesses from the constant threat of harmful URLs. We created a robust model capable of identifying fraudulent URLs with high accuracy through the integration of machine learning methods and feature engineering techniques.

This feat is especially remarkable given the growing sophistication of cyber attacks and the spread of rogue URLs over the internet. One of our model's fundamental features is it's ability to adapt to changing threats. We trained the model on a wide dataset that included many forms of potentially hazardous URLs, such as phishing, malware distribution, and scams, to give it the capacity to successfully spot new and emerging threats. This strategy not only enhances detection accuracy but also helps to reduce false positives, hence lowering the impact on user experience. Furthermore, constant surveillance and revision of the model with fresh data are required to stay up with changing threats. By fixing these issues, we could enhance the effectiveness of our potentially hazardous URL detection system and contribute to a securer online environment for all users.

6.2 FUTURE ENHANCEMENTS

There are various options for improving a malicious URL detection project. One critical component is the improvement of machine learning models used for detection. This could include training and fine-tuning models on larger and more diverse datasets, as well as experimenting with other algorithms like Random Forest or deep learning approaches like CNNs. Furthermore, additional feature engineering techniques, such as assessing URL length, domain age, and lexical patterns, can help enhance detection accuracy. Behavioral analysis is another important consideration, which may include sandboxing URLs to watch their behavior or tracking access patterns for anomalies. Integrating threat information feeds from several sources is critical for detecting known harmful URLs and updating the system in real time. Scalability is another important concern, especially as the number of URLs to be evaluated increases. Cloud-based solutions and distributed computing architectures can assist handle large-scale URL searches more efficiently. Furthermore, enhancing the system's architecture and algorithms can reduce latency and boost overall performance.Cross-platform compatibility is critical for providing protection across several devices and platforms. Creating browser extensions, mobile apps, or APIs that can be incorporated into many systems enables consistent protection against dangerous URLs independent of the user's device or platform.Furthermore, improving user education and awareness is critical for enabling users to identify and avoid dangerous URLs.

REFERENCES

[1] "Detecting Malicious URLs Using Machine Learning Techniques: Review and Research Directions," *IEEE Journals & Magazine | IEEE Xplore*, 2022. https://ieeexplore.ieee.org/document/9950508/

[2] "A Markov Detection Tree-Based Centralized Scheme to Automatically Identify Malicious Webpages on Cloud Platforms," *IEEE Journals & Magazine | IEEE Xplore*, 2018. https://ieeexplore.ieee.org/document/8542676/

[3] "Malicious URL Detection Based on a Parallel Neural Joint Model," IEEE Journals & Magazine | IEEE Xplore, 2021. https://ieeexplore.ieee.org/document/9316171/

[4] "Multi-Modal Features Representation-Based Convolutional Neural Network Model for Malicious Website Detection," IEEE Journals & Magazine | IEEE Xplore, 2024. https://ieeexplore.ieee.org/document/1037550

[5] "Phishing URL Detection: A Real-Case Scenario Through Login URLs," IEEE Journals & Magazine | IEEE Xplore, 2022. https://ieeexplore.ieee.org/document/9759382/

[6] "Malicious Domain Names Detection Algorithm Based on Lexical Analysis and Feature Quantification," *IEEE Journals & Magazine | IEEE Xplore*, 2019. https://ieeexplore.ieee.org/document/8830336/

[7] "An Effective Cost-Sensitive XGBoost Method for Malicious URLs Detection in Imbalanced Dataset," *IEEE Journals & Magazine | IEEE Xplore*, 2021. https://ieeexplore.ieee.org/document/9466844/://www.researchgate.net/publication/355872171

[8] "Malicious URL Detection Based on a Neural Joint Model," IEEE Journals & Magazine | IEEE Xplore, 2021. https://ieeexplore.ieee.org/document/9416171/

[9] "An Efficient Hybrid Feature Selection Technique Toward Prediction of Suspicious URLs in IoT Environment," *IEEE Journals & Magazine | IEEE Xplore*, 2024. https://ieeexplore.ieee.org/document/10489965/

[10] "Robust Ensemble Machine Learning Model for Filtering Phishing URLs: Expandable Random Gradient Stacked Voting Classifier (ERG-SVC)," *IEEE Journals & Magazine | IEEE Xplore*, 2021. https://ieeexplore.ieee.org/document/9597532/

[11] ”A Survey of Intelligent Detection Designs of HTML URL Phishing Attacks,” *IEEE Journals & Magazine | IEEE Xplore*, 2023. https://ieeexplore.ieee.org/document/10019269/

[12] “A URL-Based Social Semantic Attacks Detection With Character-Aware Language Model,” *IEEE Journals & Magazine | IEEE Xplore*, 2023. https://ieeexplore.ieee.org/document/10032534/

[13] ”Potential Risk Analysis Method for Malware Distribution Networks,” *IEEE Journals & Magazine | IEEE Xplore*, 2019. https://ieeexplore.ieee.org/document/8936354/

[14] “Name Filter: A Countermeasure Against Information Leakage Attacks in Named Data Networking,” *IEEE Journals & Magazine | IEEE Xplore*, 2018. https://ieeexplore.ieee.org/document/8506363/

[15] “Spammer Detection and Fake User Identification on Social Networks,” *IEEE Journals & Magazine | IEEE Xplore*, 2019. https://ieeexplore.ieee.org/document/8719906/

[16] “PhishCatcher: Client-Side Defense Against Web Spoofing Attacks Using Machine Learning,” *IEEE Journals & Magazine | IEEE Xplore*, 2023. https://ieeexplore.ieee.org/document/10155142/

[17] “Detecting Malicious URLs via a Keyword-Based Convolutional Gated-Recurrent-Unit Neural Network,” *IEEE Journals & Magazine | IEEE Xplore*, 2019. https://ieeexplore.ieee.org/document/8629082/

[18] “A Framework for Preparing a Balanced and Comprehensive Phishing Dataset,” *IEEE Journals & Magazine | IEEE Xplore*, 2024. https://ieeexplore.ieee.org/document/10497090/

[19] “eMUD: Enhanced Manufacturer Usage Description for IoT Botnets Prevention on Home WiFi Routers,” *IEEE Journals & Magazine | IEEE Xplore*, 2020. https://ieeexplore.ieee.org/document/9187209/

[20] “Smishing Strategy Dynamics and Evolving Botnet Activities in Japan,” *IEEE Journals & Magazine | IEEE Xplore*, 2022. https://ieeexplore.ieee.org/document/9931666/