MALICIOUS URL DETECTION USING MACHINE LEARNING

A Major Project report submitted

in partial fulfillment of requirement for the award of degree

BACHELOR OF TECHNOLOGY

in

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE

by

CHIDURALA ASHWINI	**(2103A52028)**
TAKKARSU SAI PRIYA	**(2103A52110)**
KORUKANTI MADHUSRI	**(2103A52090)**
KATHA SRAVANI	**(2103A52085)**

Under the guidance of

Mr. NAGURLA MAHENDER

Assistant Professor, School of CS&AI.

SR University, Ananthsagar, Warangal, Telagnana-506371

SR University

Ananthasagar, Warangal.

SR UNIVERSITY

CERTIFICATE

This is to certify that this project entitled **"MALICIOUS URL DETECTION USING MACHINE LEARNING"** is the bonafied work carried out by **CH.ASHWINI , T.SAI PRIYA , K.MADHUSRI, K.SRAVANI** as a Major Project for the partial fulfillment to award the degree **BACHELOR OF TECHNOLOGY** in **School of Computer Science and Artificial Intelligence** during the academic year 2024-2025 under our guidance and Supervision.

Mr. N MAHENDER
Assistant Professor ,
SR University
Anathasagar,Warangal

Dr M.Sheshikala
Professor & Head,
School of CS&AI,
SR University
Ananthasagar, Warangal.

Reviewer-1
Name:
Designation:
Signature:

Reviewer-2
Name:
Designation:
Signature:

ACKNOWLEDGEMENT

We owe an enormous debt of gratitude to our Major Project guide **Mr. Nagurla Mahender, Asst.Prof** as well as Head of the School of CS&AI, **Dr. M.Sheshikala, Professor** and Dean of the School of CS&AI, **Dr.Indrajeet Gupta Professor** for guiding us from the beginning through the end of the Capstone Project with their intellectual advices and insightful suggestions. We truly value their consistent feedback on our progress, which was always constructive and encouraging and ultimately drove us to the right direction.

We express our thanks to project co-ordinators **Mr. Sallauddin Md, Asst. Prof., and Dr.D.Ramesh Asst. Prof.** for their encouragement and support.

Finally, we express our thanks to all the teaching and non-teaching staff of the department for their suggestions and timely support.

ABSTRACT

One of the most important parts of cybersecurity is identifying malicious URLs, which protects against malware, phishing attacks, and other types of online threats. In this project, machine learning methods are employed to develop a trustable system to identify and classify URLs as malicious or benign. Machine learning algorithms detect patterns commonly associated with malicious sites through the analysis of numerous factors involving URLs, such as length, format, domain data, and special character inclusion. In order to enhance detection efficiency and accuracy, we utilize algorithms like Random Forest, Gradient Boosting, XGBoost, and LightGBM. We ensure a high-performance solution for real-time malicious URL detection through data preprocessing, feature extraction, and model training. The process provides an automated and scalable means of enhancing online security and mitigating the threats of bad urls.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

In today's digital age, the internet is a booming ground for communication, commerce, and data exchange. But with increasing online activity, so has the threat of cyberattacks, especially malicious URLs that attempt to mislead users, spread malware, or capture confidential information through phishing attacks. Detection and prevention of such threats can make the online space secure. Old rule-based approaches to malicious URL discovery are not in sync with the continuously changing nature of cyberattacks, and hence more sophisticated methods have to be employed.

This project addresses the problem of hazardous URL identification through the use of machine learning to develop a scalable and efficient solution. Through the use of a set of URL features, including length, pattern, domain name, and the use of special characters, the system is able to distinguish between hazardous and harmless URLs. Advanced machine learning techniques like Random Forest, Gradient Boosting, XGBoost, and LightGBM are employed for the optimum detection speed and performance.

A critical component of this process is data preparation, or the preprocessing of raw URL data into an appropriate format for machine learning models. Subsequently, patterns are identified and fake URLs separated from real ones with the aid of feature extraction tools. The trained models are tested through a variety of performance metrics to achieve low false positives and high accuracy.

By providing a scalable and automated way of detecting malicious URLs, this project assists in enhancing cybersecurity defense and mitigating the threats posed by cyberattacks.
With the implementation of machine learning algorithms for offering proactive threat detection, prevention of hostile activity in cyberspace in real time is feasible.

1.2 PROBLEM STATEMENT

To detect malicious URLs, cybersecurity today employs conventional techniques such as heuristic, rule-based, and signature-based detection methods.
As part of signature-based detection, blacklisted URLs are stored in databases and blocked when accessed.
The technique is effective for known attacks but is unable to detect malicious URLs or zero-day attacks that were recently evolved. Like this, rule-based techniques identify suspicious URLs in line with pre-defined rules, yet are subject to the need for frequent updating and can cause false alarms.

While extremely effective, this technique is computationally costly and latency-prone, making it impractical for use in real time. A second commonly used technique is reputation-based analysis, which analyzes URLs based on user opinion, domain registration information, and history. But because newly registered malicious domains might not yet have any history, this technique tends to miss new threats.

Apart from that, the majority of the existing solutions are founded on static signals such as WHOIS information, domain durations, and normal phishing behavior. While useful, these methods cannot keep pace with the evolving methods employed by cybercriminals. The attackers keep modifying their practices, making it difficult for traditional systems to remain online without much effort.

These constraints exclude classical detection algorithms from achieving a fast, scalable, and accurate solution to malicious URL detection. Hence, there is an urgent need for machine learning-based methods that can identify trends more effectively and accurately and learn from past data and respond to evolving threats.

1.3 EXSISTING METHODS

The objective of this paper is to break the constraints of conventional approaches by providing a machine learning approach for identifying perilous URLs. In the given method, the following :

• Preprocessing and Data Gathering: We will collect a large number of URLs from diverse sources, including whitelists, blacklists, and publicly available data sets. To eliminate repetitive features and inconsistencies, the data will be preprocessed, cleaned, and rearranged.

• Feature Extraction: Some of the various URL-based features to be extracted include host-based, content-based, and lexical features.

These will all make it easy for us to distinguish between secure and insecure URLs.

Among them are the length of the URL, character frequency, age of the domain, presence of special symbols, and WHOIS.

• Model Training and Selection: Machine learning models like LightGBM, XGBoost, Gradient Boosting, and Random Forest will be trained and applied utilizing the output feature set. Cross-validation and hyperparameter tuning methods will be applied to enhance the performance of the models.

• Evaluation and Validation: A comparison of trained models will be done using accuracy, precision, recall, and F1-score as performance measurements. It will determine the best-performing model for identifying real-time abusive URLs through different methods.

• Real-Time Detection System: For the ease of integration with web browsers, security software, and enterprise cyber security systems, the finished model will be provided as an API-based solution. Real-time detection and URL categorization will be integrated into the system, providing lower latency and greater dependability.

1.4 PRESENT WORK

The aim of this project is to protect people from cyberattacks and internet fraud. Hackers now steal money and personal information using phishing websites, spyware, and fake links. Most consumers do not know which links are dangerous and which are secure. Nowadays, online transactions and social media are accessed by everyone. The attackers are employing malware, fake sites, and unrecognized links to grab cash and sensitive information. Numerous individuals end up losing money or data as a consequence of falling prey to these schemes.

Such threatening links are rapidly detected and prevented from launching cyberattacks by means of machine learning. The world can potentially surf and make transactions online in a safer way because of this.

1.5 LITERATURE SURVEY

[1]. Comparative Evaluation of Machine Learning Models for Malicious URL Detection:
This study compares different machine learning algorithms for detecting malicious URLs.RandomForest and Gradient Boosting are most effective, while Neural Networks perform well with large datasets but require more computing resources. The study highlights the potential of machine learning in cybersecurity, with future refinement focused on deep learning and feature engineering advances.

[2]. Detecting malicious URLs using machine learning techniques:
The study analyzes the capacity of machine learning to detect possibly dangerous URLs. The authors compare different models and test them against URL structure and domain information. The results show that some models, such as Random Forest and Gradient Boosting, are superior to others. Moreover, the study highlights the significance of machine learning in cybersecurity and suggests enhancement through deep learning techniques and better feature engineering.

[3]. A malicious URLs detection system using optimization and machine learning classifiers:
This paper proposes a framework for identifying malicious URLs that combines optimization methods with machine learning classifiers. The authors evaluate different models to optimize

accuracy and efficiency in detection. By improving feature selection and classification methods, the system enhances cybersecurity by better detecting malicious URLs. The study highlights the need for machine learning and optimization in improving online security.

.

[4]. Malicious URL Detection and Classification Analysis using Machine Learning Models:
This study explores the identification and categorization of malicious URLs using machine learning methods. It categorizes URLs into types such as phishing, legitimate, defacement, and malware. The application of machine learning algorithms improves the accuracy in identifying such kinds of threats. The study raises concerns like the changing nature of threats and the management of large datasets, highlighting the necessity of effective cybersecurity measures.

[5].Towards the detection of malicious URL and domain names using machine learning:
The work suggests an adaptive solution for the detection of malicious URLs and domain names based on machine learning. The approach extracts semantic features from various components of a URL and can adapt to new threat information. Out of the machine learning algorithms tested, Random Forest demonstrated the best accuracy of over 96% along with advantages regarding interpretability and efficiency.

[6]. Detection of phishing URLs using machine learning:
This study explores methods for detecting phishing sites through the analysis of various URL features using machine learning methods. The authors highlight the importance of lexical features, host features, and web page purpose in distinguishing genuine URLs from phishing URLs. They also improve these features to increase the accuracy of their detection model to identify more phishing sites and improve cybersecurity.

[7]. Comparitive analysis of machine learning algorithms on malicious URL prediction:
The study conducts a comparative analysis of various machine learning models used in predicting malicious URLs. Through the comparison of various models, the study establishes the best methods of identifying malicious websites. This review seeks to enhance security against malicious activities by determining the optimal algorithms for malicious URL prediction.

[8]. An Automated System to Detect Phishing URL by Using Machine Learning Algorithm:
The study performs comparative assessment of several machine learning models used to predict dangerous URLs. From the analysis of various models, the study concludes the most efficient

techniques for identifying malicious sites. This review works towards enhancing security practices against malicious activities by determining the best algorithms to predict malicious URLs.

[9]. New Heuristics Method for Malicious URLs Detection Using Machine Learning:

The study presents a machine-learning-based technique for malicious URL detection. It compares the three models: Logistic Regression, Random Forest, and SVM. Random Forest is the most efficient one, as it had 99% training accuracy and 90.5% validation accuracy. The article emphasizes the necessity of updating models to combat changing cyberattacks.

[10]. Malicious URL Detection Using Machine Learning:

This research explores the use of machine learning to detect malicious URLs by examining Host-based features are related to data about the server, whereas lexical features relate to the composition of the URL. Combining these features provides enhanced detection precision and makes it more useful in the fight against cyber threats.

[11]. Supervised machine learning approach for identification of malicious URLs:

This paper presents supervised machine learning for malicious URL detection, which are commonly employed in cyberattacks such as phishing and malware propagation. Blacklist techniques are futile against emerging threats, hence the research investigates lexical features (relating to URL composition) and host-based features (relating to the server). Based on these characteristics, the proposed technique enhances detection accuracy and strengthens cybersecurity.

[12]. Characteristics of understanding urls and domain names features: the detection of phishing websites with machine learning methods:

This research explores the detection of phishing websites using machine learning algorithms that identify URL and domain name features. The experiment used a dataset with legitimate and phishing websites and tested several different classifiers, of which Random Forest produced the best results. The approach is more efficient at detection by focusing on URL analysis rather than doing extensive reviews of webpage content.

[13]. Malicious URL Detection using Machine Learning: A Survey:

This paper investigates machine learning methods for detecting malicious URLs and their limitations relative to traditional blacklist methods. It structures the inquiry along feature representation and algorithms, as well as challenges, design issues, and promising directions in future research within the domain of cybersecurity.

[14]. Malicious URL Detection using Machine Learning Algorithms:

This research investigates the use of machine learning for the detection of malicious URLs commonly linked with cyber attacks like phishing and malware propagation. It highlights the limitations of traditional blacklisting methods and evaluates various machine learning methods to enhance detection accuracy and enhance cybersecurity.

[15]. Detecting Malicious URLs Using Machine Learning Techniques: Review and Research Directions:

The paper gives a comprehensive review of numerous machine learning techniques employed for the detection of malicious URLs. It evaluates several algorithms.The study also emphasizes a number of challenges, including choosing suitable features, maintaining dataset quality, and adjusting detection systems to counteract new threats. The authors give directions for further research, highlighting the need for developing integrating machine learning techniques with other cybersecurity measures.

CHAPTER 2

HARDWARE / SOFTWARE TOOLS

REQUIREMENT SPECIFICATION (S/W & H/W)

Software requirements

Programming Language: We will use Python for project code development because it has a vast machine learning library and is easy to use.

Development Environment: The primary development environment will be Google Colab. It makes TPUs and GPUs available for free, which can accelerate machine learning operations such as model training by a large margin.

Machine Learning Libraries: Employ libraries such as scikit-learn, numpy and pandas for data manipulation and numerical operations.

Version Control: GitHub shall be utilized for version control in a bid to keep the integrity of the codebase intact, follow changes, and ensure effortless communication.
Documentation: Word documents and PowerPoint may be utilized to create detailed documentation, such as installation guides, API references, and code descriptions.

Hardware requirements

CPU: For effective computing, a laptop with a contemporary multicoreprocessor. For the best performance, look for AMD Ryzen series or Intel Core i5 or i7 processors.

RAM: To properly manage data processing and model training, using 8GB to 16GB of RAM. When working with larger datasets and complicated models, more RAM could be useful.

Storage: For faster data access and improved overall performance, ideally choosed a laptop with solid-state drive (SSD) storage. It is advised to have 256GB or more of storage space in order to hold project files and datasets.

GPU: Having a dedicated GPU on the laptop can be helpful for local development and testing even though Google Colab will enable access to GPUs for model training. GPUs from NVIDIA, like the GeForce

2.1 SYSTEM DESIGN

System design is the process of defining the architecture, components, modules, interfaces, and data of a system to meet specified requirements. It involves a description of the hardware, software, and networking infrastructure necessary for the system implementation, and the organization of the interaction between the system components.

2.2 FLOWCHART

A flowchart is a graphical model of a process, method, or system representing the progression of stages through symbols and arrows. It is widely utilized in multiple fields, such as software design, business process, and engineering, for planning, problem-solving, and documentation.

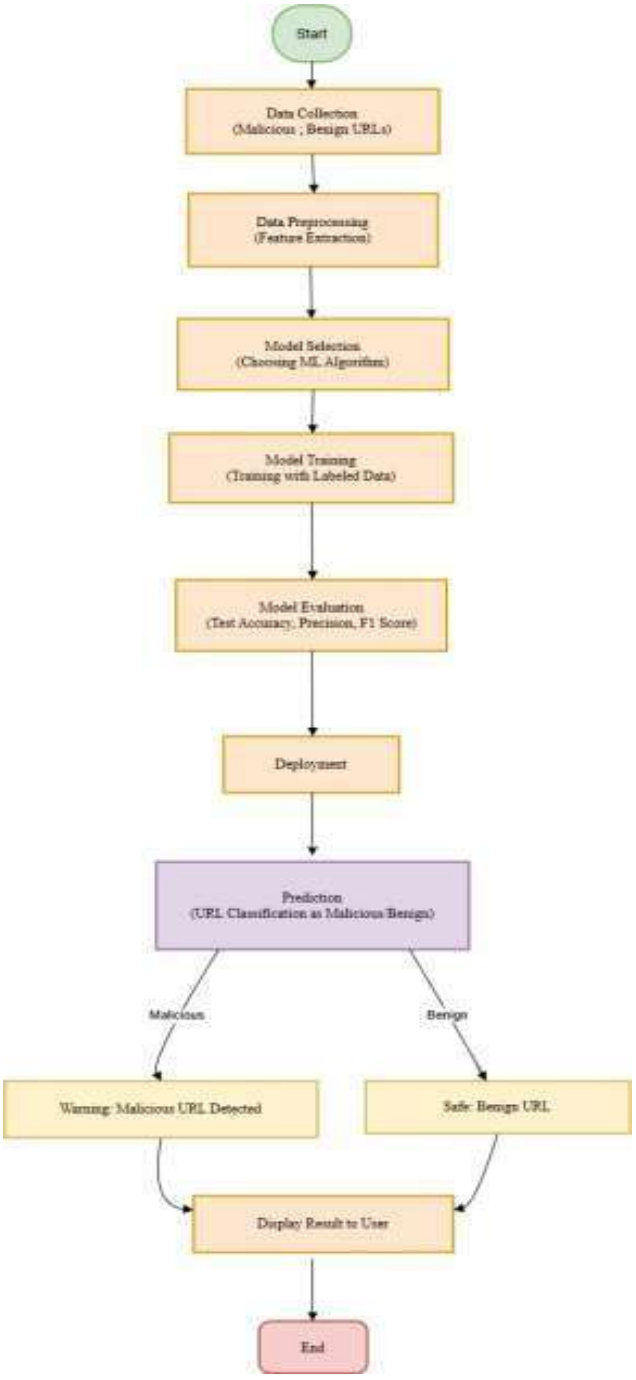

Fig 2.2.1 Flow chart

2.3 UML DIAGRAMS

A modeling language called Unified Modeling Language (UML) is employed to build, develop, document, and present a software system's artifacts. UML offers a collection of graphical notation tools for building visual representations of object-oriented software systems.

1. **COMPONENT DIAGRAM:** Component diagram is among the UML diagrams used in the display of the structural elements of a system and how they relate to each other. Its primary function in software design is to illustrate the interdependencies, interactions, and integration of various software components into a system.

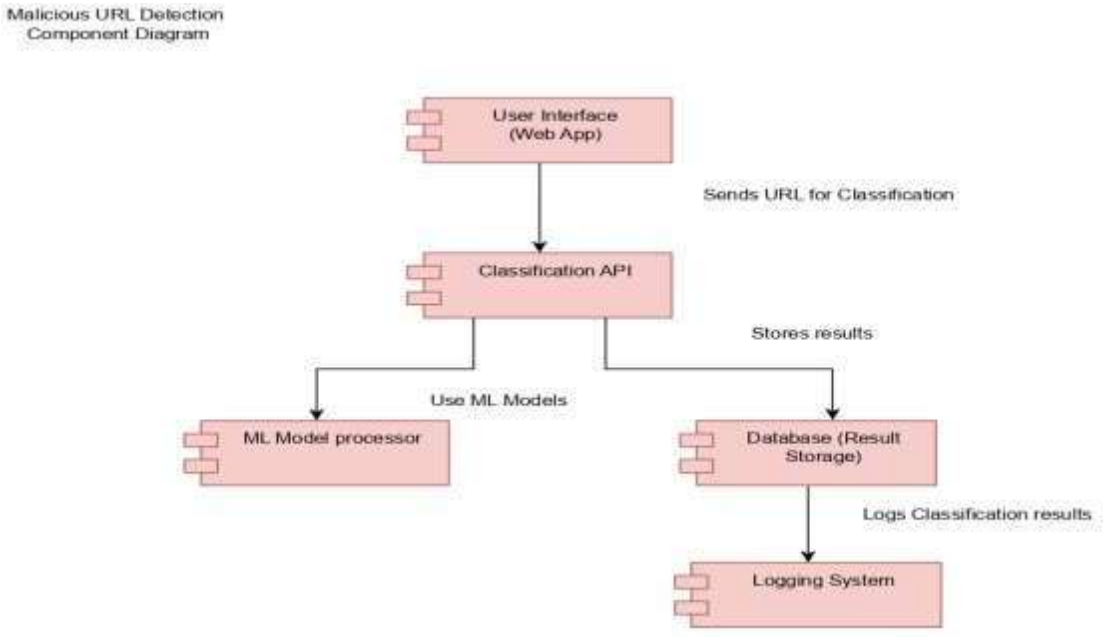

FIG 2.3.1 Component Diagram

2. USE CASE: In Unified Modeling Language (UML), a use case diagram is a behavioral diagram where the interaction of a system with actors (outside entities) and its functional requirements are shown. It shows how the system works in general and how different users (actors) can interact with the system.

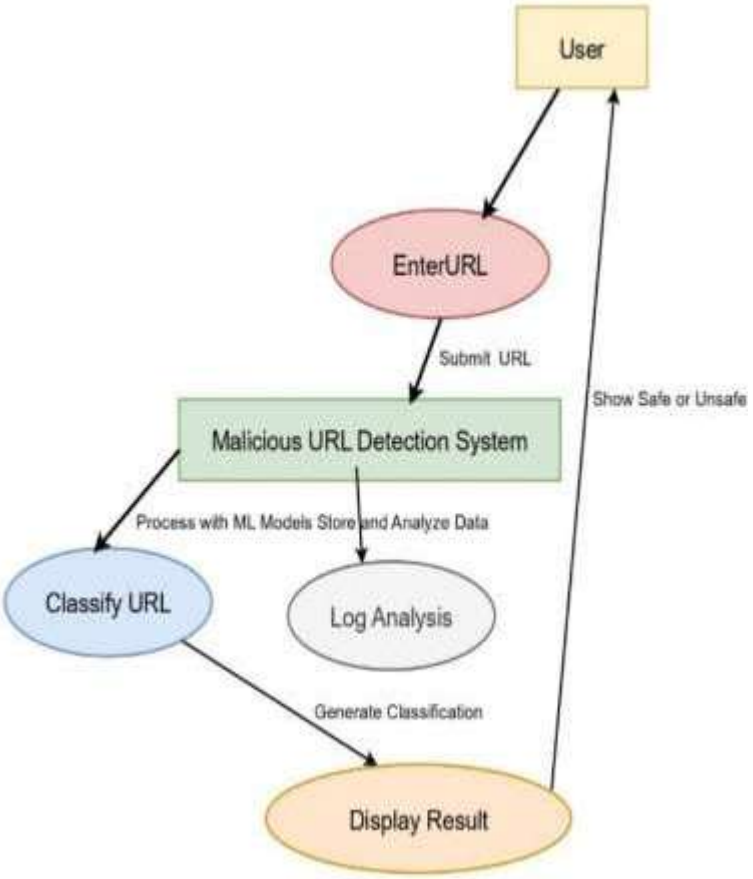

Fig 2.3.2 Use case Diagram

3. STATE CHART DIAGRAM: A behavioral UML diagram that diagrammatically depicts the various states of an object in a system and how this object alters these states based on events or a condition is a state chart diagram, or a state machine diagram, at times.

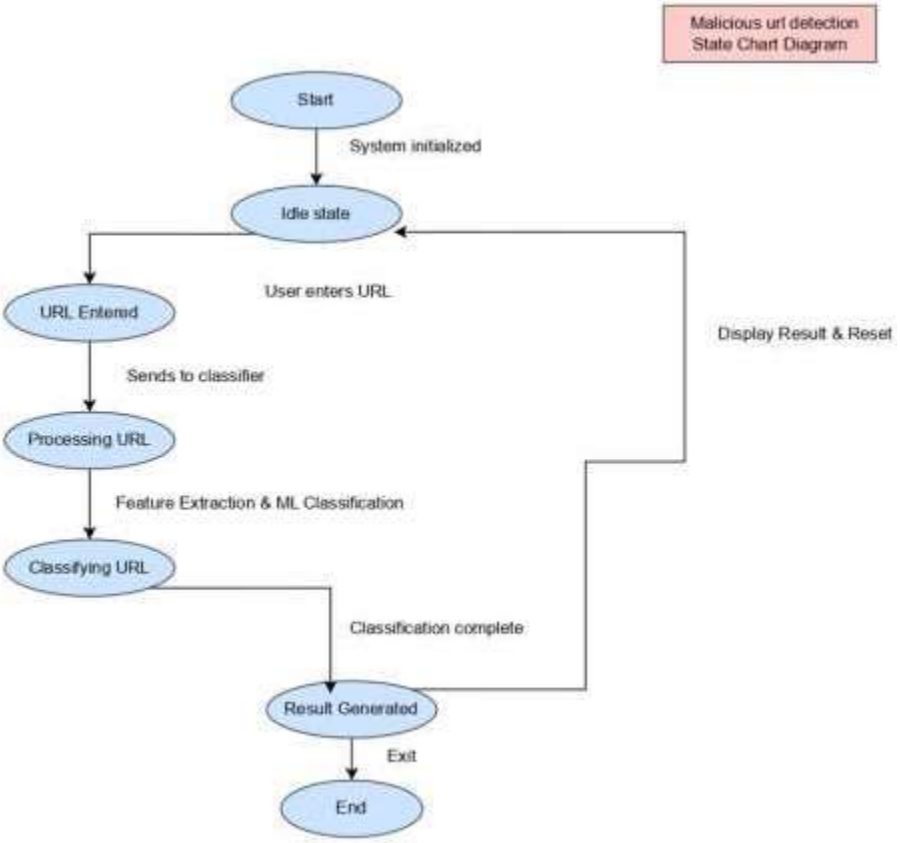

Fig 2.3.3 State chart Diagram

4. DEPLOYMENT DIAGRAM: One of the UML structure diagrams that illustrate the physical arrangement of a system's software and hardware components is referred to as a deployment diagram. It illustrates the deployment of software components (artifacts) onto hardware components (nodes) and interactions between the nodes. In designing a system's infrastructure and illustrating its physical deployment, deployment diagrams are optimally used.

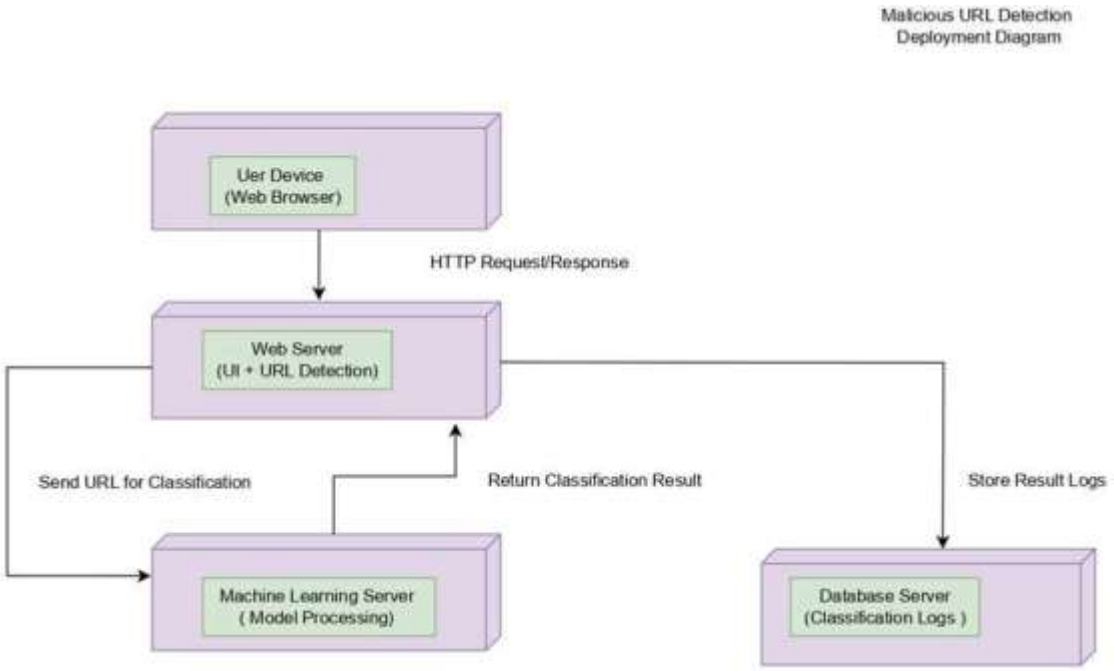

Fig 2.3.4 Deployment Diagram

5. **CLASS DIAGRAM:** A class diagram is a UML structural diagram that depicts a system's classes, properties, methods, and object-to-object relationships for specifying its structure. Class diagrams are often used to depict an object-oriented system's architecture and are needed to model the static view of an application.

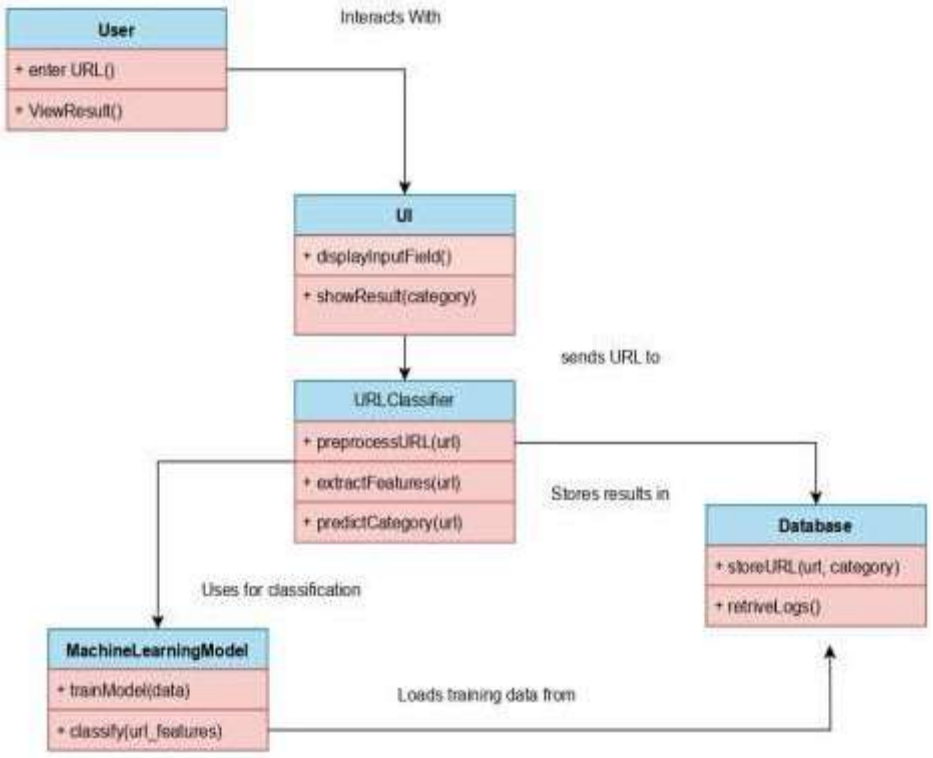

Fig 2.3.5 Class Diagram

6. **ACTIVITY DIAGRAM:** One of the UML behavior diagrams that depict a system or process workflow is an activity diagram. It is helpful when modeling the dynamic system behavior because it shows the sequence of activities, decisions, and actions carried out within a process. Activity diagrams are very helpful when describing intricate processes and parallel actions because they depict the business logic at a high level.

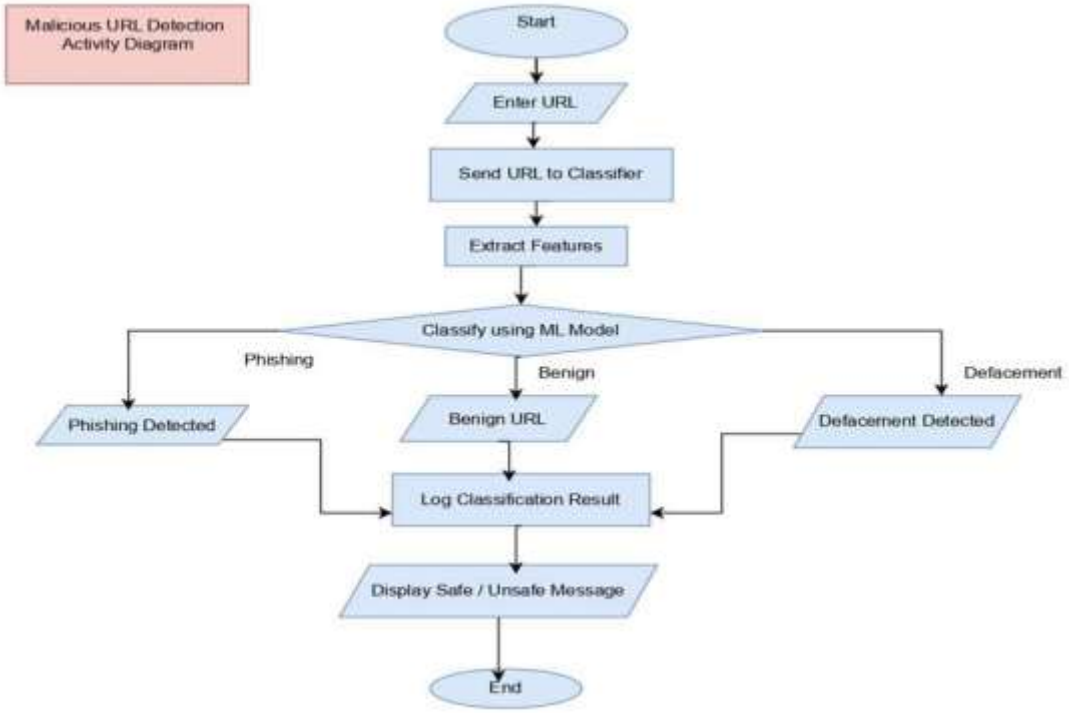

Fig 2.3.6 Activity Diagram

2.4 RISK ANALYSIS

Risk analysis is essential for identifying potential factors that could lead to the failure of a project or the inability to complete it by the deadline. Here are some factors that pose risks to the success and timely completion of a malicious url detection using machine learning project.

The Malicious URL Detection using Machine Learning project risk analysis identifies key issues that compromise efficiency, security, and accuracy. Predictions are subject to bias from data risks such as imbalanced datasets and low-quality data. Preprocessing, balanced sampling, and frequent dataset updates are necessary to mitigate this. Cross-validation, regularization, and anomaly detection can reduce model risks such as overfitting and inability to detect zero-day attacks. Attackers can bypass detection because of security risks such as adversarial attacks; hybrid detection techniques and adversarial training mitigate this. Operational risks necessitate cloud-based deployment and optimized models (e.g., LightGBM, XGBoost) because of scalability concerns and delay in real-time detection. An efficient, robust, and flexible system for fraudulent URL detection is ensured by actively mitigating these risks.

CHAPTER 3

PROJECT IMPLEMENTATION

3.1 PROPOSED SYSTEM

This project offers a machine learning approach to identify malicious URLs with a goal of overcoming the limitations of traditional approaches. The following crucial steps in the provided technique are:

Preprocessing and Data Collection: We shall obtain a big list of URLs from heterogeneous sources such as openly available data sets, blacklists, and whitelists.
The information will be cleaned, preprocessed, and reorganized to eliminate inconsistencies and repeated features.

• Feature Extraction: Some of the various URL-based features to be extracted include host-based, content-based, and lexical features.
These will all make it easy for us to distinguish between secure and insecure URLs.
Included among these are the length of the URL, character distribution, domain age, presence of special symbols, and WHOIS.

• Model Selection and Training: Different machine learning models such as LightGBM, XGBoost, Gradient Boosting, and Random Forest will be trained and executed on the output feature set. Cross-validation and hyperparameter tuning techniques will be applied to enhance the performance of the models.

• Evaluation and Validation: The learned models will be compared based on performance metrics including accuracy, precision, recall, and F1-score. By comparing different methods, the best-performing model for real-time malicious URL detection will be determined.

• Real-Time Detection System: To make integration with web browsers, security software, and enterprise-level cybersecurity systems easy, the final model will be provided as an API-based solution. Real-time detection and URL classification will be integrated into the system to provide lower latency and higher dependability.

3.2 MODEL ARCHITECTURE

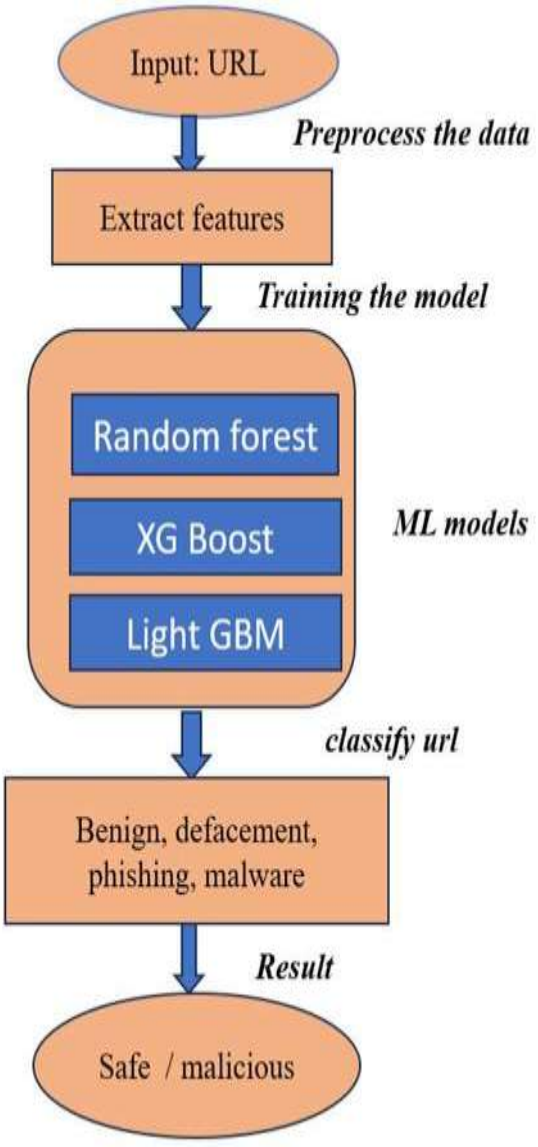

Fig 3.2.1 Model Architecture

3.3 PROCEDURE

Step 1: Gathering Information

Collect datasets from sites such as PhishTank, OpenPhish, Alexa, and Kaggle that include labeled URLs (both benign and malicious).

To enhance model generalization, make sure the datasets are diverse.

Step 2: Preparing the data

Take care of missing or incorrect data and eliminate duplicate URLs.

For analysis, transform text-based URLs into structured representations.

To ensure uniformity, normalize numerical features (such as URL length).

Step 3: Feature Extraction

Extract lexical features, including URL length, number of special characters, and presence of an IP address.

Extract Domain-Based Features (e.g., presence of HTTPS, age of the domain, and WHOIS data).

Extract content-based features (e.g., iframe detection, HTML, and JavaScript analysis).

Step 4: Transformation & Feature Engineering

Categorical variables must be encoded.

Apply word embeddings or TF-IDF to textual URL parts.

Apply ML-based or statistical approaches to select features.

Step 5: Training and Model Choice

Train a range of machine learning models, including Random Forest, Gradient Boosting, XGBoost, and LightGBM.

Split data (e.g., 80-20 split) into training and test sets.

To avoid overfitting and optimize hyperparameters, apply cross-validation.

Step 6: Model Evaluation

Utilize ROC-AUC, F1-score, recall, accuracy, and precision to measure performance.

Compare multiple models and choose the one that performs best.

Step 7: Implementation & Instantaneous Identification

To integrate the model with apps, develop an API using Flask/Django.

For real-time scanning, host the model on a cloud platform (such as AWS Lambda).

Regularly update the model to detect new threats.

Step 8: Observation and Enhancement

To improve accuracy, continue adding new data and retraining the model.

CHAPTER - 4

4.1 SIMULATION SETUP

.

Fig 4.1.1 Stimulation setup User Interface

Step -1. Overview

As hackers employ fake URLs to distribute malware, conduct phishing attacks, and try defacement, identifying malicious URLs is an important cybersecurity problem. On the basis of features that have been extracted, this project utilizes machine learning to classify URLs as benign, phishing, malware, or defacement.

The following are part of the implementation:

Features extracted from URLs were utilized to train the machine learning model.
Google Colab for model training and execution.
Real-time URL analysis via a gradient-based user interface (UI).
Threat assessment visualization to identify potential threats.

Step-2. Setting up a Machine Learning Model

Labeled data with both harmless and malicious URLs are employed to train the model. Some of the key steps are:

(a) Collecting Datasets

public datasets such as the Kaggle Malicious URL Dataset, PhishTank, and OpenPhish.
Features (length, character distribution, presence of suspicious keywords, TLD, entropy, etc.)

drawn from URLs.

Labeled data (defacement, malware, phishing, and harmless).

(b) Feature Extraction

Lexical Features: Suspicious words, length, and special characters.

Host-based features: WHOIS information and age of the domain.

Embedded links and HTML tags are some content-based features.

DNS records and IP reputation are some network-based features.

(b) Model Selection and Training Algorithms Deep Learning models (LSTM for sequential patterns), Random Forest, Decision Tree, SVM, and XGBoost were employed.

Preprocessing involves feature engineering, data cleaning, and dataset splitting for testing and training.

Training: Scikit-learn/TensorFlow and Python in Google Colab.

Step- 3. Enabling Gradio UI's URL Detection

URL classification is achievable in real-time due to the Gradio-created user interface.

(a) UI Elements:

Users type in text as a URL.

Scan Button forwards the URL for scanning.

Threat Assessment Graph: Displays the probability of each classification (defacement, malware, phishing, and benign).

Model predictions and risk notifications are indicated in the Quick Result Section.

Feature Importance Display: Highlights the most important features that influence detection.

(b) Execution Flow: The user provides a URL input to the Gradio UI.

Feature Extraction: A feature vector is generated by preprocessing the URL.

Model Prediction: The URL is predicted by the trained machine learning model.

Threat Visualization: A bar chart is used to display the probability of each category.

Warning Display: If a URL is found to be potentially dangerous, a warning message alerts users.

4. Setup of Google Colab

Used Python Libraries: Pandas, NumPy, Scikit-learn, TensorFlow, Gradio.

Processing the Dataset: The CSV datasets are loaded and prepared.

Execution during Training: The model training process is executed using the Google Colab GPU platform.

Deployment: The trained model is exported and incorporated into the Gradio web application.

Testing in Real-time: Users provide URLs and receive real-time classifying results.

5. Conclusion

This project successfully identifies malicious URLs based on machine learning and offers a real-time analysis interactive UI. Google Colab (for training) and Gradio (for UI deployment) integration guarantees end-to-end protection against cybersecurity threats.

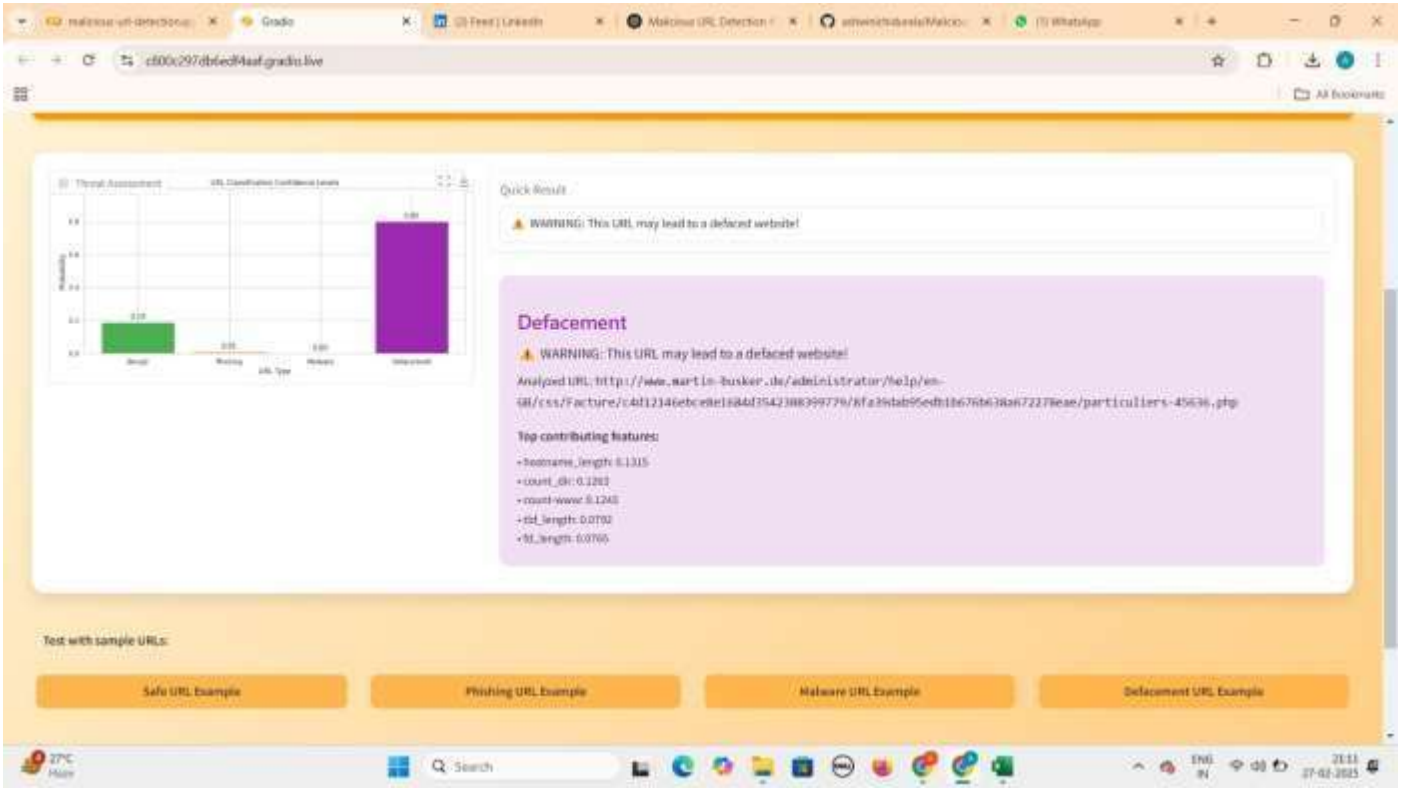

Fig 4.1.2 User Interface classification report

4.2 RESULTS

PROJECT RESULT:

Malicious URL Detection Using Machine Learning result

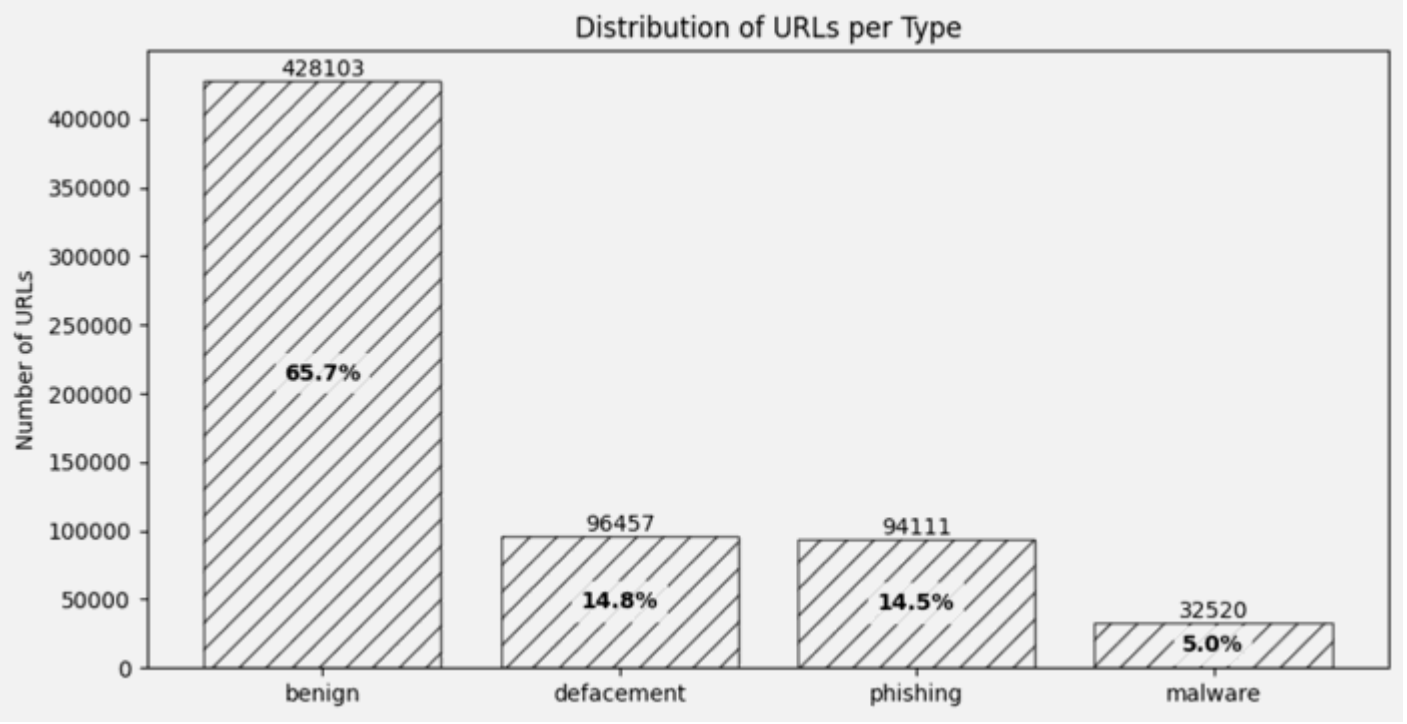

Fig 4.2.1 bar chart for types of Url's distribution

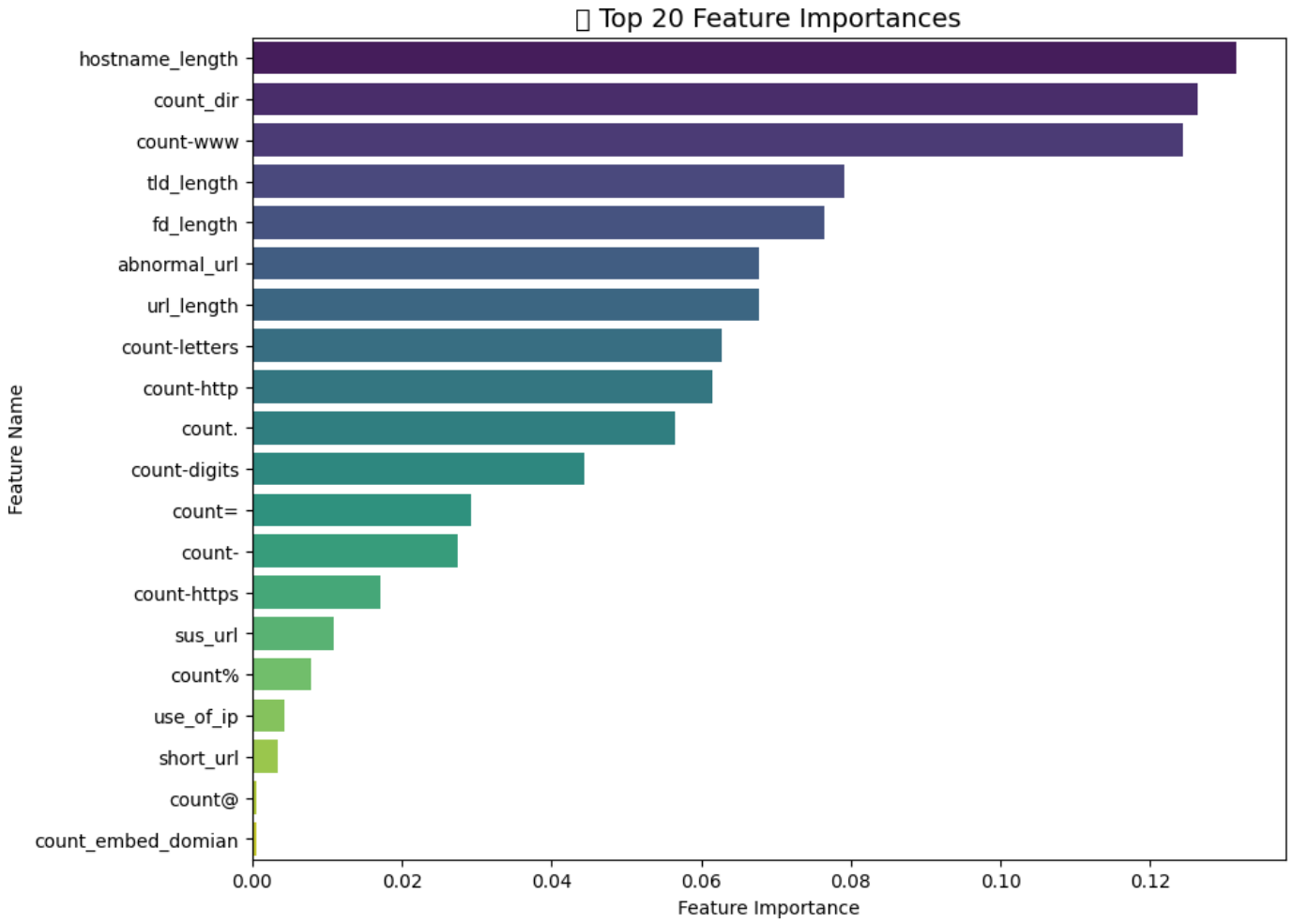

Fig 4.2.2 Top 20 important features

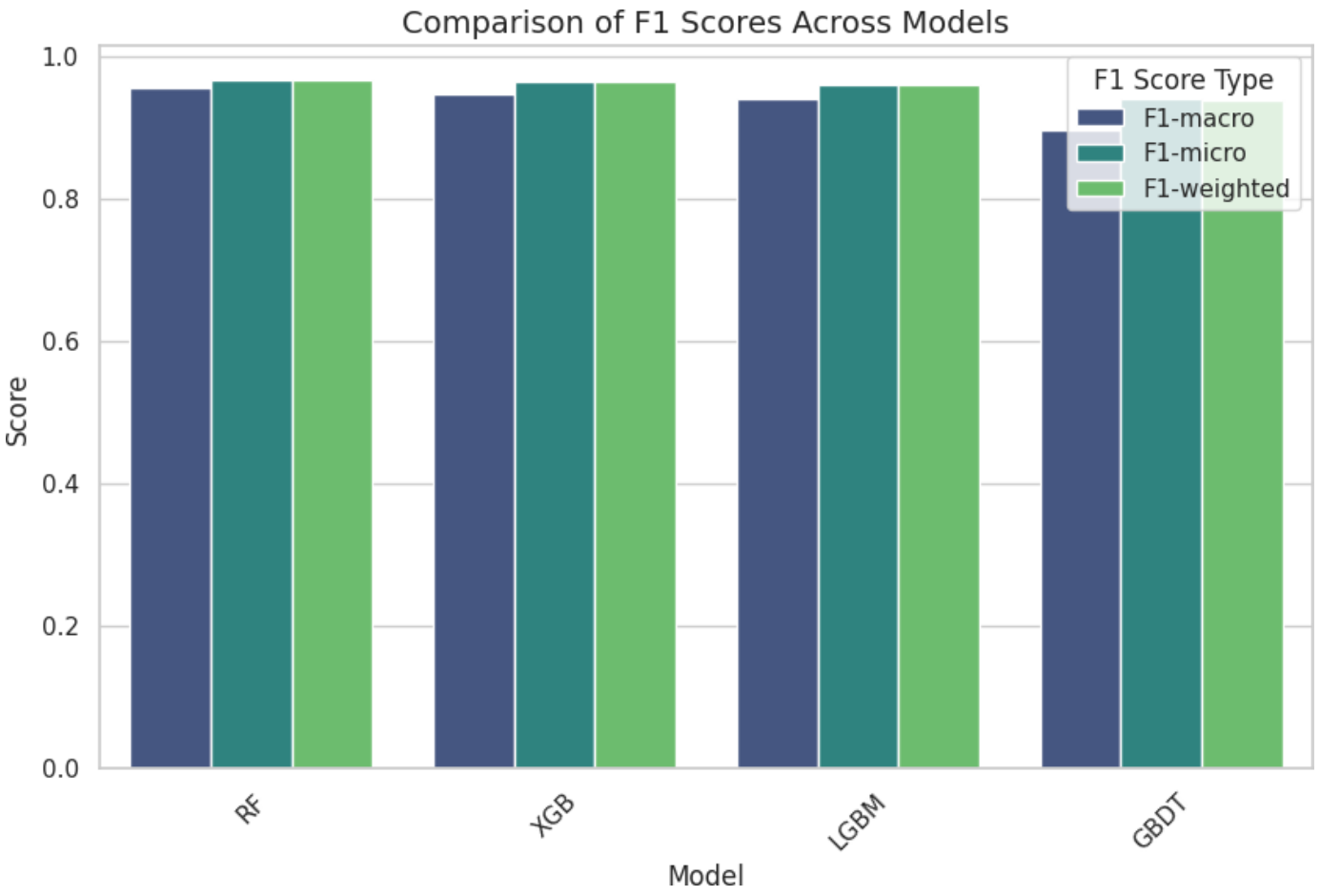

Fig 4.2.3 comparision of model performances using F1 scores

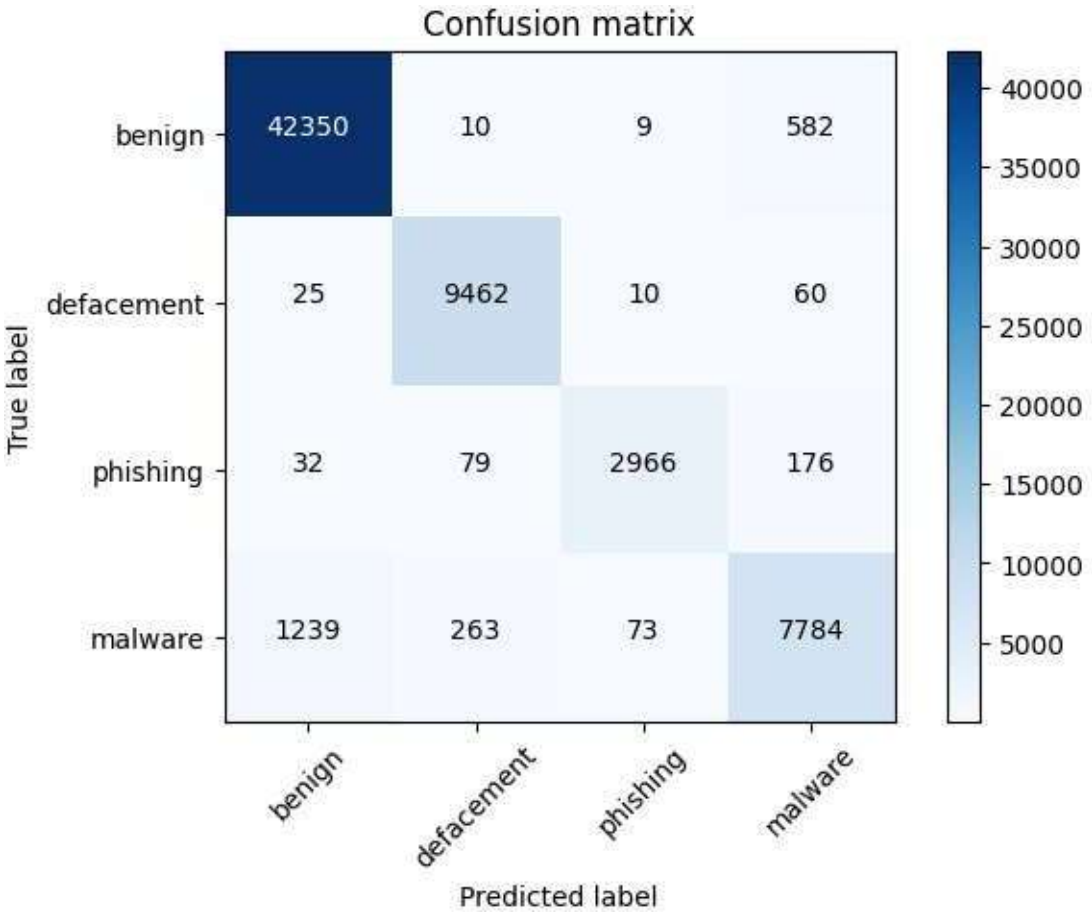

Fig 4.2.4 Confusion Matrix

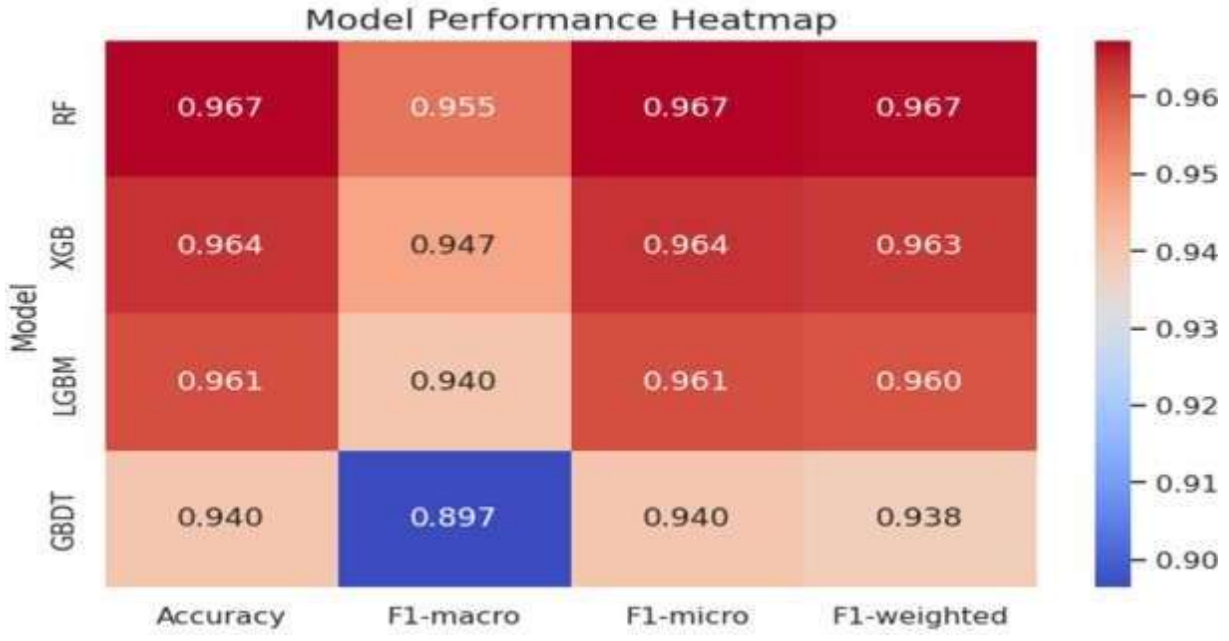

Fig 4.2.5 Heat Map

Model	Accuracy	F1-macro	F1-micro	F1-weighted
RF	0.967260	0.955395	0.967260	0.966905
XGB	0.963729	0.947367	0.963729	0.963042
LGBM	0.960719	0.940433	0.960719	0.960017
GBDT	0.940433	0.896504	0.940433	0.938472

Table 1: different model performances

Highest accuracy: 0.967

4.3 RESULT COMPARISON AND ANALYSIS

The dataset quality, feature selection, and machine learning algorithm all influence the performance of a malicious URL detection model. Here, we analyze feature importance, evaluate multiple models using performance metrics, and determine practical effectiveness.

We compare four machine learning models—Random Forest (RF), XGBoost (XGB), LightGBM (LGBM), and Gradient Boosted Decision Trees (GBDT)—on the basis of the results that have been reported. Accuracy, F1-macro, F1-micro, and F1-weighted are the main performance metrics considered.

Results Analysis

Best Performing Model: Random Forest (RF) RF performed the best among malicious URL detection options in this project because it achieved the highest accuracy (96.72%) and F1-scores in all metrics.

The model maintains performance across a number of classes (benign, phishing, malware, defacement, etc.), with minimal bias towards any single category, as can be seen by the high F1-macro (95.54%).

Even when datasets are unbalanced, RF works well, accurately classifying both frequent and infrequent events, as per F1-weighted scores (96.69%).

Although not as strong as RF and XGB, LightGBM (LGBM) Performance Accuracy (96.07%) and F1-macro (94.04%) are pretty good nonetheless.

Since LGBM is tailored for big datasets, its performance is similar to XGBoost and can be a great pick when speed is the top priority.

The poorest performing model is Gradient Boosted Decision Trees (GBDT).

GBDT was the poorest across all parameters with an accuracy of 94.04%.

Both F1-weighted (93.85%) and F1-macro (89.65%) indicate that this model suffers from class imbalance.

While GBDT remains to be helpful, RF, XGB, and LGBM are better in recall and precision.

With regards to overall performance, Random Forest performs best, especially when working with unbalanced datasets.

The second-best model, XGBoost, offers better computational efficiency and strong performance for widespread application.

4.4 LEARNING OUTCOME

The Malicious URL Detection Using Machine Learning research provided significant findings regarding the applications of supervised learning methods in cybersecurity. This project provided us with better knowledge about how machine learning methods such as Random Forest (RF), XGBoost (XGB), LightGBM (LGBM), and Gradient Boosted Decision Trees (GBDT) can utilize derived features to identify URLs as benign, phishing, malware, or defacement. We explored various feature engineering approaches to enhance model performance, such as looking into URL length, frequency of characters, and directory structure. We also learned to apply metrics like Accuracy, Precision, Recall, and F1-score (macro, micro, weighted) in order to gauge the usefulness of the model and measure the impact of imbalanced datasets on the performance of classification.

Along with machine learning, the project opened our eyes to the world of cybersecurity by exposing the different types of malicious URLs and the risks that they pose to users. We knew the way hackers modify URLs to manipulate customers and bypass security protection measures. The importance of automated threat detection mechanisms came to the forefront, outlining how machine learning can augment classical security measures such as blacklists and heuristic-based methods. Utilizing Gradio to develop a web-based user interface provided us with hands-on experience deploying machine learning models in an accessible and user-friendly manner. With Google Colab, we were able to effectively train and test our models and learned how crucial feature selection, data pre-processing, and hyperparameter tuning are to enhancing accuracy.

This research also highlighted the challenges of applied cybersecurity use cases, including adversary attacks and adaptive threat behavior. It emphasized the importance of continuously updating models and potential improvements, such as implementing deep learning methods (LSTMs, Transformers) and real-time threat intel. Overall, this project not only enhanced our technical skill set in cybersecurity and machine learning, but it also showed just how vital automation is to effectively detecting and halting attacks.

CHAPTER 5

5.1 CONCLUSION WITH CHALLENGES

To detect malicious URLs, we employed a range of machine learning models in this research, such as Random Forest (RF), XGBoost (XGB), LightGBM (LGBM), and Gradient Boosting. Accuracy, F1-macro, F1-micro, and F1-weighted scores were the building blocks of the evaluation. The Random Forest model performed the best for detecting hazardous URLs with the highest accuracy (96.73%) and overall performance. Overall, ensemble learning models (e.g., RF and XGB) were highly effective in detecting malicious URLs, demonstrating that combining multiple decision trees enhances robustness and generalization.

Challenges Faced by the Feature Engineering Complexity: Since raw URLs contain very little amount of structured data, extracting useful attributes from them was a laborious task. Designing relevant features to enhance model performance was one of the most significant challenges.

Data Imbalance: The dataset had an imbalanced ratio of benign and harmful URLs that might have influenced model performance and training. The class imbalance had to be addressed to prevent skewed predictions.

Overfitting Risks

Though robust, ensemble learning models were prone to overfitting, especially when hyperparameters were not optimally tuned.

Regularization and cross-validation techniques were needed to mitigate this issue.

Extension to Novel Threats

Since malicious URL patterns are dynamic, models trained on historical data will not generalize well to novel threats.

Retraining and adaptation should be performed continuously to be effective.

Feature Interpretability: While Random Forest and XGBoost performed with very high accuracy, there was no easy way to deduce which feature contributed the most to prediction, making it harder for security researchers to interpret the model.

Despite these challenges, the research illustrated that ensemble learning methods are a valuable tool to fight online threats and are comparatively effective at spotting malicious URLs.

5.2 FUTURE SCOPE

utilizing Natural Language Processing (NLP) techniques to analyze the URL text structure to identify phishing patterns.
applying online learning techniques to continuously adapt to evolving malicious URL patterns.
maintaining models up-to-date with advancing threats using automatic retraining mechanisms.
We can further enhance malicious URL detection systems' accuracy, efficacy, and robustness by applying these advances, making them more reliable for cybersecurity use.

5.3 REFERENCES

1) Mankar, N. P., Sakunde, P. E., Zurange, S., Date, A., Borate, V., & Mali, Y. K. (2024, April). Comparative Evaluation of Machine Learning Models for Malicious URL Detection. In 2024 MIT Art, Design and Technology School of Computing International Conference (MITADTSoCiCon) (pp. 1-7). IEEE.
2) Vanhoenshoven, F., Nápoles, G., Falcon, R., Vanhoof, K., & Köppen, M. (2016, December). Detecting malicious URLs using machine learning techniques. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)* (pp. 1-8). IEEE.
3) Lee[1], O. V., Heryanto, A., Ab Razak, M. F., Raffei, A. F. M., Phon, D. N. E., Kasim, S., & Sutikno, T. (2020). A malicious URLs detection system using optimization and machine learning classifiers. *Indonesian Journal of Electrical Engineering and Computer Science*, *17*(3), 1210-1214.
4) DR, U. S., & Patil, A. (2023, January). Malicious url detection and classification analysis using machine learning models. In *2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)* (pp. 470-476). IEEE.
5) Ghalati, N. F., Ghalaty, N. F., & Barata, J. (2020, April). Towards the detection of malicious URL and domain names using machine learning. In *Doctoral Conference on Computing, Electrical and Industrial Systems* (pp. 109-117). Cham: Springer International Publishing.
6) James, J., Sandhya, L., & Thomas, C. (2013, December). Detection of phishing URLs using machine learning techniques. In *2013 international conference on control communication and computing (ICCC)* (pp. 304-309). IEEE.
7) Vaishnavi, D., Suwetha, S., Jinila, Y. B., Subhashini, R., & Shyry, S. P. (2021, May). A comparative analysis of machine learning algorithms on malicious URL prediction. In *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)* (pp. 1398-1402). IEEE.
8) Parasar, D., & Jadhav, Y. H. (2021). An Automated System to Detect Phishing URL by Using Machine Learning Algorithm. In *International Conference on Mobile Computing and Sustainable Informatics: ICMCSI 2020* (pp. 217-225). Springer International Publishing.
9) Hasan, M. K. (2024). New Heuristics Method for Malicious URLs Detection Using Machine Learning. *Wasit Journal of Computer and Mathematics Science*, *3*(3), 60-67.
10) Catak, F. O., Sahinbas, K., & Dörtkardeş, V. (2021). Malicious URL detection using machine learning. In *Artificial intelligence paradigms for smart cyber-physical systems* (pp. 160-180). IGI Global Scientific Publishing..
11) Badugu, S., & Kolikipogu, R. (2020). Supervised machine learning approach for identification of malicious URLs. In *Advances in Computational Intelligence and Informatics: Proceedings of ICACII 2019* (pp. 187-197). Springer Singapore.
12) Kara, I., Ok, M., & Ozaday, A. (2022). Characteristics of understanding URLs and domain names features: the detection of phishing websites with machine learning methods. *IEEE Access*, *10*, 124420-124428.
13) Sahoo, D., Liu, C., & Hoi, S. C. (2017). Malicious URL detection using machine learning: A survey. *arXiv preprint arXiv:1701.07179*.
14) Ferreira, M. (2019, January). Malicious URL detection using machine learning algorithms. In *Proc. Digit. Privacy Security Conf* (pp. 114-122).
15) Aljabri, M., Altamimi, H. S., Albelali, S. A., Al-Harbi, M., Alhuraib, H. T., Alotaibi, N. K., ... & Salah, K. (2022). Detecting malicious URLs using machine learning techniques: review and research directions. *IEEE Access*, *10*, 121395-121417.