HARMFUL WEBSITE DETECTION

Real Time Project report

Submitted in partial fulfilment of the requirement for the award of the Degree of

Bachelor of Technology (B. Tech)

In

Computer Science and Engineering (AIML)

By

Nasreen Khatoon	**22AG1A66G9**
M.Rishitha	**22AG1A66G2**
B.Mounika	**22AG1A66D7**
E.Kavya prasanna	**23AG5A6620**

Under the Esteemed Guidance of

Mr C. V. Ajay Kumar

Assistant Professor

Department of Computer Science and Engineering (AI&ML)

ACE ENGINEERING COLLEGE

AN AUTONOMOUS INSTITUTION

(NBA Accredited B.Tech Courses: ECE, EEE & CSE)

(Affiliated to Jawaharlal Nehru Technological University, Hyderabad, Telanganana)

Ankushapur(V), Ghatkesar(M), Medchal- Malkajgiri Dist - 501 301.

JULY 2024.

ACE

ENGINEERING COLLEGE

AN AUTONOMOUS INSTITUTION

Website: www.aceec.ac.in E-mail: info@aceec.ac.in

CERTIFICATE

This is certify that the Real Time Project work entitled **"Harmful Website Detection"** is being submitted by **Nasreen khatoon(22AG1A66G9), M.Rishitha(22AG1A66G2), B.Mounika(22AG1A66D7)**, **E.Kavya prasanna(23AG5A6620),** in partial fulfilment for the award of Degree of **BACHELOR OF TECHNOLOGY in DEPARTMENT OF COMPUTER SCIENCE ENGINERING (AIML)** to the Jawaharlal Nehru Technological University, Hyderabad is record of Bonafide work carried out by them under our guidance and supervision.

The results embodied in this project have not been submitted by the student to any other University or Institution for the award of any Degree or Diploma.

Internal Guide	Head of the Department
Mrs C. V. Ajay Kumar	Dr S. Kavitha
Assistant Professor	Assoc. Professor and
	Head Dept. of CSE(AI&ML)

ACKNOWLEDGEMENT

We would like to express our gratitude to all the people behind the screen who have helped us transform an idea into a real time application.

We would like to express our heart-felt gratitude to our parents without whom we would not have been privileged to achieve and fulfil our dreams.

A special thanks to our General Secretary, **Prof. Y. V. Gopala Krishna Murthy**, for having founded such an esteemed institution. We also grateful to our beloved principal, **Dr. B. L. RAJU** for permitting us to carry out this project.

We profoundly thank **Dr. S. Kavitha**, Associate Professor and Head of the Department of Computer Science and Engineering (AIML), who has been an excellent guide and also a great source of inspiration to our work.

We extremely thank **Mrs. J Bhargavi**, Assistant Professor, Project coordinator, who helped us in all the way in fulfilling of all aspects in completion of our Real Time Project.

I am very thankful to my internal guide, **Mr C.V.Ajay Kumar**, Assistant Professor who has been an excellent and also given continuous support for the Completion of our project work.

The satisfaction and euphoria that accompany the successful completion of the task would be great, but incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success. In this context, we would like to thank all the other staff members, both teaching and non-teaching, who have extended their timely help and eased my task.

NasreenKhatoon	22AG1A66G9
M.Rishitha	22AG1A66G2
B.Mounika	22AG1A66D7
E.Kavya prasanna	23AG5A6620

DECLARATION

This is to certify that the work reported in the present project titled **"HARMFUL WEBSITE DETECTION"** is a record work done by us in the Department of CSE (Artificial Intelligence & Machine Learning), ACE Engineering College. No part of the thesis is copied from books/journals/internet and whenever the portion is taken, the same has been duly referred in the text; the reported are based on the project work done entirely by us not copied from any other source.

ABSTRACT

In detecting malicious websites, a common approach is the use of blacklists which are not exhaustive in themselves and are unable to generalize to new malicious sites. Detecting newly encountered malicious websites automatically will help reduce the vulnerability to this form of attack. In this study, we explored the use of ten machine learning models to classify malicious websites based on lexical features and understand how they generalize across datasets. Specifically, we trained, validated, and tested these models on different sets of datasets and then carried out a cross-datasets analysis. From our analysis, we found that K-Nearest Neighbour is the only model that performs consistently high across datasets. Other models such as Random Forest, Decision Trees, Logistic Regression, and Support Vector Machines also consistently outperform a baseline model of predicting every link as malicious across all metrics and datasets. Also, we found no evidence that any subset of lexical features generalizes across models or data sets. This research should be relevant to cyber security professionals and academic researchers as it could form the basis for real-life detection systems or further research work.

INDEX

CONTENTS **PAGE NO**

LIST OF FIGURES

LIST OF TABLES

LIST OF NOTATIONS / ABBREVATIONS

Abbreviation		Full Form
EDA	-	Exploratory Data Analysis
ML	-	Machine Learning
SMOTE	-	Synthetic Minority Over-sampling Technique
RAM	-	Random Access Memory
SSD	-	Solid State Drive
AWS	-	Amazon Web Services
ANN	-	Artificial Neural Network
IEEE	-	Institute of Electrical and Electronics Engineers
ICCSI	-	International Conference on Cyber-Physical Social Intelligence
ITM	-	Institute of Technology Management

CHAPTER 1
INTRODUCTION

Online services have become an irreplaceable part of today’s businesses, schools, banking, or personal lives. With their increasing popularity, the number of malicious websites is growing. A malicious website contains some unsolicited content with a purpose to gather sensitive data or install malware onto a user’s machine. Usually, some interaction from the user part is needed, but in the case of a drive by download, malware is installed automatically without asking for permission. Prevention from such attacks is complicated, as being cautious is sometimes not enough. Attackers can able to exploit vulnerabilities in web applications to insert malicious code without knowledge of the owner. The most common technique used to aim this problem is blacklisting. The list of ’bad URLs’ is compiled, and the browser prevents the user from accessing them. The major problem with this technique is blacklist incompleteness, URL only appear there if it was already reported before. To solve this issue, we need a more proactive solution that can spot and match patterns in malicious URLs. In this thesis, we focused on the problem of the detection of malicious URLs with machine learning techniques. We have chosen two libraries, SVM light and Tensor Flow, to train the prediction models. The goal is to determine if chosen algorithms can efficiently decide if the given URL on input is malicious or not. Based on their accuracy can be later used their estimated probability of URL being malicious as a relevant precondition for its further analysis.

1.1 Overview of Fraud in Financial Transactions

Fraudulent website detection involves using algorithms to analyze URLs, content, and user Behaviour to identify phishing, scam, and malicious sites, leveraging techniques like URL blacklists, content analysis for malicious keywords, and machine learning models for anomaly detection. Key tools include Virus Total API, Google Safe Browsing API, and WHOIS lookup, tackling challenges like evolving tactics of fraudsters and balancing detection accuracy with minimizing false positives, aiming to enhance cyber security and protect users from financial and identity theft risks online

1.2 Importance of Website Detection

Website detection is crucial as it safeguards users from malicious activities such as phishing, malware, and scams, ensuring their online safety and privacy. It protects businesses from reputational damage and financial losses by preventing fraud and unauthorized access. Compliance with regulations and industry standards is facilitated, maintaining trust and integrity in digital transactions. Detection technologies contribute to overall cyber security efforts by identifying and neutralizing threats before they can cause harm. They support a secure digital ecosystem, promoting confidence in online interactions and preserving the reliability of internet infrastructure.

1.3 Scope and Structure

The scope of website detection encompasses a broad range of activities aimed at safeguarding users and organizations from online threats. It involves:

1. Identifying Malicious Intent: Detecting phishing attempts, malware distribution, and fraudulent activities that exploit vulnerabilities.
2. Advanced Analysis Techniques: Utilizing machine learning, AI, and data analytics to analyze URLs, content, and user interactions for suspicious patterns.
3. Real-time Monitoring: Implementing systems for continuous monitoring of web traffic and immediate response to emerging threats.
4. Integration of Threat Intelligence: Leveraging APIs and databases to access real-time threat intelligence and maintain up-to-date security measures.
5. Compliance and Regulatory Alignment: Adhering to Cyber Security regulations and standards to protect sensitive data and ensure legal compliance.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview of Machine Learning in Harmful Website Detection

Machine learning (ML) has emerged as a powerful tool in detecting harmful Websites, offering superior performance over traditional rule-based systems. The ability of ML algorithms to learn from vast amounts of data and identify complex patterns has made them indispensable in the fight against financial fraud. Various ML techniques have been applied to harmful detection, including supervised learning, unsupervised learning, and semi-supervised learning.

2.2 Supervised Learning Techniques

Supervised learning techniques in machine learning involve training models on labelled data, where the algorithm learns to map input data (features) to output labels based on examples provided during training. Key supervised learning techniques include:

- **Linear Regression**: Predicts a continuous-valued output based on linear relationships between input features and output.
- **Decision Trees**: Hierarchical tree structures that recursively split data based on feature thresholds to make decisions.
- **Support Vector Machines (SVM)**: Finds an optimal hyper plane in a high-dimensional Space to separate data points into different classes.
- **Naive Bayes**: Probabilistic classifier based on Bayes' theorem with strong independence assumptions between features.
- **K-Nearest Neighbours (KNN)**: Classifies new data points based on the majority class of their nearest neighbours in the feature space.

2.3 Handling Imbalanced Data

Handling imbalanced data in harmful website detection is crucial for improving model performance and reducing bias towards the majority class:

- **Resampling Techniques**: Use oversampling (e.g., SMOTE) to increase minority class instances or under sampling to decrease majority class instances, balancing the dataset.
- **Class Weight Adjustment**: Assign higher weights to minority class examples during model training to penalize misclassifications more heavily.
- **Ensemble Methods**: Employ ensemble techniques like Balanced Random Forest or Easy Ensemble that are specifically designed to handle imbalanced datasets by adjusting sampling strategies or decision boundaries.
- Implementing these strategies helps improving overall detection capabilities and reducing the impact of skewed class distribution.

CHAPTER 3
SYSTEM REQUIREMENTS

3.1 Hardware Requirements

1.**Computing Power**:

- CPU: A multi-core processor with sufficient computational power to handle large datasets and perform model training efficiently.
- RAM: At least 8GB of RAM, preferably more for handling large-scale data processing and model training.

2.**Storage:**

- Hard Drive: A fast and reliable storage device with sufficient capacity (at least 500GB) to store transaction data, feature sets, and trained models.
- SSD: SSDs are recommended for faster data retrieval and processing speeds, especially during real-time transaction monitoring.

3.**Scalability:**

- Ensure the hardware infrastructure supports scalability to accommodate increasing transaction volumes and data growth over time.
- Cloud-based solutions (e.g., AWS, Azure) can provide scalable computing resources and storage options as needed.

3.2 Software Requirements

1. **Operating System:**

- Support for both Linux and Windows environments, depending on the organization's preferences and existing infrastructure.

2. **Programming Languages:**

- Python: Required for data pre-processing, feature engineering, model development, and deployment.

3. **Development Frameworks and Libraries**:

- Machine Learning Libraries:
- Pandas NumPy: For data manipulation and numerical computations.
- Matplotlib Seaborn: For data visualization to explore fraud patterns and model performance.

CHAPTER 4

SYSTEM ARCHITECTURE

A system architecture, also known as systems design, is the mathematical models that says describes the systems configuration, behaviour, and some aspects.

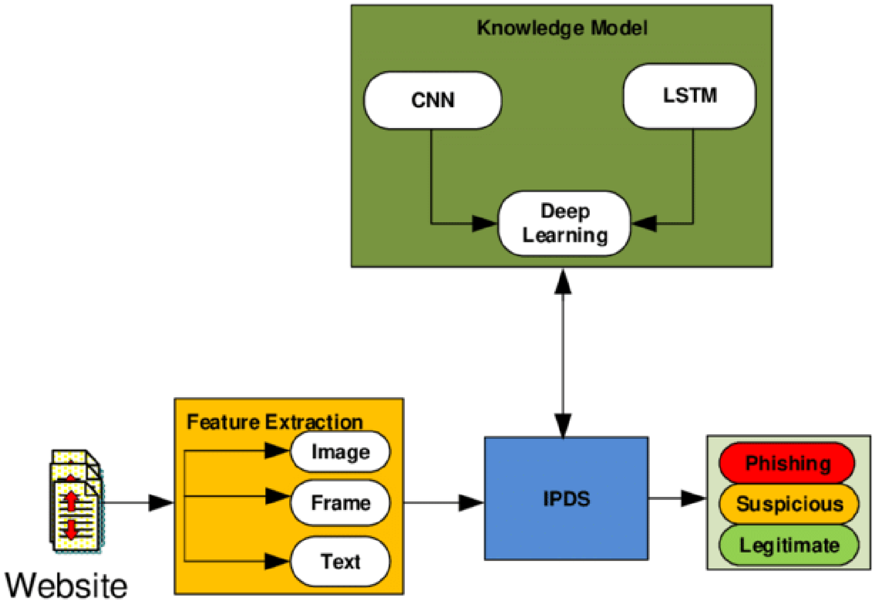


Fig. 4.1 System Architecture

1. MACHINE LEARNING ALGORITHMS

Machine learning algorithms are widely used for classifying websites based on features extracted from their content, metadata, and behaviour.

- **Supervised Learning:**
  1. **Logistic Regression:** Suitable for binary classification tasks.
  2. **Random Forest:** Ensemble method effective for handling high-dimensional data.
  3. **Support Vector Machines (SVM):** Useful for separating data points with a clear margin of separation.

4. **Gradient Boosting Machines (GBM):** Builds strong learners sequentially, improving model performance.

- **Unsupervised Learning:**
  1. **Clustering (e.g., K-means):** Groups websites based on similarity in features, identifying outliers.
  2. **Anomaly Detection:** Detects unusual behavior or patterns that deviate from normal websites.

2. FEATURE EXTRACTION TECHNIQUES

Extracting meaningful features from website data is crucial for effective classification:

Text-Based Features:

1. **Term Frequency-Inverse Document Frequency (TF-IDF):** Weighs terms based on their frequency in a document relative to the entire corpus.
2. **N-grams:** Captures sequences of words or characters to understand context and patterns.
3. **Metadata and Domain Features:**
4. **Domain Age:** Older domains may be less likely to be harmful.
5. **SSL Certificate:** Presence and validity indicate a secure connection.
6. **IP Reputation:** Evaluate IP addresses associated with the website for known malicious activity.

3. RULE BASED FILTERS

These filters apply predefined rules or heuristics to quickly identify potential harmful websites:

1. **Blacklists:** Compare website URLs or domains against known lists of malicious sites.
2. **Phishing Keywords:** Detect suspicious content such as phishing attempts using keyword analysis.
3. **URL Analysis:** Check for URL structure anomalies or patterns associated with malicious websites.

4. BEHAVORAL ANALYSIS

Analyzing website behaviour and interactions with users can also reveal malicious intent:

1. **Clickstream Analysis:** Track user navigation patterns to identify suspicious redirects or pop-ups.
2. **Session Analysis:** Monitor user sessions for abnormal behavior such as rapid page visits or erratic clicks.

5. **INTEGRATION AND OPTIMIZATION**

1. **API Integration:** Integrate with threat intelligence APIs or services for real-time updates on known malicious sites.
2. **Continuous Learning:** Implement feedback loops to continuously update models based on new data and evolving threats.

Performance Optimization: Use techniques like feature selection, dimensionality reduction, and model tuning to improve accuracy and efficiency

CHAPTER 5

SYSTEM DESIGN

5.1 Flow Chart

The flow chart serves as a visual roadmap for developing and understanding the workflow involved in detecting harmful websites, integrating advanced technologies to enhance cyber security measures effectively.

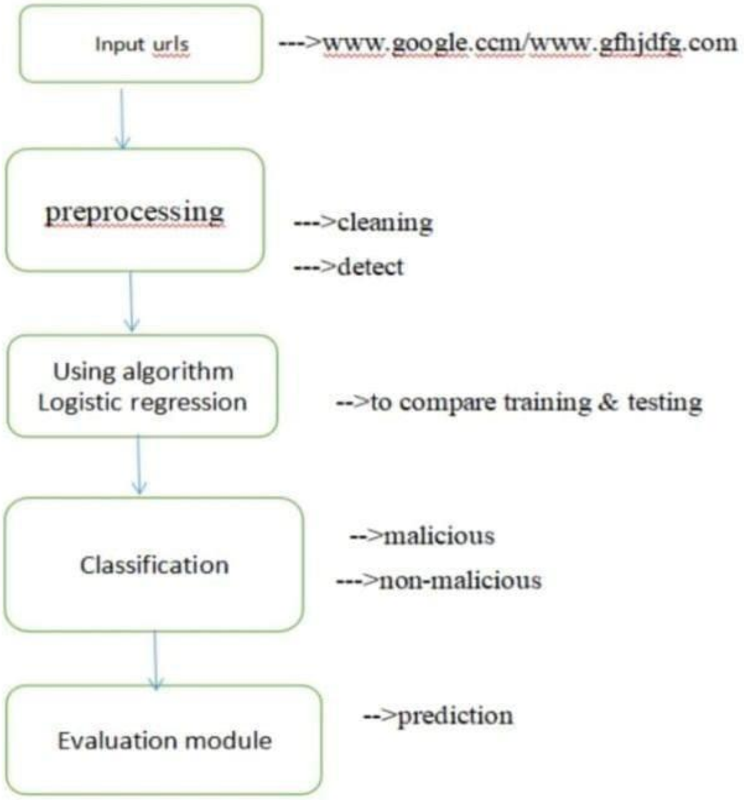


Fig. 5.1 Flow Chart for Harmful Website Detection

5.2 Class Diagram

A class diagram would be used to describe, characterize, and record numerous elements of a system, and perhaps to create executable code for a computer system. The attributes and procedures of a class, and perhaps even the system's limitations, are portrayed in a class diagram.

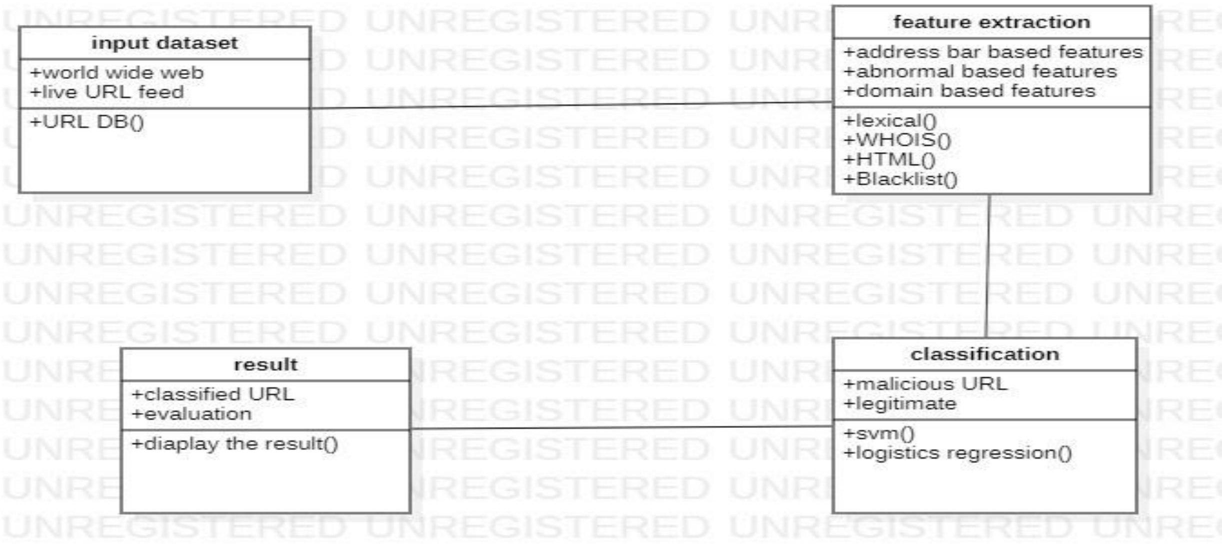


Fig. 5.2 Class Diagram

5.3 Use Case Diagram

A use cases diagram is the type of behaviour diagrams described by generated from the Use-case studies in the Unified Modelling Language (UML). It is the aim is to provide a graphical representation of a system function in common terms of actor, priorities (represents as the use cases) any dependency between these use cases.

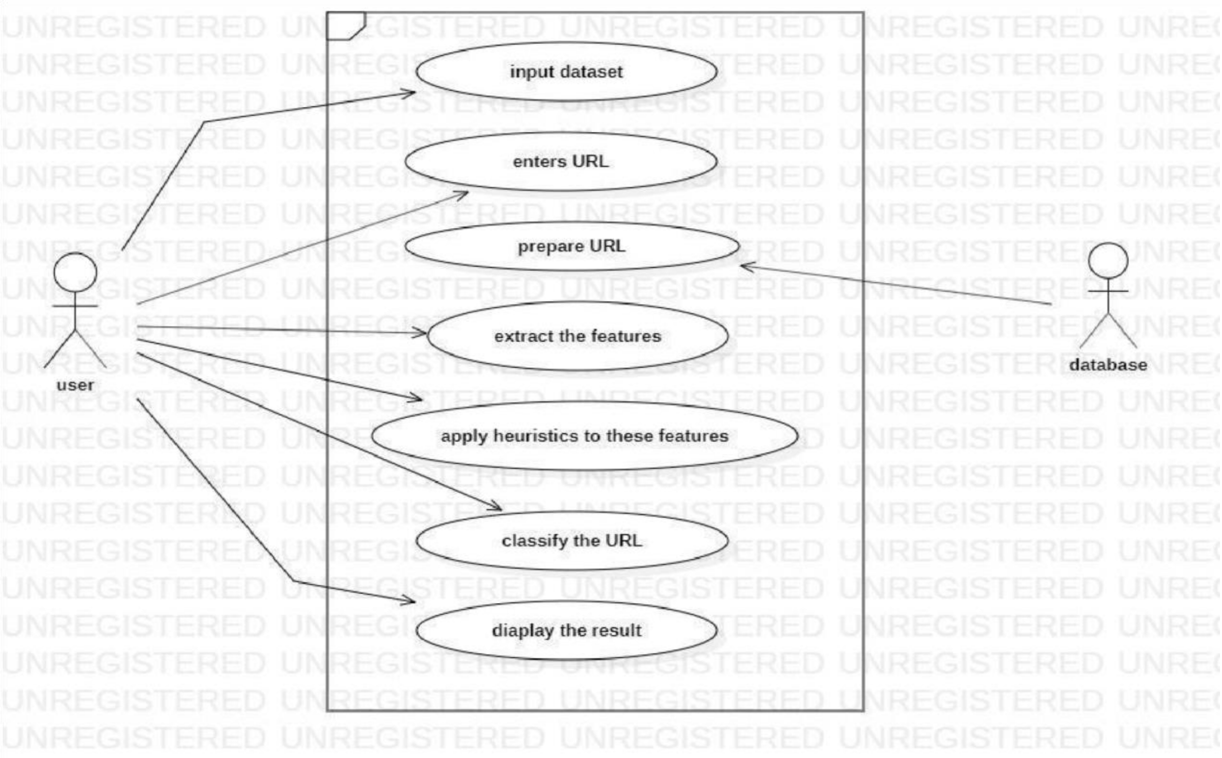


Fig. 5.3 Use Case Diagram

5.4 Sequence Diagram

In the Unified Modelling (UML), a sequence is the type that the activity diagrams that depicts however processes communicate one another and in which else order. It's a Line Chart Map construct. Case diagrams, scatter graph, and timing graphs are all terms used to describe sequence diagrams.

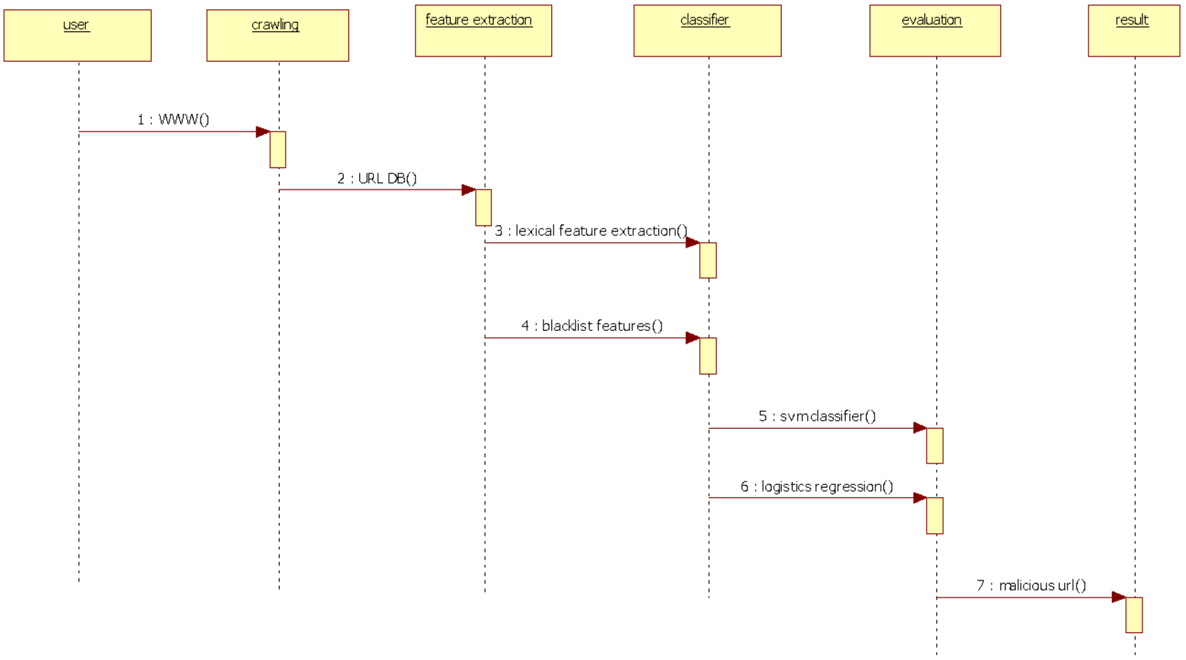


Fig. 5.4 Sequence Diagram

5.5 Activity Diagram

The activity diagram provides a structured overview of the sequence of activities involved in detecting and mitigating harmful websites, integrating advanced technologies and methodologies to enhance cyber security measures effectively.

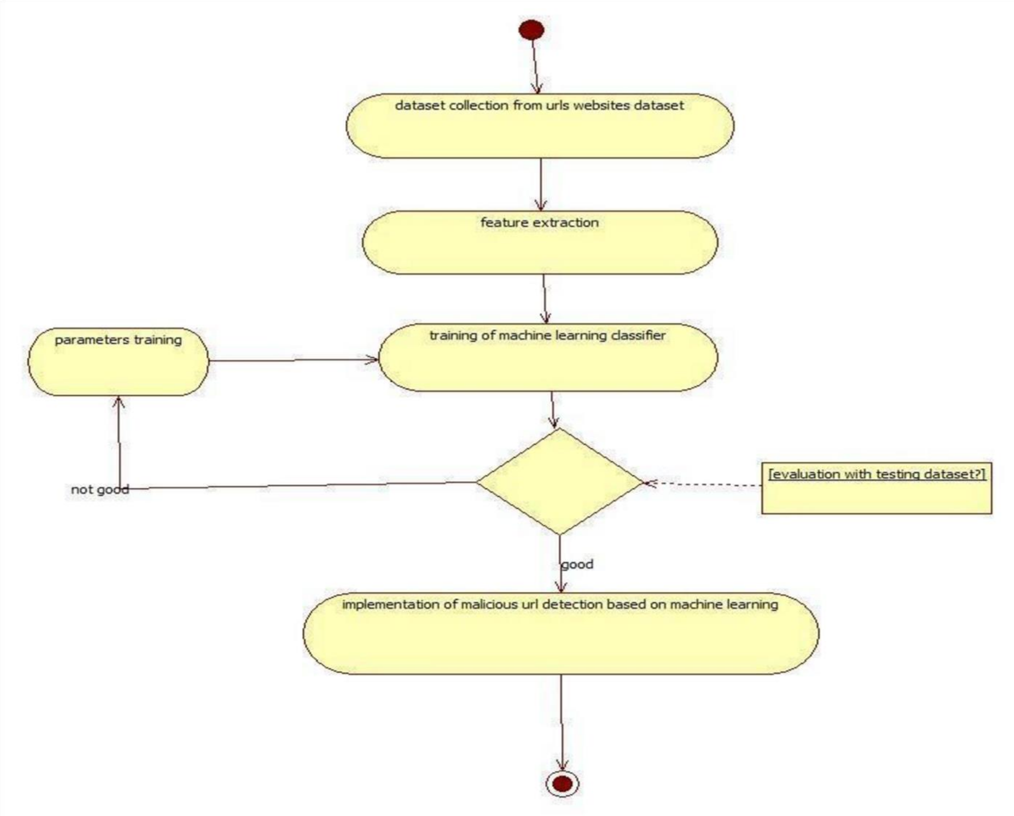

dataset collection from urls websites dataset
feature extraction
parameters training
training of machine learning classifier
not good
[evaluation with testing dataset?]
good
implementation of malicious url detection based on machine learning

Fig. 5.5 Activity Diagram

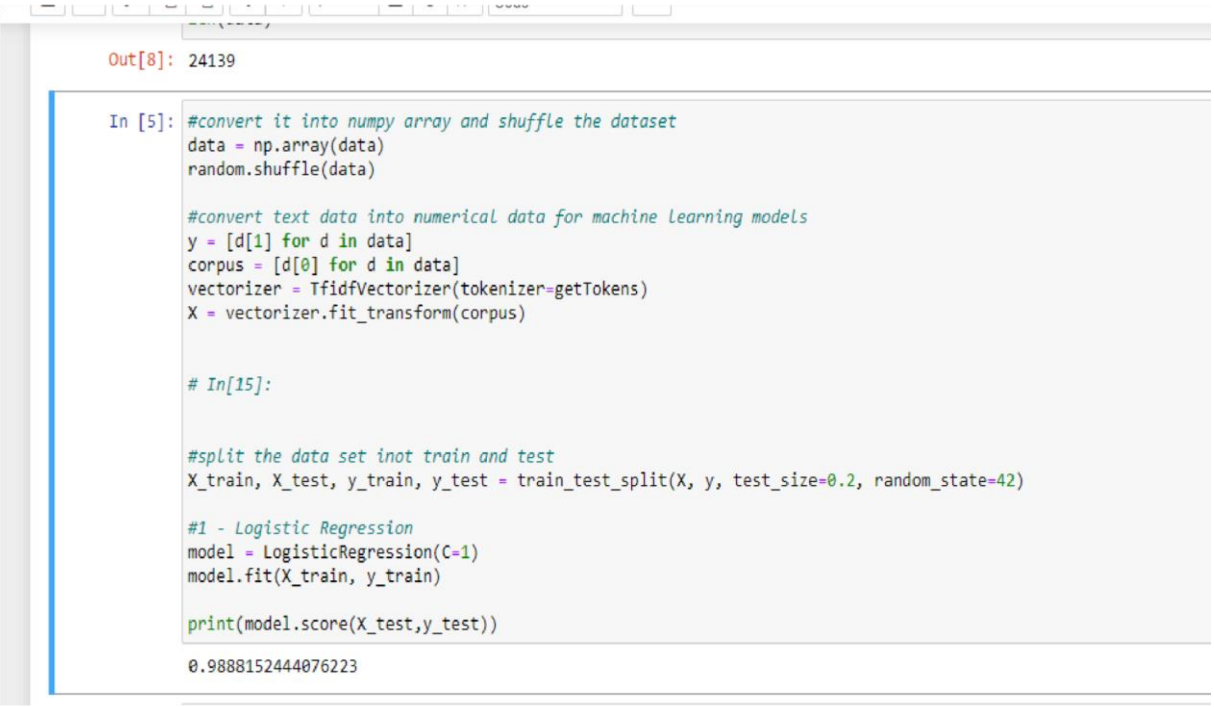
CHAPTER 6
IMPLEMENTATION

6.1 Data Set

index	having_IPh	URLURL_Le	Shortining_	having_At_	double_sla	Prefix_Suff	having_Sul	SSLfinal_St	Domain_re	Favicon	port	HTTPS_tok	Request_U	URL_of_An	Links_in_ta	SFH
1	-1	1	1	1	-1	-1	-1	-1	-1	1	1	-1	1	-1	1	-1
2	1	1	1	1	1	-1	0	1	-1	1	1	-1	1	0	-1	-1
3	1	0	1	1	1	-1	-1	-1	-1	1	1	-1	1	0	-1	-1
4	1	0	1	1	1	-1	-1	-1	1	1	1	-1	-1	0	0	-1
5	1	0	-1	1	1	-1	1	1	-1	1	1	1	1	0	0	-1
6	-1	0	-1	1	-1	-1	1	1	-1	1	1	-1	1	0	0	-1
7	1	0	-1	1	1	-1	-1	-1	1	1	1	1	-1	-1	0	-1
8	1	0	1	1	1	-1	-1	-1	1	1	1	-1	-1	0	-1	-1
9	1	0	-1	1	1	-1	1	1	-1	1	1	-1	1	0	1	-1
10	1	1	-1	1	1	-1	-1	1	-1	1	1	1	1	0	1	-1
11	1	1	1	1	1	-1	0	1	1	1	1	1	-1	0	0	-1
12	1	1	-1	1	1	-1	1	-1	-1	1	1	1	1	-1	-1	-1
13	-1	1	-1	1	-1	-1	0	0	1	1	1	-1	-1	-1	1	-1
14	1	1	-1	1	1	-1	0	-1	1	1	1	1	-1	-1	-1	-1
15	1	1	-1	1	1	1	-1	1	-1	1	1	-1	1	0	1	1
16	1	-1	-1	-1	1	-1	0	0	1	1	1	1	-1	-1	0	-1
17	1	-1	-1	1	1	-1	1	1	-1	1	1	-1	1	0	-1	-1
18	1	-1	1	1	1	-1	-1	0	1	1	-1	1	1	0	-1	-1
19	1	1	1	1	1	-1	-1	1	1	1	1	-1	-1	0	-1	-1
20	1	1	1	1	1	-1	-1	1	-1	1	1	1	1	0	0	-1
21	1	0	-1	1	1	-1	0	1	-1	1	1	1	1	0	0	-1
22	1	0	1	1	1	-1	0	1	1	1	1	-1	-1	0	-1	-1
23	1	1	1	1	1	-1	-1	-1	-1	1	1	-1	1	0	0	-1
24	1	1	1	1	1	-1	1	0	-1	1	1	1	1	0	0	-1

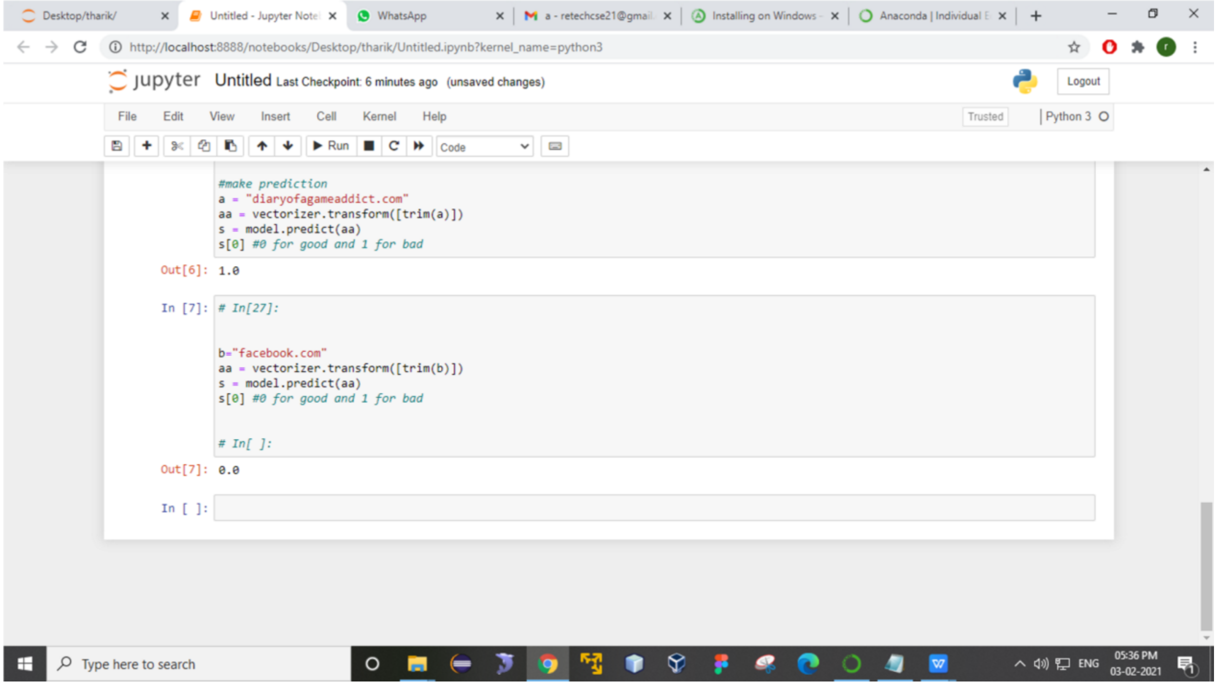
Table 6.1. Data Set

6.2FINDING ACCURACY



```
Out[8]: 24139

In [5]: #convert it into numpy array and shuffle the dataset
        data = np.array(data)
        random.shuffle(data)

        #convert text data into numerical data for machine learning models
        y = [d[1] for d in data]
        corpus = [d[0] for d in data]
        vectorizer = TfidfVectorizer(tokenizer=getTokens)
        X = vectorizer.fit_transform(corpus)


        # In[15]:


        #split the data set inot train and test
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

        #1 - Logistic Regression
        model = LogisticRegression(C=1)
        model.fit(X_train, y_train)

        print(model.score(X_test,y_test))

        0.9888152444076223
```

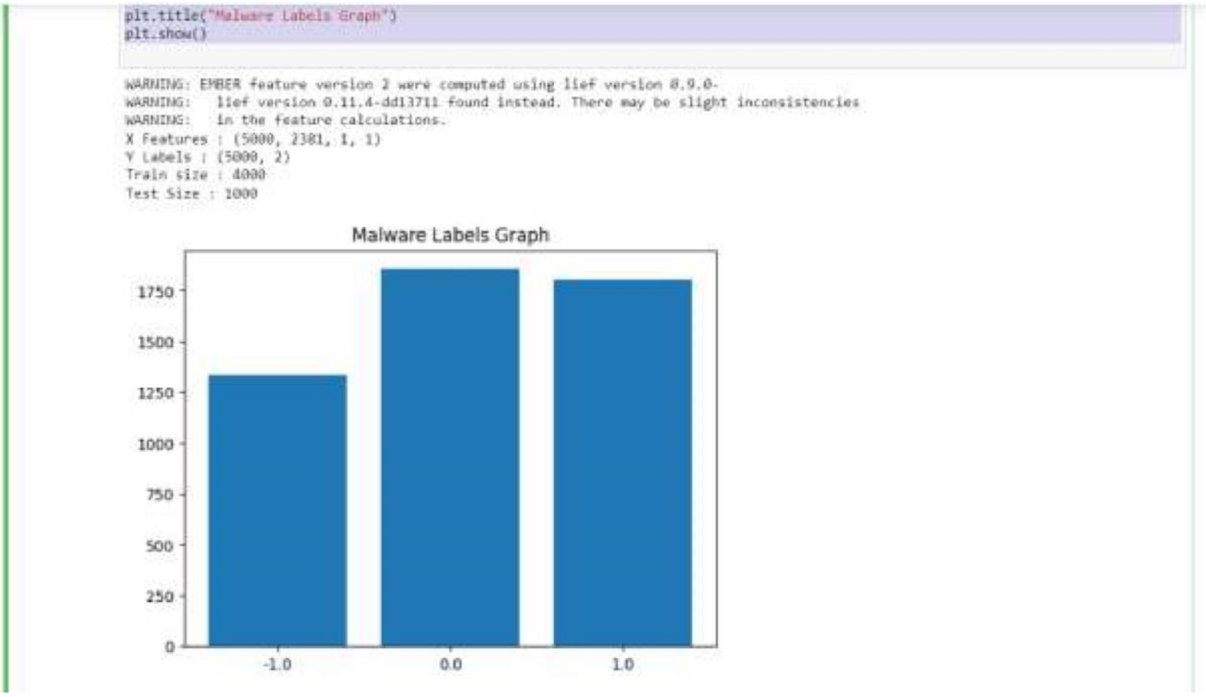
6.3DETECTING MALICIOUS URL



Desktop/tharik/
Untitled - Jupyter Note
WhatsApp
a - retechcse21@gmail
Installing on Windows
Anaconda | Individual E
http://localhost:8888/notebooks/Desktop/tharik/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: 6 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code
Type here to search
ENG
05:36 PM
03-02-2021

```
        #make prediction
        a = "diaryofagameaddict.com"
        aa = vectorizer.transform([trim(a)])
        s = model.predict(aa)
        s[0] #0 for good and 1 for bad
Out[6]: 1.0

In [7]: # In[27]:


        b="facebook.com"
        aa = vectorizer.transform([trim(b)])
        s = model.predict(aa)
        s[0] #0 for good and 1 for bad


        # In[ ]:
Out[7]: 0.0

In [ ]:
```

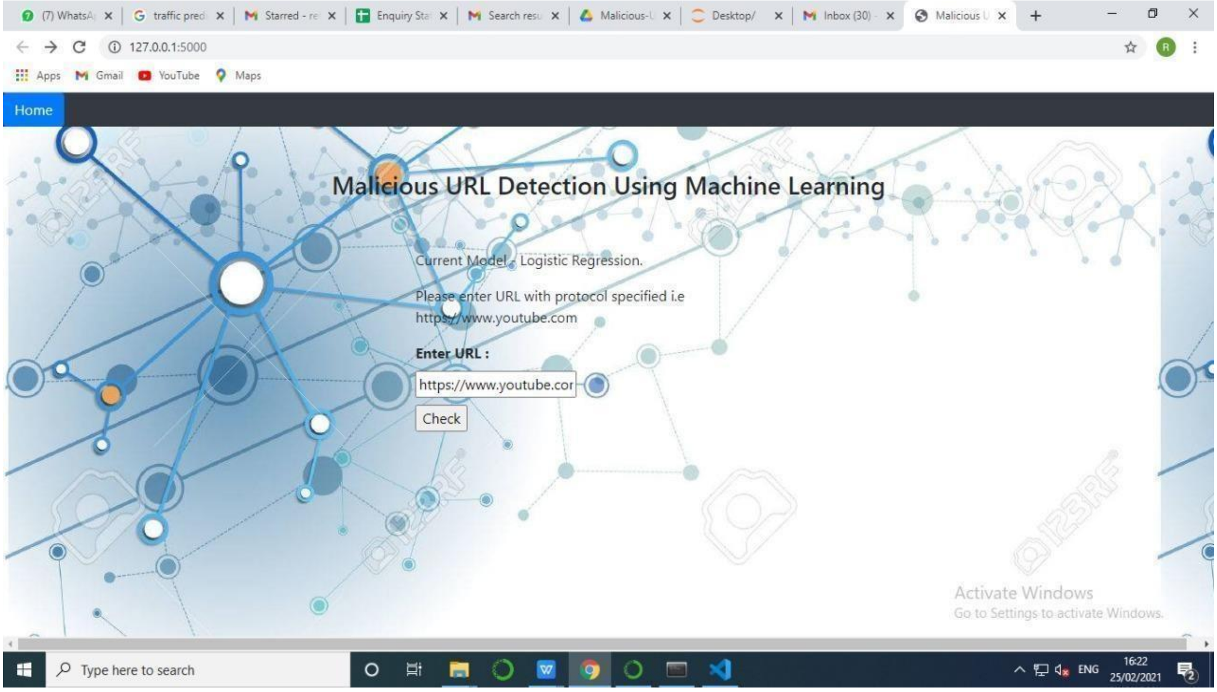
6.4 Malware label graph



```
plt.title("Malware Labels Graph")
plt.show()
```

```
WARNING: EMBER feature version 2 were computed using lief version 0.9.0-
WARNING:    lief version 0.11.4-dd13711 found instead. There may be slight inconsistencies
WARNING:    in the feature calculations.
X Features : (5000, 2381, 1, 1)
Y Labels : (5000, 2)
Train size : 4000
Test Size : 1000
```

6.5WEBPAGE USER TO CHECK



Malicious URL Detection Using Machine Learning

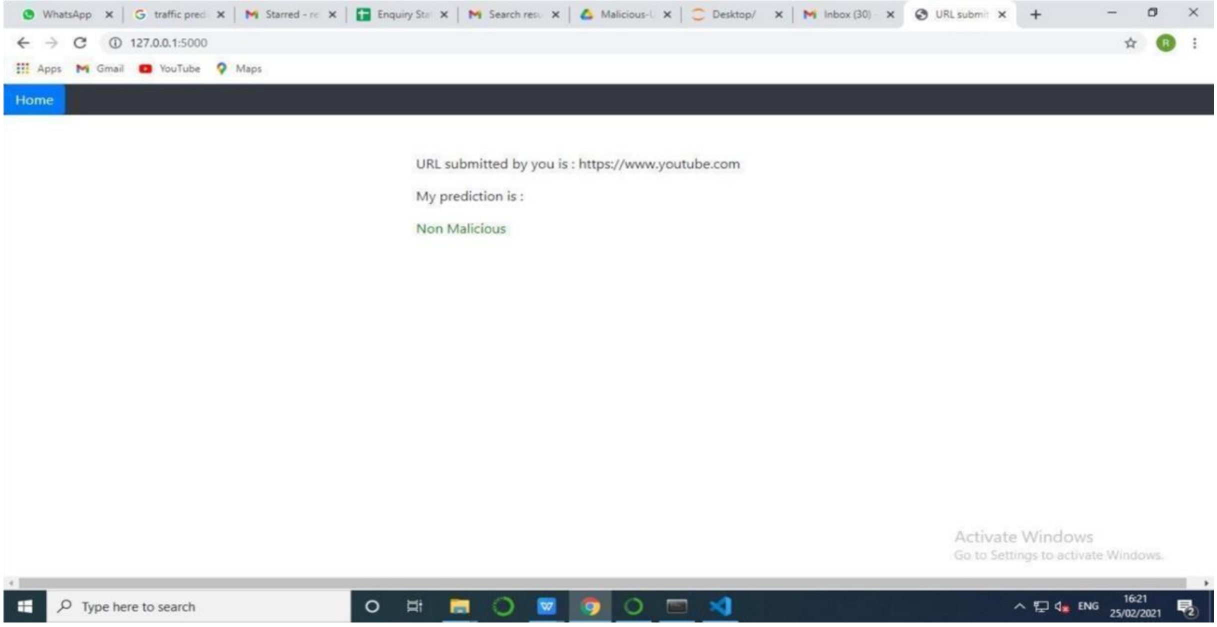
Current Model - Logistic Regression.

Please enter URL with protocol specified i.e https://www.youtube.com

Enter URL :

https://www.youtube.cor

6.6 OUTPUT PREDICTION



URL submitted by you is : https://www.youtube.com

My prediction is :

Non Malicious

CHAPTER 7

TESTING

The aim of research is to found mistakes. Testing was the methodology of attempting to find any possibility flaws and weaknesses in the work object. This allows you for testing the functional of individual parts, sub assembly, assemblies, and a completed project. There is also difference kinds of tests. Each and every test form is designed to meet a particular research need.

7.1 TYPES OF TESTS

1. **UNIT TESTING**: Unit checking requires generating test cases to ensure that the software's internal logic is running correctly and that program inputs result in correct outputs.

2. **INTEGRATION TESTING**: This check is used for seeing how two or more software modules will function together as a single application.

3. **FUNCTIONAL TESTING:** Functionality checks demonstrate that the features being evaluated are accessible in accordance with the market and operational specifications, device documents, user manuals.

4. **SYSTEM TESTING:** It checks a configuration to ensure that the outcomes are known and predictable. System testing incorporates method parameters and flows, with an emphasis on pre-driven process connections and integration points.

5. **WHITE BOX TESTING**: This is the method of software tester in which the software testing is familiar with a software's inside workings, configuration, languages, or at very least it is purpose.

7.2TEST OBJECTIVES

a. Both field entry must behave correctly.

b. The identified relation must be used to trigger the sites.

c. There must be no delays in the entry screen, calls, or replies.

Features to be tested:

1. Check that the submissions are in the proper format.
2. There should be no duplicate entries allowed.
3. All links should lead to the correct page for the user.

Test Results:

Every one of the above-mentioned test case were successful. There were no flaws found.

Acceptance Testing

Acceptance by the users Testing is an important aspect of every project, and it necessitates active input from the end user. It also guarantees that the device satisfies the operating specifications.

Test Results: All the above-mentioned test inputs were successful. There were no flaws found.

CHAPTER 8

CONCLUSION

8.1 Further Enhancements

The system can be further enhanced by exploring advanced machine learning techniques, incorporating additional data sources, and implementing real-time analytics for more advanced website detection. Continuous monitoring and periodic model retraining will be essential to adapt to evolving fraud patterns and maintain the system's effectiveness.

Overall, the project demonstrates the potential of logistic regression and machine learning in enhancing the security and integrity of website detection, providing a foundation for more sophisticated fraud detection solutions in the future.

8.2 Conclusion

Detecting harmful websites is crucial for maintaining online safety. By employing advanced algorithms and machine learning models, we can effectively identify and block websites that pose threats such as malware distribution, phishing scams, or illicit content. This proactive approach not only safeguards users from potential harm but also promotes a secure and trustworthy internet environment. Continuous refinement and collaboration within the cyber security community are essential to stay ahead of evolving threats, ensuring robust protection for all users online.

Many cyber security applications depend on malicious URL identification, and machine learning techniques are obviously a promising path. We conducted a thorough and ordered analysis of Malicious Detection using AI approaches in the work. We provided the methodical description of Malicious detection from an AI standpoint, followed by nitty gritty information.

CHAPTER 9

REFERENCES

1. Islam R, Abawajy J (2013) A multi-tier phishing detection and filtering approach. J Netw Comput Appl 36:324–335.

2. Abdelhamid N, Ayesh A, Thabtah F (2014) Phishing detection based associative classification data mining. Science-Direct 41:5948–5959.

3.Li Y, Xiao R, Feng J, Zhao L (2013) A semi-supervised learning approach for detection of phishing webpages. Optik 124:6027– 6033.

4. Chen KT, Chen JY, Huang CR, Chen JY (2009) Fighting phishing with discriminative key point features of webpages. IEEE Internet Comput 13:56–63.

5. Li Y, Xiao R, Feng J, Zhao L (2013) A semi-supervised learning approach for detection of phishing webpages. Optik 124:6027– 6033.

CHAPTER 10
APPENDIX

PYTHON

In the project, Python serves as the primary programming language for its versatility and extensive libraries suited for machine learning tasks. It facilitates data pre-processing, model development (like Logistic Regression for fraud detection), and evaluation. Python's ecosystem, including libraries like Pandas for data manipulation and Scikit-learn for machine learning, supports efficient development and testing. Additionally, Python's readability and community support enhance collaboration and maintainability across the project lifecycle.

MACHINE LEARNING

In the project focused on Harmful website detection, Machine Learning (ML) techniques, specifically Logistic Regression, are employed for their effectiveness in binary classification tasks. Logistic Regression models are trained on historical transaction data to learn patterns indicative of fraudulent behaviour. The ML models are trained iteratively to improve accuracy and adapt to evolving fraud detections.

LOGISTIC REGRESSION

In the project, Logistic Regression is chosen for its ability to model the probability of Harmful Website Detection based on input features. It applies a sigmoid function to linearly combine feature values, converting them into probabilities. Regularization techniques like L2 regularization may be employed to prevent over fitting. The model is trained using gradient descent to optimize coefficients, maximizing classification accuracy for Harmful Website Detection.