ISSN No:-2456-2165

# Web Phishing Detection using Machine Learning

[1]Seemantula Nischal; [2]Bhunesh.K; [3]Sudhish Reddy D
[1,2,3] Student, Department of CSE, R.M.D. Engineering College, Kavaraipettai

[4]Seemantula Namratha
[4]Student, Department of IT, SSN College of Engineering

ISSN No:-2456-2165


## ABSTRACT

**This research paper outlines a methodology to determine URL legitimacy and detect phishing attempts. Python modules like who is, socket, re, IP address, and BeautifulSoup are employed to extract features such as IP address, URL length, domain name, subdomains, and favicon presence. These values are stored as a list and used to train classifiers. Kernel SVM, KNN, Random Forest, and decision tree classifiers are implemented.**

**The Kernel SVM classifier (sklearn.svm.SVC) with the "rbf" kernel handles nonlinearity. Decision tree classification is based on the "entropy" criterion using the sklearn.tree module. Random Forest combines multiple decision trees, with final classification based on majority voting.The paper presents a user-friendly UI design for websites focused on phishing detection. These websites utilize machine learning algorithms to assess URL authenticity and provide user feedback. Integration of frameworks like Bootstrap and Particles.js enhances visual appeal and user experience.**

**The machine learning algorithm analyzes website content, structure, and other factors to determine legitimacy, presenting results with a legitimacy percentage and associated risks.The study explores Flask, a flexible Python web framework for rapid development of online applications. Flask provides built-in routing, templating, and supports machine learning integration, enabling user input and result retrieval. It simplifies machine learning model deployment as web services through APIs, facilitating integration with other applications.**

**Additionally, the research emphasizes Anaconda as an essential tool for data science and machine learning projects. Anaconda offers efficient package management, simplifying installation, removal, and updating of required libraries. It provides a comprehensive set of tools for the complete data science workflow, including data exploration, cleaning, model construction, and deployment. Integration with Jupyter Notebook further enhances its capabilities.**

**In conclusion, this research paper presents a comprehensive approach for URL legitimacy assessment and phishing detection, combining Python modules, machine learning classifiers, user-friendly UI design, Flask framework, and the benefits of Anaconda for data science and machine learning projects**

ISSN No:-2456-2165

# TABLE OF CONTENT

ISSN No:-2456-2165

# CHAPTER ONE
# INTRODUCTION

Artificial intelligence (AI) is a field of study that focuses on creating, expanding, and enhancing human knowledge through the development of theories, strategies, procedures, and applications. Machine learning (ML), a branch of AI, is closely related to computational measurements and involves using computers to make predictions. ML is closely intertwined with scientific progress and informs the discipline about methodologies, hypotheses, and areas of application. Data mining, although distinct from ML, is sometimes integrated with it and is often referred to as unsupervised learning. Unsupervised ML can be used to learn and establish patterns for different entities, enabling the detection of significant anomalies.

Cybersecurity refers to a range of technologies and practices aimed at protecting computers, networks, programs, and data from attacks, unauthorized access, modification, or destruction. It encompasses network protection systems and computer protection systems, which may include firewalls, antivirus software, and intrusion detection systems (IDS). IDSs help identify and determine unauthorized network behavior, such as usage, replication, modification, and destruction.

Intrusion detection systems employ different types of network analysis techniques, including misuse-based network analysis, anomaly-based network analysis, and hybrid network analysis.

- Misuse-based detection techniques aim to differentiate known attacks by utilizing specific characteristics associated with those attacks.

- Anomaly-based methods analyze the normal behavior of a system and identify anomalies as deviations from this normal behavior

Hybrid detection combines the capabilities of anomaly detection and abuse detection to enhance the effectiveness of identifying known intrusions while minimizing false positives from unknown attackers.

The application of machine learning (ML) technologies in the field of cybersecurity is experiencing rapid growth, as depicted in Figure 1.1. ML provides a powerful solution against zero-day attacks by categorizing IP traffic and isolating malicious data for effective intrusion detection. Ongoing research focuses on leveraging measured traffic parameters and machine learning approaches to strengthen cybersecurity measures.

The term "phishing" was coined in 1987 and refers to a form of online theft that aims to obtain personal information and identity. Phishing attacks often involve the creation of deceptive websites that closely resemble legitimate ones, making it challenging to discern their fraudulent nature. By gaining the trust of unsuspecting users, attackers deceive them into sharing confidential and identifying information. As online transactions, such as bill payments and money transfers, have become increasingly prevalent [12], the ability to recognize and identify fraudulent websites is of utmost importance. According to data from the Anti-Phishing Working Group, there were a total of 647,592 different phishing sites recorded as of September 2018 [13]. Once attackers gain access to user passwords, they can easily carry out malicious activities.

Due to the rise in phishing attacks, various solutions have been proposed to address this issue. Several methodologies have been developed to establish frameworks aimed at safeguarding against phishing attacks. Detection techniques for phishing attacks encompass blacklisting, fuzzy rule-based approaches, whitelisting, machine learning-based methods, heuristic approaches, and image-based techniques [14][15]. Numerous studies [16][17][18] discuss a range of strategies and tactics for detecting different types of phishing attacks [19][20][21]. Phishing websites often appear authentic, making it challenging for individuals to differentiate them from legitimate sites. Various web browsers incorporate anti-phishing measures to mitigate this risk [22].

➢ *Motivation*

A plethora of anti-phishing techniques are available to assist in the protection against phishing websites. Mozilla Firefox, Safari, and Google Chrome utilize the Google Safe Browsing (GSB) [13] service to proactively block phishing websites. Other widely used products, including McFee Site Advisor, Quick Heal, Avast, and Netcraft, offer extensive protection measures. GSB employs a blacklist strategy to analyze URLs. However, a notable limitation of GSB is its inability to detect phishing websites if the blacklist is not regularly updated. On the other hand, Netcraft labels a website as phishing only if it is explicitly blacklisted. The warning notification is displayed when the user selects the appropriate button to obtain the risk rating. The risk arises when the user fails to review the rating or proceeds despite the warning. Some software solutions, such as QuickHeal and Avast, provide online security against security breaches. During testing, the functionality of Avast antivirus software was evaluated. However, the Avast browser failed to identify a suspicious URI that was successfully detected by both Netcraft and GSB.

➢ *Problem Statement*

The problem arises from a thorough examination and analysis of the machine learning-based classification method used for identifying phishing websites. Our objective is to develop a system that can:

- *Effectively and swiftly classify websites as either legitimate or phishing.*
- *Reduce the time and cost involved in the detection process.*

➢ *Aim and Objectives*

The project aims to accomplish the following objectives:

- *Explore different automated techniques for phishing detection.*
- *Determine and define appropriate machine learning approaches.*
- *Select a suitable dataset that addresses the problem statement.*
- *Utilize relevant algorithms to devise a solution for combating phishing attacks.*

➢ *Scope*

The project specifically focuses on employing machine learning (ML) methods for network analysis and intrusion detection, with a particular emphasis on detecting phishing website attacks.

➢ *Challenges*

The project entails several challenges, including:

- *Identifying an appropriate dataset that aligns with the research goals.*
- *Conducting thorough feature extraction, necessitating a comprehensive understanding of various modules and achieving the desired outcomes from each module.*

➢ *Organization of Thesis*

Chapter 1 provides an overview of the utilization of machine learning in cybersecurity. It outlines the problem statement, objectives, scope, and challenges encountered during the project. Chapter 2 presents a comprehensive review of the relevant literature.

Chapter 3 delves into the research and exploration of phishing attacks and their detection using machine learning approaches. It provides an extensive overview of previous studies conducted in this domain, highlighting their contributions and limitations.

Chapter 4 outlines the specific software and hardware requirements needed for the system. It discusses the fundamental prerequisites for the project and provides insights into the Python modules utilized in the implementation process.

The design of the system is elaborated in Chapter 5, which includes the representation of the system through architecture diagrams, data flow diagrams, and activity diagrams. These visual representations offer a comprehensive understanding of the system's functionality from various perspectives, including the system itself, the user, and its runtime behavior.

The implementation of the project is thoroughly addressed in Chapter 6. This chapter encompasses essential aspects such as the selection and utilization of the dataset, the step-by-step implementation process, and the application of relevant classifiers.

Chapter 7 focuses on the analysis of test cases, examining the comparison between predicted and actual outputs to validate the accuracy and effectiveness of the system.

Chapter 8 presents the obtained results and provides an overview of the project's environmental setup, offering insights into the performance and efficiency of the system.

Finally, in Chapter 9, the project is concluded with a summary of the key findings. Additionally, potential future improvements and advancements are outlined, indicating areas for further enhancement and development of the system.

ISSN No:-2456-2165

# CHAPTER TWO
# LITERATURE REVIEW

- *Detecting Phishing Websites Using Machine Learning*
  Publication Year: 2019 Authors: Amani Alswailem, Bashayr Alabdullah, Norah Alrumayh, Dr. Aram Alsedrani Published In: IEEE

Phishing attacks pose a significant threat as they employ social engineering techniques and malware to deceive individuals and organizations. The attackers send emails or texts containing URLs that appear genuine but are intended to trick recipients into revealing sensitive information. Machine learning techniques have been employed to detect and mitigate these phishing links. The primary objective of this literature review is to increase awareness about phishing attacks and explore preventive strategies. Given the alarming frequency of phishing emails being sent daily, it is essential to focus on prevention through user education, technological solutions, and established protocols.

- *Phishing Website Detection Based on Machine Learning: A Survey*
  Publication Year: 2020 Authors: Smt. Meeenu, Charu Singh Published In: ICACCS

Phishing attacks, characterized by their use of social engineering tactics and malware, represent a form of cybercrime where individuals and organizations are lured into divulging sensitive information. Typically, phishers employ deceptive email or text messages containing web URLs that mimic legitimate sites but are designed to extract confidential data. Given the pervasive nature of phishing attempts, it becomes a challenge for individuals and organizations to identify and counteract them effectively. In this context, the utilization of machine learning techniques has gained prominence as a means to enhance phishing website detection. This survey aims to provide a comprehensive overview of the existing research in this domain, exploring various machine learning approaches and methodologies employed in combating phishing attacks.

- *Machine Learning Techniques for Detecting Phishing Websites: Reviewing Promises and Challenges*
  Publication Year: 2020 Authors: Eman Abdelfattah, Ismail Keshta, Ammar Odeh Published In: IEEE

This scholarly article addresses the issue of detecting phishing websites, which aim to deceive internet users and obtain sensitive information through social engineering and malware. The authors investigate the application of machine learning techniques such as Random Forest, Support Vector Machine, and Naive Bayes for identifying phishing attempts. However, they highlight that deep learning approaches have shown better performance in this area. The challenges faced by machine learning methods, including overfitting and limited training data, are also discussed. The research emphasizes the importance of user education and proposes an automated approach to detect phishing websites effectively.

- *Machine Learning-based URL Analysis for Phishing Website Detection*
  Publication Year: 2020 Authors: Mehmet Korkmaz, Banu Diri, Ozgur Korn Sahingoz Published In ICCCNT

This scholarly paper examines the growing trend of conducting real-world activities through mobile devices, which has led to security vulnerabilities, including phishing attacks. Various techniques for detecting phishing websites have been developed, with a particular focus on machine learning-based anomaly detection due to its dynamic nature. The authors propose a machine learning-based system that analyzes URLs using eight different algorithms. The effectiveness of the system is evaluated by comparing its results with previous work using three different datasets, demonstrating a high success rate in detecting phishing attempts.

- *Phishing Detection System Using Machine Learning*
  Publication Year: 2019 Authors: Che-Yu Wu, Cheng-Chung Kuo, Chu-Sing Yang Published In: ICEA

This research article investigates the issue of phishing attempts on the internet, which exploit human vulnerabilities to trick users into revealing sensitive information. The authors present a detection method based on analyzing URLs, employing string similarity calculations and the Support Vector Machine technique in machine learning. The aim is to achieve accurate detection of unknown phishing pages with a low rate of false positives.

- *Review of Phishing Detection Approaches*
  Publication Year: 2020 Author: Mouhammd Alkasassbeh AlMaha Abu Zuraiq Published In: IEEE

This comprehensive review paper explores the problem of phishing attempts on the internet and the potential financial losses they can cause for individuals. The study examines various strategies for detecting phishing, including content-based, heuristic-based, and fuzzy rule-based methods. Additionally, the article compares the performance of deep learning-based approaches with traditional machine learning methods for website recognition.

- *Detection of Phishing Websites Using Machine Learning*

Year of publishing: 2020 Authors: Mohammed Hazim Alkawaz, Stephanie JoanneSteven, Asif Iqbal Hajamydeen Published In: IEEE

This study presents a system for detecting phishing websites that aims to raise awareness among users about the potential risks associated with accessing such websites. The system employs machine learning techniques and sends email and pop-up notifications to users when they attempt to visit blacklisted or phishing websites. The main objective is to prevent users from falling victim to phishing attacks and inadvertently disclosing sensitive information. Additionally, the system can be utilized for identification and authentication purposes to enhance overall cybersecurity measures against phishing attempts.

ISSN No:-2456-2165

# CHAPTER THREE
# FUNDAMENTALS

In the domain of machine learning and statistics, classification algorithms play a crucial role in supervised learning tasks, enabling computers to learn from input data and make accurate predictions or categorizations. This chapter focuses on exploring different classification algorithms specifically designed for detecting phishing URLs.

- *Logistic Regression*

Logistic regression, also known as a logit model, is a widely used statistical technique for classification and predictive analytics. It assesses the likelihood of a specific event occurring based on a set of independent variables. By transforming the odds of success into probabilities, logistic regression restricts the dependent variable to values between 0 and 1. The model's coefficients, known as beta parameters, are typically estimated using maximum likelihood estimation (MLE). The logistic regression equation calculates log-odds, which can be converted into conditional probabilities for each observation. Model evaluation is commonly performed using measures such as the Hosmer-Lemeshow test.

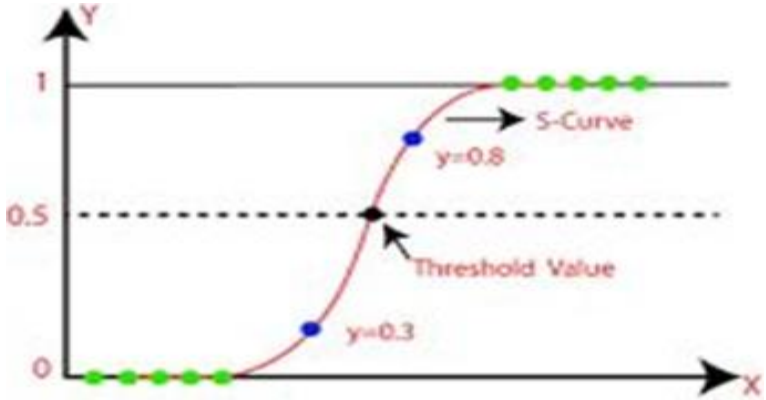



Fig 3.1 Illustrates the Graphical Representation of Logistic Regression

- *Naïve Bayes Classifier*

The Naïve Bayes classifier is a supervised machine learning technique widely used for text categorization tasks. It belongs to the generative learning algorithm family and aims to model the distribution of inputs across different classes or categories. Unlike discriminative classifiers such as logistic regression, Naïve Bayes does not prioritize learning the importance of specific features in discriminating between classes.

The Naïve Bayes classifier is often referred to as a probabilistic classifier because it relies on the principles of Bayesian statistics. Bayes' Theorem allows for the inversion of conditional probabilities, enabling the incorporation of new information to update prior probabilities into posterior probabilities. This classifier is particularly useful for tasks involving sequential events, where new data influences the initial probability estimates.

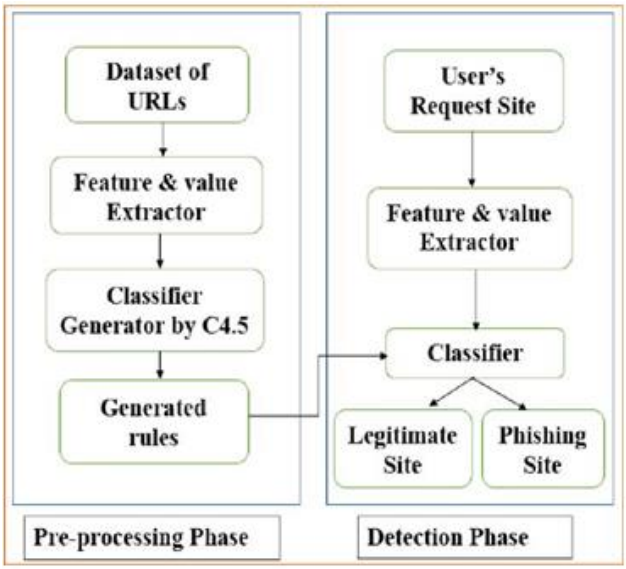



Fig 3.2 Naïve Bias Classifier

➢ *K-Nearest Neighbor Algorithm*

The K-nearest neighbors (KNN) classifier is a straightforward and easily implementable machine learning technique used for classification and regression tasks.

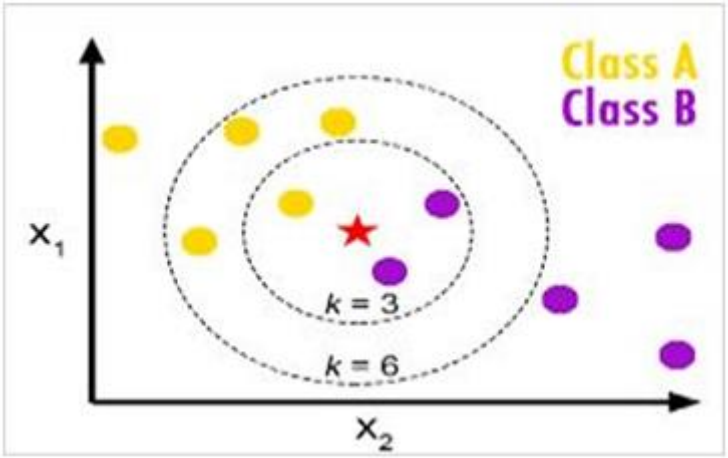



Fig 3.3 K Nearest Neighbour Algorithm

KNN is based on the assumption that similar data points are located close to each other. It captures the concept of similarity by measuring the distances between points on a graph, as depicted in Figure 3.3. Various distance metrics, such as the Euclidean distance, can be used to calculate these distances.

➢ *Kernel Support Vector Machine*

The kernel support vector machine (SVM) leverages the idea of adding additional dimensions to data to make it separable when it is indistinguishable in the current dimensions. This technique, known as the kernel trick, allows SVM to transform the data into a higher-dimensional space without explicitly increasing the dimensionality. Figure 3.4 and Figure 3.5 provide visual representations of the impact of the kernel and transformations in SVM.

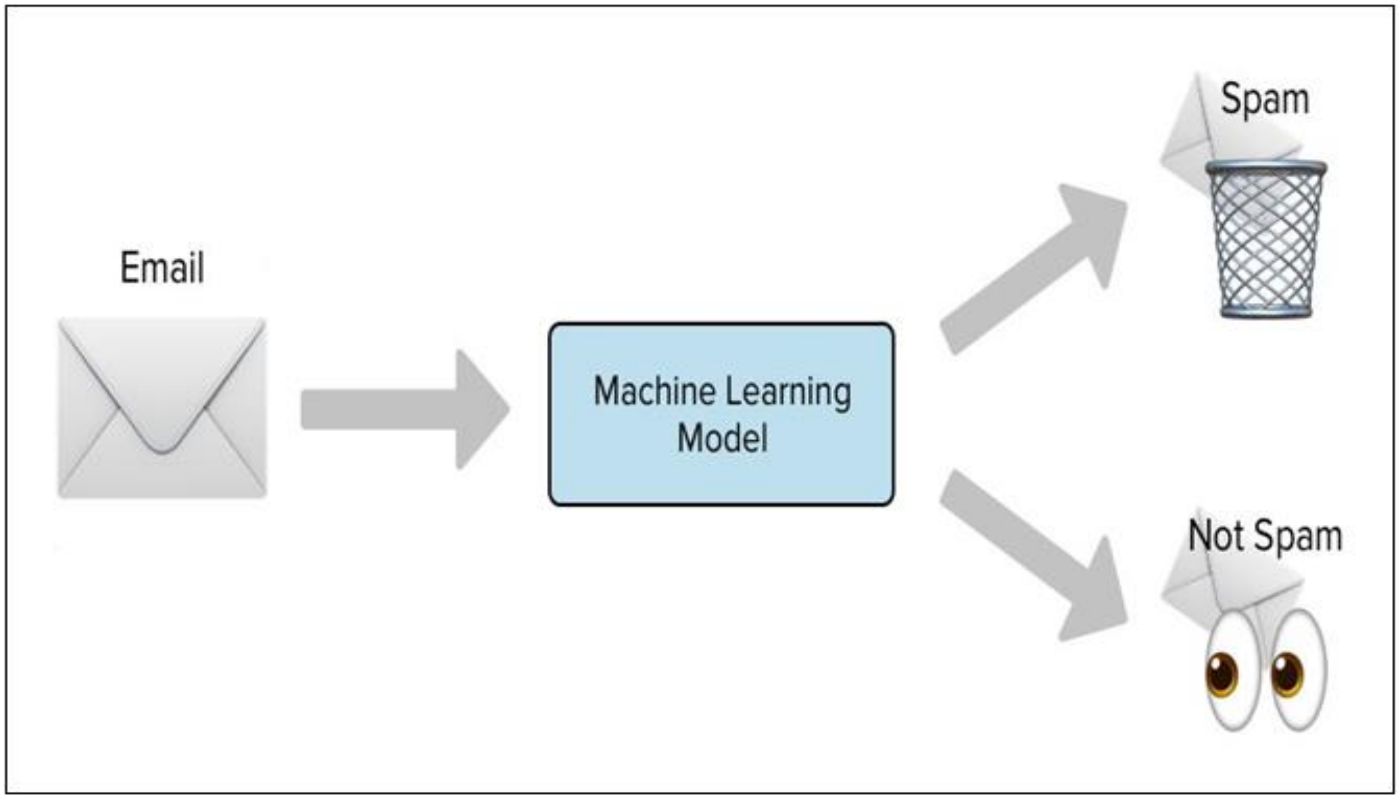



Fig 3.4 Kernelo Support5 Vector Machine

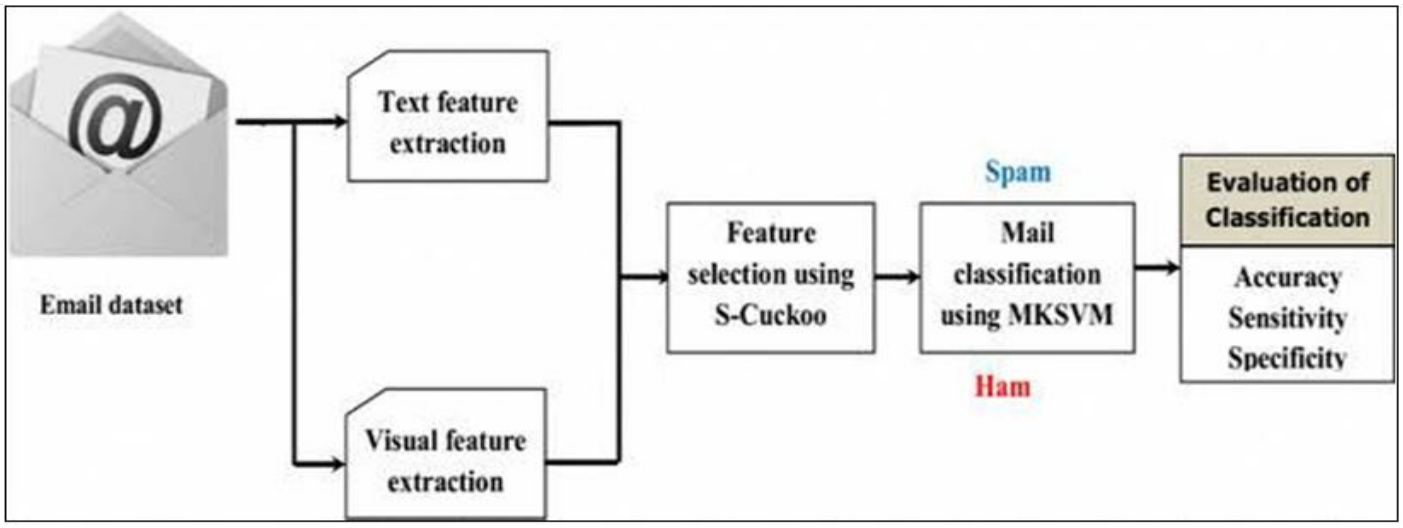



Fig 3.5 Provide Visual Representations of the Impact of the Kernel and Transformations in SVM

In SVM, a kernel is employed to consciously establish a distinct level of separation. Commonly used kernels include the Gaussian kernel, Sigmoid kernel, and Radial Basis Function.

➢ *Decision Tree*

A decision tree is a fundamental method used for categorizing instances. It consists of nodes, branches, and leaf nodes. Nodes evaluate specific properties, branches connect nodes or lead to leaf nodes, and leaf nodes represent the final outcome.

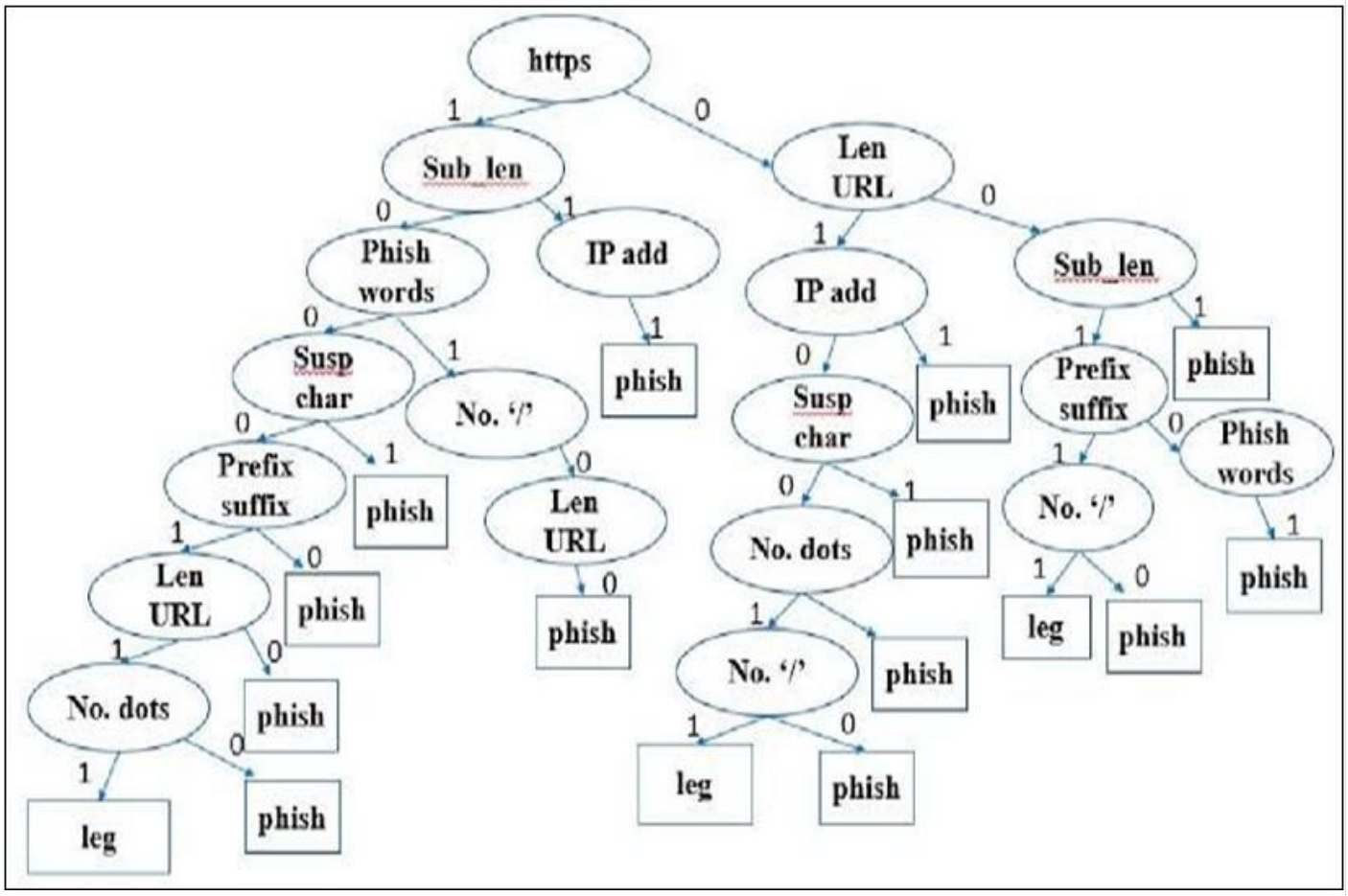



Fig 3.6 Provides an Illustrative Example of A Decision Tree.

The process of building a decision tree involves recursively partitioning the data in a binary manner. It divides the data based on certain criteria and repeats this process on each branch. In decision tree classification, a new instance is classified by following a series of tests represented by the decision tree structure.

➢ *Random Forest Classifier*

A random forest classifier is an ensemble of multiple decision trees. Each tree predicts a class, and the final prediction of the model is determined by majority voting among the individual trees. Figure 3.7 illustrates the process of random forest classification.

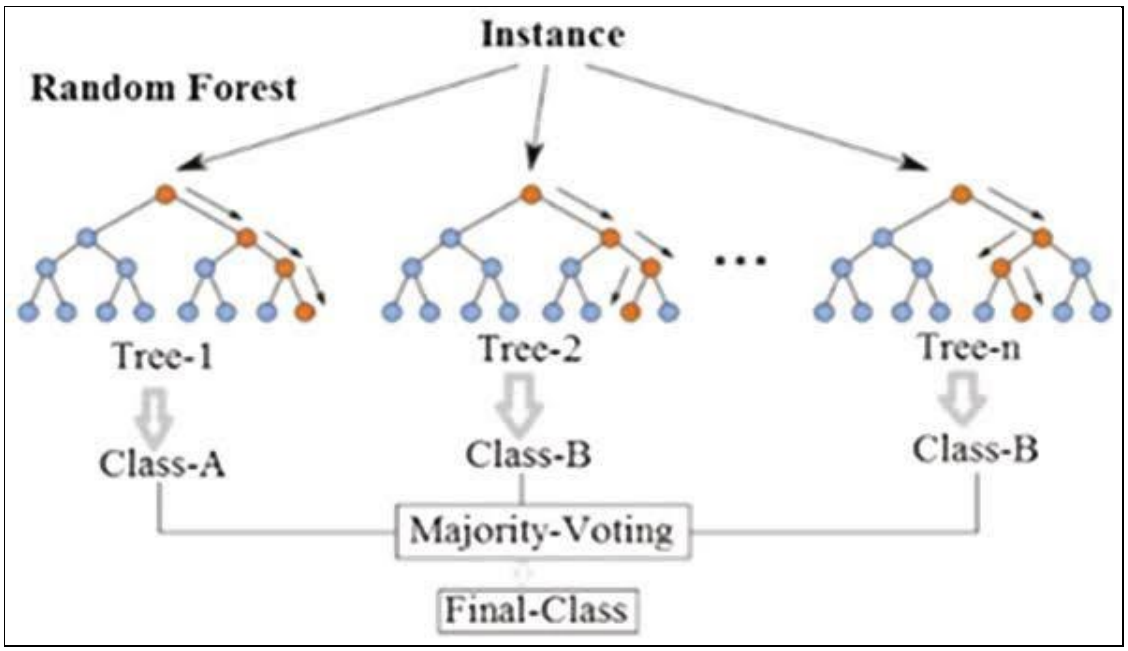



Fig 3.7 Illustrates the Process of Random Forest Classification

The random forest works by combining the knowledge from a large number of independent models (trees). This ensemble approach outperforms any single model within the random forest.

➢ *Gradient Boost Classifier*

Gradient boosting is an ensemble learning strategy that aims to minimize training errors by combining weak learners into a strong learner. The algorithm trains models sequentially, with each model compensating for the weaknesses of its predecessor. The gradient boost classifier, depicted in Figure 3.8, aggregates the weak rules from each classifier to create a strong prediction rule.

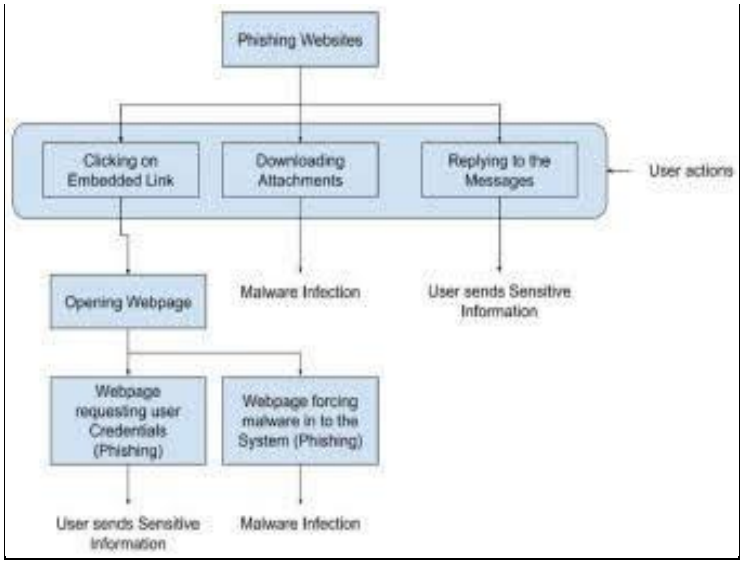



Fig 3.8 Aggregates the Weak Rules from Each Classifier to Create A Strong Prediction Rule

Boosting and bagging are two types of ensemble learning methods. Boosting trains weak learners sequentially, while bagging trains them in parallel. Boosting is often suitable for scenarios with low variance and high bias, whereas bagging is effective for models with high variance and low bias.

ISSN No:-2456-2165

- *Cat Boost Classifier*

CatBoost is a gradient boosting technique commonly used for regression and classification tasks. It extends the boosting technique by training each new model in the ensemble using the residuals of the previous model. Instead of fitting the original target variable, CatBoost aims to fit the residuals of the prior model. The algorithm employs gradient descent optimization to determine the best weights for the ensemble models.

CatBoost is a powerful algorithm capable of handling challenging scenarios, such as noisy data, missing values, and outliers. However, careful parameter tuning is essential to prevent computational inefficiency and overfitting.

ISSN No:-2456-2165

# CHAPTER FOUR
# SYSTEM REQUIREMENT SPECIFICATION

➢ *Hardware Requirements:*

- Processor CPU: Intel Pentium Dual Core or higher
- Minimum Hard Disk capacity: 512MB of space
- Minimum RAM: 4GB

➢ *Software Requirements:*

- Programming language: Python
- Operating system: Windows 8.1 or above
- IDE: Anaconda with Python version 3.x

➢ *Supporting Python Modules:*

Python provides a convenient mechanism for defining and utilizing modules within the interpreter. Modules are files that contain definitions, which can be imported and used in other modules or the main module. Table 3.1 presents some of the modules employed in this project.

Table 1 Supporting Python Modules

| S.No. | Python Modules | Description |
|---|---|---|
| 1 | Ipaddress | Enables the generation, control, and operation with IPv4 and IPv6 addresses and networks. |
| 2 | Re | Offers regular expression matching functions similar to those found in Perl. |
| 3 | urllib.request | Defines functions and classes for handling URL opening, particularly in the HTTP context. |
| 4 | BeautifulSoup | A Python package for parsing HTML and XML documents, commonly used for web scraping. |
| 5 | Socket | Provides access to the BSD interface of sockets. |
| 6 | Requests | Supports the sending of HTTP requests using Python. |
| 7 | Whois | Implements the WHOIS query and response protocol for retrieving information about Internet resource owners. |

ISSN No:-2456-2165

# CHAPTER FIVE
# SYSTEM DESIGN

➢ *System Architecture*

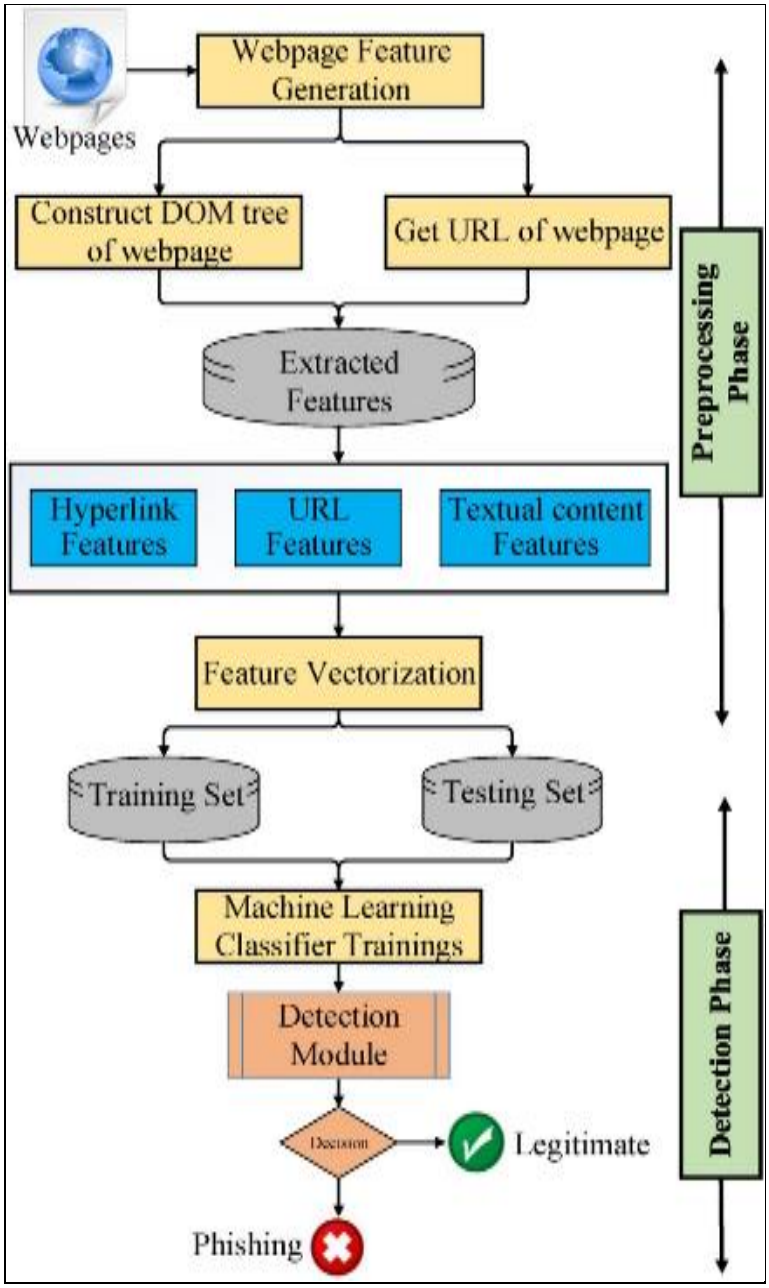
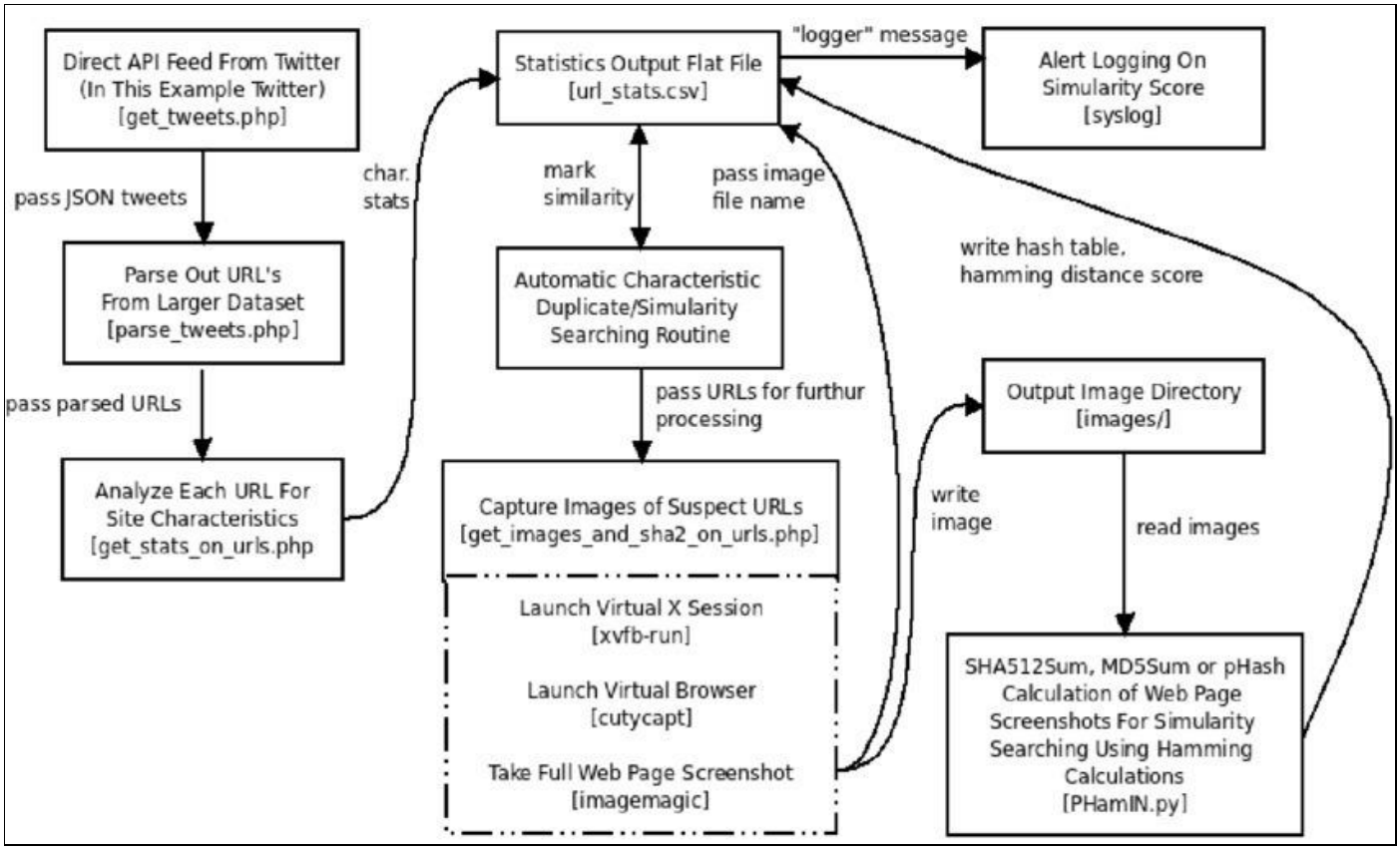



Fig 5.1 System Architecture

The system architecture, depicted in Figure 5.1, involves supplying URLs to the relevant classifier for categorization as either authentic or phishing. Trained classifiers utilize patterns identified from the training dataset to classify the provided input. Information such as IP address, URL length, domain, and favicon presence is extracted from the URLs, generating a list of their values. This list is then inputted into classifiers like KNN, kernel SVM, Decision tree, and Random Forest. The performance of these models is evaluated, resulting in an accuracy score. Based on the provided list, the trained classifier predicts whether the URL is legitimate or phishing.

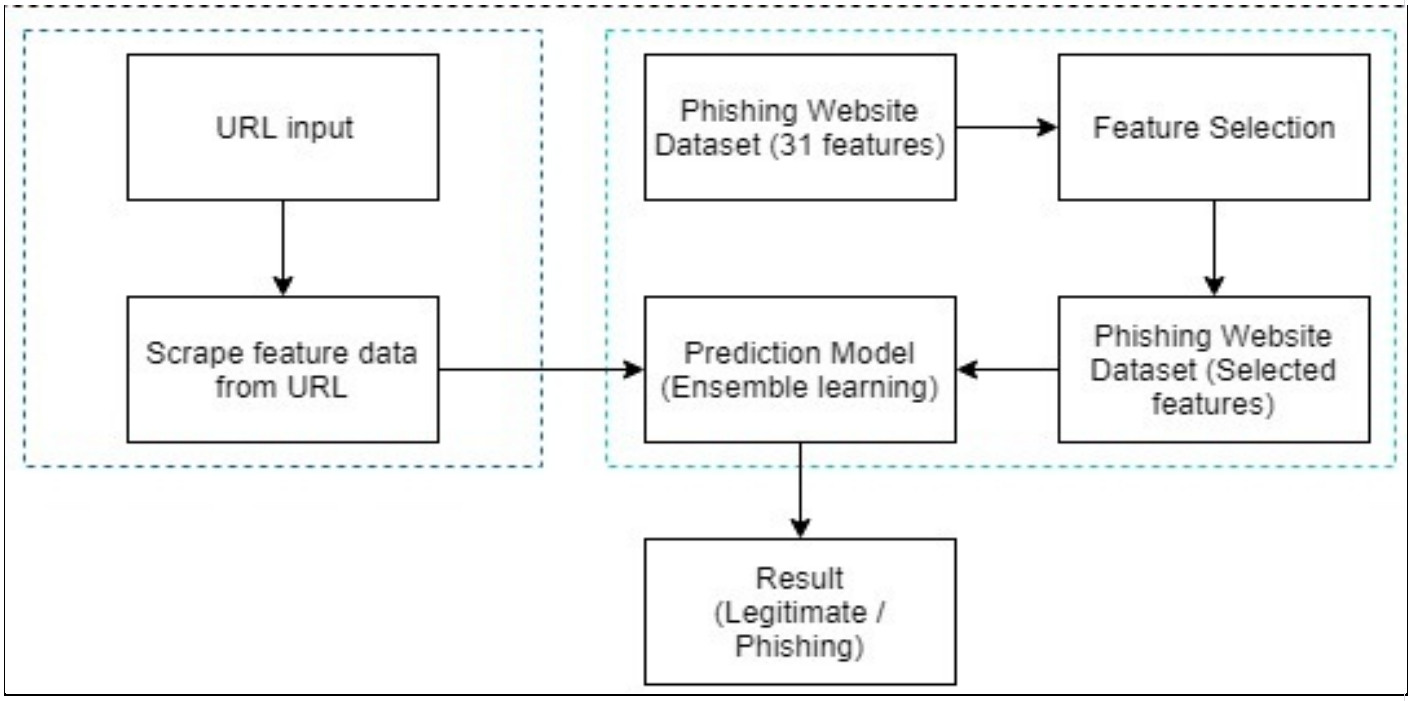
➢ *Data Flow Diagrams*

Data Flow Diagrams (DFDs) are graphical representations illustrating the flow of data within a system. They depict the processes involved, from input to report generation, and the connections between system entities. DFDs can be categorized into different levels of detail, namely 0, 1, and 2. The Gane-Sarson method is utilized for drawing DFDs in this chapter.




5.2.1 Data Flow Diagram - Level 0

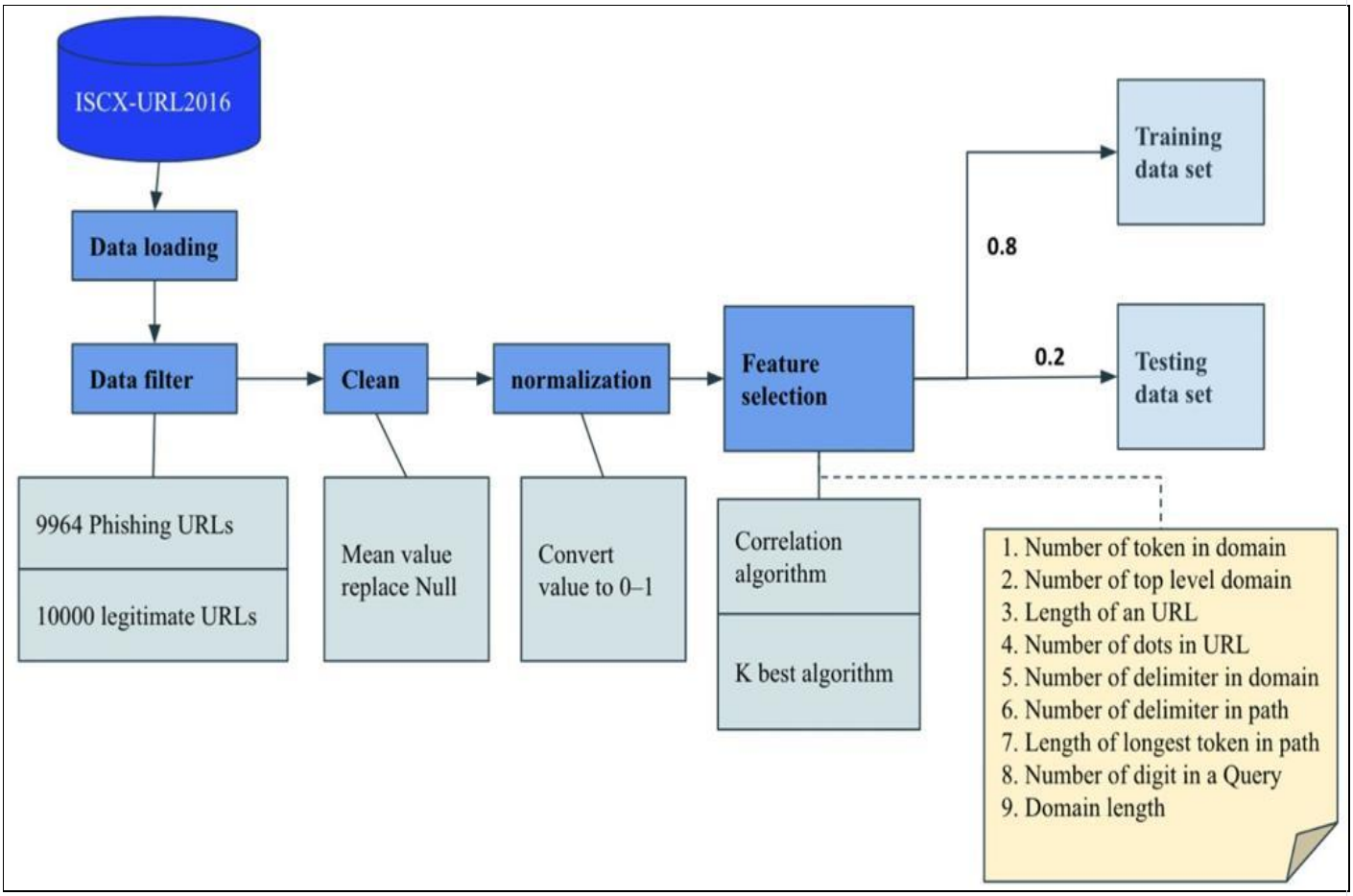
A Context Diagram, representing a DFD level 0 diagram, provides a high-level overview of the entire system. Figure 5.2 displays the system's DFD level 0, portraying the system as a high-level process connected to external entities. It aims to be easily comprehensible by stakeholders, developers, and data analysts.




5.2.2 Data Flow Diagram - Level 1

DFD level 1 offers a more detailed representation of the Context diagram, breaking down the high-level process into subprocesses. Figure 5.3 depicts the system's DFD level 1.


Fig 5.3 Depicts the System's DFD Level 1.




5.2.3 Data Flow Diagram - Level 2

DFD level 2 delves further into the processes involved in the system, encompassing feature extraction, dataset splitting, and classifier construction. Figure 5.4 illustrates the system's DFD level 2, providing a comprehensive understanding of its functioning.

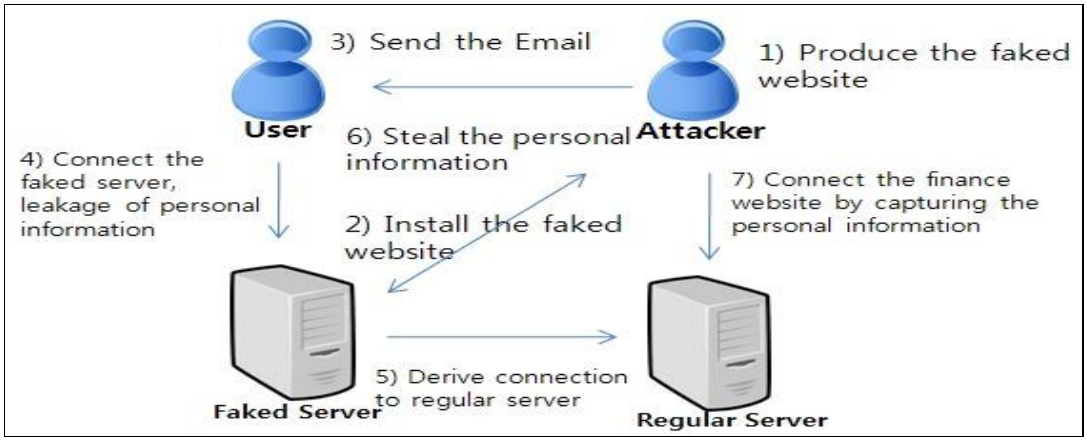



Fig 5.4 Illustrates the System's DFD Level 2, Providing A Comprehensive Understanding of its Functioning

- *UML Activity Diagram*

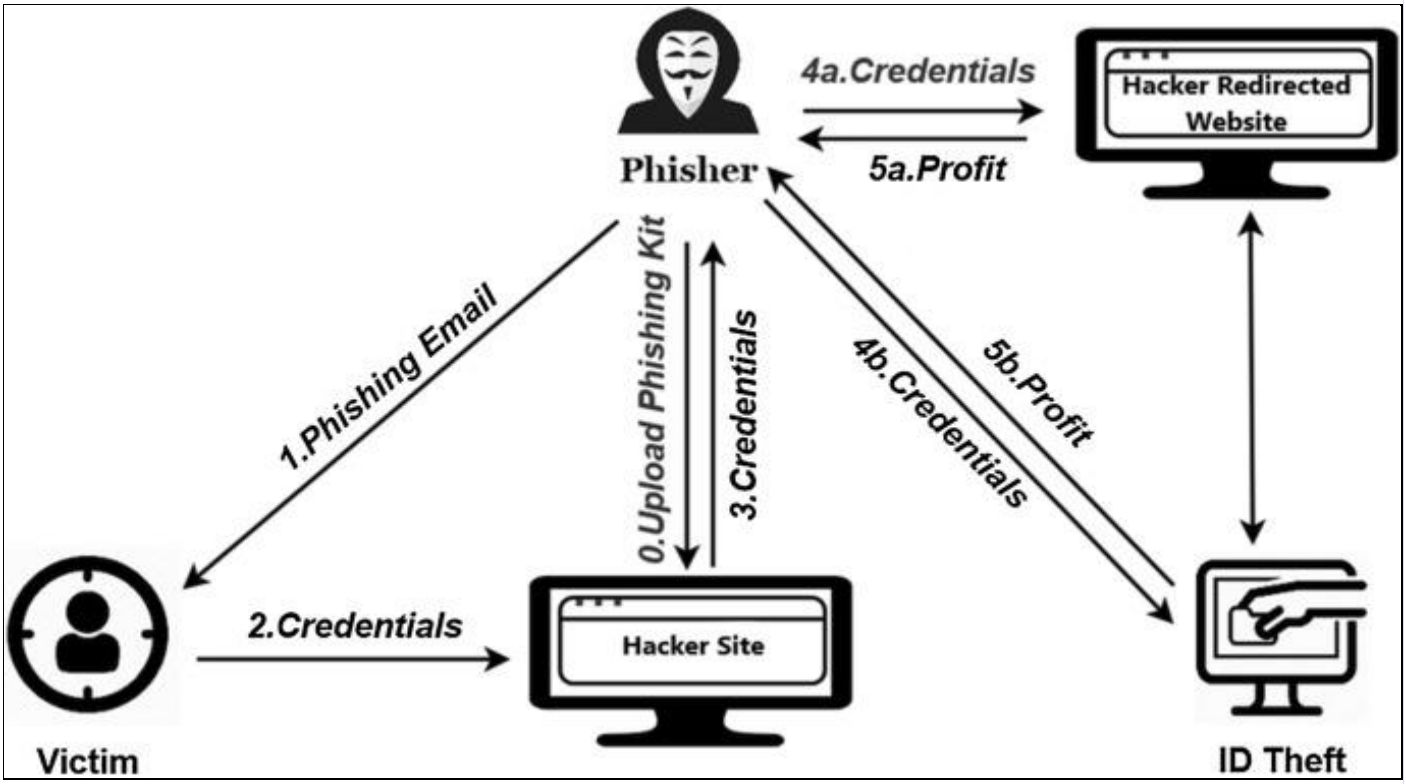



Fig 5.5 Behavioral Diagram that Visualizes the Control Flow

The system's activity diagram, presented in Figure 5.5, is a behavioral diagram that visualizes the control flow from a starting point to an ending point.

ISSN No:-2456-2165

# CHAPTER SIX
# IMPLEMENTATION

- This chapter outlines the methodology employed to determine the legitimacy of a URL and identify potential phishing attempts. Python modules such as **whois**, **socket**, **re**, **IP address**, **BeautifulSoup**, etc., are utilized to extract feature values such as IP address, URL length, domain name, subdomains, and favicon presence. These values are stored in a list format, which serves as the input for training the classifier. When a URL is entered, it is converted into a Python list representing different characteristics. The implementation includes the adoption of Kernel SVM, KNN, Random Forest, and decision tree classifiers.

For Kernel SVM, the classifier used is **sklearn.svm.SVC**, with the nonlinear algorithm specified by setting the **kernel** parameter to "rbf". The decision tree classifier is implemented using the **sklearn.tree** module and evaluates splits based on the "entropy" criterion. The Random Forest classifier comprises multiple decision trees, and the final classification is based on the majority vote from individual trees.

- User interface (UI) design of websites created specifically for web phishing detection. These websites incorporate powerful machine learning algorithms to evaluate the authenticity of URLs and provide users with feedback. The UI is designed to be user-friendly and intuitive, enabling users to easily recognize and avoid fraudulent phishing websites. Frameworks such as Bootstrap and Particles.js are integrated into the websites to enhance their visual appeal and overall user experience. The machine learning algorithm analyzes website content, structure, and other factors to determine its legitimacy, and the results are presented in an accessible format, including a legitimacy percentage and information about associated risks.

- Flask, a flexible Python web framework widely used for rapid development of online applications. Flask provides built-in functions such as routing and templating and can be extended using third-party libraries. It is commonly employed for building webpages that incorporate machine learning, enabling users to input data and obtain results. Additionally, Flask facilitates the deployment of machine learning models as web services through APIs, allowing integration with other applications.

- Anaconda, an essential tool for data science and machine learning projects. Anaconda offers efficient package management, simplifying the installation, removal, and updating of libraries necessary for data science and machine learning tasks. It provides a comprehensive set of tools and packages that support the complete data science workflow, including data exploration, cleaning, model construction, and deployment. The integration of Jupyter Notebook further enhances its capabilities

ISSN No:-2456-2165

# CHAPTER SEVEN
# TESTING AND VALIDATION

This chapter focuses on testing and validating the proposed system by comparing the algorithm's results with the actual outcomes. Each algorithm undergoes rigorous testing with both legitimate and phishing URLs, and the results are carefully analyzed and presented.

- *Unit Testing*

Unit testing is conducted to thoroughly assess the functionality of individual modules and ensure their suitability for implementation.

ISSN No:-2456-2165

# CHAPTER EIGHT
# LABORATORY INVESTIGATION

The performance of the classifier is evaluated using a confusion matrix (CM), which visually represents the accuracy of the predictions made. The CM provides important measures such as true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

ISSN No:-2456-2165

# CHAPTER NINE
# FINAL COMMENTS AND FUTURE WORKS

This chapter presents the final conclusions drawn from the study and provides recommendations for future research and improvements.

- The Verdict In our rapidly advancing technological world, phishing is evolving into a more sophisticated threat. With the global economy shifting towards cashless and paperless transactions, phishing acts as a hindrance to this progress. The trust in the internet's reliability has diminished, and while Artificial Intelligence (AI) can be a valuable tool for information gathering, individuals lacking the ability to identify security risks should avoid online financial transactions. Phishers primarily target the installation industry and cloud benefits. This research aims to address this issue by utilizing Machine Intelligence (MI) to detect phishing websites. The objective is to develop an effective, accurate, and affordable mechanism by leveraging machine learning techniques. The study was conducted using the Anaconda IDE and implemented in Python. The proposed approach involved employing four machine learning classifiers and conducting a comparative analysis. The results showed a commendable accuracy rating, with the Random Forest Classifier achieving the highest accuracy score of 96.82%. However, it should be noted that the accuracy may vary with different datasets and algorithms, potentially surpassing the performance of the Random Forest Classifier. The ensemble classifier Dashier exhibited good precision and can effectively identify legitimate URLs in real-world scenarios (Pages 99–104 of the 2016 Conference on Digital Information Processing, De Mining, and Wireless Communications (DIPDMWC)).

## REFERENCES

[1]. Giri, M., Jain, S., & Sahare, V. (2015). Cloud-based visual cryptography anti-phishing system. JAFRC, 2(01).
[2]. Dhage, S., & Patil, S. (2019). An organized process for building an anti-phishing framework and a rigorous overview of phishing detection. In Fifth International Conference on Advanced Computing Communication Systems (ICACCS), pp. 588-593.
[3]. Vaya, D., Khandelwal, S., & Hadpawat, T. (2017). Visual cryptography: A review. International Journal of Computer Applications, 174(40–43).
[4]. Saoji, S. (2015). Phishing detection system utilizing visual cryptography.
[5]. Pham, C., Tran, N. H., Huh, E., & Hong, C. S. (2018). A neuro-fuzzy method for phishing detection in fog networks. IEEE Transactions on Network and Service Management, 15(3), 1076-1089.
[6]. Yong, K. S. C., Chiew, K. L., & Tan, C. L. (2019). A survey of QR code phishing: Current attacks and countermeasures. In 7th International Conference on Smart Computing Communications (ICSCC), pp. 1-5.
[7]. Egozi, & Verma, R. (2018). Phishing email detection using robust NLP techniques. In IEEE International Conference on Data Mining Workshops (ICDMW), pp. 7-12.
[8]. Mao, J., Tian, W., Li, P., Wei, T., & Liang, Z. (2017). Phishing-alarm: Robust and efficient phishing detection via page component similarity. IEEE Access, 5, 17020-17030.
[9]. Kathrine, G. J. W., Praise, P. M., Rose, A. A., & Kalavani, E. C. (2019). Variants of phishing attacks and their detection techniques. In 3rd International Conference on Trends in Electronics and Informatics (COED), pp. 255-259.