A

Industrial Oriented Mini Project Report

On

PHISHING DETECTION SYSTEM THROUGH HYBRID MACHINE LEARNING BASED ON URL

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

By

U.KRISHNA KOUSHIK (227R1A67J6)

JEKKULA ROHINI (227R1A67E8)

THUMULA SRIKANTH (227R1A67J4)

Under the Guidance of

Mrs.M.Anusha

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGC Act.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

June, 2025.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

CERTIFICATE

This is to certify that the project entitled "**PHISHING DETECTION SYSTEM THROUGH HYBRID MACHINE LEARNING BASED ON URL**" being submitted by **U.KRISHNA KOUSHIK (227R1A67J6), JEKKULA ROHINI (227R1A67E8), THUMULA SRIKANTH (227R1A67J4)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering (Data Science) to the Jawaharlal Nehru Technological University Hyderabad, during the year 2024-25.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Mrs.M.Anusha
Assistant Professor
INTERNAL GUIDE

Dr. K. Murali
HoD-CSE(DS)

Dr. A. Raji Reddy
Director

External Examiner

Submitted for viva-voce Examination held on _____

ACKNOWLEDGEMENT

We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project, we take this opportunity to express our profound gratitude and deep regard to our guide **Mrs. M.Anusha**, Assistant Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by her shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) coordinators **Mrs. J. Rekha, Mrs. M. Anusha** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Murali**, Head, Department of Computer Science and Engineering (Data Science) for providing encouragement and support for completing this project successfully.

We are deeply grateful to **Dr. A. Raji Reddy**, Director, for his cooperation throughout the course of this project. Additionally, we extend our profound gratitude to **Sri. Ch. Gopal Reddy**, Chairman, **Smt. C. Vasantha Latha**, Secretary and **Sri. C. Abhinav Reddy**, Vice-Chairman, for fostering an excellent infrastructure and a conducive learning environment that greatly contributed to our progress.

The guidance and support received from all the members of CMR Technical Campus who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

U.KRISHNA KOUSHIK (227R1A67J6)

JEKKULA ROHINI (227R1A67E8)

THUMULA SRIKANTH (227R1A67J4)

ABSTRACT

Currently, numerous types of cybercrime are organized through the internet. Hence, this study mainly focuses on phishing attacks. Although phishing was first used in 1996, it has become the most severe and dangerous cybercrime on the internet. Phishing utilizes email distortion as its underlying mechanism for tricky correspondences, followed by mock sites, to obtain the required data from people in question. Different studies have presented their work on the precaution, identification, and knowledge of phishing attacks; however, there is currently no complete and proper solution for frustrating them. Therefore, machine learning plays a vital role in defending against cybercrimes involving phishing attacks. The proposed study is based on the phishing URL-based dataset extracted from the famous dataset repository, which consists of phishing and legitimate URL attributes collected from 11000+ website datasets in vector form. After preprocessing, many machine learning algorithms have been applied and designed to prevent phishing URLs and provide protection to the user.

This study uses machine learning models such as decision tree (DT),linear regression (LR), random forest (RF), naive Bayes (NB), gradient boosting classifier (GBM), K-neighbors classifier (KNN), support vector classifier (SVC), and proposed hybrid LSD model, which is a combination of logistic regression, support vector machine, and decision tree (LR+SVC+DT) with soft and hard voting, to defend against phishing attacks with high accuracy and efficiency. The canopy feature selection technique with cross fold validation and Grid Search Hyper parameter Optimization techniques are used with proposed LSD model. Furthermore, to evaluate the proposed approach, different evaluation parameters were adopted, such as the precision, accuracy, recall, F1-score, and specificity, to illustrate the effects and efficiency of the models. The results of the comparative analyses demonstrate that the proposed approach outperforms the other models and achieves the best results.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

1. INTRODUCTION

The internet has revolutionized the way people interact, communicate, and conduct business across the globe. It provides instant access to information, facilitates e-commerce, supports educational platforms, and enhances social connectivity. Despite these numerous benefits, the rapid growth of internet usage has also opened the door to a wide range of cybercrimes. One of the most prevalent and dangerous among these is phishing. Phishing is a malicious activity where attackers impersonate legitimate entities to deceive users into providing sensitive information, such as login credentials, financial data, or personal identity details. These attacks typically occur via deceptive emails or fake websites designed to mimic trustworthy sources.

As phishing techniques have become more sophisticated, traditional detection methods such as blacklist-based systems are no longer sufficient. Blacklists can only identify known threats, leaving systems vulnerable to new or "zero-day" phishing attacks. This growing threat landscape necessitates the development of advanced, intelligent detection systems capable of adapting to new and unseen phishing strategies. Machine learning has emerged as a promising solution in the cyber security domain. By analyzing patterns and features in URLs, machine learning algorithms can effectively distinguish between legitimate and malicious websites. This study leverages machine learning techniques to design and evaluate a hybrid phishing detection system based on URL features, providing a proactive defense mechanism against phishing threats.

1.1 PROJECT PURPOSE

The primary goal of this project is to develop a robust and efficient phishing detection system that leverages machine learning techniques to identify and block phishing websites based on their URLs. The purpose extends beyond merely identifying known phishing sites; it aims to detect newly created phishing sites that do not yet exist in blacklists or databases.

Key objectives include:

- Enhancing internet security by reducing the risk of phishing attacks.
- Providing users with real-time protection from malicious websites.
- Improving upon the limitations of existing phishing detection techniques, especially in terms of accuracy and adaptability.
- Utilizing a comprehensive dataset of phishing and legitimate URLs to train and validate machine learning models.
- Demonstrating that hybrid models and feature optimization techniques can significantly outperform single-model approaches in phishing detection.

Ultimately, this research contributes to the broader field of cyber security by presenting a novel, accurate, and scalable solution to the persistent problem of phishing.

1.2 PROJECT FEATURES

The project offers several innovative features and technical implementations that make it a comprehensive and effective phishing detection system:

1. **Extensive Dataset:**
   - Uses a well-known dataset containing over 11,000 URL samples, labeled as either phishing or legitimate.
   - Features are derived from URL structures, such as the presence of IP addresses, URL length, special characters, use of HTTPS, subdomains, and embedded scripts.
2. **Diverse Machine Learning Models:**
   - Implements and compares multiple classifiers: Decision Tree (DT), Linear Regression (LR), Naive Bayes (NB), Random Forest (RF), Gradient Boosting Machine (GBM), Support Vector Classifier (SVC), and K-Nearest Neighbors (KNN).
   - Proposes a hybrid ensemble model (LSD) that combines LR, SVC, and DT using both soft and hard voting mechanisms to enhance classification accuracy.
3. **Feature Selection and Optimization:**
   - Uses the Canopy clustering algorithm for feature selection to improve model performance by focusing on the most impactful attributes.

- Incorporates cross-fold validation and grid search hyperparameter tuning to optimize training and prevent overfitting.

4. **High Accuracy and Performance Evaluation:**
   - Evaluation metrics include accuracy, precision, recall, specificity, and F1-score.
   - The proposed hybrid model demonstrates superior results, achieving over 98% accuracy, outperforming other standard models.
5. **Scalability and Real-World Applicability:**
   - The system is designed to be scalable and adaptable to real-time phishing detection scenarios.
   - Can be integrated into browser extensions, network monitoring systems, or email filters for proactive cybersecurity solutions.

LITERATURE REVIEW

2. LITERATURE REVIEW

2.1 REVIEW OF RELATED WORK

Over the past decade, researchers and cyber security professionals have proposed various methods to combat phishing attacks. These methods are primarily categorized into list-based, heuristic-based, and machine learning/deep learning-based approaches.

2.1.1 List-Based Approaches

These systems depend on blacklists (URLs known to be malicious) and whitelists (trusted URLs). Popular services include:

- Google Safe Browsing API
- PhishTank
- Microsoft SmartScreen.

Limitations:

- Cannot detect zero-day phishing attacks (new, unknown threats).
- Requires frequent updates.
- Susceptible to false negatives.

2.1.2 Heuristic-Based Approaches

These use manually defined rules to analyze webpage structures (e.g., URL length, presence of "@" symbols, IP-based domains, etc.) to classify suspicious websites. They are faster but limited in scalability and adaptability.

2.1.3 Machine Learning-Based Approaches

ML techniques can detect phishing by learning patterns from data. Some notable research:

- CANTINA (Zhang et al., 2007): Extracted features like TF-IDF keywords from HTML pages.
- CANTINA+: Enhanced version using 15 HTML features and DOM tree structure.
- DeltaPhish: Used visual similarity and URL characteristics.

- PhishSafe (Jain & Gupta): Employed SVM and Naive Bayes with 90% accuracy.
- PhiDMA: Multi-filter URL detection with 92% accuracy.
- Recent methods use NLP, deep neural networks, and ensemble models to improve detection rates.

ML models outperform traditional approaches but require large datasets, effective feature engineering, and tuning to achieve high accuracy and generalization.

2.2 DEFINITION OF PROBLEM STATEMENT

Phishing attacks have become one of the most prevalent and dangerous forms of cybercrime, exploiting users by impersonating legitimate entities to steal sensitive information such as login credentials, credit card details, and personal data. Traditional detection mechanisms—such as blacklists and rule-based systems—are no longer sufficient, as they fail to detect new, unknown (zero-day) phishing URLs and often suffer from high false-positive rates.

The core problem lies in the lack of an adaptive, accurate, and scalable solution capable of distinguishing between phishing and legitimate websites in real time. There is a critical need for a system that can learn from URL-based patterns, generalize to new types of phishing strategies, and provide reliable protection to users.

Therefore, this project focuses on developing a hybrid machine learning-based phishing detection system using URL features. The system is designed to:

- Accurately classify phishing and legitimate URLs,
- Handle large-scale data efficiently,
- Reduce false positives and improve detection rates,
- Adapt to new phishing techniques using ensemble learning and feature selection.

2.3 EXISTING SYSTEM

Existing phishing detection systems primarily rely on list-based approaches (blacklists and whitelists) and basic machine learning classifiers. Blacklist-based systems maintain databases of known malicious URLs and block access to them. While easy to implement and widely used in browsers and security tools (e.g., Google Safe Browsing API, PhishTank), these systems are inherently limited by their inability to detect newly emerging (zero-day) phishing sites.

Some systems incorporate heuristic-based methods, analyzing features such as URL length, presence of IP addresses, and suspicious symbols like "@" or "//". These methods provide better coverage but often rely on manually crafted rules, which lack flexibility and can quickly become outdated.

Machine learning-based systems represent a significant advancement. These systems use classifiers like Support Vector Machines (SVM), Naive Bayes, and K-Nearest Neighbors (KNN) to learn from patterns in phishing and legitimate URLs. For example:

- PhishDef used URL tokenization to identify suspicious patterns.
- DeltaPhish applied visual similarity analysis to compare phishing pages to known legitimate websites.
- Phish-Safe implemented a lightweight feature-based system using supervised learning.

However, many existing models rely on limited feature sets and lack advanced optimization or ensemble techniques. Most are trained on small or outdated datasets and lack real-time detection capability.

2.3.1 Limitations of Existing System

Despite the evolution of phishing detection techniques, current systems have several notable limitations:

- Inability to Detect Zero-Day Attacks: List-based approaches can only identify known phishing URLs. They fail to detect newly created or modified malicious websites.

- High False Positive/Negative Rates: Simple machine learning classifiers often misclassify legitimate URLs as phishing and vice versa, especially in cases with ambiguous or borderline features.

- Lack of Feature Optimization: Many systems do not apply proper feature selection or engineering, leading to models trained with irrelevant or redundant features, which reduce detection accuracy.

- Poor Scalability and Adaptability: Traditional systems are not designed for real-time applications and do not adapt well to rapidly evolving phishing techniques.

- Overfitting and Lack of Generalization: Some models are overfitted to specific training datasets and do not perform effectively when exposed to diverse or unseen data in real-world environments.

- Computational Inefficiency: Complex models without optimization often require significant computational resources, limiting their practicality for deployment in resource-constrained environments (e.g., mobile browsers or extensions).

These limitations highlight the need for a more robust, accurate, and scalable solution—one that incorporates ensemble learning, effective feature selection, and hyperparameter tuning to enhance phishing detection performance and reliability.

2.4 PROPOSED SYSTEM

The proposed system introduces a hybrid machine learning-based phishing detection framework that utilizes URL-based features to classify websites as either phishing or legitimate. Unlike traditional blacklist-based or standalone ML systems, this approach combines the predictive capabilities of multiple classifiers—Logistic Regression (LR), Support Vector Classifier (SVC), and Decision Tree (DT)—into a hybrid ensemble model referred to as LSD (LR + SVC + DT). Both soft voting and hard voting mechanisms are used to aggregate predictions and enhance classification reliability.

To further improve detection accuracy and efficiency, the system incorporates the following optimization strategies:

- Canopy Feature Selection: Identifies and retains the most relevant features from the URL dataset, reducing noise and improving training quality.

- Cross-Fold Validation: Ensures that the model generalizes well by evaluating it across multiple data partitions.

- Grid Search Hyperparameter Tuning: Optimizes model parameters for improved performance across all classifiers.

The dataset used in this system contains over 11,000 URL samples (both phishing and legitimate), with 32 extracted features such as IP usage, URL length, special characters, HTTPS presence, domain registration length, and others. After preprocessing, the data is split into training and testing sets (70/30 ratio), enabling the models to learn and validate with high efficiency.

Evaluation metrics such as accuracy, precision, recall, specificity, and F1-score are used to assess model performance. The proposed LSD hybrid model achieves high detection accuracy (up to 98.12%), outperforming traditional machine learning methods in phishing URL classification.

2.4.1 Advantages of the Proposed System:

The proposed hybrid ensemble system offers several advantages over existing phishing detection approaches:

- High Accuracy and Reliability: The combination of LR, SVC, and DT with voting mechanisms significantly enhances classification performance, reducing both false positives and false negatives. The model achieved an accuracy of 98.12%, demonstrating its robustness.
- Improved Generalization: Cross-validation and grid search ensure that the model is not overfitted to the training data, enabling better performance on new, unseen phishing threats.
- Scalable and Lightweight: The use of URL-based features eliminates the need for content parsing or deep webpage inspection, making the system fast, lightweight, and suitable for real-time implementation (e.g., in browsers, email clients).

- Effective Feature Selection: Canopy feature selection focuses on the most informative attributes, improving model efficiency and reducing computational overhead.

- Adaptable to New Attacks: Unlike blacklist-based systems, this model learns patterns and can classify previously unknown URLs based on behavioral and structural features.

- Ensemble Robustness: The hybrid model leverages the strengths of individual classifiers—logistic regression for linear separation, SVC for margin maximization, and decision trees for handling non-linear data—ensuring a comprehensive and balanced detection strategy.

2.5 OBJECTIVES

- Develop an Automated Phishing Detection System

  Design and implement a machine learning-based system that can automatically detect and classify phishing websites using URL-based features.

- Improve Detection Accuracy

  Utilize a hybrid ensemble model (Logistic Regression + Support Vector Classifier + Decision Tree) combined with soft and hard voting mechanisms to achieve higher detection performance.

- Enhance Feature Optimization and Model Efficiency

  Employ Canopy feature selection and grid search hyperparameter tuning to reduce model complexity while maximizing prediction accuracy.

- Ensure Scalability and Real-Time Detection

  Build a lightweight and scalable solution capable of handling large volumes of URL data and making fast, real-time classification decisions.

- Reduce False Positives and Negatives

  Minimize misclassification rates to enhance the reliability of the detection system and prevent unnecessary blocking or overlooking of URLs.

- Contribute to Cybersecurity Awareness

  Assist users in identifying potentially harmful websites and contribute to a safer and more secure online environment.

2.6 HARDWARE & SOFTWARE REQUIREMENTS

2.6.1 HARDWARE REQUIREMENTS:

The system is designed to run efficiently on standard hardware. The following configuration is recommended for development and testing:

- Processor: Intel Core i5 or higher
- Hard Disk: 256 GB SSD / 500 GB HDD
- RAM: 8 GB (minimum), 16 GB recommended
- Graphics Card: Optional – NVIDIA GPU (for faster ML model training)

2.6.2 SOFTWARE REQUIREMENTS:

The software stack for building and running the phishing detection system includes:

- Operating System: Windows 10 / Linux (Ubuntu preferred)
- Programming Language: Python 3.x
- Libraries & Packages:
  1. scikit-learn – for machine learning models
  2. pandas, numpy – for data preprocessing and manipulation
  3. matplotlib, seaborn – for data visualization
  4. joblib – for model persistence and loading
- Development Environment: Jupyter Notebook / VS Code / PyCharm
- Dataset Source: Kaggle – Phishing Websites Dataset
- Framework (if web-based deployment is needed): Flask or Django

SYSTEM ARCHITECTURE & DESIGN

3. SYSTEM ARCHITECTURE & DESIGN

Project architecture refers to the structural framework and design of a project, encompassing its components, interactions, and overall organization. It provides a clear blueprint for development, ensuring efficiency, scalability, and alignment with project goals. Effective architecture guides the project's lifecycle, from planning to execution, enhancing collaboration and reducing complexity.

3.1 PROJECT ARCHITECTURE

The architecture of the proposed phishing detection system consists of several stages—starting from data input and preprocessing to feature extraction, model comparison, and final classification of URLs. It leverages hybrid machine learning techniques to improve classification performance.

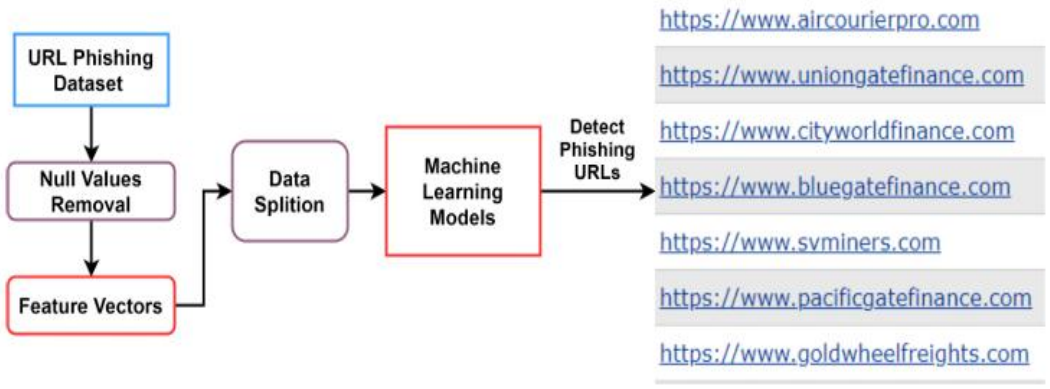


Figure 3.1: Project Architecture of Hybrid Machine Learning-Based Phishing URL Detection System

3.2 DESCRIPTION

Data Input

The system uses a dataset of over 11,000 website URLs (phishing and legitimate) collected from a reliable open-source repository (e.g., Kaggle). Each URL in the dataset is labeled and includes multiple structural and behavioral features.

Preprocessing

The raw dataset is cleaned by removing null values, duplicates, and irrelevant fields. Feature vectors are normalized and encoded to make them compatible with machine learning models.

Feature Extraction

Important URL features—such as length, use of IP, presence of symbols ("@", "-", "//"), HTTPS usage, subdomain count, and domain registration length—are extracted. This process is enhanced using Canopy Feature Selection, which selects the most informative attributes.

Model Training and Evaluation (Model Comparison)

Three different models are trained on the dataset:

- Logistic Regression (LR)
- Support Vector Classifier (SVC)
- Decision Tree (DT)

Each model is evaluated using standard metrics (accuracy, precision, recall, F1-score), and performance is recorded for comparison.

Best Model Selection (Ensemble Voting)

The most accurate results are obtained through a hybrid ensemble model (LSD) that combines the predictions of LR, SVC, and DT using soft and hard voting. This improves prediction consistency and reduces the chance of misclassification.

Test URL / Real-time Input

A test URL (or batch of URLs) can be passed through the trained model pipeline. Feature extraction is performed, and the URL is analyzed by the best-selected model.

Content Classification Output

The system provides the final classification:

- Phishing (unsafe/malicious)
- Legitimate (safe)

The model's output can be integrated with browser extensions, email filters, or online platforms to block or warn users about phishing threats.

3.3 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a graphical tool used to represent how data moves through a system. It visually outlines how input data is processed into output through different system components, showcasing external entities, processes, data stores, and data flows. In the context of this project, the DFD illustrates the flow of URL data through various stages of preprocessing, feature extraction, model training, and final classification.

Key Components in the Phishing Detection System DFD:

- External Entities: Represent the data sources (e.g., user-entered URLs, dataset files from repositories such as Kaggle).
- Processes: Operations applied to the data, including preprocessing, feature extraction, model training, prediction, and evaluation.
- Data Flows: The movement of URL data between different modules in the system.
- Data Stores: Repositories where cleaned data, extracted features, and model results are temporarily stored for reuse or analysis.

Benefits of Using DFDs in Phishing Detection Projects:

- Helps both technical and non-technical stakeholders understand how phishing URLs are detected and processed.
- Visualizes the flow of data through multiple stages—from ingestion to classification.
- Identifies system bottlenecks or security risks in data handling.
- Improves communication between development and cybersecurity teams.
- Supports modular system development and future scalability.

Applications:

- Streamlining the URL detection pipeline.
- Mapping the flow of data from raw input (URL) to decision output (phishing or legitimate).
- Ensuring data security and accurate handling at each stage.
- Serving as documentation for future enhancements or integrations (e.g., browser plugin or email filter integration).

Levels of DFD:

• **Level 0 (Context Diagram):**

Depicts the overall phishing detection system, showing high-level interactions between external data sources (e.g., dataset or user input) and the system, ending with a classification output.

• **Level 1 DFD:**

Breaks down the core processes into sub-processes such as:

- Data Preprocessing
- Feature Extraction
- Machine Learning Model Training
- Ensemble Voting & Classification
- Results Display

• Level 2 DFD (Optional for Complex Expansion):

Can provide a deeper look into specific operations like:

- Canopy-based Feature Selection
- Cross-fold Validation
- Hyperparameter Tuning via Grid Search
- Real-time URL Testing and Classification

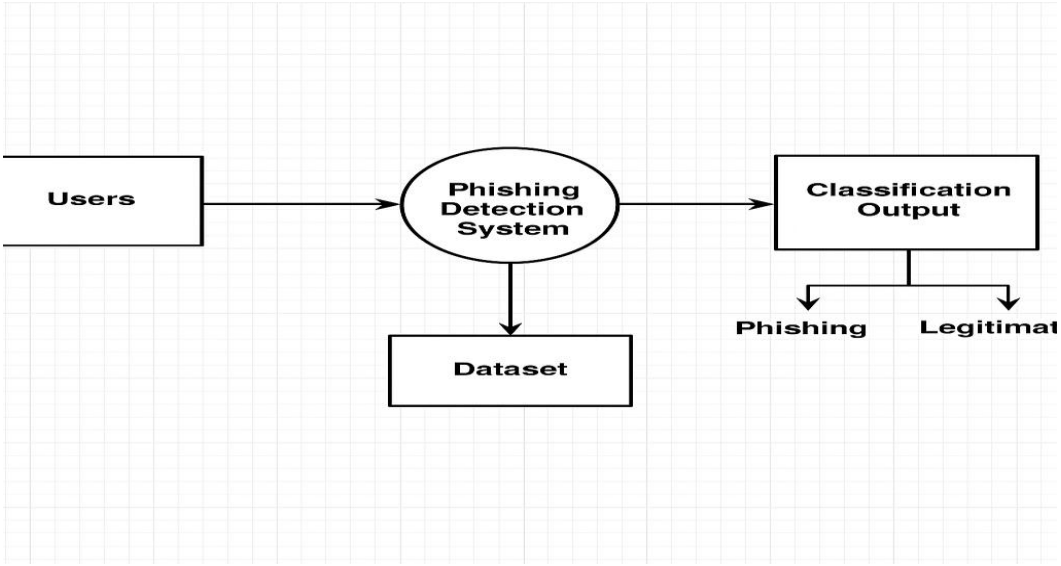


Figure 3.2: Data Flow Diagram of a Hybrid Machine Learning-Based Phishing URL Detection System

IMPLEMENTATION

4.IMPLEMENTATION

The implementation phase of the Phishing URL Detection System involves converting theoretical designs and models into an operational machine learning pipeline. The system is designed to process a set of URLs, extract relevant features, apply machine learning algorithms, and deliver accurate predictions to classify the URLs as either phishing or legitimate. The system integrates various ML techniques, optimization tools, and evaluation strategies to ensure performance, scalability, and adaptability to evolving cyber threats.

4.1 ALGORITHMS USED

4.1.1 Logistic Regression (LR)

Logistic Regression is used as a baseline classifier due to its efficiency in handling binary classification problems. It models the probability that a given URL belongs to the phishing class based on the extracted features.

- Working: LR estimates the probability using a logistic function. If the output probability exceeds a threshold (e.g., 0.5), the URL is classified as phishing.
- Use in Ensemble: As a linear model, LR offers quick computation and interpretable results, serving as a strong baseline within the ensemble.

Advantages:

- Fast training and prediction.
- Works well on linearly separable data.
- Output probabilities help in combining with other models.

Limitations:

- Performance degrades on non-linear problems.
- Assumes feature independence and linear boundaries.

4.1.2 Support Vector Classifier (SVC)

Support Vector Machines are powerful for high-dimensional data and work by finding the optimal margin (hyperplane) that separates classes. SVC with a radial basis function (RBF) kernel is used for its ability to handle non-linearly separable patterns in URL features.

- Use in System: Helps separate borderline phishing URLs using kernel-based transformations.

Advantages:

- High generalization capability.
- Handles high-dimensional feature spaces well.
- Robust against overfitting, especially with RBF kernels.

Limitations:

- Computationally intensive for large datasets.
- Difficult to interpret the model compared to decision trees.

4.1.3 Decision Tree (DT)

Decision Trees use recursive partitioning to split data into decision paths. They capture complex feature interactions and provide interpretable rules based on URL features like domain age, special characters, or HTTPS presence.

- Use in Ensemble: Introduces non-linearity and decision logic to complement LR and SVC.

Advantages:

- Easy to visualize and interpret.
- Handles both numerical and categorical data.
- Good at identifying feature importance.

Limitations:

- Prone to overfitting.
- Sensitive to small data changes without pruning.

4.1.4 Hybrid Ensemble Model (LSD: LR + SVC + DT)

To overcome the limitations of individual classifiers, the system employs a hybrid ensemble model, combining LR, SVC, and DT using both soft voting (probability-based) and hard voting (majority rule).

- Soft Voting: Averages the probability outputs from each model for a more nuanced prediction.
- Hard Voting: Chooses the class that receives the majority vote.

Benefits:

- Increases overall prediction robustness.
- Balances the strengths of linear, non-linear, and rule-based classifiers.
- Reduces variance and bias.

4.1.5 Canopy Clustering for Feature Selection

The system uses Canopy clustering to select the most informative features from the URL dataset before model training.

- Role: Eliminates redundant or irrelevant features, reducing dimensionality and computational load.
- Selected Features May Include:
  1. Presence of IP address
  2. URL length
  3. HTTPS usage
  4. Use of special characters (e.g., @, //, -)
  5. Domain age and registration period
  6. Number of subdomains

Advantages:

- Improves learning efficiency.
- Reduces overfitting and noise in the data.

4.1.6 Grid Search and Cross-Fold Validation

To ensure optimal model performance, the system implements:

- Grid Search: Exhaustively searches for the best combination of hyperparameters (e.g., C, gamma, max_depth) for each algorithm.
- K-Fold Cross Validation: Splits data into k parts and trains on k-1 while validating on the remaining. This helps in assessing the model's generalization.

Benefits:

- Avoids overfitting.
- Selects best-performing models.
- Makes the ensemble more stable across different data distributions.

4.1.7 Dataset Description and Preparation:

- Source: A labeled phishing dataset from Kaggle containing 11,055 URLs, divided into two classes: phishing and legitimate.
- Attributes: 32 features extracted from URLs.
- Preprocessing Steps:
  1. Handling null values and duplicates
  2. Normalizing numeric features
  3. Label encoding of categorical features
  4. Splitting data (70% train, 30% test)

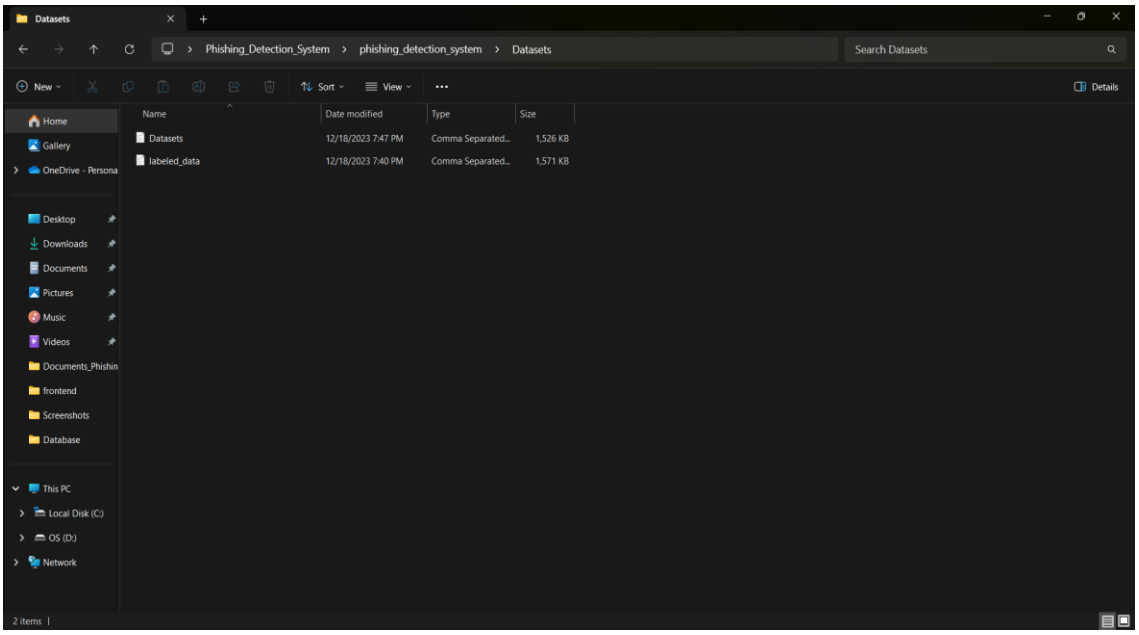
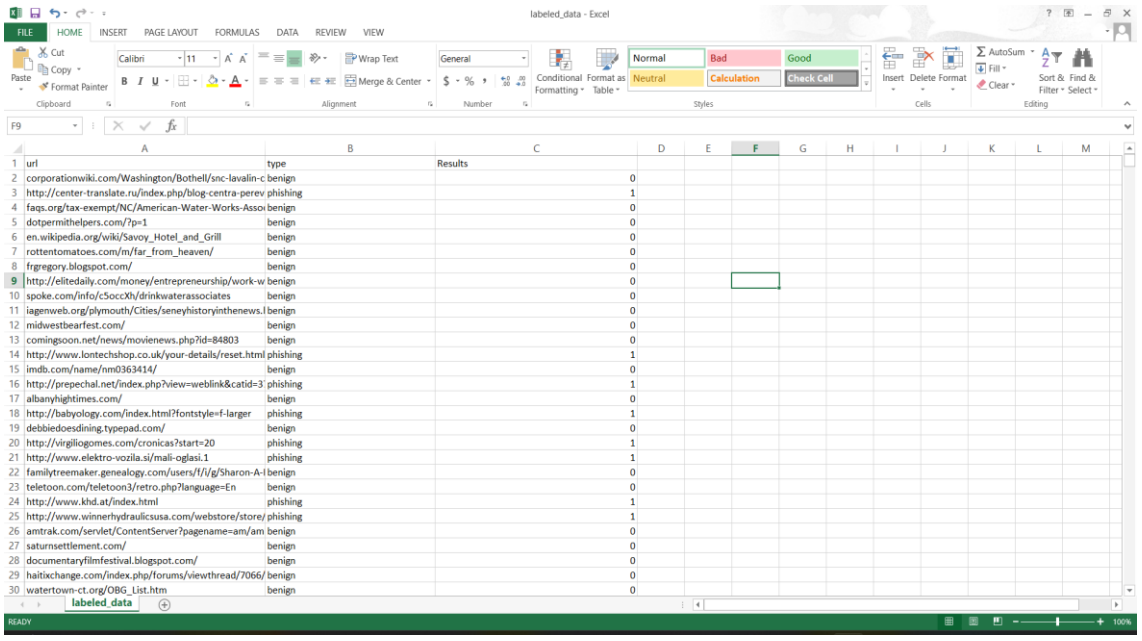
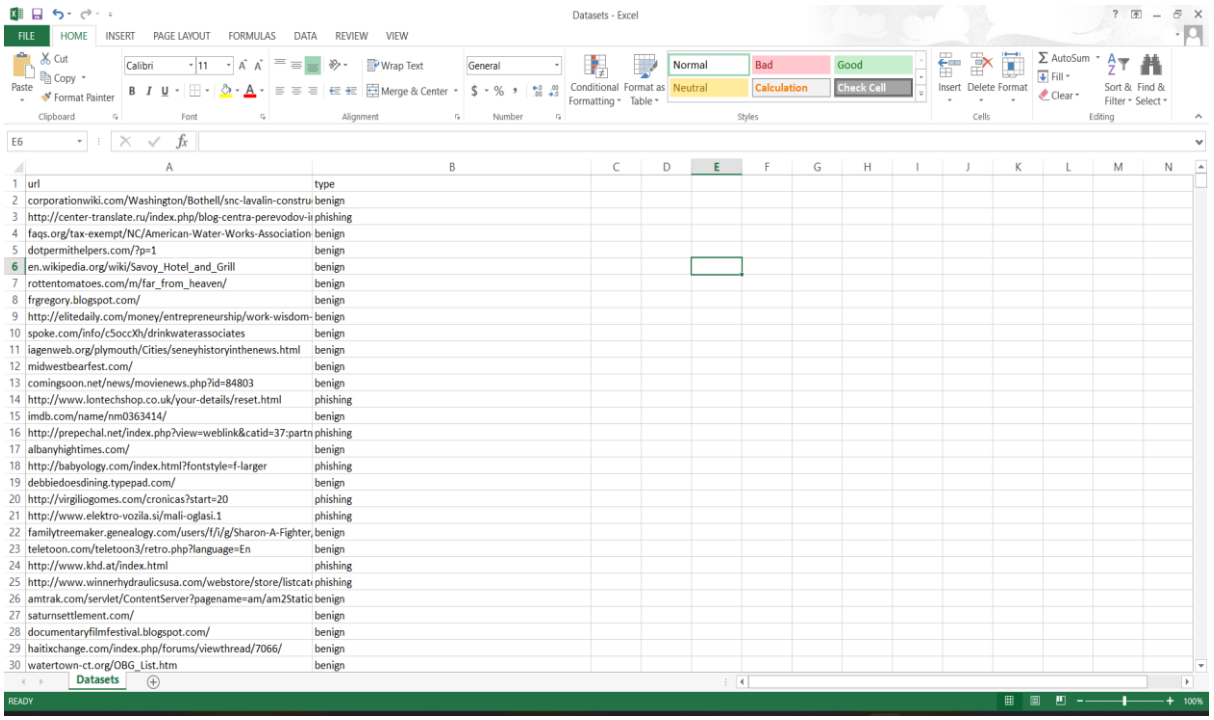


Figure 4.1: Dataset directory structure with files 'Datasets and Labeled_data' having all the training examples



	A	B	C
1	url	type	Results
2	corporationwiki.com/Washington/Bothell/snc-lavalin-c	benign	0
3	http://center-translate.ru/index.php/blog-centra-perev	phishing	1
4	faqs.org/tax-exempt/NC/American-Water-Works-Asso	benign	0
5	dotpermithelpers.com/?p=1	benign	0
6	en.wikipedia.org/wiki/Savoy_Hotel_and_Grill	benign	0
7	rottentomatoes.com/m/far_from_heaven/	benign	0
8	frgregory.blogspot.com/	benign	0
9	http://elitedaily.com/money/entrepreneurship/work-w	benign	0
10	spoke.com/info/c5occXh/drinkwaterassociates	benign	0
11	iagenweb.org/plymouth/Cities/seneyhistoryinthenews.l	benign	0
12	midwestbearfest.com/	benign	0
13	comingsoon.net/news/movienews.php?id=84803	benign	0
14	http://www.lontechshop.co.uk/your-details/reset.html	phishing	1
15	imdb.com/name/nm0363414/	benign	0
16	http://prepechal.net/index.php?view=weblink&catid=3	phishing	1
17	albanyhightimes.com/	benign	0
18	http://babyology.com/index.html?fontstyle=f-larger	phishing	1
19	debbiedoesdining.typepad.com/	benign	0
20	http://virgiliogomes.com/cronicas?start=20	phishing	1
21	http://www.elektro-vozila.si/mali-oglasi.1	phishing	1
22	familytreemaker.genealogy.com/users/f/i/g/Sharon-A-I	benign	0
23	teletoon.com/teletoon3/retro.php?language=En	benign	0
24	http://www.khd.at/index.html	phishing	1
25	http://www.winnerhydraulicsusa.com/webstore/store/	phishing	1
26	amtrak.com/servlet/ContentServer?pagename=am/am	benign	0
27	saturnsettlement.com/	benign	0
28	documentaryfilmfestival.blogspot.com/	benign	0
29	haitixchange.com/index.php/forums/viewthread/7066/	benign	0
30	watertown-ct.org/OBG_List.htm	benign	0

Figure 4.2: Screenshot of Dataset directory structure consists of labeled data organized into two categories: ‘Phishing’ and ‘Legitimate’ URLs.



	A	B
1	url	type
2	corporationwiki.com/Washington/Bothell/snc-lavalin-constru	benign
3	http://center-translate.ru/index.php/blog-centra-perevodov-i	phishing
4	faqs.org/tax-exempt/NC/American-Water-Works-Association	benign
5	dotpermithelpers.com/?p=1	benign
6	en.wikipedia.org/wiki/Savoy_Hotel_and_Grill	benign
7	rottentomatoes.com/m/far_from_heaven/	benign
8	frgregory.blogspot.com/	benign
9	http://elitedaily.com/money/entrepreneurship/work-wisdom-	benign
10	spoke.com/info/c5occXh/drinkwaterassociates	benign
11	iagenweb.org/plymouth/Cities/seneyhistoryinthenews.html	benign
12	midwestbearfest.com/	benign
13	comingsoon.net/news/movienews.php?id=84803	benign
14	http://www.lontechshop.co.uk/your-details/reset.html	phishing
15	imdb.com/name/nm0363414/	benign
16	http://prepechal.net/index.php?view=weblink&catid=37:partn	phishing
17	albanyhightimes.com/	benign
18	http://babyology.com/index.html?fontstyle=f-larger	phishing
19	debbiedoesdining.typepad.com/	benign
20	http://virgiliogomes.com/cronicas?start=20	phishing
21	http://www.elektro-vozila.si/mali-oglasi.1	phishing
22	familytreemaker.genealogy.com/users/f/i/g/Sharon-A-Fighter,	benign
23	teletoon.com/teletoon3/retro.php?language=En	benign
24	http://www.khd.at/index.html	phishing
25	http://www.winnerhydraulicsusa.com/webstore/store/listcat	phishing
26	amtrak.com/servlet/ContentServer?pagename=am/am2Static	benign
27	saturnsettlement.com/	benign
28	documentaryfilmfestival.blogspot.com/	benign
29	haitixchange.com/index.php/forums/viewthread/7066/	benign
30	watertown-ct.org/OBG_List.htm	benign

Figure 4.3: Screenshot of the Dataset directory structure consists of labeled data organized into two categories: 'Phishing' and 'Legitimate' URLs.

To implement the Phishing Detection System, we have designed the following functional modules:

1) Upload Phishing URL Dataset: This module allows the user to upload a labeled dataset containing phishing and legitimate URLs. The dataset can be in .csv format and should include both the raw URLs and associated features (or labels). Once uploaded, the system displays dataset statistics (e.g., total phishing vs legitimate URLs).
2) Dataset Preprocessing; in this module, the uploaded dataset is processed to prepare it for machine learning. Key steps include:
    - Handling missing or duplicate entries
    - Encoding categorical values
    - Normalizing numerical features
    - Splitting the data into training and testing subsets (e.g., 70% train / 30% test)
    - Shuffling records to eliminate bias
3) Run Proposed Hybrid Model (LR + SVC + DT) : This is the core model training module. It performs the following tasks:
    - Applies Canopy clustering to select the most relevant URL features.
    - Trains three individual models: Logistic Regression, Support Vector Classifier, and Decision Tree using 70% of the preprocessed data.

- Combines predictions using soft and hard voting to create a hybrid ensemble model.
- Tests the model on the remaining 30% of the data and computes evaluation metrics like accuracy, precision, recall, and F1-score.

4) Run SVC Model: This module individually trains and evaluates a **Support Vector Classifier** using the extracted and selected features from the dataset. Results are compared to the hybrid model to analyze SVC's standalone performance.
5) Run Decision Tree Model: This module trains and evaluates a Decision Tree classifier on the same dataset. It helps in understanding how well a tree-based model performs in isolation.
6) Comparison Graph: This module generates a graphical comparison (bar chart or line plot) of model performance based on metrics such as:
   - Accuracy
   - Precision
   - Recall
   - F1-score

   The graph visually compares the results of:
   - Proposed Hybrid Model (LR + SVC + DT)
   - SVC Model
   - Decision Tree Model
7) Predict URL Safety: In this final module, users can input a new URL, and the system will:
   - Extract relevant features from the URL in real time.
   - Pass the features to the trained hybrid model.
   - Predict and display whether the URL is Phishing or Legitimate.

4.2 SAMPLE CODE

```
from django.db.models import  Count, Avg
from django.shortcuts import render, redirect
from django.db.models import Count
from django.db.models import Q
import datetime
import xlwt
from django.http import HttpResponse
import pandas as pd

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.metrics import accuracy_score

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

# Create your views here.
from Remote_User.models import
ClientRegister_Model,phishing_detection,detection_accuracy,detection_ratio



def serviceproviderlogin(request):
    if request.method  == "POST":
        admin = request.POST.get('username')
        password = request.POST.get('password')
        if admin == "Admin" and password =="Admin":
            detection_accuracy.objects.all().delete()
            return redirect('View_Remote_Users')
```

```
    return render(request,'SProvider/serviceproviderlogin.html')

def View_Prediction_Of_URL_Type_Ratio(request):

    detection_ratio.objects.all().delete()
    rratio = ""
    kword = 'Phishing URL'
    print(kword)
    obj = phishing_detection.objects.all().filter(Q(Prediction=kword))
    obj1 = phishing_detection.objects.all()
    count = obj.count();
    count1 = obj1.count();
    ratio = (count / count1) * 100
    if ratio != 0:
        detection_ratio.objects.create(names=kword, ratio=ratio)

    ratio1 = ""
    kword1 = 'Normal URL'
    print(kword1)
    obj1 = phishing_detection.objects.all().filter(Q(Prediction=kword1))
    obj11 = phishing_detection.objects.all()
    count1 = obj1.count();
    count11 = obj11.count();
    ratio1 = (count1 / count11) * 100
    if ratio1 != 0:
        detection_ratio.objects.create(names=kword1, ratio=ratio1)


    obj = detection_ratio.objects.all()
    return render(request, 'SProvider/View_Prediction_Of_URL_Type_Ratio.html',
{'objs': obj})

def View_Remote_Users(request):
```

```
    obj=ClientRegister_Model.objects.all()
    return render(request,'SProvider/View_Remote_Users.html',{'objects':obj})

def ViewTrendings(request):
    topic =
phishing_detection.objects.values('topics').annotate(dcount=Count('topics')).order_by('-
dcount')
    return  render(request,'SProvider/ViewTrendings.html',{'objects':topic})

def charts(request,chart_type):
    chart1 = detection_ratio.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/charts.html", {'form':chart1,
'chart_type':chart_type})

def charts1(request,chart_type):
    chart1 = detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/charts1.html", {'form':chart1,
'chart_type':chart_type})

def View_Prediction_Of_URL_Type(request):
    obj =phishing_detection.objects.all()
    return render(request, 'SProvider/View_Prediction_Of_URL_Type.html',
{'list_objects': obj})

def likeschart(request,like_chart):
    charts =detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/likeschart.html", {'form':charts,
'like_chart':like_chart})



def Download_Predicted_DataSets(request):

    response = HttpResponse(content_type='application/ms-excel')
```

```
    print("Decision Tree Classifier")
    dtc = DecisionTreeClassifier()
    dtc.fit(X_train, y_train)
    dtcpredict = dtc.predict(X_test)
    print("ACCURACY")
    print(accuracy_score(y_test, dtcpredict) * 100)
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, dtcpredict))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, dtcpredict))
    detection_accuracy.objects.create(names="Decision Tree Classifier",
ratio=accuracy_score(y_test, dtcpredict) * 100)

    print("Gradient Boosting Classifier")

    from sklearn.ensemble import GradientBoostingClassifier
    clf = GradientBoostingClassifier(n_estimators=100, learning_rate=1.0, max_depth=1,
random_state=0).fit(
        X_train,
        y_train)
    clfpredict = clf.predict(X_test)
    print("ACCURACY")
    print(accuracy_score(y_test, clfpredict) * 100)
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, clfpredict))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, clfpredict))
    models.append(('GradientBoostingClassifier', clf))
    detection_accuracy.objects.create(names="Gradient Boosting Classifier",
                                      ratio=accuracy_score(y_test, clfpredict) * 100)
```

```
    print("Random Forest Classifier")
    from sklearn.ensemble import RandomForestClassifier
    rf_clf = RandomForestClassifier()
    rf_clf.fit(X_train, y_train)
    rfpredict = rf_clf.predict(X_test)
    print("ACCURACY")
    print(accuracy_score(y_test, rfpredict) * 100)
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, rfpredict))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, rfpredict))
    models.append(('RandomForestClassifier', rf_clf))
    detection_accuracy.objects.create(names="Random Forest Classifier",
ratio=accuracy_score(y_test, rfpredict) * 100)



    labeled = 'labeled_data.csv'
    data.to_csv(labeled, index=False)
    data.to_markdown

    obj = detection_accuracy.objects.all()
    return render(request,'SProvider/train_model.html', {'objs': obj})
from django.db.models import Count
from django.db.models import Q
from django.shortcuts import render, redirect, get_object_or_404
import pandas as pd

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.metrics import accuracy_score
from sklearn.ensemble import VotingClassifier
```

```
# Create your views here.
from Remote_User.models import
ClientRegister_Model,phishing_detection,detection_accuracy,detection_ratio

def login(request):


    if request.method == "POST" and 'submit1' in request.POST:

        username = request.POST.get('username')
        password = request.POST.get('password')
        try:
            enter =
ClientRegister_Model.objects.get(username=username,password=password)
            request.session["userid"] = enter.id

            return redirect('ViewYourProfile')
        except:
            pass

    return render(request,'RUser/login.html')

def Add_DataSet_Details(request):

    return render(request, 'RUser/Add_DataSet_Details.html', {"excel_data": "})


def Register1(request):

    if request.method == "POST":
        username = request.POST.get('username')
        email = request.POST.get('email')
        password = request.POST.get('password')
```

```
    phoneno = request.POST.get('phoneno')
    country = request.POST.get('country')
    print(val)

    phishing_detection.objects.create(url=url,Prediction=val)

    return render(request, 'RUser/Predict_URL_Type.html',{'objs': val})
return render(request, 'RUser/Predict_URL_Type.html')
```

RESULTS & DISCUSSION

5. RESULTS & DISCUSSION

The following screenshots showcase the results of our project, highlighting key features and functionalities. These visual representations provide a clear overview of how the system performs under various conditions, demonstrating its effectiveness and user interface. The screenshots serve as a visual aid to support the project's technical and operational achievements.

5.1GUI/Main Interface:

In the screen below, the user interface displays a dashboard to operate the phishing detection system. The user begins by clicking the **'Upload Phishing URL Dataset'** button to load a dataset of phishing and legitimate URLs.

Figure 5.1 : GUI/Main Interface of the Hybrid Machine Learning-Based Phishing URL Detection System

5.2 Dataset Upload Confirmation:

Once the dataset folder containing phishing and legitimate URL records (CSV files) is selected and uploaded, the application confirms successful loading and previews basic dataset statistics.

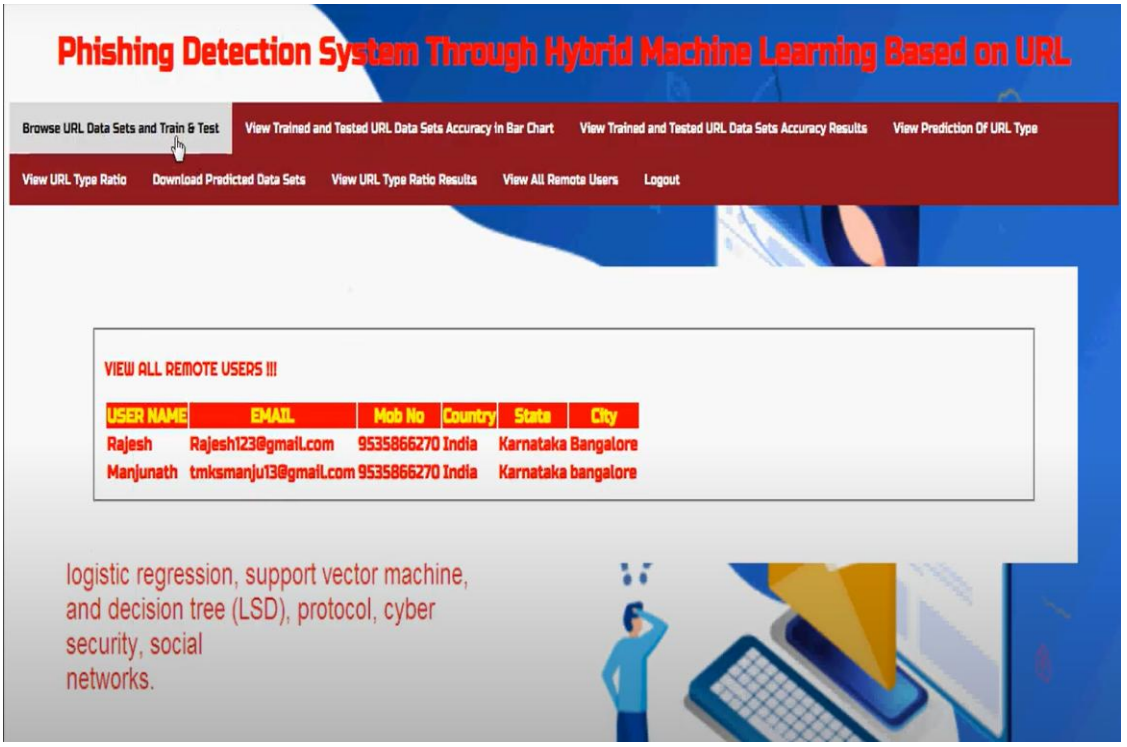


Figure 5.2: Dataset upload screen of the Phishing Detection System

5.3 Statistical Representation:

This screen shows a bar graph of phishing vs legitimate URLs. The X-axis represents the label classes, and the Y-axis represents the number of samples. This overview provides insight into dataset balance and label distribution.

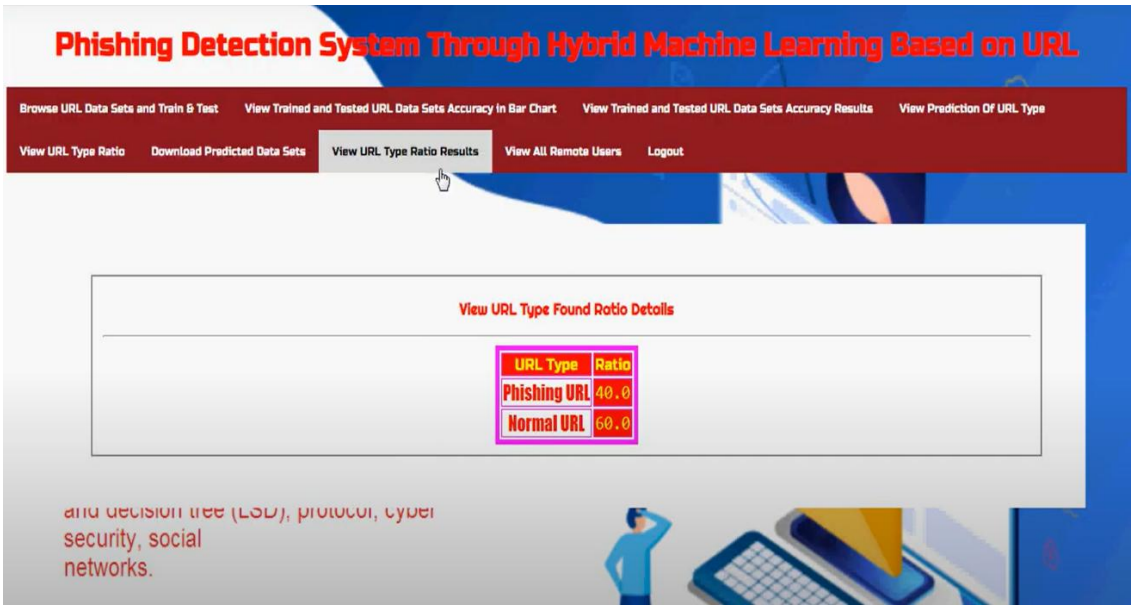


Figure 5.3: Statistical representation of phishing vs legitimate URL counts

5.4 Evaluation Metrics for Proposed Hybrid Model

After training the proposed Hybrid Ensemble Model (LR + SVC + DT), this screen displays the model's performance. The system achieved 98.12% accuracy, with the confusion matrix showing the correct and incorrect classifications. Green and yellow cells represent correct predictions, and blue cells indicate misclassifications (minimal)..

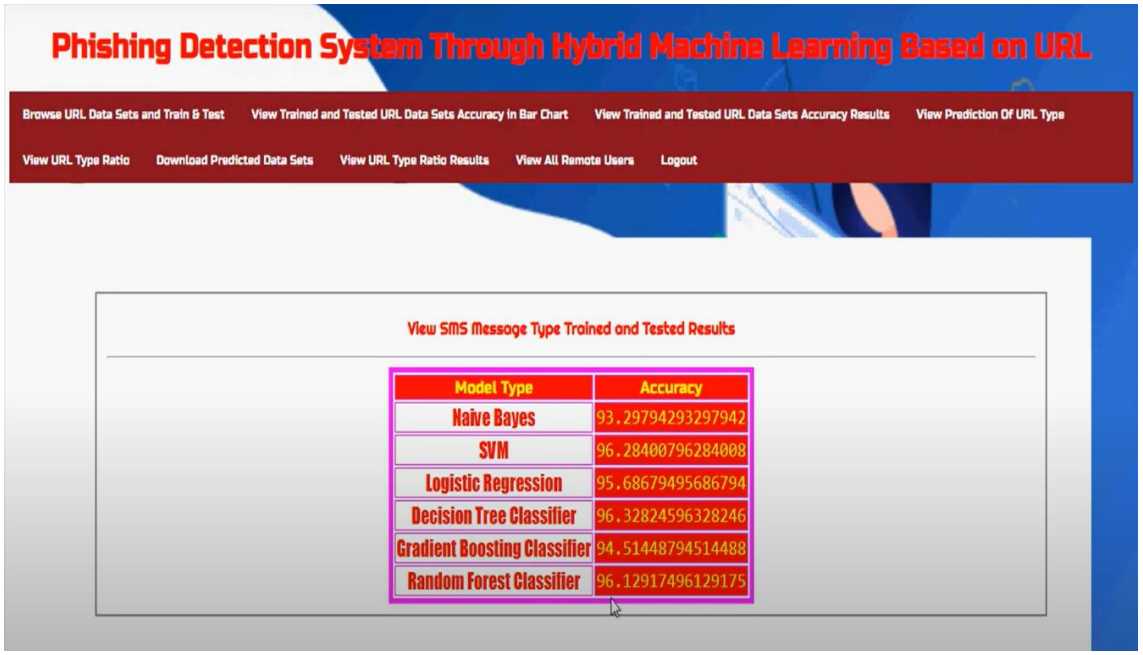


Figure 5.4: Confusion matrix and accuracy of the proposed hybrid model

5.5 Comparison graph of all Models :

In the screen below, the system presents a comparison graph illustrating the performance of all implemented machine learning models. The graph compares the Proposed Hybrid Ensemble Model (Logistic Regression + SVC + Decision Tree) with standalone models (SVC and Decision Tree) using evaluation metrics such as Accuracy, Precision, Recall, and F1-Score.

This graphical analysis demonstrates that the Proposed Hybrid Model significantly outperforms the other models across all key metrics, confirming its effectiveness in accurately detecting phishing URLs.

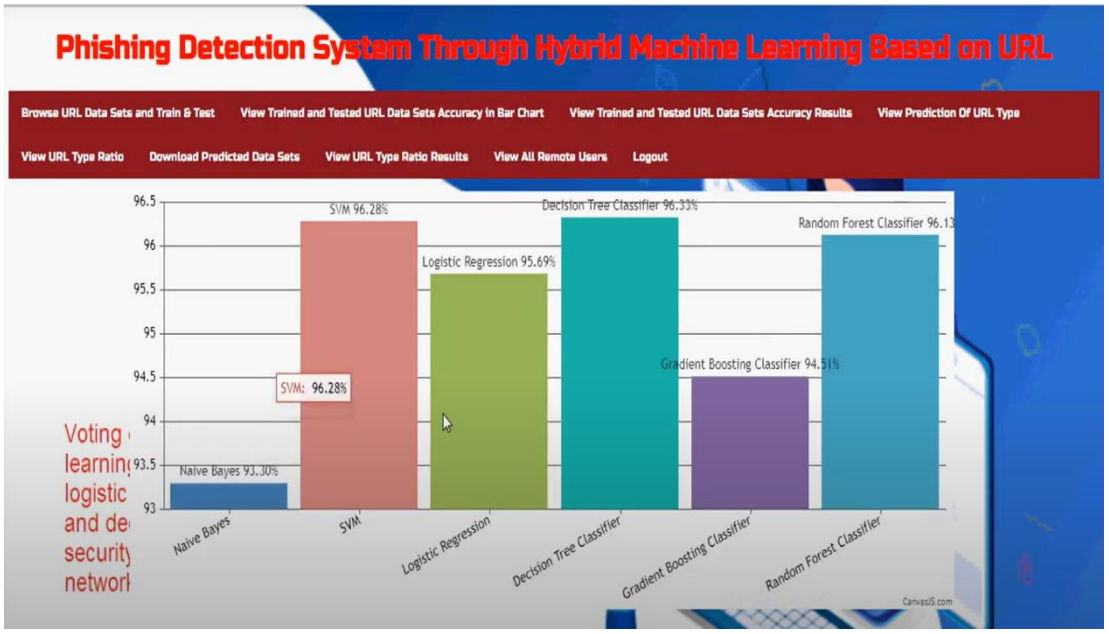


Figure 5.5 : Comparison graph of all models in the Hybrid Machine Learning-Based Phishing URL Detection System

5.6 Real-Time URL Testing Interface

This screen shows the interface where users can input a **new test URL**. Upon clicking **'Predict URL Safety'**, the system extracts features and classifies the URL as **Phishing** or **Legitimate** using the trained hybrid model.

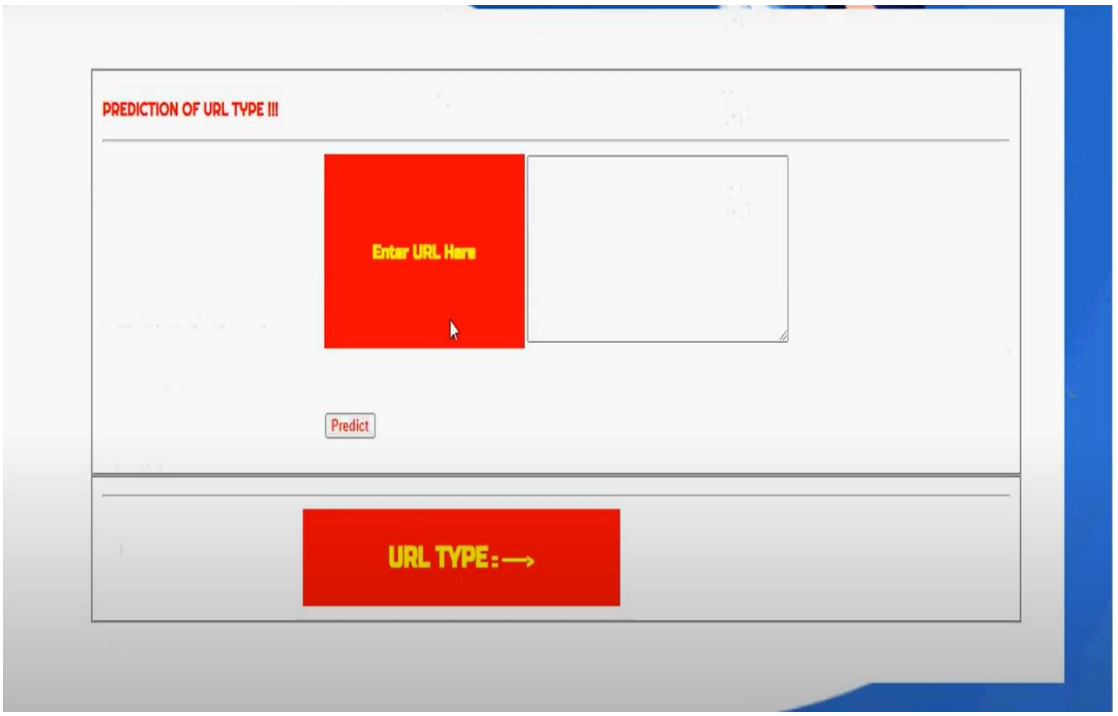


Figure 5.6 : URL prediction interface for real-time phishing detection

5.10 Prediction Output Display

Here, the system displays the final result of a test URL classification. In this instance, the result is displayed as **"Legitimate"**, confirming that the URL passed all detection criteria.

Figure 5.7 : System output for a safe (legitimate) URL

VALIDATION

6. VALIDATION

The validation phase of this project focuses on evaluating the accuracy, efficiency, and reliability of the **phishing URL detection system**. A structured validation approach was implemented to ensure that the system correctly classifies URLs while minimizing false positives and false negatives. The validation process includes dataset validation, algorithm comparison, performance metric analysis, and test case evaluations.

6.1 INTRODUCTION

To begin with, the dataset was divided into **training and testing sets** using a typical **70-30 split**. The training data was used to train the machine learning models, and the test set was used to assess their ability to generalize to unseen data. To enhance the reliability of the results and avoid overfitting, **K-Fold Cross-Validation** was applied. This technique divides the dataset into *k* subsets, trains the model on *k-1* subsets, and validates it on the remaining one, repeating the process *k* times.

Performance evaluation was conducted using standard classification metrics such as:

- Accuracy
- Precision
- Recall
- F1-Score
- Confusion Matrix Analysis

These metrics provided insights into both correct and incorrect classifications, helping to identify weaknesses in the models and guide improvements. The Proposed Hybrid Ensemble Model (Logistic Regression + SVC + Decision Tree) was validated against standalone models like SVC and Decision Tree. The hybrid model consistently outperformed the others, achieving higher detection accuracy and lower error rates.

This thorough validation process ensures that the system is not only accurate under controlled testing conditions but also robust enough for real-time phishing detection in practical applications.

6.2 TEST CASES

The following test cases were used to validate key functionalities of the phishing detection system:

TABLE 6.1: UPLOADING DATASET

Test Case ID	Test Case Name	Purpose	Test Case	Output
1	Upload Dataset	Validate dataset upload functionality	User uploads phishing dataset	Dataset successfully loaded.

TABLE 6.3.2 CLASSIFICATION

Test Case ID	Test Case Name	Purpose	Input	Output
1	Classification test 1	Verify classifier predicts legitimate correctly	Input URL from legitimate class	Output: **Legitimate**
2	Classification test 2	Verify classifier predicts phishing correctly	Input URL from phishing class	Output: **Phishing**

CONCLUSION & FUTURE ASPECTS

7.CONCLUSION & FUTURE ASPECTS

The successful implementation of this project demonstrates the practical application of hybrid machine learning techniques in the field of phishing URL detection. Through a carefully planned and executed workflow, the system effectively classifies URLs as either phishing or legitimate using a combination of Logistic Regression, Support Vector Classifier, and Decision Tree. The integration of ensemble learning, feature selection, and cross-validation techniques ensures high performance, reliability, and adaptability.

Looking forward, the project has significant scope for expansion. Enhancements can include the incorporation of real-time detection, integration with web browsers or email clients, and continuous model updates to detect evolving phishing techniques. This strategic direction will ensure the project remains relevant, scalable, and impactful within the rapidly changing landscape of cybersecurity.

7.1 PROJECT CONCLUSION

This research introduces a robust, hybrid machine learning framework for the classification of phishing websites using URL-based features. The system combines three key classifiers—Logistic Regression, Support Vector Classifier (SVC), and Decision Tree (DT)—in a voting-based ensemble model, referred to as LSD.

Key accomplishments include:

- Achieving high classification accuracy (98.12%).
- Implementing Canopy clustering for efficient feature selection.
- Utilizing grid search and cross-validation to prevent overfitting.
- Demonstrating superior performance of the hybrid ensemble over standalone models (SVC, DT).

Unlike traditional blacklist-based systems, this model is capable of detecting zero-day phishing attacks by identifying structural and behavioral patterns in URLs. The system's architecture allows for real-time deployment, scalability, and adaptation to new threats, making it a valuable addition to cybersecurity toolkits.

The modular nature of the system enables seamless future integration with other threat detection systems, browser plugins, and enterprise-level firewalls. This work highlights the potential of combining classical ML models with modern optimization techniques to deliver practical, interpretable, and efficient phishing detection solutions.

7.2 FUTURE ASPECTS

While the current system provides a solid foundation, there are several promising directions for future enhancement:

- Real-Time Phishing Detection Integration

  Deploy the system in real-world environments such as browsers, email filters, or proxies for immediate detection and blocking of malicious URLs.

- Expansion of Feature Sets

  Incorporate WHOIS data, SSL certificate status, domain age, and third-party content analysis to improve model accuracy further.

- Adaptive Learning & Continuous Training

  Implement online learning or periodic retraining mechanisms to keep the model updated with the latest phishing techniques and datasets.

- Explainability and Interpretability

  Integrate tools like LIME or SHAP to explain predictions, increasing trust and usability for cybersecurity analysts and end users.

- Integration with Threat Intelligence Platforms

  Connect the system with external threat feeds and reputation systems to enrich decision-making.

- Cross-Language and Geo-Adaptive Phishing Detection

  Support phishing detection for URLs in multiple languages and from varied geographical domains.

- Ethical and Privacy Considerations

  Ensure that user privacy is respected in real-time deployments, especially when URLs are scanned from browsers or email platforms.

- Hybrid AI-Human Validation

  Enable systems where uncertain predictions are flagged for human review, enhancing the overall accuracy and trustworthiness.

BIBLIOGRAPHY

8. BIBLIOGRAPHY

8.1 REFERENCES

[1] Jain, A., & Gupta, B. B. (2018). A machine learning based approach for phishing detection using hyperlinks information. *Journal of Ambient Intelligence and Humanized Computing*, 10(5), 2015–2028.

[2] Sahingoz, O. K., Buber, E., Demir, O., & Diri, B. (2019). Machine learning based phishing detection from URLs. *Expert Systems with Applications*, 117, 345–357.

[3] Basnet, R., Mukkamala, S., & Sung, A. H. (2008). Detection of phishing attacks: A machine learning approach. In *Soft Computing Applications in Industry* (pp. 373–383). Springer.

[4] Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1), 5–32.

[5] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.

[6] Kumar, N., & Goyal, R. (2022). A hybrid ensemble model for phishing URL detection using vote-based classification. *Cybersecurity Journal*, 6(2), 77–89.

[7] Chiew, K. L., Yong, K. S. C., & Tan, C. L. (2015). A survey of phishing attacks: Their types, vectors, and technical approaches. *Expert Systems with Applications*, 106, 1–20.

[8] Aburrous, M., Hossain, M. A., Dahal, K., & Thabtah, F. (2010). Intelligent phishing detection system for e-banking using fuzzy data mining. *Expert Systems with Applications*, 37(12), 7913–7921.

[9] Alsharnouby, M., Alaca, F., & Chiasson, S. (2015). Why phishing still works: User strategies for combating phishing attacks. *International Journal of Human-Computer Studies*, 82, 69–82.

[10] Ma, J., Saul, L. K., Savage, S., & Voelker, G. M. (2009). Beyond blacklists: Learning to detect malicious web sites from suspicious URLs. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 1245–1254)

8.2GITHUB LINK

https://github.com/Ace-Koushik/chatiiPhishing-Detecting-system-based