Phishing Website Detection Using Machine Learning

Gholap Hrishikesh Appasaheb[1] Prof. Vishwatej Pisal[2]
[1]Student [2]Assistant Professor
[1,2]Master of Computer Applications
[1,2]Anantrao Pawar College of Engineering & Research, Pune Affiliated to Savitribai Phule Pune University, India

Abstract — With the rapid growth of internet usage, cybercrime has also increased, especially phishing attacks, where fake websites are created to look like real and trusted websites in order to steal users' passwords, bank details, and personal information. Traditional phishing detection methods based on fixed rules are no longer very effective because attackers keep changing their techniques. This project presents a machine-learning-based web application that can quickly check whether a website URL is safe or phishing in real time. The system analyses different features of a website, such as the URL structure, HTTPS security, domain age, and website traffic information, and then uses four machine learning algorithms — Decision Tree, Random Forest, Logistic Regression, and Support Vector Machine to classify the website. Among these, the Random Forest algorithm gave the best performance with higher accuracy and better detection results. The proposed system is lightweight, easy to use, and accessible through a web browser, making it a practical solution to improve online safety for both individuals and organisations.

Keywords: Phishing Website Detection; Machine Learning-Based Cybersecurity; URL Analysis; Random Forest Classification; Real-Time Threat Detection; Web Security Application;

I. Introduction

The internet has woven itself into nearly every dimension of modern life — commerce, healthcare, education, banking, and social interaction all depend on it in ways that would have seemed futuristic only two decades ago. Yet this very dependence creates opportunity for malicious actors. Among the many threats lurking online, phishing remains uniquely dangerous because it exploits the most difficult vulnerability to patch: human trust.

A phishing website is a fraudulent digital copy of a trusted service — a bank, an e-commerce platform, a government portal — engineered to look authentic enough to deceive a visitor into entering sensitive information. Once submitted, those credentials are harvested by attackers who can then drain bank accounts, commit identity fraud, or sell the data on underground markets. According to cybersecurity industry reports, phishing continues to rank among the top vectors for data breaches worldwide, accounting for a significant proportion of financial losses suffered by both individuals and enterprises each year.

Historically, anti-phishing defences relied on manually maintained blacklists of known malicious domains, pattern-matching heuristics, and browser warnings. These approaches have a fundamental weakness: they react to threats that have already been identified. Sophisticated attackers can spin up a new phishing domain, harvest credentials for a few hours, and abandon it before any blacklist is updated. What is needed is a proactive detection mechanism capable of evaluating an unknown URL and rendering a judgment based on its intrinsic characteristics rather than its reputation history.

II. Problem Statement

Despite decades of awareness campaigns and technological countermeasures, phishing attacks remain stubbornly effective. Several factors contribute to this persistence:

- Attackers continuously adapt their URL structures and page layouts to evade detection by conventional filters.
- Short-lived domains — created and discarded within hours — defeat blacklist-based approaches before those lists can be updated.
- Visual mimicry has become so sophisticated that trained security professionals can be fooled by a carefully crafted phishing page.
- The enormous volume of new URLs registered daily makes manual review entirely impractical at scale.

III. Literature Survey

A substantial body of research has explored the application of machine learning to phishing detection. Early work by Mohammad et al. (2014) proposed a hybrid rule-and-learning approach using 30 URL and page features, achieving detection rates above 92% with a neural network classifier. Subsequent studies by Sahoo et al. (2017) systematically surveyed deep learning methods for malicious URL detection, finding that recurrent models could capture sequential character-level patterns that elude feature-engineered approaches.

Random Forest classifiers have attracted particular attention due to their robustness against overfitting and their natural ability to rank feature importance. Zouina and Outtaj (2017) demonstrated that Random Forest consistently outperformed single decision trees and logistic regression across several benchmark phishing datasets, a finding echoed by numerous subsequent studies.

IV. System Design and Architecture

The proposed system follows a three-tier web application architecture composed of a presentation layer, a processing layer, and a data layer. Figure 1 below summarises the overall flow.

Frontend (User Interface)	Backend Processing Engine	Database Layer
HTML / CSS / Bootstrap / JavaScript	Python + Flask / Express.js	MySQL
URL input form, result display	Feature extraction, ML model	Website records, prediction logs

Runs in web browser	Serves REST API endpoints	Stores training data and results

When a user submits a URL through the frontend interface, the request is forwarded to the backend processing engine. The engine orchestrates three sequential steps: feature extraction, model inference, and result delivery. The extracted features are passed to the pre-trained machine learning model, which returns a classification label and confidence score. This result is then rendered on the frontend and optionally stored in the database for audit purposes.

V. Feature Extraction

The discriminating power of any machine learning classifier depends critically on the quality of the features presented to it. For phishing detection, features can be derived from three primary sources: the URL string itself, the WHOIS domain registration record, and third-party traffic and reputation services. The following subsections describe the most informative features employed in this system.

A. URL-Based Features

1) URL Length – Phishing websites often use very long and confusing URLs to hide the real website address.
2) IP Address in URL – Fake websites may use numeric IP addresses instead of normal domain names.
3) Special Symbols – Symbols like "@", hyphens, or unusual characters in a URL can indicate a phishing website.
4) Too Many Subdomains – Phishing sites often use multiple subdomains to look trustworthy and confuse users.
5) HTTPS and Domain Reputation – Legitimate websites usually use HTTPS security, and trusted domain extensions are less likely to be phishing sites.

B. Domain-Based Features

- Domain Age: Phishing domains are typically registered shortly before use and abandoned quickly. A domain younger than three months warrants heightened scrutiny.
- Domain Expiry Period: Legitimate organisations tend to register domains for multiple years. A registration expiring in less than twelve months is a mild warning signal.
- DNS Record Existence: Absence of a valid DNS record for the hostname is a strong phishing indicator.

C. Traffic and Reputation Features

- Alexa Rank / Traffic Rank: Established legitimate websites generally appear in global traffic rankings. A newly created phishing domain will have no ranking or a very poor one.
- Google Index Status: Pages indexed by Google are less likely to be phishing pages, since indexing requires a degree of age and established link structure.
- Links Pointing to the Page: A very low count of inbound links from external domains indicates a recently created or isolated site.

VI. Machine Learning Classifiers

Four supervised classification algorithms were implemented and compared in this study. Each brings a different approach to the problem of separating phishing from legitimate URLs.

A. Decision Tree (DT)

Decision Tree is a machine learning algorithm that works like a flowchart to classify data. It makes decisions step by step based on different features and is easy to understand and interpret. However, very large trees can overfit the data, so techniques like pruning are used to improve accuracy.

B. Random Forest (RF)

Random Forest is a machine learning algorithm that combines multiple Decision Trees to improve prediction accuracy. Each tree is trained on different random data samples, and the final result is decided by majority voting. This method reduces overfitting, handles large datasets effectively, and provides highly accurate results, making it the best classifier for this system.

C. Logistic Regression (LR)

Logistic Regression is a machine learning algorithm that predicts the probability of a website being legitimate or phishing using input features. It is simple, fast, and effective when the data is wellstructured. The model also provides confidence scores along with predictions, helping users understand how reliable the result is.

D. Support Vector Machine (SVM)

Support Vector Machine is a machine learning algorithm that separates data into different classes by finding the best boundary between them. It works well with complex and high-dimensional data and can accurately classify phishing and legitimate websites even with smaller training datasets.

VII. Implementation

The system was implemented as a full-stack web application with a clear separation of concerns between the user interface and the analytical backend.

A. Frontend

The user interface was developed using HTML5, CSS3, Bootstrap 5, and vanilla JavaScript. The interface presents a single input field into which a user pastes or types any URL. On submission, an AJAX call is dispatched to the backend REST API endpoint so that the page need not be reloaded. The returned result — a classification label and a confidence percentage — is displayed prominently alongside a colour-coded badge: green for legitimate and red for phishing. The responsive design ensures usability across desktop and mobile devices.

B. Backend

The system backend is built using Python and Flask. It takes a URL through a POST request, extracts important features (like domain and URL details), and sends them to a pre-trained machine learning model to predict whether the site is phishing or legitimate. The model was trained on the UCI Phishing Website Dataset and saved using joblib for fast

loading. The system returns the result quickly as a JSON response.

C. Database

MySQL was selected as the relational database management system. Two primary tables are maintained: one for logging every URL submitted through the interface along with its prediction result and timestamp, and another for storing administrator-managed records from the training dataset. An admin module accessible via a protected route allows the project coordinator to view prediction logs, add new training records, and trigger model retraining.

D. Technology Stack Summary

Layer	Technology	Role
Frontend	HTML, CSS, Bootstrap, JS	User interface and interaction
Backend	Python, Flask	Feature extraction and ML inference
ML Libraries	Scikit-learn, Pandas, NumPy	Model training and prediction
Database	MySQL	Data storage and logging
Version Control	Git / GitHub	Source code management
Testing	Postman, unittest	API testing and unit testing

VIII. Experimental Results and Discussion

All four classifiers were trained on identical data splits and evaluated on the same held-out test set. Standard classification metrics — accuracy, precision, recall, and F1-score — were computed for each model. Table 3 presents the comparative results.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Decision Tree	96.2	95.8	96.5	96.1
Random Forest	97.9	97.6	98.2	97.9
Logistic Regression	92.4	91.7	93.0	92.3
Support Vector Machine	96.8	96.3	97.1	96.7

Random Forest performed the best among all models with 97.9% accuracy and F1-score, because it combines many decision trees and reduces errors, making predictions more stable and reliable.

Support Vector Machine came second with 96.8% accuracy, as it can handle complex, non-linear patterns in the data. Decision Tree ranked third with 96.2% accuracy, showing good performance but slightly less stable compared to ensemble methods.

Logistic Regression had the lowest accuracy at 92.4%, but it is still useful because it is simple, fast, and easy to interpret.

The most important features for detecting phishing websites were domain age, presence of IP address, URL length, and HTTPS status, which clearly help in identifying suspicious websites.

All models were fast, giving predictions in under 50 milliseconds, making them suitable for real-time use, while feature extraction (especially WHOIS lookup) took slightly more time.

IX. System Testing

System was tested in several ways to make sure it works correctly and reliably.

- Unit Testing: Each feature extraction function was checked using known URLs to ensure accurate results.
- Integration Testing: The full system was tested using Postman with both phishing and safe URLs to confirm proper end-to-end working.
- Cross-Validation: The model was tested multiple times using different data splits to avoid overfitting and improve reliability.
- User Testing: Non-technical users tried the web app and found it easy to use with quick responses.
- Edge Case Testing: Invalid, empty, and very long URLs were tested to ensure the system handles errors safely without crashing.

A. Scope and Limitations

system can detect phishing websites in real time using a web interface. It supports four machine learning models and uses 30 URL and domain-based features for prediction. It also includes an admin panel to manage data, retrain models, and view prediction history stored in a MySQL database.

However, there are some limitations. WHOIS lookup can slow down results on weak networks. The system does not update its features automatically with new phishing trends. It also does not analyze website images or visual content, which could improve accuracy. In addition, it only uses URL and domain information, and could be improved by adding analysis of webpage text and HTML content in the future.

X. Expected Outcomes and Future Work

project delivers a working phishing detection web app that can be used in any browser without
installation. It uses a machine learning model with over 97% accuracy, which performs better than simple rule-based systems. The study also compares four different algorithms, helping identify the best one for phishing detection.

The system helps users understand why a website is flagged by showing important features, which improves cybersecurity awareness. It is also flexible and can be easily improved in the future.

In conclusion, the system successfully detects phishing websites in real time using machine learning. Among all models tested, Random Forest performed the best. The system is fast, reliable, and built using common technologies like Python, Flask, Scikit-learn, and MySQL, making it easy to maintain and expand. While phishing threats will continue to evolve, this system provides a strong foundation that can be improved over time with new techniques and data.

REFERENCES

[1] Mohammad, R. M., Thabtah, F., & McCluskey, L. (2014). Studied how neural networks can be used to detect phishing websites. Published in Expert Systems with Applications.
[2] Sahoo, D., Liu, C., & Hoi, S. C. H. (2017). A survey paper that reviews different machine learning methods used for detecting malicious URLs.
[3] Zouina, M., & Outtaj, B. (2017). Proposed a lightweight phishing detection system using SVM and URL similarity features.
[4] Tan, C. et al. (2018). Discussed transfer learning techniques in deep learning for improving model performance.
[5] PhishTank. A community-driven platform that collects and verifies phishing website data. https://phishtank.org
[6] UCI Machine Learning Repository. Provides datasets, including the phishing websites dataset used in research. https://archive.ics.uci.edu/ml/datasets/phishing+websites
[7] Scikit-learn. A popular Python machine learning library used for building and testing models. https://scikit-learn.org
[8] Kaggle. A platform that provides datasets and machine learning competitions, including phishing datasets. https://www.kaggle.com
[9] Python Software Foundation. Official documentation for the Python programming language. https://www.python.org/doc/
[10] Krogh, J. W. (2020). Book on MySQL performance tuning that explains how to improve database speed and efficiency.

