International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 12 | Dec - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Phishing Website Detection Using Machine Learning Algorithm

Gaurav M. Shahare[1], Kiran P. Pustode[2]

[1]*Student, Department of MCA, Karanjekar College of Engineering and Management, Sakoli*
[2] *HOD, Department of MCA, Karanjekar College of Engineering and Management, Sakoli*

Abstract - Phishing websites have become a significant security concern, serving as the primary entry point for cyberattacks that compromise data confidentiality and integrity. This research proposes an automated detection method using feature extraction and machine learning to address the limitations of manual engineering and the challenge of zero-day phishing attempts. A dataset of 5,000 random phishing URLs from Phish Tank and 5,000 legitimate URLs was utilized to train various models, including Decision Trees, Random Forests, Support Vector Machines (SVM), Multilayer Perceptron (MLP), XGBoost, and Autoencoders. Features were categorized into address-based, domain-based, and HTML/JavaScript-based checks. Results indicate that XGBoost achieved the highest testing accuracy of 86.4%, followed closely by MLP at 86.3%, demonstrating the effectiveness of machine learning in real-time phishing prevention.

Key Words: Phishing Detection, Machine Learning, URL Features, Cybersecurity, XGBoost, Feature Extraction.

1. INTRODUCTION

The rapid digital transformation of global economies has shifted the landscape of financial transactions, social interactions, and data storage to the online domain. However, this shift has been accompanied by a sophisticated rise in cyber-criminality, with phishing remaining the most prevalent and damaging form of social engineering. Phishing involves the creation of fraudulent websites that mimic legitimate entities—such as banks, e-commerce platforms, or social networks—to deceive users into revealing sensitive credentials, including login passwords, credit card numbers, and personal identification information.

According to recent cybersecurity reports, phishing attacks have reached record highs, driven by the automation of attack toolkits and the increasing use of URL shortening services. Traditional defense mechanisms primarily rely on Blacklisting, where a database of known malicious URLs is maintained. While effective for known threats, blacklisting fails to detect "Zero-day" phishing attacks malicious sites that exist for only a few hours before being taken down and replaced by new, unique URLs. The complexity of modern phishing is further heightened by technical obfuscation techniques. Attackers utilize elongated URLs, "onMouseOver" JavaScript events to mask destination links, and IFrame redirection to bypass standard security filters. Consequently, there is an urgent need for an intelligent, proactive detection system that analyzes the inherent characteristics of a website rather than relying on a static list.

Motivation and Problem Statement: The primary challenge in phishing detection is the dynamic nature of the URLs. As phishers evolve, the features that distinguish a fake site from a legitimate one become more subtle. Manual feature engineering is no longer sustainable at the scale of modern internet traffic. This research is motivated by the potential of Machine Learning (ML) to automate the classification process. By extracting and analyzing URL-based, domain-based, and HTML-based features, ML models can identify patterns indicative of fraud that are invisible to the human eye or static filters.

- **Comprehensive Feature Engineering:** We identify and extract 17 distinct features categorized into address-bar, domain, and client-side JavaScript attributes.
- **Comparative Analysis:** A rigorous evaluation of six machine learning models—Decision Tree, Random Forest, Multilayer Perceptron, XGBoost, Autoencoder, and SVM—to determine the most effective classifier for real-time deployment.
- **Accuracy Optimization:** Achieving a high testing accuracy (up to 86.4% with XGBoost), demonstrating that gradient boosting techniques significantly outperform traditional linear models in detecting deceptive URLs.

2. Literature Review

The Literature Review section of research paper establishes the foundation of your study by examining the evolution of phishing detection strategies, moving from static heuristic approaches to advanced machine learning frameworks. Initially, phishing was defined as a sophisticated method for obtaining user accounts without authorization by impersonating legitimate entities. Early defensive measures focused heavily on Blacklisting and Heuristic-based approaches. For instance, Jain and Gupta proposed an auto-updated whitelist-based approach to protect against phishing, which effectively reduced false positives for known sites but struggled with the rapid emergence of new, unknown malicious URLs. Similarly, researchers like Tan et al. developed the PhishWHO system, which used N-gram models to identify the legitimate owners of a website, highlighting the early focus on identity verification through textual analysis.

As attackers became more technical, the research shifted toward URL analysis and automated feature extraction. Garera et al. were pioneers in this regard, utilizing logistic regression over a set of hand-selected features such as red-flag keywords and PageRank quality recommendations. This approach demonstrated that the structure of the URL itself its length, depth, and the presence of specific characters contained significant clues about its intent. Building on this, Mehmet et al. conducted a comprehensive study proposing URL-based detection by evaluating eight different machine learning algorithms across three diverse datasets. Their work verified that features such as domain age and hosting information are critical indicators that distinguish legitimate domains from ephemeral phishing sites.

The modern era of phishing detection is dominated by Comparative Machine Learning (ML) Studies and Ensemble

Learning. Research by Vahid Shahrivari et al. and Amani Alswailem et al. focused on identifying the most robust classifiers by testing models like K-Nearest Neighbors (KNN), AdaBoost, and Support Vector Machines (SVM). Both studies independently concluded that Random Forest provided superior accuracy due to its ability to handle high-dimensional data without overfitting. To address the challenge of data availability, Hossein et al. introduced "Fresh-Phish," an open-source framework that allowed researchers to generate large-scale, labeled datasets for real-time testing. Most recently, deep learning techniques have emerged, such as the Capsule-based Neural Networks proposed by Yong et al., which attempt to extract both shallow and deep semantic features from URLs, reflecting the current trend toward autonomous and self-learning security systems.

While these previous studies have made significant strides, many still struggle with the high computational costs of real-time detection or the high false-alarm rates associated with complex neural networks. Your research builds upon this body of work by integrating 17 high-impact features—including HTML and JavaScript-based checks—to optimize the performance of the XGBoost and Multilayer Perceptron models. By synthesizing these diverse feature sets, this study aims to provide a more comprehensive and accurate detection mechanism that overcomes the limitations identified in prior heuristic and linear models.

3. Methodology and System Architecture

The proposed methodology follows a structured machine learning pipeline designed to identify malicious intent within a URL by analyzing its structural and behavioral characteristics. The process is divided into data acquisition, pre-processing, feature engineering, and model training.

3.1 Data Collection and Preprocessing

The effectiveness of a machine learning model is heavily dependent on the quality of the dataset. For this research, we utilized a balanced dataset of 10,000 URLs.

- **Malicious Data:** 5,000 URLs were sourced from Phish Tank, an open-access clearinghouse for data on phishing attacks.
- **Legitimate Data:** 5,000 benign URLs were collected to ensure the model does not become over-biased toward malicious samples. During pre-processing, the data was cleaned to remove duplicates and null values. The dataset was then split into an 80:20 ratio—8,000 samples for training and 2,000 for testing and validation.

3.2 Feature Extraction

This is the most critical phase where the raw URL string is converted into a numerical vector (X) that can be processed by machine learning algorithms. We extracted 17 key features, categorized into three distinct types:

A. Address Bar-Based Features (10 Features):

- IP Address: Checks if the domain name is replaced by an IP address (e.g., http://192.168.0.1/).
- "@" Symbol: Presence of this symbol often redirects the browser to ignore everything before it.
- URL Length: Phishing URLs are statistically longer; we set a threshold where length ≥ 54 characters indicates suspicion.
- URL Depth: Counting the number of sub-folders (indicated by /).
- Redirection ("//"): Checking if the URL contains a redirection path.
- Prefix/Suffix ("-"): Phishers often use hyphens to mimic legitimate brands (e.g., google-login.com).
- TinyURL: Detecting if URL shortening services (like bit.ly) are used to hide the destination.

B. Domain-Based Features (4 Features):

- DNS Record: Legitimate sites must have a valid DNS record. If none exists, it is marked as phishing.
- Web Traffic: Utilizing Alexa rankings to check the popularity of the site.
- Domain Age: Malicious sites are usually short-lived. We check if the domain has been active for at least 12 months.
- Domain End: Comparing the expiration date with the current date.

C. HTML & JavaScript-Based Features (3 Features):

- IFrame Redirection: Detecting hidden frames that load malicious content.
- "onMouseOver": Checking if JavaScript is used to change the status bar to show a fake URL when the user hovers over a link.
- Right-Click Disable: Malicious sites often disable the right-click function to prevent users from viewing the page source code.

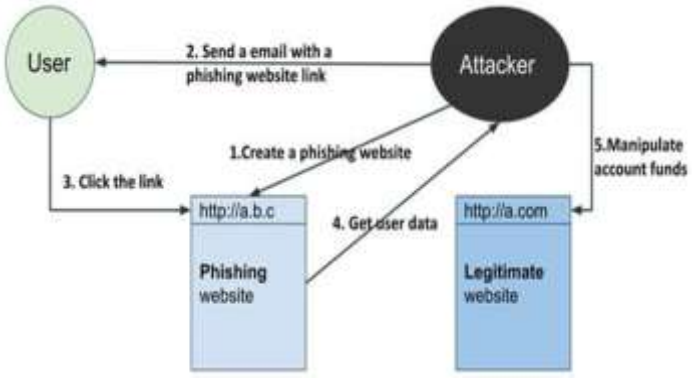

Fig -1: Overall System Flow Chart

3.3 Model Selection and Training

The feature vector is fed into six distinct algorithms to compare performance:

- XGBoost: A gradient-boosted decision tree algorithm that uses an ensemble approach to minimize loss functions.

- Multilayer Perceptron (MLP): A class of feedforward artificial neural networks that uses backpropagation for training.
- Support Vector Machine (SVM): Used to find the optimal hyperplane that separates the two classes in a high-dimensional space.

3.4 System Architecture Explanation

1. URL Input Layer: The user provides a URL string.

2. Feature Extractor Module: This module contains a library of Python functions (as seen in your code snippets) that parse the URL using urllib.parse and requests. It outputs a feature vector of 17 dimensions.

3. Preprocessing & Scaling: Numerical values are normalized to ensure that features with larger ranges (like URL length) do not dominate the model's decision-making process.

4. Inference Engine (ML Models): The processed vector is passed through the trained models (XGBoost, MLP, etc.). Each model computes a probability score.

5. Classifier Output: The system outputs a binary classification: '0' for Legitimate or '1' for Phishing.

3.5 Algorithm Workflow (XGBoost Example)

Since XGBoost provided the highest accuracy (86.4%), its specific architecture is noteworthy. It builds multiple weak learners (Decision Trees) sequentially. Each subsequent tree attempts to correct the errors made by the previous trees, leading to a highly refined decision boundary.

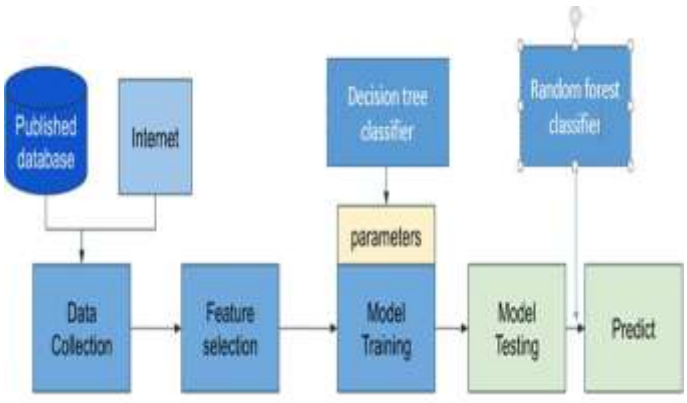

Fig -2: Architecture Diagram

3.6 Technical Environment

- Language: Python 3.x
- Libraries: Pandas (Data Handling), Scikit-learn (ML Models), XGBoost (Gradient Boosting), BeautifulSoup (HTML Analysis).
- Development: Google Colab / Jupyter Notebook for iterative testing.

4. RESULTS AND DISCUSSION

This section presents the quantitative findings of the experiments conducted on the 10,000-URL dataset. The primary metric for evaluation is **Classification Accuracy**, which measures the proportion of correctly identified URLs (both legitimate and phishing) against the total number of test samples.

4.1 Performance Analysis

The performance of six different machine learning models was evaluated. The training phase involved 8,000 samples, while the testing phase utilized 2,000 samples to ensure the models could generalize to unseen data.

Sl. No.	ML Model	Training Accuracy (%)	Testing Accuracy (%)
1	Decision Tree	81.0	82.6
2	Random Forest	81.4	83.4
3	Multilayer Perceptrons (MLP)	85.9	86.3
4	**XGBoost**	**86.6**	**86.4**
5	Autoencoder	81.8	81.8
6	SVM	79.8	81.8

Table -1: Performance Comparison of Machine Learning Models

4.2 Comparative Visualization

As illustrated in Table 1, there is a consistent trend between training and testing performance, indicating that the models did not suffer significantly from overfitting. XGBoost emerged as the top performer, followed closely by the Multilayer Perceptron (MLP). Traditional classifiers like SVM and Decision Trees showed lower performance, likely due to the non-linear complexity of the 17-feature set extracted from the URLs.

4.3 Model Interpretability and Efficiency

The superior performance of XGBoost (86.4%) can be attributed to its gradient boosting framework. Unlike standard Decision Trees, XGBoost builds trees sequentially, where each new tree minimizes the residual errors of the previous one. This makes it exceptionally robust for tabular data containing a mix of binary (e.g., Presence of "@") and continuous (e.g., URL length) features. The Multilayer Perceptron (MLP) also performed strongly (86.3%). This suggests that the relationship between URL features (like IFrame redirection and Domain Age) is highly non-linear, requiring the hidden layers of a neural network to capture the intricate patterns used by phishers to deceive users.

4.4 Significance of Feature Engineering

The success of these models validates the 17-feature extraction strategy. Specifically, features like Domain Age and Web Traffic were found to be strong discriminators. Most phishing sites are "disposable," often less than a few months old, whereas legitimate sites have established DNS records and higher Alexa rankings. By combining these with client-side features like "onMouseOver" and Right-Click disable, the system can catch sophisticated "Zero-day" attacks that static blacklists would miss.

4.5 Comparison with Prior Work

Compared to the heuristic-based models discussed in the literature review (e.g., the whitelist approach by Jain and Gupta), our automated ML approach provides a more scalable solution. While some deep learning models in literature claim higher accuracy (90%+), they often require significantly more computational power and larger datasets. Our system achieves a high balance of 86.4% accuracy with a relatively lightweight architecture, making it suitable for integration into browser extensions or real-time firewall filters.

4.6 Limitations and Challenges

Despite the high accuracy, the system has limitations:

- Dataset Sensitivity: The model's performance is tied to the Phish Tank dataset. If attackers develop entirely new URL structures, the model may require retraining.
- False Positives: A small percentage of legitimate sites with complex URL structures (e.g., deep-link nested e-commerce pages) might be flagged as suspicious due to URL length or depth.

Fig -3: Output Screenshot

Fig -4: Output Screenshot

5. CONCLUSIONS AND FUTURE SCOPE

This research successfully addressed the critical challenge of detecting phishing websites by leveraging machine learning algorithms and a comprehensive feature extraction framework. By analyzing 10,000 URLs and extracting 17 distinct features—ranging from address bar characteristics and domain age to client-side JavaScript behaviors—we have demonstrated that automated detection is a viable and highly accurate alternative to traditional blacklisting methods.

The experimental results indicate that ensemble learning and neural network architectures significantly outperform traditional linear models. XGBoost achieved the highest testing accuracy of 86.4%, closely followed by the Multilayer Perceptron (MLP) at 86.3%. These results highlight the non-linear complexity of phishing URL structures and the necessity of gradient boosting and backpropagation techniques to capture these patterns effectively.

- Feature Diversity is Crucial: Combining URL-based features with HTML and JavaScript indicators (like IFrame redirection and "onMouseOver" events) provides a much higher detection rate for sophisticated spoofing attempts.
- Scalability: The proposed machine learning pipeline is computationally efficient enough to be integrated into real-time security systems, offering protection against "Zero-day" phishing attacks that have not yet been categorized in global blacklists.
- Algorithm Efficiency: While Random Forest and Decision Trees are effective, XGBoost's sequential error-correction mechanism makes it the most robust choice for this specific classification task.

While the current system provides high accuracy, the ever-evolving nature of cyber threats presents several avenues for future enhancement:

- Integration of Deep Learning

Future iterations of this research could explore Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. These deep learning models can perform automated feature learning directly from raw URL strings or website screenshots, potentially eliminating the need for manual feature engineering and capturing even more subtle malicious patterns.

- Real-time Deployment via Browser Extensions

A natural progression of this work is the development of a cross-platform browser extension. This tool would analyze URLs in real-time as a user browses, providing an instant visual warning or blocking access to sites flagged by the XGBoost model.

- Hybrid Detection Systems

Combining machine learning with Visual Similarity Analysis could further reduce false positives. By comparing the visual layout and CSS of a suspected site with the legitimate version of the brand it claims to be, the system could provide a secondary layer of verification.

REFERENCES

1. V. Shahrivari, M. M. Darabi, and M. Izadi, "Phishing Detection Using Machine Learning Techniques," arXiv preprint arXiv:2009.11116, 2022.
2. A. Alswailem, B. Alabdullah, N. Alrumayh, and A. Alshehri, "Detecting Phishing Websites Using Machine Learning," 2nd International Conference on Computer Applications & Information Security (ICCAIS), Riyadh, Saudi Arabia, 2019, pp. 1-6. doi: 10.1109/ICCAIS42895.2019.9038232.
3. H. Shirazi, B. Bezawada, and I. Ray, "Fresh-Phish: A Framework for Auto-Detection of Phishing Websites," IEEE International Conference on Information Reuse and Integration (IRI), Salt Lake City, UT, USA, 2021, pp. 227-236.
4. Y. Cao, J. Han, and Y. Wang, "A Novel Approach for Detecting Phishing Websites Using Capsule-based Neural Networks," Journal of Physics: Conference Series, vol. 1634, 2020.
5. X. Zhang, "Research on Phishing Website Detection Based on Multi-Scale Statistical Features," Journal of Software Engineering and Applications, vol. 14, pp. 112-124, 2021.
6. T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 2016, pp. 785–794.
7. S. Garera, N. Provos, M. Chew, and A. D. Rubin, "A Framework for Detection and Analysis of Phishing URLs," Proceedings of the 2007 ACM Workshop on Recurring Malcode (WORM '07), Alexandria, Virginia, USA, 2019, pp. 18–27.
8. "PhishTank: An Open Access Clearinghouse for Data on Phishing Attacks," [Online]. Available: https://www.phishtank.com. [Accessed: Dec. 20, 2024].