An Efficient Phishing Website Detection Plugin Service Based on Website Trustworthiness Evaluation

Phoon Gar Chi, Low Choon Keat, Ng Yen Phing, Tan Bee Sian, Chong Kim Soon

Tunku Abdul Rahman University of Management and Technology (TAR UMT), Jalan Genting Kelang, Setapak, 53300 Kuala Lumpur, Federal Territory of Kuala Lumpur.

DOI: https://dx.doi.org/10.47772/IJRISS.2025.908000290

Received: 31 July 2025; Accepted: 07 August 2025; Published: 08 September 2025

ABSTRACT

Phishing attacks remain a significant threat to online users, and existing anti-phishing mechanisms often fall short in providing real-time, adaptive protection. This study proposes a browser extension that leverages machine learning to assess website trustworthiness and detect phishing threats in real time. A logistic regression classifier was trained using axx dataset comprising both legitimate and phishing URLs sourced from PhishTank and Google's site rankings. The model achieved an accuracy of 0.96 on training data and 0.91 on test data. Our browser extension provides a robust, real-time defense against phishing, with minimal latency and a user-friendly interface.

Keywords: Phishing, Browser extension, Trustworthiness evaluation, Machine learning, Website security

INTRODUCTION

Phishing is a cyberattack strategy that deceives users into revealing confidential information—such as credit card numbers, usernames, or passwords—via fake emails, malicious websites, or instant messages [15]. With phishing tactics becoming more frequent and sophisticated, it is critical to develop advanced firewalls and security tools to protect users during their online activities.

The main objective of this project is to design, develop, and evaluate an anti-phishing browser extension that serves as a reliable defense mechanism against fraudulent online activities [15]. This extension enhances user protection by leveraging advanced machine learning techniques to assess the trustworthiness of websites in real time. It alerts users to potential phishing attempts by continuously monitoring and analyzing URLs and web content. The extension integrates seamlessly into major browsers, providing a user-friendly, accessible, and convenient tool that surpasses many existing anti-phishing solutions.

To achieve comprehensive protection, the proposed system adopts a multi-layered detection strategy that includes URL analysis, blacklisting, whitelisting, and website content inspection [8]. URL analysis examines structural patterns in web addresses to detect anomalies often present in phishing links—such as subtle misspellings or added characters. Blacklisting involves maintaining an up-to-date database of known phishing websites to block user access [13], while whitelisting ensures that only verified, trusted websites are accessible without warning prompts. These lists must be updated frequently to address the continuously evolving landscape of phishing threats. Content analysis further enhances detection by scanning webpages for suspicious forms, deceptive images, and keywords commonly associated with phishing attempts.

A key innovation in this system is the integration of machine learning—an artificial intelligence approach that enables systems to improve over time without being explicitly programmed. By analyzing features such as site structure, URL patterns, SSL certificates, and language content, the machine learning model can predict the likelihood of a phishing attempt with high accuracy. Unlike traditional systems that rely on static rules, the model continually "learns" from new data, improving its performance over time. This integration of machine learning with conventional methods (URL detection, blacklisting, and content analysis) enhances overall detection robustness and reduces false positives. However, this approach introduces new challenges, such as the need for large training datasets, computational resources, and frequent retraining to ensure long-term model accuracy [9].

Despite these challenges, a machine learning-powered browser extension provides a highly efficient and scalable solution for detecting phishing attacks. Its real-time capabilities, ease of deployment, and adaptability to emerging threats make it a valuable tool in modern cybersecurity defense.

LITERATURE REVIEW

In recent years, users have become the weakest link in the cybersecurity chain, as many attacks exploit human behavior rather than system vulnerabilities [10]. Phishing, as a form of social engineering, manipulates users into revealing sensitive information via fake emails or websites. This problem is exacerbated by the rise of social networking and the increasing sophistication of phishing techniques.

The journal [18] reported that 46 million Malaysian users had their data compromised due to phishing. Such attacks often appear in spam emails or deceptive advertisements on social platforms like Facebook and Twitter. Users who unknowingly engage with these malicious links may lose access to their personal accounts or suffer identity theft.

A study by Sam Cook [7] shows that phishing attacks have steadily increased from 2019 to 2022, reaching an all-time high in 2022 (Figure 1).

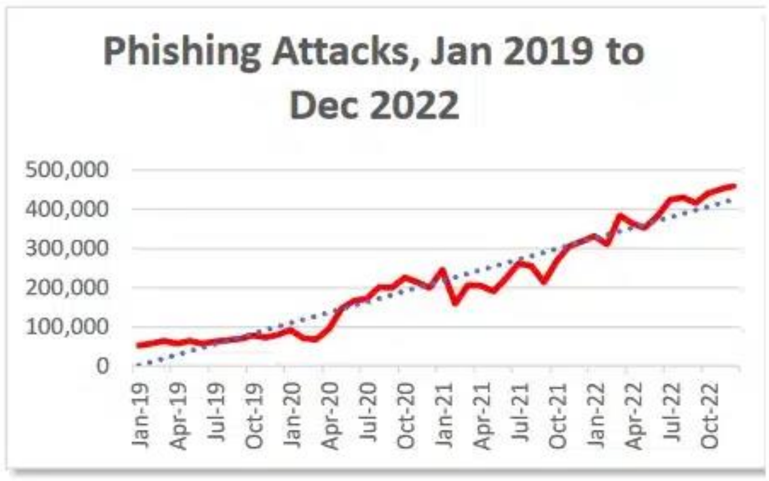

Figure 1: Phishing attack between 2019-2022

Based Similarly, data from the Anti-Phishing Working Group (APWG) [20] reveals that phishing incidents have risen by over 150% since 2019 (Table 1).

Table 1. Phishing attack growth by year

Year	Number of Attacks Observed
2019	779,200
2020	1,845,814
2021	2,847,773
2022	4,744,699

Other than that, credential phishing remains a widespread threat, even though it is no longer the most dominant technique. According to findings from the APWG study [7], attackers are still actively targeting users' credentials, albeit with a shift in focus. While the financial sector has historically been the primary target for phishing campaigns, recent trends indicate that cybercriminals are diversifying their efforts (Figure 2). Increasingly, phishing attacks are directed toward less conventional and more varied industries, highlighting the adaptive strategies employed by scammers to exploit emerging vulnerabilities.

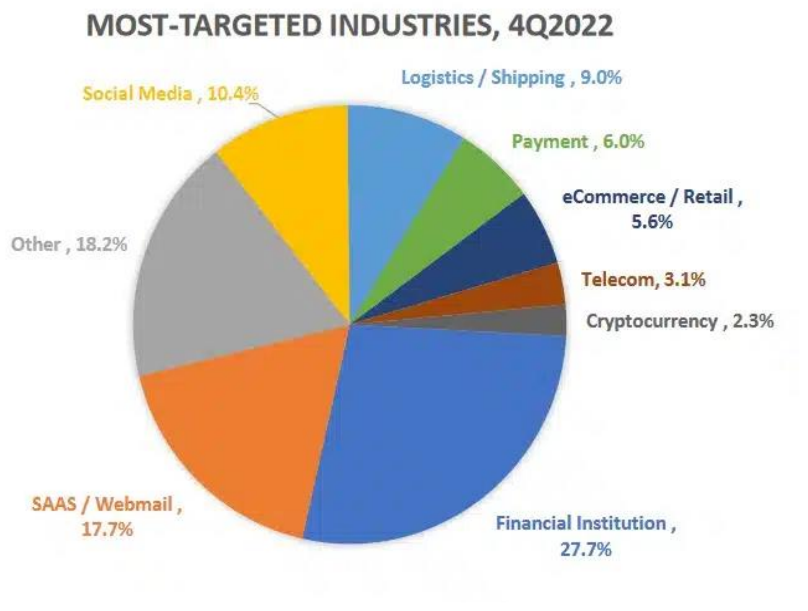

Figure 2: Target in industries

Blacklist and Whitelist Techniques for Phishing Detection

Blacklist and whitelist approaches are among the earliest and most widely adopted techniques for phishing detection. Despite their simplicity, they remain foundational components in many anti-phishing systems due to their ease of implementation and relatively low computational overhead.

A blacklist is a repository of known malicious domains, URLs, IP addresses, or email addresses that have previously been identified as associated with phishing activity. When a user attempts to access a website, the system checks the URL against this list. If the URL is present, access is blocked or a warning is issued to the user [13][15].

Blacklists are widely implemented in commercial and open-source tools, including PhishTank, Google Safe Browsing API, Spamhaus, and built-in browser security features like those in Chrome and Firefox. These services rely on frequent updates and community reporting to remain effective.

However, blacklists have a significant limitation: they are reactive. Since phishing websites are often short-lived (sometimes active for only a few hours), blacklists may not catch newly deployed phishing sites. This zero-day vulnerability poses a serious threat to users relying solely on blacklists.

To enhance their effectiveness, some systems apply heuristic rules or incorporate machine learning to identify suspicious patterns in new URLs before they are officially blacklisted. Others integrate with crowdsourced platforms that allow rapid user feedback and reporting of phishing attempts.

In contrast, a whitelist is a curated list of trusted and verified domains that users are allowed to access without restrictions. All other URLs are treated with suspicion unless explicitly approved. Whitelisting is considered a proactive strategy, especially useful in enterprise or closed environments where the range of legitimate websites is relatively fixed [11].

When a user attempts to access a URL, the system checks whether the domain exists on the whitelist. If it does, access is granted; if not, the user may be warned or denied access. This method significantly reduces false negatives (i.e., phishing sites slipping through undetected), especially for users who typically visit a predictable set of domains (e.g., banking portals, academic sites, corporate tools).

However, whitelisting also presents practical challenges:-

(1)Maintaining and updating a comprehensive whitelist is labour-intensive.

(2)It may block access to safe but newly visited websites, leading to user frustration or increased false positives.

(3)It is less suitable for general-purpose browsing environments where users explore a wide range of sites.

To overcome the limitations of each approach, many modern anti-phishing systems adopt hybrid models that combine both blacklisting and whitelisting. These systems:- (1)Use blacklists to immediately block known threats,

(2)Apply whitelists to automatically allow trusted sources, and

(3)Perform additional content analysis or machine learning inference on all other URLs.

This multi-tiered system balances security and usability by minimizing both false positives and false negatives, while also enabling faster adaptation to evolving phishing tactics.

URL and Content Analysis for Phishing Prevention

Although phishing cannot be completely eradicated, it remains a persistent and evolving cybersecurity threat that can be mitigated through consistently updated and adaptive detection techniques [1]. Within this context, URL and content analysis have emerged as vital components in the proactive defense against online fraud.

Conventional phishing detection often starts with URL inspection. This involves examining Uniform Resource Locators (URLs) using both lexical and structural analysis to identify suspicious characteristics. Traditional URL inspection is frequently tied to blacklist and whitelist mechanisms. Blacklists are maintained databases of known malicious domains, keywords, or IP addresses associated with phishing activities. They are effective against previously reported threats, but struggle to detect zero-day phishing sites—newly generated URLs that have not yet been reported or indexed [4]. As such, blacklists require constant updates to remain effective, making them more reactive than proactive.

In contrast, content analysis goes beyond the URL itself by inspecting the actual elements of the website or email suspected of phishing [12]. This includes HTML structure, images, links, embedded scripts, and layout design, along with specific phishing cues such as login forms requesting credentials or embedded redirect links [19]. Additionally, social engineering patterns, such as urgent language or imitated brand assets, are also considered. This multifaceted analysis enables systems to detect phishing even when URLs appear superficially legitimate. However, content analysis is inherently resource-intensive, requiring more computation and processing time due to the complexity and variability of web content.

To enhance both speed and accuracy, advanced feature extraction techniques have been developed. These techniques isolate and quantify key characteristics from URLs and webpage content to be fed into machine learning models. For instance, URL feature extraction may evaluate: (1)Use of HTTPS or SSL/TLS encryption,

(2)Number of redirects involved before landing on the target page,

(3)Overall URL structure, including token count and

(4)Domain depth and domain reputation, based on age, registrar, or popularity rankings.

By analysing such attributes, detection systems can uncover subtle patterns indicative of phishing behaviour, even in cases that bypass blacklists. These extracted features serve as input for predictive models or heuristic scoring systems, which assign a threat probability to each URL or page. A well-architected feature extraction pipeline significantly improves phishing detection effectiveness, especially when integrated into real-time browser extensions or email filters.

Combining URL and content analysis enhanced by advanced feature extraction enables detection systems to move beyond reactive blacklisting toward adaptive, intelligent identification of phishing threats. When applied in conjunction with machine learning, this layered approach offers a scalable and robust mechanism for real-time protection.

Leveraging Machine Learning for Enhanced Phishing Detection

Machine Learning (ML), a subfield of Artificial Intelligence (AI), enables systems to automatically learn patterns from data and make decisions or predictions without being explicitly programmed. Its ability to adapt and improve over time makes ML an ideal candidate for tackling dynamic cybersecurity threats like phishing.

Unlike static blacklists, ML models can detect phishing attempts based on behavioural patterns, URL structure, and embedded website features. Recent research has shown that ML-based approaches significantly outperform heuristic and blacklist-based methods in terms of detection accuracy and adaptability [14].

Several ML algorithms are commonly used for phishing detection:-

(1)Logistic Regression (LR): A linear classifier suitable for binary classification tasks. It is efficient and interpretable, making it ideal for modeling phishing vs. legitimate URLs.

(2)Decision Trees (DT): These build a flowchart-like structure based on feature splits. Although easy to interpret, they are prone to overfitting.

(3)Random Forests (RF): An ensemble of decision trees that reduces variance and improves generalization.

(4)Support Vector Machines (SVM): A powerful method that finds the optimal hyperplane to separate classes in high-dimensional space.

For each of these models, input features include Lexical features of the URL (length, use of symbols, domain info); Presence of HTTPS and SSL certificates; Page content and structure; Domain reputation and redirection behavior [3].

A linear model called logistic regression forecasts the likelihood of a binary result. It maps predictions to probabilities between 0 and 1 using the logistic function. Decision Trees generate a structure that is similar to a tree by recursively dividing the dataset according to features. Both numerical and category data can be handled by them. A collection of several decision trees, each trained on a distinct subset of the data, is called Random Forest. The total of each tree's individual projections makes up the final forecast. SVM seeks to maximize the margin between distinct classes by locating the hyperplane in high-dimensional space that best divides data points.

Dynamic phishing attempts are addressed with machine learning-based defenses, which have higher accuracy and fewer false positives than other approaches [21]. Model evaluation is commonly done using:-

(1)True Positives (TP): Phishing URLs correctly classified as phishing.

(2)True Negatives (TN): Legitimate URLs correctly classified.

(3)False Positives (FP): Legitimate URLs incorrectly flagged as phishing.

(4)False Negatives (FN): Phishing URLs not detected.

Performance metrics such as Accuracy, Precision, Recall, and F1-score are derived from these values to assess the model's effectiveness [9][21].

Enhancing Phishing Detection Using Machine Learning: A Random Forest Approach

An efficient phishing website detection plugin service was developed using machine learning techniques to address the growing prevalence of phishing threats in critical online transactions. The study synthesized findings from 27 published articles and utilized a dataset of 11,000 labeled URLs with 30 relevant features, sourced from PhishTank.

A unique architectural framework for phishing detection was designed using the Random Forest (RF) classifier, selected for its ability to handle high-dimensional feature sets and its robustness to overfitting. The model was implemented in Python and trained using 90% of the dataset (9,900 samples), with the remaining 10% (1,100 samples) reserved for testing.

The proposed RF model achieved an accuracy of 0.96, an error rate of 0.04, precision of 0.97, recall value of 0.99, and an F1-score of 0.98. These results significantly outperformed comparable models from existing literature, demonstrating the model's ability to maintain both high precision (low false positives) and high recall (low false negatives). The combination of strong generalization ability and high classification accuracy makes RF a compelling choice for phishing detection tasks [12].

However, due to the computational complexity of Random Forest, especially when deployed in real-time environments like browser extensions, simpler models such as Logistic Regression may be preferable for scenarios requiring low latency and lightweight processing [12]. RF models are better suited for offline training or high-performance environments where computational resources are abundant.

RELATED WORK

Several studies have explored browser-based extensions and machine learning techniques to mitigate phishing threats effectively. In journal [18], researchers developed a Google Chrome extension named 3H1M, which uses blacklist algorithms to detect phishing websites. The system capitalizes on the observation that phishing sites often mimic legitimate URLs with minor alterations—for example, using www.maybank2u2.com instead of the real www.maybank2u.com. The extension successfully prevents users from accessing such deceptive domains by comparing them against known blacklists.

Another study [17] proposed a phishing detection plugin built using a Random Forest classifier. The model was trained on 11,000 URLs (downloaded from PhishTank), using 30 unique features. The system achieved high classification performance 96% accuracy and 0.98 F1-score, making it a strong candidate for real-world deployment.

In Addition, in journal [6], a phishing email detection extension was built for Google Chrome, leveraging text mining and machine learning. The extension analyzed suspicious keywords in emails and demonstrated higher accuracy and sensitivity compared to Gmail's built-in filters. Although Gmail had a lower false positive rate, the extension offered superior detection rates for phishing emails and a user-friendly interface.

The authors of [9] also developed a plugin service using Random Forest, trained on a large dataset sourced from 27 publications and 11,000 labeled entries from PhishTank. The extension could classify URLs as either "legitimate" or "phishing," and even graded them by threat level: low, medium, or high. Real-time classification ensured an adaptive and responsive defense mechanism against phishing threats.

In journal [5], an extension named MailTrout was introduced for detecting phishing emails. This browser-based tool employed a machine learning model and achieved high accuracy in real-world testing. Test participants found the extension easy to use and effective in highlighting phishing tactics, suggesting its practical usability in real-time environments.

These works collectively illustrate that browser-integrated machine learning tools offer a promising defense against phishing—especially when designed with real-time detection, user experience, and system efficiency in mind.

System design and evaluation

Overview on dataset

For the training dataset (Figure 3), there are 40,000 URLs used for model training, comprising 20,020 phishing URLs (50.05%) and 19,980 safe URLs (49.95%). The 'url' column represents the Uniform Resource Locator, while the 'label' column indicates whether the URL is safe (1) or malicious (-1).

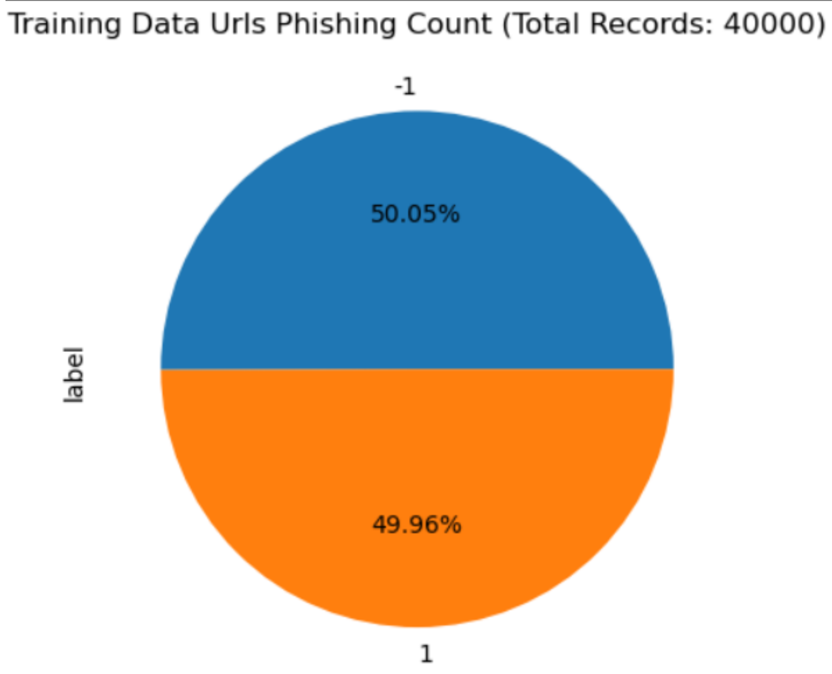

Figure 3: Training dataset

For the test dataset (Figure 4), there are 13,005 URLs used for model evaluation, comprising 7,000 phishing URLs (53.83%) and 6,005 safe URLs (46.17%). The 'url' column represents the Uniform Resource Locator, while the 'label' column indicates whether a URL is safe (1) or malicious (-1).

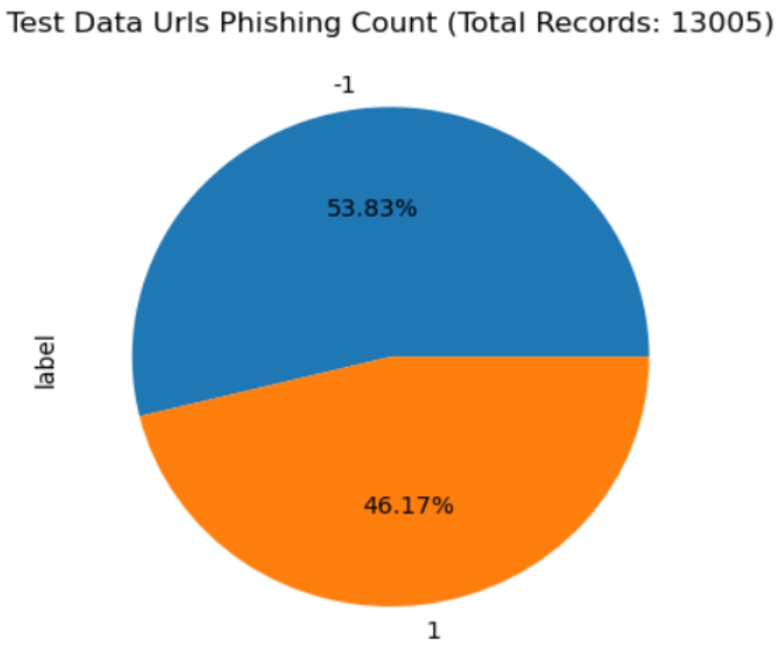

Figure 4: Test dataset

The dataset includes the most recent data retrieved from the PhishTank platform, which contains reported and verified malicious URLs, the latest confirmed malicious URLs, and high Google PageRank legitimate URLs.

This combination ensures that the dataset remains up to date and includes both malicious and trusted sources, creating a balanced and reliable foundation for training.

Logistic Regression Model

Logistic Regression was chosen as the model to predict phishing URLs because its inherent characteristics align well with the nature of the task. It is particularly suitable for phishing URL prediction, as this is essentially a binary classification problem that distinguishes between legitimate (1) and malicious (-1) URLs [2]. In the field of cybersecurity, the interpretability of the model is a valuable advantage, as it facilitates a clear understanding of how features contribute to the prediction—an essential aspect for providing explanations and validating results. Additionally, Logistic Regression performs well when the features have a clear correlation with the probability that a URL is phishing, as it is effective when the decision boundary is relatively linear [16]. Its scalability and low computational complexity make it efficient for processing large volumes of URLs, which is crucial in real-time applications. Furthermore, its regularization options help prevent overfitting, making it adaptable to datasets of varying sizes. Given its proven effectiveness in cybersecurity applications, including phishing detection, Logistic Regression remains a practical and efficient choice for developing baseline models for phishing URL prediction (Figure 5).

Pseudocode for logistic regression model

```
1: # TF-IDF Vectorization

2: vectorizer = TfidfVectorizer()

3: training_data_tfidf = vectorizer.fit_transform(training_url_corpus)

4: testing_data_tfidf = vectorizer.transform(testing_url_corpus)

5: # Labels

6: training_labels = labeled_training_data['label']

7: testing_labels = labeled_testing_data['label']

8: # Instantiate the Logistic Regression model

9: logistic_regression_model = LogisticRegression()

10: # Train the model on the training data

11: logistic_regression_model.train(training_data_tfidf, training_labels)

12: # Make predictions on the training set

13: predicted_training_labels = logistic_regression_model.predict(training_data_tfidf)
```

Figure 5:Pseudo code for logistic regression model

Firstly, a TF-IDF vectorizer commonly used in natural language processing—is initialized (Figure 5). The fit_transform method is applied to the training data (training_url_corpus), which tokenizes and converts the text data into TF-IDF vectors. Subsequently, the same vectorizer is used to transform the testing data (testing_url_corpus) into TF-IDF representations. The labels associated with the training and testing datasets are then retrieved, indicating whether each URL is classified as phishing or legitimate.

Next, a Logistic Regression model is instantiated and trained using the TF-IDF-transformed training data (training_data_tfidf) and corresponding labels (training_labels). During this training phase, the model learns to map TF-IDF features to the appropriate labels. The trained Logistic Regression model is then used to make predictions on the training set, producing predicted_training_labels that represent the model's classification for each URL. Similarly, predictions are made on the test set (testing_data_tfidf), resulting in predicted_testing_labels for each URL in the test set.

The model's low latency and lightweight architecture make it highly suitable for integration into browser extensions. Both TF-IDF transformation and model inference can be executed within milliseconds, enabling real-time phishing detection without any noticeable impact on browsing performance.

RESULT AND DISCUSSION

This section focuses on a thorough comparative analysis of the efficiency of the Logistic Regression (LG) classifier on the dataset. A performance comparison of three different classifiers: Logistic Regression (LG), Decision Tree (DT), and Random Forest (RF), using the same dataset is presented in Table 2. All performance data for this analysis was obtained from the Classification Report, a key component of scikit-learn's metrics module. This report provides a comprehensive view of model performance through metrics such as precision, recall, and F1-score.

The purpose of this analysis is to highlight the subtle differences and advantages of the Logistic Regression classifier over the Random Forest and Decision Tree classifiers when applied to the same dataset.

The performance of each model was evaluated using the following metrics:-

(1) Accuracy: Overall correctness of the model.

(2) Precision: Correctness of positive (phishing) predictions.

(3) Recall: Ability to detect all phishing URLs.

(4) F1-score: Harmonic mean of precision and recall.

Table 2: Result of classification report

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression (LG)	0.91	0.935	0.865	0.899
Decision Tree (DT)	0.748	0.769	0.65	0.704
Random Forest (RF)	0.906	0.769	0.805	0.888

Table 2 shows that the Logistic Regression classifier achieves an impressive test accuracy of 0.91, demonstrating its ability to make correct predictions on a significant proportion of the test data. Notably, its precision of 0.935 highlights the model's proficiency in correctly identifying positive instances while minimizing false positives.

In comparison, the Decision Tree classifier, although showing respectable performance with a test accuracy of 0.748, falls short in precision, recall, and F1-score when compared to Logistic Regression. Its lower precision of 0.769 indicates a higher rate of false positives, reducing its effectiveness in accurately classifying positive instances.

The Random Forest classifier achieves a commendable test accuracy of 0.906; however, its precision on the test data mirrors the pattern seen with the Decision Tree at 0.769. This suggests challenges in precisely identifying positive instances, contributing to an elevated false-positive rate.

Overall, the Logistic Regression model delivers balanced performance across all evaluation metrics, demonstrating a strong capacity to accurately classify positive cases while minimizing false positives. Its combination of high precision, recall, and F1-score places it ahead of both the Decision Tree and Random Forest classifiers, making it the preferred choice for this dataset.

According to Figure 6, the training and test accuracy for the Logistic Regression model further support its effectiveness. The training accuracy, which measures the model's performance on the data it was trained on, reaches an excellent 0.96 (96%). The test accuracy, reflecting the model's ability to generalize to unseen data, achieves 0.91 (91%). These results indicate that the model fits the training data well without significant overfitting, maintaining strong generalization performance for predicting URLs.

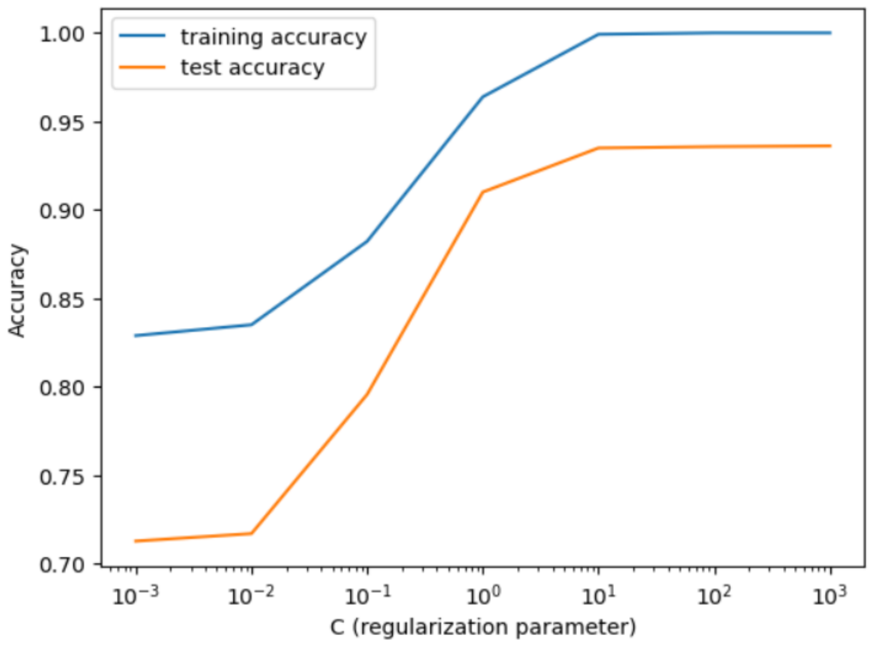

Figure 6: Performance result of training and test data

CONCLUSION

This research presents a browser-based anti-phishing extension that leverages machine learning—specifically, Logistic Regression—to evaluate website trustworthiness and detect phishing threats in real time. The model was trained on a comprehensive dataset comprising legitimate and phishing URLs sourced from PhishTank and Google site rankings, ensuring high relevance and up-to-date accuracy.

The Logistic Regression model achieved a strong test accuracy of 91%, outperforming both Decision Tree and Random Forest classifiers in terms of precision and F1-score. Its high interpretability, computational efficiency, and real-time prediction capability make it especially suitable for deployment in browser environments.

Beyond performance, the proposed solution emphasizes user accessibility and minimal system overhead. Unlike traditional antivirus software, this extension integrates directly into the user’s browsing experience, providing lightweight protection without requiring additional installations or technical expertise.

As phishing tactics continue to evolve, the ability to adapt and retrain the model periodically will be crucial. Future enhancements could include: -

(1)Incorporating hybrid models (e.g., combining Logistic Regression with deep learning),

(2)Implementing real-time URL scoring dashboards for user feedback,

(3)Addressing zero-day phishing threats using anomaly detection.

Ultimately, this work contributes a practical, scalable, and privacy-conscious tool to the fight against phishing, enhancing online safety for users with a transparent and technically sound machine learning solution.

ETHICAL CONSIDERATIONS AND DATA PRIVACY

The deployment of machine learning-based browser extensions for phishing detection must carefully consider ethical and privacy concerns. While the proposed system enhances online safety, it processes users' browsing URLs in real time. To ensure user trust and compliance with ethical standards: -

(1)Data Anonymization: No personal or identifiable user data is stored. All URL inputs are processed locally on the user's device, and no browsing history is transmitted to external servers.

(2)User Consent and Transparency: The extension includes a clear privacy policy explaining how data is handled. Users are informed that their data remains private and are given control over enabling or disabling detection features.

(3)Bias and Fairness: The model is trained using a balanced dataset representing a diverse range of legitimate and phishing websites, reducing bias in prediction. However, continued evaluation is recommended to ensure fairness across evolving domains.

(4)Security Compliance: The extension design aligns with browser extension guidelines and prioritizes secure coding practices to prevent misuse or data leakage.

Addressing these ethical dimensions reinforces the credibility and reliability of the proposed anti-phishing tool and ensures its alignment with responsible AI and cybersecurity practices.

REFERENCES

1. Aljofey, A., Jiang, Q., Rasool, A., Chen, H., Liu, W., Qu, Q., & Wang, Y. (2022). An Effective Detection Approach For Phishing Websites Using Url And Html Features. Sci Rep 12.
2. Anandkumar, V. (2019). Malicious-Url Detection Using Logistic Regression Technique. International Journal Of Engineering Business Management.
3. Awasthi, A., Goel, N. (2022). Phishing Website Prediction Using Base And Ensemble Classifier Techniques With Cross-Validation. Cybersecurity 5.
4. Azeez, N. A., Misra, S., Margaret, I. A., Fernandez-Sanz, L., & Abdulhamid, S. M. (2021). Adopting Automated Whitelist Approach For Detecting Phishing Attacks. Computers & Security, Volume 108.
5. Boyle, P., & Shepherd, L. A. (2021). Mailtrout: A Machine Learning Browser Extension For Detecting Phishing Emails. Bcs Learning And Development Ltd.
6. Chen, H., & Hossain, M. (2021). Developing A Google Chrome Extension For Detecting Phishing Emails. Epic Series In Computing.
7. Cook, S. (2023). Phishing Statistics And Facts For 2019–2023. Data Journalist, Privacy Advocate And Cord-Cutting Expert.
8. Coffie, I., Mabarani, S., Svotwa, L., Njogu, G. N., &Rubaiza, J. B. (2020). Enhancing Phishing Detection Tools: A Machine Learning Approach.
9. John-Otumu, A. M., Rahman, M. M., & Oko. (2021). An Efficient Phishing Website Detection Plugin Service For Existing Web Browsers Using Random Forest Classifier. Department Of Computer Science, Morgan State University, Baltimore, Usa.
10. Khonji, M., Iraqi, Y., Member, S., Ieee, & Jones, A. (2013). Phishing Detection: A Literature Survey. Ieee Communications Surveys & Tutorials, Vol. 15, No. 4.
11. Li, L., Berki, E., Helenius, M., & Ovaska, S. (2014). Towards A Contingency Approach With Whitelist-And Blacklist-Based Anti-Phishing Applications: What Do Usability Tests Indicate? Behaviour & Information Technology.
12. Low Choon Keat, Ang Tan Fong, Chun Yong Chong, And Tew Yiqi. 2022. “(Offloading) Qoe-Aware Application Mapping And Energy-Aware Module Placement In Fog Computing + Offloading.” International Journal Of Web Services Research (Ijwsr), Vol. 19 (1), Pages 1-28, January.
13. M, P., Mounika.B, V. S., G, M., & Sk, A. B. (2023). Chrome Extension For Detecting Phishing Websites. International Journal For Innovative Engineering And Management Research.
14. Mahajan, R., &Siddavatam, I. (2018). Phishing Website Detection Using Machine Learning Algorithms. International Journal Of Computer Applications, Volume 181.
15. Maurya, S., Saini, H. S., & Jain, A. (2019). Browser Extension Based Hybrid Anti-Phishing Framework Using Feature Selection. (Ijacsa) International Journal Of Advanced Computer Science And Applications.
16. Mbachan, F. (2022). Phishing Url Prediction Using Logistic Regression.
17. Oko, C. U., &Otuonye, A. I. (2022). Development Of Phishing Site Detection Plugin To Safeguard Online Transcation Services. Imo State Chapter Nigeria Computer Society, Conference Proceeding It For Economy Development And National Security (Iteden).

18. Ramli, M. H., Faugi, A. I., Faizal, N. M., Khadri, N. M., & Zain, J. M. (2020). Anti-Phishing With Google Extension 3h1m Using Blacklist Algorithm. Malaysian Journal Of Computing.
19. Sánchez-Paniagua, M., Fidalgo, E., Alegre, E., &Alaiz-Rodríguez, R. (2022). Phishing Websites Detection Using A Novel Multipurpose Dataset And Web Technologies Features. Expert Systems With Applications, Volume 207.
20. Smith, G. (2023). Top Phishing Statistics For 2023: Latest Figures And Trends. Algorithm. Malaysian Journal Of Computing.
21. Tang, L., & Mahmoud, Q. H. (2021). A Survey Of Machine Learning-Based Solutions For Phishing Website Detection. Mdpi.