International Journal of Data Informatics and Intelligent Computing (IJDIIC)
Vol. 4, No. 2, June 2025, pp. 41~52
DOI: 10.59461/ijdiic.v4i2.187

Malicious URL detection using machine learning techniques

Mohamed Cherradi[1], Hajar El Mahajer[2]
[1]Abdelmalek Essaâdi University (UAE), ENSAH, Tetouan, Morocco
[2]Abdelmalek Essaâdi University (UAE), FSTT, Tetouan, Morocco

Article Info

Article history:

Received March 16, 2025
Revised May 12, 2025
Accepted May 18, 2025

Keywords:

Cybersecurity
Malicious URLs
Machine Learning
Classification

ABSTRACT

With numerous new websites being created every day, it's getting increasingly challenging to tell which ones are safe and which could be dangerous. These websites frequently gather sensitive user data that may be hacked in the absence of proper cybersecurity safeguards, such as the effective identification and categorization of dangerous URLs. In order to improve cybersecurity, this study attempts to create models based on machine learning algorithms for the effective detection and categorization of harmful URLs. In this regard, our proposal uses decision trees, logistic regression, support vector machines, and Naive Bayes to reliably categorize dangerous URLs. To improve classification efficiency, we have integrated hyper-parameter tuning using the Grid Search technique, optimizing model performance for more accurate and reliable results. The results demonstrate the effectiveness of Naive Bayes in achieving high accuracy (91.9%) and reliable performance in detecting malicious URLs. Implementation as a web service of the study provides evidence of the practicality and natural fit into more generalized security frameworks. Ultimately, our approach significantly enhances the detection of unsafe URLs, offering a robust solution to address the growing challenges in cybersecurity.

Corresponding Author: Mohamed Cherradi (e-mail: m.cherradi@uae.ac.ma)

1. INTRODUCTION

Many new internet pages are made every day that use login features to gather user information. It is difficult to identify which one is trustworthy and safe due to the enormous number of websites [1]. In this situation, cybersecurity plays a crucial role. A collection of methods or instruments designed to defend consumers and businesses against cyberattacks is known as cybersecurity [2]. Malicious URLs, or hyperlinks, are a key tool used by hackers to trick Internet users into divulging private and sensitive information in this huge digital environment. Users who interact with these links put themselves at risk of negative outcomes, such as the compromise of private data or being the focus of cyberattacks.

Cybercriminals use various techniques to take advantage of human and system vulnerabilities. Phishing is one of the most popular techniques, in which attackers attempt to fool targets into disclosing private information that could have dire repercussions [3]. Another method that attackers employ to alter web pages' content is defacement, which involves altering the source code. This type of cyberattack is commonly used to compromise a company's website [4]. The ways that cybercriminals utilize misleading website addresses to spread and run malware are known as malware techniques in harmful URLs. These methods seek to send malicious payloads, trick users, and take advantage of software flaws. According to a 2013 RSA research [5], phishing attacks caused losses on approximately 450,000 websites. Blacklists made up of known malicious URLs are used to combat such threats. However, because new malicious URLs linked to spam and phishing activities are constantly appearing, their effectiveness is still restricted.

In order to detect both new and existing dangerous URLs, machine learning is crucial [6]. Computers may be taught to interpret data through a process called machine learning, which gives them the ability to forecast or decide on their own. Classification, a subfield of supervised machine learning, is the most widely used method for detecting dangerous URLs [7]. Various specific machine learning models fall under this category. However, the efficiency of models can be influenced by several factors. In this context,

we have conducted a benchmark study between four machine learning algorithms to evaluate their performance in detecting malicious URLs. To address these variations, this study presents a comprehensive benchmark of four well-known machine learning algorithms—Logistic Regression, Support Vector Machine, Naive Bayes, and Decision Tree. Our contributions include the application of Grid Search-based hyperparameter tuning to enhance model performance, the implementation of a robust web-based detection system using FastAPI, and an in-depth comparative analysis using a real-world dataset from Kaggle. The findings aim to guide the selection and deployment of effective detection models in real-time cybersecurity environments.

The rest of this study is organized as follows. Section 2 provides a summary of the pertinent literature on URL identification. The study's methodology, including data gathering, machine learning models, and instance selection techniques, is summarized in Section 3. The study's findings are reported in Section 4, along with an analysis and discussion of the models' and instance selection techniques' performances. Finally, the findings and suggested paths for further research are presented in Section 5.

2. LITERATURE REVIEW

Malicious URL detection and classification using different machine learning techniques has been the subject of numerous literature investigations. These studies helped to clarify the best methods and strategies for recognizing and categorizing dangerous websites. Building on these findings, this section examines relevant research on identifying malicious attacks. In this context, numerous researchers have looked into various features and methods for detecting harmful attacks on websites. For instance, Aldwairi et al. [8] used a simple self-learning method as the basis for their study. The open-source datasets that were used were the PhishTank dataset, which contains malicious URLs, and Alexa, which contains benign websites. There are 31 features in all, including lexical, network-based, and content-based features. The accuracy of the system was 87%.

Another study that used machine learning approaches to detect fraudulent URLs was carried out by Xuan et al. [9]. They employed lexical, network-based, and content-based criteria to categorize URLs. Similarly, He et al. [10] suggested a random forest-based feature selection technique. Moreover, Yu [11] introduced a hybrid model for fraudulent website identification that included the benefits of support vector machines (SVM) and deep belief networks (DBN). A further investigation developed by Zamir et al. [12] suggested a stacking model-based methodology for identifying phishing websites. Kaggle is the dataset used. Yet, Rao et al. [13] provided rule-based solutions for effective URL malicious detection. Thus, Adewole et al. [14] presented a hybrid rule induction system that can distinguish between malware and authentic websites. The hybrid algorithm generates rule sets by combining the advantages of the projective adaptive resonance theory (PART) algorithm and the rule induction algorithm (JRip).

Recent studies in the detection of malicious URLs continue to expand on machine learning techniques, exploring both traditional models and more recent innovations. One such study [15] delves into the use of quantum machine learning for detecting fraudulent URLs, specifically in the context of Internet of Things (IoT) devices, which are highly susceptible to phishing attacks. This research emphasizes the integration of quantum computing into cybersecurity while also paving the way for future investigations into quantum algorithms for malicious URL detection. Similarly, a study by [16] applied a variety of machine learning algorithms, such as Decision Tree, Naive Bayes, K-Nearest Neighbors, and Support Vector Machine, to both malicious URL detection and network intrusion detection. This work showed that Decision Tree, KNN, and SVM classifiers further validate the efficacy of machine learning models in real-world cybersecurity applications. Another relevant contribution [17] is a comprehensive taxonomy of malicious URL detection techniques, which contrasts rules-based approaches, such as blacklisting and heuristics, with machine learning methods. While rules-based techniques offer simplicity, they struggle to keep up with rapidly evolving malicious URL patterns, whereas machine learning techniques, though more complex, are better suited to adapt to new threats. These recent studies underscore the increasing sophistication of machine-learning approaches in detecting malicious URLs and the continuous need for innovative solutions to address emerging threats in cybersecurity.

Overall, a number of research studies have integrated lexical-content-based and lexical-network-based features, two types of URL-based features. Numerous research studies, including [18], found that the most popular combination of URL-based features was the lexical-network-based features. However, by using the machine learning classifier and combining lexical and content-based features, it was able to attain a higher level of accuracy. Therefore, the examined research concludes by emphasizing the increasing significance of machine learning methods for identifying dangerous URLs. The use of sophisticated machine learning models has greatly improved the accuracy as well as efficiency of detection systems, even though conventional techniques still play a vital role. The range of methods examined in the literature, from network-based and content-based characteristics to lexical analysis, reveals the broad spectrum of methods

used to address the ever-evolving environment of cyber threats. In light of this, the related work emphasizes whether integrating machine learning techniques has both promising potential and ongoing significance.

3. METHODOLOGY

This section outlines the process for creating machine learning models that can detect dangerous URLs. It outlines the research framework and explains the methods and approaches that were used. The methodology's workflow in this investigation is depicted in Figure 1.

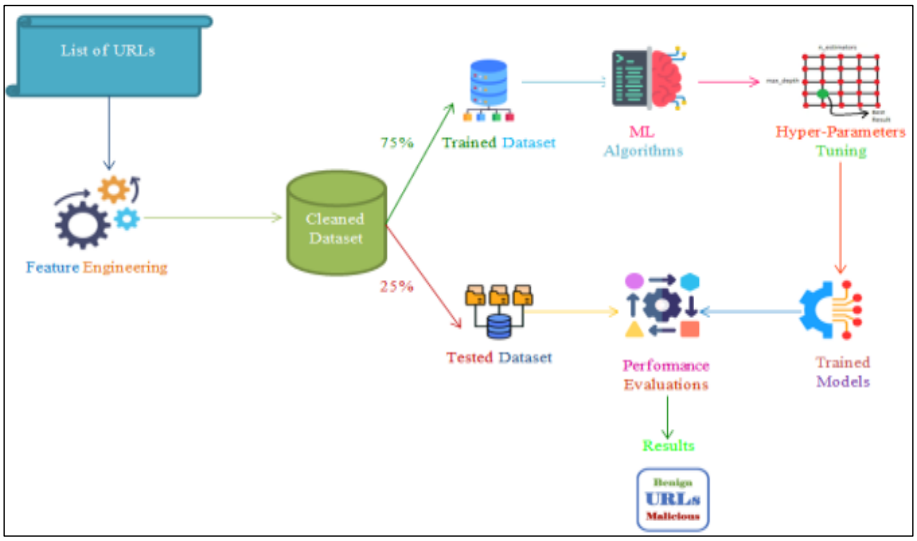


Figure 1. Proposed Workflow for Detecting Malicious URLs.

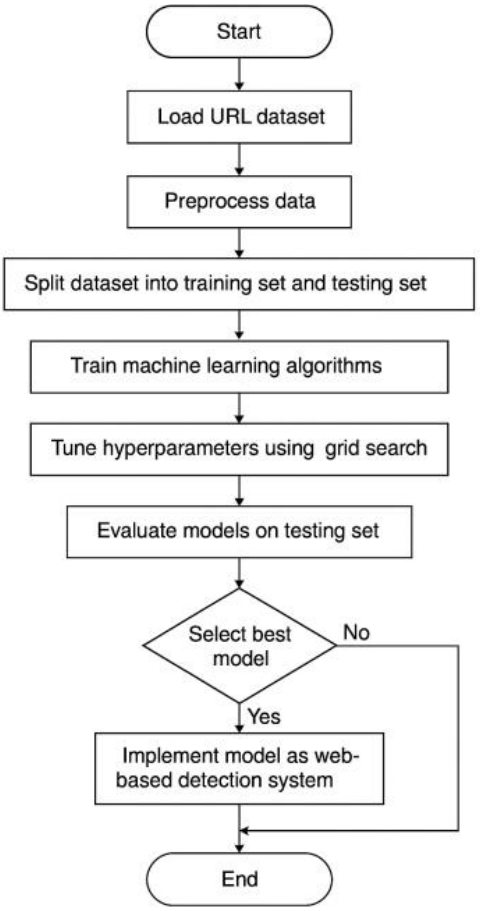


Figure 2. Flowchart of the Proposed Methodology for Malicious URL Detection.

The first stage was to prepare the dataset, which came from Kaggle and consisted of a variety of URLs that were classified as either benign or malicious. Any null values in the dataset were meticulously eliminated to guarantee data integrity. Essential characteristics for identifying malicious URLs were then extracted, producing a refined dataset in which every row denoted a distinct URL, identified by its retrieved features and a label designating whether it was malicious or benign. The dataset was then divided into two subsets: 25% for testing and 75% for training. Using a grid search strategy to find the ideal settings that would improve model performance, the training set was essential for fine-tuning each machine learning model's hyperparameters. By improving the models' ability to generalize, this hyperparameter optimization approach increased the models' accuracy and resilience. Lastly, a range of machine learning assessment metrics were used to evaluate the model performance, offering a thorough understanding of the detection models' dependability and efficacy.

Further, Figure 2 illustrates the sequential workflow of the proposed methodology for detecting malicious URLs using machine learning techniques. The process begins with collecting a labelled dataset consisting of both benign and malicious URLs. Next, the data undergoes preprocessing steps such as cleaning, normalization, and feature extraction to ensure quality input for model training. The refined dataset is then split into training and testing sets. Four machine learning algorithms—Logistic Regression, Support Vector Machine (SVM), Naive Bayes, and Decision Tree—are trained and optimized using Grid Search for hyperparameter tuning. Each model is evaluated based on key metrics, including accuracy, precision, recall, and F1-score. The model demonstrating the best overall performance is selected and deployed in a web-based application using FastAPI, enabling users to perform real-time classification of URLs. This flow ensures a structured and efficient approach to addressing cybersecurity challenges related to unsafe URLs.

3.1. Experiment setup

The experiments were performed using Google Colaboratory, a cloud-based platform that offers powerful computational resources, to efficiently execute large-scale machine learning algorithms—Linear Regression, Support Vector Machine, Naive Bayes, and Decision Tree—for malicious URL identification.

To handle the intensive data processing tasks, the experiments were also conducted in a high-performance computing environment featuring an Intel Core i7-2600 CPU, 16GB RAM, and an NVIDIA GeForce MX250 graphics card. Python was used as the programming language due to its extensive libraries and simplicity in integrating with machine learning frameworks. The models were trained over 175 epochs using the Adam optimizer with a learning rate of 1e-4. This configuration was selected to minimize the risk of overfitting while maintaining a balance between convergence speed and model performance.

The dataset used for unsafe URL detection includes labelled examples of both malicious and benign URLs, along with several features that support classification—such as URL length, domain attributes, and the presence of suspicious keywords. For proper training and evaluation, the dataset was split into 75% for training and 25% for testing. The models learn to distinguish between harmful and legitimate URLs by analyzing these characteristics. Details about the dataset, including the number of instances, class distribution, and applied classifiers, are presented in Table 1.

Table 1. Experiment Dataset for Malicious URL Classification.

Target class	Dataset			Implemented classifier
	Source	Total size	Siez per class	
Benign or Malicious	Kaggle	420.464	210.232	LR, SVM, NB, and DT

The dataset used in this study was sourced from Kaggle and is specifically designed for malicious URL classification. It comprises both benign and malicious URLs, with each record representing a single URL. Each entry includes various features that support classification, such as URL length, presence of special characters, use of suspicious or misleading keywords, and domain-related attributes like the domain is IP-based, among other features.

3.2. Data preparation

We collected 420,464 URLs from the Kaggle data repository, along with the categories that corresponded to them. A resource can be identified by its unique address or URL. Figure 3 illustrates the various components that make up a URL. The protocol that is used to get the object will first be specified; the most used protocols are HTTP (unencrypted) and HTTPS (encrypted connection). Second, the port designates the gateway that should be used to access the material, and the IP address designates the web server that is being requested. The path to the object is the third component of a URL. A list of parameters

that can be used to define keys and values that enable the execution of further actions makes up the fourth section. Lastly, a link makes it possible to navigate to a certain area of the website. Sometimes, a URL may not contain parameters.

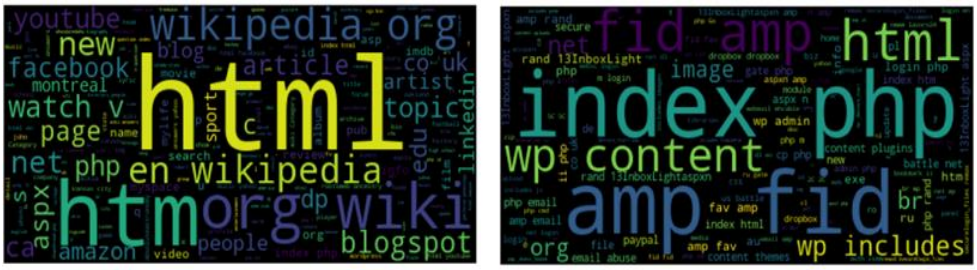
Figure 3. URL Anatomy - Exploring the Different Components of a Web Address.

Raw URL information is not enough to identify malicious URLs since it does not provide useful information about the attributes of the URLs that could help classify them. Feature extraction is required to close this gap, turning unprocessed URL data into meaningful indicators that machine learning systems can analyze efficiently. As a result, a number of input features were selected for machine learning model building and training. These characteristics recorded important data required for the classification task. The following were the features of the retrieved input: URL_Length, Domaine_Length, Has_ipv4, Has_http(s), and among other features. Figure 4 depicts an overview of the datasets used in the classification task.

	url	label
0	4tribes.com.au/wp-content/uploads/2012/11/en_u...	bad
1	ebay.com/sch/i.html?_kw=joe&_kw=weider	good
2	facebook.com/Epimpabonwoy	good
3	ellisisland.org/shipping/FormatTripPass.asp?ss...	good
4	lovemura.net/data/cheditor4/1502/bookmark/ii.p...	bad

Figure 4. Dataset overview for the malicious URL classification task.

The dataset experienced a rigorous cleaning process to ensure its accuracy and suitability for malicious URL detection. This included checking for and handling null values, removing duplicate rows, and resolving label conflicts between "good" and "bad" URLs. Text preprocessing involved converting URLs to lowercase for consistency, removing unnecessary components (such as "http://"), and splitting the URLs into meaningful tokens. Common top-level domains, numeric tokens, and irrelevant symbols were filtered out, while lemmatization was applied to reduce tokens to their base form. Additionally, the original "good" and "bad" labels were encoded as binary values, and the label column was dropped after encoding. To further explore term frequency, a Word Cloud was generated to visualize the most frequent terms in both benign and malicious URLs. These cleaning and preprocessing steps were crucial for preparing the dataset for accurate and efficient model training. Figure 5 shows the Word Cloud representation of the most frequent terms in both benign and malicious URLs.



(a) Benign URLs (b) Maliciuous URLs

Figure 5. Word Cloud Visualization of Term Frequency in Benign and Malicious URLs.

3.3. Machine learning techniques

Although the massive training dataset that was created offered a strong basis for analysis, it also presented computational efficiency issues. A number of machine learning techniques, such as Decision Trees, Support Vector Machines (SVM), Naive Bayes, and Logistic Regression, were used to address issue. These models were selected because they each offer unique benefits in terms of accuracy and interpretability, as

well as the capacity to manage immense datasets efficiently. The upcoming sections will analyze each algorithm's specifics and performance.

3.3.1. Logistic regression

An approach that is frequently used for binary classification tasks is logistic regression. It uses a logistic (sigmoid) function to represent the likelihood that a given input is a member of a specific class (either benign or malicious). The likelihood that a data point belongs to a positive class is represented by a number between 0 and 1, which is what logistic regression produces in contrast to linear regression.

The sigmoid function is the key component of logistic regression, mapping the linear combination of input features X to a probability p. The hypothesis $h_\theta(x)$ has the following formula (Equation 1):

$$h_\theta(x) = \frac{1}{1+e^{-\theta x}} \quad (1)$$

Where:

- ✓ X is the vector of input features,
- ✓ θ is the vector of model parameters (weights)

The logistic function's output, $h_\theta(x)$, indicates the likelihood that a given URL is either benign (0) or malicious (1). Depending on the use case, the decision boundary can be changed, however it is usually set at a probability of 0.5.

3.3.2. Support vector machine

For classification problems, Support Vector Machines (SVM) are effective algorithms, particularly in high-dimensional domains. SVM operates by finding the hyperplane that best divides the classes. Finding a border that optimizes the margin between benign and malicious URLs is the aim in the case of harmful URL detection. SVM seeks to identify the hyperplane that is defined by (Equation 2):

$$w^T x + b = 0 \quad (2)$$

Where:

- ✓ w is the weight vector orthogonal to the hyperplane,
- ✓ x is the feature vector of the input,
- ✓ b is the bias term.

SVM attempts to increase the gap between the positive and negative classes (malicious and benign URLs) as much as possible, which enhances the model's capacity for generalization. By transforming the input into a higher-dimensional space, the kernel method can also be applied to non-linearly separable data.

3.3.3. Naive Bayes

Based on Bayes' Theorem, which says that characteristics are conditionally independent given the class label, the Naive Bayes classifier is probabilistic. Naive Bayes frequently achieves remarkably good results despite its simplicity, particularly in text classification problems like differentiating between harmful and benign URLs based on their properties. The Bayes Theorem provides the essential formula for classification, defined by the (Equation 3):

$$P(C/X) = \frac{P(X/C)\times P(C)}{P(X)} \quad (3)$$

Where:

- ✓ P(C/X)is the posterior probability of class C given features X,
- ✓ P(X/C)is the likelihood of observing features X given class C,
- ✓ P(C) is the prior probability of class C,
- ✓ P(X) is the marginal likelihood of the features X.

The URL is assigned to the class with the highest probability after Naive Bayes calculates the posterior probability for each class. In many real-world situations, especially those with several characteristics, the independence assumption does not substantially affect performance even though it streamlines the computation.

3.3.4. Decision Tree

A decision tree is a model that recursively builds a tree structure by dividing the data into subsets according to the most important attribute at each node. This non-linear model can handle both continuous and categorical data. The decision tree in malicious URL detection divides the URLs into many branches according to feature values (such as URL length, special character usage, etc.) in order to categorize them as either benign or malicious. Gini impurity is a widely used metric for impurity and is defined by the following formula (Equation 4):

$$Gini(D) = 1 - \sum_{i=1}^{k} p_i^2 \quad (4)$$

Where:

- ✓ D is the dataset at the node,
- ✓ p_i is the proportion of samples in class iii at the node,
- ✓ k is the number of classes.

By selecting the feature that maximizes the information gain or minimizes the Gini impurity, the decision tree recursively divides the data at each node. Until a stopping condition is satisfied, the tree keeps splitting (e.g., minimum samples per leaf or maximum depth). Using feature values, the tree is traversed to arrive at the final predictions.

3.4. Evaluation metrics

The effectiveness of the machine learning models in categorizing risky URLs was evaluated using evaluation metrics such as confusion matrix, accuracy, precision, recall, and F1 score. The confusion matrix displays the numbers of true positive (TP), false positive (FP), true negative (TN), and false negative (FN) predictions, offering a thorough understanding of the model's performance. The ratio of accurate predictions to total predictions is used to determine accuracy, which quantifies the percentage of correctly predicted outcomes (Equation 5). Precision highlights the model's capacity to reduce false positives by calculating the proportion of URLs that were deemed unsafe and actually risky (Equation 6). As the model focuses on reducing false negatives, recall quantifies the percentage of real dangerous URLs that were successfully recognized (Equation 7). Lastly, a balanced indicator of a model's accuracy, the F1 score is the harmonic mean of precision and recall (Equation 8). It is especially helpful in situations where there is an imbalance between the classes.

$$Accuracy = \frac{TN+TP}{TN+FP+FN+TP} \quad (5)$$

$$Precision = \frac{TP}{TP+FP} \quad (6)$$

$$Recall = \frac{TP}{TP+FN} \quad (7)$$

$$F1\_score = 2 \times \frac{Precision \times Recall}{Precision+Recall} \quad (8)$$

The model's performance is quantitatively represented by these equations, guaranteeing a clear assessment of the model's capacity to identify dangerous URLs.

4. RESULTS AND DISCUSSION

In this section, the performance of machine learning models in detecting dangerous URLs is evaluated and compared. The models' effectiveness is measured through various assessment metrics, ensuring a comprehensive analysis of their capabilities. To complement the models' performance, we developed an innovative, easy-to-use web application designed to provide an interactive interface for users. Utilizing FastAPI as an efficient backend for fast execution, this application allows users to quickly input URLs through a simple interface. Upon entering a URL, the system interprets the input to determine whether it is malicious or benign, returning the result based on the trained machine learning algorithms immediately. This deployment significantly enhances the accessibility and usability of the URL classification system. Figure 6 depicts the user interface of the web application, showcasing how users can interact with the URL classification system.

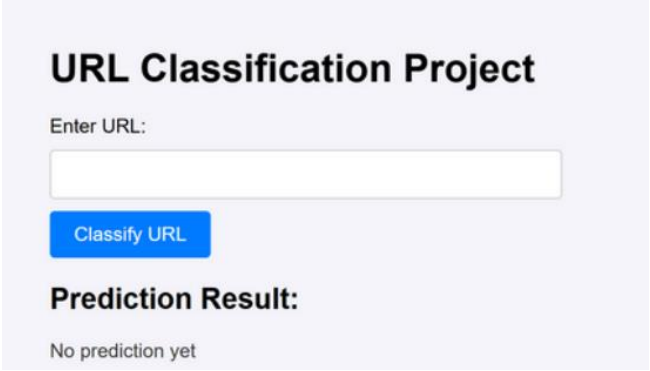


Figure 6. User Interface of the URL Classification Web Application.

The main findings of our study are based on the performance of four machine-learning algorithms for malicious URL detection. The results, as shown in the confusion matrices (Figure 7) and the performance metrics (Table Y), highlight that Naive Bayes outperforms the other algorithms, achieving the highest accuracy (91.9%), precision (94.8%), recall (88.6%), and F1-score (91.5%). Logistic Regression and Support Vector Machine (SVM) follow closely behind, delivering strong and competitive results across all metrics. While Naive Bayes consistently excels, demonstrating its reliability for URL detection, the Decision Tree, though slightly less effective, still provides respectable performance. These findings underline the effectiveness of the selected models, with Naive Bayes emerging as the most robust choice for this task.

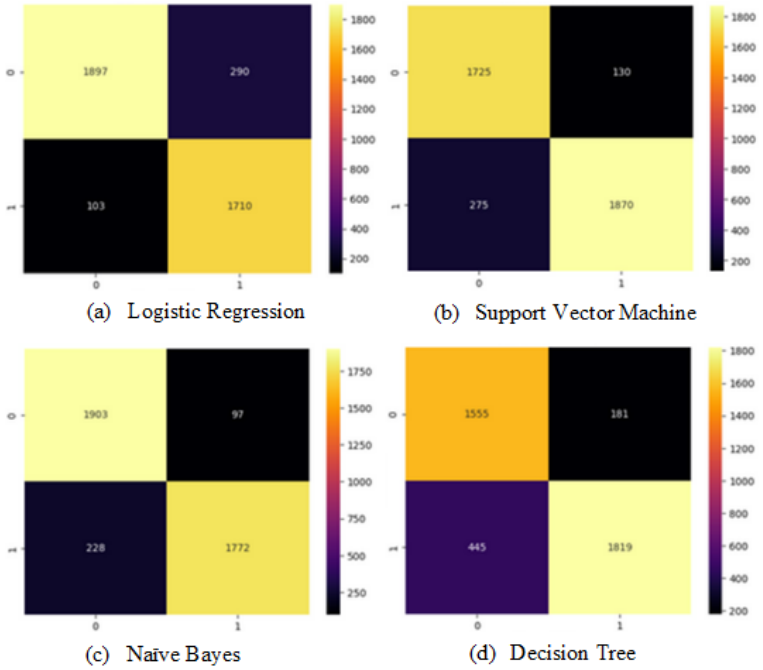


Figure 7. Confusion matrix assessment for different algorithms.

Table 2 provides a detailed breakdown of the performance metrics—accuracy, precision, recall, and F1-score—for each of the four machine learning algorithms. These metrics offer a comprehensive evaluation of the models' effectiveness in detecting malicious URLs. Notably, Naive Bayes leads across all metrics, reflecting its superior ability to correctly classify both benign and malicious URLs. Logistic Regression and Support Vector Machine (SVM) also perform admirably, with competitive scores in all categories. The Decision Tree, while slightly trailing, still demonstrates solid performance, showcasing its potential as a viable alternative for URL classification tasks. The table highlights the nuanced differences between the algorithms, emphasizing the strengths and trade-offs inherent in each model.

Table 2. Result before hyper-parameter tuning

ML Model	Accuracy	Precision,	Recall	F1-score
Logistic Regression	0.902	0.943	0.855	0.897
SVM	0.899	0.871	**0.935**	0.902
Naive Bayes	**0.919**	**0.948**	0.866	**0.915**
Decision Tree	0.844	0.803	0.909	0.853

To facilitate clear intercomparison of the machine learning models used for malicious URL classification, the performance metrics—including accuracy, precision, recall, and F1-score—are presented using bar charts. These visualizations provide a straightforward way to compare the effectiveness of each algorithm, highlighting their strengths and weaknesses in handling the classification task. Figure 8 depicts the bar chart comparison of the four models—Logistic Regression, SVM, Naive Bayes, and Decision Tree—across the key evaluation metrics. The bar charts illustrate how each model performed across different evaluation metrics, allowing for an intuitive assessment of which algorithm is best suited for identifying malicious URLs based on the dataset used.

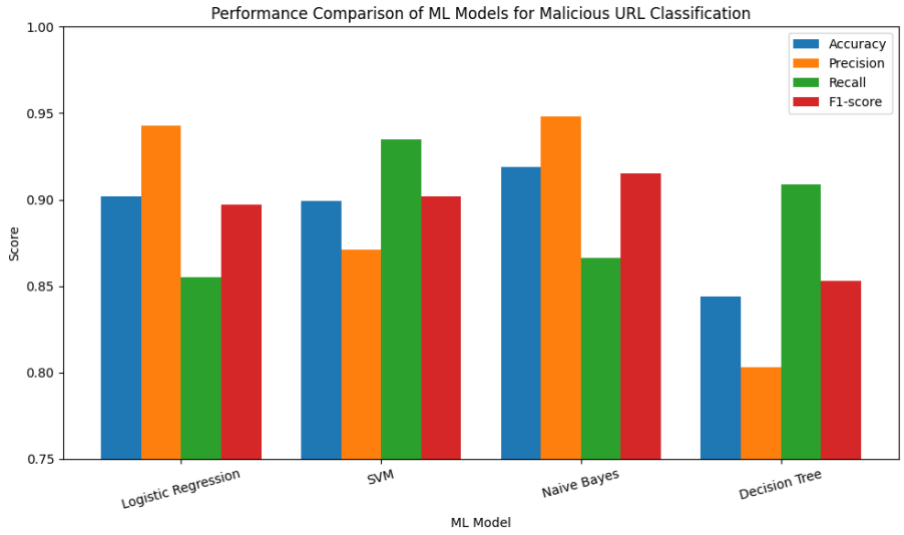


Figure 8. Comparative Analysis of Classifier Performance Using Key Metrics

4.1. Improvement of models by hyper-parameters tuning

Hyperparameter tuning plays a crucial role in optimizing machine learning models for specific tasks and datasets. In this study, we systematically fine-tuned the hyperparameters of the four selected algorithms to enhance their performance in detecting malicious URLs. By carefully adjusting parameters such as learning rate, regularization, and others, we aimed to maximize each model's ability to correctly classify URLs. This process ensured that the models were better suited to the characteristics of the dataset, leading to improvements in both accuracy and efficiency for the malicious URL classification task. Table 3 depicts the improvement in results after the fine-tuning of hyperparameters, showcasing the enhanced performance of each algorithm.

Table 3. The result after hyper-parameter tuning

ML Model	Accuracy	Precision,	Recall	F1-score
Logistic Regression	0.915	0.952	0.874	0.911
SVM	0.908	0.877	**0.949**	0.911
Naive Bayes	**0.924**	**0.955**	0.886	**0.921**
Decision Tree	0.679	0.918	0.801	0.874

After optimizing the hyperparameters, there was an overall improvement in accuracy and other performance metrics for all algorithms, with the exception of Logistic Regression, which experienced a

decline in performance. This outcome highlights the varying responses of different models to parameter adjustments, emphasizing the necessity of adaptive fine-tuning tailored to each algorithm's specific characteristics. A comparison of the models based on both accuracy and recall reveals distinct strengths in their performance. While Naive Bayes achieved the highest overall performance, the Support Vector Machine (SVM) excelled in the recall, a metric that is particularly critical for this task. In malicious URL detection, minimizing false negatives is more costly and impactful than merely maximizing detection accuracy. Therefore, SVM, with its strong recall performance, is considered a highly suitable model for this task, offering a balanced trade-off between performance and the study's primary objectives.

4.2. Research limitations

While our approach demonstrates promising results in malicious URL detection, several important limitations remain that warrant further investigation. One key challenge lies in detecting URLs that are concealed using advanced evasion techniques such as redirections, URL encoding, and obfuscated characters. These methods can manipulate the structural features that machine learning models rely on, thereby reducing detection accuracy. Although our models—particularly Naive Bayes and SVM—showed high precision and recall, they remain vulnerable to zero-day threats and adversarial manipulation, where attackers craft URLs that deliberately bypass known feature patterns.

Another limitation concerns the generalization capability of the trained models. Performance may degrade significantly when encountering URLs that differ from the training distribution, such as those written in rare languages, originating from underrepresented geographical domains, or utilizing non-standard syntax. This points to a need for more diverse and multilingual datasets to train models that can operate effectively across global web environments.

Moreover, our approach primarily leverages static feature extraction, assuming that URL characteristics remain consistent over time. In reality, attackers frequently adapt their techniques, which can render static models obsolete unless continuously updated. To address this, future work should explore adaptive learning strategies, such as online learning or incremental retraining, to ensure that detection models evolve alongside emerging threats. Additionally, the system's reliance on feature-based classification limits its capacity to detect "camouflage" attacks, where URLs mimic benign characteristics to avoid detection. Enhancing detection in such cases may require integrating context-aware analysis or dynamic features, such as domain age, traffic behaviour, and server response patterns.

To improve resilience, future studies should also consider hybrid architectures, combining traditional machine learning with deep learning models such as CNNs or RNNs, which can extract hierarchical patterns in URL strings. These models, when trained on enriched datasets, could uncover subtle indicators of malicious intent that simpler algorithms might miss. Furthermore, implementing ensemble methods or attention-based models could bolster robustness and minimize overfitting to specific URL structures. In summary, while our current methodology offers a strong foundation, future enhancements should emphasize adaptability, scalability, and deeper semantic understanding to tackle the evolving nature of cyber threats effectively.

5. CONCLUSION

Enhancing cybersecurity and protecting consumers from online dangers can be achieved by recognizing and blocking malicious URLs. As a result, there may be less chance of financial loss and an economy that is stable and sustainable. In this study, four machine learning models were developed and evaluated for the classification of unsafe URLs, demonstrating our involvement in the application of these models both prior to and following hyperparameter adjustment. In order to provide a baseline for comparison, we first evaluated the models' performance using their default parameters. We then adjusted the hyperparameters to maximize model efficiency and accuracy, which greatly enhanced the detection of dangerous URLs. By showing a noticeable improvement in classification results and highlighting the significance of hyperparameter adjustment in improving model performance, our method supports the efficacy of machine learning techniques in cybersecurity applications.

Among the evaluated models, Naive Bayes achieved the highest overall performance, with an accuracy of 92.4%, precision of 95.5%, recall of 88.6%, and F1-score of 92.1% after hyperparameter tuning. Support Vector Machine (SVM) also demonstrated strong recall performance at 94.9%, making it highly effective for minimizing false negatives. These results validate the effectiveness of traditional machine learning algorithms in detecting malicious URLs, especially when optimized properly. By showing a noticeable improvement in classification results and highlighting the significance of hyperparameter adjustment in improving model performance, our method supports the efficacy of machine learning techniques in cybersecurity applications.

Several improvements can be offered to enhance the study for future research. First, exploring additional algorithms, such as deep learning models or ensemble methods, could provide further insights into improving URL classification accuracy. Additionally, expanding the dataset to include a more diverse range of malicious and benign URLs from various sources could enhance model generalization. It would also be beneficial to investigate real-time detection systems and evaluate the models' performance in dynamic, evolving environments. Lastly, incorporating advanced feature engineering techniques and leveraging external threat intelligence data could further strengthen the models' ability to detect sophisticated malicious URLs.

DATA AVAILABILITY STATEMENT

The data presented in this study are available on request from the corresponding author.

CONFLICTS OF INTEREST

The authors declare that they have no conflicts of interest in this work.

REFERENCES

[1] B. B. Gupta, K. Yadav, I. Razzak, K. Psannis, A. Castiglione, and X. Chang, "A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment," *Comput. Commun.*, vol. 175, pp. 47–57, Jul. 2021, doi: 10.1016/j.comcom.2021.04.023.

[2] M. Veale and I. Brown, "Cybersecurity," *Internet Policy Rev.*, vol. 9, no. 4, Dec. 2020, doi: 10.14763/2020.4.1533.

[3] S. H. Ahammad *et al.*, "Phishing URL detection using machine learning methods," *Adv. Eng. Softw.*, vol. 173, p. 103288, Nov. 2022, doi: 10.1016/j.advengsoft.2022.103288.

[4] B. Wardman, "Phorecasting Phishing Attacks: A New Approach for Predicting the Appearance of Phishing Websites," *Int. J. Cyber-Security Digit. Forensics*, vol. 5, no. 3, pp. 142–154, 2016, doi: 10.17781/P002156.

[5] N. Virvilis, A. Mylonas, N. Tsalis, and D. Gritzalis, "Security Busters: Web browser security vs. rogue sites," *Comput. Secur.*, vol. 52, pp. 90–105, Jul. 2015, doi: 10.1016/j.cose.2015.04.009.

[6] F. O. Catak, K. Sahinbas, and V. Dörtkardeş, "Malicious URL Detection Using Machine Learning," 2021, pp. 160–180. doi: 10.4018/978-1-7998-5101-1.ch008.

[7] C. Crisci, B. Ghattas, and G. Perera, "A review of supervised machine learning algorithms and their applications to ecological data," *Ecol. Modell.*, vol. 240, pp. 113–122, Aug. 2012, doi: 10.1016/j.ecolmodel.2012.03.001.

[8] M. Aldwairi and R. Alsalman, "MALURLS: A Lightweight Malicious Website Classification Based on URL Features," *J. Emerg. Technol. Web Intell.*, vol. 4, no. 2, May 2012, doi: 10.4304/jetwi.4.2.128-133.

[9] C. Do Xuan, H. Dinh, and T. Victor, "Malicious URL Detection based on Machine Learning," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 1, 2020, doi: 10.14569/IJACSA.2020.0110119.

[10] S. He, J. Xin, H. Peng, and E. Zhang, "Research on Malicious URL Detection Based on Feature Contribution Tendency," in *2021 IEEE 6th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*, IEEE, Apr. 2021, pp. 576–581. doi: 10.1109/ICCCBDA51879.2021.9442606.

[11] X. Yu, "Phishing Websites Detection Based on Hybrid Model of Deep Belief Network and Support Vector Machine," *IOP Conf. Ser. Earth Environ. Sci.*, vol. 602, no. 1, p. 012001, Nov. 2020, doi: 10.1088/1755-1315/602/1/012001.

[12] A. Zamir *et al.*, "Phishing web site detection using diverse machine learning algorithms," *Electron. Libr.*, vol. 38, no. 1, pp. 65–80, Mar. 2020, doi: 10.1108/EL-05-2019-0118.

[13] R. S. Rao and A. R. Pais, "Detection of phishing websites using an efficient feature-based machine learning framework," *Neural Comput. Appl.*, vol. 31, no. 8, pp. 3851–3873, Aug. 2019, doi: 10.1007/s00521-017-3305-0.

[14] K. S. Adewole, A. G. Akintola, S. A. Salihu, N. Faruk, and R. G. Jimoh, "Hybrid Rule-Based Model for Phishing URLs Detection," 2019, pp. 119–135. doi: 10.1007/978-3-030-23943-5_9.

[15] N. Reyes-Dorta, P. Caballero-Gil, and C. Rosa-Remedios, "Detection of malicious URLs using machine learning," *Wirel. Networks*, vol. 30, no. 9, pp. 7543–7560, Dec. 2024, doi: 10.1007/s11276-024-03700-w.

[16] A. Hamza, F. Hammam, M. Abouzeid, M. A. Ahmed, S. Dhou, and F. Aloul, "Malicious URL and Intrusion Detection using Machine Learning," in *2024 International Conference on Information Networking (ICOIN)*, IEEE, Jan. 2024, pp. 795–800. doi: 10.1109/ICOIN59985.2024.10572207.

[17] D. Orozco-Fonseca, G. Marín, and A. Lara, "Taxonomy of Malicious URL Detection Techniques," 2024, pp. 73–81. doi: 10.1007/978-3-031-54235-0_7.

[18] A. Astorino, A. Chiarello, M. Gaudioso, and A. Piccolo, "Malicious URL detection via spherical classification," *Neural Comput. Appl.*, vol. 28, no. S1, pp. 699–705, Dec. 2017, doi: 10.1007/s00521-016-2374-9.

BIOGRAPHIES OF AUTHORS

Mohamed Cherradi Received a Bachelor's degree in Physical Science, a degree in Mathematics and Computer Science, and an Engineering Diploma in Computer Engineering. He later obtained a Doctorate in Computer Science. Currently, he serves as an Assistant Professor in Computer Science at the National School of Applied Science, Al Hoceima. His research interests include data mining, machine learning, artificial intelligence, and computational science. He has published quality papers in renowned journals indexed in SCI, WoS, and SCOPUS. He can be contacted at email: m.cherradi@uae.ac.ma.

Hajar El Mahajer received a Bachelor's degree in Mathematical Science and a Master's degree in Computer Science. She is currently a PhD student in Computer Science at the Faculty of Science and Technology, Tangier. Her research interests include data science, machine learning, and computational mathematics. She can be contacted at email: hajarelmahajer@gmail.com.