Journal of Information Security and Applications 77 (2023) 103553

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

A high-accuracy phishing website detection method based on machine learning

Mahdi Bahaghighat [a,*], Majid Ghasemi [a], Figen Ozen [b]

[a] Computer Engineering Department, Imam Khomeini International University, Qazvin, Iran
[b] Halic University, Türkiye

ARTICLE INFO

Keywords:
Phishing website detection
Cyber security
Machine learning
Classification
XGBoost

ABSTRACT

The rapid development of e-commerce, e-banking, and social networks has made phishing attack detection one of the most critical technologies in all cyber security systems. To improve the efficiency of anti-phishing techniques, we present an improved predictive model based on machine learning. The proposed method uses six different algorithms; Logistic Regression, K-Nearest Neighbors, Naive Bayes, Random Forest, Support Vector Machine, and Extreme Gradient Boosting (XGBoost). The experiments are based on a public dataset of 58,000 legitimate websites and 30,647 phishing ones, including 112 attributes for each sample. Our evaluations in the feature selection process show that after balancing the dataset and dropping constant features, a noticeable improvement can be achieved. We conducted our evaluation found on eight major unique scenarios. The experimental results of our phishing websites detection (PWD) method indicate remarkable performances in which each algorithm reached an accuracy of more than 93%, and the XGBoost classifier outperforms others with 99.2% overall accuracy, 99.1% precision, 99.4% recall, and 99.1% specificity. In addition, the study achieved optimal run-time of about 1500 ms for the XGBoost algorithm without dimension reduction while using Principal Component Analysis (PCA) reduces it down to just 869 ms. As a result, the proposed approach would be practical in both offline and real-time applications.

1. Introduction

Free access to information sources and public networks is considered a very primitive right in many countries. Since the internet age, fundamental changes have been imposed on human being lifestyles to be well adapted. Owing to the fact that e-commerce and users' online purchases have been increasing rapidly, phishing has become one of the most threatening forms of internet attacks. The preparation to defend our resources is set to become a vital factor in our contemporary society [1]. By entering sensitive information on fake websites that look like the original ones, the user gives access to their credit card details, passwords, and etc., to fraudulent websites.

These kinds of attacks target ordinary people who do not know much about threatening cyber-attacks and cyber security issues, as well as companies and social networks. In phishing attacks, fraudsters exploit the resemblance between authentic and phishing websites that look precisely similar. Moreover, they host their fake websites in genuine domains, available on Google or different service providers. They even advertise those websites on other legitimate websites to attract people from all walks of life [2].

The other action they take is to send fake emails that usually contain either a sense of fear or urgency, such as requesting to change their passwords to avert debit/credit card suspension or attaching a fake website link to the email. Similarly, by means of social media platforms, namely WhatsApp, Facebook, and Instagram, they convince people to give them critical information. By updating their confidential credentials unknowingly, users give cybercriminals access to their personal information. Thus, they gain access to people's credit/debit cards and can harm those individuals [3,4]. Phishing attacks can be found anywhere; in the past, they only tried to charge people; however, these days, they deliver malicious software and other things to the victims, such as computers and cell phones [5].

Approximately half of all cyber fraud affecting Internet users involves phishing attacks. In addition to the problems mentioned earlier with phishing, it must be noted that phishing attacks have trebled in 2022 compared to 2020, indicating the need for new technologies in this field. Phishing is the main threat to industries, namely e-commerce and Internet banking, especially in developing countries where facilities are

* Corresponding author.
E-mail address: Bahaghighat@eng.ikiu.ac.ir (M. Bahaghighat).

https://doi.org/10.1016/j.jisa.2023.103553

Available online 26 August 2023
2214-2126/© 2023 Published by Elsevier Ltd.

not up-to-date enough to recognize these threats before happening [6]. It is possible to protect these users from phishing attacks using various techniques, including the rule-based approach [7], the heuristic approach [8], URL-based approaches [9–11], and machine learning (ML) approaches.

With advances in machine learning and deep learning, it is possible now to achieve high-accuracy algorithms [12–16]. For practical purposes, these algorithms ought to accurately classify phishing and legitimate websites. Classification algorithms are widely deployed to categorize data into some classes or categories based on their essential differences or similarities [17–19].

As it is clear, among the machine learning techniques, classification algorithms are appropriate for phishing problems. In the phishing website detection problem, the needed features are extracted from different sources such as URLs, DNS, or information stored on other websites such as Phishtank or WHOIS [20,21].

Along with this development, unfortunately, the existing machine learning approaches have had several problems, such as low accuracy, lack of adaptations to real-world challenges, low speed, relatively small dataset, etc. Consequently, there is a requirement for a high-accuracy, fast, and dependable machine learning approach to meet those demands.

Although the mentioned approaches are helpful, supervised ml-based ones are more popular because of their reliability and high accuracy. They are primarily used for classification, which is the case in phishing problems. In [22], various techniques were used to select features and several algorithms, including Random Forest (RF), Neural Networks (NN), Bagging, etc. In [23], they generated a new classification model based on heuristic features using third-party URLs, source code, and services. The highest accuracy was achieved by Principal component analysis Random Forest (PCA-RF). [24] introduced a novel features selection based on hybrid ensemble (HEFS), which is a two-phase process. In [10], they used various classification algorithms and an NLP-based approach on a relatively large dataset with just over 73,500 instances extracted from URLs. The other URL-based approach was presented in [9], which analyses hyperlinks in the website's HTML source code to detect phishing attacks and evaluates results with several algorithms namely RF, Support Vector Machines (SVM), C4.5, etc.

In [25], four sets of features were used to identify phishing in online banking. Web pages are classified using SVM.

In [26], a novel algorithm was proposed to detect phishing, which uses Binary Modified Equilibrium Optimizer (BMEO) coupled with an AV-shape transfer function (AV-BMEO) and KNN.

Our study uses supervised machine learning to classify phishing websites. Dropping constant features as a result of the feature selection procedure tries to decrease the run time while increasing the accuracy of our algorithm. Due to the fact that phishing website classification is an imbalanced problem, in the targeted dataset [27], phishing instances are less than legitimate ones; thus, the SMOTEENN method is used by us to balance the dataset and leads to achieving higher accuracy compared to the imbalanced dataset. In this study, we compare different algorithms, including random forest (RF), SVM, logistic regression (LR), KNN, Naive Bayes, and Extreme Gradient Boosting (XGBoost), on the balanced dataset. Finally, in detail, the performance evaluation measurements including accuracy, recall, precision, F1_score, and specificity are utilized to examine the effectiveness of classification algorithms.

1.1. Motivation

Internet advancements have resulted in an increase in fraudster activities owing to a variety of factors, including the increase in online transactions and the limited knowledge of users in this field. In the first quarter of 2022, APWG recorded 1025,968 phishing attacks [6]. Phishing attacks against social media sets increased from 8.5% in the last quarter of 2021 to 12.5% in the first quarter of 2022. About 65% of cybercriminals utilize spear phishing emails as their primary attack method [6] (see Fig. 1). Phishing websites are also places where individuals are scammed and tricked into paying extra fees. Consequently, in this era, it is imperative that resources are protected from these attacks. As well as causing damage to infrastructures, phishing attacks are also causing high costs for governments and individuals. The use of phishing websites when making online purchases can result in individuals being charged up to ten times more than they should be charged. Along with the above costs, tracing the roots of these issues is extremely time-consuming for both individuals and governments. To address this issue, we have developed a real-time high-accuracy approach that can be used to detect phishing websites efficiently with low error rates.

The upcoming sections of the paper are as follows. A review of the works in the literature is presented in Section 2. Section 3 describes our methodology and techniques to develop the best classifier for phishing website detection. Section 4 examines the implementation detail, evaluation metrics, and experimental results in profound detail, and Section 5 explains the conclusion and future works.

2. Related works

Typically, phishing website detection (PWD) algorithms rely upon supervised machine learning techniques. An algorithm (classifier) maps inputs to outputs using a particular function [28,29]. In the process of learning, a classifier learns several features (variables or inputs) that help predict the output (response). A classifier learns a website's specific features (characteristics) to determine whether it is legitimate or phishing [30].

In [9], they proposed an approach to detect phishing websites incorporating various hyperlink-specific features. Machine learning algorithms were trained using 12 categories of hyperlink-specific features. A third party was not required, and it is an entirely client-side solution. Furthermore, that method is language-independent and detects websites written in any language. SMO, NB, RF, SVM, Adaboost, NN, C4.5, and Logistic Regression (LR) were used, and as far as accuracy is concerned, LR achieved the highest value of 98.42%.

A new real-time phishing detector system was suggested in [10], with the help of various algorithms and natural language processing (NLP)-based features learned from a relatively extensive URL dataset with just over 73,500 instances extracted from URLs (36,400 non-phishing and 37,175 phishing). They called this work language independence due to the usage of NLP. Decision tree, Adaboost, K-star (with only ten percent of the dataset because of the time it takes to process it), KNN ($n = 3$), RF, SMO, and NB were used, in which K-star, KNN, NB, and the hybrid features (a combination of NLP features and word vectors) involved, while for rest of the features of the algorithms were used. Moreover, NLP features outperformed the classification of

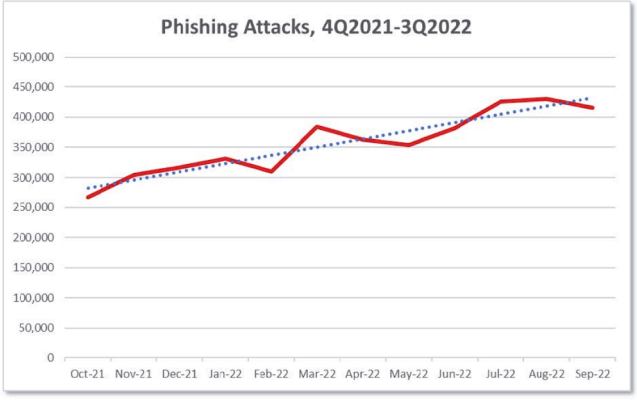


Fig. 1. Phishing attacks from the last quarter of 2021 to the third one in 2022, according to [6].

URLs in almost all ML algorithms except NB. Also, hybrid features increase the system's performance by 2.24% based on NLP features and 13.14% based on word vectors. The best accuracy (about 98%) was attained by RF, which uses NLP features only.

In [24], a feature selection framework in ML-based phishing classification was introduced, and it was called the Hybrid Ensemble Feature Selection (HEFS), which is a two-phase process in total. A Cumulative Distribution Function gradient (CDF-g) algorithm is used in the initial stage to create a primary feature subset. Data perturbation ensembles were used to generate a secondary feature subset. In the next stage, a set of base attributes was extracted from the latter subset by utilizing a function perturbation ensemble. They concluded that the best result could be achieved by integrating RF and HEFS, where they attained an accuracy of 94.6% with just 20.8% of the original attributes. Another experiment showed that the base features (in total 10) used on RF are more successful than the whole features (48 in total) used on SVM, NB, and C4.5.

In [31], an ML-based anti-phishing system was introduced, and they called it PHISH-SAFE. This system takes advantage of URL features. To evaluate the system accurately, they took only 14 features from more than 33,000 phishing and legitimate URLs from different sources extracted from Phishtank.com, including IP addresses, subdomains, length of URLs, etc. Java was used for feature extraction, and then the attributes were stored in a matrix's rows which is sparse. Two ML algorithms were used to assess the result of PHISH-SAFE, namely NB, and SVM; the SVM classifier performed better with an accuracy of just over 90% in detecting phishing websites. Although in this work authors took advantage of a relatively large dataset with more than 33,000 instances, the dataset has only 14 features, which makes it a non-practical one compared to other research projects. In terms of accuracy, it only acquired an accuracy of 90%, which is also not much practical in real-world problems. These limitations ought to be considered in this work.

In [32], they aimed to examine the usefulness of different ML techniques and algorithms in recognizing phishing attacks and report their pros and cons. They also designed a phishing classifier system to overcome widely used phishing detection techniques. The study also used a numeric representation and compared classical machine learning techniques, including RF, KNN, Decision Tree, Linear SVM classifier, One-class SVM classifier, and wrapper-based features selection, which includes URL information, and the dataset was gathered from different sources with Data Mining approaches. RF achieved the best accuracy with 96.87% overall.

An evaluation of up-to-date studies on phishing classification using auguring modeling was presented in [33], along with a discussion of whether these approaches are applicable to phishing attacks or not. Using real phishing datasets and a variety of evaluation measures, they tested four sundry rule-based classifiers, including greedy, associative, and rule induction approaches. Furthermore, they compared the derived classifiers with classical algorithms, such as Bayes Nets and Simple Logistics. Predictability models with rules were compared to determine their strengths and weaknesses and their actual performance in detecting phishing. It has demonstrated that eDRI, a recently developed greedy algorithm, produces valuable models and is highly competitive in terms of prediction accuracy and run-time when used as an anti-phishing tool. Notably, rule-based classifiers had an average error rate of 5.93%, higher than Bayes Net and Simple Logistics by 1.08% and 0.47%, respectively.

Scientists in [34] developed a phishing classification based on website URLs using a point-wise mutual information method. Additionally, a system for identifying phishing websites with the help of machine learning was created. They used several techniques, including "Sentiment, Embedding, Lexicon characteristics, and PMI-semantic orientation." Regarding algorithms, SVM, RF, NB, KNN, and Decision Trees were applied to extracted features. RF outperformed among those algorithms after ten-fold cross-validation, with an accuracy of 90.363% when the train-test ratio is 80 to 20. According to the study, both multi-class and binary scenarios show promising results in terms of "kappa values, improved accuracy, and f-values. Even though several algorithms were utilized along with some helpful techniques, 92.5% overall accuracy can definitely be considered a limitation of their work.

In [35], an ML-based classification model was introduced with improved effectiveness in detecting phishing websites. The predictive model's feature selection module constructed an effective feature vector. An incremental component-based system extracts this information from URL, webpage properties, and webpage behavior, then presents it as feature vectors to the models. The system uses SVM and NB, which were trained on a 15-dimensional feature set. They examined a tiny dataset consisting of 2541 phishing instances and 2500 legitimate ones. Both SVM and NB models demonstrate outstanding performance with only 0.04% False Positives and 99.96% accuracy using ten-fold cross--validation. The dataset has a relatively small number of instances compared to other works, which cannot be a comprehensive approach to take in phishing detection problems.

In [36], the authors put emphasis on studying phishing attacks through the eyes of the URL and Domain Identity feature, Abnormal Features, HTML and JavaScript Features, and Domain Features as semantic features to detect phishing websites, allowing the classification process to be more controlled and effective by using the semantic features. Using 16 machine learning algorithms and 10 semantic features, they were able to detect phishing webpages from two datasets using 16 machine learning algorithms. According to the comparison results, GradientBoostingClassifier and RandomForestClassifier were the most accurate (i.e., about 97%). Compared to other classifiers, GaussianNB and stochastic gradient descent (SGD) are the least accurate; 84% and 81%, respectively.

CANTINA operated on the content of the websites that are text. Search engines that use the term frequency and inverse document frequency (TF-IDF) algorithm can identify phishing websites based on the words that frequently appear in the textual content of websites. Furthermore, it includes heuristics and a straightforward linear model for classifying websites. Due to the fact that this approach relies upon the textual content of the dubious website, it stops working when images replace the explicit content. Since it relies upon third-party search engines, it has a performance problem [37].

In [38], a new approach based on CANTINA+ was proposed, which is an innovative feature-based approach that used the DOM to detect phishing and incorporates a search engine, mediator services, and ML techniques. In addition, two distinctive filters were designed to help decrease False Positives (FP) and quicken the run-time. In the first detector, hashing was used to detect highly similar phishing attacks. Secondly, there was a filter for the login system, which identifies legitimate web pages without a login form. Those methods were tested on a relatively small dataset with only 8118 phishing and 4883 non-phishing instances. Also, they compared the result of CANTINA+ with several machine learning algorithms. Based on the randomized evaluation, CANTINA+ achieved 92% True Positive (TP) on unique testing phishing, 99% TP on near-duplicate testing phishing, and 0.4% FP with 10% training phishing. In a time-based evaluation, CANTINA+ achieved more than 99% TP for near-duplicate testing phishing and approximately 1.4% FP for training phishing less than 20%.

In [39], they conducted research on recognizing web phishing with concept features. Along with the features that were introduced in CANTINA [37], they applied additional domain top-page resemblance features to an ML-based phishing classifier to achieve more accuracy. However, the dataset was relatively too small, with only 200 web data; half of them are phishing websites, and the rest are legitimate. Several different ML algorithms were used; NB, Neural Network (NN), RF, SVM, J48, and Adaboost. The f-measure of these algorithms were 0.8100, 0.9250, 0.9140, 0.9120, 0.8910, and 0.9050. The least error rate is for NN. Regarding the results, they could not manage to reach a high accuracy in general. The highest accuracy, in this case, is 92.5%. Thus, it

can definitely be considered a limitation of their work. However, acquired results indicate that even with feature alteration of CANTINA [37], the ML-based method exceeds the heuristic method.

A classifier was presented in [23] that works with heuristic attributes that were retrieved from URLs, source codes, and mediator services. The model had been assessed by several ML classification algorithms. RF outperformed all other algorithms with an accuracy of more than 99%. The remaining approach was also examined with and without third-party-based aspects to determine the efficiency of arbiter offerings in classifying dubious websites. Lastly, the results were also compared with the base methods (CANTINA and CANTINA+ [37,38]).

In [40], they proposed a new approach to record and recognize the critical features for ML-based classification of phishing websites. Machine learning uses a variety of features in the detection process. Based on research studies, the following features were collected and grouped:

- Features based on URL
- Domain-related attributes
- Features derived from pages
- Content-Based attributes

To determine if a website contains phishing URLs, it is first necessary to analyze its URL-based features. Some of these are:

- Counting the digits in the URLs
- Entire length of URLs
- Is URLs Typo squatted or no

By detecting a phishing domain, phishing attacks can be detected. Hence, passive queries related to domains that are intended to be deceptive or misleading do not provide useful information. Some of them are listed here [40]:

- An established standing service has blacklisted the domain name or IP address?
- The domain registration was acquired how many days ago?
- Does the registrant's name appear obscured?

Regarding features that rely upon pages, which help give information about the reliability of websites, some of them listed in the paper are:

- Global PageRank
- Country PageRank
- Position at the Alexa top 1 million sites

Moreover, others provide information about the users' activity on the landing page, and several of them are as follows:

- Mean of page views for each site
- The average of length of visits
- The amount of each country's traffic share
- etc.

The last introduced feature is the content-based feature, in which an active scan of the target domain is required to get this functionality. In order to determine whether the target domain is being used for phishing, page content starts processing. Page processing information includes:

- Title of the pages
- Meta tags
- Text that are hide
- Text in the body page
- Images
- etc.

As part of [41] survey, a detailed analysis of phishing attack methods and defense techniques was conducted in order to address existing phishing detection limitations. It was further divided into five folds by the authors. The first phase of this study focused on motivations, lifecycles, and phishing history. Additionally, a variety of distribution methods were introduced for the purpose of spreading phishing attacks. The next phase of the study included a taxonomy of various phishing attack techniques used on desktops and mobile devices. In the fourth phase, the development of phishing protection mechanisms will be compared. As a final point, the article presents a number of performance challenges that developers are confronted with when dealing with this important attack. It was also discussed in this paper what the consequences of phishing attacks are in emerging domains such as mobile and online social networks.

Authors in [42] proposed a new protocol that uses timestamps to avoid desynchronization and disclosure, which is a crucial step toward improving the security of RFID-enabled IoT devices, and play a vital role in securing devices against different sorts of attacks.

3. Methodology

In the age of the internet, everything is virtual, and this is causing the rise of phishing attacks, especially as banking systems and e-commerce are going online. Both Internet users and the industry suffer tremendous economic losses as a result [1]. Consequently, it is imperative to develop practical solutions that provide high levels of accuracy and fast response times when it comes to detecting phishing attacks. In this study, we develop a practical method to detect phishing websites based on machine learning approaches. Explaining the dataset with visualization tools, feature selection, performing Imblearn's SMOTEENN to make the dataset balanced, creating an innovative model, testing, and ultimately juxtaposing the results are the components of our method to detect phishing websites. In Fig. 2, the individual components that make up the proposed phishing detection system can be seen.

The dataset is then utilized for training a model. As shown in Fig. 2, the model's outcome variable falls into two categories: Training and Testing. The model is trained with Logistic Regression (LR), KNN, Naive Bayes (NB), Random Forest (RF), Support Vector Machine (SVM), and XGBoost. In order to select the best features, constant features are detected and then dropped. The results show that removing redundancies in the data source that has no helpful information can reduce the run-time and increase the overall accuracy. Additionally, we balance the dataset using the SMOTEENN method. 80% of the dataset is used to train our model, and 20% of the dataset is used for testing to evaluate its performance. We also conduct our implementations found on eight different scenarios, but first, we should elaborate more details about the public dataset used in this study.

3.1. Dataset

The collection of data is the initial step in any data science project. Having the appropriate dataset is critical to ensuring the correctness of the outcomes. We ought to use a dataset that is capable of clarifying and explaining the differences between phishing and legitimate activity. In our work, a public dataset has been evaluated with a deep investigation to conclude whether it is proper to forecast future phishing attacks or not. All of the characteristics are gathered from Phishing Websites Dataset, which was published in 2020 [27]. The dataset has 88,647 instances with 111 features and a labeled output column. According to Fig. 3, the number of non-phishing instances (0) is 58,000, and the number of phishing ones (1) is 30,647. This ratio of instances makes the dataset relatively imbalanced, and in order to resolve this issue, we use a balancing method described in the following sections of this paper. Moreover, the dataset neither does contain any categorical variables nor null values. Notably, all the features' types are numbers, 111 of them are int64 and the only float64 feature is the "time_response" feature.

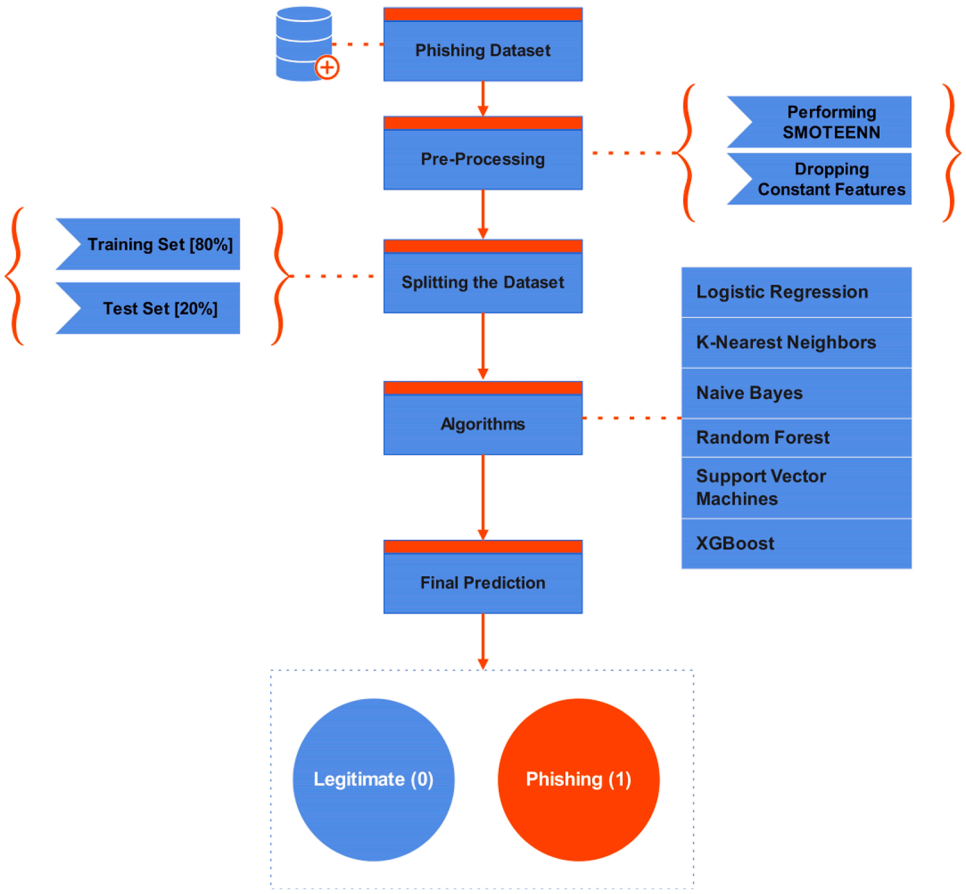


Fig. 2. Illustration of the proposed model on phishing dataset.

In Tables 1 and 2, you can see a complete list of all 112 attributes (111 features + 1 output), and the definition of each is available in [27]. The attributes of the prepared dataset can be divided into six groups:

- Attributes found on the entire URL properties
- Attributes found on the domain properties
- Attributes found on the URL directory properties
- Attributes found on the URL file properties
- Attributes found on the URL parameters
- Attributes found on the URL resolving data and external metrics

A thorough understanding of URL structure is necessary so as to understand how attackers create phishing domains. Web pages are addressed using a Uniform Resource Locator (URL). A typical URL structure is illustrated in Fig. 4.

3.2. Preprocessing

We use standardization and standard scaling to perform feature scaling, which is one of the most crucial steps in developing an ML-based application. Feature scaling is essential because ML algorithms do not work well when numeric input attributes are in widely different scales. For our study, the dataset is clean enough not to contain null values or categorical variables, so scaling the features with the standardization method is the only preprocessing step. In standardization, the data ought to conform to a Gaussian distribution with mean μ and standard deviation σ so it can be used for standardization methods. A statistical variable is standardized when it has a mean of zero and a standard deviation of one, which is known as a set of values [43]. Eq. (1) is used to standardize a value:

$$y = \frac{(x - \mu)}{\sigma} \tag{1}$$

Where the x is the value, we try to standardize it, μ is the mean, σ is the standard deviation, and N is the number of observed samples. Mean (μ) is calculated as:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{2}$$

Moreover, the standard deviation (σ) is calculated as:

$$\sigma = \sqrt{\sum_{i=1}^{N} \frac{(x_i - \mu)^2}{N}} \tag{3}$$

3.3. Feature selection

Feature selection is an important step in machine learning which refers to selecting a subset of relevant features from a larger pool of available features to improve model performance. Feature selection can also reduce the complexity and computational requirements of a model by reducing the number of variables and allowing faster training. As a result, we considered feature selection procedure in our scenarios.

3.3.1. Balancing the dataset

In a classification problem, class imbalance occurs when some classes are more prevalent than others, so standard classifiers usually tend to be overtaken by major classes and ignore minor ones. In some practical applications, the ratio of minor to major classes can be quite dramatic, such as 1 to 100, 1000 to 10,000, or even more. This problem is common in many applications such as fraud/intrusion detection, phishing

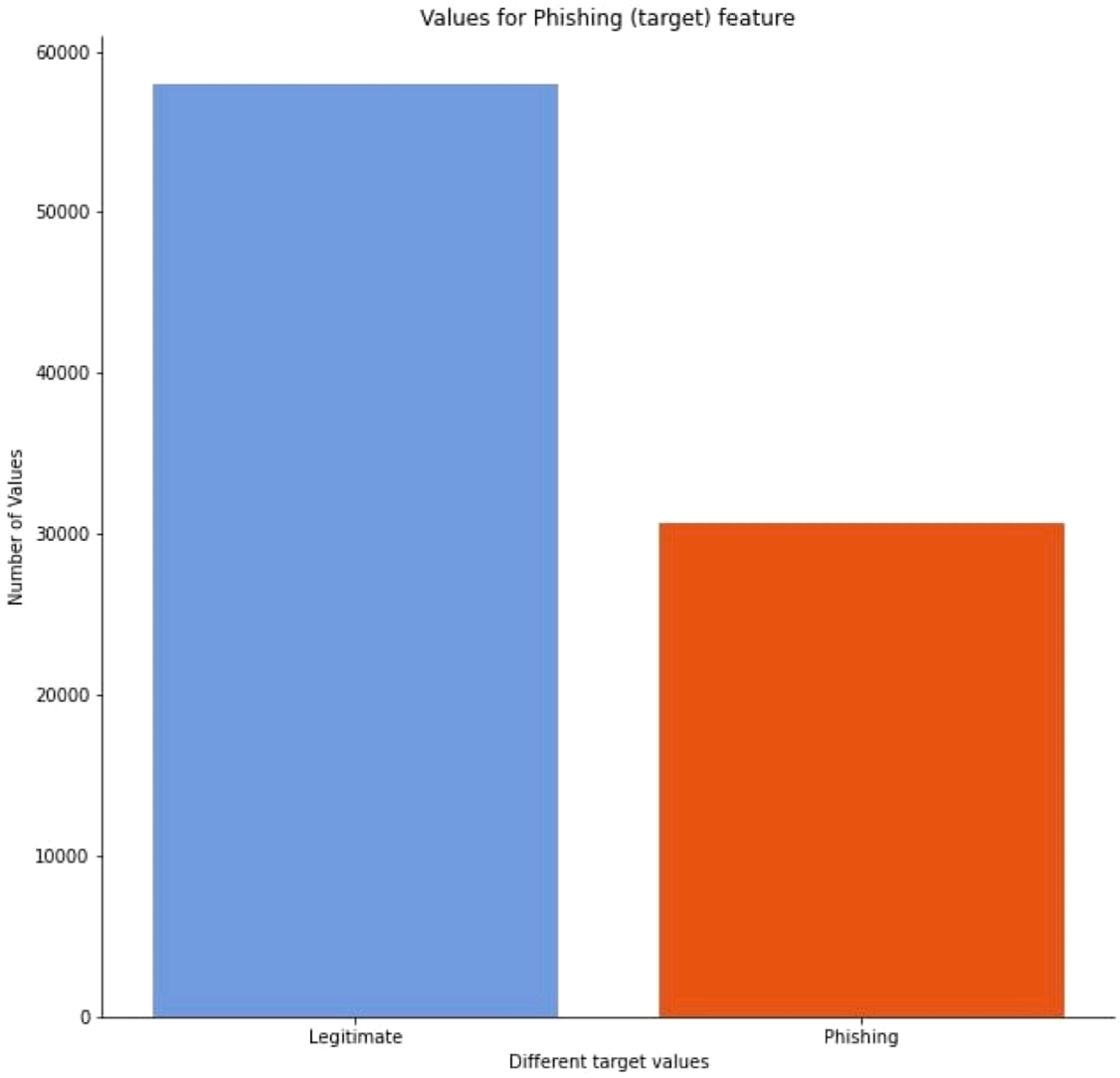


Fig. 3. The proportion of phishing and legitimate values.

Table 1
List of features in the dataset (part 1) [27].

qty_dot_url	qty_dollar_url	domain_spf
qty_hyphen_url	qty_percent_url	tls_ssl_certificate
qty_underline_url	qty_tld_url	qty_exclamation_domain
qty_slash_url	qty_dollar_file	qty_space_domain
qty_questionmark_url	qty_equal_params	qty_tilde_domain
qty_equal_url	qty_asterisk_params	qty_comma_domain
qty_at_url	time_response	qty_plus_domain
qty_and_url	ttl_hostname	qty_asterisk_domain
qty_exclamation_url	length_url	qty_hashtag_domain
qty_hashtag_file	qty_dot_domain	qty_dollar_domain
qty_questionmark_params	qty_hyphen_domain	qty_percent_domain
qty_plus_params	qty_underline_domain	file_length
email_in_url	qty_slash_domain	qty_and_params
qty_mx_servers	qty_questionmark_domain	qty_dollar_params
qty_space_url	qty_equal_domain	asn_ip
qty_tilde_url	qty_at_domain	qty_redirects
qty_comma_url	qty_and_domain	qty_vowels_domain
qty_plus_url	qty_percent_file	domain_length
qty_asterisk_url	qty_at_params	domain_in_ip
qty_hashtag_url	qty_hashtag_params	server_client_domain

Table 2
List of features in the dataset (part 2) [27].

qty_slash_directory	qty_space_params	url_shortened
qty_questionmark_directory	params_length	qty_equal_file
qty_dot_params	time_domain_expiration	qty_at_file
qty_exclamation_params	domain_google_index	qty_and_file
qty_percent_params	qty_hashtag_directory	qty_exclamation_file
time_domain_activation	qty_dollar_directory	qty_space_file
url_google_index	qty_percent_directory	qty_tilde_file
qty_equal_directory	directory_length	qty_comma_file
qty_at_directory	qty_dot_file	qty_plus_file
qty_and_directory	qty_hyphen_file	qty_asterisk_file
qty_exclamation_directory	qty_underline_file	qty_slash_params
qty_space_directory	qty_slash_file	qty_comma_params
qty_tilde_directory	qty_questionmark_file	qty_params
qty_comma_directory	qty_underline_params	qty_nameservers
qty_plus_directory	qty_tilde_params	phishing
qty_asterisk_directory	tld_present_params	
qty_hyphen_params	qty_ip_resolved	

detection, and medical diagnostics/monitoring [43].

Class imbalance issues can be addressed in several ways, which have been previously proposed at the data and algorithm level. At the former stage, these solutions involve various forms of resampling. Directed undersampling (reinforming the selection of samples to remove), oversampling by generating new samples based on information, and combinations of the above techniques [43]. SMOTEENN, a combination of oversampling and undersampling, is selected in our study out of the

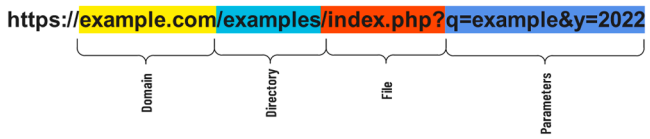


Fig. 4. The Structure of the URL.

mentioned techniques. Tomek's link and edited nearest neighbors are two cleansing ways added to the pipeline after applying SMOTE oversampling to get a cleaner space. Two out-of-the-box imbalance learning classes for merging undersampling and oversampling approaches are SMOTETomek and SMOTEENN [44].

Oversampling the minority class samples with SMOTEmek can level out the class distribution, but it does not solve the problem usually seen in datasets with skewed class distributions. The minority class space might be plagued by some majority class examples, making class clusters challenging to define. Additionally, intercalating minority class examples can extend these class clusters and introduce artificial minority ones deeper into the majority class space. Subsequently, overfitting takes place when classifiers are induced in such situations. Applying Tomek links to an oversampled training set has been proposed as a data-cleansing strategy to provide better-defined class clusters. As a result, examples from both classes are eliminated rather than just the majority class examples contributing to Tomek links. The following mathematical expression can be used to express it.

Let $d(x_i,x_j)$ be the Euclidean distance between x_i and x_j, where x_i represents the minority sample, and x_j represents the majority sample. In the absence of a sample, x_k fulfills the following condition [39] $d(x_i,x_j) > d(x_i,x_k)$ and $d(x_i,x_j) > d(x_j,x_k)$. The SMOTEmek links approach was first taken into consideration to enhance the classification of example protein annotation problems in Bioinformatics [44]. This method's rationale is similar to that of SMOTETomek links. Since Edited Nearest Neighbor (ENN) often eliminates more instances than Tomek links do, it is anticipated that it offers a more thorough data cleansing [40]. ENN is used to eliminate cases from both classes, in contrast to Neighborhood Cleaning Rule (NCL) [45] which is an undersampling technique [44]. SMOTEENN also tends to clean noisier samples than SMOTETomek [44, 46].

3.3.2. Dropping constant features

Data science allows us to deal curiously with different features in our dataset. For example, selecting important features with RF, using tree-based algorithms such as Decision Trees, removing constant features, etc. Constant features have similar/single values in all observations. From an information theory point of view, there is no information in these features. Thus, they cannot help ML models to predict the target more accurately while imposing more processing time. Thresholding variance can be used to drop constant features, but to understand how it works, we must answer two questions: What are variance and its threshold? Standard deviation is the square root of the variance (S^2). Variance can be computed with Eq. (4):

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu)^2 \tag{4}$$

Another way to calculate the standard deviation is as follows:

$$\textit{Standard deviation } (\sigma) = \sqrt{S^2} \tag{5}$$

Using the variance threshold, all features of the dataset that have low variance and are not crucial to the modeling process are removed (zero by default). As demonstrated in the following section, these features do not only add no improvements to the model but actually decrease its run-time.

3.4. Algorithms description

In this article, we perform a performance comparison of six different classification algorithms (including LR, KNN, NB, RF, SVM, XGBoost) to implement a high accuracy phishing website detection method based on machine learning.

3.4.1. Logistic regression (LR)

Although logistic regression is referred to as a regression model, it is a linear classification. In this model, the probabilities denoting the possible results of a single trial are modeled using the logistic function. As a type of sigmoidal curve (sigmoid curve), the logistic function or logistic curve follows the following formula:

$$f(x) = \frac{L}{1+e^{-k(x-x_0)}} \tag{6}$$

Where x_0 is the initial given value, x is the value of the sigmoid midpoint; L, the curve's maximum value, and k, the logistic regression growth rate of steepness of the curve. Moreover, assume $X_1,\ldots, X_p$ represent the random variables thought to be elucidating variables and let Y represent the binary response variable of interest. LR uses below formula to relate $P(Y=1|X_1,\ldots, X_p)$ to $X_1,X_2,\ldots,$ and X_p.

$$P\left(Y=1|X_1,\ldots,X_p\right) = \frac{exp\left(\beta_0+\beta_1X_1+\ldots+\beta_pX_p\right)}{1+exp\left(\beta_0+\beta_1X_1+\ldots+\beta_pX_p\right)} \tag{7}$$

Where $\beta_0,\beta_1,\ldots, \beta_p$ are regression coefficients that were calculated using the dataset under consideration and maximum likelihood. The probability that $Y=1$ for the new instance is calculated by substituting β with the calculated corresponding value and X with the realization of the new instance under consideration. The new instance is then assigned to class $Y=1$ or $Y=0$ under these circumstances:

$$Y=1|P(Y=1)\rangle c \tag{8}$$

$$Y=0|P(Y=1)\leq c \tag{9}$$

Where c is a definite threshold.

Like other model-based procedures, the effectiveness of LR's prediction depends on whether the data conforms to the model being used [47]. LR's "predict_proba" method predicts the likelihood of a positive outcome once fitted. The following cost function is minimized by binary class logistic regression with regularization term $r(w)$ [48].

$$minC\sum_{i=1}^{n}(-y_i log(\widehat{p}(X_i)) - (1-y_i)log(1-\widehat{p}(X_i))) + r(w) \tag{10}$$

According to the Table 3, there are various options for the term $r(w)$ of regularizations via the penalty argument provided by scikit-learn [48]:

3.4.2. K-Nearest neighbors (KNN)

K-Nearest Neighbor (KNN) algorithm is a straightforward instance-based supervised ML algorithm for multi-class problems. In categorizing, the distance between a fresh sample and its neighbor is used. By doing so, the K-nearest neighbors of each training set are found, and an item is assigned to the class with the most members (See Fig. 6). The KNN technique is an instance-based learning technique that does not

Table 3
Different options for the penalty argument.

Penalty	$r(w)$
None	0
$\mathscr{L}_1$	$\|w\|_1$
$\mathscr{L}_2$	$\frac{1}{2}\| w \|_2^2 = \frac{1}{2}w^Tw$
ElasticNet	$\frac{1-\rho}{2}w^Tw + \rho\| w \|_1$

presuppose any presumptions about the distribution of the data underlying the model Fig. 5 [49].

The KNN method relies on the approach used to determine how far a new instance is from existing ones. By default, KNN uses Euclidean distance, which can be computed using the equation below [50,51].

$$D^{Euclidean}(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \ldots + (p_n - q_n)^2} \tag{11}$$

where p and q are subjects to compare with n features. Various other methods are also available to calculate distance, including Manhattan distances, Minkowski distances, Jaccard distances, etc. In a plane with p_1 at (x_1,y_1) and p_2 at (x_2,y_2). Manhattan distance be calculated as:

$$D^{Manhattan}(x,y) = |x_1 - x_2| + |y_1 - y_2| \tag{12}$$

The Minkowski distance of order p (where p is an integer) is estimated:

$$D^{Minkowski}(x,y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}} \tag{13}$$

In order to calculate Jaccard distance, there is a formula, in which A and B are two points that we intend to calculate the distance between them.

$$D^{Jaccard}(A,B) = \frac{|A \cap B|}{A \cup B} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{14}$$

The parameter k in the KNN algorithm determines the number of neighbors that should be chosen. The proper choice of k greatly influences the diagnostic performance of the KNN algorithm. An enormous k minimizes the effect of variance brought on by random error, but it also boosts the chance of missing a diminutive-yet-critical pattern. Finding a balance between overfitting and underfitting is essential for choosing an appropriate k value [52]. The square root of the number of features is recommended as the value of constant k.

3.4.3. Naive Bayes (NB)

In the context of supervised learning algorithms, Naive Bayes approaches refer to a set of algorithms based on Bayes' theorem. The "Naive" presumption of independence, which is conditional, is between every pair of characteristics of a variable within a class of variables (See Fig. 7). Bayes' theorem indicates the upcoming equations for a variable of term y and a feature vector that depends x_1 to x_n:

$$P(y|x_1,\ldots,x_n) = \frac{P(y)P(x_1,\ldots,x_n|y)}{P(x_1,\ldots,x_n)} \tag{15}$$

Using the Naive conditional independence presumption of

$$P(x_i|y,x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n) = P(x_i|y) \tag{16}$$

For all i, Eq. (15) can be written simpler as:

$$P(y|x_1,\ldots,x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1,\ldots,x_n)} \tag{17}$$

Since $P(x_1,\ldots, x_n)$ is fixed with respect to the input, the following classification rule can be used:

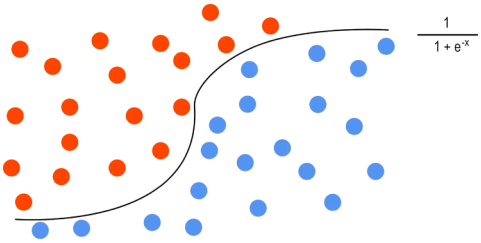


Fig. 5. Representation of Logistic Regression Classification.

$$P(y|x_1,\ldots,x_n) \propto P(y)\prod_{i=1}^{n} P(x_i|y) \tag{18}$$

Resulting in:

$$\hat{p} = argmax P(y)\prod_{i=1}^{n} P(x_i|y) \tag{19}$$

Maximum A Posteriori (MAP) calculation is used to estimate $P(y)$ and $P(x_i|y)$; $P(x_i|y)$ shows how many times class y observed in the training set. Also, the different Naive Bayes classifiers vary particularly in the presumption they make in terms of the distribution of $P(x_i|y)$. Separating the conditional feature distributions by class means that each distribution can be calculated separately as a one-way grouping, which helps reduce issues arising from the dimensional curse [53]. In this study, the Gaussian Naive Bayes algorithm has implemented by us. A Gaussian distribution is assumed for the likelihood of the features:

$$P(x_i|y) = \frac{1}{\sqrt{2\Pi\sigma_y^2}} exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2} \right) \tag{20}$$

3.4.4. Random forest (RF)

Random Forest (RF) is another chief machine learning method, and it consists of many averaged de-correlated trees or bagging. RF performs very similarly to boosting many problems and is simpler to train and tune. Consequently, RF is widely used and implemented in various software packages. Each tree in the RF depends on randomly sampled vector values, with the same distribution applied to all trees in the forest [54]. The generalization error goes towards a limit as the number of trees in the forest increases. The effectiveness of each tree and the correlations between them determine the forest generalization error of the tree classifiers. By averaging all the predictions of all the regression trees (B) on an unobserved example x' after training, predictions for an unobserved example x' can be made [55].

$$\hat{p} = \frac{1}{B}\sum_{b=1}^{B} f_\sigma(x') \tag{21}$$

Fig. 8 shows an RF classification of a new sample using majority voting. The primary advantage of RF is that it scales well to massive datasets, has excellent accuracy even when a substantial portion of the data is missing, and is a reliable method for forecasting missing data. Additionally, prediction uncertainty calculates can be made as the standard deviation of the predictions from all the individual regression trees on x'.

$$\sigma_\mu = \sqrt{\frac{\sum_{b=1}^{B}(f_\sigma(x' - \hat{f})^2}{B-1}} \tag{22}$$

3.4.5. Support vector machine (SVM)

Support Vector Machine (SVM) is one of the most prevalent methods of machine learning. SVM examines data for classifications and regressions as a supervised learning model with associated learning algorithms. Additionally, SVM can perform non-linear classification using an approach known as the kernel trick, which implicitly maps their inputs into high-dimensional feature spaces, and margin lines are drawn between classes (See Fig. 9). The classification error is minimized by drawing the margins that maximize the distance between the margin and the classes [56]. SVM takes advantage of some simple equations listed below to decide the boundary and measure distance.

$$H : w^T(x) + b = 0 \tag{23}$$

Here $b = Intercept$ and bias term of the hyperplane equation and in D-dimensional space, the hyperplane would always be $D - 1$ operator. The distance of a given point vector $\Phi(x_0)$ can be easily written as:

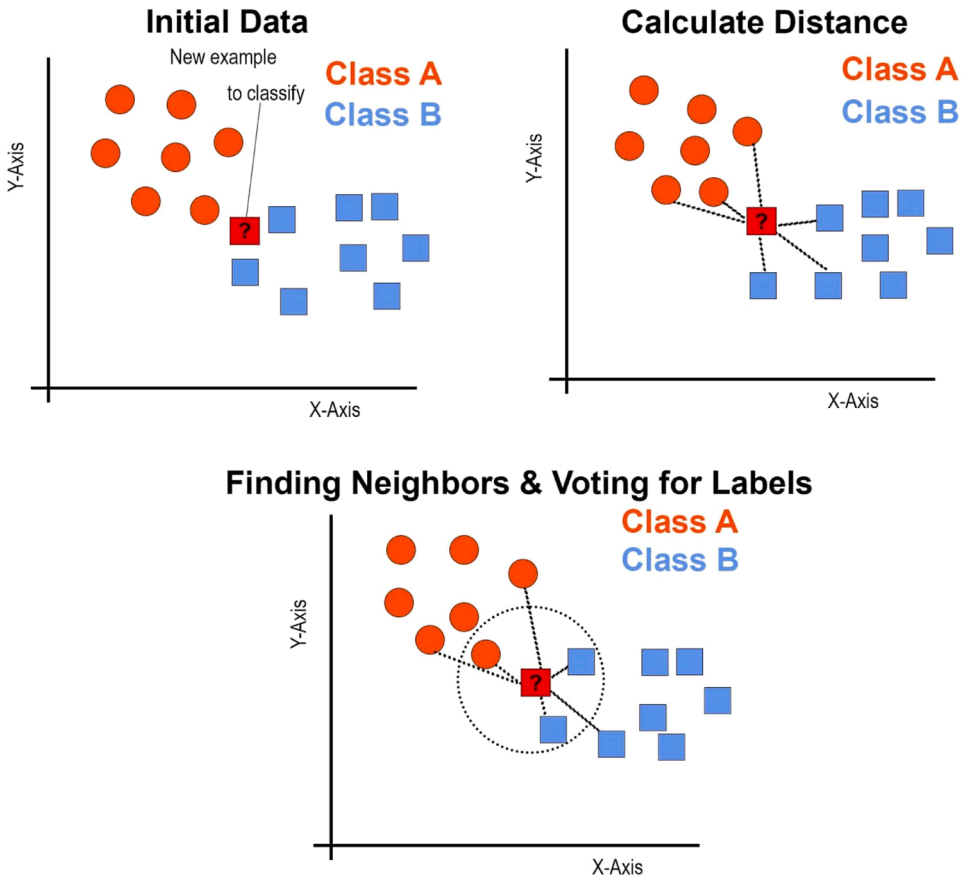


Fig. 6. Representation of K-NN Classification.

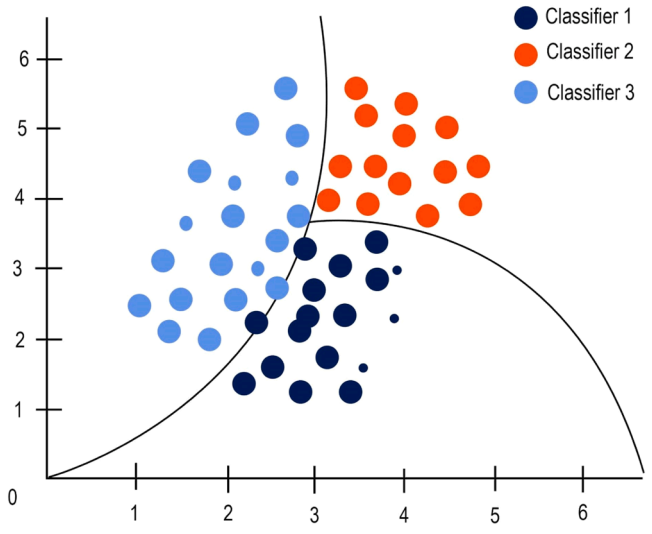


Fig. 7. Representation of Naive Bayes Classification.

$$d_H(\Phi(x_0)) = \frac{|w^T(\Phi(x_0)) + b|}{\| w \|_2} \tag{24}$$

here $\|w\|_2$ is the Euclidean norm for the length of w given by:

$$\| w \|_2 = \sqrt{w_1^2 + w_2^2 + \ldots + w_n^2} \tag{25}$$

There are several types of kernel functions available such as Linear (Eq. (26)), Polynomial (Eq. (27)), Rbf (Eq. (28)), and Sigmoid (Eq. (29)):

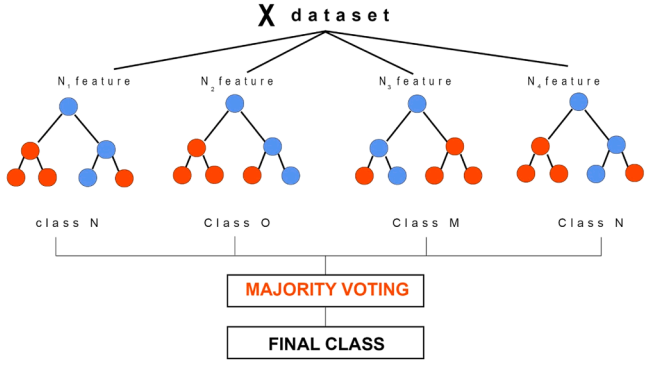


Fig. 8. Representation of Random Forest Classification.

$$K(x, x') = \langle x, x' \rangle \tag{26}$$

$$K(x, x') = (\Upsilon \langle x, x' \rangle + r)^d \tag{27}$$

Υ is the influence on classification outcomes, large value for Υ leads to overfitting while small value results in underfitting. For the "gamma" value in SVM, there are three choices; "scale", "auto", and float numbers. If "scale" is passed, then it uses $\frac{1}{n_{features} \times X_{var}}$. as value of gamma, if it equals to "auto", uses $\frac{1}{n_{features}}$. The degree parameter specifies d, controls the flexibility of the classifier result and means that the feature space has an inhomogeneous condition consisting of all monomials:

$$K(x, x') = exp\big(-\Upsilon \parallel x, x' \|^2\big) \tag{28}$$

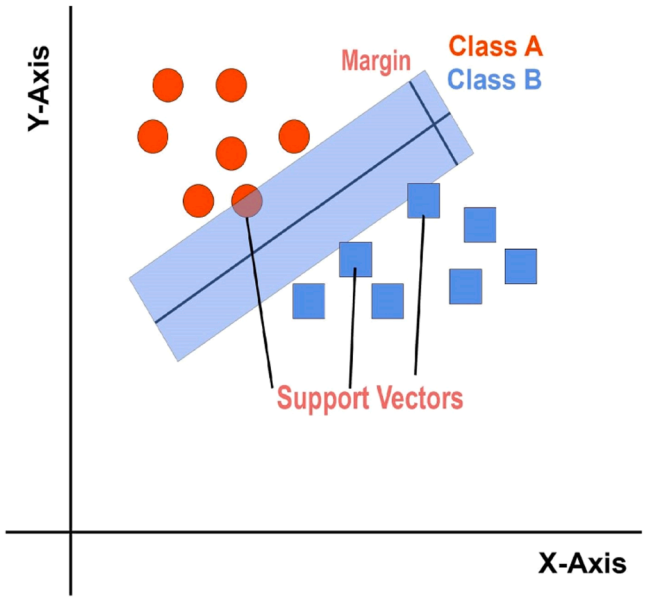


Fig. 9. Representation of Support Vector Machine (SVM) classification.

where ϒ is specified by parameter gamma, must be greater than 0.

$$K(x,x') = tanh(\Upsilon\langle x,x'\rangle + r) \tag{29}$$

3.4.6. XGBoost

The eXtreme Gradient Boosting is shortened to XGBoost. It is an expandable and valuable development of Friedman's gradient-boosting paradigm [57,58]. Sequential decision trees are created using this algorithm (See Fig. 10). XGBoost relies heavily on weights. All independent variables are weighted and entered into a decision tree that predicts outcomes. Then, the decision tree gives greater weight to mispredicted variables. Afterward, these individual classifiers/predictors are combined to give a more robust and accurate model [59]. with Eq. (30), XGBoost predicts the outcome as 0 or 1.

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) \tag{30}$$

In Gradient boosting for classification the sum of the trees is

$$F_M(x_i) = \sum_{m}^{i=1} h_m x_i \tag{31}$$

There is no homogeneity between Eq. (31) and a prediction: it cannot fall into a class because the trees predict continuous values. There is a loss-dependent mapping between value $F_M(x_i)$ and class or probability. Using the log-loss model, the probability that x_i belongs to the positive class is calculated as follows:

$$p(y_i = 1|x_i) = \sigma_F(F_M(x_i)) \tag{32}$$

Where σ_F is the sigmoid or expit function and even for classification tasks, the h_m sub-estimator is still a regressor, not a classifier. This is because the sub-estimator is trained to predict the (negative) slope, and we know that gradients are always of continuous magnitude [48]. Regularization strategies that scale the contribution of weak learners by v were proposed by [58]:

$$F_m(x) = F_{m-1}(x) + vh_m(x) \tag{33}$$

In gradient descent, v has the effect of scaling the step length; this parameter can be set using the eta parameter, which is one of the most crucial hyperparameters to tune.

3.5. Dimensionality reduction

In the scenario where the data objects consist of many attributes, it is often advantageous to reduce the dimension of the data (describe the data in fewer attributes) to improve the efficiency and accuracy of the data analysis process. In statistics, problems are sometimes referred to as "Big p, Small n;" (in a given dataset, the predictors or input features are called "p" while the number of rows in the dataset is "n "). In other words, these examples are extreme instances where dimension reduction (DR…) is vital since the number of explanatory variables p is far more than the number of samples n. In order to tackle the so-called curse of dimensionality, as part of the data analysis or as a pre-processing step. Data models are frequently simplified by dimension reduction techniques. As a result, it will be necessary to identify a low-dimensional representation that matches the original high-dimensional data set in order to achieve this task. Using a reduced representation, tasks such as classification and clustering can often be performed more accurately and more easily interpreted while also reducing the computational cost

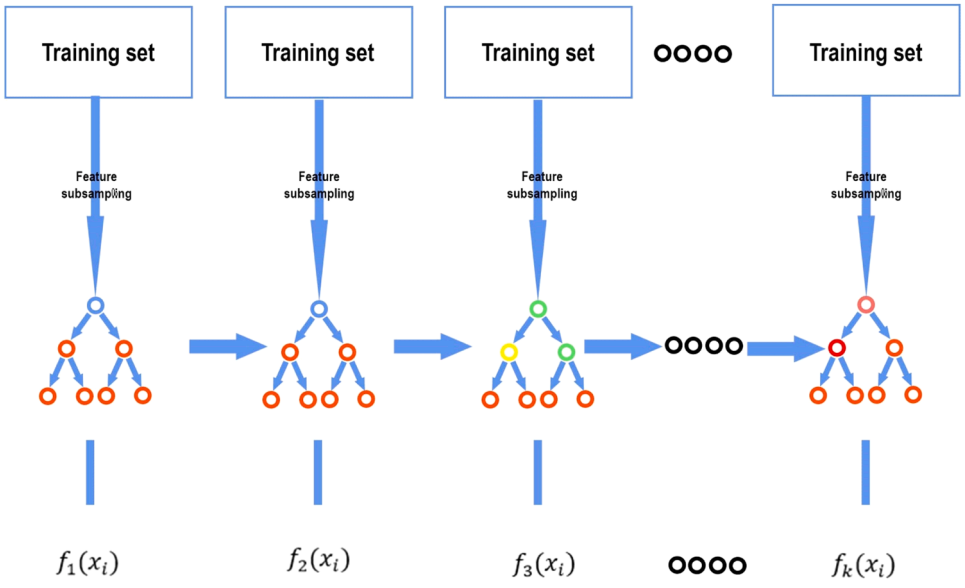


Fig. 10. Representation of XGBoost classification.

significantly [60].

There are several choices for the dimension reduction purpose, which depends on the desired model. We chose to work with Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) in our study.

3.5.1. Principal component analysis (PCA)

PCA is an unsupervised algorithm which can be used to reduce the number of features is supervised models. A key component of PCA is the covariance matrix of the data $C = \frac{1}{n-1}XX^T$. On the diagonal in C, the variance of each feature is captured, and on the off-diagonal terms, the variance between the corresponding pairs of features is quantified. PCA aims to eliminate all covariance terms from the data, for instance. C_{PCA} is produced by diagonalizing C. In this transformation, the rows of P represent the eigenvectors of XX^T, while the columns represent the rows of $Y = PX$.

$$C_PCA = \frac{1}{n-1}YY^T \Rightarrow \frac{1}{n-1}(PX)(PX)^T \tag{34}$$

C_{PCA} calculates variance by moving the i_th diagonal entry in the direction of the principal component corresponding to the data set. As a result of the reduction in dimension, the fewer principal components are discarded. The dimension of P is equal to $(k \times p)$ where k is the number of undiscarded principal components [60].

3.5.2. Linear discriminant analysis (LDA)

A dimensionality reduction technique called Linear Discriminant Analysis (LDA), Normal Discriminant Analysis (NDA), or Discriminant Function Analysis (DFA) is widely used to solve supervised classification problems. This technique is used to project features from a higher-dimensional space into a lower-dimensional one. LDA creates a new axis based on two criteria:

- Ensure that the distance between the two classes is as considerable as possible.
- Attributes found on the domain properties
- Reduce the variation within each class as much as possible

With the use of this newly generated axis, the separation between data points of the two classes can be increased. Using the aforementioned criteria, this new axis is generated, and all data points within the classes are projected onto it [60].

3.6. Performance evaluation

It is considered that the detection of phishing websites is a binary classification that reveals two possible outcomes for a particular website: it could be a phishing site or it could be legitimate. For such classification, there are typical metrics used in the literature to evaluate classifier performance, mainly accuracy, F1 score, precision, specificity, sensitivity, Jaccard index record, and Dice coefficient [29,61]. In these formulas, True Positives (TP) are the number of classified examples that belong to one of the correctly classified classes. TN or True Negatives are the number of samples that are categorized precisely and do not belong to either class. False positives (FP) are the number of samples incorrectly assigned to one of the classes or the number of samples assigned to a class that does not belong to it. Finally, FN stands for False Negatives, the number of samples classified because they belong to any class but not to that class [29].

$$Accuracy = \frac{T_P + T_N}{T_P + F_P + T_N + F_N} \tag{35}$$

$$Specificity = \frac{T_N}{T_N + F_P} \tag{36}$$

$$Sensitivity\ (Recall) = \frac{T_P}{T_P + F_N} \tag{37}$$

$$Precision = \frac{T_P}{T_P + F_P} \tag{38}$$

$$Jaccard\ index\ score = \frac{T_P}{T_P + F_P + F_N} \tag{39}$$

$$Dice\ coefficient = \frac{2 \times T_P}{2 \times (T_P + F_P + F_N} \tag{40}$$

$$F1 - score = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)} \tag{41}$$

To introduce Receiver Operating Characteristic Curve (ROC) and Area Under ROC Curve (AUC), we have to discuss True Positive Rate (TPR) and False Positive Rate (FPR). TPR and FPR can simply be calculated by the following equations.

$$TPR = \frac{TP}{P} \tag{42}$$

$$FPR = \frac{FP}{N} \tag{43}$$

A ROC curve is a plot of the performance of a classification model at every threshold at which it is tested. TPR and FPR are plotted on this curve. As we discussed earlier, AUC is the area under the ROC curve. This means that AUC measures the area beneath the entire ROC curve (think integral calculus) from (0,0) to (1,1). In addition to providing a measure of performance across all classification thresholds, AUC also provides a measure of consistency. AUC can be viewed as a measure of the probability that the model ranks a random positive example higher than a random negative example.

4. Experiments and results

In this work, we have targeted designing and implementing a high-accuracy ML-based PWD system. The public dataset collected by Vrbančič, G. [27] has been considered in our implementations. It has 88,647 instances and 112 attributes, which makes it remarkably steadfast. Our experiments were run on a Lenovo device with a 2.5 GHz Intel Core i7 processor and 16 GB DDR4 RAM. The python programming language is used with several pre-developed libraries to test the proposed PWD system. After performing SMOTEENN on the dataset to balance it, the number of non-phishing and phishing instances are shown in Table 4, and the number of instances in the imbalanced dataset.

As it has already been discussed, the next stage is to drop constant features. In our study, 13 constant features dropped, which do not change in any instances (the variance of those features was zero), as can be noticed in Table 5.

4.1. Data visualization

A general dataset consists of quantitative and qualitative data. Quantitative data has only numbers, while qualitative data consists of other kinds of information. Structured and unstructured qualitative data

Table 4
The distribution of phishing and legitimate instances in the imbalanced and the balanced dataset.

Label	No. of samples	
	Imbalanced dataset	Balanced dataset
Legitimate websites (labeled as 0)	58,000	46,728
Phishing websites (labeled as 1)	30,647	48,849
Total instances	88,647	95,577

Table 5
Constant features description.

Row	Feature	Description
1	qty_slash_domain	the number of (/) in the domain
2	qty_questionmark_domain	the number of (?) in the domain
3	qty_equal_domain	the number of (=) in the domain
4	qty_and_domain	the number of (&) in the domain
5	qty_exclamation_domain	the number of (!) in the domain
6	qty_space_domain	the number of (" ") in the domain
7	qty_tilde_domain	the number of (~) in the domain
8	qty_comma_domain	the number of (,) in the domain
9	qty_plus_domain	the number of (+) in the domain
10	qty_asterisk_domain	the number of (*) in the domain
11	qty_hashtag_domain	the number of (#) in the domain
12	qty_dollar_domain	the number of ($) in the domain
13	qty_percent_domain	the number of (%) in the domain

are classified according to how they are gathered, organized, kept, and presented. Similarly, there are two types of quantitative data: continuous and discrete. It is typically represented using decimal or floating-point numbers and can take any value within a range. In our study, as mentioned, all instances are numbers, integers, and floats. Extracting useful information from numbers is more challenging than other types, but it is indeed helpful.

Fig. 11a represents the attribute "time_response" in the dataset. It is the search time (response) domain (lookup) of websites considering the phishing feature (target feature), which is differentiated by different colors. It can be easily noticed that a vast majority of websites responded in less than ten milliseconds, regardless of being phishing or legitimate. There are several instances (5860 out of 88,647) in which the response time and the domain activation time are less than zero, which means that these websites records could not be achieved, and -1 is replaced [27].

Fig. 11b and c show the distributions of domain activation time and domain expiration time ('time_domain_activation', 'time_domain_expiration'), respectively. The 'time_domain_activation' attribute is the time of domain activation (in days), and the 'time_domain_expiration' is the time left until the domain expires (in days), which are two most significant contributors to detecting phishing websites, along with response time.

The following two plots (Fig. 12a and b) show the relation between the number of resolved IPs and the time-to-live (TTL) value associated with the hostname for the phishing (target) feature, respectively. There are some outliers, which we handle to feed our algorithms, but in general, it can be seen that legitimate websites have higher values for both resolved IP and TTL. On the contrary, phishing instances have much higher density in smaller values.

4.2. Examined scenarios

In our study, various algorithms, including LR, KNN, NB, RF, SVM, and XGBoost, are examined in different scenarios depending on a balanced/imbalanced dataset, dropping/not dropping constant features, and with/without dimension reduction. According to Table 6, this can lead to eight distinct scenarios in our experiments:

For the dimension reduction phase, we decided to examine the algorithms with PCA and LDA for ease of use and, more importantly, faster run-time. The obtained results indicate that PCA outperforms LDA and the results which are used in the following tables are from the PCA method. The overall accuracy for KNN and SVM are shown for all eight scenarios in Table 7 and Table 8, respectively:

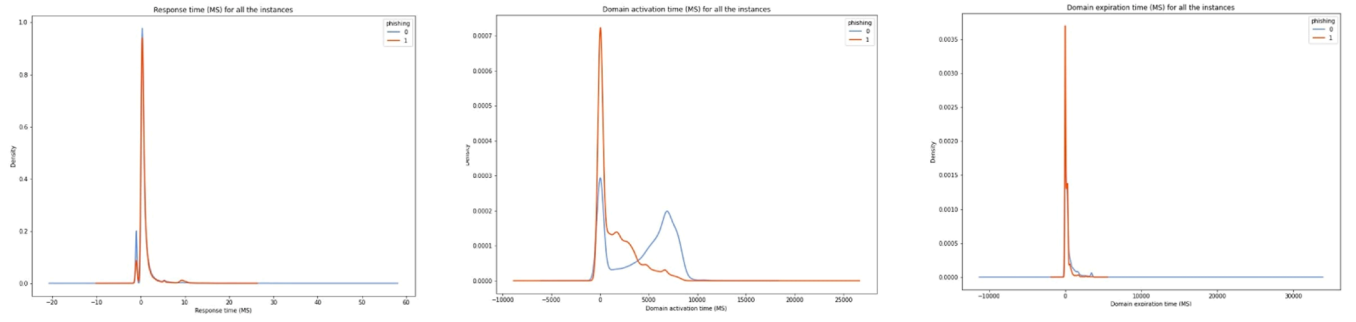
This trend is similar for all other classifiers in scenarios S5, S6, S7, and S8, which is a solid proof to deal only with the original dataset without any dimension reduction. The fixed conditions for all algorithms are the training and test set sizes (80% and 20%, respectively) and ten-fold cross-validation.

We train the classifiers with default hyperparameters as a base evaluation for further comparison. Every algorithm has several influential hyperparameters that, when they are tuned efficiently, lead to higher accuracy. With this in mind, for each algorithm, we set all but one hyperparameter fixed in each study to find the optimal value. We repeat this procedure to find the optimal value for all hyperparameters. Even though all these hyperparameters and their values are important, several of them are much more essential to increase the accuracy of classifiers, and we choose the most crucial hyperparameters which play the most vital roles in achieving the highest accuracy. Table 9 illustrates the results of hyperparameter tuning in our study for each algorithm.

Then Tables 10–13 shows the performance of all algorithms after hyperparameter tuning for the first four scenarios regarding the accuracy, precision, recall, F1 score, specificity along with TP, TN, FP, and FN values.

Fig. 13 is derived from Tables 10–13. It indicates that the best scenario is the fourth scenario (S4), when the dataset is balanced, the constant features are dropped, and dimension reduction is ignored. According to Fig. 13, the XGBoost outperforms all other algorithms with the Accuracy of about 99.2% in Scenario S4.

We also tested the validity of the fourth scenario (S4) (the best

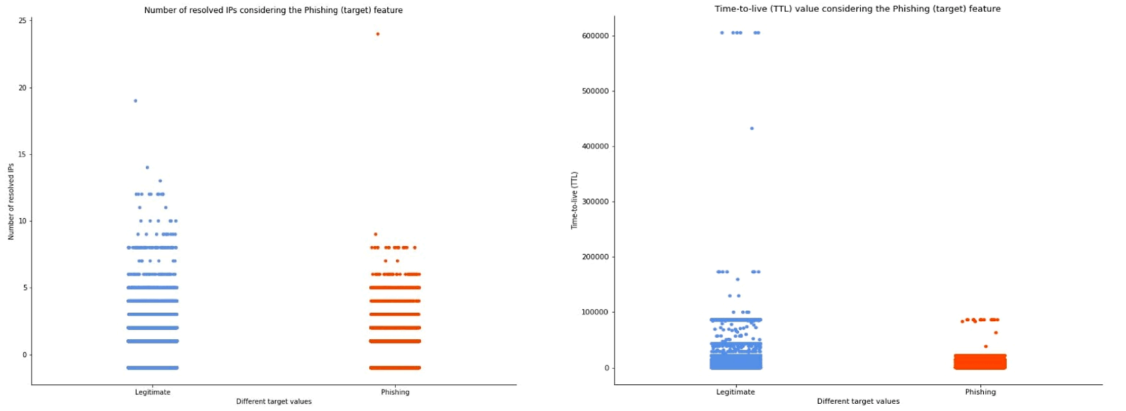


(a) Response time for the Phishing (target) feature.

(b) Domain activation time (in ms) considering the Phishing (target) feature.

(c) Domain expiration time (in ms) considering the Phishing (target) feature.

Fig. 11. Response time, Domain activation, and the expiration time for the Phishing feature.



(a) Number of resolved IPs considering the Phishing (target) feature.

(b) time-to-live (TTL) considering the Phishing (target) feature.

Fig. 12. Number of resolved IPs and time to live (TTL) considering the Phishing feature.

Table 6
Examined scenarios in our study. BDB: Balanced Database, DCF: Drop Constant Features, and DR…: Dimension Reduction.

Scenarios	BDB	DCF	…
S1	×	×	×
S2	×	✔	×
S3	✔	×	×
S4	✔	✔	×
S5	×	×	✔
S6	×	✔	✔
S7	✔	×	✔
S8	✔	✔	✔

scenario to work with) with ROC_AUC score in Table 14. The acquired results indicate that our model is absolutely reliable in detecting phishing websites.

4.3. Confusion matrix for all the algorithms

We are partial to discussing TP, TN, FP, and FN more. In this regard, the confusion matrix (CM) has depicted in Fig. 14 for all six algorithms but only for the best scenario (Scenario S4). The CM is a valuable matrix used in many analyses to evaluate the performance of a given classifier for an available test set. The results show that XGBoost in the fourth scenario not only can hit the highest record of accuracy but also it can reach a minimum FP compared to other algorithms. The FP is a fundamental index in the phishing website detection (PWD) problem. If a legitimate site is considered a phishing one by an AI-based software, from a legal perspective, it can lead to judicial complaint consequences. On the other side, if a phishing website is introduced as an ordinary website, it can be tolerated somehow as a machine error, and the owners (criminals) will certainly not be dissatisfied with this wrong result. In the following figures, the Confusion Matrix for all the algorithms for the best scenario (S4) have been depicted.

Regarding classifiers, the best results were evidently obtained by XGBoost compared to other algorithms. However, all the algorithms perform outstandingly, with accuracies of more than 93% for each, which has shown significant improvements compared to other works. In addition, Table 14 demonstrates a deeper comparison between all the algorithms with the report of overall accuracy, precision, recall, F1-score, and specificity in percentage.

As mentioned earlier, we examined all six algorithms in eight different scenarios and proved that all the algorithms perform better under the fourth scenario by comparing the results of KNN. In Table 15, we compare our best model's accuracy in all scenarios for XGBoost. After hyperparameters tuning and examining deeply, it has been revealed that XGBoost performs best out of those algorithms in this study.

Table 9
Results of hyperparameters tuning for all the algorithms.

Algorithms	Hyperparameter tuning results
Logistic Regression	$penalty = none$, $C = 100$, $solver = newton - cg$
KNN	$metric = manhattan$, $n_{neighbors}(K) = 1$
Naive Bayes	$var_{smoothing} = 0.000132195$
Random Forest	$criterion = Gini$, $max_{features} = sqrt$, $n_{estimator} = 1000$
SVM	$kernel = Rbf$, $C = 100$
XGBoost	$eta = 0.5518$

Table 7
Comparing the overall accuracy of KNN algorithm in all simulation scenarios before hyperparameter tuning.

	Without dimension reduction				With dimension reduction (PCA)			
Scenarios	S1	S2	S3	S4	S5	S6	S7	S8
Accuracy (%)	95.350	95.356	96.393	**97.702**	95.290	95.291	96.242	96.243

Table 8
Comparing the overall accuracy of the SVM algorithm in all simulation scenarios before hyperparameter tuning.

	Without dimension reduction				With dimension reduction (PCA)			
Scenarios	S1	S2	S3	S4	S5	S6	S7	S8
Accuracy (%)	94.423	94.423	95.21	**95.275**	94.145	94.140	94.939	94.940

Table 10
Performance comparison (%) of all the algorithms after hyperparameter tuning for the first scenario (S1).

	Accuracy	Precision	Recall	F1-score	Specificity	TP	TN	FP	FN
LR	93.16	89.07	91.5	90.50	94.05	5606	10,910	690	524
KNN	95.36	93.45	93.08	93.26	96.55	5706	11,200	400	424
NB	78.62	88.23	44.02	59.01	96.89	2699	11,240	360	3431
RF	97.10	95.51	95.97	95.74	97.62	5883	11,324	276	247
SVM	94.43	91.33	92.67	91.99	95.35	5681	11,061	539	449
XGBoost	96.93	95.56	93.13	95.52	95.35	5854	11,328	272	276

Table 11
Performance comparison (%) of all the algorithms after hyperparameter tuning for the second scenario (S2).

	Accuracy	Precision	Recall	F1-score	Specificity	TP	TN	FP	FN
LR	93.17	89.09	91.52	90.52	94.07	5606	10,910	690	524
KNN	95.40	93.47	93.10	93.28	96.56	5706	11,200	400	424
NB	97.12	88.23	44.02	59.02	96.89	2699	11,240	360	3431
RF	97.12	95.52	95.96	95.72	97.63	5882	11,326	274	248
SVM	94.49	91.35	92.67	92.01	95.35	5681	11,061	539	449
XGBoost	96.93	95.57	93.14	95.54	95.36	5854	11,328	272	276

Table 12
Performance comparison (%) of all the algorithms after hyperparameter tuning for the third scenario (S3).

	Accuracy	Precision	Recall	F1-score	Specificity	TP	TN	FP	FN
LR	93.95	92.50	95.38	93.93	92.13	10,833	10,443	880	490
KNN	96.40	95.71	97.14	96.44	95.64	11,000	10,829	494	323
NB	70.18	93.31	43.46	59.30	96.84	4920	10,968	355	6403
RF	97.95	97.51	98.41	98.00	97.48	11,142	11,038	285	181
SVM	95.28	95.43	96.25	95.34	94.30	10,898	10,678	645	425
XGBoost	97.81	97.60	98.02	97.84	94.31	11,098	11,046	277	225

Table 13
Performance comparison (%) of all the algorithms after hyperparameter tuning for the fourth scenario (S4).

	Accuracy	Precision	Recall	F1-score	Specificity	TP	TN	FP	FN
LR	95.3	94.1	96.8	95.5	93.7	9466	8759	587	304
KNN	98.3	97.9	98.7	98.3	97.8	9652	9145	201	118
NB	93.0	90.9	96.1	93.4	89.9	9388	8404	942	382
RF	98.9	98.5	99.3	98.9	98.4	9707	9200	146	63
SVM	98.3	98.2	98.5	98.3	98.1	9617	9142	204	153
XGBoost	99.2	99.1	99.4	99.2	99.1	9708	9260	86	62

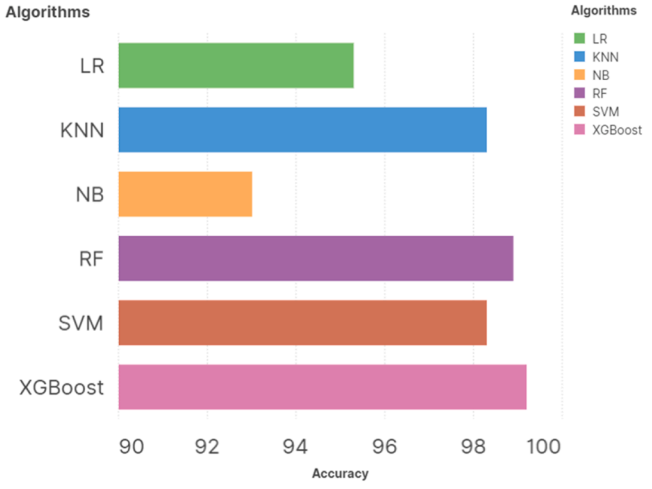


Fig. 13. Accuracy (%) of all algorithms for the best scenario (S4), after hyperparameter tuning.

Table 14
ROC_AUC score for the fourth scenario (S4) after hyperparameter tuning.

Algorithms	LR	KNN	NB	RF	SVM	XGBoost
ROC_AUC score (%)	95.30	98.32	93.01	98.91	98.13	99.21

Regarding the prediction runtime of the algorithms, we made a comparison between the fourth and the eighth scenarios in Table 16. Considering the results of our developed model, which shows its ability to detect phishing websites in around 1500 milliseconds (1.5 s), we can conclude that it is a real time model. Therefore, it can be used to detect online phishing in real-world situations with absolutely reliable results. Further, in many cases, blacklists of phishing websites are created using these kinds of models. Once this list has been compiled, it can be fed into various search engines and sources so that they can be blocked. Due to its high accuracy of 99.2%, our model is also suitable for this purpose, so it can be used in real-time or offline and is suitable for a wide range of applications.

As it is clear from Table 16, as a result of dimension reduction, algorithms run more quickly, and it is perfectly aligned with what we know about this phenomenon. It is important to note that even though the runtime in S8 is shorter in comparison to the fourth scenario, the

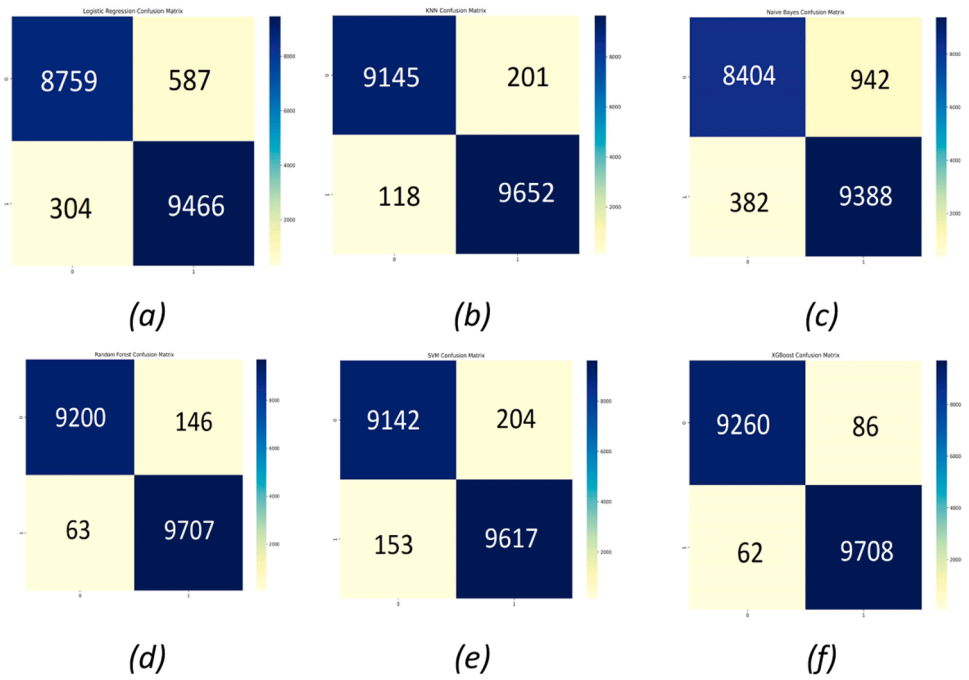


Fig. 14. Confusion matrix for all classification algorithms.
(a): LR, (b): KNN, (c): NB, (d): RF, (e): SVM, (f): XGBoost.

Table 15
Results of XGBoost in examined scenarios after hyperparameter tuning.

Scenarios	S1	S2	S3	S4	S5	S6	S7	S8
Accuracy (%)	96.9	96.9	97.8	99.2	96.1	96.1	97.1	98.5

Table 16
Comparing the runtime of algorithms for the fourth and the eighth scenarios (S4 and S8) in seconds (s).

	Run time (s)	
Algorithms	S4	S8
LR	1.1326	0.541
KNN	3.8577	1.294
NB	1.2891	0.630
RF	1.9800	1.230
SVM	6.9947	3.070
XGBoost	1.571	0.869

accuracy is relatively high, and in phishing detection studies, having an accurate model is of utmost importance. S8's runtime does not appear to be significantly lower than S4's. As a result, it is well appropriate to select S4 as a method to deal with phishing attacks.

For a machine learning model to identify fishing websites, the run time may be critical depending on the application and environment. In some circumstances, such as real-time monitoring of web traffic for potentially malicious activity, it may be essential to have a model that can generate predictions quickly and in real-time. However, when analyzing historical data in order to identify patterns and trends, the runtime of the model may not be as important. It may not be crucial to reduce the run time for identifying fishing websites when the model is used offline, such as for analyzing historical data or executing batch processing. In order for the model to be able to keep up with the incoming data streams and make accurate predictions, run time must be minimized if the model is to be used for real-time traffic monitoring. Considering the real-time runtime, our developed model has an overall runtime of 0.869 (for XGBoost and S8), 0.630 s (for NB algorithm and S8), and 0.541 s (for LR algorithm and S8), which makes it completely useful for real-world situations and real-time systems.

Table 17
Performance comparison with other studies.

Study	Accuracy (%)
Our method	99.22
[9]	98.42
[34]	90.36
[32]	96.87
[24]	94.6
[10]	97.98

Moreover, based on what has been discussed throughout the study, there is a combination of theoretical and experimental analysis in our work. In terms of experimental analysis, models were developed in the experimental analysis area since we experimentally concluded that algorithms performed well and examined the relationships between each instance. A further example of experimental analysis would be choosing the best algorithm, balancing the dataset, and removing constant features from the dataset. In terms of theoretical analysis, as discussed in our work, XGBoost outperformed other algorithms. XGBoost, which has outperformed other algorithms in this study, was developed through theoretical analysis to prove its performance and development [62]. Therefore, our work is theoretical in nature.

Finally, we made a comparison with other studies in the literature. As it can be seen in Table 17, our research outperforms the vast majority of other studies. The highest accuracy has been achieved while we examined six different algorithms in eight unique scenarios, making it an unprecedented and comprehensive study.

5. Conclusion and future work

Phishing attacks rise as everything goes through the internet, such as banking systems and e-commerce. The industry and Internet users suffer enormous economic losses as a result. As such, developing practical solutions that provide high accuracy and fast response times when

detecting phishing attacks is necessary [63,64].

As everything moves online, such as banking systems and e-commerce, phishing attacks are on the rise. A significant amount of economic damage is caused to the industry and to Internet users as a result. Thus, it is necessary to develop practical solutions that provide high levels of accuracy and fast response times when detecting phishing attacks.

The purpose of this study was to examine a relatively large dataset with multiple features that were capable of identifying phishing websites. These features are derived from hyperlink information provided in the source code of the website, along with other information gathered from multiple sources. On this dataset, which contains 88,647 instances, 58,000 legitimate, and 30,647 phishing instances, along with 111 features with labeled output, we have trained classification algorithms such as LR, KNN, NB, RF, SVM, and XGBoost. A number of constant attributes were dropped in this study in order to reduce run-time and improve the accuracy of the model. Additionally, SMOTEENN has been used to balance the dataset so that the highest level of accuracy can be achieved. As a result of balancing the dataset, 95,577 instances were identified, of which 46,728 are legitimate and 48,849 are phishing.

On a very large and high-dimensional dataset, we have conducted our experiment in a considerable amount of space, including eight different scenarios covering all six algorithms. As a result, the models are highly efficient, with an accuracy of more than 93%, which is unique among the related works. In our simulations, the XGBoost classifier outperformed the other classifiers in the classification of phishing websites, with 99.1% precision, 99.4% recall, 99.2% F1 score, 99.1% specificity, and 99.22% overall accuracy rate. In addition, the dataset has been balanced, and all constant features have been removed. According to the results, neither Principal Component Analysis (PCA) nor Linear Discriminant Analysis (LDA) was able to enhance the overall accuracy of the model. With its high accuracy and low error rates, along with fast runtime of around 1.5 s in S4 and 0.86 s in S8, the proposed approach offers an exceptional phishing website detection strategy.

However, there might be minor limitations in our work that we are planning to consider the following ideas to cover in our future works: (i) increasing the number of instances, especially phishing ones, in the next datasets (ii) implementing a variety of deep learning-based algorithms to examine the differences and obtain even higher accuracy, (iii) allowing the model to be used in real-time while reducing its run time.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

References

[1] Badotra S, Sundas A. A systematic review on the security of E-commerce systems. Int J Appl Sci Eng 2021;18(2):1–9.
[2] Ansari MF, Sharma PK, Dash B. Prevention of phishing attacks using AI-based cybersecurity awareness training. Prevention 2022.
[3] Li J, Sun L, Yan Q, Li Z, Srisa-An W, Ye H. Significant permission identification for machine-learning-based android malware detection. IEEE Trans Ind Inf 2018;14(7):3216–25.
[4] Bhuiyan MZ, Wu J, Wang G, Cao J. Sensing and decision making in cyber-physical systems: the case of structural event monitoring. IEEE Trans Ind Inf 2016;12(6):2103–14.
[5] Bhardwaj A, Al-Turjman F, Sapra V, Kumar M, Stephan T. Privacy-aware detection framework to mitigate new-age phishing attacks. Comput Electr Eng 2021;96:107546.
[6] apwg.org/trendsreports 2022.
[7] Adewole KS, Akintola AG, Salihu SA, Faruk N, Jimoh RG. Hybrid rule-based model for phishing URLs detection. In: Proceedings of the international conference for emerging technologies in computing. Springer; 2019. p. 119–35.
[8] Babagoli M, Aghababa MP, Solouk V. Heuristic nonlinear regression strategy for detecting phishing websites. Soft Comput 2019;23(12):4315–27.
[9] Jain AK, Gupta BB. A machine learning based approach for phishing detection using hyperlinks information. J Ambient Intell Humaniz Comput 2019;10(5):2015–28.
[10] Sahingoz OK, Buber E, Demir O, Diri B. Machine learning based phishing detection from URLs. Expert Syst Appl 2019;117:345–57.
[11] Alshehri M, Abugabah A, Algarni A, Almotairi S. Character-level word encoding deep learning model for combating cyber threats in phishing URL detection. Comput Electr Eng 2022;100:107868.
[12] Almomani A, Gupta BB, Atawneh S, Meulenberg A, Almomani E. A survey of phishing email filtering techniques. IEEE Commun Surv Tutor 2013;15(4):2070–90.
[13] Ghorbani M, Bahaghighat M, Xin Q, Özen F. ConvLSTMConv network: a deep learning approach for sentiment analysis in cloud computing. J Cloud Comput 2020;9(1):1–2.
[14] Hajikarimi A, Bahaghighat M. Optimum outlier detection in internet of things industries using autoencoder. in: frontiers in nature-inspired industrial optimization. Springer; 2022. p. 77–92.
[15] Khorasani F, Zanjireh MM, Bahaghighat M, Xin Q. A tradeoff between accuracy and speed for K-means seed determination. Comput Syst Sci Eng 2022;40(3):1085–98.
[16] Rostami M, Bahaghighat M, Zanjireh MM. Bitcoin daily close price prediction using optimized grid search method. Acta Univ Sapientiae Inform 2021;13(2):265–87.
[17] Bahaghighat M, Abedini F, S'hoyan M, Molnar AJ. Vision inspection of bottle caps in drink factories using convolutional neural networks. In: Proceedings of the IEEE 15th international conference on intelligent computer communication and processing (ICCP). IEEE; 2019. p. 381–5.
[18] Bahaghighat M, Abedini F, Xin Q, Zanjireh MM, Mirjalili S. Using machine learning and computer vision to estimate the angular velocity of wind turbines in smart grids remotely. Energy Rep 2021;7:8561–76.
[19] Shamseen A, Zanjireh MM, Bahaghighat M, Xin Q. Developing a parallel classifier for mining in big data sets. IIUM Eng J 2021;22(2):119–34.
[20] phishtank.org 2022.
[21] whois.com 2022.
[22] Zamir A, Khan HU, Iqbal T, Yousaf N, Aslam F, Anjum A, Hamdani M. Phishing web site detection using diverse machine learning algorithms. Electron Lib 2020;38(1):65–80.
[23] Rao RS, Pais AR. Detection of phishing websites using an efficient feature-based machine learning framework. Neural Comput Appl 2019;31(8):3851–73.
[24] Chiew KL, Tan CL, Wong K, Yong KS, Tiong WK. A new hybrid ensemble feature selection framework for machine learning-based phishing detection system. Inf Sci 2019;484:153–66.
[25] Moghimi M, Varjani AY. New rule-based phishing detection method. Expert Syst Appl 2016;53:231–42.
[26] Minocha S, Singh B. A novel phishing detection system using binary modified equilibrium optimizer for feature selection. Comput Electr Eng 2022;98:107689.
[27] Vrbančič G, Fister I, Podgorelec V. Datasets for phishing websites detection. Data Brief 2020;33:106438.
[28] Esmaeili Kelishomi A, Garmabaki AH, Bahaghighat M, Dong J. Mobile user indoor-outdoor detection through physical daily activities. Sensors 2019;19(3):511.
[29] Bahaghighat M, Akbari L, Xin Q. A machine learning-based approach for counting blister cards within drug packages. IEEE Access 2019;7:83785–96.
[30] Abu-Nimeh S, Nappa D, Wang X, Nair S. A comparison of machine learning techniques for phishing detection. In: Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit; 2007. p. 60–9.
[31] Jain AK, Gupta BB. PHISH-SAFE: URL features-based phishing detection system using machine learning. Cyber security. Springer; 2018. p. 467–74.
[32] Harinahalli Lokesh G, BoreGowda G. Phishing website detection based on effective machine learning approach. J Cyber Secur Technol 2021;5(1):1–4.
[33] Thabtah F, Kamalov F. Phishing detection: a case analysis on classifiers with rules using machine learning. J Inf Knowl Manag 2017;16(04):1750034.
[34] Awasthi A., Goel N. Feature selection & ML based prediction of phishing websites. EasyChair preprint, 2022.
[35] Orunsolu AA, Sodiya AS, Akinwale AT. A predictive model for phishing detection. J King Saud Univ Comput Inf Sci 2019.
[36] Almomani A, Alauthman M, Shatnawi MT, Alweshah M, Alrosan A, Alomoush W, Gupta BB. Phishing website detection with semantic features based on machine learning classifiers: a comparative study. Int J Semant Web Inf Syst 2022;18(1):1–24.
[37] Zhang Y, Hong JI, Cranor LF. Cantina: a content-based approach to detecting phishing web sites. In: Proceedings of the 16th international conference on World Wide Web; 2007. p. 639–48.
[38] Xiang G, Hong J, Rose CP, Cranor L. Cantina+ a feature-rich machine learning framework for detecting phishing web sites. ACM Trans Inf Syst Secur 2011;14(2):1–28.
[39] Sanglerdsinlapachai N, Rungsawang A. Using domain top-page similarity feature in machine learning-based web phishing detection. In: Proceedings of the 3rd international conference on knowledge discovery and data mining. IEEE; 2010. p. 187–90.
[40] Buber E, Demir Ö, Sahingoz OK. Feature selections for the machine learning based detection of phishing websites. In: Proceedings of the international artificial intelligence and data processing symposium (IDAP). IEEE; 2017. p. 1–5.

[41] Jain AK, Gupta BB. A survey of phishing attack techniques, defence mechanisms, and open research challenges. Enterprise Inf Syst 2022;16(4):527–65.
[42] Tewari A, Gupta BB. Secure timestamp-based mutual authentication protocol for IoT devices using RFID tags. Int J Semant Web Inf Syst 2020;16(3):20–34.
[43] Chawla NV, Japkowicz N, Kotcz A. Special issue on learning from imbalanced data sets. In: Proceedings of the ACM SIGKDD explorations newsletter. 6; 2004. p. 1–6.
[44] Batista GE, Bazzan AL, Monard MC. Balancing training data for automated annotation of keywords: a case study. WOB; 2003. p. 10–8.
[45] Wilson DL. Asymptotic properties of nearest neighbor rules using edited data. IEEE Trans Syst Man Cybern 1972:408–21.
[46] Batista GE, Prati RC, Monard MC. A study of the behavior of several methods for balancing machine learning training data. In: Proceedings of the ACM SIGKDD explorations newsletter. 6; 2004. p. 20–9.
[47] Couronné R, Probst P, Boulesteix AL. Random forest versus logistic regression: a large-scale benchmark experiment. BMC Bioinform 2018;19(1):1–4.
[48] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Scikit-learn Vanderplas J. Machine learning in Python. J Mach Learn Res 2011;12:2825–30.
[49] Khorshid SF, Abdulazeez AM. breast cancer diagnosis based on k-nearest neighbors: a review. PalArch's. J Archaeol Egypt Egyptol 2021;18(4):1927–51.
[50] Short R, Fukunaga K. The optimal distance measure for nearest neighbor classification. IEEE Trans Inf Theory 1981;27(5):622–7.
[51] Weinberger KQ, Saul LK. Distance metric learning for large margin nearest neighbor classification. J Mach Learn Res 2009;10(2).
[52] Zhang Z. Too much covariates in a multivariable model may cause the problem of overfitting. J Thorac Dis 2014;6(9):E196.
[53] Zhang H. The optimality of naive Bayes. Aa 2004;1(2):3.
[54] Zhang W, Wu C, Zhong H, Li Y, Wang L. Prediction of undrained shear strength using extreme gradient boosting and random forest based on Bayesian optimization. Geosci Front 2021;12(1):469–77.
[55] Breiman L. Random forests. Mach Learn 2001;45(1):5–32.
[56] Mahesh B. Machine learning algorithms-a review. Int J Sci Res 2020;9:381–6.
[57] Friedman J, Hastie T, Tibshirani R. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). Ann Stat 2000;28(2):337–407.
[58] Friedman JH. Greedy function approximation: a gradient boosting machine. Ann Stat 2001:1189–232.
[59] Charbuty B, Abdulazeez A. Classification based on decision tree algorithm for machine learning. J Appl Sci Technol Trends 2021;2(01):20–8.
[60] Cunningham P. Dimension reduction. Machine learning techniques for multimedia. Springer; 2008. p. 91–112.
[61] Abutair HY, Belghith A. Using case-based reasoning for phishing detection. Procedia Comput Sci 2017;109:281–8.
[62] Chen T., Guestrin C. Xgboost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining 2016, p. 785–94.
[63] Gupta BB, Arachchilage NA, Psannis KE. Defending against phishing attacks: taxonomy of methods, current issues and future directions. Telecommun Syst 2018;67:247–67.
[64] Almomani A, Alauthman M, Shatnawi MT, Alweshah M, Alrosan A, Alomoush W, Gupta BB. Phishing website detection with semantic features based on machine learning classifiers: a comparative study. Int J Semant Web Inf Syst 2022;18(1):1–24.