Data/Math (0000), **xx**:xx 2–17

RESEARCH ARTICLE

Identifying Malicious Web Domains Using Ensemble Machine Learning Technique from Multiple Feature Generation

Ikenna Miracle Umeh (A0220767)[1]

School of Computing, Engineering and Digital Technologies.
Teesside University, E-mail: a0220767@tees.ac.uk.

Keywords: Malicious Attack, Ensemble Machine Learning, Feature Engineering,

Abstract
Malicious activities using web domains have increased rapidly in recent years, since the COVID-19 pandemic and has raised a major concern to the safety of internet users and organizations. In this study, an ensemble technique from different multiple feature generation is proposed for improving malicious domain identification. 1,101,367 URLs was sourced from Kaggle and used to generate 20 lexical features, 20 performance metrics features from Google PageSpeed Insight API and bag of words. Natural language processing (NLP) technique was used to vectorize the text from the URLs, 6 different machine learning algorithms,Support Vector Classifier (SCV), Multinomial Naïve Bayes Classifier (MNBC),AdaBoost Classifier (ADC), Random Forest Classifier (RFC), XGBoost Classifier (XGBC) and Logistic Regression Classifier (LRC) were trained on the different features generated. Our proposed model was an ensemble of the top performing models evaluated on the metrics of precision, recall, accuracy, and AUC. Our proposed ensemble model produced quality result in identifying malicious URLs; it also performed better than the models which it was compared with, in terms of the evaluated performance metrics. Our experimental result confirm that the accuracy of our model, its precision, the recall and AUC score of the proposed model was 0.99, 0.99, 0.98, and 0.9884 respectively.

Contents

1. Introduction

The use of web domains in malicious activities has rapidly increased in recent years. Since the COVID-19 pandemic, there have been numerous malicious domains, used for malicious attacks in form of phishing, and scam messages with the intent to defraud, ransomware attacks have increased at an alarming rate [23] [35]. The number of these web domains used in these malicious activities has increased drastically that is making the internet unsafe and can't be overlooked, with an increasing number of new domain names being registered in bulk as part of a campaign for malicious activity and deployed simultaneously [34]. These malicious activities carried out by malicious web domain can cause not just monetary loss and the infringement of people's privacy but also hinders the effectiveness of online retail platforms and online interactions and communal activities, because of a lack of trust and fear of being victim to malicious attacks. identifying these malicious web domains is important for preserving the safety of the online community and civilization advancements and therefore is matter of utmost importance.

"Domain name system (DNS), a core component of the Internet functionality that maps the Internet Protocol (IP) addresses that may be difficult to remember to easy and memorable domain names, which has been widely used in network communications. Almost all Internet applications need to use DNS to resolve domain names and achieve resource location" [19] [40]. "DNS services are being abused to carry out malicious attacks and they play a significant role in most cyber-attacks observed nowadays, acting as a facilitator for Command and Control (C and C) communication for network attacks or impersonating legitimate websites in phishing attacks, and sending spam messages" [40].

To mitigate these malicious attacks propagated from malicious web domains, many techniques were proposed [1] [32]. Among which is maintaining an updated blacklist of malicious Uniform Resource Locators (URLs). This lengthy list of dangerous URLs has been compiled and is now prohibited from use. In the blacklist technique, a list of dangerous URLs is created manually or automatically, and it needs to be updated frequently to identify the most recent malicious URLs. [1]. One of the major disadvantages of the blacklist technique is the time and effort it takes in preparing such a huge list of malicious URLs. Furthermore, this method does not detect newly formed malicious URLs that are not on the blacklist.[20].

To solve the shortcomings of static blacklists, a machine learning technique, It entails developing a machine learning algorithm for identifying malicious from benign URLs based on their characteristics.Different algorithms for classification problems are included in machine learning techniques. The harmful URL that is already on the blacklist can be discovered, though, because it is detected quickly enough. In order to train classification algorithms (models) to predict whether a recently created URL is malicious or benign, harmful URL detection uses a set of features extracted from URLs, including both malicious and benign URLs. The literature is consequently paying more attention to "machine learning-based models" since they require "less manual intervention than blacklists" and have demonstrated remarkable outcomes [32] [4] not only in terms of identification of dangerous web domains but also for various domains[10].

1.1. Research Objective

1. To identify malicious web domains from benign ones using machine learning.
2. To do a literature review on different approaches and result achieved in identifying malicious domain.
3. To generate multiple features from the URLs dataset.
4. To analyze, design, and implement an ensemble machine learning model for identifying malicious domains.
5. To test and evaluate the machine learning model.
6. To compare the performance of the ensemble machine learning approach from multiple features to that of a single feature machine learning model.

1.2. Research Problem

Malicious attacks in form of phishing and scam messages have been increasing in numbers since the COVID-19 pandemic [5], with more people falling victim to these malicious activities. it's becoming more difficult to distinguish between these domains used in malicious activities from those not malicious because of the high similarity between the legitimate domain names and those used in malicious activities.

This project aims to identify malicious domains used in phishing and scam messages using machine learning.

This project would utilize different machine learning techniques and natural language processing techniques to identify malicious URLs. This project would combine multiple public datasets to train machine learning algorithms to detect malicious URLs using the ensemble technique.

2. Literature Review

This section focuses on reviewing the work on detecting malicious web domains in order to shed light on a main concern that our approach sought to address. Since several methods were presented, feature extraction is restricted to methods using conventional machine learning-based methodologies. Because earlier studies relied on IP-based features or NXDomain answers, the suggested strategies were only useful when practitioners could gather this data [31]. Additionally, applying the detection model may be hampered by the privacy concern [30]. To go over these restrictions, numerous studies have provided detection methods that focus on a smaller set of features. These studies extracted key domain-related features and identified distinguishing traits between harmful and benign domains. A simple detection approach containing lexical features and host-based traits was given by Ma *et al*. Lexical features could be calculated with ease from the domain, and host data like geographic characteristics might be simply acquired. They were able to get a substantial detection performance with a significantly lower feature acquisition cost by analysing these properties [20].

Previous studies demonstrated that deep neural networks performed precisely when ASCII value-based feature encoding was used [36]. Although feature encoding using the domain's ASCII values has been shown to be efficient, Park *et al.*[27] explored a linguistic feature approach to the analysis. Park *et al.*[27] proposed a technique of using domain feature-based techniques centered on the language patterns of the domain and establishing a technique that uses an autoencoder to find malicious domains. Training a detection model with only benign domains using an unsupervised machine learning method of detecting malicious web domains without as much labeling work. Although his detection model achieved a staggering 99 percent precise accuracy on a small training dataset, however, the sources of malicious domains or certain classes of Domain Generated Algorithms cannot be determined by the detection model.

Hu *et al.* [17] presented a technique for identifying malicious web domains using only numerical features generated from online credibility and performance data based on the performance and popularity metrics for the web domains. Individual machine learning classifiers and ensemble classifiers such as Random Forest, Gradient Boosting Machine, AdaBoost, and Bagging were trained using a feature selection technique based on binary particle swarm optimization (BPSO). Chiong *et al.* [10] applied a similar feature extraction technique as proposed by Hu *et al.*[17] "in an ensemble model built on the support vector machine with fuzzy-weighted least squares (EFW-LS-SVM) for enhancing the detection of malicious web domains. They handled the class imbalanced in the nature of benign and malicious web domain data, where malicious web domains tend to be the minority, this work was focused on reducing the noise sensitivity that machine learning models have in the data. Synthetic Minority Oversampling Technique (SMOTE) technique was used to improve the performance of all estimated models."

Zhang *et al.* [37] suggested a method known as the "DomainWatcher" to find malicious domains based on both regional and global textual features. By comparing the similarity between the tested domain and known domains, they were able to determine the maliciousness of the domain using 10 lexical features based on the textual features of the domain, imitation features, and bigram features of domain names. Bilge *et al.* [6]' EXPOSURE is a passive DNS analysis service that has been proposed to find malicious domains. 15 distinct features, classed as time-based features, DNS answer-based features, TTL value-based features, and domain name-based features, were actively worked on by EXPOSURE in real time. The J48 decision tree technique was used to build a classifier, and they achieved successful results with deployments operating in numerous places throughout the world.

Al Messabi, *et al.* [4] proposed an approach for detecting malicious domains based on domain name features and DNS records. Basic features, character indicator variables, top-level domain-based features, and tokens are the four main features identified. The NS lookup command line was also used to extract IP addresses for domain names. They worked hard to identify and compile a list of malicious TLDs and inappropriate words that are used in domain names to increase their maliciousness. They used 10-fold cross-validation to create a model using the J48 decision tree classifier. Han and Zhang [16] to identify benign domain names, a technique based on domain names and their TTL (time-to-live) features was proposed. They employed the naive Bayesian classifier algorithm. They were only interested in determining the benignity of a domain name, which can then be used to determine the maliciousness of a domain name. Djaballah *et al.* [11] presented a method for detecting malicious URLs in a social network like Twitter. The paper contains 25 URL features (divided into four categories). 1. Lexical characteristics 2. Features of a website 3. host-based features, 4. popularity features, and 5. Twitter user account-related features. UCI phishing datasets and 10,000 Twitter accounts are among the datasets. For the experiments, three machine learning algorithms (Logistic Regression, SVM, and Random Forest) were used, yielding 90.28 percent, 93.43 percent, and 95.51 percent accuracy, respectively.

Doyen *et al.* [12] presented a survey on malicious URL detection using machine learning and classified the features used in machine learning techniques for detecting malicious URLs as Lexical features, host-based features, content-based features, and other features (popularity features). They also examined the various techniques for detecting malicious URLs in the literature. To detect malicious URLs, a multiclass classification method was proposed. The study takes into account 63 lexical features, 34 content-related features, and 18 host-related features. SVM and multiclass CW learning were two algorithms used in the experiments. Data from PhishTank, Alexa, and jwSpamSpy are used to collect 49,935 URLs (26041 benign URLs, 8976 phishing URLs, 11,297 malware URLs, and 3621 spam URLs). The test yielded an average prediction accuracy of 98.44 percent [26].

Zhao *et al.* [39] worked on a detection algorithm for identifying malicious domain based on N-Gram natural language processing techniques of the domain name. Their detection algorithm based

on N-Gram, 100,000 domain names of the top Alexa 2013 ranking was used in the N-Gram method, substrings were segments from the domain names to lengths ranging from 3 to 7 which is then converted to its calculated numerical weights and values. In their experiment, the proposed detection algorithm yielded an accuracy rate of 94.04 percent with a very lower time complexity than other popular malicious domain names detection algorithms.

Palaniappan *et al*.[25], proposed a system that uses blacklists in addition to web-based features of the domain name, the DNS data, and lexical features to identify malicious domains. Similar method was also used by Bilge *et al.* [6] in there passive DNS analysis service called EXPOSURE. However, Palaniappan *et al.* [25]' approach which is based on the four features which was extracted from the given domain name, to build a classifier model using logistic regression algorithm to identify malicious domain from benign ones, was based on active DNS analysis rather than the passive DNS analysis approach used by Bilge *et al.* [6]. The result achieved in their experience was lower than the result achieved by approach taken by Bilge *et al.* [6].

Pradeepa and Devi, [28] proposed a lightweight machine learning model for detecting malicious domain, using a combination of fewer lexical features, content-based feature, bag-of-words, and popularity feature to train a classification model for identifying malicious domains. They trained a support vector machine algorithm and a random forest algorithm, the later model achieved a higher accuracy and precision and recall with an execution time of 0.17 seconds, outperforming the support vector machine model.

Venugopal *et al* [33] proposed a design of an ensemble machine learning technique for detecting malicious URLs, their proposed model classifies URLs and web pages into legitimate and malicious web site. The method used by the team employed domain registration attributes of the URLs, the URL text itself, also took consideration of the structure of the webpage and its contents. The approach for the method used natural language processing techniques as well as machine learning models with the vast variety of feature combined. A Bidirectional Encoder Representations from Transformers (BERT), Long short-term memory (LSTM), and Decision Tree were ensembled to tackle the challenge of achieving a better accuracy rate in identifying malicious URLS. The final model from the experiment achieved a high accuracy rate of 95.3 percent, however, the result achieved from the ensemble model was lower in accuracy rate compared to the accuracy rate of 99.98 percent they achieved from training a single Long short-term memory model on the web page text dataset.

3. Method

The phases of the proposed method are as follows: data preparation, feature extraction, normalizing, training the models, ensemble the models, testing, and evaluation as shown in Figure 1.

3.1. Dataset and Data Preparation

The labeled data used in this study was sourced from Kaggle. The dataset consists of 1,101,367 URLs, out of which 773,818 of the URLs are benign or safe, and 316,409 URLs were malicious. The size of the dataset used in this study was greater than that used by Patil and Patil [26]. The dataset was curated from five different sources. The benign, and malicious (phishing, malware, and defacement) URLs were sourced and curated from the different five sources of the URL dataset (ISCX-URL-2016); they sourced and used the malware domain black list dataset to increase phishing and malware URLs. They increased the number of benign URLs by using the Faizan GIT Repo, and they also increased the number of phishing URLs by using the Phishtank and PhishStorm datasets..

The compiled dataset had four classes: phishing, defacement, benign, and malware. However, for this study, we would be working on two classes, converting the dataset into a binary class label

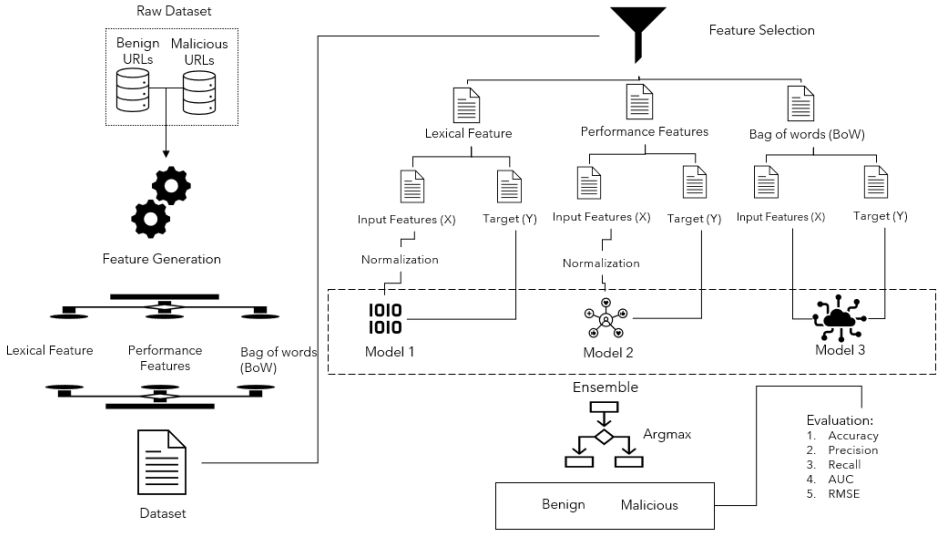

Figure 1. *Overview of the proposed system.*

of benign and malicious by converting the labels for phishing, defacement, and malware URLs into malicious as the label.

3.2. Feature Extraction

New features that can be essential and contribute greatly to the experiment can be identified during feature extraction from the raw dataset. Feature extraction is the process of extracting these critical features from the raw dataset. It is one of the pre-processing steps in data analysis and one of the major phases of our proposed method. The feature extraction phase produces a tabular list of new features and their values, which is saved as a CSV file. Table 1 and 2 below shows the extracted features of our experiment.

Table 1 contains the lexical features, while Table 2 contains the performance feature. The feature extraction phase considered shortened URLs if the URL present in the raw dataset used a shortening service to reduce the length of the URL or mask the actual URLs. The short URLs were converted to the original URL for further feature extraction. BoW (Bag of Words) technique used was the term frequency-inverse document frequency (TF-IDF), means the weight allocated to each token not only depends on its frequency in a document but also how persistent that term is in the entire corpora present in the URL. Features extracted for the proposed system were lexical features and performance features.

3.2.1. Lexical features

This study features not only absorbed prior work's experience, but also created new ones. [28][20][37], and select the common characteristics, such as the length of a domain name, whether the domain contains an IP address, and the number of special characters, but also extend some new textual features, such as the number of special characters and numeric characters, the percentage of special characters in domain length, checking for shortening service, the ratio of special characters divided by domain length, the ratio of numbers and special characters These 20 lexical characteristics are textual features of domains and are easily retrieved and normalised.

Table 1. *Lexical Features.*

Features	Description
URL Length	SLD and subdomains make up the domain's length. Because phishing URLs or malicious domains are long based on previously discovered malicious URLs, it is essential to think about the length feature.
Top Level Domain (TDL)	According to previous research, most malicious domains have been associated with specific TLDs for an extended period of time, such as.com and.pw, to name a few. When a URL's TLD has previously been used for phishing URLs, the likelihood of it being malicious increases..
Numerical Percentage	This feature displays the proportion of numerical characters in reference to the domain's length.
Special Character	This gives the individual count as well as the total count of the special characters found in the URLs.
Check Protocol (HTTP/HTTPS)	Gives out whether the website uses HTTPS or HTTP protocol. This returns 0 or 1, where 0 signifies an HTTP protocol and 1 represents an HTTPS protocol.
Count of the numerical digits in URL	Gives the count of the numerical digit found in the URLs. This returns a numerical value.
Letters Count	This feature gives the letters count of the URLs. This returns a numerical value.
Letter percentage	This is the percentage of the letters to the length of the URLs.
Check URL shortening service	A URL shortening provider is a third-party website that transforms a long URL into a short, case-sensitive alphanumeric characters.
Checking for IP	Gives a check for the IP of the web address, this returns a Boolean value of 0 or 1.
Check subdomain	This checks if the URL has a subdomain. This returns a Boolean value of 0 or 1.
Subdomain length	Gives the length of the subdomain found on the URL. This returns a numerical value.
ratio URL Dom Len	In the URL, the ratio of URL length to domain length.
ratio Alpha Num URL	The proportion of alphabets to numbers in the URL.
ratio Num Spl URL	The ratio of numbers to special characters in the URL.
ratio Spl in URL	The total number of special character appearances divided by the domain length.

3.2.2. Performance feature (Google PageSpeed Insight)

Google Page Speed is a speed metric that shows how quickly a web domain loads. These performance features in Table 2 were generated using Google PageSpeed insight API. PageSpeed Insights (PSI) analyses a page's user experience on mobile and desktop devices and makes recommendations for how to improve it.

PSI provides lab as well as field data for a page. Because it is collected in a controlled environment, lab data is useful for debugging issues. However, it is possible that it does not grasp real-world impediments. Field data is useful for capturing true, in-person consumer experience, but it has fewer metrics.

PSI analyses a given URL in a simulated environment for performance, accessibility, best practises, and SEO categories using Lighthouse.

3.3. Normalization

It's the process of reducing the huge data points within a specific range, like between 0 and 1 or between -1 and +1. Normalization is required when the features have a big difference in the range of values. The normalization of the data reduces the bias in training machine learning algorithms. It also, speeds up the computational and learning process involved in the training of a machine learning model, since all the values would be in same scale. The goal of this pre-processing step is to rescale the input vector and change the weight and bias associated with the relevant vector in order to obtain the same output features as before [21].

Min-Max Method:

The method is used in place of the Z-score Method. This method rescales any range's attributes or outputs into a new range. Typically, the features are scaled from 0 to 1 or (-1) to 1.

The equality used in the technique is as follows, where Xmin represents the minimum value, Xmax represents the maximum value, X is the input value, and Xi represents normalised data[3] [21]:

$$X^i = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3.1}$$

3.4. Natural Language Processing

This section discusses some NLP techniques that are relevant to this research. Natural language documents are separated into words using NLP techniques like Bag of Words (BoW). The corpus of words is transformed into vectors that computers can process.
The BoW method of document classification uses the frequency of each word as a feature to train a classifier. This model converts the frequency of each word in a text of the URLs or web page into vectors [29].

Documents can be converted into vectors using this model. This model does not appear to preserve the order of the words in the original documents. The number of distinct words in the original documents is equal to the dimension of converted vectors. To analyse large-scale data, the dimension should be reduced so that it can be processed in a reasonable amount of time.

TF-IDF Vectorizer

"TF-IDF stands for "term frequency-inverse document frequency", means the weight allocated to each token not only depends on its frequency in a document but also how persistent that term is in the entire corpora. This is very common algorithm to transform text into a meaningful representation of numbers which is used to fit machine algorithm for prediction" [2].

As a result, this model allows for the conversion of documents into vectors. This model does not appear to preserve the order of the words in the original documents. The number of distinct words in the original documents is equal to the dimension of converted vectors. To analyse large-scale data, the dimension must be reduced so that it can be analysed in a reasonable amount of time.

Table 2. *Performance Features.*

Features	Description
Observed First Paint	Observed time for when the first element paints the viewport.
First Contentful Paint (FCP)	"FCP is an important, user-centric metric for measuring perceived load speed because it marks the first point in the page load timeline where the user can see anything on the screen—a fast FCP helps reassure the user that something is happening. The FCP metric measures the time it takes from when the page first loads to when any part of the page's content appears on the screen." [15]
Observed First Contentful Paint	Observed FCP for the URL.
First Meaningful Paint (FMP)	"When the primary content of a page is visible to the user, FMP measures it. The time in seconds between the user initiating the page load and the page rendering the primary above-the-fold content is the raw score for FMP. FMP essentially displays the timing of the paint after which the most significant above-the-fold layout change occurs. First Meaningful Paint detects when a page's primary content is visible." [15]
Observed First Meaningful Paint	Observed FCP for the URL.
Cumulative Layout Shift	"Cumulative Layout Shift (CLS) is an important consumer metric for measuring visual stability because it quantifies how frequently users encounter unexpected layout shifts—a low CLS ensures that the page is delightful. CLS is a metric that measures the largest burst of layout shift scores for each unexpected layout shift that occurs over the course of a page's lifecycle." [15]
Observed Cumulative Layout Shift	Observed time for the CLS for the domain.
Cumulative Layout Shift Main Frame	CLS Main Frame measures of the largest burst of layout shift scores of a page fitting the viewport of the main screen. [15]
Observed Cumulative Layout Shift Main Frame	Observed time for CLS main frame at current loading of the URL.
Total Cumulative Layout Shift	"The total of all individual layout shift scores that occurred during the page's lifetime." [15]
Observed Total Cumulative Layout Shift	Observed score of the total CLS.
Interactive	"Time to interactive is the amount of time it takes for a page to become fully interactive. A page is considered fully interactive when the following conditions are met: - the page displays useful content, as measured using the First Contentful Paint; - event handlers are registered for the most visible page elements; and - the page responds to user interactions within 50 milliseconds." [15]

Observed Dom Content Loaded	"The observed time when the initial HTML document has been completely loaded and parsed, without the need to wait for stylesheets, images, or subframes to load." [15]
Largest Contentful Paint	"Largest Contentful Paint (LCP) is an important user-centric metric for measuring perceived load speed because it indicates when the page's main content has likely loaded—a fast LCP helps reassure the user that the page is useful. The LCP metric reports the render time of the largest visible image or text block within the viewport in relation to when the page first began loading" [15] .
Observed the Largest Contentful Paint	Observed time for the largest contentful element painted within the viewport.
Observed Load	Observed loading time for the page.
Total Blocking Time	"Total Blocking Time (TBT) is an important laboratory metric for measuring load responsiveness because it quantifies how non-interactive a page is before it becomes reliably interactive—a low TBT helps ensure that the page is usable. TBT is a metric that calculates the total amount of time between First Contentful Paint (FCP) and Time to Interactive (TTI) when the main thread was blocked for a long enough period of time to prevent input responsiveness. When the task length exceeded 50ms, the sum of all time periods between FCP and Time to Interactive was expressed in milliseconds." [15]
Observed Time Origin	Observed time taken to aggregate the Field Data of all the URLs on the website [15].
Speed Index Speed Index	"Measures the speed with which content is visually displayed during page load" [15].
Observed Speed Index	This is the observed speed index at a time.

3.5. Model

We apply 6 different machine learning algorithms to our generated features: a Logistic Regression Classifier (LRC), Multinomial Naive Bayes Classifier (MNBC), Random Forest Classifier (RFC), Ada Boosting Classifier (ADC), XGBoosting Classifier (XGBC), and Support Vector Classifier (SVC).

Logistic Regression Classifier (LRC)

The logistic regression classifier is a linear model for classification that is simple to understand. It includes a dependent variable that can be represented in binary (0 or 1, true or false, yes or no) values, which means that the result can only be one of two possibilities. It can be used, for example, to determine the likelihood of a positive or negative event [24].

Multinomial Naive Bayes Classifier (MNBC)

The multinomial distribution is a broadening of the binomial distribution. Multinomial Naive Bayes is an algorithm for classifying data with discrete features that is implemented for multinomially distributed data (word counts for text classification). [22]

Although the Bayes method is a good classifier, it is a poor probability estimator, so the probability outputs from predict proba are not completely accurate. The majority of the time, Naive Bayes algorithms are used in sentiment analysis, spam filtering, or recommendation systems. Multinomial Naïve Bayes classifier is a linear probabilistic classifier based on the Bayes theorem [22] with the simple assumption that each pair of features is independent given the context of the class for document classification problems (whether a document belongs to the category of music, technology, or sport). Despite the

fact that this assumption is clearly 'naive' in most real-world applications, Nave Bayes has performed admirably in a variety of tasks such as document classification and spam filtering [22].

Random Forest Classifier (RFC)

The RFC, also known as an ensemble model, is a collection of tree predictors [7], with each tree built using a different bootstrap sample from the original data. The RF model has demonstrated promising performance in both classification and regression problems by aggregating those tree predictors [14]."It re-samples several training sets based on the given data, with each sub-training set having the same number of samples; decision tree classifiers are then trained using those re-sampled training sets. The random selection of features also increases model diversity, which aids in reducing over-fitting when the models are aggregated for final prediction" [7].

Ada Boosting Classifier (ADC)

Adaboost, which stands for Adaptive Boost, is a well-known member of the Boost method [27]. Adaboost, like bagging, manipulates the samples to create a set of weak learners. Adaboost is adaptive in the sense that each weak learner is built sequentially, as opposed to bagging, which builds each learner on the bootstrapped sample independently. The subsequent learner pays more attention to instances misclassified by previous classifiers by maintaining a set of adaptive weights. A composited classifier is created by aggregating the weak learners based on their accuracy using a weighted voting mechanism [27].

XGBoosting Classifier (XGBC)

XGBC is a decision tree-based ensemble model that is a gradient-boosted decision tree implementation [9]. "It can reduce over-fitting by using a more regularised model formalisation technique such as gradient boosting" [13]. "Furthermore, XGBC has practical properties for real-world applications, such as tree construction parallelisation, distributed computing for training models, out-of-core computing for large datasets, and cache optimisation of data structures and algorithms" [8].

Support Vector Classifier (SVC)

The support vector machine classifier minimises the upper bound of the generalisation error, the support vector machine classifier implements the structural risk minimisation principle [2]. The idea is to build an optimal separating hyperplane in a high-dimensional feature space by maximising the distance between the hyperplane and the closest points of any class. As a result, SVM has excellent generalisation performance, which has been widely and successfully applied in a variety of fields[2].

3.5.1. Ensemble Model

The proposed ensemble model used for this study is the combination of the three good performing models from the evaluation of the machine learning algorithms used in fitting the different generated features.

3.6. Evaluation

The following measure parameters adopted by Pradeepa and Devi (2022) were used to evaluate predictive performance of the proposed detection algorithm:
The Accuracy Rate (AR) is calculated by dividing the total number of correctly detected domains by the total number of detected domains.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.2}$$

Precision Rate (PR) is defined as the number of correctly predicted malicious domain names divided by the total number of malicious domain names predicted.

$$Precision = \frac{TP}{TP + FP} \tag{3.3}$$

AUC is a measure of the classification model's accuracy. The Receiver Operating Characteristic (ROC) curve illustrates its manifestation. A ROC curve is created by mapping the results of each sample on a two-dimensional plane. The AUC of the model is represented by the area under the curve. The greater the value of AUC, the better the model's classification effect. The abscissa of the curve represents the False Positive Rate (FPR), and the ordinate represents the True Positive Rate (TPR). The formula is as follows:

$$Sensitivity = Recall = \frac{TP}{TP + FN} \tag{3.4}$$

$$Specificity = \frac{TN}{FP + TN} \tag{3.5}$$

$$AUC = Sensitivity - (1 - Specificity) \tag{3.6}$$

4. Experiment and Result Analysis

To verify the performance of our proposed approach, the experiment was conducted using a large volume of malicious URLs and benign URLs, the lexical features generated, and the performance feature generated from Google PageSpeed insight.

4.1. Experiments

The experiment environment is shown in Table 3

Table 3. *Experimental environment.*

Parameters	**Value**
CPU	Intel(R) Core (TM) i9-9900KF CPU @ 3.60GHz 3.60 GHz
GPU	NVIDIA GeForce RTX 3060, 12GB
Memory	24.0 GB
OS	Windows 10 Home, 64 bits
Platform	Jupyter Notebook
Python	3.9

4.2. Experimental data

The dataset consists of 1,101,367 URLs, out of which 773818 of the URLs are benign or safe, and 316409 URLs were malicious. 40 Features were generated and used to train different models, 20 lexical features were generated from the URLs and 20 performance features.

4.3. Experiment Result and Analysis

The result of the experiment is presented in the tables below.

4.3.1. Exploratory Data Analysis

The dataset used for the experience had 70:30 ratio of class proportion, favouring the benign class. Out of the total 1,101,367 URLs, only 11,140 URLs were duplicated which is about 10 percent of the URLs.

70 percent of the URLs was using a HTTP protocol while 30 percent was a more secure connection on a HTTPS protocol.

4.3.2. Lexical Feature Result

The performance evaluation of the machine learning algorithms which was trained using the lexical features generated from the URLs is shown in the table 4.

The XGBoost classifier had the highest AUC score of 0.9933, while the lowest score for AUC was observed in the multinomial Naïve Bayes classifier, with a score of 0.8595. the result achieved in this experiment compared favourable with the result obtained by Pradeepa and Devi (2022) in their lightweight approach for malicious domain detection using machine learning.

Table 4. *Lexical Features Model Result.*

Algorithms	**Accuracy**	**Precision**	**Recall**	**AUC**
LRC	0.91	0.92	0.86	0.9325
MNBC	0.81	0.82	0.69	0.8595
RFC	**0.97**	**0.97**	**0.96**	0.9912
ADC	0.91	0.92	0.85	0.9386
XGBC	**0.97**	**0.97**	**0.96**	**0.9933**
SVC	0.92	0.93	0.87	0.9451

4.3.3. Performance Metric Features Result

The performance evaluation of the machine learning algorithms which was trained using the performance metrics features generated from the URLs using Google PageSpeed Insight API is shown in the table 5

The result achieved in this experiment had lower scores in all evaluation metrics compared to the result achieved from training the models using the lexical features. The random forest classifier (RFC) scored the highest in the AUC value with a score of 0.6285, while the least value of 0.5112 was observed in the support vector classifier (SVC).

Table 5. *Performance Metric Features Model Result.*

Algorithms	**Accuracy**	**Precision**	**Recall**	**AUC**
LRC	0.74	0.65	0.50	0.5860
MNBC	0.74	0.37	0.50	0.5488
RFC	**0.75**	0.65	**0.54**	**0.6285**
ADC	0.74	0.55	0.50	0.5985
XGBC	0.74	0.61	0.54	0.6226
SVC	**0.75**	**0.76**	0.50	0.5112

4.3.4. Bag of Words – TF-IDF Features

The performance evaluation of the machine learning algorithms which was trained using the term frequency-inverse document frequency feature is shown in the table 6.

The result from the experiment showed that logistic regression classifier had better performance in accuracy, precision and recall compared to other machine learning algorithms. However, the multinomial Naïve Bayes classifier scored higher in the AUC score of 0.9724 compared to the LRC model, which scored 0.9570. The result obtained in this case using the TF-IDF feature compared favourably with the performance result obtained from training a machine learning on detecting malicious URLs using lexical features in this study and the work done by Pradeepa and Devi (2022).

Table 6. *TF-IDF Features Model Result.*

Algorithms	**Accuracy**	**Precision**	**Recall**	**AUC**
LRC	0.95	0.95	0.92	0.9570
MNBC	0.92	0.93	0.87	0.9724
RFC	0.93	0.92	0.91	0.9713
ADC	0.82	0.82	0.76	0.8545
XGBC	0.91	0.90	0.88	0.9654

4.3.5. Comparing Our Ensemble Model Performance to the best single models

Table 7 shows the performance evaluation of the ensemble model compared to the best performing single feature model. The result showed that the ensemble model from the multiple features scored higher in accuracy, precision, recall and AUC compared to using solely lexical, performance or TF-IDF feature in training a machine learning algorithm.

Table 7. *TF-IDF Features Model Result.*

Algorithms	**Accuracy**	**Precision**	**Recall**	**AUC**
Ensemble model (XGB Lexical + RF PM + LR TFIDF)	0.99	0.99	0.98	0.9884
XGB lexical	0.97	0.97	0.96	0.9933
RF PM	0.75	0.65	0.54	0.6285
LR TFIDF	0.95	0.95	0.92	0.9570

5. Conclusion

This study proposes a new method for detecting malicious domains from URLs. The work extracted several features of the Uniform Resource Locator (URL) from 1,101,367 URLs set that are retrieved from Kaggle dataset. The results reveal that the ensemble technique from multiple features method outperforms the single feature method. Compared to the malicious domain names detection methods proposed by Al Messabi, *et al.*[4], Zhang *et al.* [37] and Patil and Patil [26], our approach outperformed

the competition in terms of overall performance, with a higher detection accuracy, precision and recall rate. The research can be expanded by using word segmentation and deep learning methods to analyse URL and web page content., several techniques for class imbalance can also be used to sample minority (malicious URLs) to improve the performance of the models trained using the performance metrics generated from Google PageSpeed Insight API.

Data Availability Statement. The data used in this study can be found here https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset+.

Ethical Standards. The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

References

[1] **Agbefu, R., Hori, Y. and Sakurai, K.** (2013) 'Domain Information Based Blacklisting Method For The Detection Of Malicious Webpages', *International Journal of Cyber-Security and Digital Forensics (IJCSDF)*, 2(2), pp. 36-47.

[2] **Ahuja, R., Chug, A., Kohli, S., Gupta, S. and Ahuja, P.** (2019) 'The Impact of Features Extraction on the Sentiment Analysis', *Procedia Computer Science*, 152, pp. 341-348. Available at: https://doi.org/10.1016/j.procs.2019.05.008.

[3] **Aksu, G., Güzeller, C. and Eser, M.** (2019) 'The Effect of the Normalization Method Used in Different Sample Sizes on the Success of Artificial Neural Network Model', *International Journal of Assessment Tools in Education*, 6(2), pp. 170-192.

[4] **Al Messabi, K., Aldwairi, M., Al Yousif, A., Thoban, A. and Belqasmi, F.** (2018) 'Malware detection using DNS records and domain name features', In Proceedings of the 2nd International Conference on Future Networks and Distributed Systems (ICFNDS '18). *Association for Computing Machinery*, New York, NY, USA, Article 29, pp. 1–7. Available at: https://doi.org/10.1145/3231053.3231082.

[5] **APWG**(2021) *Phishing Activity Trends Report - 2nd Quarter - APWG*, https://apwg.org. Available at: https://docs.apwg.org/reports/apwg_trends_report_q2_2021.pdf (Accessed: January 10, 2023).

[6] **Bilge, L., Sen, S., Balzarotti, D., Kirda, E. and Kruegel, C.** (2014) 'Exposure: A Passive DNS Analysis Service to Detect and Report Malicious Domains', *ACM Transactions on Information and System Security (TISSEC)*, 16(4): 1-28.

[7] **Breiman, L.** (2001) 'Random Forests. *Machine Learning,* 45(1):5-32. 10.1023/A:1010950718922.

[8] **Chen, T. and Guestrin, C.** (2016). Xgboost: A scalable tree boosting system. *Paper presented at the Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining.*

[9] **Chen, T., He, T., Benesty, M., Khotilovich, V. and Tang, Y.** (2015). Xgboost: extreme gradient boosting. R package version 0.4-2, 1-4.

[10] **Chiong, R., Wang, Z., Fan, Z. and Dhakal, S.** (2022) 'A fuzzy-based ensemble model for improving malicious web domain identification', *Expert Systems with Applications*, 204, pp. 117243. Available at: https://doi.org/10.1016/j.eswa.2022.117243.

[11] **Djaballah, A., Boukhalfa, K., Zakaria, G.and Oussama, B.** (2020)' A new approach for the detection and analysis of phishing in social networks: the case of Twitter', 1-8. 10.1109/SNAMS52053.2020.9336572.

[12] **Doyen, S., Chenghao, L. and Steven H.** (2019) 'Malicious URL Detection using Machine Learning: A Survey', *arXiv*, 1(1), pp. 37. Available at: https://doi.org/10.1145/nnnnnnn.nnnnnnn

[13] **Friedman, J.**(2002) 'Stochastic gradient boosting', *Computational statistics and data analysis*, 38(4), pp. 367-378.

[14] **Gislason, P., Benediktsson, J. and Sveinsson, J.** (2006) 'Random Forests for land cover classification', *Pattern Recognition Letters,* 27(4), pp. 294-300. Available at: https://doi.org/10.1016/j.patrec.2005.08.011.

[15] **Google Developers** (no date) *About pagespeed insights | google developers documentation.* Google. Availableat:https://developers.google.com/speed/docs/insights/v5/about (Accessed: January 10, 2023).

[16] **Han, C. and Zhang, Y.** (2017) 'CLEAN: An approach for detecting benign domain names based on passive DNS traffic',*In Proceedings of 6th International Conference on Computer Science and Network Technology (ICCSNT)*, pp. 343-346.

[17] **Hu, Z., Chiong, R., Pranata, I., Susilo, W. and Bao, Y.** (2016) 'Identifying malicious web domains using machine learning techniques with online credibility and performance data', 2016 *IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 5186-5194, Available at: http://doi: 10.1109/CEC.2016.7748347.

[29] **Kandala, R and Dhuli, R**(2018) 'Classification of imbalanced ECG beats using re-sampling techniques and AdaBoost ensemble classifier', *Biomedical Signal Processing and Control*, 41, pp. 42-254. Available at: https://doi.org/10.1016/j.bspc.2017.12.004.

[19] **Khormali, A., Park, J., Alasmary, H., Anwar, A., Saad, M. and Mohaisen, D.** (2021) 'Domain name system security and privacy: A contemporary survey', *Computer Networks*, 185, pp. 107699. Available at: https://doi.org/10.1016/j.comnet.2020.107699.

[20] **Ma, J., Saul, L., Savage, S. and Voelker, G.** (2009) 'Beyond blacklists: learning to detect malicious web sites from suspicious URLs', In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '09), *Association for Computing Machinery*, New York, USA, pp. 1245–1254. Available at: https://doi.org/10.1145/1557019.1557153

[21] **Mazziotta, M. and Pareto, A.** (2022) 'Data Normalization for Aggregating Time Series: The Constrained Min-Max Method', *Rivista Italiana di Economia, Demografia e Statistica*, LXXV(4), pp. 101-108.

[22] **Muhammad A., Kamran, A., Saleem, M., Abdul, J., Saleemullah, M. and Anees, A.** (2019) 'Multinomial Naive Bayes Classification Model for Sentiment Analysis', *International Journal of Computer Science and Network Security*, 19(3), pp. 62-67. Available at: https://doi.org/10.13140/RG.2.2.30021.40169.

[23] **Mvula, P., Branco, P., Jourdan, G. and Viktor, H.** (2022) 'COVID-19 malicious domain names classification', *Expert Systems with Applications*, 204, pp. 117553. Available at: https://doi.org/10.1016/j.eswa.2022.117553.

[24] **Niu, L.** (2018) 'A review of the application of logistic regression in educational research: common issues, implications, and suggestions', *Educational Review.* 72, pp. 1-27. Available at: https://doi.org/10.1080/00131911.2018.1483892.

[25] **Palaniappana, G., Sangeetha, S., Rajendrana, B., Sanjaya, S., Goyal, S. and Bindhumadhava, B.** (2020) 'Malicious Domain Detection Using Machine Learning on Domain Name Features, Host-Based Features and Web-Based Features', Third International Conference on Computing and Network Communications (CoCoNet'19), *Procedia Computer Science*, 171 (2020), pp. 654–661.

[26] **Patil, D. and Patil, J.** (2018) 'Feature-based Malicious URL and attack type detection using multi-class classification', *The ISC International Journal of Information Security*, 10(2), pp. 141-162. Available at: doi: 10.22042/isecure.2021.113973.404

[27] **Park, K., Song, H., Yoo, J., Hong, S., Cho, B., Kim, K. and Kim, H.** (2022) 'Unsupervised malicious domain detection with less labeling effort', *Computers and Security*, 116, pp. 102662. Available at: https://doi.org/10.1016/j.cose.2022.102662.

[28] **Pradeepa, G and Devi, R.** (2022) 'Lightweight approach for malicious domain detection using machine learning', *Scientific and Technical Journal of Information Technologies, Mechanics and Optics,* 22(2), pp. 262 - 268. Available at: http://doi: 10.17586/2226-1494-2022-22-2-262-268

[29] **Qader, W., Ameen, M. and Ahmed, B.** (2019) 'An Overview of Bag of Words; Importance, Implementation, Applications, and Challenges', *Fifth International Engineering Conference on Developments in Civil and Computer Engineering Applications*, 2019 - (IEC2019) - Erbil – IRAQ, pp. 200-204. Available at: http://doi:10.1109/IEC47844.2019.8950616.

[30] **Rossow *et al.***(2012) 'Prudent Practices for Designing Malware Experiments: Status Quo and Outlook,' *2012 IEEE Symposium on Security and Privacy*, pp. 65-79. Available at: http://doi: 10.1109/SP.2012.14.

[31] **Schiavoni S., Maggi F., Cavallaro L.and Zanero S.** (2014) 'Phoenix: DGA-based botnet tracking and intelligence', *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, *Springer*, pp. 192-211

[32] **Vanhoenshoven, F., Nápoles, G., Falcon, R., Vanhoof, K. and Köppen, M.**(2016) 'Detecting malicious URLs using machine learning techniques',*IEEE Symposium Series on Computational Intelligence* (SSCI). Available at: doi: 10.1109/SSCI.2016.7850079.

[33] **Venugopal, S., Panale, S., Agarwal, M., Kashyap, R. and Ananthanagu, U.** (2021) 'Detection of Malicious URLs through an Ensemble of Machine Learning Techniques', 2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), Brisbane, Australia, 2021, pp. 1-6, available at: http://doi: 10.1109/CSDE53843.2021.9718370.

[34] **Weber, M., Wang, J. and Zhou, Y.** (2018) 'Unsupervised Clustering for Identification of Malicious Domain Campaigns', In Proceedings of the First Workshop on Radical and Experiential Security (RESEC '18). *Association for Computing Machinery*, New York, USA, pp. 33–39. Available at: https://doi.org/10.1145/3203422.3203423.

[35] **Xia, P., Nabeel, M., Khalil, I., Wang, H. and Yu, T.** (2021) 'Identifying and Characterizing COVID-19-Themed Malicious Domain Campaigns', In Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy (CODASPY '21), *Association for Computing Machinery*, New York, USA, pp. 209–220. Available at: https://doi.org/10.1145/3422337.3447840.

[36] **Yu, B., Gray, D., Pan, J., Cock, M. and Nascimento, A.** (2017) 'Inline DGA Detection with Deep Networks', *IEEE International Conference on Data Mining Workshops* (ICDMW). Available at: http://doi: 10.1109/ICDMW.2017.96.

[37] **Zhang, P., Liu, T., Zhang, Y., Ya, J., Shi, J. and Wang, Y.** (2017) 'Domain Watcher: Detecting Malicious Domains Based on Local and Global Textual Features', *Procedia Computer Science,* 108, pp. 2408-2412. Available at: https://doi.org/10.1016/j.procs.2017.05.204.

[38] **Zhang, L. and Zhou, W.** (2011) 'Density-induced margin support vector machines', *Pattern Recognition*, 44(7), pp. 1448-1460. Available at: https://doi.org/10.1016/j.patcog.2011.01.006.

[39] **Zhao, H., Chang, Z., Bao, G. and Zeng, X.** (2019) 'Malicious Domain Names Detection Algorithm Based on N-Gram', *Journal of Computer Networks and Communications*, 2019, pp. 1-9. Available at: https://doi.org/10.1155/2019/4612474

[40] **Zhauniarovich, Y., Khalil, I., Yu, T. and Dacier, M.** (2018) 'A Survey on Malicious Domains Detection through DNS Data Analysis', *ACM Computing Surveys,* 1 (1), pp. 35. Available at: https://doi.org/10.1145/nnnnnnn.nnnnnnn.