IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p-ISSN: 2278-8727, Volume 27, Issue 2, Ser. 4 (Mar. – Apr. 2025), PP 57-68
www.iosrjournals.org

Detecting Malicious Websites Using Machine Learning Models By Incorporating Both Lexical And Network-Based Features.

Daniel Kwadwo Nterful.
Directorate of ICT Services, Takoradi Technical University (TTU), Ghana.
Richard Appiah.
Department of Computer Science, Takoradi Technical University (TTU), Ghana.
David Ezejimofor.
Directorate of ICT Services, Takoradi Technical University (TTU), Ghana.

Abstract
The utilization of blacklists is a commonly used approach for detecting malicious websites. However, blacklists have limitations as they lack comprehensive information and cannot be easily updated to include newly discovered harmful websites. To enhance security and reduce vulnerability to these attacks, it is crucial to employ techniques that can automatically identify and manage newly emerging malicious websites. In this regard, machine learning models offer a promising solution. By utilizing eight different machine learning models, namely Random Forests (RF), Decision Trees (DT), Logistic Regression (LR), Naive Bayes (NB), K-Nearest Neighbors (KNN), Support Vector Machines (SVM), XGBoost, and LightGBM, it is possible to effectively detect and classify malicious websites. These models leverage the power of machine learning algorithms to analyze various features and patterns associated with malicious URLs, enabling accurate identification and proactive defense against such threats. Additionally, it investigates the application of ensemble methods, particularly the Stacking method, to create a brand-new model known as DKN. The study explores the experimental assessment, including the dataset source, feature extraction, and evaluation measures, and presents the architecture of the DKN model. The outcomes show how well the suggested models and the ensemble DKN stacking model predict the characteristics of URLs. The paper looks at methods like downsampling and oversampling to enhance model performance as well as the problem of imbalanced datasets. By investigating the fusion of several variables and machine-learning models to produce precise predictions, the research makes a contribution to the field of malicious website identification.

Keywords: *Machine learning, Feature Extraction, Detection, Malicious URL*

Date of Submission: 01-04-2025 Date of Acceptance: 11-04-2025

---

1. Introduction

The internet offers a wealth of information and convenience, which has become part of our everyday lives. As a result of its widespread use, malicious websites have emerged as a significant concern, posing threats to user privacy, data security, and online transactions. Malicious websites use various techniques such as phishing attacks, malware distribution, and identity theft to deceive and exploit unsuspecting users. These attacks can access incidents, where attackers obtain access to systems and networks, services, databases, and other resources, or they can be reconnaissance-only attacks, where attackers attempt to track systems and networks in order to identify faults. Making a system or network resource unavailable or extremely slow is another type of attack that makes it nearly hard to utilize(7). The prevalence of fraudulent websites has significantly increased in recent years, with cyber attackers using these websites to gain unauthorized access to unsuspecting victims' devices and even convert numerous machines into bots for launching targeted attacks. Cybercriminals employ various tactics, such as creating counterfeit institutional websites that closely resemble legitimate ones. These domains are registered to closely resemble the real domains of trustworthy organizations like banks, universities, or well-known e-commerce platforms. They often make slight alterations, such as substituting letters with visually similar characters or adding additional words or characters to the domain name. For example, an attacker might register a domain like "bannk.com" instead of the legitimate "bank.com," exploiting the visual similarity to deceive users. As a result of this escalating problem, effective detection mechanisms need to be developed that are able to identify and classify malicious websites in a real-time common approach that has been used for so many years is the blacklist approach. Blacklisting approaches have long been used to detect malicious URLs by maintaining a list of known malicious URLs.When a new URL is encountered, it undergoes a check in the blacklist database. If the URL is present in the blacklist, it is marked as malicious, prompting a warning or alert. However, blacklisting has limitations as it cannot keep up with the constantly evolving landscape of malicious URLs, as new URLs can be generated daily, rendering it ineffective in detecting new threats(21). The utilization of algorithmic methods by attackers to generate new URLs poses a significant and critical challenge to the efficacy of blacklisting strategies, as it enables them to evade detection by existing blacklists. However, despite the inherent limitations and challenges associated with blacklisting, its simplicity, and efficiency have resulted in its extensive adoption within numerous anti-virus systems in contemporary times (22). Due to the challenges posed by these traditional methods to identify new URLs, many researchers have consistently used machine learning models to detect malicious websites in previous studies. In recent times, the application of machine learning techniques has demonstrated remarkable performance in data clas-

sification tasks(11).Machine learning techniques have demonstrated their versatility and applicability beyond the domain of malicious attacks. They have been widely used in various fields such as image processing (23), weather prediction (20), price prediction,(8), stock prediction,(24) and numerous other areas. In a study conducted by the authors(10), The researchers compared the performance of different machine learning models. Specifically, three supervised models (k-nearest neighbor, support vector machine, and naive Bayes classifier) and two unsupervised models (k-means and affinity propagation) were evaluated. The results showed that the supervised models slightly outperformed the unsupervised models in terms of performance. In another study(18), the problem of dataset size was tackled by combining association rule mining with various machine-learning models to classify URLs as malicious or benign, utilizing features extracted from the URLs. To address the challenges arising from class imbalance and a small dataset, the authors employed the Synthetic Minority Over-Sampling Technique (SMOTE). Assessing the models before and after class balancing showed significant performance improvements for most models when the dataset size was increased through class balancing. In the paper (18), the authors presented ten machine-learning models aimed at classifying malicious websites using lexical features on various datasets. Among these models, K-NN showed consistent performance across different datasets. Furthermore, Random Forests, Decision Trees, Logistic Regression, and Support Vector Machines consistently outperformed baseline models that simply predict every link as malicious across all metrics and datasets. Again, in this paper (3), the authors proposed a model involving three different cases for model creation.

2. Related works

Numerous studies have been conducted on the detection of malicious websites using different types of feature extraction and machine learning models. These papers, however, only use machine-learning approaches to a narrow set of attributes in order to detect dangerous websites. The importance of key elements generated from URLs, such as lexical-based, host-based, and content-based features, has been emphasized in recent papers for the purpose of employing machine learning to identify fake websites. To specifically collect important data for the analysis, lexical-based statistical features were derived from the raw URL string. The length, digit count, number of query arguments, encoding status, and host-based attributes, such as the host-name aspects of the URL, are a few examples. These give details about the website's host, such as the nation where it was registered, the properties of the domain name, access ports, named servers, connectivity speed, and the number of days the registration will last. Domain Name Systems (DNS), host, and network-based features are combined in network-based features. These characteristics include payload size, latency, DNS query data, site registration data, and WHOIS inquiry information for domain and IP address names. Some of these papers are reviewed and summarized below.(18) In (17), The authors employed a novel extended design attribute learning algorithm to analyze web page structures, content, appearances, and trustworthiness, thereby identifying malicious websites. The algorithm proposed in this study has undergone a large-scale experiment involving over 35,000 websites. The results show that the algorithm successfully detects over 83% of malicious websites while maintaining a low false-positive rate of only 2%. Additionally, the algorithm has the capability to incorporate user feedback, allowing for the prompt detection of newly discovered suspicious websites. This makes the algorithm effective in countering zero-day attacks. The research explores the application of linear and non-linear transformations to domain-specific numeric features. Four types of features are considered: URL domain, reputation, host, and lexical features. The lexical features include Boolean features encoded as numeric values. The study also examines the effects of five different space transformation methods and evaluates the performance of various classifiers. The evaluations carried out demonstrate that the use of space-transformation methods can greatly improve the ability of classifiers to detect malicious content. Additionally, the method is compared to other techniques such as feature selection and various feature transformation methods, including PCA, Auto-Encoders, and Variational Auto-Encoders. The study also explores the effectiveness of content-based detection using contextual word embeddings like GloVe and BERT. The results show that space-transformation methods are more effective in detecting malicious content, with an average accuracy of 98.5%. This method is also more efficient and scalable than the others, making it a valuable tool for security professionals (16), Among the approaches evaluated, the Random Tree method demonstrated the highest performance, achieving an accuracy of 90.81% and an F-measure score of 91.3%. The researchers analyzed the click traffic data of unsafe and non-malicious URLs, collected from Bitly, for over 600,000 short URLs. Using chi-square and ANOVA F-values, the researchers identified the most crucial lexical features from different datasets. Their objective was to discover the significant properties of URLs. They identified the top 60 features after extracting a total of 106 lexical features and used two score algorithms for feature selection on each dataset. Surprisingly, 47 features were discovered to be shared across the two datasets. The researchers utilized various machine learning techniques, including LR, KNN, SVM, and ensemble learning, to assess the efficacy of the model. They evaluated its performance using metrics such as the confusion matrix, accuracy, precision, recall, and F1 score. Notably, the model achieved a remarkable average accuracy of 96.60%.(4). (26), Conducted a study in which they devised a method for classifying URLs into normal or malicious categories using Neural Networks. The researchers utilized the CICANDMAL2017 dataset (13)and extracted eight lexical features from the URLs. A neural network that uses feed-forward with numerous hidden layers was used to determine the URL type. The URL attacks were successfully divided into five categories by the neural network algorithms: spam, phishing, defacement, malware, and benign. The researchers conducted multiple experiments with varying dataset sizes and numbers of hidden layers. They found that the best results were obtained when using 500 data rows and 25

hidden nodes. Notably, the neural network achieved an impressive accuracy rate of 98.48% when classifying URL types. (12) aimed to distinguish between malicious and benign URLs utilizing machine learning techniques.26,054 URLs made up the dataset in total, half of which have been classified as malicious. The dataset consisted of 41 features that were obtained from a combination of attributes associated with Alexa, networks, and content. To analyze user behavior on the internet, the Amazon service Alexa utilized big data and monitored domain traffic. The researchers used two methods, ANOVA and XGBoost algorithms, to select the most relevant features. The researchers used SVM, KNN, XGBoost, and DT, four well-known machine learning algorithms, for their investigation. These models' effectiveness was measured using assessment metrics like precision, recall, F1 score, and accuracy. Remarkably, when using the XGBoost model on the dataset, an accuracy of 99.98% was achieved using only the first 9 features. The related works evaluation revealed that (12) used CNN to produce the best results, with an accuracy percentage of 99.99 percent. However, their dataset only had 26,054 records, which is less than half as many as the 65,506 URLs that will be used in this research. Further, they use only lexical features of the URLs to make predictions. This paper will focus on investigating the combination of network-based and lexical-based on eight machine learning models to generalize across multiple datasets, also the paper will investigate the use of the ensemble method using the Stacking technique to build a new model called DKN that takes in the predictions of the base models which had a close performance then combine these results as a new dataset to train the proposed model DKN.

3. Methodology

The research can be group into (7) steps which include dataset selection, data preprocessing, feature extraction engineering (Lexical and Network features), machine learning modeling, ensemble modeling, cross-data analysis, and the outcome.

3.1. Dataset Selection

The dataset used in this experiment was acquired manually search through online repositories dedicated to datasets from https://data.gov/, Kaggle.com. These platforms offer a wide range of datasets across various domains and can be valuable resources for discovering potential datasets that align with your analysis requirements. In this study, two datasets were employed to evaluate the performance of a multi-machine learning-based malicious URL prediction system. The first dataset consisted of 420,464 samples, with 344,821 labeled as good URLs and 75,643 labeled as bad URLs. The second dataset, obtained from Kaggle.com, contained 450,176 samples, including 345,738 benign URLs and 104,438 malicious URLs.

Figure 1: Methodology

3.2. Data Preprocessing

In machine learning, pre-processing is the process of transforming raw data into a format that is suitable for training models. Real-world data is often noisy, missing values, or in an inconvenient format, making it unsuitable for machine learning directly. The purpose of data pre-processing is to clean and refine the data, enabling machine learning models to work effectively and accurately. Models can extract meaningful patterns and insights from datasets when they have been prepared adequately, enabling them to perform better and be more predictive.

3.2.1. Dataset imbalance

According to the study, two datasets were used and the number of malicious 75643 was smaller than the number of non-malicious 344821. Using another dataset, the study found that the number of benign was 345738, which was more than the number of malicious (104438). This bias may result in poor performance in the minority class because the ML algorithm is biased toward the majority class. Several techniques are considered to address the problem of the imbalanced dataset to avoid bias and improve the performance of the machine learning models. Oversampling the minority class, downsampling, and generating synthetic samples are some of the techniques. A popular technique is iterative minority oversampling (SMOTE), which interpolates among neighboring minority samples to generate synthetic samples. Random forest algorithms using an oversampling technique gave the best results after analyzing metrics for benign and malicious websites(23).

3.3. Feature Extraction

The process of feature extraction was used to transform unprocessed data into numerical representations while preserving

crucial data from the original dataset. The emphasis was on network-based characteristics and lexical features specifically. The textual characteristics of URLs, such as special characters, path extensions, path, host, and URL lengths, are captured by these features. Several Python libraries and Scikit Learn tools were used to do feature extraction. These libraries and tools made it easier to convert the URL data into numeric format, preserving the essential elements of the URLs while making processing and analysis more straightforward.

3.3.1. Lexical Feature engineering

Malicious URLs and domains possess unique features that are not found in benign URLs. For example, malicious websites often have longer URLs with extended query strings, which can redirect users to phishing websites. These phishing sites may attempt to collect sensitive information like credit card details. Additionally, the researcher observed that certain malicious URLs contain more than one protocol, essentially linking to other malicious websites as query strings. The extraction of the Protocol URLs feature proved beneficial in identifying certain characteristics unique to malicious URLs and domains that are absent in benign URLs. For instance, malicious website URLs tend to be longer, often containing extensive query strings that direct users to phishing websites. These phishing sites may deceive users into providing their credit card information. Additionally, some malicious URLs contain multiple protocols, effectively linking to other malicious websites. Prior to this study, the "Protocols Count" feature had not been explored in previous research. In this research, the focus was on analyzing the lexical content of each URL to capture a wide range of URL features. Among the lexical features investigated were Dot Count, URL Length, Digits Count, Special Characters Count, Hyphen Count, Double Slash Count, Single Slash Count, @ Sign Count, and, notably, Protocols Count, along with others. These features were carefully examined to gain insights into the characteristics of URLs and their potential impact on the model's performance.

3.3.2. Network-based feature engineering

The study used network-based features to detect malicious websites by analyzing the structural characteristics and behavioral patterns of the network traffic associated with those websites. This allowed for more accurate detection of malicious websites, as the network-based features were able to more accurately capture the malicious behavior of those websites. The study hopes that this method will be more effective than traditional methods in detecting malicious websites.

3.3.3. ANOVA

The research considers several techniques for feature selection. The statistical technique known as Analysis of Variance (ANOVA) is widely used for feature selection. The change in a response variable obtained under various circumstances defined by discrete components (classification variables) is analyzed using an ANOVA (1). According to (5) An ANOVA's test statistic

No.	Attribute	Description
1	path_length	URL Path length
2	count_/	Count "/" symbols in the URL
3	count_dot	Count "." symbols in the URL
4	Count_ampersand	Count "—" symbols in the URL
5	Count_@	Count "@" symbols in the URL
6	Count_dash	Count "-" symbols in the URL
7	count_equalto	Count "=" symbols in URL
8	Count_questionmark	Count "?" symbols in the URL
9	Count_semicolon	Count the ";" in the URL.
10	count_digit	Count total number of digits in the URL

Table 1: Lexical features present in the dataset

is as follows:

$$F = \frac{\sum_{j} (X_j - x)^2 / (k-1)}{\sum_{i} (X - X_i)^2 / (N-k)} \quad (1)$$

F is the test statistic for ANOVA, and it measures the variance between groups (malicious and non-malicious URLs) relative to the variance within each group.

$$\sum_{j} (X_j - X)^2$$

represents the sum of squares between groups, indicating the variation of URL features between the different groups (malicious and non-malicious). $(k-1)$ is the degree of freedom for the variation between groups.

$$\sum_{i} \sum_{j} (X - X_i)^2$$

represents the sum of squares within groups, indicating the variation of URL features within each group (malicious and non-malicious). $(N-k)$ is the degree of freedom for the variation within groups. X is the overall mean value of the URL feature across all URLs. X_i represents the average value of a specific URL feature within ith group, where the group can be either malicious or non-malicious URLs.

3.3.4. Chi-square

Chi-square is a statistical technique utilized by researchers to measure the disparity between observed results and predicted

No.	Attribute	Description
1	HTTPS	The website employs the HTTPS protocol
2	url_length	The length of a website's URL
3	ip_Presence	The website's IP address
4	geo_IP	The location where the website is physically located
5	js_length	length of the website's JavaScript code
6	Content	Raw web page content with JavaScript code
7	Keywords	Keywords presence in the URL
8	Lang	Presence of language code
9	.com	Presence of ". Com" in the URL (Top domain name)
10	org	The URLs(top domain name) includes ".org"
11	.net	Presence of ".net" in the URL (Top domain name)

Table 2: Network features present in the dataset

outcomes. Its primary purpose is to assess the independence between two variables. In other words, it helps to determine whether the observed values (O) and the expected values (E) are significantly different from each other(2). The chi-square statistic calculates this difference between the observed and expected values using the formula:

$$X^2 = \frac{(O_i - E_i)^2}{E_i} \quad (2)$$

X^2 is the Chi-square statistic, which will be used to measure the association between the URL features and the presence of malicious content. O_i represents the observed frequency of a specific URL feature in the dataset of known malicious URLs. E_i represents the expected frequency of the same URL feature based on a representative sample of URLs (including both malicious and benign ones).

is the Chi-square statistic, which will be used to measure the association between the URL features and the presence of malicious content.

3.3.5. Recursive Feature Elimination.

Recursive Feature Elimination is a step-by-step procedure used to select important features from a set of initial features. It begins with all features and then iteratively removes the least significant ones using a machine learning model to evaluate their importance. This process continues until a specific condition is met or the desired number of features is reached.

Algorithm 1 Rf-RfE: Selecting Best Features

Require: Training set Td Set of d features $Rf = \{f_1, \ldots, f_d\}$
Rank method (Td, fe)
Ensure: Ranked list of important features k

```
fe ← Rf
for i from 1 to b do
    Rank set fe using Rf(Td, fe)
    f* ← last ranked feature in fe
    R(a − i + 1) ← f*
    Remove last ranked feature from set fe
```

3.4. Machine Learning models

Extreme Gradient Boosting (XGBOOST), Light Gradient Boosting Machine (LIGHTGBM), Random Forest (RF), Decision Tree (DT), Logistic Regression (LR), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and a naive Bayes model (NB) were the eight models used in this study. These models were chosen for comparison because they are widely used and have experience with classification tasks.

a. **Extreme Gradient Boosting (XGBOOST)**
When handling diverse supervised learning issues, the Extreme Gradient Boosting Machine (XGBoost) gradient boosting algorithm is widely used in the data science community. The algorithm was created by Chen and Guestrin in 2016 and quickly gained popularity for its quick, precise, and simple-to-understand results. XGBoost combines tree-based models with gradient boosting to improve both accuracy and scalability. The researcher builds an ensemble of decision trees in order for XGBoost to function. In this model, each tree is trained to predict the residual error of the previous tree. A greedy algorithm is employed to find the optimal split point for each node in the tree. This algorithm thoroughly evaluates all candidate split points for each feature and selects the one that minimizes the target function the most. The effectiveness of XGBoost was demonstrated by Chen and Guestrin in 2016 through the use of multiple standard datasets. It was shown to outperform other well-known algorithms, such as Random Forest and Gradient Boosted Trees. Further extensive work by Chen and Guestrin (2016) and Greenwell et al. (2018) demonstrated the ability of his XGBoost to process large datasets with high-dimensional features. Moreover, XGBoost consistently achieves the best performance on various machine-learning tasks.

$$F^{(t)} = \sum_{i=1} f(b_{(t)}(t-1)) + f_t(a_i) + \Omega(f_t) \quad (3)$$

At each iteration t of the XGBoost algorithm, the predicted output $F^{(t)}$ is calculated by considering the actual output b_i, the predicted output from the previous iteration $y_i^{(t-1)}$the loss function's gradient with regard to the anticipated result $f(b_i, b_i^{(t-1)})$, and the contribution of [illegible] term $\Omega(f)$ is used to penalize complex trees and prevent overfitting during the training process.

b. **Light Gradient Boosting Machine (LIGHTGBM)**

LightGBM, also known as Light Gradient Boosting Machine, is a potent algorithm developed by Microsoft. It employs a gradient-boosting framework to iteratively train decision trees. One of its notable advantages is its emphasis on high efficiency and scalability, making it well-suited for large datasets and complex models. The algorithm's key features include leaf-by-leaf tree growth, histogram-based gradient computation, and GPU acceleration, making it versatile for various machine-learning tasks. In a comparative study conducted by Zhang et al. in 2020, the performance of three tree-based boosting algorithms, namely LightGBM, XGBoost, and CatBoost, was evaluated using multiple datasets. The study revealed that LightGBM and XGBoost exhibited comparable performance, but LightGBM demonstrated superior speed and memory efficiency. Furthermore, the research demonstrated that LightGBM is effective in feature selection and is capable of handling imbalanced datasets, making it a valuable tool in machine learning applications (6)

c. **Random Forest (RF)**

The Random Forest (RF) ensemble approach was invented by Leo Breiman. An extensive collection of decision trees is aggregated by the algorithm, which then creates a class based on either the average of the aggregated trees or a majority voting system. The benefit of the random forest is that it reduces overfitting results by combining the results of numerous decision trees. Random forests frequently use bagging or bootstrap aggregation to educate. Given a training T=t1 . . . t with responses Y=y1 bagging repeatedly (X times) select a random sample with the training setreplaced and trees fit to these samples: For x = 1. . . X: sample, with replacement, n training examples from X, Y; call these Tx,Yx. Train a classification or regression tree fb on Tx,Yx. Averaging the predictions from all the various regression trees on x' can be used to make predictions for unseen samples x' after training:

$$f = \frac{1}{A} \sum_{x=1} f_b(x') \quad (4)$$

d. **Decision Tree (DT)**

A well-liked machine learning approach for classification uses decision trees. A decision tree is an else-if rule classifier that creates a tree-like data structure by recursively dividing training data into smaller sets according to predetermined rules (9). The split condition is applied to each node to decide, and the data is divided into two or more subsets as the tree grows. Up until there are no longer any viable splits or branches, this process continues. The leaf nodes have a class or target variable assigned to them. A new label is then used as the foundation for classifying additional instances. There are three different types of nodes that are typically used to generate decision trees: Triangles are frequently used to represent end nodes, squares are frequently used to represent choice nodes and circles represent chance nodes. The decision tree has the advantage of being easy to understand in comparison to other regression and classification algorithms since it imitates human decision-making. Additionally, it can handle qualitative predictors without the need for a dummy variable. Its disadvantage is a propensity for overfitting. This research intends to explore the decision-based interactions among the malicious website features or factors used in the study aimed to predict whether a website is malicious or non-malicious.

c. **Logistic Regression (LR)**

Logistic regression, a well-known supervised machine learning algorithm, is extensively used for classification tasks. Its primary objective is to estimate the probability of an instance belonging to a particular class. The name "logistic regression" is derived from the utilization of the logistic or sigmoid function to transform the output of a linear regression function into probability estimates for various classes. This transformation allows logistic regression to be used effectively in classification algorithms. The difference between linear and logistic regression is that the output of linear regression is an arbitrary continuous value, whereas logistic regression predicts the probability of whether an instance belongs to a particular class. Logistic regression is used to predict the outcomes of a categorical dependent variable. Consequently, its results are categorical or discrete in nature. Instead of providing exact values like 0 or 1, true or false, or yes or no, logistic regression offers a range of probability values between 0 and 1. These probabilities indicate the likelihood of an instance belonging to a specific category. The logistic regression equation can be derived from the linear regression equation, but it incorporates the logistic or sigmoid function to map the continuous output of the linear equation into the probability range (0 to 1). This enables logistic regression to handle classification tasks effectively. The logistic regression equation can be derived from the linear regression equation, which represents a straight line as $y = b_0 + b_1 x_1 + b_2 x_2 + \ldots + b_n x_n$. However, in logistic regression, the output y must be constrained between 0 and 1 to represent probabilities. To achieve this, we use the logistic or sigmoid function. Therefore Logistics Regression

equation

$$\log \frac{y}{1-y} = b_0 + x_1 b_2 x_2 \dots b_n x_n \quad (5)$$

be calculated as

$$(x, u) = \sqrt{\sum_{i=1} (u - x)^2} \quad (6)$$

f. **Support Vector Machine (SVM)**

A support vector machine (SVM) is a supervised machine learning model that utilizes linear boundaries to segregate data into separate classes. The initial concept of the algorithm, as suggested by (15), aims to achieve accurate predictions by representing data as points in a dimensionless space. The primary objective is to find a hyperplane that maximizes the distance between two distinct support vectors, enabling effective class separation. This can be expressed as given an n-point training set of the form $(x_1, y_1) \dots (x_n, y_n)$, where y_i is either 1 or -1, each indicating the class to which point x_i belongs. Each x_i is a p-dimensional real vector. We want to identify the hyperplane with the largest margin separating the set of points $y_i = 1$ from the set of points $y_i = -1$, defined such that the distance between the hyperplane and the closest data point of either class is maximized.

g. **K-Nearest Neighbors (KNNs)**

KNN is an effective supervised learning method for multifold problems, including security technology (14). K nearest neighbors are based on grouping elements with the same properties. A test example's class category is determined based on its k nearest neighbors. The value of k for KNN depends on the size of the dataset and the nature of the classification problem (19). Figure 1 shows that KNN classifies targets based on their neighbors. Details explanation of KNN as follows Find the item that most closely resembles the test data for training. Data K calculates distance based on Euclidean distance For two elements in k-dimensional space, X = [x1, x2, . . . , xk] Euclidean distance based on y = [a1, a2, . . . , ak]. The two factors can

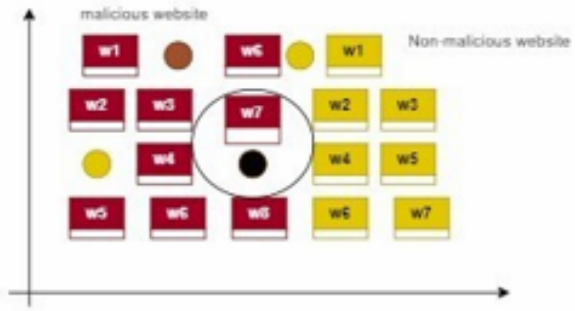

Figure 2: Methodology

After taking the K nearest neighbor the majority of the k-nearest neighbor will be considered as a class for the test data.

h. **Naïve Bayes (NB)**

The naive Bayes algorithm is a supervised learning technique that relies on Bayes' theorem to address classification tasks. It is especially useful for text classification, where datasets have high dimensions. The Naive Bayes Classifier is recognized as one of the simplest and most efficient classification methods, enabling the construction of rapid machine learning models that facilitate quick predictions. The term "Naive Bayes Algorithm" is composed of two words. i. Naïve because it always ensures that the occurrence of a certain feature is independent of the occurrence of other features. ii. Bayes because it depends on the principle of Bayes theorem The probability model for a classifier is a restrictive model over a reliant class variable. P(C— f1.....fn)

Using Bayes' theorem

$$P(C|f_1, \dots, f_n) = \frac{p(C)p[(f_1, \dots, f_n)/C]}{p(f_1, \dots, f_n)} \quad (7)$$

$$P(C|f_1, \dots, f_n)$$

is the probability that a URL belongs to the class C (i.e., whether it is malicious or not) given the observed features $f_1, \dots, f_n$ of the URL.

$$p(C)$$

is the prior probability of the class C, which represents the probability of a URL being malicious or non-malicious without considering any specific features.

$$p[(f_1, \dots, f_n)/C]$$

is the likelihood, which represents the probability of observing the specific set of features $f_1, \dots, f_n$ given that the URL belongs to class C.

$$p(f_1, \dots, f_n)$$

is the evidence or marginal likelihood, which is the overall probability of observing the features $f_1, \dots, f_n$ regardless of the class. In the context of detecting malicious URLs C

can represent the class of malicious URLs. $f_1, \ldots, f_n$ can represent various features extracted from the URL, such as URL length, presence of special characters, domain name, etc. The equation allows helps to calculate the probability that a URL is malicious given its observed features. The naive Bayes classifier makes predictions by comparing the probabilities for each class (malicious or non-malicious) and assigns the URL to the class with the highest probability.

3.5. Ensemble method

Ensemble methods refer to approaches that seek to enhance the accuracy of base models by merging multiple models rather than relying on a single one. In this research, a novel ensemble method called the DKN model was introduced, which combines the predicted outputs of various base models and utilizes a DKN model for the ultimate prediction.

3.5.1. Architecture of the DKN model

DKN model consists of two or more base or learner models that combine the predictions of the base models. The base models are at level 0 and the DKN model is at level 1. DKN model takes the base models that had a close range of accuracy and then combines their predictions and stored them as a new dataset to make a final prediction. The aim of this model is to compare the strength of the individual models against the combined model.

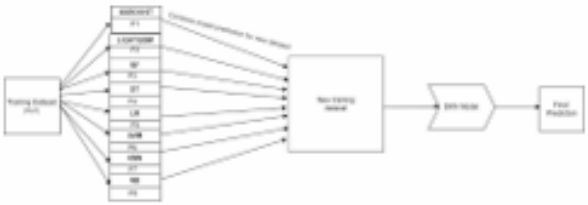

Figure 3: DKN model

4. Experimental Evaluation

In this section, the paper evaluates the effectiveness of the proposed eight models and also the ensemble DKN stacking model to combine some of the baseline models for prediction. The study used Python programming language. Python-based libraries are imported to aid in data processing and manipulation of data from Kaggle. The Python environment was set up using Anaconda. Jupiter Notebook Python coding platform was used. The Anaconda was installed and run on a Windows-based operating system platform and the hardware host was an x64-based Quad-core Processor machine with an 8GB installed memory.

4.1. Dataset Source

In this study, two datasets were obtained from kaggle.com, each comprising URLs. The first dataset consisted of 450,176 URLs, among which 345,738 were categorized as benign and 104,438 as malicious. The second dataset included 420,464 URLs, with 344,821 being benign and 75,643 being identified as bad (malicious). To prepare the datasets for analysis, pre-processing techniques such as recursive feature elimination, chi-square, and Analysis of Variance (ANOVA) were employed to normalize the data and extract 35 important features, including both lexical and network-based features, along with one target label. Subsequently, the dataset was divided into a training set with 438,176 samples of URLs and a testing set with 12,000 samples of URLs, each labeled as benign or malicious. These prepared datasets were then used to train various machine learning models, namely RF, DT, LR, NB, KNN, SVM, XGBOOST, and LIGHTGBM. Finally, the model predictions from these base models were combined to train the DKN model, which was then utilized to make the final prediction, effectively detecting whether a given URL is benign or malicious.

ut[60]:

	Unnamed: 0	url	label	result
0	0	https://www.google.com	benign	0
1	1	https://www.youtube.com	benign	0
2	2	https://www.facebook.com	benign	0
3	3	https://www.baidu.com	benign	0
4	4	https://www.wikipedia.org	benign	0

Figure 4: Data visualization

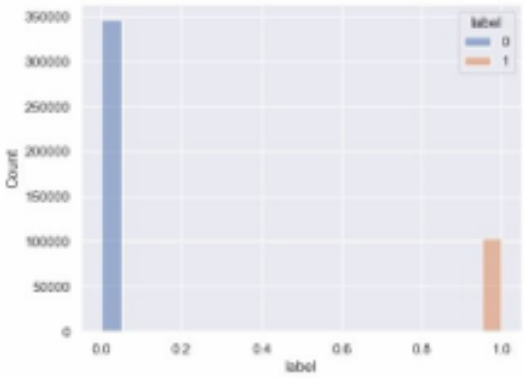

Figure 5: Benign and malicious URL

4.1.1. Feature Order of importance

After the dataset went through preprocessing these features were considered by the researcher as the most important features that can contribute to the proposed models to make the correct prediction.

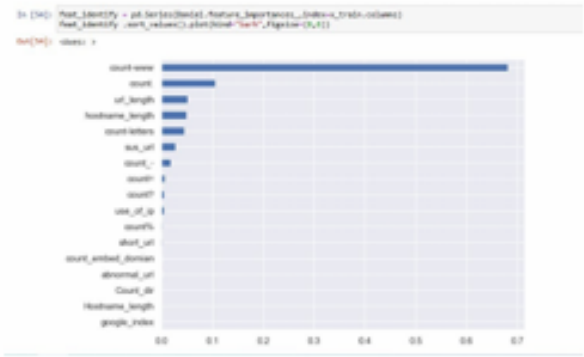

Figure 6: Benign and malicious URL

4.2. Evaluation Metrics

A confusion matrix is a valuable tool for evaluating the effectiveness of a classification model. It is represented as an n x n matrix, where n denotes the number of classes in the problem domain. The rows are labeled with the actual classes, and the columns are labeled with the predicted classes. In the context of a binary classification problem with two states, a 2 x 2 confusion matrix is generated. Each cell in the matrix represents the number of observations falling into specific categories. The documentation for presenting confusion matrices in Scikit-learn follows a similar style, providing a comprehensive overview of the classifier's performance. This matrix allows us to assess how well the model performs in predicting each class and helps in understanding potential areas for improvement.

Table 3: Confusion Matrix

Actual/Predicted Class	Predicted Negative (0)	Predicted Positive (1)
Actual Negative (0)	True Negative (TN)	False Positive (FP)
Actual Positive (1)	False Negative (FN)	True Positive (TP)

4.2.1.

The figure shows the confusion matrix pattern with rows representing the actual class and the columns representing the predicted class. TP=Confusion[1,0], TN = Confusion [0,0], FP = Confusion[0,1], FN=Confusion[0,1], FN = Confusion[1,0] Confusion [TN, FP, FN, TP].

i. True Positive (TP): To correctly predict benign websites as benign.

ii. False Positive (FP): incorrectly predicting benign website as malicious

iii. True Negative (TN): correctly predicting malicious websites as malicious. iv. False Negative (FN): incorrectly predicting malicious websites as benign.

Performance indicators derived from the confusion matrix are displayed in the table below.

Table 4: Performance Measures Formulas

Measure Metrix	Formula
Accuracy	$\frac{TP+TN}{TP+FP+FN+TN}$
Precision	$\frac{TP}{TP+FP}$
Recall	$\frac{TP}{TP+FN}$
F-Score	$\frac{2\times(Recall\times Precision)}{Recall+Precision}$

4.3. Evaluation of individual models

5. Results and Discussion

To showcase the efficiency of the proposed model, the study conducted validation on a test set consisting of 135,052 URLs using the eight models. The evaluation of these models was performed using multiple metrics, including accuracy, precision, recall, and F1 score. The classification results for all the models are presented in Table 5, providing an overview of their respective performance measures. The result shows that all the models gave a very close value. Looking at Table 5, XGBOOST and Random Forest achieved slightly higher values than the other six models, with XGBOOST achieving an accuracy of 97.2% and the Random Forest achieving 97.2%. The highest precision was obtained by XGBOOST and LIGHTGBM with 99% followed by Logistics Regression and Support vector machine with 98% while the lowest precision value was KNN with 94%. However, Logistics Regression, Support vector machine, xgboost, and LIGHTGBM achieved the highest recall value with 100% while 99% by Random forest and 98% by Decision Tree, and the rest of the models obtained the lowest recall. Finally, KNN achieved the lowest F-score while the other seven (7) models achieved the highest value range from 92% to 98%. However, the study proposed an Ensemble DKN model using a stacking technique to combine some of the baseline models to increase the accuracy of the prediction. Table 7, the DKN model combines DT and RF and the accuracy was 99.1%, 99.3% for combining LR + KNN+SVM. The parameters applied to the model are demonstrated in Table 6. The DKN model uses majority voting to combine individual base classifiers. This voting scheme helps in mitigating the risk of making incorrect predictions rather increase the overall predictive accuracy. Also, the DKN model can incorporate additional base classifiers which improves ensemble performance. The disadvantage of the model is the training time since combining multiple base classifiers which is computationally expensive therefore it takes about 2 hours before the model can predict.
Finally, a combined accuracy of 97% was achieved by combining three models: LR + KNN + SVM. This ensemble approach demonstrated improvement compared to the individual models in terms of accuracy. In the context of classifying websites as

Table 5: Evaluation of the individual models

Label	Classifier	Precision	Recall	F1-score	Accuracy (%)
Benign	Random Forest	0.97	0.99	0.98	97.2%
malicious		0.97	0.91	0.94	
Benign	Decision Tree	0.97	0.98	0.98	97%
malicious		0.95	0.91	0.93	
Benign	Logistics Regression	0.96	1.00	0.98	96.5%
malicious		0.98	0.86	0.92	
Benign	Support Vector Machine	0.96	1.00	0.98	96.4%
malicious		0.98	0.86	0.92	
Benign	K-Nearest Neighbors	0.94	0.99	0.96	94.3%
malicious		0.95	0.80	0.87	
Benign	XGBOOST	0.97	1.00	0.98	97.2%
malicious		0.99	0.89	0.94	
Benign	LightGBM	0.97	1.00	0.98	97%
malicious		0.99	0.88	0.93	
Benign	Naive Bayes (NB)	0.96	0.97	0.97	95.1%
malicious		0.90	0.88	0.89	

Table 6: Model Parameters

Model	Parameter	Value
DKN model	N_estimators	50, 100, 150
	Max_depth	10, 20, 30
	Min_samples leaf	1, 2, 4
	Max_features	Sqrt, log2

Table 7: DKN model combining Random forest classifier and Decision tree classifier

Classifier Name	Accuracy (%)
Random Forest	97.2%
Decision Tree	97%
DKN model (RF + DT)	**99.1%**

Table 8: DKN model combining Logistic Regression, KNeighborsClassifier, and SVM

Classifier Name	Accuracy (%)
Logistic Regression	96.5%
KNeighborsClassifier	94.3%
SVM	96.4%
DKN model (LR + KNN + SVM)	**97%**

benign or malicious, the correct identification of benign websites is considered true positives (TP), while the correct recognition of malicious websites is referred to as true negatives (TN). Conversely, the incorrect identification of malicious websites as benign is termed false negatives (FN), and the incorrect classification of benign websites as malicious is labeled as false positives (FP). All eight (8) models used in this study were trained independently, and then were assessed based on their confusion matrices and later combined some of the models due to their predicted accuracy range Models can be more accurate if it has more TP and TN or fewer FN and FP (25). These confusion matrices are also used to calculate metrics such as precision, recall, and the F1 score. From these matrices, it can be seen that after combining the baseline models they performed better than the individual models. The instances of TP and TN were more, compared with the individual models.

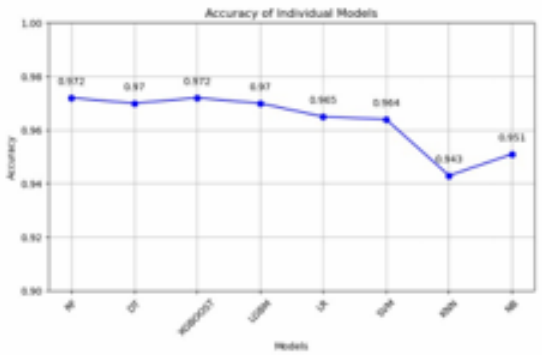

Figure 7: Individual models performance

6. Conclusion and Future work

As the number of websites increases, cybercriminals are using more harmful web addresses to sneak dangerous code into people's devices. This can seriously harm the security and functionality of computer systems. So, it's crucial to find better ways to detect and stop these attacks as they become more advanced. Using smart methods to solve this problem has become a major area of research, and many scientists are working on creating models to classify and identify these harmful URLs. Lots of researchers are working on creating models that can classify and identify harmful web addresses. The main focus of this research paper was twofold: firstly, it involved the creation

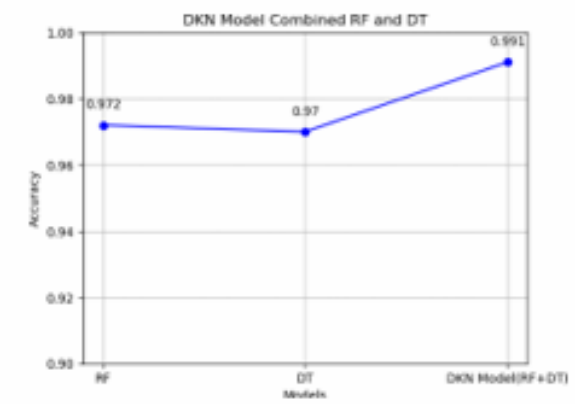

Figure 8: DKN model performance after combining 2 base models

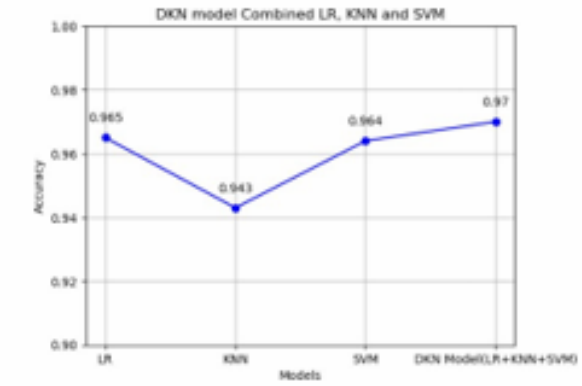

Figure 9: DKN model performance after combining 3 base models

and selection of meaningful features for analysis. Secondly, it entailed the development of multiple models, including RF, DT, LR, NB, KNN, SVM, XGBOOST, and LIGHTGBM. Additionally, the study introduced an ensemble approach known as the DKN model. This ensemble method utilized the predictions of the base models and combined their accuracies to create a new dataset, which was then used to train the DKN model for making predictions on this novel dataset. The performance evaluation of the models in the study involved several matrices, including accuracy, precision, recall, and F-score. For experimentation, the final dataset comprised a combination of lexical and network-based features. The selection of these features was carried out using three feature selection techniques: ANOVA, chi-square, and Recursive Feature Elimination (RFE). These techniques were instrumental in identifying the most relevant and informative features for the analysis. The results of the individual models show that RF and XGBOOST obtained 97.2% each followed by DT and LIGHTGBM 97%. However, the DKN model combines RF and DT which obtained an accuracy of 99.1%, and also combined LR, KNN, and SVM obtained an accuracy of 97% which serves improvement on the individual classifiers. Based on the analysis the study concluded that combining two or more base models helps to detect malicious websites since it gives a high-accuracy classification. The research contributes to the field of cybersecurity by providing valuable insights to security experts and academic researchers. It helps them understand the most effective approaches for detecting malicious websites in different situations while maintaining important qualities like speed, high accuracy, and low false-negative rates. This knowledge can be used to enhance the performance and reliability of malicious website detection systems. Future work can consider extracting content-based features and adding to the above features. Again, other research can consider the running time of the DKN model by reducing the time it takes when combining three or more models.

References

[1] Malak Aljabri, Fahd Alhaidari, Rami Mustafa A Mohammad, Samiha Mirza, Dina H Alhamed, Hanan S Altamimi, Sara Mhd Ghosn, et al. An assessment of lexical, network, and content-based features for detecting malicious urls using machine learning and deep learning models. *Computational Intelligence and Neuroscience*, 2022, 2022.

[2] Aliuch Altaher. Phishing websites classification using hybrid svm and knn approach. *International Journal of Advanced Computer Science and Applications*, 8(6), 2017.

[3] Jo Simon Aaisha, Jose Gouveia, Dan Jacinto, and Joel De Goma. Malicious url classification using extracted features, feature selection algorithms, and machine learning techniques. In *Proc. Int. Conf. Ind. Eng. Oper. Manag*, number 2016, pages 2421–2429, 2021.

[4] Mohammad Ayoob, Ashraf Ahmad, and Firas Alghanim. Enhancing detection of malicious urls using boosting and lexical features. *Intelligent Automation & Soft Computing*, 31(3), 2022.

[5] Hyunjung Choi, Bin B Zhu, and Heejo Lee. Detecting malicious web links and identifying their attack types. In *2nd USENIX Conference on Web Application Development (WebApps 11)*, 2011.

[6] Selcuk Demir and Emrehan Kutlug Sahin. Predicting occurrence of liquefaction-induced lateral spreading using gradient boosting algorithms integrated with particle swarm optimization: Performance comparison and feature selection. *Acta Geophysica*, 18(6):3403–3419, 2023.

[7] Marcelo Ferreira. Malicious url detection using machine learning algorithms. *Proceedings of the Digital Privacy and Security Conference*, 503(3):3204–3215, 2019.

[8] Enis Gegic, Becir Isakovic, Dino Keco, Zerina Masetic, and Jasmin Kevric. Car price prediction using machine learning techniques. *TEM Journal*, 8(1):113, 2019.

[9] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[10] Talha Iqbal, Adnan Elahi, William Wijns, and Atif Shahzad. Exploring unsupervised machine learning classification methods for physiological stress detection. *Frontiers in Medical Technology*, 4:782756, 2022.

[11] Sanaa Kaddoura. Classification of malicious and benign websites by network features using supervised machine learning algorithms. In *2021 8th Cyber Security in Networking Conference (CSNet)*, pages 36–40. IEEE, 2021.

[12] Tie Li, Gang Kou, and Yi Peng. Improving malicious urls detection via feature engineering: Linear and nonlinear space transformation methods. *Information Systems*, 91:101494, 2020.

[13] Chaochao Luo, Shen Su, Yanbin Sun, Qingji Tan, Meng Han, and Zhihong Tian. A convolution-based system for malicious urls detection. *Computers, Materials & Continua*, 62(1), 2020.

[14] Chaochao Luo, Shen Su, Yanbin Sun, Qingji Tan, Meng Han, and Zhihong Tian. A convolution-based system for malicious urls detection. *Computers, Materials & Continua*, 62(1), 2020.

[15] Satar Mahdevari, Hamid Shirzad Haghighat, and Seyed Rahman Torabi. A dynamically approach based on svm algorithm for prediction of tun-

url convergence during excavation. ***Tunnelling and Underground Space Technology***, 38:59–68, 2013.

[16] Mohammad Saiful Islam Mamun, Mohammad Ahmad Rathore, Arash Habibi Lashkari, Natalia Stakhanova, and Ali A Ghorbani. Detecting malicious urls using lexical analysis. In ***Network and System Security: 10th International Conference, NSS 2016, Taipei, Taiwan, September 28-30, 2016, Proceedings 10***, pages 467–482. Springer, 2016.

[17] Or Naim, Doron Cohen, and Irad Ben-Gal. Malicious website identification using design attribute learning. ***International Journal of Information Security***, pages 1–11, 2023.

[18] Adebayo Oshingbesan, Courage Ekoh, Chukwuemeka Okobi, Aime Munezero, and Kagame Richard. Detection of malicious websites using machine learning techniques. ***arXiv preprint arXiv:2209.09650***, 2022.

[19] A Saleem Raja, R Vinodini, and A Kavitha. Lexical features based malicious url detection using machine learning techniques. ***Materials Today: Proceedings***, 47:163–166, 2021.

[20] Parth Shah, A Yashwant, Umang Khaitan, and S Kayalvizhi. Weather management system using machine learning algorithm and iot. In ***2023 International Conference on Recent Advances in Electrical, Electronics, Ubiquitous Communication, and Computational Intelligence (RAEEUCCI)***, pages 1–4. IEEE, 2023.

[21] M Sridevi and KVN Sunitha. Malicious url detection and prevention at browser level framework. ***Int J Mech Eng Technol***, 8(12):536–541, 2017.

[22] Linhao Tang and Qusay H Mahmoud. A survey of machine learning-based solutions for phishing website detection. ***Machine Learning and Knowledge Extraction***, 3(3):672–694, 2021.

[23] Intesar Ul Hassan, Raja Hashim Ali, Zain Ul Abideen, Talha Ali Khan, and Rand Kouatly. Significance of machine learning for detection of malicious websites on an unbalanced dataset. ***Digital***, 2(4):501–519, 2022.

[24] Mehar Vij, Deeksha Chandola, Vinay Anand Tikkiwal, and Arun Kumar. Stock closing price prediction using machine learning techniques. ***Procedia computer science***, 167:599–606, 2020.

[25] Adarsh Vulli, Parvathaneni Naga Srinivasu, Madipally Sai Krishna Sashank, Jana Shafi, Jaeyoung Choi, and Muhammad Fazal Ijaz. Fine-tuned densenet-169 for breast cancer metastasis prediction using fastai and 1-cycle policy. ***Sensors***, 22(8):2988, 2022.

[26] Wei Wei, Qiao Ke, Jakub Nowak, Marcin Korytkowski, Rafał Scherer, and Marcin Woźniak. Accurate and fast url phishing detector: a convolutional neural network approach. ***Computer Networks***, 178:107275, 2020.