PHISHING ATTACK DETECTION USING GRADIENT BOOSTING

By

ASLIN SUSHMITHA R.

Department of Information Technology, Noorul Islam College of Engineering, Thuckalay, Tamil Nadu, India.

https://doi.org/10.26634/jdf.2.1.20840

Date Received: 16/05/2024 *Date Revised: 25/05/2024* *Date Accepted: 15/06/2024*

ABSTRACT

Phishing is a prevalent cyber attack that uses deceptive websites to trick individuals into revealing personal information. These sites mimic legitimate ones to steal data such as usernames, passwords, and financial details. Detecting phishing is crucial, and machine learning algorithms are effective tools for this task. Attackers favor phishing due to its effectiveness in tricking victims with authentic-looking yet malicious links, which can breach security measures. This method employs machine learning to innovate phishing website detection. However, attackers can manipulate features like HTML, DOM, and URLs using web scraping and scripting languages. A new approach using machine learning classifiers tackles these threats by analyzing internet URLs and domain names. A dataset sourced from globally recognized intelligence services and organizations facilitates streamlined feature extraction, reducing processing overhead by prioritizing URL and domain name traits. The Gradient Boosting Classifier is used on an 11,055-instance dataset with thirty-two features to classify phishing URLs, demonstrating superior accuracy compared to methods like Random Forest. Gradient boosting is highly effective across various machine learning tasks, leveraging aggregated weak learners such as decision trees for strong predictive accuracy. Its suitability for handling imbalanced datasets makes it particularly effective for phishing detection, which is crucial for distinguishing between legitimate and malicious URLs. This method enhances accuracy by extracting and comparing distinct characteristics of legitimate and phishing URLs. By focusing on URL and domain name attributes, a more effective approach to identifying phishing attempts in cybersecurity is proposed.

Keywords: Machine Learning, Gradient Boosting Classifier, Phishing Detection, Machine Learning for Cybersecurity, Fraud Detection, Phishing Attack Prevention.

INTRODUCTION

The aim of this paper is to predict whether a URL is malicious or not based on several features such as HTTPS, URL length, URL shortening, redirection, and website forwarding. By using the listed datasets, this paper determines whether a URL is legitimate or not in the future. This could help protect individuals from phishing attacks by predicting potential threats. Developing an effective system to identify and flag potential phishing websites enhances overall cybersecurity measures for users and organizations. Leveraging advanced machine learning techniques, such as Gradient Boosting, aims to improve the accuracy and reliability of distinguishing between phishing and legitimate websites. By detecting and notifying users about potentially malicious websites, the paper seeks to protect individuals from phishing attacks and raise awareness about online security threats. The paper also aims to contribute to the advancement of technology in cybersecurity by utilizing sophisticated algorithms and methodologies for more robust phishing detection systems. By proactively identifying and preventing access to deceptive websites, it seeks to mitigate the impact of phishing attacks on individuals, businesses, and institutions. Additionally, the paper evaluates the performance of the Gradient Boosting Classifier in comparison to other methods, validating its effectiveness in real-world scenarios.

Phishing attacks have become more complex, incorporating elements such as personalized messages and advanced social engineering to deceive recipients. If phishing attacks are detected earlier, their negative effects can be mitigated through appropriate measures. Technology can be used reliably and effectively for early detection. A data-driven model has been developed using machine learning. This type of model relies heavily on available data and employs statistical or machine learning techniques to identify patterns or make predictions without depending on explicit analytical equations or theories. These models can predict whether a URL is malicious or not. Phishing is considered one of the most dangerous forms of social engineering, so early detection is particularly beneficial. Phishing attacks occur through deceptive emails, websites, or messages that appear to come from legitimate sources. Criminals create fraudulent replicas of actual websites and email accounts, including real company logos and slogans. When a user clicks on a link provided by these attackers, the hackers gain access to the user's private information, including bank account details, personal login passwords, and images. Phishing can be categorized into various types based on the methods used, the targets, or the specific approach of the attack. Some common categories or classifications of phishing:

- Email Phishing
- Spear Phishing
- Clone Phishing
- Vishing (Voice Phishing)
- Smishing (SMS Phishing)
- Pharming
- Whaling

Phishing attacks exploit human psychology and vulnerabilities to trick individuals into taking actions that compromise their security. As technology advances, attackers continuously adapt their strategies, making it crucial for individuals and organizations to stay vigilant, employ security measures, and educate themselves to recognize and prevent these attacks.

In the second quarter of 2023, the Anti-Phishing Working Group (APWG) observed 1,286,208 phishing attacks. This represents a decrease from the 1,624,144 attacks recorded in Q1 2023, which was a record high. The average wire transfer amount requested in Business Email Compromise (BEC) attacks in Q2 2023 was $293,359, up 57 percent from Q1's average of $187,053. The financial sector remained the most-attacked sector, accounting for 23.5 percent of all phishing attacks, while attacks against online payment services made up another 5.8 percent.

Voice-mail phishing, or vishing, also continued to rise. Figure 1 shows the most targeted industries in the second quarter of 2023. It is increasingly crucial to predict phishing attacks at an early stage. Early prediction of phishing can reduce the risk of falling victim to schemes, such as clicking on malicious URLs or providing sensitive bank and credit card information. To address this, a machine learning approach has been introduced to predict malicious URLs in their initial phase. The Gradient Boosting classifier is employed in this paper for detecting phishing URLs (Kabla et al., 2022). This classifier enhances the performance of the model, and its accuracy is compared with that of other models. The method

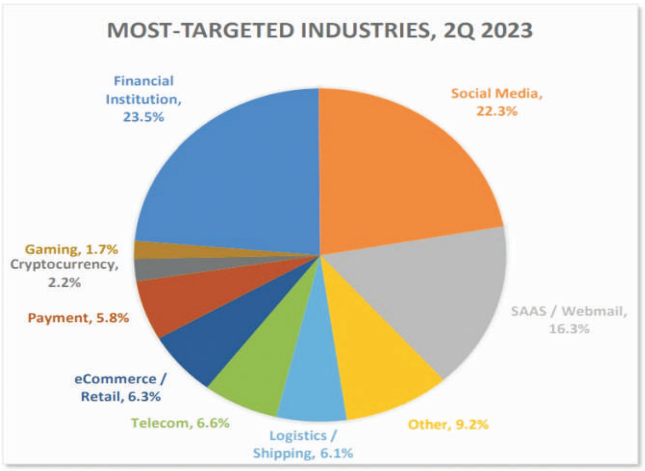

Figure.1. APWG Report

demonstrates greater accuracy compared to alternative approaches. The proposed method achieves higher accuracy when comparing with other learning techniques in the results. Gradient Boosting classifier is the proposed model which has been introduced as a modification for the existing one.

1. Related Works

Phishing detection using a Gradient Boosting Classifier represents a pioneering approach in bolstering cybersecurity measures against deceptive online threats. Leveraging the power of machine learning, this paper focuses on harnessing the capabilities of the Gradient Boosting Classifier to effectively discern phishing websites from legitimate ones. By employing a sophisticated algorithmic framework, the system aims to accurately identify suspicious online entities, providing users and organizations with an advanced defense mechanism against malicious online activities. This groundbreaking initiative not only strives to enhance the accuracy of phishing detection but also emphasizes user-centric accessibility, ensuring a user-friendly and efficient platform to safeguard against evolving cyber threats (Capuano et al., 2022; Gupta et al., 2023). Studies focused on the broad and comprehensive review of the state of the art in the field by discussing the main challenges and findings and is centered around three important categories of detection approaches, namely, list- based, similarity-based and machine learning-based. List-based detection methods are generally simple and fast, although not always effective (Salloum et al., 2022). Their reactive nature – coupled with the delays in identifying new phishing campaigns and updating the lists accordingly. Unlike list-based detection methods, similarity-based methods can generally cope with zero-hour attacks, although they are slower and more complex to implement. These methods are also storage intensive. Machine learning-based methods are generally fast and effective and allow on-the-fly detection of phishing web pages. The main limitation of list based method is they're unable to cope with zero-hour attacks and similarity-based approaches is its effectiveness, subjectivity, speed, storage (Zieni et al., 2023).

Other studies in phishing detection use PhiKitA, a novel dataset containing phishing kits and phishing websites generated using these kits. These studies have applied MD5 hashes, fingerprints, and graph representation DOM algorithms to analyze the PhiKitA dataset. In the familiarity analysis, they found evidence of different types of phishing kits and a small phishing campaign. They achieved overall accuracy, demonstrating that phishing kit data contain useful information for classifying phishing attempts. Distribution sites and Telegram, along with checking phishing website directories, can help collect phishing kits, as it is a common oversight for phishers to leave the phishing kit file on the web server. Therefore, these files can be downloaded from the server by checking directory contents. This approach has an advantage over others, as phishing kits collected from phishing attacks can provide information that relates to the attack or the phishing kit source itself. However, the hash-based method achieved minimal accuracy, indicating that this algorithm does not extract sufficient information to effectively distinguish between phishing websites and their phishing kit sources (Castaño et al., 2023).

Other studies focus on a feature-free method using the Normalized Compression Distance (NCD) for detecting phishing websites. This parameter-free similarity measure computes the similarity of two websites by compressing them, thereby eliminating the need for feature extraction. The method examines the HTML of webpages and compares their similarity with known phishing websites to classify them (Drew & Moore, 2014). They used the Furthest Point First algorithm for phishing prototype extractions. Their proposed method significantly outperforms others in detecting phishing websites and eliminates the need to retain long-term data. The advantages of using a feature-free approach include the system's ability to adapt to changes in phishing behavior or data representation. To build a detection system that can continuously learn and gain knowledge from the previous learning process resulting in a continuously learning system (El Aassal et al., 2020). There are limitations to this method, it may not be able to detect zero-day attacks or new variants of phishing websites, as the method is primarily focused on detecting variations of known attacks (Purwanto et al., 2022).

Several other studies presented Phish Farm, which is a scalable framework for methodically testing the resilience of anti-phishing entities and browser blacklists against attackers' evasion efforts. The Phish Farm framework is designed for continuous monitoring of the ecosystem and can be extended to test future state-of-the-art evasion techniques used by malicious websites. The study believes that continuous and close collaboration between all anti-abuse entities can lead to a deep understanding of current threats and the development of intelligent defenses (Karim et al., 2019). This work focuses on optimizing controls and delivering the possible long-term protection for phishing victims. The experiments revealed shortcomings in the current infrastructure, which allow some phishing sites to go unnoticed by the security community while remaining accessible to victims. They also discovered that blacklisting did not function as intended in popular mobile browsers like Chrome, Safari, and Firefox, which left users of these browsers particularly vulnerable to phishing attacks (Oest et al., 2019).

Several other studies focused on using multi-stage approach to detect phishing e-mail attacks using natural language processing and machine learning. This approach consists of feature engineering within natural language processing, feature selection, feature extraction, improved learning techniques (Razaque et al., 2021). Two methods have been applied on this approach. Exploiting the Chi-Square statistics and the Mutual Information to improve the dimensionality reduction, and the Principal Component Analysis (PCA) and Latent Semantic Analysis (LSA). These methods combined with the XGBoost and Random Forest machine learning algorithms, lead to an overall success rate. The proposed multi-stage phishing detection approach outperforms state-of-the-art schemes for an accredited data set, requiring a much smaller number of features and presenting lower computational cost. Feature engineering has data quality issues, such as missing values, outliers, and errors, can affect the quality of features and performance (Gualberto et al., 2020).

2. Proposed System

This paper aims to ensure the rapid and precise identification of potential phishing threats. The system offers an intuitive user interface, enabling the generation of efficient outputs crucial for detecting and responding to phishing attempts. The paper has been developed as a website platform for all users. This interactive and responsive website will be used to determine whether a website is legitimate or a phishing attempt. It is built using various web design technologies, including HTML, CSS, JavaScript, and the Flask framework in Python. The basic structure of the website is created with HTML. CSS is used to add effects and enhance the site's attractiveness and user-friendliness. It is important to note that the website is designed for all users, so it must be easy to operate, and no user should encounter difficulties while using it.

2.1 Data Collection

This is the first real step towards the development of a machine learning model: collecting data. Several techniques can be used to collect data, such as web scraping and manual intervention. The dataset is sourced from the popular dataset repository Kaggle, and from various intelligent sources and phishing detection databases (Kara et al., 2022).

2.2 Dataset

A dataset with 11,055 data is used in which the data without phishing websites, and data with phishing websites to be labelled according to thirty-two predetermined features to analyse and capture phishing URLs (Wei & Sekiya, 2022).

2.3 Data Preparation

The data is wrangled and prepared for training. It is cleaned as needed, including the removal of duplicates, correction of errors, dealing with missing values, normalization, and the performance of data type conversions. The data is randomized to eliminate the effects of the specific order in which it was collected and prepared. The data is visualized to help detect relevant relationships between variables, class imbalances (bias alert!), or to conduct other exploratory analyses. The data is split into training and evaluation sets (Tsinganos et al., 2022).

2.4 Model Selection

Gradient Boosting Classifier machine learning algorithm is used in this study. The training accuracy 98.9% is achieved, so this algorithm is implemented.

2.5 Test Accuracy

In the actual dataset, only 30 features are chosen. On the test set, the accuracy of 07.6% has been achieved.

3. System Architecture

Dataset: Initially, the proposed work is trained using a dataset that consists of various features. This dataset does not contain any website URLs.

Preprocessing and Feature Selection: The dataset includes features such as Long URL, Short URL, Redirecting, and HTTPS, which are considered when determining whether a website URL is legitimate or phishing.

Training Dataset: This dataset includes thirty-two predetermined features and 11,055 rows of data, which will be uploaded into the proposed system.

Gradient Boosting: The proposed system is developed using the Gradient Boosting Classifier. This algorithm combines several weak learning models to create a powerful predictive model, and it works effectively for identifying phishing URLs.

Testing Data: After training with the dataset, the classifier evaluates new URLs based on the training data.

Results: If the URL is determined to be phishing, the system will alert the user that the website is fraudulent. If the URL is identified as legitimate, the system will confirm that the website is genuine. Figure 2 shows the system architecture.

4. Algorithm

The algorithm that has been used in the proposed work is Gradient Boosting Classifier Algorithm. The principle

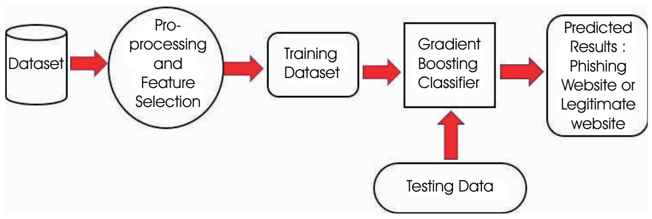
Dataset
Pro-processing and Feature Selection
Training Dataset
Gradient Boosting Classifier
Predicted Results : Phishing Website or Legitimate website
Testing Data

Figure 2. System Architecture

behind boosting algorithms is to first build a model on the training dataset, and then build a second model to rectify the errors present in the first model. There are n data points and 2 output classes (0 and 1). A model needs to be created to detect the class of test data. The approach involves randomly selecting observations from the training dataset and feeding them to model 1 (M1). Initially, all observations are assigned an equal weight, meaning each has an equal probability of being selected. In ensembling techniques the weak learners combine to make a strong model so here M1, M2, M3.... Mn all are weak learners. Since M1 is a weak learner, it will surely misclassify some of the observations. Before feeding the observations to M2, update the weights of the observations that are wrongly classified. When an observation is wrongly classified, its weight gets updated and for those which are correctly classified, their weights get decreased. The probability of selecting a wrongly classified observation gets increased hence in the next model only those observations get selected which were misclassified in model 1. Similarly, it happens with M2, the wrongly classified weights are again updated and then fed to M3. This procedure is continued until and unless the errors are minimized, and the dataset is predicted correctly. When the new datapoint comes in (test data) it passes through all the models (weak learners) and the class which gets the highest vote is the output for the test data.

4.1 Gradient Boosting Algorithm

This algorithm is to build models sequentially, with each subsequent model aiming to reduce the errors of the previous one. Achieving this involves constructing a new model based on the errors or residuals of the previous model. When the target column is continuous, a Gradient Boosting Regressor is used. For classification problems, a Gradient Boosting Classifier is employed. The primary difference between the two lies in the "loss function." The goal is to minimize this loss function by adding weak learners using gradient descent. Since it is based on the loss function, regression problems use different loss functions, such as Mean Squared Error (MSE), while classification problems use others, like log-likelihood.

Understanding the Gradient Boosting Algorithm with an example can clarify the concept. For this example, the target column is continuous, so a Gradient Boosting Regressor will be used.

Step 1: The first step in gradient boosting is to build a base model to predict the observations in the training dataset. For simplicity, the average of the target column is taken and assumed to be the predicted value. The rationale behind using the average of the target column is based on mathematical reasoning. Mathematically the first step can be written as:

$$F_0(x) = \arg_\gamma \min \sum_{i=1}^{n} L(y_i, \gamma) \quad (1)$$

Here L is the loss function, Gamma is the predicted value, and argmin indicates finding a predicted value (Gamma) for which the loss function is minimized. Since the target column is continuous, the loss function will be:

$$L = \frac{1}{n}\sum_{i=0}^{n} (y_i, \gamma_i)^2 \quad (2)$$

Here yi is the observed value. γi is the predicted value. The goal is to find the minimum value of gamma that minimizes the loss function. As studied in 12th grade, to find minima and maxima, differentiate the loss function and set it equal to 0. This approach will be used.

$$\frac{dL}{d\gamma} = \frac{2}{2}\left(\sum_{i=0}^{n} (y_i - \gamma_j)\right) = -\sum_{i=0}^{n} (y_i - \gamma_j) \quad (3)$$

yi is the observed value and gamma i is the predicted value, by plugging the values in the Equation 3: Hence for gamma=14500, the loss function will be minimum so this value will become the prediction for the base model.

Step 2: The next step is to calculate the pseudo residuals which are (observed value – predicted value). This step can be written as:

$$\gamma_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \text{ for } i=1,\dots,n \quad (4)$$

Here F(xi) is the previous model and m is the number of DT made. Taking the derivative of the loss function with respect to the predicted value has already been done.

$$\frac{dL}{d\gamma} = -(y_i - \gamma_j) = -(Observed - Predicted) \quad (5)$$

If the formula of residuals shows that the derivative of the loss function is multiplied by a negative sign, then it becomes:

$$(Observed - Predicted) \quad (6)$$

The predicted value is the prediction made by the previous model. In the example the prediction made by the previous model (initial base model prediction) is 14500, to calculate the residuals the formula becomes:

$$(Observed - 14500) \quad (7)$$

Step 3: A model will be built on these pseudo residuals to make predictions. The goal is to minimize these residuals, which will eventually improve model accuracy and prediction power. Using the residuals as the target and the original features, cylinder number, cylinder height, and engine location, new predictions will be generated. The predictions will be the error values, not the predicted car price values, as the target column is an error.

Step 4: Consider hm(x) as the decision tree made on these residuals. In this step, the output values for each leaf of the decision tree need to be determined. It is possible for one leaf to contain more than one residual, so the final output for all the leaves must be found. To determine the output, the average of all the numbers in a leaf can be taken, regardless of whether there is only one number or multiple numbers. Mathematically this step can be represented as:

$$\gamma_m = \arg_\gamma \min \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma\, {}^h m(x_i)) \quad (8)$$

Here hm(xi) is the DT made on residuals and m is the number of DT. When m = 1, it is about the 1st DT, and when it is "M," it is about the last DT. The output value for the leaf is the value of gamma that minimizes the loss function. The left-hand side "Gamma" is the output value of a particular leaf. On the right-hand side [Fm-1(xi)+yhm(xi)] is similar to Step 1, but the difference is that previous predictions are taken, whereas earlier there was no previous prediction. Figure 3 shows the residuals regressor tree.

1st residual goes in R1,1 ,2nd and 3rd residuals go in R2,1 and 4th residual go in R3,1. The output for the first leave that is R1,1 is calculated.

$$\gamma_{1,1} = \arg\min \frac{1}{2}(12000 - (14500 + \gamma))^2$$

$$\gamma_{1,1} = \arg\min \tfrac{1}{2}(-2500 - \gamma)^2 \quad (9)$$

To find the value of gamma that minimizes the function, first find the derivative of the equation with respect to gamma and set it equal to 0.

$$\frac{d}{dr}\frac{1}{2}(-2500-\gamma)^2 = 0$$

$$(-2500-\gamma)^2 = 0$$

$$\gamma = -2500 \qquad (10)$$

Hence the leaf R1,1 has an output value of -2500. Let's solve for the R2,1 Let's take the derivative to get the minimum value of gamma for which this function. Figure 4 shows the Gamma value.

$$\frac{d}{dr}\frac{1}{2}\left[(2000-\gamma)^2 + \frac{1}{2}(1000-\gamma)^2\right] = 0$$

$$2000-\gamma + 1000-\gamma = 0$$

$$\gamma = 1500 \qquad (11)$$

The average of the residuals in the leaf R2,1 is determined. Thus, if a leaf contains more than one residual, the average of that leaf will be the final output. After calculating the output for all the leaves, the results are obtained:

Step-5: This is finally the last step where the predictions of the previous model need to be updated. It can be

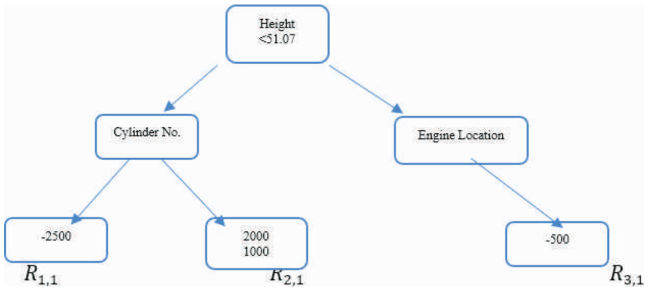

Figure 3. Residuals

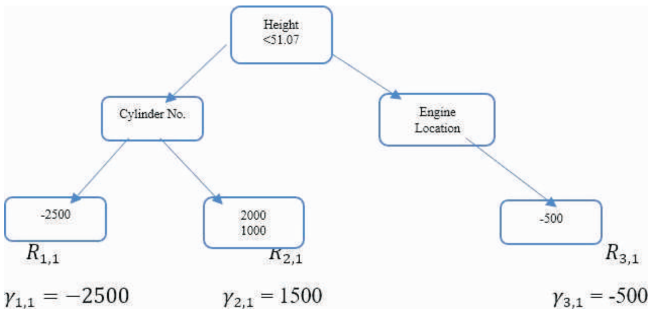

Figure 4. Gamma

updated as:

$$F_m(x) = F_{m-1}(x) + v_m h_m(x) \qquad (12)$$

Where m is the number of decision trees made.

Since the model has started being built, m = 1. To make a new decision tree (DT), the new predictions will be as follows:

Let Fm−1(x)F_{m-1}(x)Fm−1(x) represent the prediction of the base model (previous prediction). Since F1−1=0F_{1-1}=0F1−1=0 and F0F_0F0 is the base model, the previous prediction is 14,500. The learning rate, v\nuv, is typically chosen between 0 and 1. It reduces the effect each tree has on the final prediction, which improves accuracy over time. For this example, let's use v=0.1\nu = 0.1v=0.1.

Let Hm(x)H_m(x)Hm(x) be the DT created based on the residuals. To calculate the new prediction, the outcome is predicted for a data point with a car height of 48.8. This data point will pass through the decision tree, and the resulting output will be multiplied by the learning rate and then added to the previous prediction.

Assume m=2m = 2m=2, meaning two decision trees have been built. To get the new predictions, add the previous prediction, F1(x)F_1(x)F1(x), to the new DT created from the residuals. This process will be repeated until the loss is negligible. If a new data point with a height of 1.40 is introduced, it will go through all the trees, and the final prediction will be F2(x)F_2(x)F2(x) in this case, as there are only two trees.

A gradient boosting classifier is used when the target column is binary. All the steps explained for the gradient boosting regressor apply the only difference is the change in the loss function. Previously, Mean Squared Error was used when the target column was continuous, but this time, log-likelihood will be used as the loss function. To understand how this loss function works, and to read more about log-likelihood, it is recommended to review the detailed explanation provided in the accompanying material. The loss function for the classification problem is given below:

$$L = -\sum_{i=1}^{n} yi.\log(p) + (1-p)\log(1-p) \qquad (13)$$

The first step in the gradient boosting algorithm is to initialize the model with a constant value. The average of the target column is commonly used, but in this case, log(odds) will be used to determine that constant value. When differentiating this loss function, a function of log(odds) will result, and then a value of log(odds) that minimizes the loss function needs to be found. First, transform the loss function so that it becomes a function of log(odds).

$$L = -\sum_{i=1}^{n} yi.\log(p) + (1-p)\log(1-p)$$

$$L = -y * \log\left(\frac{p}{1-p}\right) - \log(1-p) \qquad (14)$$

$$Hence,\ L = -y * \log(odds) + \log(1 + e^{\log(odds)}) \qquad (15)$$

The loss function must be minimized. To achieve this, take the derivative with respect to the log(odds) and set it equal to 0. Here, y represents the observed values. The transformation of the loss function into a function of log(odds) is advantageous because it simplifies calculations. However, using the function of predicted probability p may also be convenient. Transforming the loss function is not mandatory but is done for easier calculations. The minimum value of this loss function will be the initial prediction (base model prediction).

In the Gradient Boosting Regressor, the next step involves calculating the pseudo-residuals by multiplying the derivative of the loss function by -1. With a different loss function, the probability of an outcome is considered. After finding the residuals, a decision tree is built with all independent variables and the target variables as "Residuals." For the first decision tree, the final output of the leaves needs to be determined since a leaf might contain more than one residual. A direct formula for calculating the output of a leaf will be provided. Finally, new predictions are obtained by adding the base model to the new tree created from the residuals.

5. Performance Analysis

In performance analysis, the values -1 and 1 generally represent the class labels or predictions made by a classification model. For instance, 1 represents the positive class (e.g., presence of a condition, occurrence of an event, or classification as phishing website). -1 represents the negative class (e.g., absence of a condition, non-occurrence of an event, or classification as a non- phishing website).

5.1 Confusion Matrix

The confusion matrix determines the performance of the classification models for a given set of test data.The true values for test data must be known to make this determination. It displays errors in model performance in matrix form, which is why it is also known as an error matrix. For classifiers with two prediction classes, the matrix is a 2x2 table; for three classes, it is a 3x3 table, and so on. The matrix is divided into two dimensions: predicted values and actual values, along with the total number of predictions. Predicted values are those generated by the model, while actual values are the true values for the given observations. Figure 5 shows the Confusion Matrix. The confusion matrix includes the following cases:

True Negative: The model predicted "No," and the actual value was also "No."

True Positive: The model predicted "Yes," and the actual value was also "Yes."

False Negative: The model predicted "No," but the actual value was "Yes" (also called a Type-II error).

False Positive: The model predicted "Yes," but the actual value was "No" (also called a Type-I error).

5.2 Classification Accuracy

It is one of the important parameters to determine the accuracy of the classification problems. It can be calculated as the ratio of the number of correct predictions made by the classifier to all number of

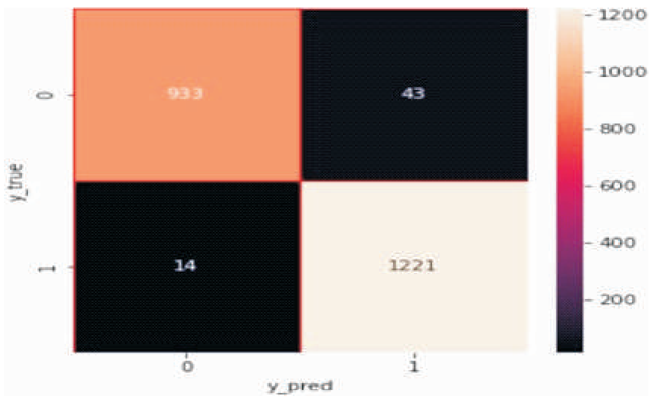

Figure 5. Confusion Matrix

predictions made by the classifiers. The formula is given as

$$\text{Accuracy} = \frac{\text{TruePositive} + \text{TrueNegative}}{\text{TruePositive} + \text{TrueNegative} + \text{FalsePositive} + \text{FalseNegative}}$$

The overall correctness of predictions for the proposed work.

5.3 Precision

It can be calculated as the number of correct outputs provided by the model or out of all positive classes that have predicted correctly by the model, how many of them were actually true. It can be calculated using the formula:

$$\text{Precision} = \frac{\text{TruePositive}}{\text{TruePositive} + \text{FalsePositive}}$$

Among the instances predicted as positive by the model, 97% were correctly identified as positive (true positives), while 3% were incorrectly classified as positive (false positives).

These metrics indicate that the model performs well in correctly identifying positive instances (phishing websites) while maintaining a good balance between precision (accuracy of positive predictions) and recall (ability to capture actual positives).

5.4 Recall

It is calculated as the out of total positive classes, how this model predicted correctly. The recall must be as high as possible.

$$\text{Recall} = \frac{\text{TruePositive}}{\text{TruePositive} + \text{FalseNegative}}$$

This implies that among the actual positive instances (label 1), the model correctly identified 99% of them as positive (true positives) and missed 1% (false negatives).

5.5 F-Measure

If two models have low precision and high recall or vice versa, it is difficult to compare these models. So, for this purpose, F-score is used. This score helps to evaluate the recall and precision at the same time. The F-score is maximum if the recall is equal to the precision. It can be calculated using the formula:

$$\text{F1 - score} = 2.\ \frac{\text{Precision.Recell}}{\text{Precision} + \text{Recel}}$$

The F1-score is the harmonic mean of precision and recall.

6. Results and Discussion

The graphs show the accuracy, F1_score, Recall and Precision values for Gradient Boosting Classifier, Random Forest, Support Vector machine, Decision trees separately. Table 1 shows the comparison of Performance Metrics. Figure 6 shows the Gradient Boosting Classifier. Figure 7 shows the Random Forest. Figure 8 shows the Support Vector Machine. Figure 9 shows the Decision Tree. Moreover, Figure 10 shows the comparison on all the machine learning models with the predicted results on Phishing Detection.

Machine Learning Model	Accuracy	Recall	Precision	F1_Score
Gradient Boosting Classifier	0.974	0.994	0.986	0.977
Random Forest	0.967	0.993	0.991	0.97
Support Vector Machine	0.964	0.98	0.965	0.968
Decision Tree	0.958	0.991	0.993	0.962

Table 1.Comparison of Performance Metrics

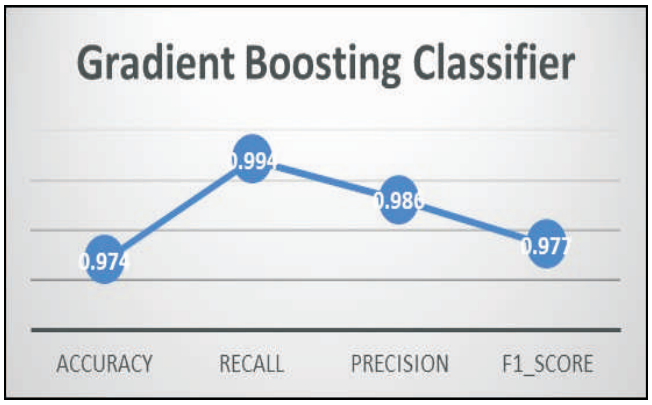

Figure 6. Gradient Boosting Classifier

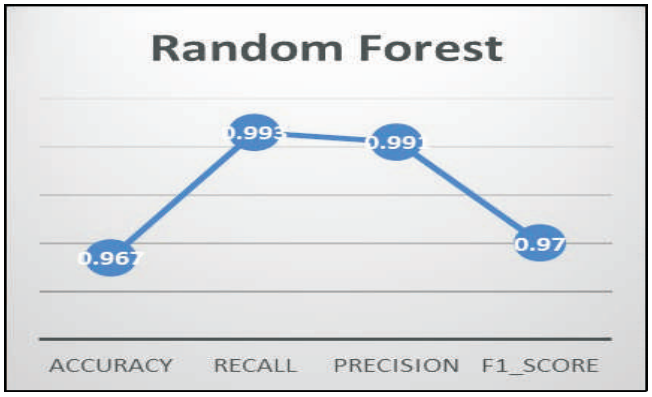

Figure 7. Random Forest

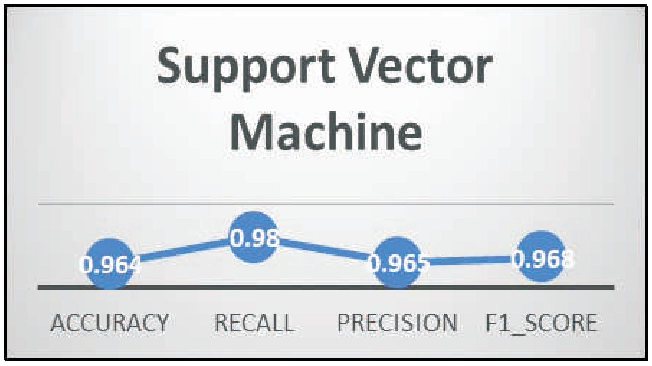

Figure 8. Support Vector Machine

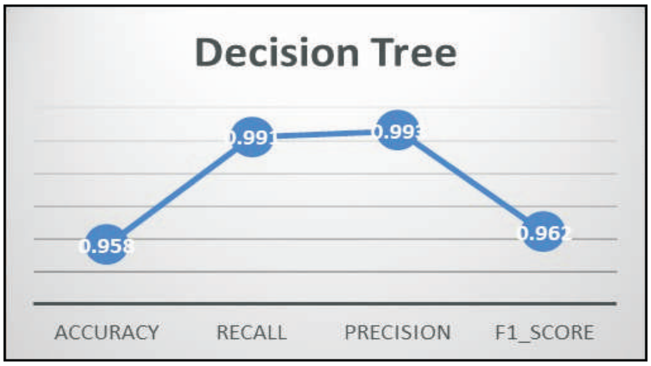

Figure 9. Decision Tree

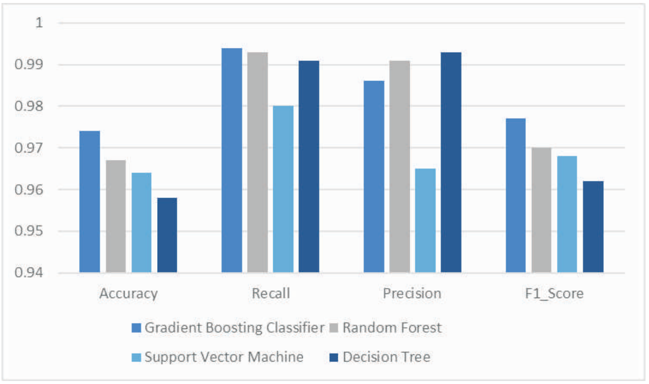

Figure 10. Performance metrics analysis on Phishing Detection

Conclusion

Many phishing detection models have been developed, but accuracy remains a major concern for investigators. Consequently, a new methodology with the highest accuracy is required for detecting phishing URLs. Machine learning, a rapidly evolving field, deals with datasets and the ways in which machines learn from experience. This paper presents a system designed to identify phishing URLs by combining the results of different machine learning techniques. The Gradient Boosting classifier is employed to detect phishing URLs and enhance performance in the proposed model. Accuracy varies across different models, and results indicate that the proposed model achieves improved accuracy compared to other machine learning methods.

Acknowledgment

I sincerely thank Dr. R. Mathusoothana S Kumar, M.E., Ph.D., and Head of the Department for his encouragement, valuable suggestions, and advice for the study. I am deeply indebted to my guide for the course and the study, Dr. V. Mary Amala Bai, M.E., Ph.D., for her valuable guidance and care shown in aiding me in completing the paper.

References

[1]. Capuano, N., Fenza, G., Loia, V., & Stanzione, C. (2022). Explainable artificial intelligence in cybersecurity: A survey. *IEEE Access*, 10, 93575-93600.

https://doi.org/10.1109/ACCESS.2022.3204171

[2]. Castaño, F., Fernañdez, E. F., Alaiz-Rodríguez, R., & Alegre, E. (2023). PhiKitA: Phishing kit attacks dataset for phishing websites identification. *IEEE Access*, 11, 40779-40789.

https://doi.org/10.1109/ACCESS.2023.3268027

[3]. Drew, J., & Moore, T. (2014, May). Automatic identification of replicated criminal websites using combined clustering. In *2014 IEEE Security and Privacy Workshops* (pp. 116-123). IEEE.

https://doi.org/10.1109/SPW.2014.26

[4]. El Aassal, A., Baki, S., Das, A., & Verma, R. M. (2020). An in-depth benchmarking and evaluation of phishing detection research for security needs. *IEEE Access*, 8, 22170-22192.

https://doi.org/10.1109/ACCESS.2020.2969780

[5]. Gualberto, E. S., De Sousa, R. T., Vieira, T. P. D. B., Da Costa, J. P. C. L., & Duque, C. G. (2020). The answer is in the text: Multi-stage methods for phishing detection based on feature engineering. *IEEE Access*, 8, 223529-223547.

https://doi.org/10.1109/ACCESS.2020.3043396

[6]. Gupta, M., Akiri, C., Aryal, K., Parker, E., & Praharaj, L. (2023). From chatgpt to threatgpt: Impact of generative AI in cybersecurity and privacy. *IEEE Access*, 11, 80218-80245.

https://doi.org/10.1109/ACCESS.2023.3300381

[7]. Kabla, A. H. H., Anbar, M., Manickam, S., & Karupayah, S. (2022). Eth-PSD: A machine learning-based phishing scam detection approach in ethereum. *IEEE Access*, 10, 118043-118057.

https://doi.org/10.1109/ACCESS.2022.3220780

[8]. Kara, I., Ok, M., & Ozaday, A. (2022). Characteristics of understanding urls and domain names features: The detection of phishing websites with machine learning methods. *IEEE Access*, 10, 124420-124428.

https://doi.org/10.1109/ACCESS.2022.3223111

[9]. Karim, A., Azam, S., Shanmugam, B., Kannoorpatti, K., & Alazab, M. (2019). A comprehensive survey for intelligent spam email detection. *IEEE Access, 7*, 168261-168295.

https://doi.org/10.1109/ACCESS.2019.2954791

[10]. Oest, A., Safaei, Y., Doupé, A., Ahn, G. J., Wardman, B., & Tyers, K. (2019, May). Phishfarm: A scalable framework for measuring the effectiveness of evasion techniques against browser phishing blacklists. In *2019 IEEE Symposium on Security and Privacy (SP)* (pp. 1344-1361). IEEE.

https://doi.org/10.1109/SP.2019.00049

[11]. Purwanto, R. W., Pal, A., Blair, A., & Jha, S. (2022). PhishSim: Aiding phishing website detection with a feature-free tool. *IEEE Transactions on Information Forensics and Security,* 17, 1497-1512.

https://doi.org/10.1109/TIFS.2022.3164212

[12]. Razaque, A., Alotaibi, B., Alotaibi, M., Amsaad, F., Manasov, A., Hariri, S., & Alotaibi, A. (2021). Blockchain-enabled deep recurrent neural network model for clickbait detection. *IEEE Access,* 10, 3144-3163.

https://doi.org/10.1109/ACCESS.2021.3137078

[13]. Salloum, S., Gaber, T., Vadera, S., & Shaalan, K. (2022). A systematic literature review on phishing email detection using natural language processing techniques. *IEEE Access,* 10, 65703-65727.

https://doi.org/10.1109/ACCESS.2022.3183083

[14]. Tsinganos, N., Mavridis, I., & Gritzalis, D. (2022). Utilizing convolutional neural networks and word embeddings for early-stage recognition of persuasion in chat-based social engineering attacks. *IEEE Access,* 10, 108517-108529.

https://doi.org/10.1109/ACCESS.2022.3213681

[15]. Wei, Y., & Sekiya, Y. (2022). Sufficiency of ensemble machine learning methods for phishing websites detection. *IEEE Access,* 10, 124103-124113.

https://doi.org/10.1109/ACCESS.2022.3224781

[16]. Zieni, R., Massari, L., & Calzarossa, M. C. (2023). Phishing or not phishing? A survey on the detection of phishing websites. *IEEE Access,* 11, 18499-18519.

https://doi.org/10.1109/ACCESS.2023.3247135

ABOUT THE AUTHOR

Aslin Sushmitha R. completed her B.Tech. in Information Technology from the University College of Engineering, Nagercoil, Tamil Nadu, India. She also completed her M.E. in Cybersecurity from Noorul Islam College of Engineering, Kumaracoil, Tamil Nadu, India.