International Journal of Humanities and Information Technology (IJHIT)
e-ISSN: 2456 –1142, Volume 6, Issue 4, (December 2024), www.ijhit.info

Efficient Phishing Detection System Using Boosting and Feature Engineering Techniques

Sivaraju Kuraku[1], Naveen Shrivastava[2], Mahesh Marri[3], Khaja Moiezuddin Mohammed[4], Rashmika Jalagam[5], Sridevi Allu[6]
[1]University of the Cumberlands
[2]T-Mobile
[3]Amazon Web Services, Senior Electrical Enginee
[4]IQVIA
[5]Matilda Cloud
[6]American Airlines

Abstract

Phishing schemes have been a constant challenge to internet security, which requires strong detection strategies. This study is the best phishing detection system that utilises the UCI Repository's Phishing Websites Data Set, which includes 11,055 occurrences and 30 characteristics of URLs, domains, and webpage content. The phishing and legitimate data, which went through an extensive preprocessing stage, including the elimination of noise, label encoding, and feature engineering to retrieve relevant URL-based, content-based, and third-party-derived features. The XGBoost model was trained to function as a classifier using gradient boosting and regularization methods. All of the following metrics ACC, PRE, REC, F1-score(F1), ROC, and SHAP show that the model performed exceptionally well: 99.8% accuracy, 99.9% precision, 99.5% recall, and 98.7% F1-score. The ROC curve and SHAP analysis illustrated that there were very strong discriminative features of the model, and those were DNS_Record, Mouse_Over, Web Forwards, Web Traffic, and URL Length. Comparative analysis with SVM and DNN models further confirmed XGBoost's superior performance in accurately and reliably detecting phishing websites, establishing it as a robust, scalable, and interpretable solution for cybersecurity applications

Keywords: Phishing Detection, XGBoost, Machine Learning, URL Features, Feature Engineering, Cybersecurity.

DOI: 10.21590/gdfg256r

I. INTRODUCTION

Social media, online banking, online shopping, and many other aspects of modern life have gone digital as a result of the exponential growth of worldwide networking and communication technology. Cyberattacks are made possible by the Internet's open, anonymous, and unmanaged architecture, which leaves networks and even experienced computer users vulnerable to significant security breaches [1]. It is

not feasible to entirely protect consumers from falling for the phishing scam, even though user awareness and experience are crucial [2]. The reason behind this is that phishers take into account the end user's personality traits, particularly when trying to trick more experienced users, in order to make their attacks more effective.

The aim of phishing emails is to trick recipients into divulging critical information by making them believe they are communicating with a legitimate business or organization [3]. In phishing attempts, the email appears to have originated from a reputable source. At least one person, often known as a "phisher" or "attacker," is responsible for planning the attack [4]. The individuals whose lives are messed up because of an attack are called victims or targets. One type of cybercrime is "phishing," which is getting malware onto computers or attempting to steal personal information like passwords or bank account data [5]. As expected, spotting the phishing attempt is no picnic. Even the most careful users could fall for this kind of attack, which involves changing a few characters in the URL to look like comparable Unicode characters [6]. On the other side, it might manifest in awkward ways, such as when an IP address masquerades as a domain name.

The process of feature engineering is essential in building high-performance classification models. It entails making, changing, and choosing informative features from the dataset in order to boost predicted accuracy, efficiency, and reduction of overfitting [7]. Equipped with necessary features, machine learning models might potentially successfully anticipate phishing situations. The fundamental challenge is determining which features are most important in order to optimize detection accuracy while keeping processing efficiency high [8].

The use of DL and ML to improve upon and overcome the shortcomings of traditional phishing detection methods has spurred a surge in research in this area in recent years [9]. The effectiveness of phishing attacks was enhanced with the development of deep learning algorithms that could detect trends and reveal data anomalies. Utilizing machine learning algorithms is a key component of modern methodologies used for cybercrime detection. Using machine learning techniques can help overcome the restrictions and constraints that classical detection methods confront [10][11]. Machine learning approaches are making their contribution both on the attacker side and the defender side. ML techniques are used on the attacker side in order to penetrate the defense wall [12]. On the defense side, on the other hand, ML techniques are utilized to develop timely and sound defense mechanisms.

A. Motivation and Contribution

Phishing attacks have also developed a high level of sophistication and, therefore, the detection of an attack has become a difficult and vital issue in cybersecurity. Conventional methods of detection fail to reveal the intelligent phishing schemes, like the URLs that have active character replacements or an IP address that has been used in place of the domain name, putting users and organizations in serious danger. This study motivated by the use of Boosting techniques that have the ability of combining a series of weak learners to form a robust predictive model to improve on phishing detection. Model performance, detection accuracy, false positive rate, and computational economy can all be enhanced by combining Boosting with good feature engineering. In order to fortify cybersecurity defences and deal with the ever-changing nature of phishing threats, this study seeks to establish a strong and dependable detection method. Listed below are a number of important contributions made by this study:

- The research made use of the well-rounded and feature-rich UCI Phishing Websites dataset, which includes 11,055 occurrences with 30 URL and website characteristics.
- Implemented robust pre-processing methods, including noise removal, label encoding, and systematic feature engineering, extracting URL-based, content-based, and third-party features to enhance model accuracy.
- Proposed and implemented an XGBoost-based phishing detection system, leveraging gradient boosting and regularization techniques for robust, scalable, and high-performance classification.
- Carried out comprehensive performance analysis using multiple assessment metrics to guarantee fair model comparison. Measurements such as ACC, PRE, REC, F1, ROC, and SHAP analysis were included in this set.

B. Novelty and Justification of the Study

This study's original contribution is that it improves phishing website detection by utilizing the robust XGBoost classifier in conjunction with extensive pre-processing and interpretability procedures. This method goes beyond traditional models' accuracy-centric focus to highlight robust classification performance and model transparency via SHAP analysis, which helps to understand the role of different features in predictions more thoroughly. The study improves the model's generalizability by training it on cleaner and more representative data through the integration of noise removal, feature engineering, and balanced data splitting. This provides strong justification for the research, as phishing websites are continuously evolving, and the need for reliable, interpretable, and scalable solutions is critical for strengthening cybersecurity defences against such threats.

C. Structure of the Paper

The following is the outline of the paper. This section II focuses on a literature study of relevant works on phishing detection systems. Section III explains how to implement, prepare, and use this model. Presented in Section IV are the experimental data together with a comparison to previous methods. In Section V, conclude with a brief summary of the work and discuss some avenues for further study.

II. LITERATURE REVIEW

A comprehensive literature survey and critical analysis of previous work on phishing detection systems provided the basis of this study, defining its scope and directing its general trajectory.

Sameen, Han and Hwang (2020) implemented URL HTML Encoding to do real-time URL classification and proactive technique comparison. Using a benchmark dataset that included 100,000 both genuine and malicious URLs, they assessed their method. With a 98.00% success rate, the results show that Phish Haven is the best algorithm available for detecting lexical-based, human-crafted phishing URLs [13].

Abedin et al. (2020) use three ML algorithms to identify phishing websites. This program was trained using features that are based on URLs, and its goal is to detect phishing websites and legitimate ones at the same time. Prevention of Zero-Day attacks was the objective of the exercise. A 97/99 recall/97 F1 Score was determined for the random forest classifier based on observations. The proposed model was efficient and fast because it simply operates using a URL and does not require any other resources to be analyzed, as was the case with previous studies [14].

Alam et al. (2020) developed an algorithm to detect phishing attempts utilizing ML techniques like DT and RF. A real-world dataset of phishing attempts was successfully processed by ML using Kaggle. In order to analyse the dataset's characteristics, the proposed model uses feature selection processes including PCA. The most accurate algorithm was the random forest one, with a 97% success rate [15].

Yazhmozhi and Janet (2019) One of the claims made in this paper is that phishing websites may be identified in real-time. It examines the efficacy of five separate classification algorithms fed word vectors and NLP together with two separate sets of features. The RF method achieved a 97.99% accuracy rate by utilizing characteristics generated from natural language processing, surpassing other ML classification algorithms like NB, LR, SVM, DT, and RF [16].

Nugraha and Rahman (2019) Many online phishing detection systems have looked to a meta-algorithm to boost their classification effectiveness. The UCI ML Repository's Web Phishing dataset was consulted throughout this testing phase. Addition of the bagging process resulted in a 97.1% increase in accuracy, addition of the boosting process 97.3%, and addition of the stacking procedure 97.5%. There is hope that the model can be utilised to refine the creation of other phishing online detection systems, thanks to the enhanced performance it achieves [17].

Shyni, Sundar and Ebby (2018). One such approach to detecting malicious websites is parse tree validation. A novel approach of detecting phishing websites was devised, which comprised constructing a parse tree from all the hyperlinks on a certain page by means of the Google API. After testing with 1,000 legitimate pages and 1,000 phishing pages, the approach had a FPR of 5.2% and a FNR of 7.3%, respectively [18].

Phishing website identification has been greatly enhanced with ML and URL-based analysis; yet, there are still certain gaps in the research. Most existing approaches heavily rely on handcrafted lexical or URL-based features, which may limit their ability to generalize to evolving phishing techniques, especially AI-generated or obfuscated URLs. While ensemble and multi-threading methods have improved real-time detection, scalability and efficiency under high-traffic conditions remain a challenge. Additionally, many models focus primarily on URL analysis without incorporating other contextual or semantic information from website content, which could enhance detection accuracy. In addition, there has been little investigation into adaptive models that can discover zero-day phishing assaults proactively, and existing methods frequently fail with small or compressed URLs. One more point: phishing detection systems can be made better and more resilient if they utilise a different combination of feature selection and dimensionality reduction approaches each time they're investigated.

Table I provides a comprehensive summary of recent studies on phishing detection systems, highlighting the models employed, datasets used, major findings, identified challenges, and possible directions for future research.

TABLE I. RECENT STUDIES ON PHISHING DETECTION SYSTEMS

Author & Year	**Proposed Work**	**Dataset**	**Key Findings**	**Challenges & Future Work**
Sameen,	PhishHaven: Random	A dataset	Reached a 98% success	Scaling to larger

Han & Hwang (2020)	voting, multi-threaded ensemble ML classification, HTML encoding of URLs, URL Hit for short URLs	that compares 100,000 safe and dangerous URLs	rate; detects upcoming AI-generated phishing URLs with a 100% success rate; employs several threads for real-time detection	datasets; adaptive detection for evolving phishing techniques
Abedin et al. (2020)	URL-based phishing detection using three ML algorithms; Random Forest used to prevent Zero-Day attacks	URL-based features dataset	Random Forest achieved 97% precision, 99% recall, 97% F1-score; efficient as it relies only on URL analysis	Could extend to multi-feature analysis for robustness against advanced phishing attacks
Alam et al. (2020)	Phishing detection utilizing ML with RF and DT, with features selected using principal component analysis	Kaggle phishing dataset	RF achieved 97% maximum accuracy; PCA enhanced feature analysis	Explore ensemble models and real-time deployment for dynamic detection
Yazhmozhi & Janet (2019)	Phishing detection in real-time using five ML techniques that utilise word vector-based characteristics and natural language processing	Phishing Websites Data Set	RF using NLP characteristics outperformed the other classifiers with a 97.99% success rate.	Integration of hybrid feature sets; optimization for faster real-time performance
Nugraha & Rahman (2019)	Bagging, boosting, and stacking are meta-algorithms that can improve phishing detection.	Database on Online Phishing Attacks (UCI ML Repository)	Accuracy improved: 97.1% (bagging), 97.3% (boosting), 97.5% (stacking)	Combine meta-algorithms with deep learning for higher accuracy
Shyni, Sundar & Ebby (2018)	Parse tree validation: intercept hyperlinks via Google API to detect phishing	1000 phishing + 1000 legitimate pages	Detection using novel hyperlinks; 7.3% false negative, 5.2% false positive	Scale to larger datasets; integrate ML techniques for improved detection

III. RESEARCH METHODOLOGY

Data gathering is the first step in the suggested method for phishing website detection; to do this, and must access the UCI ML Registry and retrieve the phishing website dataset. Feature engineering, label encoding, and noise removal are some of the pre-processing procedures used to ensure data quality and consistency after acquisition. An 80:20 train-test split is used to produce training and testing subsets from the pre-processed dataset. It is then trained to create an XGBoost classification model which is suited because it is a strong and efficient model in recognizing the intense, high-dimensional data. REC, ACC, PRE, F1, ROC curve, and SHAP values are some of the interpretable metrics used to evaluate the model's performance after classification. Lastly, the findings are analyzed in a systematic manner to come up with valuable information and prove the ability of the model in identifying phishing websites. The general process of the suggested methodology is presented in Figure 1.

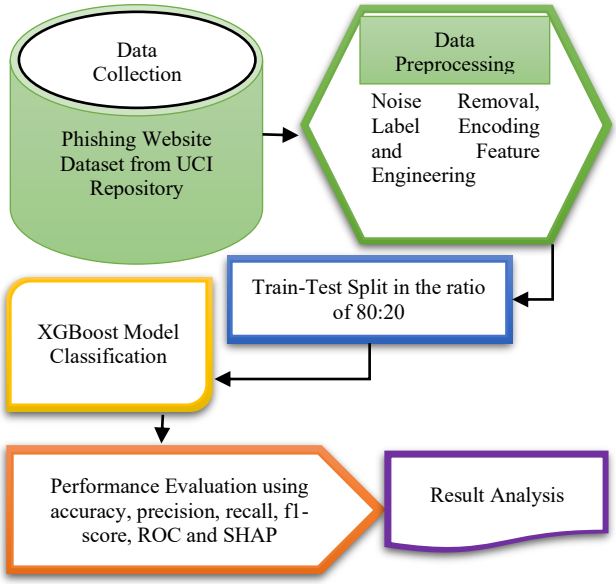


Fig. 1. Proposed Flowchart for Phishing Detection System

This section lays out the procedures that the proposed Phishing Detection System outlines.

A. Data Collection and Analysis

The study's data set for phishing websites is located in the UCI Repository and has 11,055 cases total, including 6,157 genuine and 4,898 phishing instances. Each instance is represented by 30 attributes of the URL, domain information, and webpage content. It is a balanced supervised classification dataset called phishing (-1) or legitimate (1) that is extensively used in the development and estimation of phishing detectors. The Heatmap of this data is shown below:

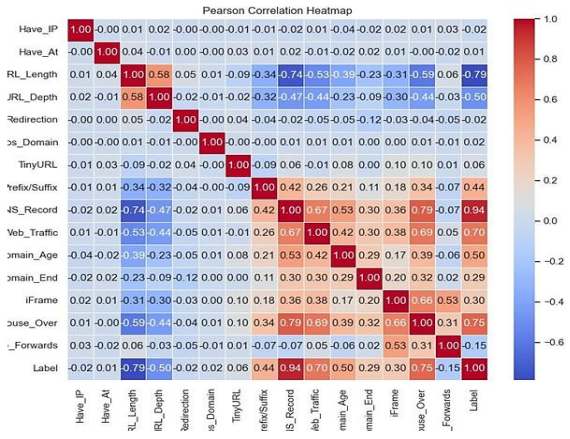


Fig. 2. Pearson Correlation Heatmap

The Pearson correlation heatmap in Figure 2 shows the relationship between the URL and website attributes. On a scale from -1 to 1, very strong positive correlation is represented by red, very strong negative correlation by blue, and very weak or non-existent correlation by light ones. The dark red cells (value 1.0) diagonal indicates the self-correlation of every feature. Such characteristics as Haveri, URL Length, Web Traffic, and Label are emphasized and assist in determining which features are most significantly correlated with phishing or legitimate classification.

B. Data Pre-processing

The accuracy of malicious URL detection systems can important to improved with the use of effective pre-processing methods. The main pre-processing techniques are the noise cancelling, label encoding and feature engineering described as follows in the present section:

- **Noise Removal:** This section aims at refining data through the identification and removal of noise. Noise constitutes irrelevant or inaccurate data points and this can distort the entire study.
- **Label Encoding:** Label Encoding is a ML technique that converts text or labels, which contain categorical data, into a numerical format [19]. The process is important in assuring that the data well-processed by the model's enabling analysis and grouping of data particularly as applied in the case of phishing detection.
- **Feature Engineering:** The feature engineering is given to the URL dataset comprising seventeen (17) lexical features. The features are purely based on the URL structure [20]. These characteristics range from the very basic, like the domain name and URL length, to the very advanced, such the IP address, special characters, redirection, domain age, web traffic, and mouse-over effects. A very successful phishing detection system is made possible through the methodical engineering of characteristics that are based on URLs, content, and third parties. This improves the model's accuracy, resilience, and interpretability.

C. Data Splitting

Two groups of data are created: training and testing. This study used an 80:20 split to construct two sets from the dataset: one for testing and one for training.

D. XGBoost Classification

The XGboost model is introduced and employed in a detection system in this section.

One end-to-end tree boosting technique that is quite scalable is XGboost. Numerous beneficial designs are incorporated. Economical proposal computation makes use of the justified weighted quantile diagram. This sparsity-aware algorithm was created for the purpose of learning trees in parallel. The implementation of out-of-core tree learning makes use of a cache-aware block structure. For these reasons, XGboost is the fastest and most accurate ML algorithm currently available [21]. Academics may also quickly implement it on a decentralized system and use GPUs to speed it up. In order to simplify the model, the XGBoost technique incorporates a penalty parameter, θ, into the incremental training of an Ensemble (Group) Tree Model.

The objective function is therefore written in Equation (1), with $\hat{y}^{(t)}$Denoting the prediction of the k-th case at the t-th increment [22]:

$$L^{(t)} = \sum_{k=1}^{n} l\left(y_k, \hat{y}_k^{(k-1)} + f_t(X_i)\right) + \theta(f_t) \quad (1)$$

where the loss function is denoted by L. In order to minimize the objective function that was previously described, the XGBoost technique makes use of the reduced second-order formula provided in Equation (2):

$$\tilde{L}^{(t)} = \sum_{k=1}^{n} \left[g_k f_t(X_k) + \frac{1}{2} h_k f_t^2(X_k)\right] + \theta(f_t) \quad (2)$$

where $g_k = \partial_{\hat{y}^{(t-1)}} l(y_k, \hat{y}^{(t-1)})$ and $h_k = \partial^2_{\hat{y}^{(t-1)}} l(y_k, \hat{y}^{(t-1)}) \cdot g_k$ $h_k$stands for the first and second order forms of the gradients with respect to the loss function.

E.Evaluation Metrics

A number of main evaluation measures are used to figure out how accurate an ML model is at making predictions. The model's overall success is measured by its ACC, PRE, REC F1, ROC, and SHAP, which count the number of correct and incorrect predictions.

- **True Positives (TP):** Situations where phishing websites have been positively identified as such.
- **True Negatives (TN):** Situations in which trustworthy websites are accurately regarded as trustworthy
- **False Positives (FP):** The occurrence of false alarms caused by reputable websites being mistakenly identified as phishing.
- **False Negatives (FN):** Problems caused by falsely categorizing malicious websites as safe (missed detection).

Accuracy: Accuracy evaluates the model's overall proportion of correct predictions across both classes, offering a broad overview of performance. It is shown in the Equation. (3):

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \quad (3)$$

Precision: The actual number of positive samples divided by the predicted number of positive samples is known as ACC. To illustrate, take Equation (4) into consideration.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad \square\square\square$$

Recall: The sum of all predicted TP samples is called recall. The suggested model appears to work effectively when the R-value is close to one. A description of this can be found in Equation (5).

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad \square\square\square$$

F1-Score: Right now, REC and PRE are harmonized to form the F1. Equation (6) demonstrates.

$$\text{F1} - \text{Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision+Recall}} \quad \square\square\square$$

ROC: A classification model's presentation over all classification criteria is graphically represented by the ROC.

SHAP: ML model predictions can be better understood using SHAP, a game-theoretic technique. Each attribute is "contributed" to a specific forecast using the Shapley values, an idea from cooperative game theory. For a certain feature i, the mathematical expression for the Shapley value, $\emptyset_i$, Is given by Equation (7):

$$\emptyset_i(f) = \sum_{S \subseteq N\{i\}} \frac{|S|!(|N|-|S|-1)!}{|N|!} \big(f(S \cup \{i\}) - f(S)\big) \quad \square\square\square$$

The prediction in $f(S \cup \{i\})$ is based on both features in S and feature i, whereas in $f(S)$The forecast is based only on features in S. $N$is the complete set of features, and S is the subset of features that excludes feature iIn this case?

IV. Results And Discussion

The study uses a Core (TM) i7-1065G7 CPU @ 1.30GHz 1.50 GHz processor, and the environment is downloaded ahead of time. Python 3.7.1 is also utilised because of the abundance of classification models and packages it provides. The suggested phishing detection system's performance results utilizing the XGBoost model are presented in Table II. With a success rate of 99.8 percent, the model was clearly very effective in its classification of almost all situations. The model's high accuracy rate of 99.9 percent in detecting phishing attempts with few false positives is shown. Its strong performance in identifying most phishing samples and reducing the number of false negatives is indicated by a recall score of 99.5%. The F1-score of 98.7, which measures the model's accuracy and consistency in detecting phishing attempts, is also indicative of its overall reliability and consistency.

TABLE II. Performance Results of the Proposed Model for Phishing Detection System

Performance matrix	**XGBoost**
Accuracy	99.8%
Precision	99.9%
Recall	99.5%
F1-Score	98.7%

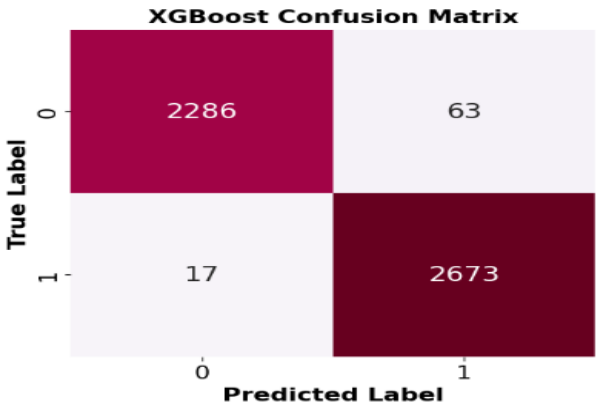


Fig. 3. Confusion Matrix of the XGBoost

Figure 3 displays the confusion matrix, which represents the XGboost model's real performance in classification. To be more precise, out of 26,73 phishing sites, the model successfully recognised 22,286 as authentic. These high percentages of right classifications show how well the program can differentiate between real and fake websites.

Figure 4 illustrates an ROC curve of the XGBoost model of phishing detection. The curve shows that there is a great separation of the two classes with the True Positive Rating being very near to 1.0 and the False Positive Rating being very near to 0.0. The model's ability to separate between phishing and legal websites is demonstrated by its AUC of 1.00, which indicates nearly perfect classification skill. The suggested phishing detection method is powerful and dependable, as its exceptional performance attests.

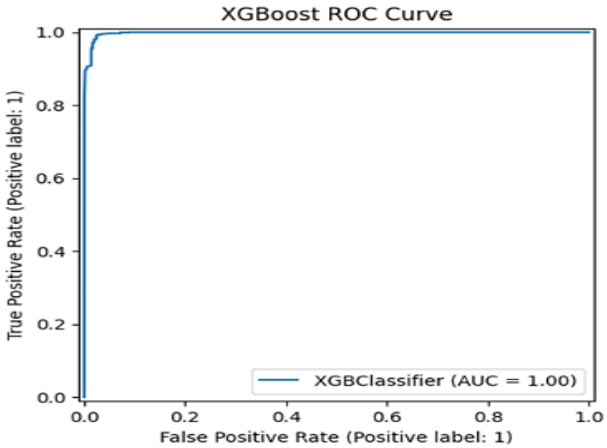


Fig. 4. ROC Analysis of the XGBoost

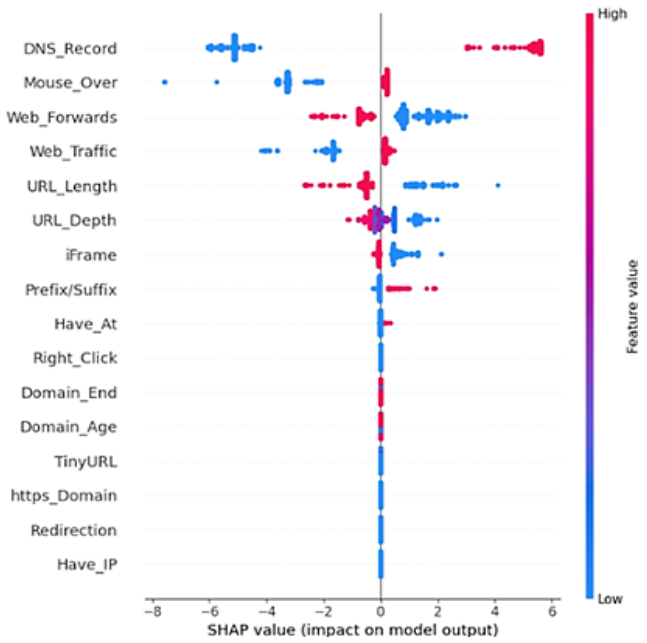


Fig. 5. SHAP Summary of XGBoost

The impact of features on the model's predictions is seen in Figure 5, a summary graphic from SHAP (Shapley Additive exPlanations). The features are displayed on the y-axis, while the SHAP values, which indicate the impact of each feature, are shown on the x-axis. Point colors reflect feature values (red = high, blue = low). For instance, high values of "DNS_Record" strongly increase predictions, whereas low values of "Mouse_Over" reduce them. The horizontal spread illustrates how the impact of each feature varies over the dataset.

A. Comparative Analysis

Table III compares the effectiveness of the phishing detection system's different models. SVM with a sigmoid kernel achieved middling results with an F1 of 84.5%, ACC of 82.7%, PRE of 84.4%, and REC of 84.6%. After comparing the models, the DNN model had the best results with an F1 of 87.3%, REC of 88.0%, PRE of 86.7%, and ACC of 90.1%. While both were considerably outperformed by the suggested XGBoost model, it came at the cost of a 98.7% F1, 99.5% REC, 99.9% PRE, and 99.8% ACC. These findings indicate that XGBoost is better than traditional and deep learning methods at accurately and reliably detecting phishing sites.

TABLE III. PERFORMANCE COMPARISON OF DIFFERENT MODELS FOR PHISHING DETECTION SYSTEM

Models	**Accuracy**	**Precision**	**Recall**	**F1-Score**
SVM_sigmoid [23]	82.7	84.4	84.6	84.5
DNN [24]	90.1	86.7	88.0	87.3
XGBoost	99.8	99.9	99.5	98.7

The model phishing detection model is XGBoost which is extremely effective among the traditional and deep learning models. Its gradient boosting model allows it to generate powerful classifiers, by continuously correcting the errors of past iterations, and therefore makes powerful decisions. XGBoost also works well with large and complicated datasets and it avoids overfitting by regularizing the data. Moreover, it allows a wide range of flexibility in using alternative methods of feature selection and can be used with interpretability tools, such as SHAP, which can be used to determine which features have the most important effect on classification. These are the strengths that qualify XGBoost to be a trustworthy and scalable option in the accurate differentiation of the phishing and genuine sites.

V. Conclusion and Future Study

Phishing detection systems are now essential in cybersecurity to protect individuals, organizations, and critical infrastructure from fraudulent activity involving deceptive websites and URLs used to steal sensitive information. The XGBoost-based phishing detection system showcased here outperformed both more conventional methods, such as SVMs with sigmoid kernels, and more sophisticated models, such as DNNs, which both registered lower accuracy rates. XGBoost is highly effective as it identifies hidden and subtle trends in URL, domain, and webpage characteristics while efficiently handling large-scale data. Its combined regularization methods enhance model strength and reduce overfitting, a common challenge in phishing detection tasks. Another key advantage is the inclusion of SHAP analysis, which significantly improves interpretability and transparency by revealing the most influential features in classification decisions. Critical attributes such as DNS records, mouse-over effects, web traffic levels, and URL length were identified as major factors in unique phishing sites from legitimate ones. Researchers and security professionals can use this to validate the model's efficiency and gain useful insights.

Future enhancements include real-time deployment, improved adversarial robustness, hybrid approaches, and scalable cloud-based solutions integrating deep learning with boosting for adaptive phishing detection.

References

1. O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, “Machine learning based phishing detection from URLs,” *Expert Syst. Appl.*, 2019, doi: 10.1016/j.eswa.2018.09.029.
2. H. P. Kapadia, “Cross-Platform UI/UX Adaptions Engine for Hybrid Mobile Apps,” *Int. J. Nov. Res. Dev.*, vol. 5, no. 9, pp. 30–37, 2020.
3. V. Bhavsar, A. Kadlak, and S. Sharma, “Study on Phishing Attacks,” *Int. J. Comput. Appl.*, vol. 182, pp. 27–29, 2018, doi: 10.5120/ijca2018918286.
4. A. Shankar, R. Shetty, and B. N. K, “A Review on Phishing Attacks,” *Int. J. Appl. Eng.*, vol. 14, no. 9, pp. 2171–2175, 2019.
5. S. S. S. Neeli, “Serverless Databases: A Cost-Effective and Scalable Solution,” *Int. J. Innov. Res. Eng. Multidiscip. Phys. Sci.*, vol. 7, no. 6, p. 7, 2019.
6. M. Zouina and B. Outtaj, “A novel lightweight URL phishing detection system using SVM and similarity index,” *Human-centric Comput. Inf. Sci.*, 2017, doi: 10.1186/s13673-017-0098-1.
7. D. D. Rao, “Multimedia-Based Intelligent Content Networking for Future Internet,” in *2009*

Third UKSim European Symposium on Computer Modeling and Simulation, 2009, pp. 55–59. doi: 10.1109/EMS.2009.108.

8. A. Kushwaha, P. Pathak, and S. Gupta, "Review of optimize load balancing algorithms in cloud," *Int. J. Distrib. Cloud Comput.*, vol. 4, no. 2, pp. 1–9, 2016.
9. A. Thapliyal, P. S. Bhagavathi, T. Arunan, and D. D. Rao, "Realizing Zones Using UPnP," in *2009 6th IEEE Consumer Communications and Networking Conference*, 2009, pp. 1–5. doi: 10.1109/CCNC.2009.4784867.
10. I. Firdausi, C. Lim, A. Erwin, and A. S. Nugroho, "Analysis of Machine learning Techniques Used in Behavior-Based Malware Detection," in *2010 Second International Conference on Advances in Computing, Control, and Telecommunication Technologies*, IEEE, Dec. 2010, pp. 201–203. doi: 10.1109/ACT.2010.33.
11. S. S. S. Neeli, "Decentralized Databases Leveraging Blockchain Technology," vol. 8, no. 1, pp. 1–8, 2020.
12. K. Shaukat *et al.*, "Performance Comparison and Current Challenges of Using Machine Learning Techniques in Cybersecurity," *Energies*, vol. 13, no. 10, May 2020, doi: 10.3390/en13102509.
13. M. Sameen, K. Han, and S. O. Hwang, "PhishHaven—An Efficient Real-Time AI Phishing URLs Detection System," *IEEE Access*, vol. 8, pp. 83425–83443, 2020, doi: 10.1109/ACCESS.2020.2991403.
14. N. F. Abedin, R. Bawm, T. Sarwar, M. Saifuddin, M. A. Rahman, and S. Hossain, "Phishing attack detection using machine learning classification techniques," in *Proceedings of the 3rd International Conference on Intelligent Sustainable Systems, ICISS 2020*, 2020. doi: 10.1109/ICISS49785.2020.9315895.
15. Bandaru, S. R., Shyam, D., Manoharan, P., Syed, M. H., Ragireddy, U. K., & Addepalli, P. V. (2023). Data-Driven Approaches for Forecasting Cost Overruns in Infrastructure Projects. *European Journal of Technology*, *7*(5), 23-34.
16. Dhuli, S., Prabu, M., Muzaffer, H. S., Uday Kumar, R., & Prasanth Varma, A. (2023). A Survey on Critical Success Factors Influencing Project Success Across Multiple Industries. *J Contemp Edu Theo Artific Intel: JCETAI-106*.
17. Ragireddy, U. K., Addepalli, P. V., Bandaru, S. R., Shyam, D., Manoharan, P., & Syed, M. H. (2023). An Overview of Different Automation Strategies Used in Cloud-Based CI/CD Pipelines for Software Deployment. *International Journal of Technology, Management and Humanities*, *9*(01), 116-127.
18. Addepalli, P. V., Bandaru, S. R., Shyam, D., Manoharan, P., Syed, M. H., & Ragireddy, U. K. (2023). Performance Analysis of Machine Learning Models for Cybersecurity Risk Classification in IT Projects. *Universal Library of Engineering Technology*, (Issue).
19. Mylavarapu, K. B., Pilli, J., Jannu, P. K., Mohammad, J. A., & Panchali, S. H. (2023). Artificial Intelligence for Continuous Quality Assurance in Cloud-Native Applications: A Review. *European Journal of Technology*, *7*(5), 35-49.
20. Jannu, P. K., Mohammad, J. A., Panchali, S. H., Mylavarapu, K. B., & Pilli, J. (2023). A Survey on Energy-Efficient Microarchitecture Design Strategies for Mobile CPUs and GPUs: Techniques and Future Directions. *International Journal of Technology, Management and Humanities*, *9*(04), 137-151.

21. Taluri, R. (2021). Cloud-Native Architectures for Enterprise Financial Data Management, Analytics, and Regulatory Reporting Compliance. International Journal of Emerging Trends in Computer Science and Information Technology, 2(2), 101-111. https://doi.org/10.63282/3050-9246.IJETCSIT-V2I2P112
22. Taluri, R. (2022). Cloud Data Engineering Strategies for Large-Scale Financial Data Integration and Intelligent Corporate Performance Reporting. International Journal of Emerging Research in Engineering and Technology, 3(4), 176-188. https://doi.org/10.63282/3050-922X.IJERET-V3I4P119
23. Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., Pathani, A. K., Dasari, A., & Chilakapati, V. K. (2023). An AI-Based Framework for Detecting IoT Botnets Through Network Traffic Analysis and Modeling. *Universal Library of Engineering Technology*, (Issue).
24. Kavuluri, H. V. R., Pathani, A. K., Dasari, A., Chilakapati, V. K., Keshireddy, S. R., & Nagumotu, V. T. (2023). Cloud Data-Center Network Verification: Approaches, Algorithms and Toolchains. *American Journal of Computing and Engineering*, *6*(2), 18-32.
25. Chilakapati, V. K., Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., Pathani, A. K., & Dasari, A. (2023). A Survey on 5G/6G Radio Access Networks: Technologies, Protocols, and Performance Metrics. *International Journal of Technology, Management and Humanities*, *9*(02), 79-93.
26. Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., Pathani, A. K., Dasari, A., & Chilakapati, V. K. (2023). An AI-Based Framework for Detecting IoT Botnets Through Network Traffic Analysis and Modeling. *Universal Library of Engineering Technology*, (Issue).
27. M. N. Alam, D. Sarma, F. F. Lima, I. Saha, R.-E.- Ulfath, and S. Hossain, "Phishing Attacks Detection using Machine Learning Approach," in *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2020, pp. 1173–1179. doi: 10.1109/ICSSIT48917.2020.9214225.
28. V. M. Yazhmozhi and B. Janet, "Natural language processing and Machine learning based phishing website detection system," in *Proceedings of the 3rd International Conference on I-SMAC IoT in Social, Mobile, Analytics and Cloud, I-SMAC 2019*, 2019. doi: 10.1109/I-SMAC47947.2019.9032492.
29. A. F. Nugraha and L. Rahman, "Meta-algorithms for improving classification performance in the web-phishing detection process," in *2019 4th International Conference on Information Technology, Information Systems and Electrical Engineering, ICITISEE 2019*, 2019. doi: 10.1109/ICITISEE48480.2019.9003952.
30. C. E. Shyni, A. D. Sundar, and G. S. E. Ebby, "Phishing Detection in Websites using Parse Tree Validation," in *2018 Recent Advances on Engineering, Technology and Computational Sciences (RAETCS)*, 2018, pp. 1–4. doi: 10.1109/RAETCS.2018.8443961.
31. A. K. Jain and B. B. Gupta, "A machine learning based approach for phishing detection using hyperlink information," *J. Ambient Intell. Humaniz. Comput.*, vol. 10, no. 5, pp. 2015–2028, May 2019, doi: 10.1007/s12652-018-0798-z.
32. R. Mahajan and I. Siddavatam, "Phishing Website Detection using Machine Learning Algorithms," *Int. J. Comput. Appl.*, vol. 181, no. 23, pp. 45–47, 2018, doi: 10.5120/ijca2018918026.
33. S. Lei, K. Xu, Y. Huang, and X. Sha, "An XGBoost-based system for financial fraud detection,"

E3S Web Conf., vol. 214, pp. 1–4, 2020, doi: 10.1051/e3sconf/202021402042.

34. S. M. Kasongo and Y. Sun, "Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset," *J. Big Data*, 2020, doi: 10.1186/s40537-020-00379-6.
35. V. Shahrivari, M. M. Darabi, and M. Izadi, "Phishing Detection Using Machine Learning Techniques," 2020.
36. M. Chatterjee and A.-S. Namin, "Detecting Phishing Websites through Deep Reinforcement Learning," in *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, IEEE, Jul. 2019, pp. 227–232. doi: 10.1109/COMPSAC.2019.10211.