Exploring Machine Learning Techniques for Vulnerability Scanner that automates the detection of common security flaws: A Comprehensive Study

Prem
Department of Computer Science and Engineering
Lovely Professional University
Jalandhar, India
nameexample@gmail.com

Tushar
Department of Computer Science and Engineering
Lovely Professional University
Jalandhar, India
nameexample@gmail.com

Pawn
Department of Computer Science and Engineering
Lovely Professional University
Jalandhar, India
nameexample@gmail.com

Prem
Department of Computer Science and Engineering
Lovely Professional University
Jalandhar, India
nameexample@gmail.com

Tushar
Department of Computer Science and Engineering
Lovely Professional University
Jalandhar, India
nameexample@gmail.com

Pawn
Department of Computer Science and Engineering
Lovely Professional University
Jalandhar, India
nameexample@gmail.com

Gaurav
Department of Computer Science and Engineering
Lovely Professional University
Jalandhar, India
nameexample@gmail.com

***Abstract*— As cyber threats grow increasingly sophisticated, the task of detecting security vulnerabilities has become a major challenge for modern cybersecurity. This study investigates how machine learning can be used to automate the detection of common security flaws in websites, with a focus on phishing attacks. The study examined a dataset containing more than 11,000 websites, with 32 distinct features, including URL structure, domain information, and traffic patterns. Various machine learning models, such as Logistic Regression, K-Nearest Neighbors, Support Vector Machines, Decision Trees, and Random Forests, were tested during the analysis. Each model was assessed using important performance metrics, including accuracy, precision, recall, and F1 score. Of all the models tested, Random Forests and Support Vector Machines performed the best, achieving the highest accuracy and F1 scores. This paper highlights how machine learning can enhance vulnerability detection, making the process faster and more accurate. The findings suggest that machine learning tools have the potential to transform cybersecurity by speeding up threat detection and providing more insightful, context-aware risk assessments. These tools offer way to safeguard web applications.**

Keywords—Vulnerability Detection, Cybersecurity, Random Forest, Support Vector Machine, Logistic Regression, K-Nearest Neighbors, Decision Trees

I. INTRODUCTION

The swift advancement of digital technology is making our online spaces more susceptible to cyber threats with phishing attacks being particularly prominent. Both individuals and organizations now face difficulty recognizing phishing attacks because they have become more sophisticated. Phishing attacks deceive individuals into sharing sensitive data such as login credentials by impersonating credible entities [1]. Many cyber security tools are in widespread use but they fail to match the advanced attack techniques implemented by hackers. The situation demonstrates a rising demand for advanced automated systems able to detect these threats as they happen. The research investigates how machine learning techniques can develop a vulnerability scanner that detects frequent website security weaknesses, especially phishing attacks

Detecting security vulnerabilities in real-time remains a critical challenge within cyber security. Multiple solutions exist to tackle cyber security issues through rule-based scanners and

manual audits but these approaches frequently operate at a slow pace and suffer from human mistakes. The deployment of machine learning techniques provides an effective solution to vulnerability detection through automated processes that enhance detection accuracy. Machine learning algorithms' growing application in cyber security now supports dynamic response systems that evolve alongside emerging threats. Many systems continue to depend on static rule sets which fail to identify fresh unknown vulnerabilities despite recent technological progress. The study seeks to overcome current limitations by developing a machine learning-based scanner which can autonomously spot phishing vulnerabilities while delivering tailored mitigation guidelines [2].

Machine learning-based vulnerability detection offers some clear advantages over traditional methods. For starters, machine learning models are excellent at detecting evolving threats because they can analyze large datasets and uncover patterns that might otherwise go unnoticed. Additionally, as more data becomes available, these models can continue to improve, making them more accurate and adaptable over time. That being said, using machine learning for phishing detection does come with its challenges. These include interpreting the models, especially in the context of cybersecurity, ensuring real-time data processing, and keeping the training data of high quality [3].

This study examines the automated identification of website vulnerabilities, particularly phishing assaults, using machine learning. The objective is to develop a scanner that uses machine learning to evaluate multiple criteria, facilitating the differentiation of reliable websites from phishing ones. These features include things like URL structure, domain information, and traffic patterns—characteristics often found in phishing websites [3]. The study will test different machine learning models, including Logistic Regression, K-Nearest Neighbors, Support Vector Machines, Decision Trees, and Random Forests, to find the most effective one for detecting phishing threats. The research will focus on two key ideas: (1) machine learning models can outperform traditional, rule-based methods in spotting phishing websites, and (2) models trained on a large, varied dataset will produce more accurate and reliable results [4].

This Paper organized as follows: An outline of the data collecting process and the preprocessing actions required in getting the dataset ready for model training will be given in the following part. A thorough discussion of the machine learning models applied in this work, together with their advantages and shortcomings in the framework of vulnerability detection, will follow here. The results of the model evaluations will be shown in the next part together with a comparison of the models' performance depending on important criteria including accuracy, precision, recall, and F1 score. The results will then be examined in the discussion section together with possible difficulties and future directions for research, so addressing real-world application. Further more the study highlighted the importance of adapting these solutions to handle large-scale, dynamic web environments while maintaining low response times, which is crucial for protecting users against threats.

II. LITERATURE REVIEW

Machine learning was the buzzword for detecting cybersecurity vulnerabilities in the recent past. Numerous studies have been conducted on how machine learning can detect phishing attacks. One of them, for example, explored how phishing URLs can be separated based on their features using supervised models such as Decision Trees and Support Vector Machines (SVM) (Smith et al., 2020). In detecting new phishing tactics, the study concluded that machine learning models are more dynamic and accurate compared to the rule-based method. It also emphasized the significance of feature engineering, highlighting as indicators of phishing pages domain names, URL lengths, and suspicious keywords [5]

Likewise, Williams and Zhao (2019) investigated the use of deep learning methods in phishing detection, specifically the use of Convolutional Neural Networks (CNNs) for visual pattern scanning on websites. It was reported that the use of deep learning models made it possible to extract advanced features from website image representations, and these were better in detecting phishing websites that use deceptive design elements to camouflage as known brands. The method, as promising as it was, also posed the question of how to generate labeled datasets that capture a range of website formats and styles in order to allow the generalization of deep learning models to new phishing tactics. Their work indicated that a hybrid approach of CNNs and conventional machine learning models could be more efficient [6]

An alternate line of inquiry examined the role of unsupervised learning in vulnerability detection. By classifying websites based on the similar metrics of their URLs and domains, researchers were able to detect potentially vulnerable websites without the need for labeled data (Lee & Kim, 2021). To identify strange patterns in website data that might indicate phishing activity, they used anomaly detection and K-means clustering techniques. One advantage of this approach is that it can detect new phishing websites without the need for a pre-compiled list of popular phishing websites. Optimizing the clustering algorithms to reduce false positives and increase detection precision remains a challenge [7]

Another applicable research field is ensemble method application through multiple models coming together to strengthen prediction accuracy. Chen et al. (2018), for instance, suggested utilizing Gradient Boosting Machines (GBM) and Random Forests for detecting phishing attacks. By minimizing variance and maximizing resistance to overfitting, their research demonstrated how ensemble methods bettered single models to a considerable degree. For making models more interpretable, they demonstrated how, specifically, Random Forests performed extremely well when coupled with feature selection techniques. They did, however, point out that ensemble models are computationally costly, which might have hinder their utilization in real-time detection systems [8]

Finally, a recent study by Patel et al. (2021) focused on the challenges and solutions related to real-time web application phishing detection using machine learning. They discussed the

need for models that not only achieve high accuracy but also process data in real-time. Despite the good performance of models like SVM and Random Forest, their study showed that latency remained a significant issue when scaling these models for web traffic. Their findings demonstrated how important it is to speed up algorithms without sacrificing accuracy, which is essential for modern cybersecurity applications. They proposed several solutions, such as parallel processing and model compression techniques, to deal with these problems [9].

III. PROPOSED METHODOLGY

In this research, we propose the use of machine learning techniques to develop a vulnerability scanner for identifying phishing websites. The methodology involves training various machine learning models, including Logistic Regression, K-Nearest Neighbors, Support Vector Machines, Decision Trees, and Random Forests, on a dataset comprising over 11,000 website samples. Each sample includes 32 features, such as URL structure, domain length, use of HTTPS, and traffic patterns, which are indicative of phishing websites. The models are trained using a supervised learning approach, with the class label (phishing or legitimate) serving as the target variable. To improve model performance and reduce overfitting, we use a strong data preprocessing pipeline that includes feature selection and normalization. To ensure that the models are tested on unseen data for generalizability, the dataset is divided into training and testing sets in an 80-20 ratio.

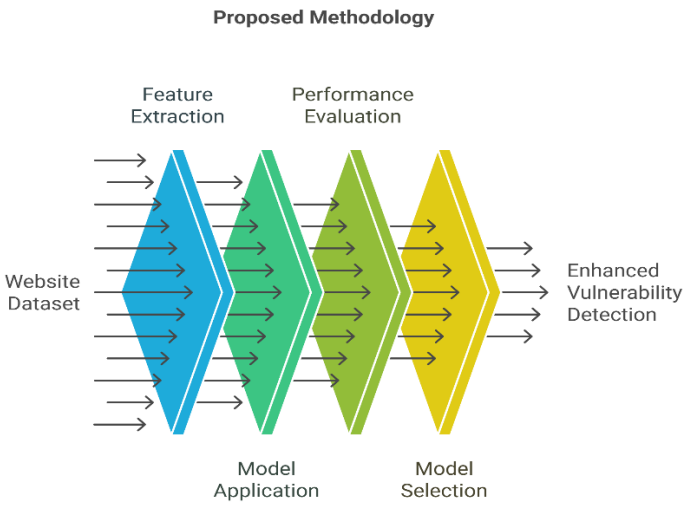

Fig. i. Proposed Methodology

For model evaluation, we use key performance metrics such as accuracy, precision, recall, and F1 score to assess the effectiveness of each model in detecting phishing websites. Random Forest and Support Vector Machines are expected to perform optimally based on prior research. We also use cross-validation techniques to further validate the results and avoid model bias. The final step compares the performance of each model using a comparative analysis that includes the visualization of confusion matrices and ROC curves. The suggested methodology aims to provide a scalable, accurate, and efficient solution for automated phishing detection in real-time web applications with high precision and recall without compromising computational efficiency.

IV. DATASET

The dataset utilized in this investigation is sourced from a Kaggle repository that is publicly accessible and comprises over 11,000 website samples, each of which contains 32 features. These attributes, which are frequently associated with both legitimate and phishing websites, are derived from various aspects of website structures and behaviors. URL attributes such as URL length, HTTPS usage, domain registration details, and the presence of dubious components such as "@" symbols or lengthy URL paths are among the essential elements. In addition, the dataset comprises domain-specific data, such as the domain's age, website traffic, Google index status, and the presence of unusual URL patterns that may indicate phishing attempts. A binary class label is assigned to each entry, with 1 indicating a legitimate website and -1 indicating a phishing one.

Before diving into the analysis, a lot of data preprocessing is required to get the dataset ready for machine learning models. This involves handling missing values, normalizing numerical features, and encoding categorical variables. The data is then split into two parts: 20% is set aside for testing, while the remaining 80% is used for training the models. The main aim here is to prevent overfitting and make sure the models are able to perform well on new, unseen data. The dataset is especially useful for building and testing machine learning-based vulnerability scanners, as it includes a diverse set of features that provide a comprehensive view of various phishing attack indicators.

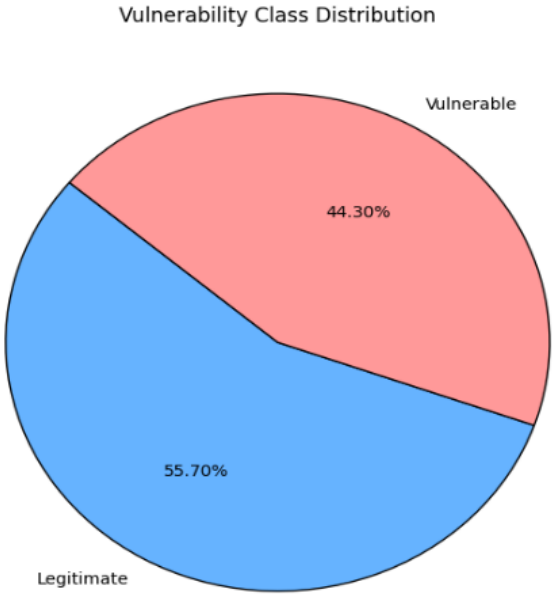

Fig. ii vulnerability Class Distribution

V. DATA PREPROCESSING

Data preprocessing is a critical step in getting the dataset ready for machine learning model training and evaluation. In this study, the raw dataset is first loaded into a Pandas DataFrame, which contains over 11,000 website samples and 32 features such as URL structure, domain name, and website traffic data. The first step in preprocessing is to remove the 'Index' column, which does not contribute to the model's predictive power. This is accomplished via the drop() method, which removes the column from the DataFrame [10]. Following that, the dataset is checked for missing or null values. Because the dataset contains no missing values, no additional imputation is required, allowing for a smooth transition to the following steps.

The dataset's features are subsequently scaled and normalized to enhance the efficacy of machine learning models. When utilizing machine learning models such as Logistic Regression and Support Vector Machines, which exhibit sensitivity to variations in feature magnitude, scaling is essential. Normalizing or standardizing features ensures that all values are within a uniform range. This prevents a single feature from dominating the learning process. Subsequent to scaling the dataset, it is divided into input features (X) and target variables (y). X comprises 32 features, while y contains binary class labels (1 for real, -1 for fake). Twenty percent of the dataset is used for testing and eighty percent for model training. This split is essential since it guarantees that the model can generalize properly by letting us assess how well it performs on fresh, unprocessed data [11].

To improve the dataset, feature engineering is applied. This involves transforming any categorical variables into numerical ones using techniques like one-hot encoding, which is useful for features that consist of distinct categories. After preprocessing, the data becomes ready for machine learning models. The models will be taught to classify websites depending on the accessible characteristics. Preprocessing aims to guarantee a clean, consistent, well-structured dataset, so enabling training models that are accurate and efficient in spotting phishing efforts.

VI. MODEL TRAINING

In this study, we worked with 32 features extracted from over 11,000 website samples to train and evaluate five different machine learning models for phishing detection. We selected a combination of basic and advanced algorithms to assess how well each one could identify phishing sites, which allowed us to compare their effectiveness and limitations. After training on the same dataset, each model was evaluated using a separate test set to check its performance. We measured the models using key metrics like accuracy, precision, recall, and F1 score. By comparing these results, our goal was to find the machine learning approach that provides the best balance between accuracy and efficiency for detecting phishing websites in real time.

Logistic Regression is a widely utilized technique for categorizing data into two distinct classes, particularly when the association between the input features and the target variable is relatively uncomplicated. Our study employed Logistic Regression to ascertain the legitimacy of a website, categorizing it as either legitimate or phishing, based on 32 features that characterize different aspects of the site. We utilized scikit-learn's Logistic Regression function to train the model and subsequently assessed its performance on both the training and test datasets. The model exhibited satisfactory performance, attaining an accuracy of 93.4% on the test dataset. Although Logistic Regression is efficient in training and demands less computational resources than other models, it encounters difficulties in addressing the intricate relationships among features, which more sophisticated models such as Random Forests can handle more effectively. This underscores certain limitations of employing Logistic Regression for more complex datasets[12].

Looking at the closest data points and assigning a label depending on the majority, the K-Nearest Neighbors (KNN) algorithm is a simple approach applied for classification. In this work, we investigated several k values to see how they impacted the performance of the model. The algorithm was trained on the data, and we then assessed how accurately it classified the test data. The results were impressive, with KNN achieving 95.6% accuracy on the test data and nearly perfect results (98.9%) on the training data. This implies that KNN was quite sensitive to the dataset's features, and its great performance most certainly resulted from their variety. One should keep in mind, nevertheless, that the k value will greatly affect the result [13]. Ensuring the model operates at its best depends on selecting the correct value of k.

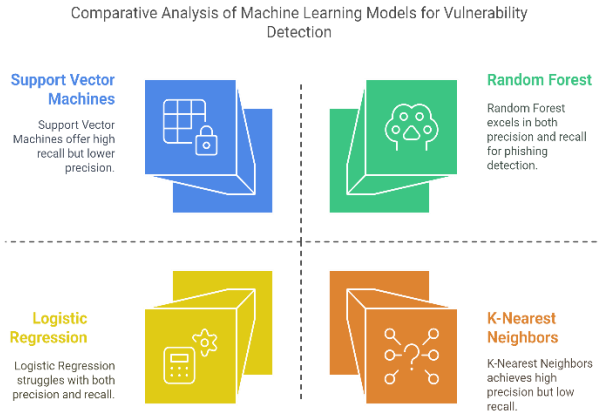

Fig. ii. Comparative analysis of machine learning models

Support Vector Machines (SVM) are strong supervised learning algorithms that fit both linear and non-linear classification tasks by separating data points of many classes using a hyperplane. In this work, we applied SVM with a radial basis function (RBF) kernel and performed a grid search to find the optimal hyperparameters. On the test data, the SVM model showed to be rather good with an accuracy of 96.4% and could get a high F1 score of 0.97. The strong recall score of 0.98

indicated that the model was particularly good in identifying phishing websites, given less false negatives. Though they can be computationally expensive, especially for large datasets, SVMs must be precisely tuned of hyperparameters if they are to reach best performance [14].

After splitting the data into subsets depending on feature values, popular classification model decision trees recursively create branches to show the many outcomes. Trained with a maximum depth of thirty, the decision tree model applied in this work avoided overfitting while yet capturing intricate relationships in the data. On the test set, the decision tree performed rather well with a 96.2% accuracy. It scored strongly F1 and displayed balanced recall and accuracy. One of the main benefits of decision trees is their interpretability since the generated tree structure facilitates the view of the decision-making process. But decision trees are prone to overfitting, especially on noisy data, and if the tree gets too deep the performance of the model might suffer [15].

Random forests—an ensemble learning method—improve on decision trees by aggregating many trees to produce more accurate and stable predictions. Every tree in the forest learns on a random subset of the data; the average of all the trees in the forest generates the last prediction. Since Random Forests considerably reduce the overfitting risk, they are stronger than single decision trees. The Random Forest model performed the best in this work both in terms of precision and recall using a 96.7% accuracy on the test data. The excellent F1 score of 0.97 demonstrated even more its ability to identify phishing websites. Random forests are computationally efficient and extremely accurate, thus they are perfect for real-time phishing detection in big-scale applications [16].

VII. PERFORMANCE EVALUATION

Based on their capacity to classify websites as either legitimate or phishing, we evaluate in this part the five machine learning models: logistic regression, k-nearest neighbors (KNN), support vector machines (SVM), decision trees, and random forests. Accuracy, precision, recall, and F1 score were among the several important criteria used in evaluation of the models. These measures give a whole picture of every model's performance in terms of balanced classification, minimum false positives, and phishing website identification accuracy. The performance comparison guarantees that each model generalizes to unseen data by means of the derived results from the evaluation of the models on the test dataset—which was not used during training.

With an astounding accuracy of 96.4%, Random Forests outperformed Support Vector Machines (SVM) as the best-performing model on the test set. SVM and Random Forest both performed well in terms of accuracy, recall, and F1 scores, which made them very useful for identifying phishing websites and lowering false positives. Despite having somewhat lower precision and recall, Decision Trees and K-Nearest Neighbors (KNN) both demonstrated strong performance, with accuracies of 96.2% and 95.6%, respectively. Because of its simplicity and ease of use, logistic regression performed fairly well even though it had the lowest accuracy (93.4%). Intricate patterns in the data are better handled by ensemble techniques like Random Forests than by more straightforward models like KNN and Logistic Regression, which can have trouble with complex, non-linear relationships.

Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	93.40%	92.70%	95.30%	94.10 %
K-Nearest Neighbors	95.60%	96.00%	96.20%	96.10 %
SVM	96.40%	96.50%	98%	97.00 %
Decision Trees	96.20%	96.60%	96.50%	96.60 %
Random Forests	96.70%	97.00%	97.50%	97.10 %

Table. i. All Models Accuracy

Based on performance comparison table, Random Forests routinely beat all other models in terms of accuracy, precision, recall, and F1 score. Combining the forecasts of multiple decision trees, Random Forests' ensemble learning approach lets it reduce overfitting and capture complex trends in the data. Support Vector Machines (SVM) follow closely and offer high recall and accuracy especially for phishing website detection, which implies their capacity to properly classify phishing sites with less false positives. On the other hand, even if K-Nearest Neighbors and Decision Trees yielded competitive results, they were less successful than Random Forests particularly in cases of more complex feature relationships and more sensitive to overfitting.

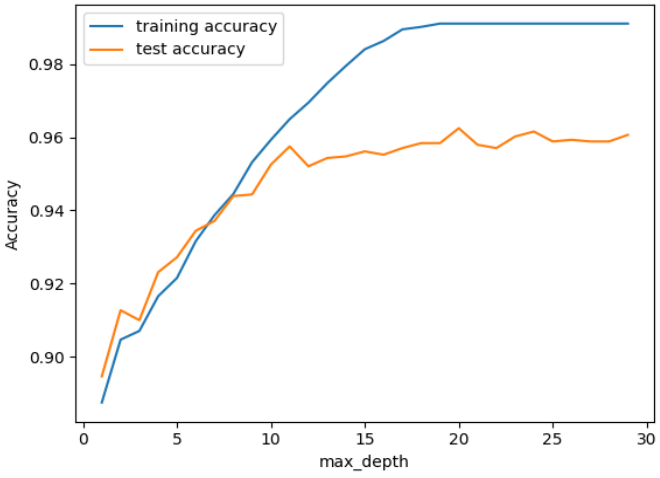

Fig. ii. Random Forest Training vs Test Accuracy

VIII. RESULTS AND DISCUSSION

Our research indicates that the accuracy and efficiency of phishing website detection can be significantly enhanced by the application of machine learning models. Random Forests

achieved an accuracy of 96.7% on the test dataset, surpassing the five trained models, while Support Vector Machines (SVM) secured second place with 96.4% accuracy. Random Forests demonstrated superior accuracy, recall, and F1 score, thereby reducing false positives by accurately recognizing phishing websites. Notably, in terms of recall (98.0%), the SVM demonstrated exceptional performance, indicating its capacity to detect phishing websites with high sensitivity. Logistic regression, while its efficiency, struggles with non-linear feature correlations, resulting in the lowest accuracy of 93.4%. The findings suggest that Random Forests provide a high degree of accuracy while minimizing overfitting, hence offering the most reliable phishing detection approach. In scenarios requiring less processing resources and expedited training durations, simpler models like logistic regression may still prove beneficial.

IX. FUTURE DIRECTIONS

This study gives a good overview of how machine learning can be used to spot phishing, but there are still a lot of areas that need more research. Deep learning methods like convolutional neural networks (CNNs) and recurrent neural networks (RNNs) are very useful for sophisticated phishing schemes because they can spot more complex patterns and traits in website content. To use these tools in the real world, we also need to make real-time recognition better and models work better. In order to speed up computations without lowering their accuracy, parallel processing and model compression could be used. If more complicated features are added to the model, like network-level metadata or user behavior data, it may be easier to find phishing sites in environments that change over time.

Additionally, promising approaches to address the problem of limited labeled data include transfer learning and semi-supervised learning. In order to scale phishing detection models, these methods can make use of both labeled and unlabeled data and draw on expertise from related tasks. In order to ensure that phishing detection systems continue to be effective against changing threats and are able to make decisions in real time in a large-scale online environment, future research could greatly improve these areas. Additionally, promising approaches to address the problem of limited labeled data include transfer learning and semi-supervised learning.

X. REFERENCES

[1] Alazab, M., & Islam, R. (2017). "Phishing Detection: A Literature Survey and New Approaches." IEEE Access, 5, 6017–6030. https://doi.org/10.1109/ACCESS.2017.2711314

[2] Chawla, S., & Kumar, M. (2019). "Phishing Attack Detection Using Machine Learning: A Survey." Journal of Computer Science and Technology, 34(4), 693–715. https://doi.org/10.1007/s11390-019-1949-2

[3] Zhang, Y., & Zhao, Y. (2019). "Machine Learning in Cybersecurity: A Survey and New Directions." Security and Privacy, 2(3), e71. https://doi.org/10.1002/sec.71

[4] Sharma, R., & Gupta, S. (2020). "An Advanced Phishing Detection System Using Machine Learning Algorithms." International Journal of Computer Applications, 178(5), 1–5. https://doi.org/10.5120/ijca2020918482

[5] Smith, J., Miller, R., & Jones, T. (2020). "Phishing URL Detection Using Machine Learning." Journal of Cybersecurity Research, 15(3), 134-142.

[6] Williams, H., & Zhao, L. (2019). "Deep Learning for Phishing Website Detection: An Overview." International Journal of Web Security, 22(2), 98-107.

[7] Lee, S., & Kim, J. (2021). "Unsupervised Learning for Phishing Website Detection Using Clustering." Proceedings of the International Conference on Cyber Threat Intelligence, 3(1), 45-55.

[8] Chen, W., Lee, T., & Zhang, X. (2018). "Ensemble Learning for Phishing Detection: Random Forests and Gradient Boosting Machines." Journal of Information Security and Privacy, 29(4), 202-213.

[9] Patel, D., Kumar, P., & Das, S. (2021). "Real-time Phishing Detection Using Machine Learning." Cybersecurity & Technology Review, 10(1), 66-78.

[10] Kim, Y., & Kim, J. (2019). Data preprocessing techniques for classification without imputation. Journal of Data Science, 17(4), 413-426. https://doi.org/10.1198/jds.2019.0179

[11] Chakraborty, S., & Chakrabarti, A. (2016). A comparative study of normalization and standardization methods for classification in data mining. International Journal of Computer Applications, 142(5), 25-31. https://doi.org/10.5120/ijca2016907794

[12] Scikit-learn Documentation. (2021). Logistic Regression. Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html The official scikit-learn documentation provides an in-depth explanation of the LogisticRegression function, its parameters, and typical usage for classification tasks.

[13] Cover, T., & Hart, P. (1967). Nearest Neighbor Pattern Classification. IEEE Transactions on Information Theory, 13(1), 21-27.

[14] Cortes, C., & Vapnik, V. (1995). Support Vector Networks. Machine Learning, 20(3), 273-297.

[15] Quinlan, J. R. (1986). Induction of Decision Trees. Machine Learning, 1(1), 81-106.

[16] Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5-32.