Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2024) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

9th International Conference on Computer Science and Computational Intelligence 2024 (ICCSCI 2024)

Using Machine Learning Models to detect Phishing URLs based on Different Feature Filtering Methods

Bryan Orville Audric[a*], Edrico Putra Pramana[a], Afdhal Kurniawan[a], Diana[a]

Computer Science Department, School of Computer Science, Bina Nusantara University, Jl. Jalur Sutera Bar. No.Kav. 21, Tangerang 15143, Indonesia

Abstract

The recent surge in online platforms has escalated cyber security threats, particularly phishing and malware attacks aimed at acquiring users' sensitive data. In this context, automated classification emerges as a promising solution. This study aims to evaluate the performance of several feature filter methods using various machine learning models. These feature filter methods were applied to the utilized dataset, with Random Forest, LightGBM, and XGBoost employed to assess the performance of each method. The research findings indicate that the Pearson correlation feature filter method obtained the fewest features yet exhibited slightly superior predictive performance compared to other methods.

Keywords: Cyber security; Feature filter methods; Automated classification; Phishing; Machine learning

1. Introduction

In recent years, the proliferation of online platforms has led to an increase in cyber threats, with phishing and malware attacks proving to be a significant threat to internet security. Phishing attacks can be defined as fraudulent activities that go beyond traditional data theft to encompass a range of malicious behaviors, including cyber terrorism, hacktivism, damaging reputations, espionage, and nation-state attack to trick individuals into divulging sensitive data, such as login credentials or bank information [1]. Related to phishing, malware refers to any software

[*]Corresponding Author. Tel. +6285933036006
E-mail: bryan.audric@binus.ac.id

created to carry out harmful actions on devices like computers and smartphones. This includes viruses, worms, Trojan horses, rootkits, and ransomware, each designed to affect systems differently, such as causing damage or stealing data [2]. To address this, researchers have increasingly use machine learning models to automate phishing detection as machine learning models offer the capability of adapting and learning a specific pattern that can indicate phishing activity from vast amounts of data. This paper focuses on the use of different features filtering methods that will help preprocessing process by selecting the most useful and informative features while discarding noise or redundant information.

In the context of selecting feature filter methods for training machine learning models, despite some methods showing superior performance, there remains considerable uncertainty regarding which approach is the most effective. Additionally, there is limited research on how factors like dataset characteristics and domain-specific attributes affect the effectiveness of these methods. This underscores the necessity for systematic investigation to understand the impact of various feature filtering techniques on dataset selection for model training, particularly in domains such as phishing detection, to improve predictive analytics' robustness and efficiency. The aim of this paper is to conduct an evaluation and comparison among several feature filters used in phishing detection using machine learning models. The paper seeks to identify the most effective feature filters in detecting phishing websites, considering factors such as prediction accuracy, precision, recall and f1 score. Thus, the paper aims to provide insights into which approach is most suitable for developing an effective and efficient phishing detection system. The research hypothesis posits that there is a significant difference in the performance of models trained with different feature filters, both in terms of prediction accuracy and training time. The experiments conducted will test this hypothesis using relevant statistical methods to validate significant differences in model performance across various feature filters.

2. Related Works

There are several attempts at creating an accurate phishing URL detection. Some of which used a specific filter method to filter out unnecessary features before inserting into a model. This research aimed to detect phishing websites using machine learning with filter features, filtered by Pearson correlation. It achieves the highest accuracy rate of 96.3% [3]. The research utilizes Random Forest Regressor for feature selection and ensemble methods as the model which produces a superior performance up to 95% accuracy [4]. This study utilized several filter methods to rank and select features for experiments aimed at detecting phishing websites. These methods included the Chi-Square Filter, Pearson Correlation Filter, Information Gain Filter, and Relief Filter. The Chi-Square Filter ranked features based on their relevance to phishing detection, demonstrating improved accuracy with a smaller feature set when used in conjunction with the Random Forest classifier. The study's analysis highlighted the Random Forest classifier's superior accuracy across various feature classes, with the Chi-Square Filter notably outperforming others with a higher accuracy of 96.83% while utilizing fewer features [5].

Other study has proved that a machine learning algorithm with no feature selection can also detect phishing URL accurately. The study focused on classifying URLs into phishing, suspicious, or legitimate categories using machine learning techniques. Four classifiers were evaluated: decision tree, Naïve Bayes' classifier, SVM, and Neural Network. The classifiers were tested, and the results showed successful classification of websites with highest accuracy of 90% achieved by pruned decision tree [6]. This study employed machine learning algorithms in detecting phishing websites, with the XGBoost algorithm achieving the highest accuracy of 94% in the training phase and 91% in the testing phase. Other algorithms like Multilayer Perceptron, Random Forest, and Decision Tree also showed accuracies ranging from 90% to 91% [7]. The research uses seven machine learning algorithms such as Decision Tree, Adaboost, Kstar, KNN, Random Forest, SMO and Naïve Bayes and all models reach over 90% accuracy [8]. This research focuses on the detection of phishing websites using machine learning methods. It compares and evaluates the performance of various machine learning models, such as Logistic Regression, Decision Tree, Random Forest, Ada-Boost, Support Vector Machine, KNN, Neural Networks, Gradient Boosting, and XGBoost [9]. This study proposed machine learning model that is RNN-LSTM model. The result shows promising accuracy of 97% when identifying phishing URLs [10]. The ensemble models, particularly Ensemble Model 5 and Ensemble Model 6, achieved high accuracy scores of 99.95% and 99.94%, respectively [11]. The research utilized a machine learning-based technique for detecting phishing websites, specifically employing a Support Vector

Machine (SVM) classifier. The Support Vector Machine model achieved an accuracy of 95.66% in detecting phishing websites [12].

Other researchers also used deep learning to create an accurate model. This study proposed a real-time phishing website detection framework based on deep learning, achieving a remarkable accuracy of 99.18% with the RNN-GRU model on the KPT-12 dataset [13]. This research aimed to enhance phishing URLs detection by applying parallel processing to machine learning and deep learning models, providing insights into the comparison between sequential and parallel ML execution [14].

3. Methodology

There are several processes that need to be followed based on Fig. 1. The process starts with gathering a dataset that will then be extracted. The result would be a numerous feature that represents the data. The features are then filtered using Pearson correlation, chi square and Linear Discriminant Analysis method. Both filtration methods will produce two sets of features that will be split into a test and training set, which will be used for the machine learning model to train and predict. The results of training from two sets of features are then evaluated and compared.

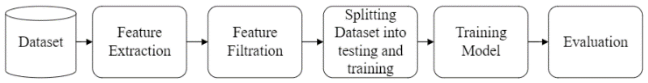

Fig. 1. Research Process

3.1. Dataset

The dataset used for this research is taken from Kaggle. The dataset consists of two columns which are a feature and a label. The label consists of benign or safe, phishing, malicious and defacement. There are 651,191 URLs, out of which are 428103 benign or safe URLs, 96457 defacement URLs, 94111 phishing URLs, and 32520 malware URLs.

3.2. Feature Extraction

Feature extraction methods for phishing URLs are employed to extract several relevant characteristics or attributes from the URLs themselves are extracted to distinguish between legitimate, phishing, benign or defacement websites.

Table 1. Features extracted.

No.	Feature	Description
1	have_ip_address	IP Address in URL
2	url_length	Number of Characters in URL
3	shortening_service	URL shortening service used in URL
4	count_attrate_symbol	Count of at symbol (@)
5	count_http_url	HTTP used in URL
6	count_https_url	HTTPS used in URL
7	count_dot_url	Number of dots (.) in URL
8	count_hyphen_url	Number of hyphen (-) in URL
9	count_underline_url	Number of underline (_) in URL
10	count_question_url	Number of question mark (?) in URL
11	count_slash_url	Number of path in URL (/)
12	count_amp_url	Number of ampersand symbol (&)

		in URL
13	count_exclam_url	Number of exclamation mark (!) in URL
14	count_percent_symbol	Number of percent symbol (%) in URL
15	count_space	Number of whitespace ( ) in URL
16	count_comma	Number of comma (,) in URL
17	count_tilde	Number of tilde (~) in URL
18	count_plus	Number of plus (+) in URL
19	count_asterisk	Number of asterisk (*) in URL
20	count_hashtag	Number of hashtag (#) in URL
21	count_dollar	Number of dollar symbol ($) in URL
22	count_equal	Number of equal symbol (=) in URL
23	find_email	Email used in URL or not
24	age_of_domain	The age of domain measure by subtracting current date by creation date
25	domain_registration_length	The registration length of domain measured by subtracting expiration date by creation date

3.3. Pearson Correlation

Pearson's correlation coefficient, a statistical metric, is instrumental in the feature selection process. This coefficient quantifies the linear relationship between features and the target variable, discerning their relevance. By calculating correlation coefficients for each feature in the dataset with respect to the target variable, researchers can gauge their strength of association. Features exhibiting strong correlations, closer to 1 or -1, are considered more pertinent for distinguishing phishing from legitimate URLs. Conversely, features with coefficients near 0 indicate minimal linear relationship and are thus deemed less relevant for classification. The formula for calculating the Pearson correlation coefficient between two variables is given by:

$$r_{xy} = \frac{\sum \left(X_i - \overline{X}\right)\left(Y_i - \overline{Y}\right)}{\sqrt{\sum \left(X_i - \overline{X}\right)^2 \left(Y_i - \overline{Y}\right)^2}} \tag{1}$$

Where r_{xy} is the Pearson correlation coefficient between variables X and Y, X_i, and Y_i are individual values of variable X and Y, and $\bar{X}$ and $\bar{Y}$ are the means of the values of X and Y, respectively. This formula measures the extent to which variables X and Y vary together, providing a deeper understanding of the relationship between features in the dataset and the target variable. Pearson correlation coefficient is broadly admitted in the feature selection algorithm to determine the best feature set [15]. The features filtered using Pearson correlation are shown in Table 2.

Table 2. Features extracted using Pearson correlation.

Features	Correlation Coefficient	Features
have_ip_address	0.18747392882130517	have_ip_address
count_http_url	0.4429366538272832	count_http_url
count_https_url	0.2609873683185638	count_https_url

count_dot_url	0.17122968496403157	count_dot_url
count_hyphen_url	-0.14552074907402165	count_hyphen_url
count_slash_url	-0.15012277812306896	count_slash_url
age_of_domain	-0.3341334004661094	age_of_domain
domain_registration_length	-0.34056981170764866	domain_registration_length

3.4. Chi Square

Chi-square (χ^2) test is a statistical method utilized in feature selection, focusing on assessing the independence between categorical variables. In the context of phishing URL detection, features are often categorical variables representing different attributes of URLs. The Chi-square test calculates the statistical significance of the association between each feature and the target variable (i.e., whether a URL is phishing or legitimate). It is computed using the formula:

$$X^2 = \sum \frac{\left(O_i - E_i\right)^2}{E_i} \tag{2}$$

Where χ^2 is the Chi-Square statistic, O_i is the observed frequency, and E_i is the expected frequency under the null hypothesis of independence. This statistical test helps to identify features that have a significant association with the target variable, aiding in the discrimination between phishing and legitimate URLs. Clearly, the Chi-square value is small if the observed value is close to the expected value. Therefore, features are more independent if Chi-Square value is small [16]. The features filtered using Chi Square are shown in Table 3.

Table 3. Features extracted using Chi Square.

Features	Chi Square Value	P value
have_ip_address	164885.72833997948	0.0
url_length	1214541.3669727864	0.0
count_http_url	261888.63426800573	0.0
count_https_url	68381.37173997957	0.0
count_hyphen_url	84601.82413794301	0.0
count_equal	521311.4973838545	0.0.
count_amp_url	510604.7321766937	0.0
count_percent_symbol	438337.15587056597	0.0
age_of_domain	12648816.120678913	0.0
domain_regstration_length	14934171.097294644	0.0

3.5. Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a dimensionality reduction technique used in machine learning and pattern classification to transform high-dimensional data into a lower-dimensional space. The main goal of LDA is to maximize the separability between different classes in the data by projecting it onto a new space. This is achieved by calculating the between-class and within-class variations in the data and constructing a lower-dimensional space that maximizes the ratio of between-class variance to within-class variance. By doing so, LDA aims to find a new feature space where the data points can be effectively separated based on their class labels. This process helps in extracting discriminative features that are crucial for classification tasks, making it easier to classify new data points accurately [17]. There are 4 subsets of features produced using LDA. Subset 3, shown in Table IV is taken as selected features.

Table 4. Features extracted using LDA.

No	Features
1	have_ip_address
2	shortening_service
3	count_attrate_symbol
4	count_http_url
5	count_https_url
6	count_dot_url
7	count_hyphen_url
8	count_underline_url
9	count_question_url
10	count_slash_url
11	count_equal_url
12	count_amp_url
13	count_exclam_url
14	count_percent_symbol
15	count_space
16	count_comma
17	count_tilde
18	count_plus
19	count_asterisk
20	count_hashtag
21	count_dollar
22	find_email

3.6. Random Forest

Random Forest is a widely used ensemble learning model comprised of multiple decision trees. Each tree is constructed independently and randomly, with a subset of features selected randomly during the tree-building process. This feature selection technique helps mitigate overfitting and decorrelate the trees within the ensemble. Through bootstrap aggregating or bagging, Random Forest generates multiple bootstrap samples from the training data and builds a decision tree on each sample. During classification tasks, the model employs a majority voting mechanism, where each tree "votes" for the most prevalent class [18].

3.7. XGBoost

XGBoost is a scalable tree boosting system widely used in machine learning for its effectiveness and efficiency. It works by sequentially building an ensemble of decision trees, where each tree corrects the errors of the previous ones through boosting. By optimizing an objective function that combines a loss function and regularization term, XGBoost minimizes errors while preventing overfitting. Leveraging gradient boosting, XGBoost fits trees to the gradient of the loss function, learning from previous mistakes. With parallel and distributed computing capabilities, XGBoost can efficiently utilize multiple CPU cores and scale to large datasets. It also provides insights into feature importance and is designed for scalability, handling billions of examples with minimal resources [19].

3.8. LightGBM Classifier

LightGBM is a state-of-the-art Gradient Boosting Decision Tree (GBDT) algorithm that revolutionizes traditional implementations by introducing novel techniques like Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB). GOSS optimizes the training process by sampling data instances based on gradient values, focusing on instances with significant gradients to accurately estimate information gain while reducing computational overhead. EFB enhances efficiency by bundling related features together, improving cache hit rates and spatial locality. By leveraging these techniques, LightGBM accelerates model training, reduces memory consumption, and maintains high accuracy, making it a powerful tool for handling large feature dimensions and datasets in machine learning tasks [20].

3.9. Accuracy

Accuracy is one of the commonly used evaluation metrics in classification modeling to assess how well a model can classify data correctly overall. In this context, "correct" means the model's predictions match the true labels of the observed data. Accuracy is calculated by dividing the number of correct predictions by the total number of predictions made by the model. The formula for calculating the accuracy of a classification model is as follows:

$$Accuracy = \frac{True\ Positives + True\ Negatives}{True\ Positives + True\ Negatives + False\ Positives + False\ Negatives} \tag{3}$$

3.10. Precision

Precision is a crucial evaluation metric in classification tasks, particularly in scenarios where minimizing false positives is important. It measures the accuracy of positive predictions made by the model, i.e., the proportion of correctly predicted positive instances out of all instances predicted as positive. The formula for calculating precision in a binary classification setting is as follows:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{4}$$

3.11. Recall Score

Recall, also known as sensitivity or true positive rate, is an essential evaluation metric in classification tasks, especially when it's crucial to capture all positive instances. It measures the ability of the model to correctly identify positive instances from all actual positive instances in the dataset. The formula for calculating recall in a binary classification setting is as follows:

$$Recall\ Score = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{5}$$

3.12. F1-Score

The F1 score is a commonly used evaluation metric in classification tasks, which combines precision and recall into a single metric. It provides a balance between precision and recall, making it particularly useful when both false positives and false negatives need to be minimized. The formula for calculating the F1 score is:

$$F1\ Score = \frac{2 \times\ Precision \times\ Recall}{Precision +\ Recall} \tag{6}$$

Where Precision and Recall are as previously defined. This formula computes the harmonic mean of precision and recall, emphasizing the balance between the two metrics. The F1 score ranges from 0 to 1, where a higher score indicates better performance. It is a useful metric for evaluating classification models, especially in scenarios where achieving a balance between precision and recall is important, such as in information retrieval systems or medical diagnosis.

4. Result and Discussion

4.1. Result

After extracting and filtering the data's features using pearson correlations, chi square and LDA, the data is then trained and predicted using random forest, XGBoost, and LightGBM classifier. The accuracy, precision, recall and f1 score are shown in Table V, VI, VII. Table V shows the evaluation results of features filtered from Pearson Correlation in which the average accuracy is around 94% across three models which are Random Forest Classifier, XGBoost Classifier and LightGBM Classifier. Amongst three classifier models that were used, Random Forest Classifier has the highest accuracy which is 95% while other classifier models top at 94% accuracy. Precision, recall and FI Score of Random Forest is relatively higher than other classifier models. The evaluation results on Table 5 also show that the precision, recall and F1 score in benign category is higher than other categories.

Table 5. Classification of filtered features using Pearson correlation.

Model	**Accuracy**	**Type**	**Precision**	**Recall**	**F1 Score**
Random Forest	0.95	Benign	0.97	0.99	0.98
		Defacement	0.90	0.97	0.94
		Phishing	0.87	0.72	0.79
		Malware	0.93	0.84	0.88
XGBoost	0.94	Benign	0.96	0.99	0.98
		Defacement	0.87	0.98	0.92
		Phishing	0.89	0.63	0.74
		Malware	0.94	0.78	0.85
LightGBM Classifier	0.94	Benign	0.96	0.99	0.98
		Defacement	0.87	0.97	0.92
		Phishing	0.88	0.62	0.73
		Malware	0.91	0.79	0.85

Table 6 shows the evaluation results of features filtered from Chi Square in which the average accuracy is around 91% across three models which are Random Forest Classifier, XGBoost Classifier and LightGBM Classifier. Amongst three classifier models that were used, Random Forest Classifier has the highest accuracy which is 92%. Other classifier models top their accuracy at 91%. Precision, recall and accuracy of Random Forest is also relatively higher than other classifier models. The evaluation results on Table 6 also showed that the preicsion, recall and FI score in benign category is higher than other categories.

Table 6. Classification of filtered features using chi square.

Model	**Accuracy**	**Type**	**Precision**	**Recall**	**F1 Score**
Random Forest	0.92	Benign	0.94	0.97	0.96
		Defacement	0.90	0.96	0.93

		Phishing	0.71	0.52	0.60
		Malware	0.92	0.84	0.88
XGBoost	0.91	Benign	0.93	0.98	0.96
		Defacement	0.87	0.96	0.92
		Phishing	0.74	0.43	0.54
		Malware	0.92	0.80	0.85
LightGBM Classifier	0.91	Benign	0.93	0.98	0.96
		Defacement	0.86	0.97	0.91
		Phishing	0.74	0.41	0.53
		Malware	0.93	0.78	0.84

Table 7 shows the evaluation results of features filtered from Linear Discriminant Analysis in which the average accuracy is around 93% across three models which are Random Forest Classifier, XGBoost Classifier and LightGBM Classifier. Amongst three classifier models that were used, Random Forest Classifier has the highest accuracy which is 94%. Other classifier models top their accuracy at 93%. Precision, recall and accuracy of Random Forest is also relatively higher than other classifier models. The evaluation results on Table VII also show that the preicsion, recall and FI score in benign category is higher than other categories.

Table 7. Classification of filtered features using chi square.

Model	**Accuracy**	**Type**	**Precision**	**Recall**	**F1 Score**
Random Forest	0.94	Benign	0.96	0.99	0.97
		Defacement	0.88	0.95	0.91
		Phishing	0.87	0.62	0.72
		Malware	0.92	0.77	0.84
XGBoost	0.93	Benign	0.95	0.99	0.97
		Defacement	0.88	0.94	0.91
		Phishing	0.85	0.61	0.71
		Malware	0.91	0.75	0.82
LightGBM Classifier	0.93	Benign	0.95	0.99	0.97
		Defacement	0.88	0.94	0.91
		Phishing	0.84	0.61	0.71
		Malware	0.91	0.75	0.82

4.2. Discussion

Based on Table 5, 6, and 7, the accuracy of each model based on every filtered feature with different method is above 0.9 with 0.95 being the highest which was achieved using random forest as a model and pearson correlations as the filtration method. The precision, recall and f1 score of every combination of models and filtration method for benign category appears to be higher than the other categories. The number of benign URL used for training is significantly higher than other URL categories which cause the evaluation score to be much higher. On the other hand, some models struggle to predict phishing URL correctly, with 0.41 in recall score being the lowest, which was achieved using XGBoost as a model and Chi Square as the filtration method. The combination of model and filtration methods that has the highest precision, recall and f1 score was also achieved by random forest and pearson correlations.

5. Conclusion

In this study, we evaluated the performance of several feature filter methods using various machine learning models to address cyber security threats, particularly phishing and malware attacks. Our main findings indicate that the Pearson correlation feature filter method showed slightly better performance in prediction compared to other methods, despite producing the fewest features. Furthermore, we found that by producing the fewest features, the Pearson correlation feature filter method provided an additional advantage in terms of cost efficiency and resource utilization.

However, it is important to note the limitations of the dataset used, as it may affect the generalization of findings and the usefulness of research outcomes. One of the major limitations of this study is the lack of phishing and malware link data, resulting in a low amount of training data for both labels. We recommend further research utilizing more adequate datasets, which can enhance the quality of cyber-attack detection and generate more accurate models. Thus, this research can make a greater contribution to enhancing overall system security.

References

[1] Z. Alkhalil, C. Hewage, L. Nawaf, and I. Khan, "Phishing Attacks: A Recent Comprehensive Study and a New Anatomy," *Frontiers in Computer Science*, vol. 3. Frontiers Media S.A., Mar. 09, 2021. doi: 10.3389/fcomp.2021.563060.

[2] O. Aslan and R. Samet, "A Comprehensive Review on Malware Detection Approaches," *IEEE Access*, vol. 8. Institute of Electrical and Electronics Engineers Inc., pp. 6249–6271, 2020. doi: 10.1109/ACCESS.2019.2963724.

[3] V. Aprelia Windarni, A. Ferdita Nugraha, S. Tri Atmaja Ramadhani, D. Anisa Istiqomah, F. Mahananing Puri, and A. Setiawan, "DETEKSI WEBSITE PHISHING MENGGUNAKAN TEKNIK FILTER PADA MODEL MACHINE LEARNING," 2023.

[4] A. A. Ubing, S. Kamilia, B. Jasmi, A. Abdullah, N. Z. Jhanjhi, and M. Supramaniam, "Phishing Website Detection: An Improved Accuracy through Feature Selection and Ensemble Learning," 2019. [Online]. Available: www.ijacsa.thesai.org

[5] A. Hannousse and S. Yahiouche, "Towards Benchmark Datasets for Machine Learning Based Website Phishing Detection: An experimental study," Oct. 2020, doi: 10.1016/j.engappai.2021.104347.

[6] A. D. Kulkarni, L. L. Brown, and A. Kulkarni, "Phishing Websites Detection using Machine Learning," 2019. [Online]. Available: http://hdl.handle.net/10950/1862www.ijacsa.thesai.org

[7] M. W. Shaukat, R. Amin, M. M. A. Muslam, A. H. Alshehri, and J. Xie, "A Hybrid Approach for Alluring Ads Phishing Attack Detection Using Machine Learning," *Sensors*, vol. 23, no. 19, Oct. 2023, doi: 10.3390/s23198070.

[8] E. Buber, B. Diri, and O. K. Sahingoz, "NLP Based Phishing Attack Detection from URLs," 2018, pp. 608–618. doi: 10.1007/978-3-319-76348-4_59.

[9] V. Shahrivari, M. M. Darabi, and M. Izadi, "Phishing Detection Using Machine Learning Techniques," Sep. 2020, [Online]. Available: http://arxiv.org/abs/2009.11116

[10] A. K. Dutta, "Detecting phishing websites using machine learning technique," *PLoS One*, vol. 16, no. 10 October, Oct. 2021, doi: 10.1371/journal.pone.0258361.

[11] A. Mittal, H. Kommanapalli, R. Sivaraman, T. Chowdhury, T. Chowdhury, and D. W. Engels, "Phishing Detection Using Natural Language Processing and Machine Learning," 2022. [Online]. Available: https://scholar.smu.edu/datasciencereviewAvailableat:https://scholar.smu.edu/datasciencereview/vol6/iss2/14http://digitalrepository.smu.edu.

[12] J. Rashid, T. Mahmood, M. W. Nisar, and T. Nazir, "Phishing Detection Using Machine Learning Technique," in *Proceedings - 2020 1st International Conference of Smart Systems and Emerging Technologies, SMART-TECH 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020, pp. 43–46. doi: 10.1109/SMART-TECH49988.2020.00026.

[13] L. Tang and Q. H. Mahmoud, "A Deep Learning-Based Framework for Phishing Website Detection," *IEEE Access*, vol. 10, pp. 1509–1521, 2022, doi: 10.1109/ACCESS.2021.3137636.

[14] N. Nagy *et al.*, "Phishing URLs Detection Using Sequential and Parallel ML Techniques: Comparative

Analysis," *Sensors*, vol. 23, no. 7, Apr. 2023, doi: 10.3390/s23073467.

[15] G. Sonowal, "Detecting Phishing SMS Based on Multiple Correlation Algorithms," *SN Comput Sci*, vol. 1, no. 6, p. 361, Nov. 2020, doi: 10.1007/s42979-020-00377-8.

[16] D. M., S. Badkul, K. Gharat, A. Vidhate, and D. Bhosale, "Detection of Phishing Websites Using Ensemble Machine Learning Approach," *ITM Web of Conferences*, vol. 40, p. 03012, Aug. 2021, doi: 10.1051/itmconf/20214003012.

[17] A. Tharwat, T. Gaber, A. Ibrahim, and A. Ella Hassanien, "Linear Discriminant Analysis: A Detailed Tutorial." [Online]. Available: http://www.egyptscience.net

[18] G. Biau and G. B. Fr, "Analysis of a Random Forests Model," 2012.

[19] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Association for Computing Machinery, Aug. 2016, pp. 785–794. doi: 10.1145/2939672.2939785.

[20] G. Ke *et al.*, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree." [Online]. Available: https://github.com/Microsoft/LightGBM.