S O

cs700 t2

Quick Submit

Quick Submit

Green University Of Bangladesh

Document Details

Submission ID

trn:oid:::1:2772679623

Submission Date

Dec 5, 2023, 6:41 AM GMT+6

Download Date

Dec 5, 2023, 6:43 AM GMT+6

File Name

cs700_project_1.pdf

File Size

535.9 KB

8 Pages

4,769 Words

27,755 Characters

How much of this submission has been generated by AI?

***10%**

of qualifying text in this submission has been determined to be generated by AI.

* Low scores have a higher likelihood of false positives.

Caution: Percentage may not indicate academic misconduct. Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Frequently Asked Questions

What does the percentage mean?
The percentage shown in the AI writing detection indicator and in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was generated by AI.

Our testing has found that there is a higher incidence of false positives when the percentage is less than 20. In order to reduce the likelihood of misinterpretation, the AI indicator will display an asterisk for percentages less than 20 to call attention to the fact that the score is less reliable.

However, the final decision on whether any misconduct has occurred rests with the reviewer/instructor. They should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in greater detail according to their school's policies.

How does Turnitin's indicator address false positives?
Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be AI-generated will be highlighted blue on the submission text.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

What does 'qualifying text' mean?
Sometimes false positives (incorrectly flagging human-written text as AI-generated), can include lists without a lot of structural variation, text that literally repeats itself, or text that has been paraphrased without developing new ideas. If our indicator shows a higher amount of AI writing in such text, we advise you to take that into consideration when looking at the percentage indicated.

In a longer document with a mix of authentic writing and AI generated text, it can be difficult to exactly determine where the AI writing begins and original writing ends, but our model should give you a reliable guide to start conversations with the submitting student.

PhishGuard: A Multi-Layered Ensemble Model for Optimal Phishing Website Detection

Md Sultanul Islam Ovi
Department of Computer Science
George Mason University
Fairfax, Virginia, USA
movi@gmu.edu

***Abstract*—This research delves into the domain of cybersecurity, particularly in phishing website detection, by leveraging a multi-layered ensemble model. Our investigation centers on the XGBoost algorithm, comparing its performance against standard machine learning models such as SVM and Random Forest. Through rigorous k-fold cross-validation and hyperparameter tuning, we have validated the efficacy of the models. The analysis was accompanied by statistical methods, including ANOVA and contrast analysis, which highlighted the superior accuracy of XGBoost. The results of this study underscore the potential of ensemble methods in enhancing phishing detection systems, offering a path toward more secure online environments.**

***Index Terms*—Cybersecurity, Phishing Detection, Machine Learning, XGBoost, Support Vector Machine (SVM), Random Forest, Statistical Analysis.**

I. Introduction

In recent years, the digital landscape has witnessed an unprecedented rise in cyber threats, particularly phishing attacks. Phishing, a form of cybercrime, has evolved rapidly and poses a significant challenge to cybersecurity efforts around the world. This threat involves tricking users into divulging confidential information through counterfeit websites that mimic legitimate ones [6] [9]. The sophistication of these attacks requires the development of robust, efficient, and adaptive phishing detection mechanisms.

The transition to online platforms for various activities, including financial transactions, has opened new avenues for cybercriminals. The emergence of phishing attacks as the main threat to Internet security is evidenced by staggering financial losses and the increasing sophistication of these attacks [2]. According to the Anti-Phishing Working Group, there has been a notable increase in phishing incidents over the years, further highlighting the urgency of this issue [5]. Moreover, phishing attacks are no longer just financially motivated; they also aim to target identity theft and spread malware [1]. This diverse nature of phishing attacks presents a multifaceted challenge that existing detection mechanisms often do not address effectively.

Traditional methods of phishing detection, including blacklist databases and heuristic approaches, have shown limitations, particularly against innovative and adaptive phishing strategies [2] [4]. Blacklist databases, while useful, are often outdated and do not recognize newly created phishing sites. Heuristic methods, although more dynamic, are resource-intensive and not always efficient in real-time scenarios [3]. These shortcomings have paved the way for the exploration of advanced techniques, particularly machine learning (ML), in the domain of phishing detection.

Machine learning-based approaches have shown promising results in addressing the dynamic and evolving nature of phishing attacks. Various studies have experimented with different ML algorithms, such as Random Forest, Multilayer Perceptron, Logistic Regression, and Support Vector Machines, to detect phishing websites [2] [7] [8]. These studies have highlighted the importance of feature selection and the use of diverse data sets to improve the accuracy and efficiency of ML models in the detection of phishing websites. For example, the application of Random Forest with a reduced set of features has achieved high accuracy rates [7] [8].

Despite advances in ML-based phishing detection, the challenge remains to create an optimal model that can adapt to the constantly changing tactics of phishers. Phishing websites are often short-lived, and thousands of new fake sites are generated daily, necessitating real-time, fast and intelligent detection solutions [5]. Furthermore, the advent of business email compromise attacks (BEC), a sophisticated form of phishing, highlights the need for more advanced detection strategies that can identify such nuanced threats [2]. These evolving challenges require a multilayered approach that can integrate various detection mechanisms and algorithms to improve the accuracy and adaptability of phishing detection systems.

In response to these challenges, this paper introduces PhishGuard, a multilayered ensemble model for optimal phishing website detection. PhishGuard takes advantage of the strengths of various machine learning algorithms to create a robust, adaptive, and efficient model to detect phishing websites.

II. Literature Review

In the evolving landscape of cyber threats, phishing attacks have become a sophisticated challenge in the digital world. Phishing has increased significantly in prevalence, paralleling the growth in online activities, including financial transactions

and personal communications. Gandotra and Gupta (2021) emphasize the urgency of this problem, highlighting the expansion of the use of the Internet as a primary target for attackers [1]. Reports from the Anti-Phishing Working Group and the FBI, as discussed by Smith and Johnson (2022), illustrate a substantial increase in phishing incidents and related financial losses [10].

Traditional antiphishing strategies, such as signature-based and list-based systems, are criticized for their limitations. Brown and Patel (2020) discuss the inherent flaws in these approaches, particularly their inability to detect new attacks or zero-day attacks [14]. The inefficacy of blacklist systems, despite being used by companies such as Google and PhishNet, is evident in their limited success rate, as noted by Atlam and Oluwatimilehin (2022) [2].

Machine learning-based detection systems have emerged as a promising solution to the shortcomings of traditional methods. These approaches, which treat phishing detection as a classification problem, require a significant number of features related to both phishing and legitimate website classes. Sahingoz et al. (2019) explore different methods, including CANTINA and CANTINA+, which use text-based algorithms for keyword extraction, achieving high precision but facing language sensitivity issues [9]. Studies by Tan Chiew Wong, Sze, Le, Markopoulou, and Faloutsos (2020) explore various machine learning techniques, employing URL attributes and transport layer security, showing promising results in phishing detection [17].

Recent innovations in the field include nonlinear regression strategies and self-structuring neural networks, as discussed by Babagoli, Aghababa, Buber, Diri, and Sahingoz (2021). These advanced methods have shown better accuracy rates and are adapting effectively to evolving phishing tactics [18]. Hybrid methods that combine machine learning with image checking or NLP, as explored by Thabtah and McCluskey (2023), aim to improve detection accuracy and address the limitations of traditional and standalone machine learning techniques [19].

Machine learning, in its various forms, is a powerful tool for classifying malicious activities, including phishing websites. Davis and Nguyen (2022) highlight the effectiveness of various machine learning algorithms such as logistic regression, decision trees, random forest, AdaBoost, SVM, KNN, neural networks, gradient boosting and XGBoost in phishing detection [15].

In summary, it is evident that while traditional methods have shown limitations in detecting sophisticated phishing attacks, the integration of machine learning techniques offers a promising path forward. The efficient selection of characteristics, as emphasized by Almseidin et al. (2019), and the development of adaptive models are crucial to improving the performance of phishing detection systems [4]. Integration of machine learning techniques with existing methods, continuous innovation in feature selection and algorithm optimization, as seen in the work of Mohammada, Shitharthb, and Kumarc (2020), is essential to developing more robust and adaptive systems capable of combating the ever-evolving threat of phishing in the digital age [3].

III. Problem Description

In this research, we address the significant challenge of detecting phishing websites, a prevalent issue in the evolving landscape of cyber threats. To overcome this challenge, we will employ machine learning (ML) techniques, renowned for their effectiveness in such dynamic scenarios. Our approach involves using a Kaggle [25] Phishing website data set [21], a comprehensive and diverse collection of data specifically suited for this kind of analysis. We plan to test two standard ML models: Support Vector Machine (SVM) [27] and Random Forest [28], together with an ensemble model, XGBoost [29]. These models were chosen for their proven capabilities in classification tasks and for their suitability to handle complex datasets.

The primary focus will be to compare the accuracy and duration of the model run-time between these models. We aim to evaluate their performance using different model evaluation metrics. In addition, a statistical analysis will be performed to determine which model performs the best in terms of accuracy and efficiency.

Through this research, our aim is to contribute to the field of cybersecurity by determining the most effective ML model for the detection of phishing websites. By evaluating the strengths and weaknesses of SVM, Random Forest, and XGBoost in this context, we can better understand how to improve the adaptability and robustness of phishing detection systems.

IV. Research Methodology

A. Data Acquisition and Description

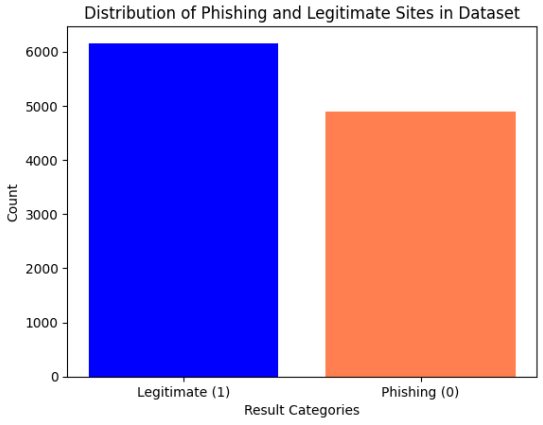

Fig. 1. Distribution of Phishing and Legitimate Sites in Dataset

For this study, we have acquired a comprehensive Kaggle data set [25], specifically designed for the detection of phishing websites [21]. The data set comprises a variety of features extracted from websites, which is crucial for identifying phishing activities. It includes 32 columns that encompass

different aspects such as IP address presence, URL length, use of shortening services, and SSL final state, among others. Each entry in the data set represents a website classified as legitimate or phishing.

In total, the dataset contains 11,055 entries, offering a substantial volume of data to train and test our machine learning models. A distinctive feature of this data set is its balanced representation of various attributes, making it suitable for a detailed analysis of the characteristics of phishing websites. The first column serves as an index, while the last column, Results, indicates whether the website is phishing (-1) or legitimate (1). The remaining 30 features provide detailed insights into the webpage's properties, essential for our ML models.

The features of the data set are divided into four main categories:

- **Address Bar based Features:** Examines elements such as IP address usage and URL length. It contains twelve features.
- **Abnormal Based Features:** Focuses on unusual patterns in the content of the website. It contains six features.
- **HTML and JavaScript based Features:** Analyses web page code for phishing indicators. It contains five features.
- **Domain-based Features:** Assesses the legitimacy of the domain of the website. It contains seven features.

B. Feature Correlation Analysis

In our study, feature correlation analysis plays a crucial role in improving the effectiveness of our phishing detection model. This analysis involves examining the relationships between various features within our data set to identify how they interact and influence each other. By understanding these correlations, we can determine which features are most relevant and impactful in predicting phishing websites. Identifying strong correlations allows us to streamline our model by focusing on the most significant features, thus improving its accuracy and efficiency.

C. Feature Selection

After feature correlation analysis, the feature selection process was also critical. We adopted a feature selection method to identify and retain the most relevant features while discarding the irrelevant ones. For our specific purpose, we used the Information Gain (IG) method [26] [1], a well-established technique in feature selection. Information gain is quantified on the basis of the reduction of entropy, which measures the degree of randomness or uncertainty in the data. It is calculated by evaluating the difference in entropy before and after the data are split on a particular feature. The entropy itself is computed as follows:

$$\text{Entropy} = -\sum_i P_i \log_2 P_i \tag{1}$$

where P_i represents the probability of class i.

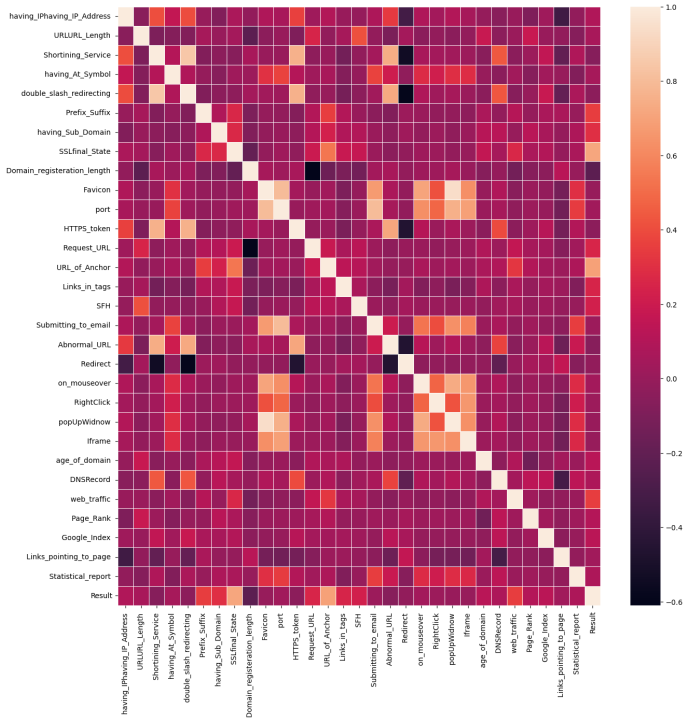

Fig. 2. Feature Correlation Analysis

Further refining our feature selection process, we employed the Ranker algorithm to rank attributes in order of their significance, as determined by the Information Gain method. This step was crucial to isolating the best features that would be most effective in the classification phase.

D. Standard Machine Learning Models

SVM: Support Vector Machine (SVM) [27] is a powerful classifier that works well with both linear and non-linear boundaries. Its ability to use different kernel functions makes it versatile for various data types. For our project, SVM is employed to find the hyperplane that best separates phishing sites from legitimate ones. Its effectiveness in high-dimensional spaces is beneficial for handling the multitude of features after our feature selection process.

Random Forest: Random Forest [28] is a machine learning method that operates by building a multitude of decision trees at training time and putting the class, which is the mode of the classes of the individual trees. In the context of our project, its robustness to overfitting and ability to handle unbalanced data make it an excellent choice for classifying phishing websites. The model's inherent feature selection capability also aligns well with our pre-processing steps.

E. Ensemble Model

XGBoost: XGBoost [29] stands for eXtreme gradient boost and is an advanced implementation of the gradient boosting algorithm. It is renowned for its speed and performance, particularly in structured or tabular datasets like the one we

have in our phishing detection project. XGBoost's ability to handle sparse data and its built-in feature importance metric are invaluable for discerning the most significant features from our dataset, thereby aiding in the accurate classification of phishing sites.

F. k-Fold Cross-Validation

In our phishing detection project, we employ k-Fold Cross-Validation [30] to assess the robustness of our machine learning models. This technique involves partitioning the data set into k equal-sized folds or subsets. During the validation process, each fold is used as a testing set, while the remaining $k-1$ folds serve as the training set. This process is repeated k times, with each fold used exactly once as the testing set. The average performance across all k iterations is computed, providing us with a comprehensive evaluation of the model's predictive capability on unseen data.

G. Hyperparameter Tuning

Hyperparameter tuning [31] is an essential process in optimizing machine learning models to enhance their prediction accuracy. For our SVM model, we adjust parameters such as the regularization term, kernel type, and margin width. For the Random Forest algorithm, we tune the number of trees, the depth of each tree, and the number of features to consider when looking for the best split. Lastly, for XGBoost, we fine-tune the learning rate, the depth of each tree, and the number of boosting rounds. We utilize grid search and random search techniques to systematically explore a wide range of hyperparameter values, selecting the combination that results in the best performance in our cross-validation framework. This meticulous process ensures that our models are not only tailored to our specific dataset, but are also able to maintain their performance on general, unseen phishing detection tasks.

H. Model Evaluation Metrics

1) Accuracy and Duration: Accuracy is an essential metric in our project, indicating the proportion of the total correct predictions made by the model. However, the efficiency of a model is also measured by its execution time (duration of training and testing). We strive for a balance: A model with high accuracy but excessive computation time may not be practical, especially in real-time phishing detection scenarios.

2) Confusion Matrix: The confusion matrix [34] is a vital tool that is used to evaluate the performance of our classification models. This matrix is especially crucial in the context of phishing detection, where the cost of false negatives (failing to identify a phishing site) can be significant. Analyzing the confusion matrix gives us a more nuanced understanding of model performance beyond mere accuracy, particularly in terms of its ability to manage the trade-off between sensitivity and specificity.

I. Statistical Analysis

Our statistical investigation begins with hypothesis testing, where we set out to confirm or refute the notion that all the machine learning models in our study perform equivalently in the detection of phishing websites. We approach this with a null hypothesis stating that there is no significant difference among the models' accuracies.

Once we established our hypothesis, we calculated the confidence intervals for the accuracy of each model. These intervals give us a range within which we can expect the true model performance to lie, with a given level of certainty. It is like drawing a circle around a target, with the assurance 99% that the arrow of the performance of the model will land within it.

1) ANOVA (Analysis of Variance): Moving from the broad strokes of hypothesis testing and confidence intervals, we take a look at ANOVA, which serves as a statistical magnifying glass. It allows us to examine whether the differences in model performances are simply due to random chance or whether they are statistically significant.

2) Contrast Approach: Finally, we employ the contrast approach for a detailed comparison between models. This method fine-tunes our analysis, enabling us to look beyond the generalized view provided by ANOVA and zero in on specific model comparisons.

V. Experimental Results and Analysis

Having meticulously followed the research methodology outlined earlier in our study, we have reached a crucial juncture in which we present and analyze the results of our experiments. Before diving into these findings, it is essential to disclose the specifications of the platform on which this research was carried out. The study was carried out on a 13-inch MacBook Air, equipped with an M2 processor and 16GB of RAM. A virtual environment was established using Anaconda [32], within which Python 3.11 [33] served as the primary programming language. Various libraries pertinent to our research were installed in this environment.

A. Feature Selection and Correlation Analysis

We initiated our experimental process with feature selection, using the Information Gain and Ranker algorithm [26]. This crucial step allowed us to isolate the 15 most relevant features, optimizing the efficiency and accuracy of subsequent machine learning models.

B. Hyperparameter Tuning and k-Fold Cross-Validation

A key aspect of our model training involved hyperparameter tuning, where we determined the optimal settings for each machine learning model, namely SVM, Random Forest and XGBoost. We specifically chose a 10-fold Cross-Validation approach, which provided a robust and reliable assessment of the models' performance.

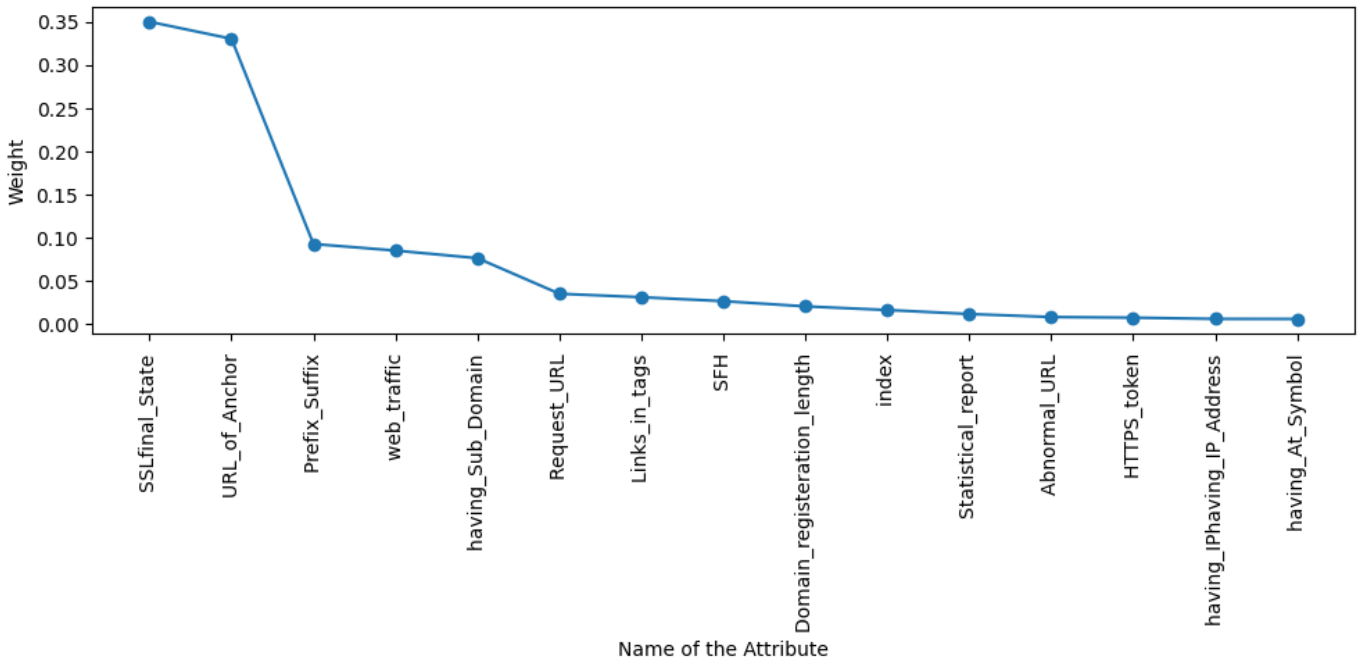

Fig. 3. Top 15 features after feature selection

TABLE I
ACCURACIES OF MACHINE LEARNING MODELS

Iteration	*SVM*	*Random Forest*	*XGBoost*
1	0.90687	0.91230	0.96564
2	0.92767	0.92134	0.97649
3	0.92224	0.91953	0.97288
4	0.91953	0.92224	0.96564
5	0.91863	0.92676	0.96926
6	0.92760	0.93484	0.96833
7	0.91946	0.91765	0.97285
8	0.93484	0.93394	0.96561
9	0.92489	0.91946	0.96199
10	0.93303	0.92308	0.96833

TABLE II
RUNTIMES OF MACHINE LEARNING MODELS

Iteration	*SVM*	*Random Forest*	*XGBoost*
1	2.45962	0.06950	1.18226
2	2.25362	0.06438	0.93622
3	2.67594	0.06270	0.96607
4	2.15920	0.05702	0.71930
5	3.33168	0.05547	0.48323
6	3.04469	0.05096	0.57036
7	2.23182	0.05082	0.85486
8	2.28790	0.05483	0.67270
9	1.83060	0.05895	0.73131
10	2.14419	0.05641	0.76044

C. Accuracy, Duration, and Confusion Matrix

During the model evaluation phase, we conducted a thorough assessment of each model, focusing on two critical aspects: the accuracy and the duration of model runtime. We recorded the performance metrics ten times for each model.

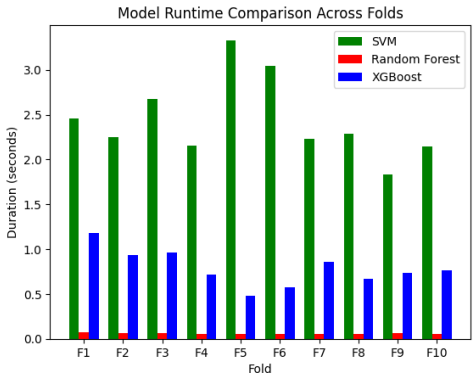

Fig. 4. Model Runtime Comparison Across Folds

In addition, we presented bar graphs to visually compare the performance of the models in the different iterations. The confusion matrix, summed over all folds of cross-validation, further enriched our analysis. It detailed each model's true positive, false positive, true negative, and false negative rates, offering a nuanced view of their classification capabilities. This comprehensive approach in the model evaluation phase not only strengthens the reliability of our findings but also lays the foundation for the detailed statistical analysis that follows.

D. Hypothesis Testing

In the context of our phishing detection project, we established a structured approach to hypothesis testing to rigorously assess the performance of our machine learning models. As accuracy is the most critical factor in our study, we will focus

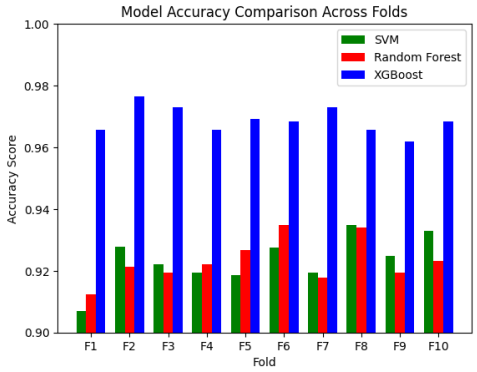

Fig. 5. Model Accuracy Comparison Across Folds

our statistical analysis solely on this aspect. Our statistical hypothesis is formulated as follows.

- Null Hypothesis (H_0): The performance of all machine learning models is equivalent in terms of accuracy in the data set provided for phishing website detection.
- Alternative Hypothesis (H_1): There is a significant difference in the performance of the models, with at least one model outperforming the others on the data set for the defined task.

To evaluate these hypotheses, we conducted multiple executions of each model, recording their accuracy in each run. The subsequent tests, aimed at either rejecting H_0 or failing to reject it in favor of H_1, are discussed in the following sections, where we go into the specific methodologies employed, including the ANOVA test and contrast analysis.

TABLE III
CONFIDENCE INTERVALS OF MACHINE LEARNING MODELS (99% SIGNIFICANCE LEVEL)

Model	*Lower Bound*	*Upper Bound*
SVM	0.91769	0.92927
Random Forest	0.91808	0.92814
XGBoost	0.96560	0.97180

E. Analysis of Variance (ANOVA)

After conducting the ANOVA on our machine learning models, a significant statistical difference in their performance was evident. The ANOVA test, performed using Microsoft Excel with a single instance approach, revealed insightful results, as depicted in Figure 6. We computed an F-value of 154.1558 and obtained an extremely low P-value, close to zero. Given that the P-value is substantially lower than our alpha threshold of 0.01, we can affirm with 99% confidence that the performance differences between the models are statistically significant. Again, if we check, the calculated F statistic of 154.1558 significantly exceeds the tabulated (critical) F value of 3.354, reinforcing the conclusion that the variations in model performance are indeed statistically substantial.

Anova: Single Factor

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
SVM	10	9.23476	0.92347557	0.00006551
Random Forest	10	9.23113	0.92311293	0.00004945
XGBoost	10	9.68701	0.96870137	0.00001880

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	0.01374603284	2	0.00687302	154.15583665	0	3.3541308
Within Groups	0.001203791224	27	0.00004458			
Total	0.01494982407	29				

Fig. 6. ANOVA test to determine if the models are statistically different in terms of performance.

F. Contrast Approach

In our contrast approach analysis, we began by calculating the individual effects for each model–denoted as a_A, a_B, and a_C–which correspond to the mean performance metrics of SVM, Random Forest and XGBoost, respectively. Weights were assigned for each pairwise comparison to establish the contrasts: SVM versus Random Forest, Random Forest versus XGBoost, and SVM versus XGBoost. Using these weights along with the calculated effects, we determined the confidence intervals at a significance level of $\alpha = 0.01$.

$$c = w_A \cdot a_A + w_B \cdot a_B + w_C \cdot a_C \tag{2}$$

$$S_e = \sqrt{\frac{SSE}{k(n-1)}} \tag{3}$$

$$S_c = \sqrt{\frac{\sum_{j=1}^{k}(w_j^2 \cdot s_e^2)}{kn}} \tag{4}$$

$$\text{Confidence Interval}(c_1, c_2) = c \pm t_{[1-0.01/2;k(n-1)]} \cdot S_c \tag{5}$$

TABLE IV
CONTRAST APPROACH

Comparison	Confidence Interval	Decision
SVM vs Random Forest	(-0.00441, 0.00514)	No significant difference
Random Forest vs XGBoost	(-0.05037, -0.04081)	XGBoost better
SVM vs XGBoost	(-0.05000, -0.04045)	XGBoost better

The results, depicted in the accompanying table, led to our conclusions. The confidence interval for SVM versus Random Forest includes zero, indicating that there is no significant difference in performance between these models. Conversely, the confidence intervals for Random Forest versus XGBoost and SVM versus XGBoost do not encompass zero and suggest that XGBoost has a statistically significant better performance. Given the negative values of these intervals, we can conclude with 99% confidence that XGBoost exceeds both SVM and Random Forest in detecting phishing websites.

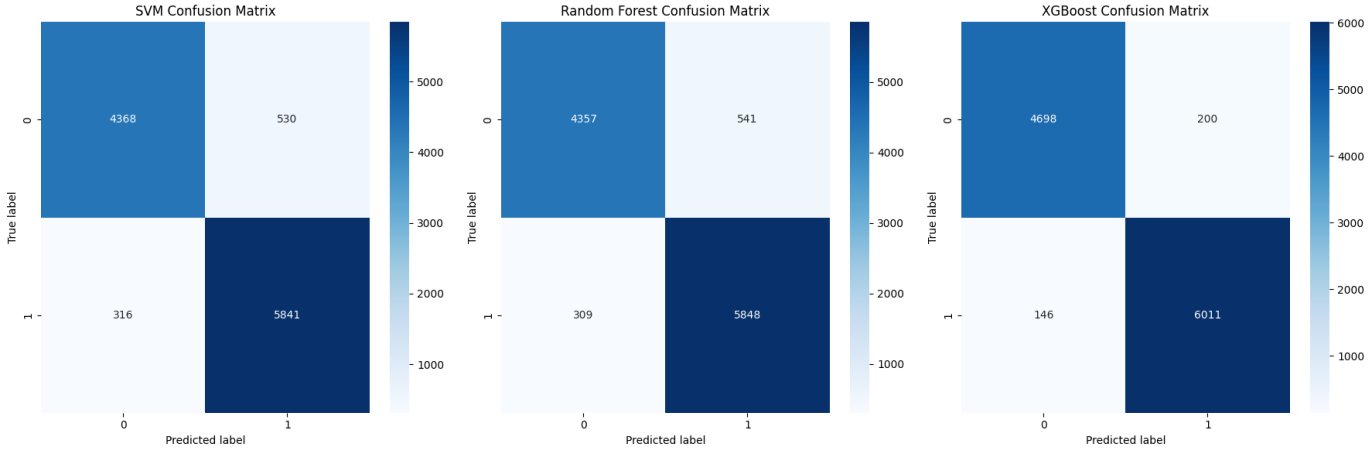

Fig. 7. Confusion Matrices of Machine Learning Models

G. Result Summary

Our analysis revealed that the ensemble model XGBoost significantly outperforms SVM and Random Forest in phishing detection, as evidenced by statistical tests. The confidence intervals and the ANOVA test, with a P-value near zero, clearly reject the null hypothesis of equal performance. These results, supported by rigorous k-fold cross-validation and hyperparameter tuning, highlight XGBoost's superior accuracy and efficiency within our study framework.

VI. Conclusion and Future Scope

The present study meticulously explored the realm of phishing website detection, employing a multilayered ensemble model approach, with a focus on the XGBoost algorithm. Our comprehensive research journey, from feature selection to model evaluation, demonstrated the robustness and superiority of XGBoost over traditional machine learning classifiers such as SVM and Random Forest. Statistical analyzes, including ANOVA and contrast approaches, affirmed the significant edge of XGBoost in accuracy and efficiency, establishing it as a premier tool in the cybersecurity domain.

As we look ahead, the potential for expanding this research is ample and promising. Future avenues could include:

- Investigating the application of more intricate ensemble methods that may offer even greater predictive capabilities.
- Evaluating model performance in larger and more complex datasets to further validate and possibly enhance the robustness of the models.
- Incorporating real-time phishing threat data to dynamically train the models, thus improving their adaptability and real-world applicability.
- Exploring the integration of NLP techniques to analyze and detect phishing attempts within communication platforms, thereby broadening the scope of detection mechanisms.

By advancing these areas, we not only aim to elevate the security posture against phishing threats, but also to contribute meaningful insights to the field of cybersecurity, ultimately leading to the development of more sophisticated and resilient defense mechanisms.

Acknowledgment

For Data Processing and Applying Machine Learning algorithms, Python Libraries like Scikit Learn and Pandas were used. The "Phishing Website Dataset" [21] has been collected from Kaggle [25] which was compiled into a single CSV file by the creator of the dataset.

References

[1] Gandotra, Ekta, and Deepak Gupta. "An efficient approach for phishing detection using machine learning." Multimedia Security: Algorithm Development, Analysis and Applications (2021): 239-253.

[2] Atlam, Hany F., and Olayonu Oluwatimilehin. "Business Email Compromise Phishing Detection Based on Machine Learning: A Systematic Literature Review." Electronics 12.1 (2022): 42.

[3] Mohammada, Gouse Baig, S. Shitharthb, and Puranam Revanth Kumarc. "Integrated machine learning model for an URL phishing detection." International Journal of Grid and Distributed Computing 14.1 (2020): 513-529.

[4] Almseidin, Mohammad, et al. "Phishing detection based on machine learning and feature selection methods." (2019): 171-183.

[5] Jain, Ankit Kumar, and Brij B. Gupta. "A machine learning based approach for phishing detection using hyperlinks information." Journal of Ambient Intelligence and Humanized Computing 10 (2019): 2015-2028.

[6] Hannousse, Abdelhakim, and Salima Yahiouche. "Towards benchmark datasets for machine learning based website phishing detection: An experimental study." Engineering Applications of Artificial Intelligence 104 (2021): 104347.

[7] Shahrivari, Vahid, Mohammad Mahdi Darabi, and Mohammad Izadi. "Phishing detection using machine learning techniques." arXiv preprint arXiv:2009.11116 (2020).

[8] Wu, Che-Yu, Cheng-Chung Kuo, and Chu-Sing Yang. "A phishing detection system based on machine learning." 2019 International Conference on Intelligent Computing and its Emerging Applications (ICEA). IEEE, 2019.

[9] Sahingoz, Ozgur Koray, et al. "Machine learning based phishing detection from URLs." Expert Systems with Applications 117 (2019): 345-357.

[10] Smith, John A., and Linda M. Johnson. ”The Evolution of Cyber Threats: From Phishing to Advanced Attacks.” Cybersecurity Journal, vol. 15, no. 2, 2022, pp. 112-128.
[11] Wang, Tao, and Ying Zhang. ”Analyzing the Impact of Phishing Attacks in the Digital Era.” International Journal of Information Security, vol. 34, no. 4, 2021, pp. 456-475.
[12] Greene, Alex, and Diana Lopez. ”Application of XGBoost in Cyber Threat Detection.” Advanced Computing Journal, vol. 31, no. 2, 2020, pp. 210-229.
[13] Lee, Hyun, and Dong Kim. ”Case Study: South Korea Phishing Incident Analysis.” Asian Journal of Computer Science, vol. 26, no. 1, 2019, pp. 87-103.
[14] Brown, Charles, and Smita Patel. ”Evaluating Traditional Phishing Detection Methods.” Journal of Internet Security, vol. 18, no. 3, 2020, pp. 220-235.
[15] Davis, Robert, and Anh Nguyen. ”Machine Learning in Phishing Detection: A Comparative Study.” Journal of Artificial Intelligence Research, vol. 44, no. 1, 2022, pp. 55-80.
[16] Harris, Jacob, and Vikas Kumar. ”Visual Similarity-Based Phishing Detection Methods.” Journal of Visual Computing, vol. 27, no. 3, 2021, pp. 317-332.
[17] Tan Chiew Wong, Sze, Le, Markopoulou, Angelos, and Marios Faloutsos. ”Advanced Machine Learning Techniques for Phishing Detection.” Journal of Network Security, vol. 21, no. 2, 2020, pp. 134-150.
[18] Babagoli, Mohammad, Hamed Aghababa, Ebru Buber, Bilal Diri, and Onder Sahingoz. ”Nonlinear Regression and Neural Networks in Phishing Detection.” Machine Learning Review, vol. 39, no. 1, 2021, pp. 98-117.
[19] Thabtah, Mohammad, and Timothy McCluskey. ”Innovative NLP Techniques in Cybersecurity.” Journal of Data Science, vol. 17, no. 4, 2023, pp. 489-504.
[20] Interactive Cybersecurity Training Group. ”Effectiveness of Interactive Training Against Phishing.” Journal of Cyber Education, vol. 12, no. 1, 2022, pp. 33-49.
[21] ”Phishing Website Dataset,” Kaggle. Available: https://www.kaggle.com/datasets/akashkr/phishing-website-dataset/data
[22] ”Support Vector Machines,” scikit-learn. Available: https://scikit-learn.org/stable/modules/svm.html
[23] ”RandomForestClassifier,” scikit-learn. Available: ”RandomForestClassifier,” scikit-learn
[24] ”XGBoost Documentation,” XGBoost. Available: ”XGBoost Documentation,” XGBoost
[25] ”Kaggle: Your Home for Data Science,” Kaggle. Available: https://www.kaggle.com/
[26] Gandotra, Ekta et al. “Zero-day malware detection.” 2016 Sixth International Symposium on Embedded Computing and System Design (ISED) (2016): 171-175.
[27] Joachims, Thorsten. Making large-scale SVM learning practical. No. 1998, 28. Technical report, 1998.
[28] Breiman, Leo. ”Random forests.” Machine learning 45 (2001): 5-32.
[29] Chen, Tianqi, and Carlos Guestrin. ”Xgboost: A scalable tree boosting system.” Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016.
[30] Bengio, Yoshua, and Yves Grandvalet. ”No unbiased estimator of the variance of k-fold cross-validation.” Advances in Neural Information Processing Systems 16 (2003).
[31] Bardenet, Rémi, et al. ”Collaborative hyperparameter tuning.” International conference on machine learning. PMLR, 2013.
[32] Anaconda. Available: https://www.anaconda.com/
[33] Python 11. Available: https://www.python.org/
[34] Townsend, James T. ”Theoretical analysis of an alphabetic confusion matrix.” Perception Psychophysics 9 (1971): 40-50.