JCBI
Journal of Computing & Biomedical Informatics Volume 09 Issue 01
ISSN: 2710 - 1606 2025

Research Article
https://doi.org/10.56979/901/2025

Phishing Website URL Detection Using a Hybrid Machine Learning Approach

Muhammad Usman Javeed[1], Shafqat Maria Aslam[2], Hafiza Ayesha Sadiqa[1], Ali Raza[1*], Muhammad Munawar Iqbal[3], and Misbah Akram[4]

[1]Department of Computer Science, COMSATS University of Islamabad, Sahiwal, 5700, Pakistan.
[2]School of Computer Science, Shaanxi Normal University, Xi'an, Shaanxi, 710062, China.
[3]Department of Computer Science, University of Engineering and Technology, Taxila, 47050, Pakistan.
[4]Department of Software Engineering, Minhaj University Lahore, 54000, Pakistan.
[*]Corresponding Author: Ali Raza. Email: alirazamscs@gmail.com

Received: April 26, 2025 **Accepted:** May 30, 2025

Abstract: In a relatively short time, the internet has grown and progressed tremendously. With more users and advancements in web development, the internet today supports a large portion of the corporate world. With it, the number of cyber-attacks and threats has skyrocketed, resulting in monetary losses, data breaches, theft of identity, brand reputation damage, and a loss in customer trust in online shopping and banking. Phishing is a type of cyber threat in which a fake person usually hacker impersonates a genuine and trustworthy organization in order to get sensitive and private information from a victim. Furthermore, phishing has been a problem for many years. The global economy has now suffered billions of dollars as a result. In this study, we will examine some techniques for addressing the issue of phishing, particularly phishing using websites, and design solution based on machine learning algorithms to identify phishing websites. In order to understand the machine learning decision-making foundation and examine which attributes in general would be utilized to classify a website as real or phishing, we also conducted feature significance analysis using the provided dataset and solution. In this study we utilized Decision Tree, Random, Forest C-Support Vector Classification and AdaBoost algorithms for the detection of phishing URLs. Random Forest consistently outperformed than the other models across all key metrics. It demonstrated optimal performance in its classifications by achieving the highest accuracy 97.7%, Precision 99% and F1 score 97%.

Keywords: Phishing; URLs; Cybersecurity; Machine Learning

1. Introduction

Everyone nowadays is connected. Because of the enormous number of users or individuals who have access to the Internet, several sectors are adopting it to replace many old processes and technology [1]. As a result, we can now complete a wide range of tasks online, including digital marketing, banking, and shopping. Furthermore, the Internet has nearly fully replaced several tasks. The Internet is always changing and expanding. But concerns about safety have increased significantly as we get closer to a world where a lot of activities are done online. Many people are still unfamiliar with a variety of security risks and how to address them. Phishing is one of the most prevalent security challenges, and this study is dedicated to addressing the issue [2].

Phishing is a cybercrime including social engineering and other advanced strategies in which an attacker impersonates a genuine and trustworthy entity to obtain sensitive information from a victim [3]. By the use of fake email accounts and communications, Naive victims are tricked by social engineering techniques into thinking they are communicating with a reliable, authentic source [4]. Phishing is a widespread and growing danger [5]. In this study, we will look into online phishing, specifically phishing websites.

The phishers usually utilize malicious websites and emails to steal information that is sensitive and confidential. Some phishers utilize certain emails or webpages to deceive their victims target to download infected attachments that would be used to track their online activities. Devices or data are stolen, often utilizing systems that intercept users' account usernames and passwords. Passwords are stolen or users are directed to fake websites, a tactic known as technical fraud. Some phishers rely solely on emails to persuade their victims to open harmful attachments. For those emails, the security provided by most email service providers is sufficient. They provide a warning and, in most cases, designate them as spam. However, sending emails is usually only one aspect of the phishing scheme. Aspiring phishers frequently develop a bogus website for a legitimate company [6]. These are phishing sites that are designed to steal sensitive data, including social security numbers, usernames, passwords, credit card numbers, and personal information. These websites have a great degree of visual resemblance to the original websites in terms of colors, themes, fonts, and images.

2. Related Work

As we can see, there are a variety of ways available for detecting phishing websites. We'll look at several Machine Learning-based approaches, as our study will be directly related to these works and studies, and these approaches have a lot of promise for addressing phishing issues. A. Shinde, A. Pandey, R. Pawar, and V. Gangule proposed a study in which they created a method based on K-means clustering techniques and naive bayes classifiers [7]. The data was extracted from 300 phishing websites out of 500 phishtank entries. After obtaining the features from the URL by using the K-Means approach the features were separated the data into clusters, one is named as less suspicious and the second is named as more suspicious. After extracting features from HTML's DOM elements, the websites in the middle were examined further. A naïve Bayes Classifier was then employed to ascertain whether or not the websites were phishing websites.

Based on traffic flow data, the authors of [8] first constructed an undirected graph with user and URL nodes. They then used an algorithm that is based on probabilistic graphical named as Markov Random Field, to repeatedly correct the reputation of nodes, which was then used as a threshold to identify web phishing. The linguistic properties of URLs, domain properties, and website content properties were all taken into consideration by the authors. Their technique incorporates two different kinds of detection algorithms, one that is based on the content of the website and the other on the URL.

This interaction between the user and the website was used by the authors. The data was collected from a major ISP. Unlike the more usual fields, their record included eight fields. Examples of these include the following: Access time of user SRC IP, Node number of user Visiting URL, IP of Access Server, Reference URL, User Agent, , and User Cookie. The client was given a changeable IP address from the ISP's own pool in addition to a unique User node number. The visiting relation graph was constructed using the visiting URL and the User Node number. The authors claim that the technique may identify potential phishing that is often undetectable through URL analysis, and they enhanced the detection rate by reducing the impact of frequently updated phishing websites, which led to producers purposefully avoiding detection. According to authors, they achieved 3% rate false positive and 92% rate of true positive on real-world traffic.

In order to identify phishing websites, W. Ali [9] suggested to utilize machine learning classifiers with wrapper feature selection. The dataset used in the study was downloaded from UCI Machine Learning Repository. In order to reduce computation time and noise, the suggested wrapper feature was created to choose a subset of significant characteristics from the dataset that would accurately reflect the website dataset. The best features were chosen based on the machine learning classifier's highest evaluation. In many supervised algorithms, the outcome of five-fold cross validation indicated an increase in Correct Classification Rate and claimed that it performed better than algorithms that used the Information Gain selection technique and the Principal Component Analysis (PCA) features selection methodology.

3. Methodology

This section outlines the research methodology used in the study.

3.1. Dataset

Data is the foundation of every machine learning model, and the more detailed, relevant, reliable and clean the data gives the exceptional results. Furthermore, there are other resources where we may access databases for both phishing and non-phishing websites, such PhishTank [10] and Alexa [11] however, obtaining appropriate attributes from these websites is a different research topic. As a result, we choose to use a standard dataset that is both credible and the subject of considerable research. The University of California, Irvine Machine Learning Repository provided the dataset for this study that is available for study purpose.

This is based on the content of several studies about phishing websites, and as of yet, there is no consensus in the literature regarding the key characteristics that define phishing websites. We encountered similar issues in this research. However, some of the most significant and widely accepted characteristics for identifying phishing websites were applied to several well-known properties in this dataset. Some additional characteristics and rules are included; these features are generally accepted in different studies and are also demonstrated by trends. In the training examples, the dataset from the UCI repository has 30 characteristics and 11055 instances.

3.2. Data Preprocessing

Data cleaning is performed during the preprocessing step of the ML process [12]. This cleaning process includes identifying and eliminating duplicates in the dataset, completing missing values or, eliminating occurrences, balancing inconsistencies, and fixing or eliminating any mistakes and anomalies in the dataset [13]. The dataset was determined to be quite clean and ready to be trained right out of the box when we looked at it. There were no features to determine whether or not the instances were duplicates. As a result, we assume that each instance in the training dataset is distinct. In addition, no null values were found in the data. Visualizing categorical data and categorical labels, as well as judging their relationship among the data, is extremely difficult. However, there are various strategies that can be used to find connection between different types of data. To begin, we can use a bar chart to represent the data. Fig 1 showing the association of each of the features in the dataset with every other feature as heat map.

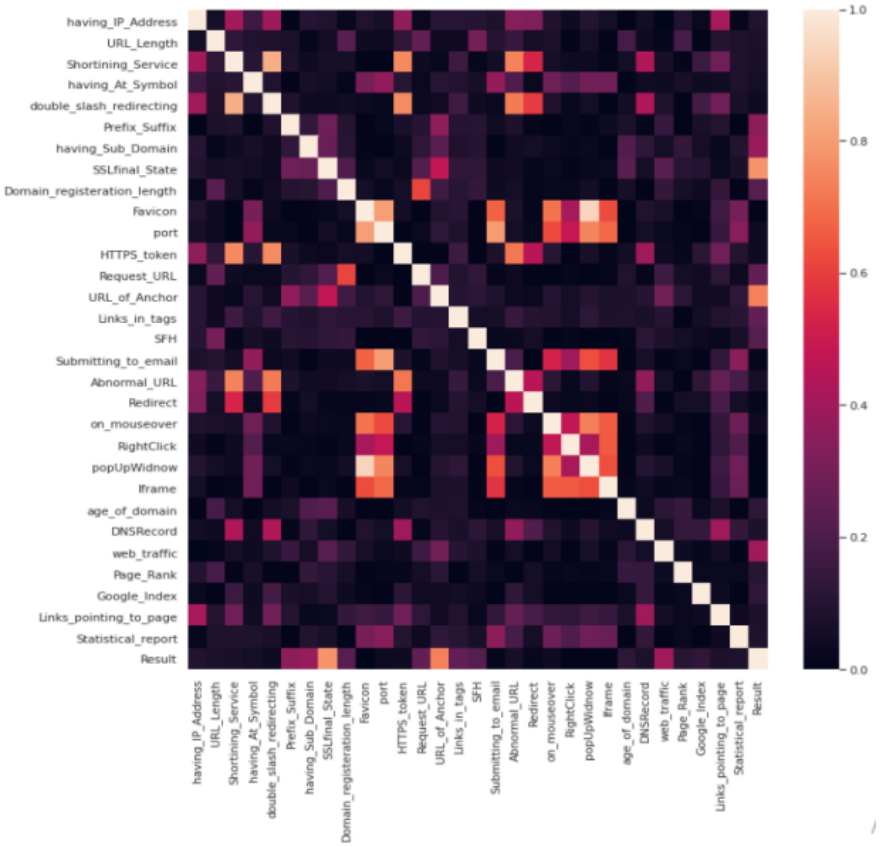

Figure 1. Heat-Map Showing the Association of Each of the Feature

3.3. Feature Selection

The act of producing new features from raw data in order to improve the learning algorithm's predictive power is known as feature engineering. The new engineered features should catch some fresh additional information that the original feature set cannot effectively express. The process of identifying a critical subset of characteristics in order to minimize the training problem's dimensionality. The procedure entails deleting features from the categorization that have very little to nearly no predictive potential. Despite the feature selection technique used, it is clear that it affects the accuracy of those traditional algorithms, even though it could enhance some of them by increasing computing performance. It learns flexible representations by leveraging raw data. In end-to-end learning, it also employs soft instance-wise feature selection with controlled sparsity.

3.4. Architecture

The Architecture of the model starts by collecting phishing URLs from a publicly available source known as PhishTank which provides updated phishing URLs data in many formats including CSV files. After that to split the URLs down into understandable parts, they are then processed using certain delimiters such dots (.), commas (,), equal signs (=), question marks (?), and slashes (/). To identify URL segments that include seemingly random or nonsensical characters a Random Word Recognizer is used, which are frequently found in phishing links. In order to help identify malicious URLs from safe ones, these parts are then compared to a predetermined library to see if they fit recognized patterns linked to phishing activity.

It is essential to gather both harmful and clean URLs in order to identify malicious ones. After that, all clean and malicious URLs are appropriately identified, and attribute extraction is carried out. Furthermore, this dataset is separated into two subsets: testing data for the testing procedure and training data for machine learning algorithm training. The machine learning model will be employed in the detection stage if its classification performance is excellent. Every input URL undergoes the detecting step. After extracting the features from URL than the classifier uses these extracted attributes to classify the URL is malicious or safe. Fig 2. Illustrate the architecture adopted in this study with each essential step.

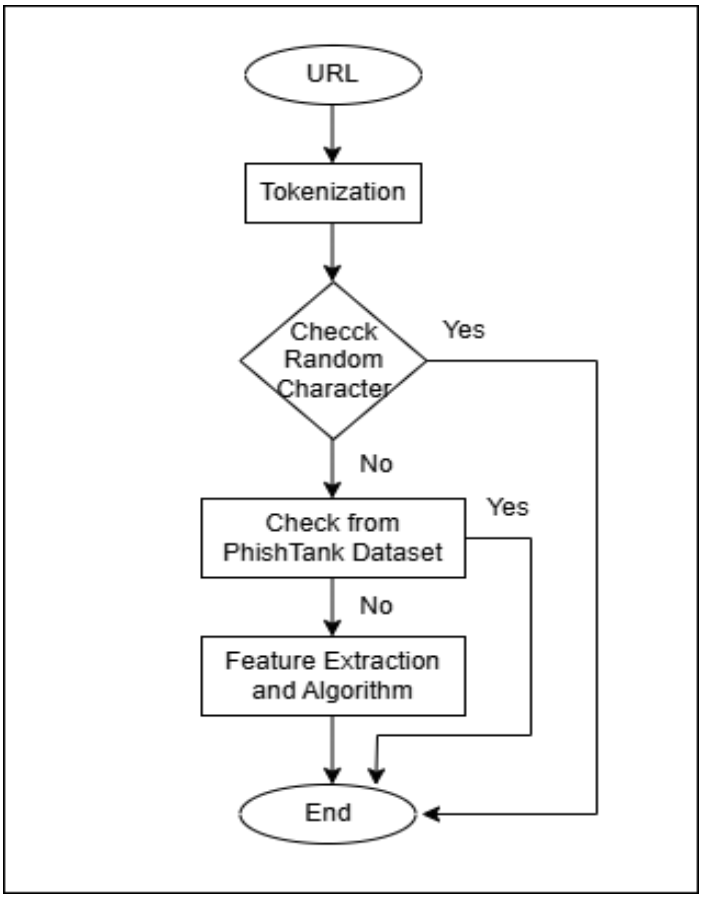

Figure 2. Architecture of Proposed Model

3.5. Experimental Setup

In this study, every step of data processing, training and assessment is performed on the Jupyter notebook. The main goal of the study was to build a model with a high level of prediction accuracy and best generalization performance. While most approaches, including CDNS (Conventional Deep Neural Networks), are criticized for overfitting the training data. These models perform well in the training data with a low error rate, but gives poor results in the unknown test data or real-world data. A technique known as 'hold-out' validation was utilized to solve this problem. In the machine learning process, this is one of the regular techniques. Training, validation, and testing datasets were created from the training set. The testing datasets account for 9.5% of the total dataset. In a 1:9 ratio, the validation dataset was partitioned from the training dataset. Stratified sampling approaches were employed for all of the splitting. This allows us to sample the data in such a way that the class variables are uniformly split over all sets of data, preserving the original class distribution. In this study we utilized Decision Tree, Random, Forest C-Support Vector Classification and AdaBoost algorithms for the detection of phishing URLs.

3.6. Hyper Parameter Tuning and Cross Validation

Depending on the many parameters used to train the model, a model can perform very well or very poorly in the same datasets. The model learns weights and other parameters in the Neural Network while training itself from the training data [14]. There are also users must provide the model with parameters such as learning rate and regularization parameters prior to attempting to fit the model to the data A model's performance can be affected by certain parameters. Changes can be made to the same sets of data. And those factors aren't the same for every piece of data. For different sets of parameters, different data is necessary. Furthermore, there are no hard and fast rules in data science. As a result, identifying the optimal solutions in machine learning is a time-consuming process. Fortunately, libraries such as sklearn offer modules that simplify most of this process. GridSearchCV is one of Sklearn's modules for this process. Hyper parameter tweaking is the term for this technique. It is an essential machine learning task that aids in maximizing the value of the data and model. We can evaluate the model's performance and train for each of the hyperparameters in the validation set. However, this introduces the issue of overfitting. In this situation, the best hyper parameters for that specific collection of training and validation data might be learned. In other words, the model that was trained using those hyperparameters can be biased either toward the validation dataset or just toward that specific division between the training and validation sets.

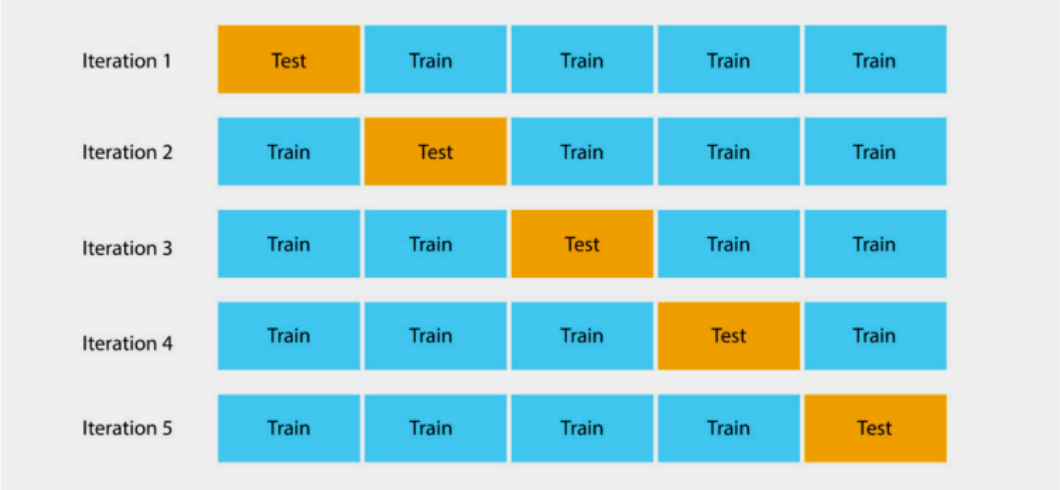

Figure 3. 5K-Fold

Cross validation is used to confirm that the model's hyper parameters are properly set and deliver the same results in unseen data [15]. There are several types of cross validation, however we will employ K-fold [16][19] cross validation in this study. Fig 2. illustrate the K-Fold technique used in this study. When using the K-fold method, the original training set is further splitted into N number of folds. In this study we set the value of N=5 which means that dataset is divided into 5 folds. Then one-fold is used as a test set

from that fold the remaining are then utilized for testing. So, after testing a model with a certain set of hyper parameters in this arrangement, the first iteration is finished. The different fold tests are utilized for assessment in the second iteration, while the remaining folds are than utilized for training with the same hyper parameters set. This process is repeatedly performed, i.e. each fold is tested exactly once. The performance of the hyper parameters on all of the folds, as well as the average evaluation metrics, are also recorded. In all folds, a good set of hyper parameters performs consistently. As a result, the mean value and deviation of the metrics are also necessary to assess their performance. The model is then trained and evaluated using the same approach with a new set of hyper parameters and the winner is the one with the best mean and the smallest deviation. GridSearchCV is used to automate the procedure.

However, going through all of the hyper parameter combinations that GridSearchCV [17] does is incredibly expensive. First, not only do we have to train and evaluate all 67 combinations, but we also have to do it N times, which is incredibly expensive. Furthermore, we virtually never receive the best parameters because the majority of hyper parameters are numerical values, and we must be picky in choosing from those numeric values, which already excludes the majority of hyper parameters. To get around this, we can utilize RandomSearchCV [18]. It is also seen that using GridSearchCV instead of RandomSearchCV does not result in a significant improvement in performance and is not worth the extra time spent training the model [19]. This module selects a random set of hyper parameters from which to train the model. The performance of the model trained with those hyper parameters is evaluated using K-fold cross validation once more. The module runs through a set number of combinations that the users have selected. We usually utilize 100-500 different hyper parameter combinations at random. Almost all of the models we tested and evaluated in the study yielded very excellent results using this method. In this approach, cross-validation aids in the development of a model with less bias and overfitting towards valid data.

4. Results

We deployed four supervised machine learning algorithms including Decision Tree, Random Forest, C-Support Vector Classification, and AdaBoost to examine the efficiency of our phishing website URL detection technique. These algorithms were chosen because to their shown effectiveness in classification tasks as well as the ability to manage high-dimensional data, noise, and non-linear patterns and features frequently seen in phishing detection datasets. A train-test split was used to train and test each model on the dataset. In almost all of the models we examined and analyzed in the study, the technique produces very excellent outcomes. Metrics including accuracy, log loss, AUC, precision, recall and F1 score are used to evaluate the performance of each model. By using these metrics, we can compare the strength and weakness of each algorithm. In following equations TP is used to for True Positive, TN for True Negative, FP for False Positive and FN for False Negative predicted instances. And Yi is used to show actual class and Yi, $\in$ {0,1}.

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

$$Precision = \frac{TP}{TP+TN} \quad \text{Eq (ii)}$$

$$Recall = \frac{TP}{TP+TN} \quad \text{Eq (iii)}$$

$$F1\ Score = 2 * (\frac{Recall}{Precision+Recall}) \quad \text{Eq (iv)}$$

$$Log\ Loss = -\ '1/n + \sum t=1n\ (Y1 \cdot Log\ (Y1) + (1 - Yi)(1 - Log(Yi)) \quad \text{Eq (v)}$$

We also utilized the above-mentioned measures for the assessment and comparison of the performance of all the models. All of the models were put to the test with the same data split. The experiment's outcomes are shown in the Table 1. In the column of Precision, Recall and F1 Score every cell have two values in the table the first value in cells is the non-phishing website measure, while the second value is the phishing website measure.

Table 1. Evaluation metrics for different Machine Learning Models

Model/Metrics	Accuracy	Log Loss	AUC Score	Precision	Recall	F1 Score
Decision Tree	0.968	0.968	0.977	[0.97, 0.97]	[0.96, 0.98]	[0.96, 0.97]
Random Forest	0.977	0.069	0.997	[0.99, 0.98]	[0.96, 0.99]	[0.97, 0.98]

Support Vector Classification	0.964	0.099	0.993	[0.96, 0.97]	[0.96, 0.97]	[0.96, 0.97]
AdaBoost	0.934	0.671	0.986	[0.93, 0.94]	[0.92, 0.95]	[0.93, 0.94]

The result shows that in terms of phishing website URL identification Random Forest consistently outperformed than the other models across all key metrics. It showed ideal performance in its classifications by achieving the highest accuracy 97%, Precision of 99% and F1 score 97%. SVM came in second, by achieving 96% accuracy.

Fig.4-7 shows the ROC AUC cure Decision Tree, SVM, Random Forest and AdaBoost classifiers respectively. Fig.7 illustrates the results comparison of all models.

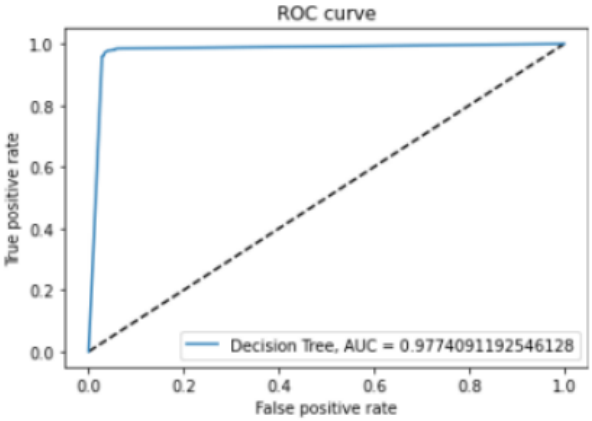

Figure 4. ROC Curve of Decision Tree

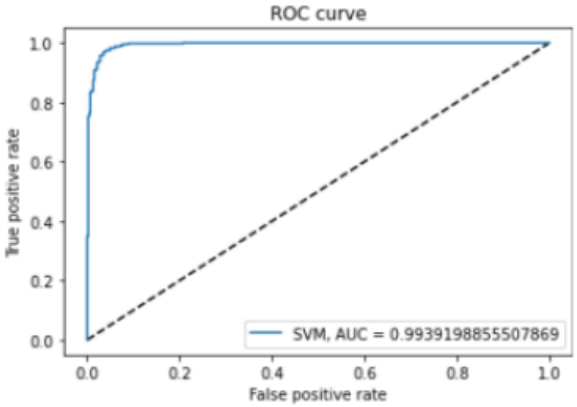

Figure 5. ROC Curve of SVM

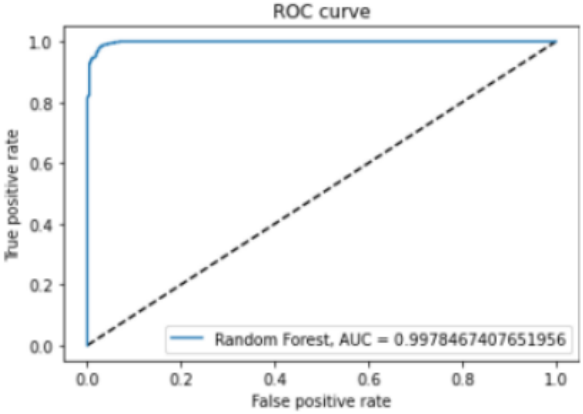

Figure 6. ROC Curve of Random Forest

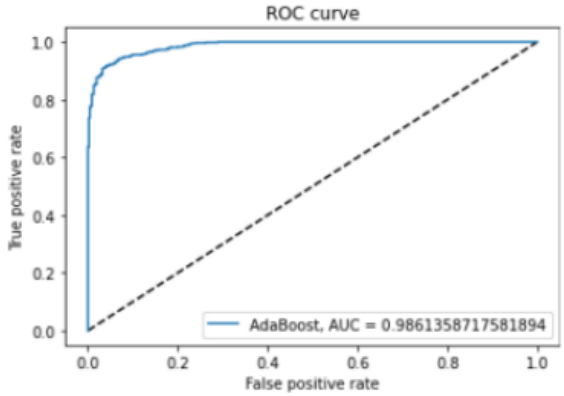

Figure 7. ROC Curve of AdaBoost

Finally, A unique architecture was introduced in this research. It aims to apply the benefits of algorithms which have been so effective in producing unstructured data solutions, to structured data. It also aims to get similar results in such structured data as some of the prominent boosting and bagging algorithms in the data science community. It also plans to explain its forecasts, which is critical when it comes to delivering machine learning-based solutions. We compared performance to that of some classic machine learning methods as well as some cutting-edge ones. In the phishing data, the model looked to perform well, with only Random Forest being able to match it. We discovered which features are most essential in evaluating whether a website is phishing or real utilizing model interpretability and correlation function. To protect consumers from phishing, SSL protection with authentic certification is critical. Fig 8. illustrate the performance comparison of all models.

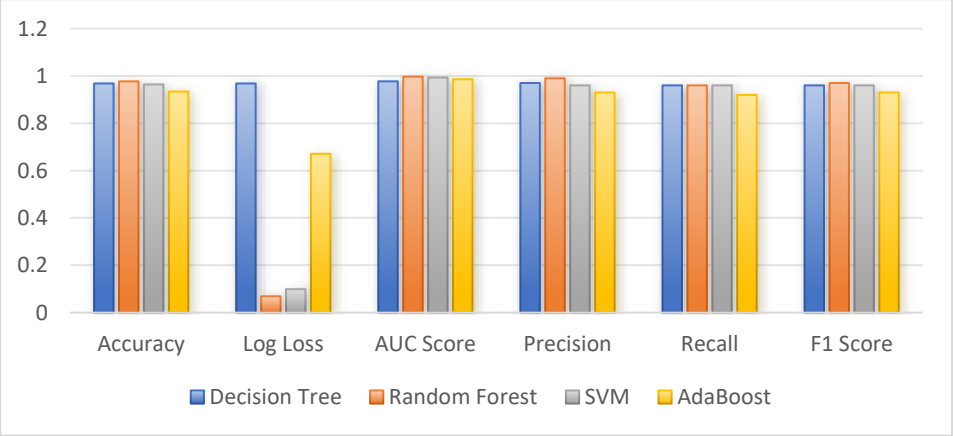

Figure 8. Result Comparisons

5. Conclusion

We developed and evaluated a hybrid machine learning technique for identifying phishing websites based on URL characteristics. We demonstrated a method for to differentiate phishing and original URLs by using machine learning classification algorithms including Decision Tree, Random Forest, C-Support Vector Classification and AdaBoost. Random Forest performed exceptionally well among all the models by achieving the maximum accuracy of 97%. The result highlights the importance of machine learning in cybersecurity. In future more advanced algorithms and detailed datasets may be used to improve the reliability of results.

References

1. DataReportal, "Digital 2024: Global Overview Report," DataReportal, Apr. 2024. [Online]. Available: https://datareportal.com/reports/digital-2024-global-overview-report.
2. A. Alsharnouby, F. Alaca, and S. Chiasson, "Why phishing still works: User strategies for combating phishing attacks," *International Journal of Human-Computer Studies*, vol. 82, pp. 69–82, Feb. 2015, doi: 10.1016/j.ijhcs.2015.05.005.
3. Al-Gharibah, A. N. Alshurideh, and A. M. Alsmadi, "Phishing Attacks in Social Engineering: A Review," *Journal of Cyber Security Technology*, vol. 7, no. 3, pp. 155–178, Aug. 2023, doi: 10.1080/23742917.2023.2121096.
4. H. Alasmary, F. Alhaidari, and A. Alzahrani, "Phishing Email Detection Based on Machine Learning Techniques: A Survey," *IEEE Access*, vol. 11, pp. 12345–12365, 2023, doi: 10.1109/ACCESS.2023.3245678.
5. M. Alenezi and M. Alhussain, "Phishing Attacks: A Comprehensive Review and Future Directions," *IEEE Access*, vol. 11, pp. 45678–45695, 2023, doi: 10.1109/ACCESS.2023.3278954.
6. Y. Zhang, M. Pan, and X. Wang, "Phishing Detection: A Review of State-of-the-Art Techniques and Emerging Trends," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 97–123, Firstquarter 2023, doi: 10.1109/COMST.2022.3145947.
7. Shinde, A. Pandey, R. Pawar, and V. Gangule, "Clustering and Bayesian Approach-based Model for Detection of Phishing," International Journal of Computer Applications, vol. 118, no. 24, pp. 30–33, May 2015, doi: 10.5120/20958-3385.
8. Z. Xiong, Y. Chen, and T. Li, "Efficient Phishing URL Detection Using Graph-based Machine Learning and Loopy Belief Propagation," arXiv:2501.06912, Jan. 2025.
9. W. Ali, "Phishing Website Detection based on Supervised Machine Learning with Wrapper Features Selection," International Journal of Advanced Computer Science and Applications, vol. 8, no. 9, pp. 72–78, Sept. 2017, doi: 10.14569/IJACSA.2017.080910.
10. PhishTank, "PhishTank: Community Site for Phishing Verification," [Online]. Available: https://www.phishtank.com. [Accessed: Jun. 6, 2025].
11. Alexa Internet, Inc., "Alexa Top Sites," [Online]. Available: https://www.alexa.com/topsites. [Accessed: Jun. 6, 2025].
12. Javeed, M., Aslam, S., Farhan, M., Aslam, M., & Khan, M. (2023). An Enhanced Predictive Model for Heart Disease Diagnoses Using Machine Learning Algorithms. Technical Journal, 28(04), 64-73. Retrieved from https://tj.uettaxila.edu.pk/index.php/technical-journal/article/view/1828.
13. S. G. F. Gomes, M. A. B. R. França, and J. Gama, "Data Preprocessing in Machine Learning: A Survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp. 3231–3248, Apr. 2023, doi: 10.1109/TKDE.2022.3154587.
14. M. U. Javeed, M. S. Ali, A. Iqbal, M. Azhar, S. M. Aslam and I. Shabbir, "Transforming Heart Disease Detection with BERT: Novel Architectures and Fine-Tuning Techniques," 2024 International Conference on Frontiers of Information Technology (FIT), Islamabad, Pakistan, 2024, pp. 1-6, doi: 10.1109/FIT63703.2024.10838424.
15. Aslam, S., Usman Javeed, M. ., Maria Aslam, S. ., Iqbal, M. M., Ahmad, H. ., & Tariq, A. . (2025). Personality Prediction of the Users Based on Tweets through Machine Learning Techniques. Journal of Computing & Biomedical Informatics, 8(02). Retrieved from https://www.jcbi.org/index.php/Main/article/view/796.
16. J. M. Gorriz, R. M. Clemente, F. Segovia, J. Ramirez, A. Ortiz, and J. Suckling, "Is K-fold cross-validation the best model selection method for machine learning?" *arXiv preprint arXiv:2401.16407*, Jan. 2024.
17. Scikit-learn Developers, "GridSearchCV — scikit-learn 1.6.1 documentation," 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html. [Accessed: Jun. 6, 2025].
18. Scikit-learn Developers, "RandomizedSearchCV — scikit-learn 1.6.1 documentation," 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html. [Accessed: Jun. 6, 2025].
19. Raza, A., Zongxin, S., Qiao, G., Javed, M., Bilal, M., Zuberi, H. H., & Mohsin, M. (2025). Automated classification of humpback whale calls in four regions using convolutional neural networks and multi scale deep feature aggregation (MSDFA). Measurement, 255, 118038. https://doi.org/10.1016/j.measurement.2025.118038.