A Cloud based Framework for Phishing Websites Detection Using Machine Learning Techniques

[1]Yasmeen, [2]Dr. Prasadu peddi

[1]Research Scholar, Dept. of CSE, Shri JJT University, Rajasthan, India, Assistant Professor, CVR College of Engineering, Telangana, India

[2] Associate Professor, Dept. of CSE, Shri JJT University, Rajasthan, India

Abstract –

Over a billion subscribers are served by cloud hosting services, which give them stable, affordable, dependable, high-speed, and internationally accessible resource access. Users frequently watch out for warning indicators of phishing attacks, such as websites with suspicious-looking domain names or those that lack an HTTPS certificate. Phishers often utilize social engineering tactics or create false websites to deceive their victims into divulging sensitive information such as account IDs, usernames, and passwords. This information can be used to steal money from individuals and corporations. Phishers have devised techniques to get around the many strategies to detect phishing websites. Nonetheless, these strategies have been put in place. Machine learning is one of the most effective methods for identifying potentially harmful behaviors. This is done so that approaches based on machine learning can identify the common features shared by the vast majority of phishing attacks. This research intends to train machine learning models and deep neural networks using the dataset produced to identify phishing websites. It is necessary to gather both phishing and benign URLs of websites to generate a dataset from which it will be possible to derive the required URL- and website content-based features. In this work, we compared the accuracy of the predictions made by several different machine-learning approaches for detecting phishing websites.

Index Terms: Cybercrime, Machine learning, Cloud, Phishing attacks, Classification

DOI Number: 10.14704/NQ.2022.20.12.NQ77263 **NeuroQuantology2022;20(12): 2700-2706**

1. INTRODUCTION

Organizations should prepare for the worst-case scenarios in a scenario where phishing attempts on the cloud are increasing daily. Instead of minimizing possible damage, this contingency planning may be accomplished through clever data backup techniques. Consumer data is always secured in the cloud and regularly backed up thanks to public platforms [1]. Cloud backup file encryption adds an extra degree of defense against unwanted external entities. In contrast to other machines, cloud service providers have unique ways of employing teams that continuously outperform cybercriminals [2, 3]. The best way to save the target's company, even if a hacker successfully encrypts the target's files and demands a ransom for the decryption key, is to restore the most recent cloud data backup file. Enterprises must also guard against internal threats typically brought on by human error. Office personnel may occasionally make unauthorized modifications to or erase corporate data out of the blue. For instance, what would have occurred if an employee had altered and deleted several slides from a PowerPoint presentation intended to be shared with a business partner for collaboration? The crucial slides are gone even if the presentation file still needs to be removed entirely, barring a backup of those slides. Utilizing a public cloud environment ensures that users may modify current permission settings and use earlier versions of documents to correct cryptographical problems [4, 5].

Implementing other protection measures should prevent consumers from ever having to fret about using their data backup. All bases must be covered, though! The industry doesn't have to worry about experiencing productivity losses or compliance absences in a cybersecurity breach scenario if a safe backup is in place. The suggested phishing anatomy provides a comprehensive breakdown of each of the four stages of a phishing attack, which are also depicted in Figure 1 as the primary flow of the process. On the other hand, as can be observed from the vast majority of efforts, the phishing procedure starts with collecting information about the victim. The first step in the planning stage is for the phisher to select the attack approach they will use.

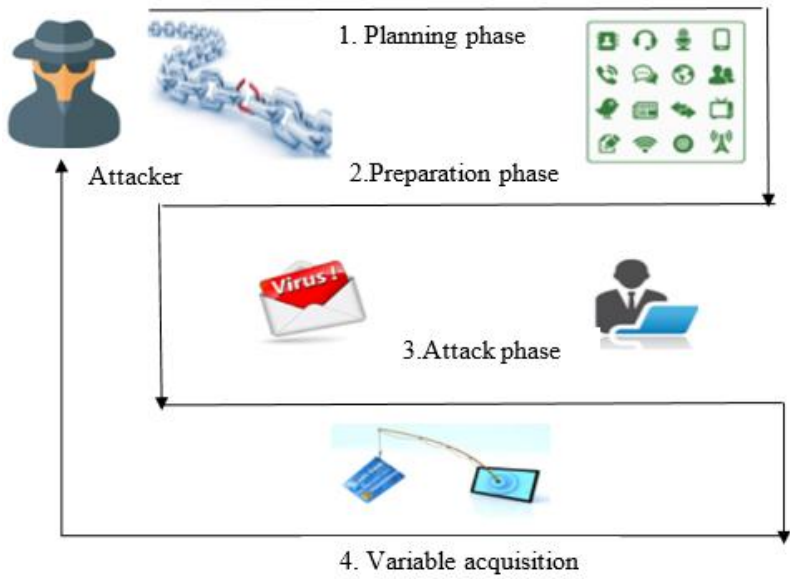

Figure 1: Process of a general phishing attack

During the preparation phase, also known as the second stage, the phisher will begin searching for potential entry points

eISSN 1303-5150

through which he might attract the victim. When the phisher enters the third attack step, they wait for the victim to answer before moving on to the next step. The final part of the phishing process is called the valuable acquisition phase, and it is at this phase the attacker may use and then collect the resources.

2. RELATED WORK

In this section, we discuss previous state-of-art works which uses machine learning for phishing attacks.

Let's look at an example to illustrate further the phishing method that was just discussed. An adversary may try to deceive a user of the internet by sending them a fake email in which they pose as the user's bank and demand that they verify their account information or risk having their account closed. Because the email uses the same logos, colors, and images as the customer's genuine bank, the user would think the email is authentic because of these similarities. The submitted data will then be sent straight to the phisher, who will utilize it for a variety of nefarious activities, including money withdrawal, blackmailing, or perpetrating more fraud [6].

In [7], the authors propose an anti-phishing strategy that guards users' data against phishing attempts by focusing on phishing URLs that take users to dubious login web pages to get around some of these constraints. The suggested method is a hybrid strategy that uses existing solutions, such as the blacklist and whitelist approaches. It adds two new anti-phishing techniques, page detection and fake data techniques, in which phishing detection is carried out in three steps. The authors are confident that this novel strategy will be able to address several issues that now outbreak existing systems, such as identifying new phishing links that have not yet been reported to blacklist databases. The experimental investigation produced positive and enticing findings, outperforming some of the commercial anti-phishing solutions that were previously available.

In [8], the authors outline a simple phishing detection algorithm for mobile devices that uses URLs to discriminate between phishing and legal websites. They used Artificial Neural Networks (ANNs) to analyze the URL-based and HTML-based properties of websites as a baseline performance. A model search yielded 15 ANN models with 96% or above accuracy levels, which is on par with other innovative approaches. The testing performance of deep ANNs is then restricted to URL-based features; however, the performance of all models is poor, with the maximum accuracy being only 86.2%. This shows that more than URL-based characteristics are needed to find phishing websites, especially when combined with deep ANNs.

The authors of [9] suggested a phishing website detection method that we named HinPhish. This creates a heterogeneous information network by extracting multiple link associations from websites and using domains and resource objects. The phish score of the target domain on the webpage is calculated with the assistance of a specialized algorithm that uses the characteristics unique to the various kinds of links. In addition, it can improve the accuracy of detection and make it more expensive for those who try to conduct phishing attacks. After careful testing, it was found that HinPhish could have an accuracy of 0.9856 and an F1-score of 0.9858.

The authors of [10] suggested a hybrid method that identifies websites as phishing, legitimate, or suspicious by intelligently combining the K-nearest neighbors (KNN) algorithm with the support vector machine (SVM) method in two steps. First, the KNN was initially used as a dependable classifier for noisy data analysis. Then, the SVM is used as a potent classifier second. The suggested method combines SVM's efficiency with KNN's ease of use. According to the experimental findings, the proposed hybrid strategy had the best accuracy compared to other methodologies, coming in at 90.04%.

According to data on the global economy, software as a service (SaaS) and webmail sites continue to be the most popular targets of phishing attacks. The widespread use of phishing has resulted in devastating losses for many businesses. Phishing websites can be removed using a variety of different approaches. Every one of these strategies has the potential to be used at multiple stages of the attack pipeline, including network-level defense, authentication, client-side tools, user training, server-side filters and classifiers, and so on.Even while every type of phishing attack has a few phishing attempts. In this section, we evaluate the efficacy of several machine-learning approaches to determine which ones are most suited to detecting phishing websites. We looked into Logistic Regression, Decision Tree, Random Forest, Ada-Boost, Support Vector Machine, KNN, Artificial Neural Networks, Gradient Boosting, and XGBoost as ways to train a machine to learn.

3. METHODOLOGY

Using the dataset that was produced, this research aims to train deep neural networks and machine learning models so that they can recognize phishing websites. First, web phishing and benign URLs are collected to produce a dataset from which the essential URL and content-based properties of websites can be retrieved. This is done for the purpose of creating a dataset. Then, comparisons and assessments are made on the performance levels of each model. Because of machine learning, conducting data analysis may now be done more efficiently and timely. It has recently demonstrated encouraging results in various real-time classification problems. The most crucial advantage of machine learning is that it allows models that can be altered to do specific jobs, such as detecting phishing. Since phishing is a classification problem, machine learning models may prove to be a helpful tool.

Furthermore, machine learning models may rapidly adapt to changing circumstances to identify fraudulent transaction patterns and contribute to the development of learning-based identification systems. This may be possible thanks to the potential for rapid adaptation offered by machine learning. In

eISSN 1303-5150

supervised machine learning, an algorithm attempts to train a function that translates input to output based on input-output pairs. This is how most of the machine learning models are categorized here. In unsupervised machine learning, an algorithm attempts to train a function that translates input to output based on unlabeled data. Then, it concludes a process based on the labeled training data comprised of a collection of training cases. In Figure 2, we show how the machine-learning strategies we used in our research worked.

Data Collection: The University of New Brunswick's dataset, accessed at https://www.unb.ca/cic/datasets/url-2016.html, is used to collect the data. This dataset contains URLs that are considered to be professional and credible. The list is combed through, and five thousand URLs are selected randomly. The open-source PhishTank project is responsible for the collection of phishing URLs. This website provides a database of phishing URLs that is kept up to date hourly and available in various formats, such as CSV, JSON, and others. The collected dataset is then combed through, and a random selection of 5000 URLs is the collected dataset is then combed through, and a random sample of 5000 URLs is made.

Data preprocessing:

In this step, the data is cleaned up with the help of pre-processing data procedures, and then it is changed before it is utilized in the models. Cleansing, instance selection, normalization, transformation, etc., are all included. Pre-processing the data might affect how the final processing's findings are understood. Data filling, noise smoothing, identifying or eliminating outliers, and resolving incompatibilities might all step in the data cleaning process. A technique for adding specific databases or data sets is called data integration. Data transformation is gathering and normalizing data to measure a particular data set.

Feature extraction and selection:

The feature extraction file, without shuffling, concatenates the extracted features of the genuine and phishing URL datasets. The top 5000 rows of authentic URL data and the lowest 5000 rows of phishing URL data were produced as a consequence. The following feature category is chosen:

- Features based on the address bar
- Dominant Features
- JavaScript and HTML-based Feature

The dataset is used to extract a total of 17 features. Next, we must shuffle the data to balance the distribution while dividing it into training and testing sets. Even the scenario of overfitting during model training is avoided by doing this.

Models for machine learning and training:

This is a supervised machine-learning problem from the dataset above. Classification and regression are the two main subtypes of supervised machine learning issues. This data collection has a categorization issue because the input URL might be either legal (0) or phishing (1). The following supervised machine learning models (classification) were taken into consideration in this study to train the dataset: Decision Tree , Random Forest, Multiple-layer perceptrons. (Autoencoder Neural Network), SVMs (Support Vector Machines) and Proposed XGBoost.

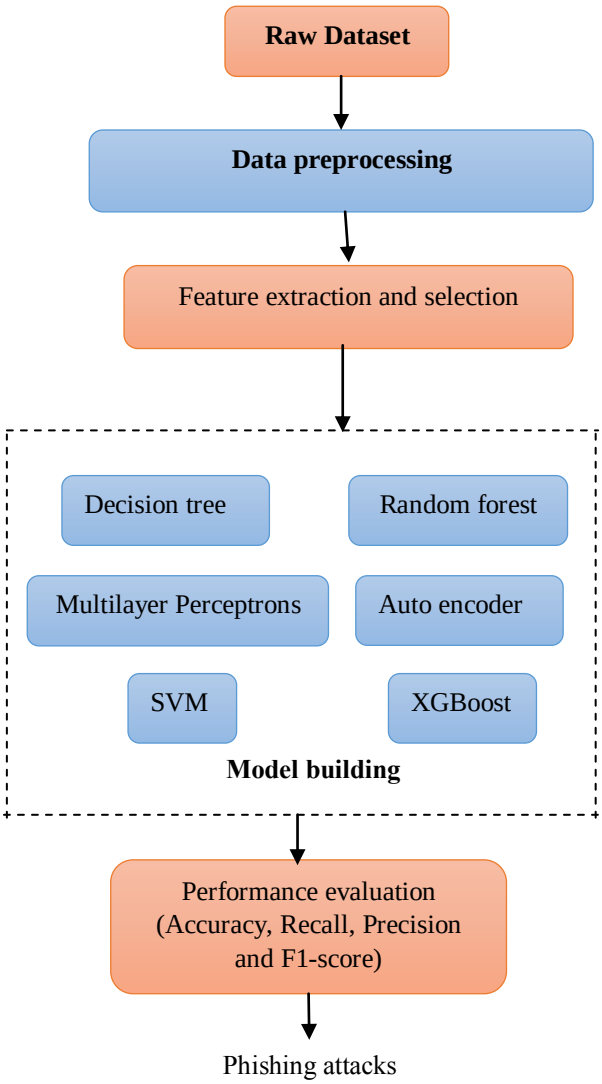

Figure 2: Proposed machine learning based Phishing attacks detection approach

Classifier using Decision Trees (DT)

A decision tree-based classifier called a decision tree classifier has tree nodes with values that have been "learned" from training cases and branches that point in the direction of the best choice that can be made about the input instance. The internal node of a decision tree is where one or more attribute tests, such as a range, are performed on one or more additional variables. Based on the attribute test, one might consider each internal node a "splitter" that partitions the instance space into two or more distinct smaller subspaces. The leaf nodes of a decision tree are the judgments or

eISSN 1303-5150

classifications of an input instance that meet the requirements along the path from the root node to the leaf nodes. It has been established that learning an ideal decision tree is a generic NP-complete issue. When decision trees produce complex trees for vast attribute spaces, overfitting is an issue, a sign of improper data space partitioning. On the other hand, the performance of decision trees in tiny attribute spaces is reasonably good. With only seven characteristics in our method, decision trees are efficient.

Random Forest

Numerous decision trees are combined to form a "forest" using the supervised machine learning method known as Random Forest. Problems with classification and regression can be solved with its help. It supported the idea of ensemble learning, which may integrate several different classifiers to tackle a complex problem and improve the model's overall performance. A Random Forest combines the results of many decision trees to get a more precise forecast. The Random Forest model is constructed on the concept that the usage of many models in combination with one another results in significantly improved performance compared to that of the individual models. Within the Random Forest classification process, every tree in the forest has a vote. The classification that received the most votes is the one that the forest uses.

In contrast, when performing regression with Random Forest, the forest takes each tree's data into account. Some decision trees may get it wrong, but the vast majority will get it right. Because of this, we can trust that the overall result will head in the right direction. It takes significantly less time to train when compared to other methods. It provides an accurate prediction of the result.

Even when dealing with large datasets, it performs effectively. In addition, its accuracy is maintained even without a large piece of data. Row sampling is carried out through Bootstrap, and sample datasets are created for each model. These sample datasets are aggregated into condensed statistics for observation and fusion. Variance is an error that results from minor differences in the training dataset. High variance tends to train noisy or irrelevant data in the dataset rather than the intended signal-producing outcomes. Overfitting is the label given to this problem. An overfitted model will perform well during training but will need help to tell the difference between the signal and noise during testing. A technology of the bootstrap method with a high difference is bagging.

Multiple-layer perceptrons

Combining a feed-forward neural network with a multilayer perceptron (MLP), also known as an MLP addition, as seen in Figure 3, is composed of three different layers: input, output, and hidden. The signal will be processed and brought into the system at the input layer. The output layer finishes the essential task, which may include categorization and prediction. In a multilayer perceptron (MLP), the true computational engine comprises an arbitrary number of hidden layers between the input and output layers. An MLP functions similarly to a feed-forward network in that the data flow from the input layer to the output layer in the forward direction. The neurons of the MLP can be educated using a technique known as backpropagation learning. Since MLPs are designed to approximate any continuous function, they may be able to tackle problems that cannot be separated linearly. MLP is mainly used for three different tasks: classifying and identifying patterns, predicting patterns, and getting close to patterns.

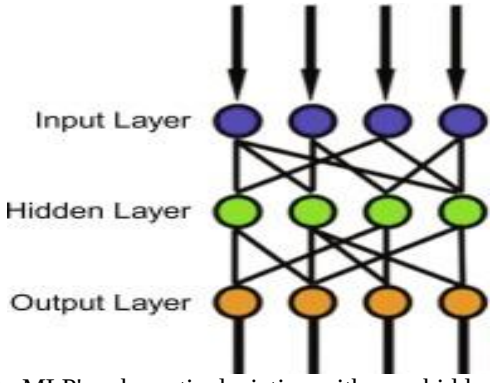

Figure 3 : MLP's schematic depiction with one hidden layer

Autoencoder

Using this particular type of neural network, it is feasible to acquire the ability to learn a compressed version of the raw input. An encoder and a decoder sub-model combine to make an autoencoder. The encoder first compresses the input, then the decoder attempts to reconstruct the input from the encoder's compressed form using the information it has received. Following the completion of the training, the decoder model is discarded, but the encoder model is kept. The encoder can then be used to clean up raw data and extract features that can be used to train a different machine-learning model. This can be done using other data cleaning and extraction methods.

Support vector machines (SVMs).

This is a prevalent example of a classification method that is utilized. The support vector machine (SVM) finds the location between two classes closest to one another by calculating the most considerable distance between the classes. This approach uses a supervised learning model to classify data in linear and nonlinear ways. In the process of nonlinear classification, a kernel function is utilized to transform the input to a feature space with a higher dimension. Even though classification is a common use for SVMs and they are compelling, they have several areas for improvement. Calculations at a high level are required to train the data. In addition, because they are sensitive to noisy data, they can easily become overfit to their data, which is a problem.

XGBoost (Proposed)

Extreme Gradient Boosting, also known as XGBoost, is a form of boosting based on the Gradient Boosting Machine. This machine combines gradient descent and boosting to achieve optimal results (GBM). Boosting is an approach to

eISSN 1303-5150

ensemble learning that involves distributing the training data distribution with a variable weight for each iteration of the algorithm. During each iteration of the boosting process, weight is added to error samples that have been wrongly classified and subtracted from error samples that have been successfully classified. This results in a shift in the distribution of training data. GBM tries to find the best balance between the regularized goals in the equation and the second-order gradient statistics (1).

$$l(\varphi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \qquad (1)$$

$$\text{Where } \Omega(\text{f}) = \gamma \text{T} + \frac{1}{2}\lambda \|w\|^2$$

l is a differentiable convex loss function that calculates the variance between the goal and prediction y_i, and penalizes the model's complexity. The goal of GBM, a tree-based technique, is to locate the best candidate split points, which is challenging for big datasets.

4. RESULTS AND DISCUSSIONS

For the analysis of model performance in our experiments, we employed 10-fold cross-validation. The data set was split into ten smaller samples. A portion of the sample is used to test the data, while the remainder is used to train the models. We must employ a binary classification model since phishing detection is a classification problem; we consider "-1" to be a valid sample and "1" to be a phishing sample. We used many machine learning models to find phishing websites in this work. These models included the Decision Tree (DT), Multilayer Perceptrons, Random Forest, Autoencoder, SVM, and XGBoost.

Performance Assessment:

A data frame is produced to compare the model's performance. The lists constructed to hold the model's findings are the columns of this data frame. When evaluating the models, accuracy is taken into account. Therefore, we assess these models' recall, accuracy, precision, F1-score, training time, and testing time. We applied various feature selection techniques and hyperparameter optimization to get the best results.

Table 1: compares the effectiveness of the models on data frames

ML Model	Train Accuracy	Test Accuracy
Decision tree	0.810	0.826
Random forest	0.814	0.834
Multilayer Perceptrons	0.858	0.863
AutoEncoder	0.819	0.818
SVM	0.798	0.818
XGBoost (Proposed)	0.866	0.864

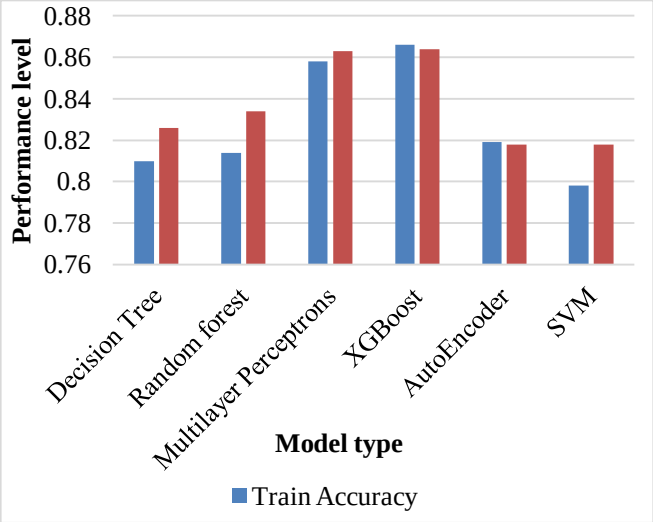

Figure 4: Comparison of the 6 models' performance on data frames before sorting

Figure 4 makes it clear that the suggested Boost's key advantages are its quick speed compared to other algorithms, such as DT and SVM, and its regularization parameter's successful variance reduction. But even without the regularization parameter, this technique uses subsamples from characteristics like random forests and a learning rate, further enhancing its generalization capacity.

Table 2: Models' data frame performance comparison

ML Model	Train Accuracy	Test Accuracy
XGBoost	0.866	0.864
Multilayer Perceptrons	0.858	0.863
Random Forest	0.814	0.834
Decision Tree	0.810	0.826
AutoEncoder	0.819	0.818
SVM	0.798	0.818

Finally, as anticipated, XGBoost's training and testing accuracy were significantly more significant than other machine learning models, at almost 0.866 and 0.864, respectively.

eISSN 1303-5150

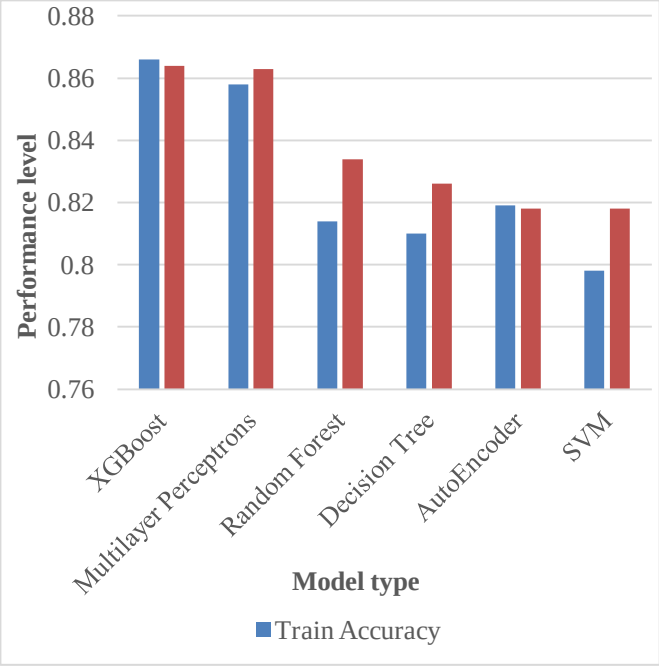

Figure 5: Comparison of the 6 models' performance on data frames after sorting

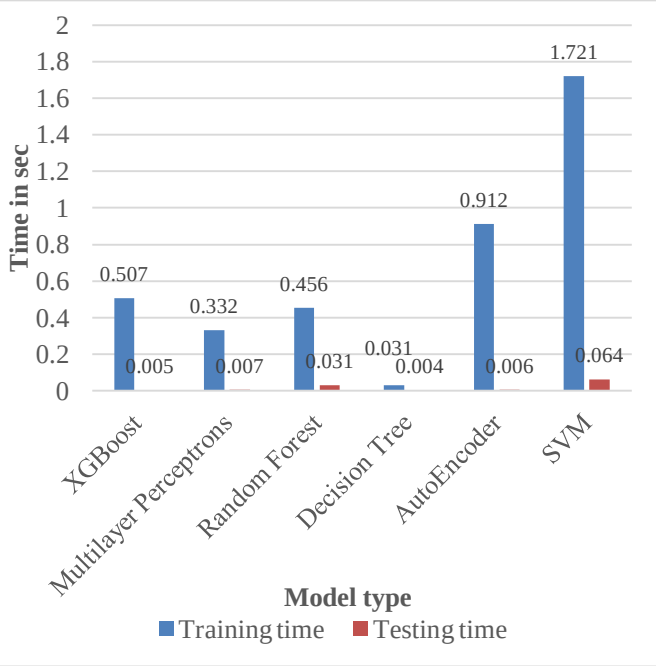

Figure 6: compares the six classifiers' performance in terms of training and testing time.

Figure 5 makes it clear that the suggested XGBoost's key advantages are its quick speed compared to other algorithms, such as DT and SVM, and its regularization parameter's successful variance reduction. But even without the regularization parameter, this technique uses subsamples from characteristics like random forests and a learning rate, further enhancing its generalization capacity. As anticipated, XGBoost had training and testing accuracy significantly greater than other machine learning models, at almost 0.866 and 0.864, respectively. Moreover, only the order of the model changes after sorting; their performance levels remain the same.

Table 3: Classification outcomes for various training and testing methodologies

Classifier name	Training time	Testing time
XGBoost	0.507	0.005
Multilayer Perceptrons	0.332	0.007
Random Forest	0.456	0.031
Decision Tree	0.031	0.004
AutoEncoder	0.912	0.006
SVM	1.721	0.064

Figure 6 demonstrates that, as predicted, SVM training time was much longer than other machine learning models. XGBoost's testing period was somewhat faster than other models (classifiers). This is a result of the limited size of our training data. An autoencoder model, like XGBoost, is unable to explain why it identified a website as a phishing one. We can more readily express important characteristics thanks to their explain ability.

Table 4: Classification results for accuracy, recall, precision, and F1-score using various techniques.

Classifier name	Accuracy	Recall	Precision	F1-score
XGBoost	0.983	0.988	0.986	0.978
Multilayer Perceptrons	0.945	0.962	0.956	0.967
Random Forest	0.976	0.982	0.967	0.976
Decision Tree	0.967	0.974	0.969	0.971
AutoEncoder	0.956	0.942	0.924	0.931
SVM	0.931	0.932	0.920	0.922

Figure 7 demonstrates the excellent accuracy, relative robustness against noise, ease of implementation, and implicit feature selection of the proposed XGBoost and Random Forest models. The primary distinction between XGBoost and Random Forest and the other four classifiers is that they are unaffected by noise. When we were creating our model, the most significant disadvantage of using random forests that we came across was the large number of hyperparameters that needed to be set to achieve the best possible performance. In addition, Random Forest introduces an element of

eISSN 1303-5150

unpredictability into both the training and testing sets of data, which is a feature that is only suitable for some data sets. Because of this, the performance of Random Forests is somewhat inferior to that of the recommended XGBoost classifier.

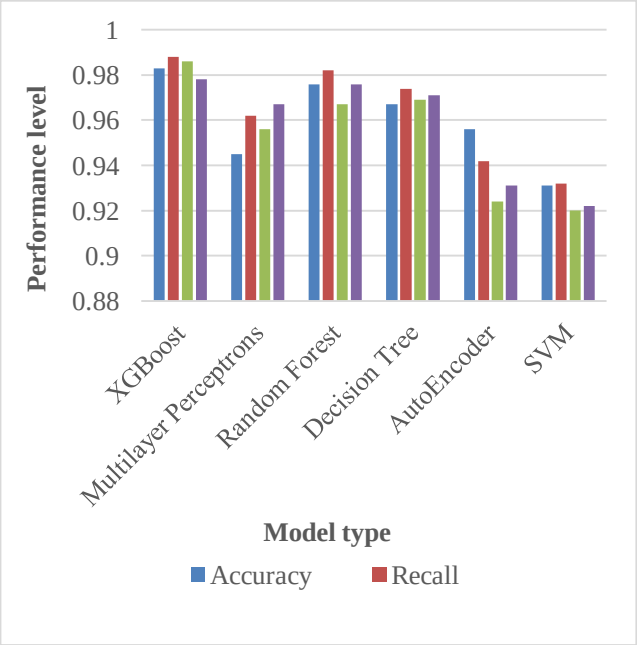

Figure 7: compares the 6 classifiers' performance in terms of accuracy, recall, precision, and F1-score

5. CONCLUSION

Phishing attacks continue to be one of the most significant threats individuals and enterprises face in the modern era. According to the information presented in the article, this is primarily caused by human participation in the phishing cycle. The classifiers investigated were Logistic Regression, Decision Tree, Support Vector Machine, Ada Boost, Random Forest, Neural Networks, KNN, Gradient Boosting, and XGBoost. Other classifiers that were investigated were Random Forest and Ada Boost. The results of our research indicate that combining several types of classifiers, such as Random Forest and XGBoost, results in a computation that is both efficient and accurate. Because the primary goal of ensemble algorithms is to combine a large number of less capable learners into a single, more capable learner, ensemble-based learning is the most common method used in practice for solving classification problems. Also, to keep up with the constantly changing ways that phishing attacks work, we will look into the possibility of coming up with a new way to get more features from the website.

REFERENCES

[1] A., Pasumpon & Smys, Smys. (2020). Effective Fragmentation Minimization by Cloud Enabled Back Up Storage. Journal of Ubiquitous Computing and Communication Technologies. 2. 1-9. 10.36548/jucct.2020.1.001.

[2] ma, Dan & Kauffman, Robert. (2014). Competition Between Software-as-a-Service Vendors. IEEE Transactions on Engineering Management. 61. 717-729. 10.1109/TEM.2014.2332633.

[3] Gul, Zulfiqar & Hussain, Zahid & Arain, Aijaz & Ali, Munwar. (2021). Cloud Service Evaluation and Selection based on User Preferences and Location. 29. 10.22937/IJCSNS.2021.21.10.4.

[4] Härting, Ralf-Christian & Möhring, Michael & Schmidt, Rainer & Reichstein, Christopher & Keller, Barbara. (2016). What Drives Users to Use CRM in a Public Cloud Environment? -- Insights from European Experts. 10.1109/HICSS.2016.495.

[5] Raja, Abdul Sattar & Razak, Shukor. (2015). Analysis of Security and Privacy in Public Cloud Environment. 2015 International Conference on Cloud Computing, ICCC 2015. 10.1109/CLOUDCOMP.2015.7149630.

[6] Shah, Kaushal & Jinwala, Devesh. (2021). Privacy preserving secure expansive aggregation with malicious node identification in linear wireless sensor networks. Frontiers of Computer Science. 15. 10.1007/s11704-021-9460-6.

[7] Alzamil, Zakarya & Aljurayyad, Abdulmalik & Almajally, Mohammed & Abuhemid, Mohammed & Alsharafi, Abdulmajeed & Alfreddi, Bader. (2020). A hybrid phishing detection approach for mobile application. International Journal of Security and its Applications. 14. 15-28. 10.33832/ijsia.2020.14.3.02.

[8] Haynes, Katherine & Shirazi, Hossein & Ray, Indrakshi. (2021). Lightweight URL-based phishing detection using natural language processing transformers for mobile devices. Procedia Computer Science. 191. 127-134. 10.1016/j.procs.2021.07.040.

[9] Guo, Bingyang & Zhang, Yunyi & Xu, Chengxi & Shi, Fan & Li, Yuwei & Zhang, Min. (2021). HinPhish: An Effective Phishing Detection Approach Based on Heterogeneous Information Networks. Applied Sciences. 11. 9733. 10.3390/app11209733.

[10] Taha, Altyeb. (2017). Phishing Websites Classification using Hybrid SVM and KNN Approach. International Journal of Advanced Computer Science and Applications. 8. 10.14569/IJACSA.2017.080611.