*algorithms*

*Article*

# Machine-Learning Techniques for Predicting Phishing Attacks in Blockchain Networks: A Comparative Study

**Kunj Joshi [1], Chintan Bhatt [1,*], Kaushal Shah [1], Dwireph Parmar [1], Juan M. Corchado [2,3,4], Alessandro Bruno [5,*] and Pier Luigi Mazzeo [6]**

1 Department of Computer Science and Engineering, School of Technology, Pandit Deendayal Energy University, Gandhinagar 382007, India; kunj.jce19@sot.pdpu.ac.in (K.J.); kaushal.shah@sot.pdpu.ac.in (K.S.); dwireph.pce19@sot.pdpu.ac.in (D.P.)
2 BISITE Research Group, University of Salamanca, 37007 Salamanca, Spain; jm@corchado.net
3 Air Institute, IoT Digital Innovation Hub, 37188 Salamanca, Spain
4 Department of Electronics, Information and Communication, Faculty of Engineering, Osaka Institute of Technology, Osaka 535-8585, Japan
5 Department of Business, Law, Economics and Consumer Behaviour "Carlo A. Ricciardi", IULM AI Laboratory, IULM University, 20143 Milan, Italy
6 ISASI Institute of Applied Sciences & Intelligent Systems-CNR, 73100 Lecce, Italy; pierluigi.mazzeo@cnr.it
* Correspondence: chintan.bhatt@sot.pdpu.ac.in (C.B.); alessandro.bruno@iulm.it (A.B.)

**Abstract:** Security in the blockchain has become a topic of concern because of the recent developments in the field. One of the most common cyberattacks is the so-called phishing attack, wherein the attacker tricks the miner into adding a malicious block to the chain under genuine conditions to avoid detection and potentially destroy the entire blockchain. The current attempts at detection include the consensus protocol; however, it fails when a genuine miner tries to add a new block to the blockchain. Zero-trust policies have started making the rounds in the field as they ensure the complete detection of phishing attempts; however, they are still in the process of deployment, which may take a significant amount of time. A more accurate measure of phishing detection involves machine-learning models that use specific features to automate the entire process of classifying an attempt as either a phishing attempt or a safe attempt. This paper highlights several models that may give safe results and help eradicate blockchain phishing attempts.

**Keywords:** blockchain; machine learning; phishing; cyberattacks; Ethereum

check for updates

**Citation:** Joshi, K.; Bhatt, C.; Shah, K.; Parmar, D.; Corchado, J.M.; Bruno, A.; Mazzeo, P.L. Machine-Learning Techniques for Predicting Phishing Attacks in Blockchain Networks: A Comparative Study. *Algorithms* **2023**, *16*, 366. https://doi.org/10.3390/a16080366

Academic Editors: Yue Duan and Frank Werner

Received: 15 June 2023
Revised: 15 July 2023
Accepted: 28 July 2023
Published: 29 July 2023

## 1. Introduction

Security in the blockchain has become a significant concern because of the recent developments in the field. One of the most common and easiest attacks carried out by the attacker is the phishing attack, where the attacker tricks the miner into mining a malicious block into the blockchain using social engineering techniques. Many blockchain projects with high-grade security implications were found to be susceptible to phishing attacks, and many members of the community and its investors have lost large sums of money [1]. A report in 2017 showed that more than 50% of attacks carried out on Ethereum were phishing attacks [2]. Phishing attacks are so common that they threaten the three pillars of security: privacy, integrity, and availability.

A phishing attack is usually undertaken with a series of aims; the attacker either has the goal of monetary gain or installing malicious code into the environment. The attack is mainly carried out using social engineering. In it, the miner is tricked into thinking that the malicious block is genuine and that linking it to the blockchain would be of benefit.

If the miner links the malicious block, it is considered incorrect mining, and the miner is not rewarded. This makes that chain of the main blockchain dysfunctional, and further blocks link to that chain impossible.

Currently, many methodologies are available to prevent this kind of attack from being carried out. The principal method is the consensus protocol, wherein a block is only linked to the blockchain if all the block owners agree upon its addition. However, it is a very time-consuming method [3], with drawbacks and biases. This has led to zero-trust policies in the blockchain and research on the same. These policies have been implemented in major blockchains such as Bitcoin and Ethereum [4] and have shown promising results. Nonetheless, it is a highly complex mechanism to deploy, and many new blockchains are still trying to set foot in the same territory. Zero-trust policies are still being researched and might also hold a promising future. However, the attackers will not wait until these policies are implemented to start their attacks. A newer and more efficient methodology is required to prevent phishing attempts. This can be achieved using machine-learning models and methods. This paper defines several models with promising results that may be helpful for the same. Using certain features, the researchers reached 98.28% accuracy in blockchain phishing attempt detection.

The project's novelty lies in using different machine-learning algorithms and deviating from the original process of thought. Whilst using machine learning, the researchers have preferred to use graph-based mapping methods and neural networks for prediction; however, the best accuracy they achieved lay in the range of 85–90%. This project also aims at a higher accuracy using other machine-learning models. Data preprocessing was carried out at the minute level to remove any discrepancies, contributing to higher accuracy. This paper focuses on blockchain phishing detection on Ethereum and achieving accuracy in that environment.

This paper is divided into nine sections. Section 1 is the introduction to the project and the topic of interest. Section 2 covers the literature review, and Section 3 defines the dataset. Section 4 outlines the procedure of the experiment. Section 5 defines the models used in the investigation. Section 6 describes the observations made during the experiment, and Section 7 concludes the achieved research, Section 8 points to the research limitations and possible future work.

This paper presents detailed research on machine-learning algorithms, which can accurately predict phishing attempts in the blockchain. The researchers developed models using their prior knowledge and the help of several experts. Hence, the paper centers on the following research questions:

RQ1 How accurately can machine-learning algorithms predict a phishing attempt in a blockchain environment, and which gives the best results?

RQ2 What is the best model to use in diagnosing phishing attempts based on accuracy analysis and performance analysis?

## 2. Literature Review

This section groups several methods for predicting phishing attacks in blockchain networks into two main approach categories: deep learning and traditional machine learning. We know that deep learning is often mentioned as part of the machine-learning family. This paper is not aimed at debating on the matter. However, here we refer to traditional ML techniques as all those methods that do not rely on deep neural networks (DNNs).

### *2.1. Traditional Machine-Learning Approaches*

Ye, Li, Cai, Gu, and Fukuda et al. [5] researched and simulated a stable security system in the blockchain which can withstand attacks such as 51% attacks and other attacks. The research was published after the WannaCry Ransomware outbreak, and security in the blockchain became a subject of intense debate. The authors' simulation made an attacking state distinguishable from an honest state. Their research also concluded that the blockchain environment became weaker and more vulnerable to other attacks because of high-power attacks. Zhang, Xue, and Liu et al. [6] propose a blockchain privacy and security model. This proposed model is considered a breakthrough in blockchain technology's cryptography and cybersecurity aspects. Security has improved due to the notion of smart contracts, smart grids, and other security practices in all fields of computer science, which imply zero-trust policies.

Aggarwal and Kumar et al. [7] mention in their analysis that the blockchain is more prone to attacks if there are more miners. The authors analyzed attacks such as 51% attacks, Sybil attacks, and denial of service (DoS) attacks. The attacks also depend on the hash computing power of the computers on which the blockchain is hosted. Phillips and Wilder et al. [8] propose that it is straightforward to perform phishing attacks on the blockchain. Many scammers try to redirect noble users to malicious websites, which ask them for advanced fees and scam them by making them believe they are mining for blocks. The authors have also used the machine-learning clustering technique DBSCAN to detect phishing. Holub and O'Connor et al. [9] developed a phishing detection system called COINHOARDER, which traced and decided whether a Bitcoin transaction was a phishing or a safe attempt. The authors' project was limited only to a Ukrainian-based cryptocurrency. Fu, Yu, and Feng et al. [10] also developed a phishing detection system using machine-learning models, namely convolution networks, to predict an Ethereum transaction as fraudulent or safe. Their models had 88.02% accuracy. Yuan, Yuan, and Wu [11] also developed a transaction-based phishing detection model, which detected and classified models as safe or phishing by mapping their transactions to subgraphs. The authors also worked on an Ethereum blockchain, improvised a Graph2Vec, and made stronger classifications. Zhang and Chen [12] used multi-channel graph classification in their phishing detection models and graph neural networks to map attempts as either phishing or safe. They achieved 91% accuracy in the attempt. In their comparative study, Bhowmik, Chandana, and Rudra et al. [13] state that the decision tree is the most accurate model for detecting phishing attacks compared to the other models based on the naïve Bayes classifier. Arshey and Viji [14] published their work on thwarting security issues in blockchain environments with machine-learning solutions. Their work was more theoretical but had many practical applications, such as COVID-19 data storage and cloud technology.

### *2.2. Deep-Learning Approaches*

Phishing attacks have been the subject of numerous research works over the last few years. Catal et al. [15] surveyed several deep-learning-based methods for phishing detection in a broader landscape. In this subsection, a description of deep-learning-based methods for phishing attack detection in the blockchain is provided.

Parra et al. [16] presented a deep-learning model to classify phishing emails and then present the results to human evaluators for validation. The feedback was used to improve the accuracy of the model over time. The approach has been shown to be effective in detecting both known and unknown phishing attacks. The deep-learning model used to detect phishing attacks is a combination of various machine-learning algorithms, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and generative adversarial networks (GANs). These models are trained on large datasets of known phishing emails and websites to identify patterns that indicate potential threats. The specific model used depends on the type of attack being detected, as well as the available resources and expertise. Fu et al. [10] proposed a high-performance model for detecting phishing fraud in blockchain cryptocurrency transactions using graph convolutional networks. The model divided the transaction graph into "sender" and "receiver" graphs and utilized a double-layer graph convolution network for feature learning. Testing on Ethereum data achieved an accuracy of 88.02% and an F1 score of 88.14%, outperforming other models and introducing a new concept for detecting phishing scams in blockchain networks.

Kim et al. [17] addressed the privacy vulnerability in a blockchain-based smart healthcare system where personal information is exposed. They presented an attack scenario that infers user identity by analyzing the blockchain graph and constructing account-transaction graphs. Graph-embedding algorithms and machine-learning techniques are used to identify participants, achieving an inference performance of up to 0.94 in the F1 score when tested on Ethereum. Ali et al. [18] addressed security and latency issues in blockchain-based healthcare systems and proposed a hybrid deep-learning-based homomorphic encryption (HE) model for the Industrial Internet of Medical Things (IIoMT). The model integrates HE with a consortium blockchain to provide privacy, resistance to attacks, and the ability to perform statistical and machine-learning operations on encrypted EMR data. Comparative simulation analysis demonstrated enhanced security, efficiency, and transparency compared to existing approaches. In addition to the aforementioned techniques, Sankar Roy et al. [19] introduced a deep-learning-based fraud detection model using Ethereum blockchain transaction data. The model relies on long short-term memory (LSTM) and dense units, with feature selection using information gain. Conversely, Yazdinejad et al. [20] designed a secure, intelligent fuzzy blockchain framework that incorporates a fuzzy deep-learning model and fuzzy control systems for network attack detection. The framework employs metaheuristic algorithms for optimization and fuzzy matching for fraud detection. Evaluation results demonstrate the efficiency and effectiveness of the framework in threat detection and decision-making in blockchain-based IoT networks.

## 3. Dataset Used

The open-source dataset used in the project was obtained from Kaggle [21]. The dataset contains detailed transactional data from the Ethereum network. Transactions are divided into phishing or safe attempts. The features involved in the dataset include average minutes between sent and received transactions, contracts created, received and sent transactions over an address, minimum value and maximum value of sent and received transactions over an address, and Ethereum-related details such as total Ethereum sent, unique Ethereum tokens, and Ethereum contracts created.

The dataset consists of 10,000 datapoints, among which 829 empty datapoints were missing in the same transaction. These datapoints were removed during preprocessing. The number of phishing attempts in the dataset amounted to 1350, while there were 7662 safe attempts. Hence, the dataset was imbalanced. A representation of the distribution of outputs over the dataset is shown. In particular, Figure 1 shows the output distribution and concerning indices in the database.

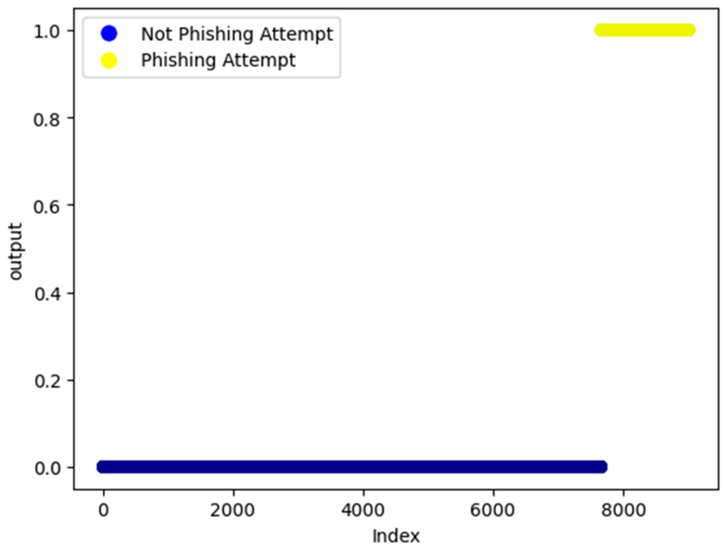

**Figure 1.** Distribution of outputs concerning index in the database.

Many of the features were co-dependent on each other, many other features either had the same value or a small number of outliers, and all other datapoints lay in the same range. All such parts were eliminated in the preprocessing stage. Some of those features are illustrated below: Figure 2 shows the datapoint distribution concerning ERC20 average time between sent TNX. From a different perspective, Figure 3 illustrates a single outlier with all other datapoints having the same value for feature: ERC20 total Ether received.

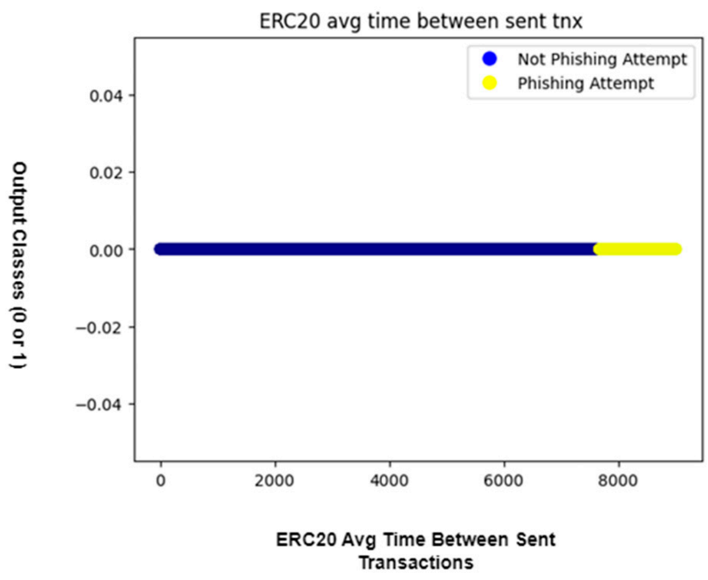

**Figure 2.** Illustrating all datapoints, which had the same value for feature: ERC20 average time between sent TNX.

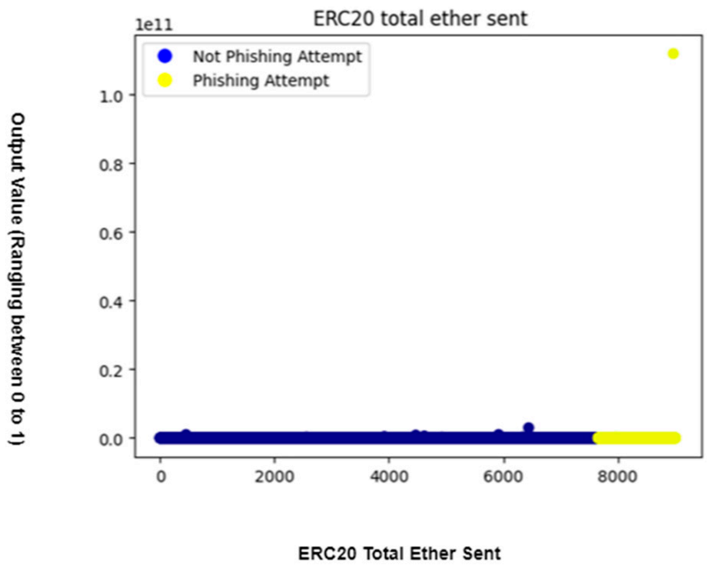

**Figure 3.** Illustrating a single outlier while all other datapoints have the same value for feature: ERC20 total Ether received.

## 4. Procedure of Experiment

Before predicting the phishing attempts in a blockchain environment, a machine-learning project was carried out in a similar domain, phishing attempt prediction on websites. The observations made from the project were later on used in this project. The observations included:

1. Best accuracy in random forest and XGBoost algorithms
2. The best accuracy was achieved when the total number of features were used in the model: 3N/5, where N is the total number of features in the dataset.

### 4.1. Data Preprocessing

The data-preprocessing step involved two phases (Figure 4): the removal of null datapoints and the removal of unwanted features. The size of the dataset was reduced from 54 columns to 30 columns by removing columns that had unique categorical data and features having the same values for all datapoints or several outliers. All features, which did not contribute anything to the actual model, were removed.

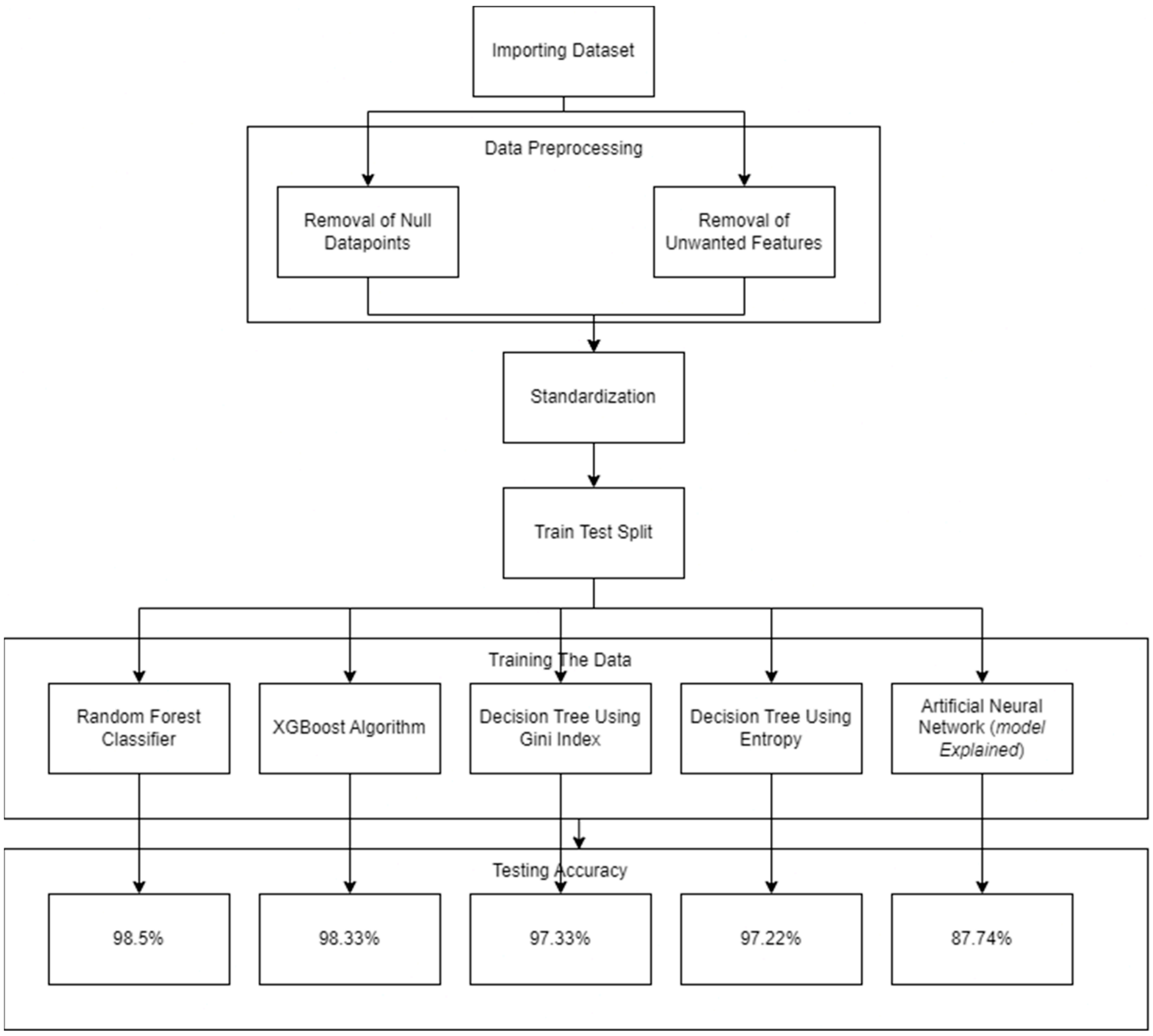

**Figure 4.** A flowchart explaining the flow of the procedure.

There were 829 null datapoints belonging to a mixed category of outputs. They all had to be removed for the smooth flow of the model. They could not be filled by mean or binning methods, as it would have made them outliers and the model would have ignored them.

### *4.2. Standardization*

The standardization was performed by using a standard scaler on the training dataset. The formula of standardization is as follows:

$$V_c - M_c / SD_c \tag{1}$$

where *Vc* represents the value in the column, *Mc* represents the mean of the column, and $SD_c$ represents the standard deviation of the column. The standardization used was column standardization.

### *4.3. Train/Test Split*

This step involved splitting the dataset into training and testing datasets. The training dataset is the dataset fed to the model to train it based on the input values. The testing dataset is the dataset used to check the accuracy of the model in predicting correct output values. We did not utilize a cross-validation dataset because as no iterations were involved in the model. The dataset was divided into 80% training and 20% test data.

### *4.4. Models for Classification*

The project focused on four different models of classification, namely the random forest classifier, XGBoost classifier, decision tree using Gini index, and decision tree using entropy. As graphically depicted in Figure 4, Random Forest noticeably provided the best accuracy rate of 98.5%, while the XGBoost classifier scored second with 98.33% accuracy rate. The used models were based on their implementations in the sklearn library in Python.

## **5. Models Used in Classification**

### *5.1. Decision Tree Classifier Using Gini Index*

A decision tree classifier is a tree-based classifier which divides the data based on the features, providing the best classification at a particular step. The best feature, which classifies the dataset, is chosen using a mathematical function. The Gini index is one such mathematical function. The leaf nodes of this tree give us classified examples. The error is calculated according to the number of wrong samples classified. Features with a lower Gini index [22] are selected as described in Equation (2).

The Gini index is calculated as:

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2 \tag{2}$$

where $D$ is the tuple in consideration, $p_i$ is the probability of tuple belonging to class $i$, and $m$ is the total number of classes.

After pre-processing, our decision tree using the Gini index gave us 97.33% accuracy, which is quite high, but we obtained better results with other options.

### *5.2. Decision Tree Classifier Using Entropy*

Entropy is another mathematical function that is used in feature selection for classification in decision trees. Entropy is a more complex mathematical notation than the Gini index [22] and provides an information gain. Thus, the features with higher information gain are selected. It makes use of the logarithmic function to calculate the predictability of a certain features.

The entropy is calculated as:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2 p_i \tag{3}$$

where $D$ is the tuple in consideration, $p_i$ is the probability of tuple belonging to class $i$, and $m$ is the total number of classes.

After pre-processing, our decision tree using entropy gave us 97.22% accuracy, which was slightly lower than the decision tree using the Gini index. It classified one sample as a false negative.

### *5.3. Random Forest Classifier*

The random forest classifier is a more complex version of decision tree classifiers. It is based on the bagging method, wherein random subsets of features and examples are selected in a predefined format, and they are made into small decision trees. It accurately measures the importance each feature gives to the decision trees and creates a final decision tree using all the subset decision trees. The final decision tree [23] has better accuracy than the subset decision trees.

The random forest classifier in our case gave us the best accuracy of 98.5% and can be implemented in real-life scenarios as well.

### *5.4. XGBoost Classification*

XGBoost classification uses the boosting method along with the bagging method used in random forest classification. The boosting method gives weights to each subset decision tree branch and adjusts them accordingly to reduce the error faced. It first develops a decision tree using a subset; it then boosts the decision tree by reducing its error and changing features and data variables in the subset decision tree. It does so in a predefined manner until it reaches an error rate that is less than the defined threshold. Using the XGBoost algorithm [24] might increase the bias in the model; however, it reduces the variance rate.

Our XGBoost classifier provided almost the same accuracy as the random forest classifier with accuracy rate of 98.33%. However, XGBoost can run in real time with more efficiency because it returns a lower false negative rate.

### *5.5. Artificial Neural Network*

An artificial neural network (ANN) [25] is a model based on human neurons and acts accordingly. It is a mixture of interconnected nodes and layers, with the first layer being the input layer and the last being the output layer. There are hidden layers [26] in between, and each layer has a set of nodes. The input layer has the same number of nodes as the input features, while the output layer has the same number of nodes as the output variables. The number of hidden layers and nodes in them is to be decided by the model creator to avoid underfitting and overfitting. The activation function of a layer takes input and converts it to output as an input for the next layer. Each branch or connection between nodes has a weight, which plays an important role in deciding the output.

We used an ANN with 28 input nodes and two hidden layers; the first has 11 nodes, and the second has six and one output node. Each layer used the sigmoid as its activation function. From among the three different configurations tried by us, this configuration gave the best output, with 88.8% accuracy.

The activation function used in the hidden and the output layers is the sigmoid function, which is mathematically represented as:

$$F(x) = \begin{cases} \mathbf{1}, & if\ (1+e^{-x}) - 1 \ \geq \ 0.5 \\ \mathbf{0}, & if\ (1+e^{-x}) - 1 \ \leq \ 0.5 \end{cases} \tag{4}$$

where $x$ is the input to the node in the layer.

The artificial neural network used here is very limited and underperforms compared to other models. As a result, it was not explored further as a possible comparison method; rather, random forest classifier and XGBoost algorithms were preferred. More research can be carried out on ANNs to reduce the algorithm's time complexity and return better accuracy on similar fields.

## 6. Observations

It was observed that the random forest classifier presented the best accuracy at 30 features from all the models tested on the dataset. It provided us with 98.5% accuracy, and the XGBoost classifier provided 98.33% accuracy as it can be noticed in Figure 4. These two models are the most accurate machine-learning implementations and can be used further on for real-time applications in Ethereum networks as well. Figures 5 and 6 represent the confusion matrix for random forest and XGBoost. Therefore, considerations are drawn on false positives, false negatives, true positives and true negatives.

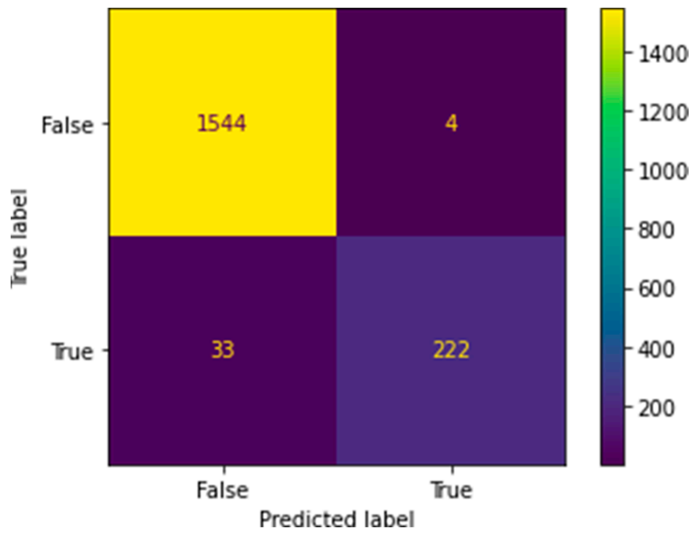

**Figure 5.** Confusion matrix for random forest classifier.

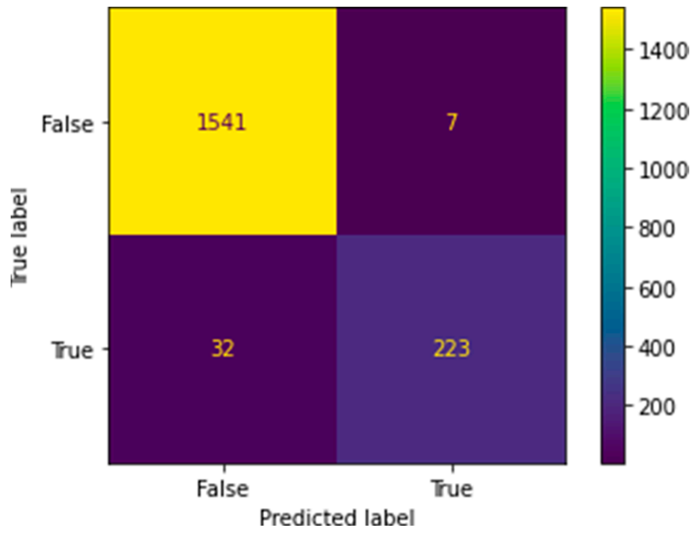

**Figure 6.** Confusion matrix for XGBoost classifier.

The major concern in the output is the notion of false negatives. False negatives refer to the values which were phishing attempts but were classified as safe attempts. This can be dangerous as malicious attackers can use this kind of feature manipulation to their advantage and still be able to successfully carry out a phishing attempt. The case of false negatives (Figures 5 and 6) is slightly higher in the random forest classifier than in the XGBoost classifier; however, the XGBoost classifier has higher cases of false positives, which is still a neglectable outcome.

With these observations, we can successfully answer the research question: How accurately can machine learning predict phishing attempts in the blockchain environment, and which model is most suitable for this purpose?

Although the following observation does answer the question of which machine-learning model will accurately predict phishing attempts, it does not mean it should be applied on a real-time basis. A performance analysis of both of the shortlisted models is shown below:

$$T_{RFC} = T_{building-trees} + T_{training-trees} \tag{5}$$

$$T_{RFC} = O(E \times M \times N \times \log(N)) + O(M \times N \times \log(N)) \tag{6}$$

In Equation (5), time complexity of a random forest classifier is divided into two main subprocesses: building trees and training trees. The building trees process takes $O(E \times M \times N \times \log(N))$ time complexity where $E$ is the number of estimators, $M$ is the number of features, and $N$ is the number of datapoints, while the time complexity for training trees is $O(M \times N \times \log(N))$. Hence, the time complexity of random forest classification was considered as $O((E + 1)(M \times N \times \log(N))$.

The performance analysis of the XGBoost algorithm is as follows:

$$T_{XGB} = T_{training-trees} \tag{7}$$

The XGBoost algorithm usually trains trees over iterations. By considering it only to edit a single tree over all its iterations and using the log loss method of loss calculation, the time complexity accounts for $O(E \times M \times N)$, where $E$ is the number of estimators, $N$ is the number of datapoints, and $M$ is number of features extracted.

From observations, it can be said that the XGBoost algorithm holds better implementation prospects and quicker results than the random forest classifier. The difference in accuracy between the two is only 0.17%, with XGBoost outperforming the random forest classifier on the number of false negatives. Due to its lower time complexity, the XGBoost algorithm is preferred to implement the system of phishing detection.

## 7. Conclusions

From the given observations, it is definitely clear that random forest classification and the XGBoost classifier show higher accuracy values than the rest of the machine-learning models used in classifying phishing attempts as either harmful or safe. Random forest classification obtained a 98.5% of accuracy rate, while XGBoost achieved an accuracy percentage of 98.33, with the difference between the two being only 0.17%. But this small number becomes significantly larger in real-time scenarios, involving attempts at phishing where the range is usually greater than a million attempts per hour.

Still, XGBoost classification should be preferred in real-time scenarios for two main reasons: It features a lower number of false negatives and executes faster than random forest classification. XGBoost classification shows a 1.77% rate of false negatives, while random forest classification scores a 1.83% rate of false negatives on the given dataset. Although the difference seems small, in real time it is a huge deciding factor because false positives may be accepted in the model, and false negatives must be filtered out.

Also, with the number of estimators as 200, datapoints as 10,000, and extracted features as 15, random forest classification provides output in 3.01 s, while XGBoost classification outputs the same result in 2.03 s. This is a huge time gap in performance between the two models which needs to be selected for real-time analysis. Hence, it can be concluded that XGBoost classification is the best model for real-time analysis and prediction of whether an attempt in the blockchain is a phishing attempt or not.

## 8. Limitations and Future Work

This research has had several limitations regarding important aspects such as cross-validation and thorough comparison. Some possibilities for future work are as follows:

1. Using a cross-validation dataset along with a training and testing dataset could improve the testing accuracy of the models.
2. Hyperparameter tuning using the cross-validation dataset to choose the best hyperparameters and higher accuracies.
3. Using newer and more advanced machine-learning algorithms and testing unsupervised deep-learning models on the dataset such as graph neural networks to improve accuracy.
4. More configurations of hidden layers and nodes and activation functions can be experimented with to achieve higher accuracy.

**Author Contributions:** K.J.: data curation, investigation, writing—original draft, C.B.: methodology, project administration, writing—review and editing, K.S.: data curation, investigation, software, D.P.: data curation, investigation, software, J.M.C.: methodology, writing—review and editing, A.B.: methodology, P.L.M.: writing—review and editing. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Andryukhin, A.A. Phishing attacks and preventions in blockchain based projects. In Proceedings of the 2019 International Conference on Engineering Technologies and Computer Science (EnT), Moscow, Russia, 26–27 March 2019; IEEE: Piscataway, NJ, USA; pp. 15–19.
2. Wen, H.; Fang, J.; Wu, J.; Zheng, Z. Transaction-Based Hidden Strategies against General Phishing Detection Framework on Ethereum. In Proceedings of the 2021 IEEE International Symposium on Circuits and Systems (ISCAS), Daegu, Republic of Korea, 22–28 May 2021; pp. 1–5. [CrossRef]
3. Oyinloye, D.P.; Teh, J.S.; Jamil, N.; Alawida, M. Blockchain consensus: An overview of alternative protocols. *Symmetry* **2021**, *13*, 1363. [CrossRef]
4. Alevizos, L.; Ta, V.T.; Hashem Eiza, M. Augmenting zero trust architecture to endpoints using blockchain: A state-of-the-art review. *Secur. Priv.* **2022**, *5*, e191. [CrossRef]
5. Ye, C.; Li, G.; Cai, H.; Gu, Y.; Fukuda, A. Analysis of security in blockchain: Case study in 51%-attack detecting. In Proceedings of the 2018 5th International Conference on Dependable Systems and Their Applications (DSA), Dalian, China, 22–23 September 2018; IEEE: Piscataway, NJ, USA; pp. 15–24.
6. Zhang, R.; Xue, R.; Liu, L. Security and Privacy on Blockchain. *ACM Comput. Surv.* **2019**, *52*, 1–34. [CrossRef]
7. Aggarwal, S.; Kumar, N. Attacks on blockchain. In *Advances in Computers*; Elsevier: Amsterdam, The Netherlands, 2021; Volume 121, pp. 399–410.
8. Phillips, R.; Wilder, H. Tracing Cryptocurrency Scams: Clustering Replicated Advance-Fee and Phishing Websites. In Proceedings of the 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Toronto, ON, Canada, 3–6 May 2020; IEEE: Piscataway, NJ, USA; pp. 15–24. [CrossRef]
9. Holub, A.; O'Connor, J. COINHOARDER: Tracking a ukrainian bitcoin phishing ring DNS style. In Proceedings of the 2018 APWG Symposium on Electronic Crime Research (eCrime), San Diego, CA, USA, 15–17 May 2018; IEEE: Piscataway, NJ, USA; pp. 1–5.
10. Fu, B.; Yu, X.; Feng, T. CT-GCN: A phishing identification model for blockchain cryptocurrency transactions. *Int. J. Inf. Secur.* **2022**, *21*, 1223–1232. [CrossRef]
11. Yuan, Z.; Yuan, Q.; Wu, J. Phishing Detection on Ethereum via Learning Representation of Transaction Subgraphs. In *International Conference on Blockchain and Trustworthy Systems*; Springer: Singapore, 2020; pp. 178–191.
12. Zhang, D.; Chen, J.; Lu, X. Blockchain Phishing Scam Detection via Multi-channel Graph Classification. In *International Conference on Blockchain and Trustworthy Systems*; Springer: Singapore, 2021; pp. 241–256.
13. Bhowmik, M.; Chandana, T.S.S.; Rudra, B. Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain. In Proceedings of the 2021 5th International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 8–10 April 2021; IEEE: Piscataway, NJ, USA; pp. 539–541.
14. Arshey, M.; Viji, K.A. Thwarting cyber crime and phishing attacks with machine learning: A study. In Proceedings of the 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 19–20 March 2021; IEEE: Piscataway, NJ, USA; Volume 1, pp. 353–357.
15. Catal, C.; Giray, G.; Tekinerdogan, B.; Kumar, S.; Shukla, S. Applications of deep learning for phishing detection: A systematic literature review. *Knowl. Inf. Syst.* **2022**, *64*, 1457–1500. [CrossRef] [PubMed]
16. Parra, G.D.L.T.; Rad, P.; Choo, K.-K.R.; Beebe, N. Detecting Internet of Things attacks using distributed deep learning. *J. Netw. Comput. Appl.* **2020**, *163*, 102662. [CrossRef]
17. Kim, J.; Lee, S.; Kim, Y.; Cho, S. A Graph Embedding-based Identity Inference Attack on Blockchain Systems. In Proceedings of the 2022 International Conference on Electronics, Information, and Communication (ICEIC), Jeju, Republic of Korea, 6–9 February 2022; pp. 1–3. [CrossRef]
18. Ali, A.; Pasha, M.F.; Guerrieri, A.; Guzzo, A.; Sun, X.; Saeed, A.; Hussain, A.; Fortino, G. A Novel Homomorphic Encryption and Consortium Blockchain-based Hybrid Deep Learning Model for Industrial Internet of Medical Things. *IEEE Trans. Netw. Sci. Eng.* **2023**, 1–18. [CrossRef]
19. Roy, K.S.; Karim, E.; Udas, P.B. Exploiting Deep Learning Based Classification Model for Detecting Fraudulent Schemes over Ethereum Blockchain. In Proceedings of the 2022 4th International Conference on Sustainable Technologies for Industry 4.0 (STI), Dhaka, Bangladesh, 17–18 December 2022; pp. 1–6. [CrossRef]
20. Yazdinejad, A.; Dehghantanha, A.; Parizi, R.M.; Srivastava, G.; Karimipour, H. Secure Intelligent Fuzzy Blockchain Framework: Effective Threat Detection in IoT Networks. *Comput. Ind.* **2023**, *144*, 103801. [CrossRef]

21. Available online: https://www.kaggle.com/datasets/xblock/ethereum-phishing-transaction-network (accessed on 20 March 2023).
22. Kotsiantis, S.B. Decision trees: A recent overview. *Artif. Intell. Rev.* **2011**, *39*, 261–283. [CrossRef]
23. Devetyarov, D.; Nouretdinov, I. Prediction with confidence based on a random forest classifier. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 37–44.
24. Chen, Z.; Jiang, F.; Cheng, Y.; Gu, X.; Liu, W.; Peng, J. XGBoost classifier for DDoS attack detection and analysis in SDN-based cloud. In Proceedings of the 2018 IEEE International Conference on Big Data and Smart Computing (BigComp), Shanghai, China, 17–18 January 2018; IEEE: Piscataway, NJ, USA; pp. 251–256.
25. Krogh, A. What are artificial neural networks? *Nat. Biotechnol.* **2008**, *26*, 195–197. [CrossRef] [PubMed]
26. Stathakis, D. How many hidden layers and nodes? *Int. J. Remote Sens.* **2009**, *30*, 2133–2147. [CrossRef]