Machine Learning Model to Identify Ransomware Payment within Bitcoin Transaction

Chang Kai Cheng, Rajat Goyal, Kwadwo Owusu, Monish Shah

Department of Computer Science, University of Nottingham, Nottingham, NG7 2RD
{hfykc9, ppxrg1, psxko2, psxms22}@nottingham.ac.uk

Abstract—**Ransomware attacks have emerged as a significant concern in the realm of cybercrime, with Bitcoin transactions frequently employed for ransom demands. Identifying and locating such transactions is critical for addressing this problem. In this paper, we propose the use of machine learning models to detect ransom-related Bitcoin transactions and categorise them by their location. The dataset spans from 2009 to 2018 and comprises over 2.9 million instances with 10 attributes.**

To handle this extensive dataset, we use big data technologies like Spark for parallel processing and faster outcomes. We assess the efficacy of various machine learning models, including logistic regression, an ensemble of logistic regression, a decision tree, and a random forest, for detecting ransomware-associated transactions. With the use of bagging, we combine the results of multiple models and evaluate their performance using metrics such as recall, precision, and the macro F1 score.

Our research highlights the value of incorporating big data technologies and multiple machine learning models in detecting ransom-related transactions. The decision tree model outperformed others in terms of accuracy and macro F1 score, demonstrating its effectiveness in differentiating between ransom-related and genuine transactions. This study offers insights into the efficiency of different machine learning models for identifying ransom-associated Bitcoin transactions and their locations, emphasising the advantages of utilising big data technologies in this context.

I. Introduction

Blockchain is a distributed public ledger used for recording transactions between two parties. It consists of a chain of blocks, each containing permanent transaction records. Since it's a public ledger, it's transparent and open for public auditing. To ensure security, cryptography methods are used to encrypt transactions, allowing only the parties involved to decrypt and view them [1]. Blockchain technology was invented in 2008, with Bitcoin being one of the first applications to utilise it [2]. The use of Bitcoin has grown rapidly in recent years due to its ability to facilitate global transactions without exchange fees and without being controlled by any organisation or government.

Bitcoin enables peer-to-peer transactions without a central authority. Transactions are anonymous, and no identity verification is needed to use Bitcoin. With just a public address, senders can easily complete transactions. The global and pseudonymous nature of Bitcoin transactions has been exploited by cybercriminals to commit cybercrimes, particularly ransomware attacks.

Ransomware, malicious software that locks files on a victim's device, has significantly increased in the realm of cybercrime by taking advantage of Bitcoin transactions [3], [4]. Attackers send ransomware to individuals, businesses, or governments to encrypt their files and demand payment to a Bitcoin address to obtain the decryption key. The impact of ransomware extends beyond victims, affecting the economy [6] and society [7]. Therefore, it is crucial to detect ransom payments among Bitcoin transactions to prevent attackers from continuing to spread crypto-ransomware.

This paper aims to implement a machine-learning solution with a big data approach to identify ransom-related Bitcoin transactions and their originating cities based on the transaction graph. The contributions of this project include developing a novel approach for detecting ransom payments within the vast network of Bitcoin transactions and potentially aiding law enforcement agencies in combating cybercrime.

II. Related Work

Ransomware attacks have become an increasingly significant issue in cybersecurity. The emergence of cryptocurrencies, particularly Bitcoin, has facilitated extortion by enabling anonymous transactions. To effectively combat this threat, it is critical to understand the underlying payment patterns and develop robust detection mechanisms.

Several studies have focused on analysing Bitcoin transactions to identify ransomware patterns. Ron and Shamir [12] were among the first to examine the Bitcoin transaction graph, highlighting the flow of funds from victims to attackers. Furtak and Kharraz [15] further explored the distinct characteristics of ransom payments and their movement within the Bitcoin network,

uncovering valuable information about attacker behaviour.

Machine learning techniques have proven useful in detecting illicit transactions related to cybercrime. Huang et al. [17] investigated shared patterns and heuristics in hacker behaviour, which informed the development of algorithms for detecting ransom payments. Talabani and Abdulhadi [21] presented a rule-based approach for identifying ransomware transactions, while Badawi and Al-Haija [20] employed machine learning classifiers to detect money laundering activities within the Bitcoin ecosystem.

This section aims to link our submission to existing knowledge by providing a comprehensive overview of the literature on ransomware payments and their detection. By building upon the findings of these studies, our work contributes to the ongoing refinement and enhancement of methods for identifying and mitigating ransomware threats.

III. Datasets

The dataset used in this study is obtained from UCI and comprises multivariate, time-series data of Bitcoin transaction graphs from 2009 to 2018. With over 2.9 million instances and 10 attributes, it encompasses a range of integer and real characteristics. The dataset was donated on June 17, 2020, and has been employed for classification and clustering tasks.

The rapid growth and increasing complexity of ransomware attacks have necessitated the use of big data techniques to better understand and mitigate these threats. By employing a comprehensive dataset of Bitcoin transactions, this study aims to leverage the power of big data analytics to identify patterns and trends associated with ransomware activities.

The rationale behind using big data techniques in this context includes the ability to process and analyse large amounts of information quickly and accurately, as well as the opportunity to uncover hidden patterns and trends that may be difficult to detect using traditional analysis methods. By employing big data analytics, we can better discern the unique characteristics of ransomware transactions and develop more effective strategies for combating these malicious activities.

Each transaction in the dataset is labelled as either a genuine transaction or as a payment associated with one of 27 different types of ransomware. For the purposes of this study, ransomware labels were modified to indicate only the city of the attack. Consequently, our predictions focus on determining whether a given address corresponds to a genuine transaction or a ransom payment in Princeton, Padua, or Montreal.

IV. Methodology

The methodology used in this paper follows the flow depicted in *Figure 5*. The various steps of the methodology are explained below:

A. Outlier Removal

As shown in Figure 1, the dataset contains outliers. Outliers in the dataset can significantly affect the accuracy and robustness of the predictions.

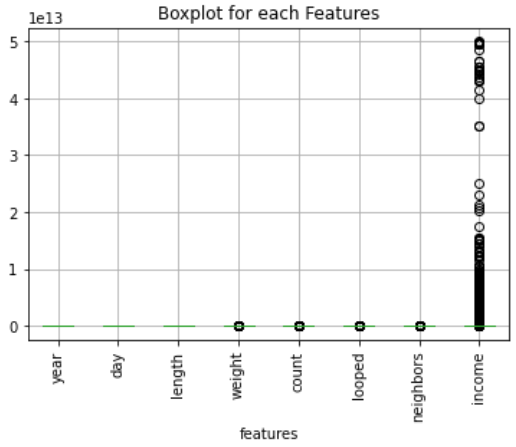

Fig.1: Boxplot of each feature

The Z-score standardised the dataset by centring the mean and removing samples that were n steps ahead or behind. We set n to 3, which means that samples 3 steps away from the mean are removed. The use of the Z-score technique ensures that the dataset used for the analysis was accurate and reliable.

B. Train-Test Split

A train-test split is performed to prevent information leakage when applying oversampling techniques during the training phase.

C. Label Encoding

There are four types of ransomware families that we are interested in, each labelled by its family name. We performed label encoding to encode the label with 0 for Princeton, 1 for Padua, 2 for Montreal and 3 for White. This technique helps enable the analysis of the impact of ransomware family names on the prediction of ransom payments.

D. Handling Imbalanced Data

The dataset contains 2,916,697 samples, with 98.6% labelled as "White" and only 1.4% associated with ransomware (see *Table 1*). This imbalance causes the models to be biased towards genuine transactions and results in poor ransom payment predictions.

Label	Count	Percentage (%)
White	2875284	98.6
Princeton	15848	0.5
Padua	12402	0.4
Montreal	13163	0.5

Table 1: Imbalance Data

To mitigate the impact of imbalanced data, we employed the SMOTE-ENN technique, which performs both oversampling and undersampling. SMOTE generates synthetic samples for the minority class by interpolating between existing minority class samples (see *Figure 2*).

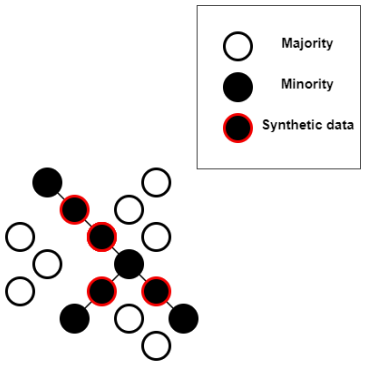

Fig. 2: SMOTE

The Edited Nearest Neighbour (ENN) technique is an undersampling method that removes misclassified samples from the majority class based on the k-nearest neighbour classifier. In this project, we used Smote to generate synthetic data for the minority class to match the number of samples in the majority class and then applied ENN to remove noisy samples from the data. This approach, known as SMOTE-ENN, was only applied to the training subset to prevent data leakage in the test set.

E. Normalising Data

Normalising the features scales their values into a specific range, which enhances model stability, especially for features with large value ranges such as "income". This process ensures that all features are on the same scale and have the same magnitude. We used a standard scaler to normalise the features, which scales them to unit variance [22].

F. Logistic Regression (LR)

LR is based on the concept of linear regression [23]. It uses a sigmoid function to map the output of linear regression to a value between 0 and 1, which represents the probability of belonging to a particular class. Specifically, the output of LR is either 1 or 0, indicating the likelihood of belonging to one class. LR was chosen for this study because of its simplicity and interpretability, it helped us analyse the impact of various features on the prediction of ransom payments.

G. Decision Tree (DT)

Decision Trees learn to form a tree representation of the data, with leaf nodes representing outcomes and decision nodes containing conditions for data splitting. The Attribute Selection Method (ASM) is an intrinsic feature selection technique in DTs that helps determine optimal splitting conditions.

H. Ensemble Method

Both logistic regression and decision tree were implemented into their own ensemble method using the bagging approach. Bagging is a popular ensemble learning technique that trains multiple classifiers with different random subsets of data and employs majority voting to obtain the final prediction.

Ensemble learning helps to reduce overfitting by combining the predictions of multiple models, each trained on a different subset of the data.

The bagging approach is particularly effective because it randomly selects a subset of data to train each model, allowing it to learn from a more diverse set of samples. The exposure to a more diverse set of samples helps to reduce variance in the model predictions and thus makes a more robust and accurate prediction. *Figure 3* illustrates the idea of bagging a subset of samples.

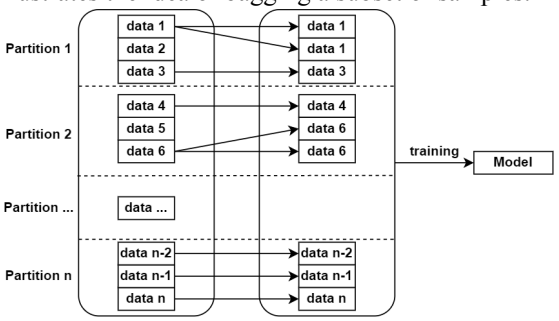

Fig. 3: Bagging

As illustrated in Figure 4, the data selected for training is done with replacement, which means that some points may appear more than once in the training subsets. However, since the size of the resampled data is the same as the original dataset, some data may not be selected during the resampling process.

The ensemble approach for the decision tree model is called the random forest (RF) model, which is available in MLlib [25]. The random forest model works by constructing multiple decision trees on different subsets of the training

data and then combining their predictions through majority voting.

For ensemble logistic regression (ELR), the pseudocode for the ensemble approach is shown in *Figure 4*. The pseudocode outlines the steps involved in training multiple logistic regression models on different subsets of the training data and then combining their predictions to obtain the final prediction.

Algorithm 1 Bagging Method

```
models ← emptylist
for i less than n do                          ▷ n is the number of model
    bag ← test_data.sample(True, 1.0)
    model.PredictionCol ← prediction_i
    model.ProbabilityCol ← prob_i
    model.RawPredictionCol ← raw_pred_i
    Append the fitted model to models
end for
```

Fig.4 Bagging Algorithm for Logistic Regression

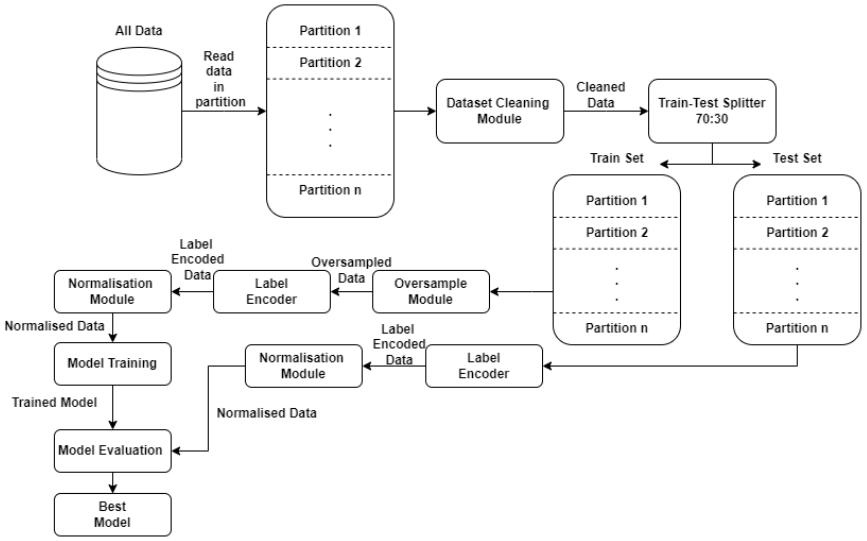

Fig. 5: General Flow

V. Experimental Set-up

A. Data Pre-processing

To prepare the Bitcoin transaction dataset for analysis, several preprocessing steps were performed. First, the dataset was loaded using the spark-CSV library. The ransomware labels were then clustered by location, reducing the number of distinct labels and improving the interpretability of the models. Outliers were removed using z-mean by computing the means and standard deviations of each column and filtering out any rows where the absolute deviation from the mean was greater than 3 times the standard deviation. The input features were then scaled using a StandardScaler. The data was split into training and testing sets using a 70/30 split, and the training set was balanced using SMOTE-ENN to address the data imbalance.

B. Hyperparameter Tuning

To optimise the performance of the models, hyperparameters were tuned using 5-fold cross-validation. For the Logistic Regression model, a LogisticRegression object was created with elasticNetParam and regParam hyperparameters. The ElasticNetParam hyperparameter grid was set to [0.0, 0.6, 1.0], and the RegParam hyperparameter grid was set to [0.01, 0.1, 1.0]. For the Random Forest model, a RandomForestClassifier object was created with numTrees and maxDepth hyperparameters. The numTrees hyperparameter grid was set to [15, 17, 19], and the maxDepth hyperparameter grid was set to [12, 15, 17]. For the Decision Tree model, a DecisionTreeClassifier object was created with maxDepth and maxBins hyperparameters. The maxDepth hyperparameter grid was set to an array range from 1 to 30, and the maxBins hyperparameter grid was set to [32, 64, 128]. The hyperparameters were tuned on k-fold cross-validation with k=5 and evaluator=MulticlassClassificationEvaluator(metricName="f1").

C. Evaluation metric

Relying solely on accuracy for model evaluation can be insufficient, as it might not show a model's true performance. To address this, the F1-score, which considers precision and recall, is used alongside accuracy. This enables a comprehensive assessment of the model's ability to predict positive and negative cases accurately.

For multiclass problems, the macro F1-score, which averages the F1-score across all labels, is appropriate. A high macro F1-score indicates the model performs well on all labels. In summary, combining macro F1-score and accuracy provides a robust evaluation of the model's performance.

V. Results and Discussion

Below are the results obtained from the experiment.

Label	Precision	Recall	F1 Score
White	1.00	0.12	0.22
Princeton	0.02	0.99	0.03
Montreal	0.01	0.53	0.02
Padua	0.02	0.77	0.02

Table. 2: Precision, Recall and F1-score for LR

Label	Precision	Recall	F1 Score
White	0.99	0.96	0.98
Princeton	0.28	0.77	0.41
Montreal	0.12	0.42	0.19
Padua	0.16	0.60	0.25

Table. 3: Precision, Recall and F1-score for DT

Label	Precision	Recall	F1 Score
White	1.00	0.17	0.28
Princeton	0.02	0.98	0.03
Montreal	0.01	0.53	0.02
Padua	0.01	0.75	0.03

Table. 4: Precision, Recall and F1-score for ELR

Label	Precision	Recall	F1 Score
White	1.00	0.88	0.94
Princeton	0.10	0.91	0.17
Montreal	0.07	0.63	0.12
Padua	0.11	0.90	0.19

Table. 5: Precision, Recall and F1-score for RF

Models	Accuracy (%)	Macro F1-score
Logistic Regression (LR)	13.31	0.07
Decision Tree (DT)	**95.65**	**0.46**
Ensemble Logistic Regression (ELR)	13.36	0.09
Random Forest (RF)	88.10	0.36

Table. 6: Accuracy and F1-score of each model

*A. Performance Comparison of Model*s

Based on *Table 6*, the Decision Tree (DT) model outperforms other models in terms of accuracy, recall and macro F1 score. The performance of the Random Forest (RF) model matches that of DT, which is expected since both are tree-based models. In contrast, the Logistic Regression (LR) and Extended Logistic Regression (ELR) models exhibit similar performance, but both perform poorly. Interestingly, the ensemble methods do not significantly outperform the single models. This suggests that complex models were not required for this problem, as the input features demonstrated a simpler relationship. As a result, simpler models were able to learn the pattern and make accurate predictions on the test dataset while saving computational costs and time.

B. Poor Performance of LR and ELR Models

The LR model assumes a linear relationship between variables. If this assumption is not met, the model cannot fit well, resulting in poor performance. Thus, the non-linear relationship present in this dataset may be the primary cause of the poor performance. Since ELR uses LR as its basis, it also suffers from poor performance.

Another possible reason for the poor performance of LR and ELR models is the imbalanced dataset. Although the Smote-Enn technique is applied, the Edited Nearest Neighbors (Enn) process removes many white samples due to their lack of clustering, making the white label the minority. *Table 2* and *Table 4* show the F1 score, recall, and precision of LR and ELR. Both models have high precision but poor recall on the white label, leading to a low F1 score. This indicates that the models struggle to predict genuine transactions. Moreover, the results also suggest that these two approaches cannot accurately identify the location of ransomware attacks, as the F1 scores for other classes are very low. Overall, the logistic regression models fail to learn the pattern of the dataset effectively.

C. Advantages of Tree-Based Models

Tree-based methods, such as DT and RF, have the advantage of learning non-linear relationships between variables. The performance of both tree-based models in Table 6 supports the presence of a non-linear relationship between variables. One advantage of using RF instead of DT is its lower susceptibility to overfitting. However, in this case, DT slightly outperforms RF, likely because DT is not overfitted to the training data. Consequently, RF does not offer a significant improvement in performance.

Additionally, tree-based methods exhibit a slight advantage in handling imbalanced data after applying SMOTE-ENN. *Table 3* and *Table 5* present the results for DT and RF in terms of precision, recall, and F1 score for each label. Notably, both models achieve high F1 scores on the white label, indicating their ability to classify ransom-related transactions and genuine transactions. However, these models struggle to group ransom-related transactions by location, resulting in high recall but low precision for three labels. This suggests that there may not be significant differences in ransom-related payments across different cities, making it difficult for the model to identify ransom payments by location. Overall, the tree-based models can effectively differentiate between normal Bitcoin transactions and ransom-related transactions but struggle to predict the location of ransomware attacks.

D. Decision Tree Model Advantages

There are several benefits to using the DT model over the RF model. First, DT is a highly interpretable model that allows for the identification of important blockchain graph features in detecting ransom payments. Additionally, DT is a lightweight algorithm capable of making predictions quickly. Although data scaling was performed in this study, DT can also be trained on unscaled data, further reducing prediction time when implemented in the Bitcoin system. This improves the model's scalability, as the rapidly growing number of Bitcoin transactions requires quick decision-making. Consequently, the decision tree model emerges as the best-performing model based on our experiment.

VI. Conclusion

In this paper we compared 4 machine learning models in a big data approach for identifying ransom-related transactions. We designed the model to have the ability to detect ransom-related transactions and the location where the attack occurred using Spark as the big data technology. The result has proven that a decision tree had the ability to detect and differentiate between ransom-related transactions and genuine transactions and also has the ability to interpret and understand important features for such detection. However, the result has also proven that all the models are struggling in grouping all the ransom-related transactions into their cities. Other than that, this study also found that Smote-Enn can be used to solve the data imbalance issue, it can also lead to the removal of a significant number of legitimate samples, resulting in poor performance on genuine transactions especially on models that are sensitive to imbalance data. In future work, we consider separate for 2 stages one for classifying a ransom-related transaction and one for clustering the transactions in the cities as well as consideration of other techniques to address data imbalance such as undersampling.

Reference

[1] "Cryptography in BlockSchain," GeeksforGeeks, May 07, 2022. https://www.geeksforgeeks.org/cryptography-in-blockchain/

[2] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008. Available: https://assets.pubpub.org/d8wct41f/31611263538139.pdf

[3] H. Orman, "Evil Offspring - Ransomware and Crypto Technology," in IEEE Internet Computing, vol. 20, no. 5, pp. 89-94, Sept.-Oct. 2016, doi: 10.1109/MIC.2016.90.

[4] L. Y. Connolly and D. S. Wall, "The rise of crypto-ransomware in a changing cybercrime landscape: Taxonomising countermeasures," Computers & Security, vol. 87, p. 101568, Nov. 2019, doi: https://doi.org/10.1016/j.cose.2019.101568.

[5] A. Hansberry, A. Lasser, and A. Tarrh, "Cryptolocker: 2013's Most Malicious Malware". Available: https://www.cs.bu.edu/~goldbe/teaching/HW55815/cryptolockerEssay.pdf

[6] A. Zimba and M. Chishimba, "On the Economic Impact of Crypto-ransomware Attacks: The State of the Art on Enterprise Systems," *European Journal for Security Research*, vol. 4, no. 1, pp. 3–31, Jan. 2019, doi: https://doi.org/10.1007/s41125-019-00039-8.

[7] "Beyond the Bottom Line: The Societal Impact of Ransomware," *www.rusi.orghttps*. https://rusi.org/explore-our-research/publications/commentary/beyond-bottom-line-societal-impact-ransomware#:~:text=First%2C%20ransomware%20causes%20downstream%20harm

[8] G. Kontaxis, P. Mavridis, and E. Markatos, "Quantifying Bitcoin Heists," in 11th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, Heraklion,

Crete, Greece, Oct. 2018, pp. 161–183, doi: https://doi.org/10.1007/978-3-030-00470-5_8.
[9] Kharraz, A., Furtak, M., & Robertson, W. (2017). Paying for ransomware: How much is data worth?. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (pp. 1439-1453). ACM.
[10] "UCI Machine Learning Repository: BitcoinHeistRansomwareAddressDataset Data Set," archive.ics.uci.edu. https://archive.ics.uci.edu/ml/datasets/BitcoinHeistRansomwareAddressDataset
[11]Böhme, R., Christin, N., Edelman, B., & Moore, T. (2015). Bitcoin: Economics, technology, and governance. Journal of Economic Perspectives, 29(2), 213-238.
[12]Ron, D., & Shamir, A. (2013). Quantitative analysis of the full bitcoin transaction graph. In Proceedings of the 2013 International Conference on Financial Cryptography and Data Security (pp. 6-24). Springer.
[13]Moser, M. (2018). Ransomware: A review. International Journal of Advanced Computer Science and Applications, 9(4), 29-37.
[14]Jaiswal, S., & Saini, H. S. (2019). A survey on ransomware: Attack, defence and mitigation techniques. Journal of Network and Computer Applications, 139, 17-37.
[15]Furtak, M., & Kharraz, A. (2019). Ransomware payment tracking and analysis. In Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP) (pp. 1129-1145). IEEE.
[16]Richter, T., & Sandner, P. G. (2019). Identifying bitcoin users using transaction metadata. Journal of Banking & Finance, 98, 1-13.
[17]Goldfeder, S., Kalodner, H. A., & Reisman, D. (2018). When the cookie meets the blockchain: Privacy risks of web payments via cryptocurrencies. In Proceedings of the 2018 IEEE Symposium on Security and Privacy (SP) (pp. 135-151). IEEE.
[18]Conti, M., Kumar, E. R., Lal, C., Ruj, S., & Choo, K. K. R. (2018). A survey on security and privacy issues in bitcoin wallets. Journal of Computer and System Sciences, 84, 81-102.
[19]Yasin, M., & Moustafa, N. (2020). Bitcoin forensics: A tutorial with case studies. Digital Investigation, 32, 101-118.
[20] A. A. Badawi and Q. A. Al-Haija, "Detection of money laundering in bitcoin transactions," 4th Smart Cities Symposium (SCS 2021), Online Conference, Bahrain, 2021, pp. 458-464, doi: 10.1049/icp.2022.0387.
[21] H. S. Talabani and H. M. Abdulhadi, "Bitcoin Ransomware Detection Employing Rule-Based Algorithms", SJUOZ, vol. 10, no. 1, pp. 5–10, Jan. 2022.
[22] "StandardScaler — PySpark 3.4.0 documentation," spark.apache.org. https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.feature.StandardScaler.html
[23] D. Opitz and R. Maclin, "Popular Ensemble Methods: An Empirical Study," Journal of Artificial Intelligence Research, vol. 11, pp. 169–198, Aug. 1999, doi: https://doi.org/10.1613/jair.614.
[24] IBM, "About Linear Regression | IBM," www.ibm.com, 2022. https://www.ibm.com/topics/linear-regression#:~:text=Resources-
[25] "Classification and regression - Spark 2.4.5 Documentation," spark.apache.org. https://spark.apache.org/docs/latest/ml-classification-regression.html#random-forest-classifier