Detecting Fradulent Ethereum Addresses: A Data Mining Approach

Anahita Hedayatifard
Professor: Dr. Arta Jamshidi

Department of Mathematics,
College of Sciences,
University of Tehran

February 2024

1 Abstract

In this project, the objective is to employ the K-Nearest Neighbors (KNN) algorithm to classify Ethereum addresses into fraudulent or non-fraudulent categories. Beginning with a concise overview of Blockchain fundamentals and cryptocurrencies, common types of fraud within these realms are explored. Subsequently, the report outlines the detailed steps involved in data mining and evaluates the KNN model on this data set.

2 Introduction

In 2009, Satoshi Nakamoto came up with a protocol enabling value transfer among non-trusting individuals: Bitcoin. [5] A peer-to-peer network enabling financial exchanges between users, based on economic incentives rather than trusted authorities. It operates on the blockchain: a public ledger keeping track of all the transactions in an immutable way using novel ideas such as Nakamoto consensus using proof of work. While all transactions are recorded on the blockchain and retrievable by any participant, Bitcoin still ensures the anonymity of its users who identify themselves solely with public keys, signing transactions with associated private keys. Benefiting from the anonymity, criminals quickly took advantage of its popularity to rip off other users by creating multiple scams. An empirical analysis of Bitcoin-based scams in 2015 showed that at least $11 million was stolen from 13,000 victims [6]. Unlike scams before Bitcoin, cryptocurrencies are not yet subject to any government regulations, making it difficult to seek recompense for damages caused by fraud. Due to the immutability of the blockchain and the anonymity of users, it is nearly impossible to revert a fraudulent transaction. The absence of regulation of the cryptocurrency ecosystem and the lack of transparency of the transactions may lead to an increased number of fraudulent cases.

Some of these fraudulent behaviors include smart-ponzi schemes, phishing, money laundering, fraud and crime related activity. The internal security mechanisms defined by blockchain technology (hash functions) are able to solve issues such as the double spending problem, but no tools are in place to detect or flag any suspicious behavior or transactions occurring over the network automatically.[3] This is where the utilization of data mining techniques may prove advantageous.

3 Data Preprocessing and EDA

The dataset contains 9841 addresses with 51 columns. [1] Here is a brief description of each column:

- Index: the index number of a row
- Address: the address of the ethereum account
- FLAG: whether the transaction is fraud or not
- Avg min between sent tnx: Average time between sent transactions for account in minutes
- Avg_min_between_received_tnx: Average time between received transactions for account in minutes
- Time_Diff_between_first_and_last(Mins): Time difference between the first and last transaction
- Sent_tnx: Total number of sent normal transactions
- Received_tnx: Total number of received normal transactions
- Number_of_Created_Contracts: Total Number of created contract transactions
- Unique_Received_From_Addresses: Total Unique addresses from which account received transactions
- Unique_Sent_To_Addresses: Total Unique addresses from which account sent transactions
- Min_Value_Received: Minimum value in Ether ever received
- Max_Value_Received: Maximum value in Ether ever received
- Avg_Value_Received5Average value in Ether ever received

- Min_Val_Sent: Minimum value of Ether ever sent
- Max_Val_Sent: Maximum value of Ether ever sent
- Avg_Val_Sent: Average value of Ether ever sent
- Min_Value_Sent_To_Contract: Minimum value of Ether sent to a contract
- Max_Value_Sent_To_Contract: Maximum value of Ether sent to a contract
- Avg_Value_Sent_To_Contract: Average value of Ether sent to contracts
- Total_Transactions(Including_Tnx_to_Create_Contract): Total number of transactions
- Total_Ether_Sent:Total Ether sent for account address
- Total_Ether_Received: Total Ether received for account address
- Total_Ether_Sent_Contracts: Total Ether sent to Contract addresses
- Total_Ether_Balance: Total Ether Balance following enacted transactions
- Total_ERC20_Tnxs: Total number of ERC20 token transfer transactions
- ERC20_Total_Ether_Received: Total ERC20 token received transactions in Ether
- ERC20_Total_Ether_Sent: Total ERC20token sent transactions in Ether
- ERC20_Total_Ether_Sent_Contract: Total ERC20 token transfer to other contracts in Ether
- ERC20_Uniq_Sent_Addr: Number of ERC20 token transactions sent to Unique account addresses
- ERC20_Uniq_Rec_Addr: Number of ERC20 token transactions received from Unique addresses
- ERC20_Uniq_Rec_Contract_Addr: Number of ERC20token transactions received from Unique contract addresses
- ERC20_Avg_Time_Between_Sent_Tnx: Average time between ERC20 token sent transactions in minutes
- ERC20_Avg_Time_Between_Rec_Tnx: Average time between ERC20 token received transactions in minutes
- ERC20_Avg_Time_Between_Contract_Tnx: Average time ERC20 token between sent token transactions
- ERC20_Min_Val_Rec: Minimum value in Ether received from ERC20 token transactions for account

- ERC20_Max_Val_Rec: Maximum value in Ether received from ERC20 token transactions for account
- ERC20_Avg_Val_Rec: Average value in Ether received from ERC20 token transactions for account
- ERC20_Min_Val_Sent: Minimum value in Ether sent from ERC20 token transactions for account
- ERC20_Max_Val_Sent: Maximum value in Ether sent from ERC20 token transactions for account
- ERC20_Avg_Val_Sent: Average value in Ether sent from ERC20 token transactions for account
- ERC20_Uniq_Sent_Token_Name: Number of Unique ERC20 tokens transferred
- ERC20_Uniq_Rec_Token_Name: Number of Unique ERC20 tokens received
- ERC20_Most_Sent_Token_Type: Most sent token for account via ERC20 transaction
- ERC20_Most_Rec_Token_Type: Most received token for account via ERC20 transactions

3.1 Unbalanced Data Set

By examining the distribution of fraudulent and non-fraudulent addresses within the dataset, it becomes evident that the dataset is unbalanced. Consequently, it is advisable to balance the training set to ensure a more unbiased training of both classes. This is done by resampling the fraudulent addresses. [2]

$$x = \frac{p(records) - rare}{1 - p}$$

- **x** := the required number of resampled records
- **p** := the desired proportion of rare values in the balanced data set
- **records** := the number of records in the unbalanced data set
- **rare** := the current number of rare target values

In this study, p is chosen to be 50.

3.2 Dropping irrelevant columns as well as duplicated records

3.2.1 Duplicated Records

During a database's history, records may have been inadvertently copied, thus creating duplicate records. Duplicate records lead to an overweighting of the data values in those records, so, if the records are truly duplicate, only one set of them should be retained. [2]

3.2.2 Nearly Constant and Zero Variance Columns

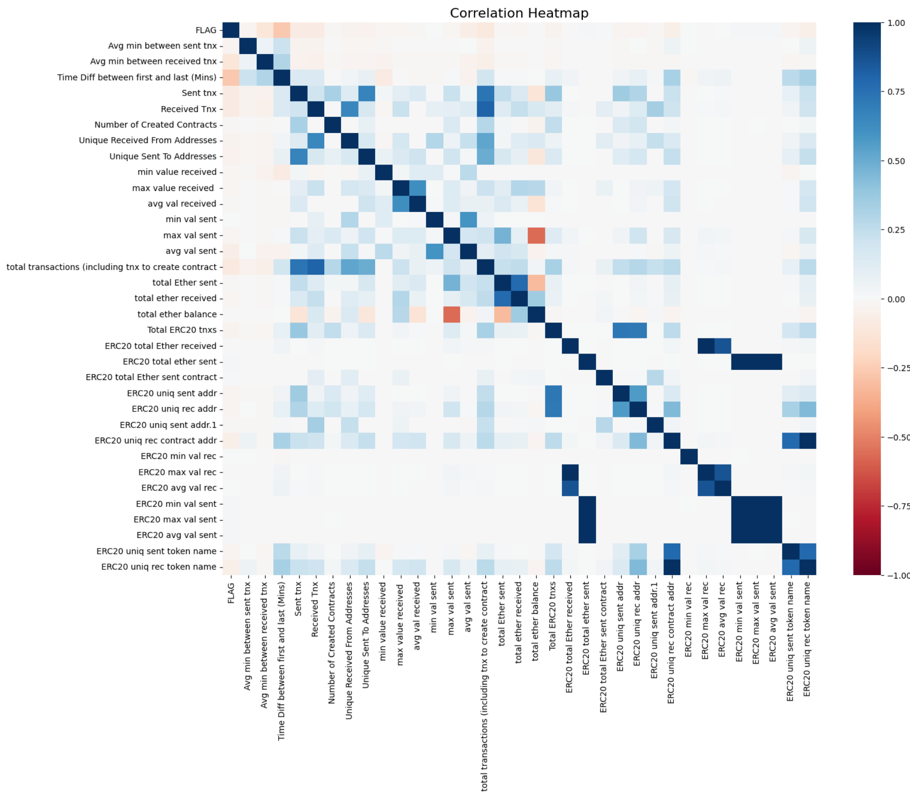
Since these columns' values are constant across all observations, they cannot have any effect on any data mining algorithm or statistical tool. These variables should be removed. [2]

3.2.3 ID Fields

The Address and Index columns should also be dropped. Since ID fields have a different value for each record, they will not be helpful for data mining algorithms. They may even be hurtful, with the algorithm finding some spurious relationship between ID field and your target. Thus it is recommended that ID fields should be filtered out from the data mining algorithms. [2]

3.2.4 Correlated Columns

Inclusion of correlated variables may at best double-count a particular aspect of the analysis, and at worst lead to instability of the model results. Therefore, we drop one of the highly correlated columns. Here is the correlation heat-map:



3.3 Missing Values

As most addresses that had missing data were due to very little or even no transactions being made, the missing data was imputed by zeros. [4]

3.4 Normalization

Variables tend to have ranges that vary greatly from each other. Such differences in the ranges will lead to a tendency for the variable with greater range to have undue influence on the results. [2] Consequently, the Min-Max Normalization technique is employed in this project to standardize the data, ensuring that variables with differing ranges are brought to a comparable scale.

4 K Nearest Neighbors(KNN)

This model is most often used for classification, although it can also be used for estimation and prediction. It works by identifying the K nearest data points to a given input, based on some distance metric (usually Euclidean distance). These nearest data points then vote to determine the classification of the input. In this project, $k = 10$ is chosen, which is a common value for k.

5 Validation and Model Evaluation

The dataset is partitioned into two distinct sets for training and testing purposes. Subsequently, the KNN model undergoes training and evaluation using the assigned test set. Each training set undergoes balancing through the technique of oversampling the minority class. This iterative process is repeated 10 times, and the resultant metrics are averaged to derive the final performance evaluation:

-	Precision	Recall	F1 measure	Support
0	0.96	0.90	0.93	765
1	0.70	0.86	0.77	218

6 Conclusion

The findings presented in section 5 demonstrate the efficacy of data mining techniques in accurately predicting fraudulent addresses and accounts of cryptocurrency. This study can be extended by exploring additional refinements to the KNN model, experimenting with alternative techniques for exploratory data analysis (EDA), and conducting comparative analyses to assess the outcomes.

References

[1] Ethereum fraud detection dataset. `https://www.kaggle.com/datasets/vagifa/ethereum-frauddetection-dataset`.

[2] Chantal D. Larose Daniel T. Larose. *Discovering Knowledge in Data: An Introduction to Data Mining*. John Wiley & Sons, 2014.

[3] Steven Farrugia, Joshua Ellul, and George Azzopardi. Detection of illicit accounts over the ethereum blockchain. *Expert Systems with Applications*, 150:113318, 2020.

[4] Han Wei Lun. Crypto fraud detection, 2023. `https://medium.com/@nusfintech.ml/crypto-fraud-detection-3d99b3298815`.

[5] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *http://bitcoin.org/bitcoin.pdf*, 2009.

[6] M. Vasek and T. Moore. There's no free lunch, even using bitcoin: Tracking the popularity and profits of virtual currency scams. *Financial Cryptography and Data Security*, 2015.