What is the problem statement?

The problem I tried to solve was deanonymizing *Ponzi Schemes* in the bitcoin blockchain network. Ponzi scheme is a scheme in which a criminal proposes a *high-yield investment program*(HYIP), takes money from investors and pays off old investors with the money of new investors. A Ponzi scheme dies when it doesn't get new investments. Since the scheme takes a lot of incoming investments and pays only a handful of previous investors, we can study the transaction attributes closely and use machine learning to deanonymize some of these Ponzi scheme addresses.

What is the proposed solution?

As mentioned above, Ponzi schemes have discernable patterns like more input investments than outputs, most of them are short-lived(link), etc. I follow Bartoletti.et al' work(link) and add some additional features to classify user's public addresses.

My solution is only for address classification(not for users) but the solution can be easily extended to users based on address clustering(on transaction inputs). The solution involves:

- Collecting HYIP or Ponzi operators' Bitcoin addresses
- Retrieving transactions using blockchain APIs
- Pre-processing transactions
- Extracting features
- Training a machine learning classifier with the features

For the first step, I use the links collected by Bartoletti.et al's work(link). This includes 32 Ponzi scheme public addresses and 3000 non-Ponzi scheme addresses(which I collected from their repo).

There are two ways to collect transactions, either by running by bitcoin node and downloading the whole blockchain or use a rate-limited API and extract attributes of an address. I used the second method and extracted addresses details from Blockchain Explorer API. Though this only returns the last 50 transactions of a non-registered user(registration takes 2-3 days to approve), there is a way to iteratively query for more transactions using the offset parameter. The API takes more than 3 hours to retrieve user transactions if his number of transactions is greater than 3000. Hence, I only extract the addresses which have transactions less than 25000.

Through the above method, I was able to extract transaction details of 3019 of 3020 addresses(31 of 32 Ponzi schemes and the remaining non-Ponzi addresses). Since the full feature dataset is not readily available and some of the users(including Ponzi

schemes) have added more transactions, this dataset extracted is a bit different from the dataset used in Bartoletti's work.

The features used are similar to Bartoletti's work were used with some additional features:

• The lifetime of the address expressed in the number of seconds. This is computed as the difference between the date of the first transaction to the address, and the date of the last transaction to/from the address.
• The activity days, i.e. the number of days in which there has been at least a transaction to/from the address
• The maximum number of daily transactions to/from the address.
• The Gini coefficient of the values transferred to (resp. from) the address.
• The sum of all the values transferred to (resp. from) the address.
• The number of incoming (resp. outgoing) transactions that transfer money to (resp. from) the address.
• The ratio between incoming and outgoing transactions to/from the address.
• The average (resp. standard deviation) of the values transferred to/from the address.
• The number of different addresses that have transferred money to the address, and subsequently received money from it.
• The minimum (resp. maximum, average) delay between the time when the address has received some bitcoins, and the time it has sent some others.
• The maximum difference between the balance of the address in two consecutive days(I normalized the dates and only considered 2-day window from the normalized date)

During feature extraction, I handle cases like coinbase transactions, transactions with no output addresses, etc.

Before training the ML classifier, EDA is performed to learn features and to gain more insights. The dataset is extremely skewed(ML algorithms assume that the data is normally distributed and skew can hurt model performance) and to account for the skew, I perform log transformation of the data. I mainly experimented with Random Forest(which was the SOTA model in the paper) and Logistic Classifiers. I measure three metrics namely precision, recall, and the AUC score. Since Recall is the most important metric(as we need to classify Ponzi schemes accurately than a non-Ponzi), I use Recall to tune the hyperparameters. Since the class imbalance is high, I follow the paper's strategies of undersampling and oversampling(using Smote)

Random Forest	Accuracy	Recall	Specificity	F-measure	Precision	G-mean	AUC
CM5 : Using Cost 5	.997	.781	.998	.714	.658	.883	.890
CM10: Using Cost 10	.995	.906	.995	.667	.527	.949	.951
CM20: Using Cost 20	.988	.969	.987	.443	.287	.978	.978
CM40: Using Cost 40	.979	.969	.979	.318	.190	.973	.974

Figure 6: Performance of Random Forest across different cost-matrices.

The above figure shows Bartolli et.al's results. The dataset this was trained on contains 6400 non-Ponzi schemes and 32 Ponzi addresses. I couldn't find the 6400 non-ponzi addresses in the link mentioned in the paper and use the 3000 non-ponzi addresses(mentioned in their repo).

Model	Precision, Recall	AUC
Random forest	100%, 22%	23%
Random forest(with minority oversampling-SMOTE(synthetic oversampling technique))	25%, 94.4%	24.6%
Random forest(with majority undersampling)	9.4%, 94.4%	8.9%

The above are my results with 3 fold cross-validation and Grid search to optimize Recall. Other methods like optimising for AUC or F1 yielded poor results and there was always a precision-recall tradeoff. The code is written with random states for reproducibility.

All the major references(code-wise and learning wise) are commented in the code and will also be added here

Merkle Tree | Brilliant Math & Science Wiki
Various Types of Cryptocurrency: How Many Cryptocurrencies are There?
What is Practical Byzantine Fault Tolerance? Complete Beginner's Guide
Blockchain analytics startup Elliptic, MIT researchers collaborate to detect money laundering in bitcoin using machine learning - The Block

Bitcoin Mixing - YouTube
GitHub - brandtoliver/Blockchain-Transaction-Classification: Bitcoin blockchain transaction classification project

An Analysis of Anonymity in the Bitcoin System: September 2011
Follow the Bitcoin With Python, BlockExplorer and Webhose.io | Automating OSINT Blog
benedekrozemberczki/RolX: An alternative implementation of Recursive Feature and Role Extraction (KDD11 & KDD12)
Massimo Bartoletti - Blockchain technologies a primer for the data scientist.pdf
https://github.com/seanconeys/Bitcoin_Ponzi_ml/
https://raw.githubusercontent.com/bitcoinponzi/BitcoinPonziTool/master/CSV/Addresses.csv
How to deal with Skewed Dataset in Machine Learning?
Titanic classification using K-fold CV | Kaggle