Deanonymizing Bitcoin

Rob Whitaker
Department of Physics
Princeton University
rwhitaker@princeton.edu

Samuel H. Cabot
Department of Astrophysics
Princeton University
shcabot@princeton.edu

Abstract

.....Graph approaches are better becuase they can use features that represent physical properties.....whatever....sailing is chill.

1 Introduction

Since several years ago, we have lived in the epoch of crytocurrency: decentralized, digitial forms of payment with encryption at their cores. Its adoption was catalyzed by a clever solution to the double-spending problem (the use of the same currency for multiple transactions): Bitcoin [**?**]. A public ledger (blockchain) hashes and stores every transaction made with the currency. Its allure and subsequent popularity, however, is perhaps attributed to its anonymity. Transactions never demand the identification of either the vendor or customer. Hence increased privacy concerns in our daily lives have driven the rise of Bitcoin. One side effect is the security it provides to criminals including, for example, sellers of illicit goods (Silk Road [**?**]), and hackers (Ransomware [**?**]). Authorities may therefore seek ways to undermine the privacy granted by bitcoin in efforts against such threats.

One approach is identification of latent structure within the available bitcoin transaction data. While actual names are not recorded, the blockchain tracks spenders and recievers through addresses linked their virtual wallets. Patterns can be extracted from these entries (e.g. an address that conistently recieves payments is likely an organized vendor). This paper uses machine learning techniques to make predictions of transactions by training on a one-year time interval of blockchain data, and hence tests the viability of different algorithms to identify underlying information in massive yet sparse datasets.

2 Data and Methods

Our dataset consists of pairs of sending and receiving addresses and the number of interactions between them over a time span from March 2012 to March 2013 (corresponding to blocks 170000 to 225000). There are 444,075 addresses and 3,348,026 recorded pairs. The number of possible transactions ($\sim 2 \times 10^{11}$) yields a sparsity of $\sim 10^{-5}$. We seek to train our models on this data to predict whether a pair of addresses will have a transaction in the future. We take two distict approaches. The first is factorization of the blockchain sparse matrix, which lets us identify eigenvalues representative of the strongest transaction patterns. The second is a graph-based analysis, in which we derive features from nodes and edges corresponding to addresses and transactions. These are two prominent and well-studied techniques, and therefore serve as the foundation of our analysis.

2.1 Matrix Factorization Methods

Factorization (Decomposition) of a matrix involves numerically approximating it as a product of two or more matrices which contain feature information. One may set a small number of features to significantly reduce dimensionality of the component matrices, which is extremely useful since it extracts the strongest latent signals from otherwise noisy data and improves prediction accuracy. We

test two decomposition models that support sparse matrices. **Singular Value Decomposition** (SVD) determines three component matrices. The outer two are unitary and are comprised of basis vectors. The central one is diagonal and contains the singular values. **Non-negative Matrix Factorization** (NMF) assumes a non-negative input matrix, and determines two constituent matrices by minimizing the Frobenius norm [**?**]. Thus vectors are always superpositions of the components. These models are summarized in Figure **??**.

2.2 Graph Feature Extraction

Lorem Ipsum.

2.2.1 Classification Methods

Use different classifiers for graph feature modeling

- **Tree** : *(DT) Decision Tree (of depth 10)*
- **Tree** : *(ET) Extra Trees Regressor (of depth 10)*
- **Linear model** : *(LR) Linear Regression*
- **Linear model** : *(RR) Ridge Regression*
- **Neighbors** : *(KNN) K Neighbors Regressor*
- **Ensemble** : *(RF) Random Forest Regressor*
- **SVM** : *(SVR) Support Vector Regression*

3 Results

3.1 Decomposition True Positive Rates

Here we present the performance of our two matrix decomposition methods, SVD and NMF. We also compare them against 'Random,' in which the component matrices are randomly generated with elements in [0, 1]. Both of the decomposition methods perform much better than the random case, as summarized in Figure **??**. NMF performs the best consistently, perhaps because of the non-negative assumption. It successfully identifies transactions at a true positive rate of 0.51, with an erroneous false positive rate of only 0.1. Both models however are far from optimal binary classification (a point at FPR = 0, TPR = 1).

Name	Formula	N_{comp}	tol	ROC	$\text{TPR}_{\text{FPR}=0.1}$
SVD	$\mathbf{X} = \mathbf{USV}$	10	10^{-10}	0.64	0.39
NMF	$\mathbf{X} = \mathbf{WH}$	20	10^{-10}	0.74	0.51
Random	$\mathbf{X} = \mathbf{R^1R^2}$	10	N/A	0.50	0.10

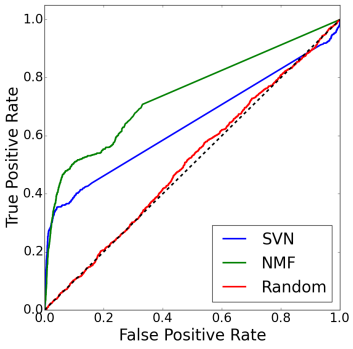

Figure 1: Properties of matrix decomposition models including name, formulation, number of components (reduced dimension), and tolerance level. Also performance including ROC area and true positive rate with at most 0.1 false positive rate. Parameters were chosen for a balance of run-time and accuracy.

3.2 Graph Classifier Performance

Graphs are cool.

Figure 2: Visualization of 50000 randomly selected transactions. The most active nodes are clearly visible. Connections between these hubs and smaller nodes provide the basis for our graph analysis.

4 Discussion and Conclusion

Lorem Ipsum