Assignment 3: Bitcoin

Barbara Engelhardt, Princeton University out 03/28/2017; due 04/18/2017

Background

Bitcoin is a virtual cryptocurrency that has reached some level of prominence in recent years. A perk of bitcoin is that it is supposed to be an anonymous way to transact money online; for example, a prominent bitcoin-only platform was Silk Road, the former online illegal drug marketplace that was shut down by authorities in 2013. We will attempt to probe the weaknesses of bitcoin anonymity.

An important operating characteristic of bitcoin is that it relies on what is called the *blockchain*, a public ledger of every bitcoin transaction ever made. The blockchain contains a list of multi-input, multi-output transactions, where the senders and receivers are referred to as *addresses*. In principle, addresses are anonymous, and treating addresses as single-use throwaways is very secure; however, users often use services to organize their bitcoins (called *wallets*) which result in multiple usages of the same address. We can use these multiple transaction observations through single addresses to look for patterns in the data.

Special thanks to Prof. Arvind Narayanan and Alex Iriza in Computer Science for their help with working with the blockchain, and Wei Hao for processing all of the blockchain data and designing the first homework assignment.

Project definition

Your goal in this homework project is to predict which currently unrelated addresses will have a bitcoin transaction in the future, given past transaction data. A key idea to use in your analysis is that there exists some sort of *latent structure* among the addresses. Approaches that shed light on which addresses are more dependent on each other than expected could allow for you to determine which addresses are likely to interact.

The training data is about a year's worth of bitcoin transactions, such that both of the input and output times of each transaction are in the time interval: Block 170000, March 2012 to Block 225000, March 2013. The data have been processed and some filters have been applied for activity, including pruning addresses that appear to be temporary addresses (or are not active). The resulting data consists of 444,075 addresses with transactions between 3,348,026 pairs of addresses. This is a sparsity rate of about 1.7×10^{-5} relative to the total number of possible transactions!

The test data is a set of 10,000 (sender, receiver) pairs that did not interact with each other in the interval of the training data. Out of these points, 1,000 represent actual transactions that happened by Block 300000, May 2014. The test data is *not* a random sample of the future data—we have added more positive test points than one would expect.

The data are available at for download at Piazza under *Resources* in a zip file *Bitcoin_data.zip*. Opening up this file, the training data are `txTripletsCounts.txt` and the test data are `testTriplets.txt`. Both data sets are presented in triplet form for sparse matrices. Each row represents an element in the sparse matrix, where the first column is the address of the sender and the second column is the address of the receiver. The last column differs depending on the data set. For the training data, it is a count of the number of times a particular sender gave bitcoins to a particular receiver, while in the test, it is simply a binary for whether or not a particular sender/receiver pair wound up interacting in the future. The addresses are indexed from 0 to 444074.

Deliverables

Your deliverables for this project include:

- A five page (not including citations) summary of the project work, which should contain (as described in the Example project write up on Piazza):
  - A title, authors' names, and abstract for the project;
  - an introduction to the problem being addressed;
  - a brief description of the data;
  - a description of the methods developed and used, and how they were fitted using training data;
  - one page describing in detail a method that you applied to the data set;
  - a presentation of the results of the methods applied to the test data;
  - a discussion of the results, including the intuition you have gained on the behavior of the models;
  - a short summary and conclusion, including extensions that you believe would be particularly valuable based on the results;
  - a *complete* bibliography to support any related work or methods that are relevant to your project.
- **please make your analysis code available to us: put a single file in the same Dropbox folder with the same name as your PDF project writeup (different extension).**

Please put your PDF write up of the project into

`https://dropbox.cs.princeton.edu/COS424_S2017/Assignment3`

by 5pm on the assignment due date, with the file name `<author1PUID>_<author2PUID>_hw3.pdf`. Please only submit one PDF per pair of authors.

We strongly recommend *writing as you go*, which means starting to write the project report as you are downloading and analyzing the data. That said, you should avoid speculative writing, and only write results once you have them.

A simple analysis

We performed a rudimentary analysis to give a baseline of what we expect. We converted the matrix of counts into a binary matrix, i.e., one that represents whether or not an address sent bitcoins to another address. Then, we performed a singular value decomposition (SVD) on this binary matrix with dimension 11 (selected by looking at a drop off in the eigenvalues). Note that, due to the size of the data, we had to use a truncated SVD routine designed for sparse data—-this is available in the R package `irlba` [1]. Finally, we multiplied the components of the SVD together, taking care to truncate values to be between $[0, 1]$, i.e., a valid probability. The interpretation of this is that we have formed a coarse estimate of a matrix of interaction probabilities between the addresses, assuming that the latent structure in the probabilities can be described by a low rank, linear, and orthogonal approximation via SVD. We can look at the probabilities for the test data locations to see how well this simple prediction procedure does in predicting future interactions. Looking directly at the probabilities in the form of a violin plot (Figure 1), we can see that in general the distribution of probabilities for the true positive locations are higher than true negative locations, though the probabilities themselves are quite low (the max is $\sim$ 0.001) and the high tail of the true negative locations extends closer to 1. The median of the true positive probabilities is 5×10^{-9} while the median of the true negative probabilities is 1.5×10^{-11}. The ROC curve (Figure 2) shows that we do a little bit better than flipping a coin.

Suggestions and Advice

This will be a difficult and open-ended assignment. Some food for thought:

- *Sparsity*: The data is very big and also very sparse. As a comparison, the data in the Netflix contest is 480,189 users by 17,770 movies, with about 1% sparsity (our data set is much larger and sparser!). The size means that you will have to be careful with how you handle the data. If you must work with the data in matrix form, sparse matrix representations will be the most efficient. Both python (`scipy.sparse`) and R (package `Matrix`) have support for sparse matrices.

- *Low signal*: Looking at the simple analysis we showed above, most of the estimated probabilities are very small. Indeed, it's very unlikely than any pair of addresses will interact in the future, simply due to the operating characteristics of the bitcoin network. This means you'll have to handle the values you are estimating carefully. You will want to think carefully about how the general lack of signal in the test set affects how you evaluate methods. For example, ROC curves informs us about our ability to detect true signal, but tells us nothing about how many spurious associations we find.

- *Modifying the data type*: The data is provided with counts of transactions between addresses, but one possible way to handle the data would be to simply turn it into binary

[1]N.b. matrices and vectors in R are indexed from 1, not 0, so you will need to add 1 to the addresses in the data provided if you want to try this yourself.

values, as we did in the sample analysis. You should be careful about the possible loss of information in this case.

- *Directionality*: The data have an explicit directionality (i.e., sender and receiver identified), which means that if you are working on the sparse matrix itself, you may want to think carefully about whether you want to apply methods to the matrix as given or to its transpose. Another possibility could be to turn sender/receiver pairs into just interactions, i.e., symmetrizing the matrix, but the directionality could be useful for determining which addresses are used mostly as vendors and which as customers.

- *Graph based methods*: We could imagine the data as a large directed graph, where addresses are nodes, and the edges are directed and have weights depending on the amount that addresses send to each other. There exist many methods for making inferences on sparse and incomplete graphs.

- *Sampling from the full matrix*: If you decide to use a stochastic optimization routine for learning, make sure that the underlying algorithm samples from the full matrix and not just from the non-zero entries. Otherwise, you will be making the missing-at-random assumption which will result in a poor generalization performance. Carefully read the implementation details of any R/Python/C++ packages you use and make sure that the algorithm handles the sparse matrix data structure correctly. If you want to implement the sampling routine and sparse matrix data structure yourself, notice that triplets are not given in a random order.

One current reference for analyzing bitcoin uses heuristic methods for prediction problems Meiklejohn et al. [2013]. Other approaches we will learn about in class or mention, including independent components analysis, collaborative filtering approaches, matrix completion methods, directed or undirected graph models, Poisson–gamma matrix factorization models, and many others. This is an unexplored data type, so feel free to use these or consider letting the data drive the development of a new method.

References

Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. A fistful of bitcoins: Characterizing payments among men with no names. pages 127–140, 2013. doi: 10.1145/2504730.2504747. URL http://doi.acm.org/10.1145/2504730.2504747.

Figure 1: Violin plots of probabilities that addresses in the test set send bitcoin to each other. Points for the probabilities themselves are “jittered” lightly in the background. The left violin plot is for the true negative test points while the the right violin plot is for the true positive test points. Note that the y-axis is on a log scale.

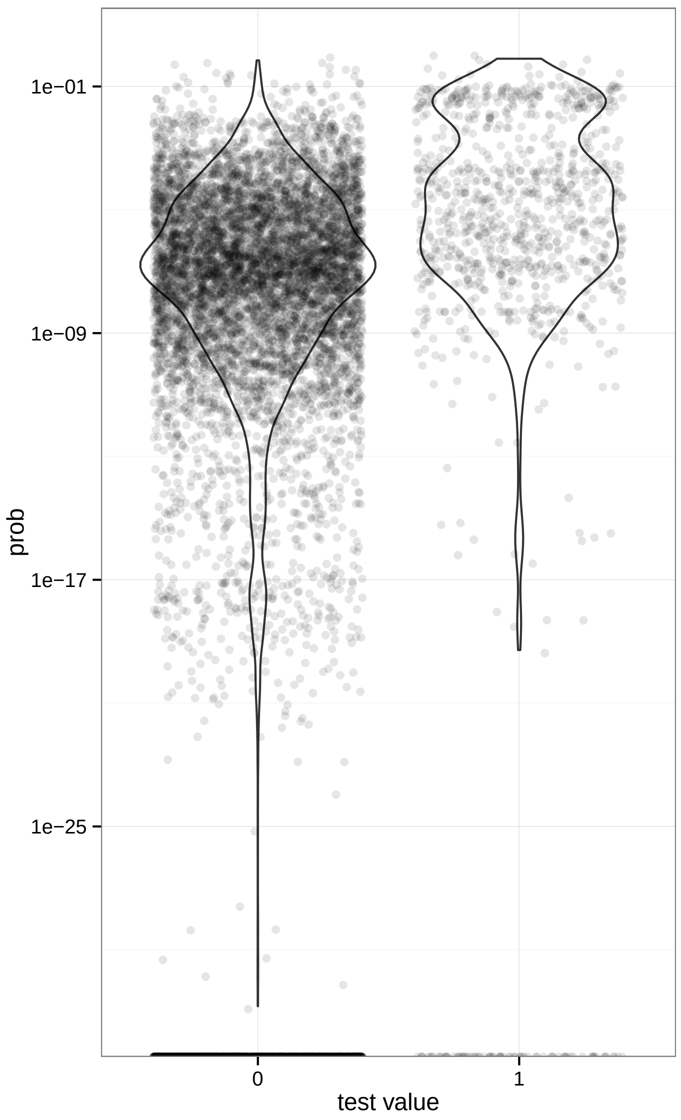
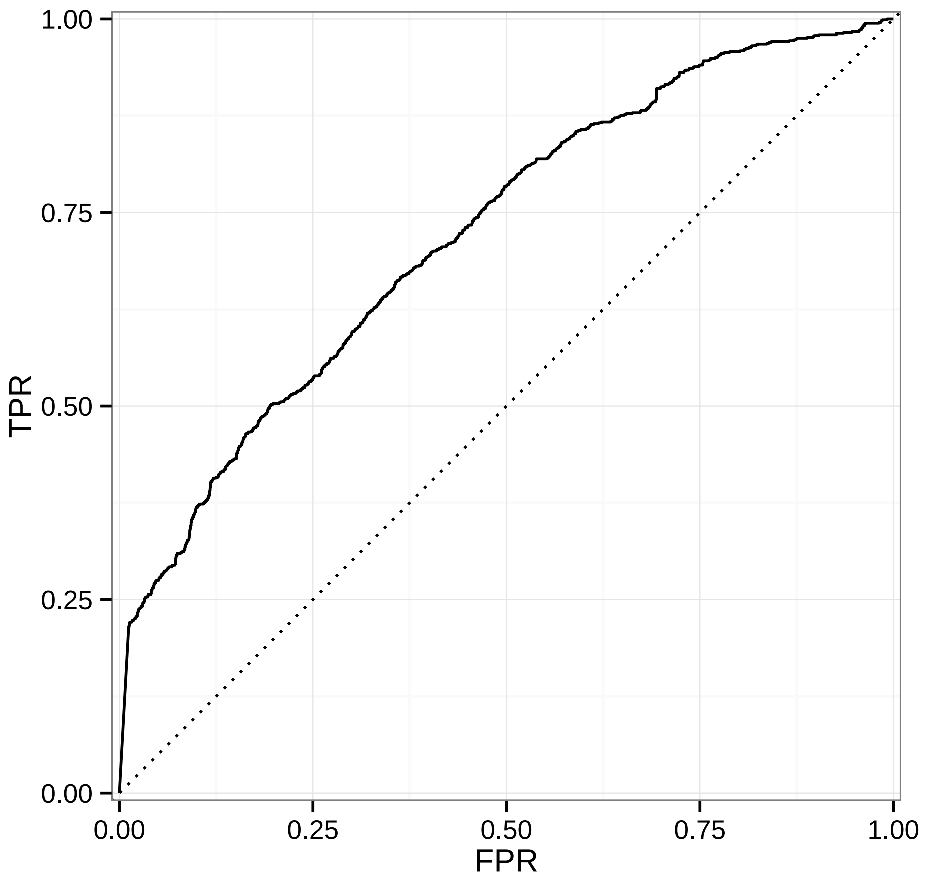


Figure 2: ROC curve for the probability of interaction computed from a naive SVD.


TPR
FPR