Link Prediction In Graphs

Brandon Dolson

May 4, 2020

1 Introduction

With real-world data, often the data can be represented in the form of a graph. The different datasets can represent completely different systems, but the thing they all have in common is a similar graph structure. There are many examples of datasets that can be transformed into graphs such as social networks, road networks, or even the internet. When a dataset is represented as a graph, it allows the use of graph algorithms to analyze the data contained within.

For the project, the datasets used are the Bitcoin OTC Trust Weighted Signed Network[1] and the Bitcoin Alpha Trust Weighted Signed Network[2], which are datasets about two bitcoin trading networks. These datasets represent trust within a bitcoin trading network where the vertices in the graph represent users and the directed edges represent the trust that one user assigns to another user after they trade. The goal of the project is to use various graph algorithms to make conclusions about the data. Specifically, I wanted to rank users from most to least trustworthy, predict whether two users would trade, and predict the trust that a user would assign to another user after they trade.

[1] https://snap.stanford.edu/data/soc-sign-bitcoin-otc.html
[2] https://snap.stanford.edu/data/soc-sign-bitcoin-alpha.html

2 Methods

2.1 Ranking Users

For ranking the users from most trustworthy to least trustworthy, I decided to use the PageRank algorithm. One issue with using these networks with PageRank is that the weights of the edges range from -10 to 10, but PageRank doesn't work with negative weights on edges. I fixed this by adding 10 to all weights as a simple way of scaling weights to be in the range of 0 to 20. I ran PageRank on the resulting graph and then ranked the results from highest to lowest as the ranking of the users.

2.2 Link Prediction

For predicting links in the graph, the approach used was similar to the method used in the homework. The method used in the homework needed to be edited slightly since in the homework edges only existed between users and products and links were only predicted between users and products, but in the bitcoin datasets the edges were between users to users. The four metrics used to predict links were:

1. Common Neighbors
$$|N(u) \cap N(v)|$$
2. Adamic-Adar Index
$$\sum_{u=N(x)\cap N(y)} \frac{1}{\log(|N(u)|)}$$
3. Preferential Attachment
$$|N(u)| \times |N(v)|$$
4. Triadic Closure
$$w(u,t) + w(t,v)$$

Which are also defined above. First, only one-fourth of the original edges were kept to calculate the indices. Next, the Jaccard Index was calculated for all pairs of vertices. Then, the metrics above were calculated for pairs of

vertices. Next, the edges were sorted by the values of the metrics. Then, the edges that already exist in the graph are removed from the list of predicted edges. Finally, the top 25% of predictions for each metric were kept as the set of predicted edges.

2.3 Weight Prediction

To predict the weights on edges, a simple linear regression model was used. The feature vector used for this model had four attributes:

1. The average weight from the incoming edges of the user doing the rating
2. The average weight from the outgoing edges of the user doing the rating
3. The average weight from the incoming edges of the user being rated
4. The average weight from the outgoing edges of the user doing the rated

The purpose of these averages is to try and measure the trustworthyness and how a user rates other users for the user doing the rating as well as the user being rated. The y value for each vector was the weight of that vector.

3 Results

3.1 Ranking Users

For the Bitcoin OTC dataset, the top five users and their PageRank were found to be:

1. 35, 0.0157
2. 2642, 0.0117
3. 1810, 0.0068
4. 7, 0.0065
5. 2028, 0.0065

And the bottom five were:

1. 5975, 3.4882×10^{-5}

2. 5976, 3.4882×10^{-5}
3. 5977, 3.4882×10^{-5}
4. 5993, 3.4882×10^{-5}
5. 6000, 3.4882×10^{-5}

3.2 Link Prediction

For the Bitcoin OTC dataset, the accuracy of each link prediction metric are:

1. Common Neighbors Accuracy: 0.641
2. Adamic-Adar Accuracy: 0.568
3. Preferential Attachment Accuracy: 0.941
4. Triadic Closure Accuracy: 0.963

3.3 Weight Prediction

The linear regression model was trained on the Bitcoin OTC dataset with a mean-squared error of: Training MSE: 5.8755 And it was tested on the Bitcoin Alpha dataset with a mean-squared error of: Testing MSE: 4.7741

4 Discussion

4.1 Ranking Users

The algorithm used to scale the weights was simply adding 10 to all of the weights. There are many other ways to do this and each will change the results of PageRank differently. Another example of a method that can be used to scale the weights is by using the sigmoid transformation. This transformation will make the relative distance between -10 and 10 further apart, which will make their presence in the graph change the PageRank more strongly. Neither method to scale the weights is better in the general sense because they only change the ranking of the users and the correct ranking depends on the context of the problem. One potential avenue for

future work is to find more ways to scale the weights and determine the context in which they would be best used.

4.2 Link Prediction

The metrics used to predict the links were the ones used in the homework. There are certainly more metrics that can be used and maybe some that could exploit the structure of these graphs more efficiently An interesting avenue for future work would be to develop new metrics for prediction for these or other types of graphs.

4.3 Weight Prediction

The model used to predict weights is a very simple linear regression model with a simple feature vector. When creating the feature vector, I tried to have the attributes represent the trust of the users as well as how that user rates other users. I believe that the feature vector used does represent that, but I couldn't figure out a way to include users other than the user being predicted as well as the user doing the prediction in the vector. For example, the the user doing the prediction may have a low rating, but that rating is from users who are considered untrustworthy or the user doing the prediction rates other users poorly, but the users rated are also considered untrustworthy. There are many possible scenarios similar to that that the current feature vector doesn't account for that may affect the results. A possible avenue of future work could be to create a more complex model or feature vector that would account for these possible scenarios.