Overall idea

Trust is a fundamental necessity since any user can initiate a trans-action, receive money from the other user and never send money back in the other currency.To achieve a web of trust, users rate each other after successful or unsuccessful trades. We analyze these datasets using the fairness and goodness method from these two measures can be used to predict the weight of a given edge (the rating that u gives to v) with good accuracy. We show that whenever the edge is reciprocated, social interaction becomes a better predictor than goodness, and users apply the talion law: "an eye for an eye". We analyze the relationship between trust in these networks.

Once we get the measures of goodness and fairness, we then predict the maliciousness of a rater and a ratee (both of them are users of bitcoin) using a classifier where the feature vectors are the [fairness,goodness] values for each trader.

Methodology

$$g(v) = \frac{1}{|in(v)|} \sum_{u \in in(v)} f(u)$$

$$f(u) = 1 - \frac{1}{|out(u)|} \sum_{v \in out(u)} 1 - g(v)$$

$$\Rightarrow f(u) = \frac{1}{|out(u)|} \sum_{v \in out(u)} g(v)$$

This equation is used to update goodness and fairness iteratively until it converges to a given value.

These values are used as feature vectors for each user to predict how good of a rater and ratee he is.

Dataset

Bitcoin-OTC (OTC for short) and Bitcoin-Alpha2(Alpha for short) : Both these exchanges allow users to rate others on a scale of -10 to +10 (excluding 0). According to OTC's guidelines, a rating of -10 should be given to fraudsters while at the other end of the spectrum, +10 means "you trust the person as you trust yourself". The other rating values have intermediate meanings. Therefore, these two exchangesexplicitly yield WSNs. We scale the edge weights to be between -1 and 1

Classification

Scikit-learn has been used for classification task. For the ground truth, the rater giving average rating less than 0.5 is considered malicious rater, otherwise non-malicious. Same applies for ratee.

The following classifiers were tried upon:

a. Nearest Neighbors

b. Linear SVM
c. RBF SVM
d. Gaussian Process
e. Decision Tree
f. Random Forest
g. Neural Net
h. AdaBoost
i. Naive Bayes
j. QDA

Gaussian Process with RBF kernel gave the best results. All of the techniques have accuracy ranging from 94-97% for rater.

Results

- - - -X

Test-set accuracy:-

Rater: 97%

Ratee: 99%

References

- - - - - X

-> https://cs.stanford.edu/~srijan/pubs/wsn-icdm16.pdf The use of this to detect rating-based behaviour is my own idea.

Libraries used

Scikit-learn

Numpy