Visualizing and Modelling the Bitcoin Blockchain with Probabilistic Graph Models

Ramiro Mata[1 2]

[1]DTU Compute

[2]Chainalysis

ramiro@chainalysis.com, s161601@student.dtu.dk

Abstract. *In this technical report we present a statistical profile of the Bitcoin network, visualizations of community structure, and finally preliminary results of applying probabilistic graph models for supervised machine learning tasks. We find that network centrality measures reveal that while exchanges dominate the bitcoin ecosystem, there are a few non-exchange services (such as darknet markets and merchant services) that are very central. Further, when using the Lovain method for community detection, we find communities that consist of entities belonging to the same category (e.g. mining pools, gambling, darknet markets) while others are more diverse. Finally, in the task of category classification of bitcoin clusters, we find that adding network centrality features improves accuracy by 13.5%, and adding GraphSAGE embeddings increases it by a further 6.6%. These improvements are seen when using a Stochastic Gradient Descent Classifier model as is done by Hamilton et al. (2017), the creators of the GraphSAGE algorithm.*

1. Introduction

The Bitcoin blockchain can be seen as a distributed database that records transactions between users. While this transactional data is open to the public, interpreting it can be challenging due to the mechanics of bitcoin transactions and pseudo-anonymity. Psuedo-anonymity refers to the fact that while a user's Bitcoin address is publicly recorded in the blockchain, the identity of the person/institution that address belongs to is unknown. There have been efforts in both academia and private industry sectors to deanonymize Bitcoin and other cryptocurrencies. Chainalysis has proprietary knowledge and software to solve the problem of knowing which addresses belong to the same entity (i.e. clustering). However, knowing the identity of that cluster is a different challenge. In this report we view the bitcoin blockchain under the lens of network science and graph theory. In particular, we use network centrality measures and features of the topology of the graph as signals for machine learning classification models with the goal to understand which category those unknown clusters belong to. That is, while this approach will not reveal the identity of the cluster, it nonetheless can provide information regarding as to whether it behaves as an exchange or as a darknet market, for instance.

2. The Bitcoin Ecosystem

The Bitcoin Ecosystem can be characterized by entities that transact with each other. At Chainalysis we call these entities clusters. These entities can be attributed to the following categories: exchanges, gambling, miningpool, darknet market, scam, ransomware, merchant services, hosted wallet, mixing, stolen bitcoins, child abuse material, terrorist financing, peer-to-peer exchanges, and other. The 'other' category is simply to collect the clusters that don't fall under the other categories. The number of clusters/entities per category are shown in Figure 1. As the figure shows most of the entities that have been identified are exchanges, gambling, miningpools and darknet markets. A category that has been omitted is that of 'Unknown.' While we might know that a set of addresses belong to the same entity, we might not know who that entity is due to pseudo-anonymity in the Bitcoin blockchain. All clusters for which we don't know their identity are in the 'Unknown' category.

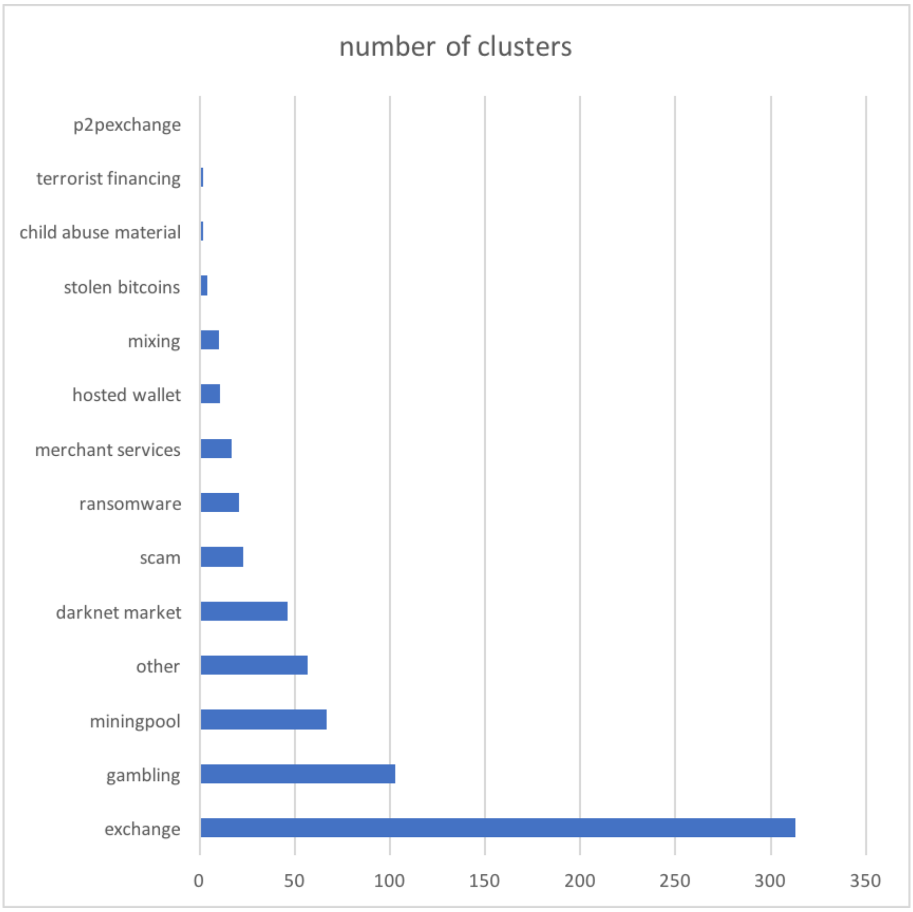


Figure 1. Cluster Categories

3. A Network Science Perspective

In this section we show the main statistical properties of the Bitcoin Network. For a detailed statistical profile, please see appendix and/or html file provided. Note that two different data abstractions were used for the network analyses in this report. The first abstraction considers all clusters and models them as a directed multigraph. We obtain Page Rank statistics and Figure 2 under this abstraction. The second data abstraction considers the subset of most active clusters in the network, where activity is measured by total number of transactions the cluster was involved in. The second data abstraction models the network as an undirected graph. The rest of the analyses in this report stemmed from the latter data abstraction, including the statistical profile in the appendix. Edge weights were not considered in any of the analyses. Note that the second data abstraction was used due to memory and processing constraints. As such, we consider these preliminary results, and we wish to include the entire data set in future analyses.

3.1. Bitcoin Network Statistical Profile

Figure 2 shows all clusters distributed according the their number of transactions received and sent in log scale. As shown in the plot, the vast majority of the clusters are the at the lower-left corner, which signifies that the vast majority of clusters have only been involved in less than 10 transactions. We would expect exchanges, as well as merchant services, to be in the upper-right corner given the volume of transactions they deal with.

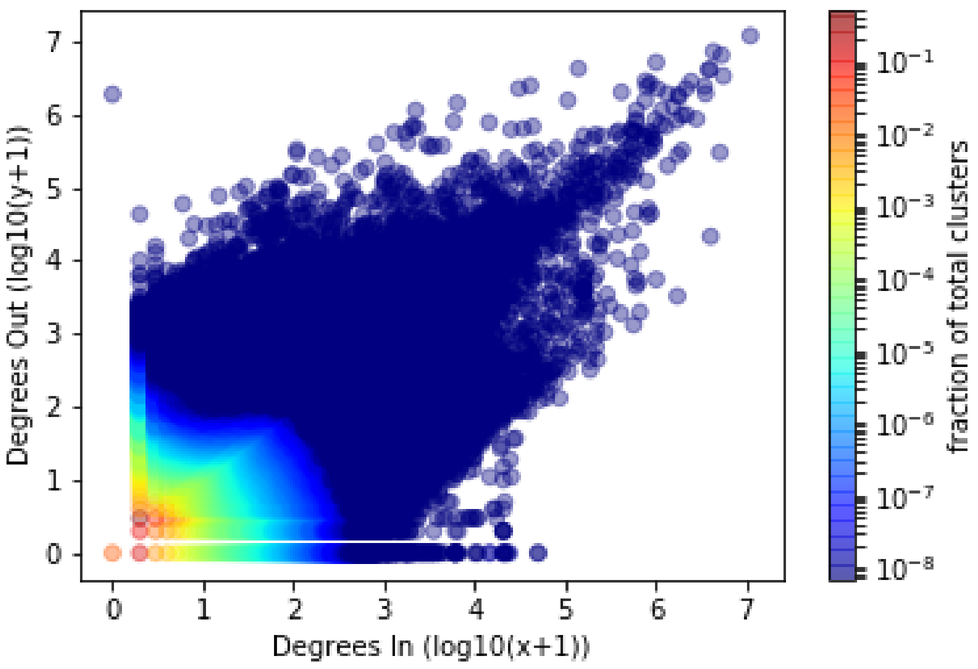


Figure 2. Cluster distribution according to transactions received and transactions sent during their entire bitcoin transaction history.

3.2. Community Structure

We used Gephi to detect and visualize communities of the Bitcoin Network. The results are shown in Figure 3. There are 9 prominent communities as detected by the Louvain modularity maximizing method. It is important to note that the community detection algorithm has no information regarding the category of each cluster, yet we see some communities that are heavily comprised of certain categories.

The Illicit Community The most upper-left community contains the majority of darknet clusters (in black), ransomware clusters (in yellow), and mixing services (in purple). Further, the orange node is LocalBitcoins.com, a peer to peer bitcoin exchange which does not enforce Anti Money Laundering (AML) nor Know Your Customer (KYC) regulations. This finding suggests that clusters associated with illicit activities are not equally distributed in the bitcoin ecosystem, and are instead localized. Presumably, this community formed as service providers that didn't enforce the above regulations created a regulation-free heaven that attracted users that use bitcoin for illicit activities. Some of the exchanges (in green) in that community have been identified with a high risk score by Chainalysis (based on other metrics) prior to this work. Thus, community structure can be used as a method to aid in finding exchanges or other bitcoin services that have high exposure to clusters associated with illicit activities.

We also find that certain communities seem to be populated by clusters of the same category. For instance, the upper-right most community is predominantly comprised of mining pools while the two adjacent to it consist mostly of gambling clusters. This is in contrast to the lower-most (and largest) community, which is populated by a more diverse set of categories and consists of the biggest exchanges and merchant services yet is almost devoid of clusters associated with illicit activities.

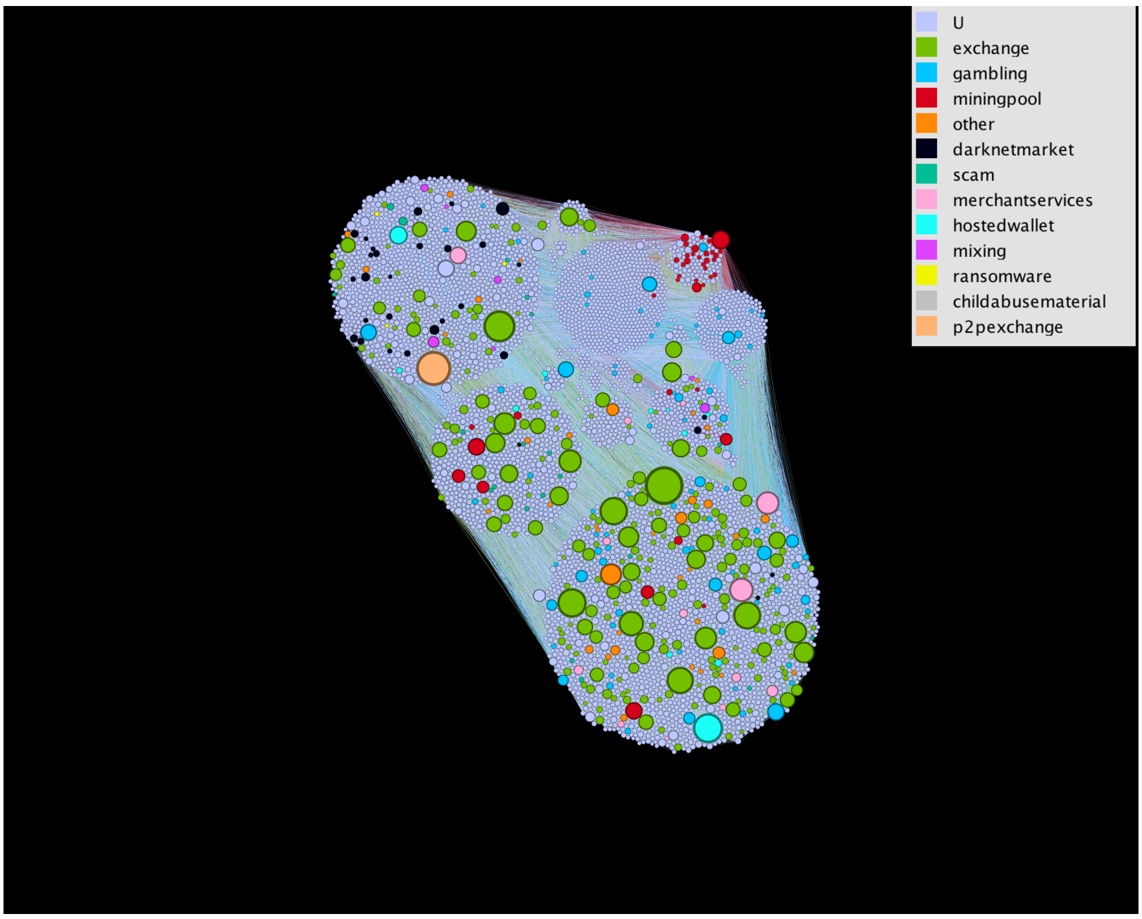


Figure 3. Bitcoin Communities: Communities are the round conglomerate of nodes. The size of the nodes is based on their degree.

4. Probabilistic Graph Models

The second goal of the project entails using probabilistic graph models for the goal of predicting what category (e.g. ransomware, terrorist financing, etc) a cluster belongs to given features about how they interact (i.e. transact) with the network, and their role in the network. At Chainalysis we call the clusters which we have identified who they are as light matter , and the clusters which we know nothing about (except for how they behave) as dark matter . With these terms the second goal of this project is to illuminate the dark matter given what we know about the light matter.
Specifically, there have been a few classification, state-of-the-art algorithms that were tailored specifically for data that has an inherent graph-structure - this is something that has so far not been explored in the machine learning literature until recently. These algorithms are able to obtain significant improvements over classical, classification machine learning algorithms on benchmark datasets by incorporating knowledge of the graph structure, and by introducing elements from deep learning models in order to make the models more expressive/flexible. In this report we focus on the GraphSAGE algorithm developed by Hamilton et al. (2017).

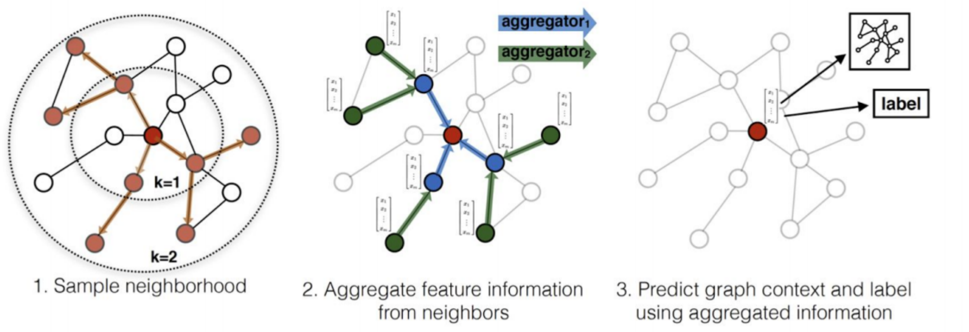
4.1. GraphSAGE

Instead of learning a function that outputs labels, GraphSAGE learns a function that generates node embeddings. These embeddings act as numerical social representations that capture neighborhood similarity and community membership by taking the graph topology information into account. This topology information is usually not captured in traditional machine learning approaches. Thus, Graph Sage can be viewed as a data augmentation tool for domains where there is an inherent graph structure to the data. These embeddings along with other node features can then be used by machine learning classification pipelines.

Particularly, the learned function samples and aggregates data from the node's local neighborhood in the graph as shown below. For every node in the network, GraphSAGE samples uniformly node features from its immediate neighborhood (as shown by k=1 below, i.e., one hop away), then samples node features from nodes that are 2 hops away. Crucially, the sampled data from nearby nodes is aggregated using different signal processing operators (e.g. LSTM, Max-Pool) used in the Deep Learning models. Just as there is a weight matrix between each layer in Deep Learning Networks, GraphSAGE contains weight matrices between each hop, such that the features that are sampled and aggregated can then be 'corrected' by these weights to optimize a loss function. The exact forward pass algorithm is presented below. For the backward pass, stochastic gradient descent algorithms can be used to optimize the model's parameters.

5. Data and Methods

To obtain the embeddings we used the second data abstraction (described in Section 3). In total we included 148,268 clusters and 7,994,203 edges corresponding to the most active clusters. We experimented with the mean-aggregator in GraphSAGE, which produced a 256 dimensional embedding vector for each node in the graph. In addition to this, we have two more sets of node features. One set captures non-network metrics, such as number of



k=1
k=2
1. Sample neighborhood
aggregator$_1$
aggregator$_2$
2. Aggregate feature information from neighbors
label
3. Predict graph context and label using aggregated information

Figure 4. Visual Illustration of GraphSAGE algorithm (from Hamilton et al (2017)).

```
Algorithm 1: GraphSAGE embedding generation (i.e., forward propagation) algorithm
  Input  : Graph $\mathcal{G}(\mathcal{V},\mathcal{E})$; input features $\{\mathbf{x}_v, \forall v \in \mathcal{V}\}$; depth $K$; weight matrices
           $\mathbf{W}^k, \forall k \in \{1,...,K\}$; non-linearity $\sigma$; differentiable aggregator functions
           $\text{AGGREGATE}_k, \forall k \in \{1,...,K\}$; neighborhood function $\mathcal{N} : v \rightarrow 2^{\mathcal{V}}$
  Output : Vector representations $\mathbf{z}_v$ for all $v \in \mathcal{V}$
1 $\mathbf{h}^0_v \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$ ;
2 for $k = 1...K$ do
3     for $v \in \mathcal{V}$ do
4         $\mathbf{h}^k_{\mathcal{N}(v)} \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}^{k-1}_u, \forall u \in \mathcal{N}(v)\})$;
5         $\mathbf{h}^k_v \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}^{k-1}_v, \mathbf{h}^k_{\mathcal{N}(v)})\right)$
6     end
7     $\mathbf{h}^k_v \leftarrow \mathbf{h}^k_v / \|\mathbf{h}^k_v\|_2, \forall v \in \mathcal{V}$
8 end
9 $\mathbf{z}_v \leftarrow \mathbf{h}^K_v, \forall v \in \mathcal{V}$
```

Figure 5. GraphSAGE pseudocode (from Hamilton et al (2017)).

addresses in cluster and number of in and out transactions. The other set captures network centrality measures for each node. Specifically, we include the following node centrality features: degree centrality, closeness centrality, betweeness centrality, katz centrality and pagerank centrality.

6. Results and Discussion

We evaluate the six machine learning models on three different datasets. The first dataset contains only the number of addresses, transaction in and transactions out as features. The second dataset contains the latter as well as the network centrality features. Lastly, the third data set contains all the aforementioned data plus GraphSAGE's embeddings. When using SGD Classifier as is done by Hamilton et al (2007), we find that adding network centrality features increases accuracy by 13.5%, and that adding embeddings increases it by a further 6.6%. The results for all models can be seen in Table 1. Note that we used a 5-K cross-validation split with 30% of the data for testing. Thus, the accuracies reported in Table 1 are computed by taking the mean of those 5 folds. The variance of the test set results can be visualized in the figures below.

Table 1. Mean Test Accuracy across Models and Data Sets

Model	Cluster Data	Network Centrality Data	GraphSAGE Embeddings
SGD Classifier	17.6%	31.2%	38.1%
AdaBoost	46.7%	43.3%	37.2%
KNN	45.5%	48.9%	47.1%
LogRegression	20%	34.8%	39.5%
Neural Network	49%	51.4%	47.9%
Random Forest	40%	46.1%	40.6%

While we find accuracy improvements in the bitcoin network, they are not as substantial as those reported on benchmark datasets. Further, while we find similar improvement for Logistic Regression, the rest of the models don't exhibit these gains. In fact in the other models adding embeddings decreases the performance. Further analyses are required to understand the decrease in performance. Preliminary results obtained by looking at the confusion matrix of those models that did not exhibit gains with embeddings, suggest that the models suffered from class imbalance. That is, they learned to predict everything as an exchange, and since the exchange class constitutes nearly 50% of the instances, they were obtaining higher performances than the rest. Only Logistic Regression and SGD Classifier were corrected for class imbalance using scikit-learn class arguments. The other models do not have options for class imbalance.

Further analyses are required to evaluate the added accuracy performance that GraphSAGE embeddings can provide. In the future, we expect to experiment more with different aggregator functions and with the dimensionality of the embeddings in search for better performance. Moreover, Hamilton et al use a loss function that encourages embedding of nearby nodes to be similar. However, in the bitcoin domain, this might not be the case. For instance, nodes near an exchange or merchant service are likely to be users rather than other exchanges or merchant services. Thus, exploring other loss functions that are more apt for the bitcoin domain might result in higher quality embeddings. Finally, GraphSAGE currently does not support mutligraphs nor directed graphs. Exploiting the information held in edges when modelling the bitcoin network as a directed, weighted multigraph is also an interesting venue for future research.

References

Inductive Representation Learning on Large Graphs. W.L. Hamilton, R. Ying, and J. Leskovec. 2017

7. Appendix

See attached HTML file for Statistical Profile of the Network.

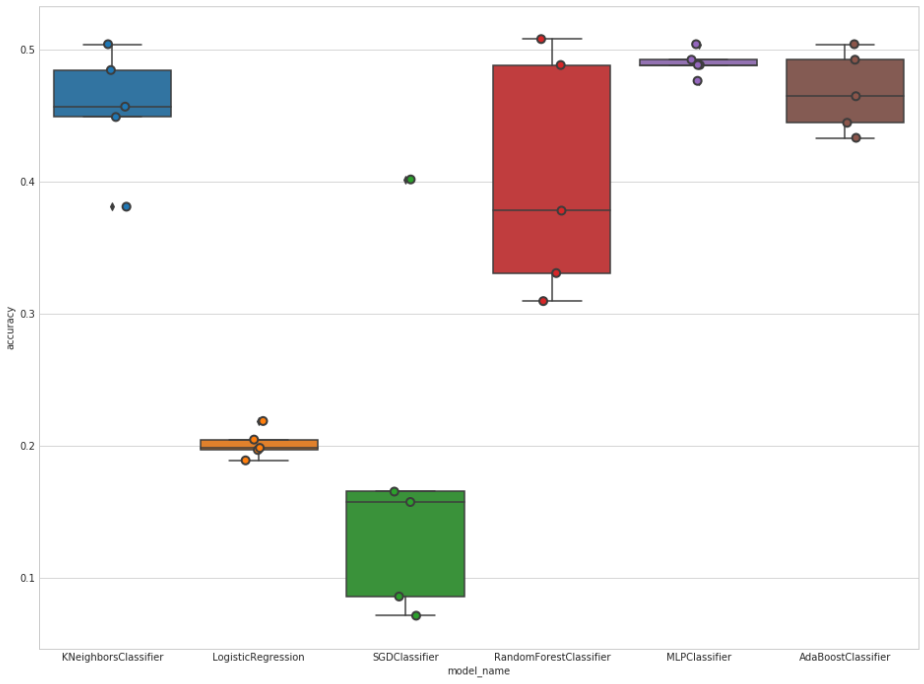


Figure 6. Test Accuracies for all models using only cluster and transaction in and out size features.

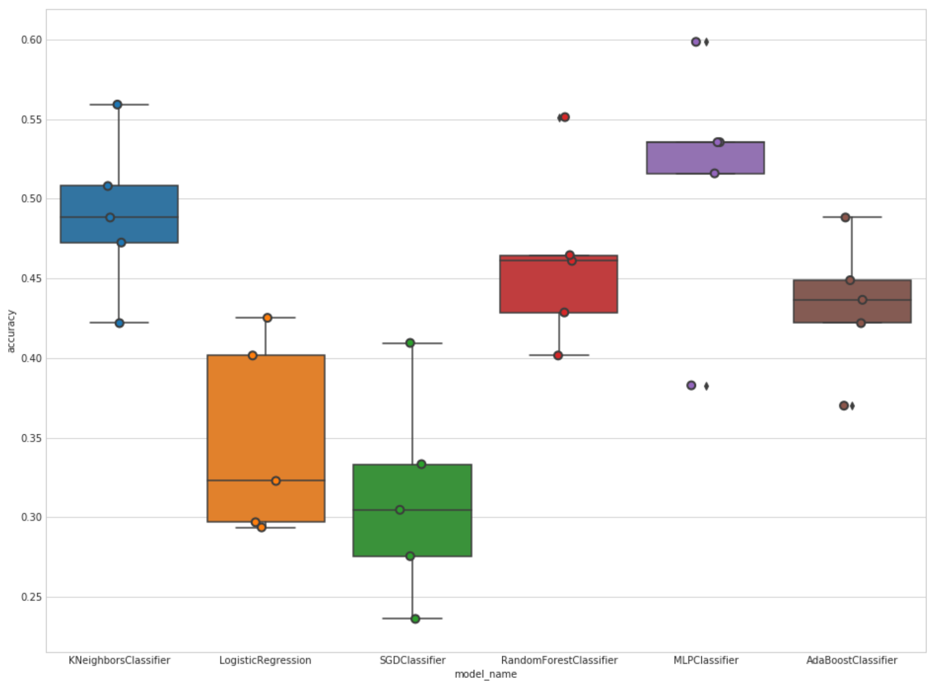


Figure 7. Test Accuracies for all models using network centrality features as well as cluster and transaction size features.

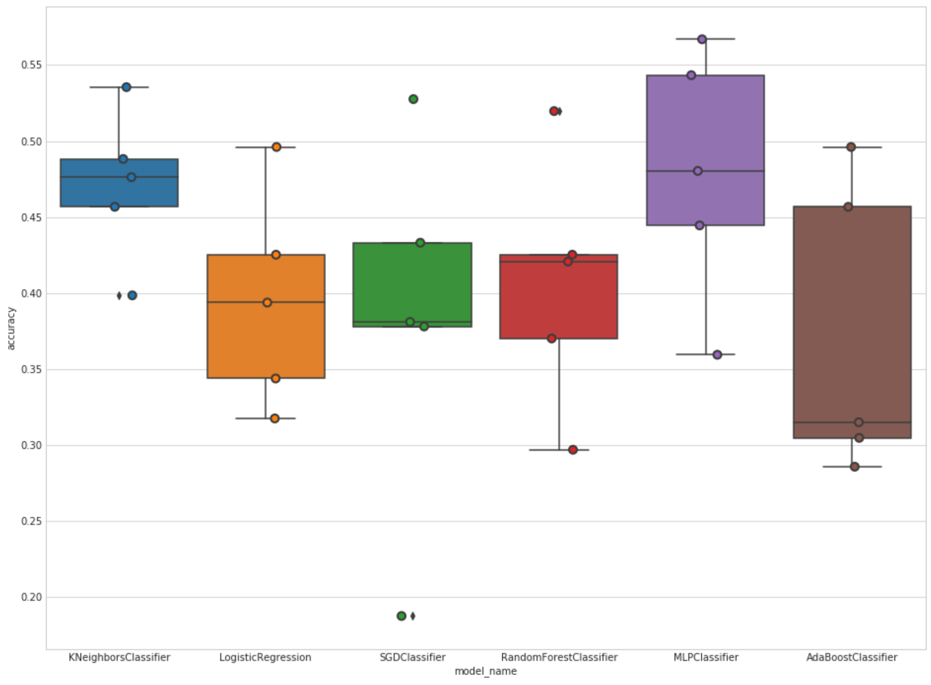


Figure 8. Test Accuracies for all models using all data including GraphSAGE embeddings.