Project: Fraud Detection in Financial Transactions using Machine Learning Techniques in Spark

Log file wit images from experiment 1

Team Members:

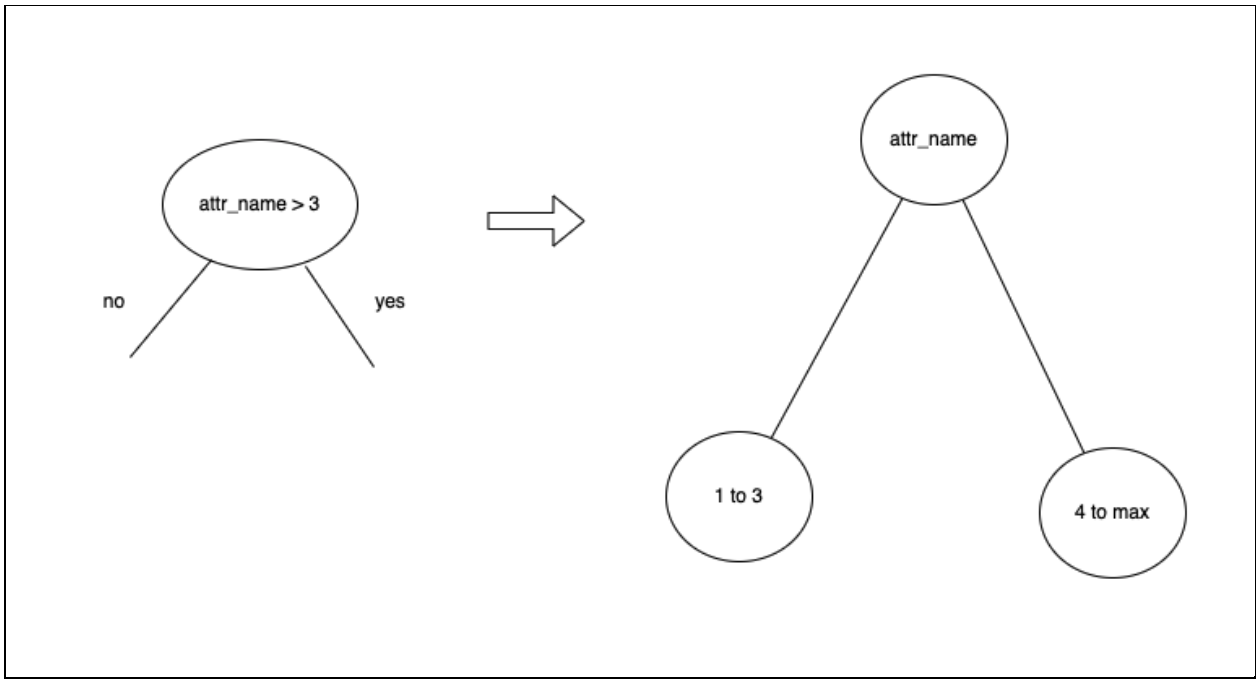
1. **Manasa M Bhat (mmb190005)**
2. **Manneyaa Jayasanker (mxj180040)**
3. **Swathi Poseti (sxp190117)**
4. **Venkatesh Sankar (vxs200014)**

Graphical representation of Decision tree explanation:

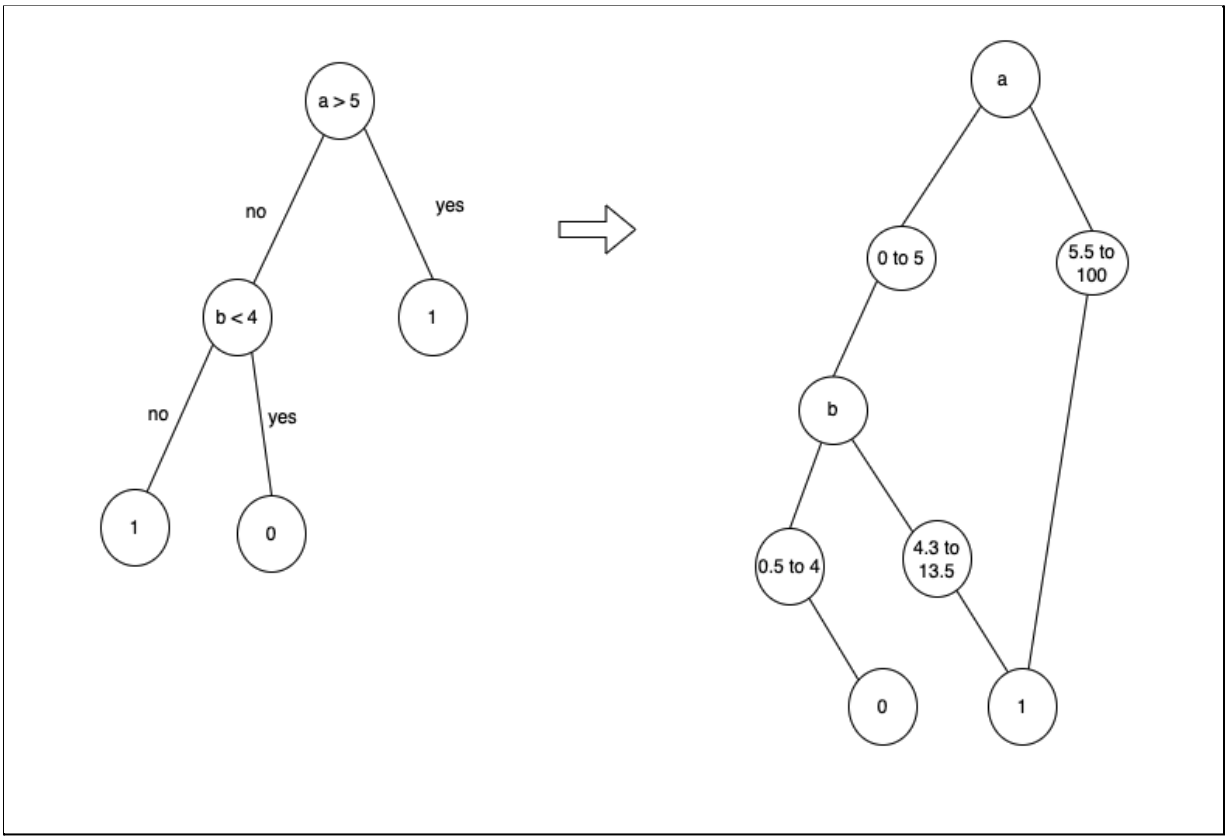
We have chosen graphical representation of the decision tree using the networkx framework.

Since we are using the directed graph, instead of adding new leaf nodes of 0s and 1s when a leaf is reached. There is a single 0 node and single 1 node. And when a leaf condition is reached an edge is created from the respective node to the 0 or 1 node.
So in the below figures all nodes finally end in either 0 or 1.

The condition over which the traversal needs to take place is also stored as a node. For example,
If the node condition is “attr_name > 3 “ then the following nodes and edges are created:
Attribute name : attr_name becomes a node, range 0 to 3 becomes a node, and range 4 to higher becomes another node with edges between attr_name to each range.



Hence the graphical representation has nodes for all the attribute names as well as for attribute values. And the overall graphical representation will look like the below transformed decision tree in graphical form



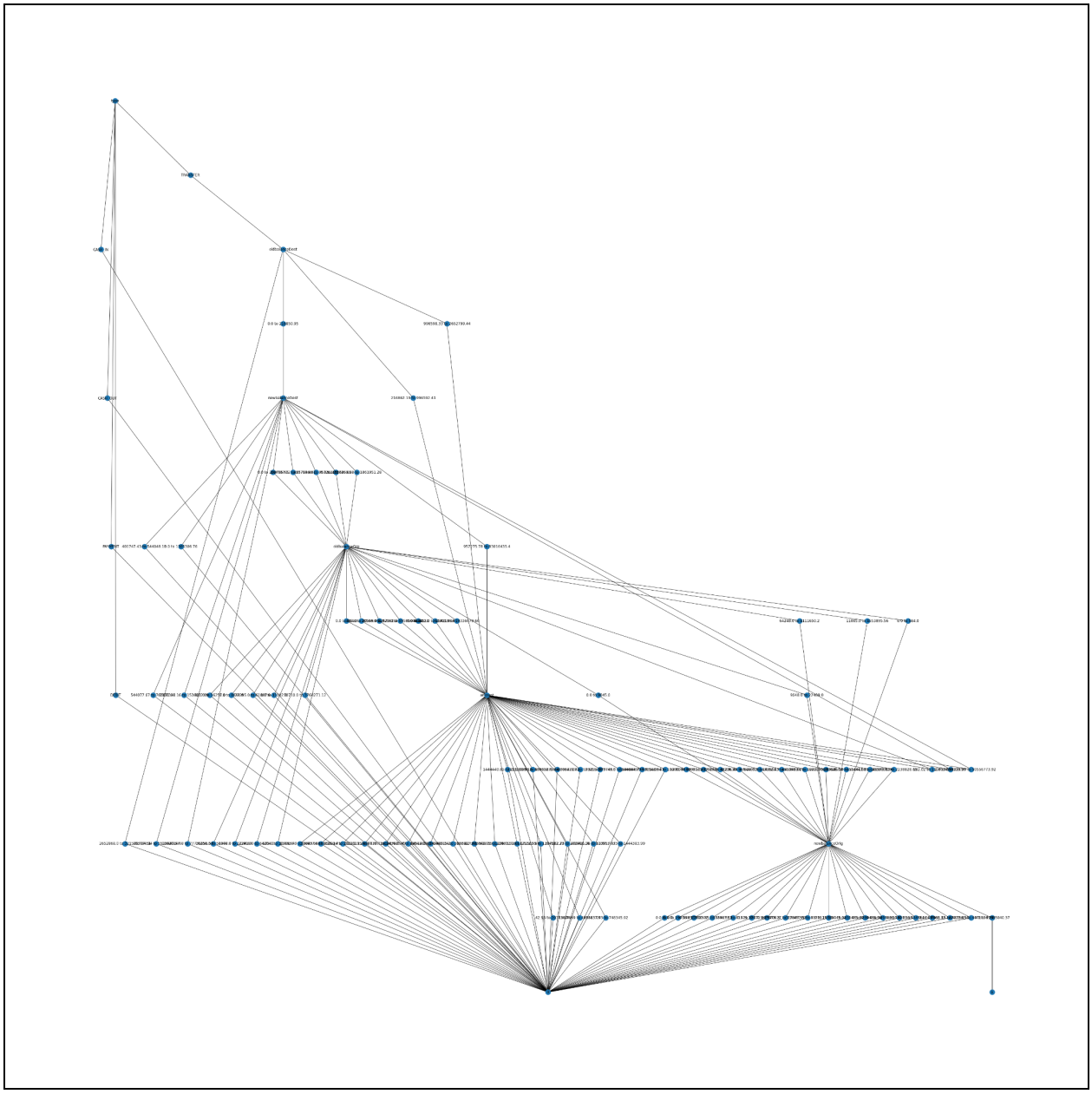
We have introduced three hyper parameters in building decision tree:

1. Bucket_count
2. Depth_limit
3. nodes_limit

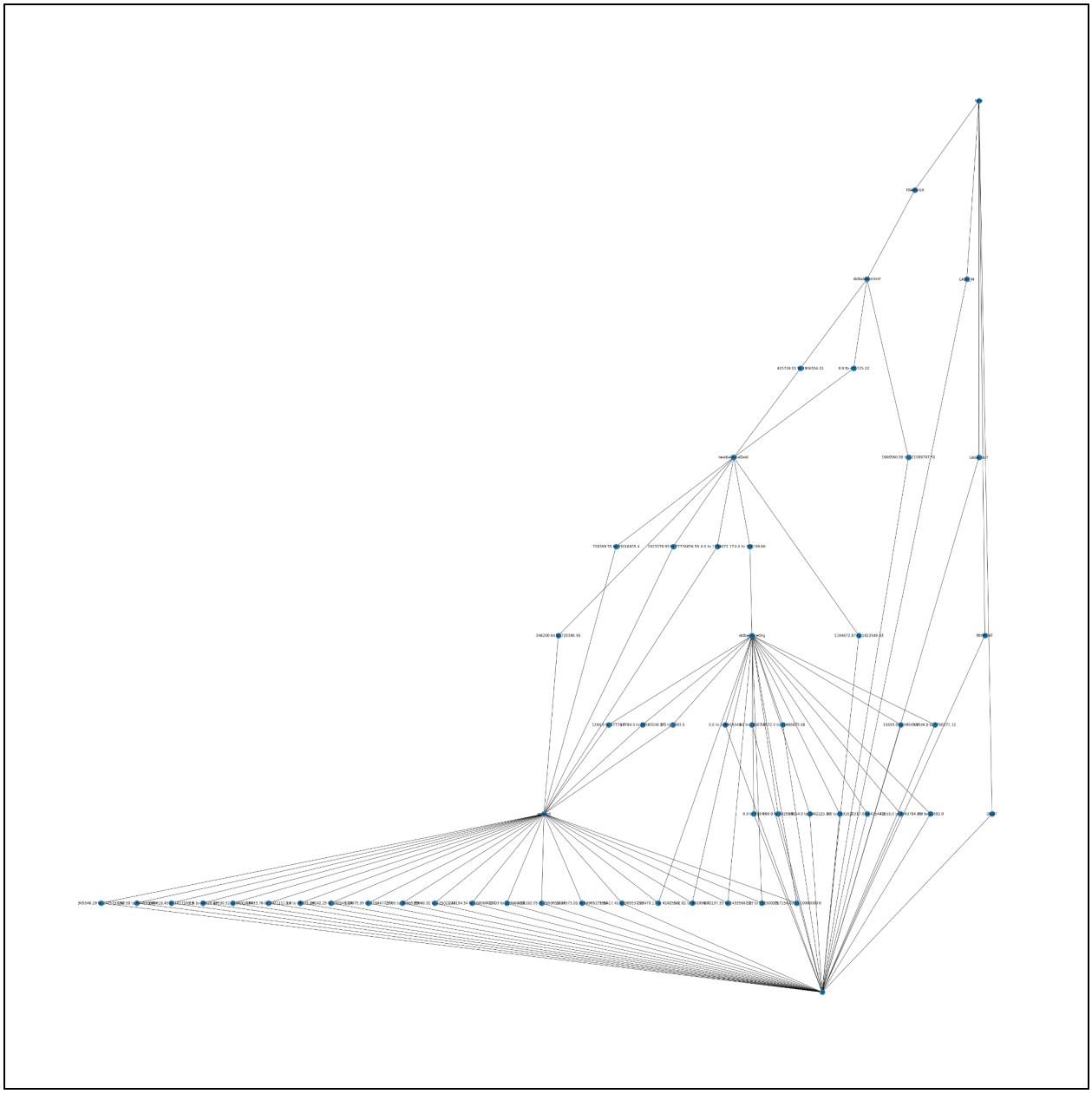
Increasing all the hyper parameters results in a naive decision tree which overfits the training data. And very less values result in underfitting. Hence the hyper parameters are tuned such that the learning is balanced.

However, the fraud detection dataset is highly unbalanced due to rows with fraud data are less than 1%

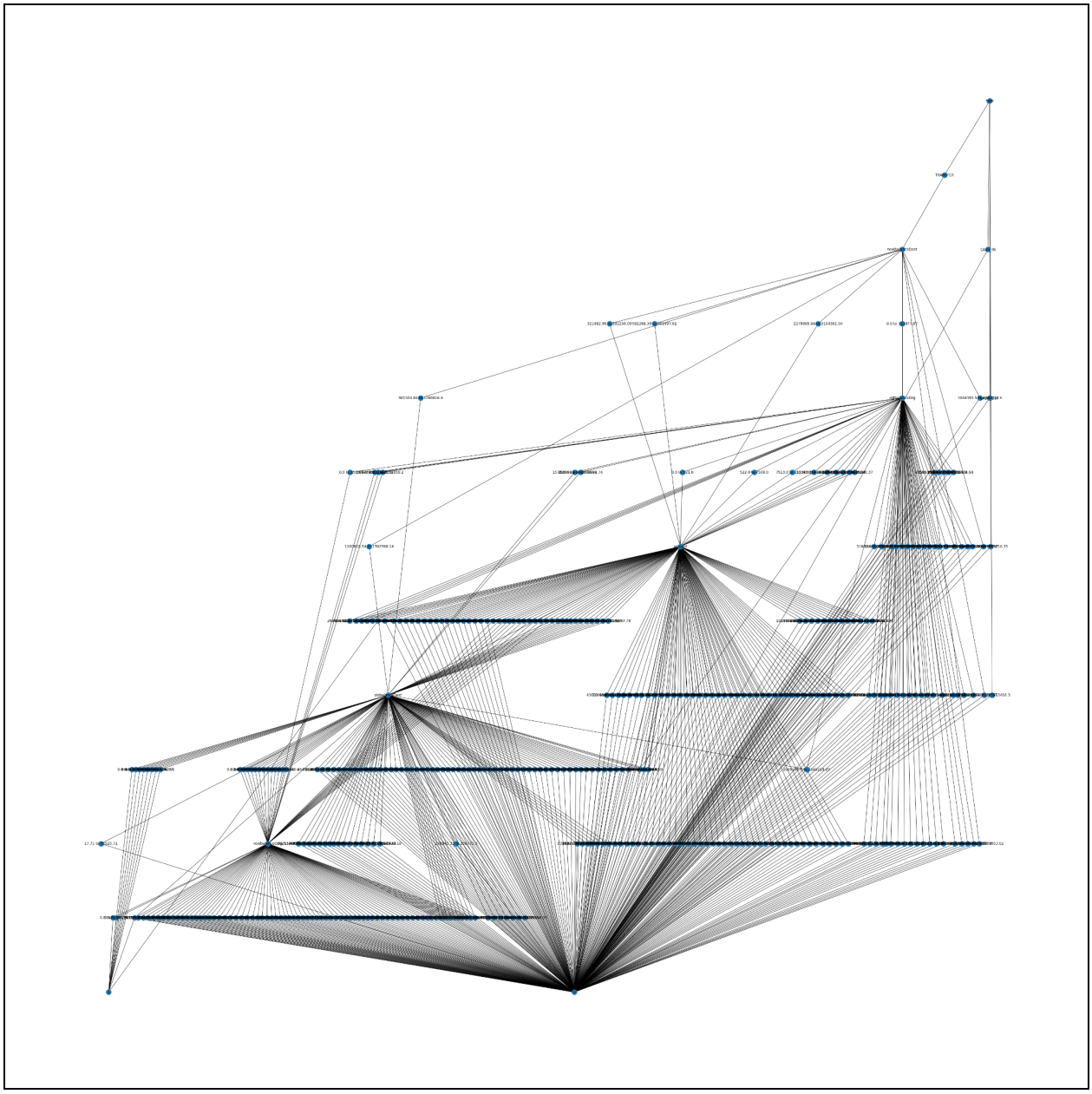
Train Test Split 80:20
bucket_count = 4
depth_limit = 8
nodes_limit = 100



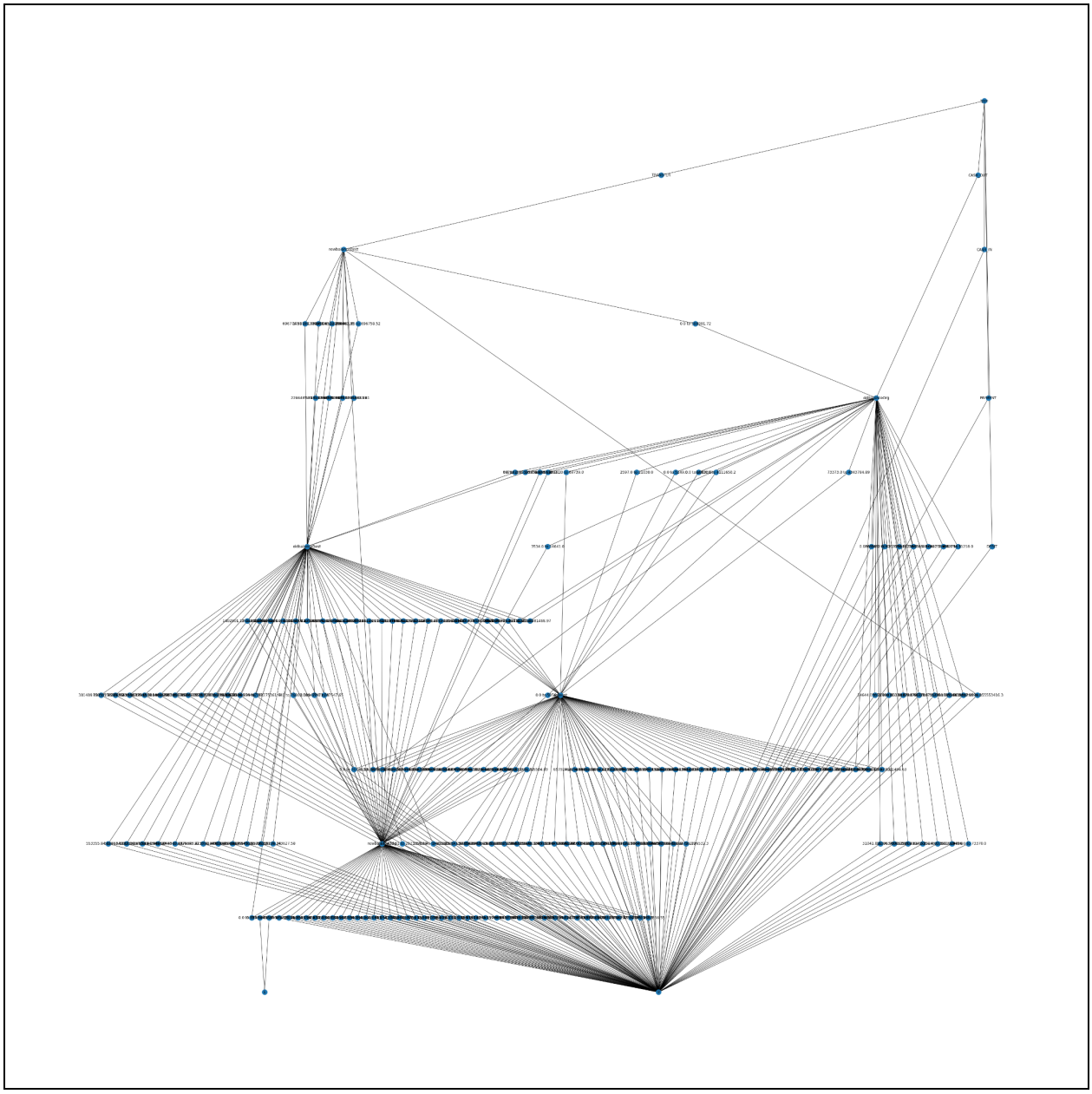
Train: Test Split 80:20
bucket_count = 3
depth_limit = 5
nodes_limit = 50



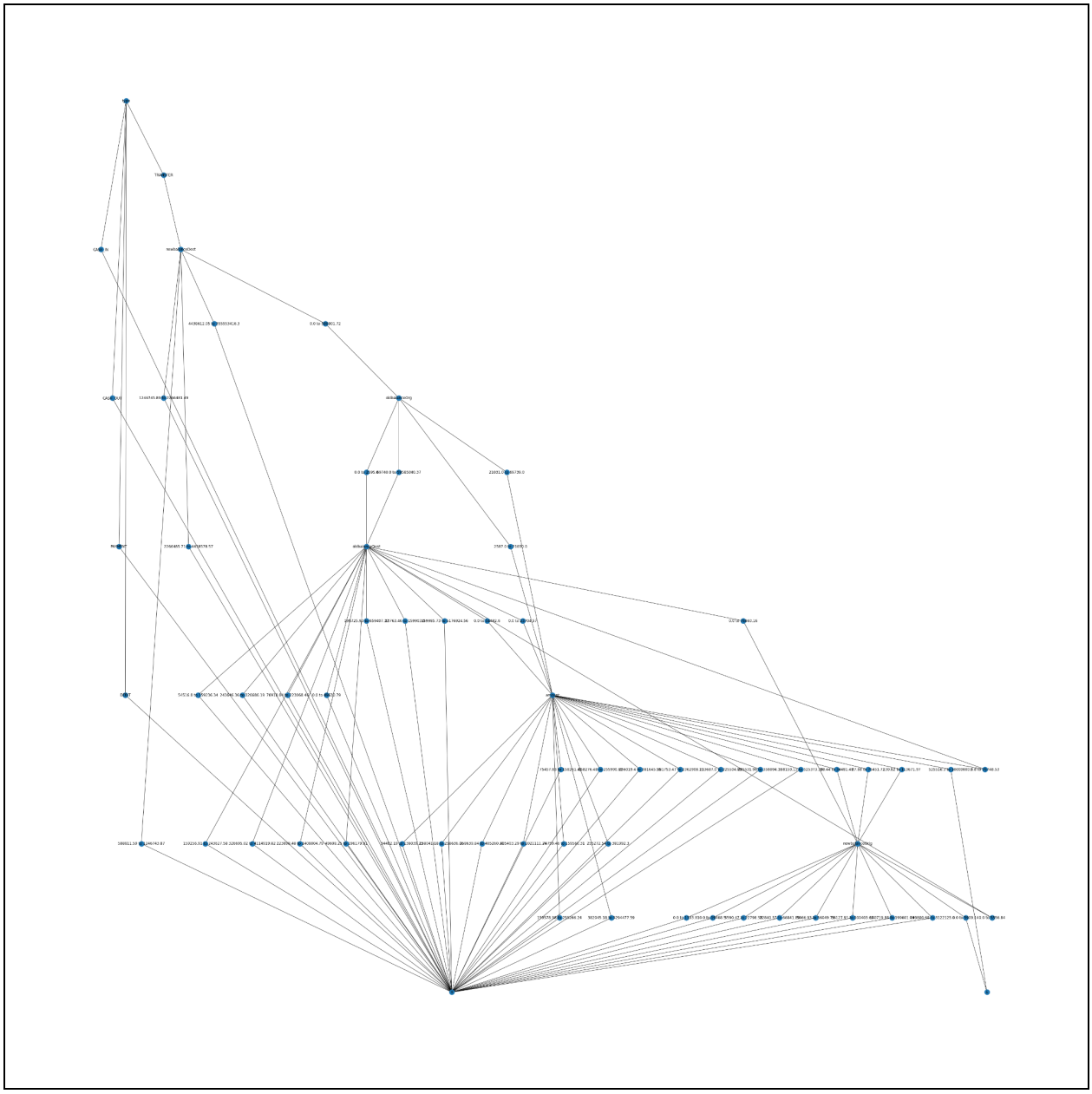
Train: Test Split 80:20
bucket_count = 10
depth_limit = 12
nodes_limit = 400



bucket_count = 5
depth_limit = 12
nodes_limit = 200
Train Test Split of 90:10



bucket_count = 5
depth_limit = 6
nodes_limit = 50
Train Test Split of 90:10



bucket_count = 3
depth_limit = 3
nodes_limit = 50
Train Test Split of 100:0