

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

Log file for all experiments

Team Members:

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

Experiments

1. Vary the hyper parameters: buckets for continuous variables, depth and number of nodes and find the impact on decision tree creation
2. Split train and test data into different ratios and find the metrics: accuracy, precision, recall, F1score, confusion matrix and area under ROC.
3. Comparison between custom decision tree implementation and MLlib classifiers
4. Feature ranking based on information gain

Experiment 1 and 2 results:

The graphical representation of decision trees are captured in the file : LogFile-Images

Test & Train Split (Train:Test) Ratio	Hyper Parameters used for building decision tree	Metrics
90 : 10	bucket_count = 5 depth_limit = 6	Area under ROC for test data: 0.53855 Precision 1.0

	<p>nodes_limit = 50</p> <p>Took 3.26 minutes to build</p>	<p>Recall 0.07711138310893513</p> <p>F1 measure 0.9999970079462085</p> <p>Accuracy 0.9988148828085706</p> <p>Confusion Matrix</p> <table><tr><th></th><th>predict ▲</th><th>label ▲</th><th>count ▲</th></tr><tr><td>1</td><td>1</td><td>1</td><td>63</td></tr><tr><td>2</td><td>0</td><td>1</td><td>754</td></tr><tr><td>3</td><td>0</td><td>0</td><td>635407</td></tr></table>		predict ▲	label ▲	count ▲	1	1	1	63	2	0	1	754	3	0	0	635407
	predict ▲	label ▲	count ▲															
1	1	1	63															
2	0	1	754															
3	0	0	635407															
90 : 10	<p>bucket_count = 5</p> <p>depth_limit = 12</p> <p>nodes_limit = 200</p> <p>Took 14.41 minutes to build</p>	<p>Area under ROC for test data: 0.50307</p> <p>Precision 1.0</p> <p>Recall 0.006142506142506142</p> <p>F1 measure 0.999959551646248</p> <p>Accuracy 0.9987284152816505</p> <p>Confusion Matrix</p> <table><tr><th></th><th>predict ▲</th><th>label ▲</th><th>count ▲</th></tr><tr><td>1</td><td>0</td><td>1</td><td>809</td></tr><tr><td>2</td><td>0</td><td>0</td><td>635400</td></tr><tr><td>3</td><td>1</td><td>1</td><td>5</td></tr></table>		predict ▲	label ▲	count ▲	1	0	1	809	2	0	0	635400	3	1	1	5
	predict ▲	label ▲	count ▲															
1	0	1	809															
2	0	0	635400															
3	1	1	5															
80 : 20	<p>bucket_count = 3</p> <p>depth_limit = 5</p> <p>nodes_limit = 50</p>	<p>Area under ROC for test data: 0.5</p> <p>Precision 0.1</p> <p>Recall 0.0</p> <p>F1 measure 0.0</p> <p>Accuracy 0.9987090017643119</p> <p>Confusion Matrix</p> <table><tr><th></th><th>predict ▲</th><th>label ▲</th><th>count ▲</th></tr><tr><td>1</td><td>0</td><td>1</td><td>1642</td></tr><tr><td>2</td><td>0</td><td>0</td><td>1270242</td></tr></table>		predict ▲	label ▲	count ▲	1	0	1	1642	2	0	0	1270242				
	predict ▲	label ▲	count ▲															
1	0	1	1642															
2	0	0	1270242															

80 : 20	bucket_count = 4 depth_limit = 8 nodes_limit = 100	Area under ROC for test data: 0.50335 Precision 1.0 Recall 0.006699147381242387 F1 measure 0.9999629332014474 Accuracy 0.9987176493435324 Confusion Matrix <table><tr><th></th><th>predict ▲</th><th>label ▲</th><th>count ▲</th></tr><tr><td>1</td><td>0</td><td>1</td><td>1631</td></tr><tr><td>2</td><td>0</td><td>0</td><td>1270241</td></tr><tr><td>3</td><td>1</td><td>1</td><td>11</td></tr></table>		predict ▲	label ▲	count ▲	1	0	1	1631	2	0	0	1270241	3	1	1	11				
	predict ▲	label ▲	count ▲																			
1	0	1	1631																			
2	0	0	1270241																			
3	1	1	11																			
80 : 20	bucket_count = 10 depth_limit = 12 nodes_limit = 400 Took 24.02 minutes to build	Area under ROC for test data: 0.53455 Precision 0.027076222980659842 Recall 0.07247259439707673 F1 measure 0.027076222980659842 Accuracy 0.9954406219435106 Confusion Matrix <table><tr><th></th><th>predict ▲</th><th>label ▲</th><th>count ▲</th></tr><tr><td>1</td><td>1</td><td>1</td><td>119</td></tr><tr><td>2</td><td>0</td><td>1</td><td>1523</td></tr><tr><td>3</td><td>1</td><td>0</td><td>4276</td></tr><tr><td>4</td><td>0</td><td>0</td><td>1265966</td></tr></table>		predict ▲	label ▲	count ▲	1	1	1	119	2	0	1	1523	3	1	0	4276	4	0	0	1265966
	predict ▲	label ▲	count ▲																			
1	1	1	119																			
2	0	1	1523																			
3	1	0	4276																			
4	0	0	1265966																			

An increase in all the three hyper parameters shows a decrease in precision, recall and consequently a decrease in F1 measure. This is attributed to overfitting due to increased branching in the decision tree as can be seen in the corresponding figure for bucket_count = 10 ,depth_limit = 12 and nodes_limit = 400. By executing build tree for various depth values, a depth of around 6-8 gave good results when trained with a data split of 90:10 ratio.

The data split of 90:10 trained the decision tree better compared to 80:20 or 70:30. This is due to the fact that the data is highly unbalanced with fraud data of less than 1% in our dataset. With a more balanced dataset other data splits for training could be considered.

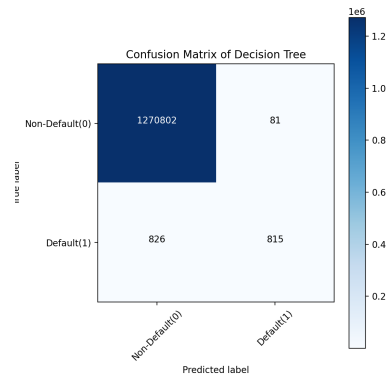
Continuous variables were divided into a range of values or buckets based on quantiles. And more the buckets, more is the number of child nodes in the graph representation. If buckets were increased the nodes limit also had to be relaxed to avoid underfitting in some branches in the decision trees. Overall a bucket count of 5 and nodes limit of 50 gave a balanced decision tree.

Note: nodes in our graphical representation not only include the attributes but also each range for continuous variables and unique values for discrete values. Hence a high number of nodes in our representation corresponds to a very low number of nodes in a normal decision tree representation. (For example, 60 nodes correspond to around 8-10 nodes in a normal decision tree representation)

Overall, the precision is very high and recall is low. This is also the reason for a low area under ROC value around 0.54 which shows a very low discrimination ability. The decision tree could be further improved with pruning techniques for higher ROC values along with obtaining a more balanced dataset.

Experiment 3 Results:

Implementing standard classifiers using Built in Libraries for comparison:

Classifier	Test & Train Split (Train:Test) Ratio	Metrics
Decision Tree	80 : 20	Accuracy: 0.9992872433054307 F1 Score: 0.6424911312573907 Confusion Matrix: 
Decision Tree	90 : 10	Accuracy: 0.9988825358107195 F1 Score: 0.6511266511266511 Confusion Matrix: 0.24442082890541977

		<p>Confusion Matrix of Decision Tree</p> <table><tr><td>Non-Default(0)</td><td>635394</td><td>42</td></tr><tr><td>Default(1)</td><td>407</td><td>419</td></tr><tr><td></td><td>Non-Default(0)</td><td>Default(1)</td></tr></table> <p>True label</p> <p>Predicted label</p> <p>Color scale: 0 to 600,000</p>	Non-Default(0)	635394	42	Default(1)	407	419		Non-Default(0)	Default(1)
Non-Default(0)	635394	42									
Default(1)	407	419									
	Non-Default(0)	Default(1)									
Random Forest	80 : 20	<p>Accuracy: 0.9988951092474484</p> <p>F1 Score: 0.25053304904051177</p> <p>Confusion Matrix:</p> <p>Confusion Matrix of Random Forest Tree</p> <table><tr><td>Non-Default(0)</td><td>1270883</td><td>0</td></tr><tr><td>Default(1)</td><td>1406</td><td>235</td></tr><tr><td></td><td>Non-Default(0)</td><td>Default(1)</td></tr></table> <p>True label</p> <p>Predicted label</p> <p>Color scale: 0.0 to 1.2e6</p>	Non-Default(0)	1270883	0	Default(1)	1406	235		Non-Default(0)	Default(1)
Non-Default(0)	1270883	0									
Default(1)	1406	235									
	Non-Default(0)	Default(1)									
Random Forest	90 : 10	<p>Accuracy: 0.9988825358107195</p> <p>F1 Score: 0.24442082890541977</p> <p>Confusion Matrix:</p> <p>Confusion Matrix of Random Forest Tree</p> <table><tr><td>Non-Default(0)</td><td>1270883</td><td>0</td></tr><tr><td>Default(1)</td><td>1406</td><td>235</td></tr><tr><td></td><td>Non-Default(0)</td><td>Default(1)</td></tr></table> <p>True label</p> <p>Predicted label</p> <p>Color scale: 0 to 1,200,000</p>	Non-Default(0)	1270883	0	Default(1)	1406	235		Non-Default(0)	Default(1)
Non-Default(0)	1270883	0									
Default(1)	1406	235									
	Non-Default(0)	Default(1)									

Experiment 4 results:

Feature Ranking based on Info gain

Rank 1 - type
Rank 2 - oldbalanceOrg
Rank 3 - amount
Rank 4 - newbalanceOrig
Rank 5 - oldbalanceDest
Rank 6 - newbalanceDest

Using the information gain it is seen that **type** attribute is highly ranked or is best suited to be chosen as the root node of the decision tree. In the below fig1. It can be seen that out of all the types of transactions: Payment, Cash in, Cash out, Transfer, Debit only Transfer and Cashout have fraudulent transactions (fig2.). From fig 1. It can be seen that Transfer makes for 8% of transactions in the data and Cash out corresponds to 35% of the transactions. Out of this collective 43% of transactions, 21.5% of the transactions are fraudulent! Hence intuitively, **type** attributes is a very good measure.

Also, surprisingly oldbalanceOrg which corresponds to original balance is higher ranked compared to amount of transaction. Intuitively the higher the original balance the higher chances of fraud involved. And by computing the average oldbalanceOrg it can be seen that the average is a high value of \$164,9667.6057116778.

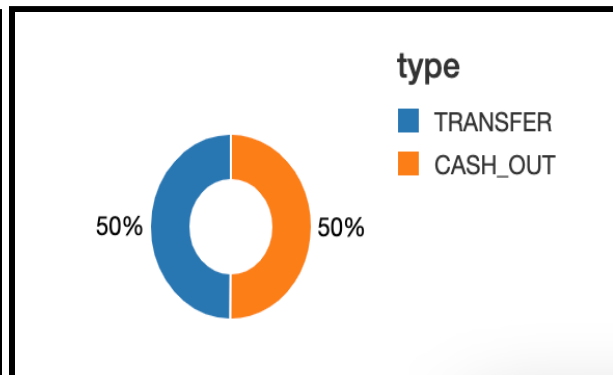
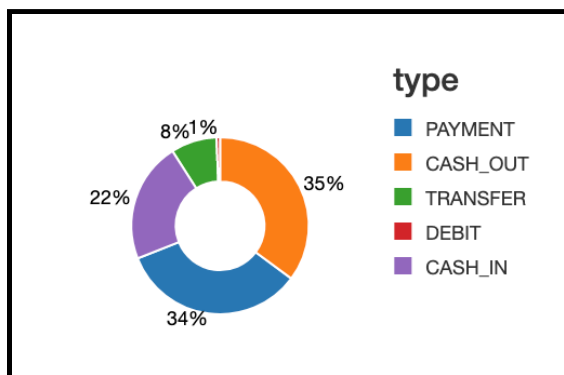


Fig1. Number of transactions of each type

Fig2. Number of fraudulent transactions