6. Implementation

6.1 Code

Random Sampling of the Data

Random sampling is one of the simplest forms of collecting data from the total population. Under random sampling, each member of the subset carries an equal opportunity to be chosen as a part of the sampling process.

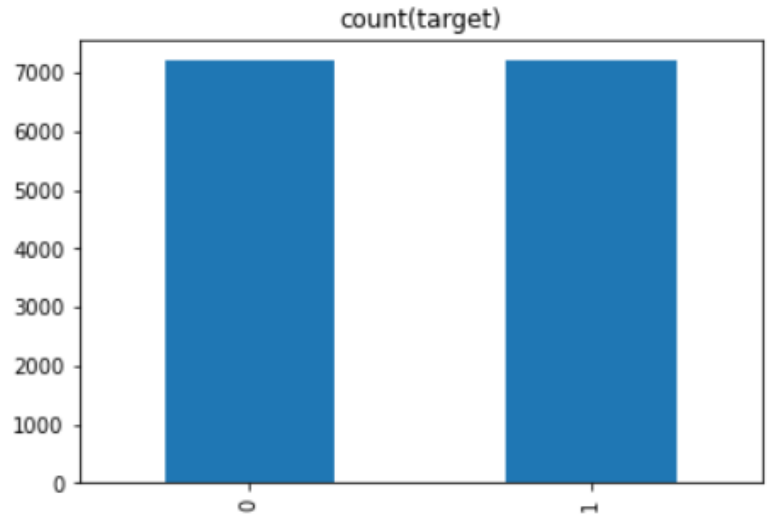
In our data, we have reduced the number of rows to 7200 for both the case i.e., Fraud class (1) and non-fraud class (0) from 587443 rows, such that both cases of fraudulence will be equally distributed among the data.

Visualize the data

Data from the above random sampling can be shown graphically as below:

```
#plot the new under sampling data
test_under['fraud'].value_counts().plot(kind = 'bar', title='count(target)')
```

```
total class of 1 and 0:
 0    7200
1    7200
Name: fraud, dtype: int64
<AxesSubplot:title={'center':'count(target)'}>
```



Classification using Random Forest

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different

samples and takes their majority vote for classification and average in case of regression. We have applied Random Forest method on our data as shown below:

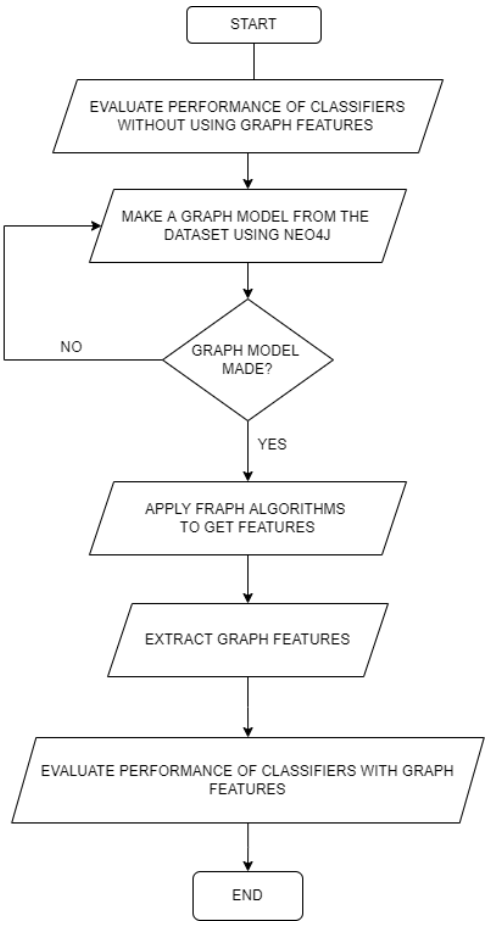
```
[ ]  from sklearn.ensemble import RandomForestClassifier
```

```
[ ]  X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=42)
     X_train.shape, X_test.shape
```

```
[ ]  rf = RandomForestClassifier()
     rf.fit(X_train,y_train)
     y_pred = rf.predict(X_test)
     print(accuracy_score(y_pred,y_test),f1_score(y_pred,y_test))

     y_ptrain = rf.predict(X_train)
     print(accuracy_score(y_ptrain,y_train),f1_score(y_ptrain,y_train))
```

6.2 Design Documentation and Flowchart



Flowchart showing methodology using graph model

7. Data Analysis and Discussion.

7.1 Output Generation

a. CSV to Graph model

the data with the columns as ‘step’, ‘customer’, ‘age’, ‘gender’, ‘zipcodeOri’, ‘merchant’, ‘zip merchant's, ‘category’, ‘amount’, ‘fraud’, ‘source’, ‘target’, ‘weight’, ‘type trans’ and ‘fraud’ from a comma separated value (.csv) file converted into Graph Model using NEO4J desktop. Using Cypher Query Language (CQL), we have generated the sample graph model of the data from the csv file using the below CQL query.

```
LOAD CSV WITH HEADERS FROM 'file:///banksim_reduced.csv' AS row

Merge (c:customer{name:row.customer,age:row.age,gender:row.gender})

merge (m:merchant{name:row.merchant})

CREATE (f:financialInstitute{Name:row.Financial_Institute})

merge (f)-[:Amount_Paid{name:row.amount}] -> (m)

merge (cat:category{name:row.category})

CREATE (ord:order{name:row.order})

merge (c) - [:purchases] -> (ord)

merge (ord) - [:From] -> (m)

merge (m) - [:supplies] -> (cat)

merge (ord) -[:type_of_category] -(cat)

merge (T:Transaction_ID{name:row.Transaction_ID})

merge (c) - [:Makes_transaction] - (T)

merge (ssn:ssn{name:row.SSN})

CREATE (add:address{name:row.Address})

merge (c) -[:has_an_ssn] -> (ssn)

merge (fraud:fraud_Trans{name:row.fraud})
```

merge(f) -[:Type{name:row.Transaction_ID}] -> (fraud)

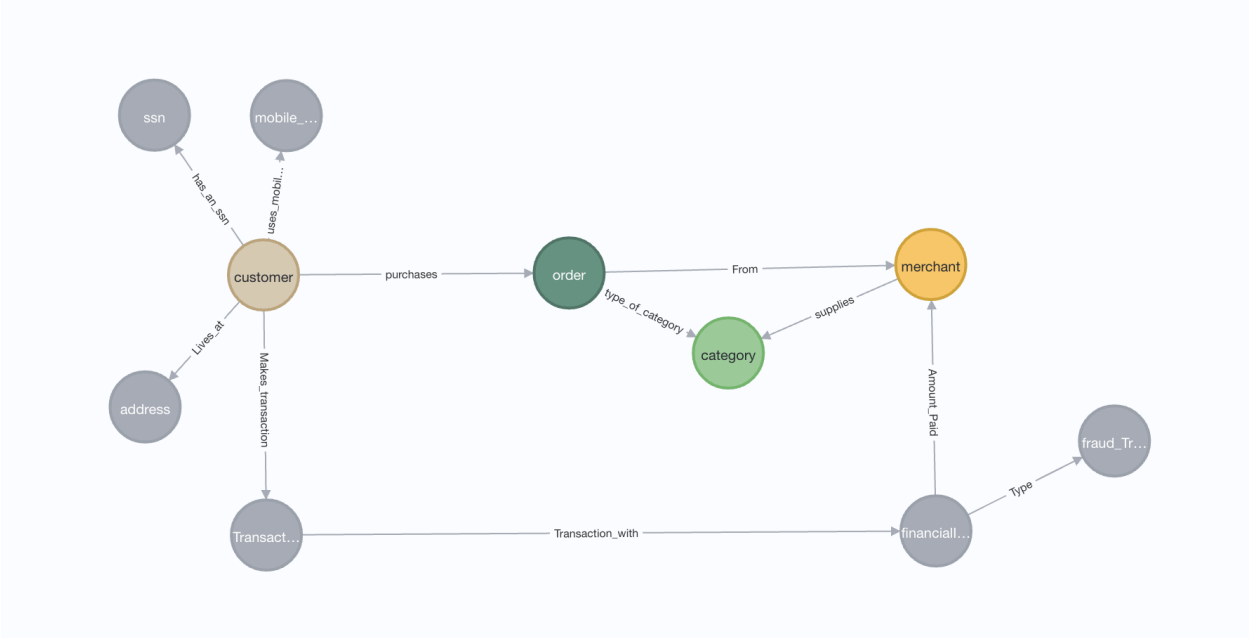
Merge(c) - [:Lives_at] -> (add)

merge(mobile:mobile_number{name:row.Mobile_number})

Merge (c) - [:uses_mobile_number] - (mobile)

merge (T) - [:Transaction_with] -> (f);

Schema (Meta graph)

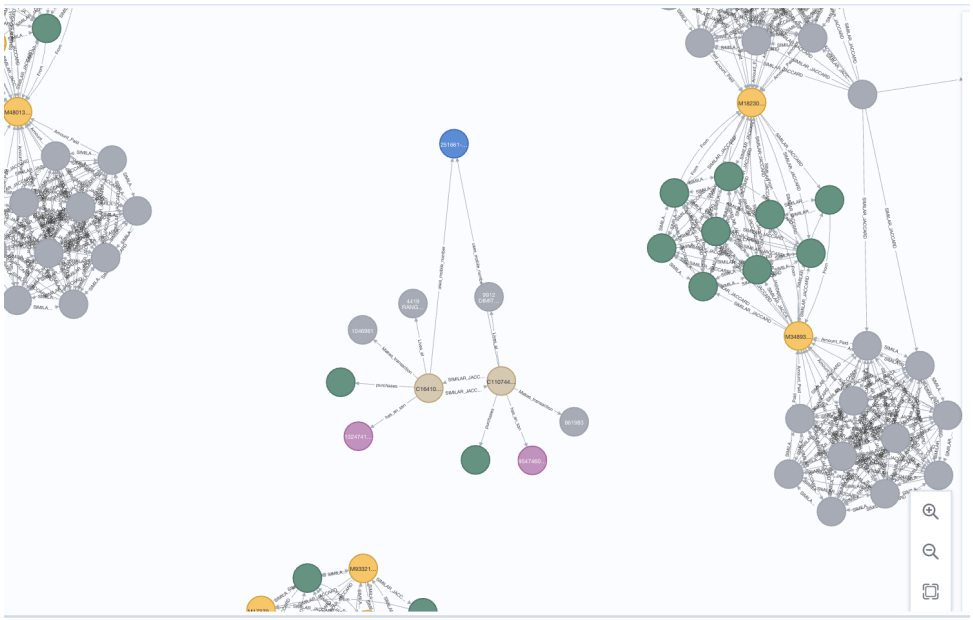


A meta-graph for the database helps to model the data in a virtual graph form with the interconnection between nodes and their relationships. The nodes of the graph represent the entities and the relationship represents the association among the nodes. A graph model is created by merging the desired features and making relationships among them as shown above. Here, a relationship ‘make’ is made from customer to transaction and another relationship ‘with’ is made from transaction to the financial institution to purchase order from merchant.

Using GDS tool:

After we extracted the graph medel out of the csv file we then ran the few graph algorithms on the in memory graph in order to extract the graph features and then run ML algorithms on the ML algorithms and evaluate the performance

comparison with and without the graph features. We used graph algorithms like PageRank, Label Propogation and Degree centrality. After extracting the graph feature we get a few new features apart from the features in the dataset. The algorithms wher run in neo4j browser as well as GDS tool. here are a few snaps of the output.



Output after running Similarity algorithm

Ask rahul to share the LPA output showing 0 and 1 nodes.

This graph shows two distinct communities “0” for non-fraud and “1” for fraud customers and their related information.

LPA code include in appendices

```
:param limit => ( 42);

:param config => ({
  relationshipWeightProperty: null
});

:param communityNodeLimit => ( 10);

:param graphConfig => ({
  nodeProjection: '*',
  relationshipProjection: {
    relType: {
      type: '*',
      orientation: 'UNDIRECTED',
      properties: {}
    }
  }
});

:param generatedName => ('in-memory-graph-1653371987187');
```

```
CALL gds.graph.create($generatedName, $graphConfig.nodeProjection, $graphConfig.relationshipProjection, {});
```

```
CALL gds.labelPropagation.stream($generatedName, $config) YIELD nodeId, communityId AS community
WITH gds.util.asNode(nodeId) AS node, community
WITH collect(node) AS allNodes, community
RETURN community, allNodes[0..$communityNodeLimit] AS nodes, size(allNodes) AS size
ORDER BY size DESC
LIMIT toInteger($limit);;
```

```
CALL gds.graph.drop($generatedName);
```

8. Conclusion

Thus, it may be considered that the inclusion of graph features from the graph algorithms helps to improve the detection of fraudulent transactions involved in any online transactions made with credit cards. In this project, we have implemented graph algorithms like Label Propagation to detect the fraud. In each case, the performance of the classification models is found to be improved with the incorporation of the graph features extracted from the graph database. The graph features included in this paper have shown significantly improved results as compared to other existing graph features.

The graph features have significant roles in increasing the evaluation metrics of the classification algorithms. With the large volume of data in the dataset, a

smaller number of fraud instances are present, which is very difficult to detect and it is intended to identify the same.

Recommendations for future studies.

Going further we can incorporate several pipelines and plugins through which we can make Neo4j to collaborate with python and application of everal Machine Learning algorithms.

include source code in appendices and send us the final pdf to submit