Project 1

IEEE-CIS Fraud Detection Report

By

Fahim Mahmood

Email: fahimmahmood@iut-dhaka.edu

▪ Abstract:

IEEE-CIS Fraud Detection competition on Kaggle is a binary classification problem where we have to generate the probability of a fraudulent transaction. Necessary preprocessing like dropping, merging, label encoding, etc. are done on the train and test dataset. Then, EDA is done on the train dataset. Finally, the LightGBM model is trained using the training dataset which gives an accuracy of 0.936113 i.e. 93.6% on the test dataset.

▪ Methodology:

Both the train dataset and test dataset are divided into two parts. First, I merged them using the left join with respect to the TransactionID feature. Then, I performed exploratory data analysis to understand the train dataset. Most of the columns had missing values. So, I dropped columns that had more than 50% missing data. I performed label encoding on categorical features. I plotted the feature distributions and also printed the necessary statistical inference of features in the console to visualize the train dataset.

Initially, train dataset had around 434 columns/features. After preprocessing, I was left with 217 features that I used for training. This report would end up being very lengthy if I try to explain all 217 features and its distribution. I am going to provide just 2 examples of EDA performed in my project.

Statistical data of C1, C2, C3, C4, C5, C6 features

```
---------ABOUT C1--------
count    590540.000000
mean         14.092458
std         133.569018
min           0.000000
25%           1.000000
50%           1.000000
75%           3.000000
max        4685.000000
Name: C1, dtype: float64
---------ABOUT C2--------
count    590540.000000
mean         15.269734
std         154.668899
min           0.000000
25%           1.000000
50%           1.000000
75%           3.000000
max        5691.000000
Name: C2, dtype: float64
```

```
---------ABOUT C3--------
count    590540.000000
mean          0.005644
std           0.150536
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max          26.000000
Name: C3, dtype: float64
---------ABOUT C4--------
count    590540.000000
mean          4.092185
std          68.848459
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max        2253.000000
Name: C4, dtype: float64
```

```
---------ABOUT C5--------
count    590540.000000
mean          5.571526
std          25.786976
min           0.000000
25%           0.000000
50%           0.000000
75%           1.000000
max         349.000000
Name: C5, dtype: float64
---------ABOUT C6--------
count    590540.000000
mean          9.071082
std          71.508467
min           0.000000
25%           1.000000
50%           1.000000
75%           2.000000
max        2253.000000
Name: C6, dtype: float64
```

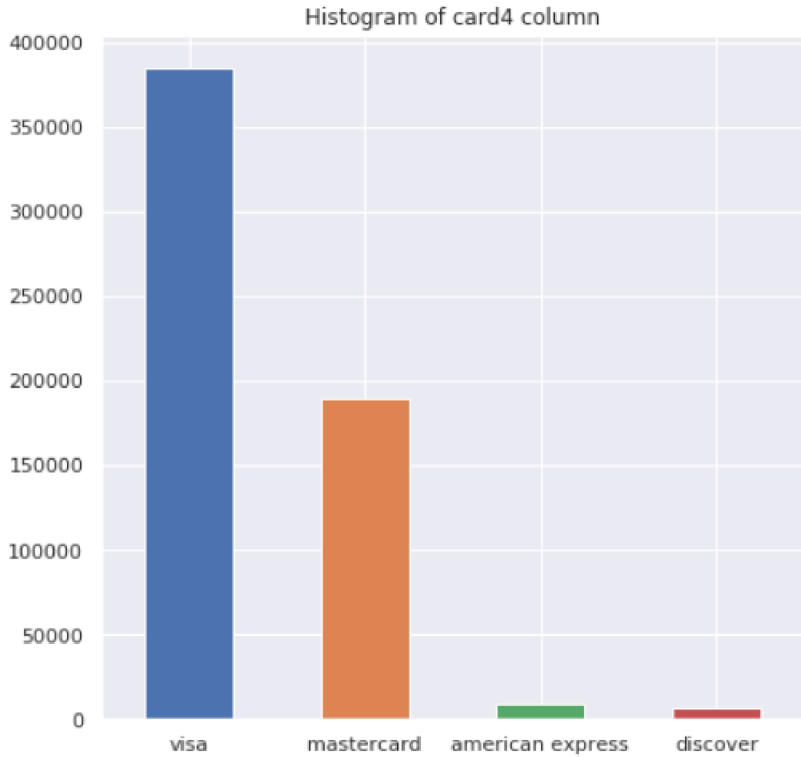


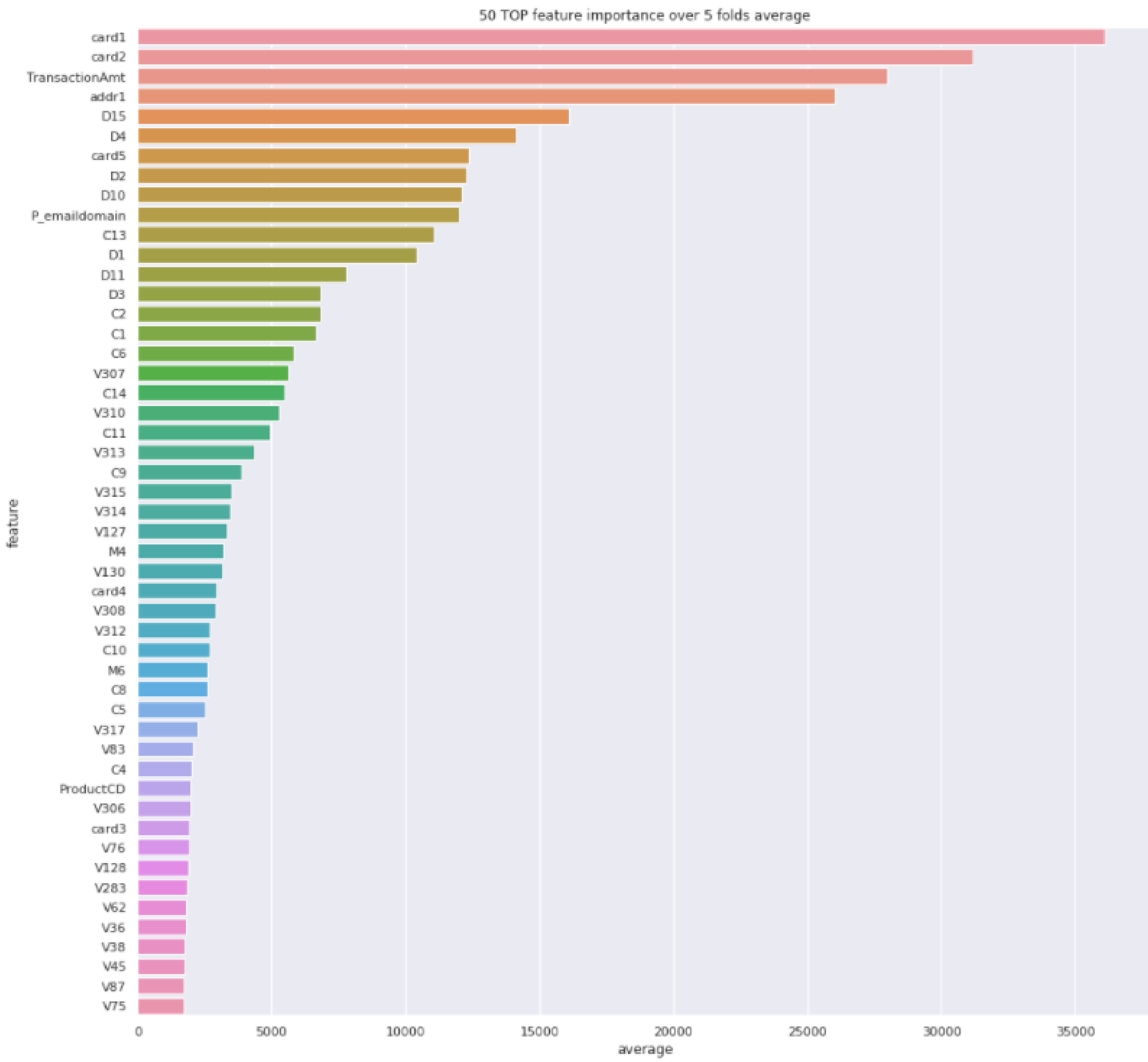
Figure: Histogram of different card users.

C1-C6 features weren't plotted as it needed normalization. Instead, I could get insights into C1-C6 by just printing the statistical data.

- ## Result Analysis:

  I used the LightGBM model for generating the probability as it is faster than most other classification models such as XGBoost, K-neighbours, Random Forest, Decision Trees, etc. Another advantage of LightGBM is that it can tolerate null values in the training dataset. LightGBM is similar to XGBoost as both use gradient boosting and they generate a prediction based on several 'weak learners' such as decision trees. The main difference between LightGBM and XGBoost is, LightGBM makes decision trees depth-wise (like DFS algorithm) whereas XGBoost makes decision trees breadth-wise (like BFS algorithm).

  The evaluation metric used is 'AUC' i.e. calculating the area under the curve of the ROC graph which is generated from the confusion matrix. I have used 217 features to train the LightGBM model and plotted the top 50 important features according to my model which is given below:



- **Conclusion:**

Even though my model's accuracy is 0.936113, it can be improved by a more sophisticated feature selection process like 'Recursive Feature Selection' and feature engineering. I also intend to implement other classification models like XGBoost, Logistic Regressor, K-neighbour's model on this dataset and compare the accuracy.