CREDIT CARD FRAUD DETECTION

(MACHINE LEARNING PROJECT REPORT)

•••

Group 20

Agnel Aaron (2017010), Anjanay Gour (2017021), Raunak Mokhasi (2017085)

Dataset

It contains details on real world e-commerce transactions through a wide variety of features (590,540 rows and 434 features). These features are anonymized and there is no Personally Identifiable Information.

We have to predict a binary attribute (isFraud) that denotes the probability of an online transaction being fraudulent.

The major difficulties that we noticed in the dataset right away was that it was extremely skewed. Now this is realistic because only a very small percentage of all credit card transactions are fraudulent.

Thus we had to use random undersampling techniques in order to solve this issue. There were a lot of outliers and NaN values so those had to handled as well.

Exploratory Data Analysis

On examining the features and plotting all the relationships as follows:

1) Desktop is being used more than Mobile for Transactions. Mobile and Desktop have relatively same number of Fraud Transactions but in Mobile comparatively the proportion is more.

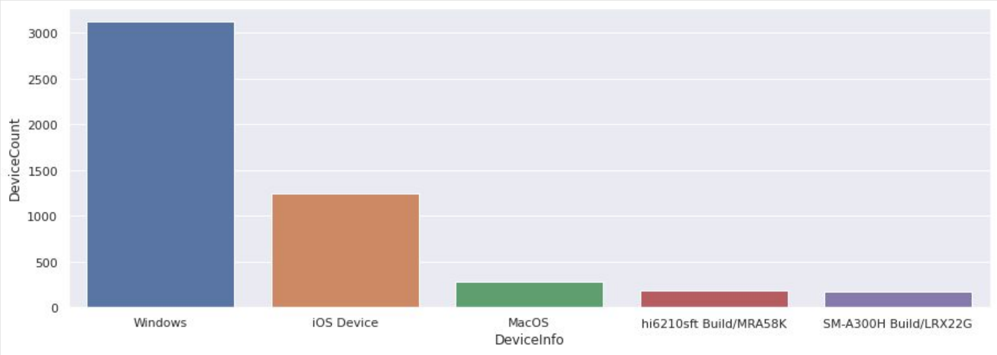
2) C and W Products have the highest number of Fraud Transactions.

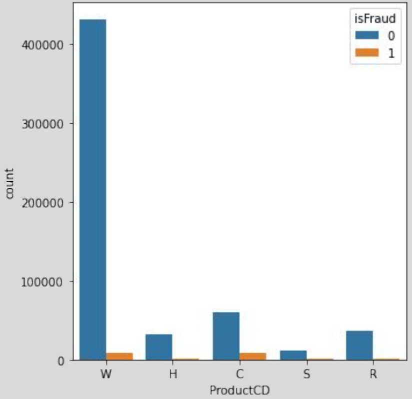
3) Most transactions and frauds take place from debit card.

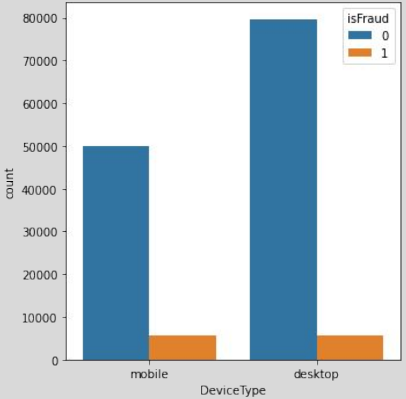
4) The plots below include these points and also show the count of Operating Systems, Fraud per Operating System, Counts and Types of card used in the transaction, Purchaser and Recipient email domain.

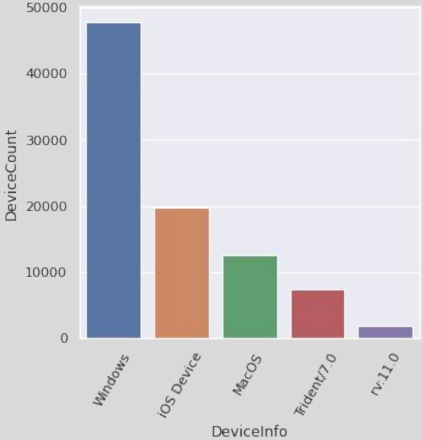
5) More plots have been attached in the submission.

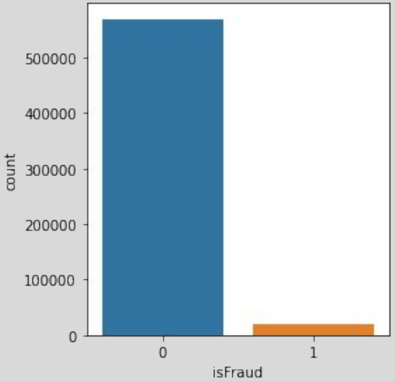
Reference Plots from EDA

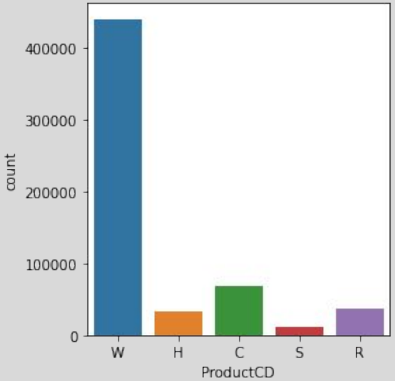


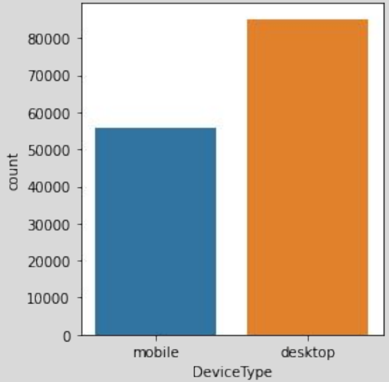












Model Methodology (Decision Tree) Model-1

- At first we used a Decision Tree in order to classify the Transaction as Fraudulent. It chose 4 particular features (for example "TransactionAmt", "ProductCD", "card4", "isFraud").

- Then we used Stratified Sampling i.e Splitting the data set to ensure that the train and test sets have approximately the same percentage of samples of each target class as the complete set. We also changed all the categorical attributes to numerical values. After that we ran the Decision Tree Classifier Algorithm, and it gave us a 69.45% Test Accuracy.

Model Methodology (RandomForest) Model-2

- To combine the multiple models and give a better AUC accuracy. We planned to use RandomForest as each tree in the forest is trained on a random subset of the data points with replacement (bagging / bootstrap aggregating).

- The RandomForestRegressor gave a higher AUC than RandomForestClassifier due to its probabilistic nature, given values between 0 and 1, and also due to the fact that we have to predict the probability of whether the transaction is fraudulent and not exactly if it is fraudulent or not (i.e. it is more of a regression than a classification). The best AUC accuracy we got for RandomForestRegressor was 76.65% for 4 Features used in the previous decision tree model and 88.99 for all features in the dataset.

Model Methodology (XGBoost) Model-3

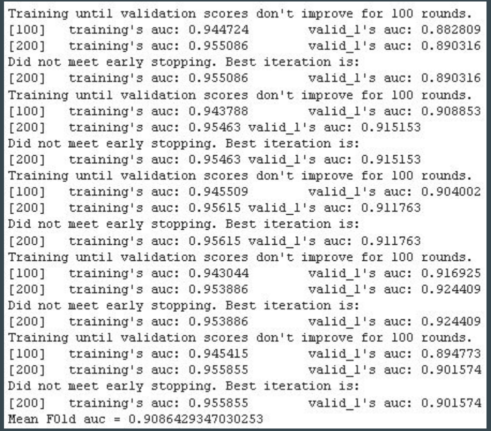
Looking at the positive response from the ensemble technique of RandomForestRegressor we decided to try XGBoost (eXtreme Gradient Boosting).

It worked by combining the predictions from several models into one by taking the predictors sequentially and modelling it based on the predecessor’s error.

Then it gave a few features (like TransactionDT) a higher weight based on whether it performed better. We used GridSearchCV in order to hypertune the parameters to get as high accuracy as possible.

Here we got an accuracy of 91% due to the appropriate combination of features and model.

Model Methodology (LightGBM) Model-4



```
Training until validation scores don't improve for 100 rounds.
[100]   training's auc: 0.944724        valid_l's auc: 0.882809
[200]   training's auc: 0.955086        valid_l's auc: 0.890316
Did not meet early stopping. Best iteration is:
[200]   training's auc: 0.955086        valid_l's auc: 0.890316
Training until validation scores don't improve for 100 rounds.
[100]   training's auc: 0.943788        valid_l's auc: 0.908853
[200]   training's auc: 0.95463 valid_l's auc: 0.915153
Did not meet early stopping. Best iteration is:
[200]   training's auc: 0.95463 valid_l's auc: 0.915153
Training until validation scores don't improve for 100 rounds.
[100]   training's auc: 0.945509        valid_l's auc: 0.904002
[200]   training's auc: 0.95615 valid_l's auc: 0.911763
Did not meet early stopping. Best iteration is:
[200]   training's auc: 0.95615 valid_l's auc: 0.911763
Training until validation scores don't improve for 100 rounds.
[100]   training's auc: 0.943044        valid_l's auc: 0.916925
[200]   training's auc: 0.953886        valid_l's auc: 0.924409
Did not meet early stopping. Best iteration is:
[200]   training's auc: 0.953886        valid_l's auc: 0.924409
Training until validation scores don't improve for 100 rounds.
[100]   training's auc: 0.945415        valid_l's auc: 0.894773
[200]   training's auc: 0.955855        valid_l's auc: 0.901574
Did not meet early stopping. Best iteration is:
[200]   training's auc: 0.955855        valid_l's auc: 0.901574
Mean F0ld auc = 0.9086429347030253
```

In this model we use decision tree with LightGBM boosting algorithm.

In this model we decided to use a data mining technique named feature engineering in order to extract more useful information from the raw_data.

We have performed feature engineering on columns : Card1-Card-5 and addr1-addr2.
Uid 1 = Card 1 + Card 2, Uid 2= Card 1 + Card 2 + Card 3 +Card 5,
Uid 3 = Card 2 + Addr 1 + Addr 2, Uid 4 = Card 3 + Card 5

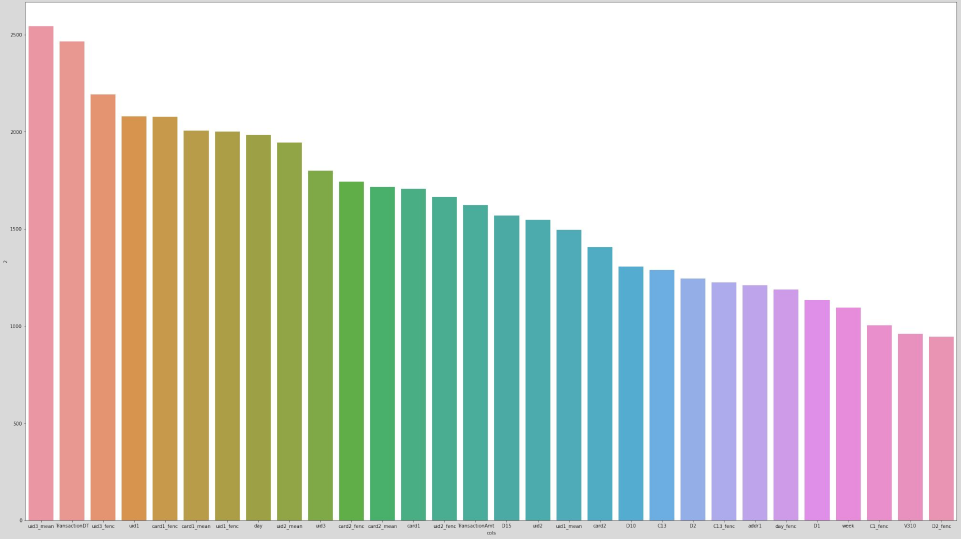
After performing feature engineering we used LightGBM boosting algorithm. It is better than XGBoost as it has lesser computation time and better accuracy.

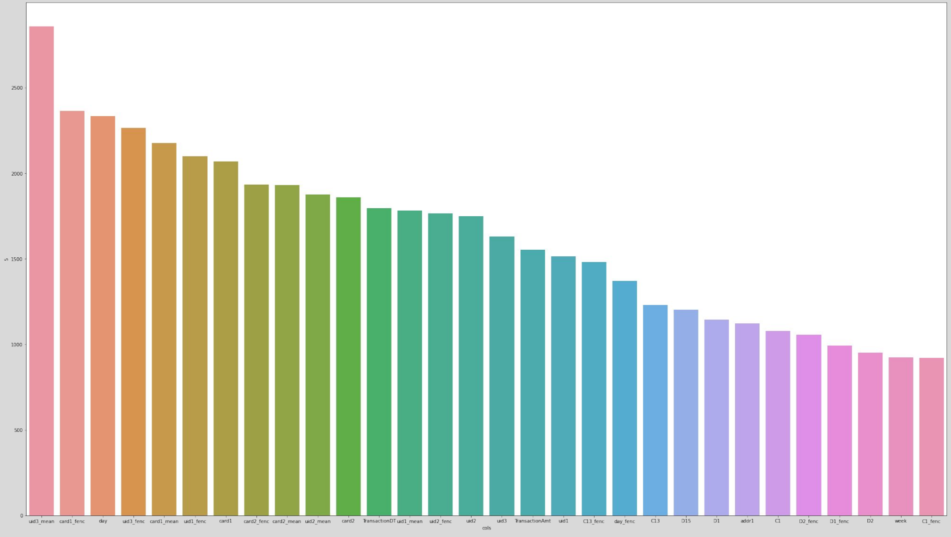
We used K-fold Cross Validation and then checked the prediction accuracy which turned out to be 92.63% .

Feature Importance

We split the P and R_emailDomain column into 2 new columns each. Reason for this is fraud transactions might have a connection to the extension or the domain they can use.

We feel TransactionDT (time delta) is important fraud transactions can be linked to the time of day or maybe fraud transactions could be grouped in day number or done in the same week etc.





Thank You!