**MS4610 – INTROUCTION TO DATA ANALYTICS**

**PROJECT REPORT**

**GROUP 21**

**TEAM MEMBERS : 1) Bhavadeep Bhukya**

**2) PSV Sai Teja**

**3) Sandeep**

**4) Hemanth**

**PROJECT : PABUDDY**

**Our project was given with 720000 customer ids for training and 180000 unseen data for testing.**

The given project was to build a propensity model for issuing credit cards in a company called paybuddy.

The given problem statement is modelled as Binary Classification. Various Machine learning techniques for the class of Binary Classification like Decision Tree, Logistic Regression, Random Forest and Boosting methods like Gradient Boosting can be used. The given dataset has target variables labeled as 1 or 0 which indicates defaulters and non-defaulters.

We are given with 7 different datasets with different customer identities and their own features and there were many nan values in 7 columns of the whole 7 datasets with more than 95% of data missing.

|  |  |
| --- | --- |
| Data set | % of Nan values |
| fraud\_update\_features | 27% |
| new\_acc\_fi | 0% |
| new\_acc\_memo\_rcv | 0% |
| new\_acc\_memo\_sent | 0% |
| new\_acc\_reciever | 0% |
| New\_acc\_sender | 0% |
| Payment\_cashout\_fail |  |

In fraud\_update\_features some columns are with 0 nan values and some are with more than 95% nan values so we removed them from the data set as we cannot completely re write a column.

Later we merged them with respect to the customer identities in the train\_df. After merging we get many number of Nan values which need to be changed to train them to get some accurate results.

**Manipulating Nan values :**

In fraud\_update\_features entries were either 1 or 0.

So while re-writing nan values we used mode function to change them.

In all other files we changed nan value with respective mean of that column using mean function.

**Correlated columns:**

Later we checked if there are columns with correlation greater than 0.9 and made them as one.

Graphical user interface, text, application

Description automatically generated

This functions gives a set as a output and we can remove the given columns in the train and test sets using drop function.

**TRAINING DATA:**

We used multiple algorithms starting from logistic Regression to boosting algorithms.

In boosting algorithms we used xgboost ( described later ) which gave a better result in less time compared to other algorithms.

After cleaning data we use train\_test\_split function to split train data to a percentage to check the accuracy for the remaining part. Example splitting 70 percent of train\_df for Training and 30 precent of train\_df for testing.

After complete training and cross-validation we used xgboost with different parameters to improve our model accuracy.

After complete training we did all the things to be done for test\_df which we did for train\_df and tested for the result. We used a function to get the probability of that customer Id being 1.

We finally got output of test\_df as a set and converted that to a dataframe as required.

**XGBOOST:** XGBOOST is an algorithm that has recently been dominating applied machine learning and Kaggle competitions for structured or tabular data.

XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.

It has given better accuaracy and F1 Score as compared to other algorithms like Logistic Regression and Random Forest.XGB is tree based learning algorithm.