Indian Institute of Technology Madras

Paybuddy

Final Project Report - Group 1

MS4610: Introduction to Data Analytics
Course Instructor: Dr. Nandan Sudarsanam

Akash Anand - ED18B002
Atharva Limkar - EP18B009
Gurmehar Singh - ED18B013
Neel Balar - ED18B019
Pranav Sharma - ED18B021
Shashank M Patil - CH18B022

Odd Semester Jul-Nov 2021

Contents

Introduction

PayBuddy is an online payments platform based out of the U.S. Its an Online Payments Platform that was split from PayBay in 2015 and became an independent company. PayBuddy facilitates a money transfer service for free of cost between any two PayBuddy users.

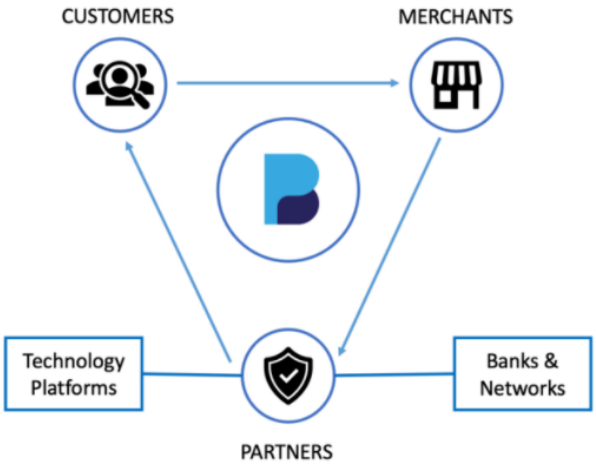


Figure 1.1: PayBuddy

1.1 Problem Statement

To ensure competitive advantage over competing brands, there is a need to build a **propensity model** to predict the customer propensity to apply for the new line of credit card called Evolution Card. The model could help the company with things such as - increasing net TPV and adoption of Evolution, providing optimal incentives and rewards, optimal customer targeting, minimizing credit risks, etc.

Data

2.1 Data Exploration

There are totally 9 data sets given, and 2 of them are train and test datasets. The remaining datasets contain information about each of the customer with the customer id as the unique identifier. The datasets given are:

- **fraud_update_features:** This dataset consists of 872890 entries and 20 features. Features 4, 5, 15, 16 are time series features and all the other features are binary and take 0/1 values. Features 4, 5, 15, 16, 17, 18, 19 have a large amount of missing data ($> 50\%$), so extrapolating data for those features would not make a lot of sense. As a result, we choose to drop those features from our analysis.

- **new_acct_fi:** This new account financial instrument consists of 569104 rows and 17 features. Most of the features are multiclass categorical and some are continuous. There are no missing values in this dataset, so there is no further data processing required.

- **new_acct_memo_rcv:** This dataset consists of 771148 rows and 52 features. There are no missing values in this dataset, so there is no further data processing required.

- **new_acct_memo_sent:** This dataset consists of 841975 entries and 52 features. There are no missing values in this dataset, so there is no further data processing required.

- **new_acct_receiver:** This dataset consists of 771148 entries and 80 features.There are no missing values in this dataset, so there is no further data processing required.

- **new_acct_sender:** This new account sender engagement consists of 841975 rows and 111 features. Most of the features are multiclass categorical and some are continuous. There are no missing values in this dataset, so there is no further data processing required.

- **payment_cashout_fail_df:** This dataset contains 49978 rows and 56 features. Most of the features are continuous and some are categorial. Payment cashout fail features 28, 42, 56 have some missing values. While there are a lot of missing values in this dataset, it is important to note that we don't want to drop any features. This is because we suspect that the number of defaulted transactions would be extremely important when it comes to the actual data. So, in this case, we assume that nan corresponds to 0.

- **train_df:** The train dataset consists of 720000 customer ids and their respective target classes. This along with the above 7 datasets is used for training.

- **test_cust_id:** The train dataset consists of 720000 customer ids and their respective target classes. This along with the above 7 datasets is used for training.

2.2 Data Preparation

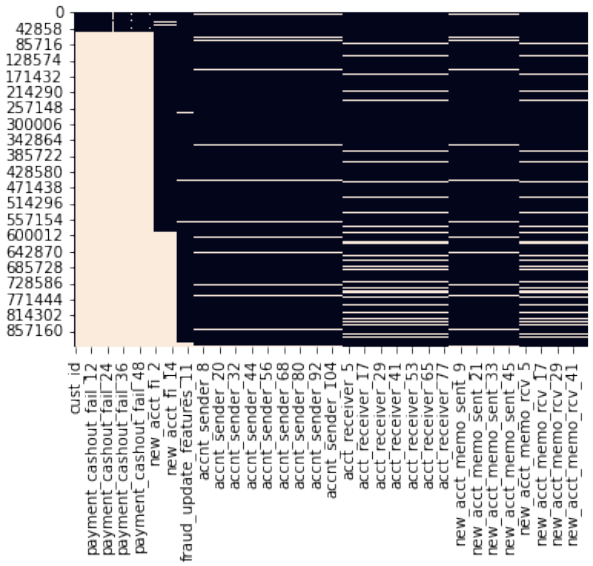
We first combine all of the above data frames into a single dataset. The merging operation is performed on the cust_id column. This is done using the OUTER merge operation. As outer merge operation gives the union of all the datasets, the dataset obtained contains all the data from the 7 datasets.

We divide this dataset into 2 datasets ("train" and "test") on the basis of the customer ids in Train_df and Test_cust_id. This is done using the INNER merge. The inner merge operation gives the intersection of the datasets. As the inner merge is performed on the cust_id column, the resulting dataset contains all the data for the customer ids of the respective datasets. We then analyse this dataset and proceed with the preparation.

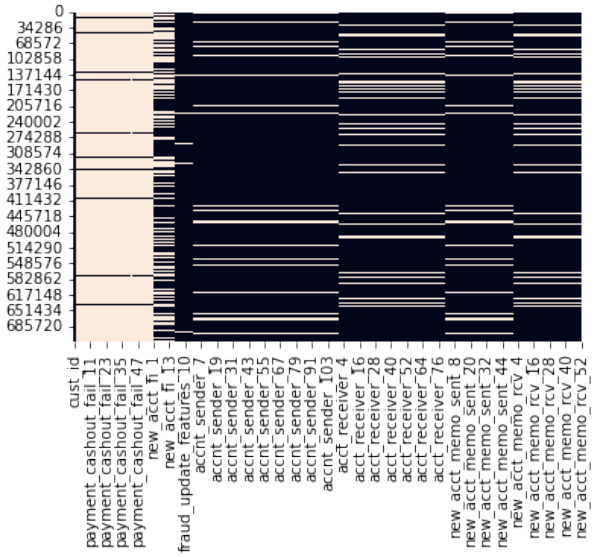
Figure 2.1 shows visual representations of the dataset. The colored spaces are the missing data. Clearly, we see that the payment_cashout_fail features contain an overwhelming amount of NaN values. We naturally choose to drop these features from our analysis.

We identify another important set of features from Figure 2.1, namely the new_account_fi features. We split the entire dataset into 2 parts, one for returning customers of PayBuddy, and one for new customers of PayBuddy. All the accounts with NaN new_account_fi features are considered to be new customers, while the remaining (what we refer to later as "old") customers are considered to be old customers.

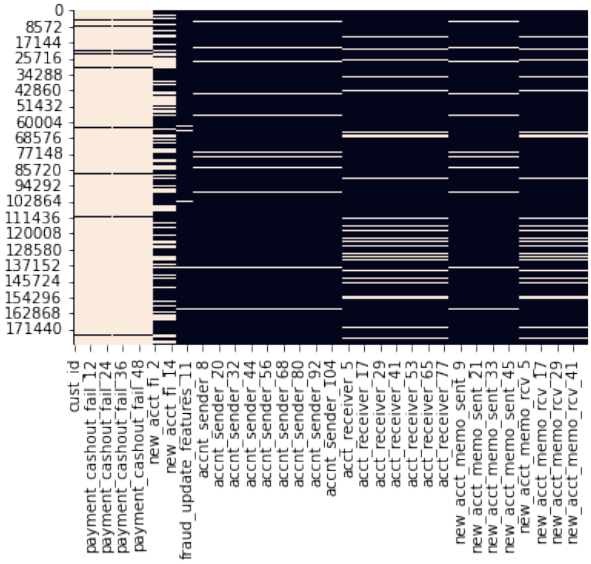
We decided to retain the new_account_fi features in this manner, simply because there are enough data points to warrant keeping the data somehow, but too many NaN data points to actually replace the missing values. The train and test data also are split in a similar fashion.



(a) Missing data in the complete dataset



(b) Missing data in the train dataset



(c) Missing data in the test dataset

Figure 2.1: Visual representation of the dataset

Model Building

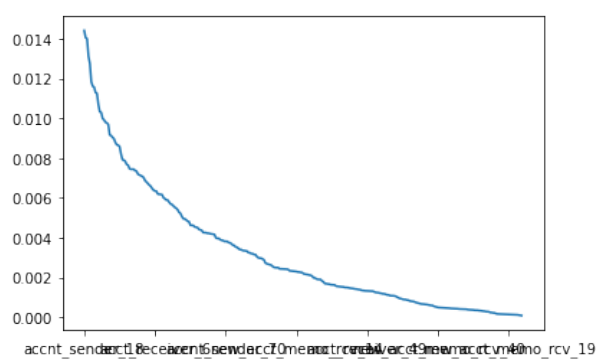
3.1 Random Forest Classifier - Without data preprocessing

- In order to have some sort of benchmark for what models could be, we decided to train a Random Forest Classifier on the entire dataset.
- We did not do any sort of splitting into new and old customers and dealt with missing values by replacing them with 0s.
- This model had very lacklustre performance (AUC score of 0.72). The large number of missing datapoints in several columns resulted in a great part of the dataset being 0.

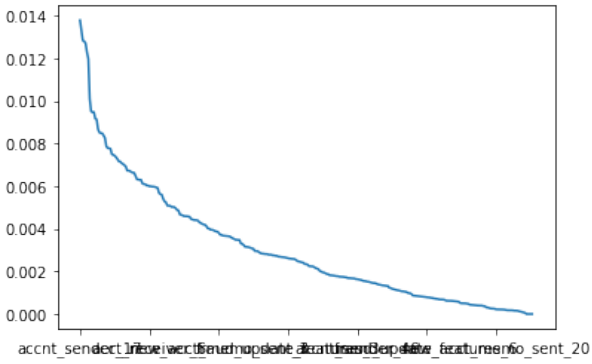
We decided to rework our data processing after seeing the performance of this model.

3.2 Random Forest Classifier - With partial data preprocessing

- We retrained the Random Forest Classifier with partially processed data. The data was processed by splitting the complete dataset into new and old customers. However, the payment_cashout_fail_df features were not dropped from the dataset.
- This partial processing was done in order to evaluate how well Random Forests would perform on the dataset.This model ran better than the previous model, with a higher overall score.
- However, this model was not very selective of the features that it used for the model, as shown in the feature importance curves in Figures 3.1. We had expected that the lowest importance features would all be given approximately the same value. This is because the features corresponding to payment_cashout_fail_df consist of over 90% 0 values.



(a) Random Forest feature importances in the new customer dataset



(b) Random Forest feature importances in the old customer dataset

Figure 3.1: Random Forest feature importances

3.3 Adaboost Classifier

- To begin with, the dataset is split into 2 components. The first component is used for training and consists of 75% of the data. The second component is used as validation data.
- The payment_cashout_fail and new_account_fi columns are dropped since they contain a big portion of missing values.
- For the resultant dataset, the missing values were substituted with zeros.
- The Adaboost classifier is fit with 100 estimators on the model, giving an AUC score of 0.719.
- Owing to the large computational time for this classifier and the sub-par performance compared to other models discussed in this document, hyper-parameter tuning was not carried out.

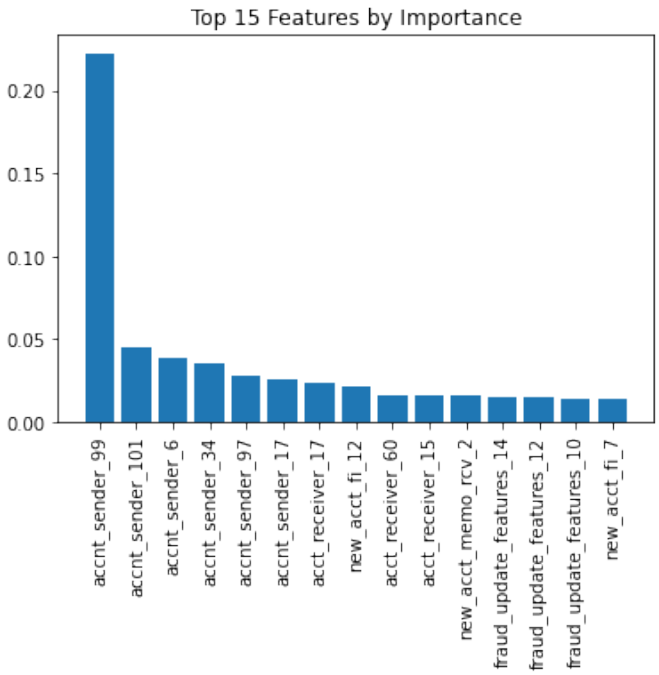
3.4 Lasso

- To begin with, the dataset is split into 2 components. The first component is used for training and consists of 75% of the data. The second component is used as validation data.
- The payment_cashout_fail and new_account_fi columns are dropped since they contain a big portion of missing values.
- For the resultant dataset, the missing values were substituted with zeros.
- As a base model, the dataset was trained with Lasso regression with the value of the parameter alpha = 1. This resulted in a AUC score of 0.73 on the validation dataset.
- After using 5-fold cross validation on the dataset and setting the maximum iterations as 10000 in order to achieve convergence, the appropriate value of alpha for Lasso regression was found to be alpha = 0.13.
- Fitting the model with Lasso regression using the newly obtained alpha = 0.13, the AUC score was seen to go up to 0.74.

3.5 XGBoost

- We begin testing this model by first starting with the ("train" and "test") created in the data preparation process in the beginning.
- Then, we seek to drop the columns which have a lot of missing values. We set the threshold as 50% missing values, and drop them.
  - This essentially dropped all the columns from the Payment Cashout Fail dataset.
  - All 56 features from that were dropped.
- For the resultant dataset, the missing values were imputed using the mean.
- In the first preliminary model, we decided to build the model with the resultant dataset after the above processing.
- Upon extensive hyperparameter tuning, we found the optimal hyperparameters as:
  - max_depth = 3 (Maximum tree depth for base learners)
  - n_estimators=100
  - eta=0.3 (Boosting learning rate)
- The below figure shows us the most important features in the model. They have been y-axis is a measure of the 'gain' of the feature. We can see how most of the important features are from the New Account Sender Engagement dataset.

Figure 3.2: Feature Importance



- We got really good performance with this model - AUC scores of above 0.75 on the validation datasets. The final model was trained on the entire dataset and reported on Kaggle, and we got a final score of 0.75+.

3.6 XGBoost - further Experimentation

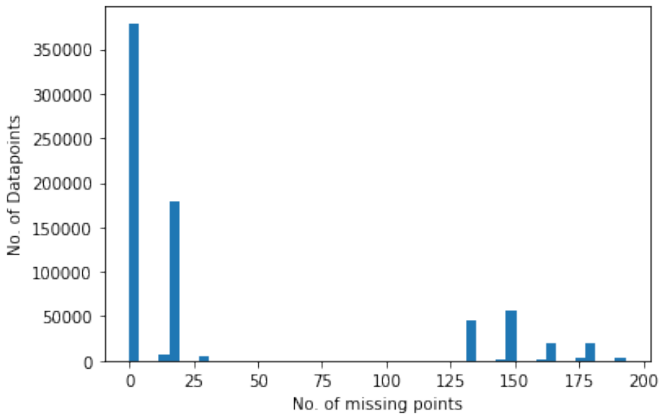
3.6.1 XGBoost - with Feature Selection

- In Figure 3.2 we saw that only some of the features of the total 300+ features might be actually be important to the model.
- By using the feature importance scores, we decided to reduce the number of features - restricting ourselves to features only having a gain metric greater than 0.002.
- As a result, we finally had only 83 features remaining in the final train data that was given to the model for training.
- The model performance on the validation sets was good, but we didn't feel it was any better than just the pure XGBoost model tried previously. It fell just shy of the pure XGBoost performance.

3.6.2 XGBoost - with PayBuddy and non-PayBuddy customers

- We plotted a histogram of the number of datapoints having a particular number of missing features:

Figure 3.3: No. of missing features



- We can clearly see there are 2 groups - one group having less than 50 missing features and the other one having more than 125 missing features.
- Hence we split the 'train' and 'test' into 2 further datasets - 'train-old' for PayBuddy customers and 'train-new' for non-PayBuddy customers.
- We trained 2 different XGBoost models separately for datasets to see if performance could be improved further.
- The model performance, again, was very good. We found it very similar to the initial XGBoost model, with AUC scores of around 0.75. There was only a difference in the 3rd or 4th decimal places.

Results

Model	AUC Score
Random Forest - Without Preprocessing	0.720
Random Forest - With Preprocessing	0.722
Adaboost	0.719
Lasso	0.740
XGBoost	**0.752**
XGBoost - with Feature Selection	0.750
XGBoost - Paybuddy and non-Pay Buddy customers	0.750

Table 4.1: Model Performance

4.1 Conclusion

In this data contest, we used various models including AdaBoost, Lasso regression, Random Forest, XGBoost, and LightGBM to analyse financial credit card data of new and returning pay-buddy customers. The best results were obtained using XGBoost.