

Elo Merchant Category Recommendation

Group 17: Jiaru Xu, Yuqi Liu, Weiteng Li

Project Overview

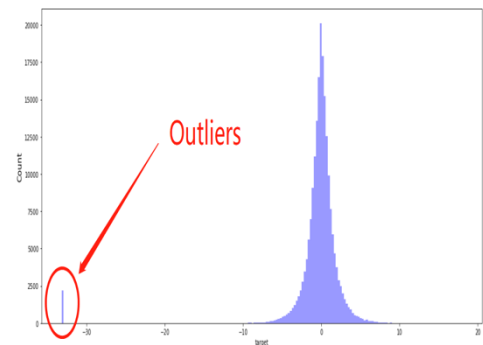
Elo is one of the largest payment brands in Brazil, building partnerships with merchants in order to offer promotions or discounts to cardholders. They wonder whether they are now running an efficient recommendation strategy to both consumers and merchants. Through this project, we will try to uncover the signal relating to customers' reaction when given promotions or discounts. In detail, we will develop models to identify individual customers' loyalty based on 3-months length customers' historical transactions. Evaluation will be based on the root mean squared error over predicted and actual loyalty scores.

Main steps of our project are listed as below:

- Exploratory Data Analysis
- Basic Feature Engineering
- Meta Features
- Modeling and Post Processing

Exploratory Data Analysis

We've done a lot EDA to understand the obvious and hidden pattern inside our data. One critical finding here is that as the right picture shows about 1% of the observations in our dataset are outliers, whose loyalty scores are significantly less than the median and mean. At first, we tried to simply exclude those points. However, after observing our leaderboard score through different trials, we notice that outliers here are playing an important role. Simply excluding those observations will not improve the accuracy but induce information loss instead. Recalling our evaluation matrix is RMSE, outliers will greatly affect the final score in a way. Therefore, we may also need to model behaviors of our outliers.



Other details for EDA are included in our jupyter notebook.

Basic Feature Engineering

Based on our understanding, we tried several kinds of feature engineering. Here's a brief summary of what we've implemented:

What we've tried that works:

- **Extract Time-based Features**
week of year / day of week / weekend / hour / purchase year / purchase month / month diff
- **Group by & Merge**
Because the data is tabular with more than 2 dimensions, we can group by & aggregate the features to extract some useful information. The aggregation function can be versatile. We've implemented min / max / sum / mean / min-max / skewness / kurtosis / var as aggregating functions, generating more than 150 variables. Some of them boost the model significantly.
- **Build Interaction Features Based on Time**
We implemented some interaction features that works well such as difference in total purchase amount in Month A and total purchase amount in Month B.

What we've tried that didn't work:

- **Target Encoding with Regularization**
target encoding for high-cardinality categorical variables: city id / merchant category / subsector id (including smoothing, out of loops encoding)
- **Mold**
mold of high-cardinality categorical variables: city id / merchant category / subsector id
- **Count Vectorizer & Dimensionality Reduction**
We tried to treat merchant id, subsector id as text data, implementing count vectorizer to them and using PCA & SVD to extract useful information but did not improve much unfortunately.

Meta Features

One of the main difficulties of this competition is how to perform feature extraction. Because every card id links to multiple transactions and multiple merchants, it's obvious that simply group by and aggregating approach can't fully deploy and extract the useful information in the data set. Therefore, in order to extract more useful information, one approach we deploy here is to train a transaction-based model with following steps:

1. Merging targets to transactions data. (every target links to multiple transactions)
2. Using transactions as features to train the model to predict target.
3. Group by & aggregating the prediction of the transaction-based model and merge the result to train/test set as meta features.

Note that this approach may lead to target leaking, so we need to implement regularization when necessary.

Modeling & Post Processing

For base model, we simply use 5 folds stratified cross validation and LightGBM. After that, chasing after accuracy, we have one interesting post processing approach with desirable results. Remember that in EDA we find about 1% of the targets are outliers and they can affect our model in the following two ways.

First, because the cost function is RMSE, who gives more weight for outliers, the overall model prediction for the none-outliers will have some negative bias. Second, the model treats the whole problem as a regression problem, but actually all outliers has the same target (about -33) while others range from about -10 to 10. Therefore, if we are mostly sure that an observation is an outlier, we should just predict the target as -33 instead of using a regression to calculate its score. Then, how to find the outliers? Just build a classifier!

To alleviate the punishment from outliers, we can build a model without outliers to make a regression prediction without negative bias. Then, we can build a classifier to predict whether an observation is outliers. However, the features we have can only build a poor classifier. So here comes the smoothed final post processing solution:

$$\text{Final Prediction} = p_{\text{outliers}} \times (-33.219) + (1 - p_{\text{outliers}}) \times \text{Prediction}_{\text{no_outliers_model}}$$





Conclusion

Finally, our solution can rank about 60 / 4129 without any blending or stacking.

Our late submission score:

Submission and Description	Private Score
combining_submission_final.csv 2 hours ago by KAGOKAGO	3.60656

Private Leaderboard Ranking of the same score:

59	 24	Atanas Atanasov		3.60638
60	 67	Madiyar Aitbayev		3.60660