Predicting Customer Loyalty

Vo Thuc Khanh Huyen
Viettel Digital Talent 2021

Abstract

In this project, we will focus on resolving the problem in the Kaggle challenge named *"Elo Merchant Category Recommendation"*. We will analyze, process data, and use machine learning to improve the understanding of customer loyalty for the Elo payment brand. The results show that LightGBM is the best performing algorithm on the given dataset.

Introduction

Elo is one of the biggest and most reliable payment brands in Brazil. *Elo Merchant Category Recommendation* problem talks about the loyalty score of credit cards for their users in the Elo brand. The loyalty score is calculated in 2 months after the historical and evaluation period. This is a regression task and we use RMSE for evaluation. In this project, we aimed to:

- Analyze, process data to generate many suitable features from our dataset.
- Apply various machine learning models to our dataset and do the assessment on achieved results.

Data Analysing

The Elo dataset contains 5 data tables:

- **Train:** shape (201917, 6), contains 6 features, which is *card_id, target, first_active_month*, 3 anonymous categorical features are *feature_1, feature_2 and feature_3*. The *target* feature is the loyalty score.
- **Test:** shape (123623, 5), contains the same feature as train data but the target feature is not present in this dataset.
- **Historical transaction:** shape (29112361, 14), contains transactions information made up to the reference month.
- **New transaction:** shape (1963031, 14), contains transactions information made in 2 months after the reference month.
- **Merchant:** shape (334696, 22), contains additional information about all merchants (*merchant_id*) in the dataset.

We will go into detail for each table data.

- **Train and test data**

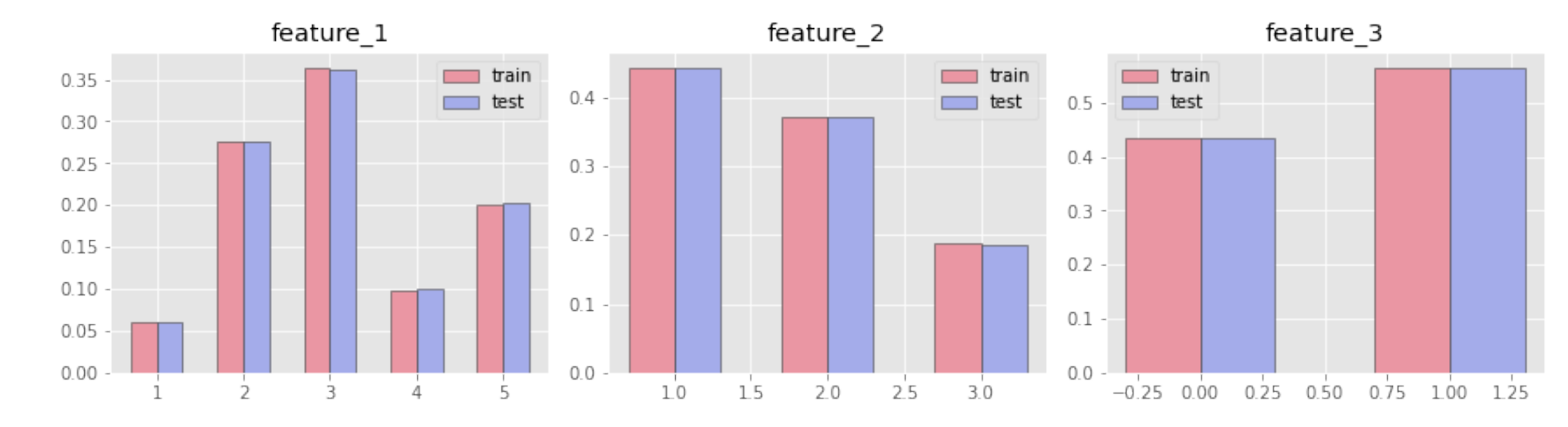

Figure 1:The distribution of features in train and test data

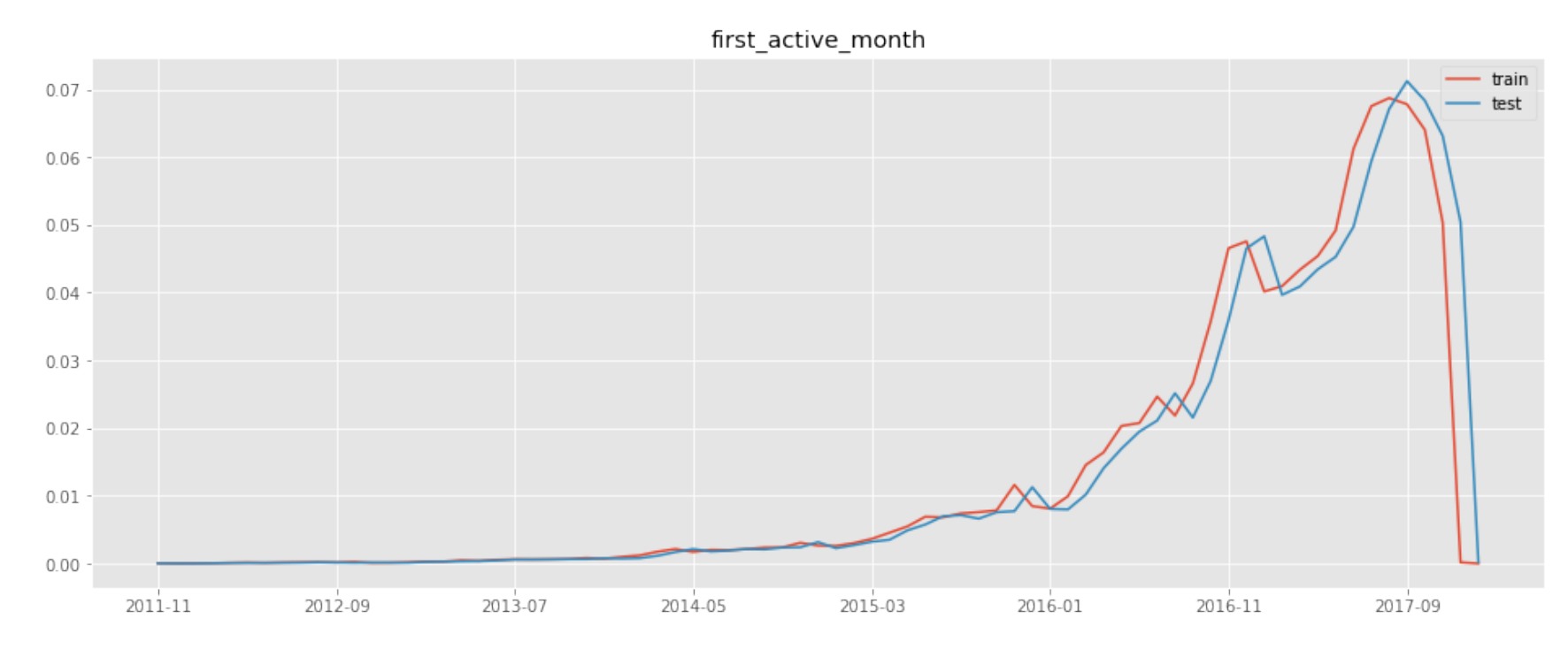

Figure 2:Train - Test - *first_active_month*

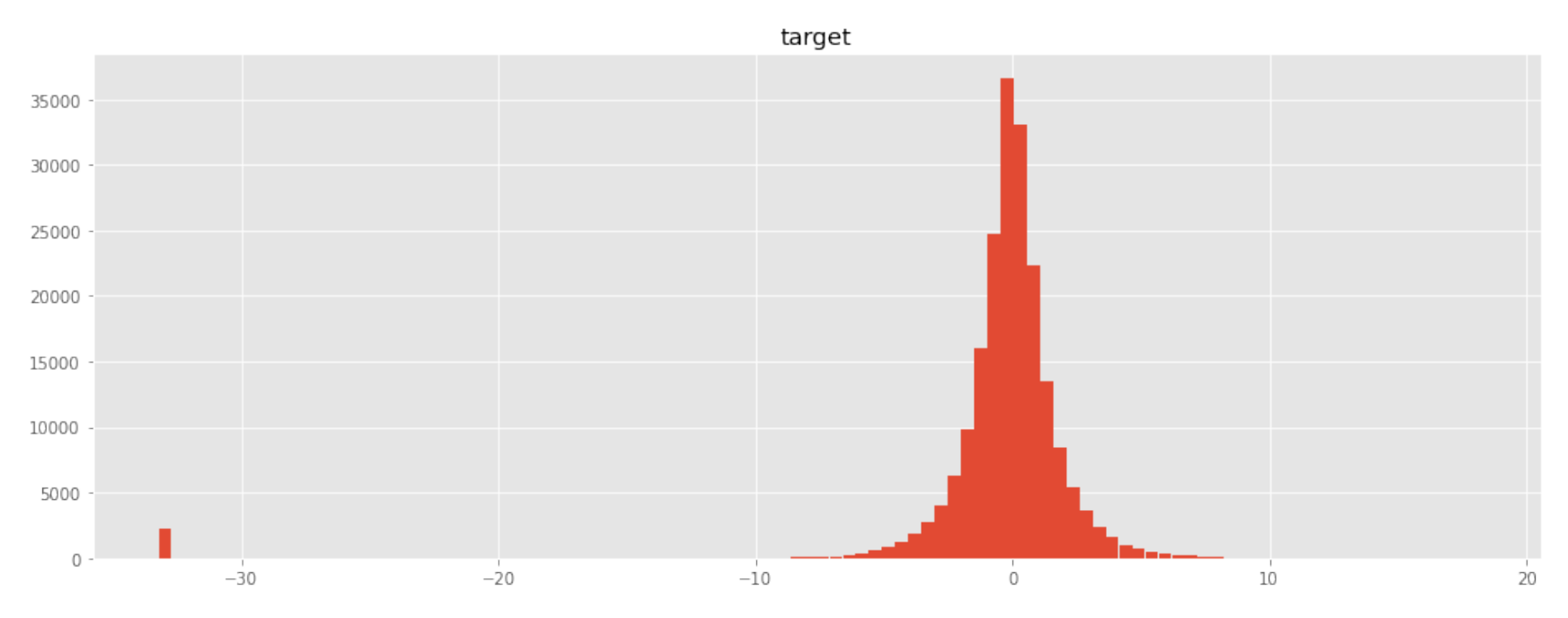

Figure 3:Train - *target*

- **Historical transaction**

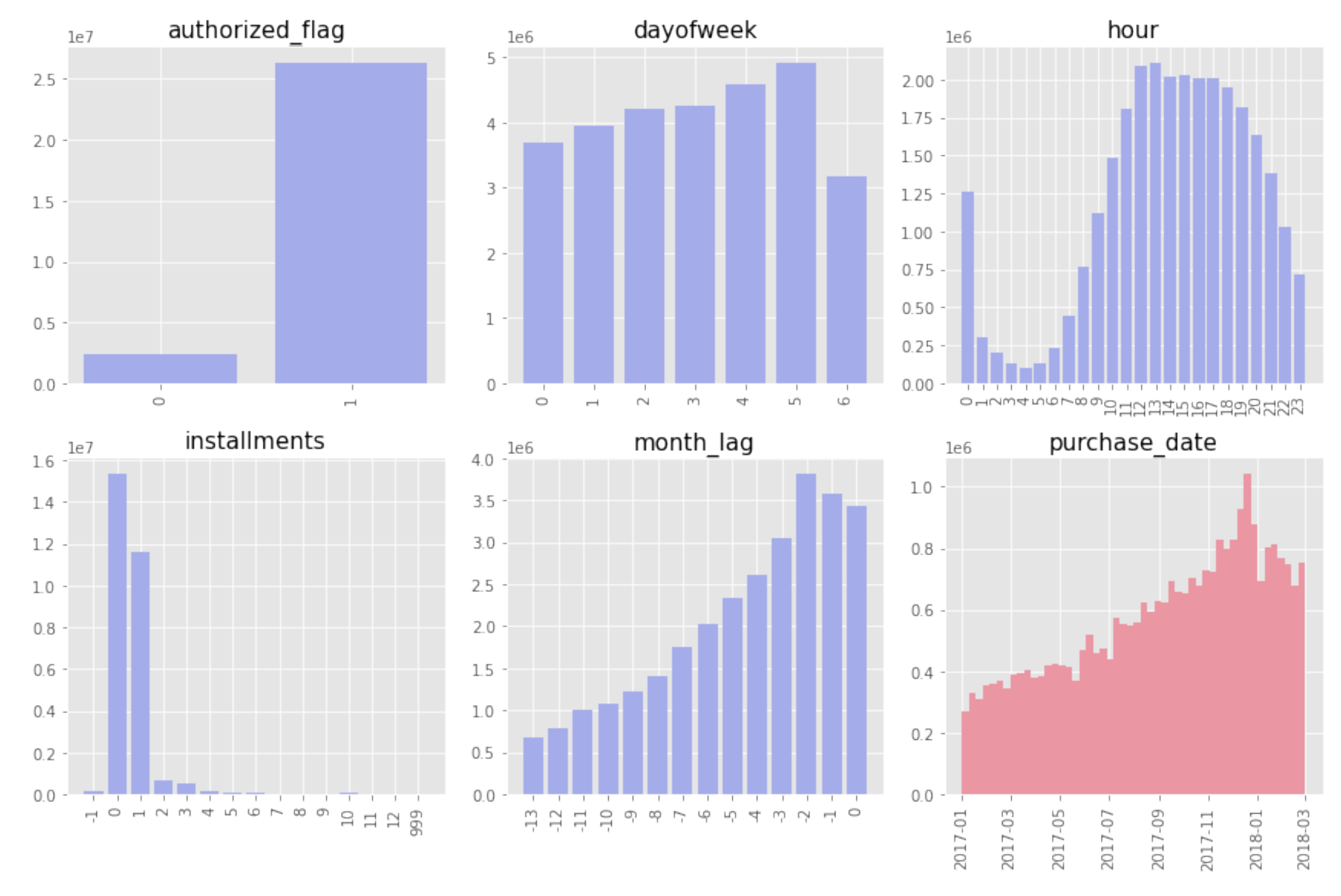

Figure 4:Historical transaction features

- **New transaction**

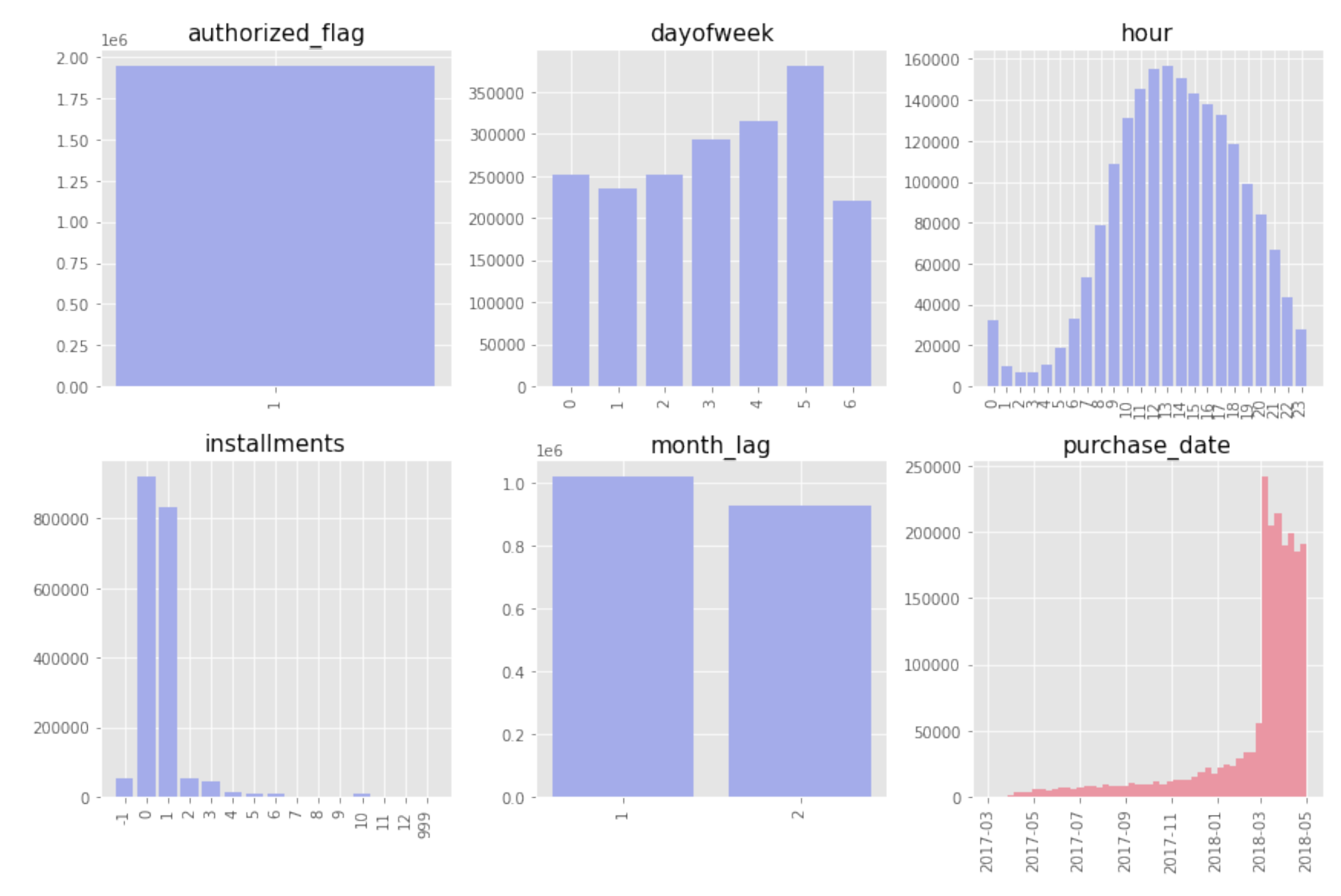

Figure 5:New transaction features

- **Merchants:** We will not use this table in training model, so we don't discuss about it.

Data Preprocessing

- **Train and test data:** We don't encode the categorical features (*feature_1, 2, 3*).
- **Historical transaction:** We reduce memory usage for historical_transactions.csv, encode categorical features like *month_lag, category_2, category_3*, replace illegal and infinite values with nan values, transfer data type of *purchase_date*, and for *purchase_amount*, we transfer by the way:

$$pa = pa/0.00150265118 + 497.06$$

- **New transaction:** Doing the same things as the historical transaction table.

Feature Engineering

We do feature engineering about *purchase_date, purchase_amount, merchant_id* and basic information related features. After, we group histrorical and new transaction table by *card_id*, we will get hist_feats and new_feats table which have *card_id* as index.

Modelling

- **Training models:** We tried 2 models LightGBM and CatBoost. These two models allows to pass categorical features into train data. For each model, we tried to pass encoded and unencoded categorical features (*feature_1, feature_2, fetures_3*). We use 5 folds cross-validation for training, we also use property *is_outlier* to split train and validation data. After all, we choose LightGBM - unencoded model because this has the best score in CV, private and public.

	CV score	Private score	Public score
LightGBM - unencoded	**3.64529**	**3.60952**	**3.69481**
CatBoost - unencoded	3.66582	3.62592	3.72180
LightGBM - encoded	3.66342	3.61376	3.70235
CatBoost - encoded	3.67428	3.62856	3.72337

Table 1:Model comparison

- **Final model:** The final result we achieve score as rank 108 on the private leaderboard.

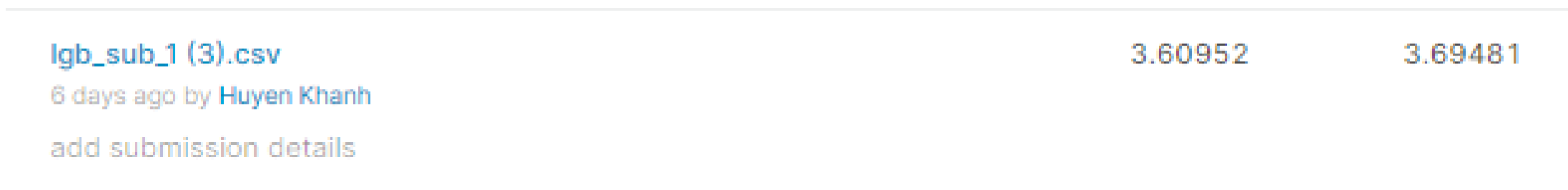
lgb_sub_1 (3).csv 3.60952 3.69481
6 days ago by Huyen Khanh
add submission details

Figure 6:Final result

objective="regression"	max_depth= 9
boosting="gbdt"	learning_rate=0.005
metric="rmse"	bagging_fraction=1
reg_alpha=0.1	bagging_freq=1
reg_lambda=20	feature_fraction=0.2
num_leaves=120	verbosity=-1
min_data_in_leaf=70	num_boost_round=10000
min_gain_to_split=0.05	early_stopping_rounds=200
max_bin=350	verbose_eval=100

Table 2:LightGBM parameters

- **Error Analysing:** After achieving the result, we calculate RMSE for each interval of the target value (see table 3). For values in the interval $(-40, -30]$, these values are outliers in our problems which cause final RMSE to rise to between 3 and 4. The first solution author implemented a classification model to classify these outliers, and get 0.015 boosts in local cv. We tried to implement as this way, but the result is not better.

target	RMSE
(10, 20]	11.52012
(0, 10]	1.95751
(-10, 0]	1.61847
(-20, -10]	11.71220
(-30, -20]	no element in this interval
(-40, -30]	30.31161

Table 3:RMSE

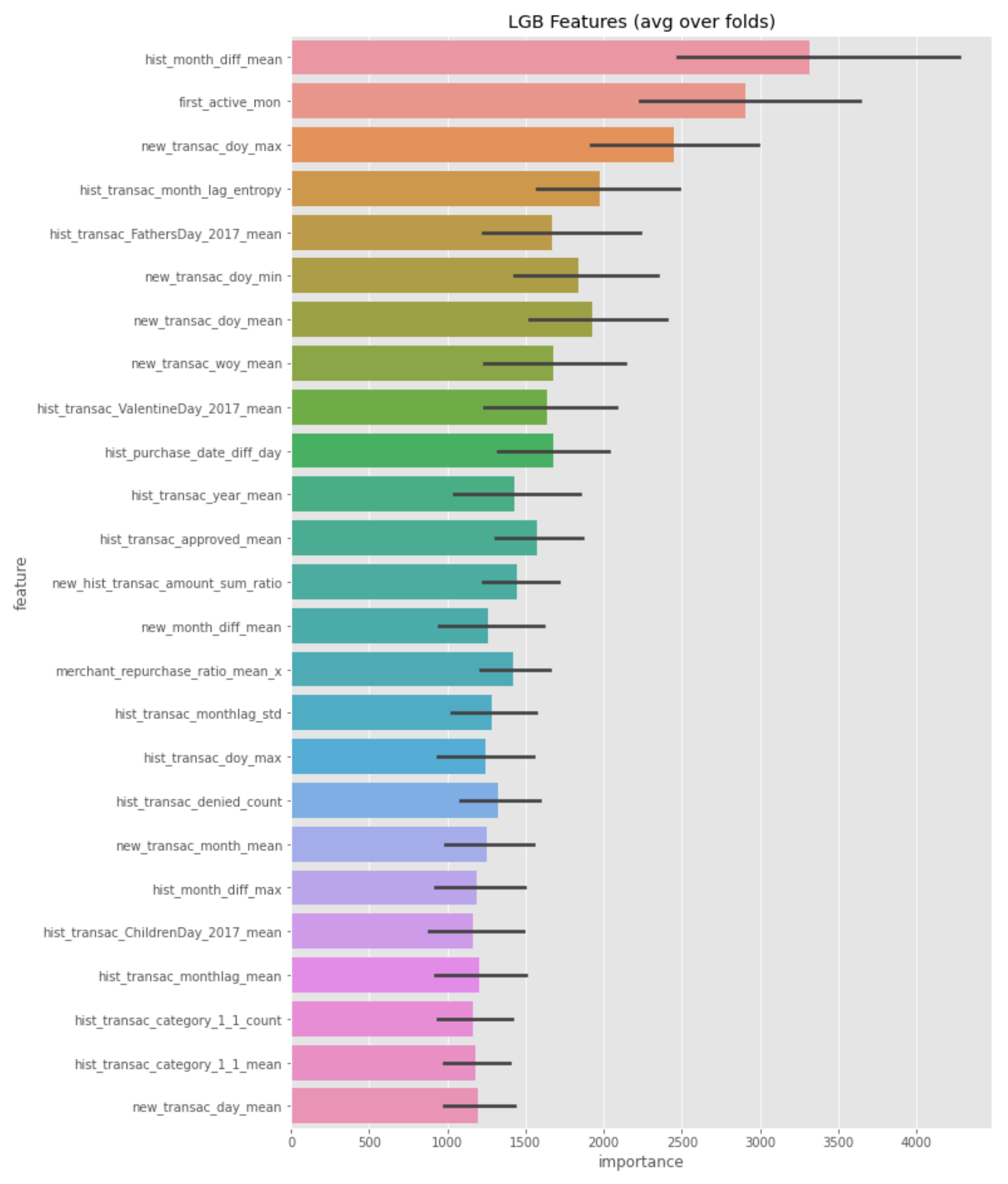

Figure 7:Feature importance

Future Work

- We have to be very careful about the preprocessing of data and feature engineering of data.
- Implement more models such as XGBoost, Neural Network, ... We want to try stacking method on this problem.
- We will try to implement a better classification model to classify outliers.

Reference

- G. Martin, *Reduce Memory Usage*
- raddar, *Purchase amount*
- CPMP, *Purchase amount*
- 30CrMnSiA, *First solution*

Related Information

- Github: *eloRepository*
- vothuckhanhhuyen@gmail.com