Predicting Customer Loyalty

Vo Thuc Khanh Huyen

Viettel Digital Talent 2021

July 1, 2021

Overview

Overview

1. **Abstract**
2. Introduction
3. Data Analysing
4. Data Preprocessing
5. Feature Engineering
6. Modelling
7. Future work

Abstract

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.

Overview

Introduction

- **Elo** is one of the biggest and most reliable payment brands in Brazil.
- 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 create many suitable features from our data.
- Apply various machine learning models to our dataset and do the assessment on achieved results.

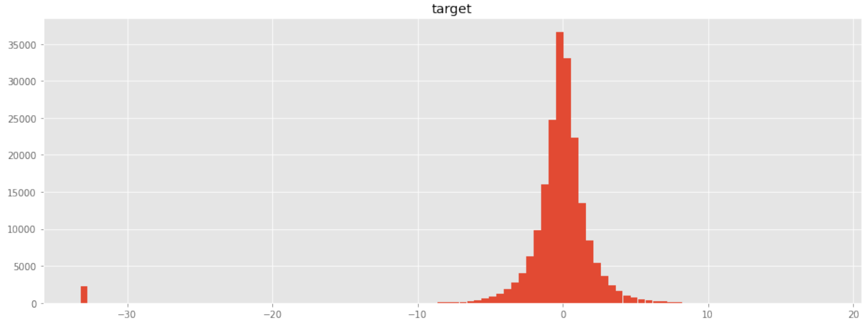
Overview

Data Analysing - Tables in dataset

1. **Train:** 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.
2. **Test:** contains the same feature as train data but the target feature is not present in this dataset.
3. **Historical transaction:** contains transactions information made up to the reference month.
4. **New transaction:** contains transactions information made in 2 months after the reference month.
5. **Merchant:** contains additional information about all merchants (*merchant_id*) in the dataset.

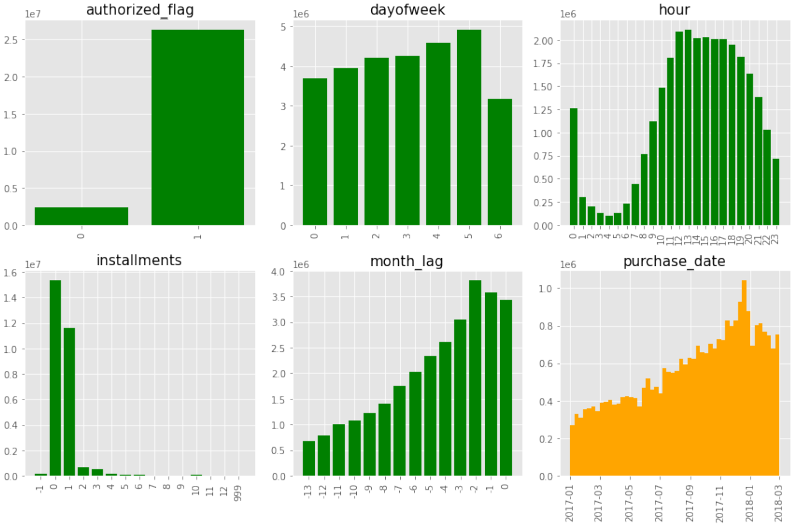
Data Analysing - Train and Test data

Data Analysing - Train and Test data

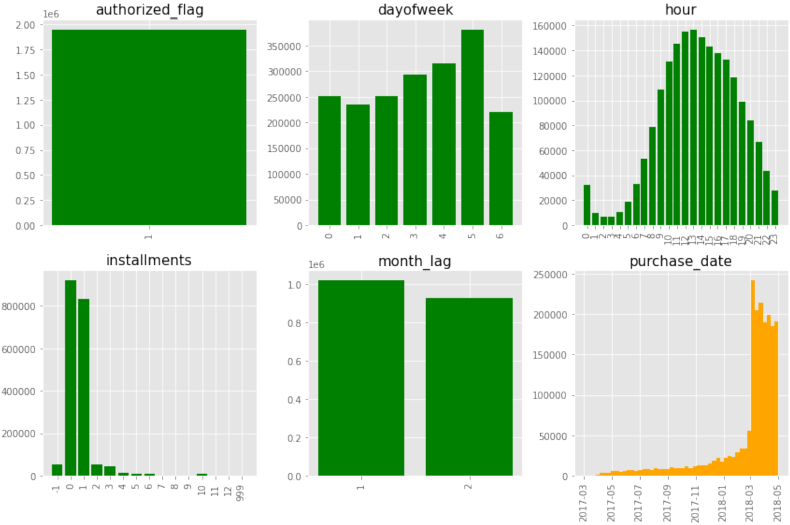


is_outlier: target $= -33.21928095$

Data Analysing - Historical transaction



Data Analysing - New transaction



Overview

1. Abstract
2. Introduction
3. Data Analysing
4. **Data Preprocessing**
5. Feature Engineering
6. Modelling
7. Future work

Data Preprocessing - Train and Test data

Encode

- We don't **encode** the categorical features (*feature_1, 2, 3*) because we will use LightGBM and CatBoost algorithm, which allow data to have categorical features.
- We tried to encode them, but the results are not good.

Data Preprocessing - Historical transaction

- **Reduce** about half of memory size of historical_transactions table.
- **Encode** categorical features like *month_lag*, *category_2*, *category_3* using *LabelEncoder* and *get_dummies* from sklearn and pandas.
- **Replace** infinite values with nan values. Replace installments which values are -1 or 999 with nan values.
- **Replace** nan values from categorical feature with mode and numerical data with the median.
- **Transfer** type of *purchase_date* into datetime type.
- **Normalize** the *purchase_amount* by the way like:

$$pa = pa/0.00150265118 + 497.06$$

Overview

Feature Engineering - Train and Test data

Train and Test data

We extract *month* and *year* of *first_active_month* feature, and then encode them.

Feature Engineering - Historical transaction

- **purchase_date related features:**
  - Extract several **elements of day** like *year*, *month*, *weekofyear*, *dayofyear*, *weekofyear*, *day*, *hour*.
  - Check whether the purchase_date is on the **weekend** or not.
  - We count the countdown days from the purchase_date and these **special days**.
  - We count **days difference** between two consecutive purchase_date of each customer. We also count **months difference** between the purchase_date and the day that this contest released.
  - And much more features related to month_lag, ...

Feature Engineering - Historical transaction

- **purchase_amount related features:**
  - We *group* historical transaction table by *card_id* using simple aggregation functions on *purchase_amount*. Also, we try to group by **2 features** which are *card_id* and one of these features: *category_1*, *2*, *3*, *installments*, and ID related features.
  - We also consider the **ratio** of total purchase_amount between 2 month_lag.

Feature Engineering - Historical transaction

- **merchant_id related features:**
  - We define **intimate merchant** for each customer which is the place that the customer visited at least 2 times. In this situation, the customer and transaction are called an **intimate customer and transaction** for this merchant.
  - For each merchant, we count the **ratio** between total intimate customers and total customers, total intimate transactions and total transactions.
- **Basic information related features:**
  - We **consider** authorized and unauthorized transactions, count unique values of ID related features, ...

Feature Engineering - New transaction

New transaction

We do the same things as in the historical transaction table, but we **don't consider** authorized and unauthorized transactions.

Overview

Modelling - Training data

We do the same things for train and test table, example for train table:

- **Merge** train, hist_feats and new_feats table into one table by using *card_id*.
- **Create** features **related to both hist_feats and new_feats table** by taking ratio, sum, ...

new_hist_transac_amount_sum_ratio
new_hist_transac_amount_max_ratio
new_hist_transac_amount_sum_log_ratio
new_hist_transac_amount_max_log_ratio
installment_total_sum
new_hist_purchase_amount_ratio_1_1
new_hist_purchase_amount_log_ratio_1_1
new_hist_purchase_amount_ratio_1_2
new_hist_purchase_amount_log_ratio_1_2
...

Modelling - Training models

- We tried **2 models** LightGBM and CatBoost.
- 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.

	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

Modelling - Final model

The final result we achieve score as rank 108 on the private leaderboard.

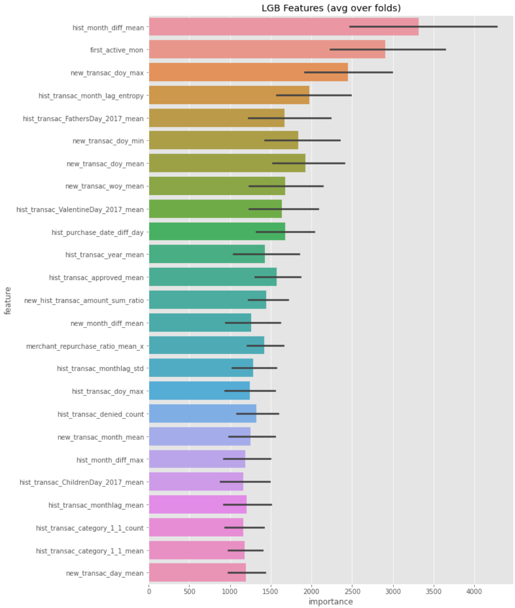
lgb_sub_1 (3).csv — 3.60952 — 3.69481

6 days ago by Huyen Khanh

add submission details

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

Modelling - Final model



Overview

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. The first solution authors tried this way and get 0.015 boosts in local cv.

Reference

1. **G. Martin**, *Reduce Memory Usage* (https://www.kaggle.com/gemartin/load-data-reduce-memory-usage)
2. **raddar**, *Purchase amount* (https://www.kaggle.com/raddar/towards-de-anonymizing-the-data-some-insights)
3. **CPMP**, *Purchase amount* (https://www.kaggle.com/cpmpml/raddar-magic-explained-a-bit)
4. **30CrMnSiA**, *First solution* (https://www.kaggle.com/c/elo-merchant-category-recommendation/discussion/82036)

Thank you

Thank you!