REPORT FOR PROJECT 1

Li Chengxin (20745826)
Li Xinyi (20745163)
Liang Xin (20745785)
Lu Lanxi (20744822)

September 20, 2021

Abstract

In this project, we analyze the Home Credit Default Risk dataset to predict the repayment abilities of clients. By conducting the feature engineering, attempting different machine learning models, and tuning parameters, we get AUC score of 0.748 from Kaggle. We find that compared to other machine learning methods, Light Gradient Boosting Machine performs best for not only in-sample data set, but also out-sample prediction.

1 Data Description

In this project, we use Home Credit Default Risk dataset from Kaggle. Our aim is to predict clients' repayment abilities by making use of a variety of alternative data, such as telco and transaction information. There are eight tables from the dataset, and there is also a diagram to show their relationships.

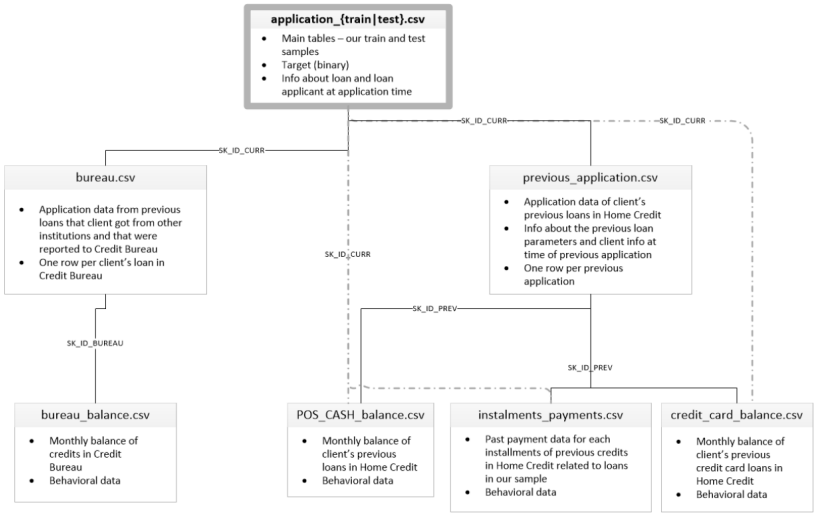

Figure 1: overview of data

In these tables, we get both numerical and dummy variables. We find some numerical variables like EXT_SOURCE have negative correlation with TARGET (-0.235). Besides, some dummy variables which seem not related to the target may have the correlation with the default rate, we just plot CODE_GENDER and OCCUPATION_TYPE as two examples, we find the difference between these categories is significant.

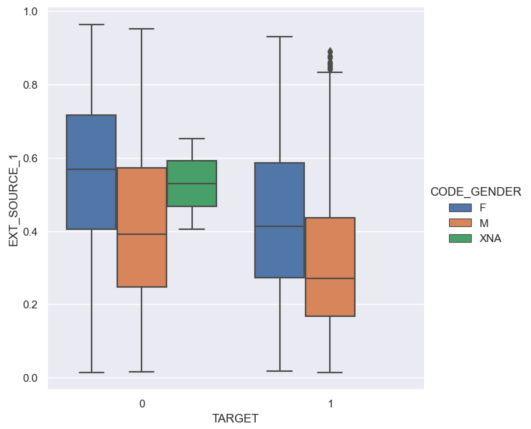

Figure 2: CODE_GENDER

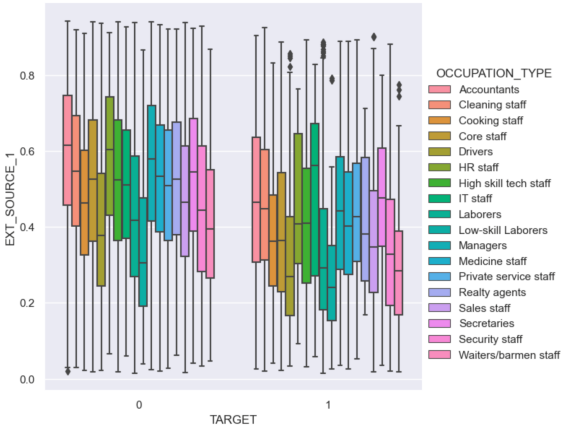

Figure 3: OCCUPATION_TYPE

We also plot the distribution of missing values and unique values.

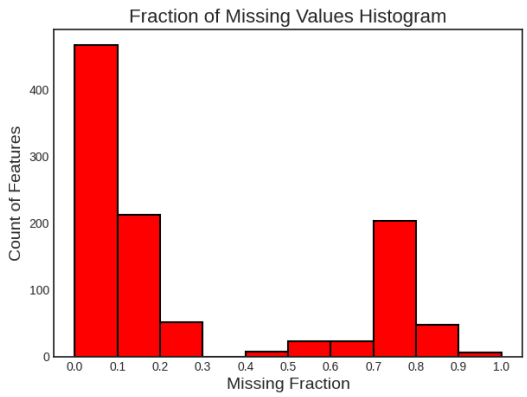

Figure 4: CODE_GENDER

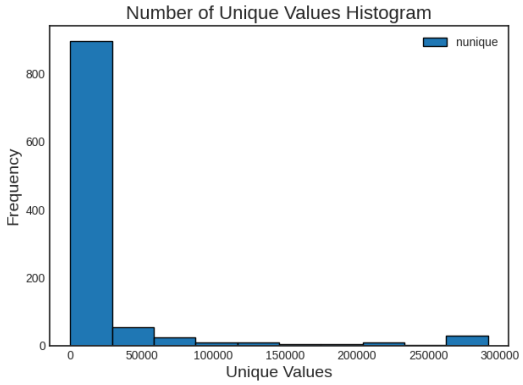

Figure 5: OCCUPATION_TYPE

2 Feature Engineering

2.1 Data Cleaning

Training and testing data:
1. Drop data if CODE_GENDER is XNA: must be male or female.
2. Drop some features not presented in the testing data: FLAG_DOCUMENT 2, 10, 12, 13, 14, 15, 16, 17, 19, 20, 21.
3. Replace outliers with Nan.

2.2 Feature Engineering

There are some general methods for data engineering.
1. Create new features: use financial knowledge to add new features from existing features.
2. Label numeric columns: for instance, we label days less than 27 to be 1, days larger than 27 and less than 40 to be 2 for column of DAYS_BIRTH.
3. Create a new column to count missing values of one application ID.
4. Aggregate numeric columns: Compute statistics for numeric columns (min, max, size, mean, variance, sum). To do so, we group the data by "clientId", aggregate the data and then merge it back to the original data.
5. Utilize get_dummy() to realize one-hot-encoding for categorical columns .

As some people have historical records, later records should be given higher weights. Considering these, we use every 7 days as a level to calculate their weights and aggregate historical records. Below is the distribution:

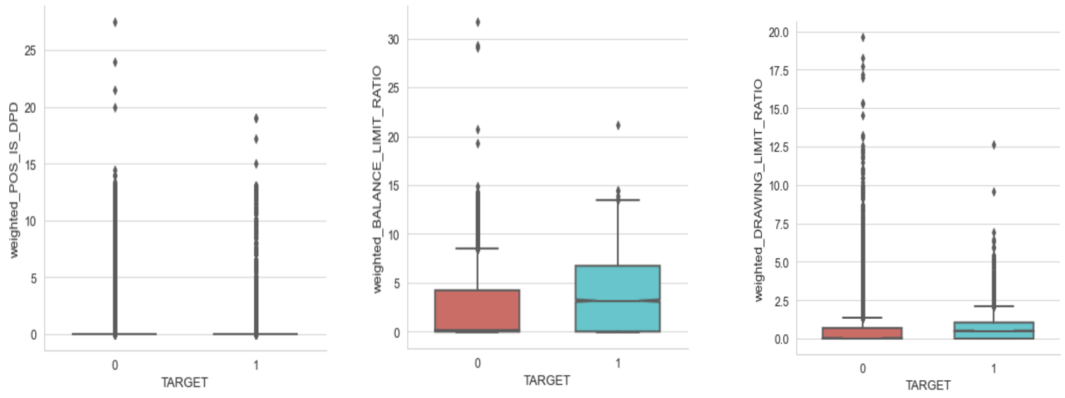

Figure 6: distribution of features

2.3 Final Data

After data cleaning and feature engineering, the shape for the data set is shown below.

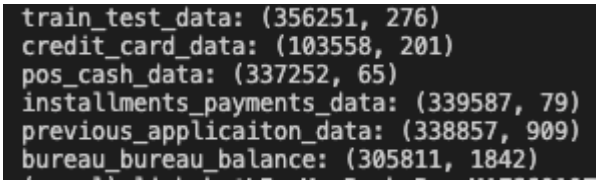

Figure 7: shape of each data

Merge all the data into one table, the size for the final data is 356251 by 3507.

3 Models and Results

3.1 Bayesian Optimization for LIGHT GBM model

We use Bayesian Optimization to get appropriate parameters for Light GBM. The range for each parameter is set as narrow as possible.

```
lgb_pool = BayesianOptimization(lgb_eval, {'learning_rate': (0.01, 1.0),
                                           'num_leaves': (24, 80),
                                           'feature_fraction': (0.1, 0.9),
                                           'bagging_fraction': (0.8, 1),
                                           'max_depth': (5, 30),
                                           'max_bin': (20, 90),
                                           'min_data_in_leaf': (20, 80),
                                           'min_sum_hessian_in_leaf': (0, 100),
                                           'subsample': (0.01, 1.0),
                                           'lambda_l1': (0, 5),
                                           'lambda_l2': (0, 3),
                                           'min_split_gain': (0.001, 0.1),
                                           'min_child_weight': (5, 50)},
                                           random_state=200)
lgb_pool.maximize(init_points=15, n_iter=15)
print(lgb_pool.res)
```

Figure 8: Bayesian Optimization

Parameters to maximize the score in the training data with 5 folders.

```
# {'bagging_fraction': 1.0, 'feature_fraction': 0.9, 'lambda_l1': 0.0, 'lambda_l2': 0.0, 'learning_rate': 0.01,
# 'max_bin': 53, 'max_depth': 30, 'min_child_weight': 44.23386978946496, 'min_data_in_leaf': 80, 'min_split_gain': 0.001,
# 'min_sum_hessian_in_leaf': 89.4965551805231, 'num_leaves': 34, 'subsample': 1.0}
```

Figure 9: parameters maximize the score

The AUC for training data is 0.7543.

We submit the predicted label to Kaggle and get the score of 0.7478.

We use more than 3000 features to get this result. In addition, we select the most important 100 features (also given by Light GBM) to fit the model and get a higher score: 0.7480 (0.22% higher).

Plot the feature importance for top 100 features:

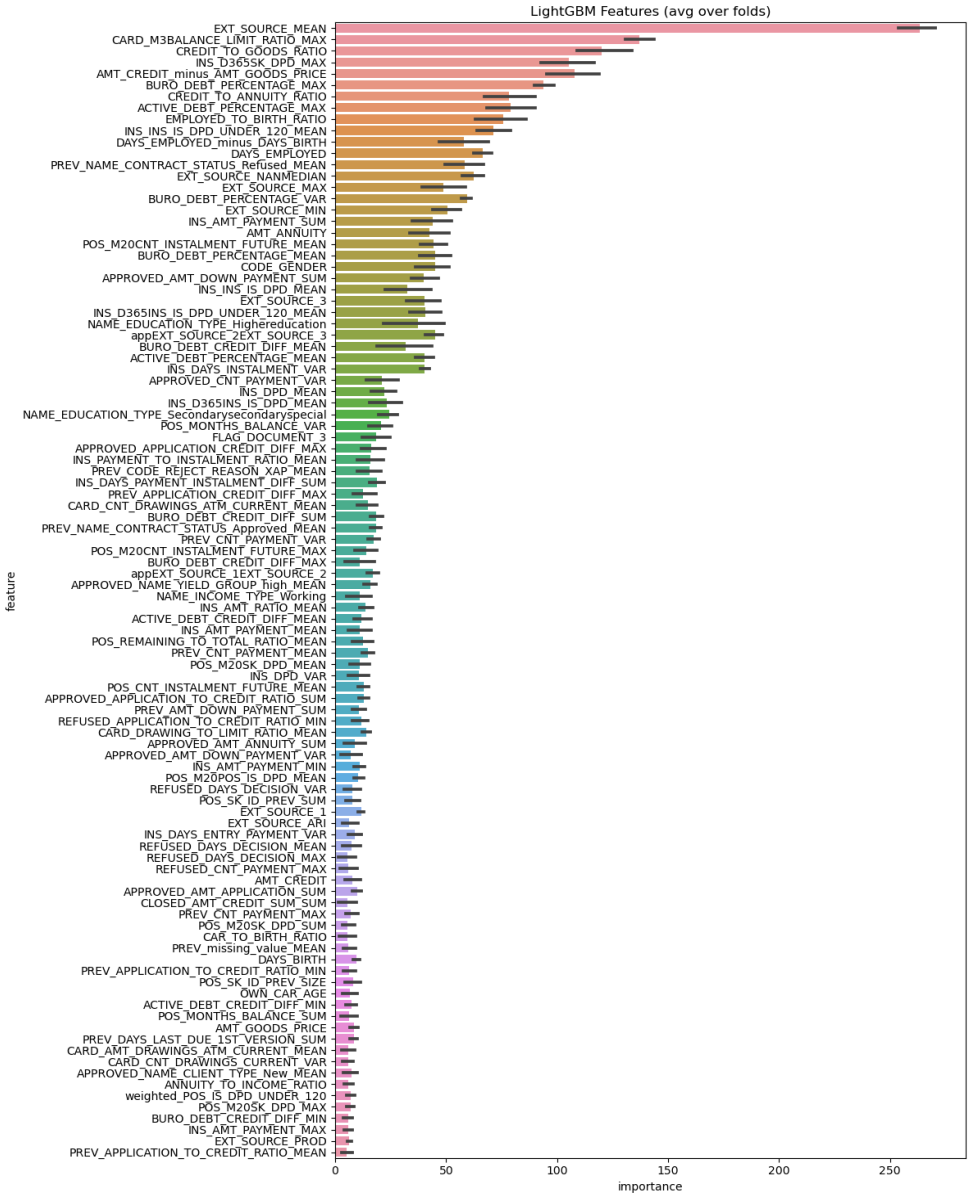

Figure 10: top 100 features

Plot the distribution for top 15 features:

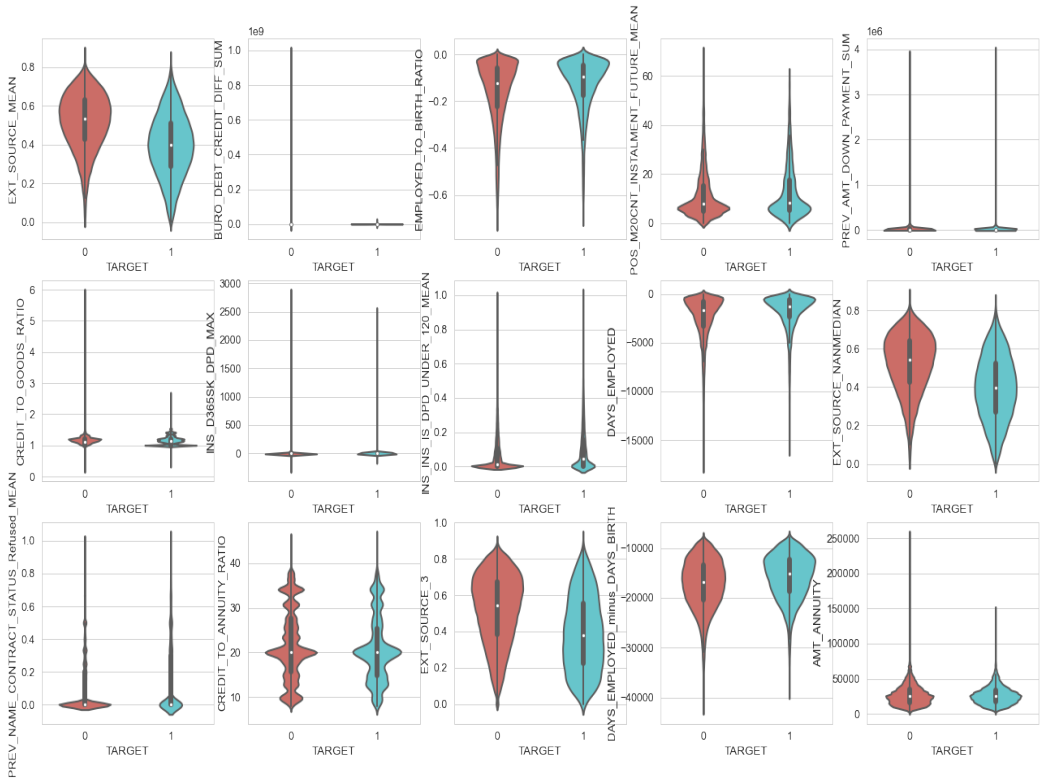

Figure 11: distribution for top 15 features

3.2 Other Models

In addition to LGBM model, we also try other models, but LGBM performed best. Their parameters and scores are shown below.

1. Random Forest
RandomForestClassifier(criterion='gini', max_features='auto',
min_samples_split=2, n_estimators=400, warm_start=True)
The AUC score is 0.7224.

2. Bagging
BaggingClassifier(max_samples=1, n_estimators=40, oob_score=False)
The AUC score is 0.6836.

3. AdaBoost
AdaBoostClassifier(algorithm='SAMME.R', learning_rate=0.5)
The AUC score is 0.7412.

4 Contribution

Li Chengxin: Feature engineering, model training, and report writing
Li Xinyi: Feature engineering and report writing
Liang Xin: Model training and report writing
Lu Lanxi: Feature engineering and report writing

5 GitHub

https://github.com/aaronliang75/MAFS6010Z_Project1