Analysis on Home Credit Default Risk

Huang Zhenyu 20744676, Luo Jiahao 20744418, Yang Yannan 20746131
Kaggle team: math6010zluoylinie
Contribution: 37%, 33%, 30%

1 Introduction

Home Credit provides credit loan service for people without bank accounts. One of their key tasks is to find out whether customers will repay the loan on time or have difficulties in it. The competition aims at using historical loan application data to predict whether applicants will be able to repay the loan.

2 Data description

There are 307511 records and 122 features in the training set. Among the 122 features, 65 are floating numbers, 41 are integers, and 16 are non-numerical objects. The number of Target = 1 is 24825, and the number of Target = 0 is 282686, which means there are far more people who are repaying their loans on time than those who do not.

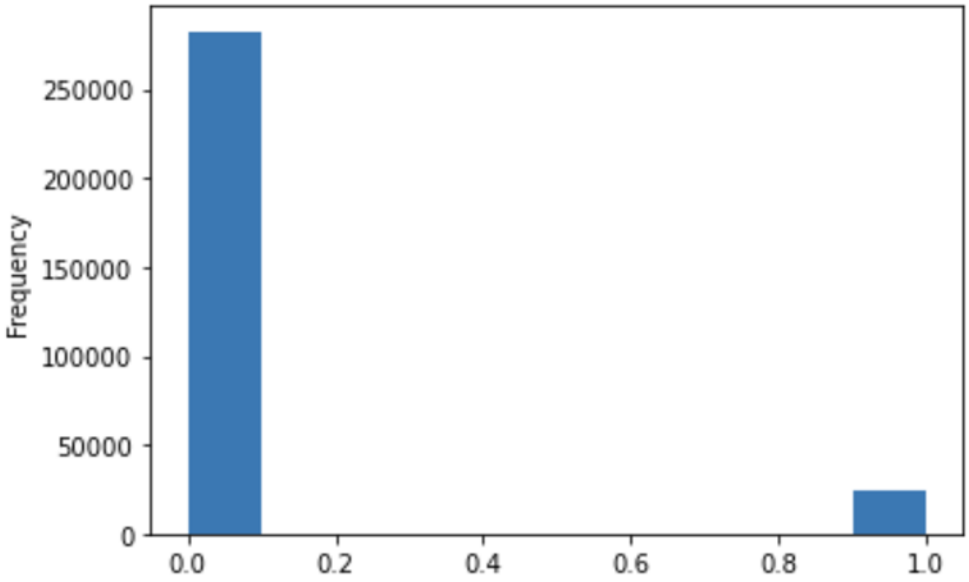

Figure 1: Distribution of clients' ability to repay the loan

2.1 Exploration in terms of loan is re-payed or not

We counted the repayment situation of applicants form different aspects, such as income sources, family status, occupation type and education. It is found that there is a significant correlation between these aspects and the repayment situation of the applicant.

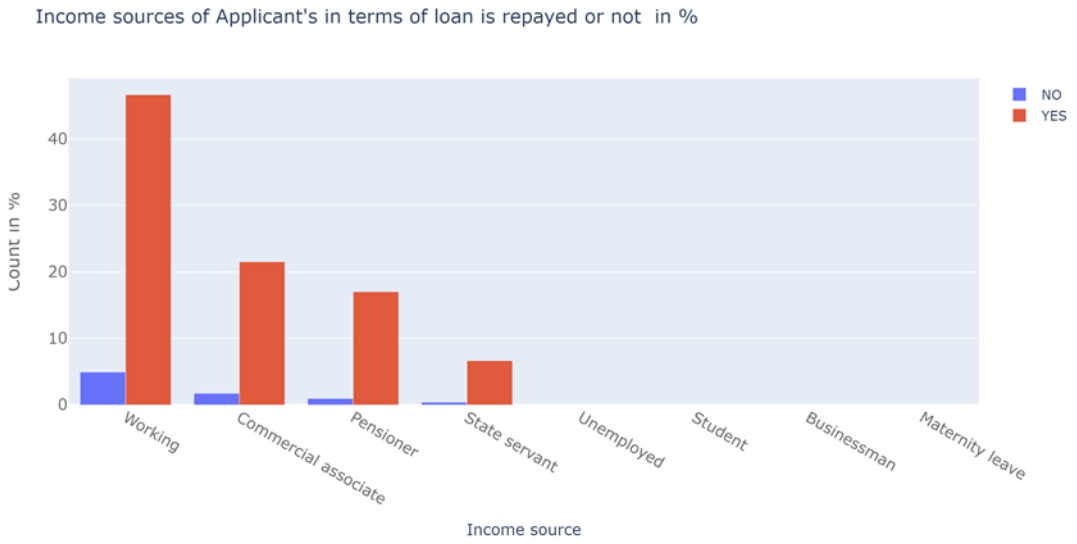

Figure 2: Relationship between applicants' income sources and loan repayment

2.2 Correlation

One effective way to understand the data is to calculate correlations between the features and the target. We can calculate the Pearson correlation coefficient between every variable and the target.

```
Most Negative Correlations:
 EXT_SOURCE_3                               -0.178919
EXT_SOURCE_2                               -0.160472
EXT_SOURCE_1                               -0.155317
NAME_EDUCATION_TYPE_Higher education       -0.056593
CODE_GENDER_F                              -0.054704
NAME_INCOME_TYPE_Pensioner                 -0.046209
DAYS_EMPLOYED_ANOM                         -0.045987
ORGANIZATION_TYPE_XNA                      -0.045987
FLOORSMAX_AVG                              -0.044003
FLOORSMAX_MEDI                             -0.043768
```

Figure 3: Most negative correlation between features and target

All the 3 EXT_SOURCE features have negative correlations with the target, indicating that as the value of the EXT_SOURCE increases, the client is more likely to repay the loan. Next, we look at the distribution of each of these features colored by the value of the target. This will let us visualize the effect of this variable on the target.

In figure 4, EXT_SOURCE_3 displays the greatest difference between the values of the target. We can clearly see that this feature has some relationship to the likelihood of an applicant to repay a loan. The relationship is not very strong, but these variables will still be useful for a machine learning model to predict whether an applicant will repay a loan on time.

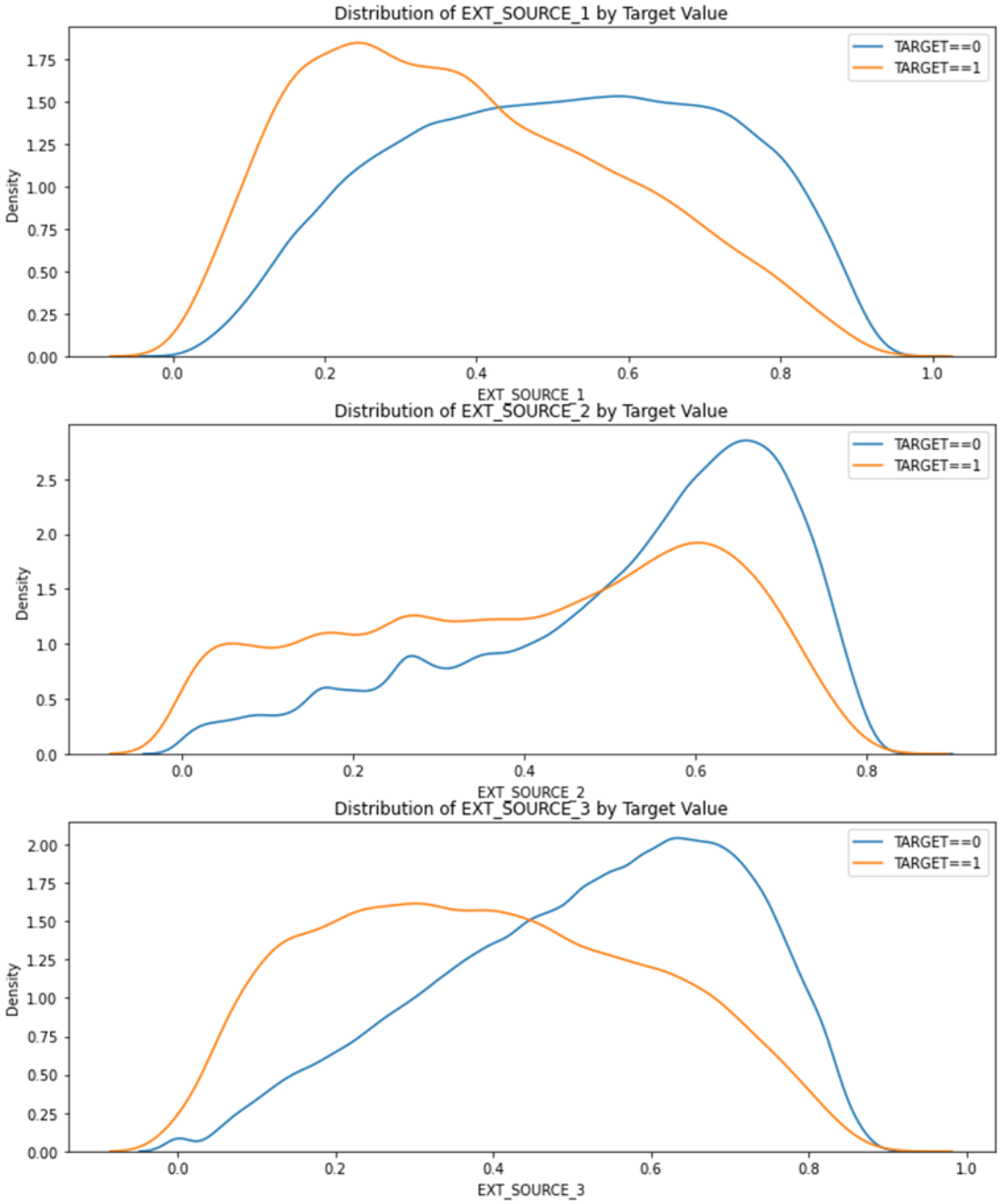

Figure 4: Relationship between EXT_SOURCE and target

2.3 Handling missing values

Before modeling, we must handle the missing values in data. We calculate the percentage of missing values of each variable in the data. To reduce the loss of information, we delete the variable data with more than 40% values missing. Then we use Randomforest to predict the estimated values of the NAs in numerical columns. Randomforest is a comparatively unbiased method for prediction so we can lower the effect of the newly added information to our classification model. For non-numerical columns, we just use one-hot-encoder method to transform the letters into numbers 0 and 1 and keep the NA values in each column as a new category.

3 Modelling and validation

Now we finish dealing with the raw data, and we are going to generate some models to predict the output in test data. An effective way to check if our model is accurate is to use cross-validation method and we set the number of folds to be 10. Here we list the models used in our experiment and their results.

Naïve Bayes is a classification method based on Bayes' Formula.

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

The main idea is to solve the probability of each category under the condition of a certain item to be classified among all features., and the item to be classified belongs to whichever category has the largest probability. In our experiment, the 10-fold cross-validation result is 0.6026643494550952.

Randomforest is a classifier that contains multiple decision trees, and the output category is determined by the mode of the category output of each the individual tree. It is capable of handling high-dimensional data and does not need much parameter tuning and feature selection. The validation score is 0.6969602699416482.

Adaboost is an iterative algorithm. Its core idea is to train different classifiers (weak classifiers) for the same training set, and then combine these weak classifiers to form a stronger final classifier (strong classifier). The algorithm itself is implemented by changing the data distribution. It determines the weight of each sample according to whether the classification of each sample in each training set is correct, and the accuracy of the previous overall classification. The new data set with modified weights is sent to the lower classifier for training. The final classifier is a weighted average of weak classifiers. In our test, adaboost gives a result of 0.7391257994257833.

XGBoost, or Extreme Gradient Boosting, is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way. 0.7461691181488385 is the result of XGBoost cross-validation.

LightGBM is a gradient boosting framework that uses tree-based learning algorithms. It is designed to be distributed and efficient with the following advantages: faster training speed and higher efficiency, lower memory usage, better accuracy, support of parallel, distributed, and GPU learning and capable of handling large-scale data. The cv score is 0.7510624343188278 in our modelling.

From above attempts, we finally select LightGBM as our classification model and predict the target values in the testing data. Our result is 0.73651 after submitting the prediction result to Kaggle.

4 Further improvement

Now we just simply look at the information in training data and do little job in grabbing the features. Next, we are going further into the provided data.

4.1 More on training dataset: feature engineering

We examine the training set and find out an “outlier” in the column DAYS_EMPLOYED as the value is extremely large compared to others. To deal with it, we set the “outlier” to be NA to keep this feature and reduce the impact on other values.

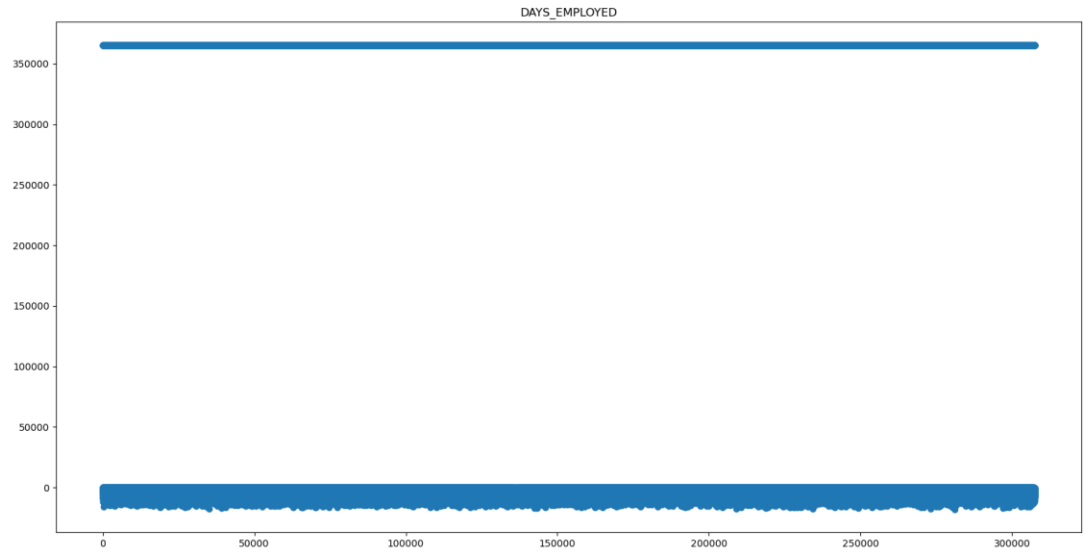

Figure 5: extremely high value occurrence in one column

In previous approach, we delete the columns with over 40% of the value missing. Then we want to know if those removed features contain significant information in our data, and we use the feature importance dataframe generated by LGBMclassifier to check if we ignore those useful messages.

From the below figure, EXT_SOURCE_1 and OWN_CAR_AGE are considered important among all features but we drop them at the very beginning since they have a large amount of missing values. Therefore, we add these 2 columns back to the processed training data.

Also, we can drop those features with 0 importance according to our model to save time and reduce dimension in our experiment with similar approach. As a result, 19 new features, the majority of which are related to FLAG_DOCUMENT, are not considered in our further investigation.

Meanwhile, we can generate new features that attempt to capture what we think may be important for telling whether a client will default on a loan with those comparatively important variables, such as CREDIT_INCOME_PERCENT, the percentage of the credit amount relative to a client's income. In total we generate 5 new features here.

Since the EXT_SOURCE features are regarded as the most important among all, we can use the statistical interpretation to describe these data, like the sum, average, minimum and maximum. 5 more features are added into the original data. This time we do not impute missing values to avoid the effect of extra information since XGboost and LightGBM can both handle missing values.

With above approaches, our local CV score increases to 0.764954671917932 and the submission score rise to 0.76223.

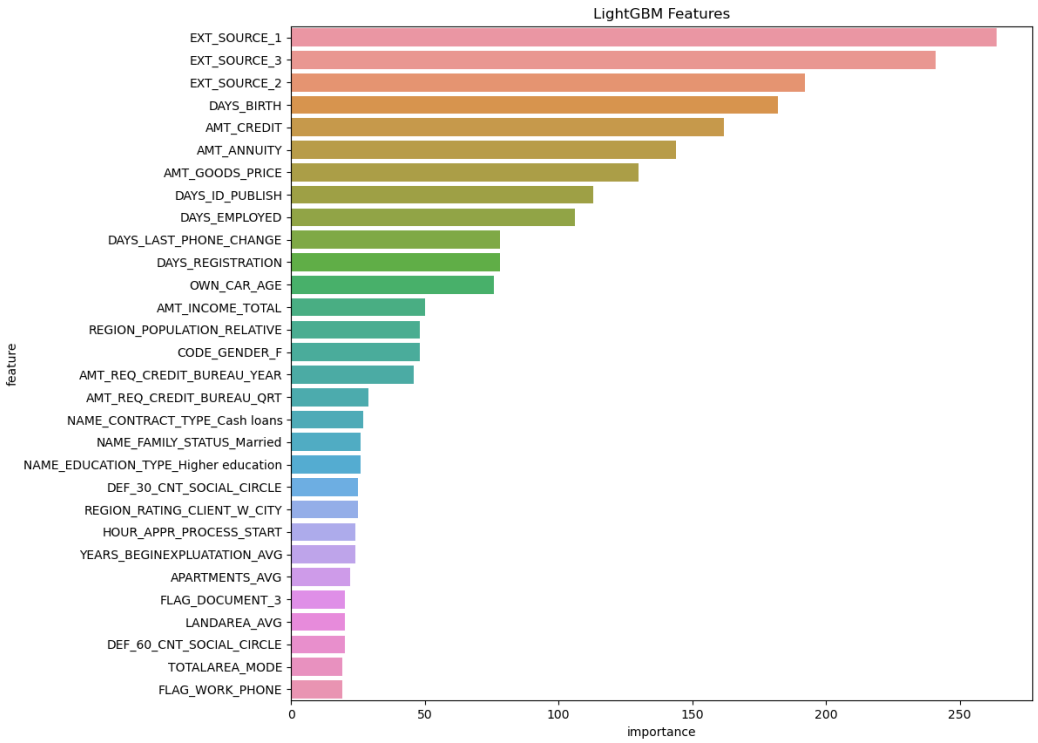

Figure 6: Top important features generated by LGBMclassifier using full training data

4.2 Adding other tables into model

There are 8 csv files given in this contest, but we currently only use 2 of them. We then try to grab the information from other tables to see if our model can give better performance.

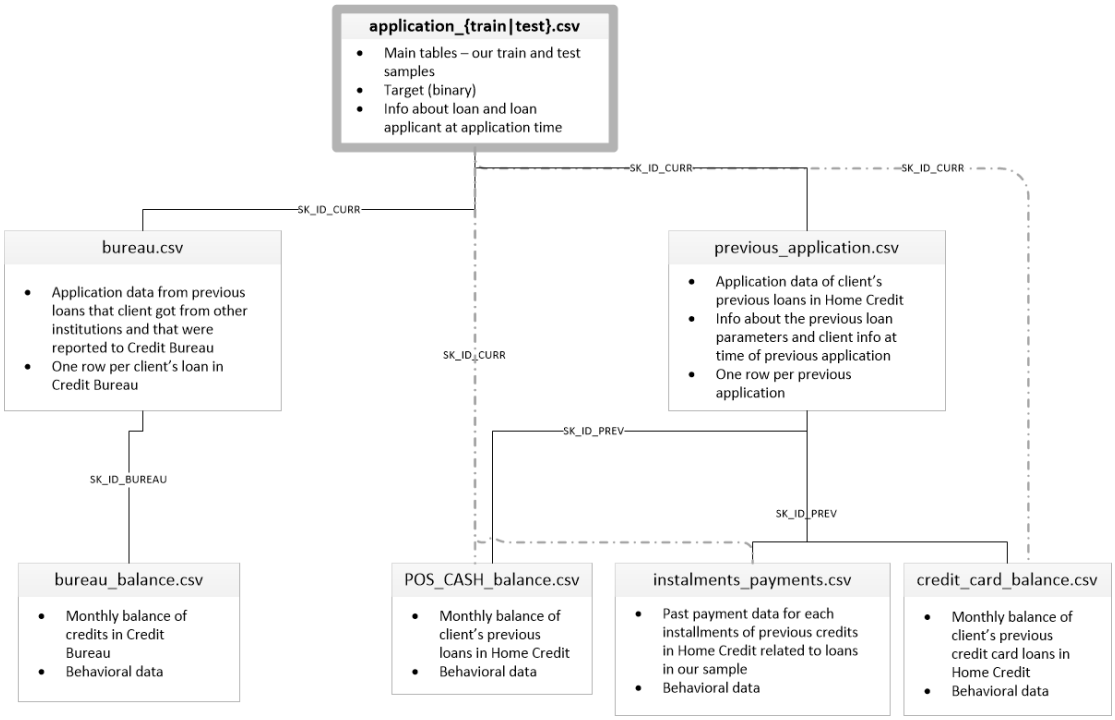

Figure 7: Relationship between the given files

Note that the other 6 files contain information in both the training and testing dataset, so we need to

merge the 2 dataset first and add features from the other tables into the new training set.

Thanks to those who have already shared their experience online, we are able to process the datasets and generate new features. In figure 8, it is shown that one ID in the new training set can correspond to multiple result in bureau dataset, and similar situation happens in other datasets. Therefore, we need to aggregate the records first before looking deeper into the information. We still apply one-hot-encoder to the non-numerical features, but we use aggregation and groups with statistical description in handling numerical variables, such as average, minimum, maximum, sum and variance. Besides, simple calculations like ratio are also utilized in our feature engineering.

	SK_ID_CURR	SK_ID_BUREAU	CREDIT_ACTIVE	CREDIT_CURRENCY	DAYS_CREDIT	CREDIT_DAY_OVERDUE
0	215354	5714462	Closed	currency 1	-497	0
1	215354	5714463	Active	currency 1	-208	0
2	215354	5714464	Active	currency 1	-203	0
3	215354	5714465	Active	currency 1	-203	0
4	215354	5714466	Active	currency 1	-629	0
5	215354	5714467	Active	currency 1	-273	0
6	215354	5714468	Active	currency 1	-43	0
7	162297	5714469	Closed	currency 1	-1896	0

Figure 8: A glance at the bureau file

We then train our LightGBM model in the new training dataset and get a cross-validation score of 0.7858858717988921. The submission file returns a score of 0.7833 on Kaggle. Surprisingly, features that rank top in the performance table are still those generated in the old training data.

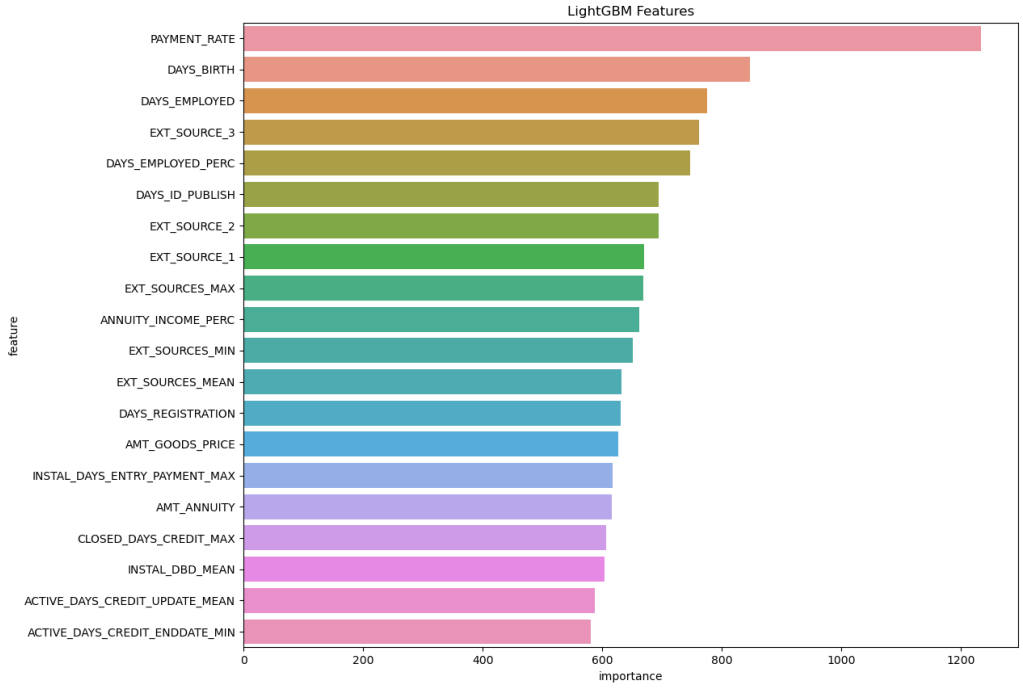

Figure 9: most importance features in the new training dataset

4.3 Dimension reduction and feature selection

In section 4.2, we get a large training sample with 732 features, but over 150 of them have 0 importance, which means they may have very low contribution to the final prediction. So we consider doing feature selection to reduce the dimension and keep the comparatively significant variables by applying approaches like removing collinearity and PCA.

To remove collinearity, we set the threshold to be 0.9, which means we remove the variables that have correlation higher than 0.9 with any other variable. In total we remove around 150 features, but with some important features included. So, we need to add them back to avoid losing information. For PCA, it cannot directly handle missing data, and we do not want to impute to introduce extra information, so it is not adopted in our experiment.

Therefore, we only utilize collinearity removement in our training and testing data and we get a validation score of 0.7858732412286537 and the submission score of 0.7829.

5 Conclusion, future work and acknowledgement

We conduct this simple analysis with various methods and get a best submission score of 0.783, which is an average result among all the contestants. We may consider following steps to improve our model.

First, we can adopt business knowledge and practice into our model so we can understand what is crucial to the final judgment instead of only looking at the information table. Second, since we drop the variables with more than 40% of missing values in the very beginning, we can just keep them and do feature selection after transformation. Third, we can check whether features in other tables have some independent influence on the target. For example, in previous_applicaion file, we can train a model directly using each previous record as a training sample, and corresponding current target as training targets (so previous applications of a same customer will share the target of this customer's current target).

Thanks to those who share their experience online!

Reference

[1] Will Koehrsen, Start Here: A Gentle Introduction, https://www.kaggle.com/willkoehrsen/start-here-a-gentle-introduction

[2] Qinghui Ge, https://github.com/NoxMoon/home-credit-default-risk

[3] Aguiar, LightGBM with Simple Features, https://www.kaggle.com/jsaguiar/lightgbm-with-simple-features

[4] Ashish Patel, https://www.kaggle.com/ashishpatel26/home-credit-default-analysis