Project 1 Home Credit Default Risk

HUO, Sixian 20810798

Introduction

This is a typical binary classification problem with labels, so boosting is a popular way to solve it.

The whole data set is relatively large, therefore, it is time-consuming to understand the meaning of each column. I finally analyzed only two csv files TRAIN and BUREAU.

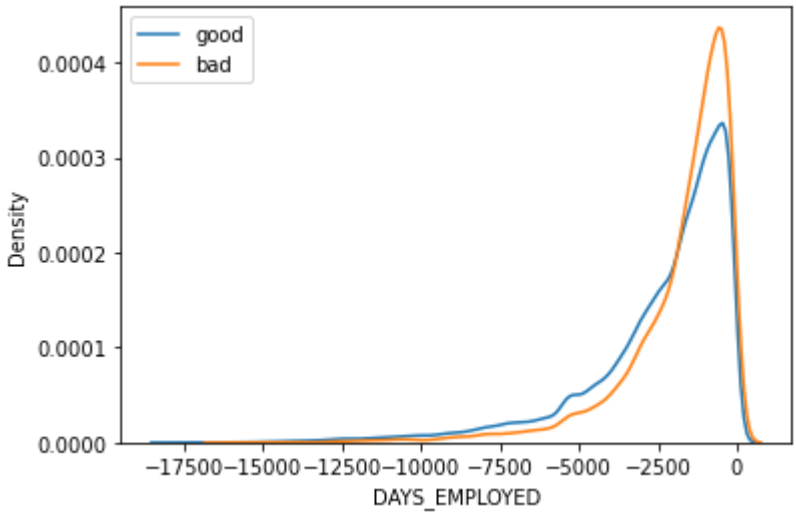
My idea is to make a statistical analysis of the column to find out which of them have a great correlation with target, then build some features based on common sense. I selected LGBM as the model, it has better performance than XGBoost. My final private score on Kaggle is 0.76201.

Submission and Description	Private Score	Public Score
sample_submission.csv 25 minutes ago by SixianHUO final sub	0.76201	0.75861

Feature Selection

First of all, from the simplest point of view, I select the columns with only two types of values for analysis, and count how many positive samples there are among them. Select a column with a threshold greater than 0.02 as the feature.

For columns with multiple types of values, I use kdeplot to observe the differences in their distribution, and select the columns with obvious differences as features.



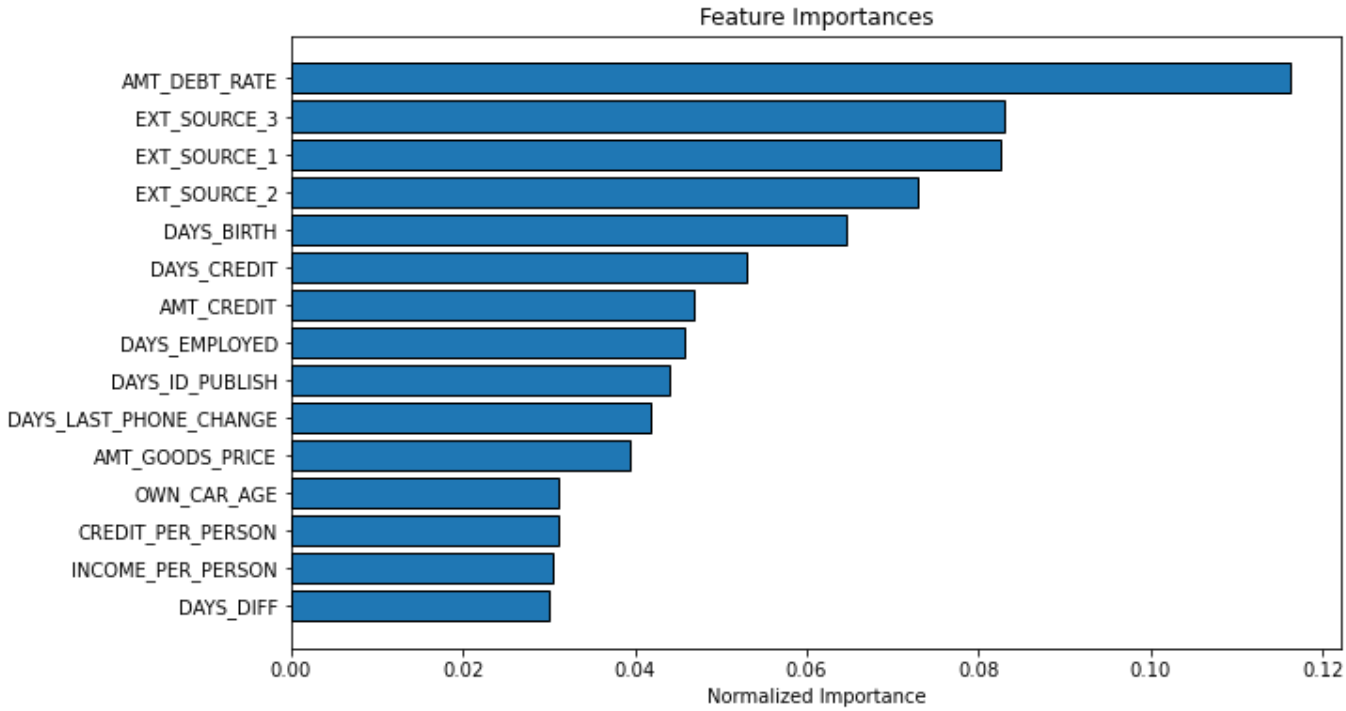
I used the same method in bureau.csv and extracted several features.

Feature Building

Here I constructed several features based on common sense, such as per capita household income, per capita household debt, and debt interest rate. I summarized the house information in one place, but it didn't seem

to perform so well.

The importance of the final features is ranked below.



Training

At first, I chose random forest as the model, but its performance was not good enough. So I find LGBM. 5-KFlod was used during training, and AUC and imp were counted as reference. Finally, the AUC score is very close to the submission score.

```
Training until validation scores don't improve for 100 rounds
[200]   train's auc: 0.799364   train's binary_logloss: 0.546691       valid's auc: 0.761008   valid's binary_logloss: 0.561224
Early stopping, best iteration is:
[285]   train's auc: 0.811855   train's binary_logloss: 0.533845       valid's auc: 0.761399   valid's binary_logloss: 0.553583
Training until validation scores don't improve for 100 rounds
[200]   train's auc: 0.798142   train's binary_logloss: 0.548133       valid's auc: 0.768769   valid's binary_logloss: 0.560114
Early stopping, best iteration is:
[274]   train's auc: 0.809221   train's binary_logloss: 0.536993       valid's auc: 0.769206   valid's binary_logloss: 0.553267
Training until validation scores don't improve for 100 rounds
[200]   train's auc: 0.799029   train's binary_logloss: 0.547155       valid's auc: 0.76222    valid's binary_logloss: 0.559851
[400]   train's auc: 0.82618    train's binary_logloss: 0.519016       valid's auc: 0.763001   valid's binary_logloss: 0.542958
Early stopping, best iteration is:
[361]   train's auc: 0.821526   train's binary_logloss: 0.523879       valid's auc: 0.76325    valid's binary_logloss: 0.545819
Training until validation scores don't improve for 100 rounds
[200]   train's auc: 0.798932   train's binary_logloss: 0.547177       valid's auc: 0.764065   valid's binary_logloss: 0.56283
Early stopping, best iteration is:
[270]   train's auc: 0.809126   train's binary_logloss: 0.536804       valid's auc: 0.764759   valid's binary_logloss: 0.55669
Training until validation scores don't improve for 100 rounds
[200]   train's auc: 0.79872    train's binary_logloss: 0.547369       valid's auc: 0.76649    valid's binary_logloss: 0.560826
[400]   train's auc: 0.825219   train's binary_logloss: 0.520375       valid's auc: 0.766764   valid's binary_logloss: 0.544485
Early stopping, best iteration is:
[358]   train's auc: 0.820332   train's binary_logloss: 0.525413       valid's auc: 0.767137   valid's binary_logloss: 0.547465
```

Conclusion

This warm-up project gives me a general understanding of the process of machine learning. The final score is not high, and there are still many areas that need to be improved. For example, introduce more data, build

more features, etc. The most important thing is to understand the principle of the model and the logic behind the data.