Home Credit Default Risk Project

Zheyue FANG * Chutian HUANG † Yue WU ‡ Lu YANG §

Abstract

This project comes from a competition hosted by Home Credit and Kaggle. In this project, we explore a variety of data about the applicants to predict the probability whether the applicants are capable of repaying their loan. Our work can be manly devided into two parts: feature engineering and model setup. We first generate various statistical features from all the seven tables and then train a LightGBM model to select some most important features. Then we use the selected features to train three different models (Logistic Regression, Random Forest, LightGBM) to fit the data and make prediction. We also apply AUC ROC to compare the results and choose the best model.

1 Problem Statement

This project is from the Kaggle contest: the Home Credit Default Risk. Nowadays, loans have always been an important part of people's lives for quite some time. Each individual has different reasons for borrowing a loan. Loans are also important to lenders. Almost all banking organizations make most of their revenues from the interests generated through loans. However, not all borrowers have the ability to repay the loan, and the lenders can make a profit only if the loan gets paid. The lending organizations therefore face the tough task of analyzing the risk associated with each client to avoid or decrease the losses and incur their huge profit.

Home Credit, a consumer finance provider in central and eastern Europe/ Asia, strives to broaden financial inclusion for the unbanked population by providing a positive and safe borrowing experience. Home Credit hopes to make use of a variety of alternative data-including technology and transaction information to predict their clients' repayment abilities and loan default probability.

We need to analyze the data provided by Home Credit and build a model to predict if each client can have the ability to repay the loan, so that the lending organizations, like Home Credit, are able to only lend money to clients who are likely to repay it and avoid much loss.

2 Data Exploration and Visualization

2.1 Data Description

The data we use is provided by Home Credit and there are seven different sources of data:

(1) application-train.csv(307511 rows and 122 columns): This is the main training data with information about each loan application at Home Credit. Each row has unique id 'SK ID CURR' and the

* zfangaf@connect.ust.hk
† chuangat@connect.ust.hk
‡ ywudb@connect.ust.hk
§ lyangbf@connect.ust.hk

Department of Mathematics, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong.

output label is in the 'TARGET' column. If the TARGET is 0, it means that the loan was paid and if it is 1, it means that the loan was not paid.

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_(
0	100002	1	Cash loans	M	N	Y	0	202500.0	4(
1	100003	0	Cash loans	F	N	N	0	270000.0	12(
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	1(
3	100006	0	Cash loans	F	N	Y	0	135000.0	3(
4	100007	0	Cash loans	M	N	Y	0	121500.0	5(

5 rows × 122 columns

Figure 1: First 5 rows of application-train.csv

(2) bureau.csv(1716428 rows and 17 columns): This dataset consists of data concerning client's previous credits from other financial institutions that were reported to Credit Bureau. Each previous credit has its own row in bureau, but one loan in the application data can have multiple previous credits.

(3) bureau-balance.csv(27299925 rows and 3 columns): It consists of monthly data about the previous credits in bureau. Each row is one month of a previous credit, and a single previous credit can have multiple rows, one for each month of the credit length.

(4) previous-application.csv(1670214 rows and 37 columns): This is the data of previous applications for loans at Home Credit of clients who have loans in the application data. Each current loan in the application data can have multiple previous loans. Each previous application has one row and is identified by the feature SK ID PREV.

(5) POS-CASH-balance.csv(10001358 rows and 8 columns): It consists of monthly balance snapshots about previous point of sales (POS) or cash loans that the clients have had with Home Credit. Each row is one month of a previous point of sale or cash loan, and a single previous loan can have many rows.

(6) credit-card-balance.csv(3840312 rows and 23 columns): It contains the monthly data about previous credit cards clients have had with Home Credit. Each row is one month of a credit card balance, and a single credit card can have many rows.

(7) installments-payment.csv(13605401 rows and 8 columns): This is the data of payment history for previous loans at Home Credit related to the selected loans. There is one row for every made payment and one row for every missed payment. One row is equivalent to one payment of one installment or one installment corresponding to one payment of one previous Home Credit credit related to the selected loans.

2.2 Visualization

We visualize the distribution of repayed loans and not repayed ones in the main table: loan repayed is 91.9% and loan not repayed is 8.07%.

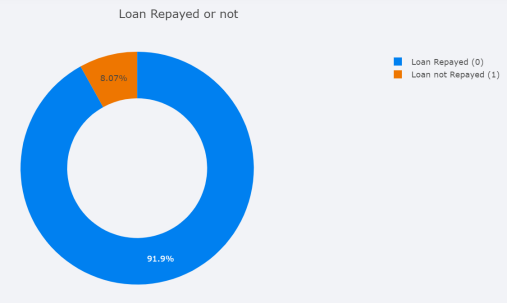

Figure 2: percentage of loan repayed and not repayed

We draw the diagram of distribution of income sources of applicants in terms of loans repayed or not percentage. We find that all the students and businessmen repay loans; working people take the highest percentage of repaying loans while take the highest percentage of not repaying loans, from the diagram.

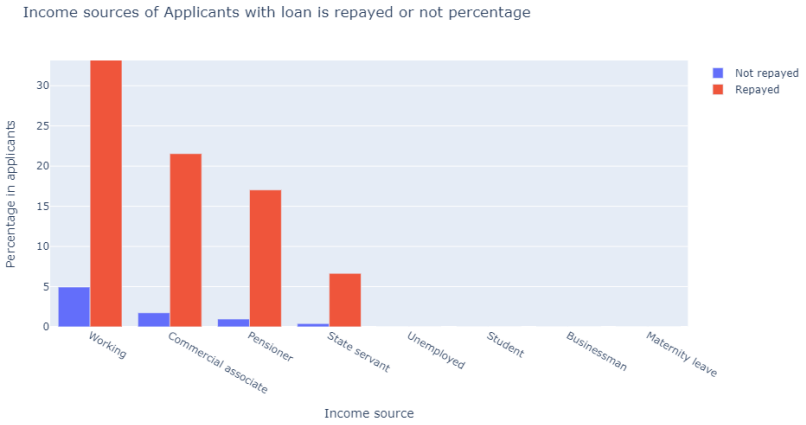

Figure 3: Income sources of applicants with loans is repayed or not percentage

We then draw the diagram of distribution of education of applicants in terms of loans is repayed or not percentage. We can see that people with academic degree are more likely to repay the loan (out of 164, only 3 applicants are not able to repay).

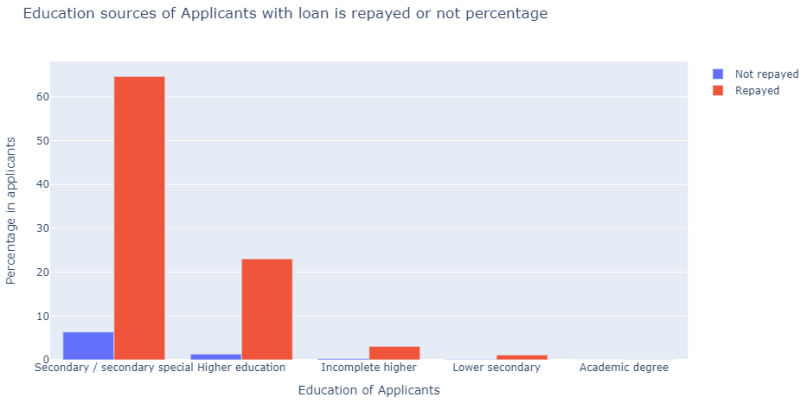

Figure 4: Education sources of applicants with loan is repayed or not percentage

These tables have many missing data so we count each column for missing values. Here we list the count and percentage of missing values for top 10 columns. Therefore we need to handle these missing values, and we will use the simple imputer to complete missing values.

	Count	Percentage
COMMONAREA_MEDI	214865	69.872297
COMMONAREA_AVG	214865	69.872297
COMMONAREA_MODE	214865	69.872297
NONLIVINGAPARTMENTS_MODE	213514	69.432963
NONLIVINGAPARTMENTS_MEDI	213514	69.432963
NONLIVINGAPARTMENTS_AVG	213514	69.432963
FONDKAPREMONT_MODE	210295	68.386172
LIVINGAPARTMENTS_MEDI	210199	68.354953
LIVINGAPARTMENTS_MODE	210199	68.354953
LIVINGAPARTMENTS_AVG	210199	68.354953

Figure 5: count and percentage of missing values for top 10 columns

3 Features Engineering

We first analyze the data in each table and generate more features, and then aggregate the features from different tables, finally use LightGBM to select significant features.

3.1 Statistical Features

We generate features based on some assumptions e.g. people are able to repay loans when their income can cover the credit or annuity; people are more likely to repay loans when they can support their families easily. For data in the application table (main table), we use simple arithmetic to generate new features, which can describe applicants' finance situation (including the ratio of credit to income, the ratio of annuity to income, average monthly credit, average monthly annuity, etc), family situation (including numbers of adults and children, per capita income and per capita consumption), house situation and life experience, all kinds of time intervals, etc.

For other tables, besides simple arithmetic on numerical information, we create more statistical features by counting the number of loans and the number of types of loans for each applicant. We use one-hot encoding to convert text data to numerical data. We also groupby different categorical variables to generate various statistics of numerical variables, filling all the NaN values with 0. We use simple imputer to complete the missing data.

We merge all the features together after processing the data in every single table. Those features include numerical variables and categorical variables, with all the categorical ones one-hot-encoded. So far, we have 515 features, and the next step is to select most important part of them before model stacking.

3.2 Model Features

We select features using the GBDT model in LightGBM (Light Gradient Boosting Machine) classifier. LightGBM is a gradient boosting framework based on decision tree algorithms, and it is mainly used for ranking and classification in machine learning tasks. It runs much faster than XGBoost but produces as accurate prediction as XGBoost does. As a tree-based model, LightGBM can evaluate the importance of features.

The importance of feature j is defined as the average of its importance in each tree:

$$\hat{J}_j^2 = \frac{1}{M} \sum_{m=1}^{M} \hat{J}_j^2(T_m),$$

where M is the number of trees, and the importance of feature j in the tree T is

$$\hat{J}_j^2(T) = \sum_{i=1}^{L-1} \hat{\imath}_t^2 1(v_t = j).$$

Here L denotes the number of terminal nodes (hence $L-1$ is the number of non-terminal nodes), and v_t is the splitting variable associated with node t, and $\hat{\imath}_t^2$ means the empirical improvement in square error after the split.

In practice, we use our 515 features as input data to train LightGBM model with early stopping, and then we estimate the feature importance and pick the features with feature importance values greater than a preset threshold. In the end, 233 features are selected. Figure 6 shows the top 15 most important features selected by LightGBM, where higher scores represent greater importance.

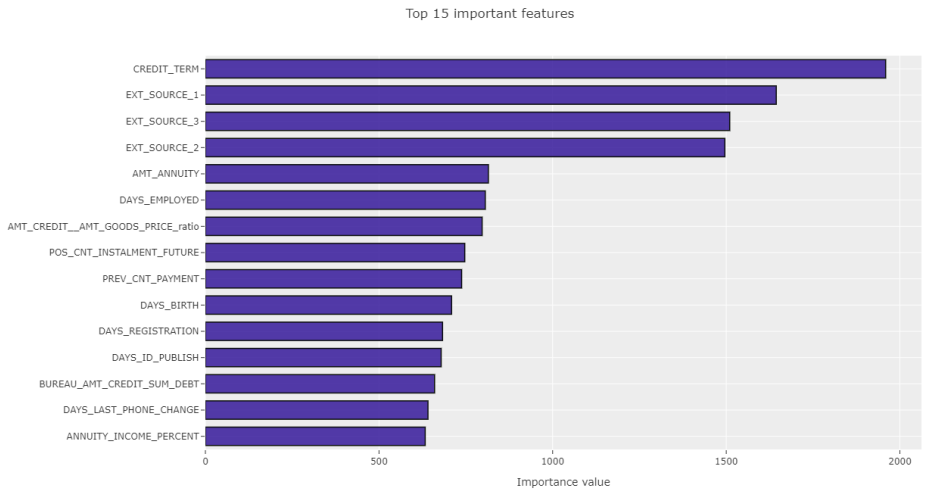

Figure 6: Top 15 most important features selected by LightGBM.

We have another idea of generating model features, also based on machine learning classifiers, in addition to feature selection. We can train a model (e.g. LightGBM, Logistic Regression, Random Forest, etc.) via cross validation and get the out-of-fold prediction values, and then join them to the current features as our new features. However, we did not try this method due to time limit.

4 Prediction

We have selected 233 features to make the prediction. We need to use these features to predict whether the applicant is a defaulter or not. This is a binary classification problem, we employ three different methods— Logistic Regression, Random Forest, LightGBM to solve it. Once we made the prediction, we use ROC AUC (Receiver Operating Characteristic Area Under the Curve) to evaluate our results. It is a common method to select the best binary classifier. We also draw the confusion matrix, recall matrix and precision matrix of these methods to compare three classifiers. These figures are drawn in a way of heat map and can represent how well a model makes its prediction.

4.1 Logistic Regression

The logistic regression uses a logistic function to model a binary variable. The basic logistic function is

$$Logit(p) = \ln(\frac{p}{1-p}). \tag{1}$$

The logistic regression model is

$$\ln(\frac{p}{1-p}) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p := \theta^T x, \tag{2}$$

$$p = P(y=1|X) = \frac{1}{1+e^{-\theta^T x}} := h_\theta(x). \tag{3}$$

For m samples, the common cost function of logistic regression is cross entropy with regularization:

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m}(y^{(i)} \ln h_\theta(x^{(i)}) + (1 - y^{(i)}) \ln(1 - h_\theta(x^{(i)})))] + \frac{\lambda}{2m}\sum_{i=1}^{p}\beta_i^2. \tag{4}$$

Through minimizing the cost function, we can train the model. We first tune the optimal inverse regularization parameter (C Value) by choosing different c, as shown in figure 7. The Optimal C value is 0.001.

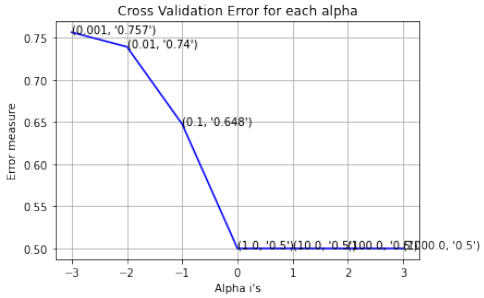

Figure 7: Line Chart of C Value and Cross Validation AUC Score

And the ROC, confusion matrix, recall matrix, precision matrix are shown in figure 8 and 13.

4.2 Random Forest

Random forest is a model made up of many decision trees. It uses the bagging sampling method to select training samples and split features randomly each time to generate a decision tree. Rather than simply averaging the prediction of different trees like bootstrap, this "featuring bagging" process promises that features won't be selected in many of the trees and strongly correlated, thus ensuring the accuracy.

The ROC, confusion matrix, recall matrix, precision matrix of random forest are shown in figure 9 and 13.

4.3 LightGBM

LightGBM (Light Gradient Boosting Machine) is a highly efficient gradient boosting framework based on decision trees. It is an elevated version of XGB (Extreme Gradient Boosting) in the following three aspects.

LightGBM grows tree vertically while other algorithm grows trees horizontally, which means that LightGBM grows tree leaf-wise while other algorithm grows level-wise. Compared to level-wise algorithms, the leaf-wise algorithm can reduce more loss when growing on the same leaf, thus bringing more accuracy and faster speed.

The decision tree use histogram algorithm rather than pre-sorted algorithm for data structure and improves cache hit rate. It also uses EFB (Exclusive Feature Bundling) to solve the sparsity of the data.

LightGBM uses GOSS (Gradient-based One-Side Sampling) for sampling, which retains instances with large gradients while performing random sampling on instances with small gradients to reduce the samples and maintaining accuracy.

Based on the above three improvements to XGB, LightGBM is a fast, distributed, and high-performance algorithm used for ranking, classification and many other machine learning tasks. Here we apply LightGBM for our Home Credit Defaulter problem and get the ROC, confusion matrix, recall matrix, precision matrix shown in figure 10 and 13.

Comparing the above results, LightGBM gives the best performance.

Table 1: AUC of three models

Method	Logistic Regression	Random Forest	LightGBM
AUC	0.764	0.750	0.785

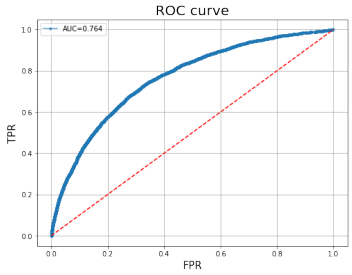

Figure 8: Logistic Regression

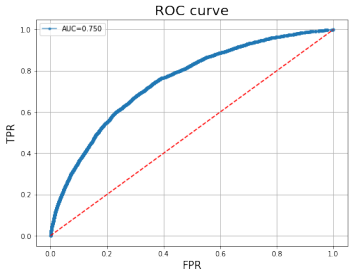

Figure 9: Random Forest

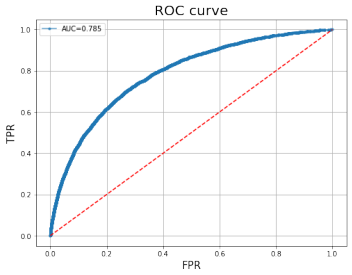

Figure 10: LightGBM

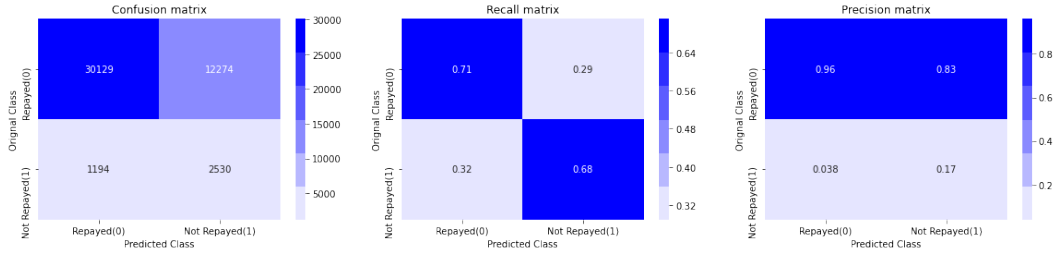

Figure 11: Logistic Regression

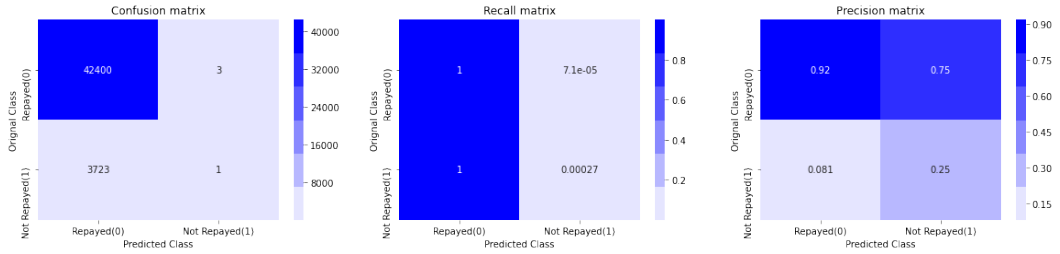

Figure 12: Random Forest

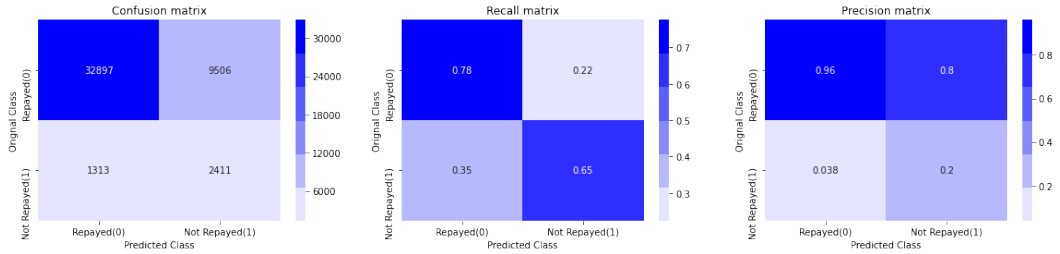

Figure 13: LightGBM

5 Conclusion

This project explores seven tables which contain information of applicants to predict the probability of not repaying loan for each applicant. We use simple arithmetic and statistic to create more features based on original data. We aggregate all the features(515) and use LightGBM model to select some important features(233). In fact, other ways to generate features may be better for prediction but our strategy is simple. Then we use logistic regression, random forest and lightgbm to predict the probability of not repaying loan. We find that lightgbm overperforms logistic regression and random forest.

6 Individual Contribution

- Zheyue FANG: Report writing; Generate statistical features
- Chutian HUANG: Report writing; Prediction based on selected features
- Yue WU: Report writing; Feature engineering and selection
- Lu YANG: Report writing; Data exploration and visualization

References

[1] Home credit default risk. `https://www.kaggle.com/c/home-credit-default-risk/`.

[2] Home credit default risk end to end machine learning project. `https://medium.com/@praveenkotha/home-credit-default-risk-end-to-end-machine-learning-project-1871f52e3ef2`.

[3] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[4] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems*, pages 3146–3154, 2017.