Featured Prediction Competition of Home Credit Default Risk

Xinyi Wei[*]
Department of Mathematics
xweiap@connect.ust.hk

Liangyawei Kuang[*]
Department of Computer Science
lkuang@connect.ust.hk

Abstract

We take the predicition problem of Home Credit Default Risk at Kaggle as a case study and used several different statistical machine learning methods based on given data. At first, we found several existing methods online and reimplement them to do the modeling part, including Stochastic Gradient Descent (SGD) methods (Logistic Regression and Support Vector Machine) and general methods for distributed data (Random Forest Classifier and Extra Trees). However, none of them could get a private score over 0.780 or a public score over 0.785. In this case, we use XGBoost Classifier and LightGBM Classifier to get better results. We also try stacking method to improve our scores. Finally, we could get a ranked 13.38% private score with our solutions. We compared our results, studied the existing top solutions, and analyzed the reasons for their success to formulate our conclusions for this case study in the end.

1 Introduction

Many people are suffering from getting loans due to their insufficient or non-existent credit histories, and Home Credit is an international consumer finance provider that focuses on responsible lending to people with poor credit history. To make sure underserved people have a better loan experience, Home Credit uses a variety of alternative data to predict the repayment abilities of clients.
In this case, Home Credit form a Kaggle challenge to let Kagglers help them unlock the potential of given data set by using various statistical and machine learning methods so that they could ensure that clients with repayment abilities will not be rejected and loans are given with a principal, maturity, and repayment calendar.

1.1 Data Description

There are 7 different sources of data given by Home Credit, including:

(1) application_train/application_test: the training data set and the testing data set with the information about each loan application at Home Credit. Every loan has its own row and is identified by the feature "SK_ID_CURR".

(2) bureau: data concerning clients' previous credits from other financial institutions. Each previous credit has its own row in bureau, but one loan in the application data can have multiple previous credits.

[*] equal contribution

Work arrangement: Both of the authors worked on information searching, coding with Jupyter, report writing, statistical analysis, and presentation preparation in this project.

Remark: Both of the authors have fully participated in this project. This report is used as a part of a final project in MATH 5470 at Hong Kong University of Science and Technology (Clear Water Bay, Kowloon, Hong Kong).

(3) bureau_balance: 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) credit_card_balance: 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.

(5) installments_payment: payment history for previous loans at Home Credit. There is one row for every made payment and one row for every missed payment.

(6) POS_CASH_BALANCE: monthly data about previous point of sale or cash loans 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.

(7) previous_application: 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".

1.2 Objectives

The main objective in this case study is to make prediction for potential defaulters based on the given data of applicants.

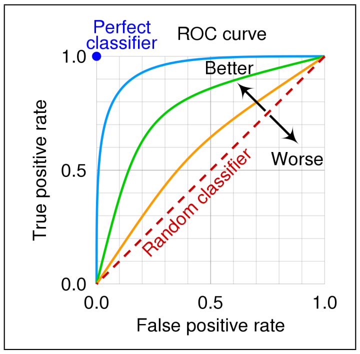


Figure 1: ROC Curve

We use "Receiver Operating Characteristic Area Under the Curve", which is known as "ROC AUC" or "AUROC", to prejudge our submission. Due to the definition, the Receiver Operating Characteristic (ROC) graphs the true positive rate versus the false positive rate, the Area Under the Curve (AUC) calculate the integral of the curve, which is a numerical metric between 0 and 1. According to the ROC AUC, the higher score of this metric means a better model.

1.3 Constraints

(1) It is hard to have a model with high interpretability when it comes to partial important to decide whether someone is a good loan or a defaulter.

(2) As this case study is a kind of decision making problem, but not an "one-shot" learning problem. So the latency could be ignored as the models may take a period to make a prediction.

(3) The fault tolerance should be low as an error on missing a potential defaulter could cost huge financial losses to the Home Credit, even though the error cost is due tot he funds related to each loan.

2 Data Processing

2.1 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is an open-ended process where we calculate statistics or make figures to do some basic data processing to find trends, anomalies, patterns, or relationships based on given information. The goal of EDA is to dive deeper in the data set (for example, to get the hidden patterns). It usually start with a high level overview, then narrows in to (a) specific expert area(s). In this case we could decide which features are going to be used.

2.2 Missing Value

An easy way to deal with missing value is to delete the columns with a high missing value rate. While at the beginning, we can not decide this cheap method is useful or not, so we keep them at first. Also, in our later work, some model con handle missing values without imputation.

2.3 Outliers Processing

(1) We found some entries at having a value equal to 365243.0 (which is about 1000 years) for the "DAYS" feature. We also found some loans could be traced to a long period of time ago, which we take them as outliers because this kind of funds would not be added a lot. In this case, we only kept the entries within a 100 years' span.

(2) Addtionally, we found some features having great values, for example, the "SELLERPLACE_AREA" had a max value of 4,000,000, and the "AMT_PAYMENT_CURRENT"also had a maxing value equal to 4,289,207 (also about 4,000,000), etc. All of those entries could be wrong as they are too large compared with other values.

(3) In this training data, we saw there are some "XNA" value in some rows at the "CODE_GENDER" feature, we delete those entries as we can not find such category in the testing data.

(4) There are also some negative values, for example, "SELLERPLACE_AREA", which csn not be less than 0. We also take them as outliers. And we take the absolute value of some negative entries as they should be positive due to it financial meaning.

3 Feature Engineering

After data processing, we start to focus on featuring engineering, where we used our understanding about data sets to come up with new ideas for data transformation.

3.1 Feature Engineering Methods

There are several methods we used in this process, including:

1. **feature scaling:** Feature scaling is aimed to normalize the range of features of data. This method is based on domain knowledge to make the data more robust by cleaning outliers and reranging data.
2. **feature aggregating:** For some tables like "previous_applications.csv", "bureau.csv", etc., which includes the data about previous loans. In this case, there are several rows in those table which are related to one loan. We aggregate them based on the current applicant ID (for example, "SK_ID_CURR") to merge this kind of data with the main table. During this process, we use "mean", "max", "sum", and other aggregation features.
   Besides, we also merged some of the categoried with a high frequence so that we can better catch the trend.
   There are also some tables related to current clients' previous loans. Those tables contain data record of transactions each month (for example, "installments_payments.csv,") so that we need to aggregate over the past application ID ("SK_ID_PREV"), then the current application ID ("SK_ID_CURR"). For these table, aggregating based on limited time span

also works (for example, for the last three years). Those method can be helpful for making further prediction.

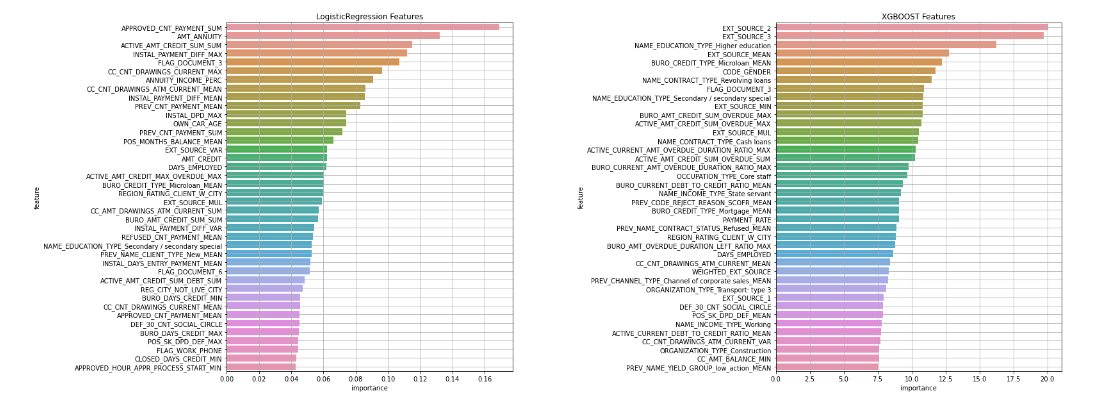


Figure 2: Feature Importance.

3. **categorical features:** For this part, we used One-Hot Encoding (OHE) for columns. To avoid increasing dimensionality by many-folds using OHE, we encode the categorical features. We also derive features based on the categorical relations by grouping the data on categorical combinations and imputing the aggregates for each group as features.
4. **historical pass & rejection features:** We also merged the "data_pre_application" tables with the "prev_agg" tables based on the current ID ("SK_ID_CURR") and "NAME_CONTRACT_STATUS_Approved" feature.
5. **active & closed features:** We also used the same method on "historical pass & rejection features" to "active & closed features", which is merged from "data_bureau" to "bureau_agg" tables based on current ID ("SK_ID_CURR") and "CREDIT_ACTIVE_Active" feature.
6. **size features:** We calculate group size based on current ID.
7. **EXT_SOURCE features :** Based on EDA, we realize that "EXT_SOURCE" features are highly related to the Target. From this point, we assume it could be used to distinguish defaulters and non-defaulters. It is also observed from the model based Feature Importances. Features generated from "EXT_SOURCE" features host the highest score among all the other features.
8. **merge:** We merged the helper tables with the main tables based on the current ID ("SK_ID_CURR"), using the left outer join, so that we could keep the current applicants, even some of them do not have any past application history.
9. **EDA analysis-filter feature:** In this part, we built a filter and take the result to deal with the training set. By defining ’missing_rate’ and ’count_unique’, we drop some isolated data before applying further methods.

3.2 Data Standardization

After feature engineering, we standardize the data through removing the ’SK_ID_CURR’ from training set and testing set and extracting the class labels for training set. At last, we replace all the "NaN" values in standardized training set and testing set to zero.

4 Methods

As the given data are related to each other. It is very natural to think of the idea to use gradient based methods or methods which are suitable for representing tree like distributed data structure.We implementing some basic gradient based methods and some other methods which are fitting distribution data set. Then we dive deeper based on gradient based methods and the XGBoost and lightGBM libraries.

4.1 SGD Logistic Regression

We used SGD Classifier and considered log-loss and L2 penalty. For Hyperparameter Optimization, we used Grid Search Cross Validation. For predictions on cross validation, we used Stratified K-Fold method with predictions like Out-of-fold predictions. Because we are optimizing the hyperparameters for AUC, and also dealing with unbalanced data, we tuned the threshold for best True Positive Rate (TPR) and least TPR, by using J-Statistic.

4.2 SGD Support Vector Machine

Support Vector Machine (SVM) is a famous discriminative model, which focus on finding a optiaml hyperplane to devide Classes with positve labels from those with negative lables. In this section, we retrained an SGD Classifier with L2 penalty but with hinge loss for the case of Linear SVM. We did not use Kernel method for SVM as for the long train time and high complexity.

4.3 Random Forest Classifier

Random Forest is a method based on decision trees. In this section, we used Bagging technique and trained a Random Forest Classifier. For some hyperparameters of the classifier, we used Randomized Search Technique to solve it.

4.4 Extra Trees Classifier

This classifier had very similar results with Random Forest Classifier. The main difference between them is that Extra Tree focus on random values for Information Gain during deviding the data for some numeric features.

4.5 LightGBM Classifier

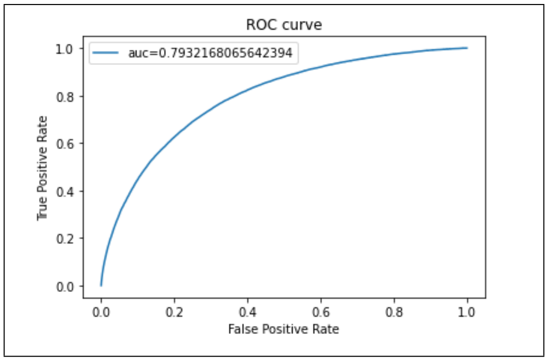


Figure 3: Experiment result for LightGBM.

When dealing with ensembles method, it always comes up with boosting methods at first, especially compared with other classifiers. We experienced this method in our case study and it showed great performance.
LightGBM, as a boosting method, works by modeling recirsively based on past errors and improve (reduce) them in the further stage. It can also work with any given loss function, which makes it superior.

4.6 XGBoost Classifier

XGBoost is also a boosting method, which is very similar to LightGBM. It works based on splitting the trees level-wise while XGBoost works by spillting trees leaf-wise.
Compared with LightGBM, it did not have better performance. However, it is widely shown that XGBoost has less difference between the training and the cross-validation scores. This fact may support that XGBoost has less overfitting problem.

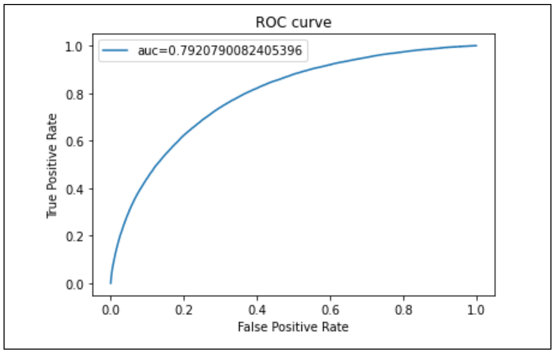


Figure 4: Experiment result for XGBoost.

4.7 Stacking Classifier

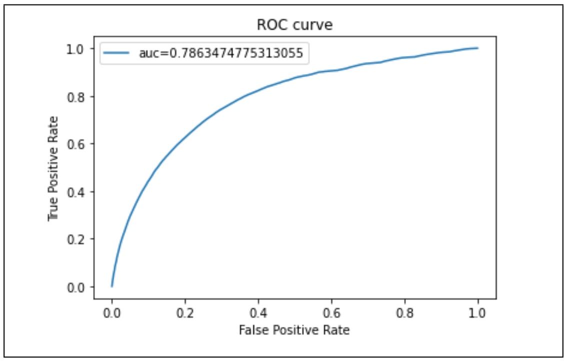
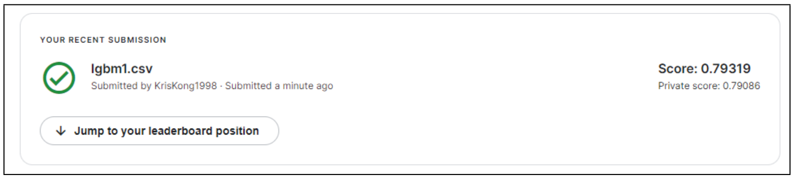


Figure 5: Experiment result for Stacking.

As we were working on ensembling methods, we gave a try for stacking method in the end. In this section, we used Logistic Regression, LightGBM Classifier, and XGBoost Classifier and a Bayesian Optimizier as a meta classifier. By ensembling in this way, we got the best performance among all methods before.
This great performance is due to the fact that the stacking classifiers usually require a amount of diverse and uncorrelated predictions.

5 Results and Comparison

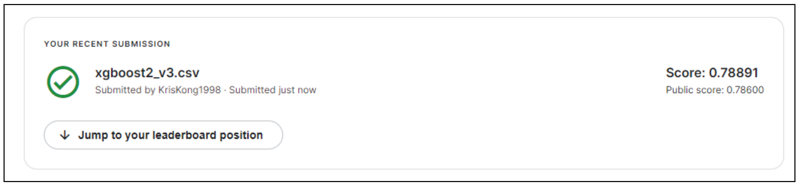
Finally, our model could have private score as 0.79319, which could be ranked as 961/7180 (13.38%) in this competition. However, our public score is only 0.79227, which is a average score among all teams.
The full results and comparison are shown as Figures below and Table 1.


YOUR RECENT SUBMISSION

lgbm1.csv
Submitted by KrisKong1998 · Submitted a minute ago

Score: 0.79319
Private score: 0.79086

Figure 6: Submission result for LightGBM.



YOUR RECENT SUBMISSION

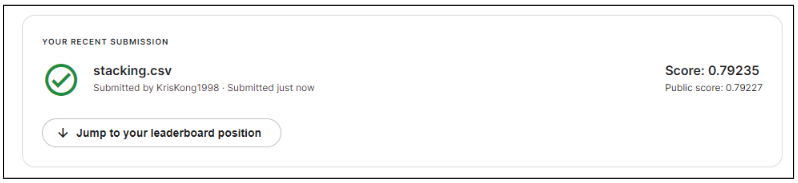
xgboost2_v3.csv

Submitted by KrisKong1998 · Submitted just now

Score: 0.78891

Public score: 0.78600

Figure 7: Submission result for XGBoost.



YOUR RECENT SUBMISSION

stacking.csv

Submitted by KrisKong1998 · Submitted just now

Score: 0.79235

Public score: 0.79227

Figure 8: Submission result for Stacking.

6 Existing Solutions

As this competition is over, we also studied some winners' solutions.

6.1 1st Place Solution

This winner's solution highlighted the importance of feature engineering, which could be proved more useful than tuning and stacking.
They use label encoding for categorical features of some tables. One of the most importance is the "neighbors_target_mean_500", which is the mean value of the targets of 500 nearest neighbors based on "EXT_SOURCE" and "CREDIT_ANNUITY_RATIO".
They did the aggregation by only taking the values for a certain period of time. They emplyed a Forward Selection Technique to deal with the feature reduction. As for modeling, they trained LightGBM and XGBoost models with stratified K-fold Cross-Validation.

6.2 2nd Place Solution

This team has a complex work arrangements. Except for feature engineering, they also use some end to end CNN and RNN to get time series features from tables.
For feature engineering, they used various methods, including PCA, UMAP, and T-SNE to reduce the dimensions.They implied LightGBM (with dart), Carboostm DAE for modeling, training and verification. They also applied blending methods in the end.

6.3 10th Place Solution

For this team, feature engineering also made lots of points. They used information from annuity amount, number of payments and credit amount to compute the interest rates from previous applica-

Table 1: Comparison among different methods

Methods	ROC-AUC Score	Private Score	Public Score
SGD Logistic Regression	0.79073	0.77848	0.78283
SGD Support Vector Machine	0.79155	0.77955	0.78299
Random Forest Classifier	0.77759	0.76457	0.77192
Extra Trees Classifier	0.77209	0.76164	0.77116
LightGBM Classifier	0.79322	0.79319	0.79086
XGBoost Classifier	0.79208	0.78891	0.78600
Stacking Classifier	0.78635	0.79235	0.79227
Highest Score on Kaggle	N/A	0.80570	0.81724

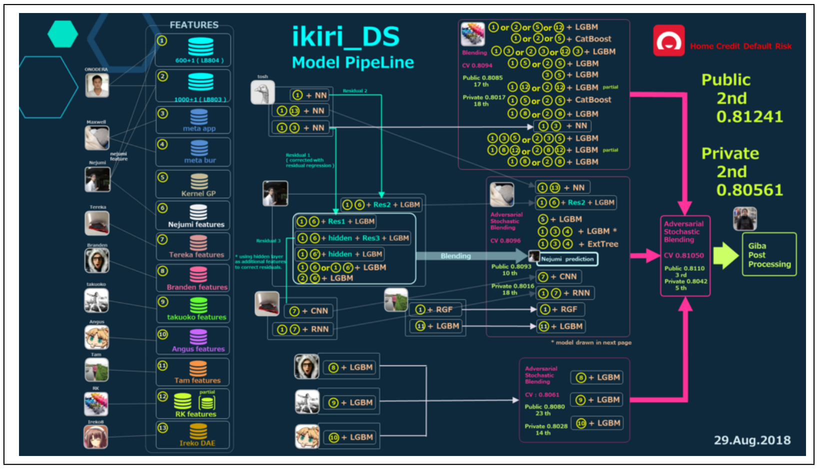


Figure 9: Work arrangement of 2nd Place team.

tions. They also used LightGBM to predict the missing value of "EXT_SOURCE" features. They used the past several months' data seperately and aggregated over current ID for aggregations and implemented LightGBM for feature selection. They also experienced stacking method in the end.

7 Conclusion

Based on statistical analysis above, we conclude several points as below:

1. Feature engineering is important when making difficult predictions. In this case, have some expert preknowledge could be more important than having great complex model as these preknowledge could help dive deeper on data set.
2. For stacking methods, we did not implement further with more diverse basic classifiers due to time and human source limits. If we trained different sets of features based on more classifiers, we may have much better results.
3. Among all the top 10% solutions, it seems that feature engineering plus end to end training is a great way to deal with this case study. While for top 10 solutions, finding important features based on human knowledge guaranteed their gold medals.

References

[1] Rao, R. (2020, Nov 1). *HOME CREDIT DEFAULT RISK — An End to End ML Case Study — PART 2: Feature Engineering and Modelling.* `https://medium.com/thecyphy/home-credit-default-risk-part-2-84b58c1ab9d5`

[2] `https://www.kaggle.com/c/home-credit-default-risk/discussion/64821`

[3] `https://www.kaggle.com/c/home-credit-default-risk/discussion/64722`

[3] `https://www.kaggle.com/competitions/home-credit-default-risk/discussion/64598`