MAFS6010z Project 1
Home Credit Default Risk Competition

Changxi Liu, Liangshu Wang, Yang Yu, Yifei Sun
MSc of Financial Mathematics
The Hong Kong University of Science and Technology

Abstract

This report details our approach to the Home Credit Default Risk competition on Kaggle, which aims to predict clients' ability to repay loans. We performed extensive exploratory data analysis on the applications dataset to understand the features, target variable distribution, missing values, anomalies, and data types. For feature engineering, techniques like feature creation, transformation, aggregation and encoding were utilized to construct an enriched dataset. A Light Gradient Boosting Model combined with bagging classifier was implemented as the prediction model. Bayesian optimization was employed for hyperparameter tuning. The model achieved an AUC of 0.77 on the validation set. On the competition test set, it attained a score of 0.7688, demonstrating its robustness. The feature importance analysis revealed that the average of external data sources and loan amount to income ratio were the top predictors. While promising, further enhancements through expanded feature engineering and modeling could potentially improve accuracy.

1 Introduction

The Home Credit Default Risk competition on Kaggle addresses a significant challenge in the financial world: predicting the ability of individuals with limited or no credit history to repay loans. Such individuals often face difficulties in securing loans and are vulnerable to predatory lenders. Home Credit aims to ensure a positive borrowing experience for this demographic by utilizing alternative data sources, including telecommunications and transactional information. This competition invites participants to enhance the predictive power of Home Credit's data, ensuring fair loan opportunities for all. This report presents our approach to the challenge, detailing the data exploration, methodology, and results obtained from our predictive models.

2 Data Exploration

Data exploration serves as an indispensable phase in analytical endeavors, offering a window into the dataset's inherent characteristics, quality, and underlying patterns. Within the ambit of the Home Credit Default Risk competition, a profound comprehension of the data stands paramount to forecast loan repayment propensities with precision. This segment embarks on a preliminary scrutiny of the furnished datasets, accentuating salient attributes, statistical metrics, and initial insights that will lay the groundwork for ensuing analytical methodologies. For the scope of this report, our emphasis is predominantly on the Applications data.

2.1 Dataset Overview

The dataset, denoted as `application_train.csv`, encompasses static data pertaining to each loan application, with each row signifying a distinct loan. This dataset comprises a total of 307,511 entries, each characterized by 122 attributes. The pivotal attribute, the target variable, delineates the client's payment behavior. Specifically, a value of 1 is assigned if the client experienced payment delays exceeding X days for any of the initial Y installments of the loan. Conversely, a value of 0 indicates adherence to the payment schedule.

2.2 Examine the Distribution of the Target Column

The target variable, which we aim to predict, represents the loan repayment status of clients. Specifically, a value of 0 indicates timely loan repayment, while a value of 1 signifies that the client encountered payment difficulties. An initial analysis of the distribution of loans across these categories reveals a class imbalance. It is evident that the majority of loans have been repaid on time, resulting in a skewed representation of the two classes. As we progress to advanced machine learning methodologies, it will be imperative to account for this class imbalance, potentially through techniques such as class weighting based on their prevalence in the dataset.

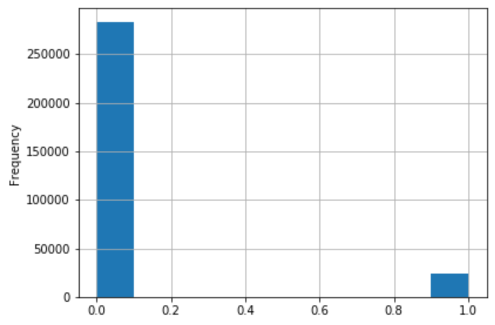


Figure 1: The Distribution of the Target Column

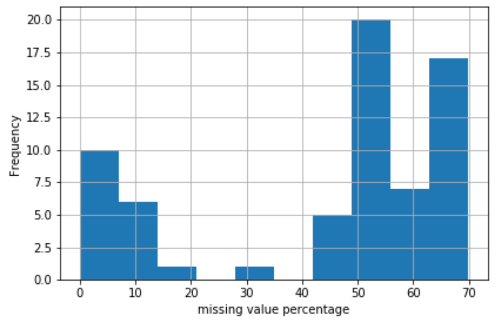


Figure 2: The Distribution of the Missing value percentage

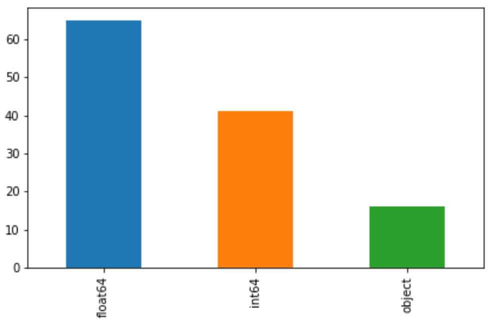


Figure 3: The Distribution of Columns Type

2.3 Examine Missing Values

Subsequently, we turn our attention to assessing the presence and proportion of missing values across columns. An examination of the application training dataset reveals that out of the 122 columns, 67 exhibit missing values.

2.4 Examine Column Types

We proceed to examine the distribution of columns based on their data types. Both ‘int64‘ and ‘float64‘ represent numeric variables, which could manifest as either discrete or continuous attributes. Conversely, columns of type ‘object‘ encompass string values, signifying categorical features.

2.5 Examine Anomalies

During EDA, it's important to watch out for unusual data points, which might be errors or extreme but valid values. A simple way to spot these anomalies is by checking the column statistics with the 'describe' method. The numbers in the DAYS_BIRTH column are negative because they are recorded relative to the current loan application. To see these stats in years, we can mutliple by -1 and divide by the number of days in a year

Statistic	DAYS_BIRTH
Count	307,511
Mean	43.94
Standard Deviation	11.96
Minimum	20.52
25th Percentile	34.01
Median	43.15
75th Percentile	53.92
Maximum	69.12

Figure 4: Age Statistics

Statistic	DAYS_EMPLOYED
Count	307,511
Mean	63,815.05
Standard Deviation	141,275.77
Minimum	-17,912
25th Percentile	-2,760
Median	-1,213
75th Percentile	-289
Maximum	365,243

Figure 5: DAYS of EMPLOYED Statistics

Upon analyzing the provided statistics, the age data appears to be relatively consistent and within expected bounds. The range, mean, and standard deviation values all suggest a typical distribution of ages. Conversely, the days of employment data raises some concerns. While negative values are expected, given that the data likely represents the number of days before the current application that the person started working, the maximum value of 365,243 days (equivalent to over 1,000 years) is clearly anomalous. Such a value is not only implausible but also indicates potential data errors or placeholders that might have been used in the dataset. It's crucial to address these anomalies, either by further investigation or data preprocessing, before proceeding with any predictive modeling, as they could significantly skew the results. As a solution, we will fill in the anomalous values with 'np.nan'.

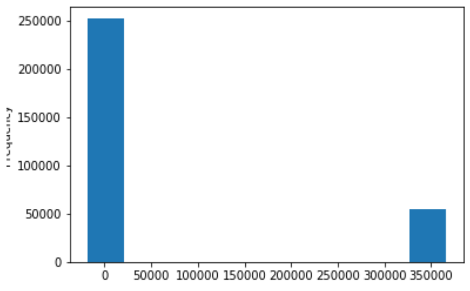


Figure 6: The Distribution of Days of Employment

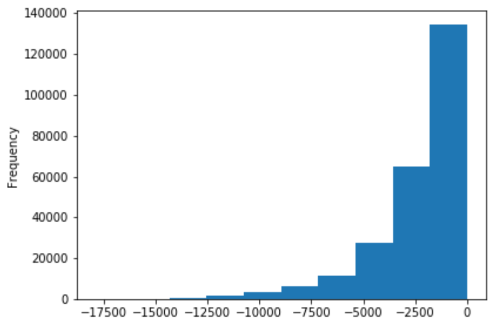


Figure 7: The Distribution of Days of Employment after Data Preprocessing

2.6 Correlation Check

By looking at the correlation between the target variable and the features, we can find the features with high predictive power. Here we check the abs value of the correlation since the relationship of both positive and negative are important.

Feature	Corr
EXT_SOURCE_3	0.178919
EXT_SOURCE_2	0.160472
EXT_SOURCE_1	0.155317
DAYS_BIRTH	0.078239
DAYS_EMPLOYED	0.074958
REGION_RATING_CLIENT_W_CITY	0.060893
REGION_RATING_CLIENT	0.058899
NAME_INCOME_TYPE_Working	0.057481
NAME_EDUCATION_TYPE_Higher education	0.056593
DAYS_LAST_PHONE_CHANGE	0.055218

Figure 8: Most Correlated TOP10

Feature	Corr
ORGANIZATION_TYPE_Industry: type 7	0.000094
ORGANIZATION_TYPE_Advertising	0.000117
FLAG_DOCUMENT_20	0.000215
ORGANIZATION_TYPE_Legal Services	0.000236
NAME_HOUSING_TYPE_Co-op apartment	0.000312
FLAG_DOCUMENT_5	0.000316
ORGANIZATION_TYPE_Business Entity Type 1	0.000339
FLAG_CONT_MOBILE	0.000370
OCCUPATION_TYPE_Realty agents	0.000394
WEEKDAY_APPR_PROCESS_START_THURSDAY	0.000446

Figure 9: Least Correlated TOP10

3 Methodology

3.1 Feature Engineering

Feature engineering is a crucial step in the machine learning pipeline, as it can significantly boost the model's performance by creating new informative features or modifying existing ones. In our analysis, we undertook a comprehensive feature engineering process, which can be summarized as follows:

1. **Data Cleaning:** We replaced certain values that seemed to be placeholders or errors with NaNs. For instance, values like 'XNA' in the `CODE_GENDER`, `ORGANIZATION_TYPE` columns and 'Unknown' in the `NAME_FAMILY_STATUS` column were replaced with NaNs.
2. **Feature Creation:** Several new features were derived from the existing ones. Some of the notable features include:
    - `NEW_DOC_IND_KURT`: Kurtosis of documentation provided.
    - `NEW_EMPLOY_TO_BIRTH_RATIO`: Ratio of days employed to age of the client.
    - `NEW_SCORES_STD`: Standard deviation of external source scores.
    - `NEW_CAR_TO_BIRTH_RATIO`: Ratio of car age to client's age.
    - `AMT_PAY_YEAR`: Ratio of credit amount to annuity.
    - And several others related to income, region, and family status.
3. **Feature Aggregation:** We aggregated certain features to create median income values based on categories like gender, car ownership, realty ownership, and family status.
4. **Feature Transformation:** We created relative features that represent the deviation of a client's feature value from the median value of their category.
5. **Feature Removal:** Certain features that were deemed non-informative or highly correlated with others were removed to reduce dimensionality and multicollinearity. Examples include `AMT_GOODS_PRICE`, `FLAG_MOBIL`, and several documentation flags.

6. **Encoding:** Categorical features were label encoded to convert them into a format suitable for machine learning algorithms. Features with less than six unique values were treated as categorical, while others were considered for mean encoding.
7. **Data Type Conversion:** To optimize memory usage, data types of the features were downcasted where possible.

After these steps, our training dataset comprised of 121 features, ready to be fed into LGBM models.

4 Model Selection and Training

4.1 Bagging Classifier

A Bagging classifier, short for Bootstrap Aggregating classifier, is an ensemble machine learning technique designed to enhance the accuracy and robustness of a classification model. It achieves this by repeatedly creating multiple subsets (or "bags") of the training dataset through random sampling with replacement (bootstrapping) and training a base classifier on each subset. Subsequently, the predictions from various base classifiers are combined to make the final classification decision, typically by majority voting.

In our project, where the target variable is imbalanced (defaulted accounts accounting for only one-tenth of the entire training set), you are employing the Bagging method to address the class imbalance issue. Specifically, we are performing down-sampling of the majority class (target=0) by dividing it into three equal parts within each fold. Then, in each run, we train the entire minority class (target=1) alongside one of the three subsets of the majority class. After three runs, we average the results to obtain a more balanced and robust prediction.

This approach helps mitigate the impact of class imbalance, enabling the model to correctly classify defaulted accounts (target=1) while avoiding strong bias toward the majority class. Bagging classifiers are well-known for their ability to enhance the stability and accuracy of classification models, especially when dealing with imbalanced datasets.

4.2 Mean Encoding

Mean Encoding is a technique where categorical features are transformed based on the mean of the target variable for each category. This method can enhance the model's predictive power by incorporating target variable information into the features.

To implement mean encoding while preventing data leakage and overfitting, we used a 5-fold cross-validation approach. For each fold, the mean of the target variable for each category is calculated and mapped to the respective categories. Missing values are handled by using the global mean of the target variable.

4.3 Optimization Model Tuning

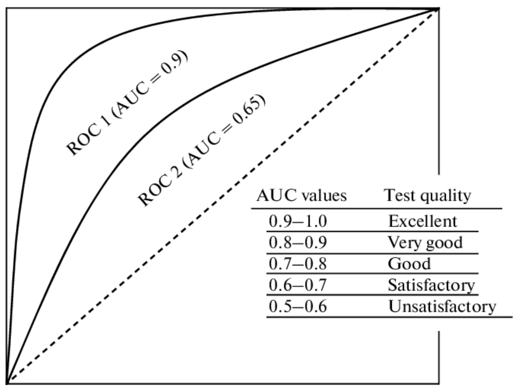
Bayesian Optimization Model Tuning is an automated method for finding the best hyperparameters of machine learning models, using probabilistic modeling to efficiently explore and exploit the hyperparameter space, ultimately improving model performance. We use the python package "optuna" do this part. The hyper parameter settings of our model:

5 Evaluation

5.1 Evaluation Metric

The evaluation metric commonly used in machine learning competitions is the **"Receiver Operating Characteristic Area Under the Curve"** (ROC AUC) metric.

ROC AUC is a metric that ranges between 0 and 1, with higher values indicating better model performance. Unlike binary predictions, ROC AUC assesses probabilities generated by models rather than simple 0s and 1s. This is because, in certain cases, accuracy is not the optimal metric for evaluating model performance, especially when dealing with imbalanced class situations.



AUC values	Test quality
0.9—1.0	Excellent
0.8—0.9	Very good
0.7—0.8	Good
0.6—0.7	Satisfactory
0.5—0.6	Unsatisfactory

Figure 10: The Example of ROC AUC

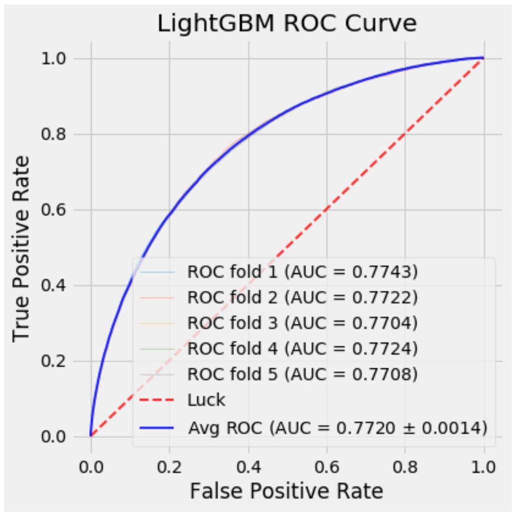


Figure 11: The Results of Model

5.2 Cross Validation

We utilized *Stratified K-Fold Cross Validation* with 5 folds to validate our model's performance. This method ensures that each fold maintains the same ratio of the target variable as the full dataset.

1. For each fold, the data is split into training and validation sets.
2. Mean encoding is applied by combining the validation and test sets temporarily.
3. A LightGBM model, optimized for the AUC metric, is trained on the training set.
4. The model's performance is evaluated on the validation set using the AUC score.

Using this cross-validation approach, we obtained a comprehensive assessment of our model's accuracy and generalizability.

5.3 Model Performance

In the process of model evaluation, both validation and test datasets were utilized to assess the performance. The validation dataset, which was derived during the training phase, provided insights into the model's capability to generalize on unseen data. The ROC curve, as depicted in Figure.9, showcases the model's performance across different folds. It is evident from the figure that the ROC curves from individual folds closely align with the average ROC curve. The Area Under the Curve (AUC) for each fold remains consistently above 0.77, with specific AUC values being 0.7743, 0.7722, 0.7704, 0.7724, and 0.7708 for the respective folds.

6 Results and Discussion

6.1 Feature Importance

From the figure above, we have found NEW_EXT_SOURCES_MEAN, EXT_SOURCE_3, EXT_SOURCE_2 and AMT_PAY_YEAR contribute most to the model. NEW_EXT_SOURCES_MEAN comes from original EXT_SOURCE data, which indicates external data sources may have a profound impact on the ability to repay on time.

And the feature AMT_PAY_Year's importance stems from a straightforward logic - the more money paid each month, the higher the likelihood of default.

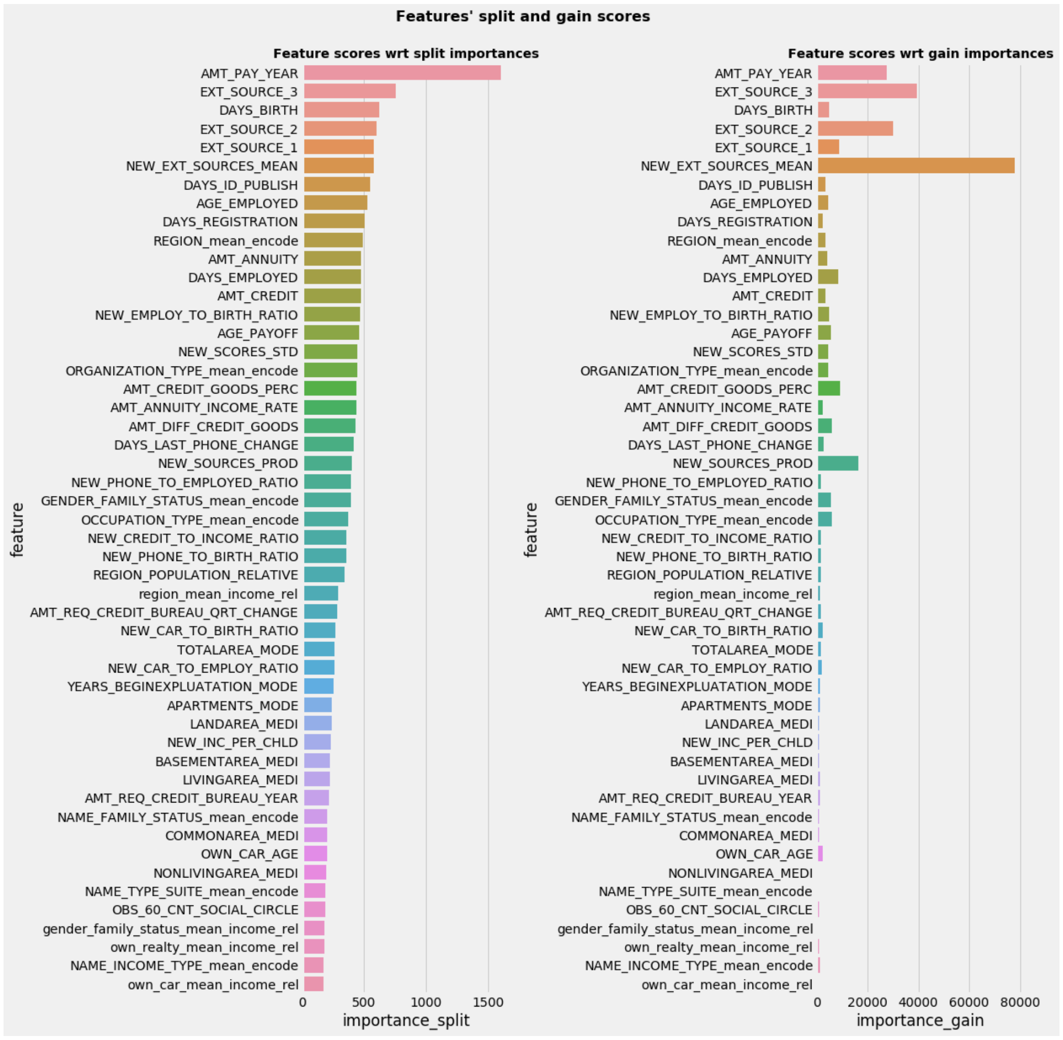
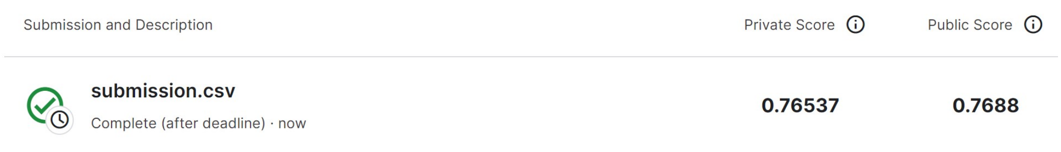


Figure 12: The Distribution of Different Features

6.2 Test Dataset Submission

Post validation, the model was subjected to the test dataset, and the predictions were submitted to the Kaggle platform for the Home Credit Default Risk competition. The model achieved a Private Score of 0.76537 and a Public Score of 0.7688. These scores are in close proximity to the AUC values obtained during validation, indicating a consistent performance of the model on both validation and test datasets. Such consistency is indicative of a well-generalized model that is likely to perform reliably on real-world data.

In conclusion, the model demonstrates robustness and reliability in predicting loan repayment capabilities, as evidenced by its performance metrics on both validation and test datasets.



Submission and Description	Private Score	Public Score
submission.csv Complete (after deadline) · now	0.76537	0.7688

Figure 13: Submissions for Evaluation

7 Conclusion

7.1 Key Findings

We conducted an in-depth analysis of the features in the application data, constructed new features for the model using various methods, and obtained the optimal parameters through Bayesian Optimization tuning. From the model's predictive results, it is evident that the amount borrowed and two undisclosed datasets play a crucial role in the model's accuracy.

7.2 Future Work

This time, we started from the application dataset, but from the results, it is clear that the newly constructed features in feature engineering are crucial for the model. Perhaps we can attempt to explore new features based on the remaining datasets to enhance the current model's performance.

Compared to teams on the leaderboard with scores above 0.80, there is room for further exploration. We can consider trying other Gradient Boosting models like XGBoost, constructing more features, in order to obtain a more robust and predictive model.

References

1. LightGBM GitHub Repository: `https://github.com/microsoft/LightGBM`

2. Bayesian Optimization for Hyperparameter Tuning: `https://en.wikipedia.org/wiki/Bayesian_optimization`