MAFS5440 Project1
Home Credit Default Risk

Congxin He 21024425: Feature Engineering, Cross Validation, Hyperparameter Tuning
Jiping Wang 21105827: Feature Engineering, Data Preprocessing
Yuwei Chen 21106455: Feature Engineering, Model Construction
Zilin Wu 21022271: Feature Engineering, Model integration and Testing

Abstract

This report investigates the use of advanced machine learning techniques to predict loan default risk for Home Credit, a financial institution dedicated to expanding access to credit for underserved populations. Leveraging a dataset of 307,511 loan applications, we performed comprehensive data exploration, feature engineering, and model selection. Key challenges such as class imbalance were addressed through methods like SMOTENC and RandomUnderSampler, while dimensionality reduction was achieved using Pearson correlation and embedded feature selection. We evaluated several models, including Logistic Regression, K-Nearest Neighbors (KNN), Random Forest, XGBoost, and LightGBM, with the latter demonstrating the highest performance in predicting default risk. Our findings contribute to improving Home Credit's ability to assess loan applicants and foster financial inclusion by offering more precise and equitable credit decisions.

1 Introduction

Home Credit is dedicated to expanding financial inclusion for the underbanked by providing a secure borrowing experience. Recognizing that individuals with insufficient or nonexistent credit histories often struggle to secure loans and are vulnerable to unscrupulous lenders, Home Credit uses diverse forms of alternative data, such as telecommunications and transactional records, to predict repayment capabilities.

This report outlines our approach to tackling the problem, detailing the steps taken in data exploration, feature engineering, model selection, and the evaluation of our predictive models. Our goal is to contribute to Home Credit's mission of providing positive loan experiences for those who are typically overlooked by traditional banking systems, thereby fostering a more inclusive financial landscape. Through our analysis, we aim to provide actionable insights that will enable Home Credit to make more accurate judgments about loan applicants' repayment abilities, ensuring that deserving individuals are granted the financial support they need to succeed.

2 Data Exploration

2.1 Dataset Overview

The main dataset 'application_train.csv' contains static information related to each loan application, where each row represents a unique loan. The dataset consists of 307,511 records, each characterized by 122 attributes.

'`SK_ID_CURR`' serves as the unique identifier for each loan application. It uniquely identifies each row in the dataset, representing an individual loan application. '`TARGET`' is the target variable that indicates the customer's payment behavior. Apart from '`SK_ID_CURR`' and '`TARGET`', the dataset

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

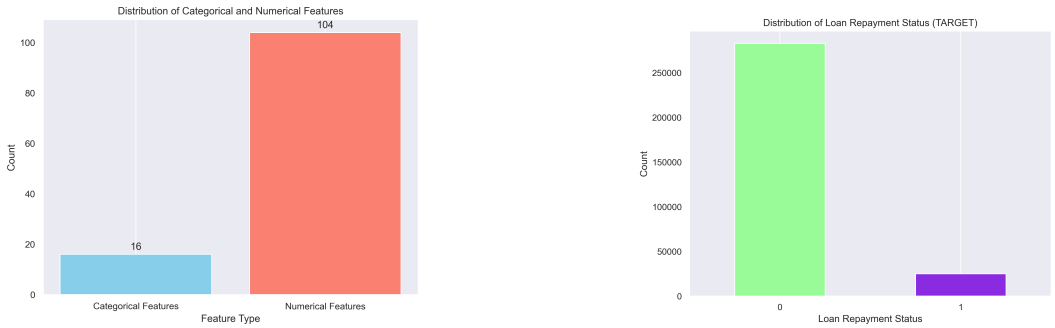
includes 120 additional features that provide various details about loan applications. These features can be categorized into two types 1a:

- **Categorical Features**: 16 categorical features represent qualitative information such as contract type, gender, car ownership, realty ownership, etc.
- **Numerical Features**: 104 numerical features represent quantitative information such as income amount, credit amount, annuity amount, number of children, and various requests made to the credit bureau.

By analyzing these features, we can gain insights into the factors that might influence a client's payment behavior and predict whether they are likely to experience payment delays.

2.2 Identifying Imbalance

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. Initial analysis of the distribution of loans 1b across these categories reveals a significant class imbalance. The majority of loans have been repaid on time, leading to a skewed representation of the two classes. This class imbalance must be addressed as we proceed with advanced methodologies. Techniques such as oversampling the minority class, and undersampling the majority class will be employed to mitigate this issue.



(a) Categorical and Numerical Features

(b) Distribution of 'TARGET'

Figure 1: Features' classification and distribution of Target Variable

2.3 Data Pre-processing

2.3.1 Outliers

In dealing with outliers in the dataset, we identified instances where the data contained infinite values (`inf` and `-inf`). These values could be due to errors during data collection or special circumstances. To handle these outliers, we adopted the following approach:

- **Detect Infinite Values**: We first detected whether the dataset contained any `inf` or `-inf` values.
- **Replace Infinite Values**: We replaced all `inf` values with the maximum value found in the respective feature, and all `-inf` values with the minimum value found in the respective feature.

This method aims to preserve the overall structure of the dataset while avoiding issues that might arise from training models with infinite values.

2.3.2 Missing Value

In this section, we systematically address missing and invalid values in the dataset.

Categorical Variables: The missing entries are replaced with the placeholder 'XNA', ensuring consistency across the dataset.

Numerical Variables: The missing values are imputed using the column's mean, preserving the overall distribution of the data.

Important Variables: Given the high importance of the variables EXT_SOURCE_1, EXT_SOURCE_2, EXT_SOURCE_3 to the model, we introduce new dummy variables exc_tf_1, exc_tf_2, exc_tf_3 that flag missing entries in these columns(1 for NA, 0 for else). This ensures that the model can differentiate between observed and imputed values for these key features, thus maintaining the integrity of their influence on the analysis.

2.3.3 Resampling

As mentioned in 2.2, our original data involves the problem of data imbalance, and direct training will lead to deviations in model predictions, so we implemented two resampling techniques: oversampling and undersampling.

- **Oversampling with SMOTENC**
  We used SMOTENC (Synthetic Minority Over-sampling Technique for Nominal and Continuous variables) for oversampling. SMOTENC is a kind of SMOTE 2 particularly suited for datasets that contain both categorical and continuous features. It generates synthetic instances for the minority class by interpolating between existing minority samples. This approach helps balance the dataset without simply replicating existing data, which can lead to overfitting.

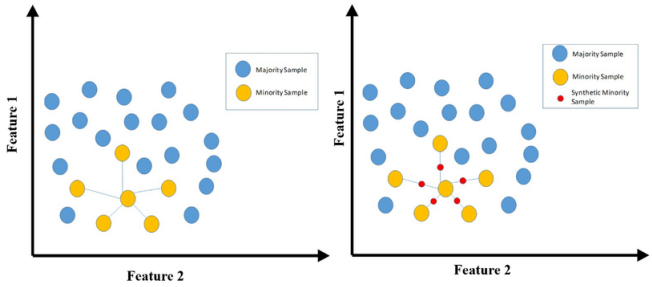


Figure 2: SMOTE

- **Undersampling with RandomUnderSampler**
  In addition to oversampling, we applied RandomUnderSampler to undersample the majority class. This method randomly removes instances from the majority class to balance the dataset. While undersampling can potentially lead to loss of information by discarding useful majority class data, it helps reduce the bias towards the majority class, forcing the model to pay more attention to the minority class.

2.3.4 Encoding

In our dataset, we dealt with a mix of categorical and numerical variables, which required different encoding techniques for the categorical features to make them suitable for machine learning models. We applied two types of encoding methods: **Label Encoding** and **One-Hot Encoding**, depending on the number of unique values in each categorical feature.

- **Label Encoding**
  For categorical features with only two unique values (binary features), we used Label Encoding. Label Encoding assigns numerical values (0 and 1) to the two possible categories, maintaining a simple and efficient representation of the data. Since binary features do not introduce ordinal relationships, Label Encoding is suitable and does not negatively affect model performance by introducing unnecessary dimensions.

- **One-Hot Encoding**
  For categorical features with more than two unique values, we applied One-Hot Encoding. One-Hot Encoding creates binary columns (dummy variables) for each unique value in the categorical feature. This ensures that no ordinal relationships are implied between categories, which could distort the model's understanding of the data.

By applying these encoding techniques, we transformed categorical variables into a format that can be efficiently used by machine learning models, while preserving the integrity of the original data structure.

3 Feature Engineering

3.1 Feature Creation

3.1.1 Hand-Crafted Features

We created various new features by leveraging the interactions between various features. The following table summarizes several selected hand-crafted features from `application_train.csv` and `previous_application.csv` and each feature is described 1.

For instance, the `credit_goods_ratio` feature calculates the ratio of the actual approved loan amount to the price of the goods. Specifically, it is obtained by dividing the actual approved loan amount (`AMT_CREDIT`) by the price of the goods (`AMT_GOODS_PRICE`). `AMT_CREDIT` and `AMT_GOODS_PRICE` alone can not reflect the repayment ability of the applicant, and they need to be compared to conclude. We get these characteristics mainly through the research of experts.

Feature Name	Description	Formula	From
income_credit_percentage	borrower's ability to repay the loan	$\frac{\text{AMT\_INCOME\_TOTAL}}{\text{AMT\_CREDIT}}$	application_train
car_to_employ_ratio	borrower's employment stability and reliability	$\frac{\text{OWN\_CAR\_AGE}}{\text{DAYS\_EMPLOYED}}$	application_train
application_credit_diff	borrower's credit risk status	$\text{AMT\_APPLICATION} - \text{AMT\_CREDIT}$	previous_application
credit_goods_ratio	borrower's repayment pressure	$\frac{\text{AMT\_CREDIT}}{\text{AMT\_GOODS\_PRICE}}$	previous_application

Table 1: Hand-Crafted Features Examples

3.1.2 Aggregations

Other supplementary tables (such as previous applications, bureau records, and installment payments) cannot be directly merged into the main table because clients have varying numbers of previous loans and different lengths of credit history. Therefore, numerous statistical features can be derived by grouping the data by `SK_ID_CURR` and then performing aggregations across different accounts and records from different months.

- **bureau_debt_credit_ratio** : This variable represents how much of the available credit is being used by the customer for each customer (`SK_ID_CURR`). It is calculated by

$$\texttt{bureau\_debt\_credit\_ratio} = \frac{\texttt{bureau\_total\_customer\_credit}}{\texttt{bureau\_total\_customer\_debt}}$$

  , where `bureau_total_customer_credit` is calculated by grouping the bureau data by the customer ID (`SK_ID_CURR`) and adding up `AMT_CREDIT_SUM` for each customer.

3.1.3 Time Series Features

Because there is a certain time series of data in some datasets, such as `MONTHS_BALANCE` in `posh_cash_balance.csv`, which represents the month of balance relative to the application date

(with -1 indicating the freshest balance date), this contains temporal information. Therefore, part of our feature engineering involves selecting only the most recent 1, 3, and 5 operations of the applicant for aggregation, respectively representing the relatively short-term, medium-term, and long-term characteristics of the applicant.

For example, the feature `SK_DPD_mean` is calculated by sorting the data by `SK_ID_CURR` and `MONTHS_BALANCE`, and then computing the mean of the most recent 1, 3, and 5 operations. This feature reflects the average days past due (DPD) over different time horizons, providing insights into the borrower's recent payment behavior and potential default risk.

3.2 Feature Selection

Correlation: Since we have added a lot of aggregation features, it is obvious that there are many features with strong correlation, such as sum and mean, which are intuitively highly correlated. To reduce the dimensionality of the data and improve model performance, we first used Pearson correlation coefficients to remove highly collinear features, which may lead to multicollinearity issues. By calculating the Pearson correlation matrix among the features, we identified and removed those with a correlation higher than a certain threshold (0.95). Figure 3a shows the correlation matrix of the top 10 features.

Embeded Feature Engineering: Subsequently, we employed embedded feature selection methods, specifically using XGBoost, to identify features with high importance. We trained an XGBoost model and selected the top features based on their importance scores. This method not only improved the model's performance but also reduced the risk of overfitting.

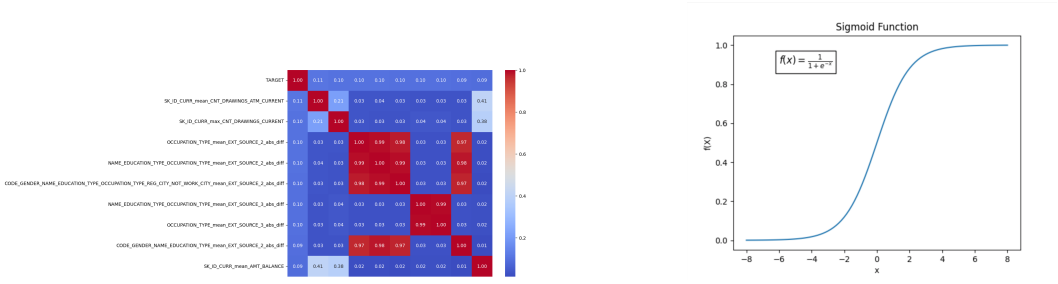
By combining Pearson correlation screening and embedded feature selection with XGBoost, we ultimately obtained a set of features that were neither highly collinear nor redundant, yet effectively captured the borrower's credit risk profile.

4 Model Construction

4.1 Overview of Models

Logistic Regression models the probability of a binary outcome by applying the logistic (or sigmoid) function 3b, which transforms any real-valued number into a probability between 0 and 1. The model is defined by the formula

$$P(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + ... + \beta_n x_n)}}.$$



(a) Correaltion Matrix of the top 10 Features (b) Sigmoid-Function

Figure 3: The correlation matrix and the sigmoid-function

KNN (K-Nearest Neighbors) is an instance-based classification method that predicts the label of a new sample by looking at the 'k' closest samples in the training set. It calculates the distance between the new sample and the training samples and assigns the most common label among the nearest neighbors. KNN does not have a separate training phase but relies on distance calculations for classification.

Linear Discriminant Analysis (LDA) is a classification method that aims to find a linear combination of features that best separates two or more classes. It works by modeling the distribution of each

class and maximizing the ratio of the variance between the classes to the variance within the classes. Quadratic Discriminant Analysis (QDA) is a supervised classification method that extends Linear Discriminant Analysis (LDA) by allowing each class to have its covariance matrix. In contrast to LDA, QDA allows for heteroscedasticity. The objective of LDA is to maximize the following criterion:

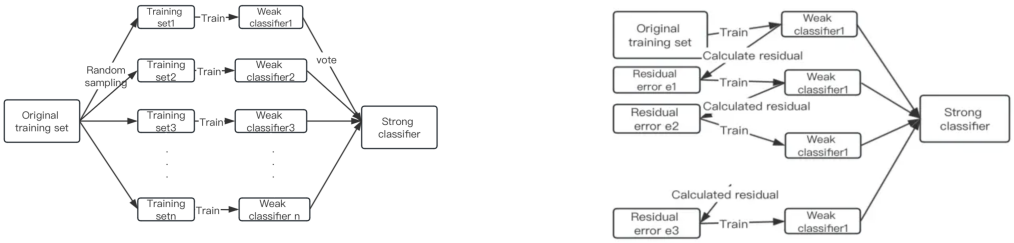
$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

Random Forest 4a is an ensemble learning method used for classification and regression tasks that combines multiple decision trees to improve predictive accuracy and control overfitting. It works by creating a "forest" of decision trees, where each tree is trained on a random subset of the data and features. During the training process, the algorithm selects a random sample of data points and a random subset of features for each tree, ensuring diversity among the trees.

Gradient Boosting Decision Trees (GBDT) build an ensemble of weak decision trees in a sequential manner, where each tree is trained on the residuals of the previous trees. One implementation of GBDT is XGBoost 4b. It builds an ensemble of weak decision trees in a sequential manner, where each tree is fitted to the residuals of the previous trees. XGBoost optimizes a loss function while incorporating regularization to prevent overfitting. The objective function in XGBoost is defined as:

$$\mathcal{L} = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{j=1}^{T} \Omega(f_j)$$

where: n is the number of training examples, $L(y_i, \hat{y}_i)$ is the loss function that measures the difference between the true label y_i and the predicted label $\hat{y}_i$, T is the number of trees, and $\Omega(f_j)$ is the regularization term for the j-th tree.



(a) Random Forest (b) XGBoost

Figure 4: Structure of Random Forest and XGBoost

Another advanced implementation of GBDT is LightGBM, which uses a leaf-wise growth strategy to select the leaf with the maximum delta loss to grow, leading to deeper trees and potentially better fit. The algorithm also incorporates a technique called Gradient-based one-sided sampling (GOSS), which retains instances with large gradients while randomly sampling those with smaller gradients, ensuring that the model focuses on harder-to-predict cases.

5 Result and Discussion

5.1 Model Comparison

We evaluated the models mentioned above using 5-fold cross-validation and calculated the performance metrics for each fold, then averaged the results. The results 2 show significant differences in performance metrics across the models. Notably, the tree-based models exhibit better performance, so we will continue to focus on tree models, including using XGBoost for embedded feature selection.

Table 2: Comparison of Model Performance Metrics

Model	Accuracy Mean	Precision Mean	Recall Mean	F1 Mean	AUC
Logistic Regression	0.6368	0.6571	0.5659	0.5903	0.64
KNN	0.8664	0.9315	0.7910	0.8555	0.92
Gaussian NB	0.6265	0.5865	0.8599	0.6971	0.72
LDA	0.7649	0.7560	0.7813	0.7681	0.84
QDA	0.8418	0.8981	0.7695	0.8261	0.91
Random Forest	0.9543	0.9978	.9108	0.9419	0.95
XGBoost	0.8274	0.8269	0.8431	0.8276	0.92
LightGBM	0.9551	0.9977	0.9122	0.9531	0.98

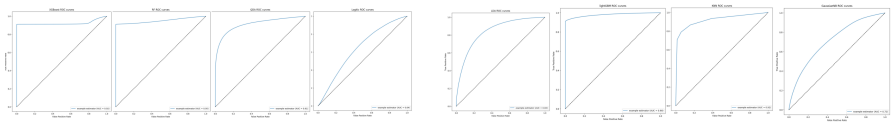


Figure 5: ROC curve

5.2 Test Results

From the below results, we find that LightGBM has the best effect and the running speed is very fast. Therefore, we choose LightGBM for the next step of model mining. And the below picture is the flow chart of the context.

We tried many different methods, including PCA and embedded feature engineering for factor dimensionality reduction. We found that PCA did not bring positive benefits to the model effect, so we finally selected the first 54 factors according to the importance of the output factors after fitting all factors with XGBoost. The cumulative importance reached 70%, earning the highest kaggle score.

After the factor selection, we used `optuna` to tune the parameters of the model. We performed cross-validation during the tuning process to ensure robustness. However, although the model's performance improved on the training data, it declined on the testing data. So we decided to retain the original parameters.

Finally, we took the predicted values of several models as new features and processed the predicted values by weighted average. We combined the prediction of a LightGBM model with 54 features and another LightGBM model with 61 features. This method brought about a 0.1 improvement to our models, indicating that different models capture different information emphases. Integrating different models is conducive to increasing the information utilization of training data, and the predicted values have a better effect 6.

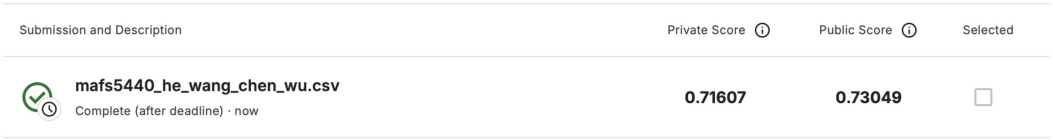


Figure 6: Kaggle Score

6 Conclusion and Future Work

6.1 Summary of Findings

In summary 7, in this report, we first mined new features based on raw data, and after dimensionality reduction, some important features were retained. The model built by using these features can well

predict the default probability of the applicant, and the robustness of the model is shown in both the training set and the test set.

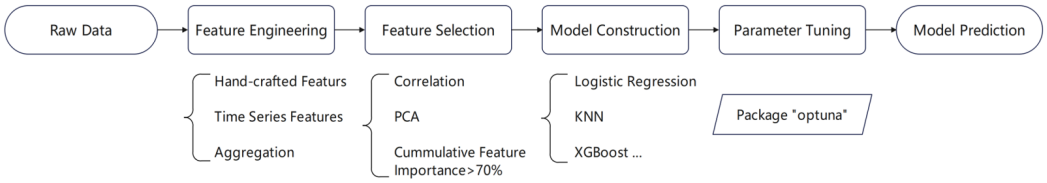


Figure 7: Flow Chart

For individuals with limited or no credit histories, leveraging these models can significantly enhance their access to credit. By providing accurate predictions, Home Credit can ensure that these individuals are not subjected to predatory lending practices, thereby improving societal security and individual welfare. Furthermore, the insights derived from the model can help borrowers understand the key factors influencing their creditworthiness, enabling them to take actionable steps to enhance their credit profiles and secure better loan terms.

6.2 Reflections

The effect of undersampling is not as good as that of oversampling.

Possible Reason: Since we randomly select one-third of the nondefault groups and all the default groups each time, and finally average the effect, we cannot guarantee that all the information can be used, so we may miss important information in the process of undersampling, resulting the effect of undersampling is not good.

PCA is not effective.

Possible Reason: Before applying PCA, we conducted factor screening based on correlation and removed variables with very high correlations, retaining only one from each set of highly correlated factors. This process solved certain collinearity problems. Therefore, we guessed that the reason for the poor effect of PCA is that the above process of selecting factors based on correlation can effectively remove collinearity in this problem, and if factors are selected based on PCA, information will be lost, resulting in poor model effect.

After parameter tuning, the effect of the training set increases, while the effect of the test set decreases.

Possible Reason: It is very likely that parameter tuning leads to overfitting of the model and lack of certain robustness, so the effect of the test set is poor.

6.3 Potential Areas for Further Research

Model Mining: We tried to use the Catboost model in this competition, but due to equipment problems, it was very slow and did not allow us to explore the model further.

Stacked Model: We tried many different models but did not explore stacked models, which combine predictions from multiple base models to enhance overall performance and accuracy. We can try in the future since stacking allows for better utilization of the strengths of various models, improving generalization and robustness in handling complex data patterns.

Feature Engineering: We tried to capture the long and short-term characteristics of the applicant, but did not aggregate the time characteristics. In our next step, we can use GRU, LSTM, and logistic regression models to calculate the trend of the time characteristics, which we guess is a very important factor.