MMF2030 MACHINE LEARNING FOR FINANCE

Home Credit Default Risk Project Group Component

Shuqi (Serena) Chang
Ruoxu (Ryan) Jiang
Xinyi (Cynthia) Shen
Surui (Alex) Zhang

Master of Mathematical Finance

University of Toronto

December 18, 2024

Table of Contents

1 Introduction **1**

2 Data Preprocessing **1**

3 Data Transformation **2**

4 Feature Engineering **3**

5 Splitting Training and Testing Datasets **7**
- 5.1 Identify Binary and Non-Binary Features 7
- 5.2 Scaling Non-Binary Features to Range of [0,1] 7
- 5.3 Splitting Training and Testing Datasets 8
- 5.4 Drop Columns with Missing Values 8

6 Feature Selection **8**
- 6.1 Stepwise Forward and Backward Selection 8
  - 6.1.1 Based on p-value 8
  - 6.1.2 Based on AIC/BIC 9
- 6.2 LASSO Regression 12
- 6.3 PCA Analysis 12
  - 6.3.1 Individual PCA Loading (Threshold: 0.1) 14
  - 6.3.2 Total PCA Loading (Threshold: 0.7) 14
- 6.4 Final Feature Selection with High Correlation Removal 15

7 Quality Assurance **20**
- 7.1 Data Validation 20
- 7.2 Preprocessing 20
- 7.3 Feature Engineering 21
- 7.4 Feature Selection 21

8 XGBoost Hyperparameter Tuning and Model Evaluation **21**
- 8.1 Objective 21
- 8.2 Hyperparameter Grid 22
- 8.3 Model Initialization and Grid Search Setup 22
- 8.4 Model Fitting and Selection 23
- 8.5 Model Evaluation 23
- 8.6 Model Evaluation Metrics 23

9 Random Forest Hyperparameter Tuning and Model Evaluation **23**
- 9.1 Objective 23
- 9.2 Hyperparameter Grid 24

9.3 Model Initialization and Grid Search Setup 24
9.4 Model Fitting and Selection 24
9.5 Model Evaluation 24
9.6 Model Evaluation Metrics 25

1 Introduction

Credit risk assessment is a critical component of modern financial institutions, as it directly impacts profitability and risk management. The Home Credit Default Risk project, based on a Kaggle competition, addresses the challenge of predicting an applicant's likelihood of defaulting on a loan. By leveraging diverse datasets that capture a wealth of applicant and transaction-related information, this project aims to enhance financial inclusion by identifying low-risk individuals who might otherwise be overlooked by traditional credit scoring models.

The dataset encompasses various data sources, including applicant demographics, previous loan applications, credit card balances, POS cash balances, and installment payment histories. This multi-dimensional data enables the development of robust machine learning models capable of capturing intricate patterns and relationships that drive credit risk.

2 Data Preprocessing

Features comes from the files

- `application_test.csv`
- `application_train.csv`
- `bureau.csv`
- `bureau_balance.csv`
- `credit_card_balance.csv`
- `installments_payments.csv`
- `POS_CASH_balance.csv`
- `previous_application.csv`

A data dictionary is enclosed in `HomeCredit_columns_description.csv` for a better understanding the meaning of data elements. Below is a summary of the type of data elements.

- **ID** (`SK_ID_CURR`): A unique identifier for each client, each labels a sample for the predictive model.
- **Target** (`TARGET`): This is the target variable for the model. The value `1` indicates clients with payment difficulties (i.e., clients who had late payment more than X days on at least one of the first Y installments of the loan in the dataset), while `0` represents all other cases. This simplifies the problem to a binary classification task.
- **Features:** There are 217 data columns in total provided in the raw data sets. We do not include all of them in the final model. We both:
  i. Make judgmental decisions on inclusion or exclusion based on our understanding of each feature's impact on a client's credit risk (payment difficulties), or
  ii. Perform feature selection using machine learning methods after initial feature engineering based on preprocessed dataset.

After uploading data, we first drop columns with missing rates greater than or equal to 40% since columns with a lot of missing values are unlikely to contribute meaningful information and can negatively impact model performance. We also remove constant columns where all rows have the same value, which have no predictive power. Then, we identify categorical columns with a `object` data type and exactly two unique values and convert them to binary encoding with 1 and 0, which would be more suitable for machine learning models. Additionally, we fill in missing values for numerical columns with `0` and for categorical columns with `'Unknown'`. Finally, we convert categorical column values to uppercase and removes leading or trailing spaces so that we standardize the format of categorical data to prevent inconsistencies.

3 Data Transformation

We transform the raw datasets into a cleaner and more machine-learning model-ready format.

First, we transform the datasets `application_train.csv` and `application_test.csv`.

- We convert the `CODE_GENDER` column into a numerical binary format with `'M'` mapped to `1` and `'F'` mapped to `0`.
- We create a new column `SUITE_ACCOMPANIED` based on the `NAME_TYPE_SUITE` column by setting `'UNACCOMPANIED'` and `'UNKNOWN'` to `0` and other values to `1`, thus creating a numerical binary variable to represent clients who were accompanied or unaccompanied when applying for the loan.
- We create a new numerical binary column `INCOME_SOURCE` based on the `NAME_INCOME_TYPE` column by encoding applicants who has income source to `1` such as `'WORKING'`, `'COMMERCIAL ASSOCIATE'`, `'PENSIONER'`, `'STATE SERVANT'`, `'BUSINESSMAN'` and who are not likely to have income source to `0` such as `'STUDENT'`, `'UNEMPLOYED'`.
- We create a new binary variable `HIGHER_EDU` based on the `NAME_EDUCATION_TYPE` column by setting applicants with a higher education degree to `1` such as `'HIGHER EDUCATION'`,`'ACADEMIC DEGREE'` and without higher education degree to `0`, including `'INCOMPLETE HIGHER'`, `'LOWER SECONDARY'`, `'SECONDARY / SECONDARY SPECIAL'`.
- We create a new binary variable `MARRIED` based on the `NAME_FAMILY_STATUS` column by setting applicants with a married family status to `1` such as `'CIVIL MARRIAGE'`, `'MARRIED'` and unmarried family status to `0` such as `'SEPARATED'`, `'SINGLE / NOT MARRIED'`, `'WIDOW'`.
- We drop columns that are irrelevant, redundant, or difficult to classify for predictive modeling, including `OCCUPATION_TYPE`, `WEEKDAY_APPR_PROCESS_START`, `ORGANIZATION_TYPE`, `HOUSETYPE_MODE`, `EMERGENCYSTATE_MODE`, `NAME_TYPE_SUITE`, `NAME_INCOME_TYPE`, `NAME_EDUCATION_TYPE`, `NAME_FAMILY_STATUS`, `NAME_HOUSING_TYPE`.

Secondly, we transform the dataset `previous_application.csv`.

- We create a new binary variable `CONTRACT_APPROVED` based on the `NAME_CONTRACT_STATUS` column by setting applicants whose previous contract status were `'APPROVED'` to `1` and other status such as `'CANCELED'`, `'REFUSED'`, `'UNUSED OFFER'` to `0`.
- We drop columns that are irrelevant, redundant, or difficult to classify for predictive modeling, including `NAME_CONTRACT_TYPE`, `WEEKDAY_APPR_PROCESS_START`, `NAME_CASH_LOAN_PURPOSE`, `NAME_PAYMENT_TYPE`,

CODE_REJECT_REASON, NAME_TYPE_SUITE, NAME_CLIENT_TYPE, NAME_GOODS_CATEGORY, NAME_PORTFOLIO, NAME_PRODUCT_TYPE, CHANNEL_TYPE, SELLERPLACE_AREA, NAME_SELLER_INDUSTRY, CNT_PAYMENT, NAME_YIELD_GROUP, PRODUCT_COMBINATION, NAME_CONTRACT_STATUS.

Third, we transform the dataset bureau.csv.

- We drop the column CREDIT_CURRENCY since the values are 'currency 1', 'currency 2', 'currency 3', and 'currency 4', which could not contribute to the predictive power.
- We drop the column CREDIT_TYPE. Since there are more than two important credit types, it is difficult to assign a new numerical binary variable to this column to provide useful information.

Finally, we transform the dataset POS_CASH_balance.csv by create a new binary variable CONTRACT_APPROVED based on the NAME_CONTRACT_STATUS column by setting applicants whose previous contract status were 'ACTIVE', 'APPROVED', 'COMPLETED', or 'SIGNED' to 1 and other status such as 'CANCELED', 'DEMAND', 'RETURNED TO THE STORE' to 0.

4 Feature Engineering

First, we construct features by feature engineering for bureau.csv and bureau_balance.csv.

- We map the categories for STATUS in bureau_balance.csv into numerical values:

STATUS	Description	Mapped Value
C	Closed	-1
X	Unknown	0
0	No Days Past Due (DPD)	1
1	1-30 DPD	2
2	31-60 DPD	3
3	61-90 DPD	4
4	91-120 DPD	5
5	120+ DPD	6

Table 1: Status Mapping in bureau_balance

- We aggregate STATUS in bureau_balance.csv grouped by SK_ID_BUREAU to calculate the last status recorded (LATEST_STATUS), the worst repayment status (WORST_STATUS), and the mean repayment status over time (AVERAGE_STATUS).
- We aggregate features in bureau.csv grouped by SK_ID_CURR and create the following features:
  1. Credit Counts:
     BUREAU_NUM_CREDIT: Total number of credit records (count of SK_ID_BUREAU).
  2. Credit Timing:
     BUREAU_DAYS_CREDIT_MIN/MAX: Minimum and maximum days since credit was granted.
     BUREAU_DAYS_CREDIT_ENDDATE_MEAN/MAX/MIN: Average, maximum, and minimum days until the credit end date.
     BUREAU_DAYS_ENDDATE_FACT_MIN/MEAN: Minimum and mean days until credit closure.

3. Overdue Metrics:
   BUREAU_CREDIT_DAY_OVERDUE_MAX/MEAN: Maximum and average number of overdue days.
   BUREAU_CNT_CREDIT_PROLONG_MAX/MIN: Maximum and minimum counts of credit prolongations.
4. Credit Amounts:
   BUREAU_AMT_CREDIT_SUM: Total credit amount.
   BUREAU_AMT_CREDIT_SUM_DEBT: Total debt amount.
   BUREAU_AMT_CREDIT_SUM_LIMIT: Total credit limit.
   BUREAU_AMT_CREDIT_SUM_OVERDUE: Total overdue credit amount.
5. Update Timing:
   BUREAU_DAYS_CREDIT_UPDATE_MEAN/MAX/MIN: Average, maximum, and minimum days since credit data update.
6. Repayment Status Features:
   BUREAU_LATEST_STATUS_MAX/SUM/MEAN: Max, sum, and mean of the latest repayment status.
   BUREAU_WORST_STATUS_MAX/SUM/MEAN: Max, sum, and mean of the worst repayment status.
   BUREAU_AVERAGE_STATUS_MAX/SUM/MEAN: Max, sum, and mean of the average repayment status.

- We calculate the credit active ratio (BUREAU_CREDIT_ACTIVE_RATIO) by computing the proportion of active credits using CREDIT_ACTIVE among all credits for the applicants.

Secondly, we perform feature engineering on dataset POS_CASH_balance.csv.

- We sort the data by the unique applicant ID (SK_ID_CURR), the unique ID for previous loans (SK_ID_PREV), and then the time relative to the current month when the status was recorded (MONTHS_BALANCE).
- We aggregate SK_DPD in POS_CASH_balance.csv grouped by SK_ID_CURR to calculate the maximum number of days past due across all recorded months for each applicant (SK_DPD_MAX), the average number of days past due across all recorded months for each applicant (SK_DPD_MEAN), and the total number of days past due across all recorded months for each applicant (SK_DPD_SUM).

Third, we perform feature engineering on credit_card_balance.csv.

- We sort the data by the unique applicant ID (SK_ID_CURR), the unique ID for previous loans (SK_ID_PREV), and then the time relative to the current month when the status was recorded (MONTHS_BALANCE).
- We aggregate features in credit_card_balance.csv grouped by SK_ID_CURR and create the following features:

  1. Balance Metrics:
     CREDIT_CARD_AMT_BALANCE_MAX: Maximum balance across all recorded months.
     CREDIT_CARD_AMT_BALANCE_MEAN: Average balance across all recorded months.
     CREDIT_CARD_AMT_BALANCE_SUM: Total balance across all recorded months.
  2. Credit Limit Metrics:
     CREDIT_CARD_AMT_CREDIT_LIMIT_ACTUAL_MAX: Maximum credit limit across all recorded months.

CREDIT_CARD_AMT_CREDIT_LIMIT_ACTUAL_MEAN: Average credit limit across all recorded months.

CREDIT_CARD_AMT_CREDIT_LIMIT_ACTUAL_SUM: Total credit limit across all recorded months.

3. Transaction Metrics:

   CREDIT_CARD_AMT_DRAWINGS_CURRENT_MAX: Maximum amount drawn using the credit card.

   CREDIT_CARD_AMT_DRAWINGS_CURRENT_MEAN: Average amount drawn using the credit card.

   CREDIT_CARD_AMT_DRAWINGS_CURRENT_SUM: Total amount drawn using the credit card.

   CREDIT_CARD_CNT_DRAWINGS_CURRENT_MAX: Maximum count of drawing transactions.

   CREDIT_CARD_CNT_DRAWINGS_CURRENT_MEAN: Average count of drawing transactions.

   CREDIT_CARD_CNT_DRAWINGS_CURRENT_SUM: Total count of drawing transactions.

4. Payment Metrics:

   CREDIT_CARD_AMT_PAYMENT_CURRENT_MAX: Maximum payment made during the months.

   CREDIT_CARD_AMT_PAYMENT_CURRENT_MEAN: Average payment made during the months.

   CREDIT_CARD_AMT_PAYMENT_CURRENT_SUM: Total payment made during the months.

   CREDIT_CARD_AMT_INST_MIN_REGULARITY_MAX: Minimum regular payment amount (maximum value).

   CREDIT_CARD_AMT_INST_MIN_REGULARITY_MEAN: Minimum regular payment amount (mean value).

   CREDIT_CARD_AMT_INST_MIN_REGULARITY_SUM: Minimum regular payment amount (total value).

5. Delinquency Metrics:

   CREDIT_CARD_SK_DPD_MAX: Maximum days past due (DPD).

   CREDIT_CARD_SK_DPD_MEAN: Average days past due (DPD).

   CREDIT_CARD_SK_DPD_SUM: Total days past due (DPD).

   CREDIT_CARD_SK_DPD_DEF_MAX: Maximum days overdue leading to default.

   CREDIT_CARD_SK_DPD_DEF_MEAN: Average days overdue leading to default.

   CREDIT_CARD_SK_DPD_DEF_SUM: Total days overdue leading to default.

Then we perform feature engineering on dataset previous_application.csv.

- We sort the data by the unique applicant ID (SK_ID_CURR) and then the unique ID for previous loans (SK_ID_PREV)
- We aggregate features in previous_application.csv grouped by SK_ID_CURR and create the following features:

1. Annuity Metrics:

   PREVIOUS_AMT_ANNUITY_MAX: Maximum annuity amount in previous applications.

   PREVIOUS_AMT_ANNUITY_MEAN: Average annuity amount in previous applications.

   PREVIOUS_AMT_ANNUITY_SUM: Total annuity amount in previous applications.

2. Application Amount Metrics:

   PREVIOUS_AMT_APPLICATION_MAX: Maximum application amount requested.

   PREVIOUS_AMT_APPLICATION_MEAN: Average application amount requested.

   PREVIOUS_AMT_APPLICATION_SUM: Total application amount requested.

3. Credit Amount Metrics:
   PREVIOUS_AMT_CREDIT_MAX: Maximum approved credit amount.
   PREVIOUS_AMT_CREDIT_MEAN: Average approved credit amount.
   PREVIOUS_AMT_CREDIT_SUM: Total approved credit amount.
4. Goods Price Metrics:
   PREVIOUS_AMT_GOODS_PRICE_MAX: Maximum price of goods requested for financing.
   PREVIOUS_AMT_GOODS_PRICE_MEAN: Average price of goods requested for financing.
   PREVIOUS_AMT_GOODS_PRICE_SUM: Total price of goods requested for financing.
5. Days Decision Metrics:
   PREVIOUS_DAYS_DECISION_MAX: Longest time since the decision on a previous application.
   PREVIOUS_DAYS_DECISION_MEAN: Average time since the decision on previous applications.
6. Number of Applications:
   PREVIOUS_NUM_APPLICATION: Total number of previous applications for a client.
7. Approval Ratio:
   PREVIOUS_APPROVED_RATIO: Ratio of approved applications to the total number of applications.

Finally, we perform feature engineering on installments_payments.csv.

- We sort the data by the unique applicant ID (SK_ID_CURR) for consistent grouping and feature computation.
- We create a new feature PAYMENT_DELAY, representing the difference between the actual payment date (DAYS_ENTRY_PAYMENT) and the scheduled installment date (DAYS_INSTALMENT). Thus, positive values indicate delayed payments, while negative values indicate payments in advance.
- To identify on-time payments, we create a new binary variable ONTIME_PAYMENT, where 1 indicates payments were made on time (PAYMENT_DELAY <= 0) while 0 indicates delayed payments.
- To identify full payments, we create a new binary column FULL_PAYMENT, where 1 indicates that the payment amount (AMT_PAYMENT) is greater than or equal to the installment amount (AMT_INSTALMENT) while 0 indicates partial payments.
- We create a new feature DELAYED_AMOUNT to represent the monetary value of delayed payments, where for delayed payments (PAYMENT_DELAY > 0) it is the product of PAYMENT_DELAY and the installment amount (AMT_INSTALMENT), otherwise it is set to 0.
- We create a feature ELIGIBLE_PAYMENT_AMOUNT to measure the monetary value of payments that are both on time and full, which is calculated as the product of AMT_PAYMENT, FULL_PAYMENT, and ONTIME_PAYMENT.
- We aggregate time series of the features in installments_payments.csv by grouping by SK_ID_CURR and create the following refined features:

  1. Delayed Payment Metrics:
     INSTALLMENTS_DELAYED_AMOUNT_MAX: Maximum delayed payment amount.
     INSTALLMENTS_DELAYED_AMOUNT_SUM: Total delayed payment amount.
  2. Installments Count:
     INSTALLMENTS_PAYMENTS: Total number of installments.

3. On-Time Payment Ratio:
   `INSTALLMENTS_ONTIME_RATIO`: Proportion of payments made on or before the due date: `ONTIME_PAYMENT.sum()/len(ONTIME_PAYMENT)`.
4. Full Payment Ratio:
   `INSTALLMENTS_FULL_PAY_RATIO`: Proportion of full payments (where `AMT_PAYMENT` is greater than or equal to `AMT_INSTALMENT`): `FULL_PAYMENT.sum()/len(FULL_PAYMENT)`.
5. Eligible Payment Amount:
   `INSTALLMENTS_ELIGIBLE_PAYMENT_AMOUNT_SUM`: Total value of payments that are both full and on-time.

5 Splitting Training and Testing Datasets

5.1 Identify Binary and Non-Binary Features

We separate binary and non-binary features in the dataset. The binary features include categorical data encoded as two distinct values `0` and `1`, which do not require further transformation, while the non-binary features contain all the other features that are continuous or have more than two categories. As a result, we can apply the scaling method only to the non-binary features while retaining the binary features. The identification is performed using the following logic:

$$\text{Binary Features} = \{\text{col} \mid \text{col has exactly two unique values}\}$$

Excluding ID and the target variable, there are 40 binary features and 138 non-binary features.

5.2 Scaling Non-Binary Features to Range of [0,1]

Based on observations from the feature engineering phase, the features in the dataset operate on vastly different scales. This can lead to issues in machine learning models which assume equal feature contributions. Without proper scaling, features with larger ranges can dominate and distort model predictions or slow convergence in optimization-based models. As a result, we apply the Min-Max Scaling method in the pipeline to transform each non-binary feature to a range of $[0, 1]$, making them comparable in magnitude. It avoid dominance by features with larger scales, ensuring balanced feature contributions. The formula is

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)},$$

where

- x is the original value.
- x' is the scaled value.
- $\min(x)$ and $\max(x)$ are the minimum and maximum values of the feature.

5.3 Splitting Training and Testing Datasets

After scaling, the dataset is split into training and testing sets using the `train_test_split` function from `sklearn.model_selection`. To maintain the class distribution of the target variable (`TARGET`), we employ *stratified splitting* by setting `stratify=y`, where the relative proportion of classes is preserved across both the training and testing sets. Specifically,

- **Training Set:** 80% of the full data is allocated to the training set.
- **Testing Set:** 20% of the full data is allocated to the testing set.

Stratified splitting is performed using: `train_test_split(X,y,test_size=0.2,stratify=y)`.

The target variable represents whether a client has payment difficulties. This ensures that the machine learning model accurately reflects the imbalance between clients with and without payment difficulties.

5.4 Drop Columns with Missing Values

Although we have dropped columns with more than 40% missing values, for the remaining columns with missing values, while their missing rates are lower than 40%, the absolute number of their missing values remains large, which could reach more than 10,000 missing values. This would negatively impact our feature selection process and model performance. Therefore, we consider dropping the 111 features that still contain missing values to ensure a cleaner dataset and minimize bias introduced by imputation or incomplete data.

6 Feature Selection

Although the number of features has been reduced to 67 by removing those with missing values, further feature selection remains essential. It enhances *model interpretability* by focusing on the most impactful features, which is critical in financial applications requiring transparency. Additionally, it helps *mitigate overfitting* by eliminating irrelevant or redundant features that introduce noise, improves computational efficiency by reducing complexity, and addresses *feature redundancy* caused by highly correlated variables that do not add predictive value.

6.1 Stepwise Forward and Backward Selection

6.1.1 Based on p-value

We first apply the iterative forward selection and backward elimination method using p-values from Ordinary Least Squares (OLS) regression. Forward selection starts with no features in the model and iteratively adds features with the smallest p-values below the inclusion threshold (`threshold_in=0.01`). Backward elimination removes features with p-values exceeding the exclusion threshold (`threshold_out=0.05`) in a stepwise manner. The process continues until no further features can be added or removed. Using this approach, we identified 35 key features with statistically significant contributions to the target variable, as shown in Table 2.

Feature	p-value
DAYS_BIRTH	0.0
EXT_SOURCE_3	0.0
EXT_SOURCE_2	0.0
HIGHER_EDU	1.425×10^{-104}
CODE_GENDER	4.71205×10^{-95}
FLAG_DOCUMENT_3	4.24941×10^{-74}
FLAG_OWN_CAR	1.53359×10^{-49}
DEF_30_CNT_SOCIAL_CIRCLE	9.64452×10^{-38}
REG_CITY_NOT_LIVE_CITY	5.69666×10^{-31}
AMT_REQ_CREDIT_BUREAU_YEAR	9.6184×10^{-27}
REGION_RATING_CLIENT_W_CITY	2.25524×10^{-18}
DAYS_ID_PUBLISH	1.17754×10^{-18}
DAYS_LAST_PHONE_CHANGE	9.20035×10^{-14}
FLAG_WORK_PHONE	2.45319×10^{-09}
DAYS_REGISTRATION	3.02482×10^{-08}
MARRIED	7.74489×10^{-09}
FLAG_DOCUMENT_18	3.59935×10^{-07}
FLAG_DOCUMENT_16	1.38333×10^{-06}
AMT_ANNUITY	5.1779×10^{-06}
AMT_GOODS_PRICE	3.45664×10^{-20}
AMT_CREDIT	1.93762×10^{-89}
REG_REGION_NOT_LIVE_REGION	0.000115793
REGION_RATING_CLIENT	0.000155483
SUITE_ACCOMPANIED	0.000204159
FLAG_DOCUMENT_6	0.000441616
FLAG_EMP_PHONE	0.000382243
DAYS_EMPLOYED	6.04157×10^{-20}
FLAG_DOCUMENT_5	0.00124104
FLAG_PHONE	0.00212506
AMT_INCOME_TOTAL	0.00218887
REGION_POPULATION_RELATIVE	0.00279706
FLAG_DOCUMENT_11	0.00492159
FLAG_DOCUMENT_13	0.00495315
DEF_60_CNT_SOCIAL_CIRCLE	0.0055991
FLAG_DOCUMENT_2	0.00824454

Table 2: Selected Features using Forward and Backward Selection based on p-values

6.1.2 Based on AIC/BIC

We applied stepwise forward and backward selection using both Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) to identify the optimal subset of features. AIC minimizes the trade-off between model fit and complexity, while BIC imposes a stronger penalty for model complexity, favoring simpler models.

The 50 selected features and their coefficients from the final model based on AIC are shown in Table 3.

Table 3: Selected Features using Forward and Backward Selection based on AIC

Feature	**Coefficient**	**Std. Error**	p**-value**
CONST	-0.3754	0.082	<0.0001
EXT_SOURCE_2	-0.1501	0.003	<0.0001
EXT_SOURCE_3	-0.0844	0.002	<0.0001
DAYS_BIRTH	0.0328	0.003	<0.0001
HIGHER_EDU	-0.0208	0.001	<0.0001
CODE_GENDER	0.0305	0.001	<0.0001
FLAG_DOCUMENT_3	0.0235	0.004	<0.0001
FLAG_OWN_CAR	-0.0169	0.001	<0.0001
DEF_30_CNT_SOCIAL_CIRCLE	0.3194	0.080	<0.0001
REG_CITY_NOT_LIVE_CITY	0.0228	0.002	<0.0001
AMT_REQ_CREDIT_BUREAU_YEAR	0.0820	0.007	<0.0001
REGION_RATING_CLIENT_W_CITY	0.0480	0.007	<0.0001
DAYS_ID_PUBLISH	0.0218	0.003	<0.0001
NAME_CONTRACT_TYPE	0.0001	0.004	0.9850
DAYS_LAST_PHONE_CHANGE	0.0226	0.003	<0.0001
FLAG_WORK_PHONE	0.0152	0.001	<0.0001
DAYS_REGISTRATION	0.0211	0.004	<0.0001
MARRIED	-0.0085	0.002	<0.0001
FLAG_DOCUMENT_18	-0.0313	0.006	<0.0001
FLAG_DOCUMENT_16	-0.0275	0.005	<0.0001
AMT_ANNUITY	0.1445	0.016	<0.0001
AMT_GOODS_PRICE	-0.8006	0.038	<0.0001
AMT_CREDIT	0.6681	0.034	<0.0001
REG_REGION_NOT_LIVE_REGION	-0.0170	0.005	<0.0001
REGION_RATING_CLIENT	-0.0234	0.007	0.0010
SUITE_ACCOMPANIED	-0.0052	0.001	<0.0001
FLAG_DOCUMENT_6	0.0194	0.004	<0.0001
FLAG_EMP_PHONE	0.5798	0.063	<0.0001
DAYS_EMPLOYED	0.5958	0.065	<0.0001
AMT_INCOME_TOTAL	0.7584	0.248	0.0020
FLAG_DOCUMENT_5	0.0181	0.005	0.0010
FLAG_PHONE	-0.0041	0.001	0.0010
REGION_POPULATION_RELATIVE	0.0104	0.003	0.0020
FLAG_DOCUMENT_11	-0.0224	0.009	0.0150
FLAG_DOCUMENT_13	-0.0299	0.009	0.0010
DEF_60_CNT_SOCIAL_CIRCLE	0.1934	0.070	0.0050
FLAG_DOCUMENT_2	0.2145	0.080	0.0070
FLAG_CONT_MOBILE	-0.0327	0.013	0.0100
FLAG_DOCUMENT_14	-0.0276	0.010	0.0060

Continued on next page

Table 3 (Continued)

Feature	Coefficient	Std. Error	p-value
FLAG_DOCUMENT_15	-0.0373	0.016	0.0160
AMT_REQ_CREDIT_BUREAU_DAY	0.0893	0.047	0.0560
INCOME_SOURCE	-0.0919	0.048	0.0540
FLAG_DOCUMENT_20	0.0447	0.024	0.0640
LIVE_CITY_NOT_WORK_CITY	0.0026	0.001	0.0750
FLAG_DOCUMENT_17	-0.0577	0.034	0.0900
FLAG_EMAIL	-0.0040	0.002	0.0920
CNT_FAM_MEMBERS	0.0269	0.016	0.0990
AMT_REQ_CREDIT_BUREAU_MON	0.0278	0.017	0.1030
FLAG_DOCUMENT_8	0.0064	0.004	0.1240

The 27 selected features and their coefficients from the final model based on BIC are shown in Table 4.

Feature	Coefficient	Std. Error	P-value
CONST	0.1144	0.005	<0.0001
EXT_SOURCE_2	-0.1506	0.003	<0.0001
EXT_SOURCE_3	-0.0852	0.002	<0.0001
DAYS_BIRTH	0.0399	0.003	<0.0001
HIGHER_EDU	-0.0213	0.001	<0.0001
CODE_GENDER	0.0309	0.001	<0.0001
FLAG_DOCUMENT_3	0.0136	0.001	<0.0001
FLAG_OWN_CAR	-0.0171	0.001	<0.0001
DEF_30_CNT_SOCIAL_CIRCLE	0.5158	0.041	<0.0001
REG_CITY_NOT_LIVE_CITY	0.0232	0.002	<0.0001
AMT_REQ_CREDIT_BUREAU_YEAR	0.0841	0.007	<0.0001
REGION_RATING_CLIENT_W_CITY	0.0468	0.007	<0.0001
DAYS_ID_PUBLISH	0.0219	0.003	<0.0001
NAME_CONTRACT_TYPE	-0.0081	0.002	<0.0001
DAYS_LAST_PHONE_CHANGE	0.0227	0.003	<0.0001
FLAG_WORK_PHONE	0.0135	0.001	<0.0001
DAYS_REGISTRATION	0.0238	0.004	<0.0001
MARRIED	-0.0068	0.001	<0.0001
FLAG_DOCUMENT_18	-0.0289	0.006	<0.0001
FLAG_DOCUMENT_16	-0.0248	0.005	<0.0001
AMT_ANNUITY	0.1537	0.016	<0.0001
AMT_GOODS_PRICE	-0.8255	0.038	<0.0001
AMT_CREDIT	0.6852	0.034	<0.0001
REG_REGION_NOT_LIVE_REGION	-0.0185	0.005	<0.0001
REGION_RATING_CLIENT	-0.0261	0.007	<0.0001
SUITE_ACCOMPANIED	-0.0051	0.001	<0.0001

Table 4: Selected Features Using Forward Selection Based on BIC

6.2 LASSO Regression

Then we apply the embedded method, Least Absolute Shrinkage and Selection Operator (LASSO) regression, to inherently perform feature selection by penalizing less important features during the optimization process. This ensures that the model selects the most relevant features while fitting the data. LASSO regression adds the L1 penalty to the loss function, which encourages sparsity in the model's coefficients. The optimization objective in LASSO regression is

$$\min_{\beta} \left(\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \alpha \sum_{j=1}^{p} |\beta_j| \right),$$

where

- β_j: Coefficients of the features.
- α: Regularization parameter controlling the trade-off between fitting the data and penalizing the model's complexity.
- $|\beta_j|$: Absolute value of feature coefficients, encouraging some coefficients to become exactly zero.

By shrinking some coefficients to zero, LASSO effectively removes irrelevant or less significant features from the model. Selected features are in Table 5.

Selected Features using LASSO Regression (α = 0.005)
CODE_GENDER
EXT_SOURCE_2
EXT_SOURCE_3
FLAG_DOCUMENT_3
HIGHER_EDU

Table 5: Selected Features using LASSO Regression (α = 0.005)

6.3 PCA Analysis

To complement the previous feature selection methods of stepwise selection and LASSO regression, which focus on statistical significance or regularization penalties, we apply Principal Component Analysis (PCA) to identify the most significant features based on their ability to explain the variability in the data.

Using PCA, we initially compute the explained variance ratio for all principal components. As illustrated in Figure 1, the explained variance ratio is plotted for each principal component. The cumulative explained variance is then calculated to determine how much of the total variability is captured by a subset of components.

Based on the cumulative explained variance in the output below, the first 10 principal components (highlighted in Figure 1) explain approximately 70.43% of the total variance. Thus, we use a threshold of 70%, which ensures that a significant portion of the data's variability is retained while avoiding overfitting caused by including components that contribute minimal variance (likely noise or redundancies).

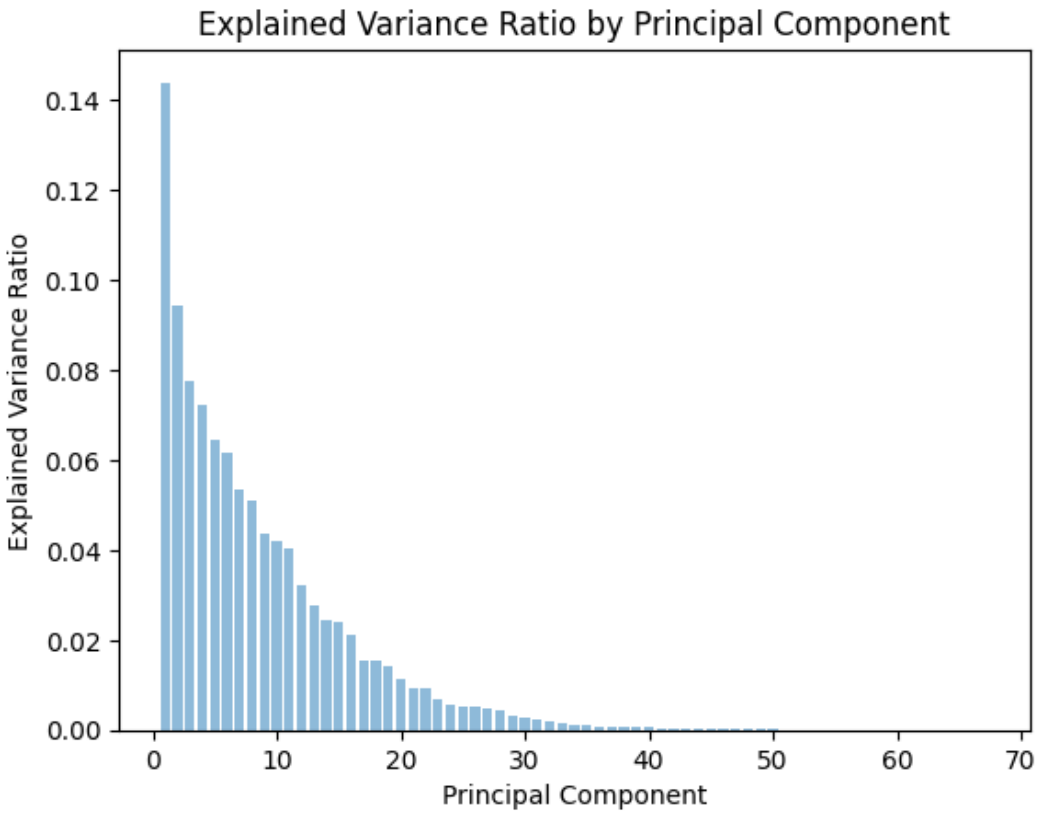

Figure 1: Explained Variance Ratio by Principal Component

```
Cumulative Explained Variance:
[0.14390363 0.2382144  0.31597894 0.3882609  0.45285535 0.51439547
 0.56783367 0.61874851 0.66220557 0.70429543 0.74473903 0.77692842
 0.80445301 0.82869502 0.85272788 0.87394175 0.88952964 0.90478461
 0.9189289  0.93015609 0.93945042 0.9485446  0.95538754 0.9608071
 0.9660411  0.97106897 0.97572583 0.9799671  0.98315252 0.98600927
 0.98834066 0.99011515 0.99162207 0.99273255 0.99370822 0.99461108
 0.99547265 0.99616276 0.99682004 0.99744606 0.99791888 0.99826964
 0.99858021 0.99886863 0.99904827 0.99921762 0.99936112 0.99947308
 0.99958468 0.99967831 0.99975082 0.99980564 0.99984247 0.99987558
 0.99990778 0.99993391 0.99995897 0.99997204 0.9999823  0.99999007
 0.99999483 0.99999622 0.99999759 0.99999878 0.99999998 1.
 1.         ]
```

The PCA loadings provide insight into the contribution of each original feature to the selected principal components. This cross-comparison of feature importance further informs the downstream model-building process.

6.3.1 Individual PCA Loading (Threshold: 0.1)

To identify features that contribute significantly to individual principal components, we set the threshold for significant contributions of features as 0.1. The selected features based on individual loadings are listed in Table 6.

Selected Features Based on Individual PCA Contributions
CODE_GENDER
DAYS_BIRTH
DAYS_EMPLOYED
EXT_SOURCE_2
FLAG_DOCUMENT_3
FLAG_DOCUMENT_6
FLAG_DOCUMENT_8
FLAG_EMP_PHONE
FLAG_OWN_CAR
FLAG_OWN_REALTY
FLAG_PHONE
FLAG_WORK_PHONE
HIGHER_EDU
LIVE_CITY_NOT_WORK_CITY
LIVE_REGION_NOT_WORK_REGION
MARRIED
NAME_CONTRACT_TYPE
REGION_POPULATION_RELATIVE
REGION_RATING_CLIENT
REGION_RATING_CLIENT_W_CITY
REG_CITY_NOT_LIVE_CITY
REG_CITY_NOT_WORK_CITY
REG_REGION_NOT_WORK_REGION
SUITE_ACCOMPANIED

Table 6: Features Selected Based on Individual PCA Contributions (Threshold: 0.1)

6.3.2 Total PCA Loading (Threshold: 0.7)

To identify features with significant overall contributions across all PCs, features with cumulative loadings greater than 0.7 are considered important and retained in the selected feature list in Table 7.

Selected Features Based on Total PCA Contributions
NAME_CONTRACT_TYPE
CODE_GENDER
FLAG_OWN_CAR
FLAG_OWN_REALTY
DAYS_EMPLOYED
FLAG_EMP_PHONE
FLAG_WORK_PHONE
FLAG_PHONE
REGION_RATING_CLIENT
REGION_RATING_CLIENT_W_CITY
REG_CITY_NOT_WORK_CITY
LIVE_CITY_NOT_WORK_CITY
FLAG_DOCUMENT_3
FLAG_DOCUMENT_6
SUITE_ACCOMPANIED
HIGHER_EDU
MARRIED

Table 7: Features Selected Based on Total PCA Contributions (Threshold: 0.7)

6.4 Final Feature Selection with High Correlation Removal

By leveraging stepwise forward and backward selection (based on p-values, AIC, and BIC), LASSO, and PCA (individual and total loading), the model benefits from improved interpretability, reduced dimensionality, and minimized redundancy. These methods ensure that only the most statistically significant and informative features are retained, enhancing model accuracy, robustness, and efficiency. Combining features selected using these methods, there are 52 unique features in Table 8.

The final step is to remove features with high correlations. We systematically identify and remove highly correlated features.

First, we compute the pairwise correlation matrix for the combined unique features to measure the linear relationships between all pairs of features, with values ranging from -1 to 1. To visualize the relationships between paired features, a heatmap is generated in Figure 2, highlighting regions of high correlation.

Then, we identify pairs of features with correlation coefficients exceeding the threshold, that is, the absolute correlation greater than 0.9. For each pair of highly correlated features, we mark only one feature for removal to avoid redundancy, and we skip those already viewed features to avoid duplicate drops.

This approach minimizes multicollinearity, which can negatively impact model stability and interpretability by introducing overlapping information. Removing highly correlated features results in a parsimonious and robust feature set, enhancing model performance and generalizability while preserving the most relevant predictors.

The correlation heatmap after removing high correlated features is shown in Figure 3, and the remaining features after filtering are shown in the final feature set in Table 9, free of strong linear dependencies. Table 10 is the data dictionary for the final feature selection.

Combined Feature Selection Before Removing High Correlated Features	
AMT_ANNUITY	FLAG_DOCUMENT_17
AMT_CREDIT	FLAG_DOCUMENT_5
AMT_GOODS_PRICE	FLAG_DOCUMENT_6
AMT_INCOME_TOTAL	FLAG_DOCUMENT_8
AMT_REQ_CREDIT_BUREAU_DAY	FLAG_EMAIL
AMT_REQ_CREDIT_BUREAU_MON	FLAG_EMP_PHONE
AMT_REQ_CREDIT_BUREAU_YEAR	FLAG_OWN_CAR
CNT_FAM_MEMBERS	FLAG_OWN_REALTY
CODE_GENDER	FLAG_PHONE
DAYS_BIRTH	FLAG_WORK_PHONE
DAYS_EMPLOYED	HIGHER_EDU
DAYS_ID_PUBLISH	INCOME_SOURCE
DAYS_LAST_PHONE_CHANGE	LIVE_CITY_NOT_WORK_CITY
DAYS_REGISTRATION	LIVE_REGION_NOT_WORK_REGION
DEF_30_CNT_SOCIAL_CIRCLE	MARRIED
DEF_60_CNT_SOCIAL_CIRCLE	NAME_CONTRACT_TYPE
EXT_SOURCE_2	REGION_POPULATION_RELATIVE
EXT_SOURCE_3	REGION_RATING_CLIENT
FLAG_CONT_MOBILE	REGION_RATING_CLIENT_W_CITY
FLAG_DOCUMENT_11	REG_CITY_NOT_LIVE_CITY
FLAG_DOCUMENT_13	REG_CITY_NOT_WORK_CITY
FLAG_DOCUMENT_14	REG_REGION_NOT_LIVE_REGION
FLAG_DOCUMENT_15	REG_REGION_NOT_WORK_REGION
FLAG_DOCUMENT_16	SUITE_ACCOMPANIED

Table 8: Combined Feature Selection Before Removing High Correlated Features

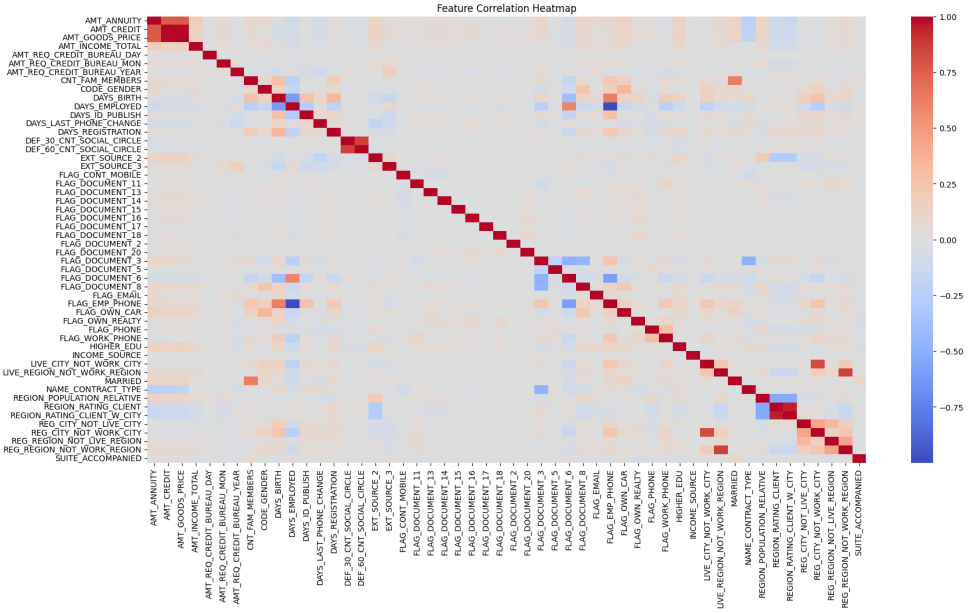

Figure 2: Correlation Heatmap before Removing High Correlated Features

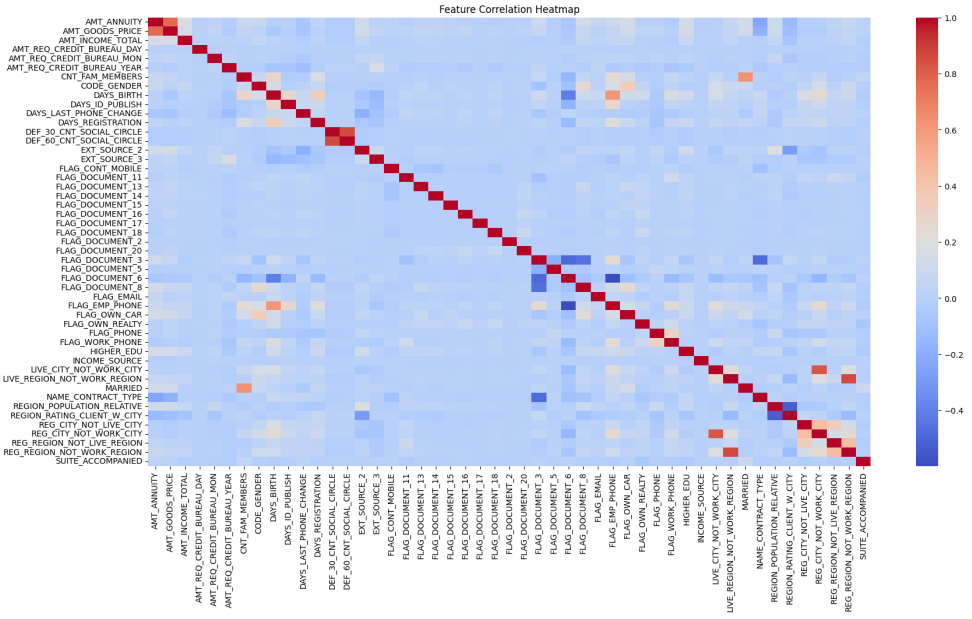

Figure 3: Correlation Heatmap after Removing High Correlated Features

Final Feature Selection After Removing High Correlated Features	
AMT_ANNUITY	FLAG_DOCUMENT_13
AMT_GOODS_PRICE	FLAG_DOCUMENT_14
AMT_INCOME_TOTAL	FLAG_DOCUMENT_15
AMT_REQ_CREDIT_BUREAU_DAY	FLAG_DOCUMENT_16
AMT_REQ_CREDIT_BUREAU_MON	FLAG_DOCUMENT_17
AMT_REQ_CREDIT_BUREAU_YEAR	FLAG_DOCUMENT_18
CNT_FAM_MEMBERS	FLAG_DOCUMENT_2
CODE_GENDER	FLAG_DOCUMENT_20
DAYS_BIRTH	FLAG_DOCUMENT_3
DAYS_ID_PUBLISH	FLAG_DOCUMENT_5
DAYS_LAST_PHONE_CHANGE	FLAG_DOCUMENT_6
DAYS_REGISTRATION	FLAG_DOCUMENT_8
DEF_30_CNT_SOCIAL_CIRCLE	FLAG_EMAIL
DEF_60_CNT_SOCIAL_CIRCLE	FLAG_EMP_PHONE
EXT_SOURCE_2	FLAG_OWN_CAR
EXT_SOURCE_3	FLAG_OWN_REALTY
FLAG_CONT_MOBILE	FLAG_PHONE
FLAG_DOCUMENT_11	FLAG_WORK_PHONE
HIGHER_EDU	INCOME_SOURCE
LIVE_CITY_NOT_WORK_CITY	REGION_RATING_CLIENT_W_CITY
LIVE_REGION_NOT_WORK_REGION	REG_CITY_NOT_LIVE_CITY
MARRIED	REG_CITY_NOT_WORK_CITY
NAME_CONTRACT_TYPE	REG_REGION_NOT_LIVE_REGION
REGION_POPULATION_RELATIVE	REG_REGION_NOT_WORK_REGION
SUITE_ACCOMPANIED	

Table 9: Final Feature Selection After Removing High Correlated Features

Descriptions of Final Selected Features	
Feature	**Description**
AMT_ANNUITY	Annualized loan annuity amount
AMT_GOODS_PRICE	Price of the goods for which the loan is given
AMT_INCOME_TOTAL	Total income of the client
AMT_REQ_CREDIT_BUREAU_DAY	Number of enquiries to Credit Bureau about the client one day before application
AMT_REQ_CREDIT_BUREAU_MON	Number of enquiries to Credit Bureau about the client one month before application
AMT_REQ_CREDIT_BUREAU_YEAR	Number of enquiries to Credit Bureau about the client one year before application
CNT_FAM_MEMBERS	Number of family members of the client
CODE_GENDER	Gender of the client (M/F)
DAYS_BIRTH	Client's age in days at the time of application (negative value)
DAYS_ID_PUBLISH	Days since the client's ID was changed (negative value)
DAYS_LAST_PHONE_CHANGE	Days since the client's last phone number change (negative value)
DAYS_REGISTRATION	Days since the client registered at the current address (negative value)
DEF_30_CNT_SOCIAL_CIRCLE	Number of the client's social surroundings defaulted on loans within 30 days
DEF_60_CNT_SOCIAL_CIRCLE	Number of the client's social surroundings defaulted on loans within 60 days
EXT_SOURCE_2	Normalized score from external data source 2
EXT_SOURCE_3	Normalized score from external data source 3
FLAG_CONT_MOBILE	Flag indicating if the client has a mobile phone (1 = yes, 0 = no)
FLAG_DOCUMENT_11	Flag indicating if the client provided document 11 (1 = yes, 0 = no)
FLAG_DOCUMENT_13	Flag indicating if the client provided document 13 (1 = yes, 0 = no)
FLAG_DOCUMENT_14	Flag indicating if the client provided document 14 (1 = yes, 0 = no)
FLAG_DOCUMENT_15	Flag indicating if the client provided document 15 (1 = yes, 0 = no)
FLAG_DOCUMENT_16	Flag indicating if the client provided document 16 (1 = yes, 0 = no)

FLAG_DOCUMENT_17	Flag indicating if the client provided document 17 (1 = yes, 0 = no)
FLAG_DOCUMENT_18	Flag indicating if the client provided document 18 (1 = yes, 0 = no)
FLAG_DOCUMENT_2	Flag indicating if the client provided document 2 (1 = yes, 0 = no)
FLAG_DOCUMENT_20	Flag indicating if the client provided document 20 (1 = yes, 0 = no)
FLAG_DOCUMENT_3	Flag indicating if the client provided document 3 (1 = yes, 0 = no)
FLAG_DOCUMENT_5	Flag indicating if the client provided document 5 (1 = yes, 0 = no)
FLAG_DOCUMENT_6	Flag indicating if the client provided document 6 (1 = yes, 0 = no)
FLAG_DOCUMENT_8	Flag indicating if the client provided document 8 (1 = yes, 0 = no)
FLAG_EMAIL	Flag indicating if the client provided an email address (1 = yes, 0 = no)
FLAG_EMP_PHONE	Flag indicating if the client provided a work phone number (1 = yes, 0 = no)
FLAG_OWN_CAR	Flag indicating if the client owns a car (Y = yes, N = no)
FLAG_OWN_REALTY	Flag indicating if the client owns real estate (Y = yes, N = no)
FLAG_PHONE	Flag indicating if the client provided a phone number (1 = yes, 0 = no)
FLAG_WORK_PHONE	Flag indicating if the client provided a work phone number (1 = yes, 0 = no)
HIGHER_EDU	Flag indicating if the client has higher education (1 = yes, 0 = no)
INCOME_SOURCE	Type of income of the client
LIVE_CITY_NOT_WORK_CITY	Flag indicating if the client's city of residence differs from the city of work (1 = yes, 0 = no)
LIVE_REGION_NOT_WORK_REGION	Flag indicating if the client's region of residence differs from the region of work (1 = yes, 0 = no)
MARRIED	Family status of the client (Married, Single, etc.)
NAME_CONTRACT_TYPE	Contract type of the loan (Cash loans, Revolving loans)
REGION_POPULATION_RELATIVE	Normalized population of the region where the client lives
REGION_RATING_CLIENT	Rating of the region where the client lives (1-3, 1 = best)

REGION_RATING_CLIENT_W_CITY	Rating of the region and city where the client lives (1-3, 1 = best)
REG_CITY_NOT_LIVE_CITY	Flag indicating if the client's registered city differs from their current city of residence
REG_CITY_NOT_WORK_CITY	Flag indicating if the client's registered city differs from their city of work
REG_REGION_NOT_LIVE_REGION	Flag indicating if the client's registered region differs from their region of residence
REG_REGION_NOT_WORK_REGION	Flag indicating if the client's registered region differs from their region of work
SUITE_ACCOMPANIED	Flag indicating if the client lives with family members (1 = yes, 0 = no)

Table 10: Descriptions of Final Selected Features

7 Quality Assurance

7.1 Data Validation

The data validation process has been conducted thoroughly. Missing values were identified, and columns with a missing rate of 40% or more were successfully dropped. For the remaining columns, appropriate imputation strategies were implemented to handle missing data (e.g., numerical columns were filled with `0`, while categorical columns were filled with `'Unknown'`). Consistency checks were performed to ensure logical relationships among features, such as verifying that the `DAYS_BIRTH` column corresponds to realistic client ages. The uniqueness of the identifier column `SK_ID_CURR` was confirmed, ensuring that each client is represented only once. Additionally, categorical values were standardized by converting them to uppercase and removing any leading or trailing spaces. These measures ensured a clean and reliable dataset for further analysis.

7.2 Preprocessing

Preprocessing steps have been rigorously validated and documented. Columns with constant values were detected and removed as they do not contribute to predictive modeling. Binary categorical columns were identified and converted into numerical format (e.g., `'M'` mapped to `1` and `'F'` mapped to `0`). A robust pipeline was implemented to ensure that preprocessing steps, including scaling of non-binary features to the range [0,1] using Min-Max Scaling, were reproducible. Detailed logs were maintained, recording each step, including columns that were dropped, encoded, or imputed. Intermediate datasets were saved for traceability and to facilitate reproducibility.

7.3 Feature Engineering

Feature engineering processes have been extensively quality-checked. All newly created features were validated against their original columns to ensure accuracy. For instance, binary features such as `MARRIED` and `HIGHER_EDU` were correctly derived from their respective categorical columns. Aggregated features, such as `BUREAU_NUM_CREDIT` and `BUREAU_DAYS_CREDIT_MIN`, were recalculated and cross-verified to confirm correctness. Redundant features introduced during the aggregation process were identified and removed through correlation analysis. The logic of engineered features was reviewed to ensure alignment with project objectives and domain knowledge, and documentation of each feature's derivation was maintained.

7.4 Feature Selection

The feature selection process was validated to ensure that only the most relevant and statistically significant features were retained. Forward selection, backward elimination, LASSO regression, and PCA were applied, and their results were cross-checked for consistency. Features were selected based on clearly defined thresholds, such as p-values, AIC, and BIC scores, as well as PCA loadings. The final feature set was confirmed to be free of multicollinearity through correlation analysis, where features with absolute correlation coefficients greater than 0.9 were systematically removed. Correlation heatmaps were generated before and after filtering to verify the elimination of highly correlated features. The final feature set was documented, including a comprehensive data dictionary for transparency and future reference.

8 XGBoost Hyperparameter Tuning and Model Evaluation

8.1 Objective

The primary goal of the process is to tune the hyperparameters of an XGBoost model using grid search and evaluate its performance on a test dataset. By identifying the best set of hyperparameters, we aim to optimize the model's predictive accuracy, particularly for binary classification tasks (e.g., predicting credit risk). The model evaluation is based on metrics such as ROC-AUC and F1 score, and the optimal classification threshold is determined to maximize the profit, which is calculated as:

$$\text{Profit} = (\text{benefit\_good} \times \text{True Negatives}) - (\text{cost\_bad} \times \text{False Negatives})$$

Note: To generate the profit curve for a binary classification model, particularly in the context of credit risk, we define the following key assumptions:

A **good customer** refers to a low-risk individual who is likely to repay the loan. When the model correctly identifies a good customer (i.e., classifies a low-risk customer as good), the company benefits as it can offer credit without the risk of default. We assume that the benefit for correctly classifying a good customer is given by:

$$\text{benefit\_good} = 1$$

This represents the unit profit gained from identifying a non-default customer.

A **bad customer** refers to a high-risk individual who is likely to default on the loan. When the model incorrectly classifies a bad customer as good (i.e., predicts a default customer as non-default), this leads to a cost for the company due to the potential loan default. We assume that the cost for incorrectly classifying a bad customer is given by:

$$\text{cost\_bad} = 5$$

This represents the financial loss the company incurs when a loan is defaulted on by a high-risk customer.

8.2 Hyperparameter Grid

A grid search is conducted over several key hyperparameters that define the behavior of the XGBoost model. The grid search will attempt different combinations of the following hyperparameters:

- **n_estimators**: The number of boosting rounds or trees to be trained. Higher values increase the model's capacity to learn from data but also increase computational complexity.
- **learning_rate**: A factor that determines the contribution of each tree to the overall prediction. Smaller values make the model more robust but require more trees (higher `n_estimators`) to converge.
- **max_depth**: The maximum depth of each tree. A deeper tree captures more complex relationships but may overfit the training data. A shallower tree may underfit.
- **colsample_bytree**: The fraction of features randomly selected for each tree. This helps to prevent overfitting by reducing correlation between trees.
- **subsample**: The fraction of training samples used to build each tree. Reducing this value can help prevent overfitting by introducing more randomness into the model.
- **scale_pos_weight**: A parameter that adjusts the balance between classes when there is an imbalanced dataset. A higher value increases the weight of the positive class, helping the model to focus more on it.

By experimenting with these parameters, the grid search identifies the best combination that maximizes model performance.

8.3 Model Initialization and Grid Search Setup

The XGBoost model is initialized with the following settings:

- **Random State**: Ensures reproducibility of results by controlling the random number generation.
- **use_label_encoder=false**: Disables the deprecated label encoder to avoid warnings.
- **eval_metric='logloss'**: Specifies the log loss metric to evaluate the model during training.

The `GridSearchCV` is used to perform an exhaustive search over the parameter grid. Cross-validation (3-fold by default) is used to assess model performance, providing a robust estimate of how well the model will generalize to unseen data.

8.4 Model Fitting and Selection

Once the grid search is completed, the best model (with the optimal set of hyperparameters) is selected. The model is refitted using the training data and optimal hyperparameters to ensure that it is trained with the most effective configuration. This model is then used to make predictions on the test set. We perform the model hyperparameter tuning based on both ROC-AUC score and F1 score.

8.5 Model Evaluation

The evaluation process involves two key steps:

- Prediction: The model's `predict_proba` method is used to generate predicted probabilities for the positive class (class 1). These probabilities are essential for generating the ROC curve and calculating the ROC-AUC score.
- ROC-AUC score is also calculated, which represents the area under the ROC curve. This score provides an aggregate measure of model performance across different classification thresholds.

8.6 Model Evaluation Metrics

The following metrics are produced as part of the evaluation:

- **ROC-AUC Score**: The Area Under the Receiver Operating Characteristic curve. This metric is crucial for assessing the performance of binary classifiers, particularly in imbalanced class scenarios.
- **Classification Report**: A detailed breakdown of the model's performance in terms of precision, recall, F1-score, and support for each class. The report helps in understanding the model's strengths and weaknesses in predicting each class.

9 Random Forest Hyperparameter Tuning and Model Evaluation

9.1 Objective

The purpose of this process is to tune the hyperparameters of a Random Forest model using grid search, evaluate its performance on a test dataset, and identify the optimal classification threshold. The Random Forest classifier is applied to a binary classification task, where the goal is to predict whether a given customer is default or non-default based on their historical data. Model performance is evaluated using metrics such as ROC-AUC, F1 score and a custom classification threshold that optimizes the profit similar as above.

9.2 Hyperparameter Grid

A grid search is performed over the following set of hyperparameters to find the optimal model configuration:

- **n_estimators**: The number of trees in the forest. Higher values typically lead to a more powerful model, but also increase computational cost.
- **max_depth**: The maximum depth of the trees. Deeper trees capture more complex patterns but may result in overfitting.
- **min_samples_split**: The minimum number of samples required to split an internal node. Higher values prevent the model from learning overly specific patterns, thus reducing overfitting.
- **min_samples_leaf**: The minimum number of samples required to be at a leaf node. This parameter controls overfitting by limiting the number of samples in terminal nodes.
- **max_features**: The number of features to consider when looking for the best split. Reducing this number can improve model generalization by introducing randomness into the decision-making process.

The grid search explores all combinations of these hyperparameters to find the best performing model.

9.3 Model Initialization and Grid Search Setup

The Random Forest model is initialized with the following parameters:

- **Random State**: Ensures that the random number generation is reproducible.

The GridSearchCV is used to perform an exhaustive search over the parameter grid. Cross-validation (3-fold by default) is used to assess model performance, providing a robust estimate of how well the model will generalize to unseen data.

9.4 Model Fitting and Selection

Once the grid search is completed, the best model (with the optimal set of hyperparameters) is selected. The model is refitted using the training data and optimal hyperparameters to ensure that it is trained with the most effective configuration. This model is then used to make predictions on the test set. We perform the model hyperparameter tuning based on both ROC-AUC score and F1 score.

9.5 Model Evaluation

After training, the model is evaluated using the following steps:

- Prediction: The model's `predict_proba` method is used to obtain the probabilities for the positive class. These probabilities are essential for evaluating the model with the ROC curve and calculating the ROC-AUC score.

The **ROC-AUC score** is computed, representing the area under the ROC curve. This score is a key metric for assessing binary classifiers, especially when dealing with imbalanced classes.

9.6 Model Evaluation Metrics

The following evaluation metrics are generated during the process:

- **ROC-AUC Score**: This score measures the model's ability to discriminate between positive and negative classes. It is especially useful when the class distribution is imbalanced.
- **Classification Report**: This report includes precision, recall, F1-score, and support for both classes. These metrics help assess the model's performance in terms of both false positives and false negatives.