Application of Machine Learning in Personal Credit Risk Assessment

By

Xinyi LI 1930018036

&

Chang YU 1930018090

Abstract

Personal credit is the foundation of the whole society credit. All transactions in the market are related to personal credit. Once the individual behaviour is out of control, there will be personal dishonesty, and then collective dishonesty. Therefore, the construction of personal credit system is extremely crucial. Prior to the widespread adoption of Internet technologies, traditional statistical models and expert analysis models often assessed credit risk by analyzing a small sample of loan customers and developing a set of standards that could be applied to all clients. In the background of big data, lending institutions can obtain the multi-dimensional information of customers. High-dimensional data brings many difficulties to the construction of credit risk assessment model, and the traditional credit risk assessment model is increasingly ineffective. Therefore, the advantages of the machine learning model in processing highly dimensional and complicated data will significantly increase the accuracy of credit risk assessment.

This paper first describes the relevant theories analysis of personal credit risk in commercial banks, credit evaluation models, evaluation metrics, time and model complexity, and feature selection methods. After that, the credit dataset of China UnionPay is taken as the research sample, aiming to accurately identify the personal fraud and overdue risks in the application of micro-credit business, help financial institutions accurately evaluate the personal credit situation, and further improve their ability to prevent fraud and reduce the failure rate. The first step is to properly preprocess the dataset, including Variance Threshold, standardization, using Isolation Forest to remove outliers, and using SMOTE algorithm to balance the dataset. Subsequently, combined with the k-fold cross-validation and parameter adjustments, five classifiers, namely Naïve Bayesian, Decision Tree, Random Forest, SVM, and Logistic Regression were selected to conduct the classification. Five evaluation metrics including accuracy, precision, recall, F1, AUC, and time complexity were obtained, so as to judge and evaluate the performances and differences of the five classifiers. In addition, the Lasso-RF two-stage feature selection method was used to process the dataset, aiming at further improving the performances of the classifiers and detecting some useful conclusions combined with the realistic meaning of the features.

The final experimental results show the following: After the data preprocessing, other classifiers except Naïve Bayesian have already shown satisfied classification effect. Among them, Random Forest has excellent performance in each metric, followed by Decision Tree, SVM, Logistic Regression and Naïve Bayesian. In detail, RF has the greatest performances on all metrics except time, which indicates it has the best classification effect but low efficiency. SVM has ordinary

performances on all metrics and the least efficiency. DT also has ordinary performances on all metrics but a better efficiency rather than SVM. LR has worse performances and medium efficiency. NB has the best efficiency and effectiveness but the worst reliability and classification effect. Moreover, the Lasso-RF two-stage feature selection method further improves the performance of Random Forest and greatly reduces the time and model complexity of all classifiers. By qualitatively analyzing the realistic meaning of important or deleted features, some useful conclusions are drawn, which can be used to help financial institutions identify what information is useful instead of analysing redundant information every time. This process is significant for financial institutions to assess personal credit risks and reduce the non-performing loan rate.

Keywords: credit risk, loan default, classification problem, Random Forest, Decision Tree, Support Vector Machine (SVM), Logistic Regression, Naïve Bayesian, Lasso regression, SMOTE Algorithm

Contents

1. Introduction

1.1 Background

With the accelerated pace of global economic integration and the rapid development of information technologies such as artificial intelligence, the economic and financial environment is becoming increasingly complex and volatile, and financial institutions are faced with more risk challenges than before. Among them, Commercial Bank, as an important part of the financial industry, is a key factor in the development of the national economy in China. It is a financial institution that undertakes credit intermediation through deposit, loan, exchange and savings operations. Its main scope of business is to take deposits from the public, grant loans and handle discounting of bills. Therefore, the smooth operation of the commercial banking system is a strong guarantee for the stable development of a country's economy. The Basel New Capital Accord classifies the risks faced by commercial banks into eight major categories, such as market risk, liquidity risk, credit risk and operational risk, of which credit risk is one of the most significant risks. A more authoritative international definition of credit risk is also given by the Basel Committee: credit risk is the risk that a debtor does not repay its debt within an agreed period of time and brings certain potential losses to economic agents. There are both macro and micro factors that affect commercial banks' credit risk. At the macro level, changes in economic policies can have an impact on the non-performing loan rates of commercial bank customers; at the micro level, basic characteristics such as the age and gender of the customers as well as their historical credit information will ultimately affect the clients' credit history.

1.2 Motivation

With the advent of the Big Data era, people's consumption patterns have gradually changed, giving rise to a variety of microloans. Compared to traditional collateral-based loans, the application threshold for microloans is low, so this requires credit institutions to monitor borrower information and control credit risks more strictly. In addition, an individual is more uncertain and mobile than a business, and is susceptible to subjective factors such as individual ideology, attitudes and behavioural habits. Therefore, credit institutions should attach great importance to personal credit risk and improve their credit assessment models and related vetting systems. The application of credit assessment models to map the diverse information of borrowers into their own detailed credit

levels has become one of the hot research issues in the field of financial risk management. This paper aims to achieve accurate identification of personal default risks in microfinance business applications, further enhancing the ability of financial institutions to prevent fraud and reduce the rate of non-performing loans.

Traditional credit risk assessment models generally use historical data as an input to predict default risk. However, traditional models are designed for business scenarios with low order volumes, long loan terms and a good quality customer base. The current business scenario for lending institutions is one of high order volumes, short loan terms and a poor customer base. In this type of business scenario, lending institutions need to consider multiple dimensions of information about the customer to assess credit risk, including personal information (e.g. age, gender, etc.), financial information (e.g. income, availability of house and car, etc.), repayment information (e.g. number of repayments, history of default, etc.), etc. However, using all these high-dimensional data to assess the risk of default is difficult for traditional credit risk assessment models and can easily lose a large amount of user information. Machine learning can help to improve these problems.

1.3 Theoretical Analysis of Machine Learning

With the continuous development of big data technology, various credit risk classification methods are becoming mature, especially the application of machine learning technology in this field has obvious advantages. Machine learning has good processing ability for solving linear and nonlinear classification problems and it can deal with a large amount of data, which can ensure the accuracy and effectiveness of personal credit default prediction to a large extent. Moreover, its development makes the prediction result of credit assessment model become more and more accurate. Previous studies have shown that credit risk assessment models based on machine learning can significantly improve the accuracy and adaptability of credit risk assessment results.

Machine learning theory makes the model construction break the constraints of operation and strong assumptions. By using the central limit theorem (CLT), the approximately distributed samples in a large sample size can be superimposed, and then the distribution of the overall sample can be inferred. The application of machine learning in credit risk assessment guarantees the universality of model prediction results to reality. Credit research using machine learning can often learn the "black box" problems in forecasting, so that the prediction results of machine learning models have more explanatory power. Machine Learning method mainly builds the model of the customer's

credit data through the supervised learning algorithm. It builds a model through a series of operations, such as data preprocessing, feature selection, classification method selection and so on, in order to realize the prediction of customer behaviours and characteristics, and judge whether the customer will default in the next transaction.

Feature Selection and the selection of classification algorithms for a given dataset are two extremely important components in the construction of credit risk assessment models. In order to improve the accuracy of assessment models and reduce the influence of non-relevant features, many scholars have used various feature selection methods to extract features that contribute highly to the model and thus improve the accuracy of assessment. Feature selection methods include Filter, Wrapper and Embedded. Filtering refers to optimising the feature set by measuring the importance of individual features through the characteristics of the sample data itself, and filtering out relatively useless features based on the scores obtained from statistical tests of the features. It can be divided into Variance Threshold and Correlation Threshold. Wrapping treats feature selection as part of the overall learning algorithm, using a supervised induction algorithm to find the optimal subset of features, which are then evaluated using a classifier algorithm. Random Forest, Support Vector Machine (SVM), K-Nearest Neighbor (KNN) and so on are the significant methods in Wrapping. Embedded is an extension of the wraparound approach and it is a feature selection method based on penalty term. It selects features through L1 regularization terms: L1 regularization method has the characteristic of sparse solution, so it naturally has the characteristic of feature selection. Lasso regression is a classical embedded method. Feature selection is often used as a significant stage of data preprocessing and can play an essential role in improving the performance metrics of a classification algorithm.

Classification algorithms discover category rules and predict the category of new data by computing and analysing a training set of known categories. It is a method for solving classification problems and is an influential area of research in data mining, machine learning and model recognition. The main use and scenario for classification is prediction, where the category of a new sample is predicted based on an existing sample, such as credit rating, risk rating, default prediction, etc. Commonly used classification algorithms include Logistic Regression, Decision Trees, Random Forests, SVM, etc.

1.4 Introduction to Dataset

This paper selected the credit dataset of China UnionPay which comes from the "Credit Customer Overdue Prediction" algorithm competition established by China UnionPay in 2018. China UnionPay was set up in March 2002 with the approval of the State Council and the People's Bank of China. Based on the merger of 18 bank card information exchange centers, it is the China Bank Card Association, which is jointly funded by 85 institutions and headquartered in Shanghai. As the banks of China joint organization, China UnionPay is on the crucial and pivotal status of bank card industry in China and plays an essential role in the development of bank card industry of our country. Through China UnionPay's inter-bank transaction clearing system, banks have realized inter-system connectivity, thus enabling bank cards to be used across banks, regions and borders.

The dataset has 11017 samples and 200 variables, among which the binary classification variable is Y, indicating whether the customer is in default (the default customer is represented by 1, and the non-default customer is represented by 0). The remaining 199 variables are explanatory variables, including six aspects which are identity information and property status, card information, transaction information, lending information, repayment information and application for loan information.

In detail, identity information and property status consider the personal age, gender, real estate, insurance, etc. Card information includes the number of cards held and the types of card can be divided to debit card, credit card and commercial bank card. Moreover, there is also a relationship among each variable of remaining four aspects. Transaction information can be divided into 21 categories and each category involves total transactions, transaction amount, number of trading months and etc. Lending information can be classified into three different periods of days, 30 days, 90 days and 180 days. In every period, there are the number of lending institutions, total number of lending and so on. The same applies to the classification of repayment information and loan information.

2. Literature Review

In recent years, many scholars have studied the issue of credit risk assessment. The research on customer credit risk assessment models of commercial banks can be roughly divided into three types. The first is the empirical model. The traditional credit risk assessment mainly relies on the

subjective experience judgment of commercial bank employees, which means through collecting personal information such as age, occupation and historical default records offline to artificially control the risk. However, this method often has the shortcoming of low efficiency and accuracy, and is mostly used in the case of insufficient data. However, Zeng Ming and Xie Jia (2019) believe that the scale of personal credit data increases dramatically and the credit data of the Internet is non-standardized, leading to different characteristics of personal credit risks and more obvious information asymmetry. So gradually transitioned to the second type of model, scholars began to apply the econometric analysis method to solve the problem of personal credit risk.

The linear regression model is one of the earliest models applied to personal credit risk assessment. Orgler (1970) selected explanatory variables from the dimension of the borrower's personal information and constructed a linear regression model to evaluate the borrower's credit risk. Wiginton (1980) was the first to introduce logistic model on the basis of linear regression for empirical analysis. Li Meng (2005) combined non-performing loan ratio, t-test and principal component analysis to establish an evaluation model for judging credit risk based on Logistic Regression. Fang Kuangnan et al. (2014) combined Lasso and logistic model to construct a personal credit risk assessment model, which significantly improved the estimation accuracy of the basic model. This econometric analysis method has the advantage of being objective, but it has great requirements for the quantity and quality of data. However, the management and utilization of complex high-dimensional data need to depend on machine learning, so the third type of model is artificial intelligence model, such as Artificial Neural Network, SVM, Random Forest, etc.

In 2004, Shen Cuihua et al. used SVM to evaluate personal credit risk and found that the classification effect of SVM was better than discriminant analysis model. Bellotti et al. (2009) applied SVM to credit scoring and concluded that variables such as the presence of property, number of loans in the past six months, and age all have an impact on credit risk assessment. Yao Xiao and Yu Lean (2012) introduced fuzzy membership into SVM and significantly improved the classification accuracy of credit risk. Eletter (2014) used artificial neural network to provide technical support for the loan decision-making of Jordanian commercial banks, and confirmed that this method could effectively improve the efficiency of credit decision-making and help financial institutions reduce the evaluation cost. Wu Chong (2009) established the fuzzy neural network to assess the credit risk of commercial banks. Empirical research results show that the prediction error of fuzzy neural network is small, and there is no the characteristic of completely black box like neural network. Decision tree has been widely used in credit scoring since 1980 as a credit

evaluation model. Frydman (1985) found that Decision Tree obtained better results than discriminant analysis model in credit risk assessment. Simha and Satchidananda (2006) used the agricultural loan data of two Indian banks to predict default risk by Logistic Regression model and Decision Tree model respectively, and found that the result of Decision Tree algorithm was better than that of Logistic Regression. However, the biggest defect of Decision Tree is instability. Small fluctuations in datasets may lead to big changes in classification results. Breiman (2001) introduced Random Forest model based on Decision Tree and pointed out that Random Forest was significantly better than single Decision Tree. Chinese scholar Fang Kuangnan (2010) applied Random Forest to credit risk assessment of credit card for the first time. Comparing this method with Logistic Regression and SVM, he found that Random Forest was the best. Cano (2017) adopted Random Forest algorithm for different datasets for feature selection, and used the important factors which were selected by Random Forest to predict risk. The results show that Random Forest has more advantages rather than artificial neural network and SVM. Zhou Yongsheng et al. (2020) used the improved Random Forest algorithm of Germany's credit dataset and verified the feasibility and superiority of the related Random Forest models.

To sum up, the theoretical system of credit risk assessment at home and overseas has been relatively mature after decades of development. From the early immature statistical model and qualitative analysis of expert, to Logistic Regression and other statistical analysis models which are relatively mature, and then to SVM and Random Forest which are more advanced data mining algorithms, the personal credit risk assessment has a lot of excellent study method through continuous comparison and optimization each model. And machine learning models have a good performance in the field of personal credit risk assessment. Additionally, in feature selection part, most of the researches for high-dimensional datasets at home and abroad focus on the optimization and application of single feature selection method. The method of selection can effectively screen out some non-critical noise features and reduce the search range of the optimal feature subset. However, the model and time complexities of this method are high. Therefore, we used the two-stage feature selection method to reduce the complexities.

In this paper, we first filled in the missing values with 0 and deleted some samples after considering the relationship among features and practical significance. Then we used Variance Threshold for the initial screening of high-dimensional dataset, and then standardized the non-categorical features by z-score. After that, Isolation Forest was applied to remove outliers. Moreover, Synthetic Minority Oversampling Technique (SMOTE) Algorithm, the combination of Undersampling and

Oversampling, was used to deal with imbalanced data. Finally, we conducted special feature selection methods like Lasso regression and two-stage selection method which combined Lasso regression and Random Forest to further screen features. The filtered data was input into the classifier including Logistic Regression, SVM, Decision Tree, Random Forest and Naïve Bayesian for analysis after SMOTE Algorithm and each step of feature selection. Since different machine learning models have different performances in personal credit risk assessment under various data patterns and business requirements, our aim is to compare which classifier has the best performance, and check whether the evaluation metrics are optimized by adjusting the parameters.

3. Methodology

3.1 The Introduction of Machine Learning Classifiers

In machine learning, supervised learning can be divided into two types of models: discriminative models and generative models. In general, discriminative models learn the conditional distribution, modelling the decision boundary between the classes, while generative models learn the joint distribution, modelling the actual distribution of each class. Although generative models can gain more information and reflect the characteristics of the data itself through learning, it is less productive and effective with small samples.

Since there are merely 11017 samples in our dataset, to obtain the better performance when predicting, we focused on the discriminative models, such as Logistic Regression, SVM, Decision Tree and Random Forest. To highlight the superiority of discriminative model in this dataset, we also used Naïve Bayesian to make the classification, one of the most common generative models.

3.1.1 Principle of Classifier

- Naïve Bayesian

Set the input space $X \subseteq R^{\mathrm{n}}$ be the set of some dimensional vectors and the eigenvector $x \in X$ serves as the input. Let the output space be the set of class label $Y = \{c_1, c_2, \dots, c_k\}$, and the class label eigenvector $y \in Y$ acts as the output. The random vectors X and Y are defined on the input and output spaces, respectively. The joint probability distribution of X and Y is $P(X, Y)$. The training dataset is

$$T = \{(x_1, y_1), (x_{2,} y_2), \dots, (x_N, y_{N)})\}$$

generated by the $P(X, Y)$ independent identical distribution.

The Naïve Bayesian method learns the joint probability distribution via the training set. Specifically, learn the distribution of prior probability and conditional probability. The prior probability distribution is

$$P(Y = c_k), \qquad k = 1,2, \dots, K$$

The conditional probability distribution is

$$P(X = x|Y = c_k) = P\left(X^{(1)} = x^{(1)}, \dots, X^{(n)} = x^{(n)}\middle|Y = c_k\right), \quad k = 1,2, \dots K$$

Therefore, learn the joint probability distribution $P(X, Y)$.

Naïve Bayesian method assumes conditional independence of conditional probability distribution. In particular, the conditional independence hypothesis is

$$\begin{aligned} P(X = x|Y = c_k) &= P\left(X^{(1)} = x^{(1)}, \dots, X^{(n)} = x^{(n)}\middle|Y = c_k\right) \\ &= \prod_{j=1}^{n} P\left(X^{(j)} = x^{(j)}\middle|Y = c_k\right) \end{aligned} \tag{3.1}$$

The learnt model calculates the posterior probability distribution $P(Y = c_K|X = x)$for the Naive Bayesian classification given the input x. The class output is determined by the class with the highest posterior probability. The Bayesian Theorem is followed in the computation of posterior probability:

$$P(Y = c_k|X = x) = \frac{P(X=x|Y=c_k)P(Y=c_k)}{\sum_k P(X=x|Y=c_k)P(Y=k)} \tag{3.2}$$

Put (3.1) into (3.2)

$$P(Y = c_k|X = x) = \frac{P(Y=k)\prod_j P\left(X^{(j)} = x^{(j)}\middle|Y = c_k\right)}{\sum_k P(Y=c_k)\prod_j P\left(X^{(j)} = x^{(j)}\middle|Y = c_k\right)}, \qquad k = 1,2, \dots K \tag{3.3}$$

This is the basic formula for Naïve Bayesian classification. Thus, Naïve Bayesian classifier is

$$y = f(x) = \arg\max_{c_k} \frac{P(Y = c_k)\prod_j P\left(X^{(j)} = x^{(j)}\middle|Y = c_k\right)}{\sum_k P(Y = c_K)\prod_j P\left(X^{(j)} = x^{(j)}\middle|Y = c_k\right)}$$

Since all the c_K are the same in the denominator of (4.3),

$$y = \arg\max_{c_k} P(Y = c_k) \prod_j P\left(X^{(j)} = x^{(j)} \middle| Y = c_k\right)$$

- Logistic Regression

The conditional probability distribution $P(Y|X)$ in the form of a parametric logistic distribution serves as the representation for the binomial logistic regression model, which is a classification model. The random variable X is a real value, while the random variable Y is either 1 or 0. Using a supervised learning strategy, we estimate the model parameters.

The binomial Logistic Regression model is the conditional probability distribution as follows:

$$P(Y = 1|x) = \frac{exp\,(w \cdot x + b)}{1 + exp\,(w \cdot x + b)} \quad (3.4)$$

$$P(Y = 0|x) = \frac{1}{1 + exp\,(w \cdot x)} \quad (3.5)$$

Here, $X \subseteq R^n$ is input and $Y \in \{0,1\}$ is output. $w \subseteq R^n$ and $b \subseteq R$ are parameters. w and b are called weight vector and bias respectively. $w \cdot x$ is the inner product of the w and x.

For a given input instance x, according to (3.4) and (3.5), we can compute $P(Y = 1|x)$ and $P(Y = 0|x)$ The Logistic Regression splits the instance x to the greater probability value after comparing the two conditional probability values.

- Support Vector Machine

Solving for the separated hyperplane that appropriately divides the training dataset and maximizes geometric separation is the fundamental concept of SVM learning.

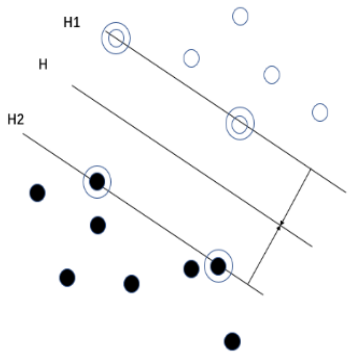

Figure 1 The Visualization of SVM

Figure 1 depicts the fundamental concept. The picture shows two different types of samples as hollow circles and solid points. The samples closest to the classification hyperplane in each type are traversed by the straight lines H1 and H2, which run parallel to the classification hyperplane H. The classification interval is the space between them. $w^T + b = 0$ is the classification hyperplane, and we normalize to create a set of samples (x_i, y_i), $i = 1, \dots, n, x \in R^m, y \in \{+1, -1\}$, that are linearly divisible and satisfy the following equations:

$$y_i(w^T x_i + b) - 1 \geq 0, \qquad i = 1, \dots, n$$

The classification interval is currently $\frac{2}{||w||^2}$. The interval must be maximized in order to minimize $||w||^2$ minimum. The samples on H1 and H2 are referred to as the support vectors, and the classification surface that meets the aforementioned criteria and minimizes $\frac{||w||^2}{2}$ is known as the optimal classification surface.

Maximizing the classification interval is the core concept of SVM. As a result, it is an effective approach for handling the classification problem.

A nonlinear classification problem is a problem that can be well classified by using nonlinear models. Let's start with one example: the left picture in Figure 2 is a classification problem, in which "·" represents a positive instance point and "×" represents a negative instance point.

As can be seen from the figure, positive and negative instances are not able to correctly partitioned by a straight line (linear model), but is able to correctly partitioned by an elliptic curve (not linear model) to separate them correctly.

Generally speaking, for a given training dataset $T = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$, where x_i belongs to the input space $x_i \in X = R^n$, corresponding to two types of tags $y_i \in Y = \{-1, +1\}, i = 1,2, \dots N$. If the positive and negative cases is able to correctly partitioned by a hyperplane in R^n, the problem is said to be non-linear classification problem.

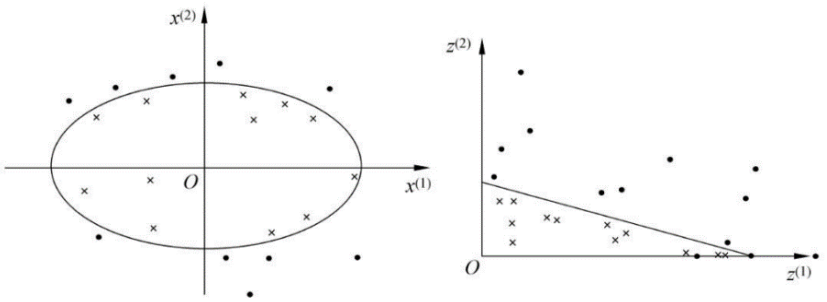

Figure 2 Example of nonlinear classification problems and kernel techniques

Nonlinear problems are poorly solved, so it is desirable to solve the problem in the same way as a linear classification problem. The method adopted is to carry out a nonlinear transformation, which transforms the nonlinear problem into a linear problem and solves the original nonlinear problem by solving the transformed linear problem. For the example shown in Figure 2, the nonlinear classification problem is transformed into a linear classification problem by transforming the centre circle in the left figure into a straight line in the right figure. Next, the principle is the same as the linear situation.

- Decision Tree

A decision tree model is a tree structure that specifies how instances are classified. There are nodes and directed edges in a decision tree. Nodes come in two varieties: internal nodes and leaf nodes. A class is represented by a leaf node, while an internal node represents a feature or property.

An instance is examined for a feature starting at the root node when using Decision Tree classification. Each child node now represents a value for that feature and the instance is assigned to them based on the test result. Up until the leaf node is reached, this is carried out recursively by testing and assigning. Finally, the instances are sent to the leaf nodes class.

A decision tree is depicted in a diagram in Figure 3. In the illustration, the circles and boxes stand for internal nodes and leaf nodes, respectively.

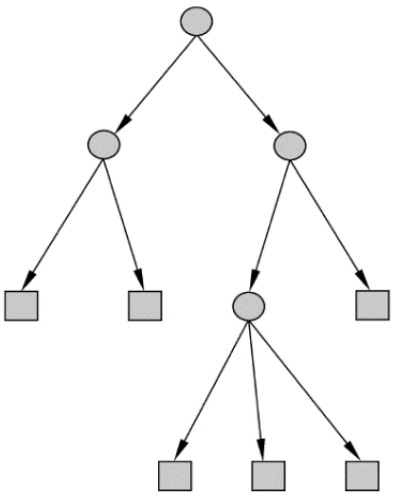

Figure 3 The Visualization of Decision Tree

The Decision Tree selects features by information gain, and determines the features with greater information gain as the basis for slicing.

Gini impurity can define information gain. Gini impurity is a gauge of a set's purity. Based on how many datapoints in the set are customers who defaulted and how many are customers who didn't, the Gini impurity could be calculated on a subset of our data. It will be a number between 0 and 0.5, where 0 is totally pure (100% of the same class survived) and 0.5 is completely impure (50% survived and 50% didn't).

Following is the Gini formula. *p* is the percent of customers who defaulted. Thus *(1-p)* is the percent of customers who didn't default.

$$Gini = 2 \times p \times (1 - p)$$

Our impurity measurement, Gini impurity is *H*. *S* stands for the original dataset, while *A* and *B* stand for the two sets we divide *S* into.

$$Information\ Gain = H(S) - \frac{[A]}{[S]}H(A) - \frac{[B]}{[S]}H(B)$$

- Random Forest

The main weakness of Decision Trees is that they have a propensity to overfit. It is possible to get a very different-looking tree if you randomly alter the training dataset, it is argued that Decision Trees have high variation. Random Forest is a model constructed using multiple trees and is designed to combine the benefits of Decision Trees with variance mitigation. Random Forest adopts the Bagging Algorithm. By performing multiple Bootstrap sampling, each sampled dataset is trained as a weak learner model, and the obtained multiple independent weak learner models are combined to predict the results. When the learner is a Decision Tree, this Bagging algorithm is called a Random Forest.

Specifically, a Random Forest can be thought of as the outcome of several Decision Trees. Each tree in the forest is an autonomous "expert," and each "expert" will classify the trees according to its area of knowledge. Finally, using a voting method to combine all of the expert classifications, we build the classifier. The most useful classification is then selected, and the Random Forest classifier's performance is evaluated using the out-of-bag error rate (OOB).

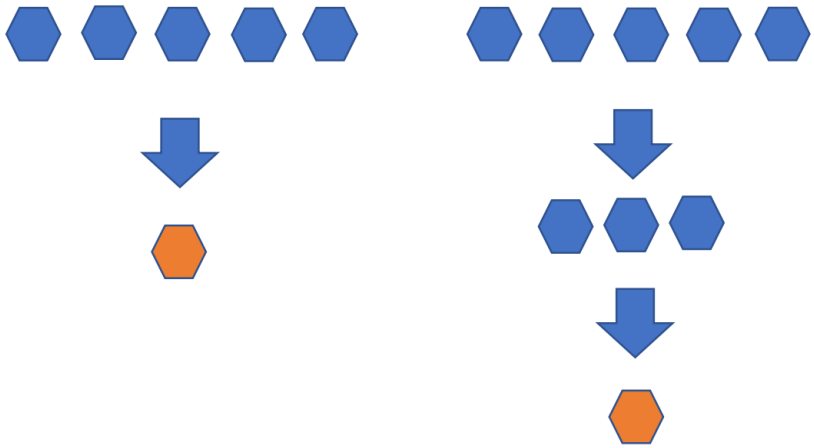

Figure 4 The selection of the classification features process by Decision Tree and Random Forest

3.1.2 Parameter Adjustment

To improve the performance metrics of the model, it is necessary to adjust model parameters. In terms of **Gaussian Naïve Bayesian**, we just needed to adjust priors and var_smoothing. As for **Logistic Regression**, there are the regularization coefficient, penalty, solver, multi_class, class_weight and sample_weight and etc. Among them, what we emphasize are the first four parameters. For **SVM**, there are C (also called regularization parameter), Kernel, Degree, Gamma, Probability, etc. Kernel is what we focused on. Concerning about **Decision Tree**, the criterion is Gini and we adjusted splitter and class_weight. As the sample size is small and missing values are few after data preprocessing, the other parameters are default values. **Random Forest** is more complicated. The more parameters there are, the more difficult it is to adjust the parameters, because there are more and more possibilities to combine them. Among the core parameters like n_estimators (the number of trees), max_depth (the depth of trees), etc., we first tried many values of different parameters and then detected the most suitable combination.

3.2 Model Evaluation

3.2.1 Confusion Matrix

The confusion matrix is a situation analysis table for summarizing the prediction results of a classifier in machine learning. The rows of the matrix represent the predicted values and the columns of the matrix represent the true values. For example, in a dichotomous classification

problem, the confusion matrix is a 2x2 situation analysis table.

		True Class	
		Positive (non-default)	Negative (default)
Predicted Class	Positive (non-default)	True Positive (TP)	False Positive (FP)
	Negative (default)	False Negative (FN)	True Negative (TN)

Table 1 Confusion Matrix

1. **Accuracy**

   The accuracy rate, which measures how well a classifier performs overall, is calculated as the ratio of correctly classified samples to all samples.

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN}$$

2. **Precision**

   The precision rate, which measures a model's reliability, is calculated as the ratio of positive samples that are correctly classified to positive samples that the classifier has judged to be positive.

$$Precision = \frac{TP}{TP+FP}$$

3. **Recall**

   Recall measures a model's effectiveness, which is calculated as the ratio of correctly identified positive samples to the total number of true positive samples.

$$Recall = \frac{TP}{TP+FN}$$

4. **F1 score**

   In general, a model cannot simultaneously have a high precision rate and a high recall rate. F1 score, which is the harmonic mean of precision and recall, is introduced to balance these two metrics.

$$F1 = \frac{2*TP}{2*TP+FN+FP}$$

3.2.2 ROC & AUC

ROC is the receiver operating characteristic curve. False Positive Rate (FPR) and True Positive Rate (TPR) are the ROC curve's horizontal and vertical coordinates, respectively.

$$FPR = \frac{FP}{TN+FP}$$

$$TPR = \frac{TP}{TP+FN}$$

The area enclosed by the coordinate axis under the ROC curve is known as the AUC (Area Under Curve). AUC has a value between 0 and 1, and the closer it is to 0.5, the less difference there is between the model's results and those of a random classifier. The effect of the model's classification is improved the closer the AUC gets near 1.

3.2.3 Cross Validation

Conducting evaluation on test set is to accurately assess the whole model's performance. However, if our dataset is small, the test set will not be large as well. As a result, it's possible that the selection of datapoints is not good. Moreover, if we just use one particular test set to obtain the evaluation metrics (accuracy, precision, recall, and F1 score), it is so limited. In order to get a measure of how well our model does in general, not just a measure of how well it does on one test set, we could break dataset into different chunks. Therefore, k-fold cross validation is applied.

In detail, the sample dataset is randomly divided into k subsets (usually equally divided), and one subset data is used as the test set, and the remaining k-1 sets of subsets are used as the training set. K subsets are used as test sets in turn, so that k results of evaluation metrics are obtained. After that, the average of the results is used to reflect the performance of the 5 classifiers. Moreover, 5-fold cross validation is more commonly used.

3.2.4 Time Complexity

Time complexity can be seen as a measure of how fast a machine learning algorithm can execute against the size of the input. It is also a reference factor to optimize the model.

Assuming that n = number of training examples, d = number of data dimensions, k = number of Decision Trees, we can get the complexity of 5 classifiers we used.

Classifier	Training Time Complexity
Logistic Regression	O(n*d)

SVM	O(n^2)
Decision Tree	O(n*log(n)*d)
Random Forest	O(n*log(n)*d*k)
Naïve Bayesian	O(n*d)

Table 2 Time Complexity of Five Classifiers

3.2.5 Model Complexity

Model complexity in machine learning frequently refers to the quantity of features used in a particular classifier, which involves the fitting ability of model. From the perspective of mathematical statistics, different training sets have different probability distribution functions. When our model has a good ability to fit the data distribution functions of the training set, it can be regarded as a good model.

The error on the training set gradually decreases as the complexity of the model gradually increases and when the complexity of the model reaches a certain level, the error on the test set instead increases as the complexity of the model increases. As can be seen from the figure 5, the corresponding x value of the lowest point of test sample curve is our desired model complexity, which performs well on both the training and test sets. We should exert utmost efforts in the process of feature selection to simplify model as long as it is effective.

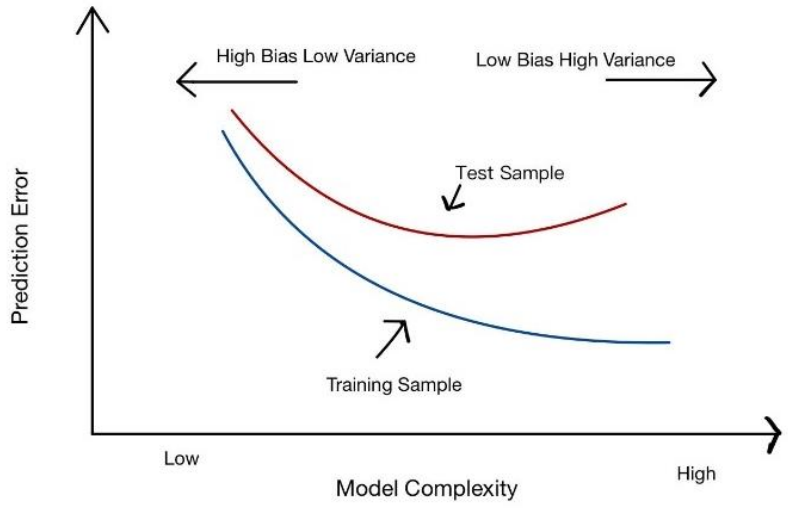

Figure 5 The Relationship of Model Complexity and Prediction Error

3.3 Introduction to Feature Selection Method

3.3.1 The significance of feature selection

We often select as many independent variables as we can at the beginning of the model establishment process for data mining and machine learning algorithms in order to reduce the model deviation brought on by the absence of significant variables. To increase the model's capacity for explanation and the precision of its predictions, it is typically required to identify a subset of independent variables that can explain the response variables. This process is called feature selection.

3.3.2 Lasso Regression

Robert Tibshirani initially proposed Lasso in 1996. Its full name is Least absolute shrinkage and selection operator. This method is a kind of compressed estimation. By building a penalty function that compresses particular regression coefficients, or compels the sum of absolute coefficient values to be less than a specific set value, a more refined model is obtained. Additionally, some regression coefficients are zero. As a result, it retains the advantage of subset shrinkage and is a biased estimator for processing data with complex collinearity.

Lasso regression model is a linear model for estimating sparse parameters, especially for parameter number reduction. Mathematically, Lasso adds an regularization term to the linear model, and its loss function is,

$$J(w) = \min_{m} \left\{ \frac{1}{2N} ||X^T w - y||_2^2 + \alpha ||w||_1 \right\}$$

Where α controls the degree of penalty of sparse parameter estimation which means to control the number of features in the model. But how to determine the value of α, there is no absolute judgment standard.

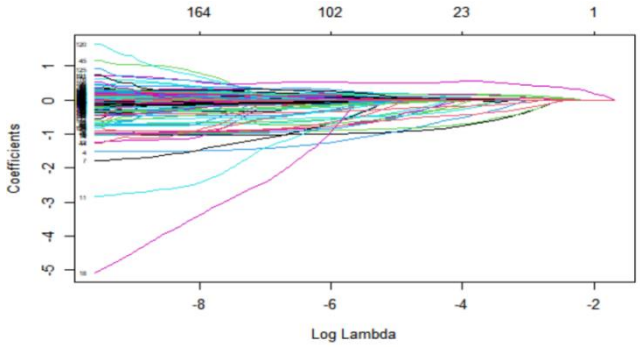

Figure 6 The Order of the Selected Features

The order in which the features are chosen is depicted in the figure 6. After Lasso regression, most parameters of the model are 0, and the features corresponding to parameters that are not 0 can be used to classify, which is the feature selection based on Lasso.

3.3.3 Random Forest

According to the Random Forest principle, each feature's contribution to each tree in the Random Forest is measured, averaged, and then the contribution of the various features is compared. The out-of-bag (OOB) error rate or the Gini index is frequently employed as an evaluation tool, and we used the Gini index to represent the average impurity decrease in the paper.

Denote Gini index by GI. The Gini index score VIMj (Variable Importance Measures), which is the average change in the impurity of the node splitting of the jth feature in all Decision Trees of the Random Forest, is calculated for each feature Xj, assuming there are m features X1, X2, ... Xm. Following is how the Gini index is written.

$$GI_m = 1 - \sum_{k=1}^{|K|} P_{mk}^2$$

The k in the formula denotes the existence of k categories, and p_{mk} denotes the percentage of category k in node m. (the amount of change in Gini index is calculated for feature m on a node-by-node basis).

The importance of feature X_j at node m, which measures how much the Gini index changes before and after node m branches, is

$$VIM_{jm}^{gini} = GI_m - GI_l - GI_r$$

where GI_l and GI_r stand for the corresponding Gini indices of the two new nodes created as a result of branching.

If the node where feature X_j appears in Decision Tree i is in the set M, then the importance of X_j in the ith tree is

$$VIM_j^{gini} = \sum_{m \in M} VIM_{jm}^{gini}$$

Assume the Random Forest contains n trees:

$$VIM_j^{gini} = \sum_{i=1}^{n} VIM_{ij}^{gini}$$

Eventually, the importance scores are calculated by normalizing all of the acquired importance values.

$$\mathrm{VIM_j} = \frac{\mathrm{VIM_j}}{\sum_i^c \mathrm{VIM_i}}$$

4. Model Construction and Solution

4.1 Data Preprocessing

4.1.1 Missing Value Handling

The original dataset totally has 11017 samples, 199 features and a categorical variable with two characteristics which indicates whether the customer is overdue (0 represents the customer is not overdue and 1 represents the customer is overdue). There are a lot of missing values in the dataset. Combing the analysis of the realistic significance of the features and the characteristics of the dataset, we filled most of the missing values with 0 and also deleted rows.

Specifically, x_001 which represents the gender has 31 missing values and we could not fill them with 0 and only deleted them. After that, the missing values of x_002 also were removed. x_003-x_039 have not missing values, so we did not need to process them. From x_041 to x_130, there exist some general rules. For instances, given the number of transactions is blank perhaps due to the number of transactions being 0 and there is no 0 in corresponding columns, so we filled the blank with 0. Since the number of transactions is 0, the number of trading months, the mean and mean divided by standard error of transactions, and the amount of transactions are all 0. If the number of missing values of transaction amount is not the same as the number of transactions, then it needs to further check x_0181 (number of failed trades). When the number of failed trades is not 0, it indicates that the value of transaction amount may be 0, and then we filled it with 0. Otherwise, we deleted the rows of missing values. In addition, x_107 (monthly average number of auto trades in

the past 6 months) has more missing values than x_105 (number of auto trades in the past 6 months) and x_106 (the number of months when there were auto trades in the past 6 months), so we filled in the 31 blanks by dividing x_105 by x_106. From x_131-x_187, the original data have 0, so we cannot fill in 0 and just delete the missing values. In the end, x_188-x_199 have no missing values. After processing missing values, the new dataset has 9120 samples.

4.1.2 Filter

Given the importance of every feature, we used the filter method for preliminarily screening features. If all the data of a feature are the same, whether Y is 0 or 1, then this feature does not play a classification effect. Therefore, we applied the **Variance Threshold** to filter features with zero variance. With the application of Python, we got the following result, which displayed the number of such features.

```
before filtration:
 (9120, 200)
after filtration:
 (9120, 199)
```

Figure 7 The Result of Variance Threshold

According to the figure 7, there is one feature with zero variance. Through the observation, it was x_012 (Health insurance mark), so we deleted it and the dataset remained 198 features and Y.

4.1.3 Data Standardization

Due to the different dimensions of the features and the large difference in the value of one feature, the effect of features with high value on Y will be larger, and those with low value will be weakened. Hence, in order to unify the dimensions, avoid numerical problems and balance the contribution of each feature, we used **z-score** to standardize data. In our dataset, there are 20 categorical features which are x_001, x_003-x_019, x_027 and x_033 and the others are non-categorical features. We only need to select those non-categorical features to do the standardization. After the z-score standardization, the mean value of the data is 0 and the standard deviation is 1. The formula is,

$$x^* = \frac{x-\mu}{\sigma}$$

where μ is sample mean, σ is sample standard deviation.

4.1.4 Outliers Detection & Removal

- Outlier visualization

Outliers generally have two characteristics: they are significantly different from most of the data and they account for a small proportion in the overall data. They usually take a relatively large error to the classification and prediction, so it is needed to detect and remove the outliers. To visualize all the data, we plotted a graph which could display the range and the distribution of the data.

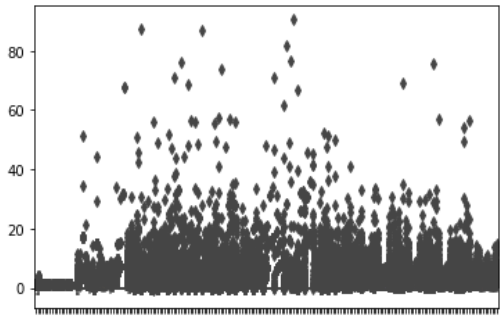

Figure 8 The Plot of Outliers

According to the figure 8, the dataset certainly existed some outliers.

- Outlier removing

Given that the dataset is high-dimensional and needs a high accuracy, we used **Insolation Forest** to remove outliers. Isolated Forest is a machine learning algorithm for outlier detection, which is an unsupervised learning algorithm. In order to simplify the principle, suppose a random hyperplane is used to cut a data space, and two sub-spaces can be generated at once. Next, continue to randomly select hyperplanes to cut the two sub-spaces obtained in the first step, and continue the cycle until each subspace contains only one data point. Intuitively, clusters with high density need to be cut many times before they stop cutting, that is, each point exists in a subspace alone. But as for the points that are sparsely distributed, most of them have been partitioned into a subspace long ago.

After deleting the outliers, the dataset remained 8974 samples.

4.1.5 Data Balancing

Currently, in the classification variable Y, the number of 0 (not overdue) is 7034 and 1 (overdue) is 1940, which is quite imbalanced and the proportion is close to 3.62 to 1. Imbalanced data can cause the predicted results of trained models to be biased towards the category with a large number of samples in the classification problem. Therefore, it is needed to balance the dataset before putting data into different kinds of classifiers.

There are mainly three methods to deal with imbalanced data, which are Under-sampling, Over-sampling and SMOTE Algorithm.

- Under-sampling means that the number of a large number of categories (denoted as majority) are sampled so that the number is equal to the number of a small number of categories (denote as minority), so as to achieve a balance. It abandons part of the data, which inevitably changes the distribution and the variance of majority.
- Over-sampling is to duplicate minority in the same number as majority in order to achieve a balance of numbers. Since multiple minority samples are copied, the variance of minority will be changed.
- SMOTE Algorithm is to conduct operations on minority and majority at the same time, by Over-sampling first and then Under-sampling.

Adopting **SMOTE Algorithm** to balance the data, we got a balanced dataset with equal numbers of 0 and 1 in Y. Finally, the total number of samples is 14052 and the number of 0 and 1 are both 7026.

4.2 Application of Five Classifiers

Put the balanced data into five classifiers including Random Forest (RF), SVM, Decision Tree (DT), Logistic Regression (LR) and Naïve Bayesian (NB) to display the classification effect. We obtain the confusion matrix respectively which can further calculate the evaluation metrics.

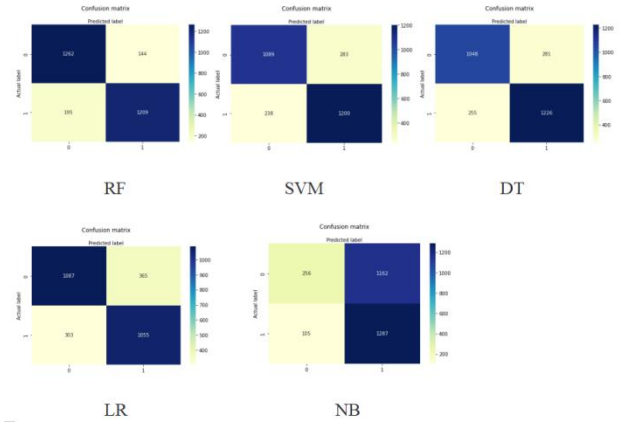

Figure 9 The Confusion Matrix of Five Classifiers

According to the figure 9, we can clearly see that RF has the best classification effect because the color of its region of diagonal is the deepest. By parity of reasoning, SVM and DT have approximate effects which rank only second to RF. And the next is LR, the color of its top right corner is deeper than the previous three. At the last, there is the worst effect in NB, which has serious error in the precision. Furthermore, the plots of AUC are as follows.

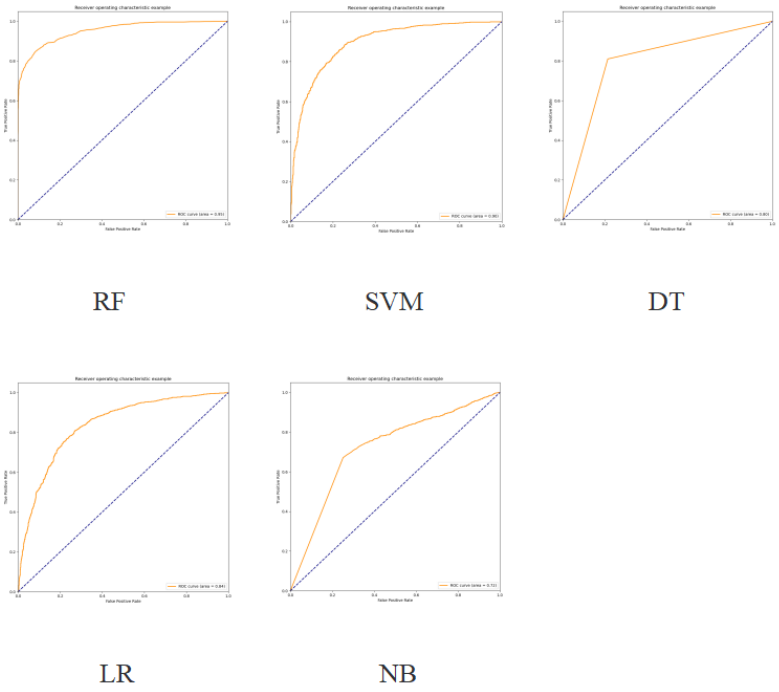

Figure 10 The Plots of ROC Curves and AUC Metrics

According to the figure 10, RF has the largest AUC which is 0.95, then followed by SVM, LR, DT, and the smallest one is NB which is 0.72. Therefore, RF has the best classification effect.

There is a summary of all the evaluation metrics.

Operation	Metrics	RF	SVM	DT	LR	NB
After SMOTE Algorithm	Accuracy	0.8745	0.8080	0.8001	0.7500	0.5601
	Precision	0.8765	0.7867	0.7924	0.7502	0.5349
	Recall	0.8717	0.8454	0.8131	0.7500	0.9199
	F1	0.8740	0.8149	0.8025	0.7499	0.6764
	AUC	0.8744	0.8081	0.8001	0.7502	0.5601
	Time	42sec	2min55sec	5sec	14sec	1sec

Table 3 Summary of the Evaluation Metrics of Five Classifiers

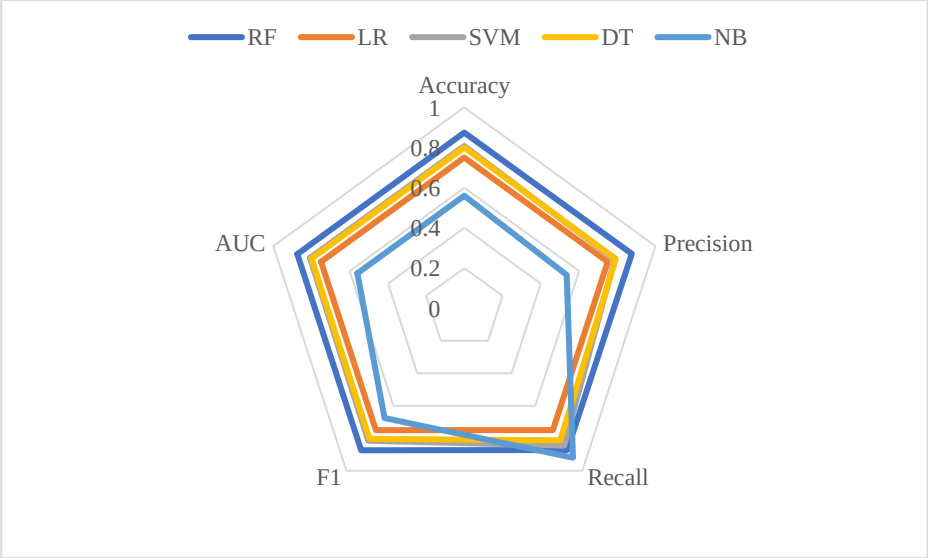

Figure 11 The Visualization of the Performances of Five Classifiers

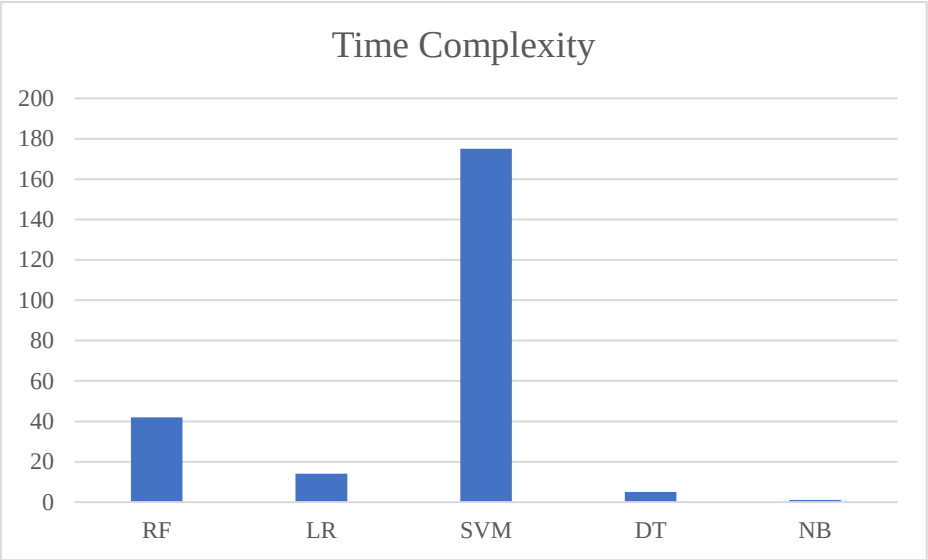

Figure 12 The Running Time of Five Classifiers

As can be seen from the above results, RF is the only classifier in which all the metrics are greater than 0.85 and others are lower than 0.85. Specially, the metrics of SVM and DT are around 0.8, and those of LR are around 0.75. NB is greatly different which has large difference among metrics. It has the largest recall in all classifiers which is 0.9199, but other four metrics are only in range (0.5, 0.7).

In addition, from the view of the time complexity, SVM has the longest running time which is close to 3 minutes, followed by RF with 42 seconds, LR with 14 seconds, DT with 5 seconds, and then NB with just 1 second to run.

On the whole, paying more attention to the classification effect, supplemented by the time complexity, RF performs best, followed by DT, SVM, LR, and NB.

Rank	Classifiers
1	Random Forest
2	Decision Tree
3	Support Vector Machine
4	Logistic Regression
5	Naïve Bayesian

Table 4 The Rank of Five Classifiers

Furthermore, since banks and credit customers perhaps have different concerns, they should be discussed separately. From the banks' perspective, they are mainly concerned with whether overdue customers are correctly predicted to be overdue. Hence, they pay more attention to the **precision** which can reflect the reliability of a model. It can be clearly seen that RF has the highest precision which is 0.8765, the values of SVM and DT are about 0.8, the value of LR is about 0.75, and the value of NB is the lowest which is 0.5349. So we recommend the bank to choose RF as the classifier. From the customers' perspective, perhaps they focus on whether banks predict them in overdue but they are not. Hence, they pay more attention to the recall which can reflect the effectiveness of a model. From the plot, NB has the highest recall which is 0.9199, the values of RF and SVM are around 0.85, the value of DT is around 0.80, and LR has the lowest value which is 0.75. So the customers may prefer the bank to use the classification results of NB first.

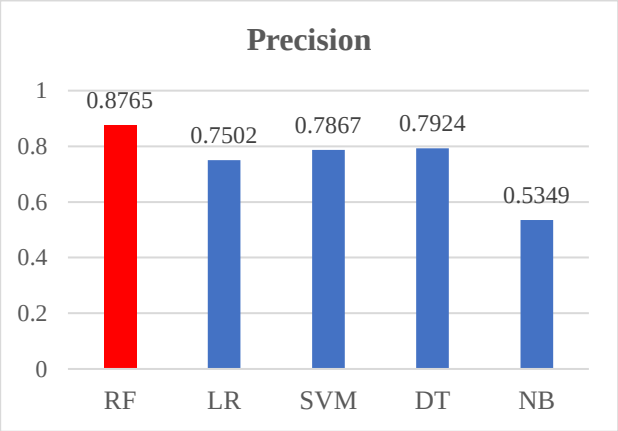

Figure 13 The Evaluation Metric of Precision

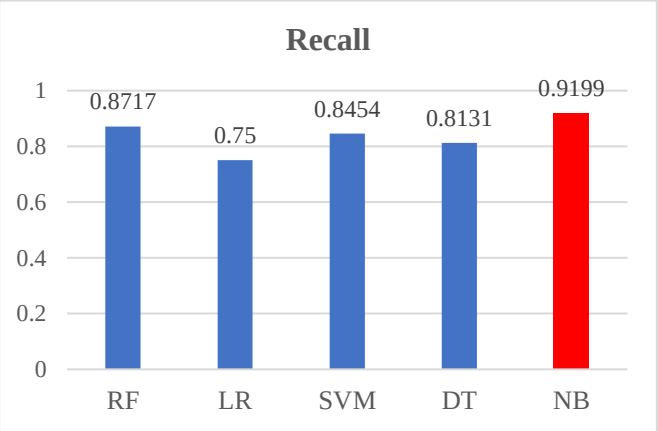

Figure 14 The Evaluation Metric of Recall

Finally, as the harmonic mean of recall and precision, F1 also deserves our attention. It can be seen that RF has the best performance in this metric. Therefore, RF is the best choice after multiple evaluations.

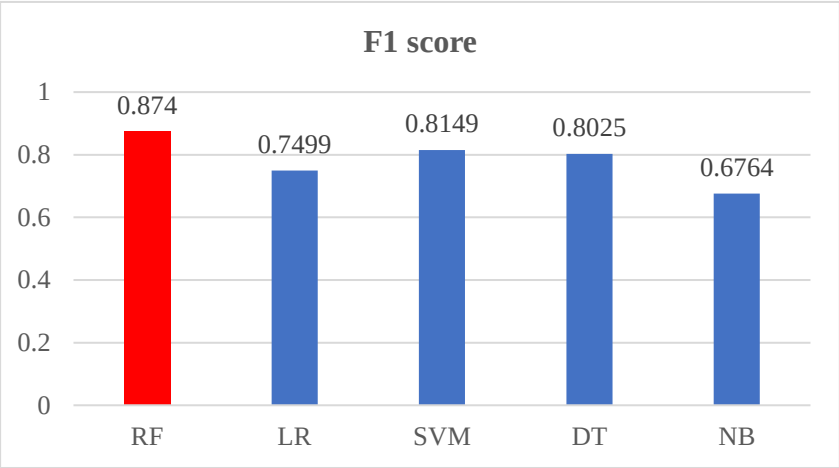

Figure 15 The Evaluation Metric of F1

4.3 Feature Selection

Trying to further improve the performance of five classifiers and get some useful, practical, and universal conclusions from all of the features, we adopted two methods of the feature selection which are Lasso regression and Random Forest.

4.3.1 Lasso Regression

Setting the parameter α to 1, 0.1, and 0.01, respectively, we obtained the table 5.

Model	α	The number of features that were selected
Lasso1	1	90
Lasso2	0.1	160
Lasso3	0.01	186

Table 5 The Number of Left Features under Different α Thresholds

If α is equal to 1, the model used 89 out of 198 features. If α fell to 0.1, the model used 160 features. And when α fell to 0.01, the model used 186 features.

Three new datasets were obtained by selecting features according to the above screening results. After that, put the new datasets into five classifiers respectively, and the results could be displayed in table 6.

Metrics	Model	RF	LR	SVM	DT	NB
Accuracy	SMOTE Algorithm	0.8745	0.75	0.808	0.8001	0.5601
	Lasso1*	0.8754	0.7589	0.8141	0.8002	0.5533
	Lasso2	0.8742	0.755	0.8102	0.7941	0.5572
	Lasso3	0.8781	0.7523	0.8108	0.7991	0.5575
Precision	SMOTE Algorithm	0.8765	0.7502	0.7867	0.7924	0.5349
	Lasso1*	0.8778	0.7559	0.7974	0.7936	0.5301
	Lasso2	0.8767	0.7538	0.7888	0.7855	0.5331
	Lasso3	0.8806	0.7528	0.7895	0.7926	0.5332
Recall	SMOTE Algorithm	0.8717	0.75	0.8454	0.8111	0.9199
	Lasso1*	0.8722	0.7644	0.8422	0.8112	0.9396
	Lasso2	0.8706	0.7573	0.8474	0.8093	0.9214
	Lasso3	0.8751	0.7515	0.8479	0.8103	0.9242
F1	SMOTE Algorithm	0.874	0.7499	0.8149	0.8022	0.6747
	Lasso1*	0.8749	0.76	0.8191	0.8025	0.6778
	Lasso2	0.8736	0.7555	0.817	0.7972	0.6754
	Lasso3	0.8778	0.7521	0.8176	0.8013	0.6761
AUC	SMOTE Algorithm	0.8744	0.7502	0.8081	0.8001	0.5601
	Lasso1*	0.8754	0.7588	0.8142	0.8002	0.5534
	Lasso2	0.8741	0.7549	0.8103	0.7942	0.5571
	Lasso3	0.8781	0.7524	0.8109	0.7991	0.5575
Time	SMOTE Algorithm	42sec	14sec	2min55sec	5sec	1sec
	Lasso1	40sec	12sec	1min35sec	5sec	0sec3789
	Lasso2*	34sec	10sec	2min41sec	4sec	0sec9894
	Lasso3	51sec	14sec	2min48sec	6sec	1sec

Table 6 The Performances of Five Classifiers after SMOTE Algorithm and Each Lasso Model

Note: The results marked in red represent the optimal performances; the signal of '' represents*

Lasso X, X = 1, 2, 3 has the most optimal performances in each metric, which indicates the model we should choose in the corresponding metric.

The results show that most of the metrics after feature selection using Lasso regression have further been improved compared with the results after SMOTE Algorithm, except the accuracy, precision and AUC of NB have been slightly reduced.

From the results of the three Lasso models, it can be seen that Lasso1 performs best in most metrics of four classifiers except RF, while Lasso3 has the greatest improvement of all the four metrics in RF. Given that Lasso3 still has too many features, which is easy to take up overfitting problem and its time complexity is also high, this paper chose Lasso1 as our final model to do the further experiment.

In this stage, from the view of banks, the precision they pay attention to was increased by 0.15%-1.36%. From the view of customers, the recall was increased by 0.06%-2.14%. And for the harmonic mean of them, the F1 score was increased by 0.037%-1.35%. Moreover, the time complexity decreased, which greatly enhanced the classification efficiency. Therefore, Lasso regression plays a positive role in the evaluation metrics of the classifiers.

4.3.2 Random Forest

After the Lasso regression, we used Random Forest to select features again. Figure 16 shows the importance of all features. The feature importance decreases from top to bottom.

```
Features sorted by their score:
[(0.1987, 'x_135'), (0.0657, 'x_138'), (0.0489, 'x_187'), (0.045, 'x_119'), (0.0337,
'x_140'), (0.0224, 'x_015'), (0.022, 'x_001'), (0.0165, 'x_141'), (0.0164, 'x_033'),
(0.0163, 'x_196'), (0.0163, 'x_188'), (0.0156, 'x_126'), (0.0151, 'x_030'), (0.0148,
'x_192'), (0.0145, 'x_179'), (0.0144, 'x_186'), (0.0141, 'x_122'), (0.0135, 'x_145'),
(0.0135, 'x_129'), (0.0133, 'x_163'), (0.0129, 'x_071'), (0.0127, 'x_137'), (0.0127,
'x_029'), (0.0126, 'x_024'), (0.0124, 'x_025'), (0.0122, 'x_058'), (0.0121, 'x_076'),
(0.0119, 'x_152'), (0.0113, 'x_175'), (0.0113, 'x_056'), (0.0112, 'x_051'), (0.0105,
'x_164'), (0.0105, 'x_155'), (0.0104, 'x_042'), (0.0101, 'x_144'), (0.0101, 'x_142'),
(0.01, 'x_185'), (0.0099, 'x_143'), (0.0098, 'x_183'), (0.0095, 'x_153'), (0.009, 'x_087'),
(0.0085, 'x_173'), (0.0084, 'x_132'), (0.0084, 'x_050'), (0.0083, 'x_053'), (0.0083,
'x_032'), (0.0075, 'x_151'), (0.0074, 'x_026'), (0.0073, 'x_040'), (0.0064, 'x_146'),
(0.0059, 'x_131'), (0.0058, 'x_028'), (0.0054, 'x_161'), (0.0047, 'x_031'), (0.0043, 'y'),
(0.0039, 'x_109'), (0.0037, 'x_060'), (0.0036, 'x_023'), (0.0034, 'x_017'), (0.0029,
'x_095'), (0.0026, 'x_091'), (0.0022, 'x_002'), (0.0017, 'x_003'), (0.0016, 'x_085'),
(0.0014, 'x_127'), (0.0013, 'x_128'), (0.0013, 'x_034'), (0.0011, 'x_084'), (0.0011,
'x_069'), (0.001, 'x_077'), (0.0009, 'x_114'), (0.0008, 'x_066'), (0.0008, 'x_022'),
(0.0006, 'x_007'), (0.0006, 'x_004'), (0.0005, 'x_016'), (0.0005, 'x_005'), (0.0004,
'x_070'), (0.0003, 'x_108'), (0.0003, 'x_106'), (0.0003, 'x_102'), (0.0002, 'x_096'),
(0.0002, 'x_019'), (0.0001, 'x_116'), (0.0001, 'x_027'), (0.0001, 'x_011'), (0.0001,
'x_006'), (0.0, 'x_037'), (0.0, 'x_018'), (0.0, 'x_009')]
```

Figure 16 The Importance of Features

We deleted the last 12 features with feature importance less than 0.0005 and then there were 77

features in the dataset.

Metrics	Model	RF	LR	SVM	DT	NB
Accuracy	Lasso 1	0.8754	0.7589	0.8141	0.8002	0.5533
	Lasso 1 and Random Forest	0.8755	0.7447	0.8042	0.8	0.5593
Precision	Lasso 1	0.8778	0.753	0.7974	0.7936	0.5301
	Lasso 1 and Random Forest	0.8844	0.7409	0.7809	0.7944	0.5335
Recall	Lasso 1	0.8722	0.7511	0.8422	0.8112	0.9396
	Lasso 1 and Random Forest	0.8638	0.7526	0.8459	0.8095	0.9449
F1	Lasso 1	0.8749	0.7519	0.8191	0.8022	0.6778
	Lasso 1 and Random Forest	0.8739	0.7466	0.8121	0.8019	0.6819
AUC	Lasso 1	0.8754	0.7524	0.8142	0.8002	0.5534
	Lasso 1 and Random Forest	0.8755	0.7447	0.8043	0.8	0.5593
Time	Lasso 1	42sec4005	13sec0630	1min35sec9097	5sec5243	0sec3789
	Lasso 1 and Random Forest	34sec5526	5sec4935	2min05sec3781	2sec3688	0sec3853

Table 7 The Performances of Five Classifiers after the Feature Selection of Lasso 1 and Lasso-RF

Note: The results marked in red represent the optimal performances.

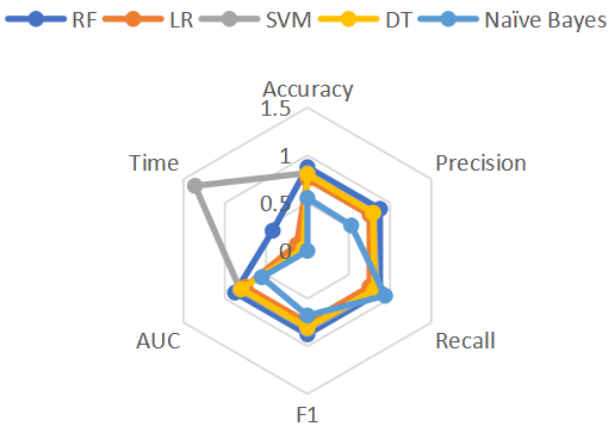

Figure 17 The Visualization of the Performances of Five Classifiers after Lasso 1

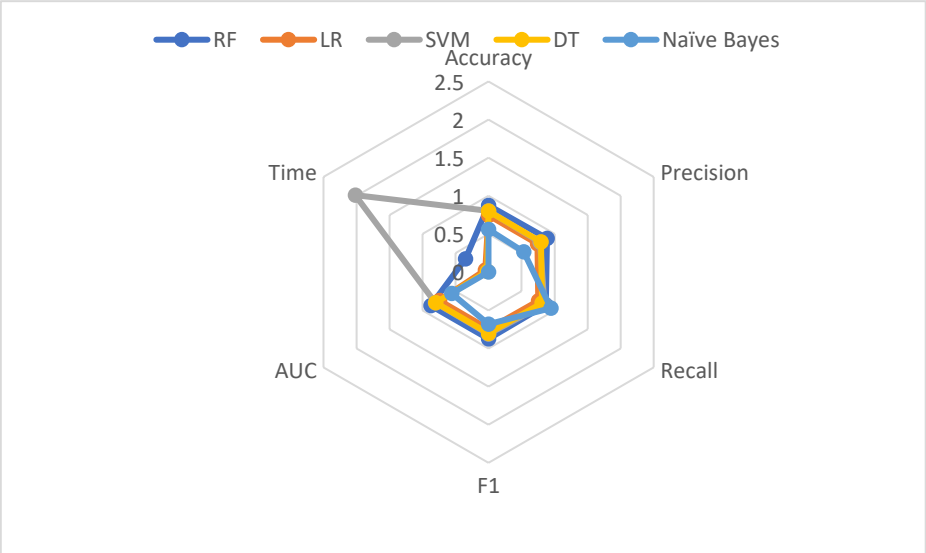

Figure 18 The Visualization of the Performances of Five Classifiers after Lasso-RF

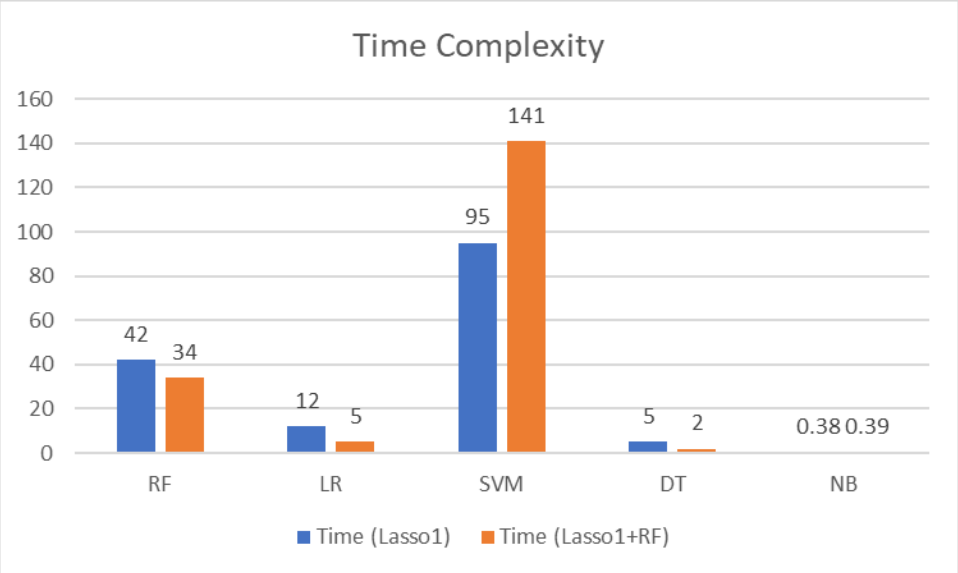

Figure 19 The Comparison of Time Complexity

From the comparison of the feature selection of Lasso 1 and Random Forest after Lasso 1, the results of accuracy, precision and AUC of Random Forest which had the best performance previously improved and all the evaluation metrics of Naïve Bayesian were optimized. Moreover, it

is obvious that the running time of most classifiers decreased a lot except SVM. Therefore, the method of Random Forest after Lasso1 is our final choice to conduct the feature selection.

5. Result & Interpretation

After the feature selection of Lasso 1 and Random Forest, we chose the top 20 features in the feature importance list. (Considering the sensitivity of gender discrimination, we will not take gender into account.)

The practical significance of the deleted features:

- Identity information and property status

From this aspect, 8 features were deleted during the process, namely x_006 (high consumption mark[1]), x_008 (wedding consumption mark), x_009 (insurance mark), x_010 (life insurance mark), x_011 (personal insurance mark), x_012 (health insurance mark), x_013 (property insurance mark), x_014 (investment and financial management mark), x_015 (bank financial management mark) and x_019 (maternal and child/education consumption). In the 11017 samples, only a few have these marks. In detail, firstly, only 5 people have wedding consumption and 4 people have maternal and child/education consumption, because these two indicators are not common in daily life, and only have a large impact on certain families. Secondly, there are just 257 people have high consumption mark which indicates most people don't spend or admit to spending more than they can afford. Thirdly, there are also few people with insurance related to life, health, property and so on. Finally, there is approximately equal number of people in having or not having investment and financial management mark and bank financial management mark. Since they have a low threshold for applicants, they have little impact on identifying whether a customer will default or not. To sum up, these features are not important.

Credit customers who own less property are at greater risk. Hence, it is necessary to collect identity information and property status of customers. Among them, x_015, whether the customer has the fund is the most essential. Fund is risky. Considering the cost of time and energy, the investors hope to obtain high return, so a considerable amount of principal is a must. Thus, people who have invested in fund has economic strength in general.

[1] High consumption: individuals or groups in the society spend more than they can afford to consume.

- Card information

From this aspect, 7 features were deleted during the process, namely x_020 (number of debit cards), x_021 (number of credit card), x_027 (the highest level of debit card which the customer has), x_035 (Number of national joint-stock commercial bank cards), x_036 (Number of Postal Savings Bank cards), x_037 (Number of local city business cards), x_038 (Number of agricultural and commercial bank cards), and x_039 (Number of foreign bank cards).

There are two views to think about it. From the view of debit card and credit card, the number of cards does not matter, and what matters is the number of cards under the subdivision level. From the view of commercial, postal savings and foreign capital banks, the number of bank cards is not important. The corresponding banks of x_035-x_039 are niche, so the bank could not need to collect the related information. Furthermore, since the threshold to entry is very low for commercial and postal savings banks, people can apply only with the ID card after the age of 16. On the other hand, foreign banks have high thresholds to entry, which leads to only 10 people have it and they only have one. All of the above mentioned features do not influence the classification effects.

Moreover, the banks ought to attach great importance to x_030 (number of credit card (gold card) holders) and x_033 (the highest level of credit card holders). The more money the customers deposit in the bank, the higher level the credit card is. However, for the ordinary wealthy people, it is unlikely to have many platinum cards and diamond cards but owning several gold cards is likely. As a result, both of them can reflect a person’s financial status.

- Transaction information

In the general direction, the number of transactions is not important, the transaction amount is more important.

x_071 (number of tax payments in the last 6 months), x_085 (amount of large transactions in the last 6 months), x_119 (number of months with education spending in the last 6 months), x_122 (number of months with financial transactions in the last 6 months), x_126 (average value of financial transactions in the last 6 months) and x_129 (number of months with gasoline transactions in the last 6 months) are all vital features. When a person gains income through more channels, they need to pay more different kinds of tax, which means the person diversifies his or her risk via

various earnings. If a person has a large amount of money in one transaction, to some degree, it means that he or she has the financial ability. Nowadays with the development of education and finance, the cost and return of investment in the educational and financial products are high as well. If a person has many gasoline transactions, it is probable that he or she has the fixed asset—car, and they can afford the expense of gasoline every month with a decent income. Therefore, the financial situation of a person can be reflected through these features.

- Lending information

The lending information within 30 days has no significant effect on the classification of Y, while the data of 90 days and 180 days are more important, especially the period of 90 days. The specific information relates to the number of lenders, the total amount, the total number of loans, the maximum and minimum amount of a single loan as well. Under normal conditions, small loans are made over a period of between one and three months. For example, x_138 (the total lending amount within 90 days) and x_145 (minimum single lending amount within 90 days) are crucial. Additionally, the reason why the data related to 30 days does not matter is borrowers need to submit a series of materials to the bank in addition to the loan application, and prepare the personal loan application materials required by the bank. However, there are preserved features with great importance such as x_135, x_140, x_145 which are maximum single lending amount within 30 days, 90 days and 180 days respectively. Therefore, the maximum amount of a single loan is a significant factor.

- Repayment information

The repayment information within 30 days and 180 days has no significant influence on the classification of Y, while the data within 90 days has more influence. The specific information relates to the number of (successful/failed/failed due to lack of balance) repayment institutions, the number of (successful/failed/failed due to lack of balance) repayments, and the detailed information about repayment amount, such as the maximum and minimum amount of a single repayment, such as x_163 (the number of successful repayment institutions within 90 days). These indicate that a quarterly repayment period is more appropriate for the classification problem of small loans. Besides, x_187 (minimum single repayment amount within 180 days) and x_179 (number of repayments within 180 days) are essential. When a customer has borrowed money for a long time

without repayment, the banks should feel more alert due to a higher probability of default.

- Application for loan information

No matter how long the time period is, 30 days, 60 days or180 days, the number of applied loan institutions which are x_188, x_192, and x_196 has a great impact on the classification of Y, while details such as the number of successfully applied loan institutions and the number of loans are not important. The reason why only the number of applied loan institutions matters is the classifiers just kept x_188, x_192 and x_196 compared with other features like the number of successful lending institutions which have a strong linear correlation with them.

6. Advantage & Disadvantage of the Model

Metrics	RF	LR	SVM	DT	NB
Accuracy	1	4	2	3	5
Precision	1	4	3	2	5
Recall	2	5	3	4	1
F1	1	4	2	3	5
AUC	1	4	2	3	5
Time	4	3	5	2	1

Table 8 The Comparison of Five Classifiers based on Evaluation Metrics

Note: The numbers in the table represent the ranking level of a classifier's performance on a metric, with "1" representing the best performance and "5" representing the worst performance.

To sum up, RF has the great performances on all metrics except time, which indicates it has the best classification effect but low efficiency. SVM has ordinary performances on all metrics and the least efficiency. DT also has ordinary performances on all metrics but a better efficiency. LR has worse performances and medium efficiency. NB has the best efficiency and effectiveness but the worst reliability and classification effect.

	Advantage	Disadvantage
DT	This model makes it easier to understand the degree to which different attributes affect the	Because the tracing result only requires changing the properties of the leaf node, it

	results (for example, at what level). Different types of data can be processed simultaneously.	is vulnerable to attack.
RF	It is a random integration of Decision Tree, which improves its vulnerability to attack to some extent. It is applicable when the data dimension is not too high (dozens) and you want to achieve high accuracy. Don't need to adjust too many parameters, suitable for use when you don't know what method to use first.	The attribute weights generated by the Random Forest on data with diverse attribute values are unreliable because attributes with more value division will have a bigger influence on the Random Forest.
SVM	It can screen the most effective training samples in the massive and even high dimensional data. The generalization ability is stronger than that of linear classifier, which can be used for nonlinear classification and the results are easy to interpret.	The training cost is high; the parameter adjustment and the choice of kernel function will affect the final effect.
LR	It can deal with uneven data without the need to standardize and quantify the data. It can be applied to continuous and categorical independent variables. The logic is intuitive and has clear interpretation.	Only local optimal results can be achieved because of random errors or noise, which leads to overfit easily.
NB	This is true under the strong hypothesis: when the target value is given, the attributes are independent of each other, which reduces the calculation parameters and saves internal consumption and time. The algorithm is simple and fast.	Prior probability is required, and classification decision has error rate. The independence hypothesis is not always satisfied.

Table 9 The Pros and Cons of a Algorithms about Five Classifiers

7. Model Improvement

In general, there are 3 directions for model improvement: choosing a better algorithm, adjusting the model parameters, and improving the data.

- Better algorithm

For the problems of classification, we can try NN (Neural Network) and XGBoost (eXtreme Gradient Boosting).

Neural Network is one kind of deep learning. Artificial neuron networks are structurally parallel and the units of the network can perform similar processes simultaneously. Therefore, the information processing in the network is carried out in a large number of units in a parallel and hierarchical manner with high computational speed. Apart from the speed, Neural Networks are highly self-learning, self-organizing and self-adaptive so they can process very complicated data. When encountering the problems of credit assessment, it is avoidable to collect abundant and complex data. Neural Network can perform better when there are more data and this ability makes Neural Network distinguished.

XGBoost is one kind of ensemble learning. The goal of traditional machine learning algorithms such as Decision Tree, artificial neural network, SVM, Naïve Bayesian is to find an optimal classifier that separates the training data as much as possible. The basic idea of ensemble learning algorithms is to combine multiple classifiers to achieve an integrated classifier with better prediction results. In addition to its distinct approach to tree generation and pruning, XGBoost has several built-in enhancements to speed up training when working with huge datasets. For instance, the approximate greedy algorithm uses weighted quantiles to choose the best node split rather than evaluating each candidate split individually. Moreover, when there are some missing data points, Aware Split Finding calculates Gain by moving the observations with the missing values to the left leaf. By putting them in the appropriate leaf and choosing the scenario with the largest Gain, it then repeats the procedure.

- Model parameters

Hyperparameter adjustment is a commonly used method for model adjustment. In machine learning models, some parameters that need to be selected before the learning process starts are called hyperparameters, which obviously affect the outcome of the learning process. Adjusting the hyperparameters allows us to obtain the best results in the learning process very quickly. The results might be better if we had used publicly available libraries to help with hyperparameter adjustment, such as optuna. Optuna provides a Bayesian-based approach to making hyperparameter optimization simple and effective.

- Size of dataset

Getting more training data is an obvious and effective way to improve model performance. More training data allows the model to find more insights and obtain better evaluation metrics. Furthermore, a new test set will check the generalizability of the model.

8. Conclusion & Suggestion

8.1 Conclusion

Firstly, the study depends on the rationality of data preprocessing like Isolation Forest to eliminate the outliers and SMOTE Algorithm to handle imbalanced data.

Secondly, we studied and compared four discriminative models: Logistic Regression, SVM, Decision Tree and Random Forest and one generative model, Naïve Bayesian. Since evaluation metrics are significant, we tested each algorithm one by one through k-fold cross-validation, compared them, adjusted the parameters to ensure that each algorithm reached the optimal solution, and finally chose the best one.

Thirdly, to further enhance the performance of five classifiers and decrease the time and model complexity, we also conducted the Lasso-RF two-stage feature selection method. The method proposed in this paper effectively optimizes the selection of credit features. It also improves the evaluation metrics in traditional ideas, combining with the advantages of the machine learning algorithm and addressing the complexity of the high-dimensional characteristics of China UnionPay credit data. The purposes of every stage of the feature selection process are as follows: the first stage uses the relationship between features and classes to remove irrelevant features. The second stage is to remove irrelevant features by calculating the importance of attributes and to remove feature variables that do not contribute to the model. The classification results can help credit institutions to carry out credit assessments and reduce risk losses.

Finally, even if the goal of machine learning business applications is to provide judgments about decisions, the more interpretable a model is, the simpler it is for people to comprehend why particular judgments or forecasts are made. Model interpretability is the ability to comprehend both the internal mechanisms of the model and its outputs. Its importance is reflected in the modelling phase, which assists developers in understanding the model, making a comparative model selection,

and optimizing and adjusting the model when necessary; and in the operational phase, explaining the internal mechanism of the model to the business side and interpreting the model results. Machine learning algorithms can manage extremely complex interactions between the dependent and independent variables and achieve exceptionally high accuracy because they are able to make abstract inferences at multiple levels. Nevertheless, this complexity also makes the model a black box, and we cannot be informed of the relationships between all these features that produce the model's prediction results, so we have to use evaluation metrics instead to assess the credibility of the model. In fact, the machine learning process for every classification problem should include model understanding and model interpretation for several reasons, as follows:

- Model improvement: understanding metric features, classification, prediction, why a machine learning model makes the decisions it does, and what features play the most essential role in the decision, allows us to determine if the model is consistent with common sense.
- Model credibility and transparency: understanding machine learning models is essential in improving model credibility and providing transparency in examining prediction results and it is unrealistic to allow black box models to dictate people's lives.

Therefore, our explanation for the practical meaning is significant to the interpretability of the model. Besides applications, we also elaborated the theories of different evaluation metrics and classifiers to make it more straightforward for readers to read and understand.

8.2 Suggestion

Based on the above findings, the following recommendations are made.

1. Improve the quality of data collection and improve big data technology.

On the one hand, we should continuously optimize and update data preprocessing technology to accurately identify customers, resolve inaccurate customer information, solve the problems of inaccurate and incomplete customer information, and ensure the authenticity and validity of customer information. On the other hand, we should constantly update and develop data storage technology, data mining technology, machine learning and other related technology to ensure the accuracy and interpretability of credit evaluation model predictions.

2. Develop scientific and unified personal credit evaluation standards.

Currently, there is no unified standard for collecting customer information by credit institutions in China. Some credit institutions use the customer's income and loan repayment time as important evaluation indexes. In contrast, some credit institutions base on customers' basic information, such as age, occupation, assets, etc., directly as indicators for credit assessment. This has increased the workload of data processing. Therefore, it is necessary to establish a unified personal credit evaluation standard to comprehensively evaluate individuals' credit status.

3. Strengthen the penalties for defaulters to purify the Internet financial environment.

We should include defaulting customers, such as those who do not repay their loans in time into the credit system and further strengthen the penalties for defaulters. The information exchange between government departments and credit agencies should be accelerated. The financial environment should be purified, and the crackdown on non-compliant and defaulting customers should be strengthened.

4. Select the features through the analysis of practical meaning

Commercial banks collect customers' financial data because of the volatility of cash flows. For the scoring purposes, it is advantageous to have an unconcentrated cash flow because risk depends in part on the regularity of cash flows, or the mandatory versus voluntary nature of cash flows. Therefore, the features that can show the customers' awareness of diversification should be considered.

Reference

Bekhet, H. A., & Eletter, S. F. (2014). Credit Risk Assessment Model for Jordanian Commercial Banks: Neural Scoring Approach. *Review of Development Finance, 4*(1).

Bellotti, T., & Crook, J. (2009). Support Vector Machines for Credit Scoring and Disco Very of Significant Features. *Expert Systems with Applications, 36*(2), 3302–3308.

Breiman, L. (2001). Random Forests. *Machine Learning, 45*(1), 5–32.

Cano, G., Garcia-rodriguez, J., Garcia-garcia, A., Sanchez, H. P., Benediktsson, J. A., Thapa, A., & Barr, A. (2017). Automatic Selection of Molecular Descriptors Using Random Forest: Application to Drug Discovery. *Expert Systems With Applications*, *72*, 151–159.

Chen, C. H., Deng, N. Y., & Xiao, R. Y. (2004). Personal credit Evaluation based on Support Vector Machine. [基于支持向量机的个人信用评估]. *Computer Engineering and Applications, 23*, 198–199+215.

Fang, K. N., Wu, J. B., & Xie, B. C. (2010). Research on Credit Risk of Credit Card under Asymmetric Credit Information. [信贷信息不对称下的信用卡信用风险研究]. *Economic Research, 45*(S1), 97–107.

Fang, K. N., Zhang, G. J., & Zhang, H. Y. (2014). Personal Credit Risk Warning Method based on Lasso-Logistic Model. [基于 Lasso-Logistic 模型的个人信用风险预警方法]. *The Journal of*

Quantitative & Technical Economics, 2.

Frydman, H., & Kao, A. (1985). Introducing Recursive Partitioning for Financial Classification: The Case of Financial Distress. *Journal of Finance, 40*(1), 269–291.

Li, H. (2019). *Statistical learning method*. Tsinghua University Press.

Li, M. (2005). Research on the Application of Logit Model in Commercial Bank Credit Risk Assessment. [Logit 模型在商业银行信用风险评估中的应用研究]. *Management Science, 2,* 33-38.

Ma, H. H. (2021). Application of Random Forest and XGBoost Model in Personal Credit Risk Assessment. [随机森林和 XGBoost 模型在个人信用风险评估中的应用]. *Minzu University of China.*
https://kns.cnki.net/KCMS/detail/detail.aspx?dbname=CMFD202201&filename=1021112273.nh

Orgler, Y. (1970). A Credit Scoring Model for Commercial Loans. *Journal of Money Credit & Banking, 2*(04).

Satchidananda, S.S., & Simha, J.B. (2006). Comparing Decision Tress with Logistic Regression For Credit Risk Analysis. *SAS APAUGC 2006 MUMB AL,* 269–291.

Trevor, H., Tibshirani, R., & Friedman, J. (2017). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction (Second)*. Springer.

Wiginton, J. C. (1980). A Note on the Comparison of Logit and Discriminant Models of Consumer Credit Behavior. *Journal of Financial and Quantitative Analysis, 15*(3).

Wu, C., Zhang, X. D., & Tian, H. X. (2009). Research on Commercial Bank Credit Risk Assessment Model based on Fuzzy Neural Network. [基于模糊神经网络的商业银行信用风险评估模型研究]. *Management Observer, 7,* 187–190.

Xiong, Z. B. (2016). Research on the Method of Feature Selection in Credit Evaluation. [信用评估中的特征选择方法研究]. *The Journal of Quantitative & Technical Economics, 33*(01), 142–155.

Yao, X., & Yu, L. A. (2012). Fuzzy Approximate Support Vector Machine Model and Its Application in Credit Risk Assessment. [模糊近似支持向量机模型及其在信用风险评估中的应用]. *System Engineering Theory and Practice, 3,* 549–554.

Zeng, M., & Xie, J. (2019). Internet Finance Personal Credit Risk Assessment Index Selection Method. [互联网金融个人信用风险评估的指标选择方法]. *Times Finance, 33,* 6-9.

Zhang, Q. (2016). Case Study on Risk Control Mode of Small and Micro Enterprises in Internet Finance based on Industry. [基于产业的互联网金融小微企业风控模式案例研究]. *Arizona State*

University.
https://www.researchgate.net/publication/347829858_Financing_Path_Analysis_of_Small_and_Micro_Enterprises_Based_on_Internet_Finance
Zhang, W., Li, Y. S., & Wang, C. F. (2000). Application of Recursive Classification Tree in Credit Risk Analysis. [递归分类树在信用风险分析中的应用]. *System Engineering Theory and Practice*. 3, 50-55+62.
Zhou, Y. S., Cui, J. L., Zhou, L. Y., Sun, H. X., & Liu, S. Q. (2020). Research on Personal Credit Risk Assessment based on Improved Random Forest Model. [基于改进的随机森林模型的个人信用风险评估研究]. *Credit Investigation, 38*(1).

Appendices

Appendix 1: Variance Threshold

```
data = pd.read_csv('model_sample.csv')
from sklearn.feature_selection import VarianceThreshold
selector = VarianceThreshold(0)
x_var0_001 = selector.fit_transform(data)
print("before filtration：\n",data.shape)
print("after filtration：\n",x_var0_001.shape)
dataframe_data = pd.DataFrame(x_var0_001)
print(dataframe_data)
outputpath='c:/Users/admin/Desktop/after_variance_threshold.csv'
dataframe_data.to_csv(outputpath,sep=',',index=True,header=True)
```

Appendix 2: Data Standardization

```
import pandas as pd
from sklearn import preprocessing
data= pd.read_csv('Missing values filled.csv')
# z-score
data_zs = zscore.fit_transform(data)
df=pd.DataFrame(data_zs)
outputpath='c:/Users/admin/Desktop/ Standardization done.csv'
df.to_csv(outputpath,sep=',',index=True,header=True)
```

Appendix 3: Outliers Detection & Removal

#Outliers Detection

```
import seaborn as sns
import matplotlib.pyplot as plt
import random
#random.seed(1)
data = pd.read_csv('after_standardization.csv')
sns.boxplot(data = data)
def find_outliers(data):
    abnomalies = []
    random_data_std = np.std(data)
    random_data_mean = np.mean(data)
    abnomaly_cut_off = random_data_std*3
    lower_limit = random_data_mean - abnomaly_cut_off
    upper_limit = random_data_mean + abnomaly_cut_off
    print(lower_limit)
    for outlier in data:
        if outlier > upper_limit or outlier < lower_limit:
            abnomalies.append(outlier)
```

```
    return abnomalies
find_outliers(data)
```

#Outliers Removal

```
import pandas as pd
from sklearn.ensemble import IsolationForest
df2=pd.read_csv('Standardization done csv')
from sklearn.ensemble import IsolationForest
predictions = IsolationForest().fit(df2).predict(df2)
df3 = df2[predictions==1]
# outputpath='c:/Users/admin/Desktop/outliers done.csv'
df3.to_csv(outputpath,sep=',',index=True,header=True)
```

Appendix 4: Data Balancing

```
#SMOTE Algorithm
df4 = pd.read_csv('outliers done.csv')
from imblearn.combine import SMOTETomek
smote_tomek = SMOTETomek(random_state=0)
x_data = data.iloc[:,1:]
y_data = data.iloc[:,0]
x_res, y_res = smote_tomek.fit_resample(x_data, y_data)
print(x_res.groupby([y_data]).size())
outputpath='c:/Users/admin/Desktop/data_balancing_done_x.csv'
x_res.to_csv(outputpath,sep=',',index=True,header=True)
outputpath='c:/Users/admin/Desktop/data_balancing_done_y.csv'
y_res.to_csv(outputpath,sep=',',index=True,header=True)
```

Appendix 5: Classifiers

```
#Evaluation metrics, the graphs of AUC and time used
# Logistic Regression
import datetime
#Starting time
aa=datetime.datetime.now()
m=0
for i in range(1000):
    m=m+i
import pandas as pd
from sklearn import metrics
from sklearn.model_selection import KFold
import numpy as np
import matplotlib.pyplot as plt
#Calculate ROC and AUC
from sklearn.metrics import roc_curve, auc
data = pd.read_csv('Data_balancing_done.csv')
x=data.iloc[:,1:].values
y=data.iloc[:,0].values
accuracy=[]
precision=[]
recall=[]
F1=[]
AUC=[]

kf=KFold(n_splits=5,shuffle=True)
for train_index,test_index in kf.split(x):
    x_train, x_test=x[train_index],x[test_index]
    y_train,y_test=y[train_index],y[test_index]
    from sklearn.linear_model import LogisticRegression
```

```
    #The model of Logistic Regression

model=LogisticRegression(penalty='l1',C=1,class_weight={0:0.5,1:0.5},solver="sag",random_state
=42)
    model.fit(x_train, y_train)
    y_score= model.predict_proba(x_test)[:,1]


    fpr,tpr,threshold = roc_curve(y_test, y_score)
    #Calculate AUC
    roc_auc = auc(fpr,tpr)


    plt.figure()
    lw = 2
    plt.figure(figsize=(10,10))
    plt.plot(fpr, tpr, color='darkorange',  lw=lw, label='ROC curve (area = %0.2f)' % roc_auc)
    plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver operating characteristic example')
    plt.legend(loc="lower right")
    plt.show()
    #Expected output of the test sample
    expected = y_test
    # Test sample prediction
    predicted = model.predict(x_test)
    accuracy.append(metrics.accuracy_score(expected, predicted))
    precision.append(metrics.precision_score(expected,predicted))
    recall.append(metrics.recall_score(expected,predicted))
    F1.append(metrics.f1_score(expected,predicted))
```

```
    AUC.append(metrics.roc_auc_score(expected,predicted))
print(accuracy)
print(precision)
print(recall)
print(F1)
print(AUC)
print(np.mean(accuracy))
print(np.mean(precision))
print(np.mean(recall))
print(np.mean(F1))
print(np.mean(AUC))
#Ending time
bb=datetime.datetime.now()

#Running time, expressed in hours: minutes: seconds
cc=bb-aa
print("Running time is: ',cc)
```

#SVM

```
import datetime
#Starting time
aa=datetime.datetime.now()
m=0
for i in range(1000):
    m=m+i
from sklearn.svm import SVC
from sklearn import svm, datasets
from sklearn import model_selection
data = pd.read_csv('Data_balancing_done.csv')
x=data.iloc[:,1:].values
```

```
y=data.iloc[:,0].values
accuracy=[]
precision=[]
recall=[]
F1=[]
AUC=[]

kf=KFold(n_splits=5,shuffle=True)
for train_index,test_index in kf.split(x):
    x_train, x_test=x[train_index],x[test_index]
    y_train,y_test=y[train_index],y[test_index]
    # The model of SVM
    model=SVC(kernel='rbf',random_state=42)
    model.fit(x_train, y_train)
    y_score =model.fit(x_train, y_train).decision_function(x_test)

    fpr,tpr,threshold = roc_curve(y_test, y_score)
    #Calculate AUC
    roc_auc = auc(fpr,tpr)

    plt.figure()
    lw = 2
    plt.figure(figsize=(10,10))
    plt.plot(fpr, tpr, color='darkorange', lw=lw, label='ROC curve (area = %0.2f)' % roc_auc)
    plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver operating characteristic example')
```

```
    plt.legend(loc="lower right")
    plt.show()

    #Expected output of the test sample
    expected = y_test
    # Test sample prediction
    predicted =model.predict(x_test)
    accuracy.append(metrics.accuracy_score(expected, predicted))
    precision.append(metrics.precision_score(expected,predicted))
    recall.append(metrics.recall_score(expected,predicted))
    F1.append(metrics.f1_score(expected,predicted))
    AUC.append(metrics.roc_auc_score(expected,predicted))
#Ending time
bb=datetime.datetime.now()

#Running time, expressed in hours: minutes: seconds
 cc=bb-aa
print("Running time is: ',cc)
```

#Decision Tree

```
import datetime
#Starting time
aa=datetime.datetime.now()
m=0
for i in range(1000):
    m=m+i
from sklearn.tree import DecisionTreeClassifier
data = pd.read_csv('Data_balancing_done.csv')
x=data.iloc[:,1:].values
y=data.iloc[:,0].values
```

```
accuracy=[]
precision=[]
recall=[]
F1=[]
AUC=[]

kf=KFold(n_splits=5,shuffle=True)
for train_index,test_index in kf.split(x):
    x_train, x_test=x[train_index],x[test_index]
    y_train,y_test=y[train_index],y[test_index]
    #The model of Decision Tree
    model=DecisionTreeClassifier(splitter="best",class_weight={0:0.5,1:0.5},
    max_depth=99,random_state=42)
    model.fit(x_train,y_train)
    y_score= model.predict_proba(x_test)[:,1]

    fpr,tpr,threshold = metrics.roc_curve(y_test, y_score)
    #Calculate AUC
    roc_auc =metrics. auc(fpr,tpr)

    plt.figure()
    lw = 2
    plt.figure(figsize=(10,10))
    plt.plot(fpr, tpr, color='darkorange', lw=lw, label='ROC curve (area = %0.2f)' % roc_auc)
    plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver operating characteristic example')
```

```
    plt.legend(loc="lower right")
    plt.show()

    #Expected output of the test sample
    expected = y_test
    # Test sample prediction
    predicted =model.predict(x_test)
    accuracy.append(metrics.accuracy_score(expected, predicted))
    precision.append(metrics.precision_score(expected,predicted))
    recall.append(metrics.recall_score(expected,predicted))
    F1.append(metrics.f1_score(expected,predicted))
    AUC.append(metrics.roc_auc_score(expected,predicted))

#Ending time
bb=datetime.datetime.now()

#Running time, expressed in hours: minutes: seconds
cc=bb-aa
print('Running time is: ',cc)

# Random Forest
import datetime
#Starting time
aa=datetime.datetime.now()
m=0
for i in range(1000):
    m=m+i

from sklearn.ensemble import RandomForestClassifier
from sklearn import
```

```
data= pd.read_csv('Data_balancing_done.csv')
x=data.iloc[:,1:].values
y=data.iloc[:,0].values
accuracy=[]
precision=[]
recall=[]
F1=[]
AUC=[]

kf=KFold(n_splits=5,shuffle=True)
for train_index,test_index in kf.split(x):
    x_train, x_test=x[train_index],x[test_index]
    y_train,y_test=y[train_index],y[test_index]
    #The model of Random Forest
    model = RandomForestClassifier(n_estimators=160,oob_score=True,random_state=1)
RandomForestClassifier
    model.fit(x_train, y_train)
    y_score= model.predict_proba(x_test)[:,1]

    fpr,tpr,threshold = roc_curve(y_test, y_score)
    #Calculate AUC
    roc_auc = auc(fpr,tpr)

    plt.figure()
    lw = 2
    plt.figure(figsize=(10,10))
    plt.plot(fpr, tpr, color='darkorange', lw=lw, label='ROC curve (area = %0.2f)' % roc_auc)
    plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
```

```
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver operating characteristic example')
    plt.legend(loc="lower right")
    plt.show()

    #Expected output of the test sample
    expected = y_test
    # Test sample prediction
    predicted = model.predict(x_test)
    accuracy.append(metrics.accuracy_score(expected, predicted))
    precision.append(metrics.precision_score(expected,predicted))
    recall.append(metrics.recall_score(expected,predicted))
    F1.append(metrics.f1_score(expected,predicted))
    AUC.append(metrics.roc_auc_score(expected,predicted))

#Ending time
bb=datetime.datetime.now()
#Running time, expressed in hours: minutes: seconds
cc=bb-aa
print("Running time is: ',cc)
```

 # Naïve Bayesian

```
import datetime
#Starting time
aa=datetime.datetime.now()
m=0
for i in range(1000):
    m=m+i
from sklearn.naive_bayes import GaussianNB
```

```
from sklearn.metrics import confusion_matrix
data = pd.read_csv('Data_balancing_done.csv')
x=data.iloc[:,1:].values
y=data.iloc[:,0].values
accuracy=[]
precision=[]
recall=[]
F1=[]
AUC=[]

kf=KFold(n_splits=5,shuffle=True)
for train_index,test_index in kf.split(x):
    x_train, x_test=x[train_index],x[test_index]
    y_train,y_test=y[train_index],y[test_index]
    from sklearn.model_selection import StratifiedShuffleSplit
    #The model of Naïve Bayesian
    model = GaussianNB()
    model.fit(x_train, y_train)
    y_score= model.predict_proba(x_test)[:,1]

    fpr,tpr,threshold = metrics.roc_curve(y_test, y_score)
    #Calculate AUC
    roc_auc =metrics. auc(fpr,tpr)

    plt.figure()
    lw = 2
    plt.figure(figsize=(10,10))
    plt.plot(fpr, tpr, color='darkorange', lw=lw, label='ROC curve (area = %0.2f)' %  roc_auc)
    plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
    plt.xlim([0.0, 1.0])
```

```
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver operating characteristic example')
    plt.legend(loc="lower right")
    plt.show()

    #Expected output of the test sample
    expected = y_test
    # Test sample prediction
    predicted = model.predict(x_test)
    accuracy.append(metrics.accuracy_score(expected, predicted))
    precision.append(metrics.precision_score(expected,predicted))
    recall.append(metrics.recall_score(expected,predicted))
    F1.append(metrics.f1_score(expected,predicted))
    AUC.append(metrics.roc_auc_score(expected,predicted))

#Ending time
bb=datetime.datetime.now()

#Running time, expressed in hours: minutes: seconds
cc=bb-aa
print("Running time is:',cc)
```

#Confusion Matrix

```
from sklearn.metrics import confusion_matrix
data = pd.read_csv('data_balancing_done.csv')
x=data.iloc[:,1:].values
y=data.iloc[:,0].values
cm_matrix = metrics.confusion_matrix(expected, predicted)
```

```
cm_matrix
class_names=[0,1] # name  of classes
fig, ax = plt.subplots()
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)

sns.heatmap(pd.DataFrame(cm_matrix), annot=True, cmap="YlGnBu" ,fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
plt.show()
```

Appendix 6: Feature Selection

```
#Lasso regression
library(glmnet)
library(tidyverse)
data <- read_csv("after_smote.csv")
set.seed(2022)
x <- as.matrix(data[,2:199])
y = as.matrix(data[, 1])
alpha1_fit <- glmnet(x,y,alpha=0.1,family="binomial")
plot(alpha1_fit,xvar="lambda",label=TRUE)
alpha1.fit <- cv.glmnet(x,y,type.measure = "class",alpha=0.1,family="binomial")
plot(alpha1.fit)
print(alpha1.fit)
coef(alpha1_fit,s=alpha1.fit$lambda.1se)
```

```
data1 <- read_csv("after_smote.csv")
x1 <- as.matrix(data1[,2:199])
y1 = as.matrix(data1[, 1])
alpha1_fit1 <- glmnet(x1,y1,alpha=1,family="binomial")
plot(alpha1_fit1,xvar="lambda",label=TRUE)
alpha1.fit1 <- cv.glmnet(x1,y1,ype.measure = "class",alpha=1,family="binomial")
plot(alpha1.fit1)
print(alpha1.fit1)
coef(alpha1_fit1,s=alpha1.fit1$lambda.1se)

data2 <- read_csv("after_smote.csv")
set.seed(2022)
x1 <- as.matrix(data2[,2:199])
y1 = as.matrix(data2[, 1])
alpha1_fit1 <- glmnet(x1,y1,alpha=0.01,family="binomial",set.seed(1234))
plot(alpha1_fit1,xvar="lambda",label=TRUE)
alpha1.fit1 <- cv.glmnet(x1,y1,ype.measure = "class",alpha=0.01,family="binomial")
plot(alpha1.fit1)
print(alpha1.fit1)
coef(alpha1_fit1,s=alpha1.fit1$lambda.1se)
```

#Random Forest

```
from sklearn.ensemble import RandomForestRegressor
data1 = pd.read_csv('Lasso1',header=None)
data=data1.iloc[1:]
X=data.iloc[:,1:]
Y=data.iloc[:,0]
names = data1.iloc[0]
```

```
rf = RandomForestRegressor(n_estimators=150,random_state=10)

rf.fit(X, Y)

print( "Features sorted by their score:")

print(sorted(zip(map(lambda x: round(x, 4), rf.feature_importances_), names),reverse=True)
```