Quantitative Finance and Economics

QFE, 6(2): 303–325.
DOI: 10.3934/QFE.2022013
Received: 16 March 2022
Revised: 26 May 2022
Accepted: 30 May 2022
Published: 09 June 2022

http://www.aimspress.com/journal/QFE

Research article

Machine learning and artificial neural networks to construct P2P lending credit-scoring model: A case using Lending Club data

An-Hsing Chang[1], Li-Kai Yang[2], Rua-Huan Tsaih[3] and Shih-Kuei Lin[1,*]

[1] Department of Money and Banking, National Chengchi University, Taiwan
[2] Department of Investment and Trading, Cathay Life Insurance Co., Ltd., Taiwan
[3] Department of Management Information Systems, National Chengchi University, Taiwan

* **Correspondence:** Email: square@nccu.edu.tw.

Abstract: In this study, we constructed the credit-scoring model of P2P loans by using several machine learning and artificial neural network (ANN) methods, including logistic regression (LR), a support vector machine, a decision tree, random forest, XGBoost, LightGBM and 2-layer neural networks. This study explores several hyperparameter settings for each method by performing a grid search and cross-validation to get the most suitable credit-scoring model in terms of training time and test performance. In this study, we get and clean the open P2P loan data from Lending Club with feature engineering concepts. In order to find significant default factors, we used an XGBoost method to pre-train all data and get the feature importance. The 16 selected features can provide economic implications for research about default prediction in P2P loans. Besides, the empirical result shows that gradient-boosting decision tree methods, including XGBoost and LightGBM, outperform ANN and LR methods, which are commonly used for traditional credit scoring. Among all of the methods, XGBoost performed the best.

Keywords: P2P lending; credit score; machine learning; artificial neural networks; feature engineering; Lending Club

JEL Codes: D12, E41, E44, G20

1. Introduction

As a financial innovation with FinTech, P2P (Peer-to-peer) lending provides direct lending between borrowers and lenders through an online platform, instead of traditional financial intermediaries such as banks (Samitsu, 2017). The earliest P2P lending platform, Zopa, was established in 2005 in UK. After Zopa development, many P2P lending platforms were established one after another, among which are more than 1,000 in China and the Lending Club in the United States of America, i.e., the most influential P2P lending platform that was listed on the New York Stock Exchange in 2014. According to Verified Market Research (2021), the P2P lending market was valued at USD 84.89 billion in 2020, and it is projected to reach USD 578.03 billion by 2028.

One of the reasons why the P2P lending platform can break through the traditional lending market and attract many investors is its great ability to meet the capital demand (Mills and McCarthy, 2016). Since the 1980s, there has been a wave of consolidation of financial institutions around the world. Many small banks have been acquired by large banks to make the increasing concentration of bank assets and loans. However, the structure of large banks makes it difficult to assess the risk of small and medium enterprises (SMEs) or personal loans, resulting in an information asymmetry issue. Furthermore, because of the high transaction costs associated with microloans, the process of underwriting the loans of SMEs and individual consumers is inherently inefficient. These two issues make large banks less willing to provide microloans, even if qualified lenders are willing to pay high interest rates. This causes a credit rationing problem. The emergence of P2P lending platforms for SMEs and individual loans has alleviated these problems and successfully filled the gap between the demand and supply of microloans, not only to meet the funding demand of those who have difficultly obtaining loans from traditional financial intermediaries, but also to provide more choices to investors who seek high returns.

However, P2P lending is quite risky for investors since there is credit risk, liquidity risk and even platform-collapse risk. The most important one is the credit risk of borrowers. Since many loans come without collateral, it is difficult for the platform to freeze the other assets of borrowers when it defaults, which leads to low loan recovery rates and high recovery costs. According to existing literature, traditionally, a credit risk can be defined as the uncertainty of a borrower's trustworthiness, including their willingness and ability to repay (Crouhy et al., 2014), and the credit score is correlated with measures of impulsivity, time preference and trustworthiness (Arya et al., 2013). Due to the differences between the traditional loans and P2P lending, as mentioned above, it is not suitable enough to calculate the credit score for P2P lending in traditional ways.

Therefore, this study explores several credit-scoring models to predict whether the loan will default, from a general perspective, by using the prior information of the loan and the borrower and excluding the repayment willingness factors, such as the loan amount and loan length, as much as possible. Compared with traditional approaches, machine learning (ML) methods can better capture the prior information of the loan and the borrower. Following Bolton (2010), Baesens et al. (2003) and Cao et al. (2018), we constructed the models by using several ML and artificial neural network (ANN) methods, including logistic regression (LR), a support vector machine (SVM), a decision tree (DT), random forest (RF), eXtreme gradient boosting (XGBoost), LightGBM and 2-layer neural networks.

Our research is related to existing literature in two aspects. First of all, due to there being information asymmetry and no collateral issues, P2P platforms try to solve these problems by improving the information transparency. By taking the Lending Club as an example, it publishes the complete information of each de-identified loan on its website, including the loan purpose, personal information, credit history,

debt status and many others, which can be used by investors to identify potential defaulters or calculate the risk-return ratio. Therefore, figuring out the significant default factors was an important purpose of this study. When it comes to traditional financial intermediaries, Baesens et al. (2004) stated that the loan purpose is the most important feature affecting the probability of default, and they divided the loans into three levels: high, medium and low risks, according to the loan purpose. They also considered the housing loans and locomotive loans as high-risk purposes, while student loans and vacation loans were low-risk purposes. Rajan et al. (2015) argued that the personal information of borrowers, such as their FICO credit score, income and debt status, were often underestimated by traditional credit-scoring models, which was one issue that led to the subprime mortgage crisis in 2008. They also found that the credit history is another important default factor. Besides, scoring from the perspective of behavior, Thomas (2000) found that incorporating credit history into scoring models can improve the ability of prediction. For small business loans, Mester (1997) found that the credit history of business owners has better prediction capability than the variables in the financial statement. On the other hand, when it comes to the P2P lending platforms, Iyer et al. (2009) found that the credit score, borrower's current and total defaulted loans, debt-to-income ratio and loan amount are the most important default factors. Everett (2015) considered the credit score, age of borrower, house ownership, guarantor and loan amount as important factors. In addition, by analyzing the Lending Club data using LR and a survival function, Serrano-Cinca et al. (2015) found that the loan interest rate, loan purpose, credit history and borrower's personal information are the most significant, while debt amount and years of work are not significant. Above all, the default factors can be divided into four categories: personal information (such as income, debt and credit score), credit history (such as the amount of defaults), loan purpose and loan information (such as interest rate and loan amount). Since we focus on credit scoring and prediction with prior information, we have paid more attention to the default factors from the first three categories in this study, which contributes to improving the efficiency of the credit-scoring process. In order to get the required training data, we preprocessed the Lending Club data by cleaning, converting and scaling according to the concept of feature engineering, and we trained the XGBoost model to filter out the important default factors.

Second, our research also contributes to the credit-scoring models. Traditionally, we have adopted statistical methods like LR to solve to the credit-scoring problem. Ohlson (1980) adopted a conditional logit model to predict corporate failure with variables from financial statements. Bolton (2010) pointed out that LR is the most popular credit-scoring model in practice. With the exception of the missing values or collinearity, it hardly needs any assumptions on the data type. Since the results of this kind of model are easy to interpret, LR methods are widely accepted in the financial industry. With the development of artificial intelligence and data mining, more and more studies have focused on the construction of credit-scoring models by using ML methods. Using 831 financial institutions' credit score data from Moody's, Baesens et al. (2003) compared the model performance of various classification algorithms, including the SVM and multilayer perceptron (MLP), and several statistical methods, including LR, ordinary least-square regression and linear discriminant analysis (LDA). The results showed that the SVM outperformed the others in terms of accuracy. Byanjankar et al. (2015) proposed a credit-scoring model by using ANNs to classify P2P loans into default and non-default groups; it performed effectively in screening default applications. Cao et al. (2018) compared the performance of eight classification methods, including LDA, LR, DT, SVM, RF, gradient-boosting decision tree (GBDT), MLP and XGBoost methods, with datasets from Kaggle's "Give Me Some Credit" and "PPDai" in China. They used accuracy, area under the curve of ROC and logistic loss for analysis; they found that the XGBoost model had better performance. Chen et al. (2019) proposed a multi-stage ensemble learning model to evaluate the borrowers' credit; they used a

GBDT to map the features and a special auto-encoder to extract the best features. Duan (2019) proposed a deep neural network-based decision-making approach to assess the risks of P2P lending. Li et al. (2021) used RF, LR, k-nearest neighbor and MLP models to predict the default borrowers; they provided an overall understanding of different models and their prediction power. On the other hand, Serrano-Cinca and Gutiérrez-Nieto (2016) used profit scoring as an alternative approach to credit-scoring systems for analyzing P2P lending issues. However, most of the past studies did not try to analyze various hyperparameter combinations in the ML models to optimize the results. In this study, we constructed a credit-scoring model of P2P loans by using several ML, ANN and GBDT methods, including XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017). We explored several hyperparameter settings for each method by performing a grid search and cross-validation to get the most suitable credit-scoring model in terms of training time and test performance.

Our empirical results show that, when comparing the accuracy and F1 score, the ensemble learning algorithms have good performance; meanwhile, LR and 2-layer neural networks are less suitable for credit scoring. When comparing the precision, LR and GBDT methods outperform other methods based on DTs. When comparing the recall, the ML algorithms and 2-layer neural networks have better performance than LR. Above all, GBDT methods, including XGBoost and LightGBM, are recommended for credit-scoring models. And, XGBoost demonstrated the best performance, with an accuracy of around 88%.

The contributions of this paper can be summarized into three aspects. First of all, we compared different hyperparameter sets for ML algorithms and 2-layer neural networks to optimize the model performance. As far as we know, past studies did not have a comprehensive discussion on this issue for credit-scoring models. Second, in this study, we focus more on using prior information, such as the personal information and credit history, to predict the default, which can provide the P2P platform a costless approach to check the loans and improve the efficiency of the process. Furthermore, we have proven that GBDT methods, especially XGBoost, outperforms not only the traditional credit-scoring approach, i.e., LR, but also the popular ANN algorithm.

The rest of this paper is organized as follows. In Section 2, we introduce the ML and ANN algorithms we used to construct the credit-scoring model. Then, in Section 3, we discuss the data from Lending Club and the data preprocessing, including the cleaning, converting and scaling. After that, we present the empirical results in Section 4. Finally, we draw the conclusion in Section 5.

2. Methods

In this section, we briefly introduce seven algorithms we used to construct the credit-scoring model, including the LR, SVM, DT, RF, XGBoost, LightGBM and 2-layer neural network algorithms. Besides, we introduce four performance measures for empirical analysis.

2.1. Logistic regression

The dependent variable in our research is a discrete variable with only two values: 1 (default) or 0 (non-default). Therefore, linear regression is not suitable for this situation. Traditionally, in statistics, we solve this kind of problem by using probit or logit models, which assume that the probability of event occurrence obeys a certain probability distribution. If the probability of event occurrence obeys the cumulative standard normal distribution, we usually use a probit model. If the probability of event

occurrence obeys the cumulative logistics distribution, we use a logit model, also known as LR (Aldrich and Nelson, 1984).

In order to predict the occurrence probability of the event, we assume the probability of default to be a linear combination of the independent variables (the features):

$$P(y = 1|x_1, \dots, x_k) = T(\beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + \varepsilon) \tag{1}$$

where $y(y \in [0,1])$ is the dependent variable, $x_1, \dots, x_k$ are the independent variables, ε is the error term and the function T is the logistic cumulative distribution function that maps $x_1, \dots, x_k$ to the real number space from 0 to 1, ensuring that the estimated value falls between [0,1], as follows:

$$T(x) = \frac{e^x}{1+e^x} \tag{2}$$

Then, we define $\pi(x) = P(y = 1|x)$ and we have

$$\pi(x) = \frac{e^{\beta_0+\beta_1 x_1+\cdots+\beta_k x_k}}{1+e^{\beta_0+\beta_1 x_1+\cdots+\beta_k x_k}} \tag{3}$$

or

$$ln\left(\frac{\pi(x)}{1-\pi(x)}\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k \tag{4}$$

We estimate the coefficients by maximizing the log-likelihood function, which is

$$logL(\beta) = \sum_{i=1}^{N} y_i log\big(F(x'_i\beta)\big) + \sum_{i=1}^{N}(1 - y_i) log\big(1 - F(x'_i\beta)\big) \tag{5}$$

where $F(x'_i\beta) = P(y = 1|x_i; \beta)$.

2.2. Support vector machine

The SVM was developed by Cortes and Vapnik (1995) from the structured risk minimum error method in statistical learning theory (Lin and Lin, 2003). This method is suitable for processing classification and prediction problems by using a hyperplane to distinguish between two or more different types of data. SVMs are widely used in many fields, such as 3D image recognition (Pontil and Verri, 1998) and biotechnology (Brown et al., 1999).

When dealing with the financial issues, since the data usually has a nonlinear structure, we need to use the kernel function to project the data into a higher-dimensional feature space and then distinguish the categories by performing linear cutting. There are four types of kernel functions that are mainly used, i.e., the linear kernel function, polynomial kernel function, radial basis kernel function (RBF) and sigmoid kernel function, which are as summarized in Table 1. For the kernel functions in Table 1, x_i and x_j are vectors of feature variables, γ and c_0 are real constants and d is an integer constant.

Table 1. SVM kernel functions.

Kernel function	Mathematical expression
RBF	$\Phi(x_i, x_j) = exp\left(-\gamma\|x_i - x_j\|^2\right), \gamma > 0$
Polynomial	$\Phi(x_i, x_j) = (\gamma x\|\ \ \|i\|\ \ \|Tx_j + c_0)^d, \gamma > 0$
Linear	$\Phi(x_i, x_j) = x_i{}^T x_j$
Sigmoid	$\Phi(x_i, x_j) = tanh\,(\gamma x\|\ \ \|i\|\ \ \|Tx_j + c_0)$

Generally speaking, the RBF is the most widely used. When there is not enough information to select the kernel function, the RBF is the most suitable one that can classify nonlinear and high-dimensional data with only two parameters: the penalty coefficient C and the parameter γ (Hsu et al., 2003). In this study, we performed a grid search for all four kernel functions, their hyperparameters and penalty coefficients to find the most suitable combination of kernel functions and hyperparameters for the P2P credit-scoring model.

2.3. Decision tree

The DT was developed by Breiman (1996); it distinguishes the data with labels (which can be categories or real numbers) according to a sequence of questions. It generates branches according to certain cutting criteria, and each node represents a category, value or decision. During this decision-making process, the amount of information will gradually increase. When the limit condition is reached, the amount of information cannot continue to increase, or it reaches a certain amount. At that moment, the tree will stop growing and produce the final result. DTs can be used as a classification tree, which is suitable for predicting the labels of discrete types, such as credit ratings. Because the model is simple and easy to understand and to implement, it is not too sensitive to deal with outliers in the data. Furthermore, since the calculation process is efficient, DTs are often applied for commercial issues (Singh and Gupta, 2014).

There are three main steps to setup the model, including defining the impurity, the branch criteria and the stopping rules. First of all, the impurity measures the disorder or uncertainty in the data. We usually use entropy (Shannon, 1948) or Gini impurity (Breiman et al., 1984) as the measure. Second, we need to build up the branches to reduce the impurity of the data in the original nodes. In detail, we traverse the features to find the best branch points in order to obtain the most additional information after the feature branch. The common criteria are information gain (Quinlan, 1987), Gini gain (Breiman et al., 1984) and gain ratio (Quinlan, 1992). Last but not least, the DT will continue to grow until the triggering of the stop rules. According to the algorithm mentioned above, the common stopping rules include the following: (1) the labels of all training sets are the same, (2) the predefined maximum growth depth is reached and (3) the sample number of all children is less than the predefined number. In this study, we tried to find the optimized combination of all hyperparameters for all steps in the DT.

2.4. Random forest

The RF is an ensemble learning method that was first proposed by Ho (1995) and developed to have a bagging (bootstrap aggregation) method (Breiman, 1996) and random subspace method (Ho, 1998). This algorithm can help solve the problem of DTs being easier to become a highly irregular pattern when they grow deeper. Regarding that problem, the variance is high even if the deviation is low, resulting in overfitting. Therefore, according to the law of large numbers, RFs construct a number of slightly different DTs (sample sampling, feature sampling) to reduce the effect of variance.

Generally speaking, with more DTs, the prediction results can be more accurate, which means better model stability. However, it also means a longer training period. Therefore, the balance between training time and model stability is what we consider in the evaluation of the model performance.

2.5. XGBoost

XGBoost, proposed by Chen and Guestrin (2016), is an extension of the GBDT that was proposed by Friedman (2001). Similar to the RF, the GBDT is an ensemble learning algorithm, but it learns incrementally. In this method, each new tree is born based on the foundation of the previous one. This means that the original model is kept unchanged every time, but a new function is added to make up for the shortcomings of the previous one.

Instead of the greedy algorithm used in GBDTs, XGBoost models use a more efficient approximation algorithm (also known as the histogram algorithm), which not only improves the computational efficiency, but also reduces the overfitting problem. In addition, we can realize parallel computing, decentralized computing and out-of-core computing in this system, which helps reduce training time and process large data efficiently. Since this study focuses more on the empirical analysis and construction of a credit-scoring model, please refer to Chen and Guestrin (2016) for more details.

2.6. LightGBM

According to Ke et al. (2017), LightGBM is a distributed gradient-boosting framework for ML algorithms originally developed by Microsoft. Similar to XGBoost, LightGBM is also an extension of the GBDT algorithm. Ke et al. (2017) proposed two methods, i.e., gradient-based one-side sampling (GOSS) and exclusive feature bundling (EFB), to reduce the amount of data and features without affecting the prediction capability of the model, which can help reach a balance between efficiency and accuracy.

GOSS is an algorithm that optimizes reducing the amount of data and maintaining accuracy. According to the method, samples with large gradients will have a significant impact on gain criteria, which means that the gradient can be used as an indicator of sample weights. The GOSS method keeps all samples with large gradients and removes some samples with small gradients by randomly sampling. After that, it weights the sampled small gradient data when calculating the gain. Therefore, the algorithm will focus more on the cases without sufficient training and will not change the distribution of the original data too much.

EFB is another algorithm that optimizes reducing the number of features and maintaining accuracy. Since the feature space of high dimensions is quite sparse and many features are almost mutually exclusive each other[1], the EFB method effectively converts many mutually exclusive features into features with higher densities, which effectively avoids the calculation of unnecessary zero-valued features. In this study, we used both algorithms to find the optimized combination of hyperparameters. For more details about LightGBM, please refer to Ke et al. (2017).

2.7. Neural network

The ANN, a nonlinear model proposed by McCulloch and Pitts (1943), imitates the structure and function of the biological neural network and constructs a model based on mathematical logic. After that, Rosenblatt (1958) proposed the concept of a perceptron, using simple addition and subtraction to realize a learning network. As a combination of the reverse transfer algorithm that was proposed by

[1]For example, after one-hot encoding, the features will not be non-zero at the same time.

Rumelhart, Hinton and Williams (1986), the MLP method overcomes the disadvantage of the perceptron being unable to identify linearly inseparable data (Cybenko, 1989).

The ANN used in this study is a 2-layer neural network, which is one of the most commonly used models in ANN applications. It has high fault tolerance and can produce good results based on incomplete information (Mijwel, 2018). The ANN algorithm is a layered structure, like the neural organization of the human brain, which has three types of layers: an input layer, a hidden layer and an output layer (Mountcastle, 1957). The input layer (bottom layer) receives and processes the data. The processed information will flow to the hidden layer, which is used to express the interaction between the input and output neurons. The output layer is used to represent the answer calculated by the ANN for the given data. The neurons in the output layer correspond to different answers to the question of default status in this research.

In this study, the default numbers of neurons in the hidden layer were set to 128, 512, 128 and 16, in sequence; we also tried to adjust the number of hidden layers with 512 neurons in order to optimize the hidden layer structure of the P2P lending credit-scoring model. Besides, in order to avoid the overfitting problem, we incorporated dropout into the hidden layers according to Srivastava et al. (2014). The initial dropout probability was set to 0.5, and it is optimized during processing. Furthermore, since we use default status as an output label, which is a one-dimensional vector, the number of output neurons is two. We have used a Softmax function to transform the output into a real number between [0,1]. The Softmax function is

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \tag{6}$$

where z is a vector of K real numbers, $z = (z_1, \dots, z_K) \in R^K$ and z_j is the jth element in z for $j = 1, \dots, k$. After the transformation, the sum of all components equals 1, which achieves normalization (Bishop, 2006). Therefore, each component value can represent the probability of the answer; finally, the two component values are compared to determine whether the loan is prone to default.

Specifically, we need to use the activation function to identify the nonlinear relationship among the data in the ANN. Empirically, the most common activation functions are the sigmoid function, tanh function, rectified linear unit (ReLU) and leaky ReLU. According to Hochreiter et al. (2001), Glorot and Bengio (2010), Krizhevsky et al. (2012), Maas et al. (2013), He et al. (2015) and Lu et al. (2019), we decided to use Xavier weight initialization for the sigmoid and tanh functions, and He weight initialization for the ReLU and leaky ReLU.

For the learning rules, the loss function we decide to use is cross-entropy. Referring to the concept of entropy, if the distribution of the predicted label is closer to the one of the real labels, the cross-entropy will be smaller, and vice versa. In addition, in order to control the situation of overfitting, we have added the L2 regularization term to penalize excessive weight values. Under the binary output, the loss function is as follows:

$$L = \frac{-1}{n} \sum_x \tag{7}$$

where n is the data size, y is the real label, a is the label predicted by the 2-layer neural networks, λ is the hyperparameter that controls the intensity of regularization and W is the weight.

For the optimizers, we compare five common types, including mini-batch gradient descent, momentum (Qian, 1999), Adagrad (Duchi et al., 2011), RMSprop (Tieleman and Hinton, 2012) and

Adam (Kingma and Ba, 2015). Please refer to Ruder (2017) for more details. Besides, we use the batch normalization proposed by Ioffe and Szegedy (2015) to improve the stability of the ANN.

2.8. Model performance measures

In ML, a confusion matrix is a good classifier evaluation tool. Each column of the matrix represents the predicted label of a class, and each row represents the real label of a class. Table 2 shows the confusion matrix architecture for binary classification.

Table 2. Confusion matrix.

		True class	
		Positive	Negative
Predicted class	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

There are four model performance measures, including accuracy, recall, precision and the F1 score.

2.8.1. Accuracy

Accuracy is one metric for evaluating classification models. Informally, accuracy is the fraction of predictions our model got right. The higher the accuracy, the better the model performance. The formula is

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{8}$$

2.8.2. Recall

Recall, also known as the sensitivity or true positive rate, indicates how much of all the data actually labeled as "positive" has been predicted correctly. The higher the ratio, the better the model established. If the model misjudges the credit status of a borrower that is actually "positive", when it is marked as "negative", it means that the model is considered to be "positive". The borrower tends to default and cannot obtain a loan, and the platform or investor loses profit opportunities. The higher the recall rate, the less likely this misjudgment will occur. The formula of recall is

$$Recall = TP/(TP + FN) \tag{9}$$

2.8.3. Precision

Precision, also known as the positive predictive value, indicates how much of the results predicted by the classifier to be labeled "positive" is actually labeled as "positive". The higher the ratio, the better the model established. If the model misjudges the credit status of a borrower who is actually "negative" and marks it as "positive", when the borrower is inclined not to default, the platform or investors will suffer losses by lending to the borrower. The higher the precision, the less likely this misjudgment will occur. Compared with the misjudgment of recall, the situation will be more serious

for precision. Therefore, the credit-scoring model will have relatively strict requirements on precision. The formula of precision is

$$Precision = TP/(TP + FP) \tag{10}$$

2.8.4. F1 score

The F1 score is the harmonic mean of the recall and precision; it comprehensively reflects that the true label is predicted as "positive". The formula is

$$F1 = 2/\left(\frac{1}{Precision} + \frac{1}{Recall}\right) \tag{11}$$

3. Data

In this section, we introduce the data source and data cleaning process, which includes removing redundant features, converting features, dealing with missing data and scaling, undersampling and feature selection.

3.1. Data source

The data we use are the open P2P loan data provided by the Lending Club. This dataset contains all of the information collected by the platform during its loan process. The main features include the borrower's personal information, loan purpose, credit history, debt status and others. Since the loans after 2015 had not yet fully expired when the data were collected in 2018, it was impossible to judge whether the loan was in default. Besides, there were too many missing values in the data before 2013. Therefore, we used three-year loan data for a two-year period, i.e., from January 2013 to December 2014, corresponding to a total of 282,763 loans with 151 features each.

Furthermore, since the loans we used expired in 2017, we also selected the data period by considering the history of Lending Club from the following two points. On the one hand, during 2016 to 2017, Lending Club suffered due to many scandals. For example, the Securities and Exchange Commission investigated Lending Club's disclosures to investors after Laplanche resigned on May 9, 2016. Hence, we excluded the loans issued during the scandal period from 2016 to 2017. On the other hand, by December 2019, Lending Club had switched its target to institutional investors focusing on delivering representative samples of loans instead of individual loans. Since December 31, 2020, Lending Club has no longer operated as a P2P lender. Hence, we selected data before 2018 to avoid the impact from the business strategy changes by Lending Club.

We have used "loan_status" as the reference label for default, where "fully Paid" means no default; "Late", "Default", and other values are regarded as defaults. There were 245,332 non-default loans, accounting for approximately 86.8%, and 37,431 default loans, accounting for approximately 13.2%, as shown as in Figure 1.

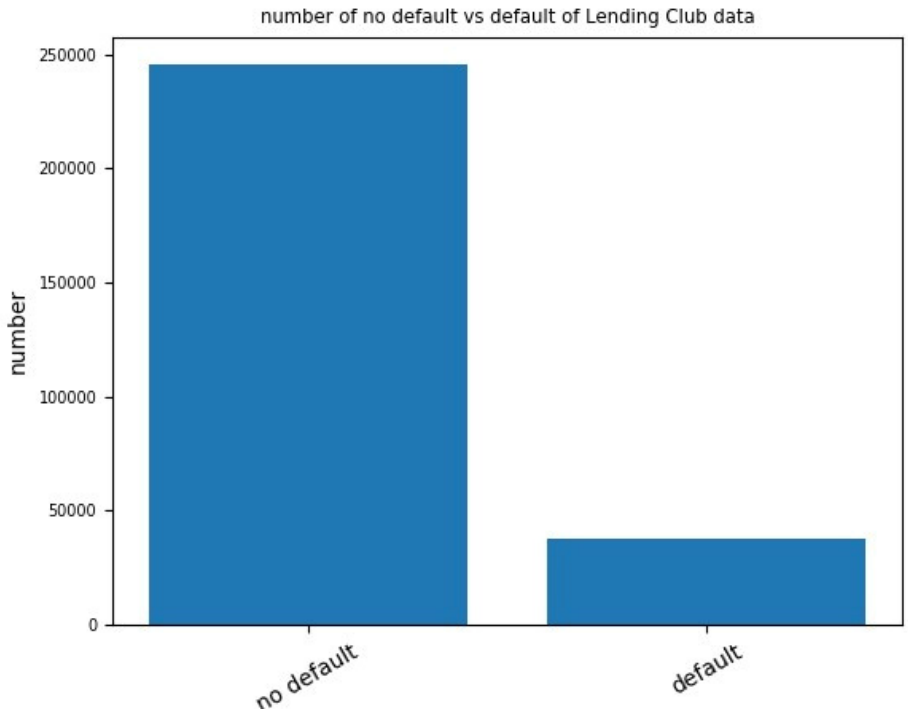

Figure 1. Statistics of the data.

3.2. Data cleaning

In order to make it easy to extract the characteristics and train the models, we cleaned the data by applying the concept of feature engineering. There are five steps, including removing redundant features, converting features, dealing with missing data and scaling, undersampling and feature selection.

3.2.1. Removing redundant features

First of all, we removed the features that are not related to loan information, such as the borrower's membership ID in Lending Club, the descriptive features (which are not easy to analyze), such as a paragraph that describes the reason for the loan, the features that are too singular (in which more than 99% data are indifferent), such as the application categories (most of which are personal loans), the features that were obtained after the loan had been approved, such as the date of the previous loan repayment, the credit features marked by Lending Club and the features with too many missing values (in which more than 99% data are missing).

3.2.2. Converting features

Since most of the data were in the form of categories, which is not suitable for model training the data needed to be converted into numerical forms.

The first one is the default reference label "loan_status". We set the "non-default" category to 0, and the "default category" to 1. The second one is "emp_length", which represents the number of working years. According to Alexander and Clifford (1996), since this is a feature with sequential meaning, we used ordinal encoding to convert it, marking this feature as orderly numbers. We set those

with longer than 10 years as 10, those with shorter than 1 year as 0 and those between 1 to 10 years as their numerical values. The third one, "revol_util", represents the revolving credit utilization rate; it is in the percentage form. We converted it into the decimal form. The fourth one, "earliest_cr_line", represents the earliest credit history; it is in month-year format. We converted it into years from September 2018 in decimal form. For example, "Sep-98" is September 1998, which was converted into 20.0 (20.0 years from September 2018). For the rest of the category features, including the loan purpose, housing ownership and others, since there is no sequential relationship, we used one-hot encoding to convert them, as described by Lantz (2013). Each one was set as an independent binary feature with only two categories of 0 or 1.

3.2.3. Dealing with missing data and scaling

Since there were missing values in the data, we needed to deal with this issue before processing the model training data. Some features were related to historical records, such as "mths_since_last_deling", which represents the number of months away from the previous default. Because the missing value ("N/A") indicates that there is no default record in the past, we could not just ignore this information. Following Kang (2013), a reasonable approach to make up the value was to set another feature that indicates whether it is a missing value in the original feature; the original missing value was filled in with values that do not appear in normal situations, such as "−1". For other features that did not have the serious situation of a missing value, we directly filled in the average of the feature, which is a reasonable approach under the assumption that the data are normal-distributed.

In addition, when using SVMs, LR or other algorithms that use the mean square error as the loss function, the scale of the features can easily affect the prediction performance because the model tends be sensitive to features with large scales. Therefore, we followed Patro and Sahu (2015) to standardize the data before training the model in order to make sure that each feature will only affect the prediction result proportionally.

3.2.4. Undersampling

Since the number of non-default loans was about 6.58 times the number of default loans, there was imbalanced classification in our dataset. Given this situation, the model will overtrain the category with more samples and consider the category with fewer samples as noise, or even ignore it, making it easy to generate bias and yield poor prediction performance. Besides, it is easy to produce a fake high accuracy, which reduces the reliability. The computational methods used by the majority of models are often biased toward the majority sample and treat the minority sample as noise, thus yielding bad prediction results (Madasamy and Ramaswami, 2017).

The simple and most popular approaches to deal with the imbalanced classification are oversampling and undersampling. Oversampling copies the minority sample, while undersampling selects part of the majority sample to reach the balance (Shelke et al., 2017). According to Elrahman and Abraham (2013), oversampling can cause overfitting and undersampling can eliminate important information about the overall pattern of the data. Therefore, there is no conclusion regarding which of the methods is better or worse.

In this study, in order to explore a more efficient approach, we adopted undersampling to greatly reduce the training time of the SVM, ANN and other ML algorithms. We randomly sampled the same

amount of data for default loans and non-default loans, which was 37,431 loans. For the default loans, we only used the totally defaulted loans, which have been labeled as "Default". Therefore, the final data size for training and testing was 74,862. Since the number of default and non-default loans is the same, if the predicted accuracy rate of the credit-scoring model is greater than 50%, the model is effective in predicting whether the borrower is inclined to default.

3.2.5. Selecting features

After the processes above, there were 144 features for each sample. According to Keogh and Mueen (2017), too many features will make the multidimensional feature space too sparse, which will easily make the distance function meaningless. This situation can affect the predictability of distance-related algorithms, such as SVMs or 2-layer neural networks, and waste computing resources. Therefore, in this study, we screened out several important features to avoid this kind of dimensional disaster.

Traditionally, a feature extraction method is used to reduce the dimensions of data. With this type of method, e.g., principal component analysis, the new feature is the projection of the original one. However, according to Wang et al. (2014), a feature extraction method will remove the original features, and the projected features may not have the empirical meanings, which means this approach is not suitable for business applications. Therefore, in this study, we followed Guyon and ElNoeeff (2003) to use a feature selection method. With this method, the new features belong to a subset of the original data features set, and only those without explanatory power are removed.

We used the XGBoost algorithm to select the features since Cao et al. (2018) proved that this algorithm can have better predicting power than RFs (Genuer et al., 2010) and other ML algorithms. After excluding the label "loan_status", we used the XGBoost method to pre-train all of the data and obtain the feature importance (in the form of percentage) according to the number of times each feature is branched in the nodes. The results are summarized in Table 3. We can find that the most important features, such as the credit score FICO ("last_fico_range_high") and debt-to-income ratio ("dti"), are consistent with past studies and business intuition, which means that the XGBoost algorithm is suitable for feature selection.

Table 3. Results of feature selection (top 10 importance).

Feature	Feature importance	Description
last_fico_range_high	0.0491525	The last upper boundary of range the borrower's FICO belongs to pulled.
dti	0.0393462	A ratio calculated using the borrower's total monthly debt payments to the total debt obligations, excluding the mortgage and requested LC loan, divided by the borrower's self-reported monthly income
mo_sin_old_il_acct	0.0365617	Months since oldest bank installment account opened
annual_inc	0.0346247	The self-reported annual income provided by the borrower during registration
revol_util	0.0329298	Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit
avg_cur_bal	0.0325666	Average current balance of all accounts
total_bc_limit	0.0305085	Total bankcard high credit/credit limit
mo_sin_old_rev_tl_op	0.0305085	Months since oldest revolving account opened
revol_bal	0.0302663	Total credit revolving balance
earliest_cr_line	0.0288136	Month the borrower's earliest reported credit line opened

After obtaining the feature importance, we screened out the features with the largest feature importance and obtained five feature subsets according to the order of feature importance. We randomly picked 80% of the data as the training set, and the remaining 20% were the test set. There were 50.01% default loans in the training set, and 49.96% in testing set. We trained the data using seven models, including LR, SVM, DT, RF, XGBoost, LightGBM and 2-layer neural network models. For the ML methods, such as LR and SVM, we used the preset hyperparameters of SKlearn. For the 2-layer neural networks, we followed the process in Section 2. After training, we predicted whether the borrower was prone to default and calculated the accuracy. Besides, we also needed to observe whether the feature setfalls into dimensional disaster. If not, we figured out the feature set with significant explanatory power for further analysis.

According to the order of feature importance, we obtained the following five feature sets:

- Feature set I: select all features, 144 in total;
- Feature set II: select features with importance that is larger than 1%, 35 in total;
- Feature set III: select top 80% features, 33 in total;
- Feature set IV: select top 70% features, 25 in total;
- Feature set V: for features in Feature set IV, we only keep those pairs with a correlation coefficient larger than 0.7.

The accuracy results for each feature set under different models are summarized in Table 4. We can find that there were no great differences between different feature sets, and that the performance of the DT was worse than that of the other models. Besides, when comparing the performance of different feature sets for the same model, we found that the accuracy of LR decreased from Feature set I to V, which means that the LR model was worse at predicting with fewer independent variables.

Table 4. Accuracy results for the feature sets for the different models.

Feature set	I	II	III	IV	V
LR	86.62%	86.50%	86.46%	86.63%	86.36%
SVM	86.26%	86.64%	86.64%	86.72%	86.78%
DT	80.63%	80.04%	80.05%	79.91%	80.16%
RF	86.56%	86.32%	86.26%	86.54%	86.55%
XGBoost	87.08%	86.78%	86.92%	86.90%	86.85%
LightGBM	87.18%	86.96%	87.04%	87.02%	86.86%
2-layer neural networks	87.10%	87.02%	87.08%	87.17%	87.00%

However, for those models that are prone to a dimensional disaster, such as the SVM and 2-layer neural network models, the situation was different. The accuracy of the SVM increased from 86.26% for Feature set I to 86.78% for Feature set V when the accuracy of the 2-layer neural networks was maximized, i.e., 87.17%, for Feature set IV. From the above results, it can be inferred that, when the feature set is too large, a dimensional disaster problem does occur. Since the purpose of this study was to compare the predictive ability of different algorithms on P2P credit scoring, we tried to avoid dimensional disasters to provide a consistent criterion for different algorithms. Therefore, we chose Feature set V for further analysis. Table 5 summarizes the 16 features in Feature set V. We can find that these 16 features belong to credit history or personal information categories and have a prediction accuracy of around 86% to 87%, which is consistent with past studies.

Table 5. 16 features in feature set V.

Feature	Category	Description
"dti"	personal information	A ratio calculated using the borrower's total monthly debt payments to the total debt obligations, excluding the mortgage and requested LC loan, divided by the borrower's self-reported monthly income
"mo_sin_old_il_acct"	credit history	Months since oldest bank installment account opened
"annual_inc"	personal information	The self-reported annual income provided by the borrower during registration
"revol_util"	credit history	Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit
"avg_cur_bal"	personal information	Average current balance of all accounts
"total_bc_limit"	credit history	Total bankcard high credit/credit limit
"mo_sin_old_rev_tl_op"	credit history	Months since oldest revolving account opened
"revol_bal"	credit history	Total credit revolving balance
"total_bal_ex_mort"	credit history	Total credit balance excluding mortgage
"mths_since_recent_bc"	credit history	Months since most recent bankcard account opened
"total_acc"	credit history	The total number of credit lines currently in the borrower's credit file
"fico_range_low"	personal information	The lower boundary of range the borrower's FICO belongs to
"last_fico_range_high"	personal information	The upper boundary of range the borrower's FICO belongs to
"mths_since_recent_inq"	credit history	Months since most recent inquiry
"mo_sin_rcnt_rev_tl_op"	credit history	Months since most recent revolving account opened
"pct_tl_nvr_dlq"	credit history	Percent of trades never delinquent

4. Empirical results

4.1. Optimized hyperparameters

Using the selected features in Feature set V, we found the optimized hyperparameters for the seven algorithms by performing a grid search and cross-validation as described by Hsu et al. (2003). Each hyperparameter set went through five rounds of cross-validation. For each time, we divided the previous trained data into five sub-samples, among which, four sub-samples were randomly chosen as the training set, and the other one was kept to verify the accuracy. Finally, we calculated the average of these five verified accuracies as the result for each algorithm and compared the optimized hyperparameter set.

We conducted the empirical analysis using Python. We operated the corresponding packages in Sklearn for the LR and ML algorithms, corresponding packages for XGBoost and LightGBM and TenserFlow for the 2-layer neural networks on a GPU (NVIDEA GTX 950M). Considering the length of the paper, for definitions of the hyperparameters for each algorithm, please refer to the corresponding packages or technology documents.

Table 6. Optimized hyperparameter sets for the LR, SVM, DT and RF algorithms.

Algorithm		Optimized hyperparameter set	Verified accuracy	Runtime
LR		{penalty = L1, C = 0.01}	86.32%	6.0 s
SVM	RBF	{C = 10, gamma = 0.01}	86.93%	5898.0 s
	Linear	{C = 10, gamma = 0.1, coef0 = 100}	86.63%	5531.4 s
	Polynomial (poly=2)	{C = 10, gamma = 1, coef0 = 10}	86.47%	5603.7 s
	Sigmoid	{C = 0.01, gamma = 0.01, coef0 = 0.01}	86.12%	5653.0 s
DT		{criterion = gini, max_depth = 5, max_features = 0.8, max_leaf_nodes = None, min_samples_leaf = 7, min_samples_split = 3, min_weight_fraction_leaf = 0}	86.93%	8.5 s
RF		{n_estimator = 70, crition = gini, max_depth = 9, max_features = 0.8, max_leaf_nodes = 1000, min_samples_leaf =5, min_samples_split = 3, min_weight_fraction_leaf = 0}	87.15%	288.0 s

Table 6 summarizes the empirical results for the LR, SVM, DT and RF models. For LR, we searched for the optimized regularization and penalty coefficient. For the SVM, we searched for the optimized hyperparameters and penalty coefficient using four different kernel functions; we found that the best kernel function was the RBF. For the DT, the hyperparameters we searched for included the branch criterion function, stopping conditions and others. For the RF, the hyperparameters we searched for included the number of trees and other related parameters. The results show that the SVM was the most time-consuming. Though the RF required a longer time than the LR and DT algorithms, it also provided better verified accuracy.

Table 7 summarizes the empirical results for the XGBoost, LightGBM and 2-layer neural network models, which were optimized by applying a multiperiod. For XGBoost and LightGBM, the whole optimization process was divided into three periods. In Period 1, we searched for the learning rate and number of DTs. After that, in Period 2, we adjusted the hyperparameters of the DTs, including the maximum depth. Finally, we adjusted the regularization-related parameters, reduced the complexity of the model, improved the training speed and reduced overfitting. For the 2-layer neural networks, the whole process was divided into five periods. In Period 1, we searched for the activation function and related hyperparameters. After that, in Period 2, we adjusted the optimizers and learning speed using ANN learning rules. In Period 3, we adjusted the network structure and searched for the optimal number of hidden layers of 512 nodes. In Period 4, we dealt with the hyperparameters related to overfitting, including the dropout probability and intensity coefficient for regularization. Finally, we obtained the optimized hyperparameter set.

Table 7. Optimized hyperparameter sets for the XGBoost, LightGBM and 2-layer neural network algorithms.

Algorithm		Optimized hyperparameter set	Verified accuracy	Runtime
XGBoost	Period 1	{learning_rate = 0.15, n_estimators = 100}	87.10%	132.0 s
	Period 2	{learning_rate = 0.15, n_estimators = 100, max_depth = 9, subsample = 1, colsample_bytree = 0.7, min_child_weight = 0, max_delta_step = 0}	87.16%	
	Period 3	{learning_rate = 0.15, n_estimators = 100, max_depth = 9, subsample = 1, colsample_bytree = 0.7, min_child_weight = 0, max_delta_step = 0, reg_lambda = 0.2, reg_lambda = 0.4}	87.22%	
LightGBM	Period 1	{learning_rate = 0.1, n_estimators = 100}	86.99%	29.8 s
	Period 2	{learning_rate = 0.1, n_estimators = 100, max_depth = 9, bagging_fraction = 0.8, feature_fraction = 0.5, bagging_freq = 6, min_child_weight = 0.0005, min_child_samples = 15, num_leaves = 360}	87.21%	
	Period 3	{learning_rate = 0.1, n_estimators = 100, max_depth = 9, bagging_fraction = 0.8, feature_fraction = 0.5, bagging_freq = 6, min_child_weight = 0.0005, min_child_samples = 15, num_leaves = 360, reg_lambda = 0, reg_alpha = 0}	87.21%	
2-layer neural networks	Period 1	{activation_function = 'leaky relu', lambda = 0.2}	87.1%	21436.5 s
	Period 2	{activation_function = 'leaky relu', lambda = 0.2, learning_rate = 0.01, optimizer = 'mini-batch gradient descent'}	87.1%	
	Period 3	{activation_function = 'leaky relu', lambda = 0.2, learning_rate = 0.01, optimizer = 'mini-batch gradient descent', 512_node_hidden_layer = 2}	87.1%	
	Period 4	{activation_function = 'leaky relu', lambda = 0.2, learning_rate = 0.01, optimizer = 'mini-batch gradient descent', 512_node_hidden_layer = 2, prob = 0.5, beta = 0.01}	87.12%	
	Period 5	{activation_function = 'leaky relu', lambda = 0.2, learning_rate = 0.01, optimizer = 'mini-batch gradient descent', 512_node_hidden_layer = 2, prob = 0.5, beta = 0.01, batch = 64}	87.12%	

4.2. Model performance

In order to evaluate the prediction capability of the credit-scoring models, we calculated the accuracy, recall, precision and F1 score for each algorithm using the corresponding optimized hyperparameter set. The results are summarized in Table 8 and Figure 2.

Table 8. Model performance for the seven algorithms.

Algorithms	Accuracy	Recall	Precision	F1 score	Runtime
LR	86.81%	86.99%	86.66%	86.82%	0.1 s
SVM	87.38%	91.52%	84.50%	87.87%	75.7 s
DT	87.23%	91.76%	84.12%	87.77%	0.2 s
RF	87.58%	91.40%	84.90%	88.03%	17.9 s
XGBoost	87.90%	91.22%	85.52%	88.28%	3.5 s
LightGBM	87.58%	90.98%	85.18%	87.98%	0.8 s
2-layer neural networks	87.06%	90.56%	84.61%	87.49%	213.6 s

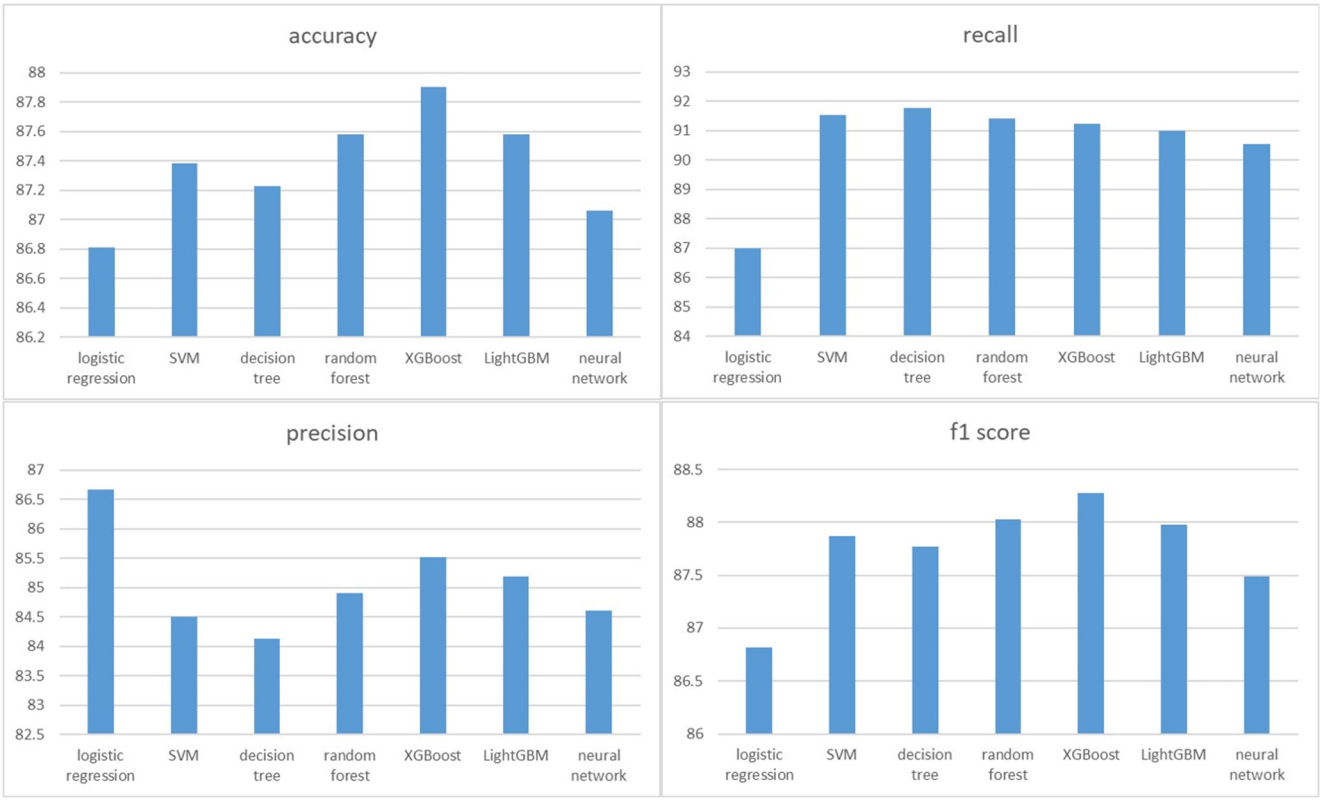

Figure 2. Model performance for the seven algorithms.

According to the results above, we obtained the following three findings:

1. For the two most comprehensive indicators, i.e., the accuracy and F1 score, the most outstanding method was XGBoost, with accuracy close to 88%, followed by LightGBM and the RF, both with an accuracy of about 87.6%. Therefore, the ensemble learning algorithms demonstrate good performance as credit-scoring models. However, the LR and 2-layer neural network algorithms performed the worst, indicating that they are less suitable for credit scoring in terms of predictive power.

2. Regarding the precision, LR performed the best, with precision above 86.5%. In addition, the precision values for XGBoost and LightGBM were both above 85%. Therefore, LR and GBDT methods are quite suitable for credit-scoring models and do not encourage excessive default risks for lending platforms and investors. However, other methods based on DTs are not suitable.

3. For the recall, the ML methods and 2-layer neural network model were above 90%, with LR demonstrating obviously worse performance. This means that, in the case of LR, the criteria for judging

whether the borrower defaults are too conservative, which makes it easy for the platform and investors to lose profiting opportunities.

Above all, GBDT methods, including XGBoost and LightGBM, are recommended as the credit-scoring models.

5. Conclusions

To reduce the information asymmetry between investors and lenders, and to allow P2P lending platforms to establish accurate lending criteria, we used the data of Lending Club to construct credit-scoring models based on ML and ANN algorithms. We cleaned the data by applying the concept of feature engineering and trained it with XGBoost to obtain the feature importance. After that, we selected 16 of the most significant default factors and categorized them into two types: personal information and credit history. Besides, we built up the training set and test set by undersampling. We found the optimized hyperparameters of seven algorithms, including LR, SVM, DT, RF, XGBoost and LightGBM algorithms, by employing a grid search and cross-validation approach. By comparing the predictive performance of each model, we found that the GBDT methods, including XGBoost and LightGBM, are the most suitable P2P credit-scoring models, as they have much better performance than 2-layer neural networks and the traditional approach of using LR. Furthermore, XGBoost had the best performance, with accuracy around 88%.

Regarding future extensions, there are several directions. First of all, it is possible to consider the use of other linear classification techniques, such as the LDA (Hastie et al., 2009). In addition, instead of adopting the undersampling a priori as we have done, future studies can try to use the classifiers on the unbalanced data as well. Furthermore, it would be an interesting extension to exploit the XGBoost algorithm by using many variable selection techniques in statistics. For example, one could fit a logistic regression using the LASSO parameter estimator (Tibshirani, 1996). Last but not least, since the data contains description features, such as the reasons for the loans and the credit document from the lender, the information in text can be converted into a numerical form via natural language processing, such as sentiment analysis. We think that following studies can focus this direction to explore more information for default classification.

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

Aldrich JH, Nelson FD (1984) *Quantitative Applications in the Social Sciences: Linear Probability, Logit, and Probit Models,* Thousand Oaks, CA: SAGE Publications. https://doi.org/10.4135/9781412984744

Alexander VE, Clifford CC (1996) *Categorical Variables in Developmental Research: Methods of Analysis*, Elsevier. https://doi.org/10.1016/B978-012724965-0/50003-1

Arya S, Eckel C, Wichman C (2013) Anatomy of the Credit Score. *J Econ Behav Organ* 95: 175–185. https://doi.org/10.1016/j.jebo.2011.05.005

Baesens B, Van Gestel T, Viaene S, et al. (2003) Benchmarking state-of-the-art classification algorithms for credit scoring. *J Oper Res Soc* 54: 627–635. https://doi.org/10.1057/palgrave.jors.2601545

Baesens B, Gestel TV, Stepanova M, et al. (2004) Neural Network Survival Analysis for Personal Loan Data. *J Oper Res Soc* 56: 1089–1098. https://doi.org/10.1057/palgrave.jors.2601990

Bishop CM (2006) *Pattern Recognition and Machine Learning*, Springer. https://doi.org/10.1007/978-0-387-45528-0_5

Bolton C (2010) *Logistic Regression and its Application in Credit Scoring*, University of Pretoria. Available from: http://hdl.handle.net/2263/27333.

Breiman L (1996) Bagging Predictors. *Mach Learn* 24: 123–140. https://doi.org/10.1007/BF00058655

Breiman L, Friedman J, Stone CJ, et al. (1984) *Classification and Regression Trees*, Taylor & Francis. https://doi.org/10.1201/9781315139470

Brown M, Grundy M, Lin D, et al. (1999) *Knowledge-Base Analysis of Microarray Gene Expression Data Using Support Vector Machines*, University of California in Santa Cruz. https://doi.org/10.1073/pnas.97.1.262

Byanjankar A, Heikkilä M, Mezei J (2015) Predicting credit risk in peer-to-peer lending: A neural network approach. *In 2015 IEEE symposium series on computational intelligence, IEEE,* 719–725. https://doi.org/10.1109/SSCI.2015.109

Cao A, He H, Chen Z, et al. (2018) Performance evaluation of machine learning approaches for credit scoring. *Int J Econ Financ Manage Sci* 6: 255–260. https://doi.org/10.11648/j.ijefm.20180606.12

Chen S, Wang Q, Liu S (2019) Credit risk prediction in peer-to-peer lending with ensemble learning framework. *In 2019 Chinese Control And Decision Conference (CCDC), IEEE,* 4373–4377. https://doi.org/10.1109/CCDC.2019.8832412

Chen TQ, Guestrin C (2016) XGBoost: A Scalable Tree Boosting System. *KDD'16 Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. https://doi.org/10.1145/2939672.2939785

Cortes C, Vapnik V (1995) Support-vector networks. *Mach Learn* 20: 273–297. https://doi.org/10.1007/BF00994018

Crouhy M, Galai D, Mark R (2014) *The Essentials of Risk Management*, 2nd Edition. McGraw-Hill. Available from: https://www.mhprofessional.com/9780071818513-usa-the-essentials-of-risk-management-second-edition-group.

Cybenko G (1989) Approximation by Superpositions of a Sigmoidal Function Mathematics of Control. *Signals Syst* 2: 303–314. https://doi.org/10.1007/BF02551274

Duan J (2019) Financial system modeling using deep neural networks (DNNs) for effective risk assessment and prediction. *J Franklin Inst* 356: 4716–4731. https://doi.org/10.1016/j.jfranklin.2019.01.046

Duchi J, Hazan E, Singer Y (2011) Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *J Mach Learn Rese* 12: 2121–2159. https://doi.org/10.5555/1953048.2021068

Elrahman SMA, Abraham A (2013) A Review of Class Imbalance Problem. *J Network Innov Comput* 1: 332–340. http://ias04.softcomputing.net/jnic2.pdf

Everett CR (2015) Group Membership, Relationship Banking and Loan Default Risk: the Case of Online Social Lending. *Bank Financ Rev* 7: 15–54. https://doi.org/10.2139/ssrn.1114428

Friedman JH (2001) Greedy Function Approximation: A Gradient Boosting Machine. *Ann Stat* 29: 1189–1232. https://doi.org/10.1214/aos/1013203451

Genuer R, Poggi JM, Tuleau-Malot C (2010) Variable selection Using Random Forests. *Pattern Recogn Lett* 31: 2225–2236. https://doi.org/10.1016/j.patrec.2010.03.014

Glorot X, Bengio Y (2010) Understanding the Difficulty of Training Deep Feedforward Neural Networks. *J Mach Learn Res* 9: 249–256. http://proceedings.mlr.press/v9/glorot10a.html

Guyon I, ElNoeeff A (2003) An Introduction to Variable and Feature Selection. *J Mach Learn Res* 3: 1157–1182. https://www.jmlr.org/papers/v3/guyon03a.html

Hastie T, Tibshirani R, Friedman JH (2009) *The elements of statistical learning: data mining, inference, and prediction*, Springer. https://doi.org/10.1007/978-0-387-84858-7

He KM, Zhang XY, Ren SQ, et al. (2015) Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *IEEE international conference on computer vision*. https://doi.org/10.1109/ICCV.2015.123

Ho TK (1995) Random Decision Forest, *Proceeding of the 3rd International Conference on Document Analysis and Recognition,* 278–282. https://doi.org/10.1109/ICDAR.1995.598994

Ho TK (1998) The Random Subspace Method for Constructing Decision Forests. *IEEE T Pattern Anal* 20: 832–844. https://doi.org/10.1109/34.709601

Hochreiter S, Bengio Y, Frasconi P, et al. (2001) Gradient Flow in Recurrent Nets: the Difficulty of Learning Long-Term Dependencies. In *A Field Guide to Dynamical Recurrent Networks, IEEE*, 237–243. https://doi.org/10.1109/9780470544037.ch14.

Hsu CW, Chang CC, Lin CJ (2003) *A Practical Guide to Support Vector Classification*. National Taiwan University, 1–12. Available from: https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.

Ioffe S, Szegedy C (2015) Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *International conference on machine learning,* 448–456. https://doi.org/10.48550/arXiv.1502.03167

Iyer R, Khwaja AI, Luttmer EF, et al. (2009) Screening in New Credit Markets: Can Individual Lenders Infer Borrower Creditworthiness in Peer-to-Peer Lending? *AFA 2011 Denver Meetings Paper*. https://doi.org/10.2139/ssrn.1570115

Kang H (2013) The Prevention and Handling of the Missing Data. *Korean J Anesthesiol* 64: 402–406. https://doi.org/10.4097/kjae.2013.64.5.402

Ke GL, Meng Q, Finley T, et al. (2017) LightGBM: A highly Efficient Gradient Boosting Decision Tree, *Neural Information Processing Systems,* 3149–3157. Available from: https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

Keogh E, Mueen A (2017) Curse of Dimensionality. *Encyclopedia of Machine Learning and Data Mining*, Boston: Springer. https://doi.org/10.1007/978-1-4899-7687-1_192

Kingma DP, Ba JL (2015) Adam: a Method for Stochastic Optimization. *International Conference on Learning Representations,* 1–13. https://doi.org/10.48550/arXiv.1412.6980

Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet Classification with Deep Convolutional Neural Networks, *Advances in Neural Information Processing Systems,* 1097–1105. https://doi.org/10.1145/3065386

Lantz B (2013) *Machine Learning with R*. Packt Publishing Limited. Available from: https://edu.kpfu.ru/pluginfile.php/278552/mod_resource/content/1/MachineLearningR__Brett_Lantz.pdf.

Li LH, Sharma AK, Ahmad R, Chen RC (2021) Predicting the Default Borrowers in P2P Platform Using Machine Learning Models. *In International Conference on Artificial Intelligence and Sustainable Computing*. https://doi.org/10.1007/978-3-030-82322-1_20

Lin HT, Lin CJ (2003) *A Study on Sigmoid Kernels for SVM and the Training of Non-PSD Kernels by SMO-type Methods*. National Taiwan University. Available from: https://www.csie.ntu.edu.tw/~cjlin/papers/tanh.pdf

Lu L, Shin YJ, Su YH, et al. (2019) Dying ReLU and Initialization: Theory and Numerical Examples. *arXiv preprint arXiv:1903.06733.* https://doi.org/10.4208/cicp.OA-2020-0165

Maas AL, Hannun AY, Ng AY (2013) Rectifier Nonlinearities Improve Neural Network Acoustic Models. *ICML Workshop on Deep Learning for Audio, Speech, and Language Processing*. Available from: https://ai.stanford.edu/~amaas/papers/relu_hybrid_icml2013_final.pdf.

Madasamy K, Ramaswami M (2017) Data Imbalance and Classifiers: Impact and Solutions from a Big Data Perspective. *Int J Comput Intell Res* 13: 2267–2281. Available from: https://www.ripublication.com/ijcir17/ijcirv13n9_09.pdf.

McCulloch WS, Pitts W (1943) A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bull Math Biophys* 5: 115–133. https://doi.org/10.2307/2268029

Mester LJ (1997) What's the Point of Credit Scoring? *Bus Rev* 3: 3–16. Available from: https://www.philadelphiafed.org/-/media/frbp/assets/economy/articles/business-review/1997/september-october/brso97lm.pdf.

Mijwel MM (2018) *Artificial Neural Networks Advantages and Disadvantages*. Available from: https://www.linkedin.com/pulse/artificial-neural-networks-advantages-disadvantages-maad-m-mijwel/.

Mills KG, McCarthy B (2016) The State of Small Business Lending: Innovation and Technology and the Implications for Regulation. *HBS Working Paper No. 17-042*. https://doi.org/10.2139/ssrn.2877201

Mountcastle VB (1957) Modality and Topographic Properties of Single Neurons of Cat's Somatic Sensory Cortex. *J Neurophysiol* 20: 408–434. https://doi.org/10.1152/jn.1957.20.4.408

Ohlson JA (1980) Financial Ratios and the Probabilistic Prediction of Bankruptcy. *J Account Res* 18: 109–131. https://doi.org/10.2307/2490395

Patro SGK, Sahu KK (2015) *Normalization: A Preprocessing Stage*. https://doi.org/10.17148/IARJSET.2015.2305

Pontil M, Verri A (1998) Support Vector Machines for 3D Object Recognition. *IEEE Trans PAMI* 20: 637–646. https://doi.org/10.1109/34.683777

Qian N (1999) On the Momentum Term in Gradient Descent Learning Algorithms. *Neural Networks* 12: 145–151. https://doi.org/10.1016/S0893-6080(98)00116-6

Quinlan JR (1987) Simplifying Decision Trees. *Int J Man-Mach Stud* 27: 221–234. https://doi.org/10.1016/S0020-7373(87)80053-6

Quinlan JR (1992) *C4.5: Programs for Machine Learning*. San Mateo: Morgan Kaufmann Publishers Inc. Available from: https://www.elsevier.com/books/c45/quinlan/978-0-08-050058-4.

Rajan U, Seru A, Vig V (2015) The Failure of Models that Predict Failure: Distance, Incentives, and Defaults. *J Financ Econ* 115: 237–260. https://doi.org/10.1016/j.jfineco.2014.09.012

Rosenblatt F (1958) The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychol Rev* 65: 386–408. https://doi.org/10.1037/h0042519

Ruder S (2017) An Overview of Gradient Descent Optimization Algorithms. *arXiv preprint arXiv:1609.04747*. https://doi.org/10.6919/ICJE.202102_7(2).0058

Rumelhart DE, Hinton GE, Williams RJ (1986) Learning Representations by Back-Propagating Errors. *Nature* 323: 533–536. https://doi.org/10.1038/323533a0

Samitsu A (2017) The Structure of P2P Lending and Legal Arrangements: Focusing on P2P Lending Regulation in the UK. *IMES Discussion Paper Series, No. 17-J-3*. Available from: https://www.boj.or.jp/en/research/wps_rev/lab/lab17e06.htm/

Serrano-Cinca C, Gutierrez-Nieto B, López-Palacios L (2015) Determinants of Default in P2P Lending. *PloS One* 10: e0139427. https://doi.org/10.1371/journal.pone.0139427

Serrano-Cinca C, Gutiérrez-Nieto B (2016) The use of profit scoring as an alternative to credit scoring systems in peer-to-peer (P2P) lending. *Deci Support Syst* 89: 113–122. https://doi.org/10.1016/j.dss.2016.06.014

Shannon C (1948) A Mathematical Theory of Communication. *Bell Syst Tech J* 27: 379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

Shelke MS, Deshmukh PR, Shandilya VK (2017) A Review on Imbalanced Data Handling using Undersampling and Oversampling Technique. *Int J Recent Trends Eng Res*. https://doi.org/10.23883/IJRTER.2017.3168.0UWXM

Singh S, Gupta P (2014) Comparative Study Id3, Cart and C4.5 Decision Tree Algorithm: A Survey. *Int J Adv Inf Sci Technol (IJAIST)* 27: 97–103. https://doi.org/10.15693/ijaist/2014.v3i7.47-52

Srivastava N, Hinton G, Krizhevsky A, et al. (2014) Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J Mach Learn Res* 15: 1929–1958. https://doi.org/https://jmlr.org/papers/v15/srivastava14a.html

Thomas LC (2000) A Survey of Credit and Behavioural Scoring: Forecasting Financial Risk of Lending to Consumers. *Int J Forecast* 16: 149–172. https://doi.org/10.1016/S0169-2070(00)00034-0

Tibshirani R (1996) Regression shrinkage and selection via the lasso. *J Royal Stat Soc (Methodological)* 58: 267–288. https://doi.org/10.1111/j.2517-6161.1996.tb02080.x

Tieleman T, Hinton G (2012) *Lecture 6.5—RMSProp, COURSERA: Neural Networks for Machine Learning*.

Verified Market Research (2021) *Global Peer to Peer (P2P) Lending Market Size by Type, by End User, by Geographic Scope and Forecast*. Available from: https://www.verifiedmarketresearch.com/product/peer-to-peer-p2p-lending-market/.

Wang Z, Cui P, Li FT, et al. (2014) A Data-Driven Study of Image Feature Extraction and Fusion. *Inf Sci* 281: 536–558. https://doi.org/10.1016/j.ins.2014.02.030

AIMS Press