mathematics

Article

A Comparative Performance Assessment of Ensemble Learning for Credit Scoring

Yiheng Li * and Weidong Chen

College of Management and Economics, Tianjin University, Tianjin 300072, China; chenweidong@tju.edu.cn
* Correspondence: liyiheng@tju.edu.cn

Received: 5 September 2020; Accepted: 29 September 2020; Published: 13 October 2020

Abstract: Extensive research has been performed by organizations and academics on models for credit scoring, an important financial management activity. With novel machine learning models continue to be proposed, ensemble learning has been introduced into the application of credit scoring, several researches have addressed the supremacy of ensemble learning. In this research, we provide a comparative performance evaluation of ensemble algorithms, i.e., random forest, AdaBoost, XGBoost, LightGBM and Stacking, in terms of accuracy (ACC), area under the curve (AUC), Kolmogorov–Smirnov statistic (KS), Brier score (BS), and model operating time in terms of credit scoring. Moreover, five popular baseline classifiers, i.e., neural network (NN), decision tree (DT), logistic regression (LR), Naïve Bayes (NB), and support vector machine (SVM) are considered to be benchmarks. Experimental findings reveal that the performance of ensemble learning is better than individual learners, except for AdaBoost. In addition, random forest has the best performance in terms of five metrics, XGBoost and LightGBM are close challengers. Among five baseline classifiers, logistic regression outperforms the other classifiers over the most of evaluation metrics. Finally, this study also analyzes reasons for the poor performance of some algorithms and give some suggestions on the choice of credit scoring models for financial institutions.

Keywords: credit scoring; ensemble learning; baseline classifiers; comparative assessment

1. Introduction

Charging interest on loan is the main business income way of financial institutions and in the meanwhile, granting loans has great contribution in socioeconomic operation. However, unreasonable lending will cause a huge loss given default, and hence to large losses at many financial institutions. Emerging and developing economies urgently need to strike a careful balance between promoting growth through lending loan and preventing credit risk caused by excessive releasing according to the Global Economic Prospects [1] from the World Bank in 4 June 2019. Credit scoring has been one of the primary approaches for principal institutions to evaluate credit risk and make rational decisions [2]. The purpose of credit scoring model is to solve problems of classifying loan customers into two categories: good customers (those are predicted to be fully paid within the specified period) and bad customers (those predicted to be default). Good customers are more likely to repay their loans on time, which will bring benefits to financial institutions. On the contrary, bad customers will lead to financial losses. Therefore, banks and financial institutions pay more and more attention to the construction of credit scoring models, because even a 1% increase in the quality of the bad credit applicants would significantly translate into remarkable future savings for financial institutions [3,4].

Since the pioneering work of Beaver [5] and Altman [6], credit scoring has been a major research subject for scholars and financial institutions, and has obtained extensive research in the past 40 years. Subsequently, many types of credit scoring models have been proposed and developed by using statistical methods, such as linear discriminate analysis (LDA) and logistic regression (LR) [7–10].

However, data is exploding in today's world, when it comes to huge quantities of data, the elastic performance of classical statistical analysis models is not very good. As a result, some of the assumptions in these models cannot be established, which in turn affects the accuracy of predictions. With the breakthrough of artificial intelligence (AI) technology, such as neural networks (NN) [11,12], support vector machines (SVM), random forests (RF) [13], and Naïve Bayes (NB) [14] can achieve the same or better results compared with the statistical models. However, LDA and LR still have a wide range of applications due to their high accuracy and ease of model operation. Machine learning belongs to the category of AI and is one of the most efficient methods in data mining that can provide analysts with more productive insights on the use of big data [15]. Machine learning models are generally subdivided into individual machine learning (NN and LR), ensemble learning (bagging, boosting, and randomspace), and integrated ensemble machine learning (RS-boosting and multi-boosting) [16]. Ensemble learning approaches are considered to be a state-of-the-art solution for many machine learning tasks [17]. After Chen and Guestrin [18] proposed XGBoost in 2016, ensemble learning, i.e., XGBoost and LightGBM, have become a winning strategy to a wide range of Kaggle competitions due to its inspiring success. Accordingly, more and more scholars introduce ensemble learning models to solve credit scoring problems [19–22]. Ensemble learning is a method to make excellent performance by building and combining multiple base learners with certain strategies. It can be used to solve classification problems, regression problems, feature selection, outlier detection, etc. Generally, ensemble learning can be separated into homogenous and heterogeneous, where the former combines classifiers of the same kind, whereas the latter combines classifiers of different kinds [22]. Boosting represented by AdaBoost and Bagging represented by random forest belong to homogenous integration in ensemble learning, otherwise, Stacking belongs to the heterogeneous (Figure 1).

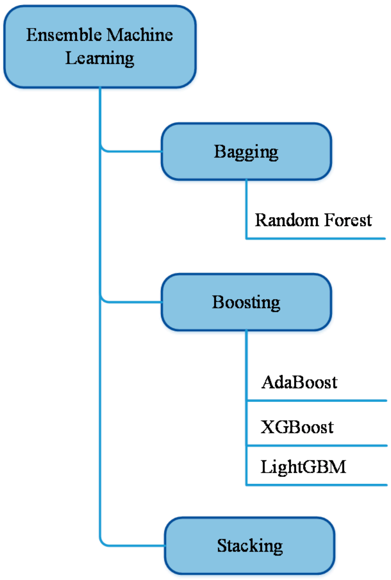


Figure 1. Ensemble machine learning framework.

Scholars have also conducted comparative studies on ensemble learning models: Bagging, Boosting, and Stacking. Wang et al. [4] investigated the performance of ensemble learning-bagging, boosting, stacking-based on four individual learners (LR, DT, NN, and SVM) on credit scoring problems across three real world credit datasets and found that bagging gets better than boosting, stacking, and bagging DT perform the best output. Barboza, Kimura, and Altman [23] conduct the comparison between machine learning methods (SVM, bagging, boosting, and RF) and some traditional methods (discriminant analysis, LR, and NN) to predict bankruptcy, showed that bagging, boosting, and RF outperform the other algorithms. Ma et al. [15] used data cleaning method of "multiple observation" and "multiple dimensional" to evaluate the performance of XGBoost and LightGBM on a prediction of P2P loan default, respectively, and observed that the result of LightGBM based on multiple observational dataset performs the best. Alazzam, Alsmadi, and Akour [24] provided the prediction of faulty modules of software systems using bagging, boosting, and stacking, and showed that boosting illustrated a better performance than bagging. Jhaveri et al. [25] predicted the Kickstarter campaign with a variety of classification and boosting algorithms and concluded that weighted RF and AdaBoost achieved the best accuracy. From the above, different datasets seem to yield different comparative research conclusions. Therefore, our study does contribute to this debate and is expected to provide a strong foundation for default prediction and credit rating.

The purpose of this study is to analyze the performance of five ensemble learning methods (RF, AdaBoost, XGBoost, LightGBM, and Stacking) in the area of credit scoring in terms of accuracy (ACC), area under the curve (AUC), Kolmogorov–Smirnov statistic (KS), Brier score (BS), and model operating time. The public credit dataset from Lending Club will be used to construct all classifiers and test their performance. Moreover, five popular individual learners, i.e., NN, SVM, LR, DT, and NB, are considered to be benchmarks. It is worth noting that RF is a model recognized by many scholars as an excellent classification of credit scoring models [26–29]. This paper also proposes some ex-ante hypotheses: first, ensemble learning models outperform traditional models; second, the application of Bagging will bring a substantial improvement for DT; therefore, RF (Bagging DT) performs better than DT; and third, the model operating time of XGBoost and LightGBM may be relatively short, on the contrary, NN and SVM may be time-consuming.

The layout of the paper is structured as follows: Section 2 introduces the mechanism of RF, AdaBoost, XGBoost, LightGBM, and Stacking. Section 3 specifies the experiment setup from four aspects: data pre-processing and feature selection, baseline models, evaluation measure and hyper-parameter tuning, meanwhile, Section 4 describes the experimental findings. Finally, Section 5 draws conclusions and addresses potential work.

2. Methodology

2.1. Overviews of Ensemble Learning

Ensemble learning improves performance by building and aggregating multiple different base learners with specific strategies. The constructed series of models are base learners, and the aggregating method of base learners is integration strategy. In general, the performance of each base learner is not required to be particularly good in ensemble learning, but just better than randomly guessing. According to the base learner generation process, ensemble learning method can be roughly divided into two types (Figure 2): 1. A parallel method represented by Bagging, and 2. sequential method represented by Boosting.

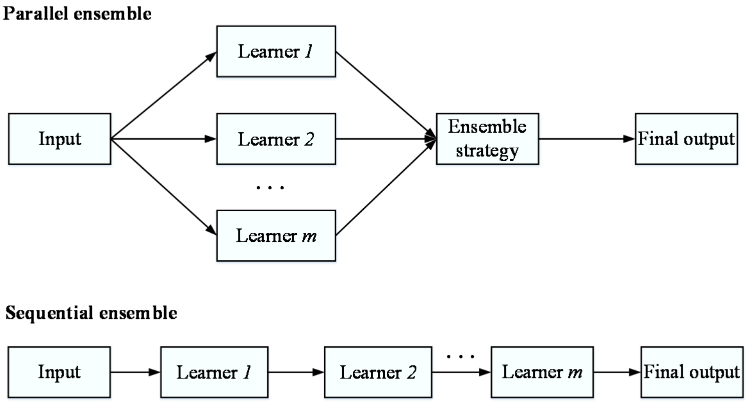


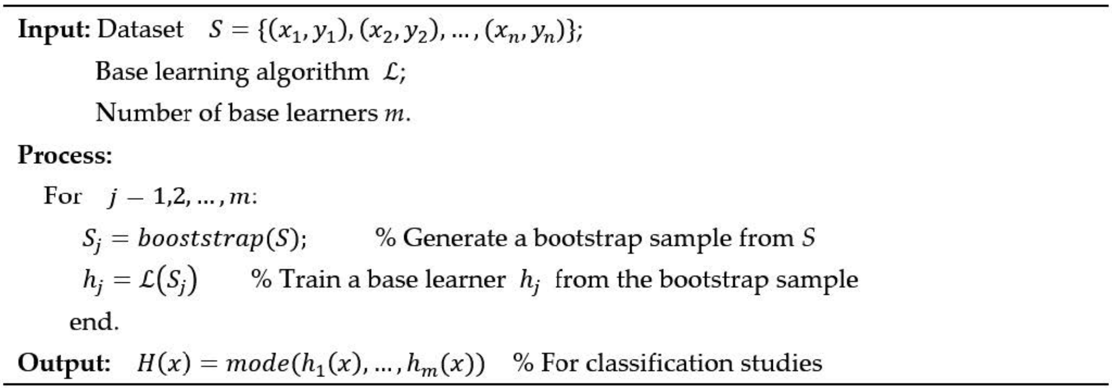
Figure 2. Flowchart of Parallel and Sequential ensemble.

In the parallel ensemble, multiple base learners are constructed simultaneously, and the independence of these base learners is utilized to improve the performance of final model. In fact, Bootstrap Sampling ensures the relative independence of base learners and the prime advantage of the parallel ensemble is that it is conducive to parallel computing and can significantly reduce the required training time. However, the sequential ensemble is to construct multiple learners in sequence, and the latter learners can avoid the errors of the former learners, thus improving the aggregated performance. Different from the parallel ensemble, new learners need to be built one by one in the sequential ensemble, so it is difficult to use parallelism to reduce the training time.

In terms of integration strategies, majority voting and its variants (e.g., weighted voting) are commonly employed in the existing credit scoring literature because of its simplicity. According to majority voting, each base leaner predicts and votes for each sample, and the sample classification with the highest votes is the final predictive category, if more than one classification gets the highest number of votes, one will be randomly chosen to be the final, that is to say, the minority is subjected to the majority. In contrast to majority voting, Stacking is also a method of integration strategies, which combines low-level learners with the high-level learning algorithm (i.e., meta-classifier) [30]. However, there are few researches on Stacking in the field of credit scoring [4]. The remainder of the section will introduce three popular ensemble learning, i.e., Bagging, Boosting, and Stacking, respectively.

2.2. *Bagging*

Bagging is one of the most popular techniques for constructing ensembles [31], which is to shape several different training sets with a bootstrap sampling method, then to train base learners on each training set, and the final model will be obtained by aggregating these base learners. Bagging has two important parts: bootstrap sampling and model aggregation. Bootstrap sampling is to take *n* samples from the data set with *n* samples by using the sampling with replacement, which can ensure the independence of different sampling training sets. In addition, the majority voting method which adopts the one with the most occurrence among the classification results of multiple base learners as the final classification result is usually as the aggregation method. The algorithm [31] of Bagging is given in Figure 3.



```
Input: Dataset  $S = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$;
       Base learning algorithm  $\mathcal{L}$;
       Number of base learners $m$.
Process:
  For  $j - 1,2, \dots, m$:
    $S_j = boostrap(S)$;        % Generate a bootstrap sample from $S$
    $h_j = \mathcal{L}(S_j)$        % Train a base learner  $h_j$  from the bootstrap sample
  end.
Output:  $H(x) = mode(h_1(x), \dots, h_m(x))$   % For classification studies
```

Figure 3. The algorithm of Bagging.

Random Forest Algorithm (RF)

RF is a bagging of decision tree which randomly restrict the features used in each split and operates by creating multiple decision trees in training process. There is no correlation between each decision trees in an RF, after generating numbers of trees, the final decision class is based on a vote when a new sample comes in and each decision tree in the RF will make a judgment on which category the sample belongs to.

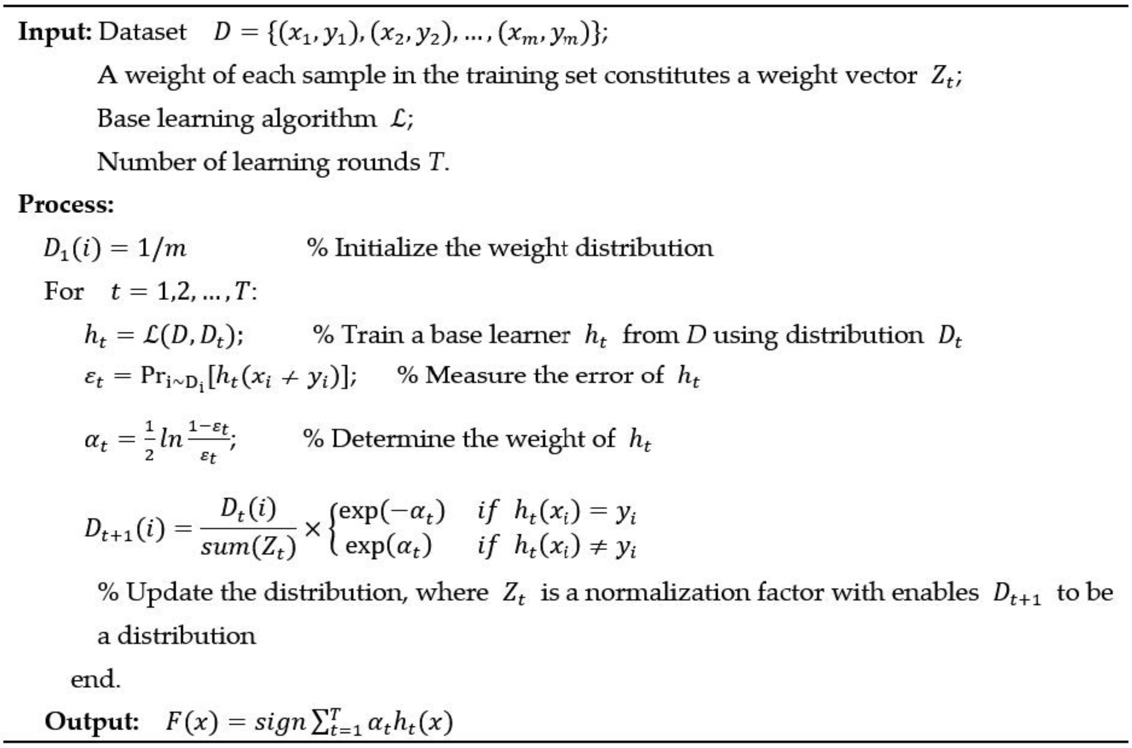
RF is considered as one of the best algorithms currently which is not sensitive to multicollinearity, and the results are relatively robust to missing data and unbalanced data. In addition, it can well predict the role of up to thousands of explanatory variables [32].

2.3. Boosting

Compared with the parallel construction of base learners in Bagging, Boosting establish a set of base classifiers in sequence, whose basic idea is firstly, a weak classifier is constructed on the training set. According to the classification result of classifier, each sample shall be assigned a weight in the training set and the weight will be relatively small if the sample is correctly classified; otherwise, it will be assigned a relatively large number. Then, in order to make the samples with large weights be accurately classified, all weight of each sample is considered to construct a second weak classifier. By repeating the process, several weak classifiers will be established in order to achieve better classification performance. The model parameters of each weak classifier are obtained by minimizing the loss function of the previous model on the training set. The final model obtained by the Boosting algorithm is a linear combination of several base classifiers weighted by their own result. In many literatures, AdaBoost is introduced as a paradigmatic example of Boosting. Thus, we introduce the AdaBoost algorithm in the study.

2.3.1. AdaBoost

Adaptive Boosting (AdaBoost) is the most widely used version of Boosting algorithm, which is proposed by Freund and Schapire [33]. In fact, the well-known AdaBoost will be obtained when the loss function is exponential type and the weights and classifiers are derived by means of forward stage-wise additive modeling. The algorithm of AdaBoost [4] is given in Figure 4.



```
Input: Dataset  $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$;
        A weight of each sample in the training set constitutes a weight vector  $Z_t$;
        Base learning algorithm  $\mathcal{L}$;
        Number of learning rounds $T$.
Process:
  $D_1(i) = 1/m$          % Initialize the weight distribution
  For   $t = 1,2,\dots,T$:
    $h_t = \mathcal{L}(D, D_t)$;      % Train a base learner  $h_t$  from $D$ using distribution  $D_t$
    $\varepsilon_t = \Pr_{i \sim D_i}[h_t(x_i \neq y_i)]$;     % Measure the error of  $h_t$
    $\alpha_t = \frac{1}{2} ln \frac{1-\varepsilon_t}{\varepsilon_t}$;       % Determine the weight of  $h_t$
    $D_{t+1}(i) = \frac{D_t(i)}{sum(Z_t)} \times \begin{cases} \exp(-\alpha_t) & if \ \ h_t(x_i) = y_i \\ \exp(\alpha_t) & if \ \ h_t(x_i) \neq y_i \end{cases}$
      % Update the distribution, where  $Z_t$  is a normalization factor with enables  $D_{t+1}$  to be
      a distribution
   end.
  Output:   $F(x) = sign \sum_{t=1}^{T} \alpha_t h_t(x)$
```

Figure 4. The algorithm of AdaBoost.

2.3.2. The Related Basic Theory-GBDT

Gradient boosting decision tree (GBDT) is also one of the Boosting algorithms and the basic idea is to combine a set of weak base learners into a strong one. Different from the traditional boosting algorithms, like AdaBoost, which use the error of previous weak learner to update the sample weight, then iterates round by round, GBDT updates the sample data in the direction of the negative gradient to make the algorithm converge globally [34]. GBDT has the advantages of strongly generalizing ability and not being easy to overfitting.

Given a dataset $\{x_i, y_i\}$, $i = 1, 2, \ldots, N$, where x_i represents a series of features and y_i denotes the label. $\Psi(y, F(x))$ is the loss function. The steps of GBDT are shown as follows [35]:

Step 1: The initial constant value of the model β is given:

$$F_0(x) = \underset{\beta}{\operatorname{argmin}} \sum\nolimits_{i=1}^{N} \Psi(y_i, \beta). \tag{1}$$

Step 2: For the number of iterations $k = 1, 2, \ldots, K$ (k is the times of iteration), the gradient of the loss function is

$$y_i^* = \left[\frac{\partial \Psi(y_i, F(x_i))}{\partial \Psi F(x_i)} \right]_{F(x) = F_{k-1}(x)}, \ i = \{1, 2, \ldots, N\}. \tag{2}$$

Step 3: The initial model $h(x_i; \theta_k)$ is formed by fitting sample data, and the parameter θ_k is obtained by using the least square method:

$$\theta_k = \underset{\theta, \beta}{\operatorname{argmin}} \sum\nolimits_{i=1}^{N} \left[y_i^* - \beta h(x_i; \theta) \right]^2. \tag{3}$$

Step 4: The new weight of the model is described as follow by minimizing the loss function

$$\gamma_k = \underset{\gamma}{\operatorname{argmin}} \sum\nolimits_{i=1}^{N} \Psi(y_i, F_{k-1}(x) + \gamma h(x_i; \theta_k)). \tag{4}$$

Step 5: Optimized the model as

$$F_k(x) = F_{k-1}(x) + \gamma_k h(x_i; \theta_k). \tag{5}$$

This loop executes until a specified number of iterations or convergence conditions are met.

2.3.3. XGBoost Algorithm

eXtreme Gradient Boosting (XGBoost) was proposed by Chen and Guestrin [18] in 2016. It offers many improvements over traditional gradient boosting algorithms, and has been recognized as an advanced estimator with ultra-high performance both in classification and regression. Different from the GBDT, the loss function has introduced the regularization in XGBoost to prevent overfitting:

$$\mathcal{L}_{\mathrm{K}}(F(x_i)) = \sum_{i=1}^{n} \Psi(y_i, F_K(x_i)) + \sum_{k=1}^{K} \Omega(f_k), \tag{6}$$

where $F_K(x_i)$ is the prediction on the i-th sample at the K-th boost, the $\Psi(*)$ is a loss function which measures the differences between the prediction and the actual label. $\Omega(f_k)$ is the regularization term, and can be expressed as

$$\Omega(f) = \gamma T + \frac{1}{2} \lambda \parallel \omega \parallel^2 . \tag{7}$$

In the regularization term, γ is the complexity parameter and represents the complexity of leaves. T denotes the number of leaves, λ indicates the penalty parameter and $\parallel \omega \parallel^2$ is the output of each leaf node.

Moreover, different from GBDT, XGBoost adopts a second-order Taylor series as the objective function. Equation (6) can be transformed as follows:

$$\mathcal{L}_{\mathrm{K}} \cong \sum\nolimits_{i=1}^{n} \left[g_i f_k(x_i) + \frac{1}{2} h_i f_k^2(x_i) \right] + \Omega(f_k), \tag{8}$$

where g_i and h_i represent the first and second order gradient statistics on the loss function, respectively. Let I_j indicate the sample set of leaf j. Equation (8) can be transformed as follows:

$$\mathcal{L}_{\mathrm{K}} = \sum_{j=1}^{T}\left[\left(\sum_{i\in I_j} g_i\right)\omega_j + \frac{1}{2}\left(\sum_{i\in I_j} h_i + \lambda\right)\omega_j^2\right] + \gamma T. \tag{9}$$

In conclusion, the objective function is transformed into the determination problem of the minimum of a quadratic function [19]. In addition, XGBoost follows the idea of GBDT and uses learning rate, boosting numbers, maximum tree depth, and subsampling to tackle over-fitting problem.

2.3.4. LightGBM Algorithm

Light Gradient Boosting Machine (LightGBM) is a GBDT framework based on decision tree algorithm which is proposed by Microsoft Research [36]. LightGBM is similar to XGBoost in that it approximates the residual (both first- and second-order) by the Taylor expansion of loss function, and introduces the regularization term to deal with the complexity of the model. Different from XGBoost which is used the pre-sorted idea of exact greedy algorithm to search split points, LightGBM can reduce the memory usage and improve the training speed by using the decision tree algorithm based on histogram. Figure 5 [19] shows the basic idea, that is to separate the continuous floating point eigenvalues into k integers and build a histogram with width k. When traversing the data, statistics are accumulated in the histogram according to the discretized value as the index, and then the optimal segmentation point can be found according to the discrete value of the histogram. The histogram algorithm can shorten the training time and reduce memory usage.

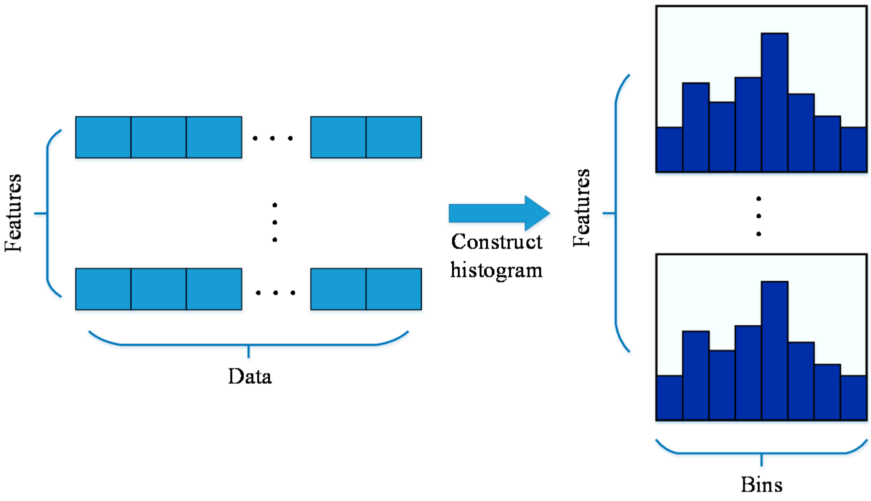


Figure 5. Histogram algorithm.

In XGBoost, trees grow by level-wise [19] growth strategy, as shown in Figure 6. According to this strategy, the same layer of leaves can be split simultaneously by traversing the data once, which is easy to carry out multithreaded optimization, control model complexity and avoid overfitting. However, level-wise is an inefficient algorithm, since it manages the same layer of leaves indiscriminately, which brings a lot of additional consumption. In fact, many leaves have low splitting information gain; thus, there is unnecessary for searching and splitting. LightGBM has optimized it and adopts the leaf-wise algorithm with depth limitation. By choosing the nodes with maximum information gain for splitting on the same layer, a great deal of extra memory consumption caused by some nodes with small gain will be avoided.

In addition, histogram algorithm [19] (Figure 7) which is introduced by LightGBM can further improve the model speed. The histogram of a leaf node can be calculated directly by the difference between parent node and sibling node histogram. Normally, it is necessary to traverse all data on the leaf to construct a histogram, but histogram method only requires traversing the k bins of the histogram. Using this method, LightGBM can construct the histogram of a leaf and obtain the histogram of its sibling leaves at quite a minimal fee, doubling the model speed.

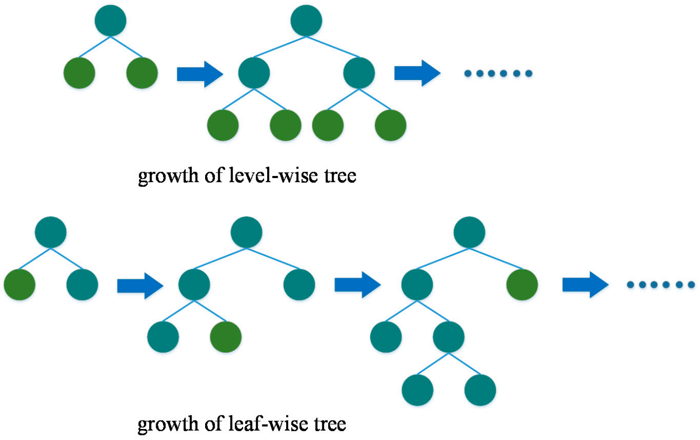


Figure 6. Process of decision tree learning using the growth of level-wise and leaf-wise.

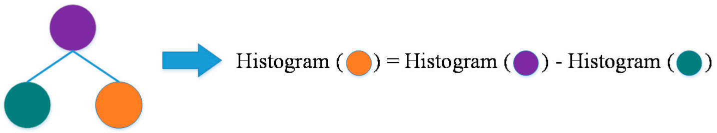
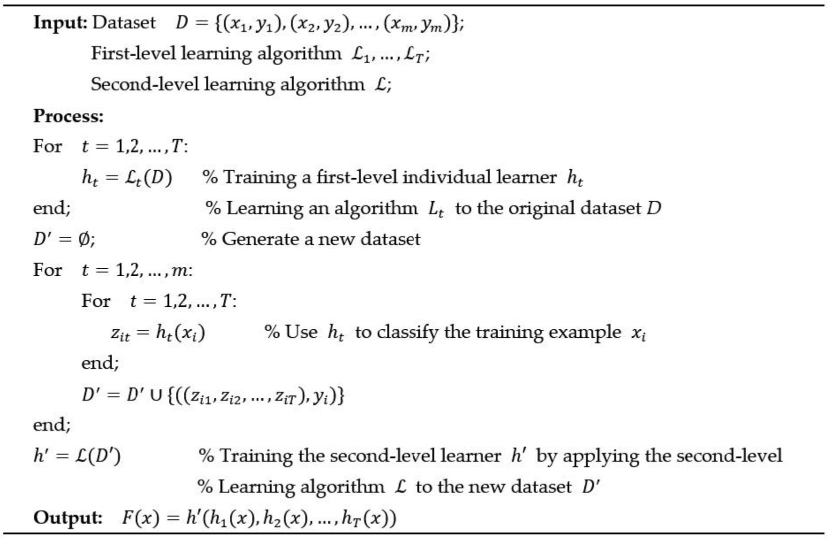


Figure 7. Histogram-based decision tree algorithm.

2.4. Stacking

As mentioned before, Stacking is one of the aggregating strategies for multiple learners and aggregates multiple models in a two-tier structure. In simple terms, the process of Stacking is to build multiple classifiers in the first layer as base-level learner, then the output of this layer is taken as new features to re-train a new learner, which is called the meta-level learner. Experience has shown that training the meta-level learner with complex models can easily lead to overfitting problem. Therefore, in many cases, the simpler models, such as linear regression, are preferred in the second layer [37]. Notably, the base-level learners are not limited to the weak learners; in fact, they are generally good performance models, such as RF, NN, SVM, etc. The algorithm of Stacking [4] is shown in Figure 8.



```
Input: Dataset  $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$;
        First-level learning algorithm  $\mathcal{L}_1, \dots, \mathcal{L}_T$;
        Second-level learning algorithm  $\mathcal{L}$;
Process:
For  $t = 1,2, \dots, T$:
    $h_t = \mathcal{L}_t(D)$     % Training a first-level individual learner  $h_t$
end;                         % Learning an algorithm  $L_t$  to the original dataset $D$
$D' = \emptyset$;                   % Generate a new dataset
For  $t = 1,2, \dots, m$:
    For  $t = 1,2, \dots, T$:
        $z_{it} = h_t(x_i)$          % Use  $h_t$  to classify the training example  $x_i$
    end;
    $D' = D' \cup \{((z_{i1}, z_{i2}, \dots, z_{iT}), y_i)\}$
end;
$h' = \mathcal{L}(D')$           % Training the second-level learner  $h'$  by applying the second-level
                             % Learning algorithm  $\mathcal{L}$  to the new dataset  $D'$
Output:  $F(x) = h'(h_1(x), h_2(x), \dots, h_T(x))$
```

Figure 8. The algorithm of Stacking.

3. Empirical Set-Up

To conduct a fair comparison assessment of the performance of five ensemble methods, contrasting with several commonly used classifiers as benchmarks, the experimental framework is demonstrated on five aspects, namely, data preparation, data pre-processing and feature selection, baseline classifiers, hyper-parameter tuning and evaluation criteria. Figure 9 demonstrates the experiment procedure. The entire experiment is coded with Python 3.6 on Jupyter, using a server with 1 Core CPU, 2 GB RAM, and Ubuntu Linux 18.04. To guarantee the reproducibility of the dividing process, this study implements the Python function *random.seed* ( ).

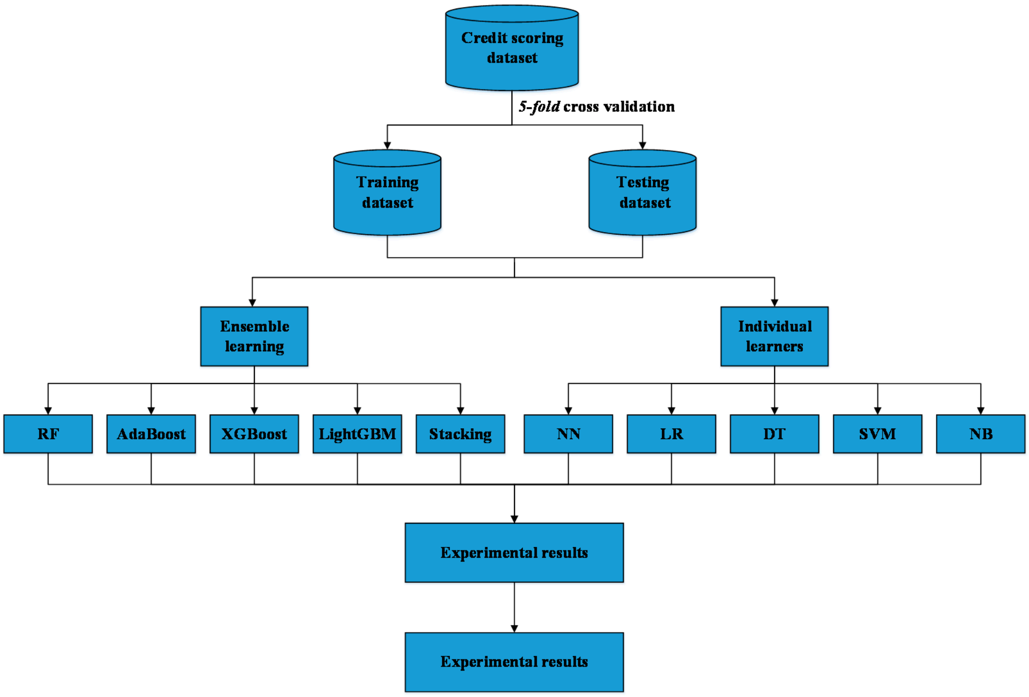


Figure 9. Experimental procedure.

3.1. Data Preparation

Lending Club is the No. 1 credit marketplace in the United States. In this experiment, a public real-world consumer credit loan dataset in Q4 of 2018 from Lending Club (https://www.lendingclub.com/info/statistics.action) is utilized as the validation of the comparison models which can be freely downloaded from the mainstream and representative website of lending platforms in the United States, has been used in several researches [15,38,39]. To be specific, Lending Club released several states of the loan data, in which fully paid means the loan has been repaid, and charged off means the loan has been written off, that is, default. This paper chooses these two loan states for study. As a result, a sum of 36,394 loans in Q4 of 2018 are utilized and analyzed, among these loans, 8016 (22.03%) are default and 28,378 (77.97%) are fully paid.

3.2. Data Pre-Processing and Feature Selection

A total of 36,394 primitive data are selected in previous sub-section, there are 150 variables, one of which is the label. Generally, there will be outlier and missing value in the raw credit data; therefore, before modeling, data pre-processing and feature selection shall be carried out to select the most discriminative variables. After filtering out loans with missing value, a total of 25,561 loans remain, 5386 (21.07%) are default and 20,175 (78.93%) are non-default loans. Unlike the traditional feature-importance scores algorithm [20], the study conduct the feature selection manually. First of all, removing variables with unique value, then deleting the obviously irrelevant variables with label, i.e., URL (Uniform Resource Locator) and ID (Identity) and variables generated during the process of loans; at last, remaining only one variable among large number of highly correlated variables. Finally, 48 variables (see Appendix A for details) and 1 label (default or non-default) are extracted, in which 0 and 1 represent bad and good customers, respectively.

The dataset is divided into two groups, training set (80% of samples) and testing set (20% of samples), which are used to train the model and to evaluate the performance, respectively. Notably, to ensure comparability across different experiments, the training set and testing set are the same across different classifiers.

3.3. Baseline Classifiers

The baseline models proposed in this paper are based on the widespread applications in research on credit scoring [22,40–42]. In this sub-section, we will give an outline of the individual classifiers employed in this study.

3.3.1. Neural Network (NN)

NN is one of the most famous machine learning algorithms for classification, and backpropagation neural network (BPNN) is the most successful neural network algorithm which is utilized to train the NN in this study. In the BPNN algorithm, the input passes through the input layer and the feedforward calculation of hidden layer to the output layer, then capturing errors between output and the label. These errors are then propagated back to the input layer through the hidden layer, during which the algorithm adjusts the connection weight and excitation threshold to reduce the errors. The whole process adjusts the value of each parameter according to the direction of gradient, and executes several rounds until the training error reduces to the predetermined target range.

3.3.2. Logistic Regression (LR)

Linear discriminant analysis is composed of weight vector $\boldsymbol{\omega}$ and deviation term b. Given a sample $\boldsymbol{x}$, the class label y is predicted according to the following formula:

$$y = \mathrm{sign}\Big(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x} + b\Big). \tag{10}$$

LR employs a logistic function which transforms linear probabilities into logit and assumes that the target y is a member of the set $\{0, 1\}$, it is a suitable algorithm for classification [43] rather than regression problems. Take the binary problem as an example, assuming that positive class is marked as 1, otherwise, 0. The key assumption in LR is that the positive class probability of the sample can be expressed as

$$P\Big(y = 1\Big|\boldsymbol{x}\Big) = \mathrm{sig}\Big(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}\Big). \tag{11}$$

$\mathrm{sig}()$ is the sigmoid function and defined as

$$\mathrm{sig}(t) = \frac{1}{1 + \exp(-t)}. \tag{12}$$

LR is very popular in many practical applications, such as in the data processing of bank and financial institutions, since it returns a probability.

3.3.3. Decision Tree (DT)

In the area of credit scoring challenge, DT is also a strategy widely used for classification because it is close to human reasoning and is easy to understand. The idea behind the DT is to summarize a series of decision tree rules from a dataset with features and labels by splitting nodes about specific features, and use a tree diagram to present the algorithm of these rules. A new sample can be simply classified by using the existing decision tree and the basic principle is to match the corresponding features and related conditions continuously until reached a leaf node; in this way, the leaf node of the class label can be used as the sample. The computational complexity of the decision tree is low, especially judging the new data.

3.3.4. Naïve Bayes (NB)

Bayesian classification algorithm which is based on Bayesian theorem is a probabilistic classification method of statistics. Naïve Bayesian classification is one of the simplest and most widely used Bayesian classifiers. The NB algorithm can construct a probability model to estimate the posterior probability $P\left(y\middle|x\right)$ in order to classify the testing samples $\boldsymbol{x}$. It is "naïve" because Bayes classifier makes only the most primitive and simplest assumptions, which are that the features of samples are independent. Supposing an example X has the features of $a_1, a_2, \ldots, a_n$, satisfying such a formula $P(X) = P(a_1, a_2, \ldots, a_n) = P(a_1) \cdot P(a_2) \cdot \ldots \cdot P(a_n)$ means that the features are statistical independent.

3.3.5. Support Vector Machine (SVM)

SVM is a large margin classifier which is used to solve binary classification problems and is a machine learning method for classifying data by finding the optimal hyperplane [44]. Specifically, in order to maximum generalization, the SVM try to find the maximum classification margin on the training dataset as the decision boundary and separates the data into two categories (good and bad loans). The resulting framework can be used to estimate the category of new samples.

3.4. Evaluation Criteria

To ensure a thorough contrast between the ensemble learning models and the baseline classifiers, 4 metrics are considered to make a comparison: accuracy (ACC), area under the curve (AUC), Kolmogorov–Smirnov statistic (KS), and Brier score (BS). ACC is a popular metric to measure the prediction power of the model, and predictive accuracy may translate into considerable benefits for financial institutions in the future. In addition, Stefan [26] point out that the evaluation criteria can classified into three categories: those that assess discriminatory ability, correctness of the models' probability predictions, and the accuracy of the categorical predictions of models. The AUC is a kind of metric, which measures the pros and cons of predictions capability of models and the classifier performance improves following the higher AUC value. The following KS statistic is used to evaluate the risk discrimination ability of model, and it measures the difference between the accumulated difference of good and bad samples. Moreover, BS is a score function which analyzes the accuracy of the probability predictions and calculates the MSE (Mean Square Error) between the possibility predictions (between 0 and 1) and the true label (0 or 1). It can be formulated as follows:

$$\mathrm{BS} = \frac{1}{N} \sum\nolimits_{t=1}^{N} (f_t - O_t)^2, \tag{13}$$

where t represents the samples, f_t and O_t indicate the predicted score, and true label of t, respectively.

3.5. Hyper-Parameter Tuning

The performance of classifiers will be influenced by hyper-parameters directly. Classifiers in this research, such as LR, DT, NN, SVM, RF, AdaBoost, XGBoost, and LightGBM, all have a few hyper-parameters which need to be significantly modified. Therefore, we determine the parameters by using the grid search which is a typical hyper-parameter optimization method and involves an exhaustive search of a pre-specified searching space. Table 1 summarizes the searching space for several learners. Firstly, for the NN model, a BPNN is designed on the basis of one hidden layer of 4 neurons, which is determined by the trial and error process. Furthermore, the training epochs and the activation function were set as 1000 and "sigmoid", respectively. According to SVM, the RBF kernel was chosen; meanwhile, two parameters, C and gamma, are set to 1.0 and

0.04832930238571752, respectively. In RF, 127 trees are constructed and the number of attributes is set to 10, which are the two critical parameters. In relation to DT, the impurity assessment is carried out according to the Gini Diversity Index in selecting the most suitable attribute to start the tree with and the maximum depth is set to 5, minimum sample leaf is set to 1. XGBoost has various hyper-parameters, some of which can be manually determined. The detail hyper-parameters are briefly displayed. XGBoost: Learning_rate = 0.3, n_estimators = 27, objective = "binary:logistic", nthread = 2, nrounds = 10, silent = 1, gamma = 1, min_child_weights = 2, and eval_metric = "auc". Analogously, paramaters of LightGBM: Learning_rate = 0.3, n_estimators = 66, max_depth = 2, feature_fraction = 0.9, bagging_fraction = 0.6, and num_leaves = 31. Finally, for Stacking, after repeated combination experiments, two learners (RF, LightGBM) are selected as first-level classifiers. As simple linear models usually work well in second-level classifier [37]; therefore, we chose LRas second-level classifier.

Table 1. Search space of parameter settings.

Classifiers	**Searching Space**
LR	$C \in (-20, 20)$
DT	max_depth $\in (1, 10)$, min_samples_leaf $\in (1, 7)$
NN	hidden layer neuron $\in (1, 10)$
SVM	gamma $\in (-10, 20)$, $C \in (-20, 20)$
RF	n_estimators $\in (0, 200)$, max_depth $\in (1, 20)$
AdaBoost	learning_rate $\in (0.1, 1)$, n_estimators $\in (1, 20)$
XGBoost	learning_rate $\in (0.1, 1)$, n_estimators $\in (1, 200)$, gamma $\in (0, 1)$, min_child_weight $\in (0, 2)$
LightGBM	learning_rate $\in (0.1, 1)$, n_estimators $\in (1, 100)$, max_depth $\in (1, 10)$, feature_fraction $\in (0.1, 1)$, bagging_fraction $\in (0.1, 1)$

It is worth noting that NB is on the basis of the idea of estimating the posterior probability according to the prior probability of a feature in the training data, so no significant parameter tuning is performed, and the parameter of Stacking depends on the chosen level-1 and level-2 classifiers. Moreover, XGBoost, LightGBM were performed with Python library "xgboost" and "lightgbm", respectively. NN used Python library "keras" and the other classifiers were performed with Python library "sklearn".

To exclude the contingency, we employ the "5-fold" cross validation accuracy as an objective to enhance the robustness of the model and overcome the consequences of overfitting and select 80% samples as the training set, the remaining 20% samples as the testing set to assess the performance of this classifier. The results and analysis of the experiment will be discussed in the following section.

4. Empirical Results

4.1. Classification Results

Our objective in this empirical study is to make a comparative performance evaluation among ensemble learnings, i.e., AdaBoost, random forest, Stacking, and two novel models, XGBoost and LightGBM, contrasting with five individual models, i.e., ANN, SVM, LR, DT, and NB. The comparison is based on five aspects, namely, accuracy (ACC), area under the curve (AUC), Kolmogorov–Smirnov statistic (KS), Brier score (BS), and model operating time. Table 2 presents the results of ensemble learning and traditional individual learners (bolded numbers are the best performance across different metrics), Table 3 shows the model operating time. With regards to ACC, RF achieves the best performance (81.05%), XGBoost and LightGBM are close challengers. They improve the performance of initial loan dataset (78.93%) by 2.12%, 0.54%, and 0.54%, respectively. The AUC of RF is 85.80%, which indicates strong discriminatory ability between classes, LightGBM and XGBoost place second and third with 72.69% and 72.55%, respectively. According to KS, RF also achieves the best, which represents the best discriminatory ability for default and non-default customers, and Stacking places second. The performance of RF is still the best in BS, while other models show slightly difference.

Table 2. Classifier results in terms of 4 measures.

Performance Measure	**Traditional Individual Classifier**					**Ensemble Learning**				
	NN	SVM	LR	DT	NB	AdaBoost	RF	XGboost	LightGBM	Stacking
ACC	0.7914	0.7919	0.7915	0.7900	0.6828	0.7861	**0.8105**	0.7947	0.7947	0.7919
AUC	0.7201	0.6589	0.7199	0.6952	0.6242	0.6952	**0.8580**	0.7255	0.7269	0.7337
KS	0.3299	0.2445	0.3309	0.2969	0.1994	0.2978	**0.5507**	0.3411	0.3438	0.3643
Brier Score	0.1497	0.1563	0.1495	0.1536	0.2644	0.1574	**0.1260**	0.1482	0.1479	0.1476

Table 3. Operating time across different classifiers (unit: second).

Classifier	**NN**	**LR**	**DT**	**NB**	**SVM**
Time (s)	118.7610	1.1919	2.0758	0.2264	696.1673
Classifier	**RF**	**AdaBoost**	**XGBoost**	**LightGBM**	**Stacking**
Time (s)	0.8949	2.7414	3.5163	1.8071	28.5911

On the whole, ensemble learning models outperform traditional models, except AdaBoost. With regards to individual classifiers, all performance considered, LR has performed the best over most of evaluation metrics and it is always a popular credit scoring model. NN becomes the second-best classifier as time-consuming is a subtractive component. SVM and DT are the third best, which can be partially explained that DT has a high variability [38] and for SVM, there is not a standard criterion for choosing a suitable kernel function. NB is the worst-performing individual classifier, the reason may be that the prediction result is based on prior probability; thus, the assumed prior model may affect the final result.

A plausible fact for credit scoring that ensemble learning can compete really well against individual classifiers, it is also interesting that AdaBoost has got worse than some individual learners. in addition to that, RF (Bagging DT) has performed excellent results, the application of Bagging has resulted in a significant improvement for DT: bagging DT; that is, RF (0.8105, 0.8580, 0.5507, and 0.1260) outperform base learner DT (0.7900, 0.6952, 0.2969, and 0.1536) in term of all four performance measures, to some extent, it shows that ensemble learning is superior to single models.

In the light of above experimental findings, we can obtain the following conclusions:

(1) Compared to traditional individual learners, the ensemble learning has brought a few improvements, except for AdaBoost, this is not consistent with our previous hypothesis.

(2) The reason for the poor performance of AdaBoost may be that the model over-emphasize examples that are noise due to the overfitting of the training data set [33]. Thus RF (Bagging DT) is a relative better choice for credit scoring and this is consistent with prior research.

(3) NN and SVM are both popular credit scoring technique which contain time-consuming training process [45], especially for the data sets of more than 50,000 examples. Thus, they can be excluded in research under the condition of computer hardware equipment is insufficient or not expect to consume too much time, RF, XGBoost, LightGBM, and LR should be the ideal choices for financial institutions in terms of credit scoring and it also provides a reference for future research.

4.2. Receiver Operating Characteristic (ROC) Curve Analysis

ROC curve derived from signal processing has also been used to study the prediction accuracy of X to Y. In the ROC curve, X-axis is false positive rate (FPR), Y-axis is referred to as true positive rate (TPR). The premise of using ROC curve is that the classification algorithm can output continuous values. The ROC curve is drawn by changing the classification threshold from maximum to minimum, from determining all samples as negative to all positive. Then, different classification results can be obtained, corresponding to different values of FPR and TPR. AUC is the area under the ROC curve which measures the quality of the algorithm. In general, the AUC of a normal classifier is between 0.5 and 1, below 0.5 means the performance of classifier is less than random guessing.

Figure 10 shows the ROC curve of each model. It can be seen that the RF ROC curve lies above all the other curves across all threshold values, it also has a convex circle-like form relative to other curves, which implies lower rates of false negative and false positive errors. This means that RF is the best for all the values of sensitivity and specificity. The Stacking ROC curve lies below RF, but for almost all threshold it is above all the other learners, which means that it is the second-best performer. Moreover, the performance of XGBoost and LightGBM are close behind with only a slight difference between the two. Although it is difficult to determine a preferred individual technique from the curves, NN and LR are the most promising candidates. Furthermore, DT, AdaBoost, SVM, and NB ROC curves are skewed, which means that the rise in specificity led to the significant decrease in sensitivity.

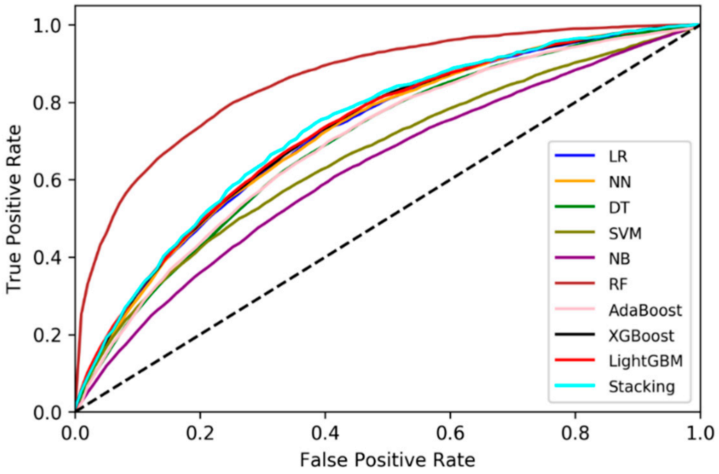


Figure 10. (Receiver Operating Characteristic) ROC curve comparing ensemble learning and individual classifiers.

5. Conclusions

The main insight of this paper is an experimental supplement in the debate on the preferred models for predicting credit risk. A comparative evaluation of five ensemble algorithms, i.e., RF, AdaBoost, XGBoost, LightGBM, and Stacking, and five traditional individual learners, i.e., NN, LR, DT, SVM, and NB, is performed. These experiments all have been implemented to a real-world credit dataset, which is from Lending Club in the United States. Experimental results reveal that the ensemble learning produces obviously higher performance than individual learners, except for AdaBoost, it is not consistent with previous hypothesis, and this is probably because the model over-emphasize examples that are noise owing to the overfitting of the training data [33]. Additionally, RF achieves the best results in five performance criteria, i.e., ACC, AUC, KS, BS, and model operating time, and XGBoost and LightGBM are close challengers. Among five base learners, LR outperforms the other classifiers over the most of evaluation metrics. Moreover, the time cost has been considered in the work, NN and SVM are time-consuming, the operating time of Stacking is up to the choice of the base models. On the whole, RF, XGBoost, LightGBM, and LR might be the first and best choice for financial institutions in the period of credit scoring under the constraint of definite time and hardware.

There are some limitations in this work. Firstly, the conclusion lies on the limited available data, in the future, more credit scoring data structures from different countries will be collected to consummate the conclusions of the research. Secondly, another limitation is only one parameter tuning method is used in this study. RF is an efficient method if the hyper-parameters are fine-tuned, indicating that hyper-parameter tuning is a major procedure during the constructing of credit scoring models; therefore, a more parameter tuning approach will be introduced in the future research, i.e., Bayesian hyper-parameter optimization. Finally, the costs of different classification are considered as the same in this research. The cost of misclassifying a default borrower may be larger than that of misclassifying a good one; therefore, the imbalanced misclassification cost will be the next research direction.

Author Contributions: Y.L. developed the idea of performance comparison of ensemble learning, performed the construction process of classifiers, software programming, and prepared the initial draft of the paper. W.C. oversaw all aspects of the research and revised it. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China [No. 71373173], National Social Science Found of China [No. 14ZDB135], and the Major Research Plan of Social Science Found of Tianjin Municipal Education Commission [No. 2019JWZD39].

Acknowledgments: The authors would like to thank the editors and anonymous reviewers for their helpful comments and suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Overview of Variable Definition for Lending Club Dataset

Table A1. Collected variables of this study.

Variable	Definition	Variable	Definition
Target variable			
Loan status	Whether the borrower is default or not		
Loan characteristic			
Fund amount	The total amount committed to the loan	Initial list status	The initial listing status of the loan.
Term	Number of payments on the loan. Values can either be 36 or 60	collections in 12 months	Number of collections in 12 months excluding medical collections
Interest rate	Interest Rate on the loan	Application type	Whether the loan is an individual application or a joint application with two co-borrowers
Purpose	Purpose of loan request, 13 purposes included		
Borrowers' creditworthiness			
Delinquency-2 years	The number of 30+ days past-due incidences of delinquency for the past 2 years	Revolving credit ratio	Total revolving high credit/credit limit
Inquires last 6 months	The number of inquiries in past 6 months	Trades last 24 months	Number of trades opened in past 24 months.
Open credit lines	The number of open credit lines.	Buying credits	Total open to buy on revolving bankcards
Public record	Number of derogatory public records	Ratio of high credit/credit limit	Ratio of total current balance to high credit/credit limit for all bankcard accounts
Revolving balance	Total credit revolving balance	Charge-offs last 12 months	Number of charge-offs within 12 months
Revolving line utilization rate	The amount of credit the borrower is using relative to all available revolving credit	Past-due amount	The past-due amount owed for the accounts on which the borrower is now delinquent
Total credit lines	The total number of credit lines currently	Months bank account opened	Months since oldest bank installment account opened
Months old revolving account opened	Months since oldest revolving account opened	Months recent revolving account opened	Months since most recent revolving account opened
Months recent account opened	Months since most recent account opened	Mortgage accounts	Number of mortgage accounts

Table A1. *Cont.*

Variable	Definition	Variable	Definition
Months recent bankcard account	Months since most recent bankcard account opened	Number of overdue accounts	Number of accounts ever 120 or more days past due
Active accounts	Number of currently active bankcard accounts	Revolving trades	Number of currently active revolving trades
Non-default record	Number of satisfactory bankcard accounts	Number of bankcard	Number of bankcard accounts
Installment accounts	Number of installment accounts	Number of open revolving accounts	Number of open revolving accounts
Number of revolving accounts	Number of revolving accounts	revolving trades with balance >0	Number of revolving trades with balance >0
Non-default accounts	Number of satisfactory accounts	Accounts past due last 24 months	Number of accounts 90 or more days past due in last 24 months
Accounts opened past 12 months	Number of accounts opened in past 12 months	Percent of non-delinquent trades	Percent of trades never delinquent
Percentage of limit bankcard accounts	Percentage of all bankcard accounts > 75% of limit.	Bankruptcies	Number of public record bankruptcies
Borrowers' solvency			
Employment length	Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years.	Verification status	The status of income verification. Verified, source verified or not verified
Home ownership	The home ownership status provided by the borrower during registration or obtained from the credit report. Our values are: rent, own, mortgage, other	Address state	The state provided by the borrower in the loan application
Annual income	The self-reported annual income provided by the borrower during registration.	DTI	Debt to income ratio
Hardship flag	Flags whether or not the borrower is on a hardship plan		

References

1. World Bank. *Global Economic Prospects: Heightened Tensions, Subdued Investment*; World Bank Group: Washington, DC, USA, 2019; ISBN 9781464813986.
2. Huang, C.L.; Chen, M.C.; Wang, C.J. Credit scoring with a data mining approach based on support vector machines. *Expert Syst. Appl.* **2007**, *33*, 847–856. [CrossRef]
3. Hand, D.J.; Henley, W.E. Statistical classification methods in consumer credit scoring: A review. *J. R. Stat. Soc. Ser. A Stat. Soc.* **1997**, *160*, 523–541. [CrossRef]
4. Wang, G.; Hao, J.; Ma, J.; Jiang, H. A comparative assessment of ensemble learning for credit scoring. *Expert Syst. Appl.* **2011**, *38*, 223–230. [CrossRef]
5. Beaver, W.H. Financial ratios as predictors of failure. *J. Account. Res.* **1966**, *4*, 71–111. [CrossRef]
6. Altman, E.I. Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *J. Financ.* **1968**, *23*, 589–609. [CrossRef]
7. Orgler, Y.E. A credit scoring model for commercial loans. *J. Money Credit Bank.* **1970**, *2*, 435–445. [CrossRef]
8. Grablowsky, B.J.; Talley, W.K. Probit and discriminant functions for classifying credit applicants-a comparison. *J. Econ. Bus.* **1981**, *33*, 254–261.
9. Eisenbeis, R.A. Pitfalls in the application of discriminant analysis in business, finance, and economics. *J. Financ.* **1977**, *32*, 875–900. [CrossRef]
10. Desai, V.S.; Crook, J.N.; Overstreet, G.A., Jr. A comparison of neural networks and linear scoring models in the credit union environment. *Eur. J. Oper. Res.* **1996**, *95*, 24–37. [CrossRef]
11. West, D. Neural network credit scoring models. *Comput. Oper. Res.* **2000**, *27*, 1131–1152. [CrossRef]
12. Atiya, A.F.; Parlos, A.G. New results on recurrent network training: Unifying the algorithms and accelerating convergence. *IEEE Trans. Neural Netw.* **2000**, *11*, 697–709. [CrossRef] [PubMed]
13. Verikas, A.; Gelzinis, A.; Bacauskiene, M. Mining data with random forests: A survey and results of new tests. *Pattern Recognit.* **2011**, *44*, 330–349. [CrossRef]
14. Hsieh, N.-C.; Hung, L.-P. A data driven ensemble classifier for credit scoring analysis. *Expert Syst. Appl.* **2010**, *37*, 534–545. [CrossRef]
15. Ma, X.; Sha, J.; Wang, D.; Yu, Y.; Yang, Q.; Niu, X. Study on a prediction of P2P network loan default based on the machine learning LightGBM and XGboost algorithms according to different high dimensional data cleaning. *Electron. Commer. Res. Appl.* **2018**, *31*, 24–39. [CrossRef]
16. Zhu, Y.; Xie, C.; Wang, G.J.; Yan, X.G. Comparison of individual, ensemble and integrated ensemble machine learning methods to predict China's SME credit risk in supply chain finance. *Neural Comput. Appl.* **2017**, *28*, 41–50. [CrossRef]
17. Sagi, O.; Rokach, L. Ensemble learning: A survey. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2018**, *8*, 1–18. [CrossRef]
18. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
19. Liang, W.; Luo, S.; Zhao, G.; Wu, H. Predicting hard rock pillar stability using GBDT, XGBoost, and LightGBM algorithms. *Mathematics* **2020**, *8*, 765. [CrossRef]
20. Xia, Y.; Liu, C.; Liu, N. Cost-sensitive boosted tree for loan evaluation in peer-to-peer lending. *Electron. Commer. Res. Appl.* **2017**, *24*, 30–49. [CrossRef]
21. Ala'raj, M.; Abbod, M.F. Classifiers consensus system approach for credit scoring. *Knowl.-Based Syst.* **2016**, *104*, 89–105. [CrossRef]
22. Li, Y.; Chen, W. Entropy method of constructing a combined model for improving loan default prediction: A case study in China. *J. Oper. Res. Soc.* **2019**, 1–11. [CrossRef]
23. Barboza, F.; Kimura, H.; Altman, E. Machine learning models and bankruptcy prediction. *Expert Syst. Appl.* **2017**, *83*, 405–417. [CrossRef]
24. Alazzam, I.; Alsmadi, I.; Akour, M. Software fault proneness prediction: A comparative study between bagging, boosting, and stacking ensemble and base learner methods. *Int. J. Data Anal. Tech. Strateg.* **2017**, *9*, 1. [CrossRef]

25. Jhaveri, S.; Khedkar, I.; Kantharia, Y.; Jaswal, S. Success prediction using random forest, catboost, xgboost and adaboost for kickstarter campaigns. In Proceedings of the 3rd International Conference Computing Methodologies and Communication (ICCMC), Erode, India, 27–29 March 2019; pp. 1170–1173.
26. Lessmann, S.; Baesens, B.; Seow, H.-V.; Thomas, L.C. Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *Eur. J. Oper. Res.* **2015**, *247*, 124–136. [CrossRef]
27. Brown, I.; Mues, C. An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Syst. Appl.* **2012**, *39*, 3446–3453. [CrossRef]
28. Saia, R.; Carta, S. Introducing a Vector Space Model to Perform a Proactive Credit Scoring. In Proceedings of the International Joint Conference on Knowledge Discovery, Knowledge Engineering, and Knowledge Management, Porto, Portugal, 9–11 November 2016; Springer: Berlin/Heidelberg, Germany, 2018; pp. 125–148.
29. Bhattacharyya, S.; Jha, S.; Tharakunnel, K.; Westland, J.C. Data mining for credit card fraud: A comparative study. *Decis. Support Syst.* **2011**, *50*, 602–613. [CrossRef]
30. Wolpert, D.H. Stacked generalization. *Neural Netw.* **1992**, *5*, 241–259. [CrossRef]
31. Breiman, L. Bagging predictors. *Mach. Learn.* **1996**, *24*, 123–140. [CrossRef]
32. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
33. Freund, Y.; Schapire, R.E. Experiments with a New Boosting Algorithm. In Proceedings of the 13th International Conference om Machine Learning, Bari, Italy, 3–6 July 1996; pp. 148–156.
34. Yuan, X.; Abouelenien, M. A multi-class boosting method for learning from imbalanced data. *Int. J. Granul. Comput. Rough Sets Intell. Syst.* **2015**, *4*, 13–29. [CrossRef]
35. Rao, H.; Shi, X.; Rodrigue, A.K.; Feng, J.; Xia, Y.; Elhoseny, M.; Yuan, X.; Gu, L. Feature selection based on artificial bee colony and gradient boosting decision tree. *Appl. Soft Comput. J.* **2019**, *74*, 634–642. [CrossRef]
36. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.Y. LightGBM: A highly efficient gradient boosting decision tree. *Adv. Neural Inf. Process. Syst.* **2017**, *2017*, 3147–3155.
37. Witten, I.H.; Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed.; Morgan Kaufmann: San Francisco, CA, USA, 2005.
38. Xia, Y.; Liu, C.; Da, B.; Xie, F. A novel heterogeneous ensemble credit scoring model based on bstacking approach. *Expert Syst. Appl.* **2018**, *93*, 182–199. [CrossRef]
39. Kennedy, K.; Namee, B.M.; Delany, S.J. Using semi-supervised classifiers for credit scoring. *J. Oper. Res. Soc.* **2013**, *64*, 513–529. [CrossRef]
40. Ala'raj, M.; Abbod, M.F. A new hybrid ensemble credit scoring model based on classifiers consensus system approach. *Expert Syst. Appl.* **2016**, *64*, 36–55. [CrossRef]
41. Louzada, F.; Ara, A.; Fernandes, G.B. Classification methods applied to credit scoring: Systematic review and overall comparison. *Surv. Oper. Res. Manag. Sci.* **2016**, *21*, 117–134. [CrossRef]
42. Xiao, H.; Xiao, Z.; Wang, Y. Ensemble classification based on supervised clustering for credit scoring. *Appl. Soft Comput.* **2016**, *43*, 73–86. [CrossRef]
43. Siddique, K.; Akhtar, Z.; Lee, H.; Kim, W.; Kim, Y. Toward Bulk Synchronous Parallel-Based Machine Learning Techniques for Anomaly Detection in High-Speed Big Data Networks. *Symmetry* **2017**, *9*, 197. [CrossRef]
44. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
45. Abellán, J.; Castellano, J.G. A comparative study on base classifiers in ensemble methods for credit scoring. *Expert Syst. Appl.* **2017**, *73*, 1–10. [CrossRef]