PLOS ONE

RESEARCH ARTICLE

Large Unbalanced Credit Scoring Using Lasso-Logistic Regression Ensemble

Hong Wang, Qingsong Xu, Lifeng Zhou*

School of Mathematics & Statistics, Central South University, Changsha, Hunan, China

* lfzhou@csu.edu.cn

Abstract

Recently, various ensemble learning methods with different base classifiers have been proposed for credit scoring problems. However, for various reasons, there has been little research using logistic regression as the base classifier. In this paper, given large unbalanced data, we consider the plausibility of ensemble learning using regularized logistic regression as the base classifier to deal with credit scoring problems. In this research, the data is first balanced and diversified by clustering and bagging algorithms. Then we apply a Lasso-logistic regression learning ensemble to evaluate the credit risks. We show that the proposed algorithm outperforms popular credit scoring models such as decision tree, Lasso-logistic regression and random forests in terms of AUC and F-measure. We also provide two importance measures for the proposed model to identify important variables in the data.

Citation: Wang H, Xu Q, Zhou L (2015) Large Unbalanced Credit Scoring Using Lasso-Logistic Regression Ensemble. PLoS ONE 10(2): e0117844. doi:10.1371/journal.pone.0117844

Academic Editor: Frank Emmert-Streib, Queen's University Belfast, UNITED KINGDOM

Received: August 5, 2014

Accepted: December 31, 2014

Published: February 23, 2015

Data Availability Statement: Data are available from Kaggle.com "Give me some credit competition" at http://www.kaggle.com/c/GiveMeSomeCredit.

Funding: The authors received no specific funding for this work.

Competing Interests: The authors have declared that no competing interests exist.

Introduction

Credit scoring analyzes the characteristics and performance of past loans and predicts the delinquency probability of loan applicants in the near future based on age, financial flows and repayment records and other characteristics [1]. The ability to discriminate good customers from bad ones is of extreme importance for lending companies or organizations and an improvement in prediction accuracy of even a percent may indicate a huge gain in profit [2].

The pioneer work to discriminate between good and bad loans using statistical method was due to [3]. Since then and particularly after the invention of credit cards in the late 1960s, various state-of-the-art machine learning and sophisticated statistical methods have been proposed to tackle the classification issue in application scoring. Among them, the most popular ones are linear discriminant analysis [4], logistic regression [5], neural networks [6, 7], decision trees [8], and support vector machines [9]. Comprehensive reviews of these methods can be found in [10–14].

Due to the success of ensemble methods in various classification tasks [15–17], ensemble algorithms have also found a place in credit scoring modeling and experimental results have revealed that ensemble methods can substantially improve the performance [18–21]. One of the first works on credit scoring using ensemble algorithms is due to [22]. In their work, an ensemble of overlapping classification trees was introduced and a Markov Chain Monte Carlo procedure was developed to choose different tree structures. Later on, a neural network ensemble

algorithm using negative correlation was proposed in [23]. However, in terms of average prediction accuracy [24], multiple neural networks classifiers may not outperform a single best neural networks classifier in many cases. Beside neural networks, support vector machines (SVM) are frequently chosen as the base learners in credit scoring ensemble algorithms [18, 25]. As is common to all SVM methods, SVM has a difficulty in illustrating the variables' importance, and suffers from the overfitting problem if its parameters are not properly chosen. In addition, it needs much time and efforts to construct a best hyper-plane, particularly when facing large datasets. Logistic regression, one of the most popular statistical methods in credit scoring [26], is also used as a benchmark base learner in some aforementioned studies. However, only classical logistic regression models have been applied so far in application credit scoring.

A very common situation in real-life credit scoring applications is that the data collected are usually highly unbalanced or skewed, i.e. most examples in the data belong to one class (the majority or the negative class) and the rest belong to the other class (the minority or the positive class). Standard learning algorithms are in favor of the majority class, and they will usually show poor performance on the minority class examples. An ideal credit scoring model should perform equally well both on the minority class and majority class. Compared to a large number of studies in credit scoring, there is relatively only a little research focusing on scoring imbalanced data. Cost-sensitive learning and re-sampling approaches are two general methods applied to alleviate the class imbalance problem. Cost-sensitive learning algorithms which incorporate costs into certain classifiers usually assume that the misclassification costs are known beforehand and require special knowledge of the classifiers themselves [27]. However, these two conditions are not easily satisfied in practice. Hence, most credit scoring algorithms have adopted a re-sampling approach [21, 28–32]. In [21, 28], simple over-sampling or under-sampling approaches are applied and the experiments in [29, 30] provide evidence that over-sampling is superior to under-sampling in terms of accuracy. The results in [31] also reveal that over-sampling outperforms under-sampling in most cases, especially with the logistic regression model. In [32], a particular version of bagging called subagging which is suitable for imbalanced learning is applied, however, a parameter controlling the proportion of positive and negative examples needs to be specified beforehand. Due to the complexity of the problem, imbalanced data classification is not fully solved in aforementioned studies and are still in need of further refinement and research.

On the other hand, big data containing millions or even billions of customers application records have become a commonplace for major credit institutions. If all the data are used to train the classification model, it would be very inefficient and time consuming. How to exploit efficiently the huge amount of data to train a satisfactory model within reasonable time and common computing facilities has become more and more imperative for banks and other credit companies.

In this paper, to demonstrate the effeteness of ensemble learning and Lasso-logistic regression(LLR) in tackling the large unbalanced data classification problem in credit scoring, a Lasso-logistic regression ensemble(LLRE) learning algorithm is proposed. First, majority class examples in the dataset are clustered into a number of subgroups, then a number of balanced and "smaller" training sets are formed by each majority subgroup and a bagged version of the whole minority data. Next, base learners namely, Lasso-logistic regression models are trained with these smaller training sets. Different from traditional cross-validation methods [33], the λ parameter in Lasso algorithm is determined by the current so-called OBS data. The OBS data are then used to evaluate the base classifiers' performance. Finally, the Lasso-logistic models are combined together into the ensemble using a weighted average scheme. As the AUC(Area Under the Curve) metric is independent of class distribution and is more suitable than

accuracy in the scenario of unbalanced learning [9, 34], we choose AUC as the evaluation metric in our experiment. F-measure, a tradeoff between precision and recall, is also chosen. Results show that the proposed algorithm outperforms decision tree, Lasso-logistic regression and popular ensemble learning algorithms such as random forests in terms of both AUC and F-measure.

Besides detecting the default customers among all loan applicants, financial institutions also have strong interest in determining which characteristics of the applicants are the most significant ones that affecting the default probability. Thus, as a minor contribution, we provide two kind of measures in LLRE for evaluating the variable importance. The first so-called LLR-occurrence measure is based on the variable selection feature of the Lasso algorithm. The second mean AUC decrease measure, inspired by [35], depends on AUC values before and after variable permutations. We apply these two measures for evaluating the top ranking variables that are important for credit scoring.

Background

In this section, we first give a brief introduction of Lasso-logistic regression in the credit scoring scenario and then we discuss a particular ensemble learning-the bagging algorithm. Later on, we shall develop our credit scoring ensemble algorithm using both Lasso-logistic regression and bagging.

Lasso-logistic regression

Application credit scoring determines the probability that a credit applicant will default on his/her credit obligation. From a statistical learning and data mining point of view, application credit scoring can be viewed as a binary classification problem. Suppose that we have credit scoring data $(\mathbf{x}^i, y_i)$, $i \in 1, 2, \cdots, n$, where $\mathbf{x}^i = (x_{i1}, x_{i2}, \cdots, x_{ip})$ are predictor variables of applicants and y_i are class labels(binary responses,1 for default and 0 for non-default). We want to determine the conditional probability $p(\mathrm{y}|\mathrm{x})$ of a specific applicant belonging to a class(0 or 1), given the variables of that credit applicant. Rather than modeling the response y directly, logistic regression computes the probability that y belongs to a particular class label via a linear function of features x:

$$\log \frac{\mathrm{Prob}(Y = 1 \mid x)}{\mathrm{Prob}(Y = 0 \mid x)} = \beta_0 + \mathrm{x}^T \beta \tag{1}$$

where β_0 denotes the intercept, $\beta = (\beta_1, \cdots, \beta_p)$ denotes the linear coefficients, and $\mathrm{Prob}(Y = 1|x)$, $\mathrm{Prob}(Y = 0|x)$ denote the conditional probabilities of the class labels 1, 0 respectively.

A maximum likelihood approach is commonly used in calculating the coefficients and the log-likelihood can be written:

$$\begin{aligned} l(\beta_0, \beta) &= \sum_{i=1}^{n} \Big\{ y_i \log \mathrm{Prob}(Y = 1; \beta) \\ &\quad + (1 - y_i) \log(1 - \mathrm{Prob}(Y = 1; \beta)) \} \\ &= \sum_{i=1}^{n} \{ y_i (\beta_0 + x_i^T \beta) \\ &\quad - \log(1 + e^{\beta_0 + x_i^T \beta}) \Big\} \end{aligned} \tag{2}$$

The above logistic regression model can be further extended into Lasso logistic regression model by imposing an L_1 constraint on β parameters [33, 36] and the problem is to minimize

the negative log-likelihood function with the penalty term

$$\sum_{i=1}^{n}\left\{\log(1+e^{\beta_0+x_i^T\beta)}-y_i(\beta_0+x_i^T\beta)\right\}+\lambda\sum_{j=1}^{p}|\beta_j| \quad (3)$$

Due to the above constraint, making λ sufficiently large will cause some coefficients in β become zero. Depending on the value of λ, a Lasso model can include any number of variables. In this way, both shrinkage and feature (variable) selection are done simultaneously and it is also this property that makes Lasso generally much easy to interpret and a very popular algorithm.

To date, there is no reported research using the latest regularized logistic regression models such as Lasso as the base learning algorithm, tough Lasso itself is efficient in dealing with large datasets and convenient for screening variables [33].

Ensemble learning

Ensemble learning is a kind of machine learning paradigm in which multiple models, such as decision trees, neural networks and SVM, are combined together to solve a particular problem [37]. Typical ensemble methods including AdaBoost [38], bagging [39], random forests [40] and gradient boosted machines [41]. And all these methods encourage diversity of the base learners to some extent to compensate individual errors and reach a better expected performance.

Bagging, achieving diversity by bootstrap aggregating the training data, improves the predication accuracy and also helps to reduce variance and alleviates the overfitting problem. Bagging was first implemented in CART(Classification And Regression Tree) [39] and usually applied with unstable decision trees methods, however, it can work with any type of base learners [42–44]. One of bagging's features is that it supports parallel computing, which is an important advantage in training large data.

A typical bagging algorithm [39] is presented in the following Table 1:

In each bootstrap training set, about 37% of the original data examples will be left out(also called Out Of Bag or OOB data) and this can be used as the validation dataset. It is noticed that bagging is particularly useful when there is a limited data [20].

The Proposed Lasso Ensemble Algorithm

In this section, we first present the idea on how to balance the data and then we describe how to generate and aggregate the diversified base learners within the ensemble. Later on, we

Table 1. Algorithm 1. Bagging.

1: Let T be the set of n training examples $(\mathbf{x}^i, y_i)$, $i \in 1, 2, \cdots, n$.
2: B is the number of base learners and L the base learning algorithm.
3: for($i = 0; i < B; i++$){
4: Create a bootstrapped training set T_i of size n by sampling with replacement.
5: Learn a specific base learner $L_i(x, y)$ on T_i by using L.
6: }
7: The final learning algorithm C is the ensemble of all base learners $\{L_i\}$ and a test example x* is classified by using a simple majority voting method: $y^* = arg \max_y \sum_{L_i \in C} L_i(x^*, y)$

doi:10.1371/journal.pone.0117844.t001

provide two kinds of measures to assess variable importance. Finally, we describe the proposed algorithm.

Balancing the data

In real credit scoring applications, most would-be-default applicants have either given up the applications or been rejected immediately after a first screening, thus in most collected data, only a small portion of the applications are default(minority) ones. However, as we do not have a good understanding of who would be defaulted in the future, we need a lot of non-default (majority) examples to help the classification. Usually, the non-default examples are collected to complement the default ones and thus these non-default examples are chosen for various reasons and may be from different sub-populations.

As shown in the upper left panel of Fig. 1, majority data greatly outnumber minority data in the training set and it is non-trivial to find a satisfactory classification model. However, when majority examples are or seem to be from different populations (in this examples, majority data are generated from three different distributions), it is easy to train a sub-model or base learner with high classification accuracy on each distribution of the majority data and all the minority data. From the upper right, lower-left, and lower-right panels of Fig. 1, we can see that when only one sub-group majority examples are considered, it is rather easy to find a satisfactory decision boundary between this sub-group of majority data and the whole minority data.

Thus, if we are given a large dataset, we can divide the available majority examples into a number (k) of subgroups according to some certain criterion (for example, variable similarity). Then, we can choose one subgroup of majority examples and all the minority examples to form a roughly balanced sub-training set. As a training set containing 1500–2000 examples of each class will be sufficient for building a near-optimal classifier in credit coring [29], the new sub-training set should be large enough for a classifier to maintain modest classification performance. Base classifiers built upon these balanced sub-training sets should have better performance on both majority and the minority class examples than classifiers trained with the original highly unbalanced dataset.

Diversifying the data

In ensemble learning, the most popular method to achieve classifier diversity is using different training datasets to train individual base classifiers [37]. This is also the approach adopted in this study. Data diversity in this paper is realized by using different sub-groups of majority examples plus bagging versions of minority examples.

First, as mentioned above, majority data can be segmented into a number (k) of sub-groups using a clustering algorithm, and the examples within each sub-group are surely quite different from data in other sub-groups. As data size may vary from sub-group to subgroup, to eliminate those with a few examples, only subgroups have sizes larger than a threshold quantile (75%) are retained. Second, in the absence of adequate minority examples, resampling techniques such as bagging can be used for drawing overlapping random subsets of the available minority data. Thus, a sub-group of majority data and a bagging version of minority data can be combined together to train a different base classifier in creating a diversified ensemble. In this way, model performance degrade due to class imbalance is alleviated and diversity of classifiers in the ensemble is increased.

In the following, for ease of explanation, the currently used balanced sub-training set, and all the training data excluding the currently used sub-training set are called Balanced Sub-training set(BS), and Out of Balanced Sub-training set (OBS) data, respectively.

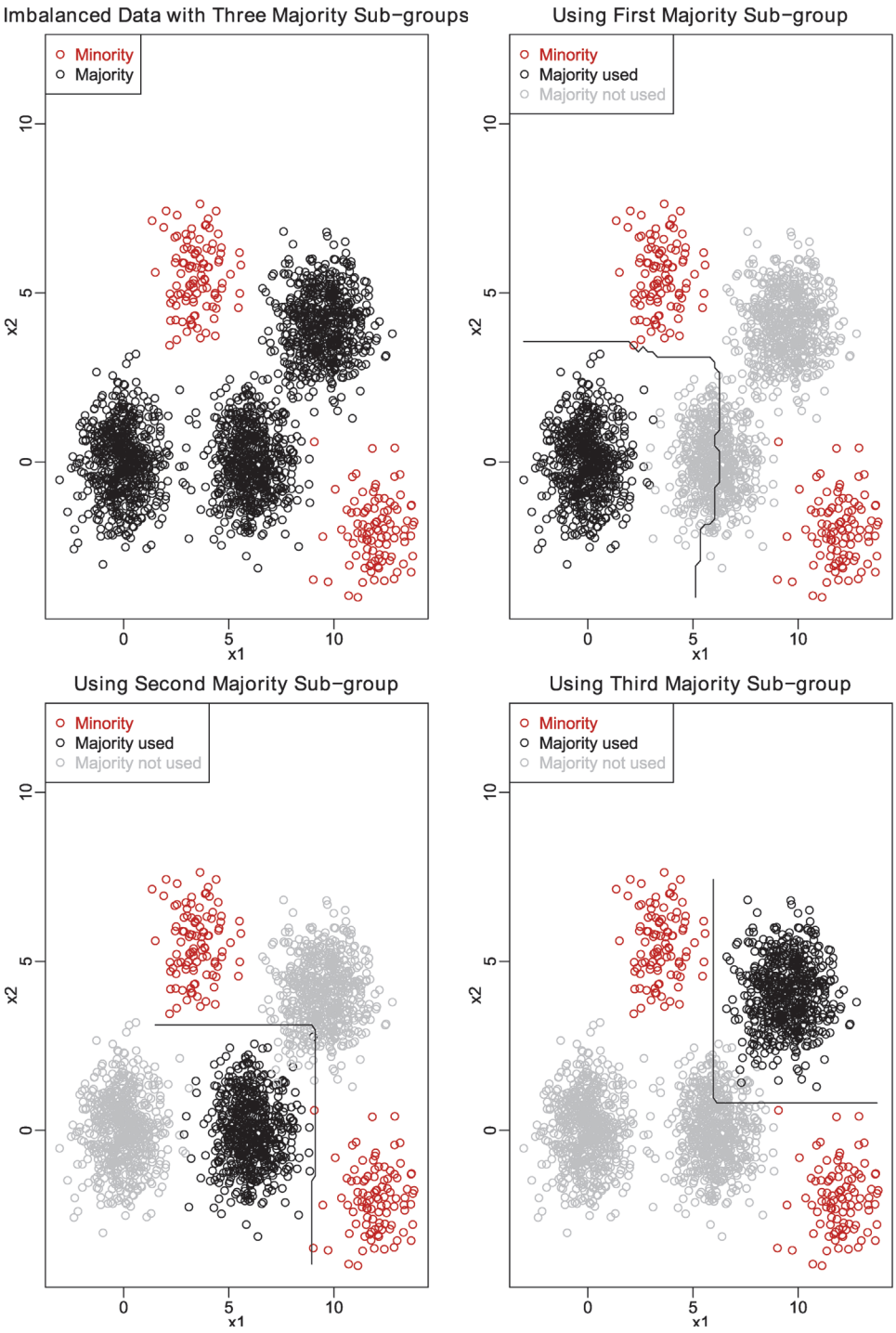

Fig 1. Classification when majority are from different populations.

doi:10.1371/journal.pone.0117844.g001

Aggregating base learners

The most popular aggregation method in ensemble algorithms is majority voting which adopts class label predicted with the most votes. However, simple majority voting does not reflect the individual confidence of the base learners and the overall predicting power might be lowered [45]. It was also proved in [46] that in most cases, weighted average is better than simple majority voting. In the proposed method, when an ensemble of classifiers is constructed, unique voting weights to each classifier in the ensemble are assigned and the final prediction is judged on the weighted average of all base learners' predictions. We also note that base classifiers with large weight values might dominate the final decision of the ensemble. Thus, to reduce the side effects of large weight values, we use the following sigmoid function as a scaling function in weight transformation:

$$w_i = \frac{1}{1 + e^{-p_i}} \tag{4}$$

where w_i denotes the weight of the i-th base classifier, and p_i the performance of classifier i on the i-th validation data-OBS_i.

Evaluating variable importance

Determining which variables in a model are its most important predictors (in ranked order) is of vital importance in the interpretation of a model. In this study, we introduce two kinds of variable important measures, namely the LLR-occurrence measure and the mean AUC decrease measure.

The LLR-occurrence measure is based on variable selection property of Lasso logistic regression. Different variables are selected automatically in building each LLR base learner in LLRE. The presence or absence of a predictor variable in the Lasso model naturally indicates whether it is closely related to the outcome variable or not. It is reasonable to evaluate the importance of different variables by ranking the variable occurrences among all iterations. A larger LLR-occurrence value will imply a more important variable and vice versa. Note that the largest possible LLR-occurrence value is the ensemble size. The variable importance for variable j in terms of LLR-occurrence is formally defined by:

$$VI_j^{occu} = \sum_{i=1}^{L} I(\beta_j^{(i)} \neq 0) \tag{5}$$

where L denotes the number of LLR model within the ensemble, $I()$ the indication function, and $\beta_j^{(i)}$ the coefficient of variable j in the i–th LLR base model.

Many ensemble learning methods such as random forests can output variable importance based on error rate [40]. However, error rate (1- accuracy) always prefers the majority class and may not be appropriate for imbalanced class problems. As AUC measure is independent of class distribution and can be used to measure variable importance when facing imbalanced classes. Thus, in this study, we introduce a new importance measure, mean decrease AUC value in base classifiers to determine variable importance. It is based on the difference between the OBS data AUC values before and after permuting the values of the variables in question. The variable importance for variable j in terms of mean AUC decrease is defined by:

$$VI_j^{auc} = \frac{100}{L} \sum_{i=1}^{L} (AUC_j - AUC_{\bar{j}}) \tag{6}$$

where AUC_j and $AUC\,j$ denotes AUC value on the current OBS data before and after the permutation of variable j. If the variable in question is not associated with the outcome, permutating its values will have no influence on the classification and hence no influence on the AUC. On the contrary, if outcome and variables are indeed associated, dropping this variable by permuting its values results in a worse classification and will lead to a decrease in the AUC value. The difference in AUC before and after randomly permuting the variable will reflect the importance level of this variable. Averaging the decrease across all the LLR base models, we will get a list of mean decrease AUC values. The higher a mean decrease AUC value is, the more important a variable will be.

The algorithm

In this study, Lasso(L1-norm penalty) is chosen as the base learner in that it is very efficient in dealing large-scale datasets and has a built-in variable selection property. However, computing the Lasso solution is a quadratic programming problem and non-trivial. In the latest related research [33, 42], the entire path of solutions with varied λ are computed using coordinate descent methods and the best λ is determined through cross-validation. Indeed, given a large dataset, we can also make use of the OBS data in each iteration as validation data to determine which value of λ is the best. The proposed Lasso-logistic regression ensemble(LLRE) algorithm is presented in Table 2:

From Table 2, we know that similar to bagging, LLRE is also a parallel algorithm and the "while" part of the algorithm can be executed concurrently. When facing large data, LLRE can make full use of the available parallel computing facilities to save training time.

Table 2. Algorithm 2. A Lasso-Logistic Regression Ensemble(LLRE) Algorithm.

1:	INPUT:
2:	$D \leftarrow$ training data; $k \leftarrow$ number of majority sub-groups
3:	OUTPUT:
4:	Ensemble of Classifiers C
5:	**procedure** LLRENS(D, k)
6:	Cluster the majority examples into k sub-groups using a clustering algorithm such as K-means and discard sub-groups whose size are less than the upper quantile (75%)
7:	The ensemble size $L \leftarrow 0.75 * k$
8:	**while** $i \leq L$ **do**
9:	Select the i–th sub-group majority data Maj_i
10:	Generate a bootstrap sample of the minority data Min_i
11:	Train a LLR model C_i on data $BS_i = Maj_i + Min_i$
12:	Evaluate C_i with all possible λs on $OBS_i = D - BS_i$ and choose λ with the best performance
13:	Record the performance p_i of C_i with the best λ on OBS_i
14:	Calculate C_i's weight w_i according to formula (4)
15:	**end while**
16:	**return** the ensemble C
17:	**end procedure**
18:	In prediction, a sample (x, y) is assigned with class label y^* according to: $$y^* = arg \max_y \sum_{C_i \in C} w_i * C_i(x, y)$$

doi:10.1371/journal.pone.0117844.t002

Experiment Results

In this section, we first describe the data set used in the experiments and then the performance metrics and statistical tests that we used. Finally, we present and discuss the experimental results.

Dataset

In this study, we want to test our LLRE algorithm's performance on large real credit scoring data. However, the only available large credit scoring dataset is from a data mining competition, publicly available on Kaggle.com. In the data, a bad customer is defined "default" (class 1) as some one would experience financial distress in the next two years as of the approval date. The size of the credit data used is 150,000 and the original variables, i.e. variables available in the above Kaggle dataset include age, debt ratio, income, number of dependents and six other personal financial variables. The vast majority (139,974) of the dataset is from class 0 and the minority (10,026) is from class 1. It is obvious that the ratio (about 14:1) between the two classes is obviously unbalanced.

Performance metrics and statistical tests

In this study, we use two measures commonly used in the unbalanced classification literature to measure the performance of all algorithms. Accuracy is inappropriate here in that when high class imbalance presents in the data, even a simple solution of classifying all default(minority) cases into the majority class will show a high and satisfactory performance value. One measure chosen in the experiments is the area under the ROC (Receiver Operating Characteristics) curve (AUC) as suggested by [9, 34] and the other measure is F-measure(the harmonic mean of precision and recall) which is also frequently used in unbalanced classification and credit scoring [47, 48].

In order to compare the performance values of different classifiers, the non-parametric Friedman test [49], which is based on the average ranked performance of the classification algorithms on each run of the dataset, is adopted. Friedman's test statistic is calculated as follows:

$$FT = \frac{12}{nm(m+1)} \sum_{j=1}^{m} \left(\sum_{i=1}^{n} r_i^j \right)^2 - 3n(m+1) \tag{7}$$

where n denotes the number of runs, m the number of classifiers, and r_i^j the rank of classifier j on the i-th run. For a large sample n (i.e. greater than 5), the statistic follows a Chi-square distribution with $m-1$ degrees of freedom: $FT \sim \chi^2(m-1)$. If the value of FT is large enough, we can reject the null hypothesis that there is no significant difference among the different classifiers and a post-hoc test such as the Nemenyi test [49] can further be applied to pinpoint where the difference lies.

In the Nemenyi test, denote by R_j the mean rank of classifier C_j on all runs of the dataset: $R_j = \frac{1}{n}\sum_{i=1}^{n} r_i^j$. Then for any two classifiers C_1 and C_2, the statistic z is calculated as follows:

$$z = \frac{R_{j1} - R_{j2}}{\sqrt{\frac{m(m+1)}{6n}}} \tag{8}$$

The performance of two classifiers is significantly different if the z value is larger than the critical difference based on the Studentized range statistic divided by $\sqrt{2}$ [49].

Results and discussions

We conduct our experiments on a system with a Pentium Dual-Core 3.20GHz CPU and 4G RAM. The proposed LLRE algorithm is implemented in the R programming language. In the experiments, training/testing sets are from a random 80% and 20% split of the Kaggle dataset.

Variables generation. In the Kaggle dataset, beside the outcome variable "SeriousDlqin2yrs", there are 10 predictor variables namely "Revolving utilization of unsecured Lines", "Age", "Number of time 30–59 days past due not worse", "Debt ratio", "Monthly income", "Number of open credit lines and loans", "Number of times 90 days late", "Number of real estate loans or lines", "Number of times 60–89 days past due not worse" and "Number of dependents". In order to create a diversified variable set for logistic regression base models, we need to generate more variables or do some variable transformation based on these original variables.

As suggested in [50], in the case of high-skewed data distribution, a log transformation for numeric variables is necessary as the logarithm function can squeeze together the larger values and stretch out the smaller values and hence make the patterns in the data more visible and interpretable. If the predictors variables contain some zeros, we may use a log transformation variant–$log(x+1)$ where x is the predictor value to ensure a similar effect [51]. This is the major technique applied in our variable generation process.

After careful study of the Kaggle data, we find some special code values(a common practice in survey data) exist for predictor variables such as "Number of time 30–59 days past due not worse". These codes do not denote the expected "Number of time" but really mean something else not mentioned by the data provider. We also notice that there are quite a few missing(NA) values in the Kaggle data and these NA values generally denote some certain financial information [32]. Consequently, replacing these NA values by a mean or median imputation could degrade classifiers' performance. It is better to keep these unusual values (include NA observations) by adding some new variables [52]. This is another technique used in obtaining new variables.

The third and last major variable generation technique applied in this study is binning or discretization, which reduces the cardinality of continuous and discrete data by grouping related values together into a small number of bins [53]. For example, variables as "Age" and "Monthly income" can be further divided into several intervals to reduce the noise or non-linearity and hence improve the model's predicative power. In our study, usually two or more generation techniques are combined together for a single variable. Take the original "Age" variable as the example, the minimum value is "0" and the maximum value is "109" and the mean value is 52.3. Three new variables based on "Age" can be generated according to the following formula:

$$GeneratedAgeVariables\begin{cases} Juvenile\_Age, & 1, if \quad age \leq 17 \\ Working\_Age\_log, & log(age-17), if \quad 59 \leq age \leq 18 \\ Senior\_Age, & 1, if \quad age \geq 60 \end{cases} \tag{9}$$

All together 80 derived variables are generated from the original variables and thus the derived data set contains 150,000 examples with 80 generated variables.

We run LLRE, LLR, RF and CART algorithms on the Kaggle dataset using the above two different set of variables and the experiment results in terms of AUC are shown in Table 3.

From Table 3, we can see that performance of most classifiers is improved and LLRE and LLR algorithms even get an increase of about 20% or more in AUC. This demonstrates the usefulness of the generated variables. We also notice that performance of RF (a recursive

Table 3. Performance on two variable sets in terms of AUC.

Classifier	Original variables	Generated variables
LLRE	0.6796	0.8597
RF	0.8488	0.8587
LLR	0.4898	0.8567
CART	0.7702	0.7632

doi:10.1371/journal.pone.0117844.t003

partitioning ensemble) does not improve too much on the generated variables, as a large part of our variable generation is based on partitioning the original variables into specific intervals.

Choice of *k*. First, we want to test the performance of LLRE with different values of *k*. As shown in Fig. 2, LLRE's AUC values increase slowly when the number of iterations increases at the very beginning. This indicates that clustering the majority ensemble does lead to a better performance. And a larger ensemble size also implies an increase in AUC but when $k \geq 10$, LLRE is no longer too much sensitive to the parameter *k*.

From our empirical study, a suggested value for *k* can be given by the function $k \geq max(ceil(N2/N1), 10)$ to ensure data balance and classification accuracy, where N_1 and N_2 denote the number of minority examples and number of majority examples in the data. Hence, the sub-group number parameter *k* for LLRE in the following experiments is set to 14.

Variables importance. We use the LLR-occurrence measure and the mean AUC decrease measure defined in the previous section to measure variable importance in the experiment. Bar plots of top 20 important variables in terms of LLR-occurrence and mean AUC decrease are shown in Figs. 3 and 4, respectively.

From Fig. 3, variables including "Has Revolving Lines", "Never 60–89 Days Past Due Not Worse", "Never 90 Days Late", "Delinquencies Per Revolving Line", "Weird0999 Utilization", "Excess Utilization", "Never Past Due", "Log Revolving Utilization Times Lines", "Log Revolving Utilization Of Unsecured Lines", "Log Number Of Times Past Due" and "Log Age" are included in all LLR base models. These variables are the most important ones in terms of LLR-occurrence measure.

From Fig. 4, we can see that "Never Past Due", "Disposable Income" (based on original variables "Debt Ratio" and "Monthly Income") and "Log Revolving Utilization Times Lines" are very important indicators of whether the applicant will default in the future, as permutation of these two variables can cause a decrease in AUC of more than 1 percent on average.

As is shown in Figs. 3 and 4, the two measures agree strongly on most important variables. Different measures give similar results provide further confidence that these variables can be used in credit scoring to evaluate the default risks of credit applicants.

According to the experiment results, variables such as number of dependents or household size provide the least information for evaluating an applicant's future credit rating.

Comparison study. Here, we want to compare our approach with other popular credit scoring approaches. The winning algorithm for the Kaggle competition is a blended approach of multiple random forests, support vector machines data cleaning, and gradient boosting machines [54]. As the algorithm is not public available and is only slighted better (0.005) than random forests in terms of AUC [54], hence random forests(RF) can be regarded as the start-of-the-art credit scoring algorithm for the Kaggle data. Moreover, as the time complexity of standard SVM algorithm and MLP neural networks are usually greater than $O(n^2)$, a single run of these algorithms on desktop machines will take even a month or more to complete and this makes them inappropriate for large scale classifications.

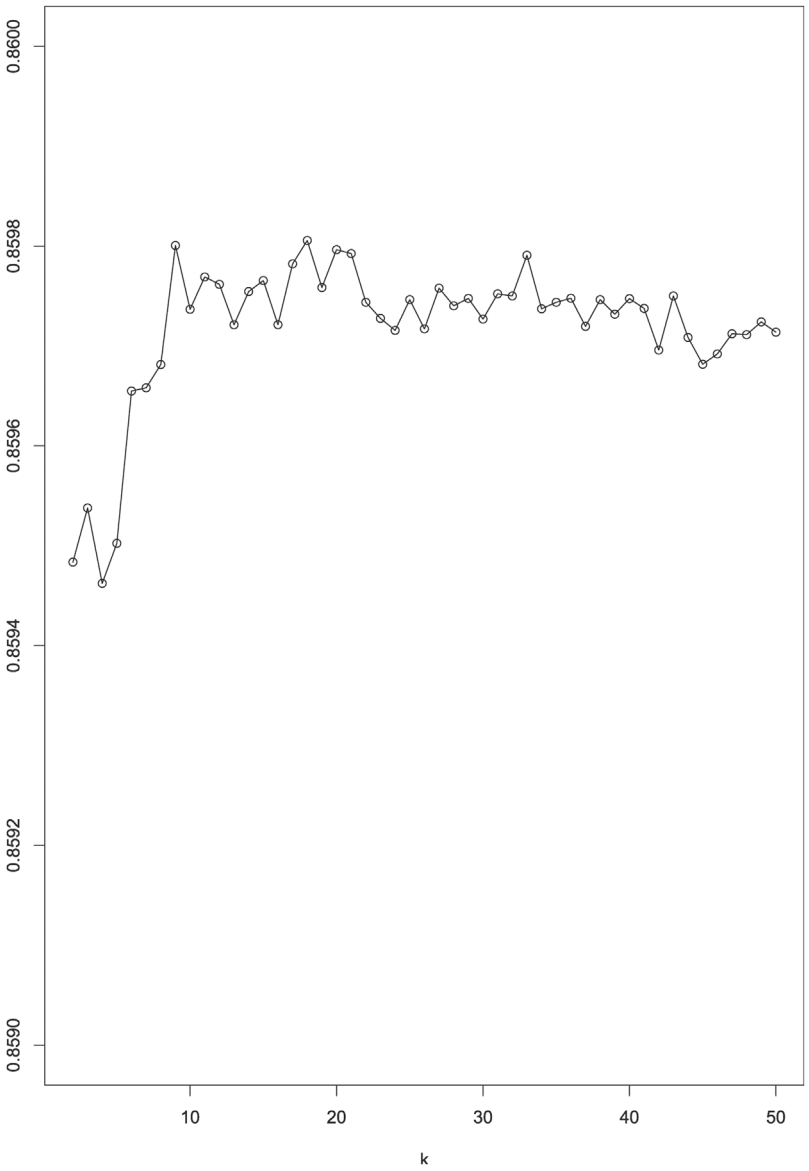

Fig 2. AUC change with LLRE using different *k*.

doi:10.1371/journal.pone.0117844.g002

Thus, in this study, we only compare LLRE with popular single and ensemble learning techniques with fast training time such as Lasso-Logistic Regression (LLR), CART decision tree and RF. Comparison with these models are conducted with corresponding "glmnet", "randomForest", "rpart" packages in R. In the experiments, we want all classifiers to have the same opportunities to achieve the best results, thus the default settings are adopted: for LLR, the error distribution and link function *family* = *binomial*, type.measure = *auc*, *alpha* = 1; for CART, the split criteria is set to Gini; for RF, number of trees *ntree* = 500 and split variables *mtry* = 3. The following Tables 4 and 5 report the AUCs and F-measure values of the proposed LLRE, RF, LLR and CART algorithms on 30 runs of the experiments.

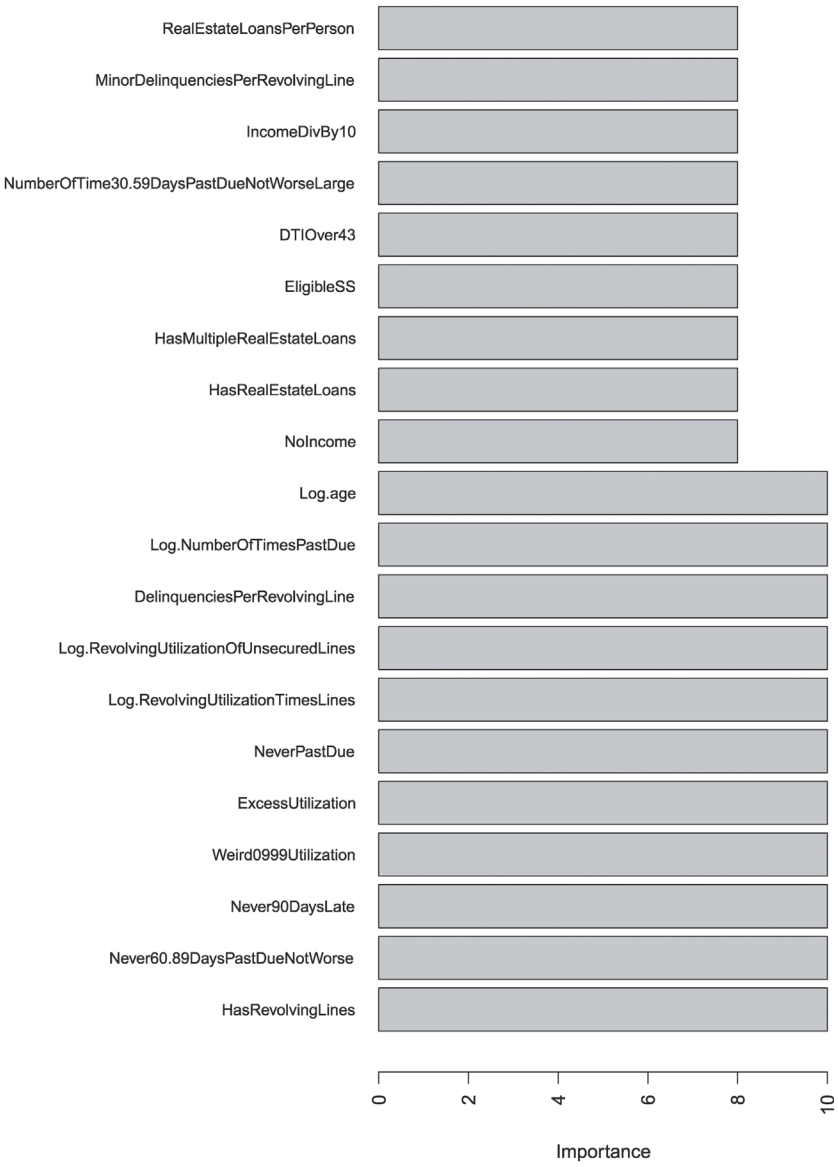

Fig 3. Top 20 important variables in terms of LLR occurrence.

doi:10.1371/journal.pone.0117844.g003

The Friedman rank sum test statistic for the above AUC results is 77.88. This is significant (the corresponding p-value is less than 2.2e-16) and a post hoc Nemenyi test was then applied to find out which pairs of algorithms are significantly different.

For the ease of notation, classifiers LLRE, RF, LLR and CART are denoted by A, B, C, D respectively. The average ranks of classifiers A, B, C and D are:

$$R_{jA} = 1.0667, R_{jB} = 2.5667, R_{jC} = 2.3667, R_{jD} = 4$$

Using the proposed algorithm (A) as the control and computing the z statistic of Nemenyi test for different classifier pairs, we obtain:

$$z_{BA} = 4.5, z_{CA} = 3.9, z_{DA} = 8.8$$

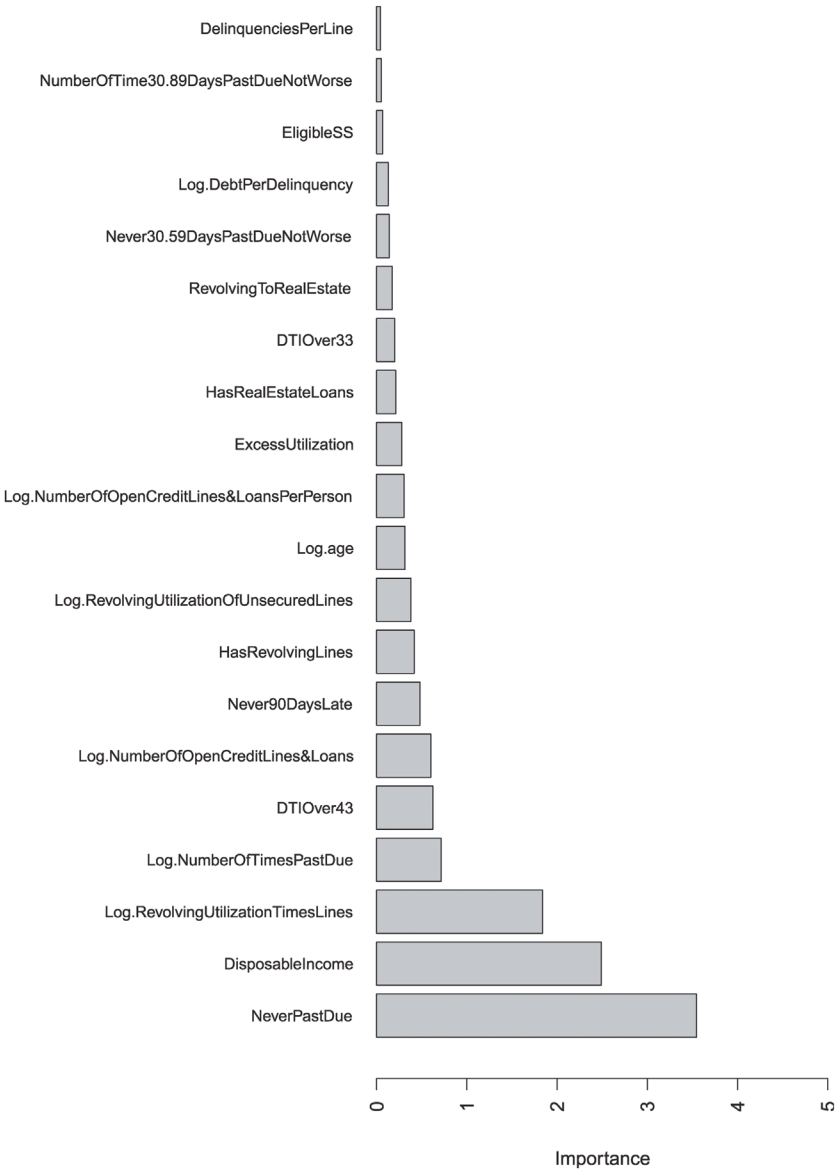

Fig 4. Top 20 important variables in terms of mean AUC decrease.

doi:10.1371/journal.pone.0117844.g004

For $\alpha = 0.05$, the critical value of Studentized range distribution $q_\alpha = 3.71$ and divide q_α value by $\sqrt{2}$, we get the Nemenyi's test critical value of 2.619. It can be seen that all three Nemenyi statistics z_{BA}, z_{CA} and z_{DA} values exceed 2.619 and thus there exists significant differences between the proposed LLRE and the other three algorithms. In other words, in terms of AUC, LLRE gives the best performance on the Kaggle dataset.

From Table 5, we can see that on the F-measure, LLRE shows remarkably the best performance, followed by RF, LLR and then CART. Similarly, the Friedman rank sum test statistic for the above F-measure results is 87.76. This is also significant (the corresponding p-value is less than 2.2e-16) and a post hoc Nemenyi test was then applied and we obtain the

Table 4. Performance comparison in terms of AUC.

Run No	LLRE	RF	LLR	CART
1	0.8598	0.857	0.8571	0.7632
2	0.8553	0.8538	0.8526	0.7676
3	0.8662	0.8609	0.8651	0.7786
4	0.8602	0.8576	0.8577	0.7778
5	0.858	0.8564	0.8559	0.7746
6	0.8662	0.8628	0.8638	0.7689
7	0.8544	0.8536	0.8526	0.77
8	0.8619	0.8617	0.8589	0.7749
9	0.8657	0.8606	0.8636	0.7832
10	0.8575	0.8569	0.8561	0.7665
11	0.8622	0.8578	0.8604	0.7762
12	0.8565	0.8551	0.8542	0.7748
13	0.8576	0.8519	0.8573	0.7763
14	0.8573	0.8537	0.8547	0.7761
15	0.8638	0.8648	0.8606	0.7699
16	0.8567	0.8535	0.8547	0.7728
17	0.8586	0.8579	0.8558	0.7783
18	0.8696	0.8631	0.8666	0.7792
19	0.8529	0.8523	0.8506	0.77
20	0.8651	0.8607	0.8609	0.7732
21	0.8537	0.8498	0.8506	0.7695
22	0.8652	0.8625	0.8635	0.7783
23	0.8603	0.8606	0.8578	0.7738
24	0.8607	0.856	0.8584	0.7692
25	0.8668	0.8642	0.8648	0.7763
26	0.859	0.8562	0.857	0.7698
27	0.8574	0.8553	0.8548	0.7807
28	0.8576	0.8543	0.8564	0.7685
29	0.8633	0.8599	0.8609	0.7685
30	0.8636	0.8617	0.8623	0.7775

doi:10.1371/journal.pone.0117844.t004

corresponding Nemenyi statistics:

$$z'_{BA} = 3.2, z'_{CA} = 5.8, z'_{DA} = 9$$

Again, it can be seen that all three Nemenyi statistics z'_{BA}, z'_{CA} and z'_{DA} values exceed 2.619. Thus, there also exists significant differences between the proposed LLRE and RF, LLR, CART when F-measure is used as an evaluation metric.

Discussions. Non-parametric models such as random forests have gained popularity in recent years with remarkable performance in credit scoring problems and it is also one of the most advanced statistical learning methods so far. Logistic regression has over the years a standard technique in credit scoring problems due to its well-understood feature and wide availability. However, its regularized version-lasso logistic regression for credit scoring problems is still limited. In this study, we examined the performance of the proposed Lasso-logistic regression ensemble, random forests, lasso-logistic regression, and classification and regression tree, for a large data credit scoring problem. Encouragingly, except classification and regression tree,

Table 5. Performance comparison in terms of F-measure.

Run No	LLRE	RF	LLR	CART
1	0.3856	0.2683	0.2599	0
2	0.3872	0.2727	0.2466	0
3	0.3777	0.2712	0.2626	0
4	0.3799	0.2694	0.2738	0
5	0.4004	0.2873	0.2793	0
6	0.3848	0.2699	0.2548	0
7	0.3756	0.2671	0.2555	0
8	0.394	0.2834	0.2575	0
9	0.4088	0.2933	0.2796	0
10	0.3729	0.2734	0.265	0
11	0.3759	0.2669	0.2647	0
12	0.3846	0.2725	0.2567	0
13	0.3939	0.2869	0.2759	0
14	0.3717	0.2815	0.2615	0
15	0.3806	0.2689	0.2672	0
16	0.3855	0.2877	0.2785	0
17	0.3963	0.2756	0.2677	0
18	0.372	0.2679	0.2466	0
19	0.3751	0.2579	0.2642	0
20	0.3807	0.2817	0.2664	0
21	0.3737	0.2804	0.2559	0
22	0.3836	0.2648	0.2604	0
23	0.3937	0.2699	0.2627	0
24	0.3876	0.2811	0.278	0
25	0.3951	0.2958	0.2823	0
26	0.3784	0.2744	0.2548	0
27	0.3895	0.2779	0.2648	0
28	0.3815	0.2621	0.2413	0
29	0.3716	0.2723	0.2517	0
30	0.3826	0.2705	0.2649	0

doi:10.1371/journal.pone.0117844.t005

all approaches have shown adequate ability in predicting default credit loans. Experiments results have shown that the combination of clustering and bagging techniques not only takes advantages of the superior capability of ensemble learning and but also provides an approach for extending the use of the traditional parametric models in credit scoring.

In the experiments, LLRE has demonstrated an overall better performance across the popular AUC and F-measure for unbalanced credit scoring. On the AUC measure, LLRE outperforms RF, LLR and CART significantly. The difference between LLRE and RF in terms of AUC appears marginal yet in fact constant. These marginal improvements should make an adequate gain in profits for financial institutions. On the F-measure, LLRE has shown a surprising well performance and surpassed all the other compared methods by a large margin. LLRE even beats RF by more than 10 percent in term of F-measure on average. We also notice that although CART might obtain a high performance in term of accuracy but its bad performance on the minority (default) cases make it the worst choice in unbalanced learning with zero F-measure values during all runs of the experiments.

Another factor contributing to the success of LLRE is possibly the carefully generated variables. Its better performance with the generated variable set implies that time-consuming exploratory data analysis and variable selection process using domain's expertise is very important in constructing logistic regression like models. This finding agrees with the assertion made in [47]. When such preprocessing efforts are unavailable, random forests could be an alternative.

Our variable importance results are in line with previous studies that credit history related variables such as past credit due information and revolving utilization data are very important factors in predicting loan applicants' quality [55–57]. As revealed by the mean AUC decrease measure, disposable income is another crucial factor in evaluating a customer's repayment quality. This finding also agrees with the results reported previously [58, 59]. The "Log age" variable which appears both in Figs. 3 and 4 is also crucial in extending credit and this finding agrees with [56, 59] but is inconsistent with a previous research [55] suggesting that age was one of the least differentiating variables. In our study, the least important factor selected by both importance measures is the number of dependents of the applicant. However, this finding does not agree with the results in [60] and needs further investigation.

Conclusions

In this paper, a new ensemble credit scoring algorithm based on Lasso-logistic regression for large unbalanced data has been presented. In our approach, clustering the majority data and bagging the minority data have been applied to generate balanced training data and maintain data diversity. Different from previous researches, the λ parameter is determined by OBS data, which also helps to reduce training time of base LLR models. Experiment studies using real world credit scoring data have demonstrated that LLRE beats CART, LLR and RF in terms of AUC and F-measure.

We also show that the proposed LLRE method can be used to evaluating variable importance by providing two kinds of importance measures. As to which variables may influence the credit decision process, experiment results with the proposed algorithm using the LLR-occurrence and AUC mean decrease measures have shown that past due information, disposable income and revolving utilization times lines are the most important indicators.

Future researches may focus on newly developed fast and accurate clustering methodologies to produce better sub-groups of majority data. Choosing other state-of-the-art models with fast training time as the base classifiers could also be considered.

Acknowledgments

The authors would like to thank the anonymous reviewers for their constructive comments and suggestions.

Author Contributions

Conceived and designed the experiments: HW. Performed the experiments: HW. Analyzed the data: QSX. Contributed reagents/materials/analysis tools: LFZ. Wrote the paper: HW.

References

1. Thomas LC, Edelman DB, Crook JN (2002) Credit scoring and its applications. Siam.
2. West D (2000) Neural network credit scoring models. Computers & Operations Research 27: 1131–1152. doi: 10.1016/S0305-0548(99)00149-5
3. Durand D (1941) Risk elements in counsumer instalment financing. National Bureau of Economic Research, New York.

4. Altman EI (1968) Financial ratios discriminant analysis, and the prediction of corporate bankruptcy. Journal of Finance 23: 589C609. doi: 10.1111/j.1540-6261.1968.tb03007.x
5. Wiginton JC (1980) A note on the comparison of logit and discriminant models of consumer credit behavior. Journal of Financial and Quantitative Analysis 15: 757–770. doi: 10.2307/2330408
6. Altman EI, Marco G, Varetto F (1994) Corporate distress diagnosis: Comparisons using linear discriminant analysis and neural networks (the italian experience). Journal of Banking & Finance 18: 505–529. doi: 10.1016/0378-4266(94)90007-8
7. Desai VS, Crook JN, Overstreet GA Jr (1996) A comparison of neural networks and linear scoring models in the credit union environment. European Journal of Operational Research 95: 24–37. doi: 10.1016/0377-2217(95)00246-4
8. Arminger G, Enache D, Bonne T (1997) Analyzing credit risk data: A comparison of logistic discrimination, classification tree analysis, and feedforward networks. Computational Statistics 12.
9. Baesens B, Van Gestel T, Viaene S, Stepanova M, Suykens J, et al. (2003) Benchmarking state-of-the-art classification algorithms for credit scoring. Journal of the Operational Research Society 54: 627–635. doi: 10.1057/palgrave.jors.2601545
10. Hand DJ, Henley WE (1997) Statistical classification methods in consumer credit scoring: a review. Journal of the Royal Statistical Society: Series A (Statistics in Society) 160: 523–541. doi: 10.1111/1467-985X.00078
11. Thomas LC (2000) A survey of credit and behavioural scoring: forecasting financial risk of lending to consumers. International Journal of Forecasting 16: 149–172. doi: 10.1016/S0169-2070(00)00034-0
12. Thomas L, Oliver R, Hand D (2005) A survey of the issues in consumer credit modelling research. Journal of the Operational Research Society 56: 1006–1015. doi: 10.1057/palgrave.jors.2602018
13. Lahsasna A, Ainon RN, Wah TY (2010) Credit scoring models using soft computing methods: A survey. Int Arab J Inf Technol 7: 115–123.
14. Abdou HA, Pointon J (2011) Credit scoring, statistical techniques and evaluation criteria: a review of the literature. Intelligent Systems in Accounting, Finance and Management 18: 59–88. doi: 10.1002/isaf.325
15. Dietterich TG (2000) Ensemble methods in machine learning. In: Multiple classifier systems, Springer. pp. 1–15.
16. Polikar R (2012) Ensemble learning. In: Ensemble Machine Learning, Springer. pp. 1–34.
17. Galar M, Fernández A, Barrenechea E, Bustince H, Herrera F (2012) A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 42: 463–484. doi: 10.1109/TSMCC.2011.2161285
18. Zhou L, Lai KK, Yu L (2010) Least squares support vector machines ensemble models for credit scoring. Expert Systems with Applications 37: 127–133. doi: 10.1016/j.eswa.2009.05.024
19. Hsieh NC, Hung LP (2010) A data driven ensemble classifier for credit scoring analysis. Expert Systems with Applications 37: 534C545. doi: 10.1016/j.eswa.2009.05.059
20. Wang G, Hao J, Ma J, Jiang H (2011) A comparative assessment of ensemble learning for credit scoring. Expert systems with applications 38: 223–230. doi: 10.1016/j.eswa.2010.06.048
21. Brown I, Mues C (2012) An experimental comparison of classification algorithms for imbalanced credit scoring data sets. Expert Systems with Applications 39: 3446–3453. doi: 10.1016/j.eswa.2011.09.033
22. Paass G, Kindermann J (1998) Bayesian classification trees with overlapping leaves applied to credit-scoring. In: Research and Development in Knowledge Discovery and Data Mining, Springer. pp. 234–245.
23. Liu Y, Yao X (1999) Ensemble learning via negative correlation. Neural Networks 12: 1399–1404. doi: 10.1016/S0893-6080(99)00073-8 PMID: 12662623
24. Tsai CF, Wu JW (2008) Using neural network ensembles for bankruptcy prediction and credit scoring. Expert Systems with Applications 34: 2639–2649. doi: 10.1016/j.eswa.2007.05.019
25. Li J, Wei H, Hao W (2013) Weight-selected attribute bagging for credit scoring. Mathematical Problems in Engineering 2013.
26. Han L, Han L, Zhao H (2013) Orthogonal support vector machine for credit scoring. Engineering Applications of Artificial Intelligence 26: 848–862. doi: 10.1016/j.engappai.2012.10.005
27. Alejo R, García V, Marqués A, Sánchez J, Antonio-Velázquez J (2013) Making accurate credit risk predictions with cost-sensitive mlp neural networks. In: Management Intelligent Systems, Springer. pp. 1–8.
28. Wang CM, Huang YF (2009) Evolutionary-based feature selection approaches with new criteria for data mining: A case study of credit approval data. Expert Systems with Applications 36: 5900–5908. doi: 10.1016/j.eswa.2008.07.026

29. Crone SF, Finlay S (2012) Instance sampling in credit scoring: An empirical study of sample size and balancing. International Journal of Forecasting 28: 224–238. doi: 10.1016/j.ijforecast.2011.07.006
30. García V, Marqués AI, Sánchez JS (2012) Improving risk predictions by preprocessing imbalanced credit data. In: Neural Information Processing. Springer, pp. 68–75.
31. Marqués A, García V, Sánchez J (2013) On the suitability of resampling techniques for the class imbalance problem in credit scoring. Journal of the Operational Research Society 64: 1060–1070. doi: 10.1057/jors.2012.120
32. Paleologo G, Elisseeff A, Antonini G (2010) Subagging for credit scoring models. European Journal of Operational Research 201: 490–499. doi: 10.1016/j.ejor.2009.03.008
33. Friedman J, Hastie T, Tibshirani R (2010) Regularization paths for generalized linear models via coordinate descent. Journal of statistical software 33: 1. PMID: 20808728
34. Huang J, Ling CX (2005) Using auc and accuracy in evaluating learning algorithms. Knowledge and Data Engineering, IEEE Transactions on 17: 299–310. doi: 10.1109/TKDE.2005.50
35. Janitza S, Strobl C, Boulesteix AL (2013) An auc-based permutation variable importance measure for random forests. BMC bioinformatics 14: 119. doi: 10.1186/1471-2105-14-119 PMID: 23560875
36. Tibshirani R (1996) Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society Series B (Methodological): 267–288.
37. Polikar R (2006) Ensemble based systems in decision making. Circuits and Systems Magazine, IEEE 6: 21–45. doi: 10.1109/MCAS.2006.1688199
38. Freund Y, Schapire RE (1995) A desicion-theoretic generalization of on-line learning and an application to boosting. In: Computational learning theory. Springer, pp. 23–37.
39. Breiman L (1996) Bagging predictors. Machine learning 24: 123–140. doi: 10.1023/A:1018094028462
40. Breiman L (2001) Random forests. Machine learning 45: 5–32. doi: 10.1023/A:1010933404324
41. Friedman JH (2001) Greedy function approximation: a gradient boosting machine. Annals of Statistics: 1189–1232. doi: 10.1214/aos/1013203451
42. Hastie TJ, Tibshirani RJ, Friedman JH (2011) The elements of statistical learning: data mining, inference, and prediction. Springer.
43. Bauer E, Kohavi R (1999) An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. Machine learning 36: 105–139. doi: 10.1023/A:1007515423169
44. Han J, Kamber M, Pei J (2006) Data mining: concepts and techniques. Morgan kaufmann.
45. Kuncheva LI (2004) Combining pattern classifiers: methods and algorithms. John Wiley & Sons.
46. Fumera G, Roli F (2005) A theoretical and experimental analysis of linear combiners for multiple classifier systems. Pattern Analysis and Machine Intelligence, IEEE Transactions on 27: 942–956. doi: 10.1109/TPAMI.2005.109
47. Bhattacharyya S, Jha S, Tharakunnel K, Westland JC (2011) Data mining for credit card fraud: A comparative study. Decision Support Systems 50: 602–613. doi: 10.1016/j.dss.2010.08.008
48. García V, Sánchez JS, Mollineda RA (2012) On the effectiveness of preprocessing methods when dealing with different levels of class imbalance. Knowledge-Based Systems 25: 13–21. doi: 10.1016/j.knosys.2011.06.013
49. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research 7: 1–30.
50. Keene ON (1995) The log transformation is special. Statistics in medicine 14: 811–819. doi: 10.1002/sim.4780140810 PMID: 7644861
51. Hyndman RJ, Athanasopoulos G (2014) Forecasting: principles and practice. OTexts.
52. Adler J (2010) R in a nutshell: A desktop quick reference. O'Reilly Media, Inc.
53. Garcia S, Luengo J, Sáez JA, López V, Herrera F (2013) A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning. Knowledge and Data Engineering, IEEE Transactions on 25: 734–750. doi: 10.1109/TKDE.2012.35
54. Sharma D (2013) Elements of optimal predictive modeling success in data science: An analysis of survey data for the give me some creditcompetition hosted on kaggle. Available at SSRN 2227333.
55. Hsieh NC (2005) Hybrid mining approach in the design of credit scoring models. Expert Systems with Applications 28: 655–665. doi: 10.1016/j.eswa.2004.12.022
56. Bellotti T, Crook J (2009) Support vector machines for credit scoring and discovery of significant features. Expert Systems with Applications 36: 3302–3308. doi: 10.1016/j.eswa.2008.01.005

57. Kao LJ, Chiu CC, Chiu FY (2012) A bayesian latent variable model with classification and regression tree approach for behavior and credit scoring. Knowledge-Based Systems 36: 245–252. doi: 10.1016/j.knosys.2012.07.004

58. Ang JS, Chua JH, Bowling CH (1979) The profiles of late-paying consumer loan borrowers: An exploratory study: Note. Journal of Money, Credit and Banking: 222–226. doi: 10.2307/1991836

59. Hian-Chye K, Chin TW, Peng GC (2004) Credit scoring using data mining techniques. Singapore Management Review 26: 25.

60. Yap BW, Ong SH, Husain NHM (2011) Using data mining to improve assessment of credit worthiness via credit scoring models. Expert Systems with Applications 38: 13274–13283. doi: 10.1016/j.eswa.2011.04.147