

Improving Risk Modeling Via Feature Selection, Hyper-Parameter Adjusting, and Model Ensembling

Yan Wang

Xuelei Sherry Ni

Jennifer Lewis Priestley

Kennesaw State University
Kennesaw, GA 30144, USA

Abstract

It has been demonstrated that methods including feature selection (FS), hyper-parameter adjusting, and model ensembling can improve the performance of binary classifiers. In this study, we propose a framework that aims at improving risk modeling by simultaneously using the above-mentioned model-improving methods. The feasibility of the framework is assessed on a dataset containing commercial information of the US companies. Three FS methods including weight by Relief, weight by information gain, and weight by correlation, are employed on each of the four classifiers including logistic regression (LR), decision tree (DT), neural network (NN), and support vector machine (SVM). After identifying the most appropriate FS method for each classifier, the hyper-parameters are then adjusted. Finally, each classifier is ensembled using bagging and boosting techniques. To investigate the effect of these model-improving methods, the model performance is evaluated using classification accuracy, area under the curve (AUC), false positive rate (FPR), and false negative rate (FNR). The results exhibited that FS and boosting on LR could largely increase its accuracy and decrease FNR. On the contrary, regularization via hyper-parameter adjusting on LR cannot further improve model performance. DT is not sensitive to any of the fore-mentioned methods. The beneficial effect of model-improving methods is obvious on NN with respect to FPR and FNR while negligible in accuracy. SVM is no longer a good base classifier to be ensembled after applying FS and hyper-parameter adjusting methods. The proposed framework provides a reference for the simultaneous utilization of these model-improving methods in business delinquency modeling.

Keywords: feature selection; hyper-parameter adjusting; ensemble model; business delinquency.

1 Introduction

One of the most popular techniques to identify risk is binary classification modeling, i.e., modeling whether or not certain business will go bankrupt or the probability of going bankrupt. According to Kim and Gu (2006), logistic regression (LR) is a traditional technique that is widely used in the financial and risk modeling domain because of its good ability to provide clear interpretability as well as consistent model performance. Decision tree (DT), neural network (NN), and support vector machine (SVM) are the three popular alternatives for LR since they can model the complex nonlinear relationships between predictors. For example, Tsai and Wu (2008), Zhang et al. (2010), West (2000) and Khandani et al. (2010), Wang and Priestley (2017), and Zhou et al. (2018) have successfully employed the above-mentioned approaches in their investigations.

Many researchers focus on investigating model-improving approaches that could benefit model performance in the financial domain. There are three main approaches that have received formal investigative attention. The first approach is feature selection (FS). Huang and Wang (2006) applied a genetic algorithm-based FS for SVM that could significantly improve the classification accuracy. Chen (2012) developed an integrated FS method in credit rating classification. In the study by Wang et al. (2012), a scatter search metaheuristic based FS method is used for credit scoring. The second approach to improve model performance in the financial domain is hyper-parameter adjusting. According to Huang and Wang (2006), optimizing the kernel functions in SVM can significantly improve the classification accuracy. Xia et al. (2017) proposed a boosted decision tree approach using Bayesian hyper-parameter optimization to improve credit risk modeling. The third approach that is beneficial to business risk modeling is model ensembling. Koutanaei et al. (2015) conducted a hybrid data mining model for credit scoring and showed that ensemble on NN can significantly improve the model performance. Yao and Lian (2016) pointed out that SVM based ensemble models can serve as an alternative for LR in financial modeling.

Based on the aforementioned studies, we propose a framework (illustrated in Fig. 1 and discussed in detail in Section 3.3) that combines three model-improving approaches including FS, hyper-parameter adjusting, and model ensembling together. This framework aims at improving the model performance using criteria including classification accuracy, area under the curve (AUC), false positive rate (FPR), and false negative rate (FNR). The feasibility of the framework is tested on a commercial dataset approved by Equifax (located at Atlanta, GA, USA). By simultaneously considering the above-mentioned three model-improving methods, the proposed framework can provide a comprehensive comparison of the performance of different classifiers when using different FS methods, different hyper-parameter adjusting values, and different ensembling approaches.

The rest of the paper is organized as follows. Since different FS methods and several binary classifiers are used in this study, we will first review the relevant algorithms in Section 2. Section 3 provides a detailed description about our experimental design, which aims at using our proposed framework to improve business delinquency modeling. The experimental results are elaborated in Section 4. Section 5 addresses the conclusions.

2 Algorithms

In this section, we review the algorithms related to the three widely used model-improving approaches. Since most of our experiments (discussed in detail in Section 3) are elaborated in the RapidMiner data mining package developed in R, the option names for parameter settings for FS as well as hyper-parameter adjusting in RapidMiner are also discussed.

2.1 Feature selection methods

FS refers to the selection of the most appropriate subset of features with strong ability to represent relevant information provided in the dataset. As discussed in Salappa et al. (2007), FS algorithms have several benefits such as decreasing the noise and reducing computational cost that is typically associated with increased classification performance. In this study, three FS methods are applied: weight by Relief, weight by information gain, and weight by correlation. Relief is an algorithm developed in Kira and Rendell (1992a) and Kira and Rendell (1992b). As quoted in the RapidMiner documentation written by Akthar et al. (2012), ‘*Relief estimates the quality of features according to how well their values distinguish between the instances of the same and different classes that are near each other*’. In RapidMiner, the features are weighted by Relief and sorted according to the weights. Similarly, according to Guyon and Elisseeff (2003), the weight by information gain method and the weight by correlation method sort the variables according to its information gain and its correlation with the dependent variable, respectively. Variables with higher weights denote higher relevance to the dependent variables. In RapidMiner, the parameter ‘weight relation’ controls the number of features selected and we investigate the effect of the above-mentioned three FS methods by changing the value of this parameter. For simplicity, we use the terms ‘Relief’, ‘information gain’, and ‘correlation’ for the three FS methods, respectively.

2.2 Binary classifiers along with hyper-parameters

Logistic regression LR describes the relationship between the input variables (x_1, x_2, \dots, x_n) and the predicted probability of the event p defined in Eq. (1). According to Menard (2018), the unknown parameters ($\beta_0, \beta_1, \dots, \beta_n$) were estimated by maximizing the likelihood function of LR defined in Eq. (2).

In the studies of Le Cessie and Van Houwelingen (1992) and Cornillon and Matzner-Lober (2011), it has been stated that combining regularization methods such as ridge regression (i.e., using L2 penalized least squares to shrink the coefficients of correlated predictors equally towards zero) and Least Absolute Shrinkage and Selection Operator (LASSO, i.e., using L1 penalized least squares to shrink some coefficients to zero) together with LR can reduce the potential multicollinearity and overfitting problems. In this study, the regularization of LR is applied by controlling the parameter ‘alpha’ in RapidMiner.

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n \quad (1)$$

$$L((\beta_0, \beta_1, \dots, \beta_n); (x_1, x_2, \dots, x_n)) = \prod_{i=1}^n p((x_1, x_2, \dots, x_n) | (\beta_0, \beta_1, \dots, \beta_n)) \quad (2)$$

Decision tree DT is a top-down tree structure that contains several nodes, leaves, and branches. Each node is built by searching the optimal splits on input variables based on different criteria such as entropy and information gain. In Witten et al. (2016), the entropy is calculated using Eq. (3), where c is the number of classes in the dependent variable and $p(j|t)$ is the relative frequency of class j at node t . The information gain is calculated using Eq. (4), where p is the parent node that has been split into k partitions, and n_i is the number of observations in partition i . In this study, we use gain ratio (default setting in RapidMiner) to select the splitting variable for each node. According to Naik and Samant (2016), gain ratio is a variant of information gain that allows the breadth and uniformity of the variable values. Song and Ying (2015) pointed out that using stopping rules on DT such as limiting the depth of the tree or limiting the number of records in a leaf can prevent the overfitting problems. In this study, we investigate the effect of the tree depth on the classification performance by changing the hyper-parameter ‘maximal depth’ in RapidMiner.

$$\text{entropy}(t) = - \sum_{j=1}^c p(j|t) \log[p(j|t)] \quad (3)$$

$$\text{gain} = \text{entropy}(t) - \left(\sum_{i=1}^k \frac{n_i}{n} \text{entropy}(i) \right) \quad (4)$$

Neural network NN aims at learning the non-linear relationship between the dependent and independent variables and has been frequently used in business risk modeling. As depicted by Gurney (2014), NN consists of input and output layers, as well as a hidden layer. The training process contains forward propagation and backward propagation. In the forward propagation, it transforms the input variables into high level features by using the activation functions in the hidden units contained in the hidden layer. In the backward propagation, it adjusts the weights by minimizing the loss function. According to Hecht et al. (2015), several hyper-parameters in NN including size of hidden layer, learning rate, and learning momentum need to be tuned for a better performance. In our study, we focus on investigating the effect of the hyper-parameter, the size of hidden layer, by changing the value of ‘hidden layers’ in RapidMiner.

Support vector machine As stated in Cristianini et al. (2000), SVM aims at looking for the optimal separating hyperplane between the two classes by maximizing the marginal distance. It can be written as an optimization problem shown in Eq. (5), where $(x_i; y_i)$ are the input data points, w and b are the parameters that defined the hyperplane and need to be trained, and $h w; x_i$ denotes the dot product between w and x_i . In SVM, the kernel function that maps the input variables to a higher separable feature space needs to be optimized. In this study, the effect of different kernel functions on the model performance is examined by changing the value of ‘kernel type’ in RapidMiner.

$$\arg \min = \frac{1}{2} \|w\|^2 \text{ subject to } y_i (< w, x_i > + b) \geq 1 \quad (5)$$

2.3 Ensemble approaches: bagging and boosting

In contrast to the single/base classifiers, ensemble models use several base classifiers such as LR and DT in order to solve one problem. In Wang and Ma (2012), the ensemble approach has been used to improve the performance of risk modeling. In our study, we focus on improving business risk modeling via two ensembling approaches: bagging and boosting. According to Bauer and Kohavi (1998) that bagging algorithm is based on the majority voting concept, where base classifiers are built in parallel on different bootstrap subsets of training dataset.

Like bagging, the AdaBoost (Adaptive Boosting, the boosting algorithm available provided in RapidMiner) algorithm generates a set of classifiers and then applies voting logic. Different from bagging, which builds the classifiers independently, the AdaBoost generates the classifiers sequentially and changes the weights of the training instances based on the previously built classifiers.

3 Experimental design

3.1 Dataset description and pre-processing

The dataset used for assessing the feasibility of the proposed framework is approved by Equifax. The dataset contains 36 separate subsets, with each subset representing quarterly financial information of d-identified US companies from 2006 to 2014. The data includes over 10 million observations and 305 independent variables including companies' financial information such as non-financial account activities, telecommunication account activities, utility account activities, service account activities, industry account activities, liabilities, and liens. The target variable *GOODBAD* denotes a binary problem and can be defined as follows: good and bad business behaviors are those that have no past due activities in service account (*GOODBAD* = 0) and have past due activities (*GOODBAD* = 1), respectively. The percentage of past due behaviors (i.e., *GOODBAD* = 1) is about 30%.

Before using the dataset on the proposed framework, a stratified sampling procedure was used to obtain a randomly sampled subset of 2000 companies. Then a series of data pre-processing procedures were applied as follows: (1) Splitting the data into a training (80%) set and a validation set (20%); (2) Re-moving observations with missing values in the target variable *GOODBAD*; (3) Removing variables with missing percentage larger than 70% due to the incomplete information; (4) Missing value imputation based on the median values; (5) Normalization due to the various range of the variables. As a result of data pre-processing, 156 features were used for the implementation of the experiment.

3.2 Performance evaluation

Several model evaluation criteria are applied in this study. The first criterion used is the classification accuracy. True Positive (TP) and False Positive (FP) are outcomes identified as having past due activities correctly or mistakenly, respectively; and True Negative (TN) and False Negative (FN) as outcomes identified as no past due activities correctly or mistakenly, respectively. The classification accuracy can be expressed in Eq. (6).

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN} \quad (6)$$

Another evaluation criterion used is the AUC of the Receiver Operating Characteristic Curve (ROC), which shows the interaction between the true positive rate (TPR) and the false positive rate (FPR). Higher AUC denotes a better model performance. According to Yamane (1973), TPR, FPR, and FNR are defined in Eq. (7), (8), and (9), respectively. For the study in this paper, classification accuracy is the most important criterion for comparing different models since the primary goal of the study is to classify the business delinquency. AUC is the second important criterion since it is a common measure to evaluate binary classification models. Although FPR and FNR are more or less equivalent in many common classification studies, in risk modeling and hazard studies, how-ever, they should be emphasized differently. This have been discussed in detail in the study of Begueria (2006). Similarly, for the study in this paper, FNR is weighted more heavily than FPR. It is because a model containing a great number of false positives can imply the beneficial loss of a potentially punctual business while a false negative error may signify the much larger loan loss from the business delinquency.

$$\text{TPR} = \frac{TP}{TP+FN} \quad (7)$$

$$\text{FPR} = \frac{FP}{FP+TN} \quad (8)$$

$$\text{FNR} = \frac{FN}{FN+TP} \quad (9)$$

3.3 The proposed framework

In this study, we propose a framework that aims at improving the performance of four binary classifiers for business delinquency classification through a series of model-improving methods. These model-improving methods consist of FS, hyper-parameter adjusting, and model ensembling. Fig. 1 shows the block diagram of the proposed framework. It contains four main stages and the details of each stage is described as follows:

In stage 1, the dataset was pre-processed following the steps described in Section 3.1. Then, four binary classifiers including LR, DT, NN, and SVM, were implemented on the training set and their performances were evaluated on the validation set using the criteria described in Section 3.2. These models are identified as baseline models and are used as benchmark models to evaluate the effect of model-improving methods in stages. As illustrated in Fig. 1, the baseline models obtained from stage 1 are labelled as LR(1), DT(1), NN(1), and SVM(1).

In stage 2, the first model-improving method, FS, is used on the dataset in order to filter out the best subset of the features. The three FS methods used in this study include Relief, information gain, and correlation. In this study, the FS parameter 'weight relation' in RapidMiner controls the number of selected features. To follow the naming convention, we use 'number of selected features' as the name for the parameter 'weight relation' in the rest of this paper. With each change in the value of 'number of selected features', the selected features were used on each of the four classifiers for classification and evaluation. Finally, the model with best performance would be stored to enter the next stage. This could identify the best FS method along with the parameter settings for each classifier. Take LR as an example. The above-mentioned three FS methods will be applied on the dataset and LR will be implemented along with each change in the value of 'number of selected features'. That is to say, for each of the FS methods, the values of 'number of selected features' will be set to 5, 10, 15, 20, 25, and 30 in this study (the reason of this setting is described in Section 4.1). For each change of 'number of selected features', a LR would be implemented resulting in 18 (i.e., three FS methods six values for the 'number of selected features') models. These models will be compared and the model with the best performance would be stored as LR(2). Similarly, DT(2), NN(2), and SVM(2) are obtained in stage 2.

In stage 3, the second model-improving method, hyper-parameter adjusting, is carried out to further improve the model performance. Although there are many hyper-parameters to be adjusted, only one critical hyper-parameter is considered for each classifier in this study. The adjusting of the hyper-parameters is shown as follows: adjusting the regularization method controlled by 'alpha' in LR; adjusting the depth of the tree structure controlled by 'maximal depth' in DT; adjusting the size of the hidden layer controlled by 'hidden layers' in NN; and adjusting the kernel function controlled by 'kernel type' in SVM. The classifiers are implemented after each adjustment and the model with the best performance would be stored. The best models acquired in this stage are labelled as LR(3), DT(3), NN(3), and SVM(3).

In stage 4, the last model-improving method, model ensembling, is performed on the models obtained from stage 3. Two ensembling methods, including bagging and boosting, are investigated in this study. The acquired models are labelled using the prefix of the ensembling method. For example, bagging and boosting method are used on LR(3) and the resulting models are labelled as LR(4bag) and LR(4boo), respectively.

Finally, the performance of the models obtained from each stage will be compared. This can evaluate the effect of the above-mentioned three model-improving methods for different classifiers comprehensively.

4 Experimental results

In this section, the results of the parameter settings in FS methods as well as the hyper-parameter settings in modeling procedures are demonstrated. The performance of the models is compared and discussed using the criteria discussed in Section 3.2. With respect to the analysis tool, hierarchical variable clustering is performed in SAS (version 9.4). RapidMiner Studio (version 9.0) is used for other analysis in the experimentation. Excluding the parameter settings discussed in this paper, other parameter values of the FS algorithms and the hyper-parameters values of the classifiers are the default settings in Rapid-Miner. All the experiments are performed on a desktop computer with 3.3 GHz Intel Core i7 processor, 16GB RAM, and macOS system.

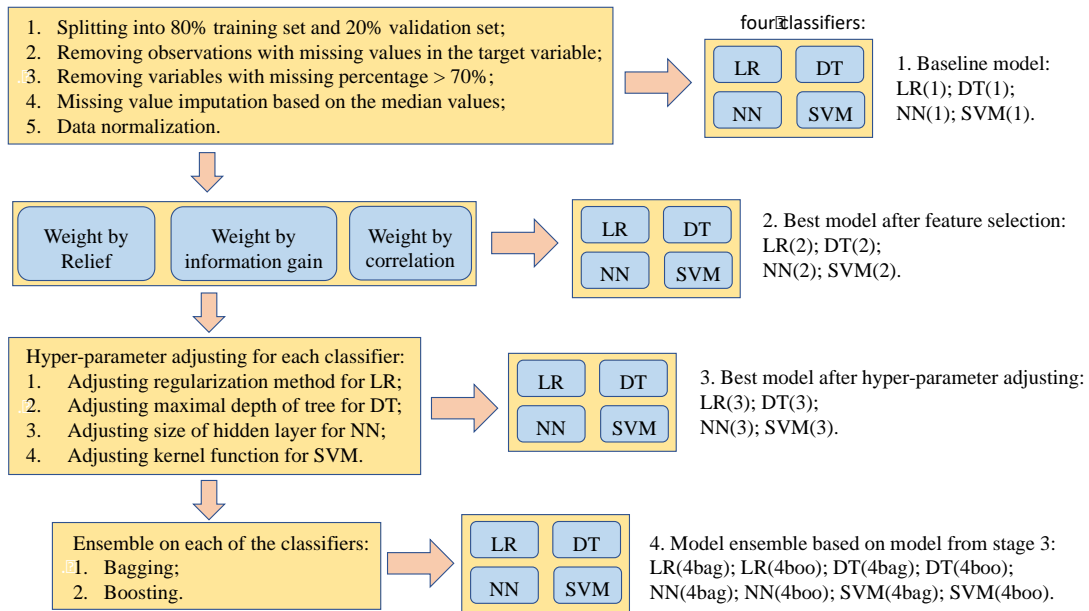


Fig. 1. The block diagram of the study framework.

4.1 Parameter settings of feature selection methods

The first model-improving method for improving risk modeling, FS along with the parameter settings, is implemented and discussed in this section. Although different FS methods have various parameters that need to be set, the parameter ‘number of selected features’ is investigated in more details in this study. This parameter plays a critical role in the modeling procedure for two reasons. One reason is that too many features tend to cause multicollinearity, which might affect the model performance. The other reason is that in risk modeling, a parsimonious while powerful model is preferred. Fig. 2 shows our initial analysis based on hierarchical variable clustering. It is found that 30 variables could explain around 80% variation of the original dataset. Therefore, for the value of the parameter ‘number of selected features’, a series of the values ranging from 5 to 30 were examined. The output of FS is then used as input variables on LR, DT, NN, and SVM classifiers.

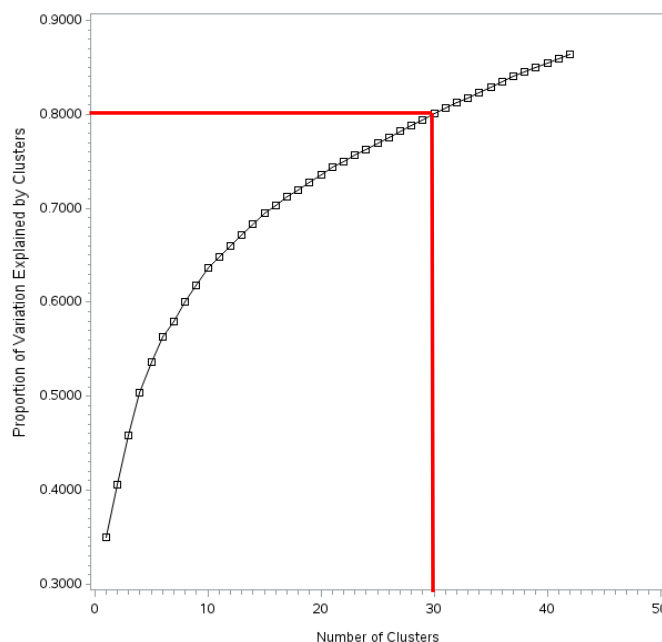


Fig. 2. Result of hierarchical variable clustering.

Parameter settings in Relief method Table 1 demonstrated the model performance by using the Relief FS method. LR4 could reach the highest accuracy of valued 0.948 by using the value of 20 for the parameter ‘number of selected features’. This kind of parameter setting could also reach the highest AUC as well as lowest FPR and FNR. By setting the parameter ‘number of selected features’ to 25, DT5 results in highest accuracy (0.975), the third highest AUC (0.978), the third lowest FPR (0.014), and the third lowest FNR (0.051), respectively. Similarly, the value of 30 for the parameter ‘number of selected features’ demonstrates the best performance on SVM6. Although NN4 and NN6 have the same values in accuracy and AUC, NN6 is considered to have better performance due to its lower FNR. Therefore, for the Relief FS method, the final settings of the parameter ‘number of selected features’ for LR, DT, NN, and SVM are 20, 25, 30, and 30, respectively. And the corresponding resulting models are LR4, DT5, NN6, and SVM6, respectively.

Table 1. Model performance based on different parameter settings in Relief feature selection method. LR, DT, NN, and SVM denote logistic regression, decision tree, neural network, and support vector machine, respectively.

Model	Number of Selected Features	Accuracy	AUC	FPR	FNR
LR1	5	0.858	0.898	0.064	0.333
LR2	10	0.888	0.960	0.046	0.274
LR3	15	0.933	0.970	0.018	0.933
LR4/LR(2)	20	0.948	0.978	0.014	0.145
LR5	25	0.935	0.971	0.025	0.162
LR6	30	0.935	0.974	0.021	0.162
DT1	5	0.943	0.974	0.057	0.060
DT2	10	0.950	0.980	0.053	0.043
DT3	15	0.970	0.977	0.068	0.970
DT4	20	0.970	0.984	0.011	0.077
DT5	25	0.975	0.978	0.014	0.051
DT6	30	0.960	0.934	0.004	0.128
NN1	5	0.928	0.966	0.042	0.145
NN2	10	0.943	0.981	0.039	0.103
NN3	15	0.968	0.983	0.103	0.968
NN4	20	0.970	0.991	0.011	0.077
NN5	25	0.968	0.991	0.014	0.077
NN6/NN(2)	30	0.970	0.991	0.018	0.060
SVM1	5	0.720	0.696	0.018	0.915
SVM2	10	0.723	0.788	0.021	0.897
SVM3	15	0.730	0.783	0.872	0.730
SVM4	20	0.733	0.794	0.018	0.872
SVM5	25	0.733	0.820	0.035	0.829
SVM6	30	0.750	0.799	0.060	0.709

Parameter settings in information gain method Table 2 shows the model performance by using information gain FS method. LR has the highest accuracy (0.935) by selecting either 15, 20, or 30 features via the information gain method. Considering that 30 features can result in relatively higher AUC and lower FNR compared with 15 and 20 features, LR12 is selected as the best model based on information gain method. According to Table 2, for DT, NN, and SVM methods, increasing numbers of selected features does not have too much effect in neither AUC nor FNR while could result in obvious changes in FPR. For DT and NN classifiers, the highest accuracy value could be reached in DT11 by selecting 25 features and in NN12 by selecting 30 features, respectively.

For SVM, by selecting 15, 20, and 25 features could result in the same values in accuracy and AUC. Considering that FNR is lower when 25 features are selected, we decide to select 25 features for SVM after FS method. In summary, for the information gain FS method, the final settings of the parameter ‘number of selected features’ for LR, DT, NN, and SVM are 30, 25, 30, and 25, respectively. And the corresponding resulting models are LR12, DT11, NN12, and SVM11, respectively.

Table 2. Model performance based on different parameter settings in information gain feature selection method. LR, DT, NN, and SVM denote logistic regression, decision tree, neural network, and support vector machine, respectively.

Model	Number of Selected Features	Accuracy	AUC	FPR	FNR
LR7	5	0.913	0.936	0.011	0.274
LR8	10	0.928	0.936	0.021	0.197
LR9	15	0.935	0.936	0.018	0.179
LR10	20	0.935	0.944	0.021	0.171
LR11	25	0.930	0.936	0.028	0.171
LR12	30	0.935	0.954	0.028	0.154
DT7	5	0.975	0.981	0.011	0.060
DT8	10	0.965	0.981	0.039	0.026
DT9	15	0.963	0.981	0.039	0.034
DT10	20	0.963	0.981	0.039	0.034
DT11/DT(2)	25	0.983	0.981	0.011	0.034
DT12	30	0.965	0.984	0.035	0.034
NN7	5	0.923	0.849	0.004	0.256
NN8	10	0.948	0.849	0.014	0.145
NN9	15	0.930	0.849	0.071	0.068
NN10	20	0.935	0.849	0.021	0.171
NN11	25	0.953	0.849	0.021	0.111
NN12	30	0.968	0.974	0.025	0.051
SVM7	5	0.913	0.967	0.011	0.274
SVM8	10	0.933	0.967	0.021	0.179
SVM9	15	0.953	0.967	0.011	0.137
SVM10	20	0.953	0.967	0.011	0.137
SVM11	25	0.953	0.967	0.014	0.128
SVM12	30	0.950	0.959	0.014	0.137

Parameter settings in correlation method The results of the model performance by using correlation FS method is presented in Table 3. As can be discerned, changing the parameter of ‘number of selected features’ does not change the FPR too much for all the four classifiers. Moreover, when number of selected features exceed 15, the FPR value does not change for DT, NN, and SVM. The highest values of accuracy are 0.945 on LR16, 0.963 on DT14, and 0.960 on SVM14, respectively. By selecting 10 features, NN14 can result in highest accuracy and AUC measures. Although NN14, NN15, and NN18 have the same highest accuracy, NN14 have the highest AUC and thus is considered as the best model. Therefore, for the correlation FS method, the final settings of the parameter ‘number of selected features’ for LR, DT, NN, and SVM are 20, 10, 10, and 10, respectively. And the corresponding resulting models are LR16, DT14, NN14, and SVM14, respectively.

Table 3. Model performance based on different parameter settings in correlation feature selection method. LR, DT, NN, and SVM denote logistic regression, decision tree, neural network, and support vector machine, respectively.

Model	Number of Selected Features	Accuracy	AUC	FPR	FNR
LR13	5	0.913	0.897	0.014	0.265
LR14	10	0.943	0.962	0.018	0.154
LR15	15	0.943	0.954	0.018	0.154
LR16	20	0.945	0.952	0.018	0.145
LR17	25	0.943	0.951	0.014	0.162
LR18	30	0.943	0.964	0.018	0.154
DT13	5	0.923	0.870	0.004	0.256
DT14	10	0.963	0.978	0.025	0.068
DT15	15	0.958	0.925	0.004	0.137
DT16	20	0.958	0.925	0.004	0.137
DT17	25	0.958	0.927	0.004	0.137
DT18	30	0.953	0.932	0.004	0.154
NN13	5	0.925	0.917	0.004	0.248
NN14	10	0.965	0.990	0.007	0.103
NN15	15	0.965	0.989	0.028	0.051
NN16	20	0.963	0.989	0.014	0.094
NN17	25	0.963	0.978	0.014	0.094
NN18	30	0.965	0.981	0.014	0.085
SVM13	5	0.918	0.812	0.004	0.274
SVM14/SVM(2)	10	0.960	0.971	0.014	0.103
SVM15	15	0.953	0.963	0.011	0.137
SVM16	20	0.953	0.943	0.011	0.137
SVM17	25	0.948	0.943	0.011	0.154
SVM18	30	0.953	0.960	0.011	0.137

Comparison of model performance by using three feature selection methods For each classifier, the best model based on each of the three FS methods is selected. The performance of these models is compared by using the accuracy measure. As mentioned above, LR4, DT5, NN6, and SVM6 are the best models based on Relief method. LR12, DT11, NN12, and SVM11 are the best models based on information gain method. LR16, DT14, NN14, and SVM14 are the best models based on correlation method. Fig. 3 compares the accuracy of these 12 models. The results demonstrate that different FS methods have large effects on SVM accuracy. Information gain and correlation methods outperform the Relief method on SVM. On the other hand, the effect on accuracy from different FS methods is not obvious for LR, DT, nor NN. The highest accuracy of LR, DT, NN, and SVM can be achieved by using FS methods of Relief ('number of selected features' valued 20), information gain ('number of selected features' valued 25), Relief ('number of selected features' valued 30), and correlation ('number of selected features' valued 10), respectively. The corresponding models are LR4, DT11, NN6, and SVM14, respectively. These four models are labelled as LR(2), DT(2), NN(2), and SVM(2) in stage 2 illustrated by Fig. 1 and will enter stage 3 for further analysis.

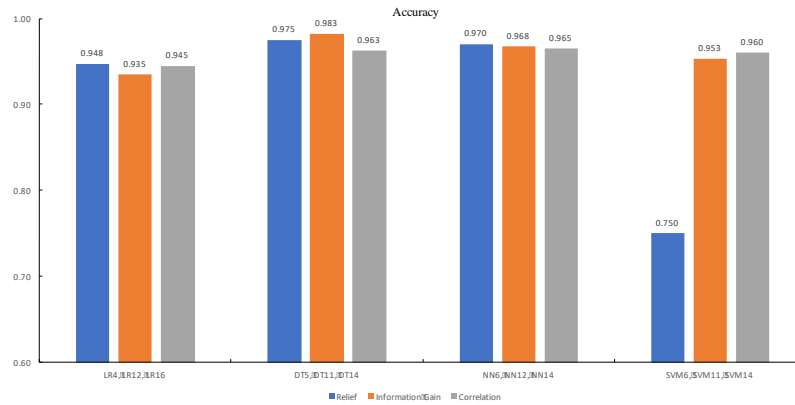


Fig. 3. Comparison of best classifiers based on three feature selection methods.

Fig. 4 demonstrated the further comparison of the performance of LR4/LR(2), DT11/DT(2), NN6/NN(2), and SVM14/SVM(2) using AUC, FPR, and FNR. As it is clear, there is no large difference in accuracy, AUC, and FPR of the four classifiers while FNR shows an obvious difference. LR(2) has the highest FNR value than the rest three classifiers, indicating the poorer power of identifying businesses’ delinquency of logistic models. DT(2) depicts the lowest FNR as well as the lowest FPR among the four classifiers, indicating the great potential in the role of classifying business delinquency.

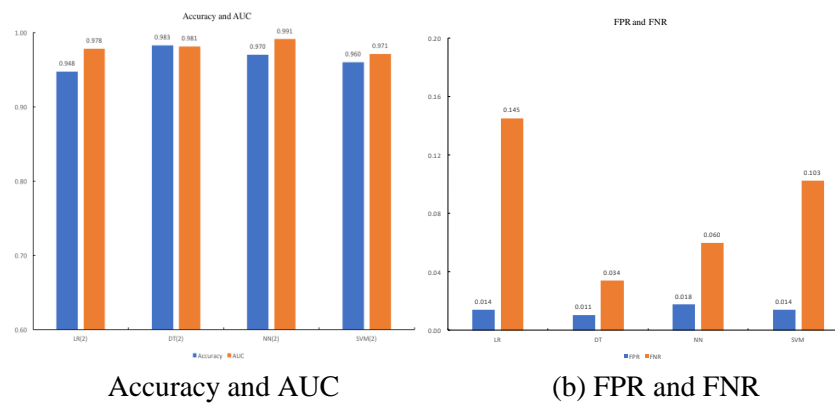


Fig. 4. Comparison of performance of LR4/LR(2), DT11/DT(2), NN6/NN(2), and SVM14/SVM(2).

4.2 Hyper-parameter settings of classifiers

The second model-improving method for improving risk modeling, hyper-parameter settings for binary classifiers, is implemented and discussed in this section. Starting from LR(2), DT(2), NN(2), and SVM(2), a series of different values of hyper-parameters was used for each classifier and the value resulting in the best performance was used as the final hyper-parameter setting.

Hyper-parameter settings in logistic regression In LR, the hyper-parameter ‘alpha’, which controls the distribution between the L1 and L2 penalty on the loss function, was altered. In RapidMiner, a value of 1 for alpha represents L1 penalty (i.e., LR with LASSO regularization), a value of 0 for represents L2 penalty (i.e., LR with ridge regularization), and a value of 0.5 represents a combination of L1 and L2 penalty (i.e., LR with elastic net regularization). Starting from LR4/LR(2), we consider the above-mentioned three regularizations. The results are demonstrated in Table 4. It is observed that the accuracy, AUC, FPR, and FNR are equal for models LR19, LR20, LR21, and LR22. This indicates that changing the regularization method (i.e., changing the setting of the hyper-parameter ‘alpha’) does not change the LR performance. In other words, it is shown that model regularization on LR is no long necessary after applying FS methods. Considering the parsimonious rule when selecting models, we selected LR without regularization (i.e., LR19) as the best model. This model is labelled as LR(3) as illustrated in Fig. 1 and will enter stage 4 for further analysis.

Hyper-parameter settings in decision tree In DT, the hyper-parameter ‘maximal depth’, which restricts the depth of the decision tree structure, was changed in RapidMiner. Since a too simple tree structure may produce poor performance while a too complex tree structure tends to result in overfitting, we limit the depth of the tree to between depths of 5 and 30. By changing the value of ‘maximal depth’, a series of experiments were implemented and DT algorithm was applied. As presented in Table 5, the performance of DT does not change with respect to accuracy, AUC, FPR, or FNR when the value of ‘maximal depth’ exceed 15. Therefore, the value of ‘maximal depth’ is set to 15 in this study and it is obvious that this setting can result in a higher accuracy compared to the default settings in RapidMiner (i.e., ‘maximal depth’ = 10). The corresponding model DT21 is labelled as DT(3) as illustrated in Fig. 1 and will enter stage 4 for further analysis.

Table 4. Logistic regression performance based on different settings of hyper-parameter ‘alpha’.

Model	Regularization	Alpha	Accuracy	AUC	FPR	FNR
LR19/LR(3)	None (default)	/	0.948	0.978	0.014	0.145
LR20	LASSO	1	0.948	0.978	0.014	0.145
LR21	Ridge	0	0.948	0.978	0.014	0.145
LR22	Elastic net	0.5	0.948	0.978	0.014	0.145

Hyper-parameter settings in neural network In NN, the hyper-parameter ‘hidden layers’, which determines the size of the hidden layer, was changed. The default setting for this hyper-parameter in RapidMiner is 1 plus half of the summation of number of attributes and number of classes. As mentioned in Section 4.1, by selecting 30 features using the Relief method can achieve the highest accuracy in NN. Therefore, for our study, the default value of ‘hidden layers’ is $1+(30+2)/2 = 17$. Since the larger size of NN tends to cause longer training time, we aimed at looking for a NN structure with a relatively small size, but strong performance. Therefore, in our experiment, the value of ‘hidden layers’ changes from 4 to 8 and the performance of the resulting NNs is demonstrated in Table 6. The highest accuracy (0.973) is achieved when value of ‘hidden layers’ is 6 and the corresponding model is NN20. Compared with the default setting in RapidMiner (i.e., hidden layers = 17), NN 20 results in a slightly lower AUC and higher FNR. On the other hand, NN20 with a hidden layer sized 6 could result in a lower FPR. Considering the much simpler structure of using a hidden layer sized 6 versus sized 17, we set the value of ‘hidden layers’ as 6 for the NN algorithm. The corresponding model NN20 is labelled as NN(3) as illustrated in Fig. 1 and will enter stage 4 for further analysis.

Table 5. Decision tree performance based on different settings of hyper-parameter ‘maximal depth’.

Model	Max depth	Accuracy	AUC	FPR	FNR
DT19	5	0.970	0.995	0.032	0.026
DT20	10 (default)	0.983	0.991	0.011	0.034
DT21/DT(3)	15	0.985	0.987	0.004	0.043
DT22	20	0.985	0.987	0.004	0.043
DT23	25	0.985	0.987	0.004	0.043
DT24	30	0.985	0.987	0.004	0.043

Table 6. Neural network performance based on different settings of hyper-parameter ‘hidden layers’.

Model	Max depth	Accuracy	AUC	FPR	FNR
NN19	4	0.963	0.982	0.007	0.111
NN20/NN(3)	6	0.973	0.971	0.004	0.085
NN21	8	0.960	0.992	0.014	0.103
NN22	17 (default)	0.970	0.991	0.018	0.060

Hyper-parameter settings in support vector machine In SVM, the hyper-parameter ‘kernel type’, which specifies the kernel function used, was altered. In this study, three types of the kernel function in RapidMiner were considered and compared: dot (i.e., inner product kernel function, default setting), polynomial (i.e., polynomial kernel function) and radial (i.e., radial basis kernel function). The performance of SVM based on different kernel functions are demonstrated in Table 7. It is obvious that radial basis kernel function has the best model performance with the highest accuracy, the highest AUC, the second lowest FPR and the lowest FNR. Then, the value of ‘kernel type’ is set to radial for SVM classifier. The corresponding model SVM21 is labelled as SVM(3) as illustrated in Fig. 1 and will enter stage 4 for further analysis.

Table 7. Support vector machine performance based on different settings of hyper-parameter ‘kernel type’

Model	Kernel type	Accuracy	AUC	FPR	FNR
SVM19	dot	0.960	0.971	0.014	0.103
SVM20	polynomial	0.930	0.958	0.004	0.231
SVM21/SVM(3)	radial	0.973	0.978	0.011	0.068

Comparison of model performance after hyper-parameter settings As mentioned above, the final hyper-parameter settings for LR, DT, NN, and SVM are as follows: LR without regularization, max depth valued 15, hidden layer sized 6, and radial as kernel function, respectively. Using these hyper-parameter settings, the resulting models are LR19/LR(3), DT21/DT(3), NN20/NN(3), and SVM21/SVM(3), respectively. Fig. 5 provides model performance with accuracy, AUC, FPR, and FNR measures for further comparison. It shows that LR(3) has the worst accuracy performance while DT(3) has the best. DT(3) also performs best with respect to the AUC, FPR, and FNR measures. Similar as the result demonstrated in Fig. 4, LR(3) has the highest FNR value than the rest three classifiers while DT(3) depicts the lowest FNR as well as the lowest FPR values. SVM(3) can reach as high an accuracy as LR(3). The performance of NN(3) and SVM(3) are better than LR(3) considering the FNR measures.

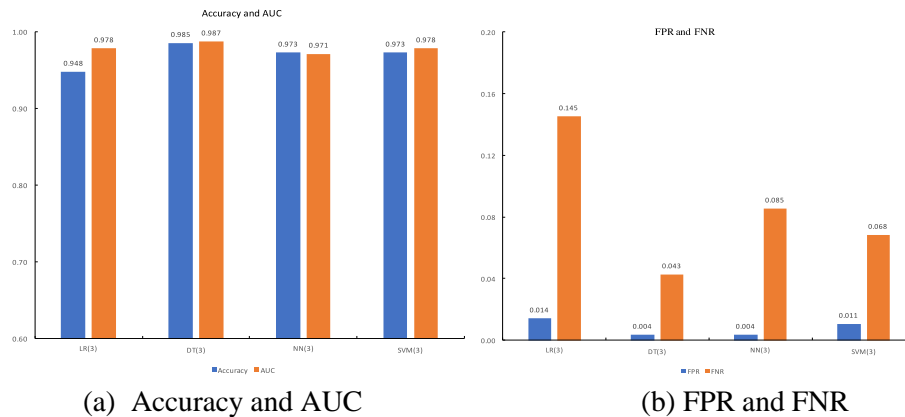


Fig. 5. Comparison of performance of R19/LR(3), DT21/DT(3), NN20/NN(3), and SVM21/SVM(3).

4.3 Ensemble models based on different classifiers along with their hyper-parameter settings

The third method for improving risk modeling, model ensembling, is implemented and discussed in this section. Starting from LR(3), DT(3), NN(3), and SVM(3), two ensembling techniques including bagging and boosting are considered in this study. The models LR(3), DT(3), NN(3), and SVM(3) are considered as base classifiers and the performance of the corresponding ensemble models are illustrated in Table 8. In general, ensembling on base classifiers including LR, DT, and NN could improve the model performance to some extent compared with the corresponding base classifiers. It is observed that by ensembling the base classifier LR, the performance has been improved by considering accuracy and FNR measures, with boosting method being superior than bagging. On the other hand, bagging on DT results in a decrease in accuracy while an increase in AUC compared with the base DT classifier. Both bagging and boosting on NN could further increase AUC and decrease FNR. However, ensembling on NN could not further improve accuracy.

After ensembling on SVM, the performance does not change with respect to accuracy, FPR, or FNR, indicating that SVM might not be an appropriate base learner to be ensembled. Or, we conclude that after FS and hyper-parameter adjusting, SVM could already reach a performance such that ensembling on SVM cannot further improve.

Table 8. Ensemble model performance based on different classifiers after FS and hyper-parameter adjusting. LR, DT, NN, and SVM denote logistic regression, decision tree, neural network, and support vector machine, respectively.

Model	Accuracy	AUC	FPR	FNR
LR(3)	0.948	0.978	0.014	0.145
LR(4bag)	0.950	0.978	0.014	0.137
LR(4boo)	0.987	0.936	0.014	0.010
DT(3)	0.985	0.987	0.004	0.043
DT(4bag)	0.975	0.998	0.021	0.034
DT(4boo)	0.980	0.997	0.011	0.043
NN(3)	0.973	0.971	0.004	0.085
NN(4bag)	0.973	0.994	0.007	0.077
NN(4boo)	0.973	0.984	0.018	0.051
SVM(3)	0.973	0.978	0.011	0.068
SVM(4bag)	0.973	0.980	0.011	0.068
SVM(4boo)	0.973	0.981	0.011	0.068

4.4 Summary and comparison of model performance after a series of model-improving methods

As mentioned above, this study aims at improving the performance of risk modeling through a series of methods, including examining different FS methods, adjusting hyper-parameters of different classifiers, and ensembling on the base classifiers. The performance of the models after applying model-improving methods has also been discussed. To further investigate the effect of these model-improving methods, we tested the performance of the baseline classifiers, i.e., implement the classifiers (LR, DT, NN, and SVM) on the original dataset. These models are labelled as LR(1), DT(1), NN(1), and SVM(1) as illustrated in Fig.1. Finally, the performance of the models from (1) baseline; (2) after FS; (3) after hyper-parameter adjusting on models obtained from (2); (4) bagging on the models obtained from (3); (5) boosting on the models obtained from (3) are summarized in Figs. 6, 7, 8, and 9 as a visual representation. As presented in these figures, the following results can be expressed based on our study:

(1): The three model-improving approaches in general have a positive effect on LR, with FS and model ensembling having a larger effect than hyper-parameter adjusting. According to Fig. 6, FS methods can significantly increase the model accuracy and decrease FNR by comparing LR(1) and LR(2). Compared with LR(2), LR(3) does not have much change in accuracy, AUC, FPR, or FNR, indicating that after FS, there is minimal effect of adjusting hyper-parameter on model performance. This indicates that model regularization through adjusting hyper-parameter 'alpha' of LR is not necessary after FS. Ensemble strategies (both bagging and boosting) can increase model accuracy and FNR measures, with boosting having a larger effect than bagging. However, with respect to FPR, it seems that none of the model-improving methods impact model performance.

(2): DT is not sensitive to any of the model-improving methods. According to Fig. 7, it is interesting to find that the baseline DT has the best performance compared to other DTs after applying model-improving methods, since the base-line DT can reach the highest accuracy and AUC, as well as the lowest FNR and FPR. This might be true because the baseline DT is built on the original dataset that contains all the variables. Moreover, DT is relatively powerful in dealing with multicollinearity, therefore retaining all variables will not hurt the model performance.

(3): The three model-improving approaches in general have a positive effect on NN and they mainly improve the performance with respect to FPR and FNR. The beneficial effect on accuracy is negligible. According to Fig. 8, FS methods can significantly increase the model accuracy and decrease FNR by comparing NN(1) and NN(2). This result is the same as that of LR. It is also found that hyper-parameter adjusting can significantly decrease FPR while hurt AUC by comparing NN(2) and NN(3). However, hyper-parameter adjusting can improve model performance with respect to FPR because NN(3) produces lower FPR than NN(2). It is worth noting that ensemble strategies (both bagging and boosting) can further improve model performance after hyper-parameter adjusting. This can be confirmed by comparing the AUC and FNR values of NN(3), NN(4), and NN(5).

(4): The three model-improving approaches show a positive effect on SVM. According to Fig. 9, FS can increase model accuracy and decrease FNR by comparing SVM(1) and SVM(2). This result is the same as those of LR and NN. SVM(3) shows a higher accuracy and AUC as well as a lower FNR and FPR than SVM(2), showing that after FS, adjusting hyper-parameters can further increase model performance. SVM(3), SVM(4bag) and SVM(4boo) show the same value in accuracy, FPR, and FNR, indicating that after FS and hyper-parameter adjusting, model ensemble cannot further improve model performance.

(5): LR(4boo) can reach the highest accuracy and the lowest FNR values among all the models. Recall that in Fig. 5, LR(3) shows the lowest accuracy and the highest FNR values compared with DT(3), NN(3), and SVM(3). However, after boosting, LR(4boo) becomes the best model. This indicates that the ensembling strategy (especially boosting) has the largest effect on LR. As the final result, the optimal candidate model for classifying business delinquency in this study is the boosting LR via Relief FS method.

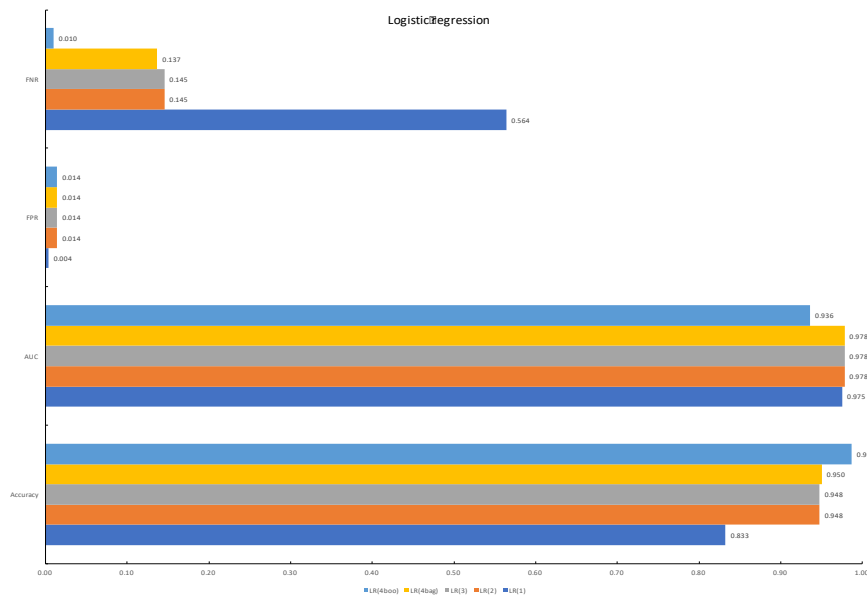


Fig. 6. Performance of LR via a series of model-improving methods.

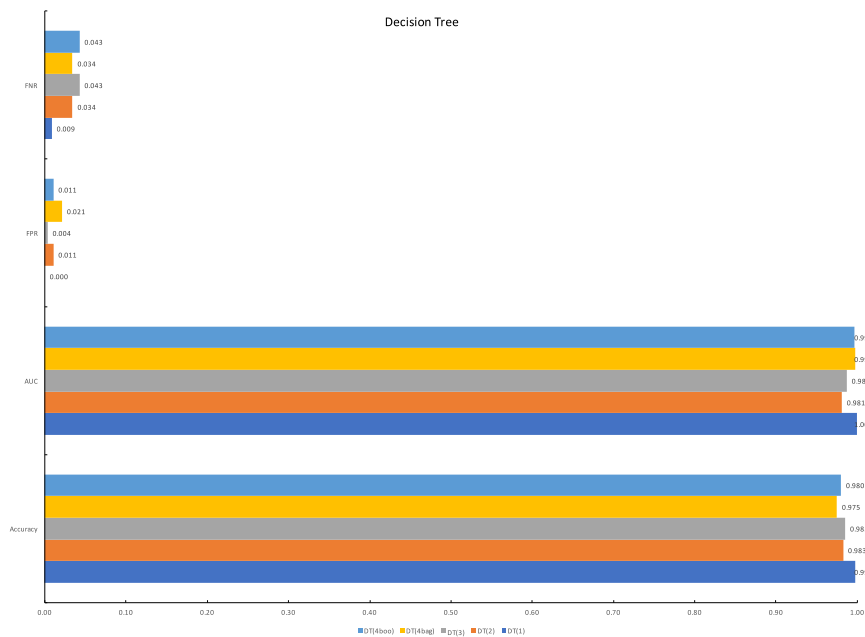


Fig. 7. Performance of DT via a series of model-improving methods.

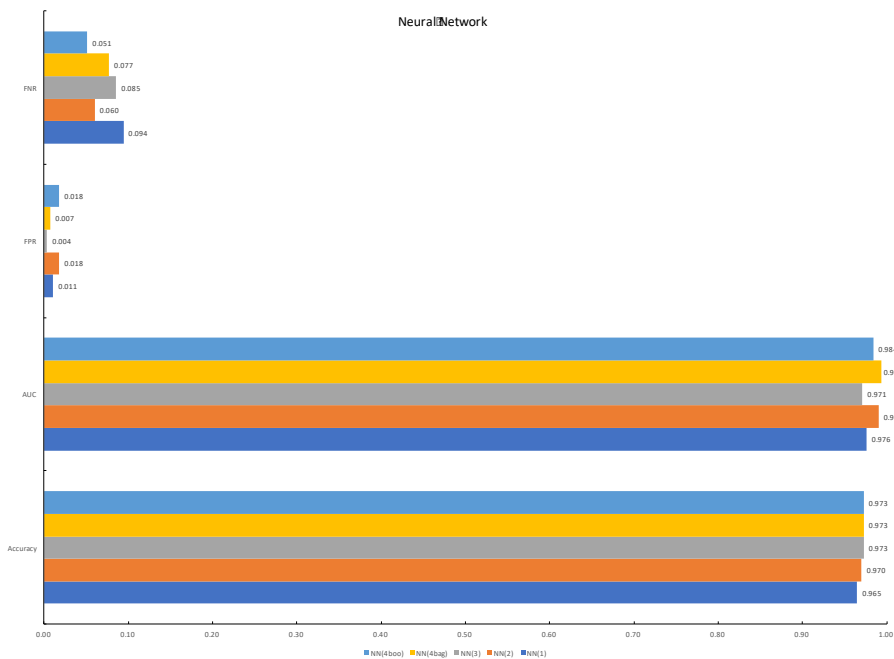


Fig. 8. Performance of NN via a series of model-improving methods.

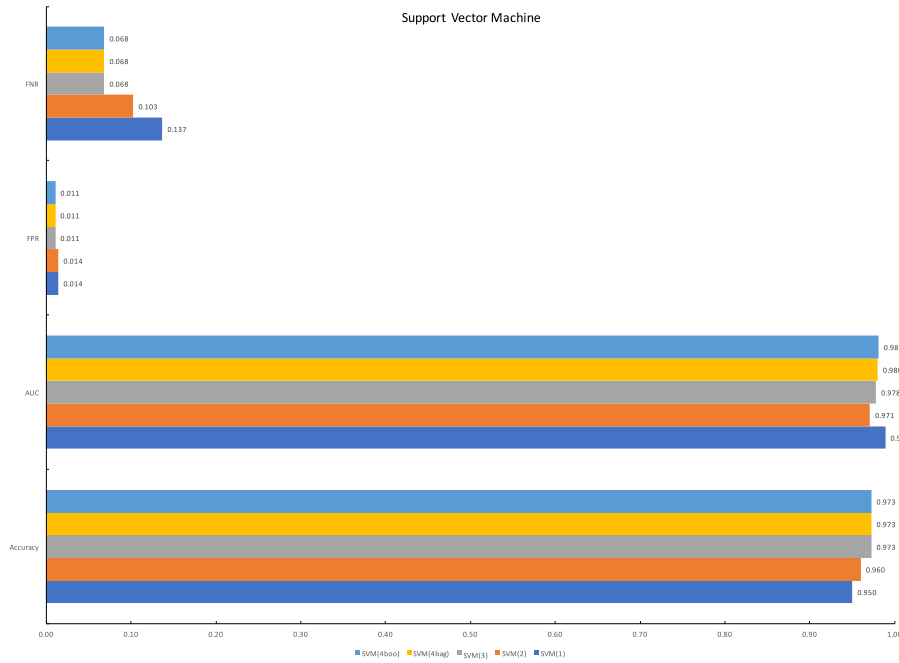


Fig. 9. Performance of SVM via a series of model-improving methods.

5 Conclusion

In this study, we proposed a framework that aims at improving business delinquency modeling via three model-improving methods: FS along with parameter settings, hyper-parameter adjusting for the classifiers, and model ensembling. The feasibility of the framework is assessed on the dataset that contains commercial information of de-identified companies in the US from 2006 to 2014. In our experiments, four binary classifiers including LR, DT, NN, and SVM are considered. For each classifier, three FS algorithms including weight by Relief, weight by information gain, and weight by correlation, are firstly applied on the dataset. By changing the parameters in the FS algorithms, each classifier would identify the best FS method along with the appropriate parameter settings by comparing the model performance resulting from each change of the parameter. Then, hyper-parameters for each classifier were adjusted to achieve a better model performance. Finally, model ensembling techniques, including bagging and boosting, are implemented on the classifiers.

Based on the dataset used in this study, the model-improving methods in general can improve the performance of LR, NN, and SVM. However, the effect varies on different classifiers. The best FS methods for LR, DT, NN, and SVM are Relief (‘number of selected features’ valued 20), information gain (‘number of selected features’ valued 25), Relief (‘number of selected features’ valued 30), and correlation (‘number of selected features’ valued 10), respectively. After FS, adjusting hyper-parameters of SVM results in a significant improvement in the model performance by considering accuracy, AUC, FPR, and FNR measures. However, model regularization via adjusting hyper-parameter ‘alpha’ of LR seems to be unnecessary after applying FS methods. The beneficial effect of model-improving methods is obvious on NN with respect to FPR and FNR while negligible in accuracy. It is worth noting that after FS and hyper-parameter adjusting, SVM is no longer an appropriate base classifier to be ensembled on, since ensembling on SVM cannot further improve model performance. On the contrary, ensembling on LR can largely increase its accuracy and decrease its FNR.

Another interesting finding is that DT is not sensitive to any of the model-improving methods and DT built on the original dataset has the best performance. The possible reason might be that building DT on the original dataset could use all the information contained by the dataset. Furthermore, DT is powerful at dealing with potential multicollinearity issues. Therefore, the advantage from keeping all the variables offsets the disadvantage caused by the potential multicollinearity issues in the DT classifier. As the final result of this study, the optimal candidate model for classifying business delinquency is the boosting LR via Relief FS method.

The proposed framework provides a comprehensive approach for simultaneous applications of FS, hyper-parameter adjusting, and model ensembling techniques to improve the performance of business delinquency modeling. Because of using different datasets in the business delinquency classification problems, the results might not be consistent in the future studies. However, the proposed framework in this study may serve as a good guidance and reference for future researchers when simultaneously using different strategies for improving model performance.

Bibliography

- F. Akthar, C. Hahne, et al. Rapidminer 5. Operator Reference. www. rapid-i.com, 2012.
- E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning*, 36(1):2, 1998.
- S. Beguer a. Validation and evaluation of predictive models in hazard assessment and risk management. *Natural Hazards*, 37(3):315{329, 2006.
- Y.-S. Chen. Classifying credit ratings for asian banks using integrating feature selection and the cpda-based rough sets approach. *Knowledge-Based Systems*, 26:259-270, 2012.
- P.-A. Cornillon and E. Matzner-Lober. Ridge et lasso. *Regression avec R*, pages 169-190, 2011.
- N. Cristianini, J. Shawe-Taylor, et al. An introduction to support vector machines and other kernel-based learning methods. Cambridge university press, 2000.
- K. Gurney. An introduction to neural networks. CRC press, 2014.
- I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157{1182, 2003.
- M. Hecht, Y. Bromberg, and B. Rost. Better prediction of functional effects for sequence variants. *BMC genomics*, 16(8):S1, 2015.
- C.-L. Huang and C.-J. Wang. A ga-based feature selection and parameters optimization for support vector machines. *Expert Systems with applications*, 31(2):231-240, 2006.
- A. E. Khandani, A. J. Kim, and A. W. Lo. Consumer credit-risk models via machine-learning algorithms. *Journal of Banking & Finance*, 34(11):2767{-2787, 2010.
- H. Kim and Z. Gu. A logistic regression analysis for predicting bankruptcy in the hospitality industry. *The Journal of Hospitality Financial Management*, 14(1):17-34, 2006.
- K. Kira and L. A. Rendell. The feature selection problem: Traditional methods and a new algorithm. In Aaai, volume 2, pages 129-134, 1992a.
- K. Kira and L. A. Rendell. A practical approach to feature selection. In *Machine Learning Proceedings 1992*, pages 249-256. Elsevier, 1992b.
- F. N. Koutanaei, H. Sajedi, and M. Khanbabaei. A hybrid data mining model of feature selection algorithms and ensemble learning classifiers for credit scoring. *Journal of Retailing and Consumer Services*, 27:11-23, 2015.
- S. Le Cessie and J. C. Van Houwelingen. Ridge estimators in logistic regression. *Applied statistics*, pages 191{201, 1992.
- S. Menard. *Applied logistic regression analysis*, volume 106. SAGE publications, 2018.
- A. Naik and L. Samant. Correlation review of classification algorithm using data mining tool: Weka, rapidminer, tanagra, orange and knime. *Procedia Computer Science*, 85:662-668, 2016.
- A. Salappa, M. Doumpos, and C. Zopounidis. Feature selection algorithms in classification problems: An experimental evaluation. *Optimisation Methods and Software*, 22(1):199-212, 2007.
- Y.-Y. Song and L. Ying. Decision tree methods: applications for classification and prediction. *Shanghai archives of psychiatry*, 27(2):130, 2015.
- C.-F. Tsai and J.-W. Wu. Using neural network ensembles for bankruptcy pre-diction and credit scoring. *Expert systems with applications*, 34(4):2639-2649, 2008.
- G. Wang and J. Ma. A hybrid ensemble approach for enterprise credit risk assessment based on support vector machine. *Expert Systems with Applications*, 39(5):5325-5331, 2012.

- J. Wang, A.-R. Hedar, S. Wang, and J. Ma. Rough set and scatter search metaheuristic based feature selection for credit scoring. *Expert Systems with Applications*, 39(6):6123-6128, 2012.
- Y. Wang and J. L. Priestley. Binary classification on past due of service accounts using logistic regression and decision tree. 2017.
- D. West. Neural network credit scoring models. *Computers & Operations Re-search*, 27(11-12):1131-1152, 2000.
- I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.
- Y. Xia, C. Liu, Y. Li, and N. Liu. A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring. *Expert Systems with Applications*, 78:225-241, 2017.
- T. Yamane. *Statistics: An introductory analysis*. 1973.
- J. Yao and C. Lian. A new ensemble model based support vector machine for credit assessing. *International Journal of Grid and Distributed Computing*, 9(6):159-168, 2016.
- D. Zhang, X. Zhou, S. C. Leung, and J. Zheng. Vertical bagging decision trees model for credit scoring. *Expert Systems with Applications*, 37(12):7838-7843, 2010.
- Y. Zhou, M. Han, L. Liu, J. S. He, and Y. Wang. Deep learning approach for cyberattack detection. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 262-267. IEEE, 2018.