Ensemble Learning or Deep Learning? Application to Default Risk Analysis

Hamori, Shigeyuki
Kawai, Minami
Kume, Takahiro
Murakami, Yuji
Watanabe, Chikara

Ensemble Learning or Deep Learning? Application to Default Risk Analysis

Shigeyuki Hamori
Minami Kawai
Takahiro Kume
Yuji Murakami
Chikara Watanabe

January 2018

Discussion Paper No.1802

GRADUATE SCHOOL OF ECONOMICS

KOBE UNIVERSITY

ROKKO, KOBE, JAPAN

Ensemble Learning or Deep Learning? Application to Default Risk Analysis

Shigeyuki HAMORI [1]*, Minami KAWAI [2] , Takahiro KUME [3] , Yuji MURAKAMI [4] , and Chikara WATANABE [5]

[1] (*Corresponding author)
Graduate School of Economics, Kobe University; hamori@econ.kobe-u.ac.jp,
[2] Department of Economics, Kobe University; minami.hehe@gmail.com
[3] Department of Economics, Kobe University; takahiro-2479@outlook.jp
[4] Department of Economics, Kobe University; yuji.murakami0410@gmail.com
[5] Department of Economics, Kobe University; 4751.power.wc@gmail.com

Abstract: Proper credit risk management is essential for lending institutions as substantial losses can be incurred when borrowers default. Consequently, statistical methods that can measure and analyze credit risk objectively are becoming increasing important. This study analyzed default payment data from Taiwan and compared the prediction accuracy and classification ability of three ensemble learning methods—specifically, Bagging, Random Forest, and Boosting—with those of various neural network methods, each of which has a different activation function. The results indicate that Boosting has a high prediction accuracy, whereas that of Bagging and Random Forest is relatively low. They also indicate that the prediction accuracy and classification performance of Boosting is better than that of deep neural networks, Bagging, and Random Forest.

Keywords: credit risk; ensemble learning; deep learning; bagging; random forest; boosting; deep neural network.

Acknowledgments: An early version of this paper was read at the Workshop of Big Data and Machine Learning. We are grateful to Prof. Zheng Zhang for helpful comments and suggestions. This research was supported by a grant-in-aid from The Nihon Hoseigakkai Foundation.

1. Introduction

Credit risk management is essential for financial institutions whose core business is lending. Thus, accurate consumer or corporation credit assessment is of utmost importance because significant losses can be incurred by financial institutions when borrowers default. To control their losses from uncollectable accounts, financial institutions therefore need to properly assess borrowers' credit risks. Consequently, they endeavor to collate borrower data and various statistical methods have been developed to measure and analyze credit risk objectively.

Because of its academic and practical importance, much research has been conducted on this issue. For example, Boguslauskas and Mileris (2009) analyzed credit risk using Lithuanian data for 50 cases of successful enterprises and 50 cases of bankrupted enterprises. Their results indicated that artificial neural networks are an efficient method to estimate the credit risk in banks. Yeh and Lien (2009) compared the predictive accuracy of probability of default among six data mining methods (specifically, K-nearest neighbor classifier, logistic regression, discriminant analysis, naive Bayesian classifier, artificial neural networks, and classification trees) using customers' default payments data in Taiwan. Their experimental results indicated that only artificial neural network can accurately estimate default probability. Khashman (2010) employed neural network models for credit risk evaluation with German credit data comprising 1000 cases: 700 instances of creditworthy applicants, and 300 instances where credit is not creditworthy[1]. The results obtained indicated that the accuracy rate for the training data and test data was 99.25% and 73.17%, respectively. In this data, however, if one always predicts that the case is creditworthy, then the accuracy rate naturally becomes 70%. Thus, the results imply that there is only a 3.17% gain for the prediction accuracy of test data using neural network models. Gante et al. (2015) also used the German credit data and compared twelve neural network models to assess credit risk. Their results indicated that a neural network with 20 input neurons, 10 hidden neurons, and one output neuron is a good neural network model for use in a credit risk evaluation system

In this study, we first employed models to predict the default risk based on clients' attributes using machine learning methods and then compared their prediction accuracy. Specifically, we employed three ensemble learning methods—Bagging, Random Forest, and Boosting—and multiple deep learning methods, with different activation functions. The performance of the methods were then compared in terms of their ability to predict the default risk using multiple indicators (accuracy rate of prediction results, receiver operating characteristic (ROC) curve, and area under the curve (AUC)).

For the customers' default payments data in Taiwan, we first conducted data mining analysis using original series and standardized data. The use of Taiwan data is beneficial for us because the sample size of the default payment data in Taiwan is 30,000 and is much larger

[1] The German credit dataset is publicly available at UCI Machine Learning data repository, https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data).

than that of German data. Next, we compared the prediction accuracy of each method using the three ensemble learning methods Bagging, Random Forest, and Boosting, and multiple deep learning methods with different activation functions.

The results obtained indicated that prediction accuracy rate is relatively higher when Boosting is used, and relatively lower when Bagging and Random Forest are used. Further, the performance of Boosting was even better than that of deep neural network (DNN). The ROC curve and the AUC value also supported these results.

The remainder of this paper is organized as follows. Section 2 summarizes the basic properties of ensemble learning and deep learning. Section 3 explores the data employed. Section 4 discusses the empirical results obtained. Section 5 presents concluding remarks.

2. Ensemble Learning and Deep Learning

2.1. Ensemble learning

There is an English adage that states the following: "Two heads are better than one." This means that, even for ordinary persons, two or more people focusing on a specific task result in a more positive outcome than only one person. This is the basis for ensemble learning, in which data-analysis algorithms are provided that resemble such ordinary daily knowledge of multiple persons. More formally, ensemble learning is a machine learning method whereby multiple models are created from different samples, after which these models are combined and integrated to improve accuracy. The methods used for this combination and integration are, in the case of regression problems, averages, and in the case of classification problems, majority-rule decision. Three ensemble learning algorithms were employed in this study: Bagging, Random Forest, and Boosting.[2]

Bagging, developed by Breiman (1996), is a machine learning method that uses bootstrapping to create multiple training datasets from given datasets. The classification results generated using the data are arranged and combined to improve the prediction accuracy. Because the bootstrap samples are mutually independent, learning can be carried out in parallel. Figure 1 summarizes the basic idea underlying Bagging.

Random Forest, also proposed by Breiman (2001), is similar to Bagging. It is a machine learning method in which the classification results generated from multiple training datasets are arranged and combined to improve the prediction accuracy. However, whereas Bagging uses all input variables to create each decision tree, Random Forest uses subsets that are random samplings of input variables to create each decision tree. This means that Random Forest is better suited than Bagging for the analysis of high-dimensional data. Figure 2 summarizes the basic idea underlying Random Forest.

[2] For the details of ensemble learning, see Jin (2017).

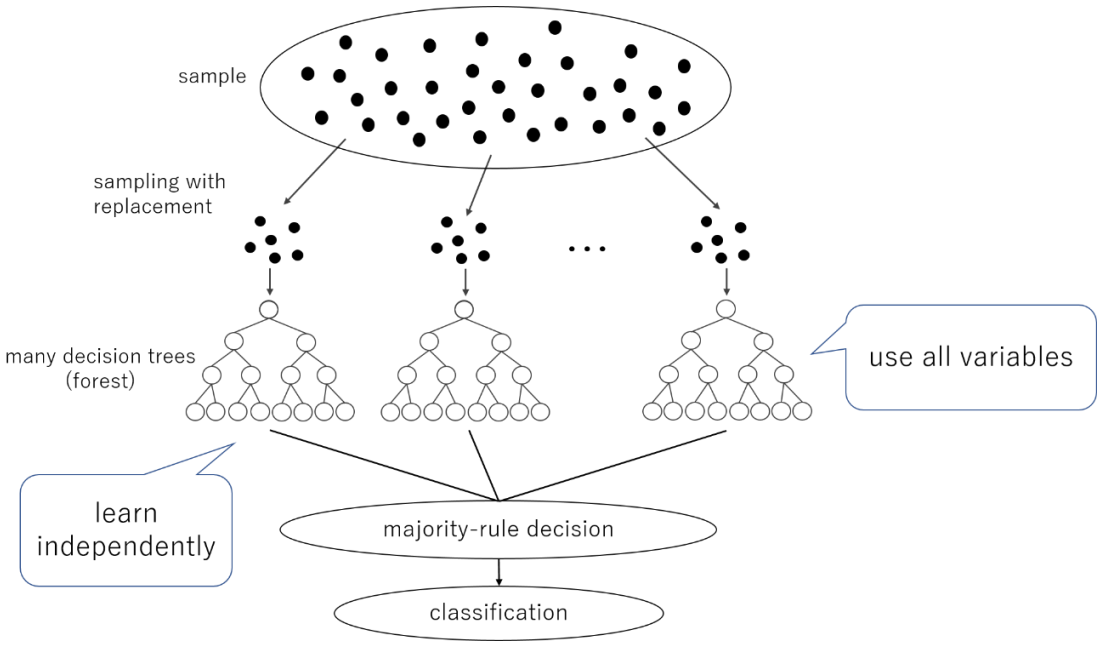


Figure 1. Basic idea underlying Bagging.

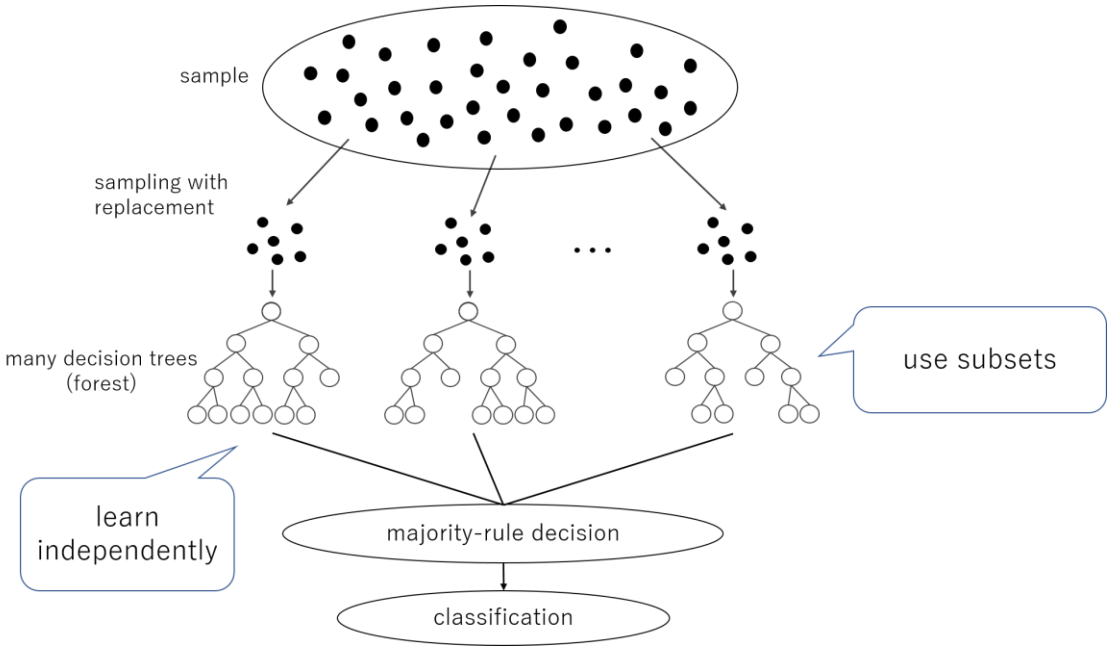


Figure 2. Basic idea underlying Random Forest.

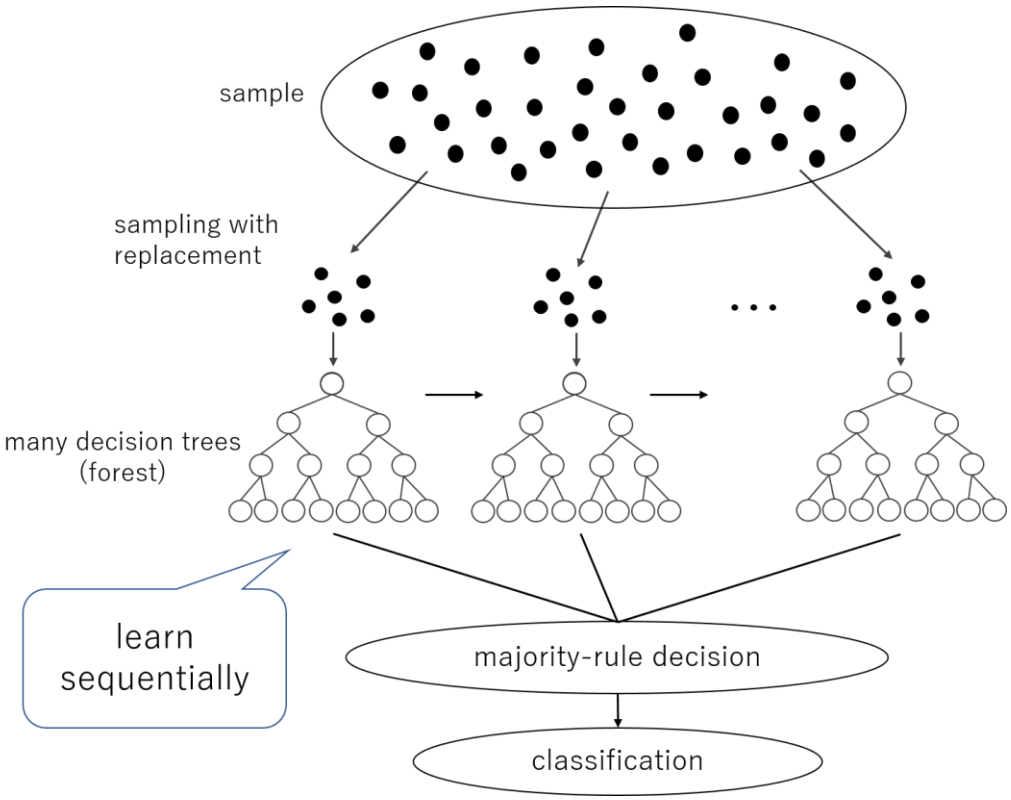


Figure 3. Basic idea underlying Boosting.

Boosting is also a machine learning method. Whereas Bagging and Random Forest employ independent learning, Boosting employs sequential learning. In Boosting, on the basis of supervised learning of data, weights are successively adjusted, and multiple learning results are sought. These results are then combined and integrated to improve accuracy. The most widely used Boosting algorithm is AdaBoost, proposed by Freund and Shapire (1996). Figure 3 summarizes the basic idea underlying Boosting.

2.2. Deep learning

A neural network is network structure comprising multiple connected units. The neural network configuration is determined by the manner in which the units are connected; different configurations enable a network to have different functions and characteristics. The feed-forward neural network is the most frequently used neural network model. Figure 4 illustrates a feed-forward neural network configured by the hierarchical connection of multiple units. (Note that the middle layer is not limited to a single layer.) When the number of middle layers is greater than or equal to two, the network is called a DNN. In Figure 4, units are arranged into three parts, i.e., input layer, middle layer, and output layer. The outputs of each unit in the input layer and the middle layer are linked to all of the units in the next layer. This kind of model is called a "fully connected" neural network.

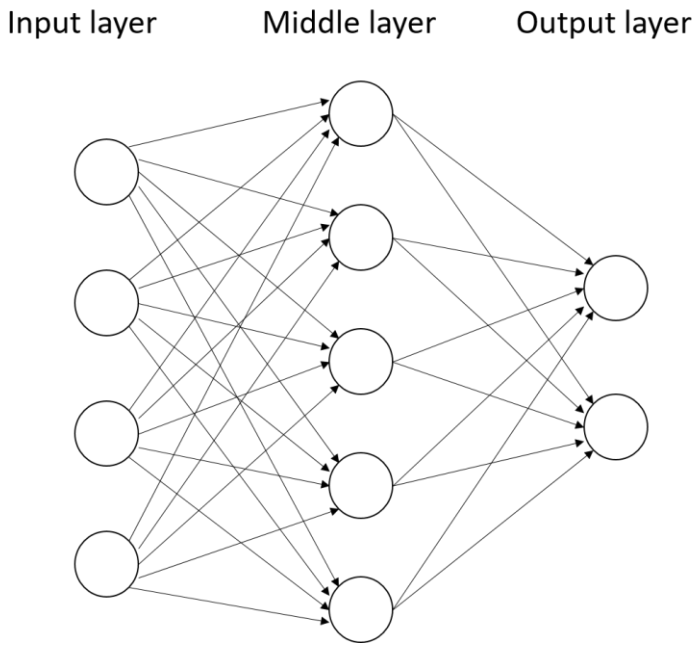


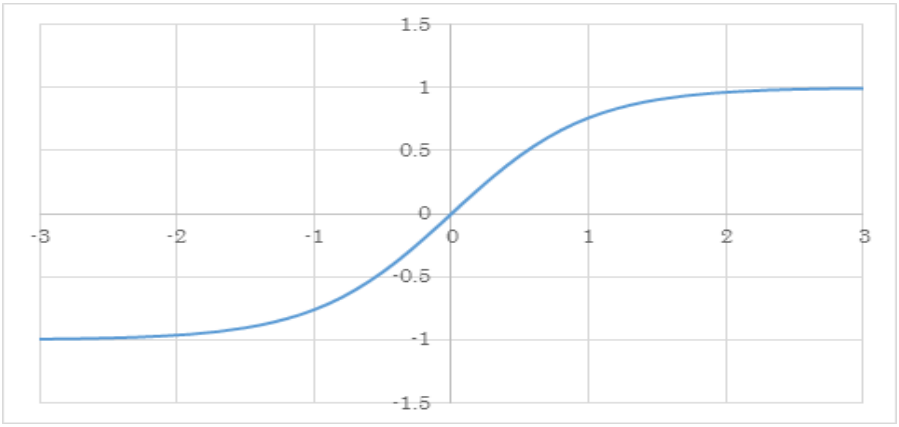
Figure 4. Structure of a Neural Network.

The activation function in a neural network is very important as it expresses the functional relationship between the input and output in each unit. In this study, we employed two kinds of activation functions: tanh and rectified linear unit (ReLU). These functions are defined as follows:

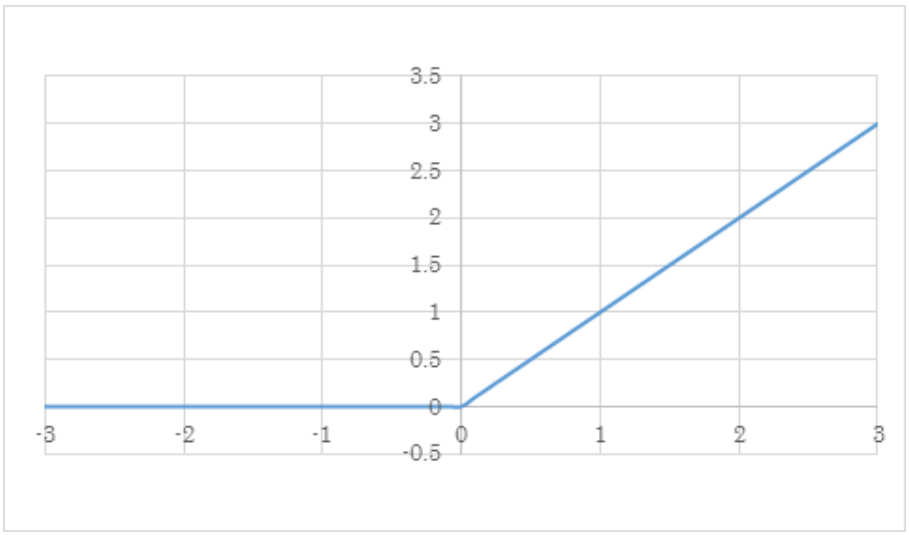
$$\text{tanh:} \quad f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

$$\text{ReLU:} \quad f(x) = \max(0, x)$$

Figures 5(a) and 5(b) illustrate the tanh function and the ReLU function, respectively.



(a)



(b)

Figure 5. (a) tanh function, (b) ReLU function

The tanh function squashes a real-valued number to the range [-1, 1]. Its activations saturate, and its output is zero-centered. The ReLU function is an alternative activation function in neural networks.[3] One of its major benefits is the reduced likelihood of the gradient to vanish.

Although DNNs are a powerful machine learning tool, they are susceptible to overfitting. This is addressed using a technique called dropout, in which units are randomly dropped (along with their incoming and outgoing connections) in the network (Figure 6). This prevents units from overly co-adapting (Srivastava et al. 2014).

[3] See LeCun et al. (2015)

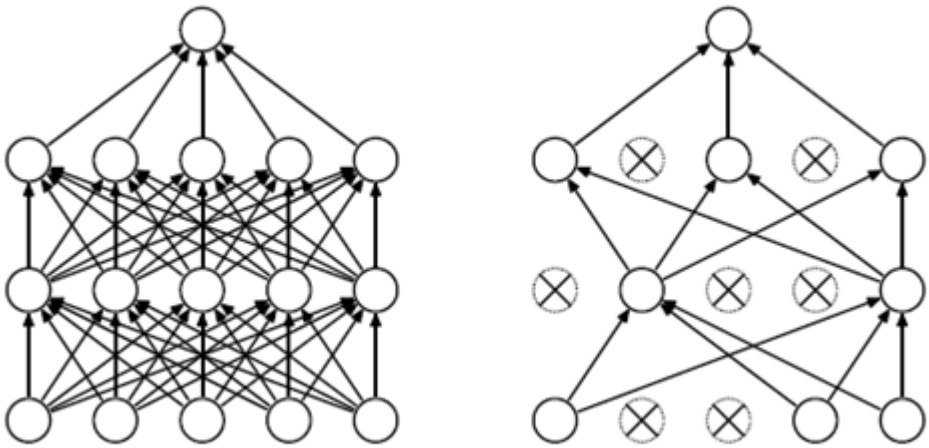


Figure 6. Illustration of the dropout technique. (Source: Srivastava et al (2014)).

3. Data

The same Taiwan payment data used by Yeh and Lien (2009) were employed in this study. The data are available as the default credit card clients' dataset in the UCI Machine Learning Repository. In the dataset used by Yeh and Lien (2009), the number of observations is 25000, in which 5529 observations are the case with default payment. However, the dataset in the UCI Machine Learning Repository has a total number of observations of 30000, in which 6636 observations are cases with default payment. Following Yeh and Lien (2009), we used default payment (No = 0, Yes = 1) as the explained variable and following 23 variables as explanatory variables:

X1: Amount of given credit (NT dollar).

X2: Gender (1 = male; 2 = female).

X3: Education (1 = graduate school; 2 = university; 3 = high school; 4 = others).

X4: Marital status (1 = married; 2 = single; 3 = others).

X5: Age (year).

X6–X11: History of past payment tracked via past monthly payment records (-1 = payment on time; 1 = payment delay for one month; 2 = payment delay for two months; …; 8 = payment delay for eight months; 9 = payment delay for nine months and above).

X6: Repayment status in September 2005.

X7: Repayment status in August 2005.

X8: Repayment status in July 2005.

X9: Repayment status in June 2005.

X10: Repayment status in May 2005.

X11: Repayment status in April 2005.

X12: Amount on bill statement in September 2005 (NT dollar).

X13: Amount on bill statement in August 2005 (NT dollar).
X14: Amount on bill statement in July 2005 (NT dollar).
X15: Amount on bill statement in June 2005 (NT dollar).
X16: Amount on bill statement in May 2005 (NT dollar).
X17: Amount on bill statement in April 2005 (NT dollar).
X18: Amount of previous payment in September 2005 (NT dollar).
X19: Amount of previous payment in August 2005 (NT dollar).
X20: Amount of previous payment in July 2005 (NT dollar).
X21: Amount of previous payment in June 2005 (NT dollar).
X22: Amount of previous payment in May 2005 (NT dollar).
X23: Amount of previous payment in April 2005 (NT dollar).

Because of the high proportions of no-default observations (77.88%), the accuracy rate inevitably remains at virtually 78% when all observations are used for analysis. As a result, in this study we extracted 6,636 observations randomly from all no-default observations to ensure that no-default and default observations re equal, thereby preventing distortion. As regards the ratio of training to test datasets, our study used two cases, i.e., 90% to 10% and 75% to 25%.

In accordance with the statement by Khashman (2009) that explanatory variables should be normalized, we normalized the data based on the following formula:

$$z_i = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}}$$

where z_i is normalized data, x_i is each dataset, x_{min} is the minimum value of x_i, and x_{max} is the maximum value of x_i. Further, we analyzed both normalized and original data to compare their accuracy.

To compare the accuracy of the models, we utilized accuracy rate and ROC curve. Furthermore, we repeated the experiments 100 times and calculated the average and standard deviation of the accuracy rate for each dataset.[4]

4. Results

We implemented the methods in R—specifically, "ipred" package for Bagging, "randomForest" for Random Forest, "ada" package for Boosting, and "h2o" package for DNN. Further, we analyzed the prediction accuracy rate of each method for the two cases; i.e.,

[4]We used set.seed(50) to remove the difference caused by random numbers in drawing the ROC curve and calculating AUC.

original and normalized data. Then, we examined the classification ability of each method based on the ROC curve and the AUC value.

Tables 1(a) and 1(b) summarize the results obtained using the original data. The tables show that Boosting has the best performance and gives more than a 70 percent prediction accuracy rate on average, with a small standard deviation for both training and test data. None of the neural network models exceed a 70 percent average accuracy rate. Further, they have a relatively large standard deviation for test data. Thus, it is clear that Boosting achieves a higher accuracy prediction than neural network. Bagging and Random Forest have a 50% to 60% prediction accuracy rate for test data and a nearly 70% prediction accuracy rate for training data. In addition, the difference of the ratio between training and test data (90%:10% or 75%:25%) does not have an obvious influence on the result of our analysis.[5]

Tables 2(a) and 2(b) summarize the results obtained using normalized data. The tables show that Boosting has the highest accuracy rate on test data, which is similar to the results obtained for the original data case. The average accuracy rate for Boosting is more than 70 percent and it has the smallest standard deviation for both training and test data. None of the deep learning methods has an average prediction accuracy rate exceeding 70 percent. Further, they have a large standard deviation for test data. The prediction accuracy rate of Bagging and Random Forest does not reach 60% on average for test date, which similar to the case for the original data. In addition, the the difference of ratio between training and test data (90%:10% or 75%:25%) does not have a major influence on the result, which is similar to the case with the original data. Our comparison of the results of the original data with the results of the normalized data revealed no significant difference in prediction accuracy rate based on type of data.

Next, we analyzed the classification ability of each method by examining the ROC curve and the AUC value. When considering whether a model is appropriate, it is not sufficient to rely solely on accuracy rate. The ratio of correctly identified instances in the given class is called the true positive rate. The ratio of incorrectly identified instances in the given class is called the false positive rate. When the false positive rate is plotted on the horizontal axis and the true positive rate on the vertical axis, the combination of these produces an ROC curve. A good model is one that shows a high true positive rate value when the false positive value is low. The AUC refers to the area under the ROC curve. A perfectly random prediction yields an AUC of 0.5. In other words, the ROC curve is a straight line connecting the origin (0,0) and the point (1,1).

[5] The number of units in the middle layers of NN and DNN is determined based on the Bayesian optimization method. (See Appendix for details.)

Table 1(a). Prediction accuracy of each method.
(Original data: The ratio of training and test data is 75% to 25%).

Method			Data	Ratio of Training and Test Data (%)	Accuracy ratio of training data		Accuracy ratio of test data		AUC
					average(%)	standard deviation	average(%)	standard deviation	
Bagging			Original	75:25	80.13	0.003	55.98	0.008	0.575
Boosting			Original	75:25	71.66	0.003	71.06	0.008	0.781
Random Forest			Original	75:25	69.59	0.544	58.50	0.844	0.604
Method			Data	Ratio of Training and Test Data (%)	Accuracy ratio of training data		Accuracy ratio of test data		AUC
Model	Activation function	Middle layer			average(%)	standard deviation	average(%)	standard deviation	
DNN	Tanh	2	Original	75:25	70.66	0.721	68.93	0.972	0.758
NN	Tanh	1	Original	75:25	71.01	0.569	69.59	0.778	0.765
DNN	Tanh with Dropout	2	Original	75:25	58.47	3.566	58.46	3.404	0.607
NN	Tanh with Dropout	1	Original	75:25	67.27	1.237	67.14	1.341	0.708
DNN	ReLu	2	Original	75:25	69.57	0.707	68.61	0.863	0.756
NN	ReLu	1	Original	75:25	68.81	0.708	68.30	1.008	0.754
DNN	ReLu with Dropout	2	Original	75:25	69.97	0.903	69.01	0.956	0.756
NN	ReLu with Dropout	1	Original	75:25	70.12	0.637	69.48	0.881	0.765

Table 1(b). Prediction accuracy of each method.
(Original data: The ratio of training and test data is 90% to 10%).

Method			Data	Ratio of Training and Test Data (%)	Accuracy ratio of training data		Accuracy ratio of test data		AUC
					average(%)	standard deviation	average(%)	standard deviation	
Bagging			Original	90:10	79.58	0.003	56.23	0.015	0.575
Boosting			Original	90:10	71.57	0.003	70.88	0.011	0.761
Random Forest			Original	90:10	68.55	0.453	58.77	1.331	0.599
Method			Data	Ratio of Training and Test Data (%)	Accuracy ratio of training data		Accuracy ratio of test data		AUC
Model	Activation function	Middle layer			average(%)	standard deviation	average(%)	standard deviation	
DNN	Tanh	2	Original	90:10	69.64	0.683	69.31	1.325	0.760
NN	Tanh	1	Original	90:10	70.49	0.550	69.61	1.312	0.761
DNN	Tanh with Dropout	2	Original	90:10	57.29	3.681	57.27	4.117	0.642
NN	Tanh with Dropout	1	Original	90:10	66.37	1.619	66.25	1.951	0.714
DNN	ReLu	2	Original	90:10	69.49	0.695	68.76	1.408	0.771
NN	ReLu	1	Original	90:10	69.16	0.728	68.54	1.261	0.751
DNN	ReLu with Dropout	2	Original	90:10	69.74	0.796	68.84	1.438	0.752
NN	ReLu with Dropout	1	Original	90:10	70.26	0.573	69.55	1.210	0.771

Table 2(a). Prediction accuracy of each method.
(Normalized data: The ratio of training and test data is 75% to 25%).

Method			Data	Ratio of Training and Test Data (%)	Accuracy ratio of training data		Accuracy ratio of test data		AUC
					average(%)	standard deviation	average(%)	standard deviation	
Bagging			Normalized	75:25	80.12	0.003	56.15	0.008	0.575
Boosting			Normalized	75:25	71.66	0.004	70.95	0.007	0.769
Random Forest			Normalized	75:25	69.67	0.565	58.39	0.880	0.605
Method			Data	Ratio of Training and Test Data (%)	Accuracy ratio of training data		Accuracy ratio of test data		AUC
Model	Activation function	Middle layer			average(%)	standard deviation	average(%)	standard deviation	
DNN	Tanh	2	Normalized	75:25	71.14	0.732	68.75	0.912	0.753
NN	Tanh	1	Normalized	75:25	70.64	0.652	69.42	0.763	0.768
DNN	Tanh with Dropout	2	Normalized	75:25	57.00	4.324	56.69	4.485	0.600
NN	Tanh with Dropout	1	Normalized	75:25	68.09	0.641	68.01	0.904	0.704
DNN	ReLu	2	Normalized	75:25	70.37	0.627	69.35	0.856	0.751
NN	ReLu	1	Normalized	75:25	70.92	0.615	69.37	0.943	0.757
DNN	ReLu with Dropout	2	Normalized	75:25	70.00	0.811	68.96	0.946	0.765
NN	ReLu with Dropout	1	Normalized	75:25	70.25	0.692	69.56	0.813	0.767

Table 2(b). Prediction accuracy of each method.
(Normalized data: The ratio of training and test data is 90% to 10%).

Method			Data	Ratio of Training and Test Data (%)	Accuracy ratio of training data		Accuracy ratio of test data		AUC
					average(%)	standard deviation	average(%)	standard deviation	
Bagging			Normalized	90:10	79.54	0.003	56.28	0.013	0.575
Boosting			Normalized	90:10	71.50	0.003	70.80	0.012	0.785
Random Forest			Normalized	90:10	68.66	0.475	58.83	1.368	0.600
Method			Data	Ratio of Training and Test Data (%)	Accuracy ratio of training data		Accuracy ratio of test data		AUC
Model	Activation function	Middle layer			average(%)	standard deviation	average(%)	standard deviation	
DNN	Tanh	2	Normalized	90:10	70.18	0.698	69.35	1.382	0.761
NN	Tanh	1	Normalized	90:10	70.52	0.594	69.51	1.309	0.763
DNN	Tanh with Dropout	2	Normalized	90:10	58.04	5.134	58.14	5.016	0.650
NN	Tanh with Dropout	1	Normalized	90:10	67.33	1.285	67.13	1.787	0.697
DNN	ReLu	2	Normalized	90:10	71.41	0.710	69.17	1.334	0.758
NN	ReLu	1	Normalized	90:10	69.55	0.772	68.97	1.426	0.759
DNN	ReLu with Dropout	2	Normalized	90:10	69.76	0.785	69.13	1.426	0.771
NN	ReLu with Dropout	1	Normalized	90:10	69.88	0.701	69.25	1.279	0.781

The various graphs in Figure 7 show ROC curves for the cases using the normalized data and the ratio between the training and test data at 75% to 25%. In each figure, sensitivity (vertical axis) corresponds to the true positive ratio, whereas 1 - specificity (horizontal axis) corresponds to the false positive ratio. The graphs indicate that the ROC curve for Boosting and some DNNs have desirable properties. The ROC curves for DNN also show good performance except for the case for tanh activation function with dropout.

The AUC values for all cases were also presented in Tables 1(a), 1(b), 2(a), and 2(b) above. The tables show that the highest AUC value was obtained for Boosting on both original and normalized data. Thus, the classification ability of Boosting is superior to that of neural networks. The second highest AUC value was obtained for neural networks for both data types. The AUC value of Bagging and Random Forest is approximately 0.60 for both data types, and their classification abilities are relatively weak.[6]

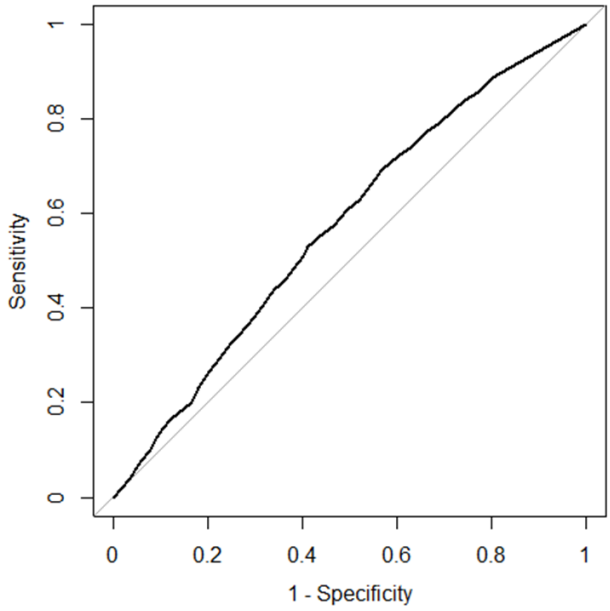


Figure 7(a). ROC curve for Bagging (normalized data at 75% to 25**%).**

[6] Boosting has the same AUC value in Tables 1(a), 1(b), 2(a), and 2(c). However, its precise value is 0.5748 for Table 1(a), 0.5751 for Table 1(b), 0.5750 for Table 2(a), and 0.5746 for Table 2(b).

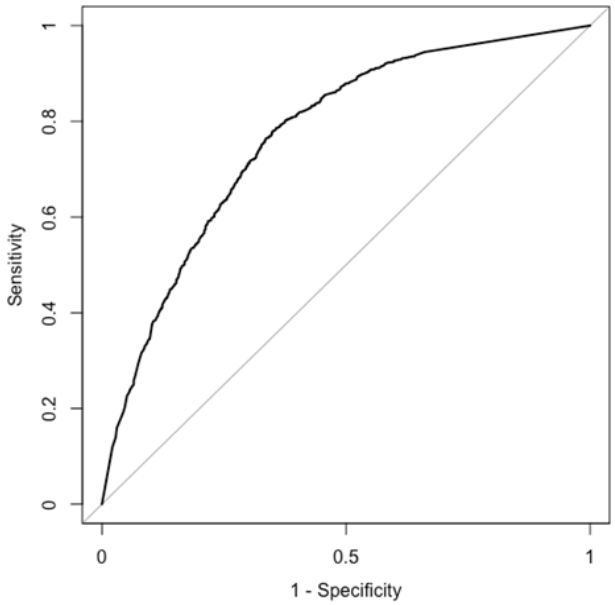


Figure 7(b). ROC curve for Boosting (normalized data at 75% to 25%).

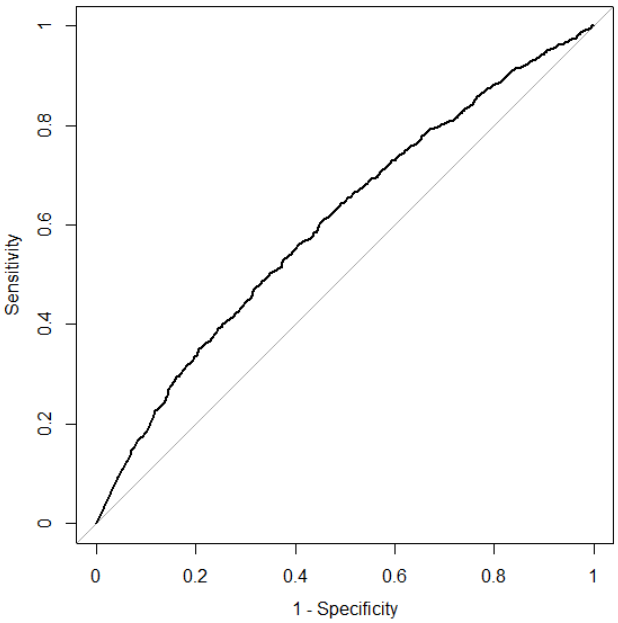


Figure 7(c). ROC curve for Random Forest (normalized data at 75% to 25%).

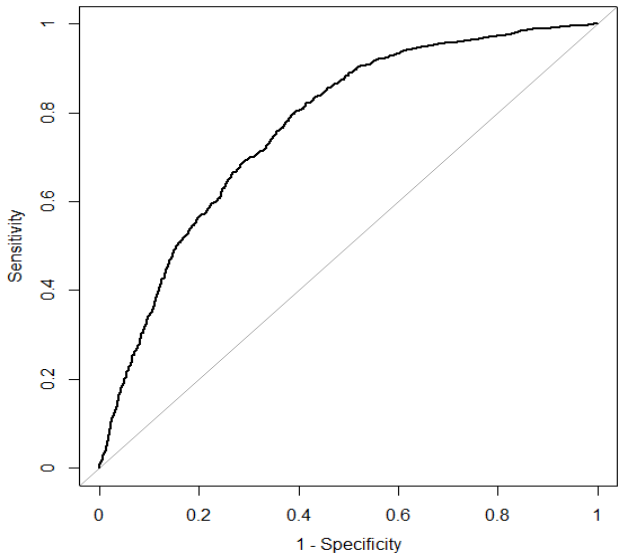


Figure 7(d). ROC curve for DNN Tanh and one middle layer (normalized data at 75% to 25%).

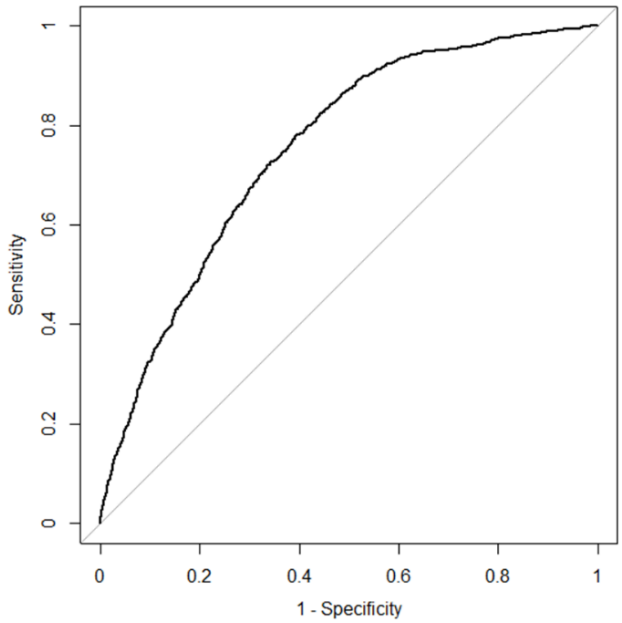


Figure 7(e). ROC curve for DNN Tanh and two middle layers (normalized data at 75% to 25%).

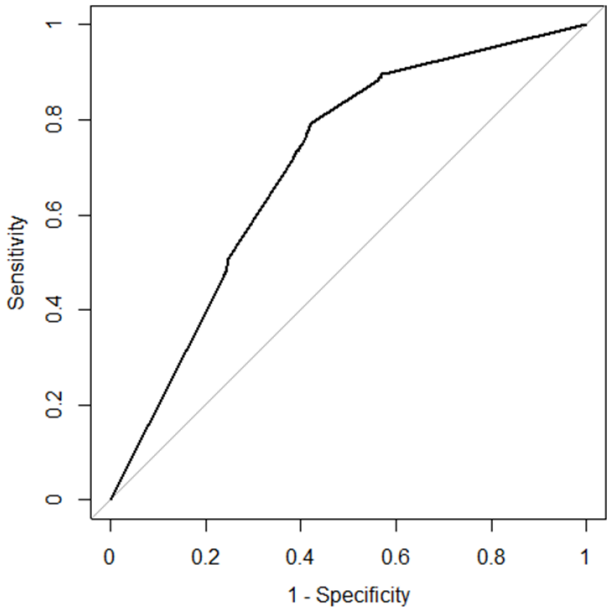


Figure 7(f). ROC curve for DNN Tanh with Dropout and one middle layer (normalized data at 75% to 25%).

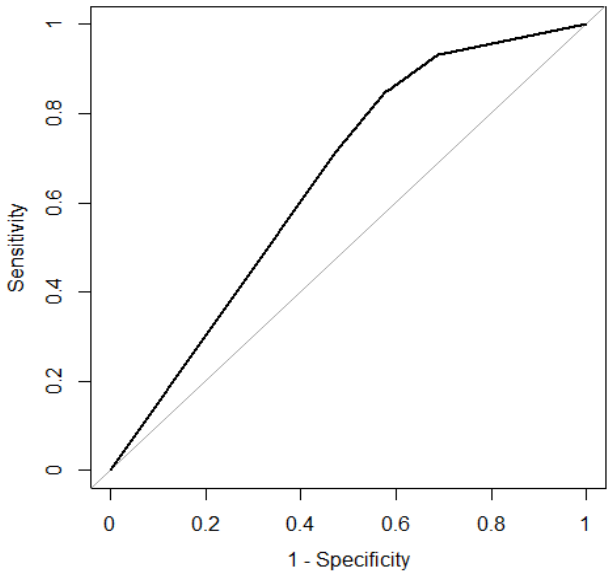


Figure 7(g). ROC curve for DNN Tanh with Dropout and two middle layers (normalized data at 75% to 25%).

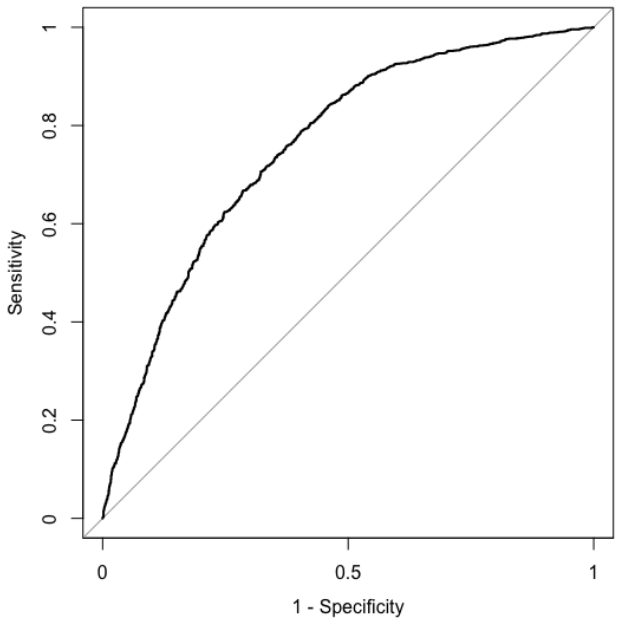


Figure 7(h). ROC curve for DNN ReLU and one middle layer (normalized data at 75% to 25%).

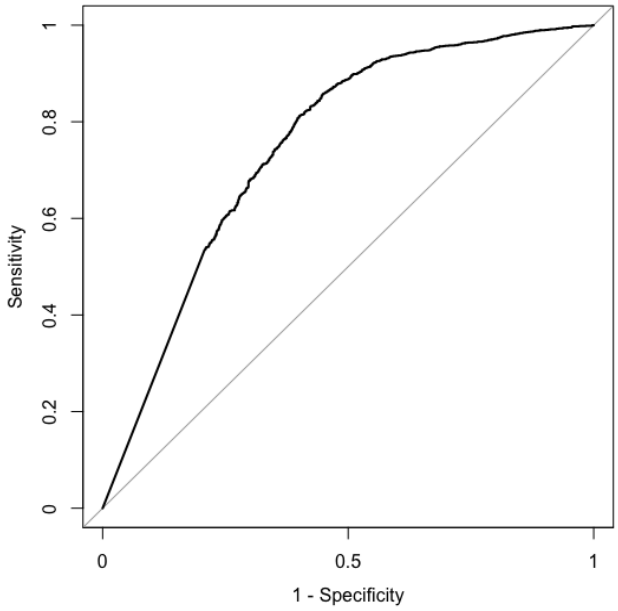


Figure 7(i). ROC curve for DNN with ReLU and two middle layers (normalized data at 75% to 25%).

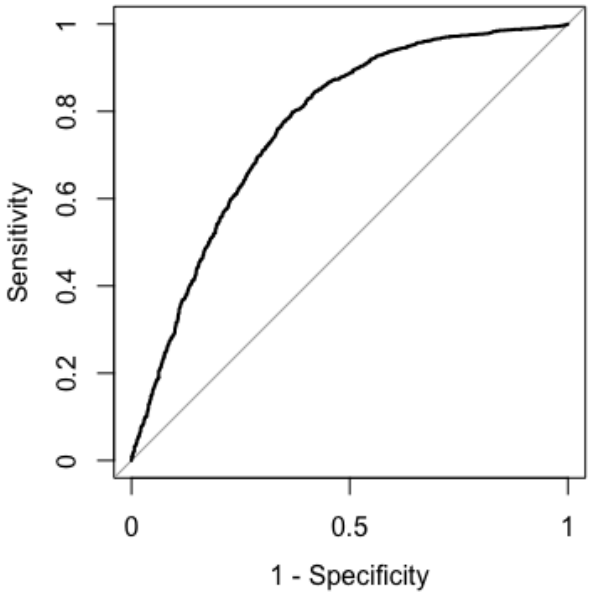


Figure 7(j). ROC curve for DNN ReLU with Dropout and one middle layer (normalized data at 75% to 25%).

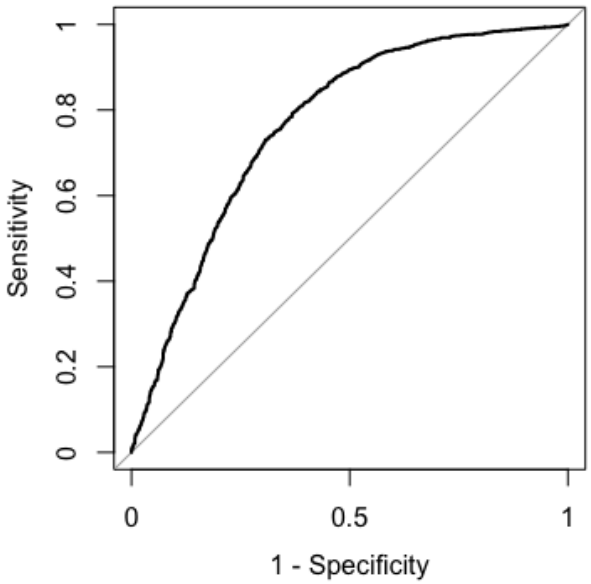


Figure 7(k). ROC curve for DNN ReLU with Dropout and two middle layers (normalized data at 75% to 25**%).**

5. Conclusions

In this study, we analyzed the default payment data in Taiwan and compared the prediction accuracy and classification ability of the three ensemble learning methods Bagging, Random Forest, and Boosting with those of various neural network methods, each of which has a different activation function. The results obtained indicate that Boosting has a higher accuracy rate than Bagging and Random Forest. Furthermore, they indicate that Boosting has better prediction accuracy and classification ability than DNNs, Bagging, and Random Forest. The usability of deep learning has recently been the focus of much attention, but our results indicate that DNNs are not panacea especially for relatively small sample. Therefore, it is necessary to make effective use of other methods such as Boosting.

Appendix

Table A1. Results based on number of middle layers in the DNN: Bayesian optimization method.

(a)

Method	Data	Ratio of Training and Test Data (%)	Input layer	Middle layer 1	Output layer
Tanh	Original	75:25	23	7	2
Tanh	Original	90:10	23	5	2
Tanh with Dropout	Original	75:25	23	14	2
Tanh with Dropout	Original	90:10	23	12	2
ReLu	Original	75:25	23	3	2
ReLu	Original	90:10	23	7	2
ReLu with Dropout	Original	75:25	23	14	2
ReLu with Dropout	Original	90:10	23	19	2
Tanh	Normalized	75:25	23	5	2
Tanh	Normalized	90:10	23	5	2
Tanh with Dropout	Normalized	75:25	23	5	2
Tanh with Dropout	Normalized	90:10	23	10	2
ReLu	Normalized	75:25	23	11	2
ReLu	Normalized	90:10	23	4	2
ReLu with Dropout	Normalized	75:25	23	16	2
ReLu with Dropout	Normalized	90:10	23	12	2

(b)

Method	Data	Ratio of Training and Test Data (%)	Input layer	Middle layer 1	Middle layer 2	Output layer
Tanh	Original	75:25	23	5	17	2
Tanh	Original	90:10	23	2	9	2
Tanh with Dropout	Original	75:25	23	9	7	2
Tanh with Dropout	Original	90:10	23	3	11	2
ReLu	Original	75:25	23	4	6	2
ReLu	Original	90:10	23	4	9	2
ReLu with Dropout	Original	75:25	23	13	9	2
ReLu with Dropout	Original	90:10	23	5	20	2
Tanh	Normalized	75:25	23	6	17	2
Tanh	Normalized	90:10	23	4	3	2
Tanh with Dropout	Normalized	75:25	23	9	4	2
Tanh with Dropout	Normalized	90:10	23	3	18	2
ReLu	Normalized	75:25	23	4	6	2
ReLu	Normalized	90:10	23	10	7	2
ReLu with Dropout	Normalized	75:25	23	16	9	2
ReLu with Dropout	Normalized	90:10	23	5	21	2

References

Boguslauskas, V. and R. Mileris, 2009. Estimation of credit risks by artificial neural networks models. *Izinerine Ekonomika-Engerrring Economics*, 4: 7-14.

Breiman, L. 1996. Bagging predictors. M*achine Learning*, 24: 123-140.

Breiman, L. 2001. Random forests. *Machine Learning*, 45: 5-32.

Gante, D.D., B.D. Gerardo, and B.T. Tanguilig, 2015.Neural network model using back propagation algorithm for credit risk evaluation. Paper presented at the 3rd International Conference on Artificial Intelligence and Computer Science, (AICS2015), pp.12 – 13.

Freund, Y. and R.E. Schapire, 1996. Experiments with a new boosting algorithm. Paper presented at the Thirteenth International Conference on Machine Learning, pp. 148-156.

Jin M., 2017. *Data Science Using R*, Morikita Shoten.

Khashman, A. 2010. Neural networks for credit risk evaluation: Investigation of different neural models and learning schemes. *Expert Systems with Applications*, 37: 6233–6239

LeCun, Y., Y. Bengio, and G. Hinton, 2015. Deep learning. *Nature*, 521(7553): 436-444.

Schapire, R.E. 1999. A brief introduction to boosting. Paper presented at the Sixteenth International Joint Conference on Artificial Intelligence, pp. 1-6.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1): 1929-1958.

Yeh, I-C. and C-H. Lien, 2009. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, 36: 2473–2480.