A novel Intelligent Credit Scoring Method using MOPSO

Yan Guo[1], Chao Dong[2]

1. Ningbo Institute of Technology, Zhejiang University, Ningbo 315100, China
E-mail: guoyanbox@126.com

2. Ningbo Dahongying University, Ningbo 315175, China
E-mail: 2821625@sina.com

Abstract: We present an intelligent credit scoring method to categorize credit applicants. Then, a novel multi-objective credit scoring model is proposed in this paper. In term of the defects of linear discriminant analysis (LDA): lack of accuracy, a multi-objective particle swarm optimization for credit scoring is designed in this paper. Finally, through the experiments with two real-world data set and one benchmark data set, we compare our approach with NaiveBayes, Logistic Regression (LR), Sequential Minimal Optimization (SMO), Neural Networks (NN), and Decision Trees (DT), the results of experiments demonstrate our proposed method outperforms the abovementioned five data-driven counterparts in term of accuracy and specificity while maintaining acceptable sensitivity.

Key Words: Credit scoring, Data classification, Particle swarm optimization

1. Introduction

Credit risk evaluation decision is a crucial issue for banking industry because even a one percent improvement in early detection of bad credit account may avoid huge amount of losses [1, 2]. Credit scoring model is the most successful method that helps financial institution to decide whether to grant or refuse a loan [3].

The popular methods used in building credit scoring models include Linear Discriminant Analysis (LDA) [2, 4], Logistic Regression (LR) [5, 6], Neural Networks (NN) [7,8], Support vector machines (SVM) [9,10] , Decision Tree (DT) [11,12] and Evolutionary computation techniques [13, 14].

We present a novel credit score method using LDA, and discriminate "good credit" group and "bad credit" group through comparing credit score with cutoff. Compared with NN and SVM, our credit scoring approach is more intelligent and easy to be implemented.

2. Mathematic model

Linear discriminant analysis (LDA) classifies two or more group using a set of independent variables. [15] LDA classifies credit applicants based on their discriminant score, which is calculated by a discriminant function.

$$Score_i = A_i X = a_{i1}x_1 + a_{i2}x_2 + ... + a_{iR}x_R$$

where $Score_i$ is the credit score of record i, R is the number of attribution, $X = (x_1,...,x_R)^T$ is the weight set of attribution, and $A_i = (a_{i1}, a_{i2},..., a_{iR})$ is the value set of record i.

For two-group classification, if $Score_i \geq b$, record i belongs to class 0 (Negative); and if $Score_i < b$, record i belongs to class 1 (Positive), where b is a boundary (cutoff), $i = 1,..., N$, and N is the sample size. The aim of credit scoring based on LDA is to determine the best coefficients of the variables, denoted by $X = (x_1,...,x_R)^T$, and value b (a scalar) to separate two classes: *Good* (Class 0)and *Bad* (Class 1). For credit scoring practice, *Bad* means a group of "bad credit" customers, *Good* means a group of "good credit" customers.

Traditional linear programming models of classification are usually based on distance, in these models, there are two kinds of objectives, one is to minimize the internal distance and the other is to maximize the external distance. But the solution with minimum internal distance or maximum external distance is not able to be proved to classify all samples correctly. In this paper, we insert an objective that is to minimize the number of misclassification to LP model of classification.

Then the credit scoring problem can be described as how to set the confident set X to achieve the following objectives:

(1) Minimize the number of misclassification

$$\text{Minimize } MCN = \sum_{i=1}^{N} [w_G d_G(i) + w_B d_B(i)] \quad (1)$$

where MCN is a coefficient that measure the quality of the classification for weight set $X = (x_1,...,x_R)^T$, $d_G(i)$, $d_B(i)$ are the 0-1 variable that $d_G(i) = 0$ if A_i is a Bad record or A_i is a Good record classified correctly and $d_G(i) = 1$ if A_i is a Good record classified as Bad; $d_B(i) = 0$ if A_i is a Good record or A_i is a Bad record classified correctly and $d_B(i) = 1$ if A_i is Bad record classified as Good.

In traditional data classification problem, accuracy is the main objective to be optimized. But to credit card customer classification, sensitivity is more important than other metrics. In this paper, we locate the trade-off between specificity and sensitivity through setting the weight $w_G, w_B \in (0,1)$. w_G is the weight of misclassification number in Good records. Then bigger

This work is supported by the Zhejiang Provincial Education Department project (Y201636906) and Ningbo innovative team project (2016C11024).

weight w_G will result in bigger specificity. Similarly, w_B is the weight of misclassification number in Bad records, and bigger weight w_B will result in bigger sensitivity.

Obviously, *MCN* is equal to half of the number of the records misclassified if $w_G = w_G = 0.5$, and *MCN*=0 if all records are classified correctly.

(2) Minimize the sum of deviations of the observations

$$\text{Minimize } MSD = \sum_{i=1}^{N} \alpha_i \quad (2)$$

where *MSD* is the sum of the deviations of the observations.

(3) Maximizing the minimal distances of observations from the critical value

$$\text{Maximize } MMD = \sum_{i=1}^{N} \beta_i \quad (3)$$

where *MMD* is the minimal distances of observations from the critical value.

A graphical representation of α_i, β_i is shown in Fig.1.

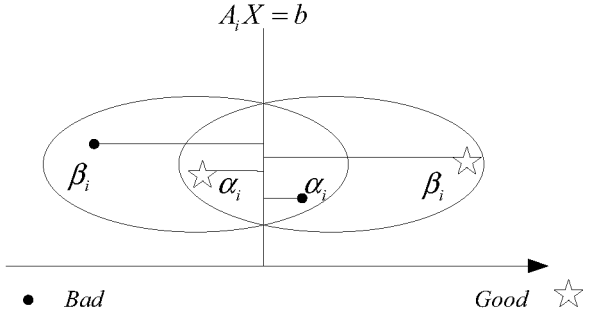

Figure 1. Graphical representation of α_i, β_i

Based on the above analysis, the constraints of this model are listed as follows:

$$\alpha_i = [d_G(i) + d_B(i)]\sqrt{(A_iX - b)^2} \quad (4)$$

$$\beta_i = c(i)\sqrt{(A_iX - b)^2} \quad (5)$$

$$d_G(i) = \begin{cases} 0, \ if: (A_i \in Bad) \text{ or } (A_iX \geq b, A_i \in Good) \\ 1, \quad if: A_iX < b, A_i \in Good \end{cases} \quad (6)$$

$$d_B(i) = \begin{cases} 0, \ if: (A_i \in Good) \text{ or } (A_iX < b, A_i \in Bad) \\ 1, \quad if: A_iX \geq b, A_i \in Bad \end{cases} \quad (7)$$

$$c(i) = \begin{cases} 0, \ if: (A_iX \geq b, A_i \in Bad) \\ \text{or } (A_iX < b, A_i \in Good) \\ 1, \ if: (A_iX < b, A_i \in Bad) \\ \text{or } (A_iX \geq b, A_i \in Good) \end{cases} \quad (8)$$

$$d_G(i) + d_B(i) + c(i) = 1 \quad , \forall i \in \{1, 2, ..., N\} \quad (9)$$

$$w_G + w_B = 1, \quad w_G, w_B \in (0,1) \quad (10)$$

where A_i, b are given, X is unrestricted and c_i is the 0-1 variable which $c(i) = 1$ if case A_i is classified correctly and $c(i) = 0$ if case A_i is misclassified.

The primary objective of classification problem is to classify all samples correctly, and then we define the solution which classifies all samples correctly as optimal solution.

Define 1 (Optimal solution): the solution X_j which classifies all records correctly.

$$P = \{X_j \mid MCN = 0, X_j \in \Omega\}. \quad (11)$$

where P is the set of feasible solutions, and Ω is the set of all solutions. According to (2) and (4), *MSD* of all feasible solutions is equal to 0.

Usually, it is not able to get an optimal solution if the data set is linear inseparable. Then the objective of classification problem is to find the feasible solution which *MCN* is the minimum.

Define 2 (Feasible solution): the solution with the minimum misclassification coefficient.

$$S = \{X_l \mid MCN = \min\{MCN | X_l \in \Omega\}, X_l \in \Omega\} \quad (12)$$

where S is the set of feasible solutions. Obviously, all optimal solutions are feasible solutions.

3. Multi-objective particle swarm optimization for credit scoring

According to three objectives for credit scoring problem, multi-objective particle swarm optimization (MOPSO) is used to deal with this problem. Multiple subswarms and co-evolutionary approach is able to improve the efficiency and effectiveness of MOPSO [16], in this paper a co-evolutionary MOPSO is adapted for Credit Scoring (MOPSO-CS). MOPSO-CS employs two subswarms ($Subswarm_1$ and $Subswarm_2$) with the same population J to probe the search space and information is exchanged between them.

3.1 Best position selection and updating process

Let the position of the particle in $Subswarm_1$ whose value of *MSD* is minimum be the personal best position of $Subswarm_1$ $Pbest_1$, and let the position of the particle in $Subswarm_2$ whose value of *MMD* is maximum be the personal best position of $Subswarm_2$ $Pbest_2$.

In order to search optimal solution as soon as possible, we select two feasible solutions from set S, and let the position of the feasible solution with minimum *MSD* be the global best position of $Subswarm_1$ $Gbest_1$, and let the position of the feasible solution with maximum *MMD* be the global best position of $Subswarm_2$ $Gbest_2$. Through this method, each swarm can share information from feasible solutions, so that MOPSO-CS is able to guide the particles to optimal solution as soon as possible.

MOPSO-CS is similar to co-evolutionary MOPSO and, in each generation t, particle j in the kth swarm updates

its current position $x_{kjr}(t)$ and velocity $v_{kjr}(t)$ through each dimension r by the personal best position $Pbest_{kjr}$ and the global best position $Gbest_{kr}$ using equation (14) and (15):

$$v_{kjr}(t+1) = wv_{kjr}(t) + c_1 r_1[Pbest_{kjr} - x_{kjr}(t)] + c_2 r_2[Gbest_{kr} - x_{kjr}(t)] \quad (14)$$

$$x_{kjr}(t+1) = x_{kjr}(t) + v_{kjr}(t+1) \quad (15)$$

where w is inertia weight, c_1, c_2 are the acceleration constants and r_1, r_2 are random real numbers drawn from $U(0,1)$.

In this paper, the following weight function is used:

$$w = w_{\max} - t\left(\frac{w_{\max} - w_{\min}}{T}\right) \quad (16)$$

where

$w_{\max}$: Initial weight, $w_{\min}$: Final weight.

T: Maximum generation number, t: Current generation number.

Because there are perhaps many particles with the same particle rank, *MSD* and *MMD* are also taken into consideration when all particles are sorted. The detailed criterion of rank for each particle is as follows:

Particle a is superior to particle b:

$\Leftrightarrow \{rank(a) < rank(b)\}$

Or $\{rank(a) = rank(b)\ and\ MSD(a) < MSD(b)\}$

3.2 The optimization algorithm

For MOPSO-CS, the scope of the solution is equal to R-dimensional search space where R is the total number of attributes of each sample (customer). The position of particle j $x_{kj} = (x_{kj1}, ..., x_{kjR})$ corresponds to a solution for the problem, and the rth dimension of the position $x_{kjr}(k = 1,2; j = 1, ..., J; r = 1, ..., R)$ denotes the rth coefficient used by each sample. The coding design of particle velocity is similar to the design of particle position, which is composed of a component controlled set of coefficient: $v_{kj} = (v_{kj1}, ..., v_{kjr})$.

Firstly the algorithm randomly generates two subswarms. Then the particle position and the particle velocity of all particles in these subswarms are initialized. Each element x_{kjr} of x_{kj} is randomly initialized within [-100, 100]. Initialize each element v_{kjr} of particle velocity within [-10,10].

Secondly evaluate these subswarms. According to x_{kj} which has been initialized, calculate *MCN*, *MSD* and *MMD* for each particle using objective (1), (2) and (3). Then, evaluate the particle rank of each particle using Eqt. (13). The algorithm initializes the global best position of each subswarm and the personal best position of each particle.

In this paper, MOPSO-CS tries to find optimal solutions of this problem and terminates when the first optimal solution is found or the maximum iteration number T is reach. Before the algorithm finds optimal solution, all feasible solutions are used as the best found solutions. At each iteration of algorithm, in order to optimize the Objective (1), MOPSO-CS updates the best found solutions in term of MCN and reserves the solutions with minimum MCN found by optimization process.

Thirdly update the position and velocity of each particle according to Eqt. (14) and (15). Then, we select top $2J$ particles with minimum particle rank from old population and updated population, and use them as the new population for the next generation. The optimization algorithm can be formally described as follows:

Pseudocode of MOPSO-CS:

Step1. Start procedure MOPSO-CS. Initialize the position of each particle of $subswarm_1$ and $subswarm_2$. Define $w_G, w_B, r_1, r_2, c_1, c_2, w_{\min}, w_{\max}$ and *Cutoff*.

Step2. Evaluate Swarm

For each $Subswarm_k$ do

For k=1 to 2

For each particle j do

For j=1 to J

Compute $MSD(j)$ and $MMD(j)$

Calculate $rank(j)$

Update $Pbest_{kj}$

Next j

End do

Update $Gbest_k$

Nest k

End do

Step 3: Update the best found solution

Step 4: Check the stopping condition. If it is not met go to Step 5. Otherwise go to Step 7.

Step 5: Update the new velocities and positions for each particle j

Step 6: Select new population for the next generation. Go to Step 2

Step 7: Finish the procedure MOPSO-CS

4. Computational experiments

4.1. The selection and preprocessing of credit data set

In computational experiments, we use various data sets that represent the credit behaviors of people from different countries which include one real-world data set from UK and one benchmark German data set. All these data sets classify people as "Normal" or "Bankrupt" credit risks based on a set of attributes/variables. In order to investigate the effectiveness of the proposed model and algorithm in unbalance data set and balance data set, the real-world credit data from UK is separated to two date set. The first is UK-1 data set [2], which collects 323 bankrupt and 902 normal customers with 14 variables. The second is balance data set UK-2, which consist of 323 normal customers and 323 bankrupt customers. The third data set comprises German Credit card records from UCI Machine

Learning databases [10], which contains 1000 records (700 normal and 300 bankrupt).

In order to improve the accuracy of credit data classification, we have preprocessed data in credit records before classification. The data preprocessing includes the conversion of unstructured data into structured data and deletion of the attributions whose correlation with other attributions are higher (the correlation coefficient $r_{ij} \le -0.5$ or $r_{ij} \ge 0.5$). Table 1 shows the selection and preprocessing of credit data set.

Table1. The selection and preprocessing of credit data set

Data set	Record No.	Attribution No. of raw data	The attributions that have been deleted
UK-1	1225	14	AES,RES,DMORT
UK-2	646	14	AES,RES,DMORT
German	1000	24	A4,A19,A21,A24

4.2. Experimental evaluation

In this section, our proposed methods, MOPSO-CS, is compared with five classic classification methods: NaiveBayes, Logistics Regression (LR), SMO, Neural Networks (NN), and Decision Tree (DT). The implementation software of these counterparts is WEKA 3.6, and all experiments were performed on a PC Core i5 with 2.5GHz and 4GB RAM running under the Windows 10 operating system. Table 2 shows the implementation software that was explored to run the three data sets.

Table2. Employed Software

Classifier	Software
MOPSO-CS	Visual Studio 6.0
NaiveBayes	WEKA3.6 Bayes (NaiveBayes)
LR	WEKA3.6 Functions (Logistics)
SMO	WEKA3.6 Functions (SMO)
NN	WEKA3.6 Functions (MultilayerPerceptron)
DT	WEKA3.6 Trees(J48)

The parameters used in MOPSO-CS are presented in Table 3. The detailed results of the experiments show as follows.

Table 3. The parameters used in MOPSO-CS

Data set	w_G	w_B	T	*Population*	*Cutoff*
UK-1	1/2	1/2	1000	40	-1.1
UK-2	1/2	1/2	1000	40	80
German	1/2	1/2	1000	40	-1.1

There are several measures of classification performance commonly used in the credit industry. [17]

True Positives (TP): the number of actual positive records which predicts positive.

False Negatives (FN): the number of actual positive records which predicts negative.

True Negatives (TN): the number of actual negative records which predicts negative.

False Positives (FP): the number of actual negative records which predicts positive.

Accuracy = (TP+TN)/(TP+FP+TN+FN)

Sensitivity = TP/(TP+FN)

Specificity = TN/(FP+TN)

Table 4 shows the result of ten-fold cross-validation of UK-1 Data Set (902 normal customers, 323 bankrupt customers). We can see from table 4, the sensitivity of our MOPSO-CS excel those of LR, SMO and NN, and only below the sensitivity of NaiveBayes and DT, but the accuracy of MOPSO-CS is superior to those of all the other counterparts.

Table 4. Ten-Fold Cross-Validation Result of UK-1

Classifier	Accuracy	Specificity	Sensitivity
MOPSO-CS	**74.64%**	**96.81%**	12.72%
NaiveBayes	65.22%	73.50%	**42.10%**
LR	73.80%	96.60%	10.20%
SMO	73.63%	100.00%	0.00%
NN	72.57%	94.50%	11.50%
DT	74.29%	94.80%	17.00%

Table 5 shows the result of ten-fold cross-validation of balance UK-2 Data Set (323 normal customers, 323 bankrupt customers).

Table 5. Ten-Fold Cross-Validation Result of UK-2

Classifier	Accuracy	Specificity	Sensitivity
MOPSO-CS	59.85%	**81.67%**	38.02%
NaiveBayes	58.20%	42.10%	**74.30%**
LR	60.99%	63.20%	58.80%
SMO	58.51%	60.70%	56.30%
NN	**61.15%**	59.10%	63.20%
DT	61.14%	57.30%	65.00%

From table 5, we can see that the accuracy of our MOPSO-CS excel those of NaiveBayes and SMO, but the Specificity of MOPSO-CS is superior to those of all the other counterparts.

Table 6 shows the result of ten-fold cross-validation of balance German Data Set (700 normal customers, 300 bankrupt customers). The specificity of MOPSO-CS excel those of all other counterparts, and the accuracy of MOPSO-CS excel those of NN, Rules and DT.

Table 6. Ten-Fold Cross-Validation Result of German

Classifier	Accuracy	Specificity	Sensitivity
MOPSO-CS	74.69%	**89.87%**	39.27%
NaiveBayes	75.10%	84.90%	**52.30%**
LR	**76.70%**	89.10%	47.70%
SMO	76.40%	89.70%	45.30%
NN	70.30%	80.00%	47.70%
DT	72.70%	83.90%	46.70%

5. Conclusion

Traditional credit scoring model are usually based on distance, in these models there are two kinds of objectives, one is to minimize the internal distance and the other is to maximize the external distance. But the solution with minimum internal distance or maximum external distance is not able to be proved to classify the samples correctly. In

this paper, a novel multi-objective credit scoring model is proposed, minimum the number of misclassification, then minimize the internal distance and maximize the external distance are all included in the objectives of our model.

To credit scoring problem, it is important for credit scoring model to provide the credit score which is easy to be comprehended by credit applicants. Compared with black box technology such as NN and SVM, the credit score function in our approach is comprehensible. Finally, our example and experimental studies based on benchmark data set and real world data sets, confirm that our proposed method outperforms the abovementioned five data-driven counterparts in term of accuracy while maintaining acceptable sensitivity.

References

[1] T.S Lee, I.F Chen, A two-stage hybrid credit scoring model using artificial neural networks and multivariate adaptive regression splines. Expert Systems with Applications, Vol 28, 743–752, 2005.

[2] J He, Y.C Zhang , Y Shi, G.Y Huang, Domain-driven classification based on multiple criteria and multiple constraint-level programming for intelligent credit scoring. IEEE transactions on knowledge and data engineering, Vol. 22, No. 6, 826-838, 2010.

[3] B.W Chi, C.C Hsu, A hybrid approach to integrate genetic algorithm into dual scoring model in enhancing the performance of credit scoring model. Expert Systems with Applications, Vol. 39, No. 3, 2650–2661, 2012.

[4] H Jo, I Han, H Lee, Bankruptcy prediction using case-based reasoning, neural networks, and discriminant analysis. Expert Systems with Applications, Vol. 13, No. 2, 97-108, 1997.

[5] D. J Hand, W. E Henley, Statistical Classification Methods in Consumer Credit Scoring: A Review. Journal of the Royal Statistical Society. Series A, Vol. 160, No. 3, 523-541, 1997.

[6] A.C Bahnsen, D Aouada, Example-Dependent Cost-Sensitive Logistic Regression for Credit Scoring. Machine Learning and Applications (ICMLA), 2014 13th International Conference, 263 – 269, 2014.

[7] P Hájek, Municipal credit rating modelling by neural networks. Decision Support Systems, Vol. 51, No. 1, 108–118, 2011.

[8] D West, Neural network credit scoring models. Computers and Operations Research, Vol. 27, No. 11, 1131–1152, 2000.

[9] T Bellotti, J Crook, Support vector machines for credit scoring and discovery of significant features. Expert Systems with Applications, Vol. 36, No. 2, 3302–3308, 2009.

[10] D Niklis, M Doumpos, C Zopounidis, Combining market and accounting-based models for credit scoring using a classification scheme based on support vector machines. Applied Mathematics and Computation, Vol. 234, No. 15, 69–81, 2014.

[11] D.L Olson, D Delen, Y Meng, Comparative analysis of data mining methods for bankruptcy prediction. Decision Support Systems, Vol. 52, No. 2, 464–473, 2012.

[12] G Wang, J Ma, L Huang, K Xu, Two credit scoring models based on dual strategy ensemble trees. Knowledge-Based Systems, Vol. 26, 61–68, 2012.

[13] R Aliehyaei, S Khan. Ant Colony Optimization, Genetic Programming and a hybrid approach for credit scoring: A comparative study. Software, Knowledge, Information Management and Applications (SKIMA), 2014 8th International Conference,1-5, 2014.

[14] J.J Huang, G.H Tzeng, C.S Ong, Two-stage genetic programming (2SGP) for the credit scoring model. Applied Mathematics and Computation, Vol. 174, No. 2, 1039-1053, 2006.

[15] S Lessmann, B Baesens, H.V Seow, Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. European Journal of Operational Research, Vol. 247, No. 1, 124–136, 2015.

[16] Goh, C.K, Tan, K.C., Liu, D.S., Chiam, S.C. A competitive and cooperative co-evolutionary approach to multi-objective particle swarm optimization algorithm design. European Journal of Operational Research, Vol. 202, 42-54, 2010.

[17] S Bhattacharyya, S Jha, K Tharakunnel. Data mining for credit card fraud: A comparative study. Decision Support Systems, Vol. 50, No. 3, 602–613, 2011.