MATHEMATICAL MODELING AND COMPUTING, Vol. 11, No. 1, pp. 109–119 (2024)

A hybrid variable neighborhood search with bootstrap resampling technique for credit scoring problem

Barhdadi M., Benyacoub B., Ouzineb M.

National Institute for Statistics and Applied Economics, Rabat, Morocco

(Received 27 June 2023; Revised 2 February 2024; Accepted 13 February 2024)

Credit scoring models have played a vitally important role in the granting credit by lenders and financial institutions. Recently, these have gained more attention related to the risk management practice. Many modeling techniques have been developed to evaluate the worthiness of borrowers. This paper presents a credit scoring model via one of local search methods – variable neighborhood search (VNS) algorithm. The optimizing VNS neighborhood structure is a useful method applied to solve credit scoring problems. By simultaneously tuning the neighborhood structure, the proposed algorithm generates optimized weights which are used to build a linear discriminant function. The experimental results obtained by applying this model on simulated and real datasets prove its high efficiency and evaluate its significant value on credit scoring.

Keywords: *credit scoring; mathematical modeling; variable neighborhood search algorithm; linear classification; resampling technique.*

2010 MSC: 62-07, 90C05, 90C06, 00C59 **DOI:** 10.23939/mmc2024.01.109

1. Introduction

Nowadays, people in our society have become perfect consumers; the fact which compels individuals to seek alternative financing solutions to cover the various expenses. One of which is getting loans. A credit can be the ideal solution, otherwise, it will become a risky operation if its applicant fails to meet the payment deadlines [1]. Credit scoring is a system analysis developed by statistical methods, operations research, artificial intelligence and machine learning techniques which enables one to make a distinction between "good" and "bad" loans, based on information about the borrower [2]. Due to the increase in demand of consumption credit, along with the rapid development of the storage infrastructure of the information, the importance of credit scoring has become more and more significant nowadays and has gained more and more attention. Credit scoring was developed in order to explore the relationship between the dependent variable describing the risk of a consumer defaulting on a loan, and independent variables characterizing the information of consumers (e.g. age, number of previous loans, Salary, Housing etc.) [3].

Many credit scoring models based statistical methods which include discriminant analysis, logistic regression [4], nonlinear regression, classification trees, nearest-neighbor approach [5]; and non-statistics based methods which include linear programming, integer programming, neural network [6], genetic algorithm, expert system and so forth; have been proposed to trade with some consumers whose credit evaluations are good and gain reasonable benefits [7,8]. The successful use of the credit scoring models in commercial banks, stockjobbers and other financial organizations depends on the sophisticated algorithms which can make appropriate decisions to do business with credit appliers or does not. Having estimated their credits overall, the organizations measure the possibilities for customers not to pay for their goods or services on time or be in arrears in purpose.

Variable neighborhood search (VNS) is a metaheuristic proposed by Mladenovic and Hansen in 1997 [9,10]. VNS systematically changes neighborhood structures during the search for an optimal (or near-optimal) solution. VNS is easy to be trapped in local optimum or lack of effective local search mechanism. The used training algorithm is able to rapidly locate good solutions, even for a difficult search space. It is a method for optimizing numerical functions and real-world problems. Actually, it

has been successfully applied to solve many hard combinatorial optimization problems such as vehicle scheduling, vehicle routing problem and timetabling problems, etc. Hence, this paper develops a VNS model in terms of practical personal credit scoring problem.

Early developed by Efron (1979, 1982) bootstrap procedure is the most widely used among resampling methods. From this time, many works has been introduced to ameliorate statistical shortcomings of the bootstrap [11,12]. In the last few years, Bootstrap variants were introduced to achieve statistical consistent estimators [13,14]. Recently, resampling methods using Bootstrap have been used to obtain more accurate estimates for massive data [15–18].

The paper is organized as follows: section 2 presents the credit scoring problem, section 3 describes the VNS method and presents the proposed model for the credit scoring, section 4 reports computational experiments that demonstrate the effectiveness of our methodology on a set of benchmark data sets, section 5 comes up with final conclusions.

2. Background

2.1. The credit scoring problem

Credit agencies must assess the level of risk associated with granting a loan to a new customer. To do this, they rely on the old credit records in their possession using classification models that can distinguish between good and bad customers. Credit scoring problem is a two-group classification problem assuming that there are two well-defined populations, G_1 and G_2 (good loans vs bad loans respectively). We assume that the information is known in advance about the input data. In this learning category, the objective is usually to provide the best discriminant function using the input data by measuring p discriminatory variables or attributes for each member of either population.

Each member represents a customer; each client is characterized by a set of attributes (information): $X = (X_1, X_2, \ldots, X_p)$ (age, account, job, income, residence and so on). This information is recorded directly from different customers.

Given a sample E that includes n customers. This sample will consist of g good customers and b bad ones; i.e. $(n = g+b)$ and $(E = G_1 \cup G_2)$. Each customer i gives answers: $X_i = (X_{i1}, X_{i2}, \ldots, X_{ip})$. The classification model used by the credit institution must differentiate between data associated with good customer and data related to bad one. To do this, each attribute is assigned a weight $w = (w_1, w_2, \ldots, w_p)$, and a threshold c is sought/found to separate the good customers from the bad ones. If $\sum_{j=1}^{p} w_j x_{ij} \geqslant c$ then customer i is a good one, and he is a bad one in case he satisfies $\sum_{j=1}^{p} w_j x_{ij} \leqslant c$.

2.2. Problem formulation

The mathematical programming approach was first applied in classification by Magasarian (1965) [19]. Freed and Glover (1981) [20] presented an evaluation of the LP approach in discriminant analysis. A classification approach has been proposed based on the idea of reducing the misclassification through by minimizing the overlaps between two groups [21]. In a linear system, the separating hyperplane maximizes the distance between two objectives.

It is not evident to perfectly separate the good customers from the bad ones. We can tolerate eventual errors by introducing positive values a_i, then we will have $\sum_{j=1}^{p} w_j x_{ij} \geqslant c - a_i$ (if customer i is a good one) $\sum_{j=1}^{p} w_j x_{ij} \leqslant c + a_i$ (if customer i is a bad one). The objective will be then to find the values of the weight vector w and the value of the separator c for which $\sum_{i=0}^{n} a_i$ is minimal. The final model is then:

$$\text{Min} \sum_{i=1}^{n} a_i \tag{1}$$

subject to:

$$\sum_{j=1}^{p} w_j x_{ij} \geqslant c - a_i \quad \forall i \in G_1, \tag{2}$$

$$\sum_{j=1}^{p} w_j x_{ij} \leqslant c + a_i \quad \forall i \in G_2, \tag{3}$$

$$\sum_{j=1}^{p} \left(n_B \sum_{\substack{i=1 \\ i \in G_1}}^{n_G} x_{ij} - n_G \sum_{\substack{i=1 \\ i \in G_2}}^{n_B} x_{ij} \right) w_j = 1, \tag{4}$$

$$a_i \geqslant 0 \quad \forall i, \quad c \in R^* \quad \text{and} \quad w_j \in R \quad \forall j. \tag{5}$$

The objective function aims at minimizing the sum of the distances between the clients from the population and the hyperplane corresponding to discriminant function. Constraint (2) ensures that the client will belong to group 1 (good loans), while constraint (3) ensures that the client will be assigned to group 2 (bad loans). Equation (4) defines the normalization constraint and it is needed to avoid a trivial solution.

For a problem of classification with two distinguished groups given by a set of observations ($i = 1, 2, \ldots, n$) from group G_1 to group G_2. The objective is to determine aseparating hyperplane presented by a linear discriminant function expressed as $Z = w_1X_1 + w_2X_2 + \ldots + w_pX_p$. This function provides the boundary between two groups and will be used to classify the new observations. The goal of solving the model is to estimate the parameters including a weighting vector $W = (w_1, w_2, \ldots, w_p)$ and a decision rule cutoff value c to minimize the number of misclassifications for given data sets. In the following we focus on specific combinatorial optimization methodology using VNS to adjust the parameters that improve the classification performance.

3. VNS approach with bootstrap resampling technique

VNS is a metaheuristic method used for optimization problems. It looks for a local or global minimum starting from an initial solution W_0. It creates a neighborhood of the solution and then checks if there is solution W^* better than the initial solution W_0. It crushes W_0 by W^* if there is solution W^* better than the initial solution W_0. It restarts until stopping condition is not met (it is defined as the maximum number of global iterations without finding an improvement in the best known solution). The algorithm needs initial solution W_0. This solution is identified as the null vector $W_0 = (0, 0, \ldots, 0)$. The proposed VNS uses three neighborhood structures N_1, N_2 and N_3.

3.1. Neighborhood structures

The neighborhood structure N_1 is based on a swap operation which strips a randomly chosen regenerator r from location and add it elsewhere. Figure 1 presents an example of neighborhood structure N_1.

Let W be a given vector $W = (w_1, w_2, \ldots, w_p)$ and i, j are two integers such as $1 \leqslant i, j \leqslant p$. w_i becomes $w_i + r$ and w_j becomes $w_i - r$.

So, $N_1 = \{(w_1, \ldots, w_i + r, \ldots, w_j - r, \ldots, w_p);\ \forall\, i \neq j\}$.

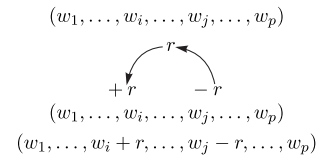

Fig. 1. Neighborhood structure N_1.

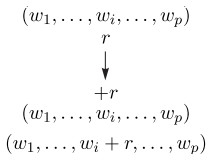

Fig. 2. Neighborhood structure N_2.

Example 1. Let W be a given vector $W = (1, 2, 3, 4)$ and $r = 1$

$$N_1(W) = \{(0,3,3,4), (0,2,4,4), (0,2,3,5), (2,1,3,4), (1,1,4,4), (1,1,3,5),$$
$$(2,2,2,4), (1,3,2,4), (1,2,2,5), (2,2,3,3), (1,3,3,3), (1,2,4,3)\}.$$

While the neighborhood structure N_2 is obtained by adding a randomly chosen regenerator r to the solution. Figure 2 presents an example of neighborhood structure N_2.

Let W be given vector $W = (w_1, \ldots, w_p)$ and i is given integer $1 \leqslant i \leqslant p$. w_i becomes $w_i + r$.

So, $N_2 = \{(w_1, \ldots, w_i + r, \ldots, w_p); \forall i\}$.

Example 2. Let W be a given vector $W = (1, 2, 3, 4)$ and $r = 1$

$$N_2(W) = \{(2, 2, 3, 4), (1, 3, 3, 4), (1, 2, 4, 4), (1, 2, 3, 5)\}.$$

The N_3 neighborhood is practically the same as the N_1, only this time the regenerator chosen is randomly stripped from a location then equally added on two other locations.

Example 3. Let W be a given vector $W = (1, 2, 3, 4)$ and $r = 2$

$$N_3(W) = \{(-1, 3, 4, 4), (-1, 3, 3, 5), (-1, 2, 4, 5), (2, 0, 4, 4), (2, 0, 3, 5),$$
$$(1, 0, 4, 5), (2, 3, 1, 4), (2, 2, 1, 5), (1, 3, 1, 5, 5), (2, 3, 3, 2), (2, 2, 4, 2), (1, 3, 4, 2)\}.$$

Figure 3 presents an example of neighborhood structures N_3.

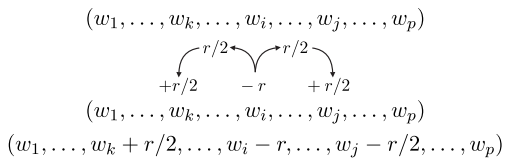

Fig. 3. Neighborhood structure N_3.

Let W be a given vector such as $W = (w_1, w_2, \ldots, w_p)$ where i, j, k are three different integers such as $1 \leqslant i, j, k \leqslant p$ $i \neq j \neq k$. w_i becomes $w_i - r$, w_j becomes $w_j + r/2$ and w_k becomes $w_k + r/2$.

So, $N_3 = \{(w_1, \ldots, w_k + r/2, \ldots, w_i - r, \ldots, w_j + r/2, \ldots, w_p)\ \forall\ i \neq j \neq k\}$.

3.2. LocalSearch function

The LocalSearch() function is able to improve the current solution. To do so, it uses one of the already defined neighborhood structures N_j $(j = 1, 2, 3)$ in order to create the set of neighbors of the current solution $\widehat{W}$. After that, it chooses a better solution from the one already found. This kind of solution is acquired by using the best improvement method, which compares the elements of $\widehat{W}$ to each other in order to find the best element W' of this set and returns its value.

3.3. VND function

In order to optimize the current solution, the $VND()$ function takes vector W as input to improve it. The next step is to use the neighborhood structures N_j $(j = 1, \ldots, n)$ which, in our case, will be N_1 and N_2, N_3. As long as the function is able to improve W j takes the value 1 in order to use the neighborhood structure N_1 (the iterations will continue as long as j is below the number of structures defined in the beginning). The function LocalSearch() is then used to create $\widehat{W}$ and return W'. If W' is better than W, then W is updated to W' and j gets the the value 1 to restart as a new iteration with N_1 as neighborhood's structure or else j is updated to $j + 1$ to use the next neighborhood's structure. If there is no possible improvement the algorithm stops. The pseudocode of the VND(W) function is provided in Algorithm 1.

Algorithm 1 VND function.

Require:
Ensure:
1: procedure VND(W);
2: **while** there is a possible improvementt
3: $j \leftarrow 1$
4: **while** $j \leqslant 3$
5: $W' \leftarrow$LocalSearch($N_j(W)$)
6: **if** W' is better than W **then**
7: $W \leftarrow W'$
8: $j \leftarrow 1$
9: **else**
10: $j \leftarrow j + 1$
11: return W_{best};
12: end procedure;

3.4. BootsVND

Bootstrap procedure. Bradley Elfon introduced the Bootstrap method for the first time, in this method we generate B samples randomly from the initial dataset $(S^1, S^2, \ldots, S^B)$, each sample has the same size as the initial set. Each time we choose only one individual, then we choose the second one with replacement until the size of the sample is reached. For each sample S^i we compute the parameters vector of the discriminant function $w^i = (w_1^i, w_2^i, \ldots, w_p^i)$, the use of B samples corresponds to the generation B vectors of parameter $(w^1, w^2, \ldots, w^B)$.

BootsVND function. In this study, we propose a hybridization of Bootstrap and the VNS (BootsVNS), to be more specific, we use a hybridization of Bootstrap and the VND() function. The bootstrap procedure applied to generate B samples then the VND() function is used to obtain coefficient estimates for the parameter W. The pseudocode of the proposed hybridization BootsVND() is shown in Algorithm 2.

Algorithm 2 BootsVND.

Require:
Ensure:
1: procedure BootsVND(W)
2: $i \leftarrow 1$
3: **while** $i <= B$
4: $S^i \leftarrow$ randomly generate a sample from the initial dataset
5: $w^i \leftarrow VND(W, S^i)$
6: $i \leftarrow i + 1$
7: $W^* \leftarrow \sum_{i=1}^{B} \frac{w^i}{B}$
8: return W^*
9: end procedure;

3.5. Shaking function

The Shaking() function is able to unsettle the current solution. To do so, it follows the same procedure to create the set of solution $\widehat{W}$. Only this time, it chooses randomly a solution from this set and repeats the process m times to make sure that it moves away from the current solution. The use of this function guarantees a certain escape from blocking domains and even avoids minimum locals. The pseudocode of the Shaking() function is provided in Algorithm 3.

Algorithm 3 Shaking(k, W_{best}).

Require:
Ensure:
1: procedure Shaking(k, W_{best})
2: $i \leftarrow 1$
3: $W \leftarrow W_{best}$
4: **while** $i \leqslant m$
5: $\widehat{W} \leftarrow N_k(W)$
6: $W \leftarrow$ randomly select one vector from $\widehat{W}$
7: $i \leftarrow i + 1$
8: return W
9: end procedure;

3.6. VNS/Bootstrap procedure

To find an optimal solution, the algorithm starts by initializing W as the null vector $(W_0 = 0, 0, \ldots, 0)$, then assigns its value to W_{best}. Next, it defines the neighborhood's structures N_j $(j = 1, \ldots, n)$ which, in our case, are N_1, N_2, N_3. While the algorithm is able to find an improvement to W_{best}, k gets the values from 1 up to K_{max} so it can shake the initial solution using the Shaking() function. Afterwards,

the BootsVND() function is then used to find the local minimum W'. After that, we check if W' is better than W_{best}. If the condition is met, W_{best} is updated to w and k gets 1 as value, or else k gets $k+1$ and restart the Shacking() function so we can move away from the local minimum solutions. Finally, if we use the Shaking() function K_{max} time without any improvements, we will assume that there will not be any possible improvements thus the algorithm stops. The pseudocode of the proposed approach is provided in Algorithm 4.

Algorithm 4 VNS/Bootstrap algorithm.

Require:
Ensure:
procedure VNS(S)
Initialize W
define a set of neighborhood structures N_j, $(j = 1, 2, 3)$ of the current solution
while stopping condition is not met
  $k \leftarrow 1$
  while $k \leqslant k_{max}$
    $W \leftarrow$ Shaking(k)
    $j \leftarrow 1$
    $W' \leftarrow$ BootsVND($N_j(W)$)
    if W' is better than W_{best} **then**
      $W_{best} \leftarrow W'$
      $k \leftarrow 1$
    else
      $k \leftarrow k + 1$
return W_{best};
end procedure;

We extend the VNS approach for the resolution of a specific case classification problems: credit scoring [22]. The vector weight W and the value c representing the classifier model will be coded and optimized in accordance with the VNS procedure using the training set until reaching the best considered neighborhood. Once W is estimated, the discriminant function is determined. Then, the model must be tested and its performance checked. Recall in our situation, that an optimal solution is determined by a discriminant function that better distinguish between good and bad customers and provide as low rate of misclassification as possible. A local search method is responsible of locally improving the measures performance of classifier.

4. Computational experiment and results analysis

VNS algorithm with combination Bootstrap resampling estimation method (BootsVNS) is used to identify parameters estimates for a discriminant mathematical programming model using a training sample data. The proposed model (BootsVNS) was coded in C++ and the executions were carried out on Intel Core i7-7500U processor on 2.7 GHz with 8 GB RAM memory under Windows 10 operating system. The resultant models are then further used to test the classification power of these models based on their performance in the testing sample data.

4.1. Datasets description

In this study, four popular credit scoring datasets were used to evaluate the performance of our proposed approach: Australian, German, Taiwan, and Tomas datasets. The Australian dataset comprises 690 customers, with 307 (44.5%) are good clients and the remaining 387 (55.5%) are bad ones, featuring 15 distinct characteristics. The German dataset contains total of 1000 observations, consisting of 700 (70%) good loans and 300 (30%) bad loans, with 21 variables. The Taiwan dataset includes data from 30 000 borrowers, with 23 364 (78%) good loans and 6 636 (22%) bad loans, it comprises 23 distinct

characteristics. While the Tomas dataset comprises 1 225 customers, with 967 (79%) good clients and 258 (21%) bad ones, with 15 features. These datasets can be accessed on the UCI Machine Learning Repository website: https://archive.ics.uci.edu/ml/datasets. Further details about these credit scoring datasets are provided in Table 1.

Table 1. Statistics of selected credit scoring and simulated datasets.

DATASET	Sample size	No. features	Training set size	Testing set size	Goods(%)/ Bads(%)
Australian	690	15	552	138	44.5/55.5
German	1000	21	800	800	70/30
Tomas	1225	15	980	245	79/21
Taiwan	30000	23	24000	6000	78/22

4.2. Performance evaluation metrics

Different evaluation metrics have been used in the literature to evaluate the performance of classification techniques in credit scoring. They include accuracy, error type I, error type II, cost function and AUC. In this study we used three of them: average accuracy, type I error and type II, which are the most used in credit scoring [23].

A: The Average Accuracy. The criterion for measuring the performance rate of a model is accuracy and is used to compare models. The following equation allows to calculate it,

$$Accuracy = \frac{The\ number\ correctly\ classified\ cases}{The\ total\ number\ of\ cases}.$$

B: Type I Error and Type II Error. The following equation is used to calculate the rate of type I error:

$$Type\ I\ error = \frac{FP}{FP + TN}.$$

Where FP (False Positive) is the number of customers mistakenly classified as "good", and who are actually "bad" and TN (True Negative) is the number of customers classified as "bad" and who are indeed "bad" (well-classified), and $1 - Type\ I\ error$ is defined as Specificity.

The following equation is used to calculate the rate of type II error:

$$Type\ II\ error = \frac{FN}{FN + TP},$$

where FN (False Negative) is the number customers mistakenly classified as "bad" and who are actually "good" customers and TP (True Positive) is the number of customers classified as "good", and who are indeed "good" (well-classified), and $1 - Type\ II\ error$ is defined as Sensitivity.

C: Cost function. The following equation is used to calculate the cost function:

$$Cost\ function = P_b\ C_{typeI}\ Type\ I\ error\ rate + P_g\ C_{typeII}\ Type\ II\ error\ rate,$$

where P_b and P_g are the population proportion of "good" and "bad" customers respectively, and C_{typeI} and C_{typeII} are the cost of $Type\ I\ error$ and $Type\ I\ error$. In credit scoring applications, $Type\ I\ error$ is considered the higher cost with and we set $C_{typeI} = 5$ and C_{typeII} is set to a small value ($C_{typeII} = 1$).

4.3. Results and analysis

In the present section, we present experimental results of our proposed evolutionary model comparing it to other existing classifiers. Before that, we split the datasets to training and test sets using the most common split ratio is 80 : 20 generated by a random selection from the whole datasets. The training set was used for models construction and the resulting models were then validated using testing sample set.

A: Resampling parameters obtained by BootsVNS. In order to harness the benefits of the Bootstrap procedure, it is crucial to determine the best value of the parameter B. The results in Figures 4 and 5 show the accuracy of the algorithm as a function of B. Specifically, Figure 4 reveals

an increasing trend in accuracy with the increment of B for the Australian dataset, reaching a peak at $B = 180$. Similarly, Figure 5 demonstrates a similar behavior for the Tomas dataset, wherein accuracy improves until reaching its maximum at $B = 240$. These observations indicate that increasing the value of B increase the model accuracy and improve its performance.

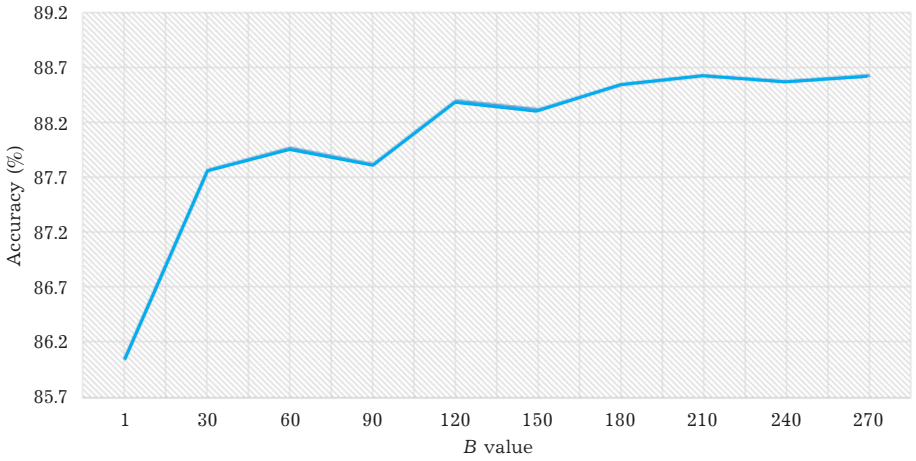

Fig. 4. Accuracy of $BootsVNS$ for various B using Australien dataset.

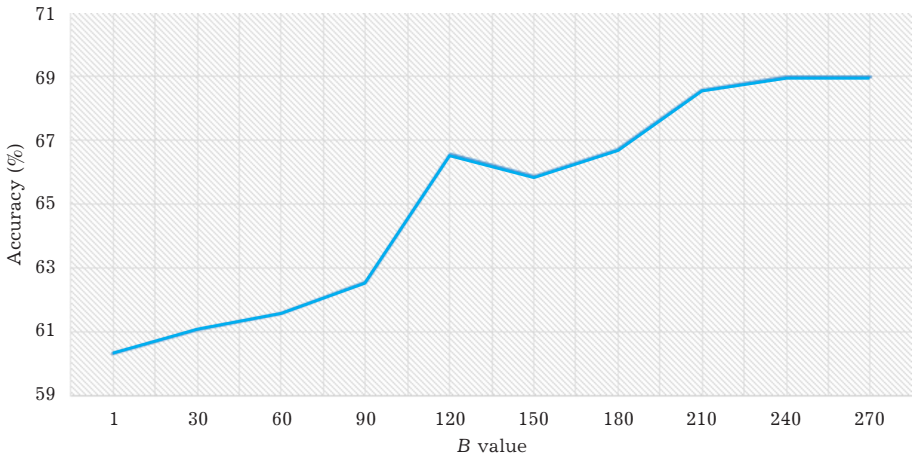

Fig. 5. Accuracy of $BootskVNS$ for various B using tomas dataset.

B: Results obtained by BootsVNS. Table 2 presents the results of classification performance (Accuracy, two type error rates and cost function) and shows the comparison of the best performances using VNS and BoostVNS for four real datasets used. From Table 2, the obtained results proves the hypothesis that, our proposed BoostVNS can improve the classification performance of credit scoring model. We have used a special notational convention whereby the best testing set metrics performance per data set is denoted in bold.

Table 2. Results of VNS and BoostVNS method for credit scoring datasets.

Datasets	VNS				BoostVNS			
	Accuracy	Error TypeI	Error TypeII	Cost function	Accuracy	Error TypeI	Error TypeII	Cost function
Australian	86.05	9.76	20.43	30.23	**88.63**	7.01	17.93	27.02
German	72.33	49.68	18.96	85.53	**74.50**	47.87	15.5	83.25
Thomas	60.32	9.20	69.32	58.8	**68.98**	5.76	64.76	53.04
Taiwan	74.11	13	74	72.51	**75.54**	11.32	72.44	70.67

C: Comparison of BootsVNS method with other existing classifiers. To examine the performance of the used method, the VNS model is compared to the well-known method, such as linear discriminant analysis (LDA), logistic regression (LR), K-nearest neighbor(knn), Decision tree (DT), Neural-Network Analysis (ANN), support vector machines (SVM) for both linear (Lin SVM) and RBF kernels (RBF SVM) and least square support vector machines (LS-SVM) for both linear (Lin LS-SVM) and RBF kernels (RBF LS-SVM) for credit scoring. All the experiments are performed with training data and test data. All computations were carried out in the Matlab software environment.

Table 3. Results of VNS and other classifiers for Australian and German datasets.

Dataset	Australian				German			
	Accuracy	Error TypeI	Error TypeII	Cost function	Accuracy	Error TypeI	Error TypeII	Cost function
BoostVNS	**88.63**	7.01	17.93	**27.02**	74.50	47.87	15.5	83.25
LDA	86.47	19.8	**5.49**	46.59	74.00	24.73	26.57	56.66
LR	85.99	10.34	18.68	33.19	**79.33**	**23.65**	19.32	**49.98**
knn	82.60	**6.03**	31.86	31.10	71.66	83.87	3.38	132.33
DT	84.54	9.48	23.07	33.76	71.33	48.38	19.80	88.65
LinSVM	84.54	23.27	**5.49**	54.22	77.00	54.83	8.69	90.98
RBFSVM	85.50	20.68	6.59	49.14	72.00	29.03	27.53	63.99
LinLS-SVM	86.47	18.10	7.69	44.09	77.33	56.98	**7.24**	93.31
RBFLS-SVM	85.99	10.34	18.68	33.19	76.33	50.53	11.59	86.31
ANN	83.09	16.37	17.58	45.83	72.33	62.36	12.07	104.98

Table 4. Results of VNS and other classifiers for Thomas and Taiwan datasets.

Dataset	Thomas				Taiwan			
	Accuracy	Error TypeI	Error TypeII	Cost function	Accuracy	Error TypeI	Error TypeII	Cost function
BoostVNS	**68.98**	5.76	64.76	**53.04**	75.54	11.32	72.44	70.67
LDA	59.67	41.80	37.39	94.89	71.65	24.07	42.84	58.67
LR	67.30	47.95	**2.43**	81.95	80.18	22.62	**10.25**	**31.27**
knn	66.21	2.86	95.12	68.02	82.61	3.56	64.33	54.81
DT	62.12	23.36	66.66	83.45	74.57	17.05	53.83	60.24
LinSVM	66.75	17.62	64.22	72.21	82.52	2.31	68.97	57.22
RBFSVM	59.40	40.57	40.65	94.99	81.17	**1.98**	76.01	62.48
LinLS-SVM	66.75	**0.40**	98.37	66.06	76.91	4.34	74.38	63.60
RBFLS-SVM	66.48	1.63	96.74	67.04	75.38	7.13	83.97	74.08
ANN	66.48	**0.40**	99.18	66.60	**83.66**	3.98	58.28	50.42

Tables 3 and 4 give experimental results of the performance of the different classifiers on four credit datasets. Notably, BoostVNS outperforms author classifiers using the Australian and Thomas datasets, with highest accuracy rates of 88.63% and 68.98%, respectively. In addition, BoostVNS has lower cost functions (27.02% and 53.04%, respectively). These robust results demonstrate the effectiveness of BoostVNS in accurately classifying instances in these datasets. Although BoostVNS performs relatively well in terms of accuracy and cost function on the German and Taiwanese datasets, its performance is comparable to that of other approaches. BoostVNS, as mathematical approach, holds promising potential for future applications.

5. Conclusion

In this paper, we have used VNS to solve the linear mathematical model for credit scoring problem as described in subsection 2.2. The VNS metaheuristic has proved its performance in estimating the values of the weight vector of our model. For the linearly separable data, the VNS algorithm found the model that perfectly separates the customers. Also, for the linearly inseparable data and the real

data it found a model that separates the data with great performance. The application of VNS with bootstrap method has brought improvements, so it is an appropriate way to apply to the field of credit scoring.

[1] Crook J. N., Edelman D. B., Thomas L. C. Recent developments in consumer credit risk assessment. European Journal of Operational Research. **183** (3), 1447–1465 (2007).

[2] Lee T.-S., Chiu C.-C., Chou Y.-C., Lu C.-J. Mining the customer credit using classification and regression tree and multivariate adaptive regression splines. Computational Statistics & Data Analysis. **50** (4), 1113–1130 (2006).

[3] Thomas L. C. A survey of credit and behavioural scoring: forecasting financial risk of lending to consumers. International Journal of Forecasting. **16** (2), 149–172 (2000).

[4] Westgaard S., van der Wijst N. Default probabilities in a corporate bank portfolio: a logistic model approach. European Journal of Operational Research. **135** (2), 338–349 (2001).

[5] Henley W. E., Hand D. j. Construction of a k-nearest-neighbour credit-scoring system. IMA Journal of Management Mathematics. **8** (4), 305–321 (1997).

[6] Khashman A. Neural networks for credit risk evaluation: investigation of different neural models and learning schemes. Expert Systems with Applications. **37** (9), 6233–6239 (2010).

[7] Rosenberg E., Gleit A. Quantitative methods in credit management: a survey. Operations Research. **42** (4), 589–613 (1994).

[8] Baesens B., Van Gestel T., Viaene S., Stepanova M., Suykens J., Vanthienen J. Benchmarking state-of-the-art classification algorithms for credit scoring. Journal of the Operational Research Society. **54** (6), 627–635 (2003).

[9] Mladenović N., Hansen P. Variable neighborhood search. Computers & Operations Research. **24** (11), 1097–1100 (1997).

[10] Hansen P., Mladenović N. Variable neighborhood search: principles and applications. European Journal of Operational Research. **130** (3), 449–467 (2001).

[11] Bickel P. J., Götze F., van Zwet W. R. Resampling fewer than n observations: Gains, losses, and remedies for losses. Statistica Sinica. **7** (1), 1–31 (1997).

[12] Politis D., Romano J., Wolf M. Subsampling. Springer, New York (1999).

[13] Bickel P. J., Sakov A. Extrapolation and the bootstrap. Sankhya: The Indian Journal of Statistics, Series A. **64** (3), 640–652 (2002).

[14] Bickel P. J., Sakov A. On the choice of m in the m out of n bootstrap and confidence bounds for extrema. Statistica Sinica. **18**, 967–985 (2008).

[15] Kleiner A., Talwalkar A., Sarkar P., Jordan M. I. The big data bootstrap. Preprint arXiv:1206.6415 (2012).

[16] Kleiner A., Talwalkar A., Sarkar P., Jordan M. I. A scalable bootstrap for massive data. Journal of the Royal Statistical Society: Series B (Statistical Methodology). **76** (4), 795–816 (2014).

[17] Sengupta S., Volgushev S., Shao X. A Subsampled Double Bootstrap for Massive Data. Journal of the American Statistical Association. **111** (515), 1222–1232 (2016).

[18] Caravagna G., Ramazzotti D. Learning the structure of Bayesian Networks via the bootstrap. Neurocomputing. **448**, 48–59 (2021).

[19] Magasarian O. L. Linear and non linear separation of patterns by linear programming. Operations Research. **13** (3), 444–452 (1965).

[20] Freed N., Glover F. Simple but powerful goal programming models for discriminant problems. European Journal of Operational Research. **7** (1), 44–60 (1981).

[21] Freed N., Glover F. Evaluating alternative linear, programming models to solve the twogroup discriminant problem. Decision Science. **17** (2), 151–162 (1986).

[22] Bequé A., Lessmann S. Extreme learning machines for credit scoring: An empirical evaluation. Expert Systems with Applications. **86**, 42–53 (2017).

[23] Teng G.-E., He C.-Z., Xiao J., Jiang X.-Y. Customer credit scoring based on HMM/GMDH hybrid model. Knowledge and Information Systems. **36** (3), 731–747 (2013).

Гібридний пошук за змінною околицею з технікою повторної вибірки початкового завантаження для проблеми кредитного рейтингу

Бархдаді М., Беньякуб Б., Узінеб М.

Національний інститут статистики та прикладної економіки, Рабат, Марокко

Моделі кредитного рейтингування зіграли життєво важливу роль у наданні кредитів кредиторами та фінансовими установами. Останнім часом їм приділяють більше уваги у зв'язку з практикою управління ризиками. Було розроблено багато методів моделювання для оцінки добробуту позичальників. У цій статті подано модель оцінки кредитоспроможності за допомогою одного з методів локального пошуку — алгоритму пошуку за змінною околицею (VNS). Оптимізація структури околиці VNS є корисним методом, застосованим для вирішення проблем кредитного рейтингування. Одночасно налаштовуючи структуру околиці, запропонований алгоритм генерує оптимізовані ваги, які використовуються для побудови лінійної дискримінантної функції. Експериментальні результати, отримані шляхом застосування цієї моделі на змодельованих і реальних наборах даних, доводять її високу ефективність і високо оцінюють її значення для кредитного рейтингування.

Ключові слова: *кредитне рейтингування; математичне моделювання; алгоритм пошуку за змінною околицею; лінійна класифікація; техніка повторної вибірки.*