Neural Comput & Applic (2019) 31:337–346
DOI 10.1007/s00521-017-3018-4

ORIGINAL ARTICLE

A new algorithm of modified binary particle swarm optimization based on the Gustafson-Kessel for credit risk assessment

F. O. Sameer[1] · M. R. Abu Bakar[1] · A. A. Zaidan[2] · B. B. Zaidan[2]

Received: 22 September 2016 / Accepted: 13 April 2017 / Published online: 8 July 2017

Abstract To increase the quality of loans provision and reduce the risk involved in this process, several credit scoring models have been developed and utilized to improve the process of assessing credit worthiness. Credit scoring is an evaluation of the risk connected with lending to clients (consumers) or an organization. The Gustafson-Kessel (GK) algorithm has become one of the most valuable tools for credit scoring. However, this algorithm demonstrates a relatively poor capability to identify a subset of features from a large dataset. Most methods that use the GK algorithm require a predefined number of clusters. This paper presents a new GK-based modified binary particle swarm optimization (MBPSO) approach to increase the classification accuracy of the GK algorithm. The proposed MBPSO consists of three parts. First, the figure of particles is utilized to determine the optimal number of clusters automatically and overcome the drawback of the GK algorithm that requires a predefined number of clusters. A subset of features is identified because the same dataset may contain influencing features or a high level of noise. The two procedures are then combined in the same optimization method to increase the classification accuracy of the GK algorithm. Second, the updating function uses velocity and position to update the next position for every particle in the swarm. Third, a kernel fuzzy clustering method (KFCM) is used as the fitness function because this function can analyze high- dimensional data. These modifications are utilized as preprocessing steps before the classification of credit data is performed. Internal measures of clustering are conducted on Australian, German, and Taiwan standard datasets that contain 690, 1,000, and 30,000 instances, respectively, with several feature properties. Results show that the GK algorithm is good at separating the data into clusters. Furthermore, the fuzzy Rand validity measures of the three credit datasets derived by using the proposed method of combining the GK algorithm with a MBPSO are greater than the values of the two other compared methods. This finding means that fuzzy partitioning (classification) is robust therefore, the risk associated with loans provision can be reduced when the proposed method is used.

Keywords Credit scoring · Gustafson-Kessel algorithm · Feature subset selection problem · Binary particle swarm optimization

✉ A. A. Zaidan
aws.alaa@fskik.upsi.edu.my

F. O. Sameer
fadfhaa55@fsm.upm.edu.my

M. R. Abu Bakar
Bakar@fsm.upm.edu.my

B. B. Zaidan
bilal@fskik.upsi.edu.my

[1] Department of Mathematics, Universiti Putra Malaysia, UPM, 43400 Serdang, Selangor, Malaysia

[2] Department of Computing, Faculty of Arts, Computing and Creative Industry, Universiti Pendidikan Sultan Idris, Kajang, Malaysia

1 Introduction

The most successful operational research technique in the financial sector is credit scoring [1]. One of the most crucial procedures in a bank is credit evaluation, which is also known as a credit management decision. This procedure includes the collection and analysis of credit management

decisions. This procedure involves collection and analyzation of data and classification of different credit variables to arrive at a credit decision [2]. Empirical credit scoring has replaced judgment credit scoring. Empirical credit scoring is essentially the method which produces a "score" that a bank can use to rank its loan applicants or borrowers in terms of risk. To build an empirical credit scoring, historical data and statistical techniques were used; credit scoring tries to isolate the effects of various applicant characteristics on delinquencies and defaults. For example, the different between good and bad account [3]. Empirical credit scoring is more formal and accurate than judgment credit scoring so it has been improved based on the results of judgment credit scoring. Different statistical and artificial intelligence (AI) system techniques have been utilized to improve the performance of empirical credit scoring models [3, 4], such as discriminate analysis [5], linear probability, logit [6], and probit models [7]. These statistical techniques are suitable for identifying a linear relationship between independent and dependent variables. However, when these variables have non linear relationships, an AI technique such as a neural network is more appropriate than statistical techniques. Examples of AI techniques include the Bayesian network, support vector machine (SVM) and integer programming [8], k-nearest neighbor (KNN) [9], and classification tree [10]. These techniques are commonly used to model and assess credit risks [11]. However, these methods possess weaknesses. For example, representation of knowledge by an artificial neural network (ANN)is difficult because an ANN requires a large number of training samples and a long learning time [12].

Clustering is the most recommended classification method. Cluster analysis is a non-parametric statistical method that can be used for credit scoring. One of its principal advantages is that it does not assume presence of a specific data distribution. Thus, it is suitable when prior knowledge is insufficient. Cluster analysis is typically applied when no prior hypotheses exist. The method is exploratory and identifies the most likely solution so it is suitable for credit risk analysis [13]. In clustering, similar data points or elements are grouped into the same cluster and different data points are placed into different clusters. Clustering is performed through partitioning and hierarchical clustering. Partitioning clustering employs either the hard or the fuzzy clustering technique. Hard clustering assumes that each element of the dataset belongs to only one cluster, whereas fuzzy clustering assumes that each element (point) in a dataset could belong fully or partially to all or several clusters. Fuzzy clustering is more flexible than hard clustering and thus more suitable for real-world problems [14].

The GK algorithm is a powerful fuzzy clustering technique that has been used in various applications, such as image processing, data classification, and system identification. The GK algorithm is similar to the fuzzy clustering method (FCM) algorithm. The main difference between them is how they calculate distances. The FCM algorithm uses the square Euclidean distance measure, whereas the GK algorithm uses the Mahalanobis distance measure. Mahalanobis distance employs a covariance matrix, in which the clusters are in ellipsoidal, hyper-ellipsoidal, or other forms. Clusters in the FCM algorithm are in a spherical form, and the cluster shape does not change according to the type of data.

In cluster analysis, one of the most challenging problems is the number of clusters in a dataset. The number of clusters is usually known or fixed in advance, and this value is a crucial parameter for most clustering algorithms. However, having a predetermined number of clusters is unrealistic for many data analyses in the real world. Therefore, for algorithms such as the GK algorithm, a cluster validation technique needs to be used to determine the number of clusters [14].

Clusters in high dimensions cannot be visualized because high-dimensional data have their own distinct characteristics. This impossibility makes the identification and definition of clusters within these data challenging [15].

Credit scoring databases are often large and contain many redundant and irrelevant features. Classifying these data is therefore demanding in terms of computation. This difficulty can be overcome by using a feature selection method.

This study proposes a modified binary particle swarm optimization (MBPSO) algorithm to address the problems of feature subset selection and determination of the number of clusters in credit scoring data. Classification accuracy is improved to build a simple and robust credit scoring model. The most relevant features are selected, and an evolutionary computing optimization-based approach is utilized. Binary particle swarm optimization (BPSO) and kernel fuzzy clustering are the sources of inspiration. Although the proposed method is under BPSO, it is different from other BPSO methods in terms of the representation of the positions of particles (problems of feature subset selection and integrating the number of clusters) and in terms of updating the positions of the particles (inclusion/exclusion of features and number of clusters). The proposed approach is essentially a modification of the discrete PSO method [16].

The remainder of the paper is organized as follows. Section 2 provides a brief background and review of prior studies on feature subset selection and number of clusters. Section 3 describes the proposed MBPSO+GK. Section 4 presents the GK algorithm. Section 5 shows the measures of cluster validity. Section 6 shows the results of the proposed MBPSO+GK and a comparison of these results with those of two baseline algorithms, namely, BPSO+GK and GK.

The conclusions and several recommendations for future research are presented in Section 7.

2 Background and review of prior studies

2.1 Feature selection

Several methods have been proposed to improve feature selection. These include filter and wrapper methods. Filter methods use factor, principal component, discriminant, or independent component analysis to statistically test the variables and identify the best features. These methods can also use other distance and information measures for indirect performance measurement. Filter methods assess the key properties of the data and the properties of the classifier. Hence, these methods are quick and straightforward. However, filter methods are sensitive to redundancy [17].

By contrast, wrapper methods consider the accuracy of the classifier in selecting the best features. Consequently, the results are dependent on the type of classification algorithm used. Given that the resulting subset of features is closely linked to the classifier used, these methods are often not generalizable. Another drawback of wrapper approaches is that the search space for the selection of (n) features is 2^n; searching for features is therefore computationally expensive [18]. Regardless of the approach adopted for feature selection, the search strategy can significantly influence the results. Many wrapper techniques have been utilized for feature selection, but most of them tend to become stuck in local optima [19]. Most wrapper algorithms are either exact methods that apply the branch and bound principle or "involve" methods that apply greedy sequential subset selection, mathematical programming, nested partitioning, and meta-heuristics [20]. Two examples of greedy methods are sequential forward and backward selection [21]. An example of mathematical programming is that proposed in [22], which uses successive linearization and a bilinear algorithm to select the feature subset through a parametric objective function [22]. Meanwhile, the use of nested partition method was demonstrated in the work of [23], and this approach was later extended to an adaptive version [24].

Another study proposed a stochastic gradient descent algorithm, in which each feature is given a weight based on its importance to classification to find the best subset of features [25]. A comparative study of four feature selection methods was presented, which use data mining approach in reducing the feature space. The final results show that among the four feature selection methods, the Gini index and information gain algorithms perform better [26]. Various meta-heuristic algorithms have also been proposed to improve feature selection. These include genetic algorithms (GA), simulated annealing (SA), and PSO. For instance, a GA was used in a previous study to optimize the feature subset and effectively model the parameters for SVM; this approach demonstrated good performance [27]. A method of genetic algorithm (GA) based neural network was proposed for feature selection [28]. In [29], the use of a particle swarm optimization (PSO) and a genetic algorithm (GA) (both augmented with support vector machines SVM) for the classification of high-dimensional microarray data. An SA approach was developed in another study to obtain the parameters for feature selection in SVM [30]. The authors named the approach $SA - SVM$. Furthermore, a hybrid method that combines artificial bee colony optimization and a differential evolving algorithm was proposed in [31] to improve feature selection and enhance classification accuracy. The method was tested on 15 datasets from the UCI repository and was determined to be successful [31]. This review of relevant literature indicates that a global search method is required to develop a successful feature selection algorithm. Evolutionary computation techniques are necessary because of their global search ability, and one of the best techniques is the particle swarm algorithm [32]. In a PSO-based approach, each particle moves toward its actual best position, so the best position is determined by the full swarm over several iterations. This concept is what guides PSO toward the optimal solution and is the basis upon which most types of PSO are developed [33]. For instance, the PSOSVM model, which is a hybrid of discrete PSO and SVM, was proposed to choose a appropriate feature subset of simulated data [34]. A modified PSO algorithm was also proposed to select a feature subset [35]. In another work, four feature selection approaches for feature preprocessing were combined; one of them is hybrid fuzzy a priori with PSO [36]. Meanwhile, a weighted binary swarm optimization method proposed for feature selection was modeled as a discrete optimization task [37].

2.2 Number of clusters

In most clustering methods, the number of clusters is already known. Thus, finding a way to enable an algorithm to automatically estimate the number of clusters when it is not known in advance remains a major challenge. A previous study attempted to overcome this problem [38]. In this previous study, a dynamic clustering approach based on PSO was proposed. A binary PSO was used to find the best number of clusters. Then, the center of the selected cluster was refined by K-means clustering. The authors applied their approach called DCPSO to an unsupervised image classification task with some success [38]. Another promising method is that proposed by [39], in which the classical PSO is modified to a kernel-induced similarity measure instead of a sum of squares distance. The proposed approach, which the authors named the multi-elitist PSO

(MEPSO) method, can find the optimal number of clusters automatically [39]. A clustering method that can deal with different numbers of clusters has also been proposed; this method combines CPSO and K-means algorithms [40].

An improved version was subsequently proposed and called CPSOII. CPSOII, which is a combination of PSO and a dynamic clustering algorithm, can automatically find the best number of clusters and categorize data objects [41]. A robust PSO-based clustering method that considers the local density of data to measure how compact clusters are and that can also automatically estimate the number of clusters was put forward by [42] that method can deal well with noise.

Overall, this review indicates that most previous studies did not use BPSO to select the feature subset and estimate the number of clusters.

3 The proposed algorithm (MBPSO)

In BPSO, the position or state of each particle is a binary that can be changed from 1 to 0 or from 0 to 1 [16]. Particle velocity is defined as the probability that the state could change or mutate from 0 to 1 or vice versa. To discover the optimal solution, each particle changes its direction during its search for the feature space either because of its own particular best experience or cognitive learning (p_{best}) resulting from the best experience of all the other particles (i.e., the swarm's collective social learning (g_{best}).

Each particle retains the ((p_{best}) value, which is the best fitness value, at position (P_i) and the best value at position (p_{best}). Each particle represents a candidate or solution and is considered a point in a D-dimension space. It is represented by its position and velocity.

In BPSO, the velocity of every particle is updated as follows

$$v_{id}^{t} = v_{id}^{t-1} + c_1 r_1 \left(p_{id}^{t} - x_{id}^{t}\right) + c_2 r_2 \left(p_{gd}^{t} - x_{id}^{t}\right) \quad (1)$$

The changing positions of the particles are calculated by update functions. For instance, if a particle $s\left(v_{id}^{t+1}\right)$ is larger than a random number between 0 and 1,its position is represented by 1 (i.e., the position is selected for update).However, if $s\left(v_{id}^{t+1}\right)$ is smaller than a random number between 0 and 1,its position is represented by 0 (i.e., the position is not selected for update) [43].

$$s\left(v_{id}^{t+1}\right) = \frac{1}{1 + e^{-v_{id}^{t+1}}} \quad (2)$$

where s is the sigmoid function.

$$If\,(\text{rand} < s\left(v_{id}^{t+1}\right) \quad \text{then } x_{id} = 1 \quad \text{else } x_{id} = 0 \quad (3)$$

$d = 1, 2, \ldots, D$ where c_1 indicates the cognition learning and c_2 indicates social learning. Usually $c_1 = c_2 = 2$, and r_1 and r_2 are random numbers uniformly distribution in $U(0, 1)$. Also, the velocity of the dimension is limited to $v_{\max}$, and $v_{\max}$ is specified by the parameters set by the user of the algorithm. Each particle then moves to a new potential solution based on (3).

Notably, the update function of BPSO does not consider the current position of a particle in a binary search space, so the choice of the next position is not influenced by the current position. Thus, velocity alone represents the particle, although the binary position already exists in the BPSO. Therefore, we used velocity and position similar to the original PSO. The update function of the original BPSO is changed as follows:

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t} \quad (4)$$

$$If\left(\text{rand} < \exp\left(x_{id}^{t+1} - x_{id}^{t}\right)\right) \quad \text{then } x_{id} = 1 \quad \text{else } x_{id} = 0 \quad (5)$$

where $\exp\left(x_{id}^{t+1} - x_{id}^{t}\right)$ is the exponential function of the particle in two successive steps. In our approach, first, the number of particles required is set. Second, each particle's initial coding string is produced randomly. In the process of selecting features, each particle is coded to mimic a chromosome in GA. In other words, each particle is coded into a binary alphabetical string $x = a_1, a_2, \ldots, a_n, f_1 f_2 \ldots f_D$, where the first section of the particle $(a_1, a_2, \ldots, a_n)$ represents the numbers of clusters (0 or 1). The number of maximum clusters is n, and the numbers of 1s are counted to represent the number of clusters, where D is the number of features in the data. A bit value of 1 in second section $f_1 f_2 \ldots f_D$ denotes a selected feature and a bit value of 0 denotes an unselected feature.

If $\exp\left(x_{id}^{t+1} - x_{id}^{t}\right)$ is larger than a randomly produced number that is within (0, 1),its position value f_m, m = 1, 2, …, D is represented as 1, which means that the feature is selected because it is required for the next update.

In the proposed MBPSO, the combination of PSO with a kernel-based fuzzy clustering algorithm allows the cluster number to be determined and feature selection to be performed. Each particle's fitness is evaluated by KFCM. When the fitness of a particle is better than its best fitness, the position vector is saved for the particle. However, when its fitness is better than the global best fitness, then the position vector is saved for the global best. The particle's velocity and position are updated until the stopping criterion or criteria are met. Using PSO can lead to fast convergence during optimization. Moreover, precision can be improved by combining PSO with a KFCM algorithm.

The MBPSO consists of the following steps:

Step 1: Initialize the particles in the swarm population to provide them random positions and velocity vectors.

Step 2: Measure the fitness of each particle position by using the KFCM algorithm as follows:

1- Input the dataset $X = (x_1, x_2, ..., x_n)$ and remove the features from the data as the practicals positions.
2- Identify the number of groups (clusters)from the positions of the initial practicals and select the stopping criterion(the number of iterations or generations reach a prespecified value).
3- Choose the initial centers of the clusters from the data randomly and calculate the partition matrix u_{ij} by:

$$u_{ij} = \frac{\left(\frac{1}{(1-K(x_i,c_j))^{\frac{1}{(m-1)}}}\right)}{\sum_{j=1}^{k}\left(\frac{1}{(1-K(x_i,c_j))^{\frac{1}{(m-1)}}}\right)} \tag{6}$$

where

$$K(x, c) = \phi(x)^T \phi(c) \tag{7}$$

and is an inner product kernel function if we adopt the Gaussian function as a kernel function, i.e., $K(x_i, c_j) = \exp\left(-\frac{|x_i - c_j|^2}{\sigma^2}\right)$.

Since σ is presented as a dispersion, the sample variance is used to estimate σ^2 with $\sigma^2 = \frac{\sum_{j=1}^{n}(x_j - \overline{x})^2}{n}$, $\overline{x} = \frac{\sum_{j=1}^{n} x_j}{n}$.

4- Update the center matrix c_j by the formula:

$$c_j = \frac{\sum_{i=1}^{n} u_{ij}^m K(x_i, c_j) x_i}{\sum_{i=1}^{n} u_{ij}^m K(x_i, c_j)} \tag{8}$$

where $i = 1, 2, ..., n$, $j = 1, 2, ..., k$, and $m = 2$ is the fuzziness parameter.

5- Calculate the objective function of each partition and select the minimum value for the fitness function value of each particle.

$$J(X, U, C) = \sum_{i=1}^{n}\sum_{j=1}^{k}(u_{ij})^m \mid \phi(x_i) - \phi(c_j) \mid^2 \tag{9}$$

where

$$\mid \phi x_i - \phi c_j \mid^2 = K(x_i, x_i) + K(c_j, c_j) - 2K(x_i, c_j) \tag{10}$$

6- For convergence, test if the termination tolerance satisfies the following: $||C^I - C^{I+1}|| \leq \varepsilon$ where I is the iteration number.
7- Select the particle that has the minimum fitness function value as the local and global solution.

Step 3: Calculate the velocity of each particle for the next update.
Step 4: Move each particle to its next updated position according to (4, 5) and return to step 2 if the best position has not been found.
Step 5: Stop the algorithm if the stopping criterion or criteria are satisfied or if the number of iterations reaches the predetermined maximum number.

4 The Gustafson –Kessel algorithm

The GK algorithm is a powerful fuzzy clustering technique that can be used in many applications, such as image processing and classification. The algorithm estimates the cluster covariance matrix, which enables it to match the distance metric to the cluster shape [44], which is a key advantage. The GK algorithm needs a set of n samples in the p dimensional space and the number of clusters k as the input parameters. A fuzzy partition of a dataset X can be represented by a $(n * k)U = [u_{ij}]$, where u_{ij} gives the degree of membership that the ith object belongs into the jth cluster, where $(1 \leq i \leq n)$ and $(1 \leq j \leq k)$.

The GK algorithm has several similarities with the FCM algorithm. The main difference between them is that the FCM algorithm uses the square Euclidean distance measure, and the GK algorithm uses the Mahalanobis distance measure. The clusters formed by the FCM algorithm are spherical, and the cluster shape does not change according to the type of data. By contrast, in the GK algorithm, the cluster shape can change according to the data and can be created in several different forms, such as ellipsoidal and hyper-ellipsoidal. Hence, the GK algorithm employs a covariance matrix. The GK algorithm consists of the following steps.

1- Dataset $X = (x_1, x_2, ..., x_n)$ is given.
2- Select the number of groups or clusters and the subset of features by modify binary particle swarm optimization (MBPSO) and select the termination condition.
3- Generate initial values for partition matrix U_{ij} to denote the degree of membership of x_i to the cluster j. This degree of membership satisfies the following constraints

- $0 \leq u_{ij} \leq 1$ for $i \in 1, 2, ..., n$, $j \in 1, 2, ..., k$
- $\sum_{j=1}^{k} u_{ij} = 1$ for $i \in 1, 2, ..., n$

4- Calculate the C-means matrix c_j with a dimension $(k \times p)$ where $1 \leq j \leq k$, that represents the center of the clusters by the following formula

$$c_j = \frac{\sum_{i=1}^{n} u_{ij}^m x_i}{\sum_{i=1}^{n} u_{ij}^m} \tag{11}$$

5- Compute the cluster covariance matrix by:

$$F_j = \frac{\sum_{i=1}^{n} u_{ij}^m (x_i, c_j)(x_i - c_j)^T}{\sum_{i=1}^{n} u_{ij}^m} \tag{12}$$

6- Compute the Mahalanobis distance by:

$$d_{ij}^2 = (x_i - c_j) A_j (x_i - c_j)^T \tag{13}$$

A_j defined as following:

$$A_j = V_j [\det(F_j)]^{\frac{1}{p}} F_j^{-1} \tag{14}$$

where x_i is the ith data, p is the number of features or attributes, c_j is the center of cluster j, V_j is the volume of cluster, and j $V_j = 1$, F_j^{-1} is the inverse of matrix F_j.

7- Update the partition matrix u_{ij} by:

$$u_{ij} = \frac{1}{\sum_{r=1}^{k} \left(\frac{d_{ij}}{d_{rj}}\right)^{\frac{2}{m-1}}} \tag{15}$$

Where $i = 1, 2, ..., n$, $j = 1, 2, ..., k$ and $i \neq r$.

8- For convergence, test if the termination tolerance satisfies the following: $||U^I - U^{I+1}|| \leq \varepsilon$

5 Measures of cluster validity

5.1 Internal measures

Several internal indices are used simultaneously, and the most important ones are described below.

1- The partition coefficient (PC) measures the amount of overlapping between clusters and is defined as follows [45]:

$$\mathrm{PC} = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} u_{ij}^2 \tag{16}$$

where u_{ij} is the membership degree of the i_{th} data point in the j_{th} cluster. The best algorithm for partitioning the data is the one that produces the highest value of PC.

2- Classification entropy (CE) measures only the fuzziness of the cluster partitions, so it is similar to PC [45].

$$\mathrm{CE} = -\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} u_{ij} \log(u_{ij}) \tag{17}$$

The best clustering algorithm is the one with the lowest value of CE.

3- The partitions index (SC) [46] is the ratio between the sum of the separation and the compactness of the clusters. It is the sum of the cluster validity measures for each individual divided by the fuzzy cardinality for each cluster.

$$\mathrm{SC} = \sum_{j=1}^{k} \frac{\sum_{i=1}^{n} u_{ij}^m ||x_i - c_j||^2}{\sum_{i=1}^{n} u_{ij} \mathrm{sum}_{d=1}^{k} ||c_d - c_j||^2} \tag{18}$$

SC is useful for comparing different partitions with an equal number of clusters. A good partition is obtained by a low value of SC.

4- The separation index (S) [46] in contrast to the SC, uses a minimum-distance separation for partition validity. A lower value of S indicates a good partition.

$$S = \sum_{j=1}^{k} \frac{\sum_{i=1}^{n} u_{ij}^2 ||x_i - c_j||^2}{n \min_{i \neq j} ||c_i - c_j||^2} \tag{19}$$

5- The Xie and Beni's (XB)index [47] aims to measure the proportion between the total variation within clusters and the separation of clusters. It is defined as follows:

$$XB = \sum_{j=1}^{k} \frac{\sum_{i=1}^{n} u_{ij}^m ||x_i - c_j||^2}{n \min_{ij} ||x_i - c_j||^2} \tag{20}$$

This index focuses on separation and compactness properties. The clusters are well separated if XB has a small value.

6- The Dunn's Index (DI) [47] aims to recognize dense and well-separated clusters. It is defined as the proportion between the minimal intra-cluster distance and the maximal inter-cluster distance. For each cluster partition, this index can be identified as follows:

$$DI = \min_{j \in k} \left\{ \min_{j \in k (i \neq j)} \left\{ \frac{\min_{x \in c_i, y \in c_j} d(x, y)}{\max_{x, y \in k} d(x, y)} \right\} \right\} \tag{21}$$

A high Dunn's index denotes that the a desirable algorithm is suitable for producing clusters.

7- Davies-Bouldin index (DB) [48] is defined as follows:

$$\mathrm{DB} = \frac{1}{n} \sum_{i=1}^{n} \max_{i \neq j} \left(\frac{d_i + dj}{d(c_i, c_j)} \right) \tag{22}$$

where (k) is the number of clusters, c_j and c_i are the centers of clusters, and respectively d_j and d_i are the average distances of all the elements in clusters (j) and (i) respectively and $d(c_i, c_j)$ is the distance between the centers c_i and c_j. The best algorithm is the clustering algorithm that produces a collection of clusters with the smallest DB index.

5.2 External measures

External measures indicate the quality of the resulting partitioning; thus, they can be considered tools that can help experts evaluate the clustering results. The fuzzy Rand

Table 1 Dataset description

Dataset	Classes	Instances	Features
Australian	2	690	14
German	2	1000	24
Taiwan	2	30000	23

index is a well-known measure of similarity between two partitions of a dataset [49].

Given a fuzzy partition $w = \{W_1, W_2, \ldots, W_k\}$ of X, each element $x \in X$ can be characterized by its membership vector

$$W(x) = (W_1(x), W_2(x), \ldots, W_k(x)) \in [0, 1]^k \quad (23)$$

where $W_i(x)$ is the degree of membership of x in the ith cluster W_i. A similarity measure for associated membership vectors can be formed as follows:

$$E_W(x, x') = 1 - ||W(x) - W(x')|| \quad (24)$$

where $||.||$ is a proper metric on $[0, 1]^k$. If W and Z are two fuzzy partitions to generalize the concept of concordance, a pair $(x; x')$ is defined and the degree of concordance is:

$$\text{conc}(x, x') = 1 - ||E_W(x, x') - E_Z(x, x')|| \in [01] \quad (25)$$

the degree of discordance is

$$\text{disc}(x, x') = 1 - ||E_W(x, x') - E_Z(x, x')|| \quad (26)$$

The distance measure for the fuzzy partitions is then defined by the normalized sum of the degrees of discordance as follows:

$$d(W, Z) = \frac{\sum_{(x,x')\in X} ||E_W(x, x') - E_Z(x, x')||}{(N(N-1)/2)} \quad (27)$$

Likewise,

$$\text{RE}(W, Z) = 1 - d(W, Z) \quad (28)$$

This condition corresponds to the normalized degree of concordance and is a direct generalization of the original Rand index. The Rand index is a similarity measure that assumes values between 0 and 1. If the value is near 1 this means that i_{th} the cluster in W and the i_{th} cluster in Z are identical thus $W = Z$.

6 Results and discussion

6.1 Data description

To evaluate the performance of the proposed approach, Australian, German, and Taiwanese datasets from the UCI machine learning repository were used. Table 1 shows the characteristics of the datasets. The input variables were scaled during the data preprocessing stage. The main advantage of scaling is that it prevents attributes in greater numerical ranges from dominating over those in smaller numerical ranges. Another advantage is that it can prevent numerical difficulties during calculation. Scaling of the feature value can also help increase accuracy according to our experimental results. Generally, each feature can be linearly scaled to the [0, 1] range by using the following formula

$$x^1 = \frac{x - \min_x}{\max_x - \min_x} \quad (29)$$

where x is original value, x^1 is the scaled value, max is the maximum value of feature x, and min is the minimum value of feature x.

The following tables show the internal indices for the clustering of the Australian, German, and Taiwanese credit data. The first column shows the values of the indices of the GK algorithm, and the second column presents the proposed algorithm (GK+BPSO) with three cases. The first one is for feature selection only, the second one is for determining the

Table 2 The validity measures of Australian credit data

Algorithms	GK	GK+BPSO			GK+MBPSO		
Validity measures		*F*	*N*	*F*+*N*	*F*	*N*	*F*+*N*
PC	0.7277	0.5000	0.3333	0.3333	0.9973	0.2766	0.8776
CE	3.3897	0.6931	1.0986	1.0986	0.0071	1.4169	0.2071
SC	2.9343	1.3830	2.3813	2.8578	1.1641	0.9723	0.8730
S	4.7605	0.6915	0.7938	0.9578	2.4276	0.2569	0.8540
XB	1.8017	0.3458	0.2646	0.8480	1.9358	0.0760	0.8086
DI	0.4938	0.0065	0.0395	0.0231	2.5094	0.0191	0.0338
DB	1.4972	0.7322	0.7782	0.7782	0.7782	0.7782	0.6929
J (objective function)	94.24	185.95	150.92	139.9	74.281	51.55	45.02
No. iteration	31	22	20	16	16	13	12
No. clusters	2		3	3		5	5
Features selection	14	6		11	6		8

Table 3 The validity measures of German credit data

Algorithms	GK	GK+BPSO			GK+MBPSO		
Validity measures		*F*	*N*	*F*+*N*	*F*	*N*	*F*+*N*
PC	0.5400	0.5000	0.2000	0.1667	0.7458	0.7066	0.9761
CE	2.6931	0.6931	1.6094	1.7918	0.4157	1.5919	0.6009
SC	4.7122	0.7108	2.3813	0.3901	1.7756	2.0262	0.0715
S	2.3600	0.3554	0.1589	0.0650	1.1608	1.0553	0.0054
XB	3.1800	0.1777	0.0318	0.9886	0.9339	2.0011	0.8080
DI	0.0790	0.3694	0.0892	0.0334	0.3078	0.4229	0.8145
DB	3.7400	0.7240	0.7117	0.7382	0.7387	1.7387	0.0159
J (objective function)	477.33	467.10	262.92	148.97	288.27	350.95	110.02
No. iteration	19	14	6	9	9	9	9
No. clusters	2		5	3		5	4
Features selection	24	9		13	13		10

number of clusters, and the third is for both feature selection and determining the number of clusters. The third column shows GK+MBPSO for the three cases.

Table 2 shows the values of the internal indices for Australian data for GK, GK+BPSO, and GK+MBPSO. The value of the first index (PC) for the GK+MBPSO algorithm is near 1, which is greater than its value for GK and GK+BPSO. The value of the second index (CE) for the GK+MBPSO algorithm is less than the value for GK and GK+BPSO. The values of the other indices (SC, S, and XB) for the GK+MBPSO algorithm are lower than the values for the GK and GK+BPSO algorithms. However, the value of DI for the proposed method is greater than the values for the other methods, and the value of DB for the GK+MBPSO algorithm is lower than the values for GK and GK+BPSO algorithms. This result means that the algorithms separated clusters well. The proposed method determined that the number of clusters is five, and it selected eight features.

Table 3 shows the internal indices for German data for GK, GK+BPSO, and GK+MBPSO. The value of the first index (PC) for the GK+MBPSO algorithm is near 1, which is greater than its value for GK and GK+BPSO. The value of the second index (CE) for the GK+MBPSO algorithm is less than the value for GK and GK+BPSO. The values of the other indices (SC, S, and XB) for the GK+MBPSO algorithm are lower than the values for GK and GK+BPSO algorithms. However, the value of DI for the proposed method is greater than the values for the other methods, and the DB for GK+MBPSO is lower than the values for GK and GK+BPSO. This result means that the algorithms separated clusters well. The proposed method determined that the number of clusters is four, and it selected 10 features.

Table 4 shows the internal indices for Taiwan data for GK, GK+BPSO, and GK+MBPSO. The value of the first index (PC) for the GK+MBPSO algorithm is near 1, which is greater than its value for GK and GK+BPSO. The value

Table 4 The validity measures of Taiwan credit data

Algorithms	GK	GK+BPSO			GK+MBPSO		
Validity measures		*F*	*N*	*F*+*N*	*F*	*N*	*F*+*N*
PC	0.5277	0.6819	0.2009	0.3808	0.7622	0.6276	0.8082
CE	3.6931	0.4904	1.6072	1.2109	0.3822	1.0584	0.2423
SC	6.1690	4.4189	2.3873	2.3281	1.7746	2.0268	0.2071
S	8.1324	2.5224	5.1368	5.3894	4.1628	3.6466	1.0540
XB	3.5423	1.8456	1.0347	2.5067	0.9439	1.5700	0.9804
DI	0.0343	0.0027	0.0010	0.0016	1.8740	0.5329	0.7374
DB	1.8709	0.5849	0.5847	0.4172	0.5847	1.5847	0.3868
J (objective function)	155	244.88	162.92	163.30	150.23	140.25	90.02
No. iteration	30	25	16	41	40	17	20
No. clusters	2		5	5		4	5
Features selection	23	14		14	10		10

Table 5 The fuzzy rand validity measure of three credit data

Data set	GK	GK+BPSO	GK+MBPSO
Australian	0.8600	9517	0.9911
German	0.7888	0.8057	0.9955
Taiwan	0.8476	0.9022	0.9534

of the second index (CE) for the GK+MBPSO algorithm is less than the value for GK and GK+BPSO. The values of the other indices (SC, S, and XB) for the GK+MBPSO algorithm are lower than the values for GK and GK+BPSO algorithms. However, the value of DI for the proposed method is greater than the values for the other methods, and the DB for the GK+MBPSO algorithm is lower than that for GK and GK+BPSO algorithms. This result means that the algorithms separated clusters well. The proposed method determined that the number of clusters is five, and it selected 10 features.

As shown in the summarized results, the proposed modified method (MBPSO) for determining the number of clusters and for feature selection with the GK algorithm (GK+MBPSO) exhibits the best performance for the three datasets because it has a smaller distance function (objective function) and a smaller number of iterations for the three datasets. A *t* test was conducted on the internal index values of the proposed method (GK+MBPSO) and GK for the three datasets. The results demonstrate that significant differences exist between them at 95%, and the *P* value of proposed method (MBPSO+GK) and GK for the three datasets is 0.025, 0.033, and 0.043. As we compare with method of [31] the number of iterations is 300 iterations and repeated 10 times but our method do not exceed 40 iteration for three datasets. The number of data in their method is from 30–50 but our method the size of data 690,1000 and 30000.

Table 5 shows that the results of the fuzzy Rand validity measures for the Australian, German, and Taiwanese credit datasets are 0.9911, 0.9955, and 0.9933, respectively. The values of the (GK+MBPSO algorithm) are greater than the values of the two other methods. This finding means that fuzzy partition (classification) is robust, so the risk associated with loans can be reduced with this method.

7 Conclusion

We proposed a new modified BPSO-KFCM method for determining the number of clusters and for selecting features in fuzzy data clustering. We developed and improved the GK algorithm to increase classification accuracy for cluster analysis. The three algorithms were applied to Australian, German, and Taiwanese credit datasets, and their performance was compared. The cluster internal validity indices of the proposed method (GK+MBPSO) are better than those of the other algorithms. The *t* test on the internal indices of the proposed method (GK+MBPSO) demonstrated that significant differences exist among the methods at 95%. The results of the fuzzy Rand validity measures show that fuzzy partition (classification) is robust, so the risk associated with loans can be reduced with this method. In future work, other validation measures can be utilized to test the effectiveness of the proposed approach for cluster analysis. Moreover, the modified BPSO-KFCM can be improved to select the initial centers of clusters integrated with feature selection.

The cluster internal validity indexes confirm that the performance of the proposed algorithm (GK+MK) is better than that of the GK and GKK algorithms. A fuzzy validity index is applied in this paper for evaluating the fitness of clustering to data sets.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Gan G, Ma C, Wu J (2007) Data clustering: theory, algorithms, and applications, vol 20. SIAM
2. Sánchez JFM, Lechuga GP (2016) Assessment of a credit scoring system for popular bank savings and credit. Contad Adm 61(2):391–417
3. Abdou HA, Pointon J (2011) Credit scoring, statistical techniques and evaluation criteria: a review of the literature. Financ Manag 18(2–3):59–88
4. Leung K, Cheong F, Cheong C (2010) A comparison of traditional and simple artificial immune system (sais) techniques in consumer credit scoring. Int J Artif Intell Soft Comput 2(1–2):1–25
5. Fisher RA (1936) The use of multiple measurements in taxonomic problems. Ann Eugen 7(2):179–188
6. Wiginton JC (1980) A note on the comparison of logit and discriminant models of consumer credit behavior. J Financ Quant Anal 15(03):757–770
7. Grablowsky BJ, Talley WK (1981) Probit and discriminant functions for classifying credit applicants—a comparison. J Econ Bus 33(3):254–261
8. Mangasarian OL (1965) Linear and nonlinear separation of patterns by linear programming. Oper Res 13(3):444–452
9. Henley W, Hand DJ (1996) A k-nearest-neighbour classifier for assessing consumer credit risk. Statistician 45(1):77–95
10. Lee T-S, Chiu C-C, Chou Y-C, Lu C-J (2006) Mining the customer credit using classification and regression tree and multivariate adaptive regression splines. Comput Stat Data Anal 50(4):1113–1130
11. Lahsasna A, Ainon RN, Teh YW (2010) Credit scoring models using soft computing methods: a survey. Int Arab J Inf Technol 7(2):115–123
12. Abdou H, Pointon J, El-Masry A (2008) Neural nets versus conventional techniques in credit scoring in Egyptian banking. Exp Syst Appl 35(3):1275–1292

13. Bezdek JC, Ehrlich R, Full W (1984) Fcm: the fuzzy c-means clustering algorithm. Comput Geosci 10(2–3):191–203
14. Klawonn F, Höppner F (2009) Fuzzy cluster analysis from the viewpoint of robust statistics. In: Views on fuzzy sets and systems from different perspectives. Springer, pp 439–455
15. Klawonn F (2013) What can fuzzy cluster analysis contribute to clustering of high-dimensional data? In: International workshop on fuzzy logic and applications. Springer, pp 1–14
16. Kennedy J, Eberhart RC (1997) A discrete binary version of the particle swarm algorithm. In: 1997 IEEE international conference on systems, man, and cybernetics, 1997. Computational Cybernetics and Simulation, vol 5. IEEE, pp 4104–4108
17. Kabir MM, Shahjahan M, Murase K (2012) A new hybrid ant colony optimization algorithm for feature selection. Exp Syst Appl 39(3):3747–3763
18. Kohavi R, John GH (1997) Wrappers for feature subset selection. Artif Intell 97(1):273–324
19. Hu Q, Yu D, Liu J, Wu C (2008) Neighborhood rough set based heterogeneous feature subset selection. Inf Sci 178(18):3577–3594
20. Wang L, Li H, Huang JZ (2008) Variable selection in non-parametric varying-coefficient models for analysis of repeated measurements. J Am Stat Assoc 103(484):1556–1569
21. Somol P, Pudil P, Kittler J (2004) Fast branch & bound algorithms for optimal feature selection. IEEE Trans Pattern Anal Mach Intell 26(7):900–912
22. Yang J, Olafsson S (2006) Optimization-based feature selection with adaptive instance sampling. Comput Oper Res 33(11):3088–3106
23. Pudil P, Novovičová J, Kittler J (1994) Floating search methods in feature selection. Pattern Recogn Lett 15(11):1119–1125
24. Ólafsson S, Yang J (2005) Intelligent partitioning for feature selection. INFORMS J Comput 17(3):339–355
25. Bradley PS, Mangasarian OL, Street WN (1998) Feature selection via mathematical programming. INFORMS J Comput 10(2):209–217
26. Aryuni M, Madyatmadja ED (2015) Feature selection in credit scoring model for credit card applicants in xyz bank: a comparative study. Int J Multimed Ubiquitous Eng 10(5):17–24
27. Huang C-L, Wang C-J (2006) A ga-based feature selection and parameters optimization for support vector machines. Expert Syst Appl 31(2):231–240
28. Li T-S (2006) Feature selection for classification by using a ga-based neural network approach. J Chin Inst Indust Eng 23(1):55–64
29. Talbi E-G, Jourdan L, Garcia-Nieto J, Alba E (2008) Comparison of population based metaheuristics for feature selection: application to microarray data classification. In: 2008 IEEE/ACS international conference on computer systems and applications. IEEE, pp 45–52
30. Lin S-W, Lee Z-J, Chen S-C, Tseng T-Y (2008) Parameter determination of support vector machine and feature selection using simulated annealing approach. Appl Soft Comput 8(4):1505–1512
31. Zorarpacı E, Özel SA A hybrid approach of differential evolution and artificial bee colony for feature selection. Exp Syst Appl
32. Gadat S, Younes L (2007) A stochastic algorithm for feature selection in pattern recognition. J Mach Learn Res 8:509–547
33. Zhou Z, Liu X, Li P, Shang L (2014) Feature selection method with proportionate fitness based binary particle swarm optimization. In: Asia-Pacific conference on simulated evolution and learning. Springer, pp 582–592
34. Huang C-L, Dun J-F (2008) A distributed pso–svm hybrid system with feature selection and parameter optimization. Appl Soft Comput 8(4):1381–1391
35. Unler A, Murat A (2010) A discrete particle swarm optimization method for feature selection in binary classification problems. Eur J Oper Res 206(3):528–539
36. Sadatrasoul S, Gholamian M, Shahanaghi K (2015) Combination of feature selection and optimized fuzzy a priori rules: the case of credit scoring. Int Arab J Inf Technol 12(2):138–145
37. Moayedikia A, Jensen R, Wiil UK, Forsati R (2015) Weighted bee colony algorithm for discrete optimization problems with application to feature selection. Eng Appl Artif Intell 44:153–167
38. Omran MG, Salman A, Engelbrecht AP (2006) Dynamic clustering using particle swarm optimization with application in image segmentation. Pattern Anal Appl 8(4):332–344
39. Das S, Abraham A, Konar A (2008) Automatic kernel clustering with a multi-elitist particle swarm optimization algorithm. Pattern Recogn Lett 29(5):688–699
40. Kao Y, Lee S-Y (2009) Combining k-means and particle swarm optimization for dynamic data clustering problems. In: IEEE international conference on intelligent computing and intelligent systems, 2009. ICIS 2009, vol 1. IEEE, pp 757–761
41. Masoud H, Jalili S, Hasheminejad SMH (2013) Dynamic clustering using combinatorial particle swarm optimization. Appl Intell 38(3):289–314
42. H-L Ling, J-S Wu, Y Zhou, W-S Zheng How many clusters? A robust pso-based local density model. Neurocomputing
43. Kennedy J (2011) Particle swarm optimization. In: Encyclopedia of machine learning. Springer, pp 760–766
44. Gustafson D, Kessel W (1978) Fuzzy clustering with a fuzzy covariance matrix. Scientific Systems, Inc., Cambridge
45. Bezdek JC (2013) Pattern recognition with fuzzy objective function algorithms. Springer Science & Business Media
46. Bensaid AM, Hall LO, Bezdek JC, Clarke LP, Silbiger ML, Arrington JA, Murtagh RF (1996) Validity-guided (re) clustering with applications to image segmentation. IEEE Trans Fuzzy Syst 4(2):112–123
47. Xie XL, Beni G (1991) A validity measure for fuzzy clustering. IEEE Trans Pattern Anal Mach Intell 13(8):841–847
48. Wu K-L, Yang M-S (2005) A cluster validity index for fuzzy clustering. Pattern Recogn Lett 26(9):1275–1291
49. Hullermeier E, Rifqi M, Henzgen S, Senge R (2012) Comparing fuzzy partitions: a generalization of the rand index and related measures. IEEE Trans Fuzzy Syst 20(3):546–556