Determine Optimal Number of Clusters with an Elitist Evolutionary Approach

Lydia Boudjeloud-Assala and Ta Minh Thuy

Laboratory of Theoretical and Applied Computer Science
LITA EA 3097, University of Lorraine
Ile du Saulcy, Metz, F-57045, France
{lydia.boudjeloud-assala,minh-thuy.ta}@univ-lorraine.fr

Abstract. This article proposes an elitist evolutionary approach to determine the optimal number of clusters for clustering data sets. The proposed method is based on the cluster number optimization and in the same time, finds the potential clusters seeds. This method can be used as an initialization of k-means algorithm or directly as a clustering algorithm without prior knowledge of the clusters number. In this approach, elitist population is composed of the individuals with potential clusters seeds. We introduce a new mutation strategy according to the neighborhood search and new evaluation criteria. This strategy allows us to find the global optimal solution or near-optimal solution for clustering tasks, precisely finding the optimal clusters seeds. The experimental results show that our algorithm performs well on multi-class and large-size data sets.

Keywords: Elitist approach, Evolutionary algorithm, Clustering, Optimal number of clusters, Cluster seed initialization.

1 Introduction

Clustering is a challenging research area in data mining. A common form of clustering is partitioning the data set into homogeneous clusters such that members of the same cluster are similar and members of distinct clusters are dissimilar. Determining the optimal clusters number is one of the most difficult issues in clustering data. In this work, we deal with the clustering problem without prior knowledge on the appropriate clusters number and we try to propose the global optimal or near-optimal cluster seeds. Clustering algorithms can be broadly classified into two groups: hierarchical and partitional [1]. Hierarchical algorithms recursively find nested clusters either in a divisive or agglomerative method. In contrast, partitional algorithms find all the clusters simultaneously as a partition of the data and do not impose a hierarchical structure. Common formulation of the clustering problem is assuming S is the given data set include n data points: $S = \{x_1, x_2, ..., x_n\}$ where x_i $(i = 1, ..., n)$ is a real vector $d-dimensions$ and an integer k. The goal of clustering is to determine a set of k clusters $C_1, C_2, ..., C_k$ such that the points belonging to the same cluster are similar, while the points

V.S. Tseng et al. (Eds.): PAKDD 2014, Part II, LNAI 8444, pp. 324–335, 2014.

belonging to different clusters are dissimilar in the sense of the given metric. The problem of finding an optimal solution to the partition of n data into k clusters is $NP-complete$, and heuristic methods are widely effective on $NP-complete$ global optimization problems and they can provide good sub-optimal solutions in reasonable time. We propose a clustering algorithm that can detect compact and hyperspherical clusters that are well separated using an Euclidean distance. To detect hyperellipsoidal clusters, we can use a more general distance function such as the Mahalanobis distance for example [2]. Some recent research has shown that the problem of searching efficient initialization methods for k-means clustering algorithm is a great challenge. Numerous initialization methods have been proposed to address this problem. Celebi and al. [3] present an overview of these methods with an emphasis on their computational efficiency. In their study, they investigate some of the most popular initialization methods developed for the k-means algorithm. They describe and compare initialization methods that can be used to initialize other partitional clustering algorithms such as fuzzy c-means and its variants and expectation maximization, and they conclude that most of these methods can be used independently of k-means as standalone clustering algorithms. Some others methods are proposed, based on metaheuristics such as simulated annealing [4] and genetic algorithms [5]. These algorithms start from a random initial configuration (population) and use k-means to evaluate their solutions in each iteration (generation). There are two main drawbacks associated with these methods. First, they involve numerous parameters that are difficult to tune [6]. Second, due to the large search space, they often require a large number of iterations, which renders them computationally prohibitive for most of the data sets. This paper presents a method that proposes, in the same time, the optimal cluster number with the initial clusters seeds. We don't use k-means to evaluate our solution because it depends on several parameters itself (k, initial seeds, ...), or any other clustering algorithm, so, our method can be used as standalone clustering algorithms without a prior knowledge of cluster number. Generally metaheuristics approaches involve numerous parameters that are difficult to tune, to deal with this problem, we propose an Elitist Evolutionary Approach that involves numerous evolutionary algorithms (EAs). The difference between them is implemented by the parameters and we select only the best concurrent solution. The remainder of this paper is organized as follows: Section 2 describes the related work on initialization and elitist methods. Section 3 gives the details of the new proposed algorithm, including the algorithm description, motivation of population initialization, as well as the mutation process based on neighborhood search. Section 4 shows the experimental results on data sets clustering. Finally, we draw the conclusions.

2 Related Work

In this section, we briefly review some of the commonly used initialization methods and elitist methods.

2.1 Initialization Methods

Celebi and al. [3] investigate some of the most popular initialization methods developed for the k-means algorithm. Their motivation is threefold. First, a large number of initialization methods have been proposed in the literature. Second, these initialization methods can be used to initialize other partitional clustering algorithms such as fuzzy c-means and its variants and expectation maximization. Third, most of these initialization methods can be used independently of k-means as standalone clustering algorithms. They review some of the commonly used initialization methods with an emphasis on their time complexity. They conclude that the super linear methods often have more elaborate designs when compared to linear ones. An interesting feature of the super linear methods is that they are often deterministic, which can be considered as an advantage especially when dealing with large data sets. In contrast, linear methods are often non-deterministic and/or order-sensitive. A frequently cited advantage of the more elaborate methods is that they often lead to faster k-means convergence, i.e. require fewer iterations, and as a result the time gained during the clustering phase can offset the time lost during the initialization phase. This may be true when a standard implementation of k-means is used. However, convergence speed may not be as important when a fast k-means variant is used as such methods often require significantly less time compared to a standard k-means implementation. Some other initialization methods such as the binary-splitting method [7] takes the mean of data as the first center. In iteration t, each of the existing 2^{t-1} centers is split into two new centers by subtracting and adding a fixed perturbation vector. These 2^t new centers are then refined using k-means. There are two main disadvantages associated with this method. First, there is no guidance on the selection of a proper value for the vector, which determines the direction of the split [8]. Second, the method is computationally demanding since after each iteration k-means has to be run for the entire data set. Some other methods based on metaheuristics such as simulated annealing [4] and genetic algorithms [5]. These algorithms start from a random initial configuration and use classical algorithm such as k-means to evaluate their solutions in each iteration. There are two main disadvantages associated to these methods. First, they involve numerous parameters that are difficult to tune (initial temperature, cooling schedule, population size, crossover/mutation probability, etc.) [6]. Second, due to the large search space, they often require a large number of iterations, which renders them computationally prohibitive for all but the smallest datasets. Interestingly, with the recent developments in combinatorial optimization algorithms, it is now feasible to obtain globally minimum clusterings solution for small data sets without resorting to metaheuristics [9].

2.2 Elitist Methods

Prevent promising individuals from being eliminated from the population during the application of genetic operators is a very important task. To ensure that the best chromosome is preserved, elitist methods copy the best individual found

so far into the new population. Different EAs variants achieve this goal of preserving the best solution in different ways. However, elitist strategies tend to make the search more exploitative rather than explorative and may not work for problems in which one is required to find multiple optimal solutions [10]. Elitist methods are widely applied on different domains, Qasem and Shamsuddin [11] developed a Mimetic Elitist Pareto Differential Evolution algorithm, in order to deal with the hybrid learning problem (unsupervised and supervised learning), they use the multi-elitist approach to help the learning algorithm to get out of local minimum, therefore improving the accuracy of the proposed learning model. Das and al. [12] proposed a method based on a modified version of classical Particle Swarm Optimization algorithm, known as the Multi-Elitist Particle Swarm Optimization model. The proposed algorithm has been shown to meet or beat the other state of the art clustering algorithms in a statistically meaningful way over several benchmark datasets. The unique disadvantage of this algorithm is choosing the best suited parameters to find optimal solution. Gou and al. [13] apply Multi Elitist approach on quantum clustering problems. They use these methods to avoid getting stuck in local extremes. They used the mechanism of cluster center updating with a property of k-means clustering, that can influence the clustering results. This is one of disadvantages of this method, adding parameters that the method is based on. According to elitist strategy that work for problems in which one is required to find multiple optimal solutions, we introduce a new approach where multi evolutionary algorithms run together at the same time to compare their proposed solutions, and we select only the best one which is the optimal or nearest optimal solution.

3 Proposed Approach

We propose an Elitist Evolutionary Clustering Algorithm (EECA) (figure 1) that combine different techniques such as evolutionary algorithm with elitist approach and local search approach. This approach allows us to determine the optimal cluster number as well as finding cluster seeds.

3.1 Evolutionary Algorithm

In this section, we try to explain succinctly the evolutionary algorithm which is shown in right of the figure 1.

Gene Representation. A potential solution of our problem is a combination of potential optimal seeds, we try to find optimal number of these seeds. According to this, we consider a genetic individual (chromosome) as a combination of k_{max} potential optimal seeds, k_{max} being parameter of the algorithm. Each gene in genetic individual is an integer number, which takes values from $\{1, 2, .., n\}$. This value indicates the data point identification. The gene with value 0 indicates that there is no point selected as center. The number of genes in the optimal solution different from 0 represents the optimal cluster number. Each data point is a

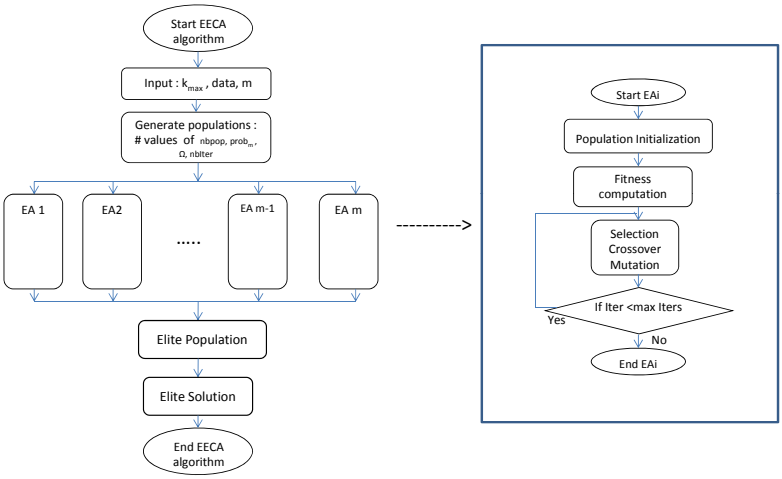

Fig. 1. EECA schema

$d - dimensions$ vector containing the d real values. We want to diversify the population, for this, we associate a frequency rate at each data point, each new genetic individual will be composed by the data point had not been used before, that have frequencies equal or close to zero.

Population Initialization. The population initialization is created by nb_{pop} genetic individuals with nb_{pop} given before. Each gene in the genetic individual is selected randomly on : $\{0, 1, 2, ..., n\}$ data set identification. Each genetic individual corresponds to a specific clustering solution in terms of cluster seeds. We impose in the genetic individual to have a different gene to obtain specific and different clusters seeds. We also impose to have an initial population without redundant genetic individuals. Studying individual is applied in each genetic individual which is created. And then, we verify that is no identical genetic individual, in the population which has the same gene. At this step, the first population is ready. Once the population is evaluated and sorted according to the fitness function we operate the genetic operator described bellow.

Genetic Operators. The crossover operation produces new offspring genetic individuals from parent individual. Two new genetic individuals are created by exchanging genes from two parent chromosomes. This exchange may start from one or several positions in the chromosomes called cut point. We can use two types of crossover: a randomly determined cut point or an optimized cut point. In the latter case, we determine the best point before the cut, this implies an evaluation of each possible cut for the individual. For our case, we first use a randomly chosen cut point which is modified to obtain optimized cut point, which implies evaluation of each new created individual. The first child resulting from

by repeated randomly crossover which is better than initial child will determine the optimal cut point. The mutation makes gene inversion in the genetic individual. This genetic operator is used to avoid the degeneration of the population in order to prevent a too fast convergence of the genetic algorithm. If implemented appropriately, the operator can make the algorithm able to leave from local optimum. The mutation is usually applied with a small probability ($prob_m$, algorithm parameter). The mutated gene is chosen according to neighborhood search (section 3.2), we should verify if a new gene value is not in neighborhood of other genes composing the genetic individual. It is evident that we study the composition of each creation of new individual to have a different gene in a specific genetic individual.

Evaluation Criteria. The first objective is to find the optimal number of clusters, where clusters are compact and separated between them. Several measures have been proposed, Milligan and Cooper [14] presented a survey and comparison of 30 internal validity indexes for clustering algorithms and out-perform that $CH(k)$ [15] is one of the best solutions.

This index represents a ratio of the sum of between-cluster and the sum of within-cluster, which is expressed as follows:

$$CH(k) := \frac{[traceB/(k-1)]}{[traceW/(n-k)]}$$

Where n is the number of data points, k is clusters number.

$$traceB := \sum_{i=1}^{k} |C_i| \ \|\overline{C_i} - \overline{x}\|^2$$

$$traceW := \sum_{i=1}^{k} \sum_{j \in C_i} \|x_j - \overline{C_i}\|^2$$

With $|C_i|$ is the number of assigned objects to the cluster C_i $(i = 1, \ldots, k)$; $\overline{C_i}$ is a center of class C_i and $\overline{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$ is the global center of all data points. The optimal clusters number is found by maximizing Calinski and Harabasz index $(CH - index)$.

The second objective is to find clusters that are compact and separate, so we introduce a measure that minimizes the overlapping between clusters, this criteria is defined by :

$$OP = \sum_{i \neq j}^{k} Card(C_i \cap C_j)$$

We compare our method with two fitness functions, first test series focus on maximizing $CH(k)$ in each EAi and in the elite population, where second test series focus on minimizing OP in each EAi and sort the elite population with $CH(k)$.

3.2 Neighborhood Search

When we operate the mutation we should firstly verify if all genes are different; secondly, according to the obtained potential clusters seeds, we introduce a new verification process where we impose that a new gene must not be in the neighborhood of others genes composing the genetic individual. For this, we introduce an automatic method to detect the limit of the cluster.

Cluster limit detection. In order to obtain the neighborhood seed, we search the data points contained in the cluster, we select the ones close enough to the seed. We choose the threshold Ω (algorithm parameter) by computing the distance between all the data points and the cluster seed and ordering them from the closest object to the farthest. We then try to find an abrupt increasing of distance that will indicate the cluster limit. We choose to use the peak detection method presented by Palshikar [16] applied on differential distances. Other cluster limit detection might be used, but this one is fast and gives the algorithm a complexity of $O(n \times d \times (p + g))$, where n is the number of data points, d the number of dimensions, p the population size and g the number of generations.

Then, we operate mutation by changing gene value by new value which is not in the neighborhood of other genes composing the genetic individual, and if we don't find this new gene, we change the value by 0 (no center selected).

3.3 Elite Population

Generally, the goal of the adaptive elitist-population search method is to adaptively adjust the population size according to the features of our technique to achieve, firstly, a single elitist individual searching for each solution; and secondly, all the individuals in the population searching for different solutions in parallel. For satisfying multi modal optimization search, we define the elitist individuals in the population as the individual with the best fitness on different solutions of the multiple domain. These elitist individuals define the elite population. Then we propose the elitist genetic operators that can maintain and even improve the diversity of the population and performing different evolutionary algorithms. A major advantage of using EAs over traditional learning algorithms is the ability to escape from local minimum using genetic operators [17]. The evolutionary algorithm depend on 4 parameters : nb_{pop}, $prob_m$, nb_{iter} and Ω. We perform as much as possible many evolutionary algorithms according to different values of these parameters and then we select the best solution, which compose elite population, without focusing on setting parameters. From this elite population we select the best solution (elite individual), which is the optimal solution for our problem.

4 Experimental Results

To evaluate the performance of the proposed method, we proceed several experiments on data sets from University of California at Irvine (UCI) machine

Table 1. Datasets description

Dataset	No. Points	No. Attributes	No. Clusters
Iris	150	4	3
Vehicule	846	18	4
Haberman	306	3	2
Synthetic	500	2	5
Wine	178	13	3
Blood Tranfusion	748	4	2
Seed	210	7	3
Ecoli	336	7	8

learning benchmark repository [18]. Data sets information are summarized in Table 1.

In our experiments, we perform an Elitist Algorithm with varying different parameters values in each Evolutionary Algorithm. We vary Evolutionary Algorithm parameters nb_{pop}, $prob_m$, nb_{iter} and Ω as follow :

- $nb_{pop} \in \{50, 100, 150\}$
- $prob_m \in \{0.1, 0.2, 0.3\}$
- $nb_{iter} \in \{200, 300, 500\}$
- $\Omega \in \{1, 2, 3\}$

For each data set we perform the elitist algorithm with 18 EAs and $k_{max} = 10$. The results of finding optimal k are illustrated in table 2. We compare our method with the two fitness functions presented before. First test series focus on maximizing $CH(k)$ in each EAi and in the elite population ($EACH$ corresponding column in table 2). The second test series focus on minimizing OP in each EAi and sort the elite population with $CH(k)$ ($EECA$ corresponding column in table 2).

Table 2. Results description

Dataset	k-real	$EACH$	$EECA$
Iris	3	2	3
Vehicule	4	2	4
Haberman	2	2	2
Synthetic	5	5	5
Wine	3	2	3
Blood Tranfusion	2	2	2
Seed	3	2	3
Ecoli	8	7	8

As we can see in the table 2, we find exactly the same number of clusters as in real data sets using overlapping fitness function combined with $CH(k)$ index. When only $CH(k)$ index is used, we always find $k = 2$, (except for 2

data sets, Ecoli and Synthetic data sets). This result can be explained by the formula of $CH(k)$ index, when we try to maximize only this index, we converge to small number of clusters. To find the data set partitions, we use a cluster limit detection method based on hyperspherical cluster forms. For Synthetic data set which presents prefect hyperspherical clusters, the two fitness functions performs perfectly. But in other data sets it seems important to consider the overlapping fitness in the first, and then to select the better solution according to the $CH(k)$ index. To confirm our results, we visualize some data sets using scatter plot methods [19] which represent all 2D projection of the data set. The figure 2 represents the projection of iris data set, and the figure 3 represents the projection of Haberman data set, as we can see in the figures the data points colors (different forms) represent the real clusters, and in red (square form) we can see the clusters seeds that are detected by our method. We can note that each of them corresponds to the real clusters, and can be considered as the center of the different clusters or as initial seed for any clustering algorithm.

These results and the visualizations showed the effectiveness of our methods on data sets that have compactness clusters structures. As we can see in the figure 4, that represent a synthetic data set, composed by five clusters, with Gaussian distribution. We can easily adapt our method with changing and adapting distance measures to extract other cluster structures. In fact, for this we need to

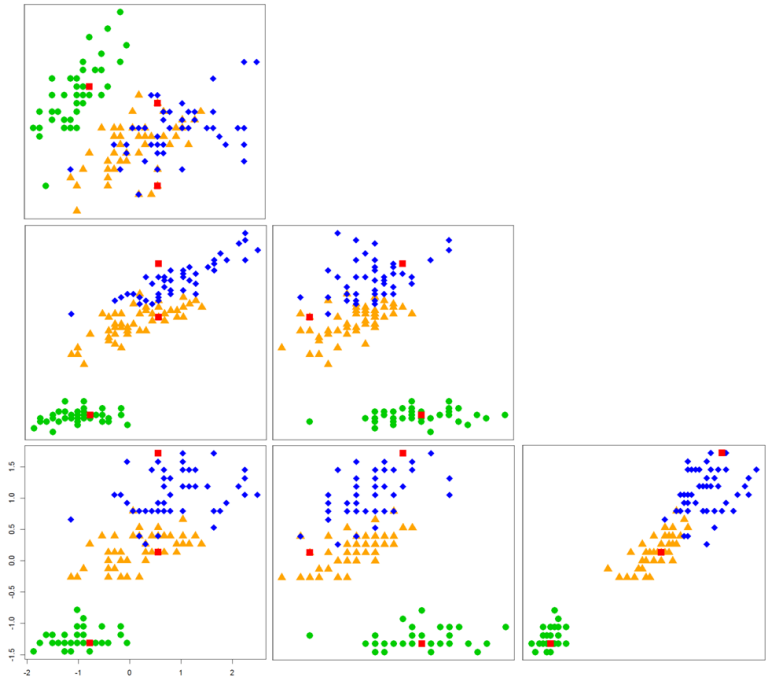

Fig. 2. Scatter plot visualization of Iris data set with detected seeds

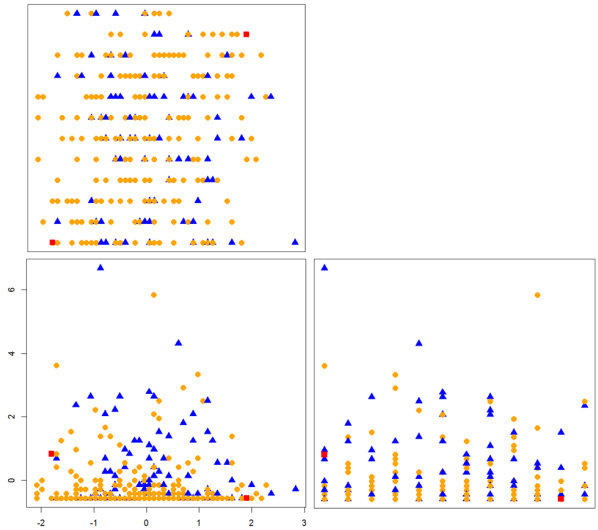

Fig. 3. Haberman data set with detected seeds

build a good benchmark data bases. We can also improve our method with allowing an overlapping degree between different extracted clusters. This approach allows us to apply our method on more different data sets structures that can be added on the benchmark.

5 Conclusion

This article proposes a new method that finds at the same time the optimal cluster number and proposes the initial clusters seeds without a prior knowledge of cluster number. We don't use any other clustering algorithm to evaluate our solution, so, our method can be used as standalone clustering algorithms dealing with hyperspherical clusters. Unlike other algorithms such as k-means, our approach do not need to fix a priori the clusters number. We propose a new mutation process using neighborhood search and we use, for this, an automatic cluster limit detection method. We also introduce a new combined fitness function to evaluate our solution. To deal with the problem of involving numerous parameters, we propose an Elitist Evolutionary Approach that involve numerous evolutionary algorithms (EAs). The difference between them is implemented through parameters and we select only the best concurrent solution. This proposition can also address the problem of exploration of large search space. The initial population of numerous evolutionary algorithms (EAs) is substantially different, so, we can deal with the large data sets in few number of iterations. First results are encouraging, as meaningful clusters seeds have been found from different data sets. Results showed the effectiveness of our methods on data sets that have compactness clusters structures. We can easily adapt our method with changing

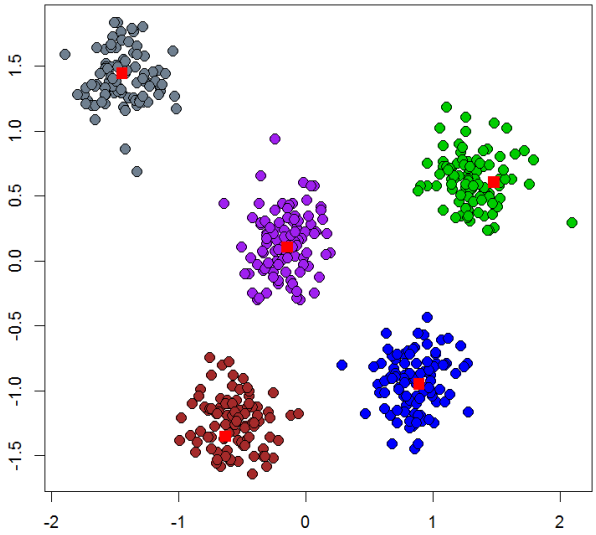

Fig. 4. Synthetic data set with detected seeds

and adapting distance measures to extract other cluster structure. Concerning further work, we plan to test our approach to different benchmark data bases to detect hyperellipsoidal clusters or other data sets structures and we think improving our methods with allowing an overlapping degree between different extracted clusters. This approach allows us to apply our method on more different data sets structures that can be added on the benchmark. We want also to apply our approach on different subspaces, we think that the optimal clustering can be different according to the subspace data projection. And then, our method can be applied on multiview clustering or on subspace clustering.

References

1. Jain, A.K.: Data clustering: 50 years beyond k-means. Pattern Recognition Letters 31(8), 651–666 (2010)
2. Mao, J., Jain, A.K.: A self-organizing network for hyperellipsoidal clustering (HEC). IEEE Transacations on Neural Networks 7(1), 16–29 (1996)
3. Celebi, M.E., Kingravi, H.A., Vela, P.A.: A comparative study of efficient initialization methods for the k-means clustering algorithm. Journal Expert Systems with Applications: An International Journal Archive 1(40), 200–210 (2013)
4. Babu, G.P., Murty, M.N.: Simulated annealing for selecting optimal initial seeds in the k-means algorithm. Indian Journal of Pure and Applied Mathematics 25(1-2), 85–94 (1994)
5. Babu, G.P., Murty, M.N.: A near-optimal initial seed value selection in k-means algorithm using a genetic algorithm. Pattern Recognition Letters 14(10), 763–769 (1993)

6. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: A review. ACM Computing Surveys 31(3), 264–323 (1999)
7. Linde, Y., Buzo, A., Gray, R.: An algorithm for vector quantizer design. IEEE Transactions on Communications 28(1), 84–95 (1980)
8. Huang, C.M., Harris, R.W.: A comparison of several vector quantization codebook generation approaches. IEEE Transactions on Image Processing 2(1), 108–112 (1993)
9. Aloise, D., Hansen, P., Liberti, L.: An improved column generation algorithm for minimum sum-of-squares clustering. Mathematical Programming, 1–26 (2010)
10. Sarma, J., De, J.: Generation gap methods. In: Handbook of Evolutionary Computation, vol. 2(7), pp. 1–5 (1997)
11. Qasem, S.N., Shamsuddin, S.M.: Memetic Elitist Pareto Differential Evolution algorithm based Radial Basis Function Networks for classification problems. Original Research Article Applied Soft Computing 8(11), 5565–5581 (2011)
12. Das, S., Abraham, A., Konar, A.: Automatic kernel clustering with a Multi-Elitist Particle Swarm Optimization Algorithm. Pattern Recognition Letters 5(29), 688–699 (2008)
13. Gou, S., Zhuang, X., Li, Y., Xu, C., Jiao, L.C.: Multi-elitist immune clonal quantum clustering algorithm. Neurocomputing 101(4), 275–289 (2013)
14. Milligan, G., Cooper, M.: An examination of procedures for determining the number of clusters in a data set. Psychometrika 50, 159–179 (1985)
15. Calinski, T., Harabasz, J.: A dendrite method for cluster analysis. Communications in Statistics Simulation and Computation 3(1), 1–27 (1974)
16. Palshikar, G.: Simple algorithms for peak detection in time-series. In: Proceedings of 1st International Conference on Advanced Data Analysis Business Analytics and Intelligence (2009)
17. Radcliffe, N.J.: Equivalence class analysis and presentation of strong rules. In: Knowledge Discovery in Database, vol. 11, pp. 229–248 (1991)
18. Blake, C.L., Merz, C.J.: UCI repository of machine learning databses. University of California, Irvine, Dept. of Information and Computer Sciences (1998), `http://archive.ics.uci.edu/ml/datasets.html` (accessed on May 2013)
19. Carr, D.B., Littlefield, R.J., Nicholson, W.L.: Scatter-plot matrix techniques for large N. Journal of the American Statistical Association 82(398), 424–436 (1987)