Special issue of the 2nd International Conference on Computational and Experimental Science and Engineering (ICCESEN 2015)

HK-Means: A Heuristic Approach to Initialize and Estimate the Number of Clusters in Biological Data

D. Reddy Edla*, V. Gondlekar and V. Gauns
National Institute of Technology Goa, Farmagudi, Goa, India

K-means algorithm is one of the simplest and fastest clustering algorithms existing since more than four decades. One of the limitations of this algorithm is estimating number of clusters in advance. This algorithm also suffers from random initialization problem. This paper proposes a heuristic which initializes the cluster centers and estimates the number of clusters as a discrete value. The method estimates the number of clusters and initializes many cluster centers successfully for the clusters that are dense and separated significantly. The method selects a new cluster center in each iteration. The point selected is the point which is most dissimilar from the previously chosen points. The proposed algorithm is experimented on various synthetic data and the results are encouraging.

DOI: 10.12693/APhysPolA.130.78

PACS/topics: 07.05.Tp, 29.85.Fj, 93.85.Bc

1. Introduction

Clustering [1] is defined as a process of grouping a set of physical or abstract objects into classes of similar objects. Clustering is a main task of exploratory data mining and is used for data analysis in variety of applications such as machine learning [2], image analysis [3], pattern recognition [4] and medicine [5]. Clustering methods can broadly be divided into partitional, hierarchical, distribution based, grid based and density based methods. Among these, partitional clustering techniques have been very popular due to their simplicity and low time complexity. K-means is a partitional based unsupervised learning algorithm that has been developed four decades ago.

The main idea behind K-means is to select k cluster centers at first and then partition N observations into k clusters such that each observation belongs to the closest cluster center. Cluster is a group of similar objects. Given a set of X observations where each observation is d-dimensional real vector, K-means aims to partition these X observations into K sets. Algorithm tries to minimize within the cluster sum of squares. It has been shown that the problem of minimizing this sum is NP-Hard [6] even for 2 clusters and so it is for any given K clusters even in 2D Euclidean space. K-means attempts to find a local minima for sum, and hence requires that K to be known in advance, the global minima being cost $= 0$ when $K = N$. Hence greedy approaches such as K-means are used to minimize the Euclidean sum of squares. K-means may terminate at local optimum and we cannot guarantee global optimum.

Specifying number of clusters in advance can be seen as huge drawback of K-means. This drawback however can be overcome by running k-means over a range of K values and then choosing the k which gives the best result. K-means is not suitable for convex shape clusters and clusters with very different size and is very sensitive to noise. Another problem is efficiently selecting clusters centers at start. More efficient selection of centers will lead to the faster result. This paper proposes a new technique to estimate number of clusters i.e. K value. The method gives best result for separated clusters than other types of clusters. K-means is applied partially on data set to define intermediate clusters. Procedure to check validity of these clusters is derived from silhouette coefficient [7].

2. Literature survey

K-means is the most popular clustering technique of this model developed by MacQueen [8] in 1967. However, it is sensitive to the random selection of initial cluster centers. In addition to that, a prior knowledge of the number of clusters is necessity to input to K-means. Many researches proposed various methods [10, 11] to overcome these problems.

Kanungo et al. [12] proposed a novel initialization method for K-means using kd-tree. This scheme does not pass information from one stage to its next. Du et al. [13] developed an initialization scheme for K-means clustering called PK-means to cluster the gene expression data. The convergence rate of this technique is fast and the computational load is less. A novel clustering algorithm called modified filtering algorithm (MFA) has been proposed in [14]. It is the improvement of the algorithm in [12]. A fast K-means clustering algorithm named FKMCUCD was proposed in [15] using cluster center displacement. This method is significant for high-dimensional large data. Zalik [16] proposed an efficient algorithm named K'-means to enhance the K-means algorithm by exploiting a cost function. This scheme fails when the clusters are of various shapes such as elliptical. Redmond et al. [17] proposed a novel seed selection algorithm using kd-tree [9]. This scheme is unable to deal with the noise. Cao et al. [18] proposed an algorithm

corresponding author; e-mail: dr.reddy@nitgoa.ac.in by defining the cohesion degree of the neighborhood of a given point and the coupling degree between neighborhoods of the points. This algorithm has quadratic time complexity. Khan et al. [19] designed an algorithm called CCIA. This method first develops $K'(> K)$ cluster centers from which the desired k centers are chosen. Lu et al. [20] contributed with a hierarchical initialization approach in which the clustering problem has treated as a weighted clustering problem. A genetic clustering algorithm named GAGR [21] has been proposed to cluster the genome data using K-means. It uses the genetic algorithm with gene rearrangement process. Ahmad et al. [22] proposed an enhanced K-means clustering algorithm for mixed numeric and categorical data based on co-occurrence of the values. An algorithm called KGA [23] was proposed using the genetic algorithm. This method may not produce fine results whenever the number of clusters is unknown. An improved version of K-means called K-means has been developed in [24]. It is unable to deal with the noisy data. Likas et al. [25] proposed a global K-means clustering algorithm in which the clusters are formed using a global search procedure. A recursive method is proposed by Duda and Hart [26]. Milligan [27] developed an enhanced algorithm based on Ward's hierarchical method [28] that helps in finding the initial cluster centers. The algorithm proposed by Fisher [29] generates good seeds by constructing initial hierarchical clustering based on [30] method using MaxMin algorithm to choose a subset of the original database as initial cluster centers. Bradley et al. [31] formed the initial clusters based on the bilinear program. Tou and Gonzales [32] presented a method which entirely depends on the order of the points and the threshold value. Linde et al. [33] proposed a method based on binary splitting (BS). Here, the clusters quality depends on the selection of a random vector. Kaufman and Rousseeuw [34] developed a method based on the reduction in the distortion. Babu and Murty [35] proposed a technique for the near optimal seed selection based on genetic programming. This is not robust for large data bases. Huang and Harris [36] projected a method called direct search binary splitting (DSBS) based on the principal component analysis (PCA) and the vector of Linde et al. [33]. Thiesson et al. [37] designed an algorithm that depends on the mean value of the given data. Bradley and Fayyad [38] proposed an initialization approach for K-means using the Forgy method [39].

3. Proposed HK-means algorithm

The algorithm works well for separated clusters. Separation means that distance of any cluster center from any other point in that cluster should be less than distance of that cluster center from any other cluster center. R is specified as input and represents the degree after which newly selected object should be checked if it lies in previously selected cluster. $R = 2$ will run Algorithm 1 (see Appendix) for almost all the iterations while $R = 1$ will not run Algorithm 1 for any selected object. The first cluster center is chosen at a random. At any given instance new cluster center is a object whose minimum distance from the previously chosen centers divided by the average distance between cluster centers is the maximum. Clearly this maximum value will be larger for valid object i.e. object that does not lie in previously chosen clusters and low for invalid object i.e. object that lies in previously chosen cluster. For separated clusters it is guaranteed that newly chosen cluster center does not lie in previously chosen clusters. Cluster centers chosen by this method will always lie on boundary of clusters, and thus to refine them, single iteration of k-means is applied on data which results in partial formation of clusters which may not be the accurate ones. Also cluster centers are recomputed so that they lie inside the clusters. Verification process to check if two objects lie in same cluster is as follows. Algorithm 1 is run over the m selected points and value of si is found for cluster cluster$_m$. si value indicates how nicely any point in cluster$_m$ is allocated to that particular cluster. In process of calculating si cluster which is most similar to cluster$_m$ is found and is indicated by clusters. In second step, as illustrated in Algorithm 2, cluster$_m$ and clusters are merged temporarily and centers are recomputed. Again Algorithm 1 is run again over new set of centers and clusters and $si1$ is calculated.

4. Experimental results

The proposed algorithm has been executed on MATLAB R2008a on Windows 7 Home Premium 64-bit running on Intel(R) Core(TM) i5 2410M CPU @ 2.30 GHz, with 6 GB RAM.

4.1. Synthetic data

The algorithm was applied on various two dimensional synthetic data sets with $R = 1.3$ and some of the results are given below. Following are the experimental

TABLE I

si and $si1$ values as the result of 4 or 7 iterations.

min	*si*	*si*1
2	–	–
2.1107	–	–
1.5868	–	–
1.0302	0.9461	0.9871

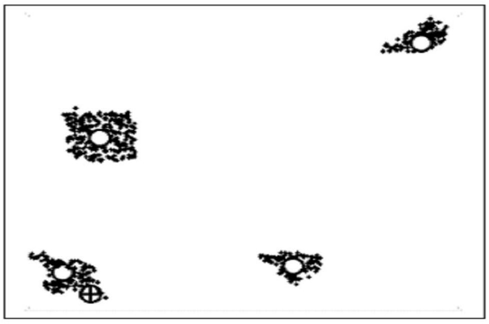


TABLE II

si and *si*2 values used in third case.

min	*si*	*si*2
2	–	–
1.9806	–	–
1.3743	–	–
1.1816	0.9708	0.9284
1.0650	0.9058	0.9795

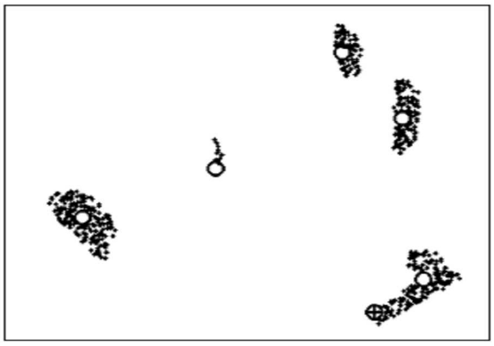
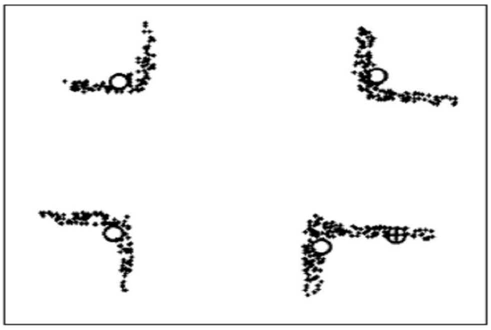


TABLE III

si and *si*1 values used in fourth case.

min	*si*	*si*1
2	–	–
3.6115	–	–
2.0598	–	–
1.1875	0.9421	0.9631



results when the proposed algorithm was applied on four data sets. These results included data plots of the 2-dimensional data objects along with the table indicating *si* and *si*1 values. *si* and *si1* are calculated only when *min* is less than R. Algorithm stops when *si* value is less than *si1* value.

In general, valid object is represented by empty circle and rejected object by a circle with cross. In case of Table I algorithm terminates after 4th or 7th iteration and object marked by circle with cross is rejected. First 7 objects are added to ClusterPoints and 8th object is rejected. Tables II and III can be similarly interpreted.

4.2. Biological data

We have experimented the proposed algorithm on various biological data sets taken from UCI machine learning repository [40]. The experimental results are shown in Table IV. As depicted in the table, the proposed algorithm has succeeded to initialize and estimate the appropriate number of clusters. The estimated number of clusters produced by the proposed algorithm for the biological datasets are also validated against K-means clustering algorithm using dynamic validity index (DVI) [41] defined as follows.

Let N be the number of data points, K be the pre-defined upper bound number of clusters, and z_i be the centre of the cluster C_i. The dynamic validity index is given by

$$\mathrm{DVI} = \min_{k=1,2,\ldots,K}\{\mathrm{IntraRatio}(k) + \gamma^{*}\mathrm{InterRatio}(k)\}, \tag{1}$$

$\mathrm{IntraRatio}(k) = \frac{\mathrm{Intra}(k)}{\mathrm{MaxIntra}}$, $\mathrm{InterRatio}(k) = \frac{\mathrm{Inter}(k)}{\mathrm{MaxInter}}$, $\mathrm{Intra}(k) = \frac{1}{N}\sum_{i=1}^{k}\sum_{x-C_i}\|x - z_i\|^2$, $\mathrm{MaxIntra} = \max_{i=1,2,\ldots,k}(\mathrm{Intra}(i))$, and $\mathrm{MaxInter} = \max_{i=1,2,\ldots,k}(\mathrm{Inter}(i))$.

Initially, Algorithm 1 is applied over the selected points and value of *si* is found for cluster. *si* value attracts compactness within the clustrer. Then, Algorithm 2 is applied and clusters are merged temporarily and centers are recomputed. Again Algorithm 1 is run over different set of centers and clusters and si1 is computed. Here, IntraRatio represents the overall compactness of clusters and InterRatio represents overall separation of the clusters. The term Intra is average distance of all the points within a cluster from cluster centre. Then the Inter term is composed of two parts, both of them based on cluster centers. The value of Inter increases with the increment in k. The less value of DVI indicates more quality of the clusters and vice versa. The value of γ in Eq. (1) represents the modulating parameter to balance the noisy data points.

TABLE IV

Number of attributes (1), data size (2), actual (3) and estimated (4) number of clusters and comparison of K-means (5) with the our proposed (6) scheme using dynamic validity index (DVI).

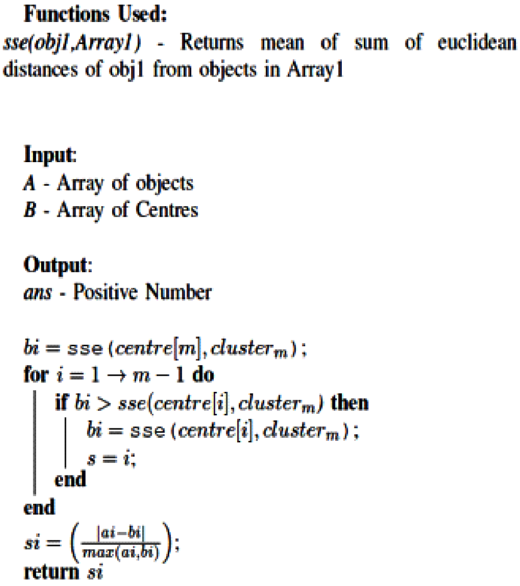
Name	(1)	(2)	(3)	(4)	(5)	(6)
Iris	4	150	3	3	1.1155	0.5324
Wine	13	178	3	4	0.1362	0.0923
St. Heart	13	270	2	2	0.2825	0.2011
Br. Tissue	9	106	2	2	0.4322	0.1008
P.I. Diabetes	8	768	2	2	0.7134	0.0645
Cloud	10	1024	2	3	2.8754	0.8637
B. Transfusion	5	748	2	2	0.7747	0.3189
Yeast	8	1484	10	10	1.6543	1.0067

5. Conclusion

We have proposed a heuristic approach for estimating the number of clusters in the given data set and finding initial set of clusters centers in order to apply K-means algorithm, provided that clusters are separated enough. The main significance of the algorithm is that it runs in linear time. However, real world data is not always separated, future efforts can be made to extend this algorithm to random data by estimating K as a range. In proposed algorithm, it is not guaranteed that the algorithm will halt for all the cases. Future efforts can be put in designing algorithm that can be proved to halt for all cases. Also, future endeavors will be made to give K value as a range when separation criterion is not satisfied. Efforts can also be put to eliminate threshold value R and reducing the number of times Algorithm 1 is run so as to make it faster. It has been shown that the proposed algorithm estimates the number of clusters in most of the biological datasets considered for experimentation. The results of the proposed algorithm were evaluated in terms of the dynamic validity index. The results have shown the effectiveness of the algorithm.

Appendix

Algorithm 1: Compute algorithm



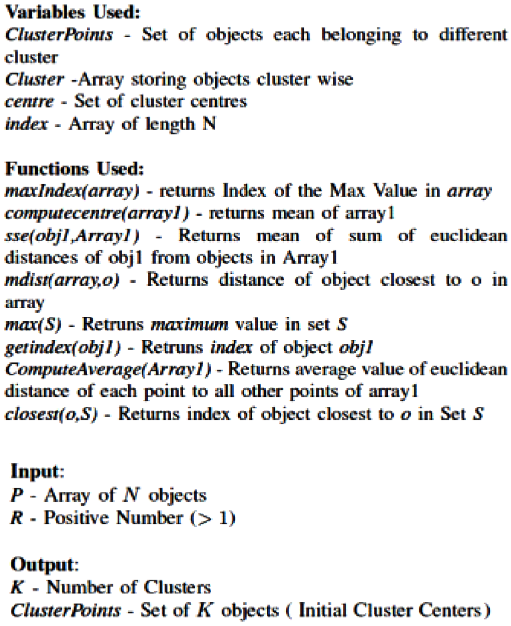
Functions Used:
sse(obj1,Array1) - Returns mean of sum of euclidean distances of obj1 from objects in Array1

Input:
A - Array of objects
B - Array of Centres

Output:
ans - Positive Number

```
$bi = \texttt{sse}\,(centre[m], cluster_m)$;
for $i = 1 \to m-1$ do
    if $bi > sse(centre[i], cluster_m)$ then
        $bi = \texttt{sse}\,(centre[i], cluster_m)$;
        $s = i$;
    end
end
$si = \left(\frac{|ai-bi|}{max(ai,bi)}\right)$;
return $si$
```

Algorithm 2: Partial algorithm

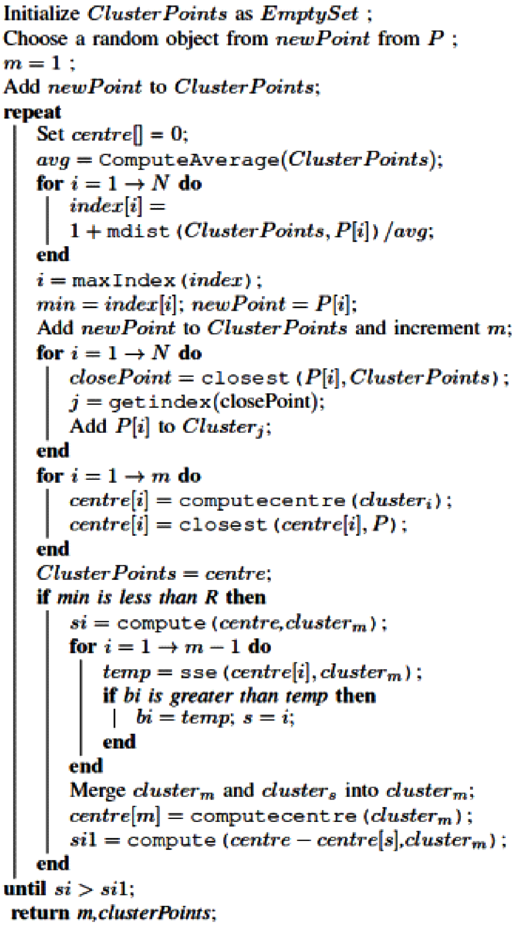


Variables Used:
ClusterPoints - Set of objects each belonging to different cluster
Cluster -Array storing objects cluster wise
centre - Set of cluster centres
index - Array of length N

Functions Used:
maxIndex(array) - returns Index of the Max Value in *array*
computecentre(array1) - returns mean of array1
sse(obj1,Array1) - Returns mean of sum of euclidean distances of obj1 from objects in Array1
mdist(array,o) - Returns distance of object closest to o in array
max(S) - Retruns *maximum* value in set *S*
getindex(obj1) - Retruns *index* of object *obj1*
ComputeAverage(Array1) - Returns average value of euclidean distance of each point to all other points of array1
closest(o,S) - Returns index of object closest to *o* in Set *S*

Input:
P - Array of *N* objects
R - Positive Number (> 1)

Output:
K - Number of Clusters
ClusterPoints - Set of *K* objects (Initial Cluster Centers)



```
Initialize $ClusterPoints$ as $EmptySet$ ;
Choose a random object from $newPoint$ from $P$ ;
$m = 1$ ;
Add $newPoint$ to $ClusterPoints$;
repeat
    Set $centre[] = 0$;
    $avg = \texttt{ComputeAverage}(ClusterPoints)$;
    for $i = 1 \to N$ do
        $index[i] =$
        $1 + \texttt{mdist}\,(ClusterPoints, P[i])\,/avg$;
    end
    $i = \texttt{maxIndex}\,(index)$;
    $min = index[i]$; $newPoint = P[i]$;
    Add $newPoint$ to $ClusterPoints$ and increment $m$;
    for $i = 1 \to N$ do
        $closePoint = \texttt{closest}\,(P[i], ClusterPoints)$;
        $j = \texttt{getindex}(\text{closePoint})$;
        Add $P[i]$ to $Cluster_j$;
    end
    for $i = 1 \to m$ do
        $centre[i] = \texttt{computecentre}\,(cluster_i)$;
        $centre[i] = \texttt{closest}\,(centre[i], P)$;
    end
    $ClusterPoints = centre$;
    if $min$ is less than $R$ then
        $si = \texttt{compute}\,(centre, cluster_m)$;
        for $i = 1 \to m-1$ do
            $temp = \texttt{sse}\,(centre[i], cluster_m)$;
            if $bi$ is greater than $temp$ then
                $bi = temp$; $s = i$;
            end
        end
        Merge $cluster_m$ and $cluster_s$ into $cluster_m$;
        $centre[m] = \texttt{computecentre}\,(cluster_m)$;
        $si1 = \texttt{compute}\,(centre - centre[s], cluster_m)$;
    end
until $si > si1$;
return $m, clusterPoints$;
```

References

[1] J. Han, M. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed., Morgan Kaufmann Publ., Waltham, USA 2006.

[2] M. Castelnovi, P. Musso, A. Sgorbissa, R. Zaccaria, in: *Proc. IEEE Int. Symp. on Computational Intelligence in Robotics and Automation*, Vol. 1, 2003, p. 229.

[3] P.K. Chang, Wen Chen, Jiebo Luo, *IEEE Trans. Image Process.* **7**, 1673 (1998).

[4] A. Jain, R. Duin, J. Mao, *IEEE Trans. Pattern Anal. Machine Intellig.* **22**, 4 (2000).

[5] N. Srinivasan, V. Vaidehi, in: *Proc. BroadNets 2005, 2nd Int. Conf. on Broadband Networks*, Vol. 2, 2005, p. 1007.

[6] D. Aloise, A. Deshpande, P. Hansen, P. Popat, *Machine Learning* **75**, 245 (2009).

[7] J. Rousseeuw, *J. Computat. Appl. Math.* **20**, 53 (1987).

[8] J.B. MacQueen, in: *5th Berkeley Symp. on Mathematical Statistics and Probability*, Vol. 1, University of California Press, Berkeley, CA 1967, p. 281.

[9] M.D. Berg, O. Cheong, M.V. Kreveld, M. Overmars, *Computational Geometry: Algorithms and Applications*, 3rd ed., Springer-Verlag, Berlin 2008.

[10] L. Galluccio, O. Michel, P. Comon, A.O. Hero, *Sign. Process.* **92**, 1970 (2012).

[11] A.K. Jain, *Pattern Recogn. Lett.* **31**, 651 (2010).

[12] T. Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, A.Y. Wu, *IEEE Trans. Pattern Anal. Machine Intellig.* **24**, 881 (2002).

[13] Z. Du, Y. Wang, Z. Ji, *Computat. Biol. Chem.* **32**, 243 (2008).

[14] J.Z.C. Lai, Y.C. Liaw, *Pattern Recogn.* **41**, 3677 (2008).

[15] J.Z.C. Lai, T.J. Huang, Y.C. Liaw, *Pattern Recogn.* **42**, 2551 (2009).

[16] K.R. Zalik, *Pattern Recogn. Lett.* **29**, 1385 (2008).

[17] S.J. Redmond, C. Heneghan, *Pattern Recogn. Lett.* **28**, 965 (2007).

[18] F. Cao, J. Liang, G. Jiang, *Comput. Math. Appl.* **58**, 474 (2009).

[19] S.S. Khan, A. Ahmad, *Pattern Recogn. Lett.* **25**, 1293 (2004).

[20] J.F. Lu, J.B. Tang, Z.M. Tang, J.Y. Yang, *Pattern Recogn. Lett.* **29**, 787 (2008).

[21] D.X. Chang, X.D. Zhang, C.W. Zheng, *Pattern Recogn.* **42**, 1210 (2009).

[22] A. Ahmad, L. Dey, *Data Knowledge Eng.* **63**, 503 (2007).

[23] S. Bandyopadhyay, U. Maulik, *Inform. Sci.* **146**, 221 (2002).

[24] Y.M. Cheung, *Pattern Recogn. Lett.* **24**, 2883 (2003).

[25] A. Likas, N. Vlassis, J.J. Verbeek, *Pattern Recogn.* **36**, 451 (2003).

[26] R.O. Duda, P.E. Hart, *Pattern Classification and Scene Analysis*, Wiley, New York 1973.

[27] G.W. Milligan, *Psychometrika* **45**, 325 (1980).

[28] J.H. Ward, Jr., *J. Am. Statist. Assoc.* **58**, 236 (1963).

[29] D. Fisher, *J. Artif. Intellig. Res.* **4**, 147 (1996).

[30] D.H. Fisher, *Machine Learn.* **2**, 139 (1987).

[31] P.S. Bradley, O.L. Mangasarian, W.N. Street, in: *10th Annual Conf. on Advances in Neural Information Processing System, USA*, 1996, Vol. 9, p. 368.

[32] J. Tou, R. Gonzales, *Pattern Recognition Principles*, Addison Wesley, Massachusetts 1974.

[33] Y. Linde, A. Buzo, R.M. Gray, *IEEE Trans. Commun.* **28**, 84 (1980).

[34] L. Kaufman, P.J. Rousseeuw, *Finding Groups in Data — An Introduction to Cluster Analysis*, Wiley, Canada 1990.

[35] G.P. Babu, M.N. Murty, *Pattern Recogn. Lett.* **14**, 763 (1993).

[36] C. Huang, R. Harris, *IEEE Trans. Image Process* **2**, 108 (1993).

[37] B. Thiesson, B. Meck, C. Chickering, D. Heckerman, *Microsoft Technical Report* (MSR-TR-97-30), 1997.

[38] P.S. Bradley, U.M. Fayyad, in: *15th Int. Conf. on Machine Learning (ICML-1998), Wisconsin (USA)*, 1998, p. 91.

[39] E. Forgy, *Biometrics* **21**, 768 (1965).

[40] UCI Machine Learning Repository, `archive.ics.uci.edu/ml/datasets.html`.

[41] J. Shen, S.I. Chang, E.S. Lee, Y. Deng, S.J. Brown, *Appl. Math. Comput.* **169**, 1172 (2005).