A K-means-like Algorithm for K-medoids Clustering and Its Performance

Hae-Sang Park*, Jong-Seok Lee and Chi-Hyuck Jun
Department of Industrial and Management Engineering, POSTECH
San 31 Hyoja-dong, Pohang 790-784, S. Korea
shoo359@postech.ac.kr, jongseok@postech.ac.kr, chjun@postech.ac.kr

Abstract

Clustering analysis is a descriptive task that seeks to identify homogeneous groups of objects based on the values of their attributes. This paper proposes a new algorithm for K-medoids clustering which runs like the K-means algorithm and tests several methods for selecting initial medoids. The proposed algorithm calculates the distance matrix once and uses it for finding new medoids at every iterative step. We evaluate the proposed algorithm using real and artificial data and compare with the results of other algorithms. The proposed algorithm takes the reduced time in computation with comparable performance as compared to the Partitioning Around Medoids.

KEY WORDS : Clustering, K-medoids, K-means

1. Introduction

Clustering is the process of grouping a set of objects into classes or clusters so that objects within a cluster have similarity in comparison to one another, but are dissimilar to objects in other clusters (Han et al 2001). K-means clustering (MacQueen, 1967) and Partitioning Around Medoids (PAM) (Kaufman and Rousseeuw, 1990) are well known techniques for performing non-hierarchical clustering.

K-means clustering finds the k centroids, where the coordinate of each centroid is the means of the coordinates of the objects in the cluster and assigns every object to the nearest centroid. The algorithm can be summarized as follows.
Step 1 : Select k objects randomly. These objects represent initial group centroids.
Step 2 : Assign each object to the group that has the closest centroid.
Step 3 : When all objects have been assigned, recalculate the positions of the k centroids.
Step 4 : Repeat Steps 2 and 3 until the centroids no longer move.

Unfortunately, K-means clustering is sensitive to the outliers and a set of objects closest to a centroid may be empty, in which case centroids cannot be updated. For this reason, K-medoids clustering are sometimes used, where representative objects called medoids are considered instead of centroids. Because it uses the most centrally located object in a cluster, it is less sensitive to outliers compared with the K-means clustering. Among many algorithms for K-medoids clustering, Partitioning Around Medoids (PAM) proposed by Kaufman and Rousseeuw (1990) is known to be most powerful. However, PAM also has a drawback that it works inefficiently for large data sets due to its complexity (Han et al, 2001). This is main motivation of this paper. We are interested in developing a new K-medoids clustering method that should be fast and efficient.

The remaining parts of this paper are organized as follows: The proposed method is introduced in the next section and performance comparison is presented with some simulation results. Other methods to find initial medoids are discussed and finally some conclusions are given.

2. Proposed K-medoids algorithm

Suppose that we have n objects having p variables that will be classified into k ($k < n$) clusters (Assume that k is given). Let us define j-th variable of object i as X_{ij} ($i = 1,...,n;\ j = 1,...,p$). The proposed algorithm is composed of the following three steps.

Step 1 : (Select initial medoids)

1-1. Using Euclidean distance as a dissimilarity measure, compute the distance between every pair of all objects as follows:

$$d_{ij} = \sqrt{\sum_{a=1}^{p}(X_{ia} - X_{ja})^2} \quad i = 1,...,n;\ j = 1,...,n \tag{1}$$

1-2. Calculate p_{ij} to make an initial guess at the centers of the clusters.

$$p_{ij} = \frac{d_{ij}}{\sum_{l=1}^{n} d_{il}} \quad i = 1,...,n;\ j = 1,...,n \tag{2}$$

1-3. Calculate $\sum_{i=1}^{n} p_{ij}$ $(j = 1,...,n)$ at each objects and sort them in ascending order. Select k objects having the minimum value as initial group medoids.

1-4. Assign each object to the nearest medoid.

1-5. Calculate the current optimal value, the sum of distance from all objects to their medoids.

Step 2 : (Find new medoids)

Replace the current medoid in each cluster by the object which minimizes the total distance to other objects in its cluster.

Step 3 : (New assignment)
3-1. Assign each object to the nearest new medoid.
3-2. Calculate new optimal value, the sum of distance from all objects to their new medoids. If the optimal value is equal to the previous one, then stop the algorithm. Otherwise, go back to the Step 2.

The above algorithm runs just like K-means clustering and so this will be called as ‘K-means-like’ algorithm. In Step 1, we proposed a method of choosing the initial medoids. The performance of the algorithm may vary according to the method of selecting the initial medoids. The followings may be other possibilities of choosing the initial medoids, whose performance will be compared with each other in our simulation study in Section 3.

Method 1. Random selection
Select k objects randomly from all objects.

Method 2. Systematic selection
Sort all objects in the order of values of the chosen variable (first variable will be used in this study). Divide the range of the above values into k equal intervals and select one object randomly from each interval.

Method 3. Sampling
Take 10% random sampling from all objects and perform a preliminary clustering phase on these sampled objects using the proposed algorithm. The clustering result is used as the initial medoids.

Method 4. Outmost objects
Select k objects which are furthest from the center.

Method 5. Gaussian mixture
Assuming that the objects are derived from k Gaussian components, estimate each mean vector of k Gaussian models through Expectation-Maximization (EM) algorithm (Vlassis and Likas, 2002) and find the closest object to the estimated mean vector.

3. Numerical experiments

3.1 Artificial data

In order to evaluate the performance of the proposed method, some artificial data will be generated and clustered by using the proposed method, K-means clustering and PAM.

We generate 120 objects having 2 variables for each of three classes shown in Fig. 1. We call the first group marked by square as class A, the second group marked by circle as class B and third group marked by triangle as class C for the sake of convenience.

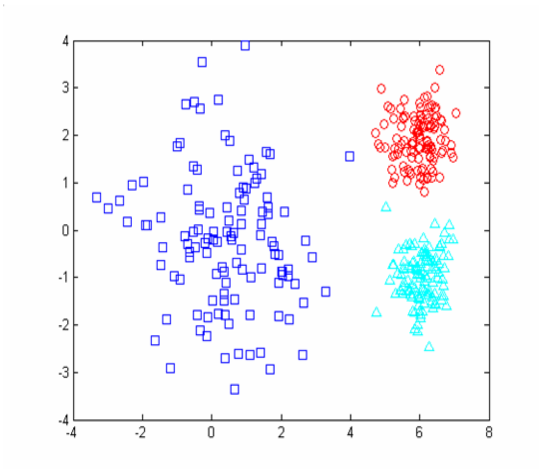


Figure 1 Artificial Data for Comparison

Data is generated from multivariate normal distribution, whose mean vector and variance of each variable (variance of each variable is assumed to be equal and covariance is zero) are given in Table 1. In order to compare the performance when some outliers are present among objects, we add outliers to the class B. The outliers are generated from a multivariate normal distribution which has the equal mean with class B but larger variance as shown in Table 1.

Table 1. Mean and variance when generating objects

	Class A	Class B	Class C	Outliers (Class B)
Mean vector	$(0\,,0)$	$(6\,,2)$	$(6\,,-1)$	$(6\,,2)$
Variance of each variable	1.5^2	0.5^2	0.5^2	2^2

We compare the performance of the proposed method with K-means clustering and PAM. The adjusted Rand index will be used as the performance measure, which proposed by Hubert and Arabie (1985) and is popularly used for comparison of clustering results. The adjusted Rand index is calculated as

$$RI_{adj} = \frac{2(ad-bc)}{(a+b)(b+d)+(a+c)(c+d)} \tag{3}$$

where

a = number of pairs which are in the identical cluster of compared clustering solution for pairs of objects in certain cluster of correct clustering solution

b = number of pairs which are not in the identical cluster of compared clustering solution for pairs of objects in certain cluster of correct clustering solution

c = number of pairs which are not in the identical cluster of correct clustering solution for pairs of objects in certain cluster of compared clustering solution

d = number of pairs which are not in the identical cluster of both correct clustering solution and compared clustering solution.

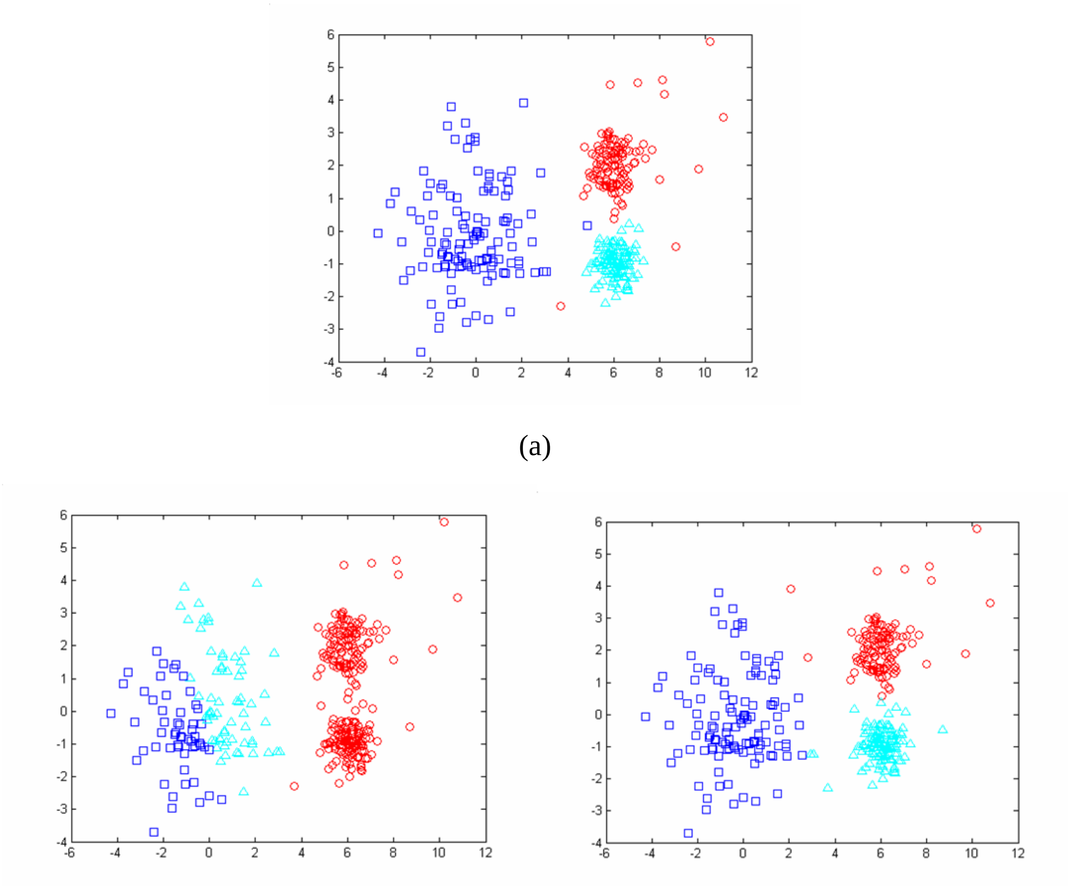
Performance of each method in terms of the adjusted Rand index is reported in Table 2. Here, outlier % means the proportion of outliers (in class B) among 120 objects. For example, when the outliers % is 10, 108 objects plus 12 outlier objects belonging class B will be generated while 120 objects for each of class A and class C will be generated. The result in Table 2 is actually the average adjusted Rand index from 100 repetitions.

Table 2. Adjusted Rand indices by various clustering methods

outliers %	K-means	PAM	Proposed method
0 %	0.7903	0.9679	0.9629
5 %	0.8376	0.9534	0.9335
10 %	0.7836	0.9430	0.9430
15 %	0.7957	0.9288	0.9189
20 %	0.7305	0.9150	0.9115
25 %	0.7708	0.9053	0.8904
30 %	0.7750	0.8952	0.8915
35 %	0.7595	0.8782	0.8609
40 %	0.7624	0.8667	0.8671

From Table 2, it can be clearly seen that PAM and the proposed method perform much better than K-means clustering. The performance of the proposed method and PAM is very similar to each other, although it seems to be degraded as the proportion of outliers increase.

Fig. 2 shows the one of the simulation results. Instead of partitioning class B and C, K-means clustering divide class A into two groups. This may be caused by K-means clustering’s weakness, which is sensitive to outliers.



(a)

(b)

(c)

Figure 2 (a) True cluster solution (b) Cluster result from K-means

(c) Cluster result from PAM and the proposed method

To compare the proposed method with PAM, we calculated the computation time with the artificial data sets. Fig. 3 shows how the computation time of each method increases as the number of objects increases. It is seen that PAM requires increasing computation time according to the number of objects, whereas the proposed method takes about the constant time.

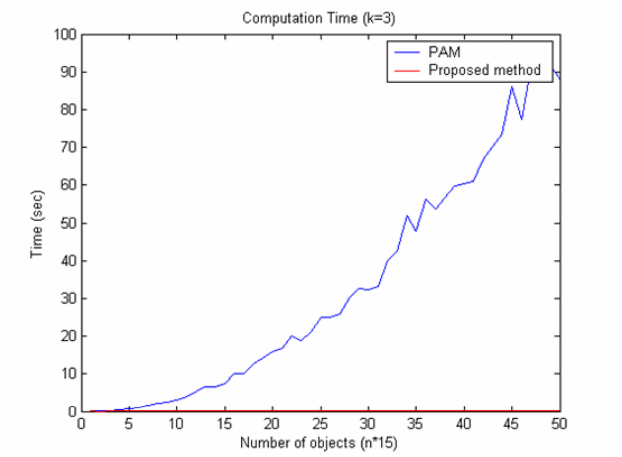


Figure 3 Time comparison of the proposed method with PAM

In fact, the complexity of PAM is $O(k(n-k)^2)$ but that of the proposed method is $O(nk)$ which is equivalent to K-means clustering (Ng and Han, 1994). So, we may conclude that the proposed method is more efficient than PAM.

3.2 Performance comparison of several methods for selecting initial medoids

To compare several methods for selecting initial medoids listed in Section 2, data set is generated by the same way as before with 10% outliers. Table 3 summarizes the results, where the adjusted Rand indices were reported in (a), the computation time were in (b), the distance from medoids to all other objects in each cluster were in (c), and the number of iterations according to the increased number of objects were in (d). Here again, the result is the average of 100 times of repetitions.

Table 3. (a) Adjusted Rand index

n	Proposed	Method1	Method2	Method3	Method4	Method5
300	0.93927	0.8456	0.68002	0.91237	0.71532	0.94416
600	0.92889	0.82134	0.6562	0.93896	0.78439	0.94455
900	0.92832	0.81601	0.65237	0.92356	0.70749	0.94448
1200	0.93135	0.84926	0.63543	0.92593	0.76650	0.94231
1500	0.92939	0.81001	0.63680	0.92376	0.75256	0.94491
1800	0.93771	0.83955	0.63531	0.93278	0.77791	0.94310
2100	0.92736	0.79899	0.59487	0.91689	0.72579	0.94308
2400	0.93755	0.82880	0.67166	0.92734	0.74584	0.94275
2700	0.93284	0.78849	0.65120	0.94318	0.73119	0.94342
3000	0.92201	0.80911	0.65068	0.9322	0.71507	0.94273

(b) Computation time (in seconds)

n	Proposed	Method1	Method2	Method3	Method4	Method5
300	0.088	0.082	0.082	0.078	0.090	0.490
600	0.278	0.264	0.264	0.235	0.280	0.835
900	0.584	0.551	0.564	0.503	0.595	1.290
1200	1.052	0.969	0.993	0.897	1.030	1.921
1500	1.609	1.475	1.566	1.383	1.574	2.596
1800	2.273	2.109	2.226	1.990	2.247	3.424
2100	3.231	3.061	3.091	2.791	3.194	4.487
2400	4.753	4.201	4.391	3.942	4.410	5.848
2700	5.499	5.167	5.356	4.813	5.472	6.942
3000	6.901	6.474	6.778	6.070	6.912	8.380

(c) Distance from medoids to all objects

n	Proposed	Method1	Method2	Method3	Method4	Method5
300	326.9	356.0	403.8	336.6	393.6	325.3
600	662.9	723.1	822.4	652.5	744.6	649.5
900	993.0	1088.0	1233.8	990.6	1181.5	971.8
1200	1321.3	1409.5	1666.9	1327.0	1506.1	1300.0
1500	1651.6	1839.3	2087.9	1670.7	1909.6	1628.8
1800	1956.0	2145.7	2498.0	1964.9	2239.6	1947.1
2100	2306.2	2572.8	3003.9	2344.3	2721.7	2274.1
2400	2609.6	2922.9	3246.8	2634.4	3063.0	2597.3
2700	2952.0	3338.0	3715.1	2924.5	3489.7	2924.5
3000	3312.2	3646.5	4124.5	3280.7	3923.4	3250.1

(d) Number of iterations

n	Proposed	Method1	Method2	Method3	Method4	Method5
300	3.66	3.63	3.66	2.55	4.42	2.07
600	3.61	3.87	3.98	2.21	4.70	2.13
900	3.84	4.09	4.31	2.25	5.17	2.16
1200	3.91	3.95	4.36	2.30	5.01	2.13
1500	4.10	3.89	4.92	2.33	5.06	2.21
1800	3.94	4.10	4.99	2.47	5.22	2.24
2100	4.19	4.78	4.80	2.38	5.42	2.32
2400	4.09	4.33	5.30	2.47	5.47	2.35
2700	3.91	4.37	5.32	2.42	5.75	2.29
3000	4.00	4.34	5.34	2.35	5.86	2.34

The adjusted Rand index by Method 5 (Gaussian mixture) is reported as the best in Table 3(a). It means that its clustering performance is better than others. However, it takes a little more time when estimating the means of the Gaussian mixture model. It is expected that the computational time by Method 5 rapidly increases as the number of clusters increases. But the proposed method is as good as Method 5 in clustering performance and runs faster than Method 5.

3.3 Iris data

We used 'Iris' data set in UCI repository (ftp://ftp.ics.uci.edu/pub/machine-learning-databases/) in order to see the performance of the proposed algorithm. This data set includes 150 objects (50 in each of three classes, 'Setosa', 'Versicolor', 'Virginica'), each objects having 4 variables ('sepal length', 'sepal width', 'petal length', and 'petal width').

Table 4 shows the confusion matrix by K-means clustering method, whereas Table 5 shows that by the proposed method. The accuracy by K-means is 88.7 percent, whereas the accuracy by the proposed method is 92 percent. This example also shows the performance dominance of the proposed method over K-means clustering.

Table 4. Cluster result by K-means

	Setosa (predicted)	Versicolor (predicted)	Virginica (predicted)
Setosa	50	0	0
Versicolor	0	47	14
Virginica	0	3	36

Table 5. Cluster result by the proposed method

	Setosa (predicted)	Versicolor (predicted)	Virginica (predicted)
Setosa	50	0	0
Versicolor	0	41	3
Virginica	0	9	47

4. Conclusion

In this paper, we propose a new algorithm for K-medoids clustering which runs like the K-means clustering. The algorithm has excellent feature that it requires the distance between every pairs of objects only once and uses this distance at every iterative step.

The result from various simulations shows that the proposed method has better performance than K-means clustering and it takes the less computation time than PAM.

Also various methods for selecting initial medoids are presented and compared. Though the Gaussian mixture method is a little better in terms of clustering performance, its computation time is large. So, even the method of selecting initial medoids described in the proposed method is good enough to use when considering both the performance and the computation time.

Acknowledgement

This work was supported by KOSEF through System Bio-Dynamics Research Center at POSTECH.

References

Han, J., Kamber, M. and Tung, A. (2001). Spatial clustering methods in data mining: A survey. In Miller, H., and Han, J., eds., Geographic Data Mining and Knowledge Discovery. Taylor & Francis.

Hubert, L. & P. Arabie (1985). Comparing partitions. Journal of Classification, 2, 193-218.

Kaufman, L. and Rousseeuw, P.J. (1990). Finding Groups in Data: An Introduction to Cluster Analysis. Wiley, New York

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. Berkeley, University of California Press, 1, 281-297.

Ng, R. and J. Han. (1994). Efficient and Effective Clustering Methods for Spatial Data Mining. Proceedings of the 20th International Conference on Very Large Databases, Santiago, Chile.

Vlassis, N. and Likas, A. (2002). A greedy EM algorithm for Gaussian mixture learning. Neural Processing Letters, 15(1), 77-87.

ftp://ftp.ics.uci.edu/pub/machine-learning-databases/