International Journal of Mathematical Archive-3(1), 2012, Page: 285-291
Available online through www.ijma.info ISSN 2229 – 5046

AN ENHANCED ALGORITHM FOR IMPROVING THE EFFICIENCY OF K-MEANS AND K-MEDOID CLUSTERING USING NORMAL DISTRIBUTION DATA POINTS

D. NAPOLEON
Assistant Professor, Department of Computer Science, School of Computer Science and Engineering
Bharathiar University, Coimbatore-641046, India
E-mail: mekaranapoleon@yahoo.co.in

M. SIVASUBRAMANI
Research scholar, Department of Computer Science, School of Computer Science and Engineering
Bharathiar University, Coimbatore-641046, India
E-mail: sivasu4all@gamil.com

S. SATHYA
Research scholar, Department of Computer Science, School of Computer Science and Engineering
Bharathiar University, Coimbatore-641046, India
E-mail: selvarajsathya72@gmail.com

M. PRANEESH*
Research scholar, Department of Computer Science, School of Computer Science and Engineering
Bharathiar University, Coimbatore-641046, India
E-mail: raja.praneesh@gmail.com

(Received on: 31-10-11; Accepted on: 28-11-11)

ABSTRACT

Clustering is one of the unsupervised learning method in which a set of essentials is separated into uniform groups. The K-Means method is one of the most widely used clustering techniques for various applications. paper proposes a method for making the K-Means algorithm more effective and Efficient, so as to get better clustering with reduced complexity. In this research, the most representative algorithms K-Means and K-Medoids and proposed K-Means were examined and analyzed based on their basic approach. The best algorithm in each category was found out based on their performance using Normal Distribution data points. The accuracy of the algorithm was investigated during different execution of the program using Normal Distribution input data points. colors and the execution time is calculated in milliseconds. This paper deals with a method for improving efficiency of the K-Means algorithm and analyze the elapsed time is taken by Efficient K-Means is less than K-Means and K-Medoid algorithm.

Key words: *Data Clustering, Efficient K-Means clustering, Normal Distribution data points.*

1. INTRODUCTION:

A fundamental problem that frequently arises in a great variety of fields such as data mining and knowledge discovery [1], data compression and vector quantization [2], and pattern recognition and pattern classification [3] is the clustering problem. It also has been applied in a large variety of applications, for example, image segmentation, object and character recognition, document retrieval, etc [4]. The main advantage of clustering is that interesting patterns and structures can be found directly from very large data sets with little or none of the background knowledge. The cluster results are subjective and implementation dependent. The quality of a clustering method depends on The similarity measure used by the method and its implementation second, its ability to discover some or all of the hidden patterns, atlast The definition and representation of cluster chosen A number of algorithms for clustering have been proposed by researchers, of which this study establishes with a comparative study of K-Means and K-Medoids clustering algorithms [7,5].

2. K-MEANS CLUSTERING:

This segment describes the original K-Means clustering algorithm. The idea is to classify a given set of data into *k* number of transfer clusters, where the value of *k* is fixed in advance. The algorithm consists of two separate phases: the first stage

*Corresponding author: *M. PRANEESH*,* *E-mail: *raja.praneesh@gmail.com* is to define *k* centroids, one for each cluster [4, 17]. The next stage is to take each point belonging to the given data set and associate it to the nearest centroid. Euclidean distance is generally considered to determine the distance between data points and the centroids. When all the points are included in some clusters, the first step is completed and an early grouping is done. At this point we need to recalculate the new centroids, as the inclusion of new points may lead to a change in the cluster centroids [1, 21]. Once we find *k* new centroids, a new binding is to be created between the same data points and the nearest new centroid, generating a loop. As a result of this loop, the *k* centroids may change their position in a step by step manner. Eventually, a situation will be reached where the centroids do not move anymore. This signifies the convergence criterion for clustering. Pseudo code for the K-Means clustering algorithm is listed as Algorithm 1 [13].

Algorithm 1: The K-Means clustering algorithm

Input: D = {d1, d2,......,dn} //set of *n* data items.
k // Number of desired clusters

Output: A set of *k* clusters.

Step 1: Arbitrarily choose *k* data-items from D as initial centroids;

Step 2: Repeat
Assign each item *d*i to the cluster which has the closest centroid;

Calculate new mean for each cluster; Until convergence criteria is met

The process, which is called "K-Means", appears to give partitions which are reasonably Efficient in the sense of within-class variance, corroborated to some extend by mathematical analysis and practical experience. Also, the K-Means procedure is easily programmed and is computationally economical, so that it is feasible to process very large samples on a digital computer[17]. K-Means algorithm is one of first which a data analyst will use to investigate a new data set because it is algorithmically simple, relatively robust and gives "good enough" answers over a wide variety of data sets [18].

3. MODIFIED APPROACH:

In the enhanced clustering method discussed in this paper, both the phases of the original K-Means algorithm are personalized to improve the efficiency [15]. The enhanced method is outlined as Algorithm 2.

Algorithm 2: The Enhanced method

Input: D = {d1, d2,......,dn} // set of *n* data items
k // Number of desired clusters

Output: A set of *k* clusters.

Phase 1: Determine the initial centroids of the clusters by using Algorithm 3.

Phase 2: Assign each data point to the appropriate clusters by using Algorithm 4.

In the first phase, the initial centroids are determined systematically so as to produce clusters with better accuracy [16]. The second phase makes use of a variant of the clustering method discussed in [8]. It starts by forming the initial clusters based on the relative distance of each data-point from the initial centroids. These clusters are subsequently fine-tuned by using a heuristic approach, thereby improving the efficiency. The two phases of the enhanced method are described below as Algorithm 3 and Algorithm 4.

Algorithm 3: Finding the initial centroids

Input: D = {d1, d2,......,dn} // set of *n* data items
k // Number of desired clusters

Output: A set of *k* initial centroids

Step 1: Set m = 1;

Step 2: Compute the distance between each data point and all other data- points in the set D;

Step 3: Find the closest pair of data points from the set D and form a data-point set Am (1<= m <= k) which contains these two data- points, Delete these two data points from the set D;

Step 4: Find the data point in D that is closest to the datapoint set Am, Add it to Am and delete it from D;

Step 5: Repeat step 4 until the number of data points in Am reaches 0.75*(n/k);

Step 6: If m<k, then m = m+1, find another pair of datapoints from D between which the distance is the shortest, form another data-point set Am and delete them from D, Go to step 4;

Step 7: For each data-point set Am (1<=m<=k) find the arithmetic mean of the vectors of data points in Am, these means will be the initial centroids [23].

Algorithm 3 describes the method for finding initial centroids of the clusters [12]. Initially, compute the distances between each data point and all other data points in the set of data points. Then find out the closest pair of data points and form a set A1 consisting of these two data points, and delete them from the data point set D. Then determine the data point which is closest to the set A1, add it to A1 and delete it from D. Repeat this procedure until the number of elements in the set A1 reaches a threshold. At that point go back to the second step and form another data-point set A2. Repeat this till 'k' such sets of data points are obtained. Finally the initial centroids are obtained by averaging all the vectors in each data-point set. The Euclidean distance is used for determining the closeness of each data point to the cluster centroids. The distance between one vector X = (x1, x2, ...xn) and another vector Y = (y1, y2, …….yn) is obtained as (,) (1 1)2 (2 2)2 .... ( )2 *d X Y = x − y + x − y + + xn − yn* The distance between a data point X and a data-point set D is defined as d(X, D) = min (d (X, Y), where Y ∈ D). The initial centroids of the clusters are given as input to the second stage, for assigning data-points to appropriate clusters. The steps involved in this phase are outlined as Algorithm 4[11].

Algorithm 4: Assigning data-points to clusters

Input: D = {d1, d2,......,dn} // set of *n* data-points.
C = {c1, c2,.......,ck} // set of *k* centroids

Output: A set of *k* clusters

STEPS:

Step 1: Compute the distance of each data-point *di* (1<=i<=n) to all the centroids

cj (1<=j<=k) as *d(di, cj)*;

Step 2: For each data-point *di*, find the closest centroid *cj* and assign *di* to cluster *j*.

Step 3: Set ClusterId [i]=j; // j:Id of the closest cluster

Step 4: Set Nearest_Dist [i]= *d(di, cj)*;

Step 5: For each cluster *j* (1<=j<=k), recalculate the centroids;

```
1. Repeat
2. For each data-point di,
   (a)  Compute its distance from the centroid of the present nearest cluster;
   (b)  If this distance is less than or equal to the present nearest distance, the data-point stays in the cluster;
   (c)  Else for every centroid cj (1<=j<=k) compute the distance d(di, cj);
        End for;
3. Assign the data-point di to the cluster with the nearest centroid cj
4. Set ClusterId[i]=j;
5. Set Nearest_Dist[i] = d(di, cj);
   End for (step(2));
6. For each cluster j (1<=j<=k), Recalculate the centroids until the convergence criteria is met.
```

4. K-MEDOIDS ALGORITHM:

The objective of K-Medoid clustering [KR90] is to find a non-overlapping set of clusters such that each cluster has a most representative point, i.e., a point that is most centrally located with respect to some measure, e.g., distance. These representative points are called medoids. In K-Medoids methods a cluster is represented by one of its points. We have already mentioned that this is an easy solution since it covers any attribute types and that medoids have embedded resistance against outliers since peripheral cluster points do not affect them. When medoids are selected, clusters are defined as subsets of points close to respective medoids, and the objective function [22] is defined as the averaged distance or another dissimilarity measure between a point and its medoid. A typical KMediods algorithm for partitioning based on Medoid or central objects is as follow as

Input: K: The number of clusters
D: A data set containing n objects

Output: A set of k clusters that minimizes the sum of the dissimilarities of all the objects to their nearest medoid.

Method: Arbitrarily choose k objects in D as the initial representative objects;

Repeat: Assign each remaining object to the cluster with the nearest medoid; Randomly select a non medoid object

Orandom; compute the total points S of swap point Oj with Oramdom

if S < 0 then swap Oj with Orandom to form the new set of k medoid

Until no change;

Like this algorithm, a Partitioning Around Medoids (PAM) was one of the first K-Medoids algorithms introduced. It attempts to determine k partitions for n objects. After an initial random selection of k medoids, the algorithm repeatedly tries to make a better choice of medoids. [6]

5. EXPERIMENTAL RESULTS

In this study, the K-Means algorithm is explained with an example first, followed by enhanced K-Means algorithm. The experimental results are discussed for the K-Means algorithm. The resulting clusters of the Normal Distribution of K-Means algorithm is presented in Fig. 1. The number of clusters and data points is given by the user during the execution of the program. The number of data points is 1000 and the number of clusters given by the user is 10 (k = 10). The algorithm is repeated 1000 times to get Efficient output. The cluster centers (centroids) are calculated for each cluster by its mean value and clusters are formed depending upon the distance between data points. For different input data points, the algorithm gives different types of outputs. The enhanced K-Means is better than K-Means in experimental results. In cluster size has to be differing in different run.

Table-(1): Cluster results for K-Means using Normal Distribution

No. of clusters	1	2	3	4	5	6	7	8	9	10	Time in ms
Run1	102	95	81	75	86	117	76	102	135	131	2124.6
Run2	118	101	93	119	99	90	94	61	99	126	2141.4
Run3	136	99	102	132	96	88	84	86	90	87	2141.1
Run4	78	96	110	120	146	50	100	80	94	126	2140.2
Run5	102	96	99	100	97	115	65	126	105	95	2140.3

Table-(2): Cluster results for enhanced K-Means using Normal Distribution

No. of clusters	1	2	3	4	5	6	7	8	9	10	Time in ms
Run1	102	95	81	75	86	117	76	102	135	131	2124.6
Run2	118	101	93	119	99	90	94	61	99	126	2141.4
Run3	136	99	102	132	96	88	84	86	90	87	2141.1
Run4	78	96	110	120	146	50	100	80	94	126	2140.2
Run5	102	96	99	100	97	115	65	126	105	95	2140.3

Table-(3): Cluster results for K-Medoid using Normal Distribution

No.of clusters	1	2	3	4	5	6	7	8	9	10	Time in ms
Run1	42	77	65	148	125	104	67	95	111	166	62.2ms
Run2	102	112	108	74	103	121	1102	97	87	94	45.97
Run3	120	91	106	101	122	97	88	87	111	77	44.76
Run4	92	133	71	95	95	101	74	101	121	117	34.45
Run5	123	86	93	64	96	104	133	84	121	96	28.59

Fig-1: Normal Distribution output in K-Means

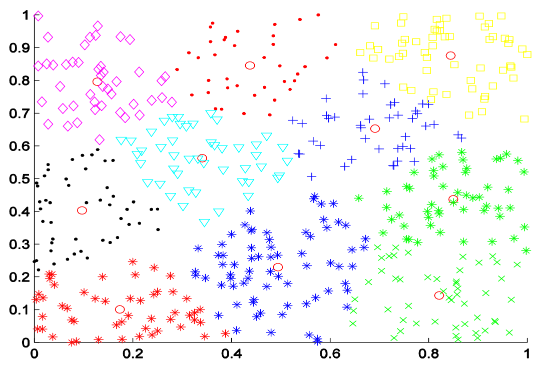


Fig.-(2): Normal Distribution output in enhancedK-Means

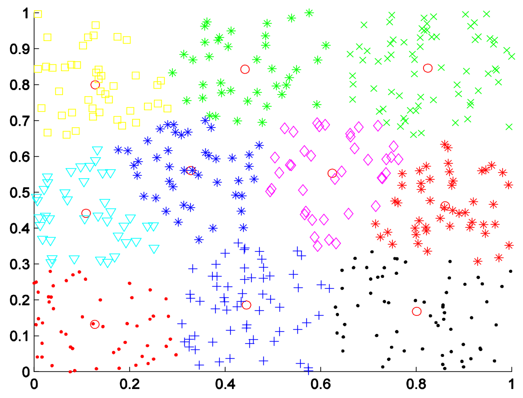
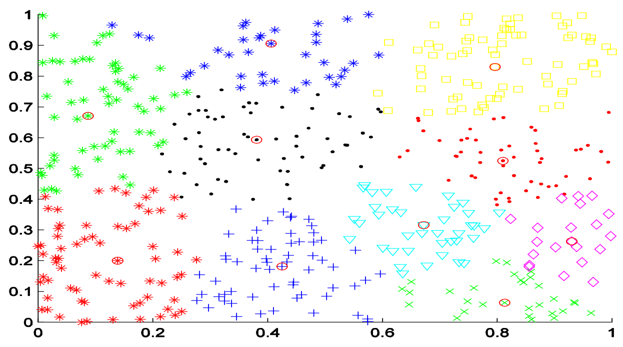


Fig. - (3): Normal Distribution output in K-Medoid



6. CONCLUSION:

The time taken for one execution of the program for the Normal Distribution data points . Usually the time complexity varies from one processor to another processor, which depends on the speed and the type of the system. The partition based algorithms work well for finding spherical-shaped clusters in small to medium-sized data points. K-Medoids algorithm seems to perform better for large data sets and Efficient K-Means algorithm is more efficient than K-Medoid and K-Means algorithm.

7. REFERENCES:

[1] U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press (1996)

[2] A. Gersho, R.M. Gray, Vector Quantization and Signal Compression, Kluwer Academic, Boston, MA (1992)

[3] R.O. Duda, P.E. Hart, Pattern Classification and Scene Analysis, Wiley, New York (1973)

[4] M.N. Murty, A.K. Jain, P.J. Flynn, Data clustering: a review, ACM Comput. Surv. 31(3) (1999) 264-323

[5] Khan, S.S. and A. Ahmad, 2004. Cluster center initialization algorithm for K-Means clustering. Patt. Recog. Lett., 25:1293-1302

[6] Park, H.S., J.S. Lee and C.H. Jun, 2006. A K-Means like algorithm for K-Medoids clustering and its performance.

[7] Berkhin, P., 2002. Survey of clustering data mining techniques. Technical Report, Accrue Software, Inc.

[8] Chaturvedi J. C. A, Green P, "K-modes clustering," *J. Classification,* (18):35–55, 2001.

[9] Daxin Jiang, Chum Tong and Aidong Zhang, "Cluster Analysis for Gene Expression Data," *IEEE Transactions on Data and KnowledgeEngineering*, 16(11): 1370-1386, 2004.

[10] Fahim A.M, Salem A. M, Torkey A and Ramadan M. A, "An Efficient enhanced K-Means clustering algorithm," *Journal of Zhejiang University*, 10(7) :1626–1633, 2006.

[11] Huang Z, "Extensions to the K-Means algorithm for clustering large data sets with categorical values," *Data Mining and Knowledge Discovery*, (2):283–304, 1998.

[12] Jiawei Han M. K, *Data Mining Concepts and Techniques*, Morgan Kaufmann Publishers, An Imprint of Elsevier, 2006.

[13]Margaret H. Dunham, *Data Mining- Introductory and Advanced Concepts*, Pearson Education, 2006.

[14] McQueen J, "Some methods for classification and analysis of multivariate observations," *Proc. 5th Berkeley Symp. Math. Statist.Prob.*, (1):281–297, 1967.

[15] K. A. Abdul Nazeer, M. P. Sebastian" Improving the Accuracy and Efficiency of the K-Means Clustering Algorithm" Proceedings of the World Congress on Engineering 2009 Vol I WCE 2009, July 1 - 3, 2009, London, U.K.

[16] Pang-Ning Tan, Michael Steinback and Vipin Kumar, *Introduction toData Mining*, Pearson Education, 2007.

[17] Stuart P. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, 28(2): 129-136.

[18] Yuan F, Meng Z. H, Zhang H. X and Dong C. R, "A New Algorithm to Get the Initial Centroids," *Proc. of the 3rd International Conference on Machine Learning and Cybernetics*, pages 26–29, August 2004.

[19] MacQueen, J.: Some Methods for Classification and Analysis of Multivariate Observations. In Proceedings Fifth Berkeley Symposium Mathematics Statistics and Probability.Vol. 1. Berkeley, CA (1967) 281-297.

[20] Wesan, Barbakh And Colin Fyfe. "Local vs global interactions in clustering algorithms: Advances over K-Means." International Journal of knowledge-based and Intelilligent Engineering Systems 12 (2008).83 – 99.

[21] Performance analysis of AIM-K-Means and K-Means in quality cluster generation. J. Comput., 1: 175-178.

[22] Data Mining: Introductory and Advanced Topics. 1st Edn. Prentice Hall, USA, ISBN: 10: 0130888923, pp: 315.

[23] Han, J. and M. Kamber, 2006. Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, 2nd Edn., New Delhi, ISBN: 978-81-312-0535-8.

[24] Jain, A.K. and R.C. Dubes, 1988. Algorithms for Clustering Data. Prentice Hall Inc., Englewood Cliffs, New Jersey, ISBN: 0-13-022278-X, pp: 320.

[25] Jain, A.K., M.N. Murty and P.J. Flynn, 1999. Data clustering: A review. ACM Comput. Surveys.

****