Analysis And Study Of K-Means Clustering Algorithm

Sudhir Singh and Nasib Singh Gill
Deptt of Computer Science & Applications
M. D. University, Rohtak, Haryana

Abstract

Study of this paper describes the behavior of K-means algorithm. Through this paper we have try to overcome the limitations of K-means algorithm by proposed algorithm. Basically actual K-mean algorithm takes lot of time when it is applied on a large database. That's why the proposed clustering concept comes into picture to provide quick and efficient clustering technique on large data set. In this paper performance evaluation is done for proposed algorithm using Max Hospital Diabetic Patient Dataset.

Keywords: *Clustering, K-means, Threshold, outlier, Square Error.*

1. Introduction

Clustering is the process of partitioning or grouping a given set of patterns into disjoint clusters. This is done such that patterns in the same cluster are alike and patterns belonging to two different clusters are different. Clustering has been a widely studied problem in a variety of application domains. Several algorithms have been proposed in the literature for clustering: CLARA, CLARANS [6], Focusing Techniques [4], P-CLUSTER [5]. DBSCAN [3] and BIRCH [7]. The k-means method has been shown to be effective in producing good clustering results for many practical applications. However, a direct algorithm of k-means method requires time proportional to the product of number of patterns and number of clusters per iteration. This is computationally very expensive especially for large datasets. We propose a novel algorithm for implementing the k-means method. Our algorithm produces the same or comparable (due to the round-off errors) clustering results to the direct k-means algorithm. It has significantly superior performance than the direct k-means algorithm in most cases. The rest of this paper is organized as follows. We review previously proposed approaches for improving the performance of the k-means algorithms in Section 2. We present our algorithm in Section 3, time complexity of algorithms in Section 4, we describe the experimental results in Section 5 and we conclude with Section 6.

2. K-MEANS CLUSTERING

K-means algorithm is one of the partitioning based clustering algorithms [2]. The general objective is to obtain the fixed number of partitions/clusters that minimize the sum of squared Euclidean distances between objects and cluster centroids.

Let $X=\{x_i| \ i=1,2,\ldots\ldots,n\}$ be a data set with n objects, k is the number of clusters, m_j is the centroid of cluster cj where $j=1,2,\ldots\ldots,k$. Then the algorithm finds the distance between a data object and a centroid by using the following Euclidean distance formula [1].

The Euclidean distance between two points/objects/items in a dataset, defined by point X and point Y is defined by Equation below [5].

$$\textbf{EUCLIDEAN DISTANCE}(X,Y) = \left(|X_1-Y_1|^2 + |X_2-Y_2|^{2+\cdots+} |X_{N-1}-Y_{N-1}|^{2+} |X_N-Y_N|^2 \right)^{1/2}$$

OR Euclidean distance formula$=\sqrt{\sum|x_i-m_j|^2}$

where **X** represents is the first data point, **Y** is the second data point, **N** is the number of characteristics or attributes in data mining terminology.

Starting from an initial distribution of cluster centers in data space, each object is assigned to the cluster with closest center, after which each center itself is updated as the center of mass of all objects belonging to that particular cluster. The procedure is repeated until convergence.

2.1. K-MEANS ALGORITHM [1]

INPUT: // Set of n items to cluster

D= {d1, d2, d3,………………, dn}

// No. of cluster (temporary cluster) randomly chosen i.e. k

// So below, K is set of subset of D as temporary cluster and C is set of centroids of those clusters.

K= {k1, k2, k3,………………, kk },

C= {c1, c2, c3,………………, ck}

Where k1= {d1}, k2= {d2}, k3= {d3}…… kk= {dk}

And c1=d1, c2=d2, c3=d3,………. ck=dk,

// here k<=n

Output: // // K is set of subset of D as final cluster and C is set of centroids of these cluster.

K= {k1, k2, k3,………………, kk },

C= {c1, c2, c3,………………, ck}

Algorithm:

K-means (D, K, C)

1. Arbitrarily choose k objects from D as the initial cluster centers.
2. **Repeat**
3. (re) assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster.
4. Update the cluster means, i.e., calculate the mean value of the objects for each cluster.
5. **Until** no change.

2.2. LIMITATTIONS OF K-MEANS CLUSTERING ALGORITHM

A critical look at the available literature indicates the following shortcomings are in the existing K-means clustering algorithms [13].

1. In partitioning based K-means clustering algorithms, the number of clusters (k) needs to be determined beforehand.
2. The algorithm is sensitive to an initial seed selection (starting cluster centroids). Due to selection of initial centroid points it is susceptible to a local optimum and may miss the global optimum. It may converge to suboptimal solutions. This means suboptimal classification may be found, requiring multiple runs with different initial conditions. The selection of spurious data points as a center may lead to no data points in the class, with the outcome that the center cannot be updated.
3. It can model only a spherical shape of clusters. Thus the non convex shape of clusters cannot be modeled in center based clustering.
4. It is sensitive to outliers since a small amount of outliers can substantially influence the mean value.
5. Due to the nature of iteration scheme in producing the clustering result, it begins at starting cluster centroids and iteratively updates these centroids to decrease the square error. But it is not confirmed how many time it iterates which is not relevant for bigger data set. It may take a huge number of iterations to converge. Such number of iterations cannot be determined beforehand and may change from run to run. Result may be bad with high dimensional data.
6. It cannot be used for clustering problems whose results cannot fit in main memory, which is the case when data set has very high dimensionality or desired number of cluster is too big.

3. PROPOSED CLUSTERING ALGORITHM

Input: // A set D of n objects to cluster. A threshold value Tth.

D= {d1, d2, d3,………………, dn}, Tth

Output:// A set K of k subsets of D as final clusters and a set C of centroids of these clusters.

K= {k1, k2, k3,………………, kk},

C= {c1,c2,c3,………………,ck}

Algorithm:

Proposed cluster algorithm (D,Tth)

1. Let k=1
2. // Randomly choose a object from D, let it be p
   k1= {p}
3. K={k1}
4. c1=p
5. C={c1}
6. Assign a constant value to Tth
7. for l=2 to n do
8. Choose next random point from D other than already chose points let it be q.
9. Determine m, distance between q and centroid cm(1<=m<=k) in C such that distance is minimum using eq. (1).
10. If (distance<=Tth) then
11. km=km union q
12. Calculate new mean (centroid cm) for cluster km using eq. (2).
13. Else k=k+1
14. kk={q}
15. K=K union {kk}
16. ck=q
17. C=C union {ck}

3.1. ADVANTAGES OF PROPOSED CLUSTERING

Having looked at the available literature indicates the following advantages can be found in proposed clustering over K-means clustering algorithm.

1. In K-means clustering algorithms, the number of clusters (k) needs to be determined beforehand but in proposed clustering algorithm it is not required. It generates number of clusters automatically.
2. K-means depends upon initial selection of cluster points, it is susceptible to a local optimum and may miss global optimum. Proposed clustering algorithm is employed to improve the chances of finding the global optimum.
3. K-means is sensitive to outliers since a small amount of outliers can substantially influence the mean value. In proposed clustering algorithm outliers can't influence the mean value. They can be easily identified and removed (if desired).
4. In K-means it is not confirmed that how many times it iterates but in proposed clustering it is known.
5. Data are stored in secondary memory and data objects are transferred to main memory one at a time for clustering. Only the cluster representations i.e. centroid are stored permanently in main memory to alleviate space limitations thus space requirements of proposed algorithm is very small, necessary only for the centroids of clusters. In K-means memory space is more required to store each object permanently in memory along with centroids.

4. TIME COMPEXITY

4.1 TIME COMPLEXITY OF K-MEANS CLUSTERING ALGORITHM [1]

To calculate the running time of K-means algorithm it is necessary to know the number of times each statement run and cost of running. But sometimes number of steps is not known so it has been assumed. For example let number of times first statement runs with cost m1 is q (>=1). For each q next statement , for i=1,2,………..n where n is number of data objects, runs n+1 times with cost m2. For each q and for each n, next statement runs k+1 times, where k is number of cluster with cost m3. 4th statement runs one time for each q and for each n with cost m4. Calculating new mean for each cluster requires k+1 runs for each q with cost m5.

Running time for algorithm is the sum of running time for each statement executed i.e.

$$T(n) = m1*q+m2*{}_1\sum^q (n+1)+m3*{}_1\sum^q {}_1\sum^n (k+1)+m4*{}_1\sum^q {}_1\sum^n 1+m5*{}_1\sum^q (k+1).$$

$$= m1*q+m2*q*(n+1)+m3*q*n*(k+1)+m4*q*n*1+m5*q*(k+1).$$

$$= m1*q+m2*q*n+ m2*q+m3*q*n*k+ m3*q*n+m4*q*n+m5*q*k+ m5*q.$$

=(m1+m2+m5)*q+(m2+m3+m4)*q*n+m3*q*n*k.

For **worst case** it will be $O(n^i)$ where 2<=i<3

For **best case** it will be $O(n)$

For **average case** it will be $O(n^2)$.

4.2 Time complexity of proposed clustering algorithm.

Time taken by an algorithm depends on the input data set. Clustering a thousand data objects takes longer time than clustering one object. Moreover K-means and proposed algorithm takes different amounts of time to cluster same data objects. In general, the time taken by an algorithm grows with the size of input, so it is traditional to describe the running time of program as a function of size of its input. To do so, there is need to define the terms "Running Time " and "Size of Input" more carefully. Most natural measure is the number of objects in the input. In this analysis number of objects is represented by n. Running time of an algorithm on a particular input is the number of primitive operations or "steps" executed. It is convenient to define the notion of steps so that it is as machine –independent as possible. A constant amount of time is required to execute each line of algorithm. One line may take different amount of time than another line, but it is assumed that each execution of ith line takes time m_i where m_i is a constant. In the following discussion , expression for running time of both algorithms evolves from a messy formula that uses all the statement costs m_i to a much simpler notation that concise and more easily manipulated. This simpler notation makes it easy to determine whether one algorithm is more efficient than another.

In proposed clustering algorithm, like incremental K-means, number of times each statement runs is known. 1st, 2nd, 3rd, 4th, 5th and 6th statement runs one time only with cost m1, m2, m3, m4, m5 and m6 respectively. Next statement for i=2,3,…………..n, runs n times with cost m7 where n is number of data objects. 8th statement finds next random object to cluster. 9th statement scans centroid of each cluster with cost m9. So it runs k+1 times where k is number of clusters. Rest of statement is part of if-then-else body which runs for n-1 times. Let if-then part body runs for r times with cost m11, m12 and then else part body runs for n-1-r times with cost m13, m14, m15, m16.

Running time algorithm is the sum of running time for each statement executed i.e.

T(n)=m1*1+ m2*1+ m3*1+ m4*1+ m5*1 +m6*1+ m7*n+ m8*q+ m9*$_{i=2}\sum^{n}$ (k+1)+ m10*(n-1)+m11*r+ m12*r+ m13*(n-1-r)+ m14*(n-1-r)+ m15*(n-1-r)+ m16*(n-1-r)+ m17*(n-1-r).

T(n)=m1+ m2+ m3+ m4+ m5 +m6+ (m7+ m10+ m13+m14+m15)*n-(m10+m13+ m14+ m15+ m16+ m17)+(m11+ m12- m13- m14- m15- m16- m17)*r+m9* $_{i=2}\sum^{n}$ (k+1)+m8*q.

For worst case let p increases with increase in i then

$$_{i=2}\sum^{n} (k+1)=2+3\ldots\ldots\ldots.n$$

$$=n*(n+1)/2-1$$

So T(n)=m1+ m2+ m3+ m4+ m5 +m6+ (m7+ m10+ m13+m14+m15)*n-(m10+m13+ m14+ m15+ m16+ m17)+(m11+ m12- m13- m14- m15- m16- m17)*r+m9* n*(n+1)/2-1+m8*q.

$T(n)=O(n^2)$

For best case let p=1 for 2<=i<=n then $_{i=2}\sum^{n}$ (k+1)=2*n

T(n)=m1+ m2+ m3+ m4+ m5 +m6+ (m7+ m10+ m13+m14+m15)*n-(m10+m13+ m14+ m15+ m16+ m17)+(m11+ m12- m13- m14- m15- m16- m17)*r+m9* 2*n+m8*q.

T(n)=O(n)

For average case it will be $O(n^i)$ for 1<=i<=2.

Table1 Comparison of algorithm's running time

Name of algorithm	Worst case	Average case	Best case
k-means	$O(n^i)$ where 2<=i<3	$O(n^2)$	$O(n)$
Proposed Algorithm	$O(n^2)$	$O(n^i)$ where 1<=i<=2	$O(n)$

5. Experimental Result

The implementation of proposed algorithm is using Dot Net Visual Studio 2008 using language C# and backend Microsoft SQL Server 2008. We have evaluated our algorithm on Max hospital data set of diabetic patients. All the experimental results reported are on Intel Core i3 whose clock speed of processor is 3.0GHz and the memory size is 4 GB running on window7 home basic.

Table 2: Experimental Result obtained by Proposed Algorithm

TEST CASE	THERSHOLD VALUE	SQUARE ERROR *100	MIN. NO. OF OBJECT IN A CLUSTER.	NO. OF OBJECT. AS OUTLIERS	NO. OF CLUSTER FOR-MED
1	12	17.57	2	2	9
	11	15.18	2	1	11
	10	9.14	2	4	12
	9	7.64	2	3	13
	8	6.22	2	6	12
	7	4.84	2	8	12
	6	3.78	2	11	12
2	12	17.2	3	6	7
	11	14.79	3	7	8
	10	8.42	3	12	8
	9	6.9	3	11	9
	8	5.58	3	14	8
	7	4.35	3	14	9
	6	3.56	3	15	10
3	12	17.21	4	6	7
	11	14.13	4	10	7
	10	7.49	4	18	6
	9	5.8	4	20	6
	8	5.32	4	17	7
	7	3.92	4	20	7
	6	2.78	4	27	6

Above table shows three test cases having minimum number of object in a cluster as 2,3 and 4, threshold value varies from 6 to 12 for each test case. On different –different threshold value we have obtained different values of square error, number of object as Outlier and number of cluster form.

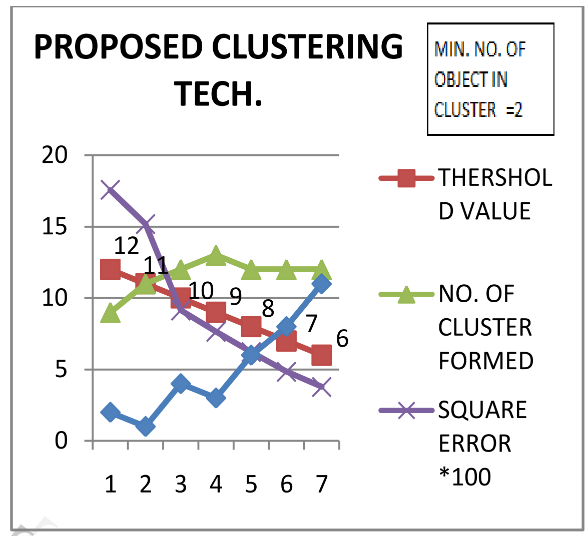

Figure 1: Graph representing test case1.

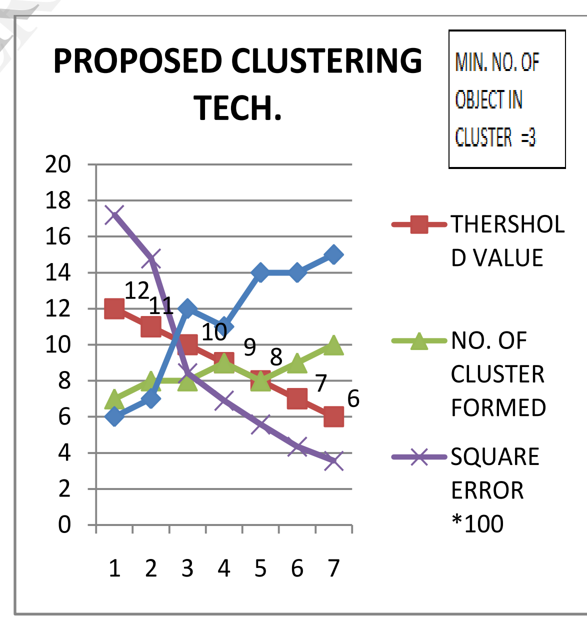

Figure 2: Graph representing test case2.

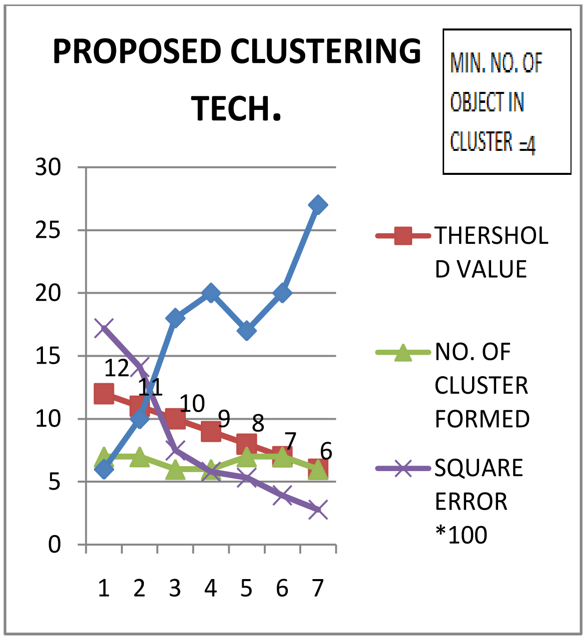

Figure 3: Graph representing test case3.

Above graph shows that

1. As threshold value decreases Square Error decreases. Lower the value of Square Error higher the compactness of cluster and as separate as possible. Hence as we decrease the threshold value cluster quality increases.
2. As we decreases the threshold value number of cluster form increases.
3. As we decrease the threshold value number of object as Outlier increases.

6. Conclusions

In this paper we presented an algorithm for performing K-means clustering. Our experimental result demonstrated that our scheme can improve the direct K-means algorithm. This paper also explains the time complexity of K-means and our purposed algorithm.

There are several improvements possible to the basic strategy presented in this paper. One approach will be to use the concept of Nearest Neighbor Clustering Algorithm to improve the compactness of clusters.

7. References

1. Han, J. &Kamber, M. (2012). Data Mining: Concepts and Techniques. 3rd.ed. Boston: Morgan Kaufmann Publishers.
2. Sudhir Singh, Dr. Nasib Singh Gill,Comparative Study Of Different Data Mining Techniques : A Review, www. ijltemas.in, Volume II, Issue IV, APRIL 2013 IJLTEMAS ISSN 2278 – 2540.
3. M. Ester, H. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Proc. of the 2nd Int'l Conf. on Knowledge Discovery and Data Mining, August 1996.
4. M. Ester, H. Kriegel, and X. Xu. Knowledge Discovery in Large Spatial Databases: Focusing Techniques for Efficient Class Identification. Proc. of the Fourth Int'l. Symposium on Large Spatial Databases, 1995.
5. D. Judd, P. McKinley, and A. Jain. Large-Scale Parallel Data Clustering. Proc. Int'l Conference on Pattern Recognition, August 1996.
6. R. T. Ng and J. Han. Efficient and Effective Clustering Methods for Spatial Data Mining. Proc. of the 20th Int'l Conference on Very Large Databases, Santiago, Chile, pages 144–155, 1994.
7. T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An Efficient Data Clustering Method for Very Large Databases Proc. of the 1996 ACM SIGMOD Int'l Conf. on Management of Data, Montreal, Canada, pages 103–114, June 1996.
8. Performance Evaluation of Incremental K-means Clustering Algorithm, Sanjay Chakraborty , N.K. Nagwani National Institute of Technology (NIT) Raipur, CG, India, IIJDWM, Journal homepage: www.ifrsa.org.
9. PERFORMANCE ANALYSIS OF PARTITIONAL AND INCREMENTAL CLUSTERING, Seminar National Aplikasi Teknologi Informasi 2005 (SNATI 2005) ISBN: 979-756-061-6 Yogyakarta, 18 June 2005.
10. Performance Evaluation of Incremental K-means Clustering Algorithm, IFRSA International Journal of Data Warehousing & Mining |Vol1|issue 1|Aug 2011.
11. M H Dunham, "Data Mining: Introductory and Advanced Topics," Prentice Hall, 2002.
12. R C Dubes, A K Jain, "Algorithms for Clustering Data," Prentice Hall, 1988.
13. Data Mining and Statistics for Decision Making, Page no. 251, Stephane Tuffey, Wiley Publication.