International Journal of Management, IT & Engineering

CONTENTS

Chief Patron

Editorial Board

Editorial Advisory Board

Associate Editors

Reviewers

Title

AN EFFICIENT CENTROID SELECTION ALGORITHM FOR K-MEANS CLUSTERING

Author(s)

Saranya

Research Scholar

Bharathiyar University,

Coimbatore, India

Dr.Punithavalli

Dean, School of Computer Studies

Dr SNS College of Arts & Science,

Coimbatore, India

Abstract:

This paper, we proposes an algorithm for performing data partitioning along the data axis with the highest variance to improve the accuracy. The data partitioning tries to divide data space into small cells or clusters where inter cluster distance are large and intra cluster distance are small as possible. Cells are partitioned one at a time until the number of cells equals to the predefined number of clusters, K. The centers of the K cells become the initial cluster centers for K-means. The experimental results shows that the proposed algorithm will be more effective and efficient converge to better clustering results than the existing clustering.

Keyword: *Data clustering, k-means algorithm, Data partitioning.*

INTRODUCTION:

Clustering is the process of partitioning or combination a given set of patterns into displaces clusters. the goal of the clustering is to group data in to cluster such that similarity among data members within the same cluster are maximal while similarities among data members from different are minimal. The k-means algorithm is one of a group algorithm called partitioning method.

Likas, N. Vlassis and J.J. Verbeek,(2003) proposed the global k-means algorithm has presented which is an incremental approach to clustering that dynamically adds one cluster center at a time through a deterministic global search procedure consisting of N (with N being the size of the data set) executions of the k-means algorithm from suitable initial positions. The propose method will reduce the computational load without significantly affecting solution quality. The proposed clustering methods are tested on well-known data sets and they compare favorably to the k-means algorithm with random restarts.

Fahim A.M, Salem A. M, Torkey A and Ramadan M. A, , "An Efficient enhanced k-means clustering algorithm" has proposed idea that makes k-means more efficient Since, in each iteration, the k-means algorithm computes the distances between data point and all centers, this is computationally very expensive especially for huge datasets.For each data point, it keep the distance to the nearest cluster. At the next iteration, we compute the distance to the previous nearest cluster. If the new distance is less than or equal to the previous distance, the point stays in its cluster, and there is no need to compute its distances to the other cluster centers. This saves the time re-quired to compute distances to k−1 cluster center.

S.S khan and A.Ahmad(2004) has proposed cluster center initalization algorithm (CCIA) based on the considering values for each attribute of the given data set. this leads to the some information leading to a good initial cluster center. The initial cluster centers computed using this methodology are found to be very close to the desired cluster centers, for iterative clustering algorithms. This procedure is applicable to clustering algorithms for continuous data.

Yuan F, Meng Z.H has proposed the systematic method for finding the initial centroids. The centroids obtained in this method are consistent. Hence it will produce the cluster with better accuracy, and it has been compared to the standard k-means algorithm.

EXISTING APPROACH:

A. *The k-means clustering algorithm:*

The segment describes the original k-means clustering algorithm. The k-means cluster is the method of cluster analysis which aims to partition n observation in to k centroid in which each observation belongs to the centroid with the nearest mean. Euclidean distance is generally considered to determine the distance between data points and the centroid once we find k new centroids a new binding is to created between the same Data points and the nearest new centroid as a results, the k- centroid may change their position in a step by step manner. This process will continue until convergence criteria for clustering.

Algorithim1: The Standard K-means clustering algorithm

```
  Input
  D= {d1, d2, d3………dn} // set of elements
  K   // number of desired cluster
  Output:
       K // set of clusters
Steps:
        Assign initial centroid for means in k-data items
Repeat
        Assign each item $d_i$ to the cluster which has the closes
       Calculate new means for each cluster
       Until convergences criteria is met;
```

B. ***The cluster used calculating the initial centroid algorithm k-means.***

Algorithm 2: modified Algorithm

```
 Input
  D= {d1, d2, d3………dn} // set of elements
  K   // number of desired cluster
Output:
       K // set of clusters
```

Steps:

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

Phase 2: Assign each data point to the appropriate clusters by algorithm 4

In the phase1 initial centroids determined systematically so as to produce clusters with better accuracy. In the phase2 it will make use of a variant of the clustering method.

Algorithm 3: Finding the initial centroids

Input

D={ d1,d2,d3………dn} // set of elements

K // number of desired cluster

Output: A set of *k* initial centroids

Steps:

1. Set m = 1;

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

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;

4. Find the data point in D that is closest to the data point set Am, add it to Am and delete it from D;

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

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

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.

Algorithm 3 describes the method for finding initial centroids of the clusters. 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 phase, for assigning data-points to appropriate clusters. The steps involved in this phase are outlined as Algorithm 4.

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:

1. Compute the distance of each data-point *di* (1<=i<=n) to all the centroids *cj* (1<=j<=k) as *d (di, cj)*;

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

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

```
4. Set Nearest_Dist[i] = d(di, cj);
5. For each cluster j (1<=j<=k), recalculate the centroids;
6. Repeat
7. For each data-point di,
7.1 Compute its distance from the centroid of the Present nearest cluster;
7.2 If this distance is less than or equal to the present nearest distance, the data-point stays in the cluster;
Else
7.2.1 For every centroid cj (1<=j<=k) Compute the distance d (di, cj);
 End for;
7.2.2 Assign the data-point di to the cluster with the nearest centroid cj
7.2.3 Set ClusterId[i] =j;
7.2.4 Set Nearest_Dist[i] = d (di, cj);
End for;
8. For each cluster j (1<=j<=k), recalculate the centroids;
Until the convergence criteria is met.
```

PROPOSED APPROACH:

C. *Cluster using Proposed Data Partitioning Based K-means:*

Algorithm 5: Cluster using Proposed Data Partitioning Based K-Means

```
  Input
  D= {d1, d2, d3………dn} // set of elements
```

K // number of desired cluster

Output:

K // set of clusters

Steps:

Phase 1: Determine the cluster using proposed data partitioning based k-means by algorthim6

Phase 2: Assign each data point to the appropriate clusters by algorithm 4

The proposed cluster center initialization algorithm(CCLA) based on considering values For each attributes of the given data set this provides the some information leading to a good initial cluster center, the algorithm are describe below.

Algorithm 6: cluster using proposed data partitioning based k-means

Input:

D = {d1, d2...dn} // set of *n* data-points.

C = {c1, c2...ck} // set of *k* centroids

Output:

1. Set m = 1;

2. Sort all data in the cell *c* in ascending order on each attribute value and links data by a linked list for each attribute.

3. Compute variance of each attribute of cell *c*. Choose an attribute axis with the highest variance as the principal axis for partitioning.

4. Compute squared Euclidean distances between adjacent data along the data axis with the highest variance

$D_j = d(c_j, c_{j+1})^2$ and compute the $dsum_i = \sum_{j=1}^{i} D_j$

5. Compute centroid distance of cell *c*:

$$centroidDist = \frac{\sum_{i=1}^{n} dsum_i}{n}$$

Where *dsumi* is the summation of distances between the adjacent data.

6. Divide cell *c* into two smaller cells. The partition boundary is the plane perpendicular to the principal axis and passes through a point *m* whose *dsumi* approximately equals to *CentroidDist*. The sorted linked lists of cell *c* are scanned and divided into two for the two smaller cells accordingly

7. Compute Delta clustering error for *c* as the total clustering error before partition minus total clustering error of its two sub cells and insert the cell into an empty Max heap with Delta clustering error as a key.

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.

The data axis with the highest variance will be chosen as the principal axis for data partitioning. The reason is to make the inter distance between the centers of the two cells as large as possible while the sum of total clustering errors of the two cells are reduced from that of the original cell. To partition the given data into *k* cells, we start with a cell containing all given data and partition the cell into two cells. Later on we select the next cell to be partitioned that yields the largest reduction of total clustering errors (or Delta clustering error). This can be defined as *Total clustering error of the original cell – the sum of Total clustering errors of the two sub cells of the original cell*. This is done so that every time we perform a partition on a cell, the partition will help reduce the sum of total clustering errors for all cells, as much as possible. We can now use the partitioning algorithm to partition a given set of data into *k* cells. The centers of the cells can then be used as good initial cluster centers for the *K*-means

EXPERIMENTAL AND RESULTS:

We evaluated the proposed algorithm on Wine , leukemia, glass dataset taken from the UCL repository of machine learning databases, is used for testing the accuracy of the cluster using proposed data partitioning based k-means. The proposed algorithm has compared with the standard k-means algorithm and the modified k-means algorithm and cluster using proposed data partitioning based k-means. The value of k, the number of clusters, is taken as 3

The results of the experiments are tabulated below in table 1&2. The accuracy of clustering is determined by comparing the clusters obtained by the experiment with the three clusters already available in the UCL data set. The measurements used to compare the clustering results are

1. The sum of squared error distance between the data and the centroid of the clusters. The SSE results on UCL Dataset are shown in table 1&2 and figure 1&2
2. Entropy to measure the impurity of each cluster

$$E = -\sum_{j=1}^{c} P_j \log P_j$$

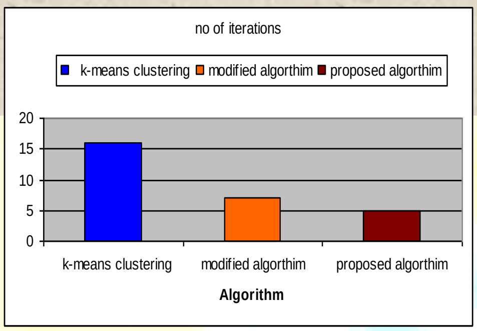
TABLE 1: Leukemia Dataset

	K-means algorithm	Modified algorithm	Proposed algorithm
No of Iterations	9	7	5
Total SSE	2.37E+06	1.18E+06	1.18E+06
Accuracy	0.7187	0.7955	0.7955

TABLE2: Wine Dataset

	K-means algorithm	Modified algorithm	Proposed algorithm
No of Iterations	5	3	2
Total SSE	75.4294	37.7147	28.5651
Accuracy	0.8797	0.9055	0.9124

To achieve the final clustering in UCL Dataset the k-means algorithm and modified algorithm has taken much iteration to calculated the final centroid and accuracy but proposed algorithm takes only less iteration respectively. So the results shows that proposed k-means will convergence quickly.



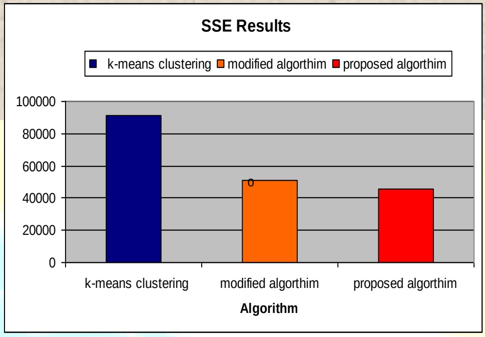


Figure1. No of iteration on UCl Dataset: glass

Figure2: SSE results on UCl Dataset: glass

CONCLUSION

The k-means algorithm is widely used for clustering large sets of data. The standard k-means algorithm will not produce the good results for the accuracy of final cluster. A given data set was partitioned into *k* clusters in such a way the sum of the total clustering errors for all clusters was reduced as much as possible while inter distances between clusters are maintained to be as large as possible. The proposed algorithm is very effective and quick converges to give the better clustering results .the experimental results show that the proposed algorithm performs better than existing algorithm and can reduce running time of K-Means significantly for large data sets.

REFERENCES:

- *A. Likas, N. Vlassis and J.J. Verbeek, "The Global k-means Clustering algorithm", Pattern Recognition , Volume 36, Issue 2, 2003, pp. 451- 461.*
- *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.*

- *P.S. Bradley and U.M. Fayyad, "Refining initial points for K-means Clustering", Proceeding of The Fifteenth International Conference on Machine Learning, Morgan Kaufmann, San Francisco, CA, 1998, pp. 91-99.*
- *Merz C and Murphy P, UCI Repository of Machine Learning Databases, Available:*
- *ftp://ftp.ics.uci.edu/pub/machine-learning-databases*
- *S. S. Khan and A. Ahmad, "Cluster Center Initialization for K-mean Clustering", Pattern Recognition Letters, Volume 25, Issue 11, 2004, pp. 1293-1302*
- *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.*