International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 2, February- 2013

New Fast K-Means Clustering Algorithm using Modified Centroid Selection Method

Mrs. S. Sujatha , Associate Professor,

School of IT & Science,

Dr. G. R. Damodaran College of Science,

Coimbatore.

Mrs. A. Shanthi Sona, Assistant Professor,

Tiruppur Kumaran College for Women, Tirupur.

***Abstract*---**Cluster analysis is a major technique for classifying a 'mountain' of information into manageable meaningful piles. It is a data reduction tool that creates subgroups that are more manageable than individual datum. The fundamental data clustering problem may be defined as discovering groups in data or grouping similar objects together. The goal of clustering is to find groups of similar objects based on a similarity metric. However, a similarity metric is mainly defined by the user to ensure it suits his needs. Until now, there is still no absolute measure that always fit all applications. Some of the problems associated with current clustering algorithms are that they do not address all the requirements adequately, and need high time complexity when dealing with a large number of dimensions and large data sets. K-Means is one of the algorithms that solve the well known clustering problem. The algorithm classifies objects to a pre-defined number of clusters, which is given by the user (assume k clusters). The idea is to choose random cluster centers, one for each cluster. These centers are preferred to be as far as possible from each other. Starting points affect the clustering process and results. Here the Centroid initialization plays an important role in determining the cluster assignment in effective way. Also, the convergence behavior of clustering is based on the initial centroid values assigned. This paper focuses on the assignment of cluster centroid selection so as to improve the clustering performance by K-Means clustering algorithm. This paper uses Initial Cluster Centers Derived from Data Partitioning along the Data Axis with the Highest Variance to assign for cluster centroid. Experimental result suggests that the proposed approach results in better clustering result when compared to the conventional technique.

Keywords---Clustering, K-Means, Centroid Selection, Partitioning

I. INTRODUCTION

The grouping or clusters are defined through an analysis of the data. Subsequent multi-variate analyses can be performed on the clusters as groups [1]. The concept of clustering has been around for a long time. It has several applications, particularly in the context of information retrieval and in organizing web resources. The main purpose of clustering is to locate information and in the present day context, to locate most relevant electronic resources. The research in clustering eventually led to automatic indexing to index as well as to retrieve electronic records. Clustering is a method in which we make cluster of objects that are some how similar in characteristics. The ultimate aim of the clustering is to provide a grouping of similar records. Clustering is often confused with classification, but there is some difference between the two. In classification the objects are assigned to pre defined classes, whereas in clustering the classes are formed. The term "class" is in fact frequently used as synonym to the term "cluster". Cluster analysis (CA) is an exploratory data analysis tool for organizing observed data (e.g. people, things, events, brands, companies) into meaningful taxonomies, groups, or clusters, based on combinations, which maximizes the similarity of cases within each cluster while maximizing the dissimilarity between groups that are initially unknown. In this sense, CA creates new groupings without any preconceived notion of what clusters may arise, whereas discriminate analysis [2] classifies people and items into already known groups. CA provides no explanation as to why the clusters neither exist nor is any interpretation made. Each cluster thus describes, in terms of the data collected, the class to which its members belong. Items in each cluster are similar in some ways to each other and dissimilar to those in other clusters.

The basic Clustering setup is

Preprocessing and Feature Selection- involves choosing an appropriate feature, and doing appropriate preprocessing and feature extraction on data items to measure the values of the chosen feature set. It will often be desirable to choose a subset of all the features available, to reduce the dimensionality of the problem space. This step often requires a good deal of domain knowledge and data analysis.

Similarity Measure- plays an important role in the process of clustering where a set of objects are grouped into several clusters, so that similar objects will be in the same cluster and dissimilar ones in different cluster [3].

Clustering Algorithm-which use particular similarity measures as subroutines. The particular choice of clustering algorithms depends on the desired properties of the final clustering. A clustering algorithm attempts to find natural groups of components (or data) based on some similarity and also finds

the centroid of a group of data sets. To determine cluster membership, most algorithms evaluate the distance between a point and the cluster centroids. The output from a clustering algorithm is basically a statistical description of the cluster centroids with the number of components in each cluster.

Result Validation-Do the results make sense? If not, want to iterate back to some prior stage. It may also be useful to do a test of clustering tendency, to try to guess if clusters are present at all; note that any clustering algorithm will produce some clusters regardless of whether or not natural clusters exist.

Result Interpretation and Application-Typical applications of clustering include data compression (via representing data samples by their cluster representative), hypothesis generation (looking for patterns in the clustering of data), hypothesis testing (e.g. verifying feature correlation or other data properties through a high degree of cluster formation), and prediction. Among the various clustering algorithms, K-Means (KM) is one of the most popular methods used in data analysis due to its good computational performance. K-means clustering is a method of classifying/grouping items into k groups (where k is the number of pre-chosen groups). The grouping is done by minimizing the sum of squared distances (Euclidean distances) between items and the corresponding centroid [4]. In K-Means choosing the proper initial centroids is the key step of the basic K-means procedure. It is easy and efficient to choose initial centroids randomly, but the results are often poor [5]. Hence the Modified Centroid selection method is introduced Instead of updating the centroid of a cluster after all points have been assigned to clusters, the centroids can be updated as each point is assigned to a cluster. In addition, the relative weight of the point being added may be adjusted. The goal of these modifications is to achieve better accuracy and faster convergence.

II. RELATED WORK

Bradley et al., [6] put forth a technique for refining initial points for clustering algorithms, in particular k-means clustering algorithm. They presented a fast and efficient algorithm for refining an initial starting point for a general class of clustering algorithms. The iterative techniques that are more sensitive to initial starting conditions were used in most of the clustering algorithms like K-means, and EM normally converges to one local minima. They implemented this iterative technique for refining the initial condition which allows the algorithm to converge to a better local minimum value. The refined initial point is used to evaluate the performance of K-means algorithm in clustering the given data set. The results illustrated that the refinement run time is significantly lower than the time required to cluster the full database.

Likas et al., [7] put forth a global K-Means clustering algorithm. The technique was an incremental move toward to clustering that animatedly adds one cluster center at a time in the course of a deterministic global exploration procedure consisting of N (with N being the size of the data set) executions of the k-means algorithm from appropriate initial positions. They also proposed a method to reduce the computational load. Moreover this reduction can be achieved without significantly affecting the solution quality. The indispensable thought essential for the proposed method is that an optimal solution for a clustering problem with M clusters can be obtained using a series of local searches.

Chen Zhang et al., [8] presented a new clustering method based on K-means that have avoided alternative randomness of initial center. This paper focused on K-means algorithm to the initial value of the dependence of K selected from the aspects of the algorithm is improved. First, the initial clustering number is radicN. Second, through the application of the sub-merger strategy the categories were combined. The algorithm does not require the user is given in advance the number of cluster. Experiments on synthetic datasets are presented to have shown significant improvements in clustering accuracy in comparison with the random K-means.

In this paper, Juanying Xie et al., [9] proposed a new version of the global K-means algorithm. The outstanding feature of our algorithm is its superiority in execution time. It takes less run time than that of the available global K-means algorithms. This great advantage is due to that we improved the way of creating the next cluster center in the global K-means algorithm. We defined a novel function to select the optimal candidate center for the next cluster enlightened by the idea of K-medoids clustering algorithm suggested by Park and Jun.

In this approach, Yanfeng Zhang et al., [10] presented a new Neighbor Sharing Selection based Agglomerative fuzzy K-means (NSS-AKmeans) algorithm for learning optimal number of clusters and generating better clustering results. The NSS-AKmeans can identify high density areas and determine initial cluster centers from these areas with a neighbor sharing selection method. To select initial cluster centers, we propose agglomeration energy (AE) factor for representing global density relationship of objects, and a Neighbors Sharing Factor (NSF) for estimating local neighbor sharing relationship of objects.

Xue Sun et al., [12] proposed a semi-supervised K-means algorithm based on the global optimization. It can select an appropriate number of clusters as the K value directly and plan

a great amount of supervision data by using only a small amount of the labeled data. Combining the distribution characteristics of data sets and monitoring information in each cluster after clustering, we use the voting rule to guide the cluster labeling in the data sets. The experiments indicated that the global optimization algorithm for semi-supervised K-means is quite helpful to improve the K-means algorithm, it can effectively find the best data sets for K values and clustering center and enhancing the performance of clustering.

Jieming Wu et al., [13] proposed the method of seeking the initial cluster center embarking from the data object distribution, moreover in order to accurately appraise the cluster result, it also proposed cluster assessment method based on the data object. Through analyzes and contrast of the experiment, the improved cluster algorithm surpasses the traditional K-means cluster algorithm, and also can obtain high and stable classified accuracy.

In this paper, Mingwei Leng et al., [14] presented a new algorithm, called an efficient k-means clustering based on influence factors, which is divided into two stages and can automatically achieve the actual value of k and select the right initial points based on the datasets characters. Propose influence factor to measure similarity of two clusters, using it to determine whether the two clusters should be merged into one. In order to obtain a faster algorithms theorem is proposed and proofed, using it to accelerate the algorithm.

III. FAST K-MEANS CLUSTERING ALGORITHM USING MODIFIED CENTROID SELECTION METHOD

K-Means (KM) is considered one of the major algorithms widely used in clustering. However, it still has some problems, and one of them is in its initialization step where it is normally performed randomly. Another problem for KM is that it converges to local minima. This paper focuses on the initialization phase of K-Means so that the performance of clustering is enhanced.

A. K-means Clustering Algorithm

K-Means is one of the algorithms that solve the well known clustering problem. The algorithm classifies objects to a pre-defined number of clusters, which is given by the user (assume k clusters). This algorithm aims at minimizing an objective function, which is in this case a squared error function. The algorithm is expressed as follows

Algorithm 1: 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.

Steps:

1. Arbitrarily choose k data-items from D as initial centroids;
2. Repeat Assign each item d_i to the cluster which has the closest centroid;
3. Calculate new mean for each cluster;

Until convergence criteria is met.

One drawback of KM is that it is sensitive to the initially selected points, and so it does not always produce the same output. Furthermore, this algorithm does not guarantee to find the global optimum, although it will always terminate. To reduce the effect of randomness, the user can run the algorithm many times before taking an average values for all runs, or at least take the median value. The main purpose of clustering algorithm modifications is to improve the performance of the underlying algorithms by fixing their weaknesses. And because randomness is one of the techniques used in initializing many of clustering techniques, and giving each point an equal opportunity to be an initial one, it is considered the main point of weakness that has to be solved. However, because of the sensitivity of K-Means to its initial points, which is considered very high, we have to make them as near to global minima [11] as possible in order to improve the clustering performance.

B. Enhanced K-Means Clustering Algorithm

In the enhanced clustering method discussed in this thesis, both the phases of the original k-means algorithm are modified to improve the accuracy and efficiency. The enhanced method is outlined as below.

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.

Steps:

1. ***Phase 1:*** Determine the initial centroids of the clusters by using Algorithm 3.
2. ***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. The second phase data points are assigned to appropriate clusters. 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.

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 datapoint set A_m, Add it to Am and delete it from D;
5. Repeat step 4 until the number of data points in A_m reaches 0.75*(n/k);
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;
7. For each data-point set Am (1<=m<=k) find the arithmetic mean of the vectors of data points in A_m, 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

$$d(X,Y) = \sqrt{(x1 - y1)^2 + (x2 - y2)^2 + \cdots + (xn - yn)^2}$$

The distance between a data point X and a data-point set D is defined as

$$d(X,D) = \min\left(d(X,Y),\ \ where\ Y \in D\right)$$

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 $d(d_i, c_j)$;
2. ***For*** each data-point di, find the closest centroid c_j and assign d_i to cluster j.
3. Set ClusterId[i]=j; // j:Id of the closest cluster
4. Set Nearest_Dist[i]= $d(d_i, c_j)$;
5. For each cluster j (1<=j<=k), recalculate the centroids;
6. ***Repeat***
7. ***For*** each data-point d_i,

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 $d(d_i, c_j)$;

Endfor;

7.2.2 Assign the data-point di to the cluster with the nearest centroid c_j

7.2.3 Set ClusterId[i]=j;

7.2.4 Set Nearest_Dist[i]= $d(d_i, c_j)$;

Endfor;

8. ***For*** each cluster j (1<=j<=k), recalculate the centroids;

Until the convergence criteria is met.

The first step in Phase 2 is to determine the distance between each data-point and the initial centroids of all the clusters. The data-points are then assigned to the clusters having the closest centroids. This results in an initial grouping of the data-points. For each data-point, the cluster to which it is assigned (ClusterId) and its distance from the centroid of the nearest cluster (Nearest_Dist) are noted. Inclusion of data-

points in various clusters may lead to a change in the values of the cluster centroids. For each cluster, the centroids are recalculated by taking the mean of the values of its data-points. Up to this step, the procedure is almost similar to the original k-means algorithm except that the initial centroids are computed systematically.

The next stage is an iterative process which makes use of a heuristic method to improve the efficiency. During the iteration, the data-points may get redistributed to different clusters. The method involves keeping track of the distance between each data-point and the centroid of its present nearest cluster. At the beginning of the iteration, the distance of each data-point from the new centroid of its present nearest cluster is determined. If this distance is less than or equal to the previous nearest distance, that is an indication that the data point stays in that cluster itself and there is no need to compute its distance from other centroids. This results in the saving of time required to compute the distances to k-1 cluster centroids. On the other hand, if the new centroid of the present nearest cluster is more distant from the data-point than its previous centroid, there is a chance for the data-point getting included in another nearer cluster. In that case, it is required to determine the distance of the data-point from all the cluster centroids. Identify the new nearest cluster and record the new value of the nearest distance. The loop is repeated until no more data-points cross cluster boundaries, which indicates the convergence criterion. The heuristic method described above results in significant reduction in the number of computations and thus improves the efficiency. However, in order to improve the classification with better accuracy, this thesis uses a new technique to determine the cluster centers which is derived from data partitioning.

A. Initial Cluster Centers Deriving from Data Partitioning

The algorithm follows a novel approach that performs data partitioning along the data axis with the highest variance. The approach has been used successfully for color quantization [36]. The data partitioning tries to divide data space into small cells or clusters where intercluster distances are large as possible and intracluster distances are small as possible.

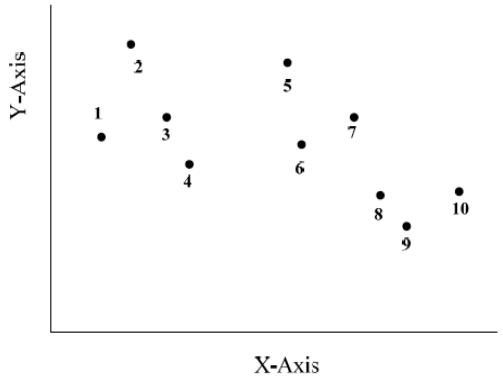

Figure 1: Diagram of ten data points in 2D, sorted by its X value, with an ordering number for each data point

For instance, consider Figure 1. Suppose ten data points in 2D data space are given.

The goal is to partition the ten data points in Figure 1 into two disjoint cells where the sum of total clustering errors of the two cells is minimal, see Figure 2. Suppose a cutting plane perpendicular to X-axis will be used to partition the data. Let C_1 and C_2 be the first cell and the second cell respectively and $\overline{C_1}$ and $\overline{C_2}$ be the cell centroids of the first cell and the second cell, respectively. The total clustering error of the first cell is thus computed by:

$$\sum_{c_i \in C_1} d(c_i, \overline{C_1}) \quad (2)$$

and the total clustering error of the second cell is thus computed by:

$$\sum_{c_i \in C_2} d(c_i, \overline{C_2}) \quad (3)$$

where c_i is the i^{th} data in a cell. As a result, the sums of total clustering errors of both cells are minimal (as shown in Figure 2.)

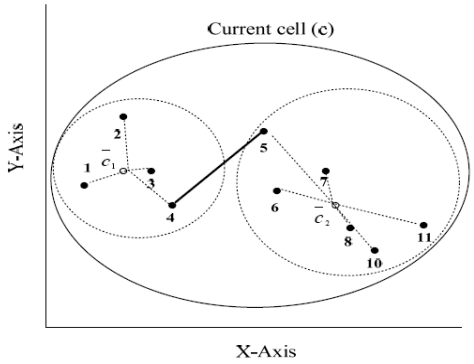

Figure 2: Diagram of partitioning a cell of ten data points into two smaller cells, a solid line represents the intercluster distance and dash lines represent the intracluster distance

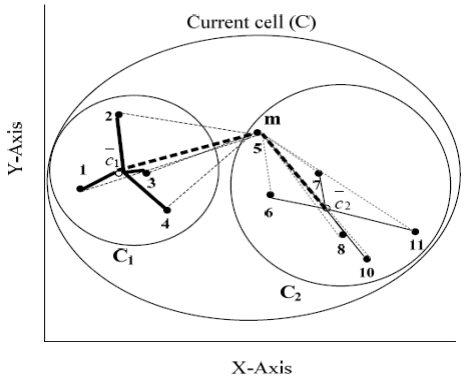

Figure 3: Illustration of partitioning the ten data points into two smaller cells using m as a partitioning point. A solid line in the square represents the distance between the cell centroid and a data in cell, a dash line represents the distance between m and data in each cell and a solid dash line represents the distance between m and the data centroid in each cell

The partition could be done using a cutting plane that passes through m. Thus

$$d(c_i,\overline{c_1}) \leq d(c_i,c_m) + d(\overline{c_1},c_m)\ and$$

$$d(c_i,\overline{c_2}) \leq d(c_i,c_m) + d(\overline{c_2},c_m) \quad (4)$$

(as shown in Figure 3). Thus

$$\sum_{c_i \in C_1} d(c_i,\overline{c_1}) \leq \sum_{c_i \in C_1} d(c_i,c_m) + d(\overline{c_1},c_m).|C_1|$$
$$\sum_{c_i \in C_2} d(c_i,\overline{c_2}) \leq \sum_{c_i \in C_2} d(c_i,c_m) + d(\overline{c_2},c_m).|C_2| \quad (5)$$

m is called as the partitioning data point where |C1| and |C2| are the numbers of data points in cluster C1 and C2 respectively. The total clustering error of the first cell can be minimized by reducing the total discrepancies between all data in first cell to m, which is computed by:

$$\sum_{c_i \in C_1} d(c_i,c_m) \quad (6)$$

The same argument is also true for the second cell. The total clustering error of the second cell can be minimized by reducing the total discrepancies between all data in second cell to m, which is computed by:

$$\sum_{c_i \in C_2} d(c_i,c_m) \quad (7)$$

where $d(c_i,c_m)$ is the distance between m and each data in each cell. Therefore the problem to minimize the sum of total clustering errors of both cells can be transformed into the problem to minimize the sum of total clustering error of all data in the two cells to m.

The relationship between the total clustering error and the clustering point may is illustrated in Fig. 3.4, where the horizontal-axis represents the partitioning point that runs from 1 to n where n is the total number of data points and the vertical-axis represents the total clustering error. When m=0, the total clustering error of the second cell equals to the total clustering error of all data points while the total clustering error of the first cell is zero. On the other hand, when m=n, the total clustering error of the first cell equals to the total clustering error of all data points, while the total clustering error of the second cell is zero.

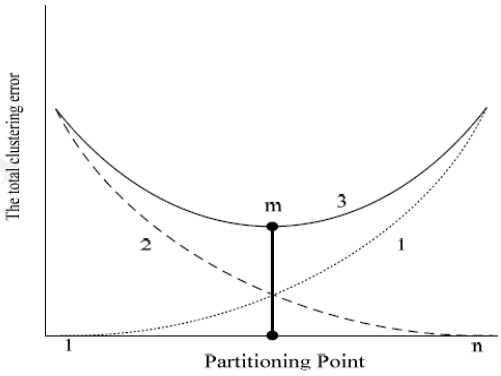

Figure 4: Graphs depict the total clustering error, lines 1 and 2 represent the total clustering error of the first cell and second cell, respectively,

Line 3 represents a summation of the total clustering errors of the first and the second cells

A parabola curve shown in Figure 4 represents a summation of the total clustering error of the first cell and the second cell, represented by the dash line 2. Note that the lowest point of the parabola curve is the optimal clustering point (m). At this point, the summation of the total clustering error of the first cell and the second cell is minimum. Since time complexity of finding the optimal point m is $O(n^2)$, the distances between adjacent data is used along the X-axis to find the approximated point of n but with time of O(n).

Let $D_j = d(c_j,c_{j+1})^2$ be the squared Euclidean distance of adjacent data points along the X-axis.

If i is in the first cell then $d(c_m, c_i) \leq \sum_{j=i}^{m} D_j$. On the one hand, if i is in the second cell then $d(c_m, c_i) \leq \sum_{j=m}^{m} D_j$ (as shown in Figure 5).

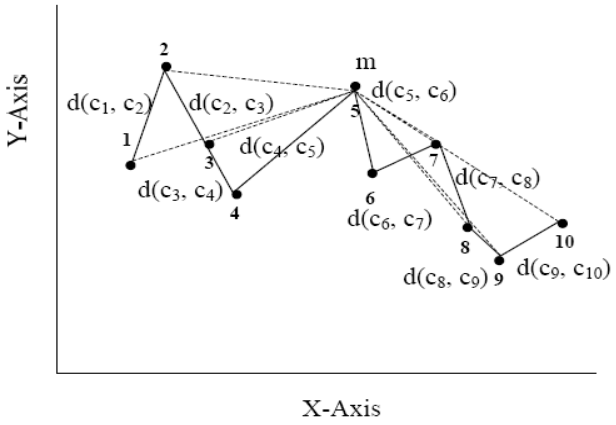

Figure 5: Illustration of ten data points, a solid line represents the distance between adjacent data along the X-axis and a dash line represents the distance between m and any data point

The task of approximating the optimal point (m) in 2D is thus replaced by finding m in one-dimensional line as shown in Figure 6.

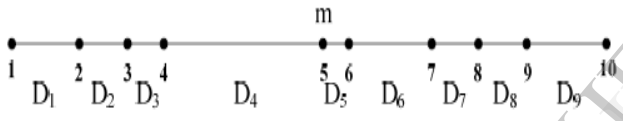

Figure 6: Illustration of the ten data points on a one-dimensional line and the relevant D_j

The point (m) is therefore a centroid on the one dimensional line (as shown in Fig. 3.6), which yields

$$\sum_{i=1}^{m-1} d(c_m, c_i) \approx \sum_{i=m}^{n} d(c_m, c_i) \quad (8)$$

Let $dsum_i = \sum_{j=1}^{i} D_j$ and a *centroidDist* can be computed

$$centroidDist = \frac{\sum_{i=1}^{n} dsum_i}{n} \quad (9)$$

It is possible to choose either the X-axis or Y-axis as the principal axis for data partitioning. However, 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, it is started with a cell containing all given data and partition the cell into two cells. Later on the next cell is selected 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 a partition on a cell is performed, the partition will help reduce the sum of total clustering errors for all cells, as much as possible.

The partitioning algorithm can be used now 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 algorithm. Following are the steps of the initial centroid predicting algorithm.

Algorithm: Deriving Initial Cluster Centers using Data Partitioning

1. Let cell c contain the entire data set.
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 $dsum_i$, 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 $dsum_i$ 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.
8. Delete a max cell from Max heap and assign it as a current cell.
9. For each of the two sub cells of c, which is not empty, perform step 3 - 7 on the sub cell.
10. Repeat steps 8 - 9 until the number of cells reaches K.

IV. EXPERIMENTAL RESULTS

The proposed semi-unsupervised gene selection method is experimented using the following data sets:

- Wine
- Iris
- Glass
- Leukemia

First, the number of iterations required for various techniques are compared. Table 1 represents the comparison of number of iterations required for various techniques with different dataset. From the table, it can be observed that the proposed clustering results in lesser number of iteration when compared to K-Means and modified K-Means techniques.

Table 1: Comparison of Number of Iterations Required for the Proposed and Existing Technique for Various Datasets

Dataset	Iterations		
	K-Means	Modified K-Means	Proposed K-Means
Wine	7	5	5
Iris	10	11	8
Glass	13	5	5
Leukemia	10	2	2

Next, the cluster distance resulted for various techniques are compared. Table 2 represents the comparison of resulted cluster distance for various techniques with different dataset. From the table, it can be observed that the proposed clustering results in maximum cluster distance when compared to K-Means and modified K-Means techniques.

Table 2: Comparison of Cluster Distance Resulted for the Proposed and Existing Technique for Various Datasets

Dataset	Cluster Distance		
	K-Means	Modified K-Means	Proposed K-Means
Wine	2.36936	3.124	4.7082
Iris	75.4294	85.625	114.26
Glass	9.213	11.01	12.2154
Leukemia	365.366	400.235	443.3769

Next, the elapsed time clustering using various techniques are compared. Table 2 represents the comparison of resulted elapsed time for various techniques with different dataset. From the table, it can be observed that the proposed clustering results in lesser time for clustering when compared to K-Means and modified K-Means techniques.

Table 3: Comparison of Required Time for the Proposed and Existing Technique for Various Datasets

Dataset	Elapsed Time		
	K-Means	Modified K-Means	Proposed K-Means
Wine	0.703	0.25	0.195
Iris	0.719	0.485	0.438
Glass	0.437	0.297	0.215
Leukemia	0.313	0.219	0.136

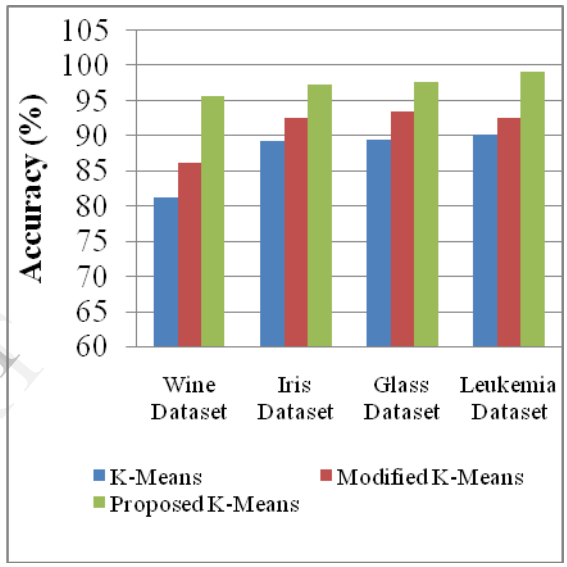

Figure 1: Comparison of Classification Accuracy for the Proposed and Existing Technique for four Datasets

Next, the classification accuracy using various techniques is compared. Figure 1 represents the comparison of resulted accuracy for various techniques with different dataset. From the table, it can be observed that the proposed clustering results in better clustering accuracy when compared to K-Means and modified K-Means techniques.

V. CONCLUSION

Clustering is playing a vital role in many applications. The most commonly used efficient clustering technique is k-means clustering. Initial starting points that are generated randomly by K-means often make the clustering results reaching the local optima. So to overcome this disadvantage a new technique is proposed. This thesis uses the partitioned data along the data axis with the highest variance for assigning the initial centriod for K-Means clustering. The proposed clustering technique is evaluated using different dataset, namely, Wine, Iris, Glass and Leukemia. The parameters used for comparison are number of iterations,

cluster distance, elapsed time and accuracy of clustering. From the results, it can be observed that the proposed technique results in lesser number of iteration which in turn reduces the clustering time. When cluster distance is considered, the proposed clustering technique results in maximum cluster distance which indicates that the proposed technique produces better accuracy for clustering. Considering all these results, the proposed clustering results in better clustering result when compared to the other existing techniques. This is satisfied for all the considered dataset. The future work will be to increase the classification accuracy of the proposed approach. More over, the time taken by the proposed approach should also be considered. The time taken for classification should be very less with high accuracy.

REFERENCES

[1] Shi Yong; Zhang Ge; "Research on an improved algorithm for cluster analysis", International Conference on Consumer Electronics, Communications and Networks (CECNet), Pp. 598 – 601, 2011.

[2] Gkalelis, N.; Mezaris, V.; Kompatsiaris, I.; "Mixture Subclass Discriminant Analysis", IEEE Signal Processing Letters, Vol. 18, No. 5, Pp. 319 – 322, 2011.

[3] Weijiang Jiang; Jun Ye; "Decision-making method based on an improved similarity measure between vague sets", IEEE 10th International Conference on Computer-Aided Industrial Design & Conceptual Design (CAID & CD), Pp. 2086 – 2090, 2009.

[4] de Souza, R.M.C.; de Carvalho, F.A.T.; "A Clustering Method for Mixed Feature-Type Symbolic Data using Adaptive Squared Euclidean Distances", 7th International Conference on Hybrid Intelligent Systems (HIS), Pp. 168 – 173, 2007.

[5] Chen, B.; Tai, P.C.; Harrison, R.; Yi Pan; "Novel hybrid hierarchical-K-means clustering method (H-K-means) for microarray analysis", IEEE Computational Systems Bioinformatics Conference, Pp. 105 – 108, 2005.

[6] P. S. Bradley, and U. M. Fayyad, "Refining Initial Points for K-Means Clustering," ACM, Proceedings of the 15th International Conference on Machine Learning, pp. 91-99, 1998.

[7] Aristidis Likas, Nikos Vlassis, and Jakob J. Verbeek, "The global k-means clustering algorithm," The Journal of Pattern Recognition society, Elsevier, vol. 36, no. 2, pp. 451-461, 2003.

[8] Chen Zhang; Shixiong Xia; "K-means Clustering Algorithm with Improved Initial Center", Second International Workshop on Knowledge Discovery and Data Mining (WKDD), Pp. 790 – 792, 2009.

[9] Juanying Xie; Shuai Jiang; "A Simple and Fast Algorithm for Global K-means Clustering", Second International Workshop on Education Technology and Computer Science (ETCS), Vol. 2, Pp. 36 – 40, 2010.

[10] Yanfeng Zhang; Xiaofei Xu; Yunming Ye; "NSS-AKmeans: An Agglomerative Fuzzy K-means clustering method with automatic selection of cluster number", 2nd International Conference on Advanced Computer Control (ICACC), Vol. 2, Pp. 32 – 38, 2010.

[11] Tasoulis, D.K.; Plagianakos, V.P.; Vrahatis, M.N.; "Clustering in evolutionary algorithms to efficiently compute simultaneously local and global minima", The 2005 IEEE Congress on Evolutionary Computation, Vol. 2, Pp. 1847 – 1854, 2005.

[12] Xue Sun; Kunlun Li; Rui Zhao; Xikun Hu; "Global Optimization for Semi-supervised K-means", Asia-Pacific Conference on Information Processing (APCIP), Vol. 2, Pp. 410 – 413, 2009.

[13] Jieming Wu; Wenhu Yu; "Optimization and Improvement Based on K-Means Cluster Algorithm", Second International Symposium on Knowledge Acquisition and Modeling (KAM '09), Vol. 3, Pp. 335 – 339, 2009.

[14] Mingwei Leng; Haitao Tang; Xiaoyun Chen; "An Efficient K-means Clustering Algorithm Based on Influence Factors", Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD), Vol. 2, Pp. 815 – 820, 2007.