IJRSET November 2015 Volume 2, Issue 12 Pages: 44-49

ISSN 2394-739X

International Journal for Research in Science Engineering and Technology

ACCURACY AND TIME PERIOD COMPARISON USING MODIFIED K-MEAN ALGORITHM

[1] P. VANITHA, [2] S. RANJANI,
[1] Assistant Professor, [2] M.Phil Research Scholar,
[1] Dept of IT & CT, [2] PG & Research Dept of Computer Science,
[1, 2] Hindusthan Colleges,
[1, 2] Coimbatore.

Abstract:-

The K-mean algorithm is a popular clustering algorithm and has its application in data mining, image segmentation, bioinformatics and many other fields. This algorithm works well with small datasets. In this paper we proposed an algorithm that works well with large datasets. Modified k-mean algorithm avoids getting into locally optimal solution in some degree, and reduces the adoption of cluster -error criterion. The algorithm involves initial centroid selection, which is done randomly in existing Algorithm. Hence it proposes an algorithm which selects initial centroids based on the distances calculated from the origin. One of the most popular clustering algorithms is k-means clustering algorithm.

Keywords: - [K Means, Modified K Means, K-means clustering]

1. INTRODUCTION

The aim of the proposed algorithm is to improve the computational efficiency of the K Means algorithm. The algorithm involves initial centroid selection, which is done randomly in existing Algorithm. Hence it proposes an algorithm which selects initial centroids based on the distances calculated from the origin. One of the most popular clustering algorithms is k-means clustering algorithm, but in this method the quality of the final clusters relies heavily on the initial centroids, which are selected randomly. Moreover, the k-means algorithm is computationally very expensive also. The proposed algorithm is found to be more accurate and efficient compared to the original k-means algorithm. This proposed method finding the better initial centroids and provides an efficient way of assigning the data points to the suitable clusters. This method ensures the total mechanism of clustering in O(nlogn) time without loss the correctness of clusters. This approach does not require any additional inputs like threshold values. The proposed algorithm produces the more accurate unique clustering results. The value of k, desired number of clusters is still required.

1.1 Steps

1: In the given data set D, if the data points contain the both positive and negative.

2: Find the minimum attribute value in the given data set D.

3: For each data point attribute, subtract with the minimum attribute value.

4: For each data point calculate the distance from origin.

5: Sort the distances obtained in step 4. Sort the data point's accordance with the distances.

6: Partition the sorted data points into k equal sets.

7: In each set, take the middle point as the initial centroid.

8: Compute the distance betiten each data point di (1 <= i<= n) to all the initial centroids cj (1 <= j <= k).

9: Repeat 10: For each data point di, find the closest centroid cj and assign di to cluster j.

11: Set Clustered[i]=j. // j:Id of the closest cluster.

12: Set Nearest_Dist[i]= d(di, cj).

13: For each cluster j (1 <= j <= k), recalculate the centroids.

14: For each data point di,

14.1 Compute its distance from the centroid of the present nearest cluster.

In cluster analysis, the k-means algorithm can be used to partition the input data set into k partitions (clusters). Hoitver, the pure k-means algorithm is not very flexible, and as such of limited use (except for when vector quantization as above is actually the desired use case!). In particular, the parameter k is known to be hard to choose (as discussed below) when not given by external constraints. In contrast to other algorithms, k-means can also not be used with arbitrary distance functions or be use on non-numerical data. The basic k-means algorithm is commonly measured by any of intra-cluster or inter-cluster criterion. A typical intra-cluster criterion is the squared-error criterion (Equation 1). It is the most commonly used and a good measure of the within-cluster variation across all the partitions. For the current work, it considers intra-cluster squared-error function to evaluate the present scheme of clustering. In basic k-means algorithm, a set D of d dimensional data is partitioned into K clusters, starting with a set of K randomly generated initial center vectors.

The process iterates through the following steps:

- Assignment of data to representative centers upon minimum distance, and
- Computation of the new cluster centers.

The process stops when cluster centers (or the metric M) become stable for two consecutive iterations. The basic k-means algorithm is greedy in nature.

K-means is the most popular and easy-to-understand clustering algorithm. The main idea of K-means is summarized in the following steps:

Arbitrarily choose k objects to be the initial cluster centers/Centroids;

Assign each object to the cluster associated with the closest Centroids;

Compute the new position of each Centroids by the mean value of the objects in a cluster; and

Repeat Steps 2 and 3 until the means are fixed.

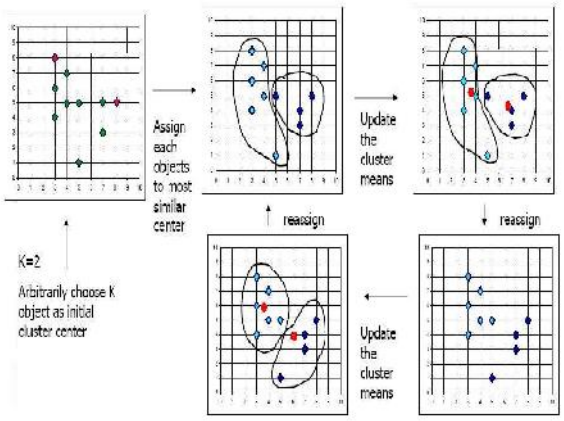


Figure 1. K-means clustering algorithm.

The k-means is possibly the most commonly-used clustering algorithm. It is most effective for relatively smaller data sets. The k-means finds a locally optimal solution by minimizing a distance measure betiten each data and its nearest cluster center. Many parallel versions of the k-means algorithm use the basic serial k-means at their core. Besides, a number of stochastic clustering algorithms make use of the basic k-means or some of its variations. Very often these algorithms are based on Simulated FNN dealing or Genetic Algorithms. The k-means clustering algorithm faces two major problems. One is the problem of obtaining non-optimal solutions. As the algorithm is greedy in nature, it is expected to converge to a locally optimal solution only and not to the global optimal solution, in general. This problem is partially solved by applying the k-means in a stochastic framework like simulated FNN and genetic algorithm (GA) etc. The second problem is that of empty cluster generation. This problem is also referred to as the singularity problem in literature. Singularity in clustering is obtained when one or more clusters become empty. Both the problems are caused by bad initialization. Algorithms that use the k-means at their core suffer from the empty cluster problem too.

A very common practice to solve these problems is to repeat the initialization until it receives a set of good initial center vectors. In practice, after center initialization, it assigns elements to the concerned clusters. If an empty cluster is found, at this early stage, a re-initialization takes place. This process is repeated until all non-empty initial clusters are formed. This is, hoitver, an ad-hoc technique. Several other methodical approaches are also found, a refinement approach is proposed, and where starting with a number of initial samples of the data set it can obtain a number of sets of center vectors. These center vectors then pass through a refinement stage to generate a set of so called good starting vectors. In [21], a genetically guided k-means has been proposed where possibility of generation of empty clusters is treated in the mutation stage. Several k-d-tree based methods are found in [17] and [18]. Another approach to initialize cluster centers based on values for each attribute of the data set has been proposed in [19]. These methods are time costly and may not be applicable by keeping the mean's inherently simple structure.

2. PROPOSED ALGORITHM

Proposed steps for first iteration of k-means algorithm

1- Input k random points as centerfolds for k clusters

2- Assign parameter r with value range between 0.1and1.0

3- Repeat

4- Present data point x 5- Assign point p to the cluster i with minimum distance 6- Recomputed centriod c of cluster i as : c(i,j) = c(i,j) + r * (x(p.j) - c(i,j)) j : features of data point.

The clustering schemes employing the basic k-means algorithm (Section II) face a significant problem of generation of empty clusters at different instances due to bad initialization. In this section, we report a scheme to address the above issue of generation of empty clusters and propose an algorithm to modify the center vector updating procedure of the basic k-means that denies the possibility of empty clusters. We denote this new algorithm as m_k-means algorithm.

2.1 Basis of the Proposed Scheme

In the basic k-means algorithm, iteration starts with a set of old center vectors **z**k (old), the data elements are distributed among clusters depending on minimum Euclidean distance, and then a set of new cluster centers **z**k (new) is generated by averaging the data elements. This center updation procedure in

the original k-means algorithm can be mathematically described as:

$$\mathbf{z}_k^{(new)} \leftarrow \frac{1}{n_k}\{\sum_{x_j \in C_k}(\mathbf{x}_j)\}$$

where, nk is the number of elements in cluster Ck. If the new centers **z**k (new) do not match exactly with the old centers **z**k (old), the k-means algorithm enters into the next iteration assuming **z**k (new) as **z**k (old).

In case of the m_k-means algorithm, the computation of new center vectors differs from that in the k-means algorithm. Here, the old center vectors **z**k (old) are assumed to be members of the concerned clusters along with the allocated data items. In this scheme, we deny the

formation of empty clusters. Therefore, center updation procedure in m_k-means can be written as :

$$\mathbf{z}_k^{(new)} \leftarrow \frac{1}{n_k+1}\{\sum_{x_j \in C_k}(\mathbf{x}_j) + \mathbf{z}_k^{(old)}\}$$

Equation 3 indicates that every cluster should always contain at least one element. One may guess that incorporation of **z**k (old) in computation of **z**k (new), may affect the rate of convergence to final cluster centers and the quality of clustering solutions. However, since the value of **z**k (old) soon becomes well inside the concerned cluster (within a few iterations), the effect of **z**k (old) will be very insignificant, and for a large data set it can be assumed to be negligible. The following subsection reports the design of the present algorithm, referred to as m_k-means clustering algorithm

2.2 The m_k-means Algorithm

The execution steps of the m_k-means algorithm to form clusters are essentially similar to those of the original k-means algorithm. The processor maintains the cluster structures in its own local memory and iterates through the steps of the m_k-means algorithm to evaluate a final set of cluster centers **Z**. The execution steps to be followed are summarized below.

2.3 The m_k -means Algorithm

Input: a set D of d-dimensional data and an integer K.

Output: K clusters

begin

randomly pick K points D to be initial means;

while measure M is not stable **do**

begin

Compute distance dkj = ||**x**j – **z**k||2 for each k, j where 1 ≤ k ≤ K and 1 ≤ j ≤ N, and

Determine members of new K subsets based upon minimum distance to **z**k for 1 ≤ k ≤ K;

compute new center **z**k for 1 ≤ k ≤ K using (3);

compute M;

end

end

The above algorithm reveals that the new clustering scheme is exactly similar to the original k-means algorithm except the only difference at the center computation step. In the following subsection, we shall try to prove that the m_k-means algorithm converges to kmeans centers and the rate of convergence is almost equal to that of the original k-means algorithm.

Proof of Convergence for the m_k-means Algorithm, incorporation of **z**k (old) in computing **z**k (new) may affect the rate of convergence of the present algorithm. However, this effect is insignificant and for a large data set it can be assumed to be negligible. An analytical measure is described in this subsection to estimate the effect of the old centers while computing new ones.

Let us consider that at iteration t, data items are distributed among correct clusters, but center vectors are yet to be stable. Let, at this stage, a certain cluster be represented by C = {**x**1, **x**2, …, **x**m} and the corresponding center is **z**(t) = **x** (say), where,

$$\mathbf{x} \neq \frac{\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_m}{m}$$

due to the initial center effect. Therefore, the following is a sequence of center vectors obtained in the subsequent iterations:

$$\mathbf{z}^{(t)} = \mathbf{x}$$

$$\mathbf{z}^{(t+1)} = \frac{\sum_{i=1}^{m}\mathbf{x}_i + \mathbf{z}^{(t)}}{m+1} = \frac{\sum_{i=1}^{m}\mathbf{x}_i}{m+1} + \frac{\mathbf{x}}{m+1}$$

$$\mathbf{z}^{(t+2)} = \frac{\sum_{i=1}^{m}\mathbf{x}_i + \mathbf{z}^{(t+1)}}{m+1} = \frac{\sum_{i=1}^{m}\mathbf{x}_i}{m+1} + \frac{\sum_{i=1}^{m}\mathbf{x}_i}{(m+1)^2} + \frac{\mathbf{x}}{(m+1)^2}$$

$$\mathbf{z}^{(t+3)} = \frac{\sum_{i=1}^{m}\mathbf{x}_i + \mathbf{z}^{(t+2)}}{m+1} = \frac{\sum_{i=1}^{m}\mathbf{x}_i}{m+1} + \frac{\sum_{i=1}^{m}\mathbf{x}_i}{(m+1)^2} + \frac{\sum_{i=1}^{m}\mathbf{x}_i}{(m+1)^3} + \frac{\mathbf{x}}{(m+1)^3}$$

For large values of m or u, 1/(m+1)u → 0. Hence

$$\mathbf{z}^{(t+u)} = \frac{\sum_{i=1}^{m}\mathbf{x}_i}{m+1} \times \frac{m+1}{m} = \frac{\sum_{i=1}^{m}\mathbf{x}_i}{m}$$

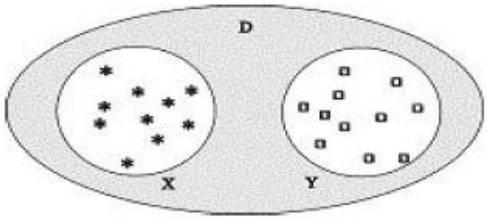
which is the desired cluster center (using k-means). And the additional u iterations needed for final convergence can be expressed by the condition 1/(m+1)u → 0. For large data sets (large m), even for a small u this condition holds. That is, the effect of the initial centers toward convergence is negligible.

Proof of Generation of Non-empty Clusters

In the above subsection, we have shown that the m_kmeans algorithm converges to the k-means centers. In the following we show that the m_k-means algorithm generates non-empty clusters only as opposed to the conventional k-means. The problem of empty clusters occurs when the initial center vectors $\mathbf{z}(0), \mathbf{z}(0), \mathbf{z}(0), \mathbf{z}1, \mathbf{z}2, \ldots, \mathbf{z}K$ are such that any two or more of them are either equal or very close to each other. In such a situation, after assignment of data to clusters, data elements will be assigned to only one of the clusters with nearly equal centers, and the others remain empty.

Let D = {**x**1, **x**2, …, **x**m, **y**1, **y**2, …, **y**n } be a data set with (m+n) data elements forming two well separated clusters X and Y as shown in Figure 1. If we apply serial k-means algorithm to partition this data set into two clusters, we get the following center vectors

$$\mathbf{z}_1 = \frac{\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_m}{m} \text{ and } \mathbf{z}_2 = \frac{\mathbf{y}_1 + \mathbf{y}_2 + \cdots + \mathbf{y}_n}{n}.$$



Now let us consider execution of k-means and m_kmeans on the above data set with identical initial centers.

3. ADVANTAGES

1. K-means clustering is very Fast, robust and easily understandable. If the data set is well separated from each other data set, then it gives best results.
2. The clusters do not having overlapping character and are also non-hierarchical in nature.

4. RESULT

S.No	Algorithm	Accuracy	Time period
1	KMEANS	88.4	3.2
2	DBSCAN	92.5	2.6
3	MK MEANS	95.6	1.8

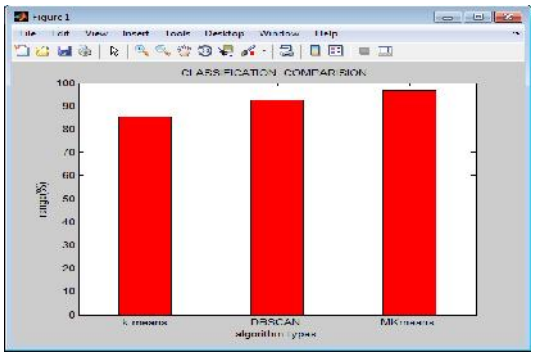


Figure 2: accuracy comparison

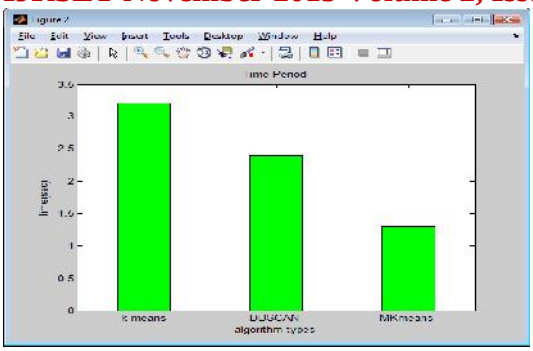


Figure 3: Time period comparison

CONCLUSION

The proposed algorithm will overcome this problem by finding the optimal number of clusters on the run. The main drawback of the proposed approach is that it takes more computational time than the K-means for larger data sets. Future work can focus on how to reduce the time complexity without compromising cluster quality and optimality. More experiments will be conducted with natural datasets with different features. Modified approach Kmean takes less time of computation time as well as than the K-mean and if the number of clusters is more, then it is again true that Modified approach K-mean takes minimum time to execute than the K-mean.

REFERENCES

[1] Joshua Zhexue Huang, Michael K. Ng, Hongqiang Rong, and Zichen Li," Automated Variable Weighting in k-Means Type Clustering", IEEE Transactions on Pattern Analysis and Machine Intelligence, VOL. 27, NO. 5, PP. 657-668,

[2] Jian Yu," General C-Means Clustering Model", IEEE Transactions on Pattern Analysis and Machine Intelligence, VOL. 27, NO. 8, PP.1197-2111,

[3] Carlos Ordonez ," Integrating K-Means Clustering with a Relational DBMS Using SQL", IEEE Trans. Knowl. Data Eng.,, VOL. 18, NO. 2, PP. 188-201,

[4] Francesco Masulli, Stefano Rovetta," Soft Transition From Probabilistic to Possibilistic Fuzzy Clustering", IEEE Trans. on Fuzzy Systems, VOL. 14, NO. 4, PP.516-527,

[5] Michael K. Ng, Mark Junjie Li, Joshua Zhexue Huang, and Zengyou He," On the Impact of Dissimilarity Measure in k-Modes Clustering Algorithm," IEEE Transactions on Pattern Analysis and Machine Intelligence, VOL. 29, NO. 3, PP. 503-507,

[6] Hung-Leng Chen, Kun-Ta Chuang, and Ming-Syan Chen, "On Data Labeling for Clustering Categorical Data", IEEE Trans. Knowl. Data Eng.,VOL. 20, NO. 11, PP.1458-1471,

[7] Eduardo Raul Hruschka, Ricardo J. G. B. Campello, Alex A. Freitas, and Andre C. Ponce Leon F. de Carvalho ," A Survey of Evolutionary Algorithms for Clustering", IEEE Trans. Syst., Man, Cybern.—Part C: Appl. And Review, Vol. 39, No. 2,PP.133-155,

[8] Lin Zhu, Fu-Lai Chung, and Shitong Wang," Generalized Fuzzy C-Means Clustering Algorithm With Improved Fuzzy Partitions", IEEE Trans. Syst., Man, Cybern. B, Cybern ,VOL. 39, NO. 3, PP.578-591,

[9] Pradipta Maji and Sankar K. Pal," Rough Set Based Generalized Fuzzy C-Means Algorithm and Quantitative Indices," IEEE Trans. Syst., Man, Cybern. B, Cybern , vol. 37, no. 6, Dec

[10] J iawei Han, Micheline Kamber,"Data Mining:Concepts and Techniques",Second Edition,Elesvier Publications,