ISSN (Online): 2230-7893
www.IJCEM.org

Evolving limitations in K-means algorithm in data mining and their removal

Kehar Singh [1], Dimple Malik [2] and Naveen Sharma [3]

[1] Prof., Department of CSE, SITM Rewari, MD University
Rohtak, Haryana, India
keharsingh51@yahoo.co.in

[2] Department of CSE, SITM Rewari, MD University
Rohtak, Haryana, India
dimplemalik1234@gmail.com

[3] Department of CSE, SITM Rewari, MD University
Rohtak, Haryana, India
nvn_85@yahoo.com

Abstract

In the modern world large amount of information is stored in our database. These information are stored at different places in our data bases, therefore we have to develop some methods for extracting essential Information from these databases. Therefore we have to develop some methods for extracting essential Information from these databases; data mining is the process of extracting this information. There are various types of algorithms in data mining process. From these algorithm k-means algorithm is evolved. *K*-means is very popular because it is conceptually simple and is computationally fast and memory efficient but there are various types of limitations in k means algorithm that makes extraction some what difficult. In this paper we are discussing these limitations and how these limitations will be removed.

Keywords: *SSE (sum of squared error), HAM (Hamiltonian access method).*

1. Introduction

1.1 K-means algorithm:

K-means **clustering** is a method of cluster analysis which aims to partition *n* observations into *k* clusters in which each observation belongs to the cluster with the nearest mean. It is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The k-means algorithm is an evolutionary algorithm that gains its name from its method of operation. The algorithm clusters observations into k groups, where k is provided as an input parameter. It then assigns each observation to clusters based upon the observation's proximity to the mean of the cluster. The cluster's mean is then recomputed and the process begins again. Here's how the algorithm works:

1. The algorithm arbitrarily selects k points as the initial cluster centers ("means").
2. Each point in the dataset is assigned to the closed cluster, based upon the Euclidean distance between each point and each cluster center.
3. Each cluster center is recomputed as the average of the points in that cluster.
4. Steps 2 and 3 repeat until the clusters converge. Convergence may be defined differently depending upon the implementation, but it normally means that either no observations change clusters when steps 2 and 3 are repeated or that the changes do not make a material difference in the definition of the clusters.

1.2 Terms:

Cluster: A cluster is an ordered list of objects, which have some common characteristics. The objects belong to an interval [a, b], in our case [0, 1].

Distance between Two Clusters: The distance between two clusters involves some or all elements of the two clusters. The clustering method determines how the distance should be computed.

Similarity: A similarity measure SIMILAR (Di, Dj) can be used to represent the similarity between the documents. Typical similarity generates values of 0 for documents exhibiting no agreement among the assigned indexed terms, and 1 when perfect agreement is detected. Intermediate values are obtained for cases of partial agreement.

Average Similarity: If the similarity measure is computed for all pairs of documents (Di, Dj) except when

ISSN (Online): 2230-7893
www.IJCEM.org

i=j, an average value AVERAGE SIMILARITY is obtainable. Specifically, AVERAGE SIMILARITY = CONSTANT SIMILAR (Di, Dj), where i=1, 2….n and j=1, 2….n and i < > j

Threshold: The lowest possible input value of similarity required to join two objects in one cluster.

Similarity Matrix: Similarity between objects calculated by the function SIMILAR (D_i, Dj), represented in the form of a matrix is called a similarity matrix.

Dissimilarity Coefficient: The dissimilarity coefficient of two clusters is defined to be the distance between them. The smaller the value of dissimilarity coefficient, the more similar two clusters are.

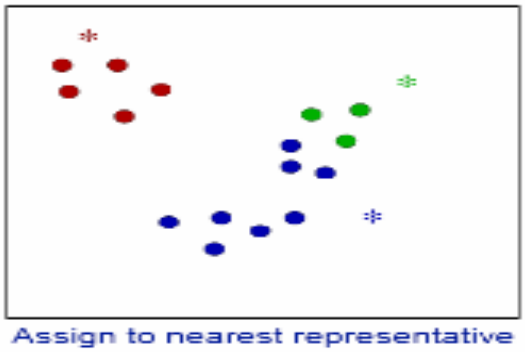
Cluster Seed: First document or object of a cluster is defined as the initiator of that cluster i.e. every incoming object's similarity is compared with the initiator. The initiator is called the cluster seed.

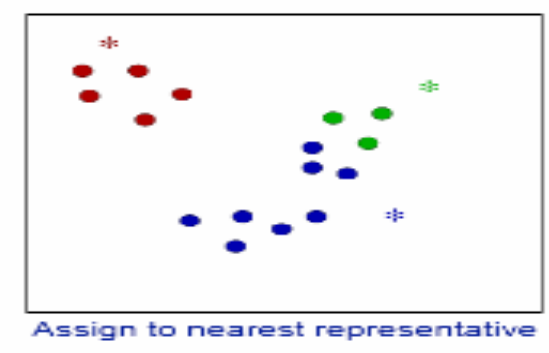
1.3 Description:

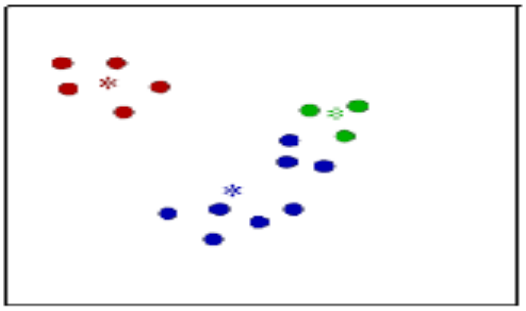
Given a set of observations (x1, x2, …, xn), where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into k sets (k ≤ n) S = {S1, S2, …, Sk} so as to minimize the within-cluster sum of squares (WCSS):

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2$$

Where $\left\| x_i^{(j)} - c_j \right\|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre c_j, is an indicator of the distance of the n data points from their respective cluster centers.







(Steps in k means algorithm)

Given an initial set of k means m1 (1)… mk (1) (see below), the algorithm proceeds by alternating between two steps:

Assignment step: Assign each observation to the cluster with the closest mean (i.e. partition the observations according to the Voronoi diagram generated by the means).

$$S_i^{(t)} = \left\{ \mathbf{x}_j : \left\| \mathbf{x}_j - \mathbf{m}_i^{(t)} \right\| \le \left\| \mathbf{x}_j - \mathbf{m}_{i^*}^{(t)} \right\| \text{ for all } i^* = 1, \ldots, k \right\}$$

Update step: Calculate the new means to be the centroid of the observations in the cluster.

$$\mathbf{m}_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{\mathbf{x}_j \in S_i^{(t)}} \mathbf{x}_j$$

Complexity of k means algorithm is given by:
Complexity is O (n * K * I * d)
n = number of points, K = number of clusters,
I = number of iterations, d = number of attributes

ISSN (Online): 2230-7893
www.IJCEM.org

1.3 Advantages and disadvantages:

The main advantages of this algorithm are its simplicity and speed which allows it to run on large datasets. Its disadvantage is that it does not yield the same result with each run, since the resulting clusters depend on the initial random assignments (the k-means++ algorithm addresses this problem by seeking to choose better starting clusters). It minimizes intra-cluster variance, but does not ensure that the result has a global minimum of variance. Another disadvantage is the requirement for the concept of a mean to be definable which the case is not always. For such datasets the k-medoids variants is appropriate. An alternative, using a different criterion for which points are best assigned to which centre is k-medians clustering.

2. Limitations in K-means algorithm:

Given an integer K, K-means partitions the data set into K non overlapping clusters. It does so by positioning K "centroïds" or "prototypes" in densely populated regions of the data space. Each observation is then assigned to the closest centroid ("Minimum distance rule"). A cluster therefore contains all observations that are all closer to a given centroid than to any of the other centroids (lower image of the illustration).

2.1 Limitation 1:

Handling Empty Clusters:

One of the problems with the basic K-means algorithm given earlier is that empty clusters can be obtained if no points are allocated to a cluster during the assignment step. If this happens, then a strategy is needed to choose a replacement centroid, since otherwise, the squared error will be larger than necessary.

Removal:

One approach is to choose the point that is farthest away from any current centroid. If nothing else, this eliminates the point that currently contributes most to the total squared error. Another approach is to choose the replacement centroid from the cluster that has the highest SSE. This will typically split the cluster and reduce the overall SSE of the clustering. If there are several empty clusters, then this process can be repeated several times.

2.2 Limitation2:

Outliers:

When outliers are present, the resulting cluster centroids (prototypes) may not be as representative as they otherwise would be and thus, the SSE will be higher as well.

Removal:

To remove this it is often useful to discover outliers and eliminate them beforehand. It is important, however, to appreciate that there are certain clustering applications for which outliers should not be eliminated. When clustering is used for data compression, every point must be clustered, and in some cases, such as financial analysis, apparent Outliers.

2.3 Limitation 3:

Reducing the SSE with Post processing:

In k-means to get better clustering we have to reduce the SSE that is most difficult task. There are various types of clustering methods available which reduces the SSE.

Removal:

An obvious way to reduce the SSE is to find more clusters, i.e., to use a larger K. However, in many cases, we would like to improve the SSE, but don't want to increase the number of clusters. This is often possible because K-means typically converges to a local minimum. Various techniques are used to "fix up" the resulting clusters in order to produce a clustering that has lower SSE. The strategy is to focus on individual clusters since the total SSE is simply the sum of the SSE contributed by each cluster.

Two strategies that decrease the total SSE by increasing the number of clusters are the following:

Strategy 1:

Split a cluster: The cluster with the largest SSE is usually chosen, but we could also split the cluster with the largest standard deviation for one particular attribute.

Strategy 2:

Introduce a new cluster centroid: Often the point that is farthest from any cluster center is chosen. We can easily determine this if we keep track of the SSE contributed by each point. Another approach is to choose randomly from all points or from the points with the highest SSE.

Two strategies that decrease the number of clusters, while trying to minimize the increase in total SSE, are the following:

Strategy 1*:*

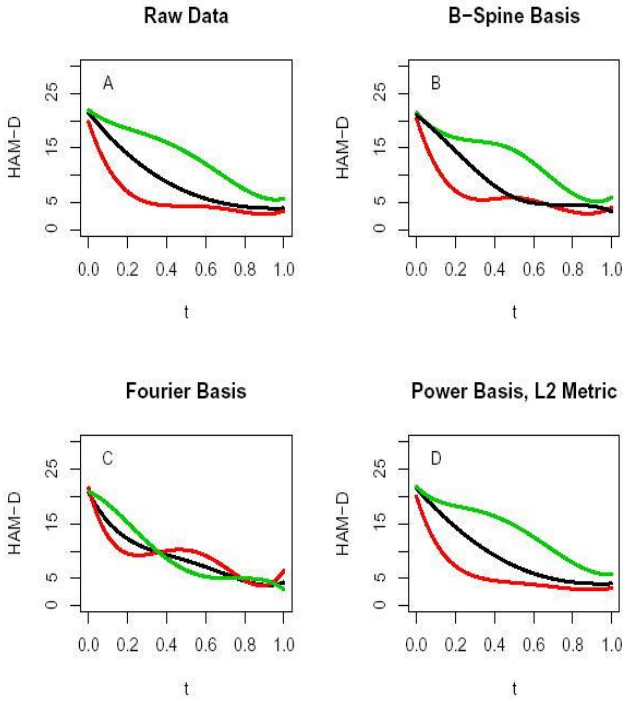
Disperse a cluster: This is accomplished by removing the centroid those cor-responds to the cluster and reassigning the points to other clusters. Ideally, the cluster that is dispersed should be the one that increases the total SSE the least.

Strategy 2:

Merge two clusters: The clusters with the closest centroid are typically chosen, although another, perhaps better, approach is to merge the two clusters that result in the smallest increase in total SSE.

Updating Centroid Incrementally:

Instead of updating cluster centroid after all points have been assigned to a cluster, the centroid can be updated incrementally, after each assignment of a point to a cluster. Notice that this requires either zero or two updates to cluster centroids at each step, since a point either moves to a new cluster (two updates) or stays in its current cluster (zero updates).In addition, if incremental updating is used, the relative weight of the point being added may be adjusted; e.g., the weight of points is often decreased as the clustering proceeds.



(k = 3 cluster mean curves for the Prozac data from four different approaches: **A.** results from clustering the raw data y_i's; **B.** results from clustering estimated *B*-spline coefficients; **C**: results from clustering estimated Fourier coefficients; **D**: results from clustering estimated coefficients from a power basis fit to the c)

Removal:

For updating centroid inclemently we have to evaluate the objective function, suppose that we are given an arbitrary objective function to measure the goodness of a set of clusters. When we process an individual point, we can compute the value of the objective function for each possible cluster assignment, and then choose the one that optimizes the objective. On the negative side, updating centroid incrementally introduces an order dependency. In other words, the clusters produced may depend on the order in which the points are processed. Although this can be addressed by randomizing the order in which the points are processed, the basic K-means approach of updating the centroid after all points have been assigned to clusters has no order dependency. Also, incremental updates are slightly more expensive. However, K-means converges rather quickly, and therefore, the number of points switching clusters quickly becomes relatively small.

2.4 Limitation 4:

Difficult to measure the no of clusters:

The user has to choose the value of *K*, the number of clusters .Although for 2D data this choice can easily be made by visual inspection, it is not so for higher dimension data, and there are usually no clues as to what number of clusters might be appropriate.

Removal:

To calculate the no of clusters we have to make the possible no of groups of observations and for each no of group we have to assign the cluster. From this we have calculate the no of clusters.

3. Conclusion:

K means data mining algorithm is most popular and efficient algorithm because it is very simpler in their operations, but some limitations in this algorithm makes it somewhat difficult. By removing these limitations we have used this algorithm in various field of our life such that in financial analysis, image segmentation and various fields in which we have extracting the information.

4. References:

[1] Athman Bouguettaya "On Line Clustering", IEEE Transaction on Knowledge and Data Engineering Volume 8, No. 2, April 1996.
[2] Euripides G.M. Petrakis and Christos Faloutsos "Similarity Searching in Medical Image Databases", IEEE

ISSN (Online): 2230-7893
www.IJCEM.org

Transaction on Knowledge and Data Engineering Volume 9, No. 3, MAY/JUNE 1997.
[3] Rob Short, Rod Gamache, John Vert and Mike Massa "Windows NT Clusters for Availability and Scalability" Microsoft Online Research Papers, Microsoft Corporation.
[4] Jim Gray "QqJim Gray's NT Clusters Research Agenda" Microsoft Online Research Papers, Microsoft Corporation.
[5] Bruce Moxon "Defining Data Mining, The Hows and Whys of Data Mining, and How It Differs From Other Analytical Techniques" Online Addition of DBMS Data Warehouse Supplement, August 1996.
[6] Willet, Peter "Parallel Database Processing, Text Retrieval and Cluster Analyses" Pitman Publishing, London, 1990.