A Comparative Analysis of Partitioning Based Clustering Algorithms and Applications

Sumit Pandey[1], Sanjay Kumar Dubey[2]
Amity School of Engineering & Technology,
Amity University, Noida, India

Abstract

A set of data points are taking in the process of Clustering and assigning them to groups or clusters where the points in the same cluster are more similar to each another than the members of the other clusters. These clusters are then used to help find patterns in the data set that may allocate relevant correlations to be inferred. Clustering has many different applications in a variety of fields including: image segmentation, market research for advertising, data mining, several different biological applications, and clustering can also be used in file compression. Because clustering is used in many different fields, it is important that an algorithm is able to generate the most accurate clusters, and converge on those clusters as quickly as possible, because clustering is often performed on massive data sets, or the cluster assignments may need to be determined in real-time.

In this paper, two well known partitioning based methods k-means and k-medoids – are compared. The study given here explores the behavior of these two methods.

Key-words: Clustering, K-Means, K-Medoids, Partitioning

1. Introduction

Cluster analysis, also called segmentation analysis or taxonomy analysis [1]. That is, cluster analysis seeks to identify a set of groups which both minimize within-group variation and maximize between-group variation.

Clustering is a method of unsupervised learning and a well known technique for statistical data analysis. It is used in many fields such as machine learning, image analysis, pattern recognition, outlier detection, and bioinformatics to name a few.

2. Type of Clustering Methods

In general, the major clustering methods can be classified into the following categories:

- Partitioning methods
- Hierarchical methods
- Density-based methods
- Grid-based methods
- Model-based methods

2. Clustering Algorithm

Given a database of n objects or data tuple, a partitioning method constructs K partitions of the data, where each partition represents a cluster and K ≤ n. That is, it classifies the data into K groups, which together satisfy the following requirement [2].

i) Each group must contain at least one object.

ii) Each object must belong to exactly one group.

Given K, the number of partitions to construct, a partitioning method creates an initial partitioning. It then uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another. The general criterion of good partitioning is that objects in the same cluster are "close" or related to each other, where as objects of different clusters are "far apart" or very different. There are various kinds of criteria for judging the quality of partitions. On the basis of the concepts various methods are proposed

i) K-mean Methods

ii) K-Medoid Methods

iii) Probabilistic Clustering

The most well known and commonly used partitioning methods are K-Mean, K-Medoids method and their variations[3]. Two methods k-means and k-medoids algorithms are partitional (breaking the dataset up into groups) and both attempt to minimize the distance between points labeled to be in a cluster and a point designated as the center of that cluster.

2.1 Partitioning Based Clustering Methods

A partitioning method creates *k* partitions, called clusters, from given set of *n* data objects. Initially, each data objects are assigned to some of the partitions. An iterative relocation technique is used to improve the partitioning by moving objects from one group to another. Here, each partition is represented by either a centroid or a medoid. A centroid is an average of all data objects in a partition, while the medoid is the most representative point of a cluster. The fundamental requirements of the partitioning based methods are each cluster must contain at least one data object, and each data objects must belong to exactly one cluster. In this category of clustering, various methods have been developed[3][4]. A distance measure is one of the feature space used to identify similarity or dissimilarity of patterns between data objects. Some of the well known methods are k-means, k-medoids;

i) Partitioning around Medoids (PAM)
ii) Clustering LARge Applications (CLARA)
iii) Clustering Large Applications based upon RANdomized Search (CLARANS).

Out of these methods k-means and k-medoids are reviewed here and also similarity measure for both algorithm is carried out by distance measure.

2.1.1 K-mean Clustering Algorithm

K-Means is one of the simplest unsupervised learning methods among all partitioning based clustering methods. It classifies a given set of *n* data objects in *k* clusters, where *k* is the number of desired clusters and it is required in advance. A centroid is defined K-Mean classify a given data set through certain number of clusters (assume k clusters) fixed apriori. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. It is well known that the basic K-means algorithm does not produce an analytic solution. The solution will depend on how the objects are initially assigned to clusters. The K-means algorithm gave better results only when the initial partition was close to the final solution. Several attempts have been reported to solve the cluster initialization problem[3].

Formally, the k-means clustering algorithm follows the following steps.

i) Choose a number of desired clusters, *k*.

ii) Choose *k* starting points to be used as initial estimates of the cluster centroids. These are the initial starting values.

iii) Examine each point (i.e., job) in the workload data set and assign it to the cluster whose centroid is nearest to it.

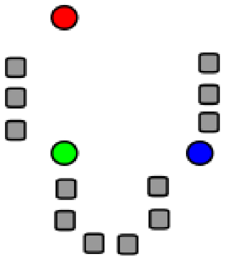
iv) When each point is assigned to a cluster, recalculate the new k centroids.

v) Repeat steps 3 and 4 until no point changes its cluster assignment, or until a maximum number of passes through the data set is performed.

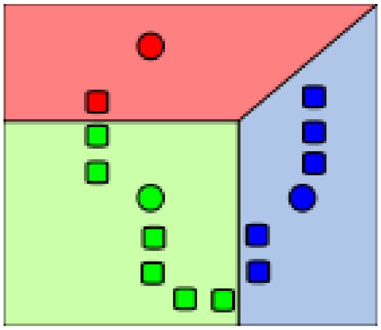
Algorithm 1: **The k-means Clustering Algorithm**

```
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 di to the cluster which
has the closest centroid;
Calculate new mean for each cluster;
Until convergence criteria is met.
```

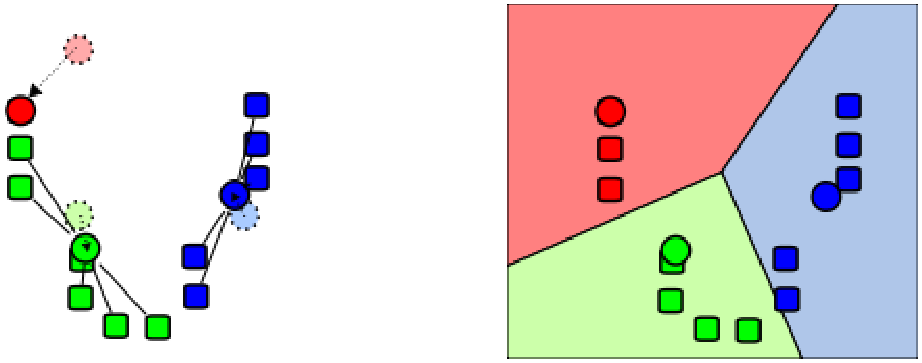
The k-means algorithm is the most extensively studied clustering algorithm and is generally effective in producing good results. The major drawback of this algorithm is that it produces different clusters for different sets of values of the initial centroids. Quality of the final clusters heavily depends on the selection of the initial centroids. The k-means algorithm is computationally expensive and requires time proportional to the product of the number of data items, number of clusters and the number of iterations[3].



k initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).



k clusters are created by associating every observation with the nearest mean.



The centroid of each of the *k* clusters becomes the new mean.

Steps 2 and 3 are repeated until convergence has been reached.

Figure 1: Demonstration of the standard K-Mean Algorithm[4]

Numerical Example: Cluster the following eight points (with (x, y) representing locations) into three clusters A1(2, 10) A2(2, 5) A3(8, 4) A4(5, 8) A5(7, 5) A6(6, 4) A7(1, 2) A8(4, 9). Initial cluster centers are: A1(2, 10), A4(5, 8) and A7(1, 2). The distance function between two points a=(x1, y1) and b=(x2, y2) is defined as: ρ(a, b) = |x2 – x1| + |y2 – y1| .

First we list all points in the first column of the table above. The initial cluster centers means, are (2, 10), (5, 8) and (1, 2) - chosen randomly. Next, we will calculate the distance from the first point (2, 10) to each of the three means, by using the distance function:

point	mean1
x1, y1	x2, y2
(2, 10)	(2, 10)

ρ(a, b) = |x2 – x1| + |y2 – y1|

ρ(point, mean1) = |x2 – x1| + |y2 – y1|

= |2 – 2| + |10 – 10|

= 0 + 0

point	mean2
x1, y1	x2, y2

(2, 10)   (5, 8)

ρ(a, b) = |x2 – x1| + |y2 – y1|

ρ(point, mean2) = |x2 – x1| + |y2 – y1|

= |5 – 2| + |8 – 10|

= 3 + 2

= 5

point	mean3
x1, y1	x2, y2
(2, 10)	(1, 2)

ρ(a, b) = |x2 – x1| + |y2 – y1|

ρ(point, mean2) = |x2 – x1| + |y2 – y1|

= |1 – 2| + |2 – 10|

= 1 + 8

= 9

= 0

1. Next we need to re-compute the new cluster centers. We do so by taking the mean of all points in each cluster.
2. The K-means algorithm is sensitive to outliers, since an object with an extremely large value may substantially distort the distribution of the data.
3. In k-means algorithm, the prototype , called the center, is the mean of all objects belonging to a cluster.
4. It requires several passes on the entire dataset, which can make it very expensive for large dataset, which can make it very expensive for large dataset.
5. The k-medoids approach is more robust in this aspect.

Table 1: k-mean Iteration[4]

		(2, 10)	(5, 8)	(1, 2)	
	Point	**Dist Mean 1**	**Dist Mean 2**	**Dist Mean 3**	**Cluster**
A1	(2, 10)	0	5	9	1
A2	(2, 5)	5	6	4	3

A3	(8, 4)	12	7	9	2
A4	(5, 8)	5	0	10	2
A5	(7, 5)	10	5	9	2
A6	(6, 4)	10	5	7	2
A7	(1, 2)	9	10	0	3
A8	(4, 9)	3	2	10	2

2.1.2. K-Mediods Clustering Algorithm

The k-means method uses centroid to represent the cluster and it is sensitive to outliers. This means, a data object with an extremely large value may disrupt the distribution of data. K-medoids method overcomes this problem by using medoids to represent the cluster rather than centroid. A medoid is the most centrally located data object in a cluster. Here, k data objects are selected randomly as medoids to represent k cluster and remaining all data objects are placed in a cluster having medoid nearest (or most similar) to that data object. After processing all data objects, new medoid is determined which can represent cluster in a better way and the entire process is repeated. Again all data objects are bound to the clusters based on the new medoids. In each iteration, medoids change their location step by step. Or in other words, medoids move in each iteration. This process is continued until no any medoid move. As a result, *k* clusters are found representing a set of n data objects. An algorithm for this method is given below.

Algorithm 2: The k- Mediods clustering algorithm

A typical k-Mediods algorithm for partitioning based on medoid or central objects is as follows:

Input: k: The number of clusters

D: A data set containing n objects

Output: A set of k clusters that minimizes the sum of the dissimilarities of all the objectsto their nearest medoid.

Method: Arbitrarily choose k objects in D as the initial representative objects;

Repeat assign each remaining object to the cluster with the nearest medoid;

randomly select a non medoid object random;

```
compute the total points S of swaping object Oj withOramdom;
if S < 0 then swap Oj with Orandom to form the new set of k medoid;
Until no change;
```

Numerical Example:

Cluster the given data set of ten objects into two cluster i.e. k= 2.

Consider a data set of then objects as follows :

Table 2 : Data Set- Ten Objects

X1	2	6
X2	3	4
X3	3	8
X4	4	7
X5	6	2
X6	6	4
X7	7	3
X8	7	4
X9	8	5
X10	7	6

Step 1

Initialize k centre, Let us assume c1 =(3,4) and c2 =(7 , 4).here c1 and c2 are selected as medoid. Calculating distance so as to associate each data object to its nearest medoid Calculating distance so as to associate each data object to nearest medoid. Cost is calculating using Minkowski distance metric

Table 3 : Cost Calculation[4]

C1		Data Object Xi		Cost(Distance)
3	4	2	6	3
3	4	3	8	4
3	4	4	7	4
3	4	6	2	5
3	4	6	4	3
3	4	7	3	4
3	4	8	5	6
3	4	7	6	6

C1		Data Object Xi		Cost(Distance)
3	4	2	6	3
3	4	3	8	4
3	4	4	7	4
3	4	6	2	5
3	4	6	4	3
3	4	7	3	4
3	4	8	5	6
3	4	7	6	6

Then the clusters become

Cluster1 = {(3,4),(2,6),(3,8),(4,7)}

Cluster2={(7,4),(6,2),(6,4),(7,3),(8,5),(7,6)}

Since the points (2,6) (3,8) and (4,7) are closer to c1 hence they from one cluster whilst remaining points from another cluster .

So the total cost involved is 20.

Where cost is the summation of the cost of data object from its medoid in its cluster so here.

Total cost =

{ cost((3,4),(2,6))+ cost((3,4),(3,8))+cost((3,4),(4,7))}

+

{cost((7,4),(6,2)) + cost((7,4),(6,4))+ cost((7,4),(7,3)) + cost ((7,4),(8,5))+

cost((7,4),(7,6))}

=(3+4+4)+(3+1+1+2+2)

=20

Cluster1={(3,4)(2,6)(3,8)(4,7)}

Cluster1={(7,4)(6,2)(6,4)(7,3)(8,5)(7,6)}

3. Strength and weakness of K-Mean[5][6]

3.1 Strength of K-Mean

i) Relatively scalable and efficient in processing large data sets; complexity is O (i k n), where i is the total Number of iterations, *k* is the total number of clusters, and n is the total number of objects. Normally, k<<n and i<<n.

ii) Easy to understand and implement.

3.2 Weaknesses :

i) Applicable only when the mean of a cluster is defined; not applicable to categorical data.

ii) Need to specify k, the total number of clusters in advance.

iii) Not suitable to discover clusters with non-convex shape, or clusters of very different size.

iv)Unable to handle noisy data and outliers.

v) May terminate at local optimum.

vi)Result and total run time depends upon initial partition.

4. Strength and weakness of K-medoids

4.1 Strength of K-Medoids

i) More robust than k-means in the presence of noise and outliers; because a medoid is less influenced by outliers or other extreme values than a mean.

4.2 Weaknesses

i) Relatively more costly; complexity is *O(i k (n-k)2),* where i is the total number of iterations, is the total

ii) Number of clusters, and *n* is the total number of objects.

iii) Relatively not so much efficient.

iv) Need to specify *k*, the total number of clusters in advance.

v) Result and total run time depends upon initial partition.

Table 4: Comparison of K-means & K-medoids[7][8]

Different Settings	k-means	k-medoids
Complexity	O (i k n)	O (i k (n-k)2)
Efficiency	Comparatively more	Comparatively less
Implementation	Easy	Complicated
Sensitive to Outliers?	Yes	No
Necessity of convex shape	Yes	Not so much
Advance specification of no of clusters 'k'	Required	Required
Does initial partition affects	Yes	Yes

result and runtime?		
Optimized for	Separated clusters	Separated clusters, Small Dataset

5. Clustering Algorithm Applications

5.1. Clustering Algorithm in Identifying Cancerous Data[9]

Clustering algorithm can be used in identifying the cancerous data set. Initially we take known samples of cancerous and non cancerous data set. Label both the samples data set. We then randomly mix both samples and apply different clustering algorithms into the mixed samples data set (this is known as learning phase of clustering algorithm) and accordingly check the result for how many data set we are getting the correct results (since this is known samples we already know the results beforehand) and hence we can calculate the percentage of correct results obtained. Now, for some arbitrary sample data set if we apply the same algorithm we can expect the result to be the same percentage correct as we got during the learning phase of the particular algorithm. On this basis we can search for the best suitable clustering algorithm for our data samples.

It has been found through experiment that cancerous data set gives best results with unsupervised non linear clustering algorithms and hence we can conclude the non linear nature of the cancerous data set.

5.2. Clustering Algorithm in Search Engines[9]

Clustering algorithm is the backbone behind the search engines. Search engines try to group similar objects in one cluster and the dissimilar objects far from each other. It provides result for the searched data according to the nearest similar object which are clustered around the data to be searched. Better the clustering algorithm used, better are the chances of getting the required result on the front page. Hence, the definition of similar object play a crucial role in getting the search results, better the definition of similar object better the result is. Most of the brainstorming activities needs to be done for defining the criteria to be used for similar object.

5.3. Clustering Algorithm in Academics[10][11]

The ability to monitor the progress of students' academic performance has been the critical issue for the academic community of higher learning. Clustering algorithm can be used to monitor the students' academic performance. Based on the students' score they are grouped into different-different clusters (using k-means, fuzzy c-means etc), where each clusters denoting the different level of performance. By knowing the number of students' in each cluster we can know the average performance of a class as a whole.

5.4. Clustering Algorithm in Wireless Sensor Network's based Application[11][12]

Clustering Algorithm can be used effectively in Wireless Sensor Network's based application. One application where it can be used is in Landmine detection. Clustering algorithm plays the role of finding the Cluster heads(or cluster center) which collects all the data in its respective cluster.

6. Conclusion

From the above study, it can be concluded that partitioning based clustering methods are suitable for spherical shaped clusters in small to medium sized data sets. K-means and k-medoids – both the methods find out clusters from the given database. Both the methods require to specify *k*, no of desired clusters, in advance. Result and runtime depends upon initial partition for both of these methods. The advantage of k-means is its low computation cost, while drawback is sensitivity to noisy data and outliers. Compared to this, k-medoid is not sensitive to noisy data and outliers, but it has high computation cost.

References

[1] Salem A.M., Fahim A.M ,Torkey F.A.,Ramdan M.A., “An efficient enhance k-means clustering algorithm”, Journal of Zhejiang university Science,2006 7(10):1626-1633

[2] Oyelade, O. J , Oladipupo, O. O, Obagbuwa, I. C “Application of k-Means Clustering algorithm for prediction of Students’ Academic Performance” (IJCSIS) International Journal of Computer Science and Information Security, Vol. 7, 2010

[3] Depa Pratima , Nivedita Nimmakanti , “Pattern Recognition Algorithms for Cluster Identification Problem”, International Journal of Computer Science & Informatics (IJCSI), ISSN (PRINT) : 2231–5292, Vol.- II, Issue-1, 2

[4] Dr. Aishwarya Batra, “Analysis and Approach: K-Means and K- Medoids Data Mining Algorithms”, 5th IEEE International Conference on Advanced Computing & Communication Technologies [ICACCT-2011] ISBN 81-87885-03-3

[5] Fahim A.M., Salem A.M., “Modified enhanced k-means clustering algorithm”, Journal of Zhejiang University Science, 1626 – 1633, 2006.

[6] Frank Robinson, Amy Apon, Denny Brewer and Larry Dowdy, " Initial Starting Point Analysis for K-Means Clustering: A Case Study" , IEEE Transactions on Pattern Analysis and Machine Intelligence, 2006,pp. 345-254

[7] D T Pham_, S S Dimov, and C D Nguyen "Selection of K in K-means clustering" Proc. IMechE Vol. 219 Part C: J. Mechanical Engineering Science.

[8] Kurt Hornik , Ingo Feinerer, Martin Kober, Christian Buchta "Spherical k-Means Clustering" Journal of Statistical Software September 2012, Volume 50, Issue 10.

[9] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Wu, "An efficient k-means clustering algorithm: analysis and implementation", IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, pp. 881-892.

[10] Khalid Hammouda, Prof. Fakhreddine Karray, "A Comparative Study of Data Clustering Techniques", University of Waterloo, Ontario, Canada N2L 3G1

[11] Osama, Abu Abbas, "Comparison Between Data Clustering Algorithm", The International Arab Journal Of Information Technology, vol.5, No.3, July 2008

[12] Azhar Rauf, Sheeba, Saeed Mahfooz, Shah Khusro and Huma Javed ,"Enhanced K-Mean Clustering Algorithm to Reduce Number of Iterations and Time Complexity", Middle-East Journal of Scientific Research 12 (7): 959-963, 2012