IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

Unsupervised K-Means Clustering Algorithm Analysis

Prof. Ankush Patil, Kalyani Raghatate, Vaibhav Mokale, Sunil Chavan, Payal Gawande

Computer Department, Government College of Engineering, Yavatmal

ankushpatil48@gmail.com, kalyaniraghatate01541@gmail.com, vaibhavmokale123@gmail.com, chavansunil15501@gmail.com, payalgawande2092000@gmail.com

***Abstract*- Among many clustering algorithms, the K-means clustering algorithm is widely used because of its simple algorithm and fast convergence. Conventional database querying methods are inadequate to extract useful information from huge data banks. Cluster analysis is one of the major data analysis methods and the k-means clustering algorithm is widely used for many practical applications. Although it is an unsupervised learning to clustering in pattern recognition and machine learning, the k-means algorithm and its extensions are always influenced by initializations with a necessary number of clusters a priori. k-means algorithm is computationally expensive and the quality of the resulting clusters heavily depends on the selection of initial centroids. Clustering and type, K-means clustering algorithm, working of K-means clustering, Relation to other algorithms its primary topics in this paper.**

Keywords- Algorithm, Machine Learning, Data Analysis, Clustering, k-means clustering, Unsupervised learning.

I. Introduction

Advances in scientific data collection methods have resulted in the large-scale accumulation of promising data pertaining to diverse fields of science and technology. Owing to the development of novel techniques for generating and collecting data, the rate of growth of scientific databases has become tremendous. Hence it is practically impossible to extract useful information from them by using conventional database analysis techniques. Effective mining methods are absolutely essential to unearth implicit information from huge databases. Clustering is a way that classify the raw data reasonably and searches the hidden patterns that may exist in datasets. It is a process of grouping data objects into disjointed clusters so that the data in the same cluster are similar, yet data belonging to different cluster differ. The demand for organizing the sharp increasing data and learning valuable information from data, which makes clustering techniques are widely applied in many application areas such as artificial intelligence, biology, customer relationship management, data compression, data mining, information retrieval, image processing, machine learning, marketing, medicine, pattern recognition, psychology, statistics and so on.

Clustering is the process of partitioning a given set of objects into disjoint clusters. This is done in such a way that objects in the same cluster are similar while objects belonging to different clusters differ considerably, with respect to their attributes. The k-means algorithm is effective in producing clusters for many practical applications. But the computational complexity of the original k-means algorithm is very high, especially for large data sets. Moreover, this algorithm results in different types of clusters depending on the random choice of initial centroids. Several attempts were made by researchers for improving the performance of the k-means clustering algorithm. This paper deals with a method for improving the accuracy and efficiency of the k-means algorithm.

II. What Is Clusteing?

Clustering is a set of techniques used to partition data into groups, or clusters. Clusters are loosely defined as groups of data objects that are more similar to other objects in their cluster than they are to data objects in

other clusters. In practice, clustering helps identify two qualities of data:

1. Meaningfulness
2. Usefulness

Meaningful clusters expand domain knowledge. For example, in the medical field, researchers applied clustering to gene expression experiments. The clustering results identified groups of patients who respond differently to medical treatments.

Useful clusters, on the other hand, serve as an intermediate step in a data pipeline. For example, businesses use clustering for customer segmentation. The clustering results segment customers into groups with similar purchase histories, which businesses can then use to create targeted advertising campaigns.

Overview of Clustering Techniques

You can perform clustering using many different approaches so many, in fact, that there are entire categories of clustering algorithms. Each of these categories has its own unique strengths and weaknesses. This means that certain clustering algorithms will result in more natural cluster assignments depending on the input data.

Selecting an appropriate clustering algorithm for your dataset is often difficult due to the number of choices available. Some important factors that affect this decision include the characteristics of the clusters, the features of the dataset, the number of outliers, and the number of data objects.

You'll explore how these factors help determine which approach is most appropriate by looking at three popular categories of clustering algorithms:

1. Partitional clustering
2. Hierarchical clustering
3. Density-based clustering

It's worth reviewing these categories at a high level before jumping right into k-means. You'll learn the strengths and weaknesses of each category to provide context for how k-means fits into the landscape of clustering algorithms.

1.Partitional Clustering

Partitional clustering divides data objects into nonoverlapping groups. In other words, no object can be a member of more than one cluster, and every cluster must have at least one object.

These techniques require the user to specify the number of clusters, indicated by the variable k. Many partitional clustering algorithms work through an iterative process to assign subsets of data points into k clusters. Two examples of partitional clustering algorithms are k-means and k-medoids.

These algorithms are both nondeterministic, meaning they could produce different results from two separate runs even if the runs were based on the same input.

Partitional clustering methods have several strengths:

- They work well when clusters have a spherical shape.
- They're scalable with respect to algorithm complexity.

They also have several weaknesses:

- They're not well suited for clusters with complex shapes and different sizes.
- They break down when used with clusters of different densities.

2.Hierarchical Clustering

Hierarchical clustering determines cluster assignments by building a hierarchy. This is implemented by either a bottom-up or a top-down approach:

- Agglomerative clustering is the bottom-up approach. It merges the two points that are the most similar until all points have been merged into a single cluster.
- Divisive clustering is the top-down approach. It starts with all points as one cluster and splits the least similar clusters at each step until only single data points remain.

These methods produce a tree-based hierarchy of points called a dendrogram. Similar to partitional clustering, in hierarchical clustering the number of clusters (k) is often predetermined by the user. Clusters are assigned by cutting the dendrogram at a specified depth that results in k groups of smaller dendrograms.

Unlike many partitional clustering techniques, hierarchical clustering is a deterministic process, meaning cluster assignments won't change when you run an algorithm twice on the same input data.

The strengths of hierarchical clustering methods include the following:

- They often reveal the finer details about the relationships between data objects.
- They provide an interpretable dendrogram.

The weaknesses of hierarchical clustering methods include the following:

- They're computationally expensive with respect to algorithm complexity.
- They're sensitive to noise and outliers.

3.Density-Based Clustering

Density-based clustering determines cluster assignments based on the density of data points in a region. Clusters are assigned where there are high densities of data points separated by low-density regions.

Unlike the other clustering categories, this approach doesn't require the user to specify the number of clusters. Instead, there is a distance-based parameter that acts as a tuneable threshold. This threshold determines how close points must be to be considered a cluster member.

Examples of density-based clustering algorithms include Density-Based Spatial Clustering of Applications with Noise, or DBSCAN, and Ordering Points To Identify the Clustering Structure, or OPTICS.

The strengths of density-based clustering methods include the following:

- They excel at identifying clusters of nonspherical shapes.
- They're resistant to outliers.

The weaknesses of density-based clustering methods include the following:

- They aren't well suited for clustering in high-dimensional spaces.
- They have trouble identifying clusters of varying densities.

III. THE K-MEANS CLUSTERING ALGORITHM

The K-means clustering algorithm computes centroids and repeats until the optimal centroid is found. It is presumptively known how many clusters there are. It is also known as the flat clustering algorithm. The number of clusters found from data by the method is denoted by the letter 'K' in K-means.

In this method, data points are assigned to clusters in such a way that the sum of the squared distances between the data points and the centroid is as small as possible. It is essential to note that reduced diversity within clusters leads to more identical data points within the same cluster.

This section describes the original k-means clustering algorithm. The idea is to classify a given set of data into k number of disjoint clusters, where the value of k is fixed in advance. The algorithm consists of two separate phases: the first phase is to define k centroids, one for each cluster. The next phase is to take each point belonging to the given data set and associate it to the nearest centroid. Euclidean distance is generally considered to determine the distance between data points and the centroids. When all the points are included in some clusters, the first step is completed and an early grouping is done. At this point we need to recalculate the new centroids, as the inclusion of new points may lead to a change in the cluster centroids. Once we find k new centroids, a new binding is to be created between the same data points and the nearest new centroid, generating a loop. As a result of this loop, the k centroids may change their position in a step-by-step manner. Eventually, a situation will be reached where the centroids do not move anymore.

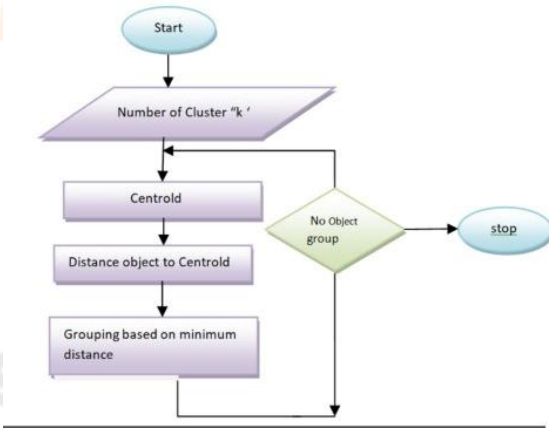


Fig. k-means clustering algorithm

RELATED WORK

The K-Means algorithm for grouping data: -

An approach for unsupervised learning called K-Means Clustering divides the unlabelled dataset into various clusters.

It is an iterative technique that separates the unlabelled dataset into k distinct clusters, with each dataset belonging to just one group with comparable qualities.

Two tasks are primarily carried out by the k-means clustering algorithm:

a method of iteration is used to get the optimal value for K centre points or centroids.

chooses the nearest k-centre for each data point. The data points that are close to a specific k-centre group together as a cluster.

One of the most straightforward unsupervised learning techniques to handle the well-known clustering problem is k-means. The process uses a predetermined number of clusters (let's assume k clusters) fixed apriority to categorise a given data set. To define k centres, one for each cluster, is the main notion. These centres should be strategically positioned because different locations yield various effects. The preferable option is to situate them as far apart from one another as you can. The following phase is connecting each point from a given data set to the closest centre.

The first step is finished and an early group age is finished when there are no points still open. Now that the clusters produced by the previous phase have their barycentre, we need to recalculate k new centroids. The same data set points must now be bound to the closest new centre once we have these k new centroids. There has been created a loop. This loop may cause the k centres to gradually shift positions until no more modifications are made, or, to put it another way, the centres stop moving altogether. The algorithm's final goal is to minimise the squared error function, which is represented by:

$$J(V) = \sum_{i=1}^{c} \sum_{j=1}^{c_i} (\|x_i - v_j\|)^2$$

where "xi - vj" denotes the distance in Euclidean space between xi and vj.

The quantity "ci" refers to the amount of data in the ith cluster

"c" represents the quantity of cluster centres.

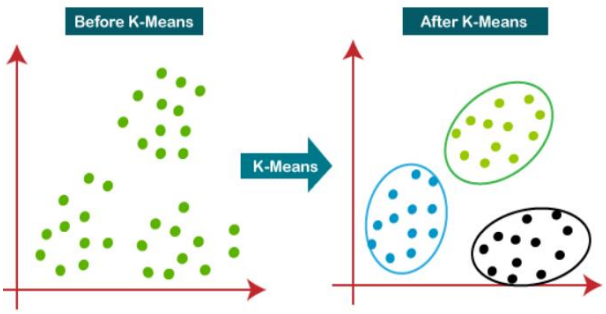


Fig. working of the K-means Clustering Algorithm

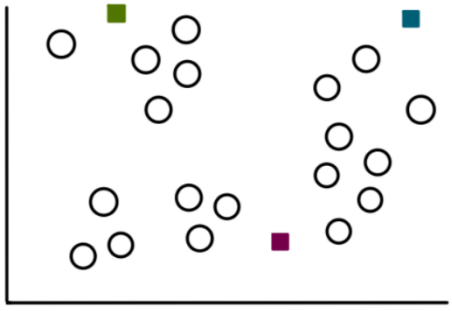
Working of K-Means Algorithm

Step 1: Select the Number of Clusters, k

The number of clusters we want to identify is the k in k-means clustering. In this case, since we assumed that there are 3 clusters, k = 3.

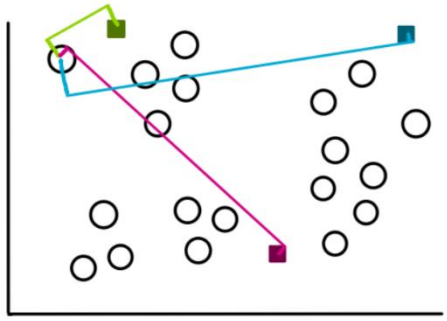
Step 2: Select k Points at Random

We start the process of finding clusters by selecting 3 random points (not necessarily our data points). These points will now act as centroids, or the center, of clusters that we are going to make:



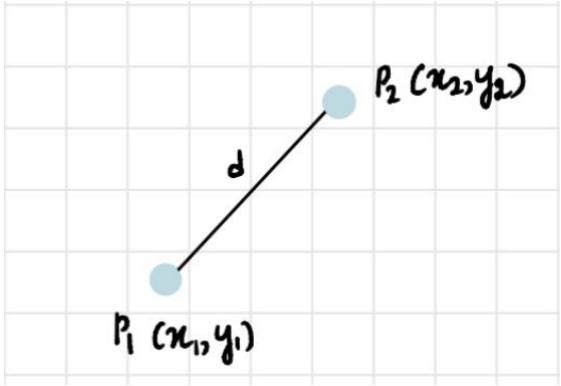
Step 3: Make k Clusters

To make the clusters, we start by measuring the distance from each data point to each of the 3 centroids. And we assign the points to the cluster closest to it. So for a sample point, the distances will look like this:



By just looking at it, we see that the distance from the point to the green centroid is the least, so we assign the point to the green cluster.

In two dimensions, the formula to find the distance between two points is:



$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Using the above formula, we repeat this process for the rest of the points and the clusters will look something like this:

Step 4: Compute New Centroid of Each Cluster

Now that we have our 3 clusters, we find the new centroids formed by each of them. For instance, the way we calculate the coordinates of the centroid of the blue cluster is:

$$(x', y') = \left(\frac{x_1+x_2+x_3}{3}, \frac{y_1+y_2+y_3}{3}\right)$$

where x1, x2, and x3 are the x-coordinates of each of the 3 points of the blue cluster. And y1, y2, and y3 are the y-coordinates of each of the 3 points of the blue cluster. We divide the sum of the coordinates by 3 because there are 3 data points in the blue cluster. Similarly, the coordinates of the centroids of the pink and green clusters are:

$$(x'', y'') = \left(\frac{x_1+\cdots+x_{11}}{11}, \frac{y_1+\cdots+y_{11}}{11}\right)$$

$$(x''', y''') = \left(\frac{x_1+\cdots+x_5}{5}, \frac{y_1+\cdots+y_5}{5}\right)$$

So, the new centroids look like this:

Step 5: Assess the Quality of Each Cluster

Since k-means can't see the clustering as we can, it measures the quality by finding the variation within all the clusters. The basic idea behind k-means clustering is defining clusters so that the within-cluster variation is minimized. We calculate something called Within-Cluster Sum of Squares (WCSS) to quantify this variance:

$$\text{WCSS} = \sum_{C_k}^{C_n}\left(\sum_{d_i in C_i}^{d_m} distance(d_i, C_k)^2\right)$$

Where,
C is the cluster centroids and d is the data point in each Cluster.

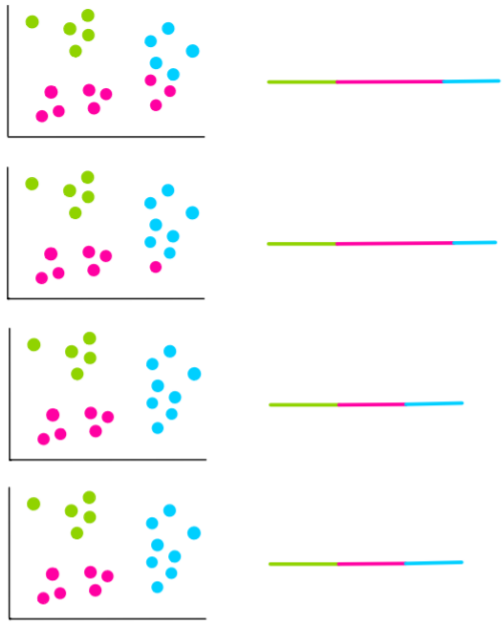
This is a scary-looking formula, so if you don't really understand it that's okay; just try to understand the intuition behind it.

For simplification purposes, let's represent the variation visually like this:

Step 6: Repeat Steps 3–5

Once we have previous clusters and the variation stored, we start all over. But only this time we use the centroids we calculated previously to - make 3 new clusters, recalculate the center of the new clusters, and calculate the sum of the variation within all the clusters.

Let's suppose the next 4 iterations look like this:



From the last two iterations, we see that the clusters haven't changed. This means that the algorithm has converged and we stop the clustering process. We then choose the clusters with the least WCSS. This also happens to be those of the last two iterations. So, they are going to be our final clusters.

How do we choose k?

In our example, we conveniently knew that we need 3 clusters. But what if we don't know how many clusters we have, then how do we choose k?

In this case, we try multiple k values and calculate the WCSS.

k=1:

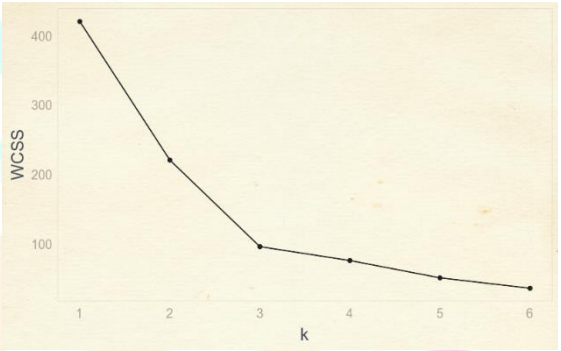
k=2:

k=3:

k=4:

We notice that each time we add a new cluster, the total variation within each cluster is smaller than before. And when there is only one point per cluster, the variation = 0.

So, we need to use something called an elbow plot to find the best k. It plots the WCSS against the number of clusters or k.



This is called an elbow plot because we can find an optimal k value by finding the "elbow" of the plot, which is at 3. Until 3 you can notice a huge reduction in variation, but after that, the variation doesn't go down as quickly.And that's about it. A simple, but effective clustering algorithm!

I. Implementation of the k-means method in Python

1. Data processing is the first stage; to begin, we must import the data, and then we must import the dataset. The independent variable
2. must then be extracted. The ideal number clusters
3. will be located in the second stage using the elbow approach. The K-means algorithm must be trained using the training data set
4. in the third stage. The fourth phase is seeing the cluster.

Relation to other algorithms

1. Gaussian mixture model

The slow "standard algorithm" for k-means clustering, and its associated expectation-maximization algorithm, is a special case of a Gaussian mixture model, specifically, the limiting case when fixing all covariances to be diagonal, equal and have infinitesimal small variance. Instead of small variances, a hard cluster assignment can also be used to show another equivalence of k-means clustering to a special case of "hard" Gaussian mixture modelling. This does

not mean that it is efficient to use Gaussian mixture modelling to compute k-means, but just that there is a theoretical relationship, and that Gaussian mixture modelling can be interpreted as a generalization of k-means; on the contrary, it has been suggested to use k-means clustering to find starting points for Gaussian mixture modelling on difficult data.

2. k-SVD

Another generalization of the k-means algorithm is the k-SVD algorithm, which estimates data points as a sparse linear combination of "codebook vectors". k-means corresponds to the special case of using a single codebook vector, with a weight of

3. Principal component analysis

The relaxed solution of k-means clustering, specified by the cluster indicators, is given by principal component analysis The intuition is that k-means describe spherically shaped (ball-like) clusters. If the data has 2 clusters, the line connecting the two centroids is the best 1-dimensional projection direction, which is also the first PCA direction. Cutting the line at the center of mass separates the clusters (this is the continuous relaxation of the discrete cluster indicator). If the data have three clusters, the 2-dimensional plane spanned by three cluster centroids is the best 2-D projection. This plane is also defined by the first two PCA dimensions. Well-separated clusters are effectively modelled by ball-shaped clusters and thus discovered by k-means. Non-ball-shaped clusters are hard to separate when they are close. For example, two half-moon shaped clusters intertwined in space do not separate well when projected onto PCA subspace. k-means should not be expected to do well on this data.[54] It is straightforward to produce counterexamples to the statement that the cluster centroid subspace is spanned by the principal directions.

4. Mean shift clustering

Basic mean shift clustering algorithms maintain a set of data points the same size as the input data set. Initially, this set is copied from the input set. Then this set is iteratively replaced by the mean of those points in the set that are within a given distance of that point. By contrast, k-means restricts this updated set to k points usually much less than the number of points in the input data set, and replaces each point in this set by the mean of all points in the input set that are closer to that point than any other (e.g. within the Voronoi partition of each updating point). A mean shift algorithm that is similar then to k-means, called likelihood mean shift, replaces the set of points undergoing replacement by the mean of all points in the input set that are within a given distance of the changing set.[56] One of the advantages of mean shift over k-means is that the number of clusters is not pre-specified, because mean shift is likely to find only a few clusters if only a small number exist. However, mean shift can be much slower than k-means, and still requires selection of a bandwidth parameter. Mean shift has soft variants.

5. Independent component analysis

Under sparsity assumptions and when input data is pre-processed with the whitening transformation, k-means produces the solution to the linear independent component analysis (ICA) task. This aids in explaining the successful application of k-means to feature learning.

6. Bilateral filtering

k-means implicitly assumes that the ordering of the input data set does not matter. The bilateral filter is similar to k-means and mean shift in that it maintains a set of data points that are iteratively replaced by means. However, the bilateral filter restricts the calculation of the (kernel weighted) mean to include only points that are close in the ordering of the input data.[56] This makes it applicable to problems such as image denoising, where the spatial arrangement of pixels in an image is of critical importance.

IV. Conclusion

The k-means algorithm is widely used for clustering large sets of data. But the standard algorithm do not always guarantee good results as the accuracy of the final clusters depend on the selection of initial centroids. Moreover, the computational complexity of the standard algorithm is objectionably high owing to the need to reassign the data points a number of times, during every iteration of the loop. This paper presents an enhanced k-means algorithm which combines a systematic method for finding initial centroids and an efficient way for assigning data points to clusters.

The proposed U-k-means algorithm uses the number of points as the initial number of clusters for solving the initialization problem. During iterations, the U-k-means algorithm will discard extra clusters, and then an optimal number of clusters can be automatically found according to the structure of data. The advantages of U-k-means are free of initializations and parameters that also robust to different cluster volumes and shapes with automatically finding the number of clusters. The proposed U-k-means algorithm was performed on several synthetic and real data sets and also compared with most existing algorithms. The results actually demonstrate the superiority of the U-k-means clustering algorithm.

V. References

1. K. A. Abdul Nazeer, M. P. Sebastian - Improving the Accuracy and Efficiency of the k-means Clustering Algorithm [2009 Vol I WCE 2009, July 1 - 3, 2009, London, U.K.]
2. Zhang Chen , Xia Shixiong - K-means Clustering Algorithm with improved Initial Center [Second International Workshop on Knowledge Discovery and Data Mining]
3. KRISTINA P. SINAGA AND MIIN-SHEN YANG , K-Means Clustering Algorithm
   [Received April 5, 2020, accepted April 16, 2020, date of publication April 20, 2020, date of current version May 13, 2020]
4. https://en.wikipedia.org/wiki/K-means_clustering
5. https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/
6. https://www.javatpoint.com/k-means-clustering-algorithm-in-machine-learning
7. https://realpython.com/k-means-clustering-python/
8. Juntao Wang, Xiaolong Su- An improved K-Means clustering algorithm, School of Computer Science and Technology China University of Mining & Technology Xuzhou, China