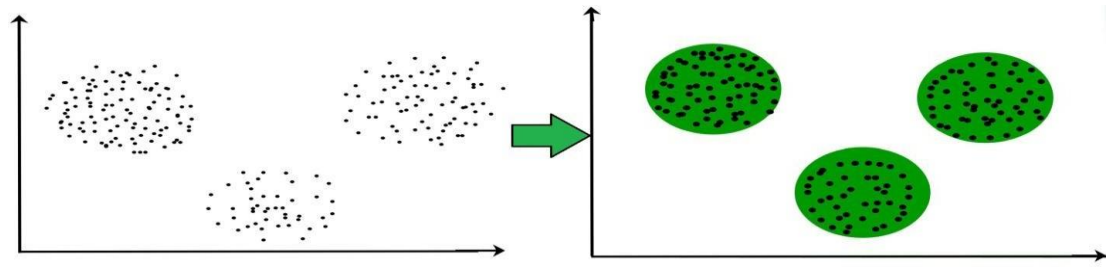
UNIT-V

CLUSTERING:

The task of grouping data points based on their similarity with each other is called Clustering or Cluster Analysis. This method is defined under the branch of unsupervised learning, which aims at gaining insights from unlabeled data points.

Think of it as you have a dataset of customers shopping habits. Clustering can help you group customers with similar purchasing behaviors, which can then be used for targeted marketing, product recommendations, or customer segmentation



Types of Clustering

Broadly speaking, there are 2 types of clustering that can be performed to group similar data points:

- **Hard Clustering:** In this type of clustering, each data point belongs to a cluster completely or not. For example, Let's say there are 4 data point and we have to cluster them into 2 clusters. So each data point will either belong to cluster 1 or cluster 2.

Data Points	Clusters
A	C1
B	C2
C	C2
D	C1

- **Soft Clustering:** In this type of clustering, instead of assigning each data point into a separate cluster, a probability or likelihood of that point being that cluster is evaluated. For example, Let's say there are 4 data point and we have to cluster them into 2 clusters. So we will be evaluating a probability of a data point belonging to both clusters. This probability is calculated for all data points.

Data Points	Probability of C1	Probability of C2
A	0.91	0.09
B	0.3	0.7
C	0.17	0.83
D	1	0

PARTITIONING OF DATA:

Using data partitioning techniques, a huge dataset can be divided into smaller, easier-to-manage portions. These techniques are applied in a variety of fields, including distributed systems, parallel computing, and database administration.

Why do we need Data Partitioning?

Data partitioning is essential for several reasons:

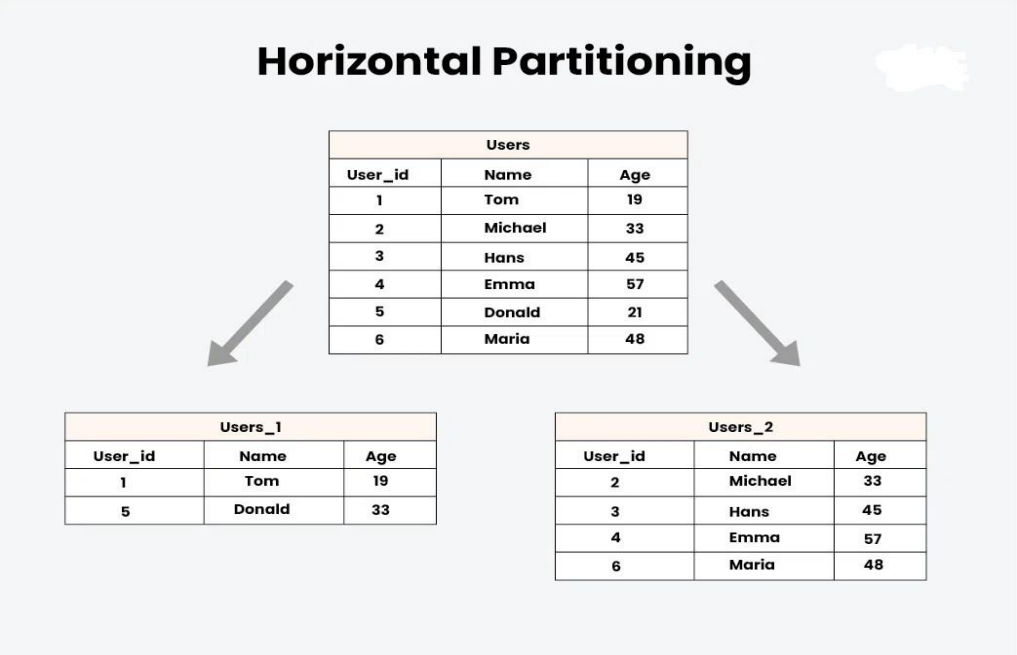
- **Performance Improvement**: By breaking data into smaller segments, systems can access only the relevant partitions, leading to faster query execution and reduced load times.
- **Scalability**: As datasets grow, partitioning allows for easier management and distribution across multiple servers or storage systems, enabling horizontal scaling.
- **Efficient Resource Utilization**: It helps optimize the use of resources by allowing systems to focus processing power on specific partitions rather than the entire dataset.
- **Enhanced Manageability**: Smaller partitions are easier to back up, restore, and maintain, facilitating better data governance and maintenance practices.

Methods of Data Partitioning

Below are the main methods of Data Partitioning:

1. Horizontal Partitioning/Sharding

In this technique, the dataset is divided based on rows or records. Each partition contains a subset of rows, and the partitions are typically distributed across multiple servers or storage devices. Horizontal partitioning is often used in distributed databases or systems to improve parallelism and enable load balancing. Horizontal Partitioning



- **Advantages of Horizontal Partitioning/Sharding**

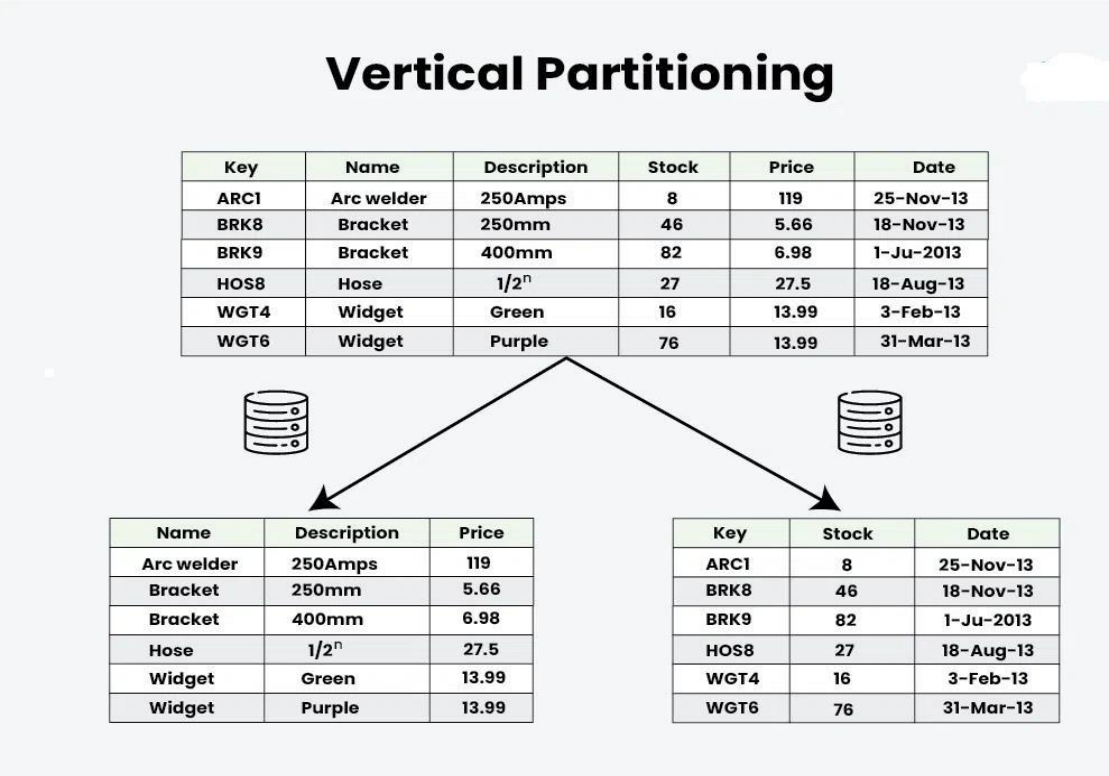
  - **Greater scalability:** Large datasets can be processed in parallel thanks to horizontal partitioning, which divides data among multiple computers or storage devices.
  - **Load balancing:** Data partitioning allows for the equitable distribution of workload across multiple nodes, preventing bottlenecks and improving system performance.
  - **Data separation:** Data isolation and fault tolerance are enhanced since each partition can be controlled separately. Even if one partition fails, the others can continue to function.

- **Disadvantages of Horizontal Partitioning/Sharding**

  - **Join operations:** Horizontal partitioning can make join operations across multiple partitions more complex and potentially slower, as data needs to be fetched from different nodes.
  - **Data skew:** If the distribution of data is uneven or if some partitions receive more queries or updates than others, it can result in data skew, impacting performance and load balancing.

2. Vertical Partitioning

Vertical partitioning separates the dataset according to columns or attributes, in contrast to horizontal partitioning. Each partition in this method has a subset of columns for every row. When certain columns are visited more frequently than others or when different columns have different access patterns, vertical partitioning might be helpful.



MATRIX FACTORIZATION | CLUSTERING OF PATTERNS:

This mathematical model helps the system split an entity into multiple smaller entries, through an ordered rectangular array of numbers or functions, to discover the features or information underlying the interactions between users and items. his approach recommends items based on user preferences. It matches the requirement, considering the past actions of the user, patterns detected, or any explicit feedback provided by the user, and accordingly, makes a recommendation.

Example: If you prefer the chocolate flavor and purchase a chocolate ice cream, the next time you raise a query, the system shall scan for options related to chocolate, and then, recommend you to try a chocolate cake.

How does the System make recommendations

Let us take an example. To purchase a car, in addition to the brand name, people check for features available in the car, most common ones being safety, mileage, or aesthetic value. Few buyers consider the automatic gearbox, while others opt for a combination of two or more features. To understand this concept, let us consider a two-dimensional vector with the features of safety and mileage.

Car features	Car A	Car B	Car C	Car D
Safety	4	1	2	1
Mileage	1	4	2	2

Individual preferences	*Safety*	*Mileage*				
Person A	1	0	4	1	2	1
Person B	0	1	1	4	2	?
Person C	1	0	4	1	?	1
Person D	1	1	?	5	4	3

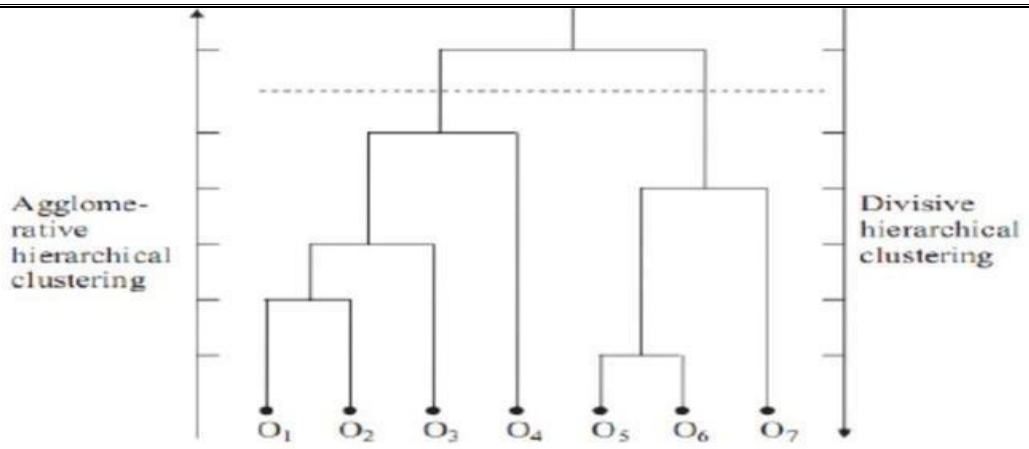
Divisive Clustering:

Divisive clustering **starts with one, all-inclusive cluster**. At each step, it **splits a cluster until each cluster contains a point** (or there are k clusters).

Agglomerative Clustering:

Clustering is the broad set of techniques for finding subgroups or clusters on the basis of characterization of objects within dataset such that objects with groups are similar but different from the object of other groups. Primary guideline of clustering is that data inside a cluster should be very similar to each other but very different from those outside clusters. There are different types of clustering techniques like Partitioning Methods, Hierarchical Methods and Density Based Methods.

Method	Characteristics
Partitioning Method	• Uses mean/medoid to represent cluster Centre • Adopts distance-based approach to refine clusters • Finds mutually exclusive clusters of spherical / nearly spherical shape • Effective for datasets of small - medium age
Hierarchical Method	• Creates tree-like structure through decomposition • Uses distance between nearest / furthest points in neighboring clusters for refinement • Error can't be corrected at subsequent levels
Density Based Method	• Useful for identifying arbitrarily shaped clusters • May filter out outliers



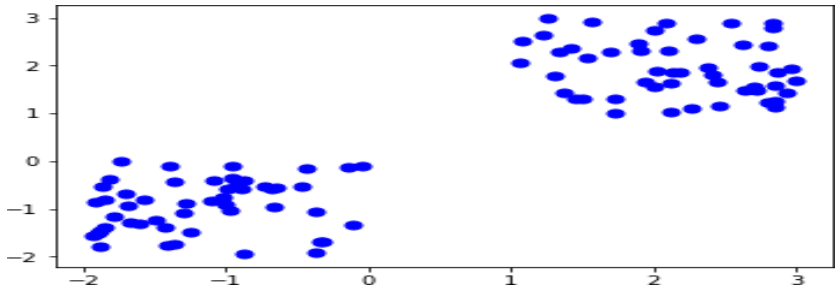
Partitional Clustering:

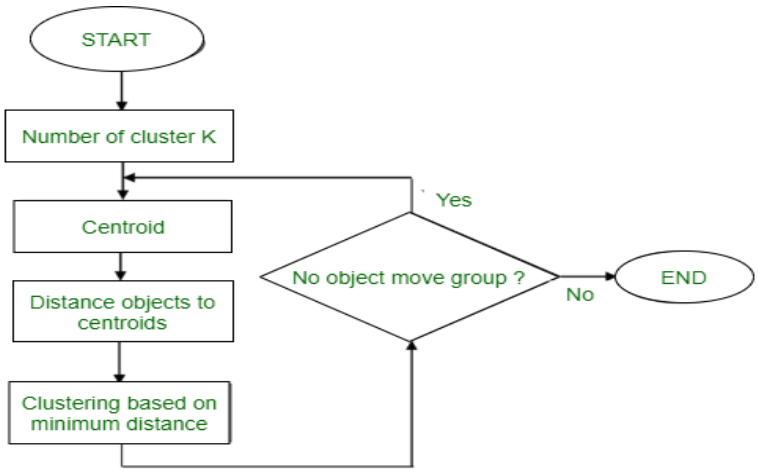
Partitional clustering, is a type of clustering algorithm that divides a dataset into a predefined number of clusters (k).

Method:

1. Randomly assign K objects from the dataset(D) as cluster centres(C)
2. (Re) Assign each object to which object is most similar based upon mean values.
3. Update Cluster means, i.e., Recalculate the mean of each cluster with the updated values.
4. Repeat Step 2 until no change occurs.

.





Example: Suppose we want to group the visitors to a website using just their age as follows:

16, 16, 17, 20, 20, 21, 21, 22, 23, 29, 36, 41, 42, 43, 44, 45, 61, 62, 66

Initial Cluster:

K=2
Centroid(C1) = 16 [16]
Centroid(C2) = 22 [22]

Note: These two points are chosen randomly from the dataset.

Iteration-1:

C1 = 16.33 [16, 16, 17]
C2 = 37.25 [20, 20, 21, 21, 22, 23, 29, 36, 41, 42, 43, 44, 45, 61, 62, 66]

Iteration-2:

C1 = 19.55 [16, 16, 17, 20, 20, 21, 21, 22, 23]
C2 = 46.90 [29, 36, 41, 42, 43, 44, 45, 61, 62, 66]

Iteration-3:

C1 = 20.50 [16, 16, 17, 20, 20, 21, 21, 22, 23, 29]
C2 = 48.89 [36, 41, 42, 43, 44, 45, 61, 62, 66]

Iteration-4:

C1 = 20.50 [16, 16, 17, 20, 20, 21, 21, 22, 23, 29]
C2 = 48.89 [36, 41, 42, 43, 44, 45, 61, 62, 66]

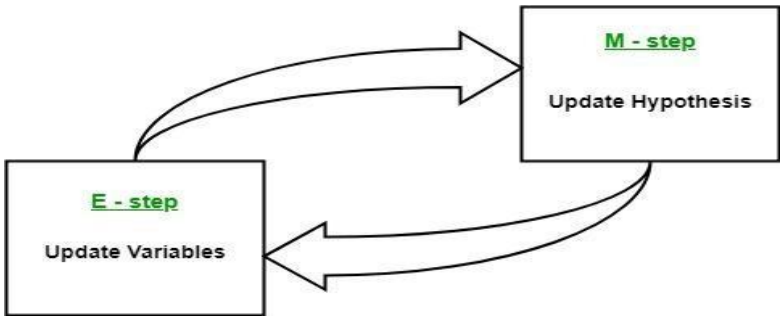
No change Between Iteration 3 and 4, so we stop. Therefore we get the clusters **(16-29)** and **(36-66)** as 2 clusters we get using K Mean Algorithm.

Expectation Maximization-Based Clustering:

Expectation-Maximization (EM) algorithm is a **iterative method** used in unsupervised machine learning to find unknown values in statistical models. **It helps to find the best values for unknown parameters especially when some data is missing or hidden.** It works in two steps:

- **E-step (Expectation Step):** Estimates missing or hidden values using current parameter estimates.
- **M-step (Maximization Step):** Updates model parameters to maximize the likelihood based on the estimated values from the E-step.

This process repeats until the model reaches a stable solution as it improve accuracy with each iteration. It is widely used in clustering like **Gaussian Mixture Models** and handling missing data.



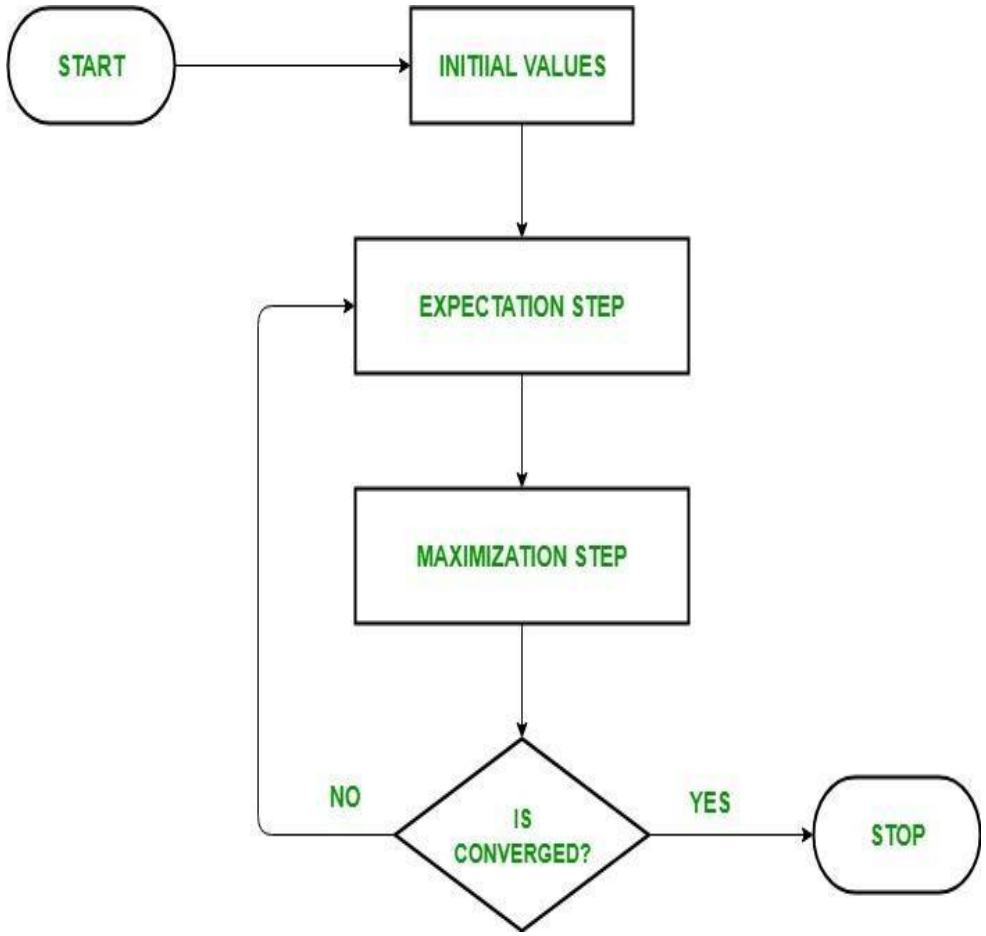
Key Terms in Expectation-Maximization (EM) Algorithm

Lets understand about some of the most commonly used key terms in the Expectation-Maximization (EM) Algorithm:

- **Latent Variables:** These are hidden parts of the data that we can't see directly but they still affect what we do see. We try to guess their values using the visible data.
- **Likelihood:** This refers to the probability of seeing the data we have based on certain assumptions or parameters. The EM algorithm tries to find the best parameters that make the data most likely.
- **Log-Likelihood:** This is just the natural log of the likelihood function. It's used to make calculations easier and measure how well the model fits the data. The EM algorithm tries to maximize the log-likelihood to improve the model fit.
- **Maximum Likelihood Estimation (MLE):** This is a method to find the best values for a model's settings called parameters. It looks for the values that make the data we observed most likely to happen.
- **Posterior Probability:** In Bayesian methods this is the probability of the parameters given both prior knowledge and the observed data. In EM it helps estimate the "best" parameters when there's uncertainty about the data.
- **Expectation (E) Step:** In this step the algorithm estimates the missing or hidden information (latent variables) based on the observed data and current parameters. It calculates probabilities for the hidden values given what we can see.
- **Maximization (M) Step:** This step update the parameters by finding the values that maximize the likelihood based on the estimates from the E-step.
- **Convergence:** Convergence happens when the algorithm has reached a stable point. This is checked by seeing if the changes in the model's parameters or the log-likelihood are small enough to stop the process.

Working of Expectation-Maximization (EM) Algorithm

So far, we've discussed the key terms in the EM algorithm. Now, let's dive into how the EM algorithm works. Here's a step-by-step breakdown of the process:



1. **Initialization:** The algorithm starts with initial parameter values and assumes the observed data comes from a specific model.

2. E-Step (Expectation Step):

- Find the missing or hidden data based on the current parameters.
- Calculate the posterior probability of each latent variable based on the observed data.
- Compute the log-likelihood of the observed data using the current parameter estimates.

3. M-Step (Maximization Step):

- Update the model parameters by maximize the log-likelihood.
- The better the model the higher this value.

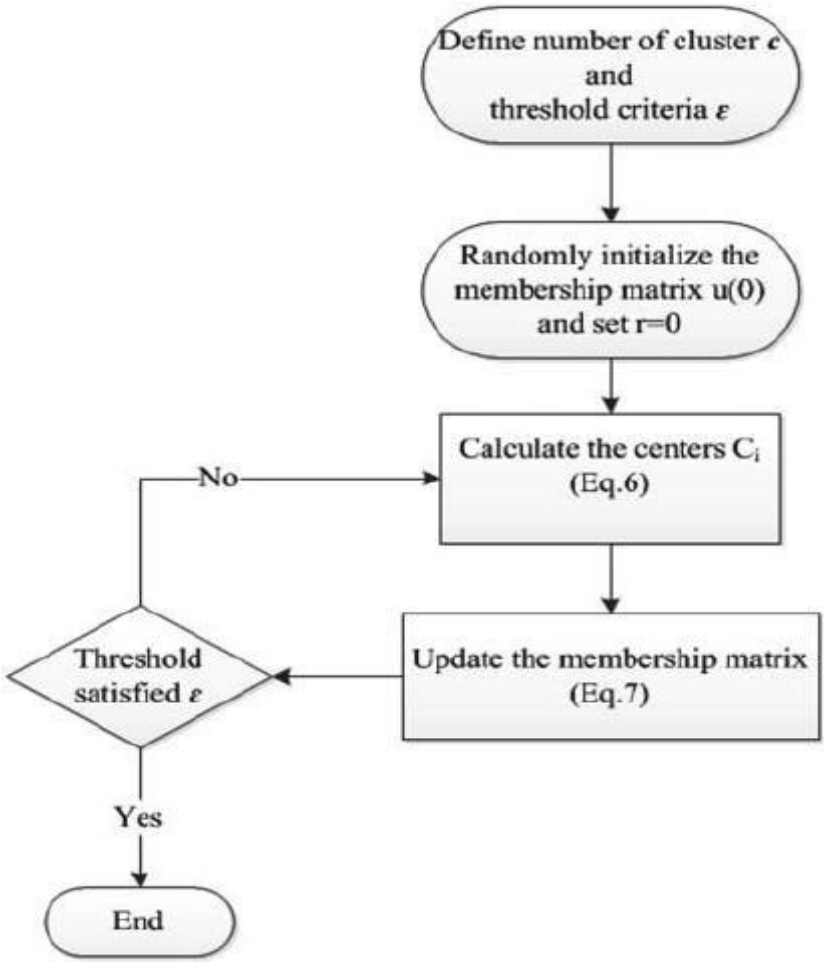
4. Convergence:

- Check if the model parameters are stable and converging.
- If the changes in log-likelihood or parameters are below a set threshold, stop. If not repeat the E-step and M-step until convergence is reached

Fuzzy C-Means Clustering:

It is an unsupervised clustering algorithm that permits us to build a fuzzy partition from data. The algorithm depends on a parameter m which corresponds to the degree of fuzziness of the solution. Large values of m will blur the classes and all elements tend to belong to all clusters. The solutions of the optimization problem depend on the parameter m. That is, different selections of m will typically lead to different partitions. Given below is a gif that shows the effect of the selection of m obtained from the fuzzy c-means.

Steps:



1. **Assume** a fixed number of clusters *k.*
2. **Initialization**: Randomly initialize the k-means μk associated with the clusters and compute the probability that each data point xi is a member of a given cluster *k*, P(point xi has label k|xi, k).

3. **Iteration**: Recalculate the centroid of the cluster as the weighted centroid given the probabilities of membership of all data points xi:

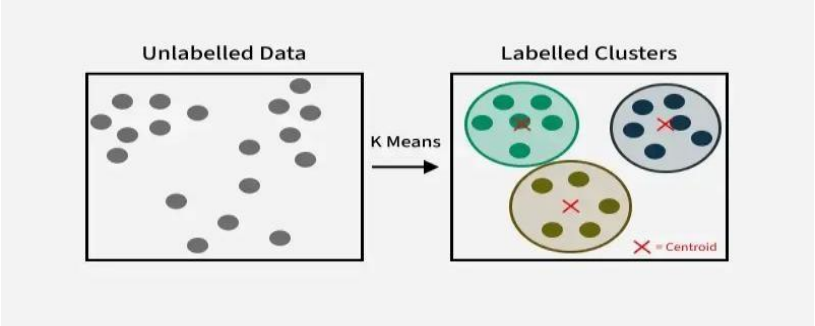
$$\mu_k(n+1) = \frac{\sum_{x_i \epsilon k} x_i * P(\mu_k|x_i)^b}{\sum_{x_i \epsilon k} P(\mu_k|x_i)^b}$$

4. **Termination**: Iterate until convergence or until a user-specified number of iterations has been reached (the iteration may be trapped at some local maxima or minima).

K-Means Clustering Algorithm:

K-Means Clustering is an Unsupervised Machine Learning algorithm which groups unlabeled dataset into different clusters. It is used to organize data into **groups based on their similarity**.

We are given a data set of items with certain features and values for these features like a vector. The task is to categorize those items into groups. To achieve this we will use the K-means algorithm. 'K' in the name of the algorithm represents the number of groups/clusters we want to classify our items into.



1. First we randomly initialize k points called means or cluster centroids.
2. We categorize each item to its closest mean and we update the mean's coordinates, which are the averages of the items categorized in that cluster so far.
3. We repeat the process for a given number of iterations and at the end, we have our clusters.

The "points" mentioned above are called means because they are the mean values of the items categorized in them. To initialize these means, we have a lot of options. An intuitive method is to initialize the means at random items in the data set. Another method is to initialize the means at random values between the boundaries of the data set. For example for a feature *x* the items have values in [0,3] we will initialize the means with values for x at [0,3].

Rough Clustering

Rough clustering is a data mining technique that handles overlapping clusters by using rough set theory, which represents clusters using lower and upper approximations instead of a single, definitive boundary. This allows objects on the edge of a cluster to potentially belong to multiple clusters, creating a more flexible and realistic representation of data than hard clustering methods like K-Means

Rough clustering, such as the Rough K-Means (RKM) algorithm, assigns objects to clusters based on their relationship to cluster centroids.

Objects are divided into three groups:

Lower approximation: Objects definitely belonging to a cluster.

Upper approximation: Objects that might belong to a cluster.

Boundary region: Objects in the upper approximation that are not in the lower approximation, meaning they could potentially belong to multiple clusters.

This process is iterative, with cluster centroids being recomputed after each assignment until the assignments stabilize.

Advantages

Handles uncertainty: It can deal with the "inseparability of the border of clusters," where objects are not clearly defined as belonging to one group or another.

Flexibility: Allows for overlapping cluster boundaries, which is more realistic for many datasets than methods that force every object into a single cluster.

Applications

Image segmentation: Helps segment images with overlapping regions, which is a problem for traditional algorithms like K-Means.

Data mining: Extracts more nuanced knowledge by providing a more realistic representation of clusters and their relationships.

Pattern recognition: Can be used in various pattern recognition applications where data may have unclear boundaries.

The Rough K-Means clustering algorithm

The Rough K-Means clustering algorithm is an extension of the traditional K-Means algorithm that incorporates concepts from rough set theory to handle uncertainty and imprecision in data, particularly regarding cluster boundaries.

Key Idea:

Instead of assigning each data point to a single cluster, Rough K-Means assigns each data point to a lower approximation (certainly belonging to a cluster) and an upper approximation (possibly belonging to a cluster). The difference between these two approximations forms the boundary region, which represents data points whose cluster membership is ambiguous.

Rough Steps of the Algorithm:

- **Initialization:**
  - Choose the number of clusters, K.
  - Randomly select K initial cluster centroids.
- **Assignment Phase (Rough Set Approach):**
  - For each data point:
  - Calculate its distance to all K centroids.
  - **Lower Approximation:** Assign the data point to the lower approximation of a cluster if its distance to that cluster's centroid is significantly smaller than its distance to any other centroid. This signifies a clear membership.
  - **Upper Approximation:** Assign the data point to the upper approximation of a cluster if its distance to that cluster's centroid is within a certain threshold of its distance to other centroids. This suggests a possible membership, including the boundary region.
  - The boundary region is comprised of data points that are part of the upper approximation of multiple clusters but not definitively part of any single lower approximation.

Advantages:

- Handles uncertainty in cluster boundaries more effectively than traditional K-Means.
- Provides a more nuanced understanding of cluster membership by identifying ambiguous data points.

Disadvantages:

- More complex to implement than standard K-Means.
- Requires additional parameters (e.g., thresholds for defining lower/upper approximations) which can be challenging to tune.

.

Spectral Clustering

Spectral Clustering is a variant of the clustering algorithm that uses the connectivity between the data points to form the clustering. It uses eigenvalues and eigenvectors of the data matrix to forecast the data into lower dimensions space to cluster the data points. It is based on the idea of a graph representation of data where the data point are represented as nodes and the similarity between the data points are represented by an edge.

Steps performed for spectral Clustering

Building the Similarity Graph Of The Data: This step builds the Similarity Graph in the form of an adjacency matrix which is represented by A. The adjacency matrix can be built in the following manners:

Epsilon-neighborhood Graph: A parameter epsilon is fixed beforehand. Then, each point is connected to all the points which lie in its epsilon-radius. If all the distances between any two points are similar in scale then typically the weights of the edges ie the distance between the two points are not stored since they do not provide any additional information. Thus, in this case, the graph built is an undirected and unweighted graph.

K-Nearest Neighbors A parameter k is fixed beforehand. Then, for two vertices u and v, an edge is directed from u to v only if v is among the k-nearest neighbors of u. Note that this leads to the formation of a weighted and directed graph because it is not always the case that for each u having v as one of the k-nearest neighbors, it will be the same case for v having u among its k-nearest neighbors. To make this graph undirected, one of the following approaches is followed: -

1.Direct an edge from u to v and from v to u if either v is among the k-nearest neighbors of u OR u is among the k-nearest neighbors of v.

2.Direct an edge from u to v and from v to u if v is among the k-nearest neighbors of u AND u is among the k-nearest neighbors of v.

3.Fully-Connected Graph: To build this graph, each point is connected with an undirected edge-weighted by the distance between the two points to every other point. Since this approach is used to model the local neighborhood relationships thus typically the Gaussian similarity metric is used to calculate the distance.

Projecting the data onto a lower Dimensional Space: This step is done to account for the possibility that members of the same cluster may be far away in the given dimensional space. Thus the dimensional space is reduced so that those points are closer in the reduced dimensional space and thus can be clustered together by a traditional clustering algorithm. It is done by computing the Graph Laplacian Matrix.