Unit-III

Clustering in Machine Learning

Clustering or cluster analysis is a machine learning technique, which groups the unlabelled dataset. It can be defined as **"A way of grouping the data points into different clusters, consisting of similar data points. The objects with the possible similarities remain in a group that has less or no similarities with another group."**

It does it by finding some similar patterns in the unlabelled dataset such as shape, size, color, behavior, etc., and divides them as per the presence and absence of those similar patterns.

It is an unsupervised learning method, hence no supervision is provided to the algorithm, and it deals with the unlabeled dataset.

After applying this clustering technique, each cluster or group is provided with a cluster-ID. ML system can use this id to simplify the processing of large and complex datasets.

The clustering technique is commonly used for **statistical data analysis.**

Clustering is somewhere similar to the classification algorithm, but the difference is the type of dataset that we are using. In classification, we work with the labeled data set, whereas in clustering, we work with the unlabelled dataset.

Example: Let's understand the clustering technique with the real-world example of Mall: When we visit any shopping mall, we can observe that the things with similar usage are grouped together. Such as the t-shirts are grouped in one section, and trousers are at other sections, similarly, at vegetable sections, apples, bananas, Mangoes, etc., are grouped in separate sections, so that we can easily find out the things. The clustering technique also works in the same way. Other examples of clustering are grouping documents according to the topic.

Similarity Measures A similarity measure can be defined as the distance between various data points. While, similarity is an amount that reflects the strength of relationship between two data items, dissimilarity deals with the measurement of divergence between two data items.

The similarity is subjective and is highly dependent on the domain and application.

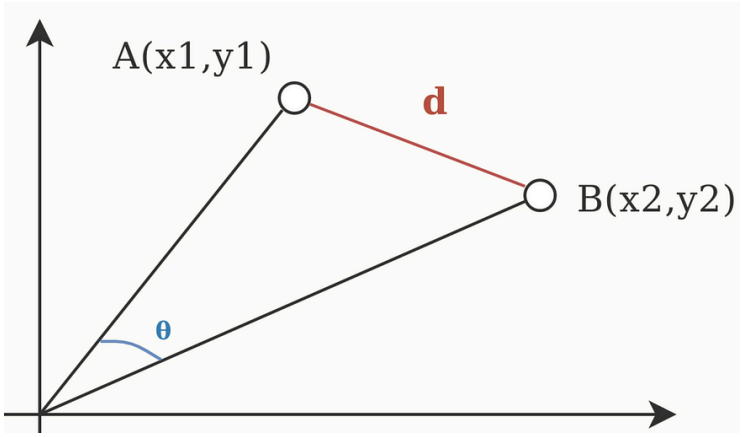
For example, two fruits are similar because of color or size or taste. Special care should be taken when calculating distance across dimensions/features that are unrelated. The relative values of each element must be normalized, or one feature could end up dominating the distance calculation.

There are lots of similarity distance measures. But here we will look into 5 most important measures

1)Cosine Similarity:

Cosine similarity is a metric used to measure how similar the documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. The cosine similarity is advantageous because even if the two similar documents are far apart by the Euclidean distance (due to the size of the document), chances are they may still be oriented closer together. The smaller the angle, higher the cosine similarity.

Formula:



$$Cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \, \|\vec{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2} \sqrt{\sum_1^n b_i^2}}$$

2) Manhattan distance:

Manhattan distance is a metric in which the distance between two points is the sum of the absolute differences of their Cartesian coordinates. In a simple way of saying it is the total sum of the difference between the x-coordinates and y-coordinates.

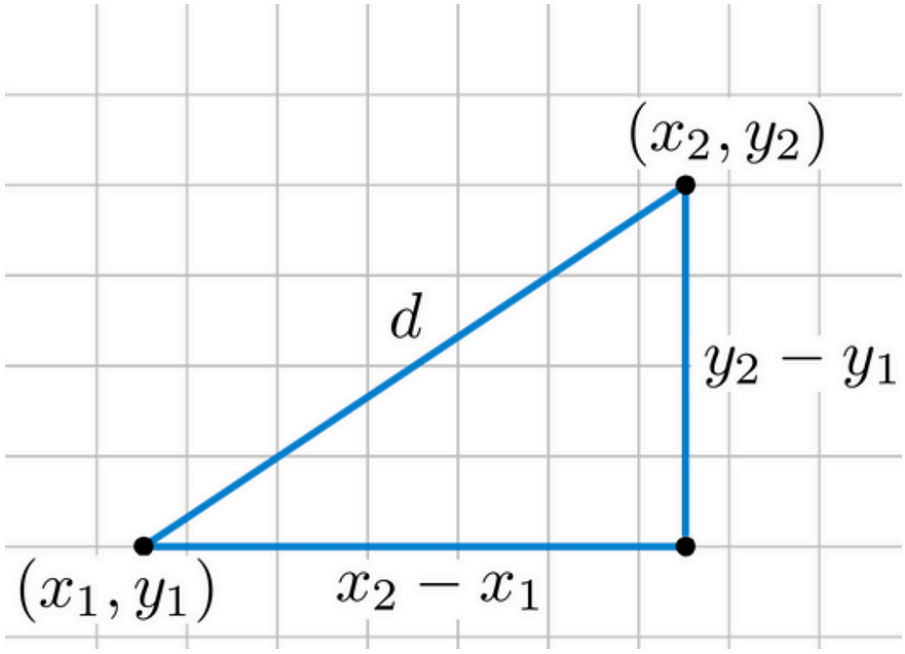
Formula: In a plane with p1 at (x1, y1) and p2 at (x2, y2)

$$|x_1 - x_2| + |y_1 - y_2|$$

3) Euclidean distance:

The Euclidean distance between two points in either the plane or 3-dimensional space measures the length of a segment connecting the two points. It is the most obvious way of representing distance between two points.

The Pythagorean Theorem can be used to calculate the distance between two points, as shown in the figure below.



Formula: If the points (x1,y1)(x1,y1) and (x2,y2)(x2,y2) are in 2-dimensional space, then the Euclidean distance between them is

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

4) Jaccard similarity:

The Jaccard index, also known as Intersection over Union and the Jaccard similarity coefficient is a statistic used for gauging the similarity and diversity of sample sets. The Jaccard coefficient measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets.

Formula:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

5) Minkowski distance

Minkowski distance is a generalisation of the Euclidean and Manhattan distances.

Formula: The Minkowski distance of order p between two points is defined as

$X = (x_1, x_2, \ldots, x_n)$ and $Y = (y_1, y_2, \ldots, y_n) \in \mathbb{R}^n$

$$D(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$$

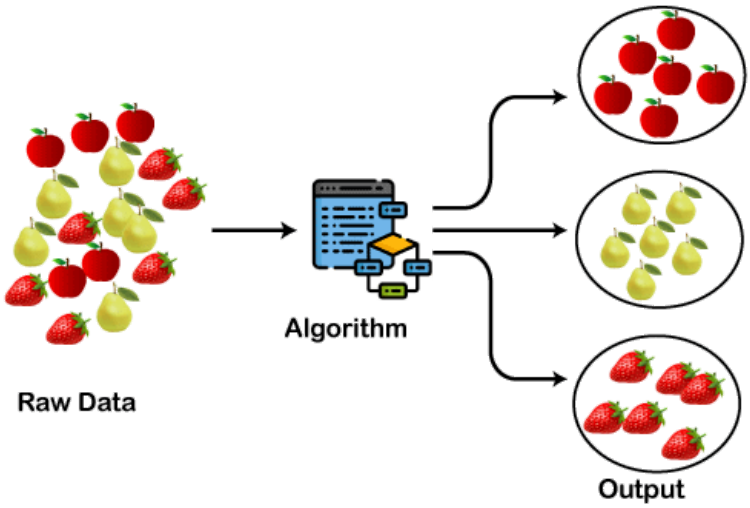
The clustering technique can be widely used in various tasks. Some most common uses of this technique are:

- Market Segmentation

- Statistical data analysis
- Social network analysis
- Image segmentation
- Anomaly detection, etc.

Apart from these general usages, it is used by the Amazon in its recommendation system to provide the recommendations as per the past search of products. Netflix also uses this technique to recommend the movies and web-series to its users as per the watch history.

The below diagram explains the working of the clustering algorithm. We can see the different fruits are divided into several groups with similar properties.



Types of Clustering Methods

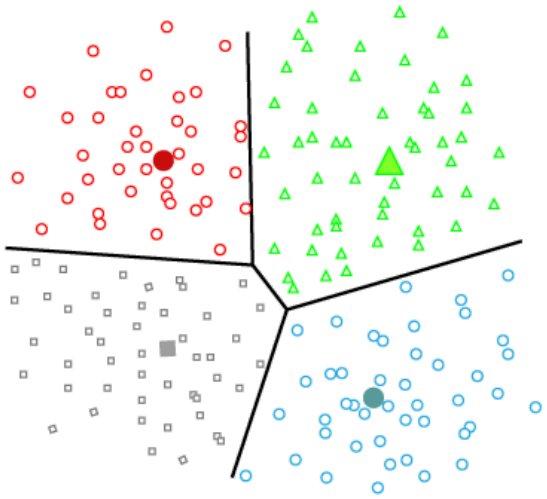
The clustering methods are broadly divided into **Hard clustering** (datapoint belongs to only one group) and **Soft Clustering** (data points can belong to another group also). But there are also other various approaches of Clustering exist. Below are the main clustering methods used in Machine learning:

1. **Partitioning Clustering**
2. **Density-Based Clustering**
3. **Distribution Model-Based Clustering**
4. **Hierarchical Clustering**
5. **Fuzzy Clustering**

Partitioning Clustering

It is a type of clustering that divides the data into non-hierarchical groups. It is also known as the centroid-based method**.** The most common example of partitioning clustering is the K-Means Clustering algorithm.

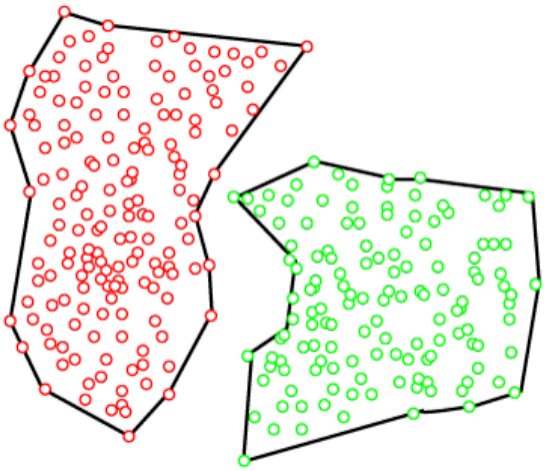
In this type, the dataset is divided into a set of k groups, where K is used to define the number of pre-defined groups. The cluster center is created in such a way that the distance between the data points of one cluster is minimum as compared to another cluster centroid.



Density-Based Clustering

The density-based clustering method connects the highly-dense areas into clusters, and the arbitrarily shaped distributions are formed as long as the dense region can be connected. This algorithm does it by identifying different clusters in the dataset and connects the areas of high densities into clusters. The dense areas in data space are divided from each other by sparser areas.

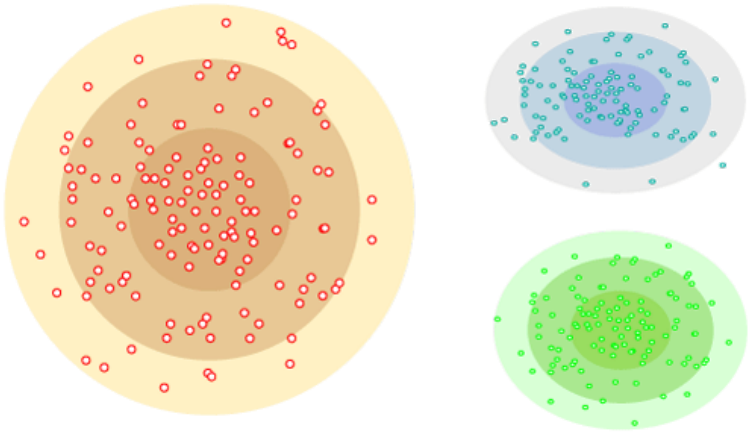
These algorithms can face difficulty in clustering the data points if the dataset has varying densities and high dimensions.



Distribution Model-Based Clustering

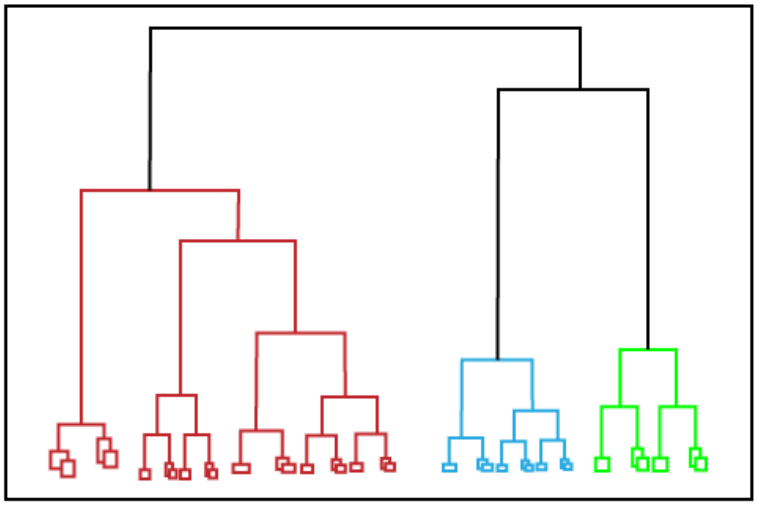
In the distribution model-based clustering method, the data is divided based on the probability of how a dataset belongs to a particular distribution. The grouping is done by assuming some distributions commonly **Gaussian Distribution**.

The example of this type is the **Expectation-Maximization Clustering algorithm** that uses Gaussian Mixture Models (GMM).



Hierarchical Clustering

Hierarchical clustering can be used as an alternative for the partitioned clustering as there is no requirement of pre-specifying the number of clusters to be created. In this technique, the dataset is divided into clusters to create a tree-like structure, which is also called a **dendrogram**. The observations or any number of clusters can be selected by cutting the tree at the correct level. The most common example of this method is the **Agglomerative Hierarchical algorithm**.



Fuzzy Clustering

Fuzzy clustering is a type of soft method in which a data object may belong to more than one group or cluster. Each dataset has a set of membership coefficients, which depend on the degree of membership to be in a cluster. **Fuzzy C-means algorithm** is the example of this type of clustering; it is sometimes also known as the Fuzzy k-means algorithm

Clustering Algorithms

The Clustering algorithms can be divided based on their models that are explained above. There are different types of clustering algorithms published, but only a few are commonly used. The clustering algorithm is based on the kind of data that we are using. Such as, some algorithms need to guess the number of clusters in the given dataset, whereas some are required to find the minimum distance between the observation of the dataset.

Here we are discussing mainly popular Clustering algorithms that are widely used in machine learning:

1. **K-Means algorithm:** The k-means algorithm is one of the most popular clustering algorithms. It classifies the dataset by dividing the samples into different clusters of equal variances. The number of clusters must be specified in this algorithm. It is fast with fewer computations required, with the linear complexity of **O(n).**
2. **DBSCAN Algorithm:** It stands **for Density-Based Spatial Clustering of Applications with Noise**. It is an example of a density-based model similar to the mean-shift, but with some remarkable advantages. In this algorithm, the areas of high density are separated by the areas of low density. Because of this, the clusters can be found in any arbitrary shape.

3. **Expectation-Maximization Clustering using GMM:** This algorithm can be used as an alternative for the k-means algorithm or for those cases where K-means can be failed. In GMM, it is assumed that the data points are Gaussian distributed.
4. **Agglomerative Hierarchical algorithm:** The Agglomerative hierarchical algorithm performs the bottom-up hierarchical clustering. In this, each data point is treated as a single cluster at the outset and then successively merged. The cluster hierarchy can be represented as a tree-structure.

Applications of Clustering

Below are some commonly known applications of clustering technique in Machine Learning:

- **In Identification of Cancer Cells:** The clustering algorithms are widely used for the identification of cancerous cells. It divides the cancerous and non-cancerous data sets into different groups.
- **In Search Engines:** Search engines also work on the clustering technique. The search result appears based on the closest object to the search query. It does it by grouping similar data objects in one group that is far from the other dissimilar objects. The accurate result of a query depends on the quality of the clustering algorithm used.
- **Customer Segmentation:** It is used in market research to segment the customers based on their choice and preferences.
- **In Biology:** It is used in the biology stream to classify different species of plants and animals using the image recognition technique.
- **In Land Use:** The clustering technique is used in identifying the area of similar lands use in the GIS database. This can be very useful to find that for what purpose the particular land should be used, that means for which purpose it is more suitable.

K-Means Clustering Algorithm

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering.

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

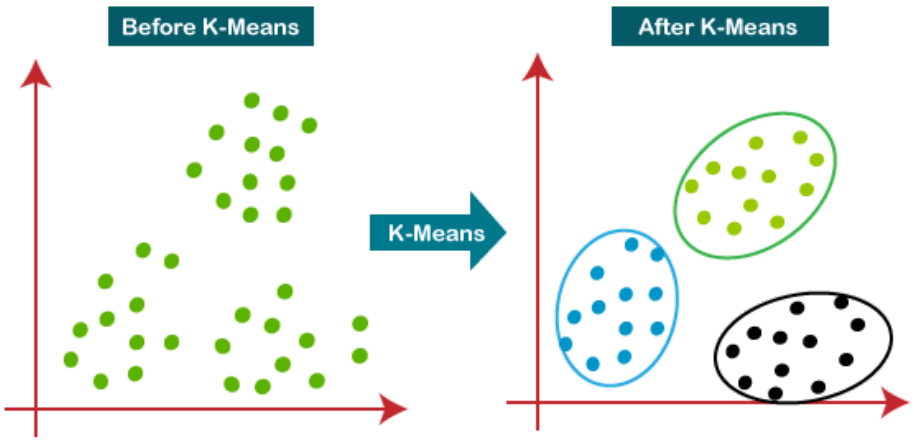
The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

Hence each cluster has datapoints with some commonalities, and it is away from other clusters.

The below diagram explains the working of the K-means Clustering Algorithm:



Working of K-Means Algorithm:

Step-1: Select the number K to decide the number of clusters.

Step-2: Select random K points or centroids. (It can be other from the input dataset).

Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

Step-4: Calculate the variance and place a new centroid of each cluster.

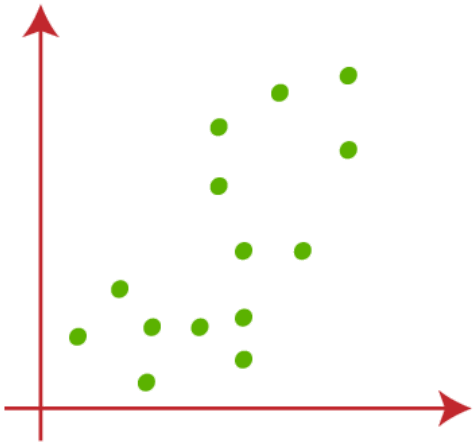
Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

Step-7: The model is ready.

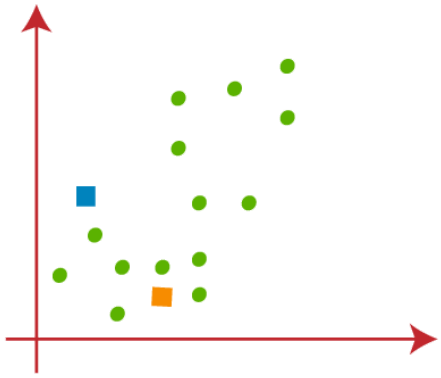
Let's understand the above steps by considering the visual plots:

Suppose we have two variables M1 and M2. The x-y axis scatter plot of these two variables is given below:

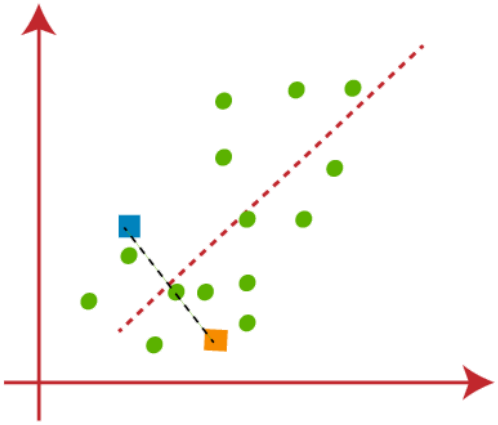


- Let's take number k of clusters, i.e., K=2, to identify the dataset and to put them into different clusters. It means here we will try to group these datasets into two different clusters.
- We need to choose some random k points or centroid to form the cluster. These points can be either the points from the dataset or any other point. So, here we are

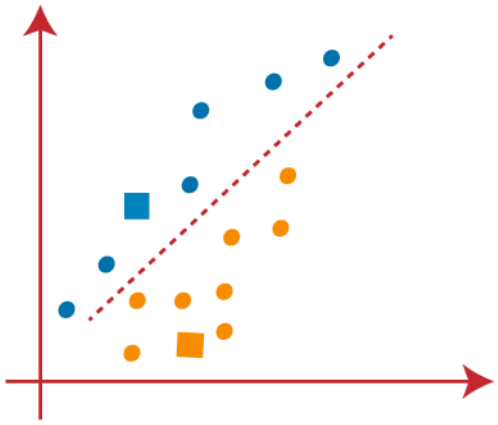
selecting the below two points as k points, which are not the part of our dataset. Consider the below image:



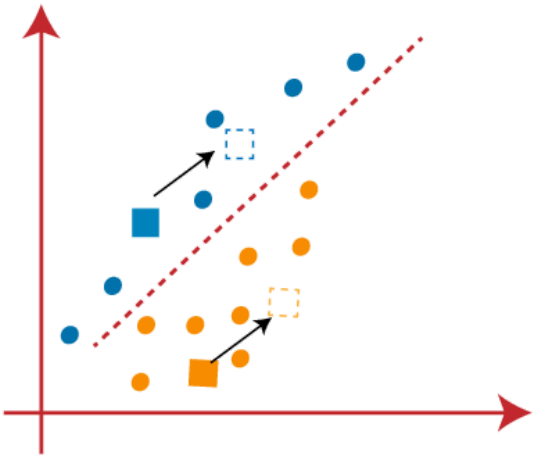
- Now we will assign each data point of the scatter plot to its closest K-point or centroid. We will compute it by applying some mathematics that we have studied to calculate the distance between two points. So, we will draw a median between both the centroids. Consider the below image:



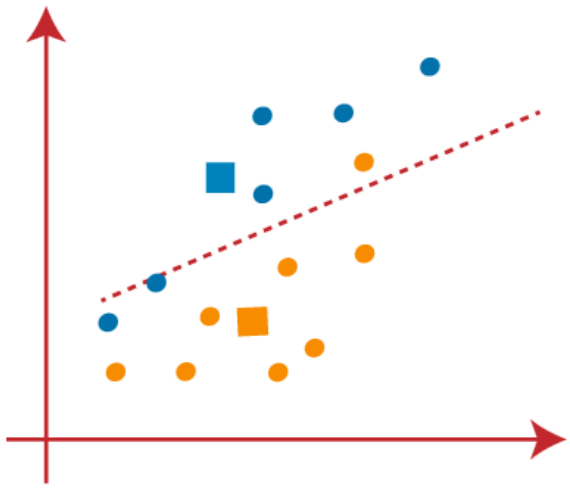
From the above image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.



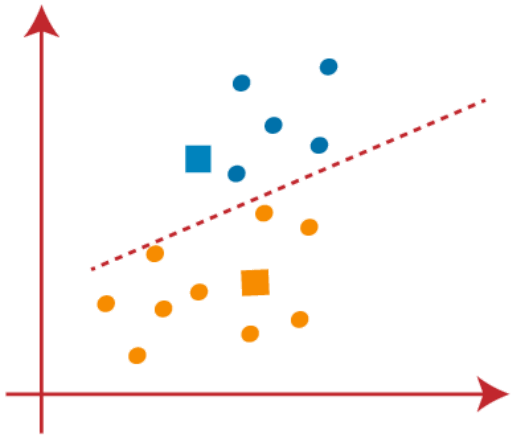
- As we need to find the closest cluster, so we will repeat the process by choosing a new centroid**.** To choose the new centroids, we will compute the center of gravity of these centroids, and will find new centroids as below:



- Next, we will reassign each datapoint to the new centroid. For this, we will repeat the same process of finding a median line. The median will be like below image:



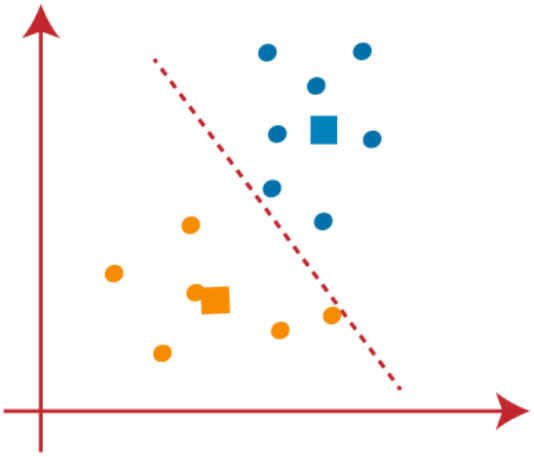
From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.



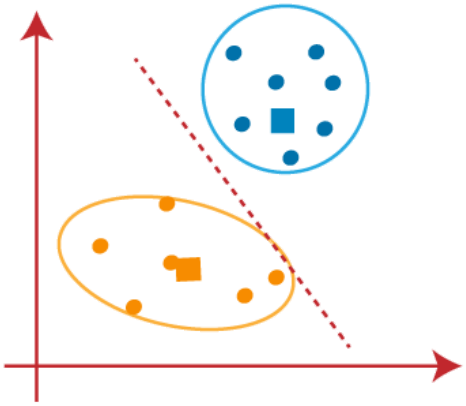
As reassignment has taken place, so we will again go to the step-4, which is finding new centroids or K-points.

- We will repeat the process by finding the center of gravity of centroids, so the new centroids will be as shown in the below image:

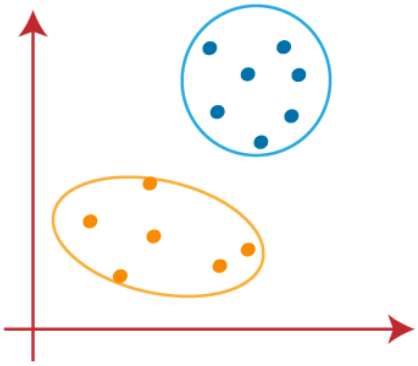
- As we got the new centroids so again will draw the median line and reassign the data points. So, the image will be:



- We can see in the above image; there are no dissimilar data points on either side of the line, which means our model is formed. Consider the below image:



As our model is ready, so we can now remove the assumed centroids, and the two final clusters will be as shown in the below image:



Choosing the value of "K number of clusters" in K-means Clustering

The performance of the K-means clustering algorithm depends upon highly efficient clusters that it forms. But choosing the optimal number of clusters is a big task. There are some different ways to find the optimal number of clusters, but here we are discussing the most appropriate method to find the number of clusters or value of K. The method is given below:

Elbow Method

The Elbow method is one of the most popular ways to find the optimal number of clusters. This method uses the concept of WCSS value. **WCSS** stands for **Within Cluster Sum of Squares**, which defines the total variations within a cluster. The formula to calculate the value of WCSS (for 3 clusters) is given below:

$$WCSS= \sum_{Pi \text{ in Cluster1}} distance(P_i\ C_1)^2 + \sum_{Pi \text{ in Cluster2}} distance(P_i\ C_2)^2 + \sum_{Pi \text{ in CLuster3}} distance(P_i\ C_3)^2$$

In the above formula of WCSS,

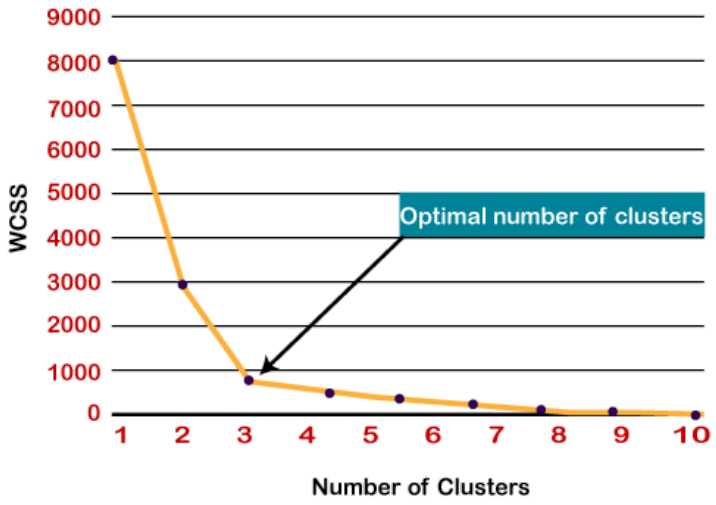
$\sum_{Pi \text{ in Cluster1}} distance(P_i\ C_1)^2$: It is the sum of the square of the distances between each data point and its centroid within a cluster1 and the same for the other two terms.

To measure the distance between data points and centroid, we can use any method such as Euclidean distance or Manhattan distance.

To find the optimal value of clusters, the elbow method follows the below steps:

- It executes the K-means clustering on a given dataset for different K values (ranges from 1-10).
- For each value of K, calculates the WCSS value.
- Plots a curve between calculated WCSS values and the number of clusters K.
- The sharp point of bend or a point of the plot looks like an arm, then that point is considered as the best value of K.

Since the graph shows the sharp bend, which looks like an elbow, hence it is known as the elbow method. The graph for the elbow method looks like the below image:



Note: We can choose the number of clusters equal to the given data points. If we choose the number of clusters equal to the data points, then the value of WCSS becomes zero, and that will be the endpoint of the plot.

Hierarchical Clustering

Hierarchical clustering is another unsupervised machine learning algorithm, which is used to group the unlabeled datasets into a cluster and also known as hierarchical cluster analysis or HCA.

In this algorithm, we develop the hierarchy of clusters in the form of a tree, and this tree-shaped structure is known as the dendrogram**.**

Sometimes the results of K-means clustering and hierarchical clustering may look similar, but they both differ depending on how they work. As there is no requirement to predetermine the number of clusters as we did in the K-Means algorithm.

The hierarchical clustering technique has two approaches:

1. **Agglomerative:** Agglomerative is a bottom-up approach, in which the algorithm starts with taking all data points as single clusters and merging them until one cluster is left.
2. **Divisive:** Divisive algorithm is the reverse of the agglomerative algorithm as it is a top-down approach**.**

Need for hierarchical clustering

As we already have other clustering algorithms such as **K-Means Clustering**, then why we need hierarchical clustering? So, as we have seen in the K-means clustering that there are some challenges with this algorithm, which are a predetermined number of clusters, and it always tries to create the clusters of the same size. To solve these two challenges, we can opt for the hierarchical clustering algorithm because, in this algorithm, we don't need to have knowledge about the predefined number of clusters.

Agglomerative Hierarchical clustering

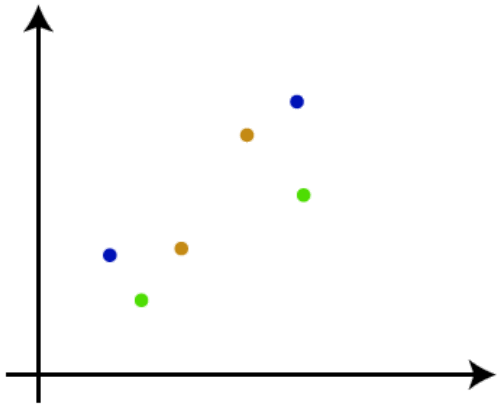
The agglomerative hierarchical clustering algorithm is a popular example of HCA. To group the datasets into clusters, it follows the bottom-up approach. It means, this algorithm considers each dataset as a single cluster at the beginning, and then start combining the closest pair of clusters together. It does this until all the clusters are merged into a single cluster that contains all the datasets.

This hierarchy of clusters is represented in the form of the dendrogram.

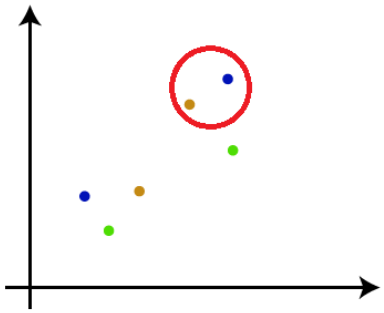
Working of Agglomerative Hierarchical clustering

The working of the AHC algorithm can be explained using the below steps:

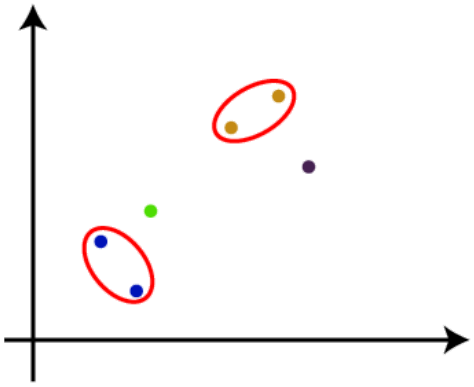
- **Step-1:** Create each data point as a single cluster. Let's say there are N data points, so the number of clusters will also be N.



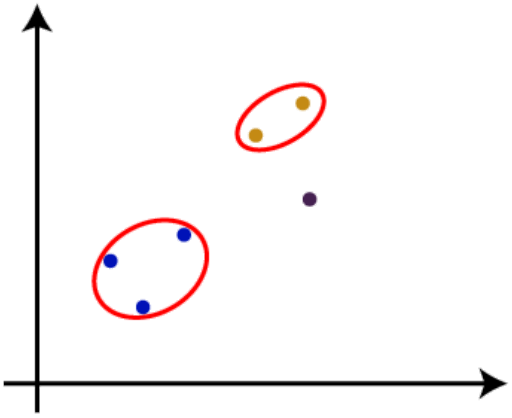
- **Step-2:** Take two closest data points or clusters and merge them to form one cluster. So, there will now be N-1 clusters.

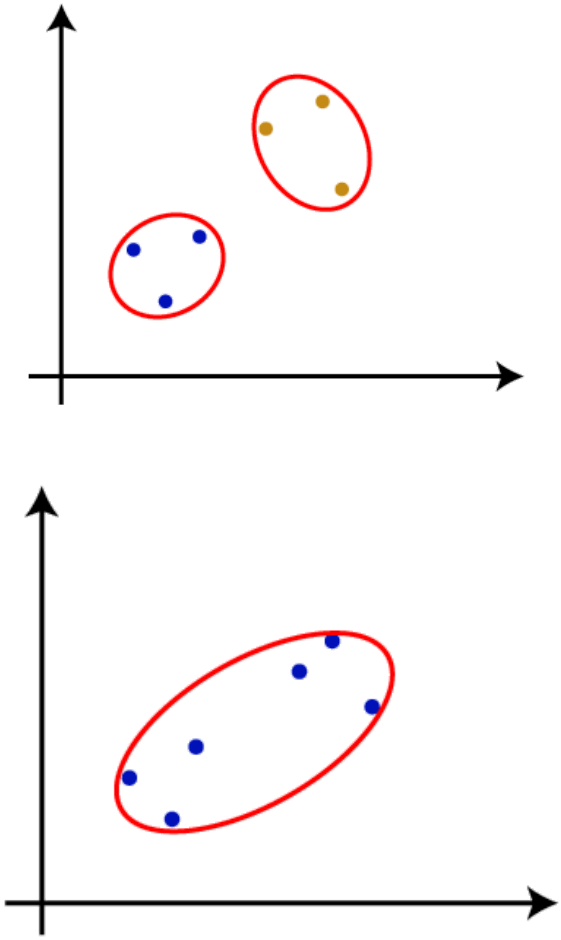


- **Step-3**: Again, take the two closest clusters and merge them together to form one cluster. There will be N-2 clusters.



- **Step-4:** Repeat Step 3 until only one cluster left. So, we will get the following clusters. Consider the below images:





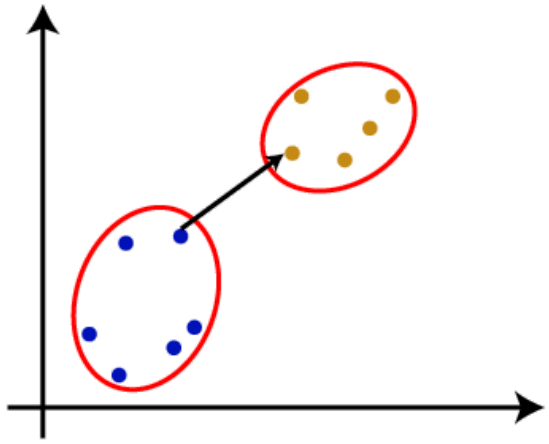
- **Step-5:** Once all the clusters are combined into one big cluster, develop the dendrogram to divide the clusters as per the problem.

Note: To better understand hierarchical clustering, it is advised to have a look on k-means clustering

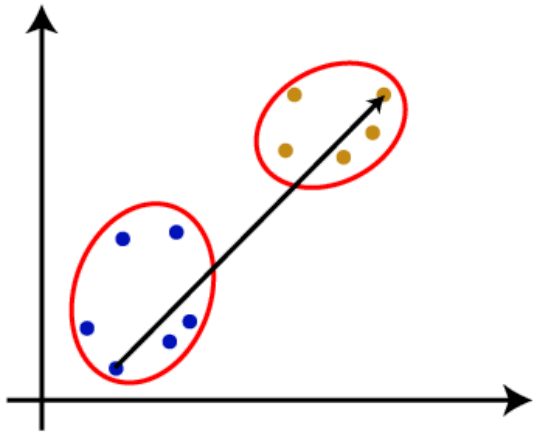
Measure for the distance between two clusters

As we have seen, the **closest distance** between the two clusters is crucial for the hierarchical clustering. There are various ways to calculate the distance between two clusters, and these ways decide the rule for clustering. These measures are called **Linkage methods**. Some of the popular linkage methods are given below:

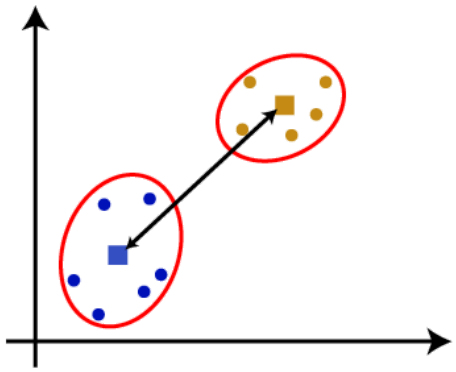
1. **Single Linkage:** It is the Shortest Distance between the closest points of the clusters. Consider the below image:



2. **Complete Linkage:** It is the farthest distance between the two points of two different clusters. It is one of the popular linkage methods as it forms tighter clusters than single-linkage.



3. **Average Linkage:** It is the linkage method in which the distance between each pair of datasets is added up and then divided by the total number of datasets to calculate the average distance between two clusters. It is also one of the most popular linkage methods.
4. **Centroid Linkage:** It is the linkage method in which the distance between the centroid of the clusters is calculated. Consider the below image:

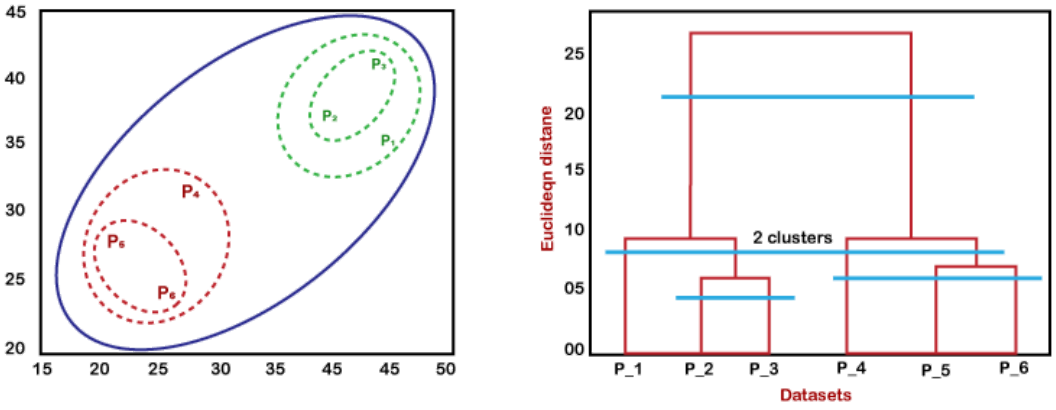


From the above-given approaches, we can apply any of them according to the type of problem or business requirement.

Woking of Dendrogram in Hierarchical clustering

The dendrogram is a tree-like structure that is mainly used to store each step as a memory that the HC algorithm performs. In the dendrogram plot, the Y-axis shows the Euclidean distances between the data points, and the x-axis shows all the data points of the given dataset.

The working of the dendrogram can be explained using the below diagram:



In the above diagram, the left part is showing how clusters are created in agglomerative clustering, and the right part is showing the corresponding dendrogram.

- As we have discussed above, firstly, the datapoints P2 and P3 combine together and form a cluster, correspondingly a dendrogram is created, which connects P2 and P3

with a rectangular shape. The hight is decided according to the Euclidean distance between the data points.

- In the next step, P5 and P6 form a cluster, and the corresponding dendrogram is created. It is higher than of previous, as the Euclidean distance between P5 and P6 is a little bit greater than the P2 and P3.
- Again, two new dendrograms are created that combine P1, P2, and P3 in one dendrogram, and P4, P5, and P6, in another dendrogram.
- At last, the final dendrogram is created that combines all the data points together.

We can cut the dendrogram tree structure at any level as per our requirement.

EM Algorithm in Machine Learning

The EM algorithm is considered a latent variable model to find the local maximum likelihood parameters of a statistical model, proposed by Arthur Dempster, Nan Laird, and Donald Rubin in 1977.

The EM (Expectation-Maximization) algorithm is one of the most commonly used terms in machine learning to obtain maximum likelihood estimates of variables that are sometimes observable and sometimes not. However, it is also applicable to unobserved data or sometimes called latent. It has various real-world applications in statistics, including obtaining the mode of the posterior marginal distribution of parameters in machine learning and data mining applications**.**

In most real-life applications of machine learning, it is found that several relevant learning features are available, but very few of them are observable, and the rest are unobservable. If the variables are observable, then it can predict the value using instances. On the other hand, the variables which are latent or directly not observable, for such variables Expectation-Maximization (EM) algorithm plays a vital role to predict the value with the condition that the general form of probability distribution governing those latent variables is known to us. In this topic, we will discuss a basic introduction to the EM algorithm, a flow chart of the EM algorithm, its applications, advantages, and disadvantages of EM algorithm, etc.

EM algorithm

The Expectation-Maximization (EM) algorithm is defined as the combination of various unsupervised machine learning algorithms, which is used to determine the **local maximum likelihood estimates (MLE)** or **maximum a posteriori estimates (MAP)** for unobservable variables in statistical models. Further, it is a technique to find maximum likelihood

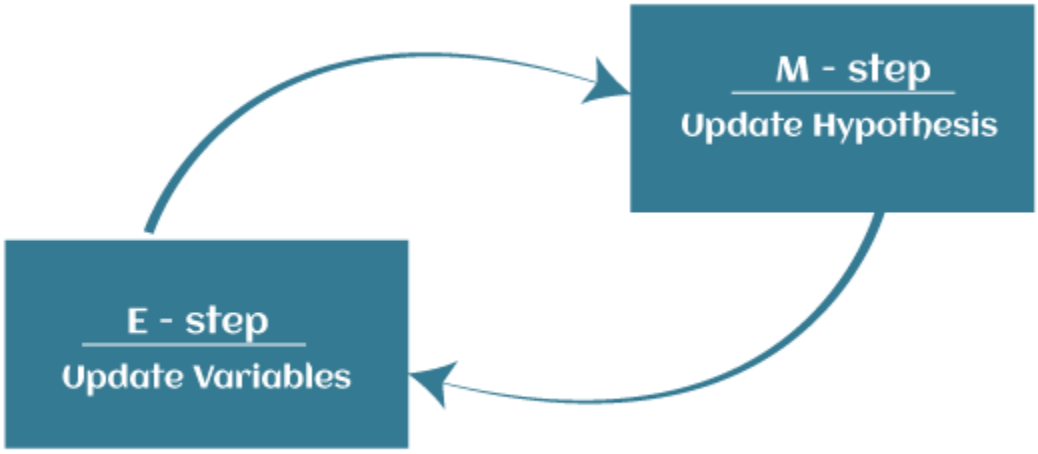
estimation when the latent variables are present. It is also referred to as the **latent variable model.**

A latent variable model consists of both observable and unobservable variables where observable can be predicted while unobserved are inferred from the observed variable. These unobservable variables are known as latent variables.

Key Points:

- It is known as the latent variable model to determine MLE and MAP parameters for latent variables.
- It is used to predict values of parameters in instances where data is missing or unobservable for learning, and this is done until convergence of the values occurs.

The EM algorithm is the combination of various unsupervised ML algorithms, such as the **k-means clustering algorithm**. Being an iterative approach, it consists of two modes. In the first mode, we estimate the missing or latent variables. Hence it is referred to as the **Expectation/estimation step (E-step**). Further, the other mode is used to optimize the parameters of the models so that it can explain the data more clearly. The second mode is known as the **maximization-step or M-step.**



- **Expectation step (E - step):** It involves the estimation (guess) of all missing values in the dataset so that after completing this step, there should not be any missing value.
- **Maximization step (M - step):** This step involves the use of estimated data in the E-step and updating the parameters.

- **Repeat E**-step and M-step until the convergence of the values occurs.

The primary goal of the EM algorithm is to use the available observed data of the dataset to estimate the missing data of the latent variables and then use that data to update the values of the parameters in the M-step.

Convergence in the EM algorithm

Convergence is defined as the specific situation in probability based on intuition, e.g., if there are two random variables that have very less difference in their probability, then they are known as converged. In other words, whenever the values of given variables are matched with each other, it is called convergence.

Steps in EM Algorithm

The EM algorithm is completed mainly in 4 steps, which include I**nitialization Step, Expectation Step, Maximization Step, and convergence Step**. These steps are explained as follows:

- **1st Step:** The very first step is to initialize the parameter values. Further, the system is provided with incomplete observed data with the assumption that data is obtained from a specific model.

- **2nd Step:** This step is known as Expectation or E-Step, which is used to estimate or guess the values of the missing or incomplete data using the observed data. Further, E-step primarily updates the variables.
- **3rd Step:** This step is known as Maximization or M-step, where we use complete data obtained from the 2nd step to update the parameter values. Further, M-step primarily updates the hypothesis.
- **4th step:** The last step is to check if the values of latent variables are converging or not. If it gets "yes", then stop the process; else, repeat the process from step 2 until the convergence occurs.

Applications of EM algorithm

The primary aim of the EM algorithm is to estimate the missing data in the latent variables through observed data in datasets. The EM algorithm or latent variable model has a broad range of real-life applications in machine learning. These are as follows:

- The EM algorithm is applicable in data clustering in machine learning.
- It is often used in computer vision and NLP (Natural language processing).
- It is used to estimate the value of the parameter in mixed models such as the **Gaussian Mixture Model**and quantitative genetics.
- It is also used in psychometrics for estimating item parameters and latent abilities of item response theory models.
- It is also applicable in the medical and healthcare industry, such as in image reconstruction and structural engineering.
- It is used to determine the Gaussian density of a function.

Advantages of EM algorithm

- It is very easy to implement the first two basic steps of the EM algorithm in various machine learning problems, which are E-step and M- step.
- It is mostly guaranteed that likelihood will enhance after each iteration.
- It often generates a solution for the M-step in the closed form.

Disadvantages of EM algorithm

- The convergence of the EM algorithm is very slow.
- It can make convergence for the local optima only.

- It takes both forward and backward probability into consideration. It is opposite to that of numerical optimization, which takes only forward probabilities.

In real-world applications of machine learning, the expectation-maximization (EM) algorithm plays a significant role in determining the local maximum likelihood estimates (MLE) or maximum a posteriori estimates (MAP) for unobservable variables in statistical models. It is often used for the latent variables, i.e., to estimate the latent variables through observed data in datasets. It is generally completed in two important steps, i.e., the expectation step (E-step) and the Maximization step (M-Step), where E-step is used to estimate the missing data in datasets, and M-step is used to update the parameters after the complete data is generated in E-step. Further, the importance of the EM algorithm can be seen in various applications such as data clustering, natural language processing (NLP), computer vision, image reconstruction, structural engineering, etc.

Fuzzy C-means clustering algorithm

C-means clustering, or fuzzy c-means clustering, is a soft clustering technique in machine learning in which each data point is separated into different clusters and then assigned a probability score for being in that cluster

Clustering is an unsupervised machine learning technique that divides the given data into different clusters based on their distances (similarity) from each other.

The unsupervised k-means clustering algorithm gives the values of any point lying in some particular cluster to be either as 0 or 1 i.e., either true or false. But the fuzzy logic gives the fuzzy values of any particular data point to be lying in either of the clusters. Here, in fuzzy c-means clustering, we find out the centroid of the data points and then calculate the distance of each data point from the given centroids until the clusters formed becomeconstant.

Suppose the given data points are **{(1, 3), (2, 5), (6, 8), (7, 9)}**

Fuzzy Clustering is a type of clustering algorithm in machine learning that allows a data point to belong to more than one cluster with different degrees of membership. Unlike traditional clustering algorithms, such as k-means or hierarchical clustering, which assign each data point to a single cluster, fuzzy clustering assigns a membership degree between 0 and 1 for each data point for each cluster.

Applications in several fields of Fuzzy clustering :

1. **Image segmentation**: Fuzzy clustering can be used to segment images by grouping pixels with similar properties together, such as color or texture.

2. **Pattern recognition**: Fuzzy clustering can be used to identify patterns in large datasets by grouping similar data points together.
3. **Marketing:** Fuzzy clustering can be used to segment customers based on their preferences and purchasing behavior, allowing for more targeted marketing campaigns.
4. **Medical diagnosis**: Fuzzy clustering can be used to diagnose diseases by grouping patients with similar symptoms together.
5. **Environmental monitoring:** Fuzzy clustering can be used to identify areas of environmental concern by grouping together areas with similar pollution levels or other environmental indicators.
6. **Traffic flow analysis:** Fuzzy clustering can be used to analyze traffic flow patterns by grouping similar traffic patterns together, allowing for better traffic management and planning.
7. **Risk assessment:** Fuzzy clustering can be used to identify and quantify risks in various fields, such as finance, insurance, and engineering.

Advantages of Fuzzy Clustering:

1. Flexibility: Fuzzy clustering allows for overlapping clusters, which can be useful when the data has a complex structure or when there are ambiguous or overlapping class boundaries.
2. Robustness: Fuzzy clustering can be more robust to outliers and noise in the data, as it allows for a more gradual transition from one cluster to another.
3. Interpretability: Fuzzy clustering provides a more nuanced understanding of the structure of the data, as it allows for a more detailed representation of the relationships between data points and clusters.

Disadvantages of Fuzzy Clustering:

1. Complexity: Fuzzy clustering algorithms can be computationally more expensive than traditional clustering algorithms, as they require optimization over multiple membership degrees.
2. Model selection: Choosing the right number of clusters and membership functions can be challenging, and may require expert knowledge or trial and error.
3. If you're interested in learning more about fuzzy clustering, you might consider reading "Fuzzy Clustering and Its Applications" by James C. Bezdek or "An Introduction to Fuzzy Clustering" by Witold Pedrycz and Fernando Gomide.

The steps to perform the algorithm are:

Step 1: Initialize the data points into the desired number of clusters randomly.

Let us assume there are 2 clusters in which the data is to be divided, initializing the data point randomly. Each data point lies in both clusters with some membership value which can be assumed anything in the initial state.

The table below represents the values of the data points along with their membership (gamma) in each cluster.

Cluster	(1, 3)	(2, 5)	(4, 8)	(7, 9)
1)	0.8	0.7	0.2	0.1
2)	0.2	0.3	0.8	0.9

Step 2: Find out the centroid.

The formula for finding out the centroid (V) is:

Where, μ is fuzzy membership value of the data point, m is the fuzziness parameter (generally taken as 2), and xk is the data point.

Here,

V11 = (0.8^2 *1 + 0.7^2 * 2 + 0.2^2 * 4 + 0.1^2 * 7) / ((0.8^2 + 0.7^2 + 0.2^2 + 0.1^2) = 1.568

V12 = (0.8^2 *3 + 0.7^2 * 5 + 0.2^2 * 8 + 0.1^2 * 9) / ((0.8^2 + 0.7^2 + 0.2^2 + 0.1^2) = 4.051

V21 = (0.2^2 *1 + 0.3^2 * 2 + 0.8^2 * 4 + 0.9^2 * 7) / ((0.2^2 + 0.3^2 + 0.8^2 + 0.9^2) = 5.35

V22 = (0.2^2 *3 + 0.3^2 * 5 + 0.8^2 * 8 + 0.9^2 * 9) / ((0.2^2 + 0.3^2 + 0.8^2 + 0.9^2) = 8.215

Centroids are: (1.568, 4.051) and (5.35, 8.215)

Step 3: Find out the distance of each point from the centroid.

$D11 = ((1 - 1.568)^2 + (3 - 4.051)^2)^{0.5} = 1.2$

$D12 = ((1 - 5.35)^2 + (3 - 8.215)^2)^{0.5} = 6.79$

Similarly, the distance of all other points is computed from both the centroids.

Step 4: Updating membership values.

For point 1 new membership values are:

$= [\{ [(1.2)^2 / (1.2)^2] + [(1.2)^2 / (6.79)^2]\} \wedge \{(1 / (2 - 1))\}]^{-1} = 0.96$

$= [\{ [(6.79)^2 / (6.79)^2] + [(6.79)^2 / (1.2)^2]\} \wedge \{(1 / (2 - 1))\}]^{-1} = 0.04$

Alternatively,

Similarly, compute all other membership values, and update the matrix.

Step 5: Repeat the steps(2-4) until the constant values are obtained for the membership values or the difference is less than the tolerance value (a small value up to which the difference in values of two consequent updations is accepted).
Step 6: Defuzzify the obtained membership values.