**Module 3: Unsupervised learning**

Unsupervised learning is the training of a machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of the machine is to group unsorted information according to similarities, patterns, and differences without any prior training of data.

Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore, the machine is restricted to find the hidden structure in unlabeled data by itself.

**For instance**, suppose it is given an image having both dogs and cats which it has never seen. Thus, the machine has no idea about the features of dogs and cats so we can’t categorize it as ‘dogs and cats ‘. But it can categorize them according to their similarities, patterns, and differences, i.e., we can easily categorize the above picture into two parts. The first may contain all pics having **dogs** in them and the second part may contain all pics having **cats** in them. Here you didn’t learn anything before, which means no training data or examples.

It allows the model to work on its own to discover patterns and information that was previously undetected. It mainly deals with unlabelled data.

Unsupervised learning is classified into two categories of algorithms: 

* **Clustering**: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
* **Association Rule Learning**: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y. An example of this is Market Basket Analysis.

**Clustering**: Clustering is the grouping of objects or data points that are similar to each other and dissimilar to objects in other clusters.

Machine learning engineers and data scientists can use different algorithms for clustering, with the algorithms themselves falling into different categories based on how they work. The categories include the following:

* Exclusive clustering (Partitioning Method/ Hard Clustering)
* Overlapping clustering (Soft Clustering)
* hierarchical clustering
* Density-based clustering

**K-means Clustering:**

K-Means Clustering is an [Unsupervised Learning algorithm](https://www.javatpoint.com/unsupervised-machine-learning), 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 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](https://www.javatpoint.com/clustering-in-machine-learning) 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.

## How does the K-Means Algorithm Work?

The working of the K-Means algorithm is explained in the below steps:

**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.

## How to choose the value of "K number of clusters" in K-means Clustering?

### **Elbow Method:**

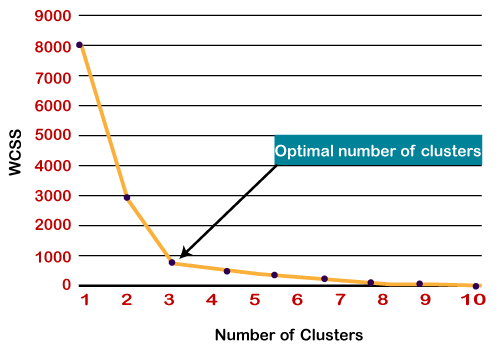
The Elbow method is one of the most popular ways to find the optimal number of clusters. This method uses the concept of WCSS value. **WCSS** stands for **Within Cluster Sum of Squares**, which defines the total variations within a cluster. The formula to calculate the value of WCSS (for 3 clusters)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

WCSS= ∑Pi in Cluster1 distance(Pi C1)2  + ∑Pi in Cluster2distance(Pi C2)2 + ∑Pi in CLuster3 distance(Pi C3)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.

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:



**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**.

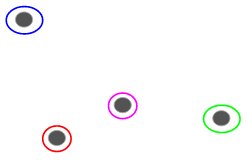
## Types of Hierarchical Clustering

There are mainly two types of hierarchical clustering:

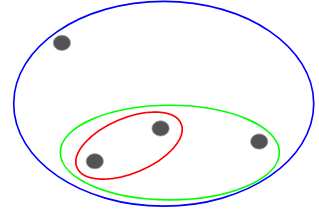
1. Agglomerative hierarchical clustering
2. Divisive Hierarchical clustering

### **Agglomerative Hierarchical Clustering**

We assign each point to an individual cluster in this technique. Suppose there are 4 data points. We will assign each of these points to a cluster and hence will have 4 clusters in the beginning:

[](https://cdn.analyticsvidhya.com/wp-content/uploads/2019/05/Screenshot-from-2019-05-15-13-11-28.png)

Then, at each iteration, we merge the closest pair of clusters and repeat this step until only a single cluster is left:

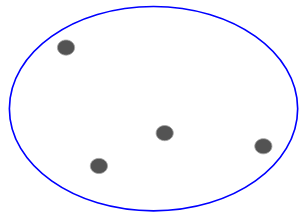
[](https://cdn.analyticsvidhya.com/wp-content/uploads/2019/05/Screenshot-from-2019-05-15-13-31-06.png)

We are merging (or adding) the clusters at each step, right? Hence, this type of clustering is also known as **additive hierarchical clustering.**

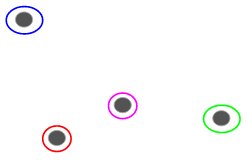
### **Divisive Hierarchical Clustering**

Divisive hierarchical clustering works in the opposite way. Instead of starting with n clusters (in case of n observations), we start with a single cluster and assign all the points to that cluster.

So, it doesn’t matter if we have 10 or 1000 data points. All these points will belong to the same cluster at the beginning:

[](https://cdn.analyticsvidhya.com/wp-content/uploads/2019/05/Screenshot-from-2019-05-15-13-12-35.png)

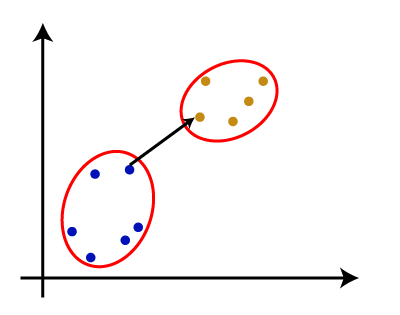
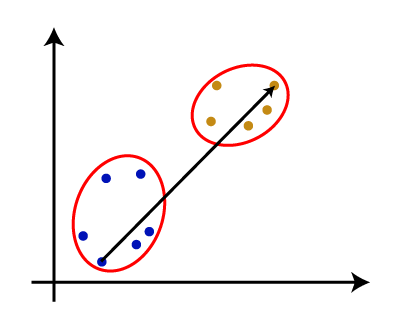
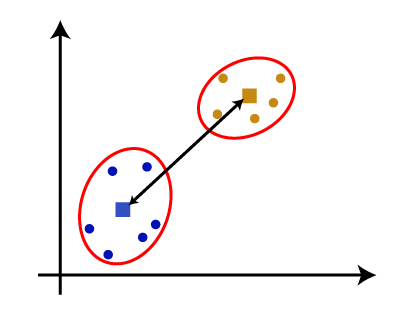
Now, at each iteration, we split the farthest point in the cluster and repeat this process until each cluster only contains a single point:



We are splitting (or dividing) the clusters at each step, hence the name divisive hierarchical 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:  
   

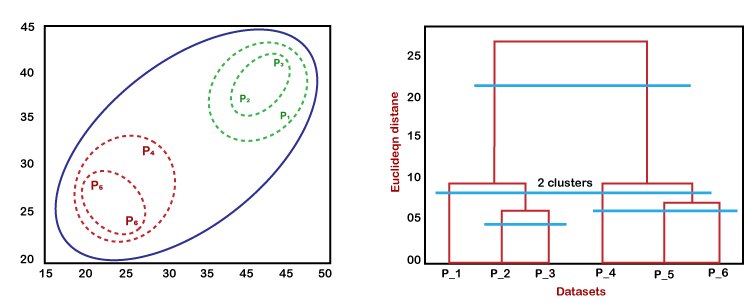
From the above-given approaches, we can apply any of them according to the type of problem or business requirement.

5. **Ward´s Linkage:** Ward´s linkage is a method for hierarchical cluster analysis . The linkage function specifying the distance between two clusters is computed as the increase in the "sum of squares" after fusing two clusters into a single cluster. Ward´s Method seeks to choose the successive clustering steps so as to minimize the increase in sum of squares at each step.

### **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.

**Density-based spatial clustering of applications with noise** **(DBSCAN):**

Clusters are dense regions in the data space, separated by regions of the lower density of points.

The key idea is that for each point of a cluster, the neighborhood of a given radius must contain at least a minimum number of points.

**Why Density -Based clustering?**

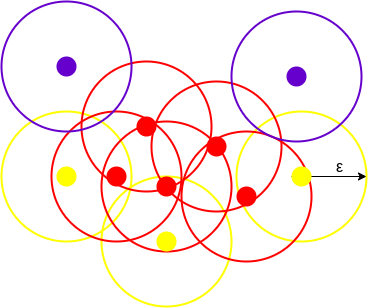
* Partitioning methods (K-means, PAM clustering) and hierarchical clustering work for finding spherical-shaped clusters or convex clusters.
* In other words, they are suitable only for compact and well-separated clusters.
* Moreover, they are also severely affected by the presence of noise and outliers in the data.

**DBSCAN algorithm requires two parameters:**

1. **eps** : It defines the neighborhood around a data point i.e. if the distance between two points is lower or equal to ‘eps’ then they are considered as neighbors. If the eps value is chosen too small then large part of the data will be considered as outliers. If it is chosen very large then the clusters will merge and majority of the data points will be in the same clusters. One way to find the eps value is based on the ***k-distance graph***.
2. **MinPts**: Minimum number of neighbors (data points) within eps radius. Larger the dataset, the larger value of MinPts must be chosen. As a general rule, the minimum MinPts can be derived from the number of dimensions D in the dataset as, MinPts >= D+1. The minimum value of MinPts must be chosen at least 3.

**In this algorithm, we have 3 types of data points:**

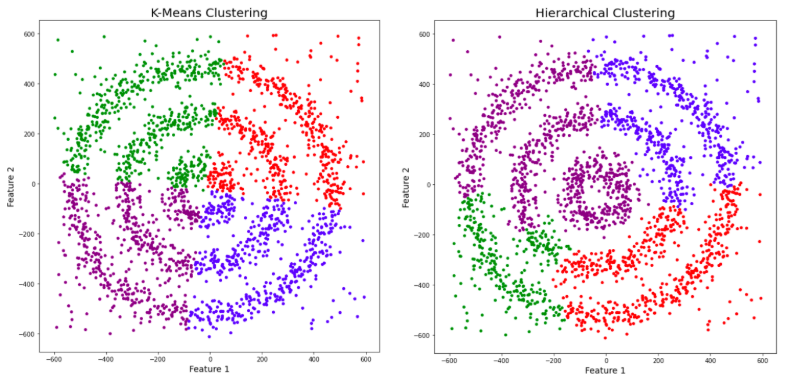
* ***Core Point****:* A point is a core point if it has more than MinPts points within eps.
* ***Border Point****:* A point which has fewer than MinPts within eps but it is in the neighborhood of a core point.
* ***Noise or outlier****:* A point which is not a core point or border point.

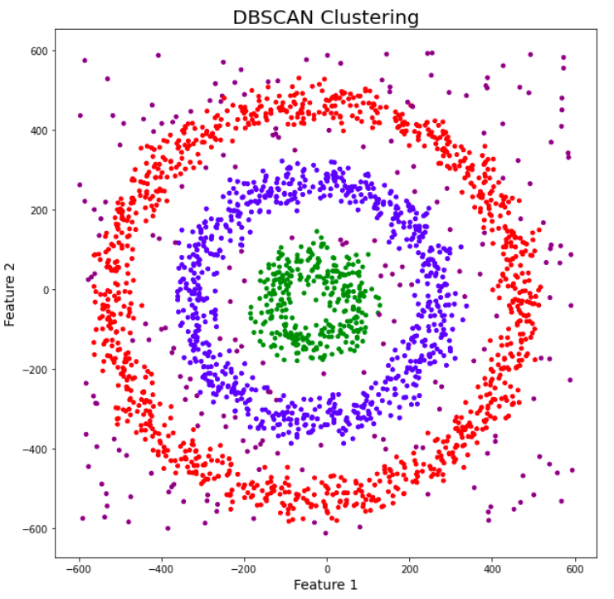


This figure shows us a cluster created by DBCAN with *minPoints = 3*. Here, we draw a circle of equal radius *epsilon* around every data point.

* All the data points with at least 3 points in the circle including itself are considered as **Core** points represented by red color.
* All the data points with less than 3 but greater than 1 point in the circle including itself are considered as **Border** points. They are represented by yellow color.
* Finally, data points with no point other than itself present inside the circle are considered as **Noise** represented by the purple color.

**Example to compare cluster formation by different clustering algorithms:**





**What Is Good Clustering?**

A good clustering method will produce high quality clusters with

* high intra-class similarity (i.e high within cluster similarity)
* low inter-class similarity (i.e low between clusters similarity)

**Measure the Quality of Clustering:**

* The quality of a clustering result depends on both the similarity measure used by the method and its implementation.
* The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns.
* Dissimilarity/Similarity metric: Similarity is expressed in terms of a distance function, which is typically metric: d(i, j)

There are two types of cluster quality measures:

1. Supervised cluster analysis techniques

2. Unsupervised cluster analysis techniques

**1. Supervised cluster analysis techniques:**

* requires knowledge of the ground truth classes.
* or requires manual assignment by human annotators.

**Example**: Rand Index, Fowlkes-Mallows scores, etc.

**Rand Index**: It computes a similarity measure between two clusters by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clusterings.

The formula of the Rand Index is:

Text, schematic

Description automatically generated

The RI can range from zero to 1, a perfect match will have RI score as 1. The only drawback of Rand Index is that it assumes that we can find the ground-truth clusters labels and use them to compare the performance of our model, so it is much less useful than the Silhouette Score for pure Unsupervised Learning tasks.

To calculate the Rand Index: (sklearn function)

***sklearn.metrics.rand\_score(labels\_true, labels\_pred)***

## Adjusted Rand Index: Rand index adjusted for chance.

The Rand Index computes a similarity measure between two clusterings by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clusterings.

The raw RI score is then “adjusted for chance” into the ARI score using the following scheme:

A picture containing text, watch

Description automatically generated

The Adjusted Rand Index, similarly to RI, ranges from zero to one, with zero equating to random labelling and one when the clusters are identical.

Similarly to RI, to calculate the ARI:

sklearn.metrics.**adjusted\_mutual\_info\_score**(labels\_true, labels\_pred, \*, average\_method='arithmetic')

**2. Unsupervised Clusters Analysis Techniques :**

* If the ground truth labels are not known, evaluation must be performed using the model itself.
* Example: Silhouette Coefficient, Davies-Bouldin Index, etc.

**Silhouette Score:**

* The Silhouette Score and Silhouette Plot are used to measure the separation distance between clusters. It displays a measure of how close each point in a cluster is to points in the neighbouring clusters. This measure has a range of [-1, 1] and is a great tool to visually inspect the similarities within clusters and differences across clusters.
* The Silhouette Score is calculated using the mean intra-cluster distance (i) and the mean nearest-cluster distance (n) for each sample. The Silhouette Coefficient for a sample is (n - i) / max(i, n).
* n is the distance between each sample and the nearest cluster that the sample is not a part of while i is the mean distance within each cluster.

The higher the Silhouette Coefficients (the closer to +1), the further away the cluster’s samples are from the neighbouring clusters samples. A value of 0 indicates that the sample is on or very close to the decision boundary between two neighbouring clusters. Negative values, instead, indicate that those samples might have been assigned to the wrong cluster.