Module Summary

Unsupervised Learning

In the previous modules, you saw various supervised machine learning algorithms. Supervised learning is a type of machine learning algorithm that uses a known dataset to preform predictions. This dataset (referred to as the training dataset) includes both response values and input data. From this, the supervised learning algorithm seeks to build a model that can predict the response values for a new dataset.

If you are training your machine learning task only with a set of inputs, it is called unsupervised learning, which will be able to find the structure or relationships between different inputs. The most important unsupervised learning technique is clustering, which creates different groups or clusters of the given set of inputs and is also able to put any new input in the appropriate cluster. While carrying out clustering, the basic objective is to group the input points in such a way as to **maximise the inter-cluster variance and minimise the intra-cluster variance**.

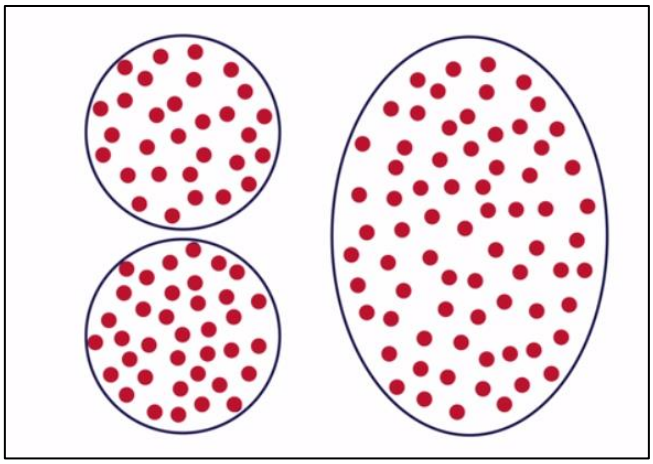


Fig 1: Objective of clustering is to **maximise the inter-cluster distance and minimise the intra-cluster variance**

The two most important methods of clustering are the K-Means algorithm & the Hierarchical clustering algorithm.

K-Means Algorithm

K-Means algorithm is the process of dividing the N data points into K groups or clusters. Here the steps of the algorithm are:

1. Start by choosing K random points the initial cluster centres.
2. Assign each data point to their nearest cluster centre. The most common way of measuring the distance between the points is the Euclidean distance.
3. For each cluster, compute the new cluster centre which will be the mean of all cluster members.
4. Now re-assign all the data points to the diffrent clusters by taking into account the new cluster centres.
5. Keep iterating through the step 3 & 4 until there are no further changes possible.

At this point, you arrive at the optimal clusters.

Let's apply the K-Means algorithm on a set of 10 points, which we want to divide into 2 clusters. Thus the value of K here is 2.

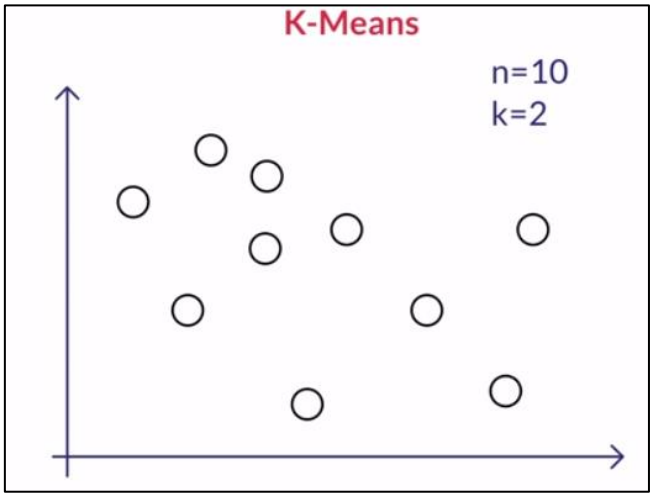


Fig 2: A set of 10 points to be divided into 2 clusters

We begin with choosing 2 random points as the 2 cluster centres.

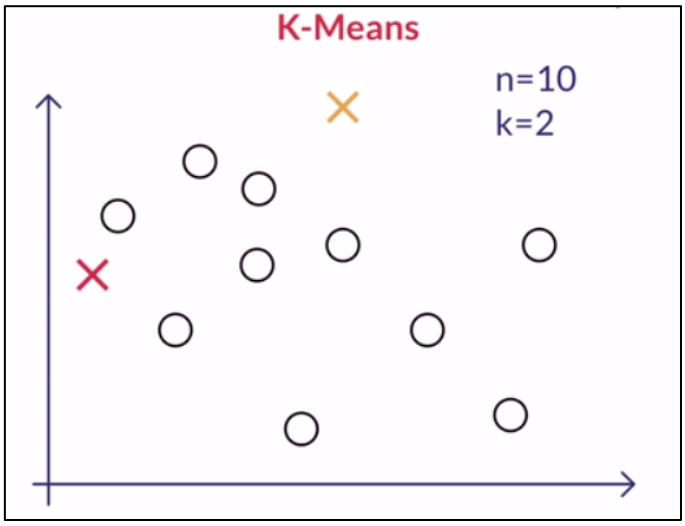


Fig 3: Choosing K random initial cluster centres

We then assign each of the data points to their nearest cluster centres based on the Euclidean distance. This way all the points are divided among the K clusters.

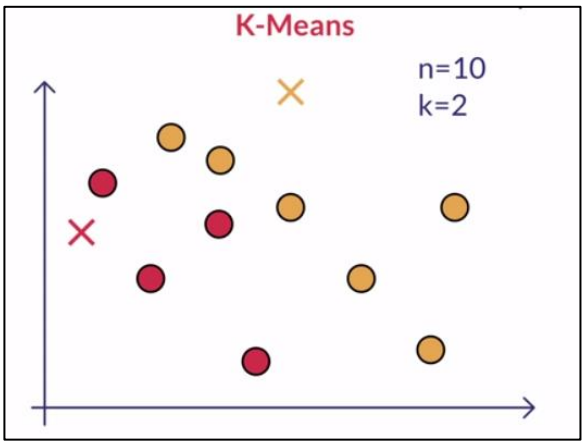


Fig 4: Assigning each data point to their nearest cluster centre

Now we update the position of each of the cluster centres to reflect the mean of each cluster.

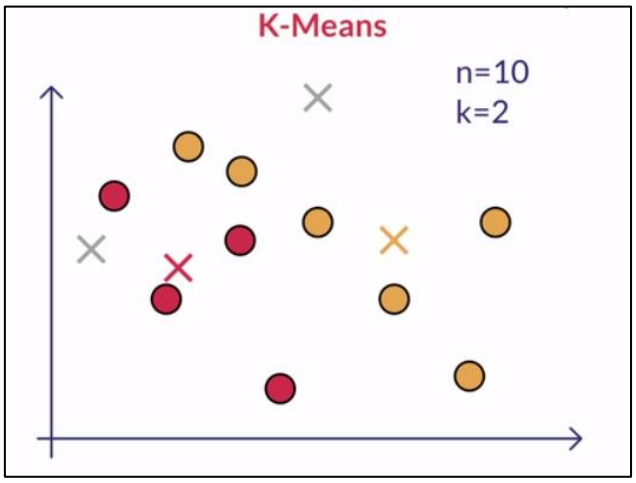


Fig 5: Updating the cluster centres

This process continues iteratively till the clusters converge; that is, there are no more changes possible in the position of the cluster centres. At this point, we achieve the two optimal clusters.

Practical considerations in K-Means algorithm

Some of the points to be considered while implementing K-Means algorithm are:

1.) **The choice of initial cluster centre has an impact on the final cluster composition.**

You saw the impact of the initial cluster centres through the visualisation tool. In the 3 cases with a different set of initial cluster centres, we obtained 3 different clusters at the end.

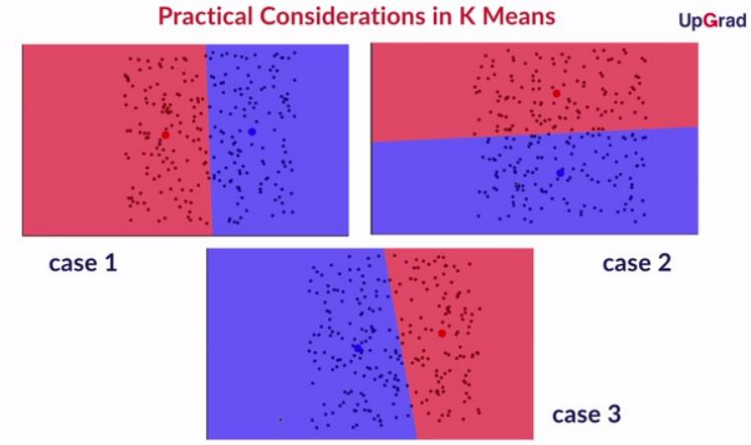


Fig 6: Impact of initial cluster centres on the final result

2.) **Choosing the number of clusters K in advance**

There are a number of pointers that can help us decide the K for our K-means algorithm:-

- The business problem/understanding/constraints
- Mathematically, we aim to maximise the inter-cluster distance and minimise the intra-cluster variance. This can be achieved in R using the R_sq statistic. Here the output of the kmeans() function is utilised. R_sq measures the fraction of the total sum of the squared distance between the data points that can be explained by the sum of squared distance between the clusters. The higher this fraction, the better the clustering. Thus, we plot the curve for the R_sq vs the number of clusters and use the elbow method to find the optimal range of K. You will learn more about it when you implement the K-Means algorithm in R.
- Also, the division of points among different clusters should make business sense and should be feasible and viable.

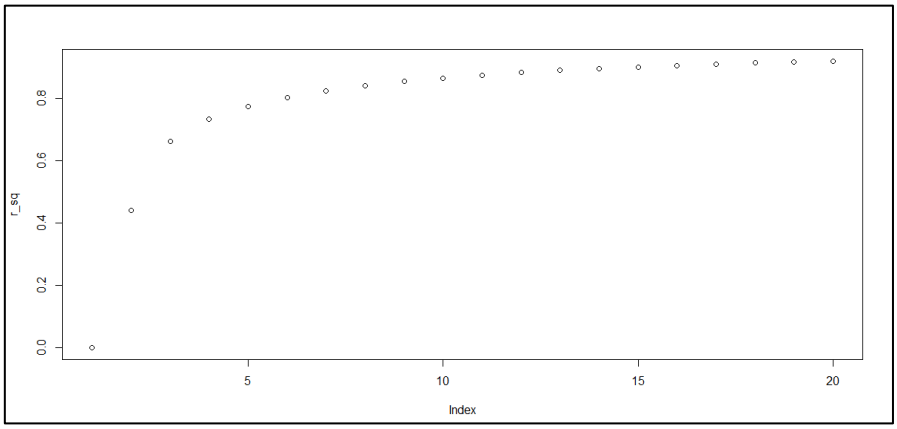


Fig 7: R_sq vs the K (number of clusters)

3.) **Impact of outliers**

Since, the K-Means algorithm tries to allocate each of the data point to one of the clusters, outliers have serious impact on the performance of the algorithm and prevent optimal clustering.

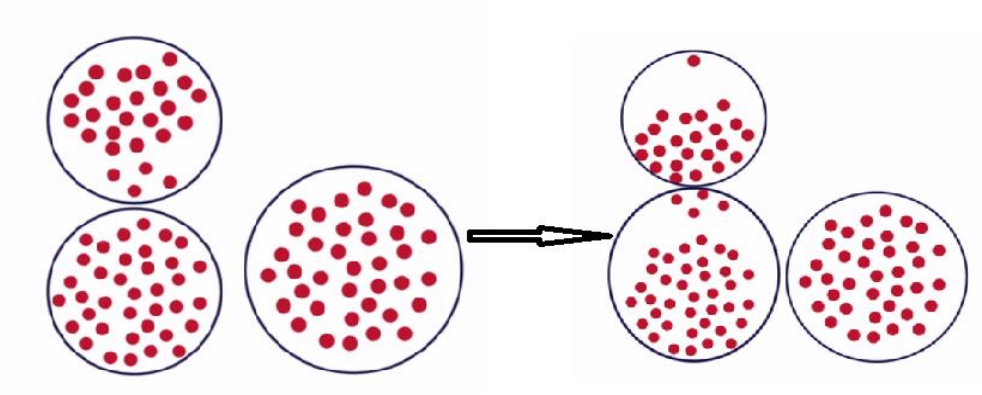


Fig 8: Impact of outliers on clustering

4.) **Standardisation of data**

Standardisation of data, that is, converting them into z-scores with mean 0 and standard deviation 1, is important for 2 reasons in K-Means algorithm:

- Since we need to compute the Euclidean distance between the data points, it is important to ensure that the attributes with a larger range of values do not out-weight the attributes with smaller range. Thus, scaling down of all attributes to the same normal scale helps in this process.
- The different attributes will have the measures in different units. Thus, standardisation helps in making the attributes unit-free and uniform.

5.) Non-applicability with the categorical data

The K-Means algorithm cannot be used when dealing with categorical data as the concept of distance for categorical data doesn't make much sense. So, instead of K-Means, we need to use different algorithms.

Earlier you saw the difference between the classification and the clustering problem. Then you saw the K-Means algorithm as a way to achieve the clusters. Hierarchical Clustering is another algorithm to obtain such clusters.

Hierarchical Clustering algorithm

Given a set of N items to be clustered, the steps in the hierarchical clustering are:

1. Calculate the NxN distance (similarity) matrix, which calculates the distance of each data point from the other.
2. Start by assigning each item to its own cluster, so that if you have N items, you now have N clusters, each containing just one item.

3. Find the closest (most similar) pair of clusters and merge them into a single cluster, so that now you have one less cluster.
4. Compute distances (similarities) between the new cluster and each of the old clusters.
5. Repeat steps 3 and 4 until all items are clustered into a single cluster of size N.

Thus, what we have at the end is the dendrogram, which shows us which data points group together in which cluster at what distance.

Let's consider an example. Let's take 10 points and try to apply hierarchical clustering algorithm over them.

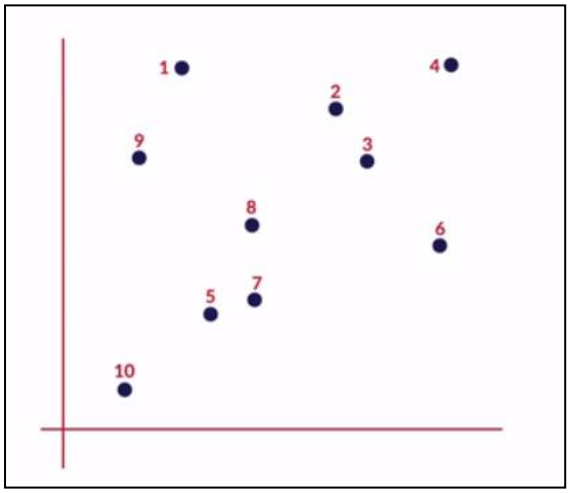


Fig 1: 10 random points

Now initially, we treat each of these points as individual clusters. Thus we begin with 10 different clusters.

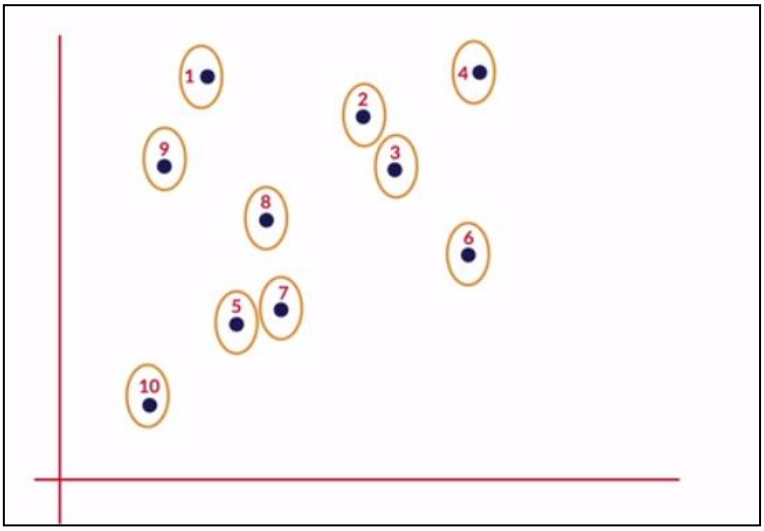


Fig 2: 10 initial clusters

Now we calculate the distance matrix for the 10 clusters, that is, the distance of each cluster from every other cluster. Then we combine the 2 clusters with the minimum distance between them. In this case point 5 & 7 were the closest clusters to each other. Thus, they would be merged first. Correspondingly, they appear at the lowest level in the dendrogram.

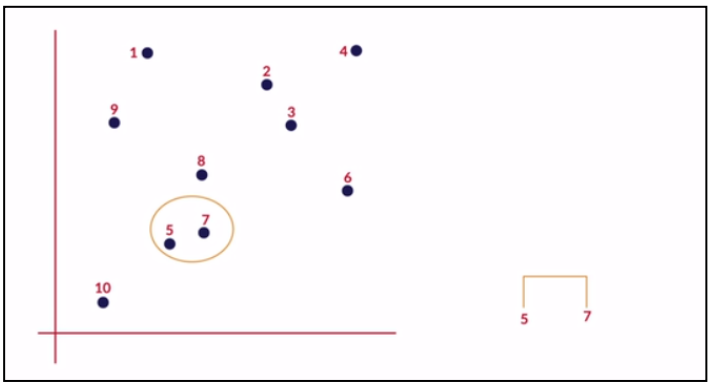


Fig 3: Merging of point 5 & 7 to form a single cluster

Thus now you are left with only 9 clusters. 1 of them has a single element, while 1 of them have 2 elements i.e. 5 & 7. Now again we calculate the distance of each cluster from every other cluster. But here the problem is how can you measure the distance of a cluster having 2 points with a cluster having a single point? It is here that the concept of linkage becomes important. Linkage is the measure of dissimilarity or similarity between the clusters having multiple observations.

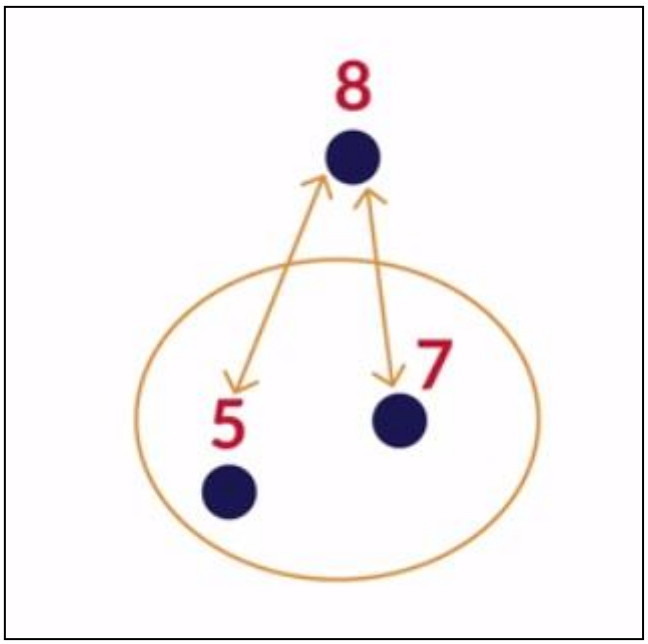


Fig 4: Calculating dissimilarity measure between 2 clusters

Here, we calculate the distance between the points 5 & 8 and then 7 & 8, and the minimum of these 2 distances is taken as the distance between the 2 clusters. Thus, in next iteration, we obtain 8 clusters.

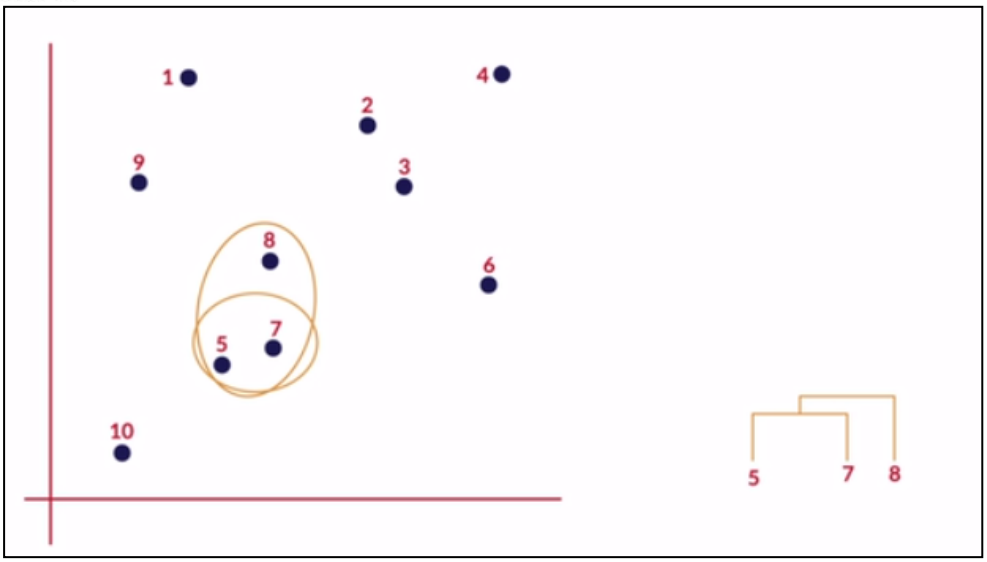


Fig 5: After iteration 2, we have 8 clusters

These iterations continue until we arrive at 1 giant cluster.

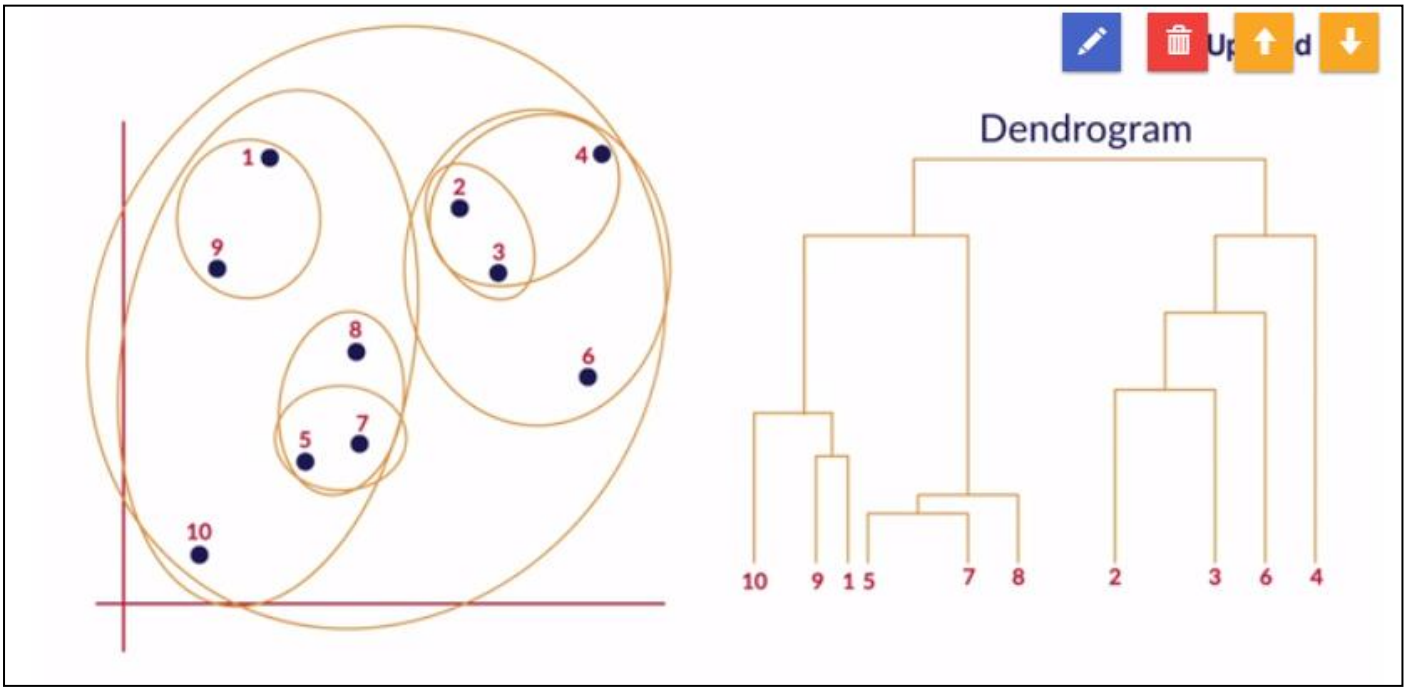


Fig 6: The HC algorithm results into a dendrogram

The result of the hierarchical clustering algorithm is shown by a dendrogram, which starts with all the data points as separate clusters and indicates at what level of dissimilarity any two clusters were joined.

Interpreting the dendrogram

The y-axis of the dendrogram is some measure of the dissimilarity or distance at which clusters join.

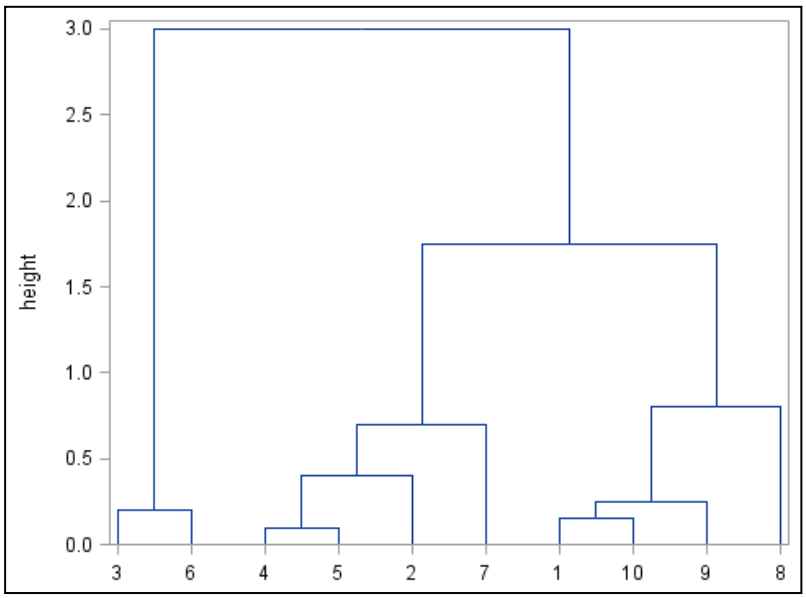
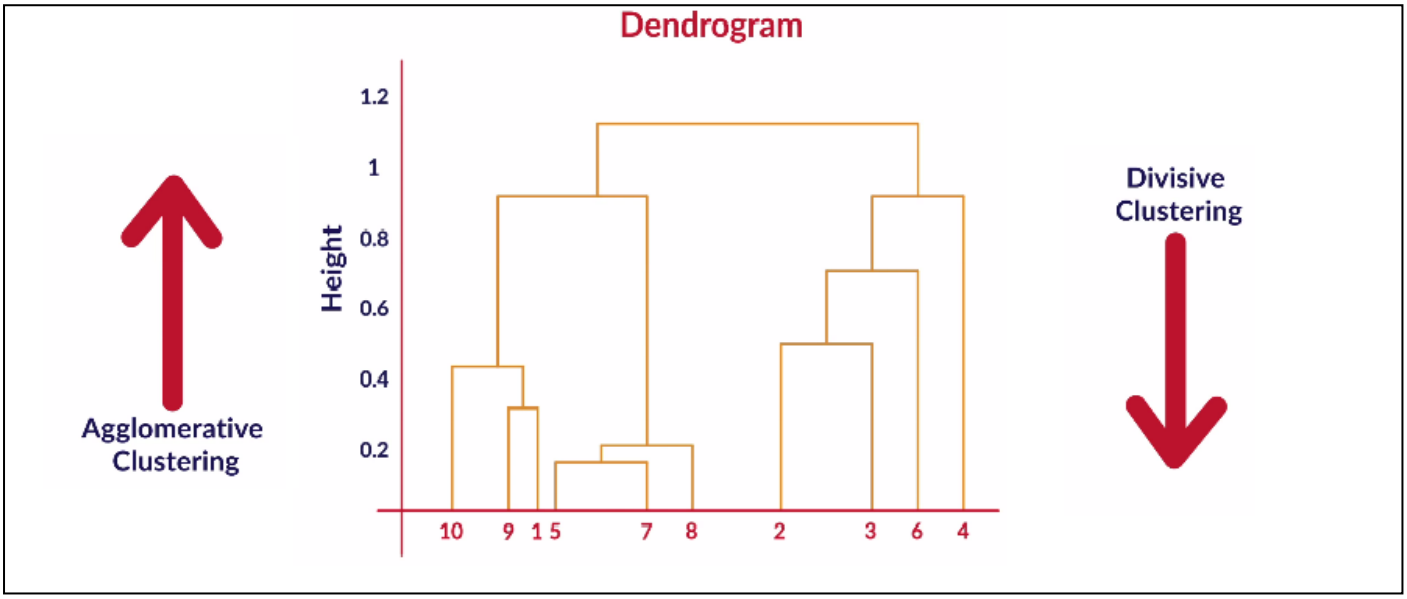


Fig 7: A sample dendrogram

In the dendrogram shown above, samples 4 and 5 are the most similar and join to form the first cluster, followed by samples 1 and 10. The last two clusters to fuse together to form the final single cluster are 3-6 and 4-5-2-7-1-10-9-8.

Determining the number of groups in a cluster analysis is often the primary goal. Typically, one looks for natural groupings defined by long stems. Here, by observation, you can identify that there are 3 major groupings: 3-6, 4-5-2-7 and 1-10-9-8.

We also saw that the hierarchical clustering can proceed in 2 ways - agglomerative and divisive. If we are starting with n distinct clusters and iteratively reach to a point where we have only 1 cluster in the end, it is called agglomerative clustering. On the other hand, if we begin with 1 big cluster and subsequently keep on partitioning this cluster to reach n clusters, each containing 1 element each, it is called divisive clustering.



Cutting the dendrogram

Once we obtain the dendrogram, the clusters can be obtained by cutting the dendrogram at an appropriate level. The number of vertical lines intersecting the cutting line represents the number of clusters.

Thus if we cut the dendrogram at dissimilarity measure of 0.8, we obtain 4 clusters.

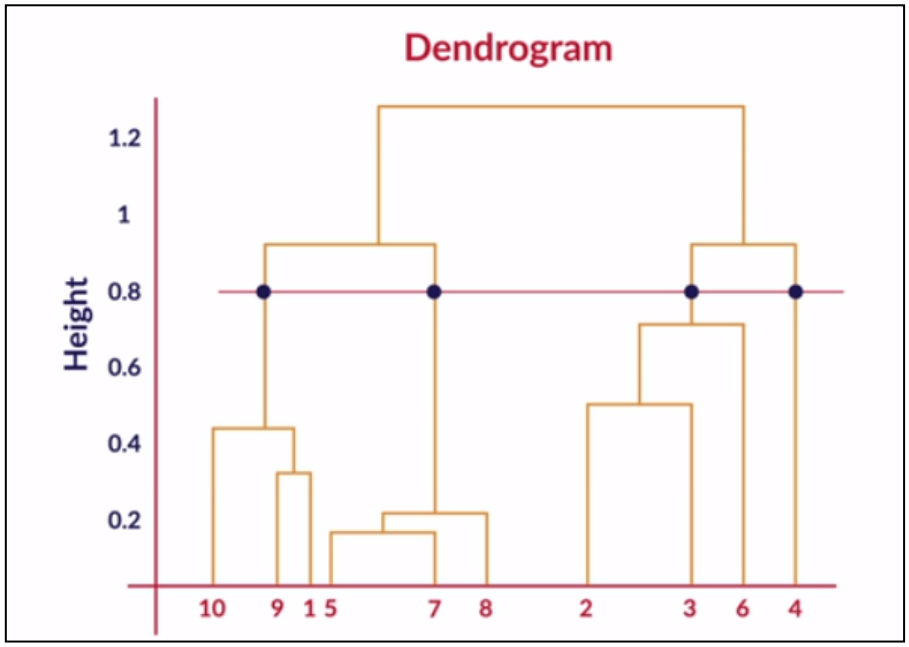


Fig 9: Cutting the dendrogram at 0.8

Instead, if we make the cut at height of 1.2, we get only 2 clusters.

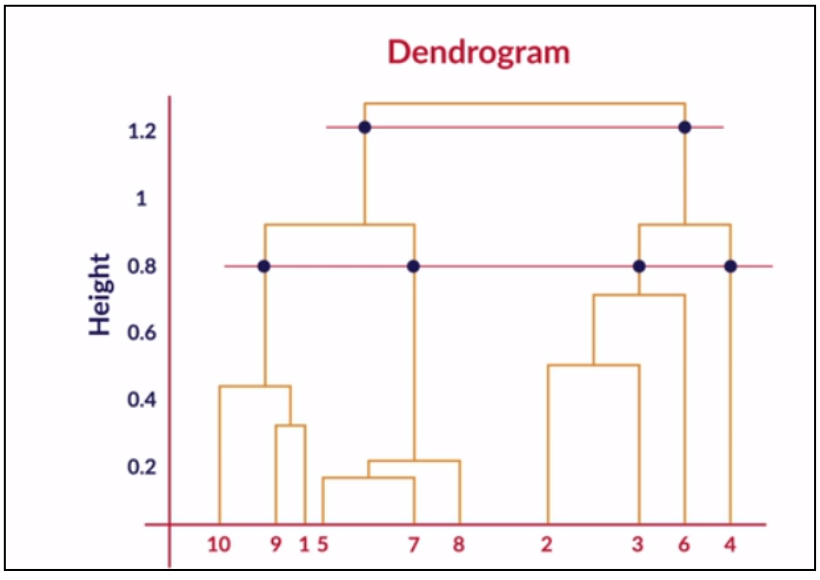


Fig 10: Cutting the dendrogram at 1.2

Types of linkages

In our earlier example, we took the minimum of all the pairwise distances between the data points as the representative of the distance between 2 clusters. This measure of distance is called **single linkage**. Apart from using the minimum, we can use other methods to compute the distance between the clusters. Let's consider the common types of linkages:-

Single Linkage

Here, the distance between 2 clusters is defined as the shortest distance between points in the two clusters

Complete Linkage

Here, the distance between 2 clusters is defined as the maximum distance between any 2 points in the clusters

Average Linkage

Here, the distance between 2 clusters is defined as the average distance between every point of one cluster to every other point of the other cluster.

You also looked at the difference K-Means and Hierarchical clustering and saw how these methods are used in the industry.

Here are some important commands to cluster data that you should remember when clustering the data

- scale(x, center = TRUE, scale = TRUE)
- kmeans(x, centers, iter.max = ___, nstart = ___)
- hclust(d, method = "complete")
- cutree(tree, k = NULL, h = NULL)