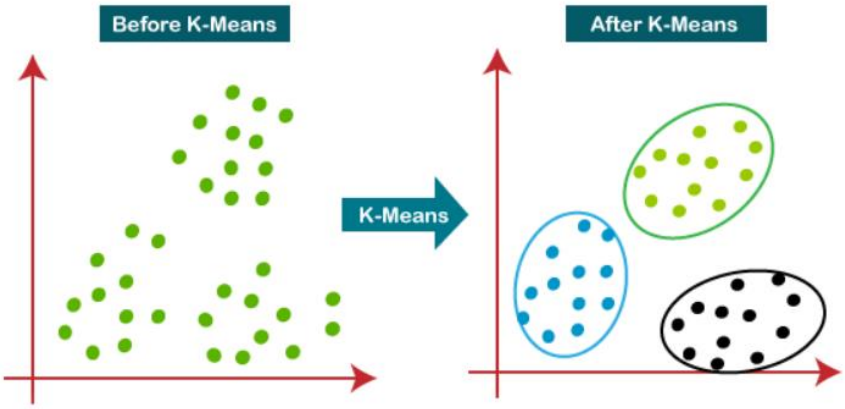
CSE 574 – Introduction to Machine Learning
Assignment - 3

K-means Clustering:

k-means clustering is a method that aims to partition “n” observations into “k” clusters in which each observation belongs to the cluster with the nearest mean (cluster centroid). K, here is the pre-defined number of clusters to be formed by the algorithm. K-means clustering minimizes a metric called within-cluster sum of squares (WCSS). It is a measure of the compactness of the clusters; it represents the sum of squared distances between each data point and its centroid within a cluster.



Advantages:

- Scales well to large datasets.
- Computationally efficient and easy to implement.
- Works well when clusters are well-separated, spherical, and of similar sizes.

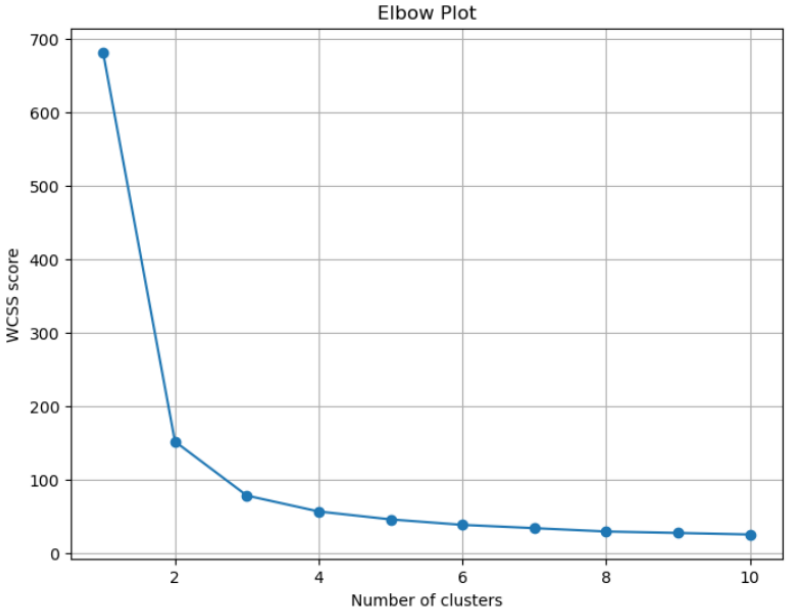
Disadvantages:

- Requires the number of clusters (k) to be specified in advance.
- Sensitive to initial centroid positions, which may lead to different final cluster assignments.

To find the optimal number of clusters there is a technique called the **Elbow method**, which consists of the following steps:

- Execute the k-means clustering on a given dataset for a range of values.
- For each value of k, calculate the WCSS value.
- Plot a graph between WCSS values and the respective number of clusters k.
- The sharp point of bend or a point on the plot, like an arm, will be considered as the best value of k.

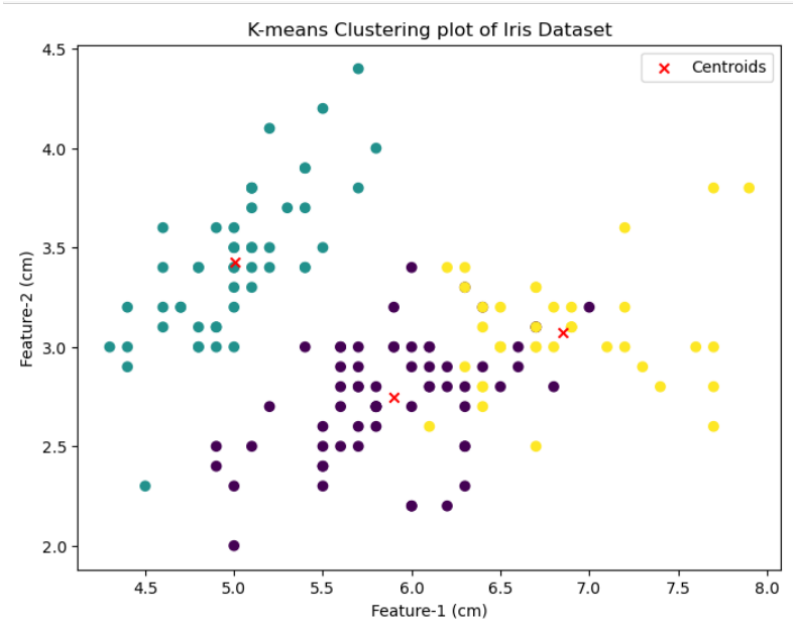
Elbow method:



From the above elbow plot we can see that there is a sharp bend/ turn at k = 3, thus we can say that the optimal number of clusters for the iris data set is three.

Visual Inspection of k-means:

Although Iris dataset has 4 input features which cannot be directly visualized, we can see the results of clustering by plotting them against any 2 random features. The visual allows us to see the centroids and the clusters themselves, which can give us a good intuition about the output of the algorithm.



By randomly selecting two features we plot the results of k-means on a scatter plot. The above plot gives us a good intuition of the clusters formed. Notice that centroids are marked by the little **X**'s. Each color of the data point denotes the cluster that point belongs to and we can see a group of well-defined clusters.

Silhouette score:

The silhouette score is a metric used to calculate the goodness of a clustering technique. It measures how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette score ranges from -1 to 1.

- A score close to +1 indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters.
- A score around 0 indicates that the object is on or very close to the decision boundary between two neighboring clusters.
- A score close to -1 indicates that the object is poorly matched to its own cluster and well matched to neighboring clusters.

With the k-value as 3, the silhouette score for the Iris dataset comes around to **0.553,** which shows a good cluster quality.

Hierarchical Clustering - Agglomerative clustering:

Hierarchical clustering is an unsupervised learning technique used to group similar objects into clusters. It creates a hierarchy of clusters by merging or splitting them based on similarity measures.

It merges similar clusters iteratively, starting with each data point as a separate cluster. This creates a tree-like structure called dendrogram, that shows the natural groupings in the data and provides a visual representation of the relationships between clusters.

There are mainly two types of hierarchical clustering:

- Agglomerative hierarchical clustering – Starts with each data point assigned to one unique cluster. Iteratively merge the closest pair of clusters until only a single cluster is left that contains all the data points.
- Divisive Hierarchical clustering – Starts with all the data points assigned to a single cluster. Iteratively splits the farthest point in the cluster until each cluster only contains a single point.

In Agglomerative clustering, the two closest clusters are merged based on a distance metric until all points belong to a single cluster. The choice of distance metric (such as

Euclidean distance, Manhattan distance, etc.) and linkage criteria (how to measure the distance between clusters) are important parameters in the process.

Single and complete linkages are two common methods used to measure the distance between clusters in hierarchical clustering. Both methods are based on the distances between individual points in different clusters but differ in how they calculate these distances.

Single linkage computes the distance between two clusters as the shortest distance between any point in one cluster and any point in the other cluster.

Complete linkage computes the distance between two clusters as the maximum distance between any point in one cluster and any point in the other cluster.

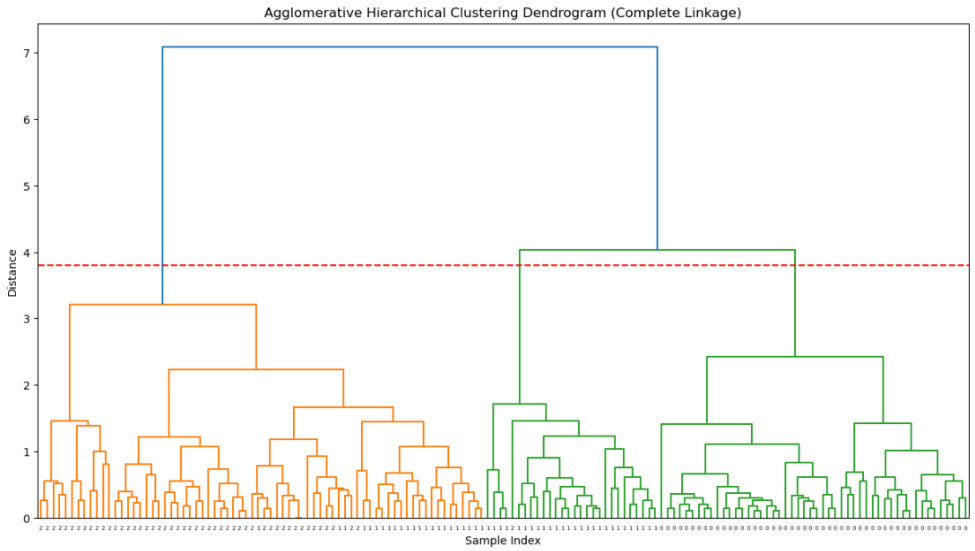
Advantages:

- Does not require the number of clusters to be specified in advance.
- Robust to noise and outliers, as it builds clusters based on pairwise distances.

Disadvantages:

- Computationally expensive, especially for large datasets
- Dendrograms can be difficult to interpret, especially for large data sets with many data points.

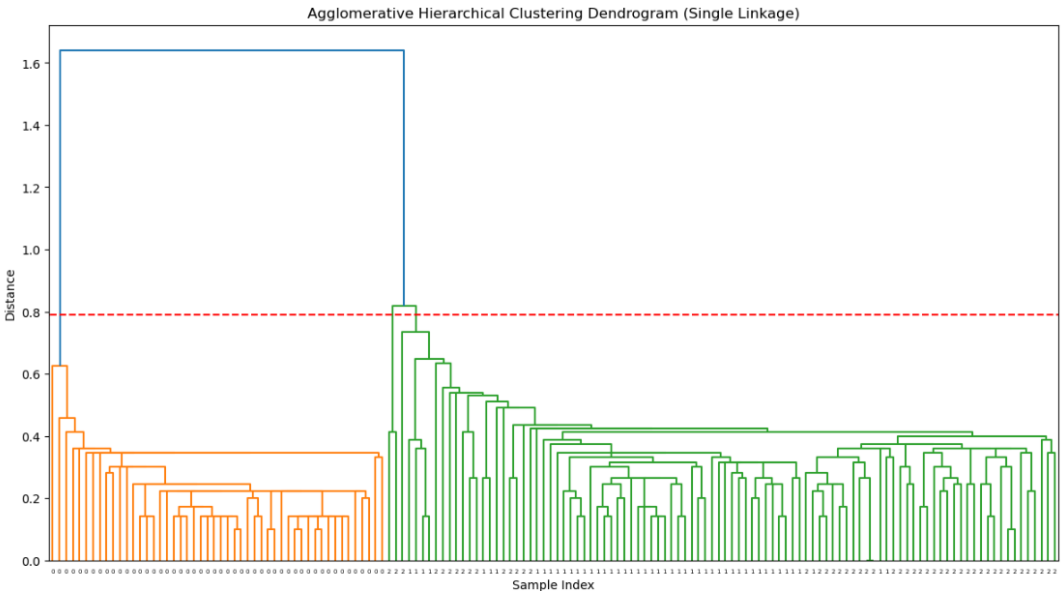
Visual Inspection of Dendrograms:



Above is a plot of the dendrogram formed by performing Agglomerative clustering on Iris data set. As per the definition, we aim to merge all data points into a single cluster. This dendrogram is formed by a Euclidian distance metric and follows complete linkage.

The number of clusters is determined by choosing a horizontal line(dotted-red) that intersects the vertical lines at a specific height. This horizontal line represents the threshold for clustering, and the number of clusters is determined by counting the number of vertical lines it intersects.

Similarly, a dendrogram for Iris dataset that follows Euclidian distance and single linkage looks like:



Density-Based Spatial Clustering of Applications with Noise (DBSCAN):

DBSCAN is a popular clustering algorithm used to identify clusters in a dataset based on the density of data points. Unlike K-means, which assumes that clusters are spherical and of similar density, DBSCAN can find clusters of arbitrary shape and size.

DBSCAN has two parameters:

Epsilon (ε): Defines the radius within which to search for neighboring points. Points within ε distance of each other are considered neighboring points.

Minimum-Points (MinPts): Minimum number of points required to form a dense region.

<u>DBSCAN Algorithm</u>:

- Core Points: For each point in the dataset, DBSCAN counts the number of points within ε distance. If a point has at least MinPts neighboring points (including itself), it is considered a core point.
- Directly Density Reachable: Two points are said to be directly density reachable if they are within ε distance of each other.

- Density Reachable: A point is density reachable from another point if there is a chain of core points connecting them, where each successive core point is within ε distance of the previous one.
- Density Connected: Two points are density connected if there is a common core point that both points are density reachable from.
- Clusters: DBSCAN forms clusters by assigning each core point and its density-reachable neighbors to the same cluster. Points that are not core points or reachable from any core points are considered noise and are not assigned to any cluster.

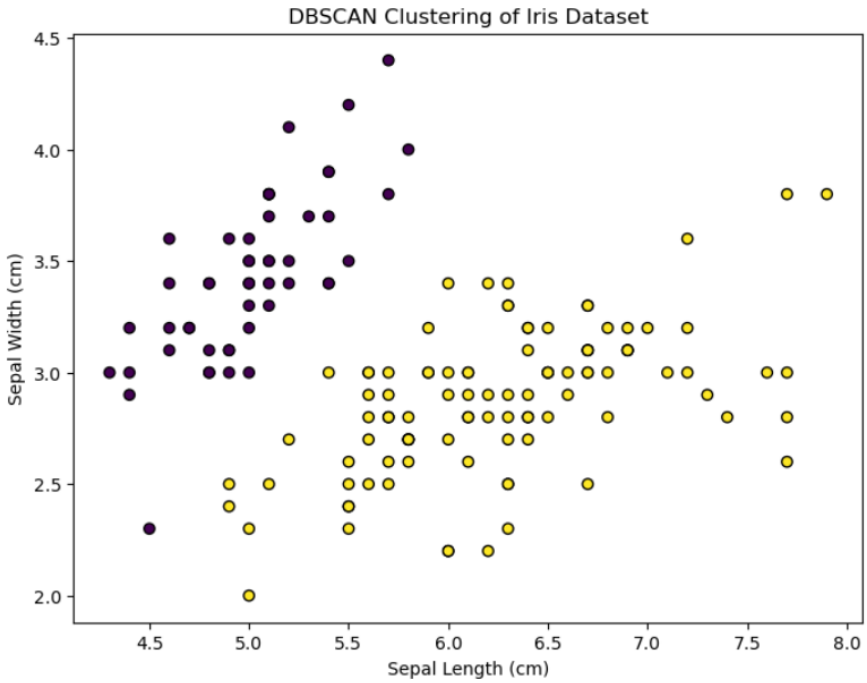
Advantages:

- Can identify clusters of arbitrary shape and size.
- Does not require the number of clusters to be specified in advance.
- Robust to noise and outliers, as it identifies low-density regions as noise.

Disadvantages:

- Requires the tuning of two parameters: epsilon (ε) and minimum points.
- Computationally expensive, especially for large datasets.
- Performance may degrade in high-dimensional spaces.

Visual Inspection of DBSCAN:



The scatter plot below is plotted based on the optimum values for epsilon and minimum points of 0.831 and 2 respectively for the Iris dataset. We can see a clear distinction between the decision boundary and separation of classes.

The optimum hyperparameters are found by performing a grid-search algorithm over different values of epsilon and minimum-points based on the Silhouette score.

K-means vs Hierarchical vs DBSCAN:

To check for the quality of the fit between the 3 algorithms we can use the Silhouette score. The higher the score, the better the quality of the clustering.

Based on the results, highlighted in the code we can say that we have K-means as the best score of 0.553 and was clearly able to form better clusters.