# PRACTICAL – 10

**Aim:** Implementation and analysis of clustering algorithms like K-means, Agglomerative.

## Theory:

**K- Means**

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

The algorithm will categorize the items into k groups or clusters of similarity. To calculate that similarity, we will use the Euclidean distance as measurement.

The algorithm works as follows:

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

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

**The above algorithm in pseudocode is as follows:**

Initialize k means with random values

--> For a given number of iterations:

--> Iterate through items:

--> Find the mean closest to the item by calculating the Euclidean distance of the item with each of the means

--> Assign item to mean

--> Update mean by shifting it to the average of the items in that cluster

**Steps:**

- Read Data
- Initialize Means
- Euclidean Distance
- Update Means
- Classify Items
- Find Means
- Find Clusters

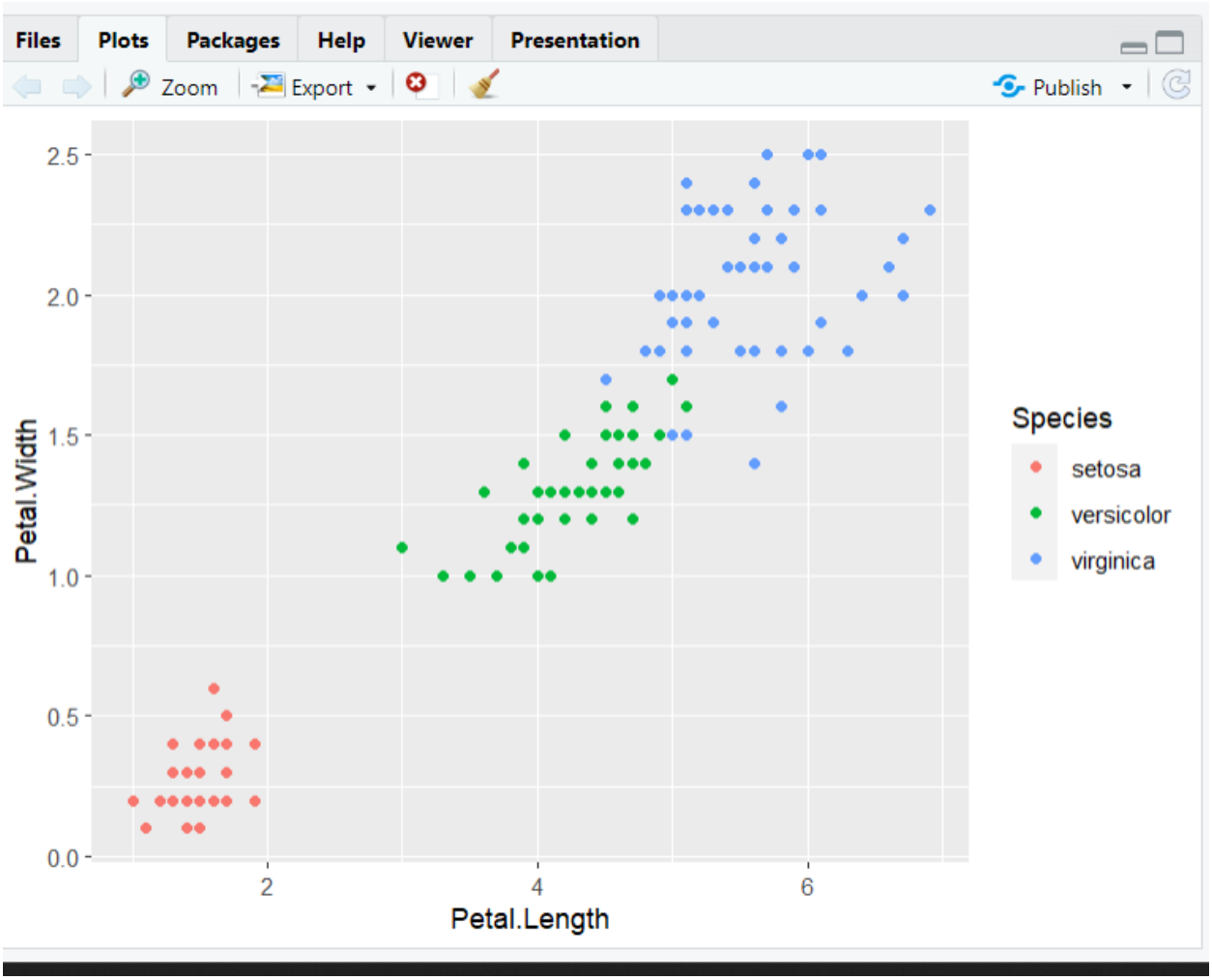
**Setting working directory.**

```
> getwd()
[1] "C:/Users/Suraj/Documents"
>
```

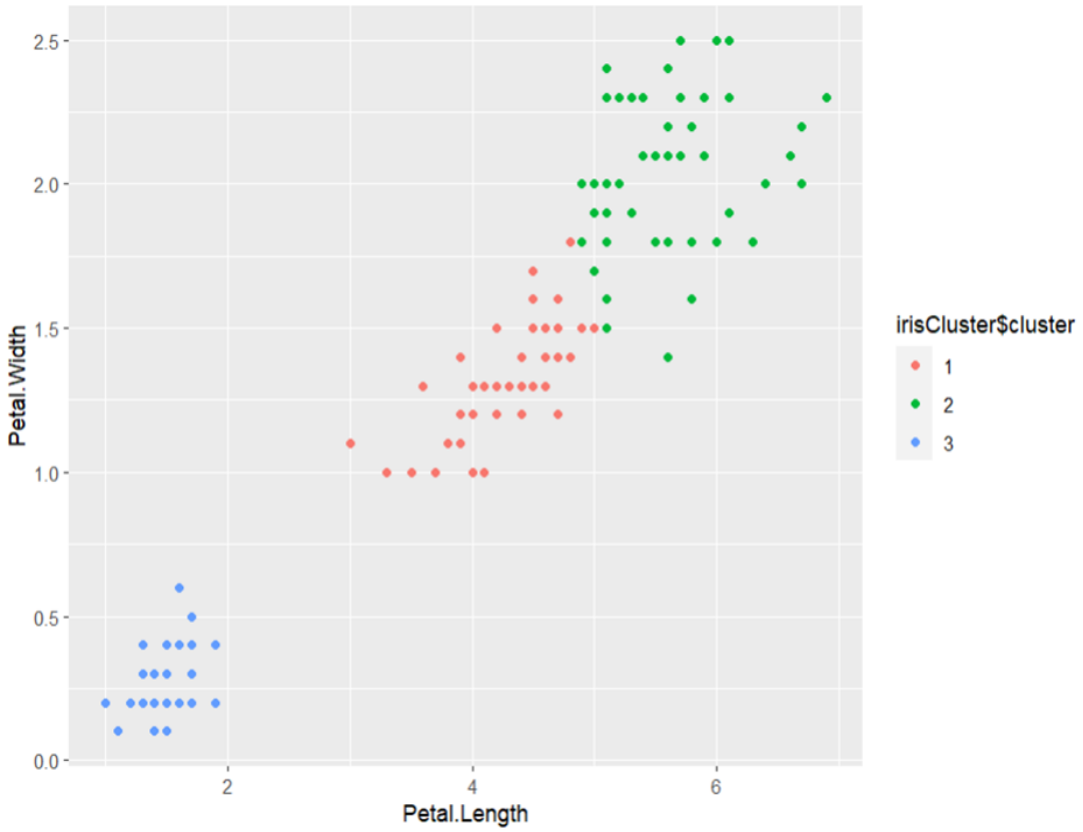
**Loading iris data set and printing from top**

```
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> install.packages("ggplot2")
```

```
> library(ggplot2)
> ggplot(iris,aes(Petal.Length, Petal.Width,color=Species))+geom_point()
>
```


Files
Plots
Packages
Help
Viewer
Presentation
Zoom
Export
Publish
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
0.5
0.0
2
4
6
Species
setosa
versicolor
virginica

```
> ggplot(iris,aes(Petal.Length, Petal.Width,color=Species))+geom_point()
>  set.seed(20)
> irisCluster <- kmeans(iris[,3:4],3,nstart = 20)
> irisCluster
K-means clustering with 3 clusters of sizes 52, 48, 50

Cluster means:
  Petal.Length Petal.Width
1     4.269231    1.342308
2     5.595833    2.037500
3     1.462000    0.246000

Clustering vector:
  [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 [31] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1
 [61] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 1 1
 [91] 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1
[121] 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2

Within cluster sum of squares by cluster:
[1] 13.05769 16.29167  2.02200
 (between_SS / total_SS =  94.3 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"
[5] "tot.withinss" "betweenss"    "size"         "iter"
[9] "ifault"
> table(irisCluster$cluster,iris$Species)

    setosa versicolor virginica
  1      0         48         4
  2      0          2        46
  3     50          0         0
> table(irisCluster$cluster, iris$Species)

    setosa versicolor virginica
  1      0         48         4
  2      0          2        46
  3     50          0         0
> irisCluster$cluster <- as.factor(irisCluster$cluster)
> ggplot(iris, aes(Petal.Length, Petal.Width,color=irisCluster$cluster))+geom_point()
>
```




**Agglomerative Clustering**

The agglomerative clustering is the most common type of hierarchical clustering used to group objects in clusters based on their similarity. It's also known as *AGNES* (*Agglomerative Nesting*). The algorithm starts by treating each object as a singleton cluster. Next, pairs of clusters are successively merged until all clusters have been merged into one big cluster containing all objects. The result is a tree-based representation of the objects, named *dendrogram*.

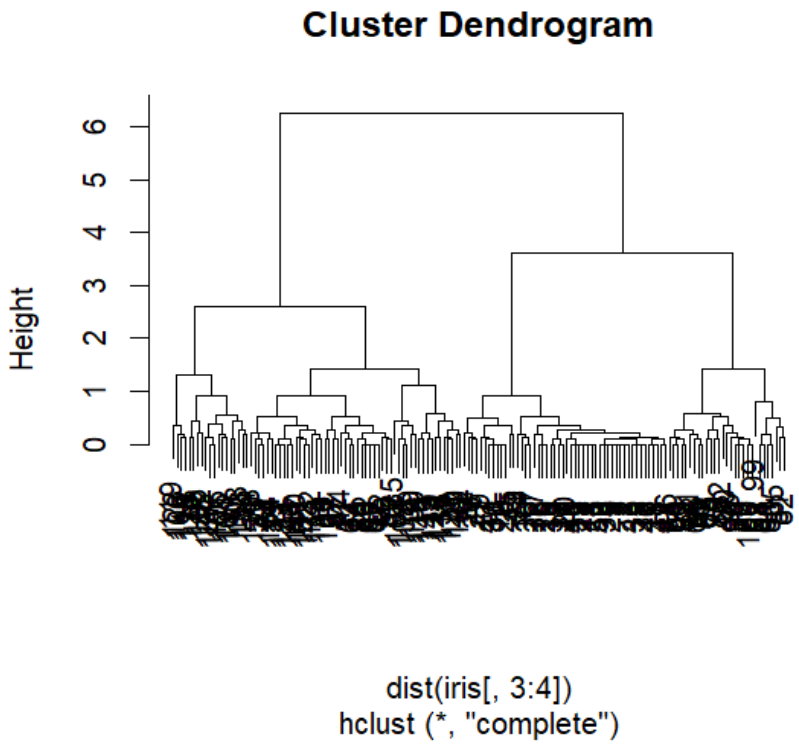
Agglomerative clustering works in a "bottom-up" manner. That is, each object is initially considered as a single-element cluster (leaf). At each step of the algorithm, the two clusters that are the most similar are combined into a new bigger cluster (nodes). This procedure is iterated until all points are member of just one single big cluster (root) (see figure below).

The inverse of agglomerative clustering is *divisive clustering*, which is also known as DIANA (*Divise Analysis*) and it works in a "top-down" manner. It begins with the root, in which all objects are included in a single cluster. At each step of iteration, **the most** heterogeneous cluster is divided into two. The process is iterated until all objects are in their own cluster

**Steps to agglomerative hierarchical clustering**

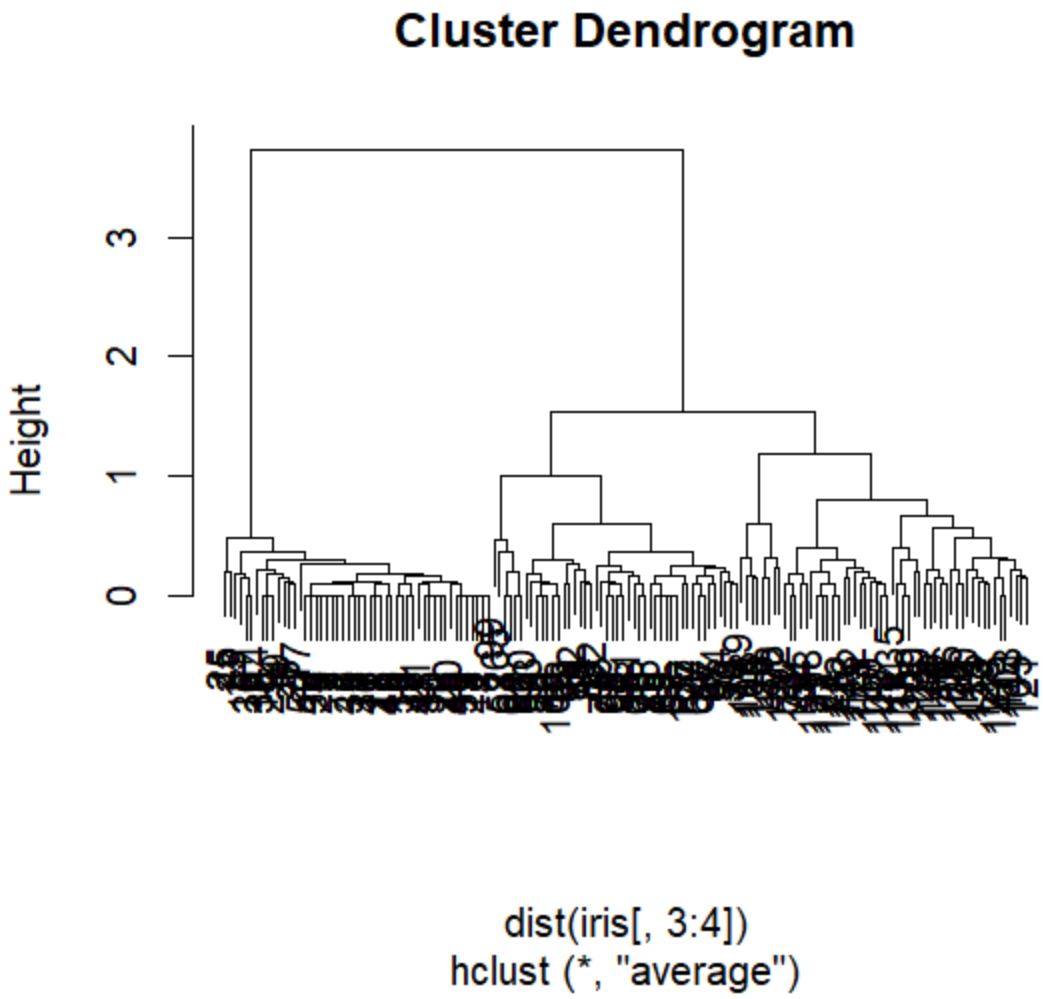
We'll follow the steps below to perform agglomerative hierarchical clustering using R software:

1. Preparing the data
2. Computing (dis)similarity information between every pair of objects in the data set.
3. Using linkage function to group objects into hierarchical cluster tree, based on the distance information generated at step 1. Objects/clusters that are in close proximity are linked together using the linkage function.
4. Determining where to cut the hierarchical tree into clusters. This creates a partition of the data.

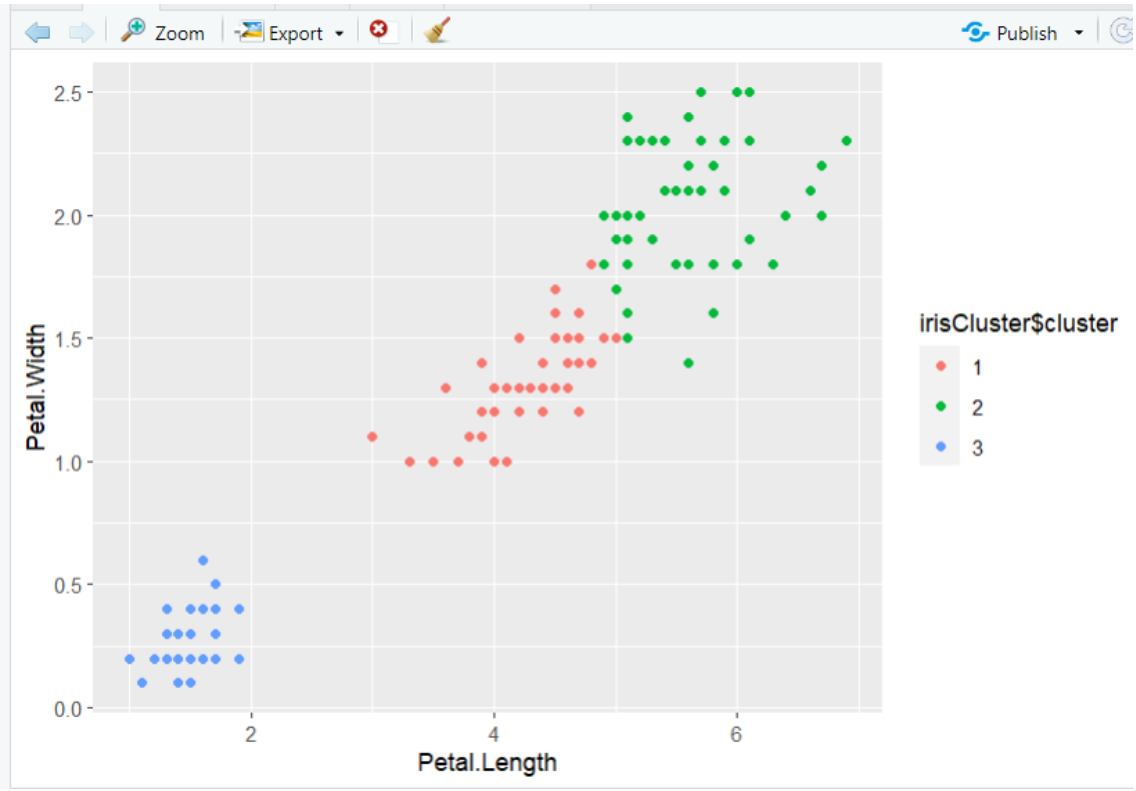



```
> clusterCut<- cutree(clusters, 3)
> table(clusterCut, iris$Species)

clusterCut setosa versicolor virginica
         1     50          0         0
         2      0         21        50
         3      0         29         0
```

```
> clusters <- hclust(dist(iris[,3:4]), method = 'average')
> plot(clusters)
>
```




```
  3     50          0         0
> table(irisCluster$cluster, iris$Species)

   setosa versicolor virginica
1      0         48         4
2      0          2        46
3     50          0         0
> irisCluster$cluster <- as.factor(irisCluster$cluster)
> ggplot(iris, aes(Petal.Length, Petal.Width,color=irisCluster$cluster))+geom_point()
>
```




## Conclusion:

I have successfully implemented and analyzed clustering algorithms like K-means and agglomerative clustering.