ASSIGNMENT NO.3:

# Aim – Implement K-means using R programming.

# Theory:

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

# K-Means Algorithm: K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of predefined 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 to only one group that has similar properties. 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 the particular k-center, create a cluster. Hence each cluster has data points with some commonalities, and it is away from other clusters. 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 than the input dataset).

# Step-3: Assign each data point to its 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 reassigning each data point to the new closest centroid of each cluster.

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

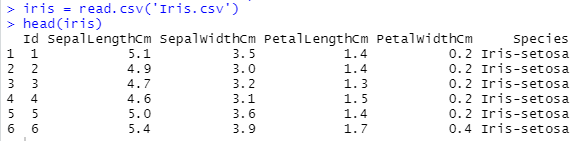
# Step-7: The model is ready.

# Elbow Method: In cluster analysis, the elbow method is a heuristic used in determining the number of clusters in a data set. The method consists of plotting the explained variation as a function of the number of clusters and picking the elbow of the curve as the number of clusters to use. Using the "elbow" or "knee of a curve" as a cutoff point is a common heuristic in mathematical optimization to choose a point where diminishing returns are no longer worth the additional cost. In clustering, this means one should choose a number of clusters so that adding another cluster doesn't give a much better modeling of the data.

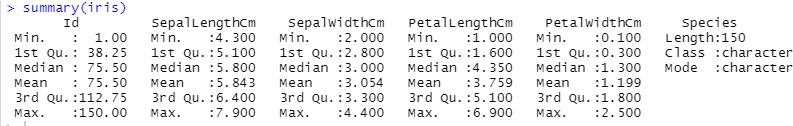
# The Elbow method looks at the total WSS as a function of the number of clusters: One should choose a number of clusters so that adding another cluster doesn't improve much better the total WSS. The optimal number of clusters can be defined Calculate the Within-Cluster-Sum of Squared Errors (WSS) for different values of k, and choose the k for which WSS first starts to diminish. In the plot of WSS-versus-k, this is visible as an elbow. Within-Cluster-Sum of Squared Errors sounds a bit complex. Let’s break it down: The Squared Error for each point is the square of the distance of the point from its representation i.e. its predicted cluster center.

**R Program CODE:**

**Reading the data**

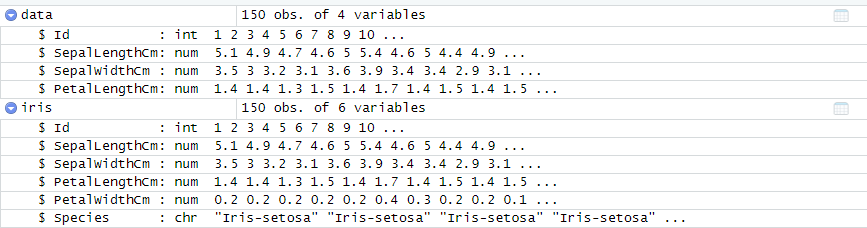


**Summary of data**



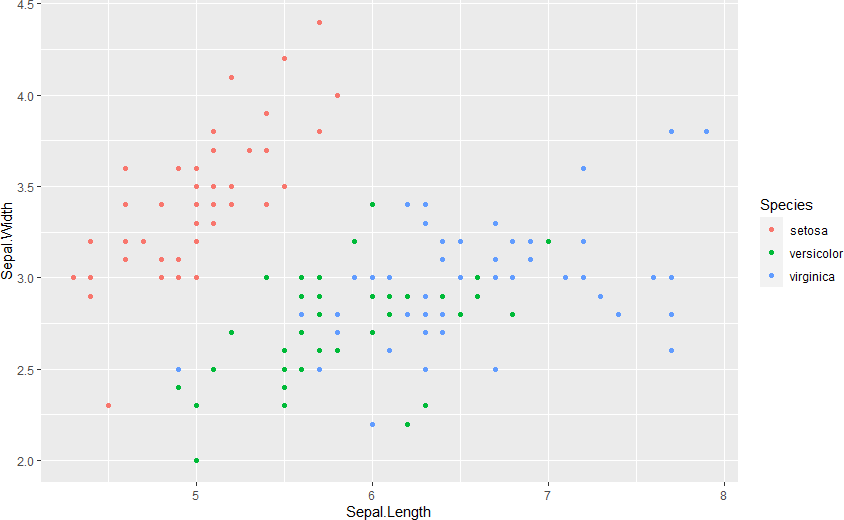
**Selecting the columns for clustering excluding the classes of dataset**

data <- select(iris, c(1:4))

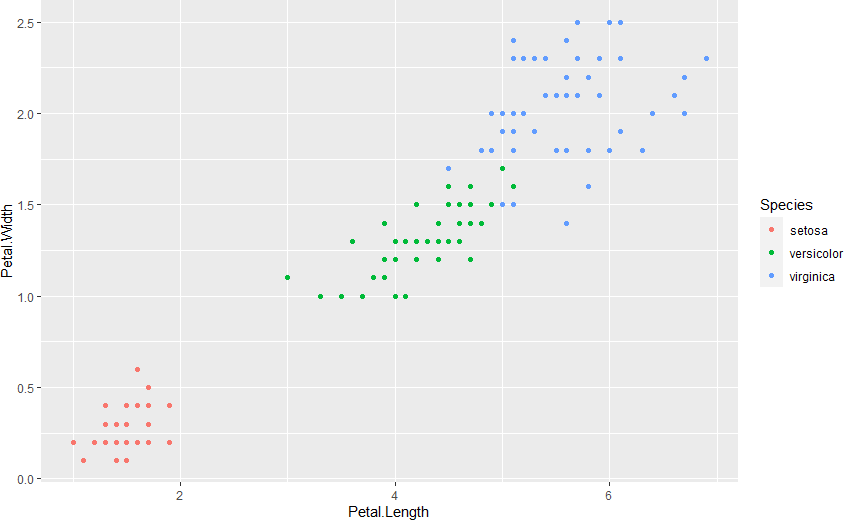


# Lets check the datapoint variance using scatterplot

ggplot(iris,aes(x = Sepal.Length, y = Sepal.Width, col= Species)) + geom\_point()



ggplot(iris,aes(x = Petal.Length, y = Petal.Width, col= Species)) + geom\_point()



# Initializing the K means object

K.max = 10

wss = rep(NA, K.max = 10) nClust = list()

# lets find the WCSS score to find the no of clusters

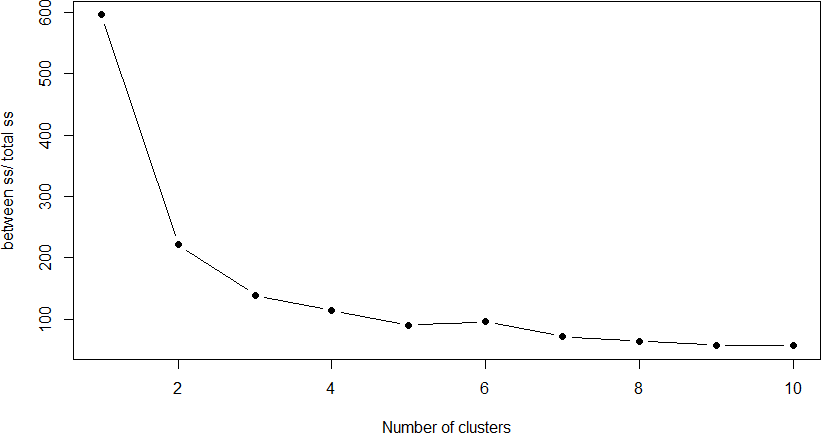
for(i in 1:K.max){

irisclasses= kmeans(irisScale,i) #object stored in iris class

wss[i] = irisclasses$tot.withinss #in irisclass,total wss distance allocated

nClust[[i]] = irisclasses$size #size allocated to componant of list you have created

}

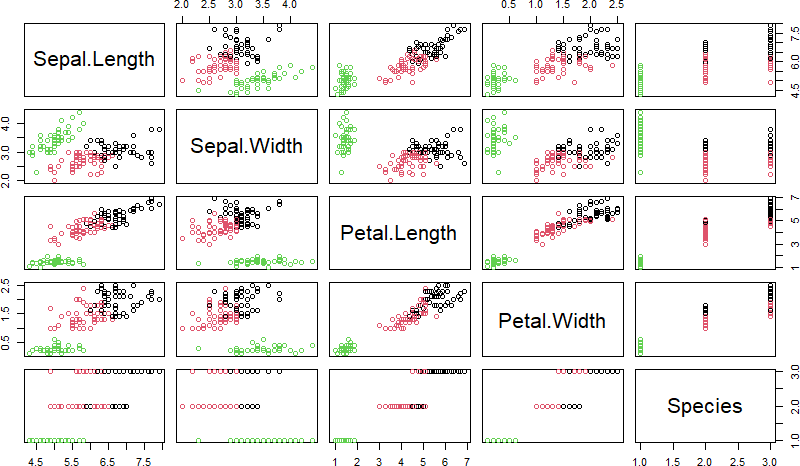


***In the above plot WCSS can observe that if we use 3 cluster then it is best choice for us because after k=3 the WCSS is gradual.***

# Lets train the kmeans for k = 3 and will observe the output

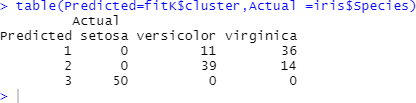
fitK = kmeans(irisScale, 3) str(fitK)

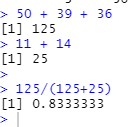
plot(iris,col = fitK$cluster)



# Let’s find how much accurate our k means is by understanding the actual and clustered data

table(Predicted=fitK$cluster,Actual =iris$Species)





***By above observation we can say that our algorithm has 83.33% capability to cluster the new data point in accurate cluster***

**Conclusion:**

WSS helps to identify the K value