**Date:08/03/2021**

# Practical No: 04

**Aim:** Practical of Clustering.

**Theory:**

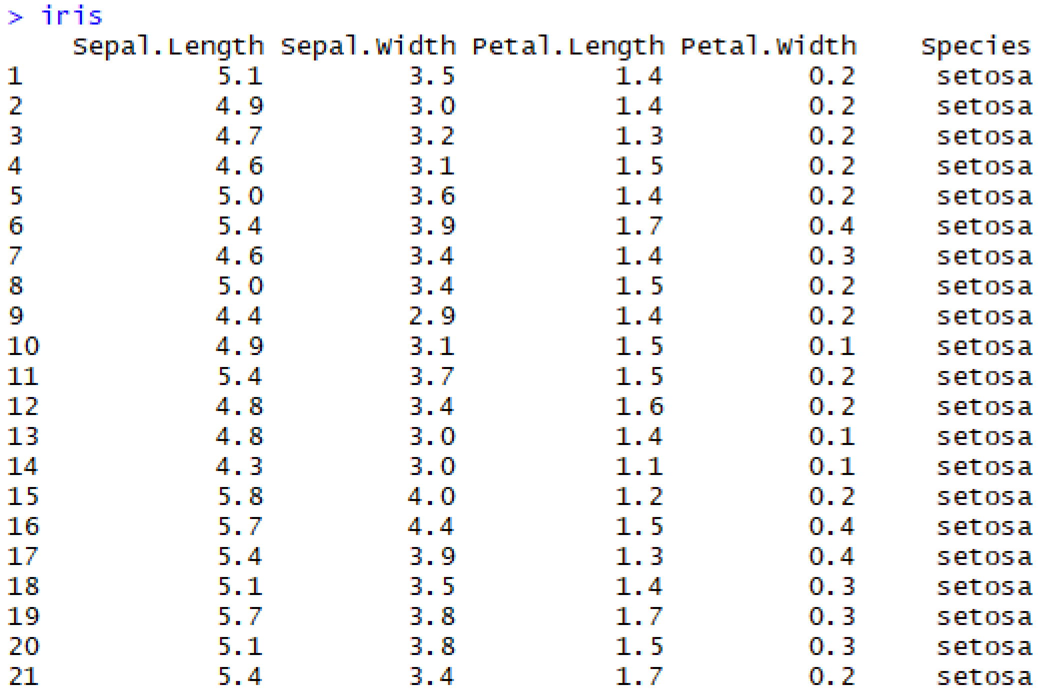
**1. Clustering:-** Clustering is a technique of data segmentation that partitions the data into several groups based on their similarity. Basically, we group the data through a statistical operation. These smaller groups that are formed from the bigger data are known as clusters.

**2. k-means clustering:-** *k*-means clustering is a method of vector quantization, originally from signal processing, that aims to partition *n* observations into *k* clusters in which each observation belongs to the cluster with the nearest mean (cluster centers or cluster centroid), serving as a prototype of the cluster.

**3. What is Iris dataset?**
The ***Iris* flower data set** or **Fisher's *Iris* data set** is a multivariate data set introduced by the British statistician, eugenicist, and biologist Ronald Fisher in his 1936 paper *The use of multiple measurements in taxonomic problems* as an example of linear discriminant analysis.

**Input:**

**1)iris**

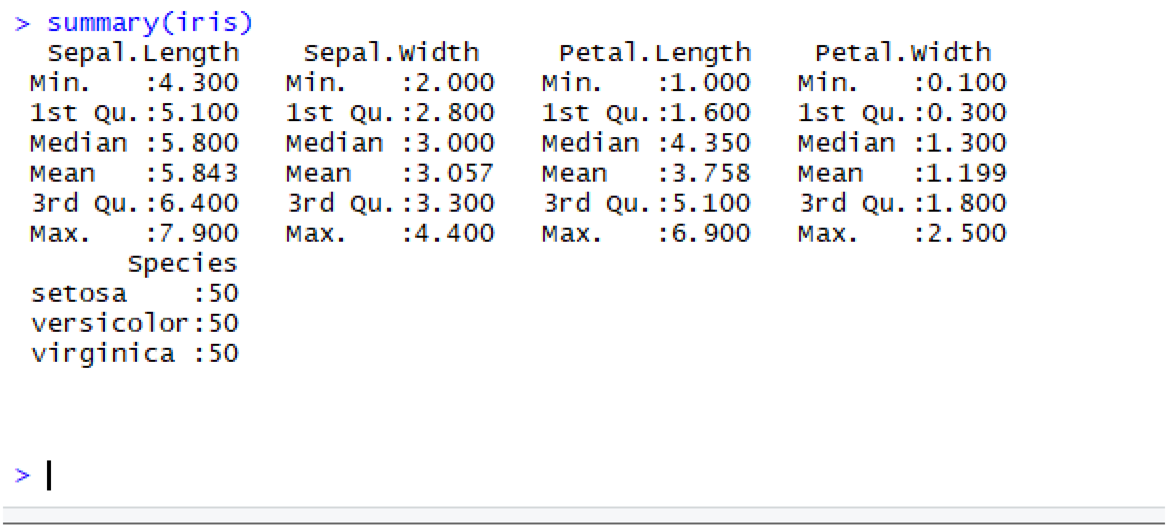


```
> iris
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1            5.1         3.5          1.4         0.2     setosa
2            4.9         3.0          1.4         0.2     setosa
3            4.7         3.2          1.3         0.2     setosa
4            4.6         3.1          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
7            4.6         3.4          1.4         0.3     setosa
8            5.0         3.4          1.5         0.2     setosa
9            4.4         2.9          1.4         0.2     setosa
10           4.9         3.1          1.5         0.1     setosa
11           5.4         3.7          1.5         0.2     setosa
12           4.8         3.4          1.6         0.2     setosa
13           4.8         3.0          1.4         0.1     setosa
14           4.3         3.0          1.1         0.1     setosa
15           5.8         4.0          1.2         0.2     setosa
16           5.7         4.4          1.5         0.4     setosa
17           5.4         3.9          1.3         0.4     setosa
18           5.1         3.5          1.4         0.3     setosa
19           5.7         3.8          1.7         0.3     setosa
20           5.1         3.8          1.5         0.3     setosa
21           5.4         3.4          1.7         0.2     setosa
```

2)**summary** is a generic function used to produce result summaries of the results of various model fitting functions.

**INPUT:**

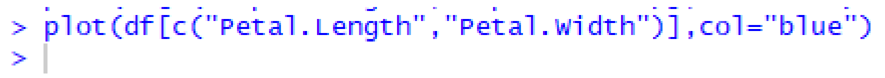
> summary(iris)



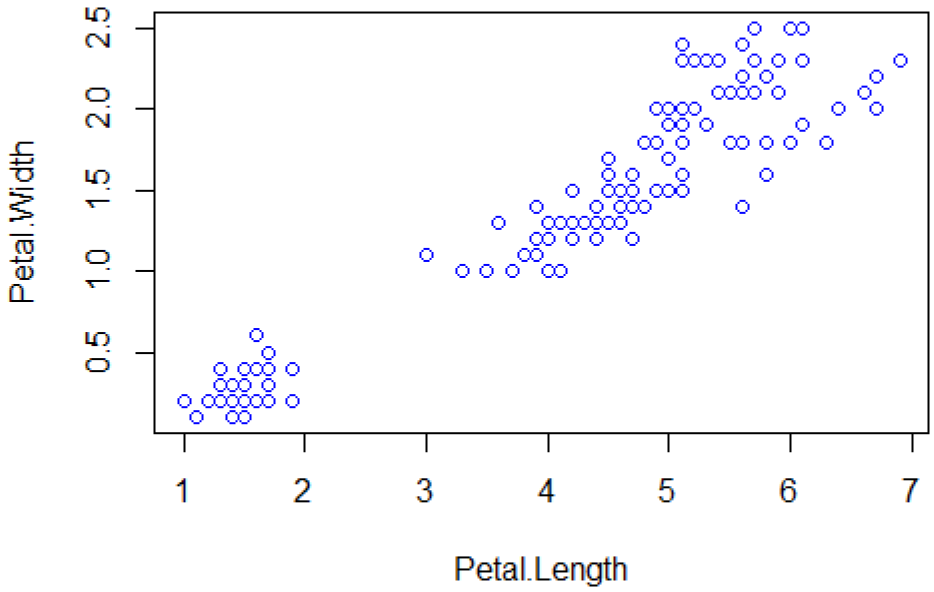
```
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
 Median :5.800   Median :3.000   Median :4.350   Median :1.300  
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500  
       Species  
 setosa    :50  
 versicolor:50  
 virginica :50  


> |
```

- ❖ Plot function is used to plot graphs in R Studio .If we don't specify the graph type the R Studio consider default type i.e. Scatter Plot.

**3)INPUT:**

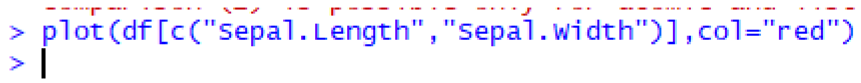


```
> plot(df[c("Petal.Length","Petal.Width")],col="blue")
> 
```

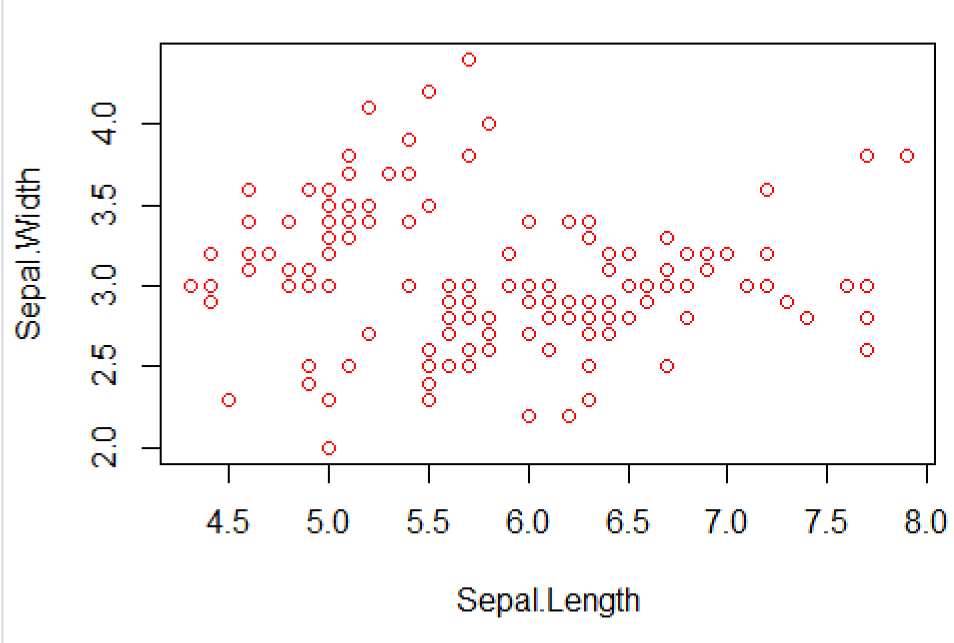


## INPUT:

**4) Plot is a Generic function for plotting of R objects. Here we plot a scatterplot of (Sepal Length,SepalWidth) as points.**



```
> plot(df[c("Sepal.Length","Sepal.Width")],col="red")
> 
```

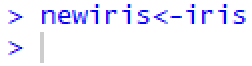


- ❖ In R studio for assigning a dataset or value to variable we use **(<-)** arrow with dash(minus sign).

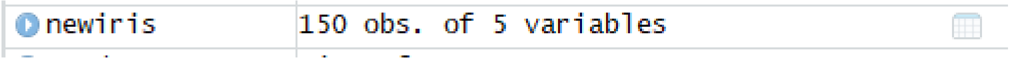
## INPUT:

**5) We copy iris dataset into a new variable newiris for future manipulation.**

> newiris<- iris



```
> newiris<-iris
> 
```

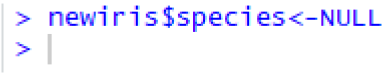


```
newiris          150 obs. of 5 variables
```

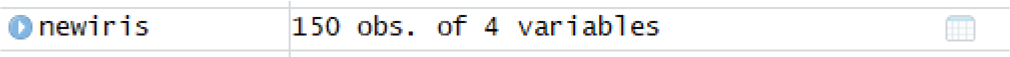
## INPUT:

**6) Here we disable the species column by providing NULL values to all rows. Command: newiris$Species<-NULL**

> newiris$Species<-NULL



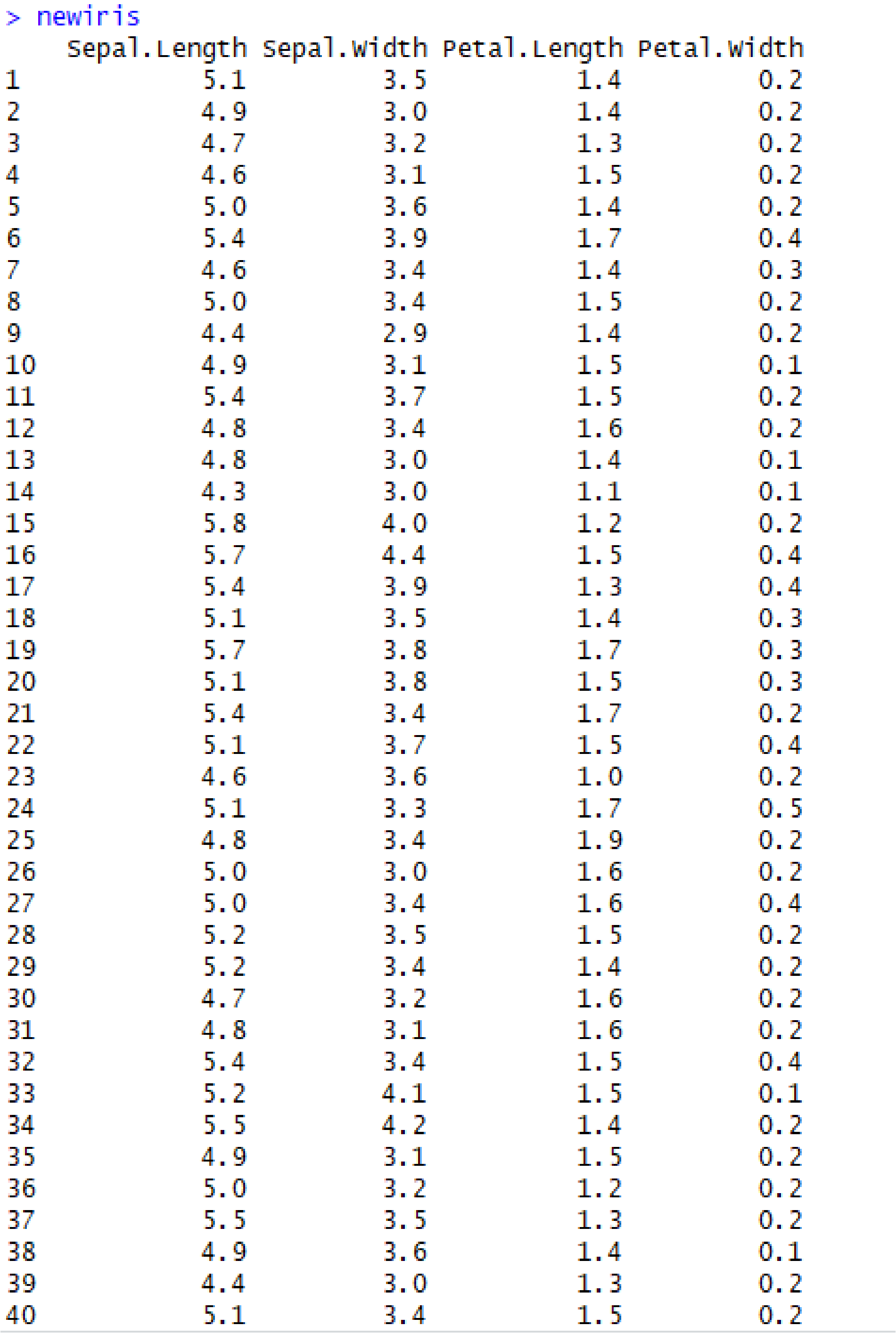
```
> newiris$species<-NULL
> 
```



```
newiris          150 obs. of 4 variables
```

7)INPUT:

> newiris



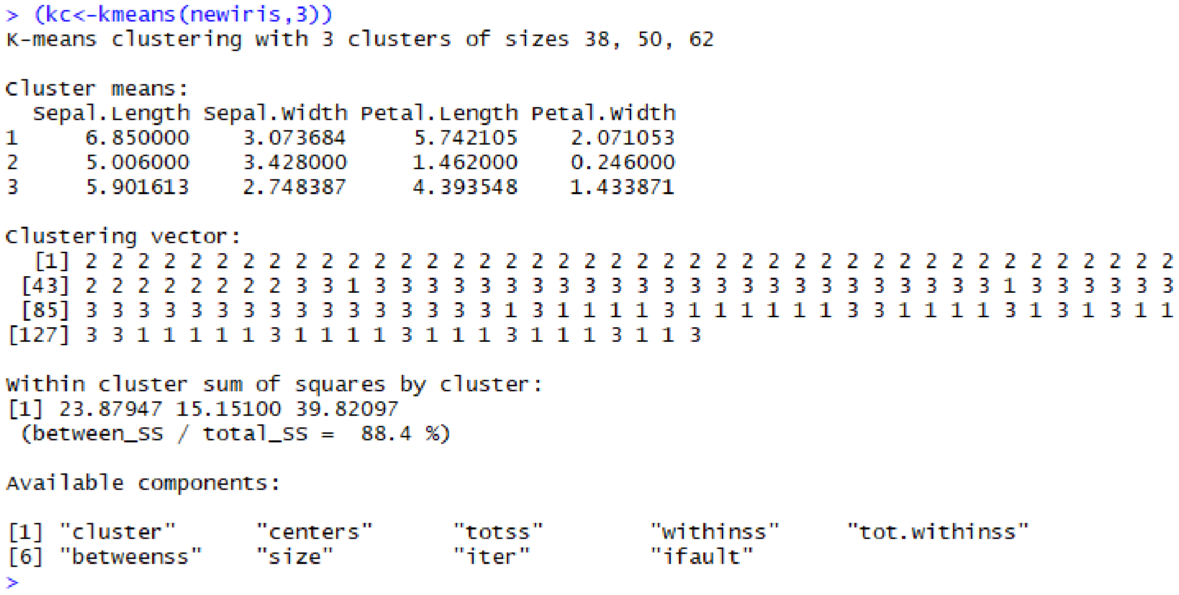
```
> newiris
    Sepal.Length Sepal.Width Petal.Length Petal.Width
1            5.1         3.5          1.4         0.2
2            4.9         3.0          1.4         0.2
3            4.7         3.2          1.3         0.2
4            4.6         3.1          1.5         0.2
5            5.0         3.6          1.4         0.2
6            5.4         3.9          1.7         0.4
7            4.6         3.4          1.4         0.3
8            5.0         3.4          1.5         0.2
9            4.4         2.9          1.4         0.2
10           4.9         3.1          1.5         0.1
11           5.4         3.7          1.5         0.2
12           4.8         3.4          1.6         0.2
13           4.8         3.0          1.4         0.1
14           4.3         3.0          1.1         0.1
15           5.8         4.0          1.2         0.2
16           5.7         4.4          1.5         0.4
17           5.4         3.9          1.3         0.4
18           5.1         3.5          1.4         0.3
19           5.7         3.8          1.7         0.3
20           5.1         3.8          1.5         0.3
21           5.4         3.4          1.7         0.2
22           5.1         3.7          1.5         0.4
23           4.6         3.6          1.0         0.2
24           5.1         3.3          1.7         0.5
25           4.8         3.4          1.9         0.2
26           5.0         3.0          1.6         0.2
27           5.0         3.4          1.6         0.4
28           5.2         3.5          1.5         0.2
29           5.2         3.4          1.4         0.2
30           4.7         3.2          1.6         0.2
31           4.8         3.1          1.6         0.2
32           5.4         3.4          1.5         0.4
33           5.2         4.1          1.5         0.1
34           5.5         4.2          1.4         0.2
35           4.9         3.1          1.5         0.2
36           5.0         3.2          1.2         0.2
37           5.5         3.5          1.3         0.2
38           4.9         3.6          1.4         0.1
39           4.4         3.0          1.3         0.2
40           5.1         3.4          1.5         0.2
```

**8) kmeans Performs k-means clustering on a data matrix, with k cluster centers.**

**COMMAND**: (kc<-kmeans(newiris,3))

**INPUT:**

> (kc<-kmeans(newiris,3))

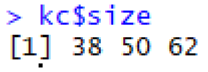


```
> (kc<-kmeans(newiris,3))
K-means clustering with 3 clusters of sizes 38, 50, 62

Cluster means:
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     6.850000    3.073684     5.742105    2.071053
2     5.006000    3.428000     1.462000    0.246000
3     5.901613    2.748387     4.393548    1.433871

Clustering vector:
  [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [43] 2 2 2 2 2 2 2 2 3 3 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3
 [85] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 1 1 1 1 3 1 1 1 1 1 1 3 3 1 1 1 1 3 1 3 1 3 1 1
[127] 3 3 1 1 1 1 1 3 1 1 1 1 3 1 1 1 3 1 1 1 3 1 1 3

Within cluster sum of squares by cluster:
[1] 23.87947 15.15100 39.82097
 (between_SS / total_SS =  88.4 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"      
>
```

**9) kc$size gives the number of points in each cluster.**

**INPUT:**

> kc$size



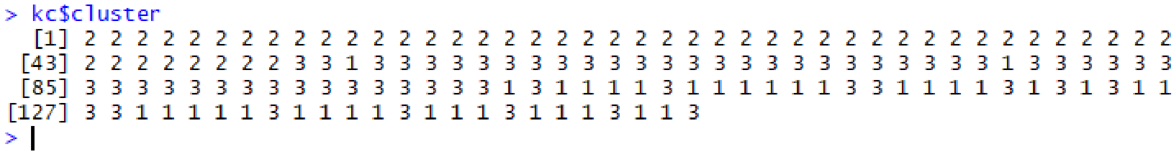
```
> kc$size
[1] 38 50 62
```

**10) kc$cluster** A vector of integers (from 1:k) indicating the cluster to which each point is allocated.

**COMMAND**:kc$cluster

**INPUT:**

> kc$cluster



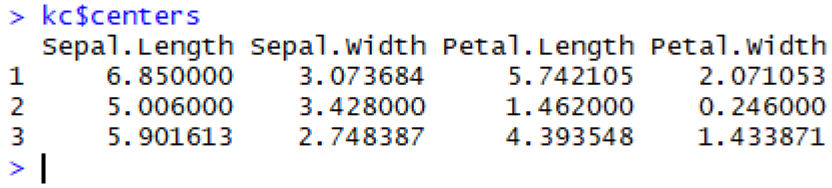
```
> kc$cluster
  [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [43] 2 2 2 2 2 2 2 2 3 3 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3
 [85] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 1 1 1 1 3 1 1 1 1 1 1 3 3 1 1 1 1 3 1 3 1 3 1 1
[127] 3 3 1 1 1 1 1 3 1 1 1 1 3 1 1 1 3 1 1 1 3 1 1 3
> |
```

**11) kc$centers** returns a matrix of cluster centres.

[**COMMAND**:kc$centers

**INPUT:**

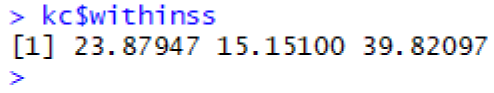
> kc$centers



```
> kc$centers
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     6.850000    3.073684     5.742105    2.071053
2     5.006000    3.428000     1.462000    0.246000
3     5.901613    2.748387     4.393548    1.433871
> 
```

12) **kc$withinss** returns a Vector of within-cluster sum of squares, one component per cluster.

**COMMAND :kc$withinss**
**INPUT:**
> kc$withinss

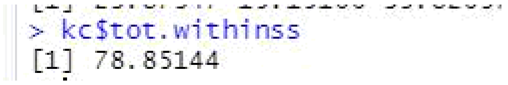


```
> kc$withinss
[1] 23.87947 15.15100 39.82097
> 
```

13) **kc$tot.withinss** returns the Total within-cluster sum of squares, i.e.sum(withinss)

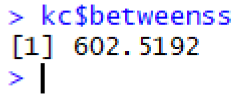
**COMMAND** :kc$tot.withinss

**Output:**



```
> kc$tot.withinss
[1] 78.85144
```

**14) kc$betweenss returns the between-cluster sum of squares, i.e.totss-tot.withinss.**
**INPUT:**
> kc$betweenss

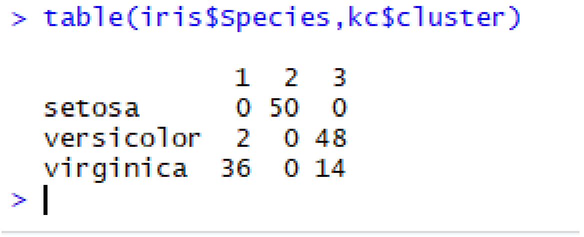


```
> kc$betweenss
[1] 602.5192
> 
```

**15) table uses the cross-classifying factors to build a contingency table of the counts at each com bination of factor levels.Here we construct a table with species wise breakdown of clusters**

**COMMAND: table(iris$Species,kc$cluster)**

**INPUT:**
> table(iris$Species,kc$cluster)

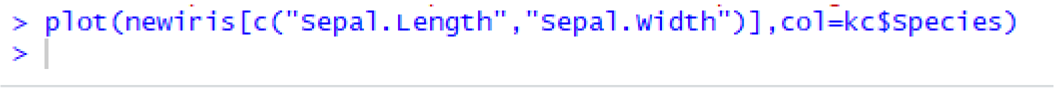


```
> table(iris$Species,kc$cluster)
            
              1  2  3
  setosa      0 50  0
  versicolor  2  0 48
  virginica  36  0 14
> 
```

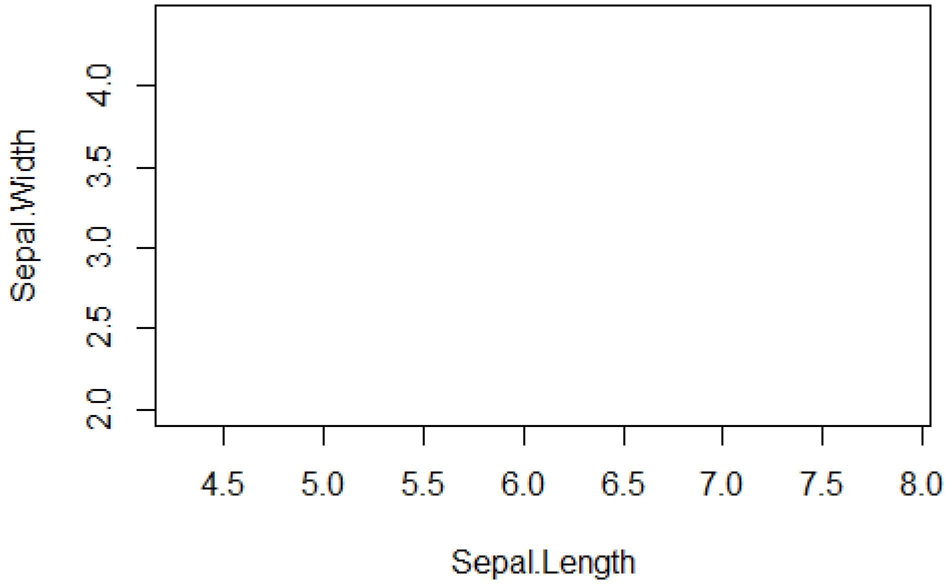
**16) COMMAND:**

plot(newiris[c("Sepal.Length","Sepal.Width")],col=kc$Species)

**INPUT:**



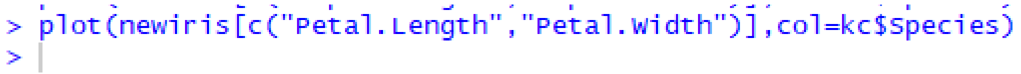
```
> plot(newiris[c("Sepal.Length","Sepal.Width")],col=kc$Species)
> 
```



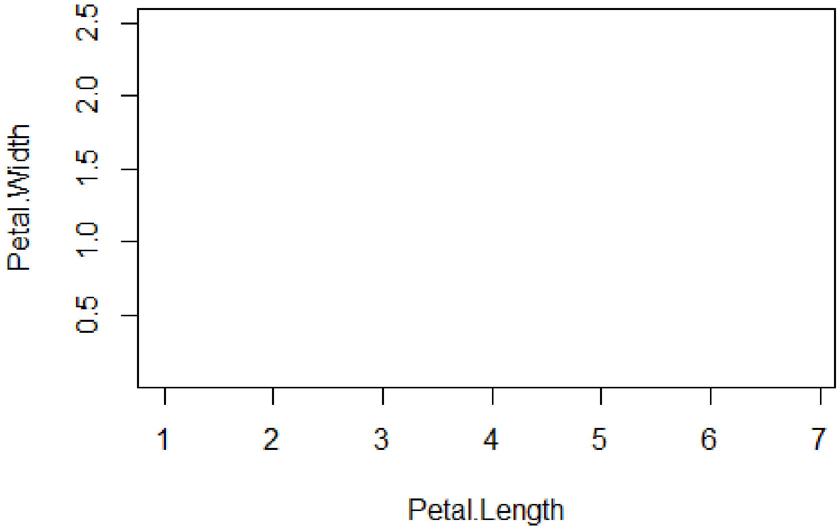
**17) COMMAND:**

plot(newiris[c("Petal.Length","Petal.Width")],col=kc$Species)

**INPUT:**



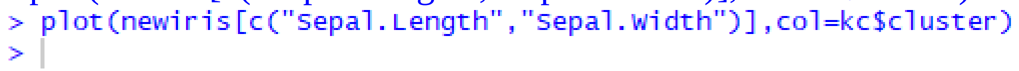
```
> plot(newiris[c("Petal.Length","Petal.Width")],col=kc$Species)
> 
```



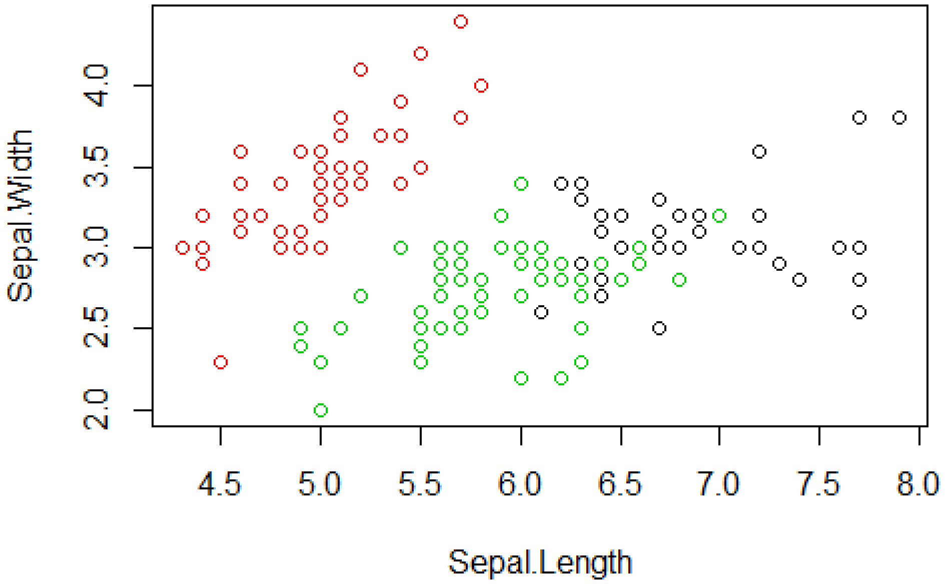
**18) COMMAND:**
**plot(newiris[c("Sepal.Length","Sepal.Width")],col=kc$cluster)**

**INPUT:**

```
> plot(newiris[c("Sepal.Length","Sepal.Width")],col=kc$cluster)
```



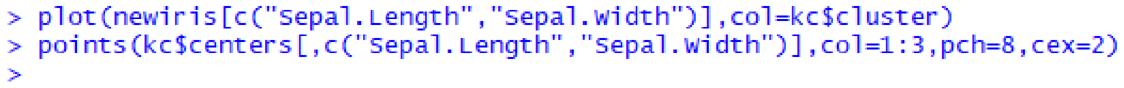
```
> plot(newiris[c("Sepal.Length","Sepal.Width")],col=kc$cluster)
>
```



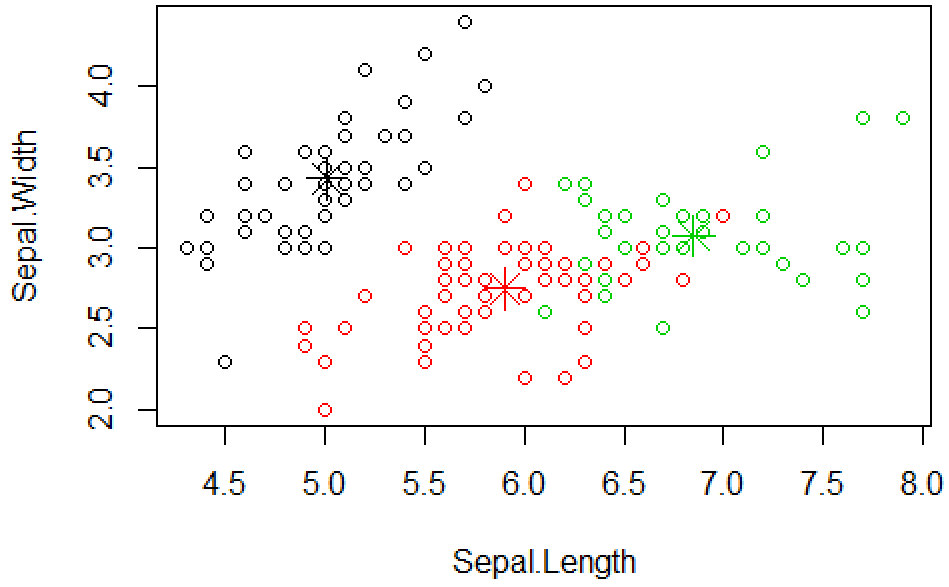
## 19) COMMAND:

**plot(newiris[c("Petal.Length","Petal.Width")],col=kc$cluster)INPUT:**

```
> plot(newiris[c("Sepal.Length","Sepal.Width")],col=kc$cluster)
> points(kc$centers[,c("Sepal.Length","Sepal.Width")],col=1:3,pch=8,cex=2)
```



```
> plot(newiris[c("Sepal.Length","Sepal.Width")],col=kc$cluster)
> points(kc$centers[,c("Sepal.Length","Sepal.Width")],col=1:3,pch=8,cex=2)
>
```



**20) points** is a generic function to draw a sequence of points at the specified coordinates. The specified character(s) are plotted, centered at the coordinates.

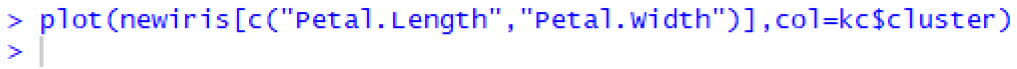
**COMMAND:**

plot(newiris[c("Sepal.Length", "Sepal.Width")], col=kc$cluster)

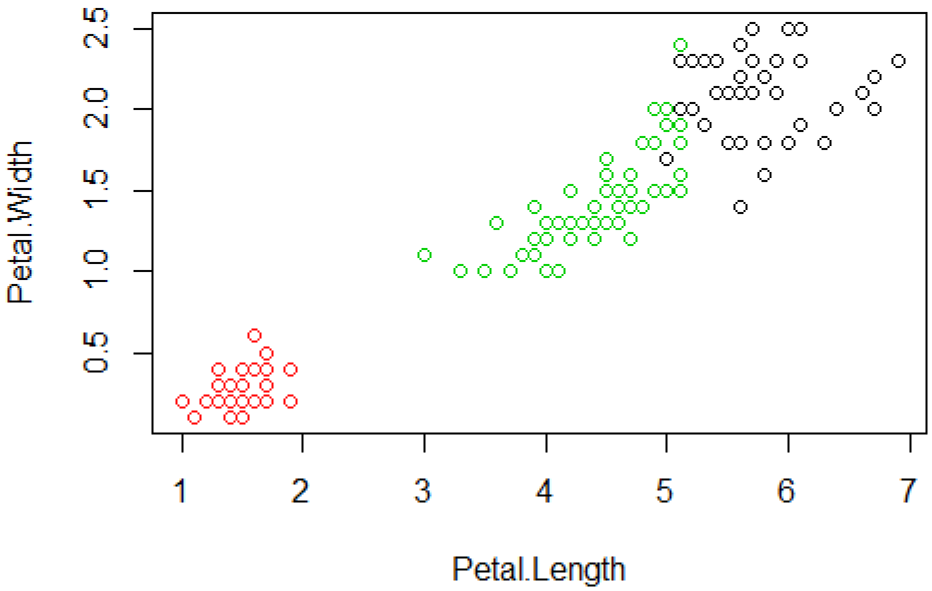
points(kc$centers[,c("Sepal.Length", "Sepal.Width")], col=1:3, pch=8, cex=2)

**INPUT:**

> plot(newiris[c("Petal.Length","Petal.Width")],col=kc$cluster)

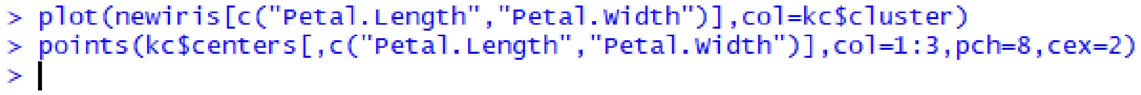


```
> plot(newiris[c("Petal.Length","Petal.Width")],col=kc$cluster)
> 
```

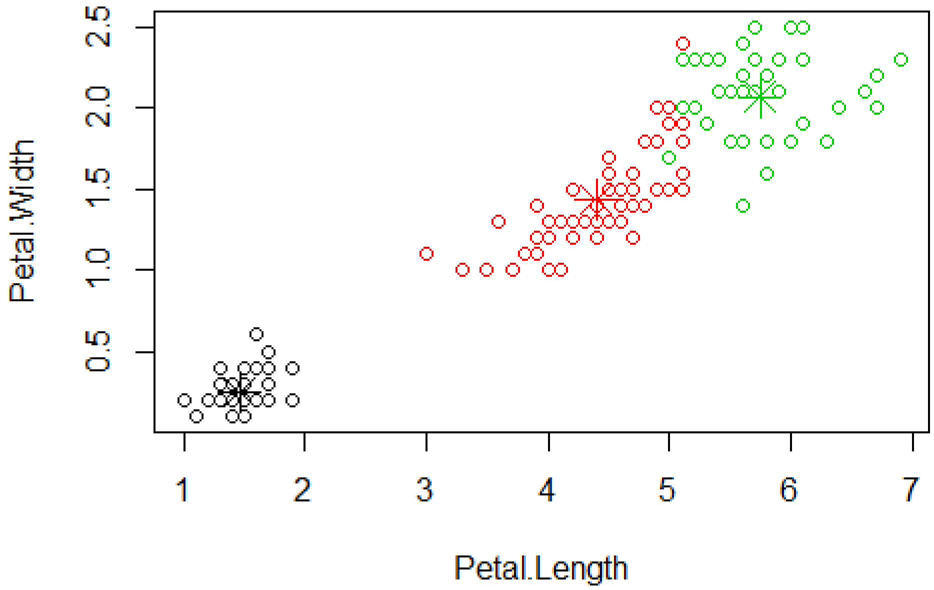


**INPUT:**

> plot(newiris[c("Petal.Length","Petal.Width")],col=kc$cluster)
> points(kc$centers[,c("Petal.Length","Petal.Width")],col=1:3,pch=8,cex=2)



```
> plot(newiris[c("Petal.Length","Petal.Width")],col=kc$cluster)
> points(kc$centers[,c("Petal.Length","Petal.Width")],col=1:3,pch=8,cex=2)
> |
```



**Conclusion :**

**We can segregate data clusters using k means clustering and find centers of cluster**