lab 6

Lab 6: Ensemble Methods

The topics that are covered in this lab worksheet are:

1. K-Means Clustering
2. Hierarchical Clustering
3. Exercises

1. K-Means Clustering

The `kmeans()` function performs K-Means clustering in R.

We begin with a simple simulated dataset, in which there truly are two clusters.

- First, we use `rnorm()` function to generate 20 observations in R^2, which follow the 2-D Gaussian/normal distribution.
- Then we shift the first 10 observations by a vector of $\begin{bmatrix}4,\\2\end{bmatrix}$, and we shift the last 10 observations by a vector of $\begin{bmatrix}-1\\-3\end{bmatrix}$.

```
set.seed(12)
x <- matrix(rnorm(20*2) , ncol=2)
x
```

```
##              [,1]       [,2]
##  [1,] -1.48056759  0.2236414
##  [2,]  1.57716947  2.0072015
##  [3,] -0.95674448  1.0119791
##  [4,] -0.92000525 -0.3024592
##  [5,] -1.99764210 -1.0252448
##  [6,] -0.27229604 -0.2673848
##  [7,] -0.31534871 -0.1991057
##  [8,] -0.62825524  0.1311226
##  [9,] -0.10646388  0.1457999
## [10,]  0.42801480  0.3620647
## [11,] -0.77771958  0.6739812
## [12,] -1.29388230  2.0720358
## [13,] -0.77956651 -0.5410286
## [14,]  0.01195176 -1.0704922
## [15,] -0.15241624 -0.3724567
## [16,] -0.70346425 -0.4851414
## [17,]  1.18887916  0.2747842
## [18,]  0.34051227 -0.4795126
## [19,]  0.50696817  0.7981053
## [20,] -0.29330515 -1.0044512
```

```
x[1:10,1] <- x[1:10,1] + 4
x[1:10,2] <- x[1:10,2] + 2
x[11:20,1] <- x[11:20,1] - 1
x[11:20,2] <- x[11:20,2] - 3
x
```

```
##             [,1]       [,2]
##  [1,]  2.5194324  2.2236414
##  [2,]  5.5771695  4.0072015
##  [3,]  3.0432555  3.0119791
##  [4,]  3.0799948  1.6975408
##  [5,]  2.0023579  0.9747552
##  [6,]  3.7277040  1.7326152
##  [7,]  3.6846513  1.8008943
##  [8,]  3.3717448  2.1311226
##  [9,]  3.8935361  2.1457999
## [10,]  4.4280148  2.3620647
## [11,] -1.7777196 -2.3260188
## [12,] -2.2938823 -0.9279642
## [13,] -1.7795665 -3.5410286
## [14,] -0.9880482 -4.0704922
## [15,] -1.1524162 -3.3724567
## [16,] -1.7034643 -3.4851414
## [17,]  0.1888792 -2.7252158
## [18,] -0.6594877 -3.4795126
## [19,] -0.4930318 -2.2018947
## [20,] -1.2933051 -4.0044512
```

For observations in R^2, we can plot them for visualization.

We use the following code snippet to output the plot to a pdf file.

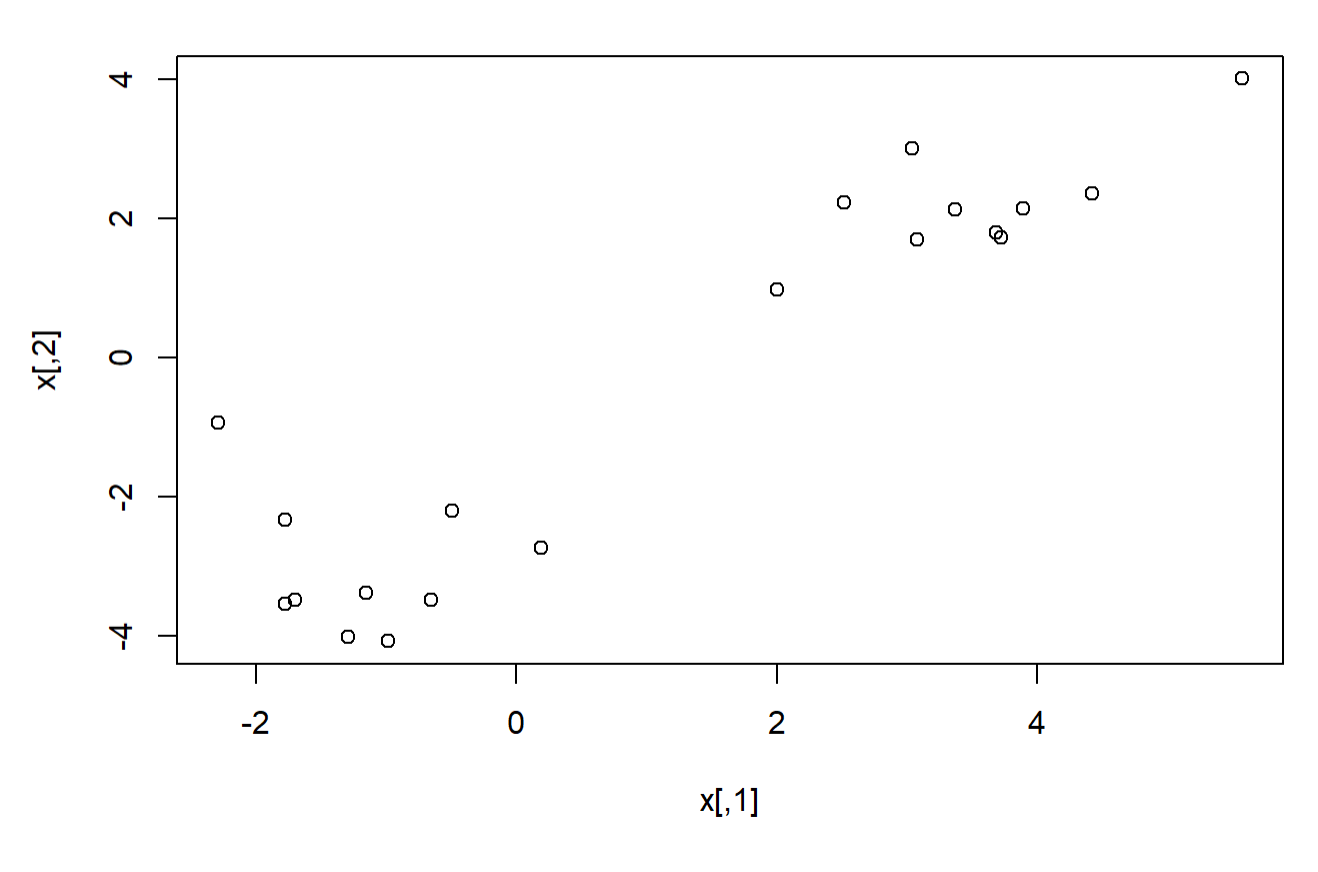
If you are using Windows OS, the filename might need to be specified in full path.

`dev.off()` instructs the computer to complete the generation of the pdf file.

```
pdf(file="clusters.pdf",width=10,height=10)
plot(x,pch=19)
dev.off()
```

```
## png
##   2
```

```
plot(x)
```



We now perform K-Means clustering with K = 2.

Recall from the lecture that the output of K-Means clustering depends on the random initial assignments. Thus, in practice we want to run it multiple times and choose the best output.

`nstart` specifies the number of times K-Means is to be run.

`km.output$cluster` is the output, which, for each observation, indicates which cluster it belongs to.

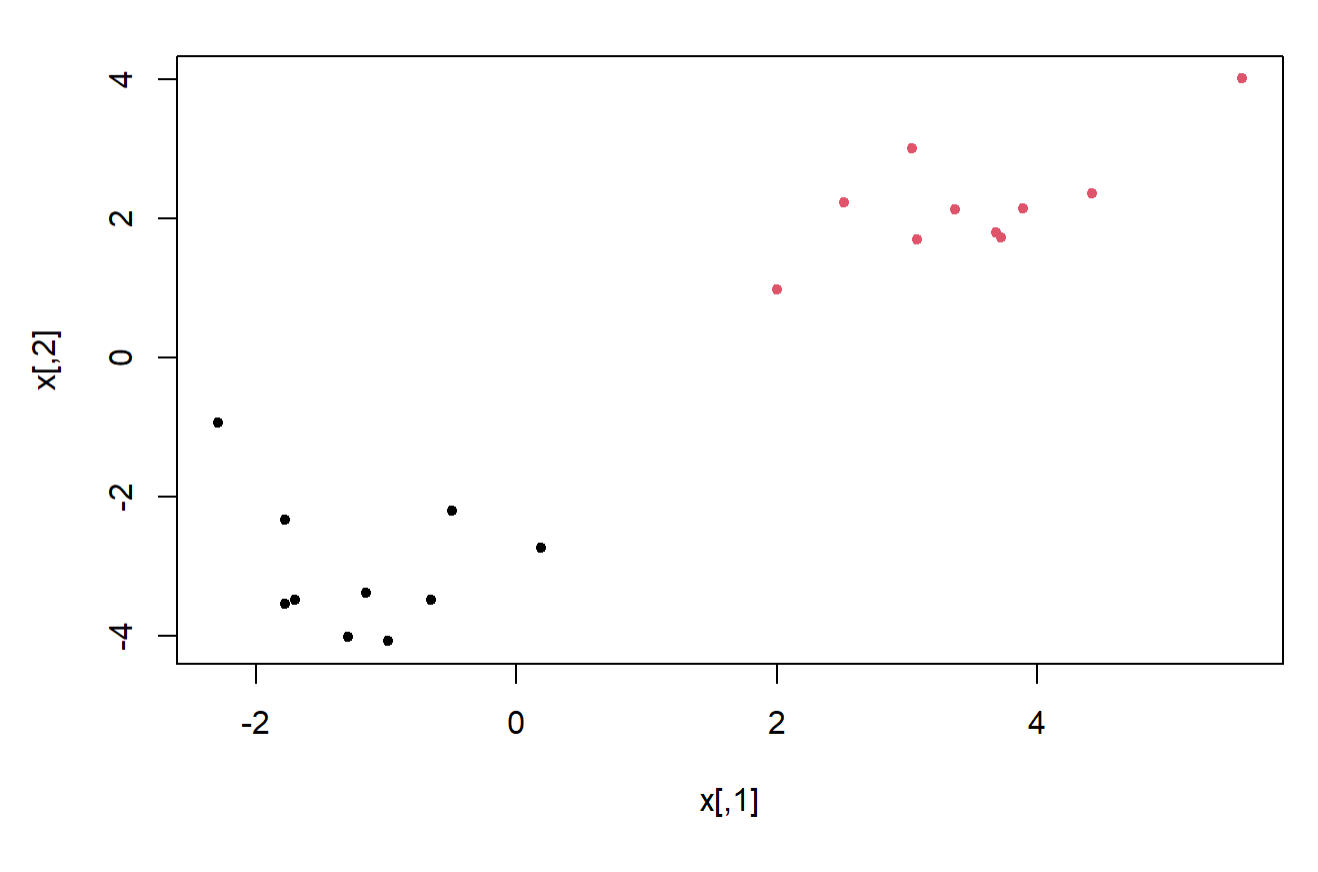
By "best", it means the cluster output C with the smallest value of Φ(C), where Φ is the function we discussed in the lecture, which computes the total distance between each observation and its cluster's centroid.

```
km.output <- kmeans(x,2,nstart=10)
km.output$cluster
```

```
##  [1] 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1
```

Now, we plot the two clusters with different colors.

```
plot(x,col=km.output$cluster,pch=20)
```

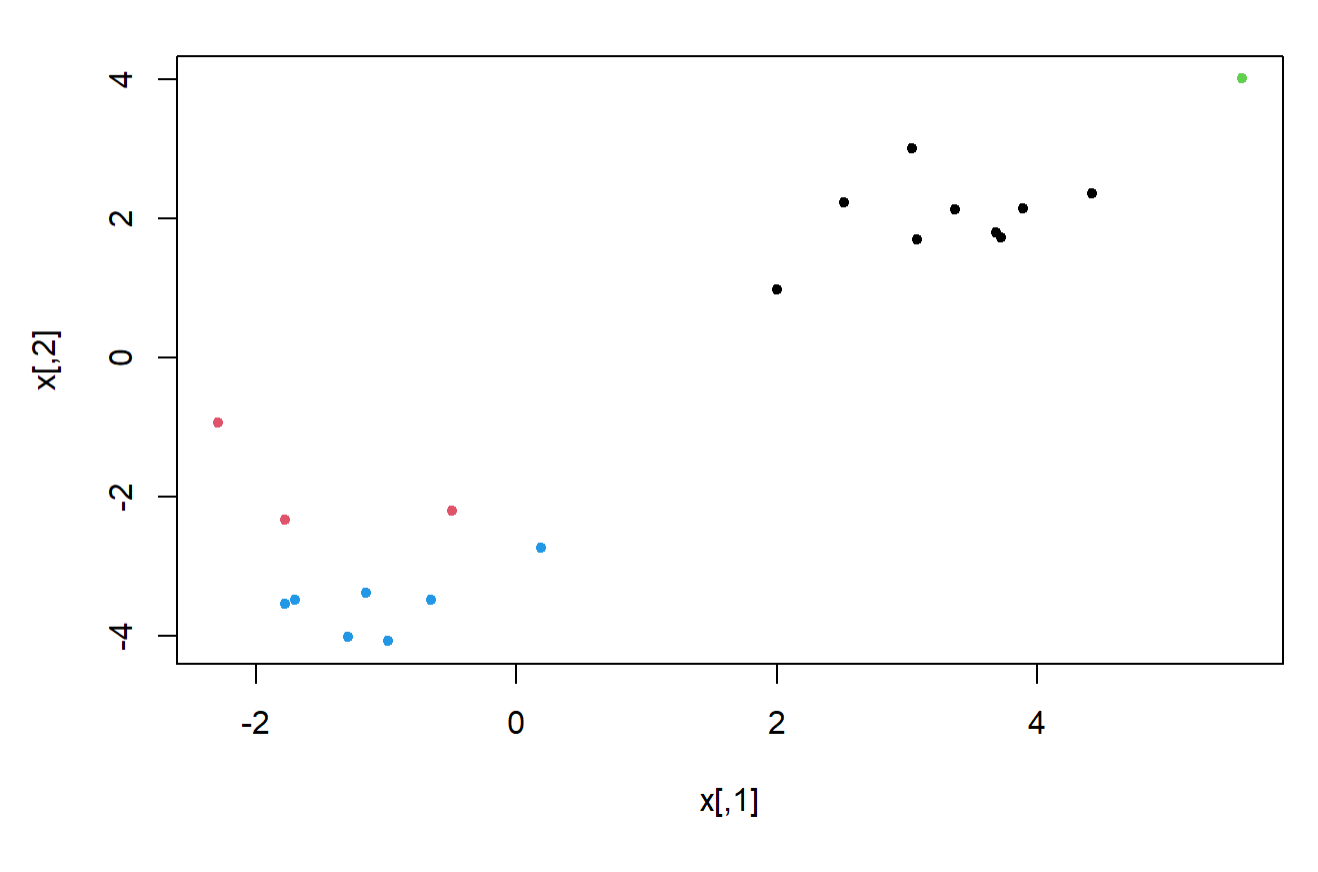


Now, try to do the same but with K = 4.

```
km.output4 <- kmeans(x,4,nstart=10)
km.output4$cluster
```

```
##  [1] 1 3 1 1 1 1 1 1 1 1 2 2 4 4 4 4 4 4 2 4
```

```
plot(x,col=km.output4$cluster,pch=20)
```



Under km.output, there are some other information stored in different variables. You may want to explore their meanings yourselves.

```
names(km.output)
```

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

`km.output$tot.withinss` is the value of Φ(C) of the cluster output.

```
km.output$tot.withinss
```

```
## [1] 28.50846
```

You should always run K-Means clustering with a large value of nstart, such as 20 or 50, since otherwise an undesirable local optimum may be obtained.

2. Hierarchical Clustering

The hclust() function performs hierarchical clustering (HC) in R.

- First, we generate a dataset of 50 observations in R^2.
- Then we apply HC with different linkage functions.

```
set.seed(12)
x <- matrix(rnorm(50*2) , ncol=2)
head(x, 10)
```

```
##             [,1]        [,2]
##  [1,] -1.4805676 -0.04268491
##  [2,]  1.5771695 -0.11267058
##  [3,] -0.9567445  0.45682725
##  [4,] -0.9200052  2.02033484
##  [5,] -1.9976421 -1.05089006
##  [6,] -0.2722960  0.73465211
##  [7,] -0.3153487  0.53924974
##  [8,] -0.6282552 -1.31427280
##  [9,] -0.1064639 -0.25003872
## [10,]  0.4280148  0.31420460
```

```
hc.complete <- hclust( dist(x) , method="complete" )
hc.complete
```

```
##
## Call:
## hclust(d = dist(x), method = "complete")
##
## Cluster method   : complete
## Distance         : euclidean
## Number of objects: 50
```

```
hc.average <- hclust( dist(x) , method="average" )
hc.average
```

```
##
## Call:
## hclust(d = dist(x), method = "average")
##
## Cluster method   : average
## Distance         : euclidean
## Number of objects: 50
```

```
hc.single <- hclust( dist(x) , method="single" )
hc.single
```

```
##
## Call:
## hclust(d = dist(x), method = "single")
##
## Cluster method   : single
## Distance         : euclidean
## Number of objects: 50
```

```
hc.centroid <- hclust( dist(x) , method="centroid" )
hc.centroid
```

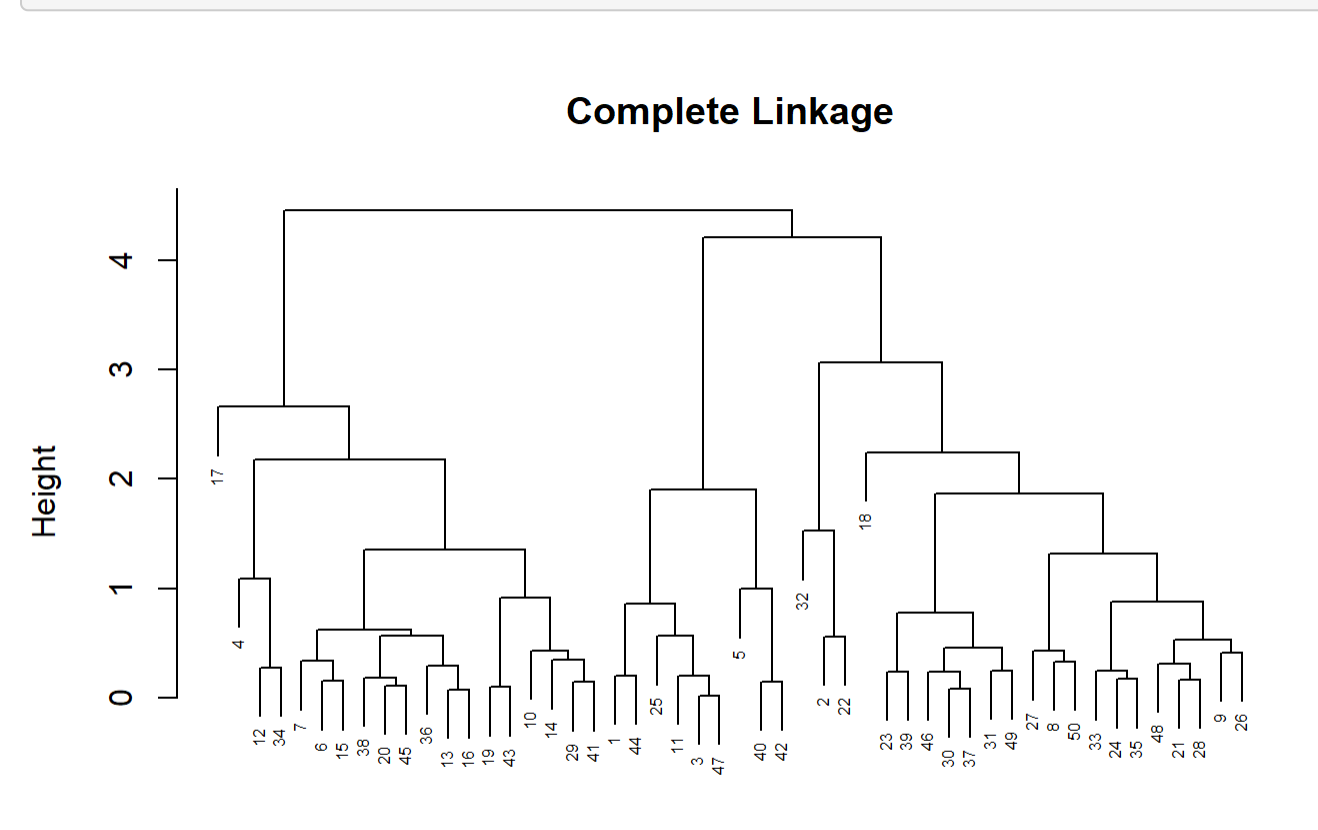
```
##
## Call:
## hclust(d = dist(x), method = "centroid")
##
## Cluster method   : centroid
## Distance         : euclidean
## Number of objects: 50
```

We can now plot the dendrograms obtained using the usual plot() function. T he numbers at the bottom of the plot are labels which identify each observation. `cex` controls the size of the labels.

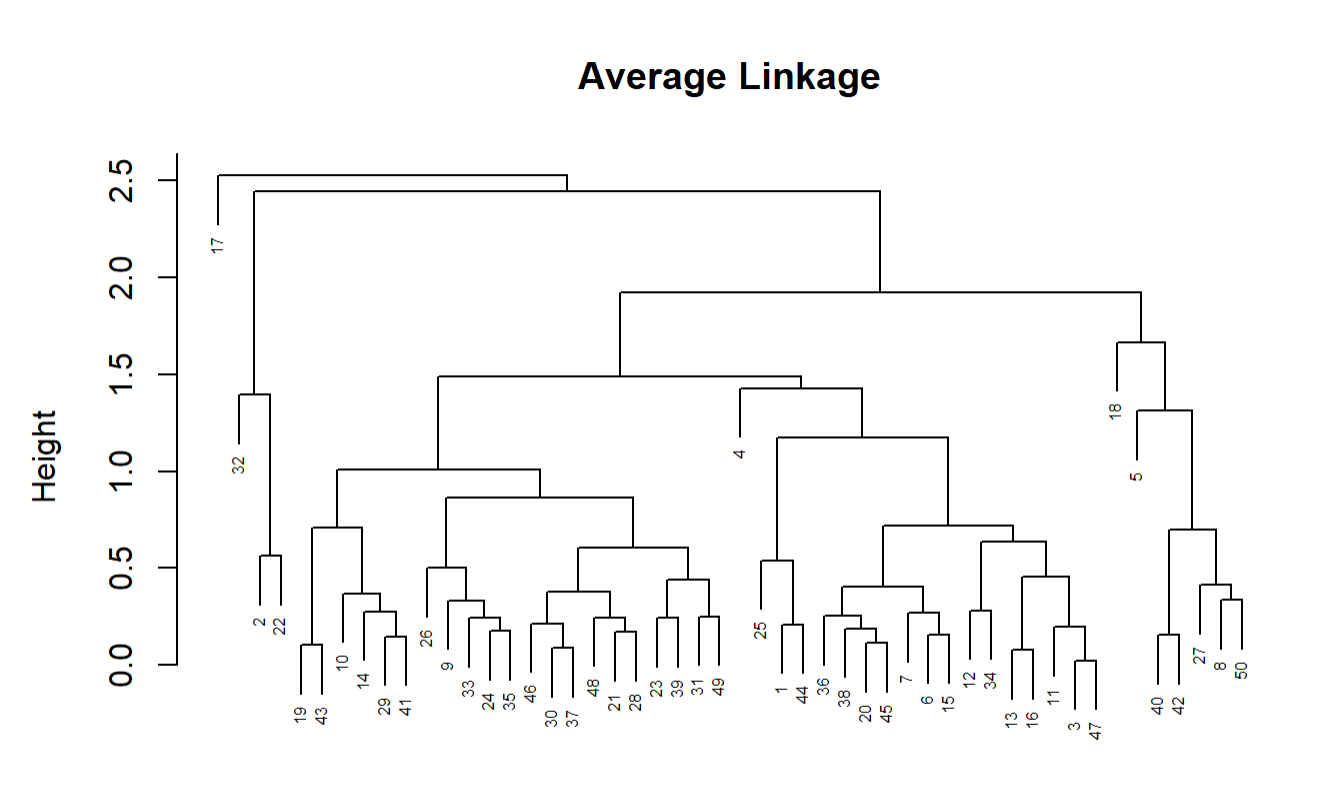
The code below plots the dendrogram for complete linkage.

Plot the other dendrograms with different linkages, and make observations yourselves.

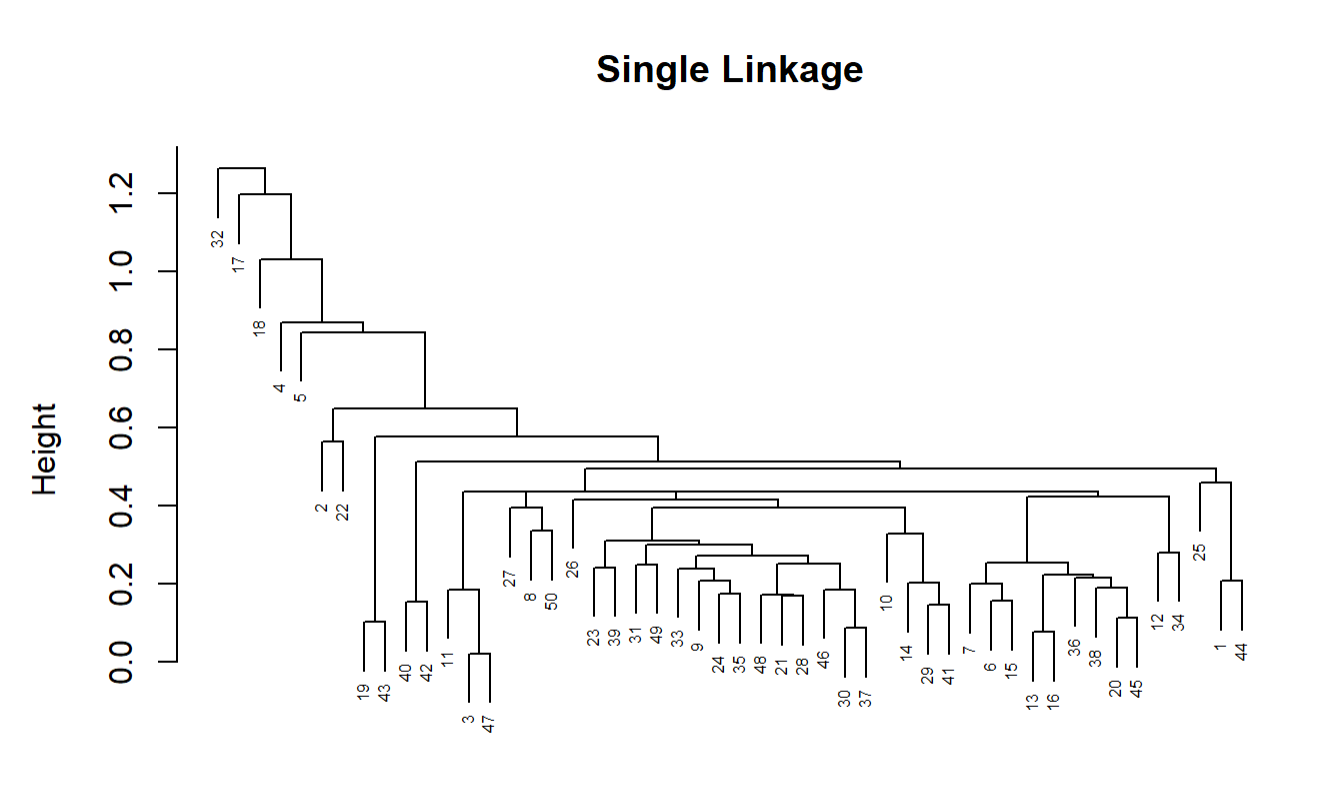
```
plot(hc.complete,main="Complete Linkage",xlab="",sub="",cex=.5)
```



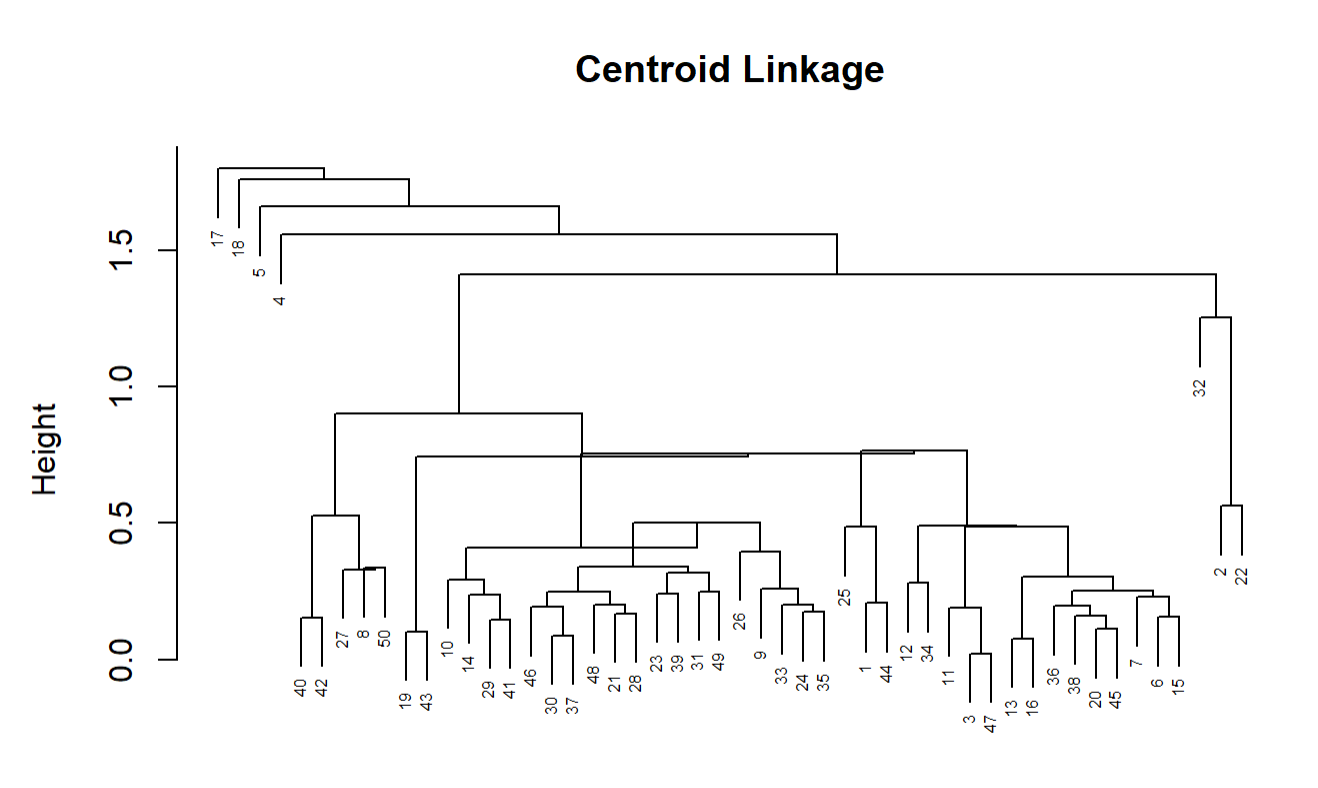

```
plot(hc.average,main="Average Linkage",xlab="",sub="",cex=.5)
```




```
plot(hc.single,main="Single Linkage",xlab="",sub="",cex=.5)
```




```
plot(hc.centroid,main="Centroid Linkage",xlab="",sub="",cex=.5)
```



Recall that for a given integer K, we cut a dendrogram at an appropriate height to generate K clusters.

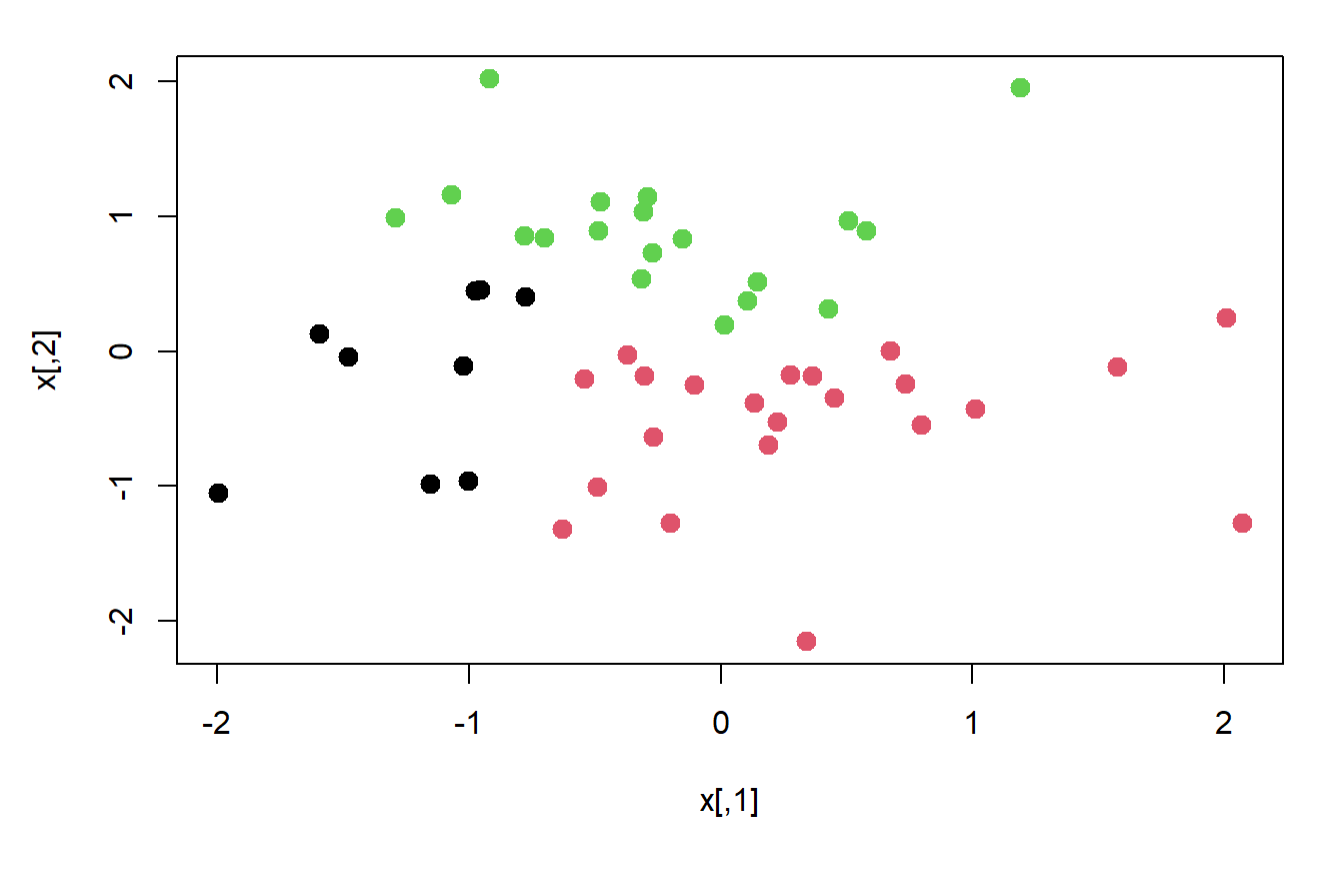
In R, this is done via the cutree() function.

Then as you did in K-Means, you can plot the clusters.

```
hc.complete.cluster <- cutree(hc.complete, 3)
hc.complete.cluster
```

```
##  [1] 1 2 1 3 1 3 3 2 2 3 1 3 3 3 3 3 3 2 3 3 2 2 2 2 1 2 2 2 3 2 2 2 2 3 2 3 2 3
## [39] 2 1 3 1 3 1 3 2 1 2 2 2
```

```
plot(x,col=hc.complete.cluster,pch=20,cex=2)
```



3. Exercises

1. Consider `the USArrests dataset`, which is part of the base R package.

(a). Perform K-Means clustering on the dataset, with K = 5. Plot the clusters.

(b). Perform hierarchical clustering on the dataset with different linkage functions, and plot the clusters.

(c). Compare the outputs. What do you observe?

1.

```
data("USArrests")
names(USArrests)
```

```
## [1] "Murder"   "Assault"  "UrbanPop" "Rape"
```

```
dim(USArrests)
```

```
## [1] 50  4
```

```
head(USArrests, 10)
```

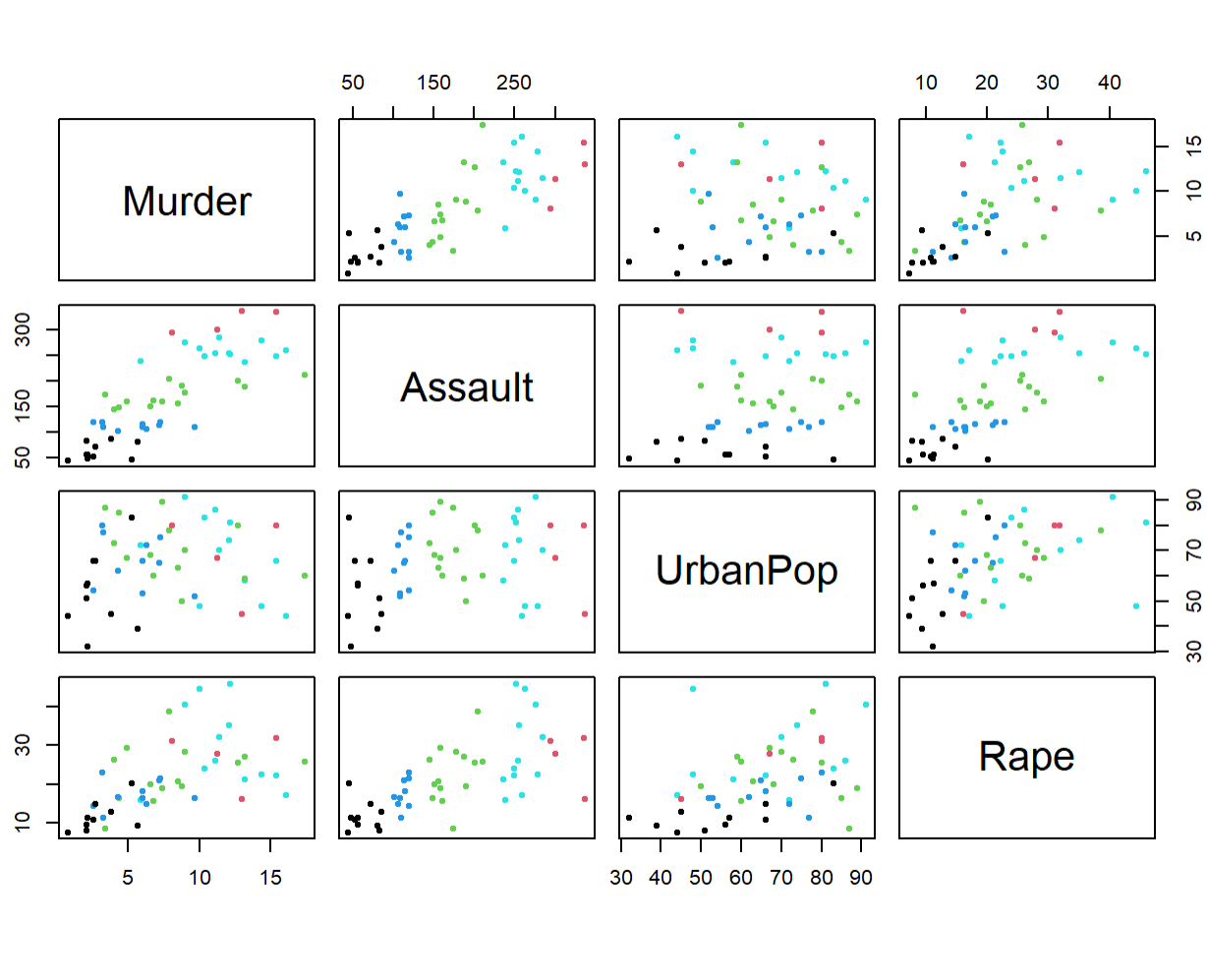
```
##             Murder Assault UrbanPop Rape
## Alabama       13.2     236       58 21.2
## Alaska        10.0     263       48 44.5
## Arizona        8.1     294       80 31.0
## Arkansas       8.8     190       50 19.5
## California     9.0     276       91 40.6
## Colorado       7.9     204       78 38.7
## Connecticut    3.3     110       77 11.1
## Delaware       5.9     238       72 15.8
## Florida       15.4     335       80 31.9
## Georgia       17.4     211       60 25.8
```

(a). Perform K-Means clustering on the dataset, with K = 5. Plot the clusters.

```
km.clu <- kmeans(USArrests,5,nstart=10)
km.clu
```

```
## K-means clustering with 5 clusters of sizes 10, 4, 14, 10, 12
##
## Cluster means:
##      Murder  Assault UrbanPop     Rape
## 1  2.950000  62.7000 53.90000 11.51000
## 2 11.950000 316.5000 68.00000 26.70000
## 3  8.214286 173.2857 70.64286 22.84286
## 4  5.590000 112.4000 65.60000 17.27000
## 5 11.766667 257.9167 68.41667 28.93333
##
## Clustering vector:
##        Alabama         Alaska        Arizona       Arkansas     California
##              5              5              2              3              5
##       Colorado    Connecticut       Delaware        Florida        Georgia
##              3              4              5              2              3
##         Hawaii          Idaho       Illinois        Indiana           Iowa
##              1              4              5              4              1
##         Kansas       Kentucky      Louisiana          Maine       Maryland
##              4              4              5              1              2
##  Massachusetts       Michigan      Minnesota    Mississippi       Missouri
##              3              5              1              5              3
##        Montana       Nebraska         Nevada  New Hampshire     New Jersey
##              4              4              5              1              3
##     New Mexico       New York North Carolina   North Dakota           Ohio
##              5              5              2              1              4
##       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
##              3              3              4              3              5
##   South Dakota      Tennessee          Texas           Utah        Vermont
##              1              3              3              4              1
##       Virginia     Washington  West Virginia      Wisconsin        Wyoming
##              3              3              1              1              3
##
## Within cluster sum of squares by cluster:
## [1] 4547.914 2546.350 9136.643 1480.210 6705.907
##  (between_SS / total_SS =  93.1 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

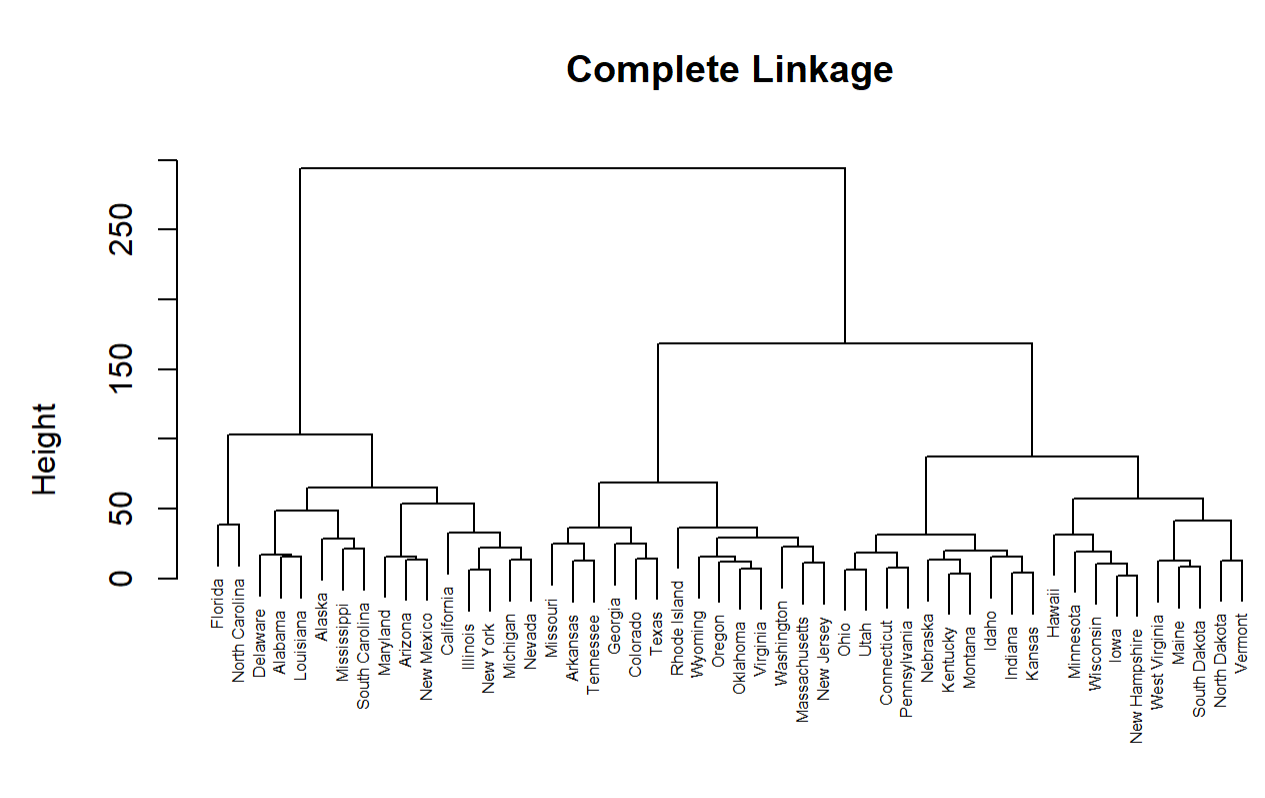
```
plot(USArrests,col=km.clu $cluster,pch=20)
```



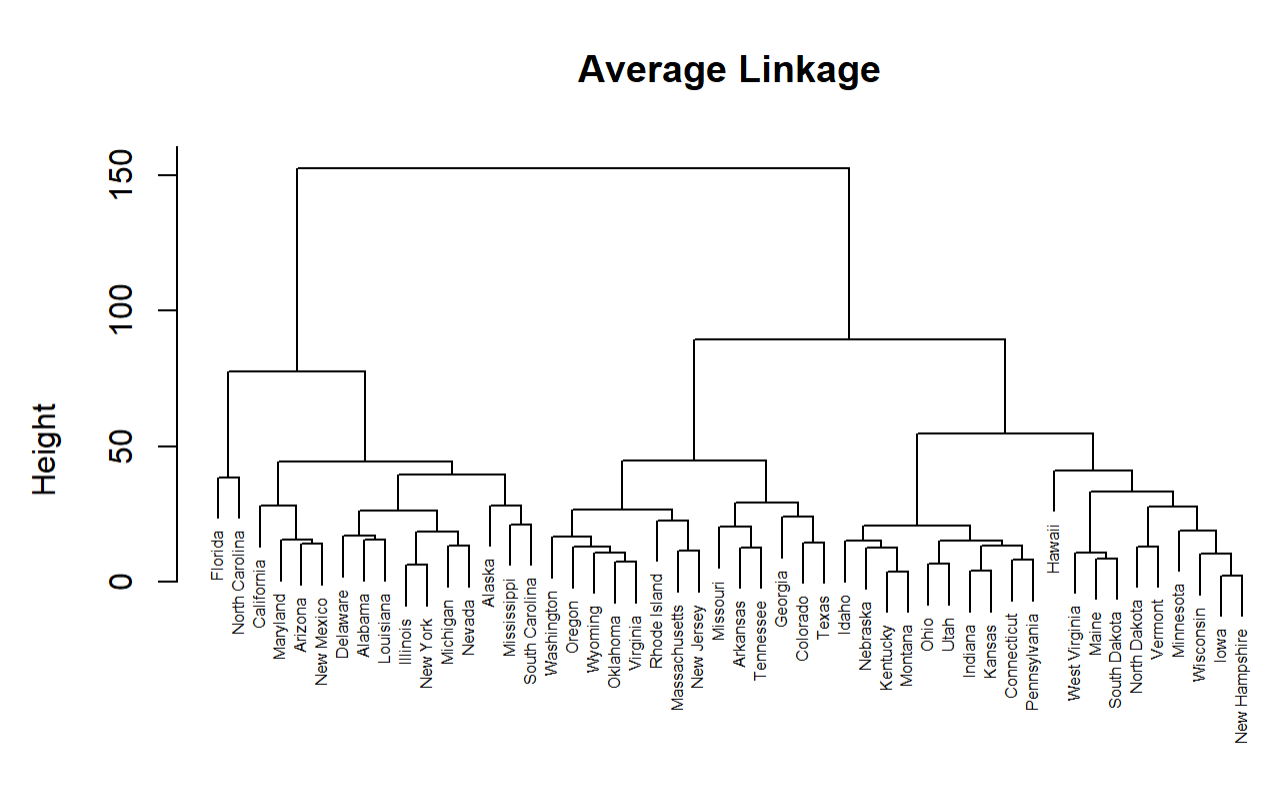
b. Perform hierarchical clustering on the dataset with different linkage functions, and plot the clusters.

```
hc.complete.b <- hclust( dist(USArrests) , method="complete" )
hc.average.b <- hclust( dist(USArrests) , method="average" )
hc.single.b <- hclust( dist(USArrests) , method="single" )
hc.centroid.b <- hclust( dist(USArrests) , method="centroid" )
```

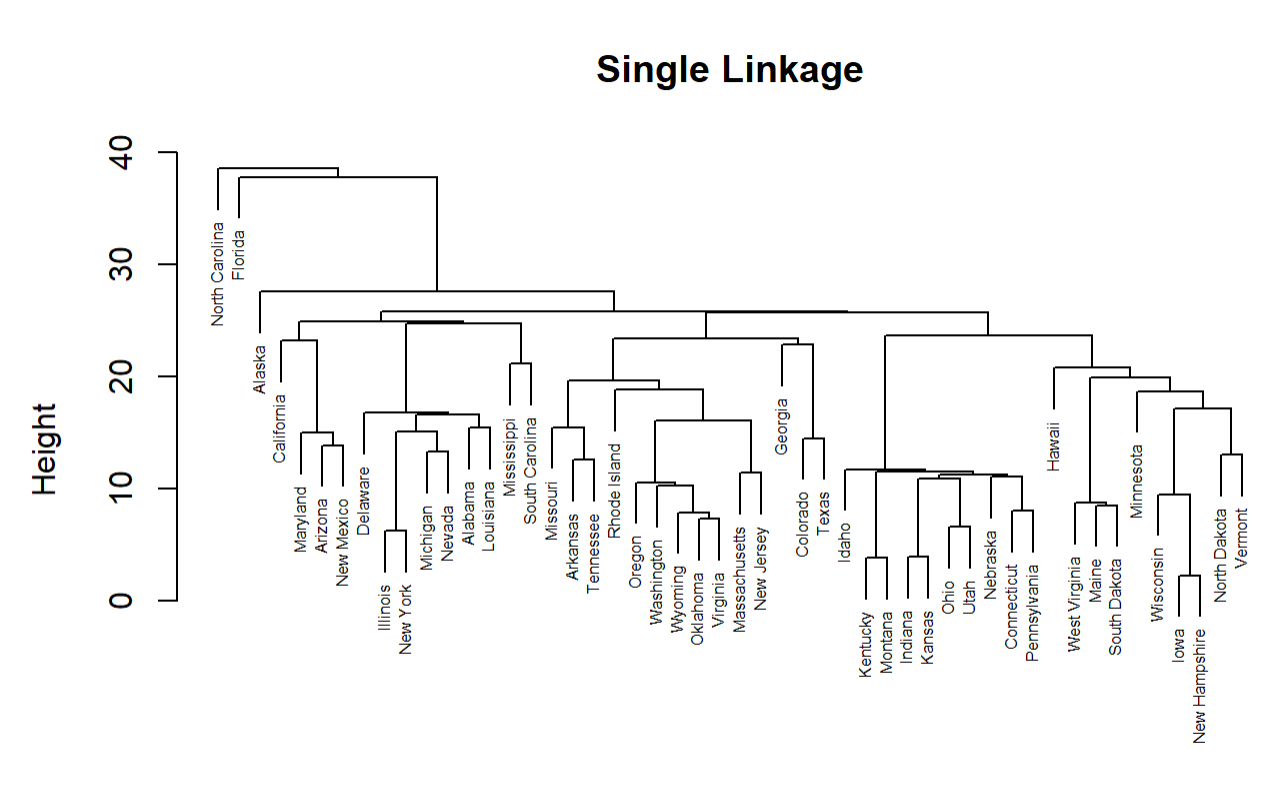
```
plot(hc.complete.b,main="Complete Linkage",xlab="",sub="",cex=.5)
```




```
plot(hc.average.b,main="Average Linkage",xlab="",sub="",cex=.5)
```




```
plot(hc.single.b,main="Single Linkage",xlab="",sub="",cex=.5)
```




```
plot(hc.centroid.b,main="Centroid Linkage",xlab="",sub="",cex=.5)
```