Meher Shrishti Nigam
20BRS1193

EDA LAB – 9
10 / 3 / 23

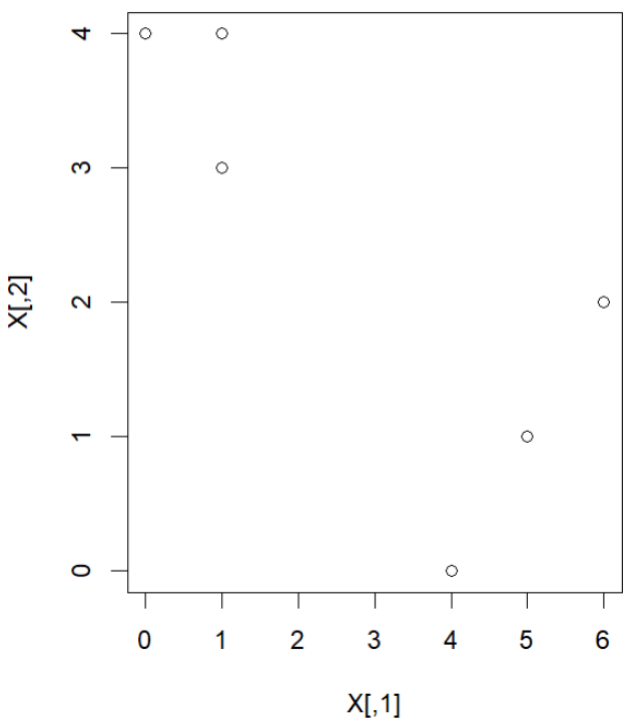
```
# Meher Shrishti Nigam
# 20BRS1193
# EDA Lab 9
options(prompt="MEHERSHRISHTI>", continue =" ")
# options(prompt=">", continue =" ")
# EDA-LAB-EXPERIMENT-9 (Date-10/3/2023)
library(dplyr)
library(ggplot2)

# Q1)
# In this problem, you will perform K-means clustering, with K = 2, on a small
# example with n = 6 observations and p = 2 features. The observations are as
# follows.

# (a) Plot the observations.

n = 6
X = matrix(c(1, 4, 1, 3, 0, 4, 5, 1, 6, 2, 4, 0),
        nrow = n, byrow = T)
plot(X)
```

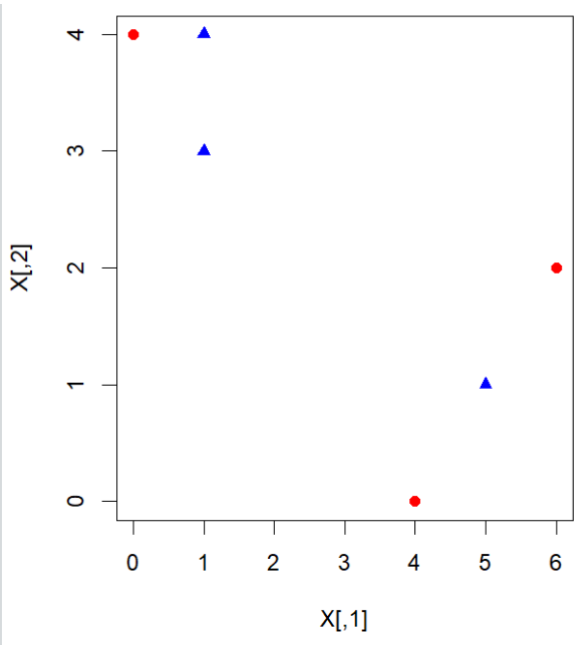
```
MEHERSHRISHTI># EDA-LAB-EXPERIMENT-9 (Date-10/3/2023)
MEHERSHRISHTI>library(dplyr)
MEHERSHRISHTI>library(ggplot2)
MEHERSHRISHTI>n = 6
MEHERSHRISHTI>X = matrix(c(1, 4, 1, 3, 0, 4, 5, 1, 6, 2, 4, 0),
            nrow = n, byrow = T)
MEHERSHRISHTI>plot(X)
MEHERSHRISHTI>
```



(b) Randomly assign a cluster label to each observation. You can use the
sample() command in R to do this. Report the cluster labels for each observation.

```
set.seed(2^17 - 1)
clusters = sample(1:2, n, replace = T)
clusters
col = rep("red", n)
col[clusters == 2] = "blue"
pch = rep(16, n)
pch[clusters == 2] = 17
plot(X, col = col, pch = pch)
```

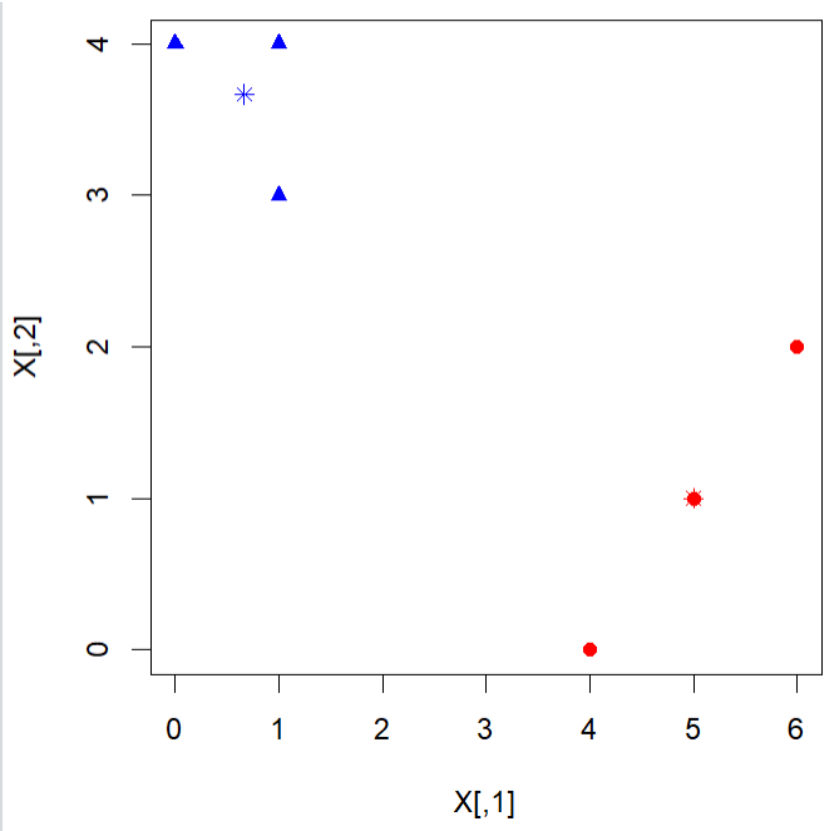
```
MEHERSHRISHTI>set.seed(2^17 - 1)
MEHERSHRISHTI>clusters = sample(1:2, n, replace = T)
MEHERSHRISHTI>clusters
[1] 2 2 1 2 1 1
MEHERSHRISHTI>col = rep("red", n)
MEHERSHRISHTI>col[clusters == 2] = "blue"
MEHERSHRISHTI>pch = rep(16, n)
MEHERSHRISHTI>pch[clusters == 2] = 17
MEHERSHRISHTI>plot(X, col = col, pch = pch)
MEHERSHRISHTI>
```



(c) Compute the centroid for each cluster.

```
centroids = aggregate(X, list(Cluster = clusters), mean)
centroids
plot(X, col = col, pch = pch)
points(centroids[1,2:3], col = "red", pch = 8)
points(centroids[2,2:3], col = "blue", pch = 8)
```

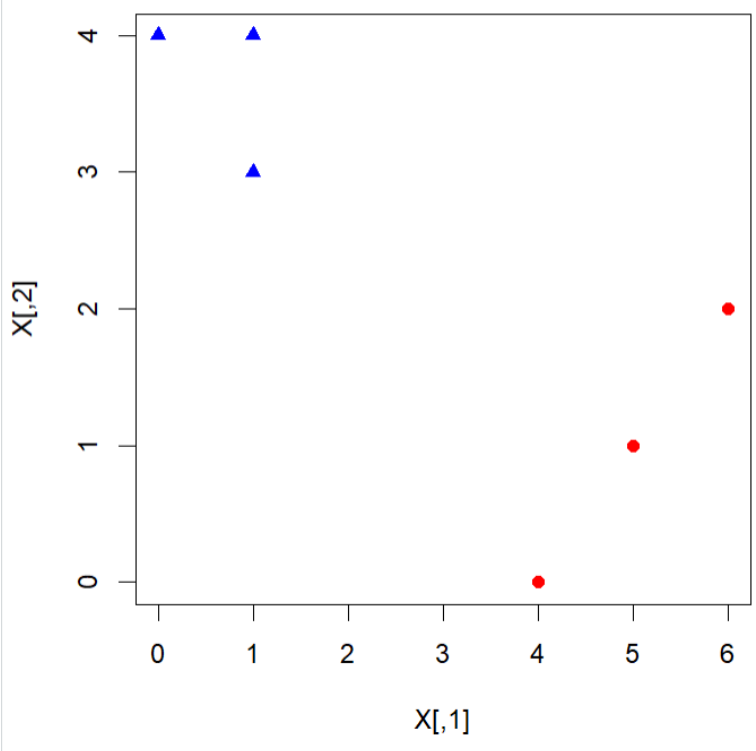
```
MEHERSHRISHTI># (c) Compute the centroid for each cluster.
MEHERSHRISHTI>centroids = aggregate(X, list(Cluster = clusters), mean)
MEHERSHRISHTI>centroids
  Cluster       V1       V2
1       1 3.333333 2.000000
2       2 2.333333 2.666667
MEHERSHRISHTI>plot(X, col = col, pch = pch)
MEHERSHRISHTI>points(centroids[1,2:3], col = "red", pch = 8)
MEHERSHRISHTI>points(centroids[2,2:3], col = "blue", pch = 8)
MEHERSHRISHTI>
```



(d) Assign each observation to the centroid to which it is closest, in terms of
Euclidean distance. Report the cluster labels for each observation.
Create a function to get the closest centroid
```
library(class)
clusters = knn(centroids[,2:3], X, factor(centroids[,1]))
clusters

col = rep("red", n)
col[clusters == 2] = "blue"
pch = rep(16, n)
pch[clusters == 2] = 17
plot(X, col = col, pch = pch)
```

```
MEHERSHRISHTI># (d) Assign each observation to the centroid to which it is
 closest, in terms of
MEHERSHRISHTI># Euclidean distance. Report the cluster labels for each obs
ervation.
MEHERSHRISHTI># Create a function to get the closest centroid
MEHERSHRISHTI>library(class)
MEHERSHRISHTI>clusters = knn(centroids[,2:3], X, factor(centroids[,1]))
MEHERSHRISHTI>clusters
[1] 2 2 2 1 1 1
Levels: 1 2
MEHERSHRISHTI>col = rep("red", n)
MEHERSHRISHTI>col[clusters == 2] = "blue"
MEHERSHRISHTI>pch = rep(16, n)
MEHERSHRISHTI>pch[clusters == 2] = 17
MEHERSHRISHTI>plot(X, col = col, pch = pch)
MEHERSHRISHTI>
```



(e) Repeat (c) and (d) until the answers obtained stop changing.
```
centroids = aggregate(X, list(Cluster = clusters), mean)
centroids
plot(X, col = col, pch = pch)
points(centroids[1,2:3], col = "red", pch = 8)
points(centroids[2,2:3], col = "blue", pch = 8)
clusters = knn(centroids[,2:3], X, factor(centroids[,1]))
clusters
centroids = aggregate(X, list(Cluster = clusters), mean)
```

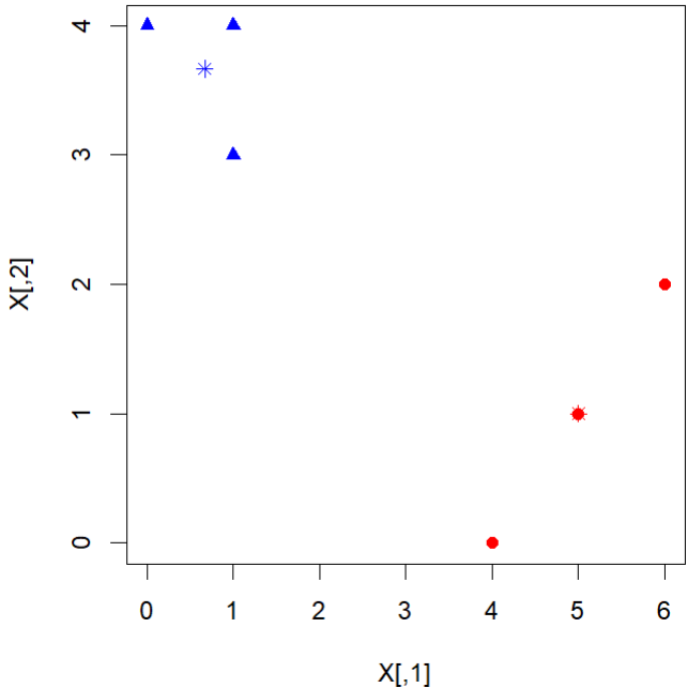
centroids

```
MEHERSHRISHTI># (e) Repeat (c) and (d) until the answers obtained stop ch
anging.
MEHERSHRISHTI>centroids = aggregate(X, list(Cluster = clusters), mean)
MEHERSHRISHTI>centroids
  Cluster        V1       V2
1       1 5.0000000 1.000000
2       2 0.6666667 3.666667
MEHERSHRISHTI>plot(X, col = col, pch = pch)
MEHERSHRISHTI>points(centroids[1,2:3], col = "red", pch = 8)
MEHERSHRISHTI>points(centroids[2,2:3], col = "blue", pch = 8)
MEHERSHRISHTI>clusters = knn(centroids[,2:3], X, factor(centroids[,1]))
MEHERSHRISHTI>clusters
[1] 2 2 2 1 1 1
Levels: 1 2
MEHERSHRISHTI>centroids = aggregate(X, list(Cluster = clusters), mean)
MEHERSHRISHTI>centroids
  Cluster        V1       V2
1       1 5.0000000 1.000000
2       2 0.6666667 3.666667
MEHERSHRISHTI>
```

(f) In your plot from (a), color the observations according to the cluster labels
obtained.

```
col = rep("red", n)
col[clusters == 2] = "blue"
pch = rep(16, n)
pch[clusters == 2] = 17
plot(X, col = col, pch = pch)
points(centroids[1,2:3], col = "red", pch = 8)
points(centroids[2,2:3], col = "blue", pch = 8)
```

```
MEHERSHRISHTI># (f) In your plot from (a), color the observations accordi
ng to the cluster labels
MEHERSHRISHTI># obtained.
MEHERSHRISHTI>col = rep("red", n)
MEHERSHRISHTI>col[clusters == 2] = "blue"
MEHERSHRISHTI>pch = rep(16, n)
MEHERSHRISHTI>pch[clusters == 2] = 17
MEHERSHRISHTI>plot(X, col = col, pch = pch)
MEHERSHRISHTI>points(centroids[1,2:3], col = "red", pch = 8)
MEHERSHRISHTI>points(centroids[2,2:3], col = "blue", pch = 8)
MEHERSHRISHTI>
```



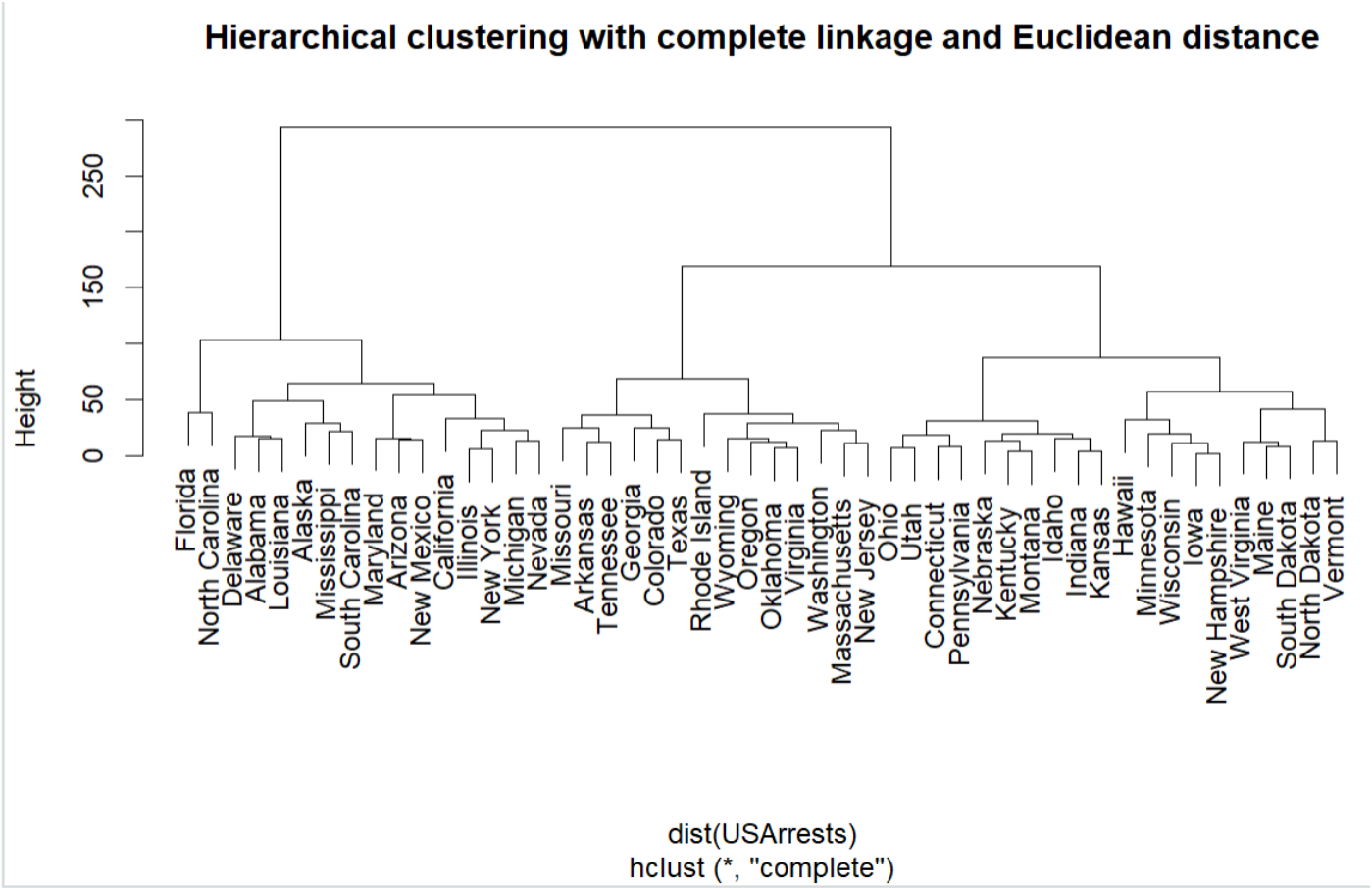
# Q2)
# Consider the USArrests data. We will now perform hierarchal clustering on the states.
library(ISLR)
data("USArrests")

# (a) Using hierarchical clustering with complete linkage and Euclidean distance,
# cluster the states.
set.seed(702)
hclust.mod <- hclust(dist(USArrests), method='complete')

plot(hclust.mod, main='Hierarchical clustering with complete linkage and Euclidean distance')

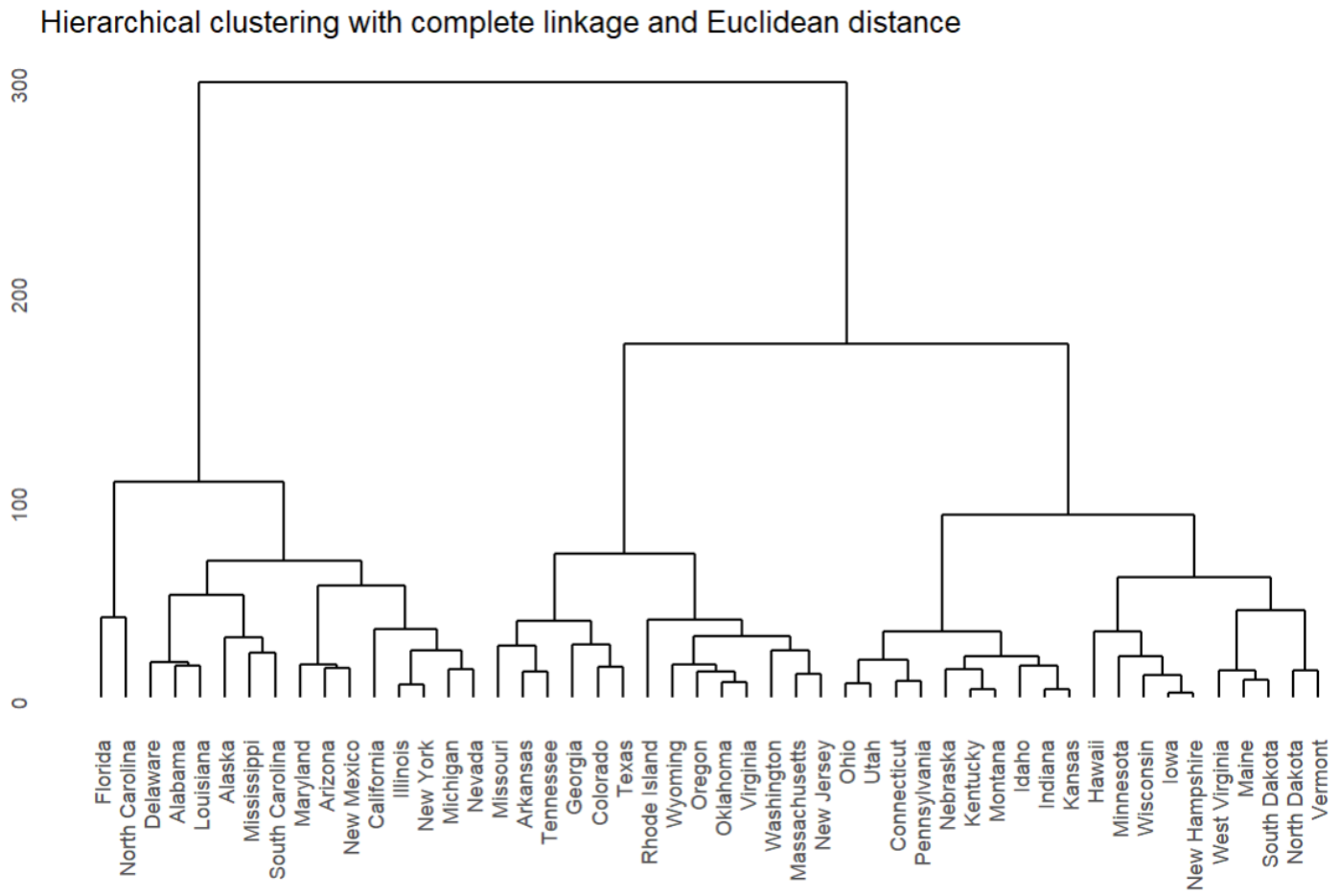
# install.packages("ggdendro")
library(ggdendro)
library(tidyverse)

```
MEHERSHRISHTI># Consider the USArrests data. We will now perform hierarchal clus
tering on the states.
MEHERSHRISHTI>library(ISLR)
MEHERSHRISHTI>data("USArrests")
MEHERSHRISHTI># (a) Using hierarchical clustering with complete linkage and Eucl
idean distance,
MEHERSHRISHTI># cluster the states.
MEHERSHRISHTI>set.seed(702)
MEHERSHRISHTI>hclust.mod <- hclust(dist(USArrests), method='complete')
MEHERSHRISHTI>plot(hclust.mod, main='Hierarchical clustering with complete linka
ge and Euclidean distance')
MEHERSHRISHTI># install.packages("ggdendro")
MEHERSHRISHTI>library(ggdendro)
MEHERSHRISHTI>library(tidyverse)
MEHERSHRISHTI>
```




```
ggdendrogram(hclust.mod, segements=TRUE, labels=TRUE, leaf_labels = TRUE, rotate=FALSE,
theme_dendro = TRUE) +
  labs(title='Hierarchical clustering with complete linkage and Euclidean distance')
```

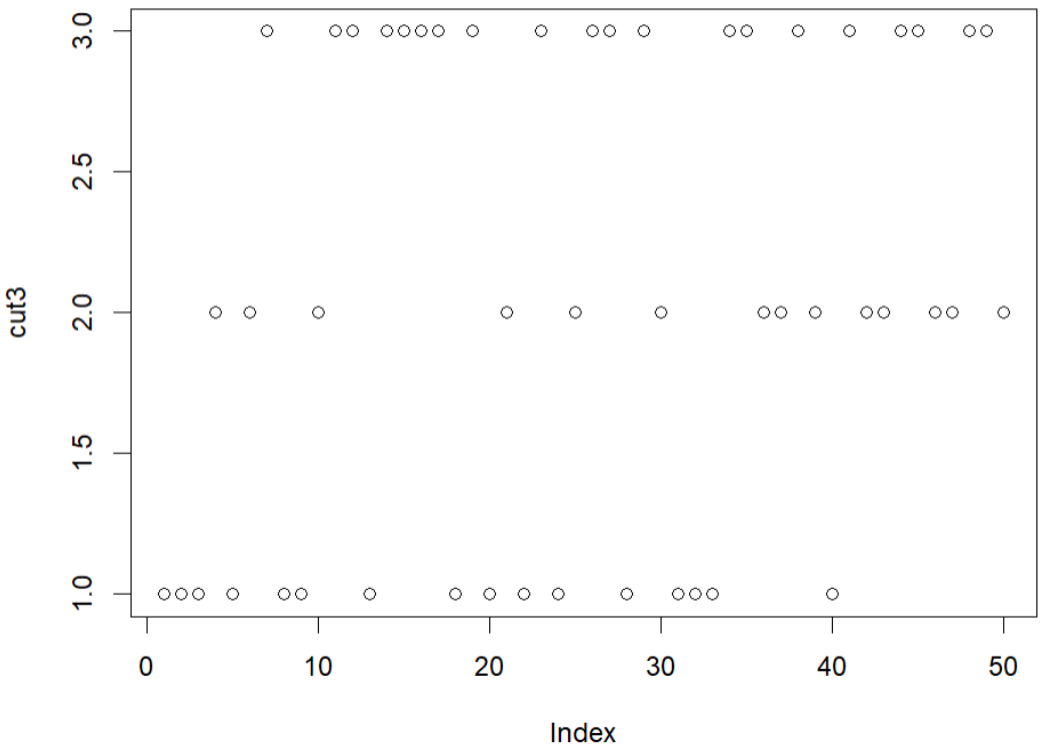
```
MEHERSHRISHTI>ggdendrogram(hclust.mod, segements=TRUE, labels=TRUE, leaf_labels = TRU
E, rotate=FALSE, theme_dendro = TRUE) +
    labs(title='Hierarchical clustering with complete linkage and Euclidean distance')
```



(b) Cut the dendrogram at a height that results in three distinct clusters. Which
states belong to which clusters?

```
set.seed(702)
cut3 <- cutree(hclust.mod, 3)
table(cut3)
cut3
plot(cut3)
```

```
MEHERSHRISHTI># (b) Cut the dendrogram at a height that results in three distinct clusters. Which
MEHERSHRISHTI># states belong to which clusters?
MEHERSHRISHTI>set.seed(702)
MEHERSHRISHTI>cut3 <- cutree(hclust.mod, 3)
MEHERSHRISHTI>table(cut3)
cut3
 1  2  3
16 14 20
MEHERSHRISHTI>cut3
       Alabama         Alaska        Arizona       Arkansas     California
             1              1              1              2              1
      Colorado    Connecticut       Delaware        Florida        Georgia
             2              3              1              1              2
        Hawaii          Idaho       Illinois        Indiana           Iowa
             3              3              1              3              3
        Kansas       Kentucky      Louisiana          Maine       Maryland
             3              3              1              3              1
 Massachusetts       Michigan      Minnesota    Mississippi       Missouri
             2              1              3              1              2
       Montana       Nebraska         Nevada  New Hampshire     New Jersey
             3              3              1              3              2
    New Mexico       New York North Carolina   North Dakota           Ohio
             1              1              1              3              3
      Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
             2              2              3              2              1
  South Dakota      Tennessee          Texas           Utah        Vermont
             3              2              2              3              3
      Virginia     Washington  West Virginia      Wisconsin        Wyoming
             2              2              3              3              2
MEHERSHRISHTI>plot(cut3)
MEHERSHRISHTI>
```

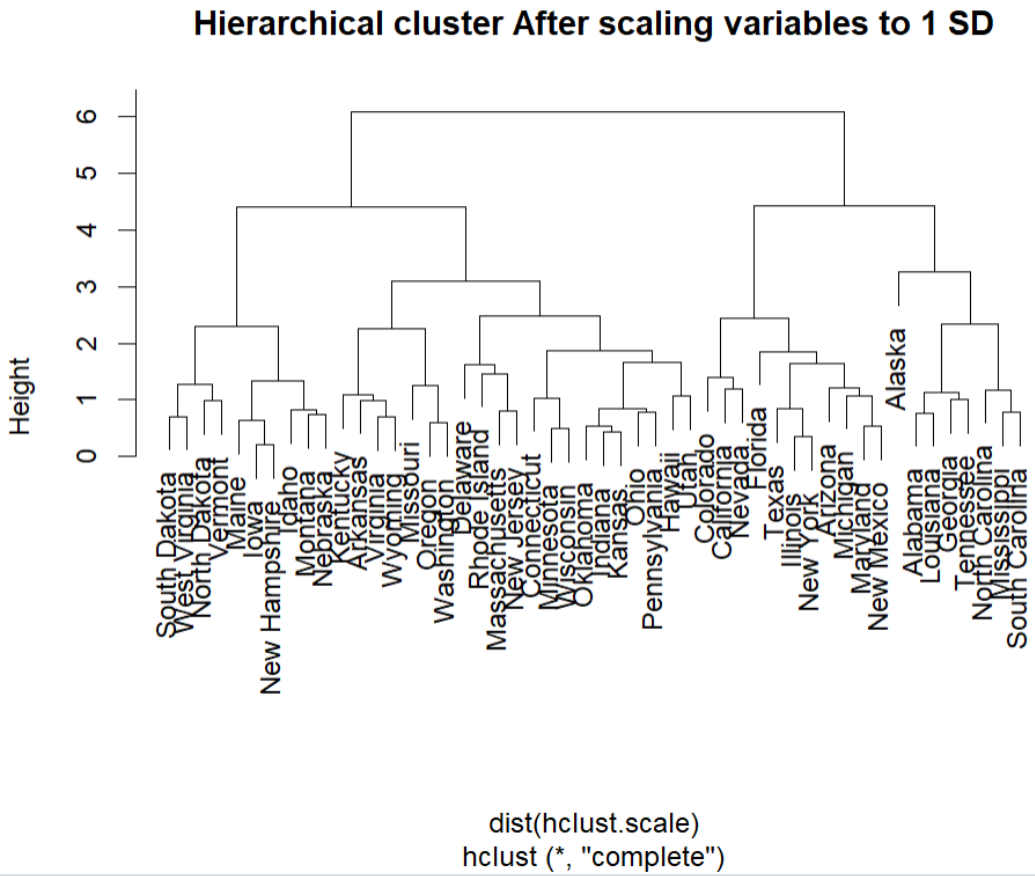


(c) Hierarchically cluster the states using complete linkage and Euclidean distance,
after scaling the variables to have standard deviation one.

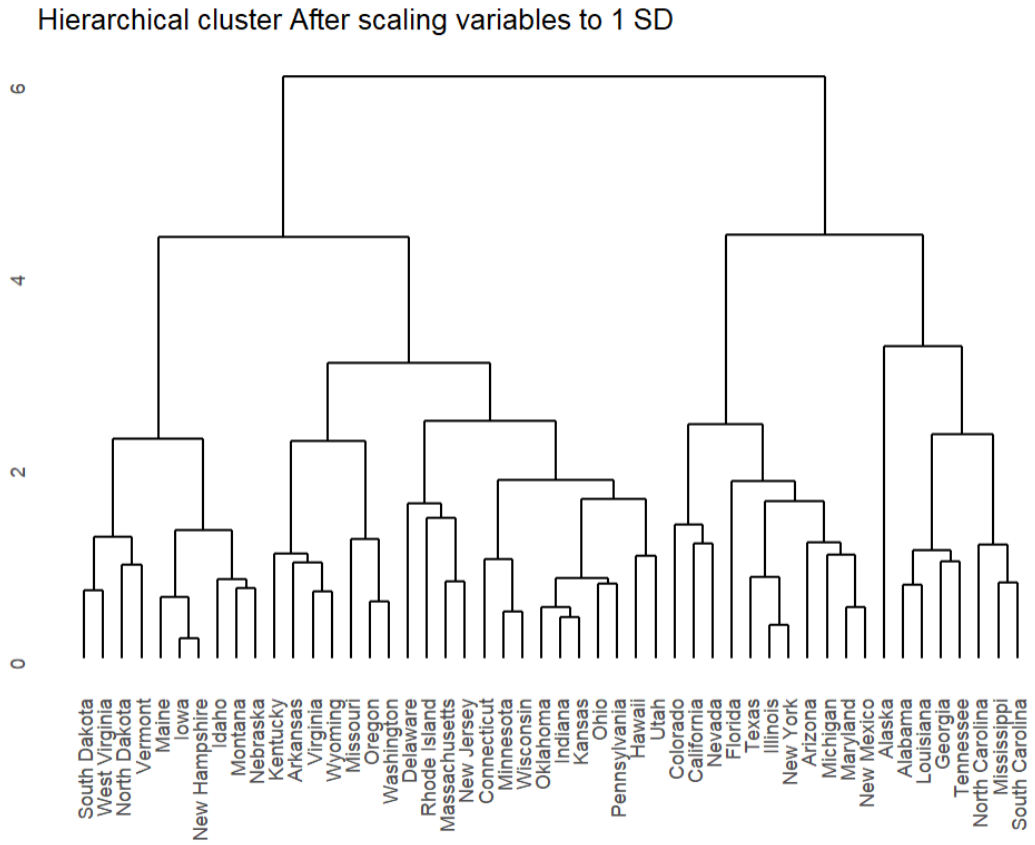
```
hclust.scale <- scale(USArrests)
set.seed(702)
hclust.scale.mod <- hclust(dist(hclust.scale),method='complete')

plot(hclust.scale.mod, main='Hierarchical cluster After scaling variables to 1 SD')
```

```
MEHERSHRISHTI># (c) Hierarchically cluster the states using complete linka
ge and Euclidean distance,
MEHERSHRISHTI># after scaling the variables to have standard deviation on
e.
MEHERSHRISHTI>hclust.scale <- scale(USArrests)
MEHERSHRISHTI>set.seed(702)
MEHERSHRISHTI>hclust.scale.mod <- hclust(dist(hclust.scale),method='comple
te')
MEHERSHRISHTI>plot(hclust.scale.mod, main='Hierarchical cluster After scal
ing variables to 1 SD')
MEHERSHRISHTI>
```




```
ggdendrogram(hclust.scale.mod, segements=TRUE, labels=TRUE, leaf_labels = TRUE,
rotate=FALSE, theme_dendro = TRUE) +
  labs(title='Hierarchical cluster After scaling variables to 1 SD')
```



(d) What effect does scaling the variables have on the hierarchical clustering
obtained? In your opinion, should the variables be scaled before the interobservation dissimilarities are computed?

```
set.seed(702)
cut.hclust.scale.mod <- cutree(hclust.scale.mod,3)

table(cut.hclust.scale.mod)
cut.hclust.scale.mod
Non.SD <- cut3
SD.1 <- cut.hclust.scale.mod

tab <- table(Non.SD, SD.1)
tab
same <- (tab[1,1] + tab[2,2] + tab[3,3]) / sum(tab)
cat('It appears that ', same*100,'% of the observations are assigned to the same clusters')
```

```
MEHERSHRISHTI># (d) What effect does scaling the variables have on the hierarchical clustering
MEHERSHRISHTI># obtained? In your opinion, should the variables be scaled before the interobservation dissimilarities are computed?
MEHERSHRISHTI>set.seed(702)
MEHERSHRISHTI>cut.hclust.scale.mod <- cutree(hclust.scale.mod,3)
MEHERSHRISHTI>table(cut.hclust.scale.mod)
cut.hclust.scale.mod
 1  2  3
 8 11 31
MEHERSHRISHTI>cut.hclust.scale.mod
       Alabama         Alaska        Arizona       Arkansas
             1              1              2              3
    California       Colorado    Connecticut       Delaware
             2              2              3              3
       Florida        Georgia         Hawaii          Idaho
             2              1              3              3
      Illinois        Indiana           Iowa         Kansas
             2              3              3              3
      Kentucky      Louisiana          Maine       Maryland
             3              1              3              2
 Massachusetts       Michigan      Minnesota    Mississippi
             3              2              3              1
      Missouri        Montana       Nebraska         Nevada
             3              3              3              2
 New Hampshire     New Jersey     New Mexico       New York
             3              3              2              2
North Carolina   North Dakota           Ohio       Oklahoma
             1              3              3              3
        Oregon   Pennsylvania   Rhode Island South Carolina
             3              3              3              1
  South Dakota      Tennessee          Texas           Utah
             3              1              2              3
       Vermont       Virginia     Washington  West Virginia
             3              3              3              3
     Wisconsin        Wyoming
             3              3
MEHERSHRISHTI>Non.SD <- cut3
MEHERSHRISHTI>SD.1 <- cut.hclust.scale.mod
MEHERSHRISHTI>tab <- table(Non.SD, SD.1)
MEHERSHRISHTI>tab
       SD.1
Non.SD  1  2  3
     1  6  9  1
     2  2  2 10
     3  0  0 20
MEHERSHRISHTI>same <- (tab[1,1] + tab[2,2] + tab[3,3]) / sum(tab)
MEHERSHRISHTI>cat('It appears that ', same*100,'% of the observations are assigned to the same clusters')
It appears that  56 % of the observations are assigned to the same clusters
MEHERSHRISHTI>
```

Provide a justification for your answer.
We experiment above with results for non scaled. Scaling should be done because the units of measure are very different.

Q3)

In this problem, you will generate simulated data, and then perform K-means
clustering on the data.

(a) Generate a simulated data set with 20 observations in each of three classes (i.e.
60 observations total), and 50 variables.
Hint: There are a number of functions in R that you can use to generate data. One
example is the rnorm() function; runif() is another option. Be sure to add a mean
shift to the observations in each class so that there are three distinct classes.

```
X <- rbind(matrix(rnorm(20*50, mean = 0), nrow = 20),
        matrix(rnorm(20*50, mean=0.7), nrow = 20),
        matrix(rnorm(20*50, mean=1.4), nrow = 20))
```

```
MEHERSHRISHTI># (a) Generate a simulated data set with 20 observations in each of three classes (i.e.
MEHERSHRISHTI># 60 observations total), and 50 variables.
MEHERSHRISHTI># Hint: There are a number of functions in R that you can use to generate data. One
MEHERSHRISHTI># example is the rnorm() function; runif() is another option. Be sure to add a mean
MEHERSHRISHTI># shift to the observations in each class so that there are three distinct classes.
MEHERSHRISHTI>X <- rbind(matrix(rnorm(20*50, mean = 0), nrow = 20),
            matrix(rnorm(20*50, mean=0.7), nrow = 20),
            matrix(rnorm(20*50, mean=1.4), nrow = 20))
MEHERSHRISHTI>
```

(b) Perform K-means clustering of the observations with K = 3.
How well do the clusters that you obtained in K-means clustering compare to the
true class labels?

Hint: You can use the table() function in R to compare the true class labels to the
class labels obtained by clustering. Be careful how you interpret the results:
Kmeans clustering will arbitrarily number the clusters, so you cannot simply check
whether the true class labels and clustering labels are the same.

```
res = kmeans(X, centers = 3)
true_class = c(rep(1,20), rep(2,20), rep(3,20))
table(res$cluster, true  class)
```

```
MEHERSHRISHTI>res = kmeans(X, centers = 3)
MEHERSHRISHTI>true_class = c(rep(1,20), rep(2,20), rep(3,20))
MEHERSHRISHTI>table(res$cluster, true_class)
   true_class
     1  2  3
  1  0 20  0
  2  0  0 20
  3 20  0  0
MEHERSHRISHTI>
```

They are all precisely clustered

(c) Perform K-means clustering with K = 2. Describe your results.

```
res = kmeans(X, centers = 2)
true = c(rep(1,20), rep(2,20), rep(3,20))
table(res$cluster, true_class)
```

```
MEHERSHRISHTI># (c) Perform K-means clustering with K = 2. Describe your
 results.
MEHERSHRISHTI>res = kmeans(X, centers = 2)
MEHERSHRISHTI>true = c(rep(1,20), rep(2,20), rep(3,20))
MEHERSHRISHTI>table(res$cluster, true_class)
   true_class
     1  2  3
  1 20  8  0
  2  0 12 20
MEHERSHRISHTI>
```

An incorrect class is assigned to the middle class. The classification of the extreme classes is accurate.

(d) Now perform K-means clustering with K = 4, and describe your results.

```
res = kmeans(X, centers = 4)
true = c(rep(1,20), rep(2,20), rep(3,20))
table(res$cluster, true_class)
```

```
MEHERSHRISHTI># (d) Now perform K-means clustering with K = 4, and descri
be your results.
MEHERSHRISHTI>res = kmeans(X, centers = 4)
MEHERSHRISHTI>true = c(rep(1,20), rep(2,20), rep(3,20))
MEHERSHRISHTI>table(res$cluster, true_class)
   true_class
     1  2  3
  1  0  0 10
  2  0 20  0
  3  0  0 10
  4 20  0  0
MEHERSHRISHTI>
```

Two classes have been created out of one class.