Tanmay Gupta

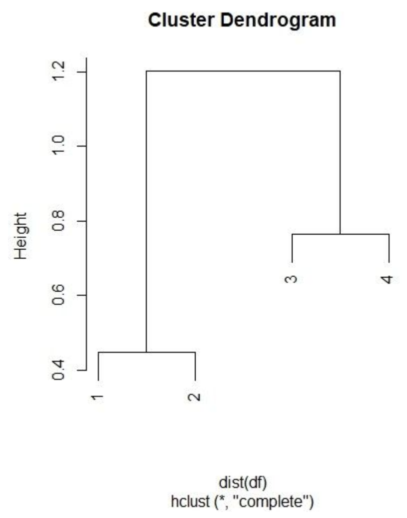
Math 678, Spring 2020
Homework 4 Due: Wednesday, 4/1/2020

1. Complete R Lab 3 from Section 10.6 of our textbook and submit associated R codes and outputs.

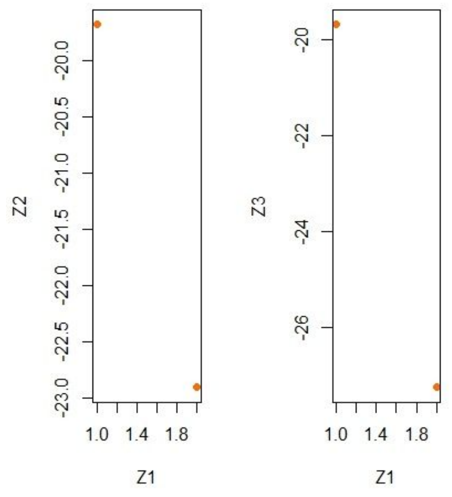
Answer->
Code-

```
library (ISLR)
nci.labs= NCI60$labs
nci.data= NCI60$data
dim(nci.data)
> dim(nci.data)
[1]   64 6830

nci.labs [1:4]
table(nci.labs)
> nci.labs [1:4]
[1] "CNS"   "CNS"   "CNS"   "RENAL"
> table(nci.labs)
nci.labs
    BREAST        CNS       COLON K562A-repro
         7          5           7           1
K562B-repro    LEUKEMIA MCF7A-repro MCF7D-repro
         1          6           1           1
  MELANOMA       NSCLC     OVARIAN    PROSTATE
         8          9           6           2
     RENAL     UNKNOWN
         9           1
> pr.out =prcomp (nci.data , scale=TRUE)
> Cols=function (vec ){
+   + cols=rainbow (length (unique (vec )))
+   + return (cols[as.numeric (as.factor (vec))])}
```



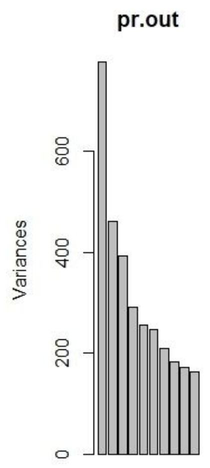
```
> Cols=function(vec){cols=rainbow (length(unique(vec)))
+ +    return(cols[as.numeric(as.factor(vec))])}
> par(mfrow =c(1,2))
> plot(pr.out$x [1:2], col=Cols(nci.labs), pch =19,xlab ="Z1",ylab="Z2")
> plot(pr.out$x[c(1,3)], col=Cols(nci.labs), pch =19,xlab ="Z1",ylab="Z3")
```



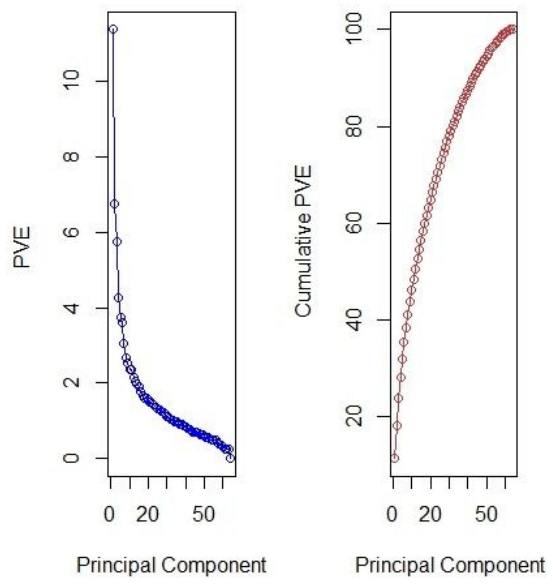
```
> summary (pr.out)
Importance of components:
                         PC1      PC2      PC3
Standard deviation     27.8535 21.48136 19.82046
Proportion of Variance  0.1136  0.06756  0.05752
Cumulative Proportion   0.1136  0.18115  0.23867
                         PC4      PC5      PC6
Standard deviation     17.03256 15.97181 15.72108
Proportion of Variance  0.04248  0.03735  0.03619
Cumulative Proportion   0.28115  0.31850  0.35468
                         PC7      PC8      PC9
Standard deviation     14.47145 13.54427 13.14400
Proportion of Variance  0.03066  0.02686  0.02529
Cumulative Proportion   0.38534  0.41220  0.43750
                        PC10     PC11     PC12
Standard deviation     12.73860 12.68672 12.15769
Proportion of Variance  0.02376  0.02357  0.02164
Cumulative Proportion   0.46126  0.48482  0.50646
                        PC13     PC14     PC15
Standard deviation     11.83019 11.62554 11.43779
Proportion of Variance  0.02049  0.01979  0.01915
Cumulative Proportion   0.52695  0.54674  0.56590
                        PC16     PC17     PC18
Standard deviation     11.00051 10.65666 10.48880
Proportion of Variance  0.01772  0.01663  0.01611
Cumulative Proportion   0.58361  0.60024  0.61635
                        PC19    PC20     PC21    PC22
```

```
Standard deviation     10.43518 10.3219 10.14608 10.0544
Proportion of Variance  0.01594  0.0156  0.01507  0.0148
Cumulative Proportion   0.63229  0.6479  0.66296  0.6778
                          PC23    PC24    PC25    PC26
Standard deviation     9.90265 9.64766 9.50764 9.33253
Proportion of Variance 0.01436 0.01363 0.01324 0.01275
Cumulative Proportion  0.69212 0.70575 0.71899 0.73174
                          PC27   PC28    PC29    PC30
Standard deviation     9.27320 9.0900 8.98117 8.75003
Proportion of Variance 0.01259 0.0121 0.01181 0.01121
Cumulative Proportion  0.74433 0.7564 0.76824 0.77945
                          PC31    PC32    PC33    PC34
Standard deviation     8.59962 8.44738 8.37305 8.21579
Proportion of Variance 0.01083 0.01045 0.01026 0.00988
Cumulative Proportion  0.79027 0.80072 0.81099 0.82087
                          PC35    PC36    PC37    PC38
Standard deviation     8.15731 7.97465 7.90446 7.82127
Proportion of Variance 0.00974 0.00931 0.00915 0.00896
Cumulative Proportion  0.83061 0.83992 0.84907 0.85803
                          PC39    PC40    PC41   PC42
Standard deviation     7.72156 7.58603 7.45619 7.3444
Proportion of Variance 0.00873 0.00843 0.00814 0.0079
Cumulative Proportion  0.86676 0.87518 0.88332 0.8912
                          PC43   PC44    PC45   PC46
Standard deviation     7.10449 7.0131 6.95839 6.8663
Proportion of Variance 0.00739 0.0072 0.00709 0.0069
Cumulative Proportion  0.89861 0.9058 0.91290 0.9198
                          PC47    PC48    PC49    PC50
Standard deviation     6.80744 6.64763 6.61607 6.40793
Proportion of Variance 0.00678 0.00647 0.00641 0.00601
Cumulative Proportion  0.92659 0.93306 0.93947 0.94548
                          PC51    PC52    PC53    PC54
Standard deviation     6.21984 6.20326 6.06706 5.91805
Proportion of Variance 0.00566 0.00563 0.00539 0.00513
Cumulative Proportion  0.95114 0.95678 0.96216 0.96729
                          PC55    PC56    PC57   PC58
Standard deviation     5.91233 5.73539 5.47261 5.2921
Proportion of Variance 0.00512 0.00482 0.00438 0.0041
Cumulative Proportion  0.97241 0.97723 0.98161 0.9857
                          PC59    PC60    PC61    PC62
Standard deviation     5.02117 4.68398 4.17567 4.08212
Proportion of Variance 0.00369 0.00321 0.00255 0.00244
Cumulative Proportion  0.98940 0.99262 0.99517 0.99761
                          PC63      PC64
Standard deviation     4.04124 2.148e-14
Proportion of Variance 0.00239 0.000e+00
Cumulative Proportion  1.00000 1.000e+00
```

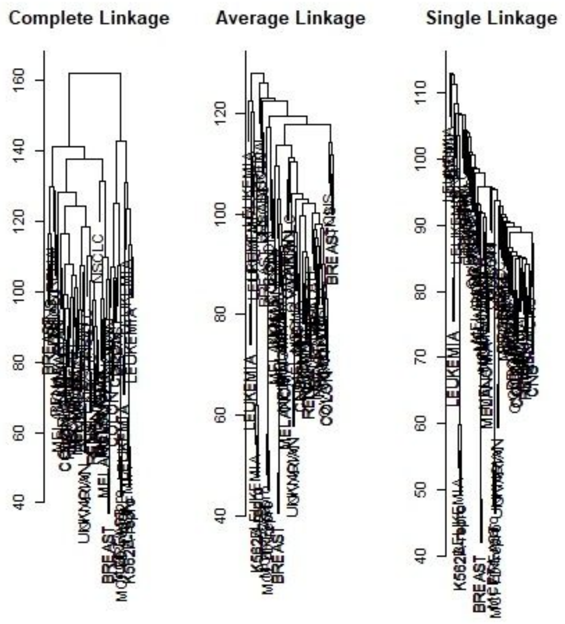
```
> plot(pr.out)
```



```
> pve =100* pr.out$sdev ^2/ sum(pr.out$sdev ^2)
> par(mfrow =c(1,2))
> plot(pve , type ="o", ylab="PVE ", xlab=" Principal Component ",col =" blue")
> plot(cumsum (pve ), type="o", ylab =" Cumulative PVE", xlab="Principal Component ", col ="
brown3 ")
```

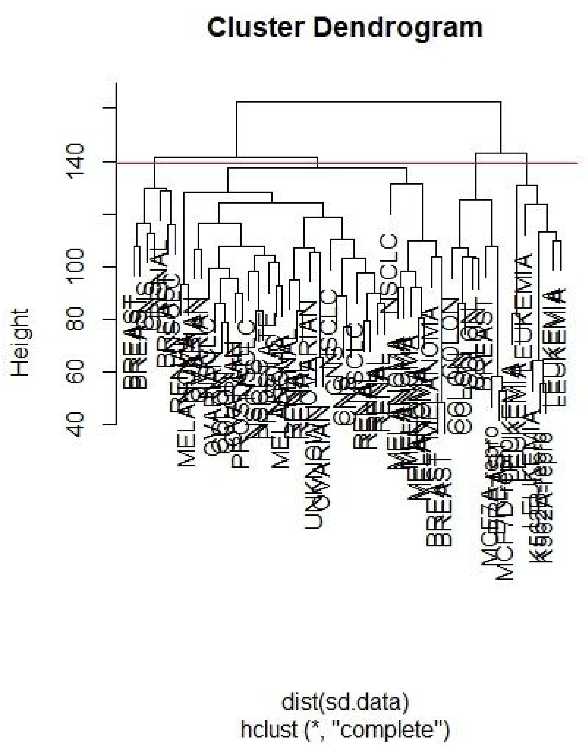


```
> par(mfrow =c(1,3))
> data.dist=dist(sd.data)
> plot(hclust (data.dist), labels =nci.labs , main=" Complete Linkage ", xlab ="", sub ="", ylab ="")
> plot(hclust (data.dist , method ="average"), labels =nci.labs , main=" Average Linkage ", xlab
="", sub ="", ylab ="")
> plot(hclust (data.dist , method ="single"), labels =nci.labs , main=" Single Linkage ", xlab="",
sub ="", ylab ="")
```



```
> hc.out =hclust (dist(sd.data))
> hc.clusters =cutree (hc.out ,4)
> table(hc.clusters ,nci.labs)
           nci.labs
hc.clusters BREAST CNS COLON K562A-repro K562B-repro
          1      2   3     2           0           0
          2      3   2     0           0           0
          3      0   0     0           1           1
          4      2   0     5           0           0
           nci.labs
hc.clusters LEUKEMIA MCF7A-repro MCF7D-repro MELANOMA
          1        0           0           0        8
          2        0           0           0        0
          3        6           0           0        0
          4        0           1           1        0
           nci.labs
hc.clusters NSCLC OVARIAN PROSTATE RENAL UNKNOWN
          1     8       6        2     8       1
          2     1       0        0     1       0
          3     0       0        0     0       0
          4     0       0        0     0       0

> par(mfrow =c(1,1))
> plot(hc.out , labels =nci.labs)
> abline (h=139, col =" red ")
```



```
> hc.out

Call:
hclust(d = dist(sd.data))

Cluster method   : complete
Distance         : euclidean
Number of objects: 64

> set.seed (2)
> km.out =kmeans (sd.data , 4, nstart =20)
> km.clusters =km.out$cluster
> table(km.clusters ,hc.clusters)
           hc.clusters
km.clusters  1  2  3  4
          1 11  0  0  9
          2 20  7  0  0
          3  9  0  0  0
          4  0  0  8  0

> hc.out =hclust (dist(pr.out$x [ ,1:5]) )
> plot(hc.out , labels =nci.labs , main=" Hier. Clust . on First Five Score Vectors ")
> table(cutree (hc.out,4) , nci.labs)
   nci.labs
    BREAST CNS COLON K562A-repro K562B-repro LEUKEMIA
  1      0   2     7           0           0        2
  2      5   3     0           0           0        0
  3      0   0     0           1           1        4
```

```
 4     2  0    0          0          0       0
  nci.labs
   MCF7A-repro MCF7D-repro MELANOMA NSCLC OVARIAN
 1          0          0       1     8       5
 2          0          0       7     1       1
 3          0          0       0     0       0
 4          1          1       0     0       0
  nci.labs
   PROSTATE RENAL UNKNOWN
 1        2     7       0
 2        0     2       1
 3        0     0       0
 4        0     0       0
```

2. Exercises 2, 3, 9, and 10 from Section 10.7 of our textbook.

2. Suppose that we have four observations, for which we compute a
dissimilarity matrix, given by

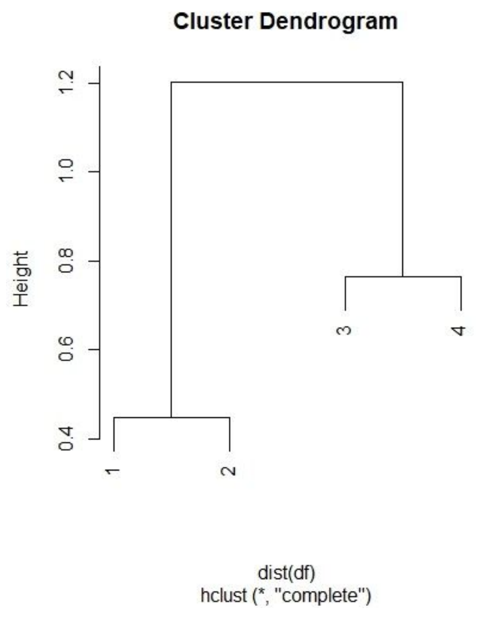
```
[     0.3 0.4   0.7
 0.3       0.5   0.8
 0.4 0.5         0.45
 0.7 0.8 0.45        ]
```

For instance, the dissimilarity between the first and second observations is 0.3, and the dissimilarity between the second and fourth observations is 0.8.

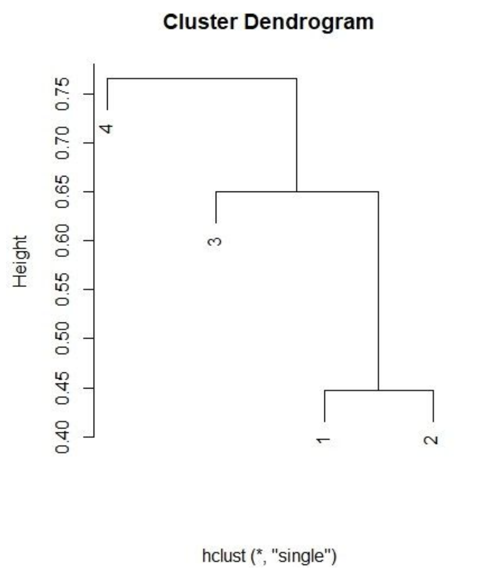
(a) On the basis of this dissimilarity matrix, sketch the dendrogramthat results from hierarchically clustering these four observations using complete linkage. Be sure to indicate on the plot the height at which each fusion occurs, as well as the observations
corresponding to each leaf in the dendrogram.

Annswer->

```
> library(knitr)
> df=data.frame(c(0,0.3,0.4,0.7),c(0.3,0.0,0.5,0.8),c(0.4,0.5,0.0,0.45),c(0.7,0.8,0.45,0.0))
> #print(d)
> colnames(df)=c(paste('Col',1:4))
> kable(df)
| Col 1| Col 2| Col 3| Col 4|
|-----:|-----:|-----:|-----:|
|   0.0|   0.3|  0.40|  0.70|
|   0.3|   0.0|  0.50|  0.80|
|   0.4|   0.5|  0.00|  0.45|
|   0.7|   0.8|  0.45|  0.00|
> hcl<- hclust(dist(df))
> plot(hcl)
```



(b) Repeat (a), this time using single linkage clustering.



(c) Suppose that we cut the dendogram obtained in (a) such that two clusters result. Which observations are in each cluster?

Answer-> If we cut the dendogram in two clusters, We will obtain (1,2) in first and (3,4) in second.

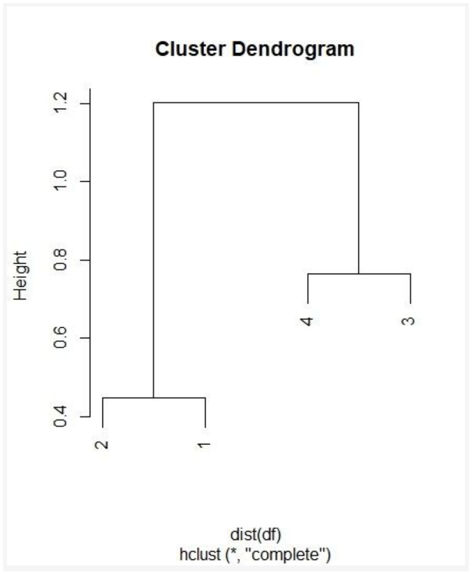
(d) Suppose that we cut the dendogram obtained in (b) such that two clusters result. Which observations are in each cluster?

Answer-> The clusters obtained here are (4) and (1,2,3).

(e) It is mentioned in the chapter that at each fusion in the dendrogram, the position of the two clusters being fused can be swapped without changing the meaning of the dendrogram. Draw a dendrogram that is equivalent to the dendrogram in (a), for which two or more of the leaves are repositioned, but for which the meaning of the dendrogram is the same.

Answer-> A dendogram is read bottom up, where the height indicates where clusters are fused. Thus there is no horizontal meaning, the leafs are be swapped but they still represent clusters that are fused at the same height.

```
row.names(DissMatrix)=c(2,1,4,3)
plot(hclust(dist(DissMatrix)))
```

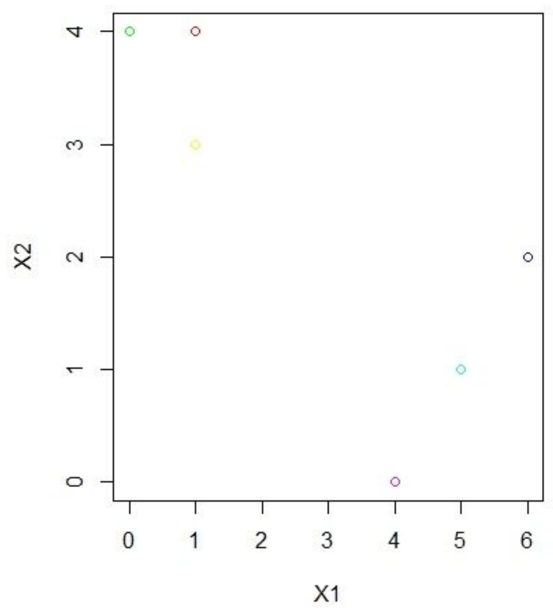


3. In this problem, you will perform K-means clustering manually, with
K = 2, on a small example with n = 6 observations and p = 2
features. The observations are as follows.

Obs.	X1	X2
1	1	4
2	1	3
3	0	4
4	5	1
5	6	2
6	4	0

(a) Plot the observations.
Answer->

```
tb=data.frame(c(1,1,0,5,6,4),c(4,3,4,1,2,0))
colnames(tb)=c('X1','X2')
rownames(tb)=1:6
plot(tb,col=rainbow(6))
```



(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.

Answer->

```
set.seed(1)
> x = cbind(c(1, 1, 0, 5, 6, 4), c(4, 3, 4, 1, 2, 0))
> labels = sample(2, nrow(x), replace=T)
> labels
[1] 1 2 1 1 2 1
```

(c) Compute the centroid for each cluster.

Answer->

```
> centroid1 = c(mean(x[labels==1, 1]), mean(x[labels==1, 2]))
> centroid2 = c(mean(x[labels==2, 1]), mean(x[labels==2, 2]))
> centroid1
[1] 2.50 2.25
> centroid2
[1] 3.5 2.5
```

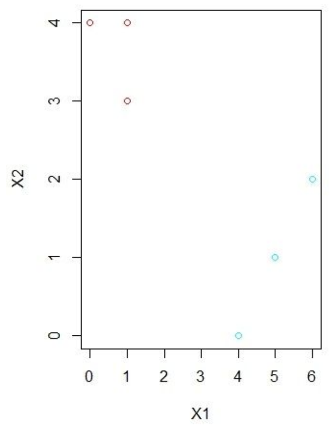
(d) Assign each observation to the centroid to which it is closest, in terms of Euclidean distance. Report the cluster labels for each Observation.

Answer->

```
> euclid = function(a, b) {
+     return(sqrt((a[1] - b[1])^2 + (a[2]-b[2])^2))
+ }
> assign_labels = function(x, centroid1, centroid2) {
+     labels = rep(NA, nrow(x))
+     for (i in 1:nrow(x)) {
```

```
+         if (euclid(x[i,], centroid1) < euclid(x[i,], centroid2)) {
+             labels[i] = 1
+         } else {
+             labels[i] = 2
+         }
+     }
+     return(labels)
+ }
> labels = assign_labels(x, centroid1, centroid2)
> labels

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

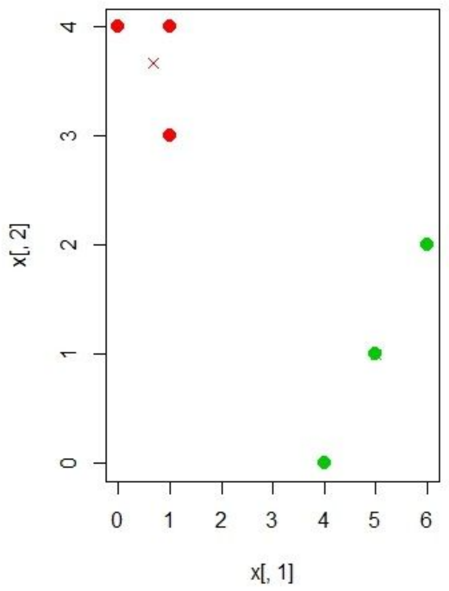


(e) Repeat (c) and (d) until the answers obtained stop changing.
Answer->

```
> last_labels = rep(-1, 6)
> while (!all(last_labels == labels)) {
+     last_labels = labels
+     centroid1 = c(mean(x[labels==1, 1]), mean(x[labels==1, 2]))
+     centroid2 = c(mean(x[labels==2, 1]), mean(x[labels==2, 2]))
+     print(centroid1)
+     print(centroid2)
+     labels = assign_labels(x, centroid1, centroid2)
+ }
[1] 0.6666667 3.6666667
[1] 5 1
> labels
[1] 1 1 1 2 2 2
```

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

```
> plot(x[,1], x[,2], col=(labels+1), pch=20, cex=2)
> points(centroid1[1], centroid1[2], col=2, pch=4)
> points(centroid2[1], centroid2[2], col=3, pch=4)
```

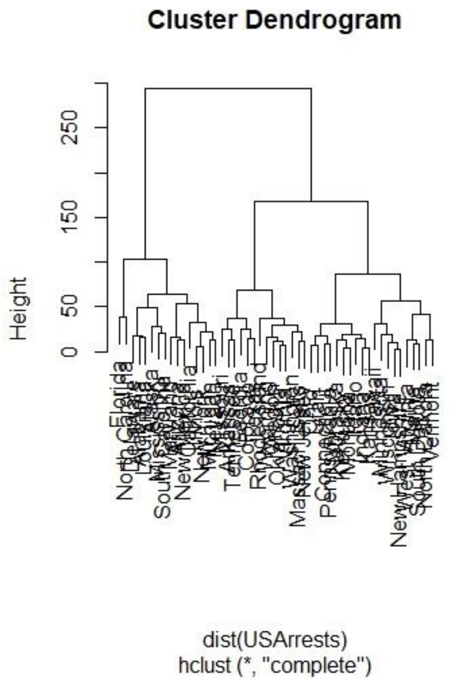


9. Consider the USArrests data. We will now perform hierarchical clustering on the states.

(a) Using hierarchical clustering with complete linkage and Euclidean distance, cluster the states.

Answer->

```
> set.seed(2)
> hc.complete = hclust(dist(USArrests), method="complete")
> plot(hc.complete)
```



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

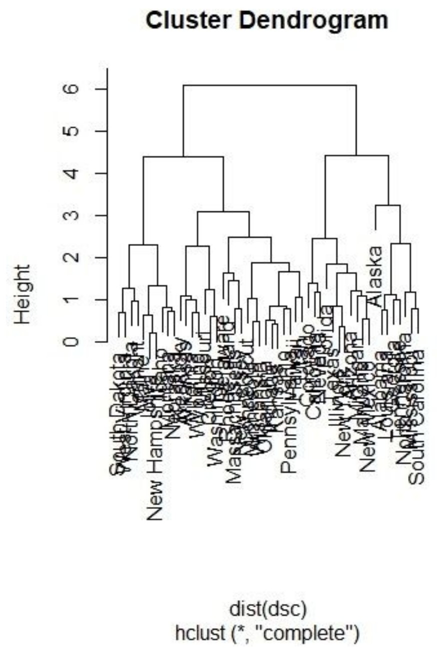
Answer->

```
> cutree(hc.complete, 3)
       Alabama         Alaska        Arizona       Arkansas
             1              1              1              2
    California       Colorado    Connecticut       Delaware
             1              2              3              1
       Florida        Georgia         Hawaii          Idaho
             1              2              3              3
      Illinois        Indiana           Iowa         Kansas
             1              3              3              3
      Kentucky      Louisiana          Maine       Maryland
             3              1              3              1
 Massachusetts       Michigan      Minnesota    Mississippi
             2              1              3              1
      Missouri        Montana       Nebraska         Nevada
             2              3              3              1
 New Hampshire     New Jersey     New Mexico       New York
             3              2              1              1
North Carolina   North Dakota           Ohio       Oklahoma
             1              3              3              2
        Oregon   Pennsylvania   Rhode Island South Carolina
             2              3              2              1
  South Dakota      Tennessee          Texas           Utah
             3              2              2              3
       Vermont       Virginia     Washington  West Virginia
             3              2              2              3
     Wisconsin        Wyoming
             3              2
> table(cutree(hc.complete, 3))
 1  2  3
16 14 20
```

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

Answer->

```
> dsc = scale(USArrests)
> hc.s.complete = hclust(dist(dsc), method="complete")
> plot(hc.s.complete)
```



(d) What effect does scaling the variables have on the hierarchical clustering obtained? In your opinion, should the variables be scaled before the inter-observation dissimilarities are computed? Provide a justification for your answer.

Answer-> Scaling the variables effects the max height of the dendogram obtained from hierarchical clustering. From a cursory glance, it doesn't effect the bushiness of the tree obtained. However, it does affect the clusters obtained from cutting the dendogram into 3 clusters. In my opinion, for this data set the data should be standardized because the data measured has different units (UrbanPop compared to other three columns).

```
> cutree(hc.s.complete, 3)
     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
```

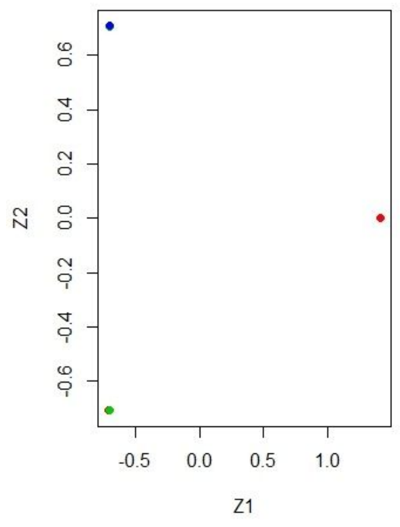
```
           3            3
> table(cutree(hc.s.complete, 3))
 1  2  3
 8 11 31
> table(cutree(hc.s.complete, 3), cutree(hc.complete, 3))
    1  2  3
  1  6  2  0
  2  9  2  0
  3  1 10 20
```

10. In this problem, you will generate simulated data, and then perform PCA and 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.
Answer->

```
set.seed(42)
data= matrix(sapply(1:3,function(x){ rnorm(20*50,mean = 10*sqrt(x))  }),ncol=50)    # 20 obs. in
each class with 50 features.
class=unlist(lapply(1:3,function(x){rep(x,20)}))
```

(b) Perform PCA on the 60 observations and plot the first two principal component score vectors. Use a different color to indicate the observations in each of the three classes. If the three classes appear separated in this plot, then continue on to part (c). If
not, then return to part (a) and modify the simulation so that there is greater separation between the three classes. Do not continue to part (c) until the three classes show at least some separation in the first two principal component score vectors.
Answer->

```
> pca.out = prcomp(x)
> plot(pca.out$x[,1:2], col=2:4, xlab="Z1", ylab="Z2", pch=19)
```



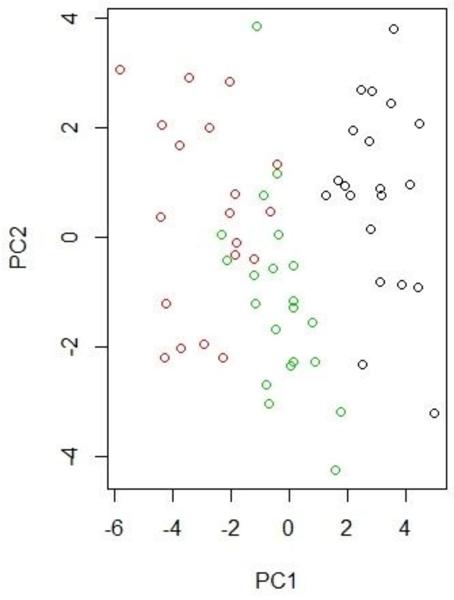
(c) Perform K-means clustering of the observations with K = 3. How well do the clusters that you obtained in K-means clustering compare to the true class labels? Hint: You can use the table()

function in R to compare the true class labels to the class labels obtained by clustering. Be careful how you interpret the results: K-means clustering will arbitrarily number the clusters, so you cannot simply check whether the true class labels and clustering labels are the same.

Answer->

```
> set.seed(1)
> kmeans.out=kmeans(data,3)
> table(kmeans.out$cluster)
 1  2  3
20 19 21
> table(class)
class
 1  2  3
20 20 20
```

We can see that there is only one observation that is miss classified.



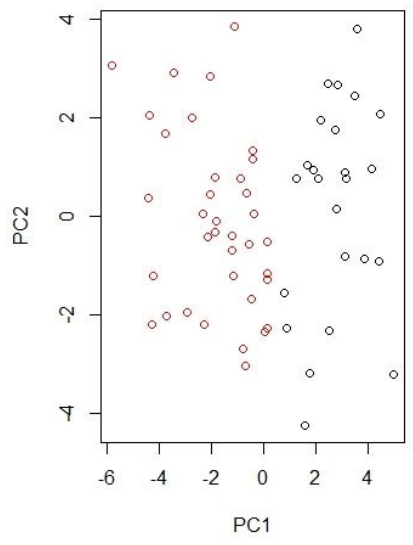
(d) Perform K-means clustering with K = 2. Describe your results

Answer->

```
> set.seed(1)
> kmeans.out=kmeans(data,2)
> table(kmeans.out$cluster)
 1  2
24 36
> table(class)
class
 1  2  3
20 20 20
```

K-means seem to find a single cluster that is the same as before. This can clearly be observed in the picture below as the red cluster closely matches the original green cluster.

```
> plot(pr.out$x[,c(1,2)],col=kmeans.out$cluster)
```

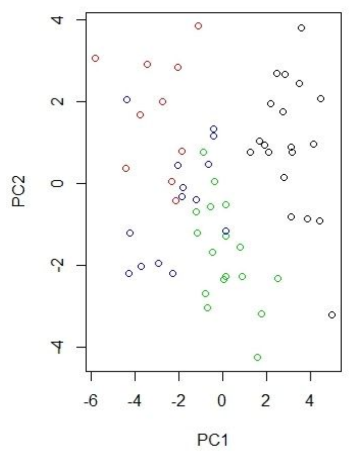


(e) Now perform K-means clustering with K = 4, and describe your results.
Answer->

```
> set.seed(1)
> kmeans.out=kmeans(data,4)
> table(kmeans.out$cluster)
 1  2  3  4
19 10 17 14
> table(class)
class
 1  2  3
20 20 20
```

When using 4 clusters it becomes more difficult to determine the difference between the new found clusters and the actual class values. However, by examining the plot we can see that it again find the original green cluster with some overlap between it and the remaining ones. Overlap between clusters in the two principal components is also clear, as should be expected since they may be close in the remaining dimensions.

```
> plot(pr.out$x[,c(1,2)],col=kmeans.out$cluster)
```



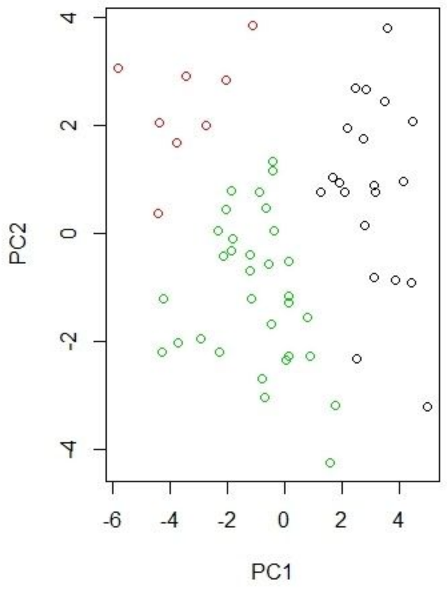
(f) Now perform K-means clustering with K = 3 on the first two principal component score vectors, rather than on the raw data. That is, perform K-means clustering on the 60 × 2 matrix of which the first column is the first principal component score vector, and the second column is the second principal component score vector. Comment on the results.

Answer->

```
> set.seed(1)
> kmeans.out=kmeans(pr.out$x[,c(1,2)],3)
> table(kmeans.out$cluster)
 1  2  3
20  8 32
> table(class)
class
 1  2  3
20 20 20
```

The algorithm performs well when clustering on the first two principal components, however, since it is missing information about the remaining dimensions observations that are close in the first two components are assigned to the same cluster which leads to mistakes. By examining the plot as before we can see that this is true, as there is no overlap.

```
> plot(pr.out$x[,c(1,2)],col=kmeans.out$cluster)
```



(g) Using the scale() function, perform K-means clustering with K = 3 on the data after scaling each variable to have standard deviation one. How do these results compare to those obtained in (b)? Explain.

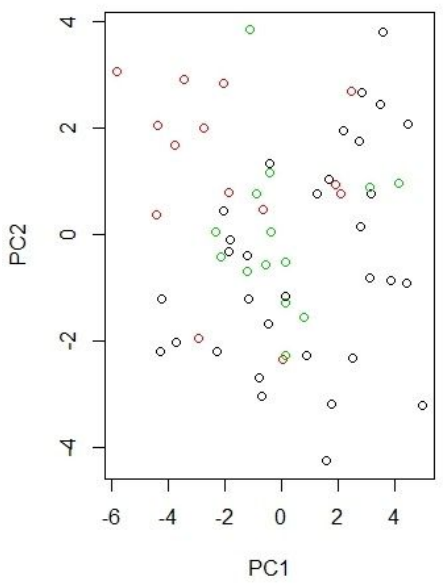
Answer->

```
> set.seed(1)
> kmeans.out=kmeans(scale(data,center = T,scale = T),3)
> table(kmeans.out$cluster)
 1  2  3
32 14 14
> table(class)
class
```

```
 1  2  3
20 20 20
```

There is significant overlap in the first two clusters, and the algorithm performs poorly.

```
> plot(pr.out$x[,c(1,2)],col=kmeans.out$cluster)
```


PC2
PC1