Data Mining II: Clustering Assignment

Allen Rahrooh

14 April 2020

Contents

Problem 10.9

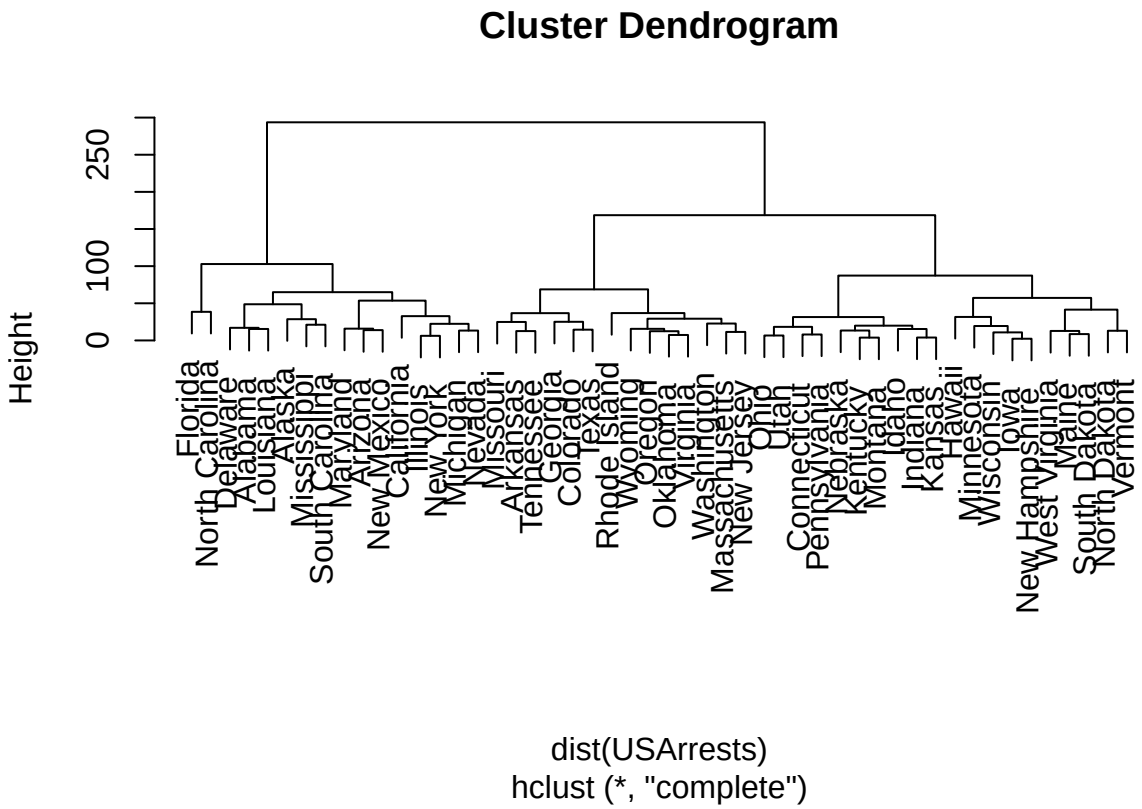
Consider the USArrests data. We will now perform hierarchical clustering on the states.

Part A

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

```
library(ISLR)
set.seed(2)

hc.complete = hclust(dist(USArrests), method="complete")
plot(hc.complete)
```



Part B

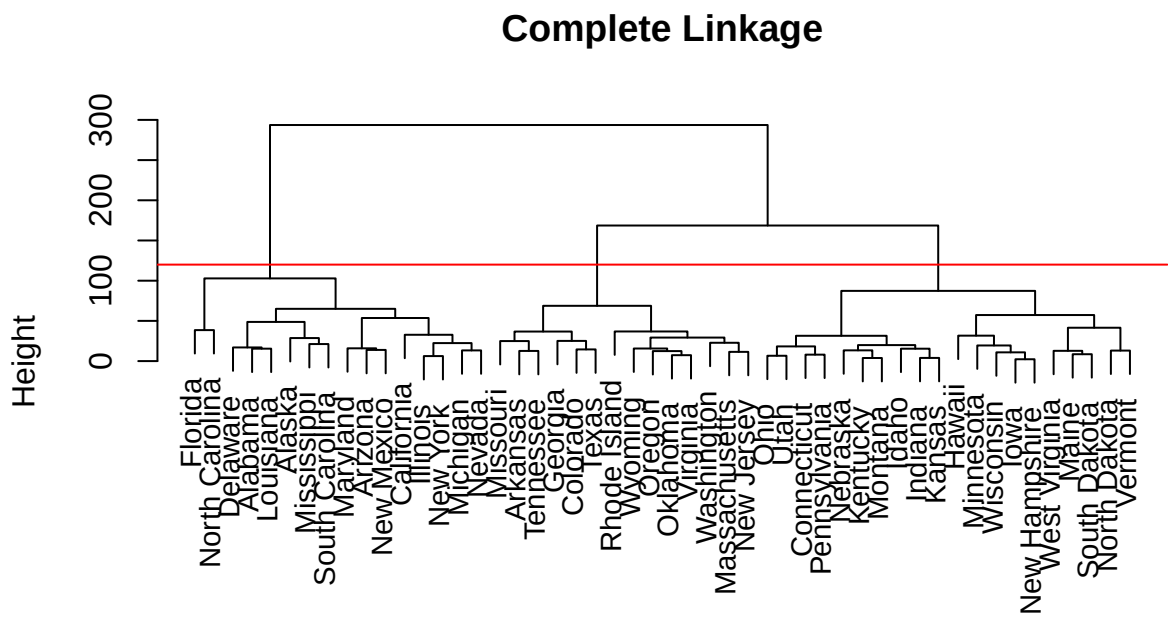
Cut the dendrogram at a height that results in three distinct clusters. Which states belong to which clusters?

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

```
##        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
```

```
plot(hc.complete, main = "Complete Linkage", xlab = "", sub = "", cex = .9)
abline(h = 120, col = "red")
```




```
table(cutree(hc.complete, 3))
```

```
##
##  1  2  3
## 16 14 20
```

Cluster 1: Alabama, Alaska, Arizona, Calfornia, Delaware, Florida, Illinois, Louisiana, Maryland, Michigan, Mississippi, Nevada, New Mexico, New York, North Carolina, South Carolina

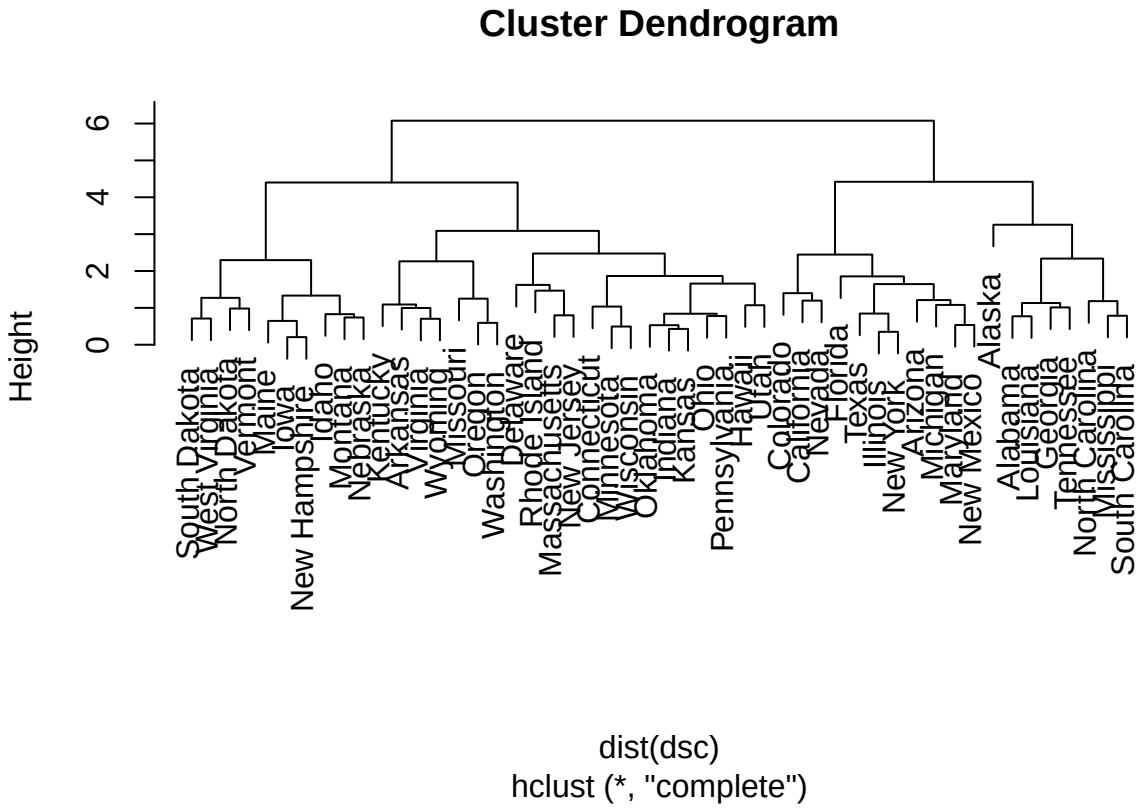
Cluster 2: Arkansas, Colorado, Georgia, Massachusetts, Missouri, New Jersey, Oklahoma, Oregon, Rhode Island, Tennessee, Texas, Virginia, Washington, Wyoming

Cluster 3: Connecticut, Hawaii, Idaho, Indiana, Iowa, Kansas, Kentucky, Maine, Minnesota, Montana, Nebraska, New Hampshire, North Dakota, Ohio, Pennsylvania, South Dakota, Utah, Vermont, West Virginia, Wisconsin

Part C

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

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



Part 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.

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

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

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

```
##
##  1  2  3
##  8 11 31
```

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

```
##
##      1  2  3
##   1  6  9  1
##   2  2  2 10
##   3  0  0 20
```

There is not much difference between the scaled and unscaled data cluster dendrogram's. Scaling the variables effects the max height of the dendogram but for this data it should be standardized because the measured data has different units.

Problem 10.10

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

Part 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 = matrix(rnorm(20*3*50, mean=0, sd=0.001), ncol=50)
x[1:20, 2] = 1
x[21:40, 1] = 2
x[21:40, 2] = 2
x[41:60, 1] = 1
true.labels <- c(rep(1, 20), rep(2, 20), rep(3, 20))
true.labels <- factor(true.labels, levels = c(1,2,3), labels = c("True Clust. 1",
                                                                 "True Clust. 2",
                                                                 "True Clust. 3"))
```

Part B

Perform PCA on the 60 observations and plot the first two prin- cipal 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.

```
pca.out = prcomp(x)
summary(pca.out)
```

```
## Importance of components:
##                           PC1    PC2      PC3      PC4      PC5      PC6
## Standard deviation     1.008 0.5821 0.001731 0.001673 0.001648 0.001582
## Proportion of Variance 0.750 0.2499 0.000000 0.000000 0.000000 0.000000
## Cumulative Proportion  0.750 1.0000 0.999970 0.999970 0.999970 0.999970
##                            PC7      PC8      PC9     PC10     PC11     PC12
## Standard deviation     0.001543 0.001497 0.001474 0.001411 0.001393 0.001335
## Proportion of Variance 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
## Cumulative Proportion  0.999980 0.999980 0.999980 0.999980 0.999980 0.999980
##                           PC13     PC14     PC15     PC16    PC17     PC18
## Standard deviation     0.001297 0.001257 0.001244 0.001226 0.00116 0.001118
## Proportion of Variance 0.000000 0.000000 0.000000 0.000000 0.00000 0.000000
## Cumulative Proportion  0.999980 0.999990 0.999990 0.999990 0.99999 0.999990
##                           PC19     PC20     PC21      PC22      PC23      PC24
## Standard deviation     0.001091 0.001021 0.001012 0.0009849 0.0009378 0.0009316
## Proportion of Variance 0.000000 0.000000 0.000000 0.0000000 0.0000000 0.0000000
## Cumulative Proportion  0.999990 0.999990 0.999990 0.9999900 0.9999900 0.9999900
##                            PC25      PC26      PC27     PC28      PC29
```

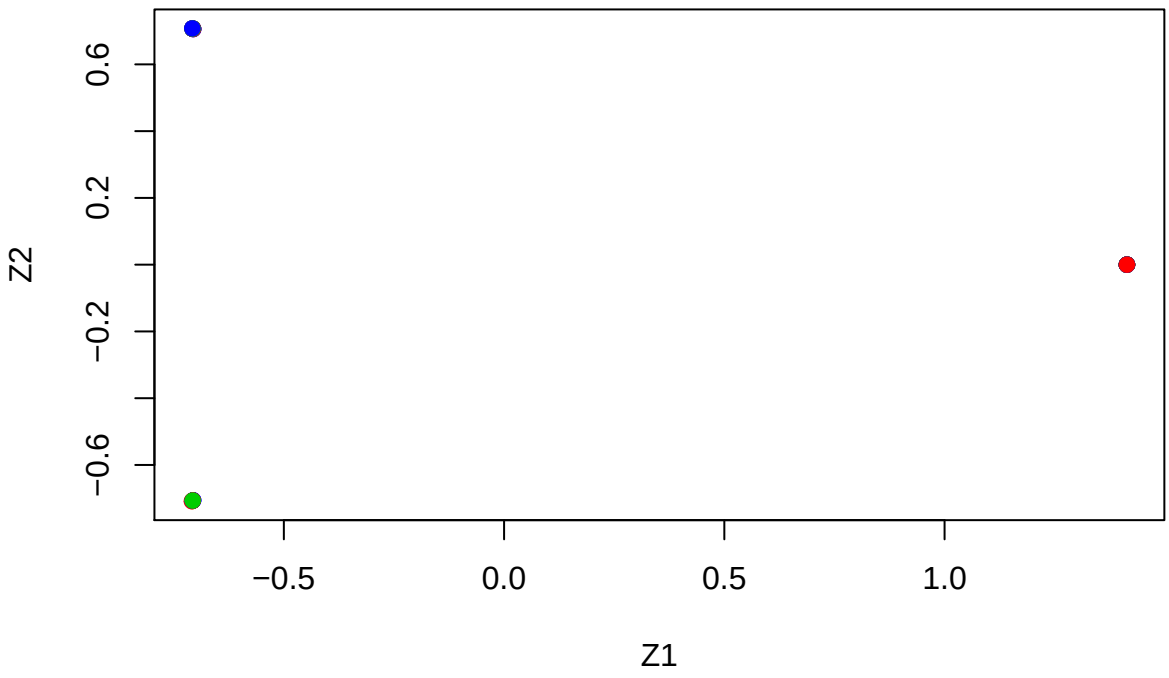
```
## Standard deviation     0.0009081 0.0008668 0.0008228 0.000801 0.0007486
## Proportion of Variance 0.0000000 0.0000000 0.0000000 0.000000 0.0000000
## Cumulative Proportion  0.9999900 1.0000000 1.0000000 1.000000 1.0000000
##                            PC30      PC31      PC32      PC33      PC34
## Standard deviation     0.0007124 0.0006966 0.0006733 0.0006323 0.0005909
## Proportion of Variance 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
## Cumulative Proportion  1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
##                            PC35      PC36      PC37      PC38      PC39
## Standard deviation     0.0005654 0.0005381 0.0005325 0.0004756 0.0004476
## Proportion of Variance 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
## Cumulative Proportion  1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
##                            PC40      PC41      PC42      PC43      PC44
## Standard deviation     0.0004261 0.0003914 0.0003774 0.0003144 0.0002964
## Proportion of Variance 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
## Cumulative Proportion  1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
##                            PC45      PC46      PC47      PC48     PC49
## Standard deviation     0.0002732 0.0002495 0.0001915 0.0001466 0.000129
## Proportion of Variance 0.0000000 0.0000000 0.0000000 0.0000000 0.000000
## Cumulative Proportion  1.0000000 1.0000000 1.0000000 1.0000000 1.000000
##                            PC50
## Standard deviation     7.787e-05
## Proportion of Variance 0.000e+00
## Cumulative Proportion  1.000e+00
```

```
pca.out$x[,1:2]
```

```
##              PC1           PC2
##  [1,] -0.7079228 -7.076535e-01
##  [2,] -0.7071573 -7.068897e-01
##  [3,] -0.7061651 -7.058937e-01
##  [4,] -0.7080866 -7.078204e-01
##  [5,] -0.7073449 -7.070720e-01
##  [6,] -0.7071940 -7.069282e-01
##  [7,] -0.7067857 -7.065196e-01
##  [8,] -0.7074565 -7.071901e-01
##  [9,] -0.7058831 -7.056148e-01
## [10,] -0.7073860 -7.071167e-01
## [11,] -0.7069927 -7.067245e-01
## [12,] -0.7065928 -7.063271e-01
## [13,] -0.7075651 -7.072968e-01
## [14,] -0.7080215 -7.077547e-01
## [15,] -0.7060274 -7.057605e-01
## [16,] -0.7089206 -7.086542e-01
## [17,] -0.7066636 -7.063975e-01
## [18,] -0.7072613 -7.069955e-01
## [19,] -0.7065710 -7.063041e-01
## [20,] -0.7069825 -7.067125e-01
## [21,]  1.4140879 -7.088995e-05
## [22,]  1.4140874 -6.893673e-05
## [23,]  1.4140878 -7.309105e-05
## [24,]  1.4140860 -6.934807e-05
## [25,]  1.4140859 -7.045088e-05
## [26,]  1.4140878 -6.983408e-05
## [27,]  1.4140866 -7.447988e-05
```

```
## [28,]  1.4140855 -6.910345e-05
## [29,]  1.4140863 -7.290071e-05
## [30,]  1.4140863 -7.054766e-05
## [31,]  1.4140874 -6.989590e-05
## [32,]  1.4140872 -7.065443e-05
## [33,]  1.4140855 -7.013529e-05
## [34,]  1.4140874 -7.005627e-05
## [35,]  1.4140877 -7.296213e-05
## [36,]  1.4140863 -6.608230e-05
## [37,]  1.4140894 -7.285675e-05
## [38,]  1.4140867 -6.920393e-05
## [39,]  1.4140872 -6.744255e-05
## [40,]  1.4140861 -6.949980e-05
## [41,] -0.7064221  7.064353e-01
## [42,] -0.7069992  7.070155e-01
## [43,] -0.7074064  7.074178e-01
## [44,] -0.7077105  7.077255e-01
## [45,] -0.7077918  7.078046e-01
## [46,] -0.7057358  7.057486e-01
## [47,] -0.7065186  7.065305e-01
## [48,] -0.7057616  7.057767e-01
## [49,] -0.7074790  7.074930e-01
## [50,] -0.7074294  7.074432e-01
## [51,] -0.7079055  7.079200e-01
## [52,] -0.7073606  7.073721e-01
## [53,] -0.7068480  7.068625e-01
## [54,] -0.7062227  7.062363e-01
## [55,] -0.7067821  7.067967e-01
## [56,] -0.7068591  7.068727e-01
## [57,] -0.7063127  7.063250e-01
## [58,] -0.7063711  7.063849e-01
## [59,] -0.7071057  7.071214e-01
## [60,] -0.7077361  7.077523e-01
```

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



Part 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.

```
set.seed(36215)
km.out = kmeans(x, 3, nstart=20)
km.out$cluster
```

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

```
km.out$cluster <- factor(km.out$cluster, levels = c(3,2,1), labels = c("K-means Cluster 1",
                                                                       "K-means Cluster 2",
                                                                       "K-means Cluster 3"))
table(true.labels, km.out$cluster)
```

```
##
## true.labels     K-means Cluster 1 K-means Cluster 2 K-means Cluster 3
##   True Clust. 1                20                 0                 0
##   True Clust. 2                 0                20                 0
##   True Clust. 3                 0                 0                20
```

The observations are perfectly clustered.

Part D

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

```
km.out = kmeans(x, 2, nstart=20)
km.out$cluster <- factor(km.out$cluster, levels = c(1,2), labels = c("K-means Cluster 1",
                                                                     "K-means Cluster 2"))
table(true.labels, km.out$cluster)
```

```
##
## true.labels     K-means Cluster 1 K-means Cluster 2
##   True Clust. 1                20                 0
##   True Clust. 2                 0                20
##   True Clust. 3                20                 0
```

All the observations from the one of the three clusters in the true labels are absorbed into the one of two clusters from K-means.

Part E

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

```
set.seed(100)
km.out = kmeans(x, 4, nstart=20)
km.out$cluster
```

```
##  [1] 1 1 1 1 1 3 1 3 1 1 3 3 3 3 1 3 1 3 3 1 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
## [39] 4 4 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
km.out$cluster <- factor(km.out$cluster, levels = c(1,2,3,4), labels = c("K-means Clus. 1",
                                                                         "K-means Clus. 2",
                                                                         "K-means Clus. 3",
                                                                         "K-means Clus. 4"))
table(true.labels, km.out$cluster)
```

```
##
## true.labels     K-means Clus. 1 K-means Clus. 2 K-means Clus. 3 K-means Clus. 4
##   True Clust. 1              11               0               9               0
##   True Clust. 2               0               0               0              20
##   True Clust. 3               0              20               0               0
```

The first cluster from the true label is split into two clusters from K-means.

Part 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.

```
set.seed(65656)
km.out = kmeans(pca.out$x[,1:2], 3, nstart=20)
km.out$cluster
```

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

```
km.out$cluster <- factor(km.out$cluster, levels = c(1,2,3), labels = c("K-means Cluster 1",
                                                                       "K-means Cluster 2",
                                                                       "K-means Cluster 3"))
table(true.labels, km.out$cluster)
```

```
##
## true.labels     K-means Cluster 1 K-means Cluster 2 K-means Cluster 3
##   True Clust. 1                 0                 0                20
##   True Clust. 2                 0                20                 0
##   True Clust. 3                20                 0                 0
```

The observations are perfectly clustered.

Part 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.

```
km.out = kmeans(scale(x), 3, nstart=20)
km.out$cluster
```

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

```
km.out$cluster <- factor(km.out$cluster, levels = c(1,2,3), labels = c("K-means Cluster 1",
                                                                       "K-means Cluster 2",
                                                                       "K-means Cluster 3"))
table(true.labels, km.out$cluster)
```

```
##
## true.labels     K-means Cluster 1 K-means Cluster 2 K-means Cluster 3
##   True Clust. 1                15                 3                 2
##   True Clust. 2                 0                 0                20
##   True Clust. 3                 6                12                 2
```

We see that scaling the data results in worse clustering, since scaling affects the distance between the observations.