Tarea 2

Galindo Torres Bernardo Antonio, Miranda Peñafiel Melissa, Pacheco Martínez Mariana

8. In Section 10.2.3, a formula for calculating PVE was given in Equation 10.8. We also saw that the PVE can be obtained using the $sdev$ output of the $prcomp()$ function. On the USArrests data, calculate PVE in two ways:

(a) Using the $sdev$ output of the $prcomp()$ function, as was done in Section 10.2.3.

```
library(MASS)
data("USArrests")
attach(USArrests)
str(USArrests)
```

```
## 'data.frame':    50 obs. of  4 variables:
##  $ Murder  : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
##  $ Assault : int  236 263 294 190 276 204 110 238 335 211 ...
##  $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
##  $ Rape    : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...
```

```
usa_pc = prcomp(USArrests, scale. = TRUE)
x1<- usa_pc$sdev
s <- summary(usa_pc)
s$importance[2,] #proportion of variance
```

```
##     PC1     PC2     PC3     PC4
## 0.62006 0.24744 0.08914 0.04336
```

```
round(usa_pc$sdev^2,4) #Eigenvalues
```

```
## [1] 2.4802 0.9898 0.3566 0.1734
```

```
round(usa_pc$rotation,3) #Eigenvectors
```

```
##             PC1    PC2    PC3    PC4
## Murder   -0.536  0.418 -0.341  0.649
## Assault  -0.583  0.188 -0.268 -0.743
## UrbanPop -0.278 -0.873 -0.378  0.134
## Rape     -0.543 -0.167  0.818  0.089
```

$$\therefore PVE = \begin{cases} 0.62 & PC1 \\ 0.247 & PC2 \\ 0.089 & PC3 \\ 0.043 & PC4 \end{cases}$$

(b) By applying Equation 10.8 directly. That is, use the $prcomp()$ function to compute the principal component loadings. Then, use those loadings in Equation 10.8 to obtain the PVE.

```
pve <- round(usa_pc$sdev^2/sum(usa_pc$sdev^2),3)
pve
```

```
## [1] 0.620 0.247 0.089 0.043
```

$$\therefore PVE = \begin{cases} 0.62 & PC1 \\ 0.247 & PC2 \\ 0.089 & PC3 \\ 0.043 & PC4 \end{cases}$$

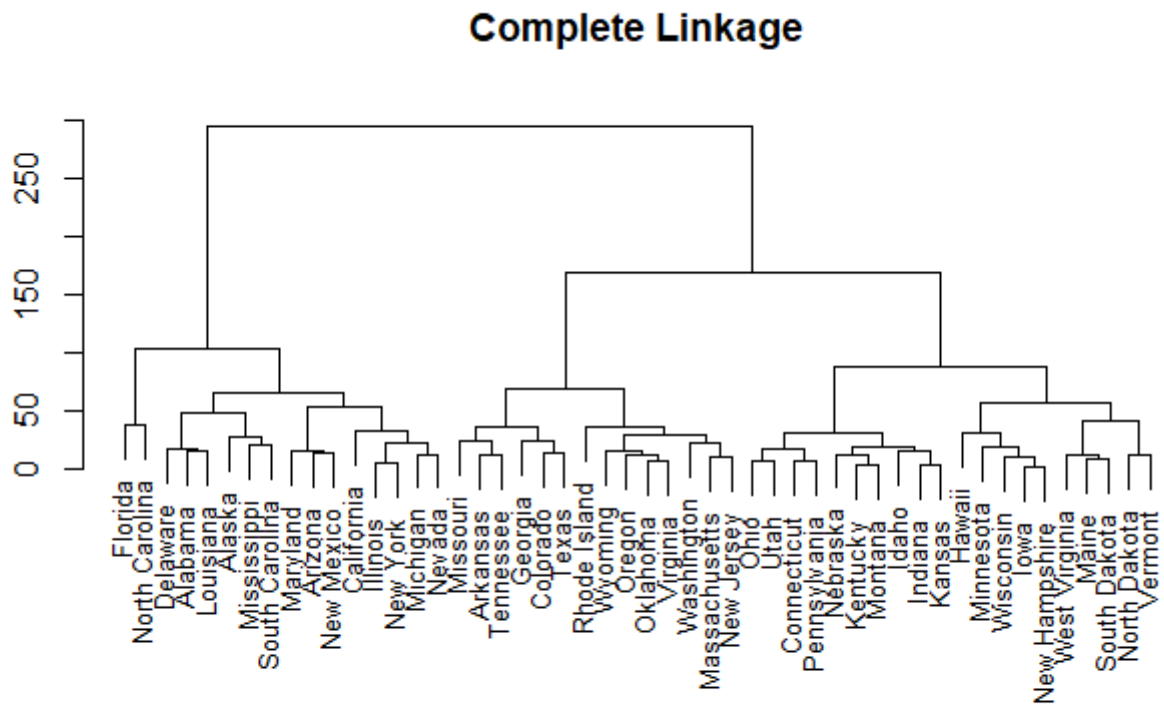
9. Consider the $USArrests$ data. We will now perform hierarchical clustering on the states.

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

```
set.seed(1706)

usa_h_comp=hclust(dist(USArrests), method="complete")
```

```
plot(usa_h_comp,main="Complete Linkage", xlab="", sub="",ylab="",cex=.8)
```



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

```
cluster3=cutree(usa_h_comp, k=3)
for( k in 1:3 ){
  print(k)
  print( rownames( USArrests )[ cluster3 == k ] )
}
```

```
## [1] 1
##  [1] "Alabama"        "Alaska"         "Arizona"        "California"
##  [5] "Delaware"       "Florida"        "Illinois"       "Louisiana"
##  [9] "Maryland"       "Michigan"       "Mississippi"    "Nevada"
## [13] "New Mexico"     "New York"       "North Carolina" "South Carolina"
## [1] 2
##  [1] "Arkansas"      "Colorado"      "Georgia"       "Massachusetts"
##  [5] "Missouri"      "New Jersey"    "Oklahoma"      "Oregon"
##  [9] "Rhode Island"  "Tennessee"     "Texas"         "Virginia"
## [13] "Washington"    "Wyoming"
## [1] 3
```

```
##  [1] "Connecticut"   "Hawaii"        "Idaho"         "Indiana"
##  [5] "Iowa"          "Kansas"        "Kentucky"      "Maine"
##  [9] "Minnesota"     "Montana"       "Nebraska"      "New Hampshire"
## [13] "North Dakota"  "Ohio"          "Pennsylvania"  "South Dakota"
## [17] "Utah"          "Vermont"       "West Virginia" "Wisconsin"
```

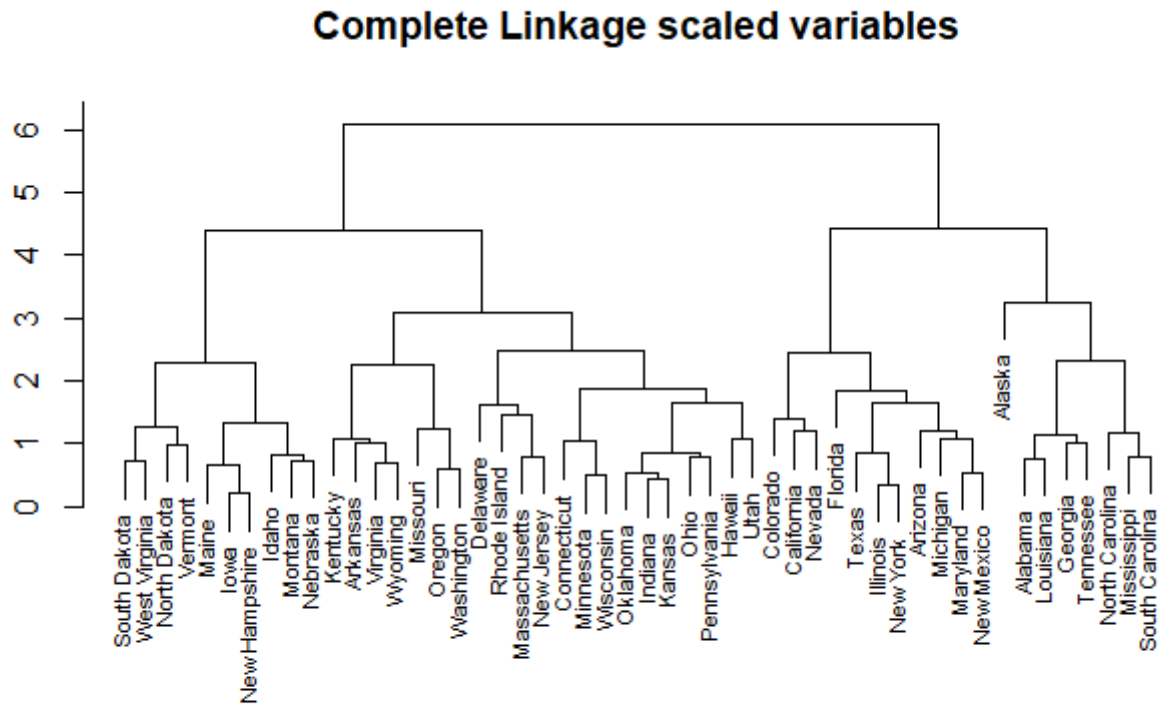
```r
table(cutree(usa_h_comp, 3))
```

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

∴ Cuando divides en 3 ramos, en el primero hay 16 estados, en el segundo 14 y en el tercero 20. Tal como se muestra en la parte de arriba.

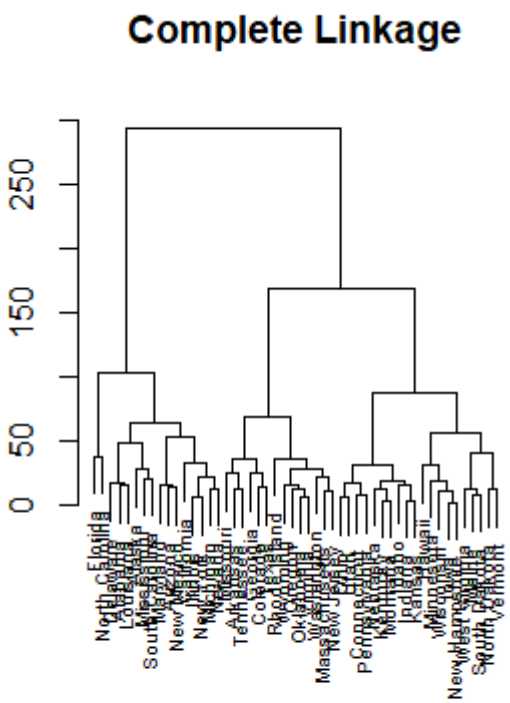
(c) Hierarchically cluster the states using complete linkage and Euclidean distance, $after scaling the variables to have standard deviation one.$

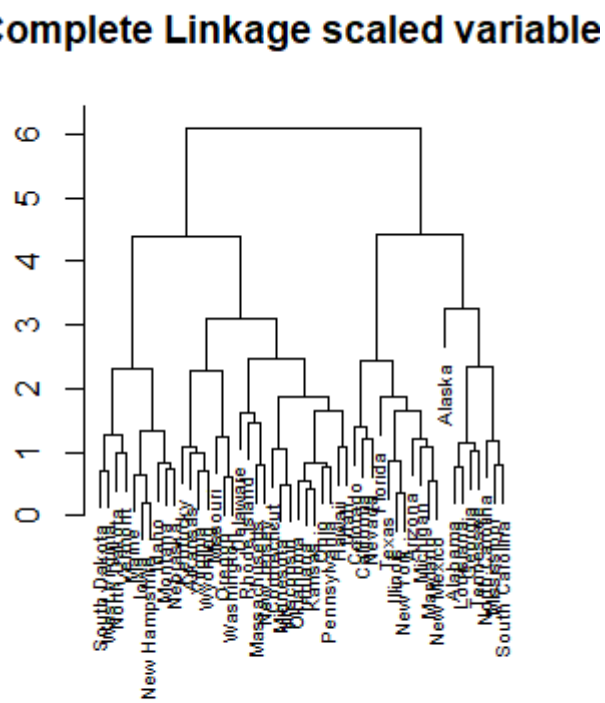
```r
usa_h_comp_s=hclust(dist(scale(USArrests)), method="complete")
par(mfrow=c(1,1))
plot(usa_h_comp_s,main="Complete Linkage scaled variables", xlab="",
  sub="",ylab="", cex=.7)
```



(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.

```
par(mfrow=c(1,2))
plot(usa_h_comp,main="Complete Linkage", xlab="", sub="",ylab="",cex=.6)
plot(usa_h_comp_s,main="Complete Linkage scaled variables", xlab="",
  sub="",ylab="", cex=.6)
```






```
cluster4=cutree(usa_h_comp_s, k=3)
for( k in 1:3 ){
  print(k)
  print( rownames( USArrests )[ cluster4 == k ] )
}
```

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

```
table(cutree(usa_h_comp_s, 3))
```

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

∴ Como podemos ver, al "escalar" las varibales cambia la altura máxima del arreglo y aunque a primera vista parece que no cambia mucho, al dividir en tres los datos ya estandarizados, vemos que sí afecta. En nuestra opinión es mejor trabajar con los datos escalados, ya que los datos de las columnas están medidos en distintas unidades.($Asault$ y $UrbanPop$ son enteros)

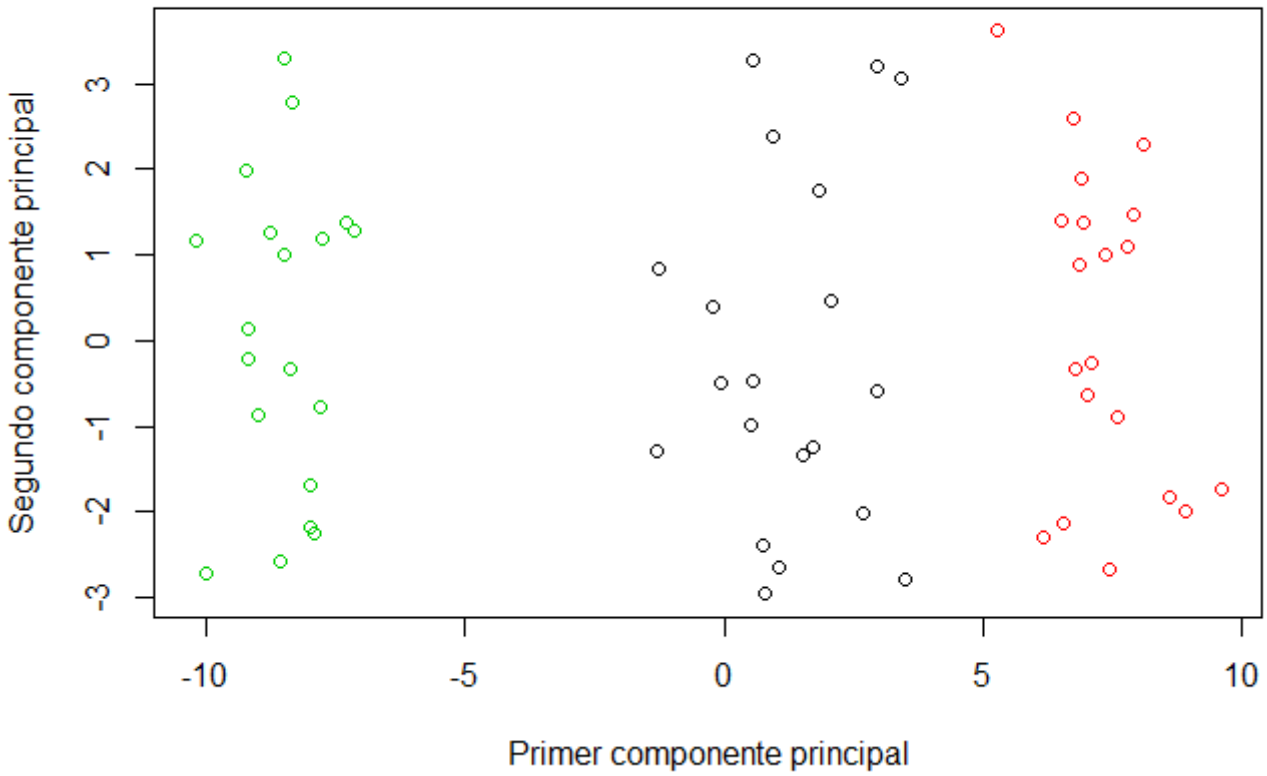
10. In this problem, you will generate simulated data, and then perform PCA and K-means clustering on the data.

(a) Generate a simulated data set with 20 observations in each of three classes (i.e. 60 observations total), and 50 variables.

```
set.seed(30)
datos10 <- rbind(matrix(rnorm(20*50), nrow = 20),
                 matrix(rnorm(20*50), nrow = 20) + .8,
                 matrix(rnorm(20*50), nrow = 20) - 1.4)
```

(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.

```
pca10 = prcomp(datos10)
plot(pca10$x[,1:2], col = c(rep(1,20), rep(2,20), rep(3,20)), xlab = "Primer
  componente principal", ylab = "Segundo componente principal")
```



(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?

```
km3 = kmeans(datos10, centers = 3, nstart = 20)
clases = c(rep(1,20), rep(2,20), rep(3,20))
table(km3$cluster, clases, dnn = c("Clusters", "Clases verdaderas"))
```

```
##         Clases verdaderas
## Clusters  1  2  3
##        1 20  0  0
##        2  0 20  0
##        3  0  0 20
```

∴ Parece ser que se lograron clasificar bien todos los datos ya que se formaron 3 clases con 20 observaciones cada una como los datos originales.

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

```
km2 = kmeans(datos10, centers = 2, nstart = 20)
table(km2$cluster, clases, dnn = c("Clusters", "Clases verdaderas"))
```

```
##         Clases verdaderas
## Clusters  1  2  3
##        1 20 20  0
##        2  0  0 20
```

∴ Se observa que la clase 1 fue absorbida por la clase 2, de aquí sólo los de clase 3 fueron clasificados apropiadamente.

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

```
km4 = kmeans(datos10, centers = 4, nstart = 20)
table(km4$cluster, clases, dnn = c("Clusters", "Clases verdaderas"))
```

```
##         Clases verdaderas
## Clusters  1  2  3
##        1  0 20  0
##        2  6  0  0
##        3  0  0 20
##        4 14  0  0
```

∴ Toda la clase 1 fue asignada al cluster 4, la clase 2 fue asignada a la mitad entre cluster 1 y 2, la única clase que parece que sí fue respetada es la 3 con 20/20 observaciones asignadas correctamente.

(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 x 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.

```
km3pca = kmeans(pca10$x[,1:2], centers = 3, nstart = 20)
table(km3pca$cluster, clases, dnn = c("Clusters", "Clases verdaderas"))
```

```
##         Clases verdaderas
## Clusters  1  2  3
##        1 20  0  0
##        2  0  0 20
##        3  0 20  0
```

∴ Se aprecia que se separaron adecuadamente las observaciones en 3 clusters.

(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.

```
km3scale = kmeans(scale(datos10), centers = 3, nstart = 20)
table(km3scale$cluster, clases, dnn = c("Clusters", "Clases verdaderas"))
```

```
##         Clases verdaderas
## Clusters  1  2  3
##        1  1 20  0
##        2 19  0  0
##        3  0  0 20
```

∴ Podríamos decir que son buenos resultados ya que, de nuevo, se categorizaron las 3 clases con 20 observaciones cada una.

11. On the book website, *www.StatLearning.com*, there is a gene expression data set ($Ch10Ex11.csv$) that consists of 40 tissue samples with measurements on 1,000 genes. The first 20 samples are from healthy patients, while the second 20 are from a diseased group.

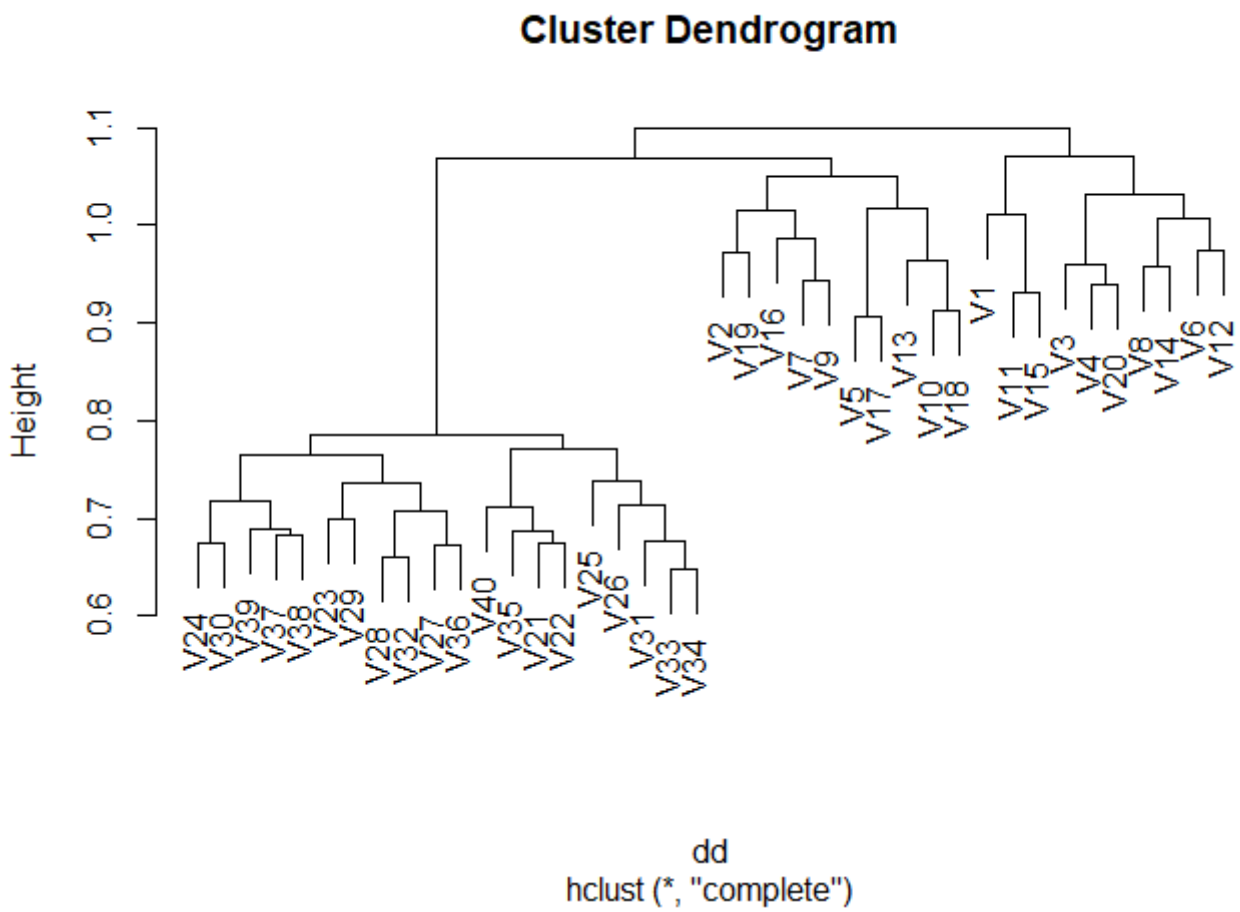
(a) Load in the data using $read.csv()$. You will need to select $header = F$.

```
datos<-read.csv('Ch10Ex11.csv', header = F)
dim(datos)
```

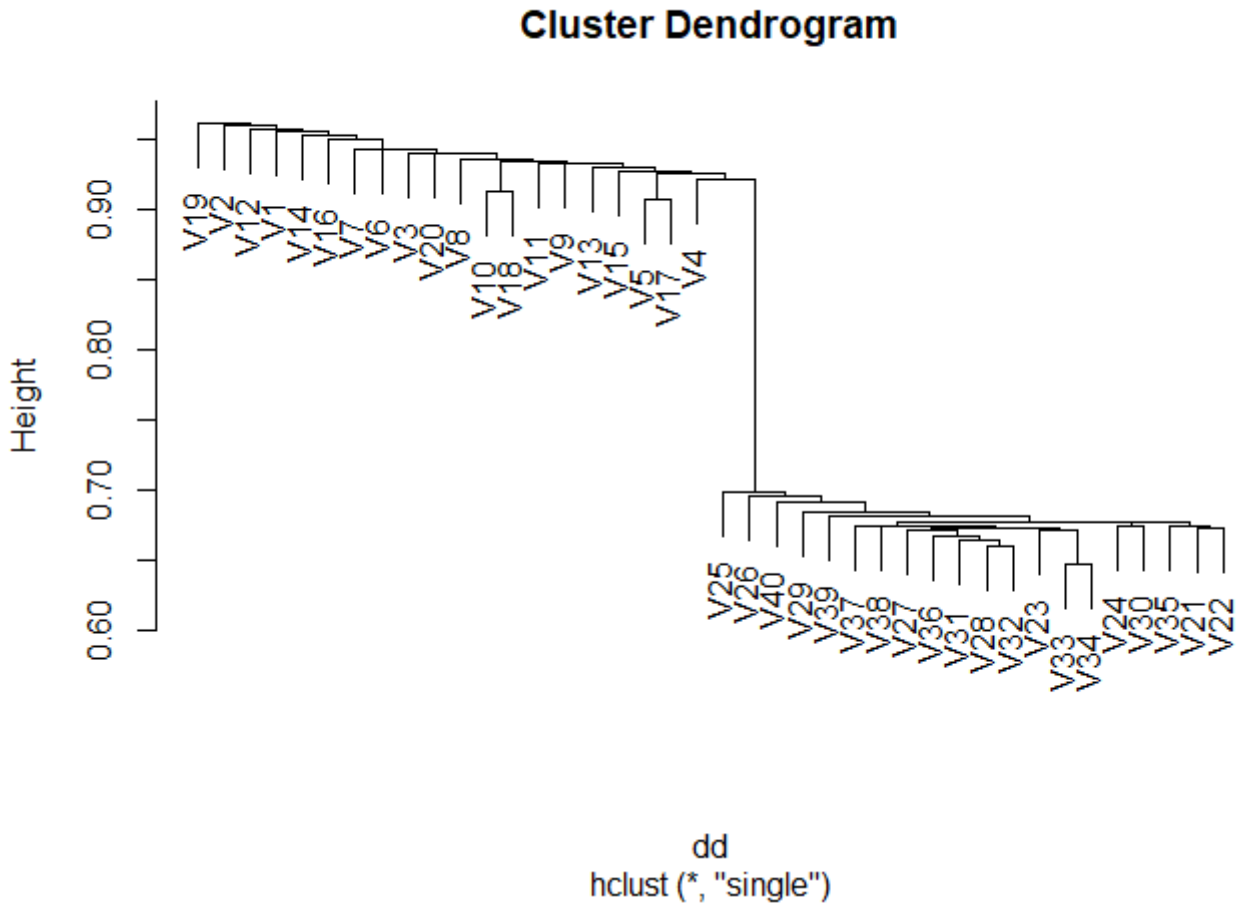
```
## [1] 1000   40
```

(b) Apply hierarchical clustering to the samples using correlationbased distance, and plot the dendrogram. Do the genes separate the samples into the two groups? Do your results depend on the type of linkage used?

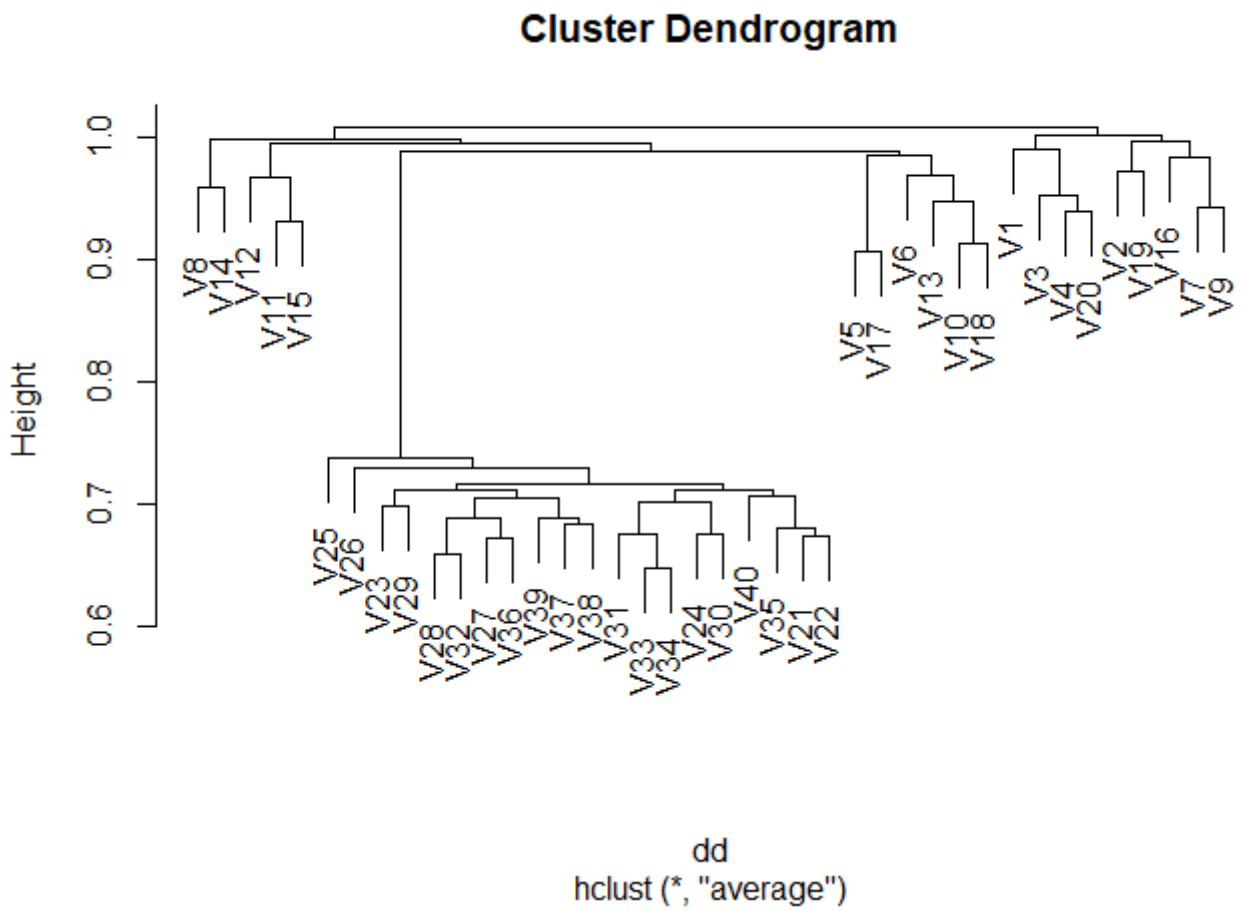
```
dd = as.dist(1 - cor(datos))
plot(hclust(dd, method="complete"))
```




```r
plot(hclust(dd, method="single"))
```



```
plot(hclust(dd, method="average"))
```



∴ Podemos ver que, dependiendo del método, los datos se separan en 2 o 3 grupos.

(c) Your collaborator wants to know which genes differ the most across the two groups. Suggest a way to answer this question, and apply it here.

∴ Para ver qué genes difieren más entre sanos y enfermos podemos aplicar PCA y ver qué genes describen mejor la variabilidad de los datos.

```
x <- variable.names(datos)
pca = prcomp(datos, scale = TRUE)
summary(pca)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
## Standard deviation     2.5401 1.1455 1.12727 1.10915 1.09034 1.06729
## Proportion of Variance 0.1613 0.0328 0.03177 0.03076 0.02972 0.02848
## Cumulative Proportion  0.1613 0.1941 0.22587 0.25663 0.28635 0.31483
##                           PC7     PC8     PC9    PC10   PC11   PC12
## Standard deviation     1.06464 1.05514 1.04623 1.03430 1.0218 1.0040
## Proportion of Variance 0.02834 0.02783 0.02736 0.02674 0.0261 0.0252
## Cumulative Proportion  0.34316 0.37100 0.39836 0.42510 0.4512 0.4764
```

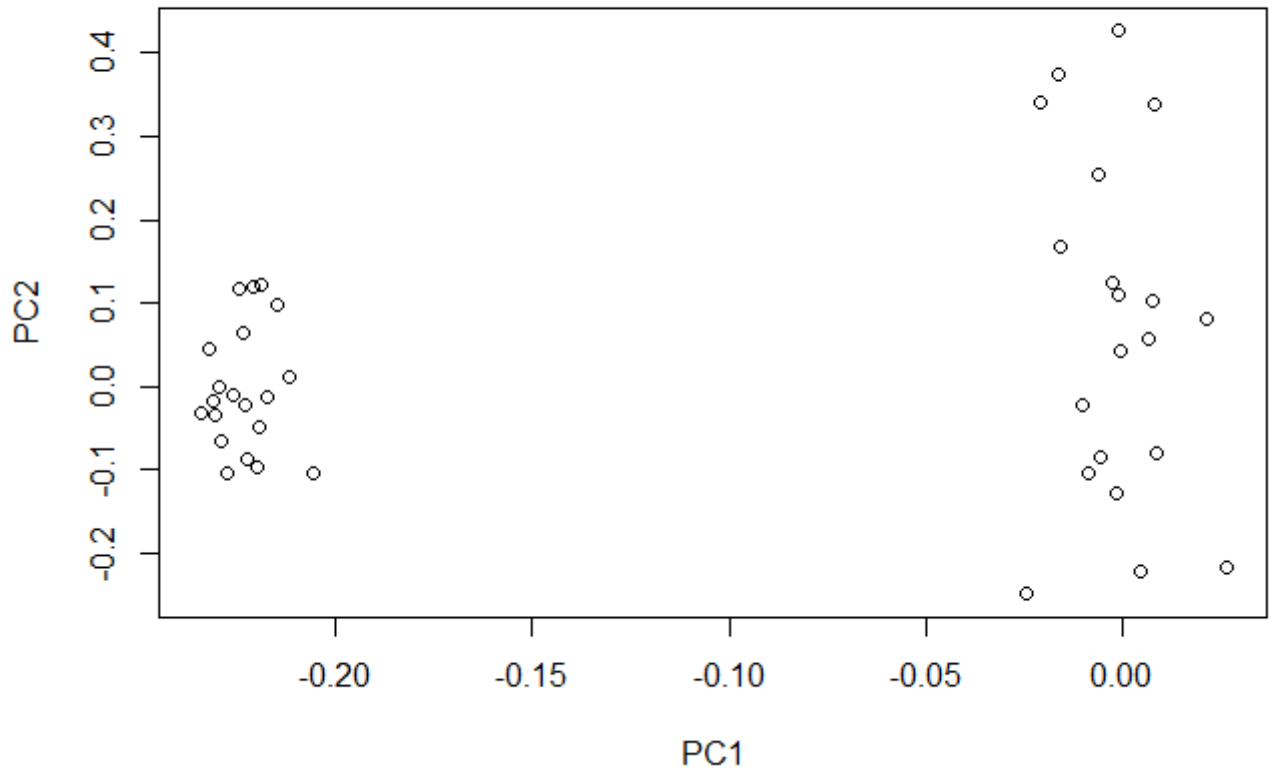
```
##                           PC13    PC14    PC15   PC16    PC17    PC18
## Standard deviation     1.0000 0.99680 0.97913 0.9674 0.96207 0.95900
## Proportion of Variance 0.0250 0.02484 0.02397 0.0234 0.02314 0.02299
## Cumulative Proportion  0.5014 0.52625 0.55021 0.5736 0.59675 0.61974
##                           PC19    PC20    PC21    PC22    PC23    PC24
## Standard deviation     0.94880 0.93664 0.91466 0.90370 0.89974 0.87668
## Proportion of Variance 0.02251 0.02193 0.02092 0.02042 0.02024 0.01921
## Cumulative Proportion  0.64225 0.66418 0.68509 0.70551 0.72575 0.74496
##                           PC25    PC26   PC27    PC28    PC29    PC30
## Standard deviation     0.86341 0.85710 0.8532 0.84544 0.83839 0.82676
## Proportion of Variance 0.01864 0.01837 0.0182 0.01787 0.01757 0.01709
## Cumulative Proportion  0.76360 0.78196 0.8002 0.81803 0.83561 0.85269
##                           PC31    PC32   PC33    PC34    PC35    PC36
## Standard deviation     0.81452 0.79681 0.7924 0.78095 0.77890 0.77086
## Proportion of Variance 0.01659 0.01587 0.0157 0.01525 0.01517 0.01486
## Cumulative Proportion  0.86928 0.88515 0.9009 0.91610 0.93126 0.94612
##                           PC37    PC38    PC39    PC40
## Standard deviation     0.7509 0.74522 0.72987 0.70944
## Proportion of Variance 0.0141 0.01388 0.01332 0.01258
## Cumulative Proportion  0.9602 0.97410 0.98742 1.00000
```

```r
round(pca$sdev^2,4)#Eigenvalores
```

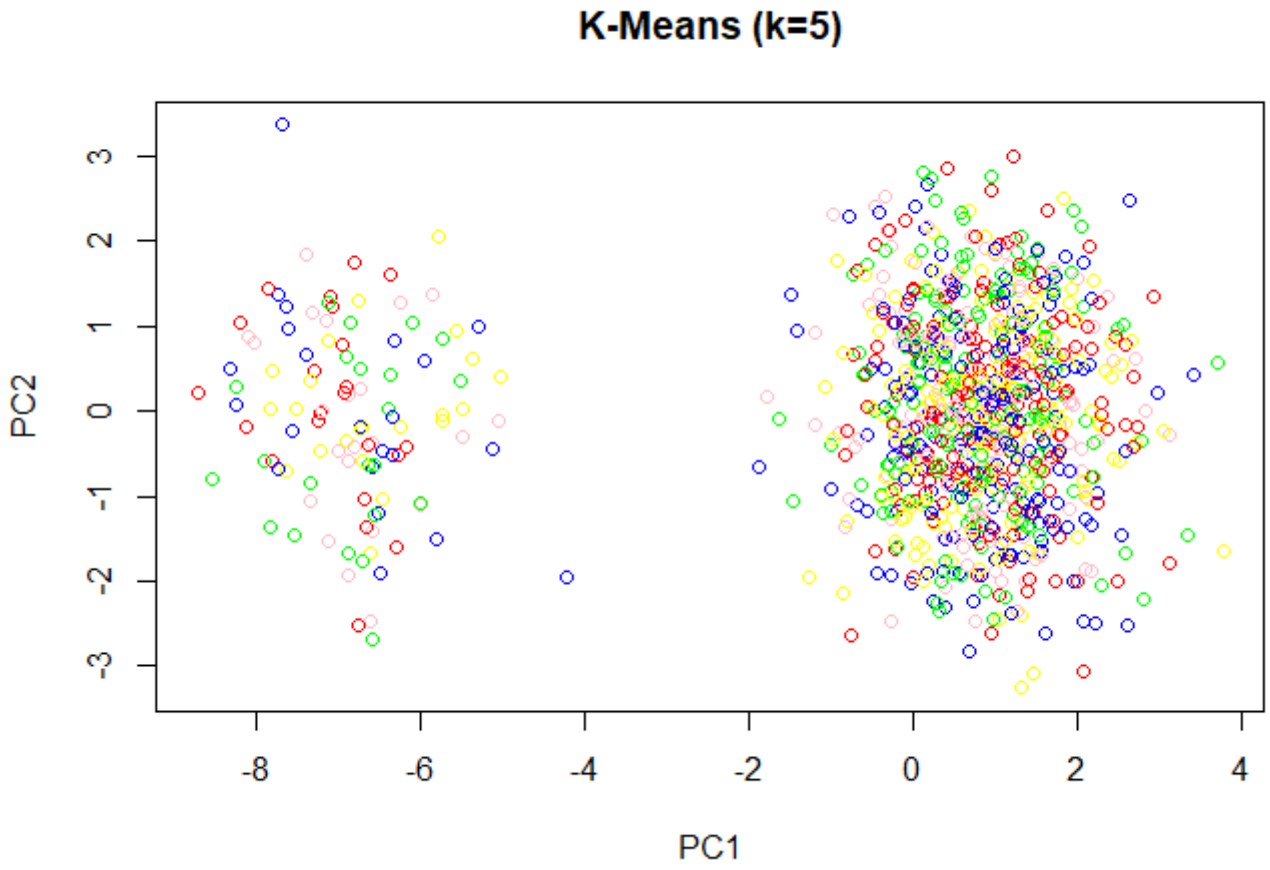
```
##  [1] 6.4521 1.3121 1.2707 1.2302 1.1888 1.1391 1.1335 1.1133 1.0946 1.0698
## [11] 1.0441 1.0080 0.9999 0.9936 0.9587 0.9358 0.9256 0.9197 0.9002 0.8773
## [21] 0.8366 0.8167 0.8095 0.7686 0.7455 0.7346 0.7280 0.7148 0.7029 0.6835
## [31] 0.6634 0.6349 0.6279 0.6099 0.6067 0.5942 0.5638 0.5554 0.5327 0.5033
```

∴ Es claro ver que los primeros 12 componentes son los que mantienen la mayor cantidad de información y variabilidad de los datos.

```r
#round(pca$rotation,3)#Eigenvectores 40 pca x 40 variables
plot(pca$rotation)
```




```
set.seed(1); km5 = kmeans(pca$x[,1:15], 5, nstart = 5)
plot(predict(pca), col = c('red', 'yellow', 'blue', 'green', 'pink'), main = "K-
  Means (k=5)")
```


K-Means (k=5)
PC2
3
2
1
0
-1
-2
-3
-8
-6
-4
-2
0
2
4
PC1