Tarea 4

Ejercicio 1

Comenzaremos cargando las librerías necesarias y cargando nuestra tabla de distancias.

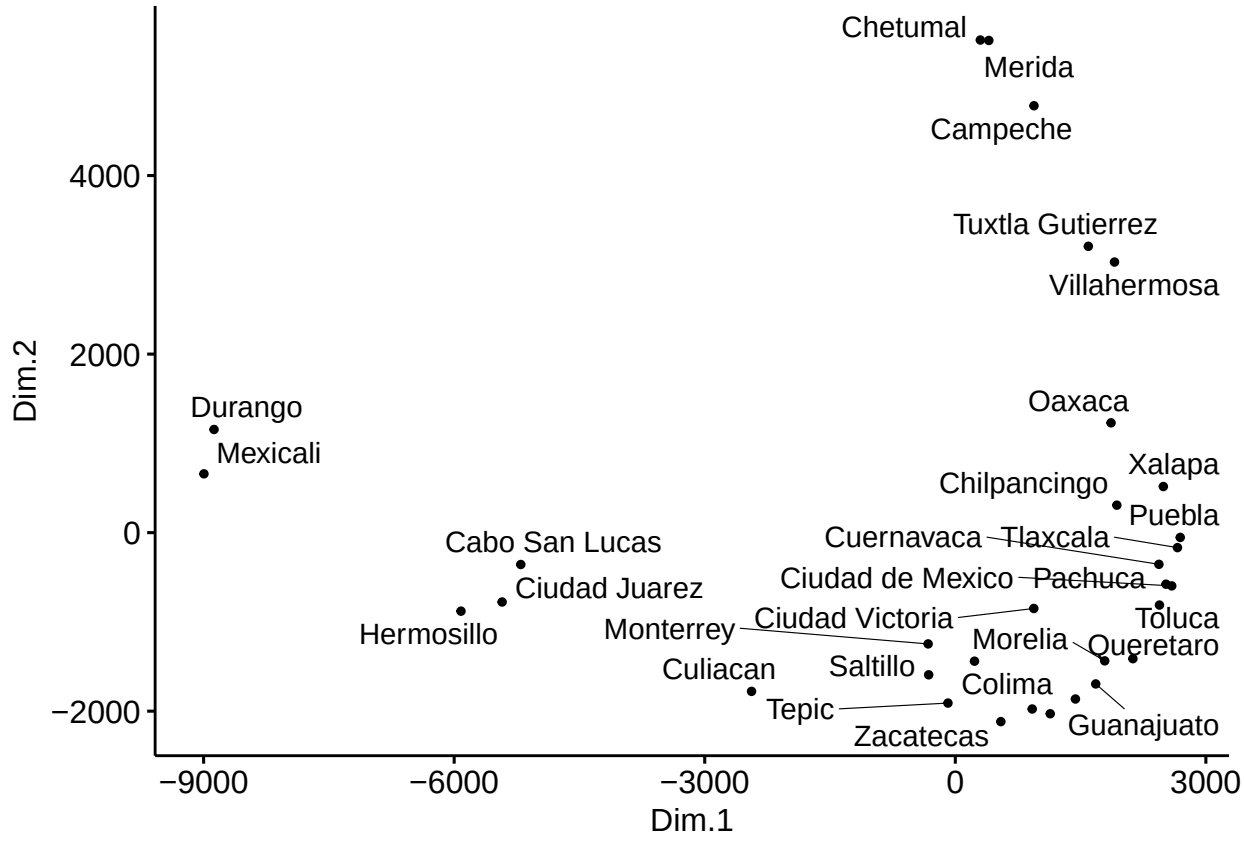
```
library(magrittr)
library(dplyr)
library(ggpubr)
library(MASS)

dist<-read.csv("distanciasMex.csv", header=TRUE, row.names=1)
```

Procedemos a ajustar el EMD y renombramos las coordenadas de los puntos.

```
mds <- dist %>%
  dist() %>%
  cmdscale() %>%
  as_tibble()
colnames(mds) <- c("Dim.1", "Dim.2")
```

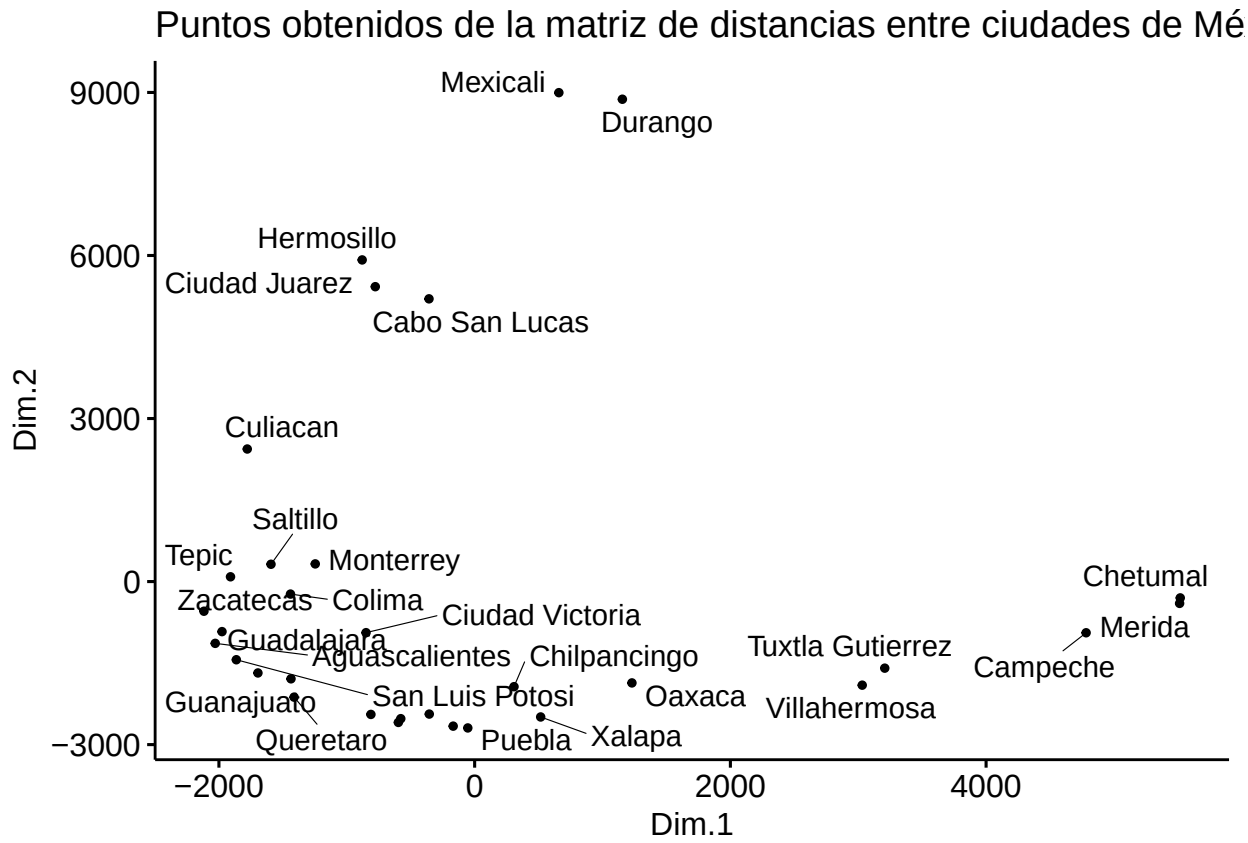
Grafiquemos los puntos ahora. Esperaríamos encontrar un gráfico bastante equivalente al de la República Mexicana. Veamos lo obtenido.

```
ggscatter(mds, x = "Dim.1", y = "Dim.2",
          label = rownames(dist),
          size = 1,
          repel = TRUE)
```



No tuvimos tanta suerte. Sin embargo, si ponemos atención no es tan diferente al mapa de la República, sólo hay que hacer una pequeña rotación de los puntos. Hagámosla y veamos el resultado.

```
A<-matrix(data=c(0,1,-1,0),nrow=2,ncol=2,byrow=TRUE)
mds_rot<- t(A%*%t(mds))
mds_rot<- data.frame(mds_rot)
colnames(mds_rot)<- c("Dim.1","Dim.2")

ggscatter(mds_rot, x = "Dim.1", y = "Dim.2",
          label = rownames(dist),
          size = 1,
          repel = TRUE, main="Puntos obtenidos de la matriz de distancias entre ciudades de México")
```



Definitivamente conseguimos un mejor resultado. Quizás el resultado no es exactamente igual al mapa de la República Mexicana pero podemos observar que hay muchas similitudes, salvo por algunas ciudades (como Mexicali), podríamos decir que es una muy buena aproximación.

Ejercicio 2

Sea $X \in \mathbb{R}^p$ una muestra para dos clases con distribución normal multivariada con matrices de covarianza Σ_1 y Σ_2 diferentes, y medias μ_1 y μ_2. Sean f_1 y f_2 las funciones de densidad de las distribuciones $N_p(\mu_1, \Sigma_1)$ y $N_p(\mu_2, \Sigma_2)$ respectivamente, a continuación construimos un cociente de las funciones de densidad Q para obtener un discriminante cuadrático:

$$
\begin{aligned}
Q(X) &= \log\left(\frac{f_1(X)}{f_2(X)}\right) \\
&= \log\left(\frac{(2\pi)^{-\frac{p}{2}}\left|\Sigma_1\right|^{-\frac{1}{2}}\exp -\frac{1}{2}\big(X-\mu_1\big)^t\Sigma_1^{-1}\big(X-\mu_1\big)}{(2\pi)^{-\frac{p}{2}}\left|\Sigma_2\right|^{-\frac{1}{2}}\exp -\frac{1}{2}\big(X-\mu_2\big)^t\Sigma_2^{-1}\big(X-\mu_2\big)}\right) \\
&= \frac{1}{2}\log\left(\frac{\left|\Sigma_2\right|}{\left|\Sigma_1\right|}\right) + \log\left(\exp\Big(-\frac{1}{2}\big(X-\mu_1\big)^t\Sigma_1^{-1}\big(X-\mu_1\big) + \frac{1}{2}\big(X-\mu_2\big)^t\Sigma_2^{-1}\big(X-\mu_2\big)\Big)\right) \\
&= \frac{1}{2}\log\left(\frac{\left|\Sigma_2\right|}{\left|\Sigma_1\right|}\right) + \frac{1}{2}\big(X-\mu_2\big)^t\Sigma_2^{-1}\big(X-\mu_2\big) - \frac{1}{2}\big(X-\mu_1\big)^t\Sigma_1^{-1}\big(X-\mu_1\big) \\
Q(X) &= \frac{1}{2}\left(\big(X-\mu_2\big)^t\Sigma_2^{-1}\big(X-\mu_2\big) - \frac{1}{2}\big(X-\mu_1\big)^t\Sigma_1^{-1}\big(X-\mu_1\big)\right) + \frac{1}{2}\left(\log\left|\Sigma_2\right| - \log\left|\Sigma_1\right|\right)
\end{aligned}
$$

Ejercicio 3

Leemos los datos y cargamos los paquetes necesarios.

```
library(cluster)
library(factoextra)
library(mclust)
library(ggplot2)
data(ruspini)
```

Comencemos escalando los datos para hacer las variables comparables y obteniendo la matriz de distancias. Tomaremos como base la distancia euclidiana entre nuestras muestras ya que no tenemos mucha información de los datos, y en particular estos no son de tipo factor.

```
datos <- scale(ruspini)
distance <- get_dist(datos)
```

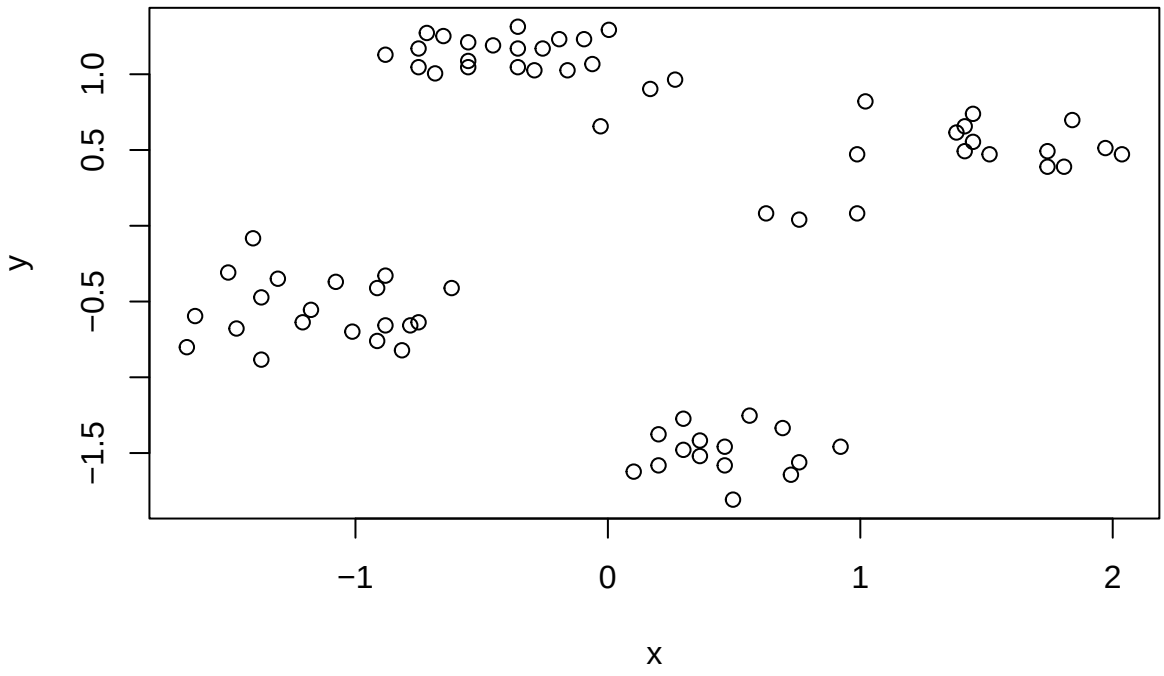
Hagamos diferentes métodos de análisis de conglomerados para hacer una aproximación de cuántos grupos es que hay en estos datos.

K-Medias

Ahora estimemos un número adecuado de clusters.

Afortunadamente, tenemos únicamente dos variables en los datos, por lo que los podemos visualizar en un scatterplot normal y hacernos una idea.

```
plot(datos)
```

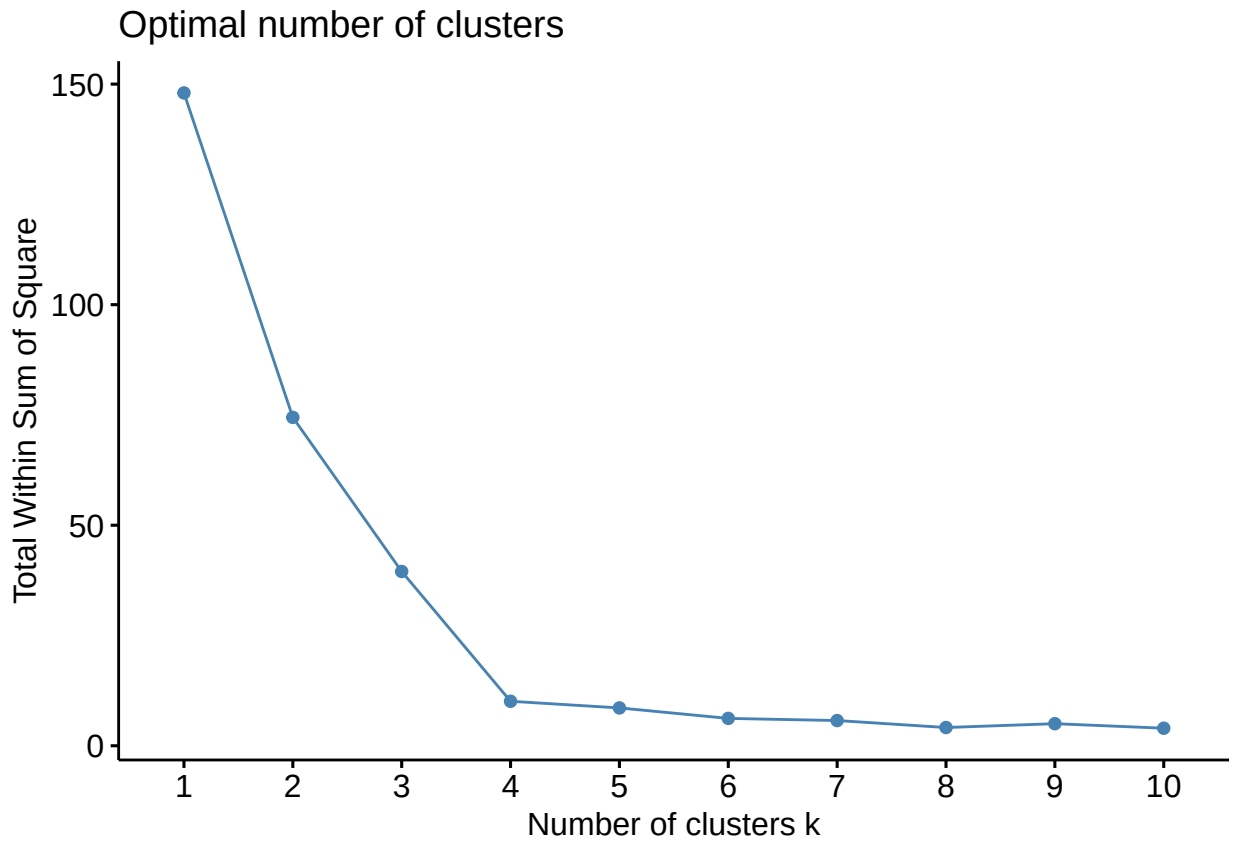


Gracias al sactterplot estaríamos tentados a decir que son 4 grupos. Sin embargo esto no es nada riguroso y

tuvimos suerte en este caso de sólo tener dos variables (aunque en otros casos podríamos usar componentes principales, aún seguiría siendo muy poco riguroso).
Por lo que veamos cuántos clusters nos indican diferentes métodos.

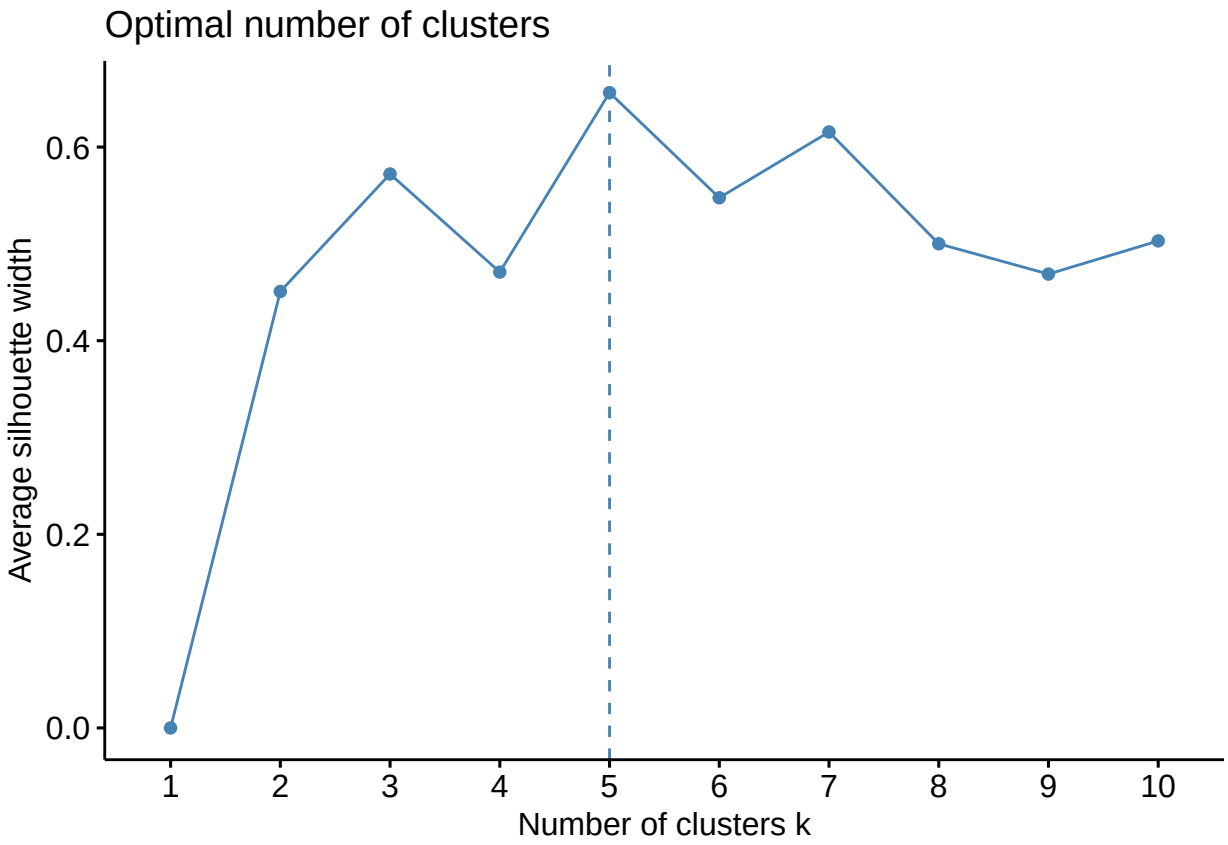
Usando el método del codo, nos indica que 4 clusters es lo adecuado, como habíamos intuido. Lo podemos ver en la siguiente gráfica.

```
set.seed(1)
fviz_nbclust(datos, kmeans, method = "wss")
```



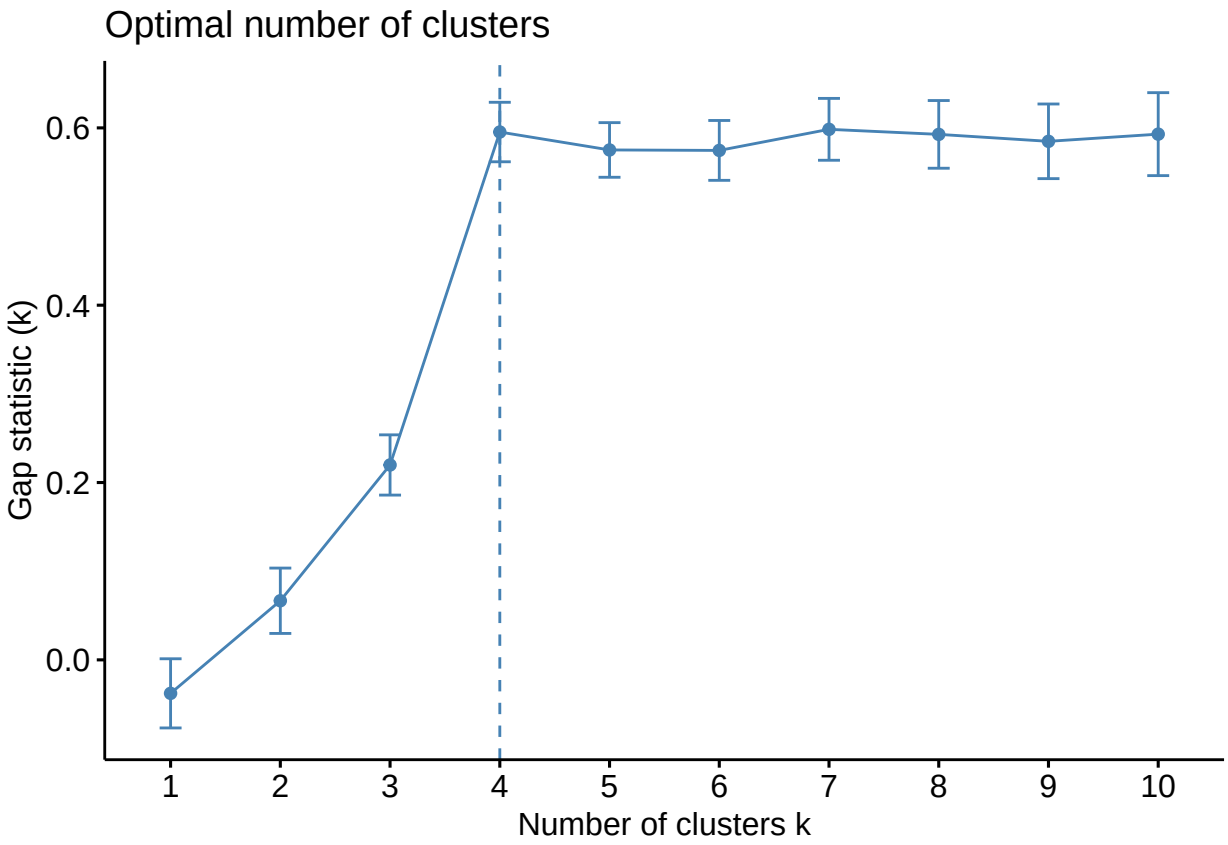
Por el método de promedio de siluetas, nos indica que 5 son los estimados.

```
fviz_nbclust(datos, kmeans, method = "silhouette")
```



Y finalmente, con el método estadístico Gap, también nos indica que hay 4 grupos:

```
set.seed(1)
gap_stat <- clusGap(datos, FUN = kmeans, nstart = 25,
                    K.max = 10, B = 50)
fviz_gap_stat(gap_stat)
```



Por lo que decidiremos que 4 serán el número de clusters que tomaremos para nuestro análisis final.

Veamos cómo se ven agrupados nuestros datos por medio de K-Medias.

```
set.seed(1)
kmeans4 <- kmeans(datos, 4, nstart = 25)
fviz_cluster(kmeans4, datos)
```

Jerárquico (Aglomerativo)

Analicemos con las diferentes ligas cuál de estas nos da un mejor coeficiente de aglomeración.

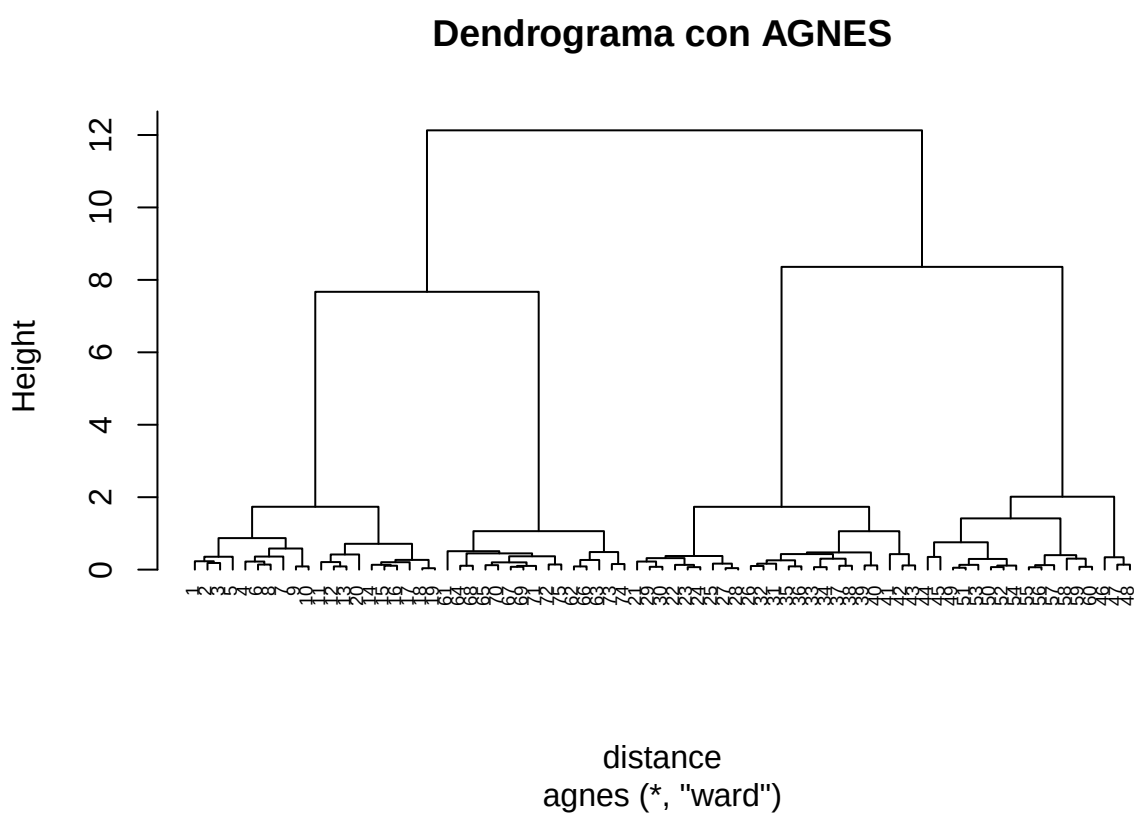
```
m <- c( "average", "single", "complete", "ward")
names(m) <- c( "average", "single", "complete", "ward")

ac <- function(x) {
  agnes(datos, method = x)$ac
}
library(purrr)
map_dbl(m, ac)
```

```
##   average    single  complete      ward
## 0.9420112 0.8982060 0.9624500 0.9880271
```

Como podemos ver, el método Ward nos dio estructuras aglomerativas más fuertes, pues su coeficiente es el más cercano a 1.

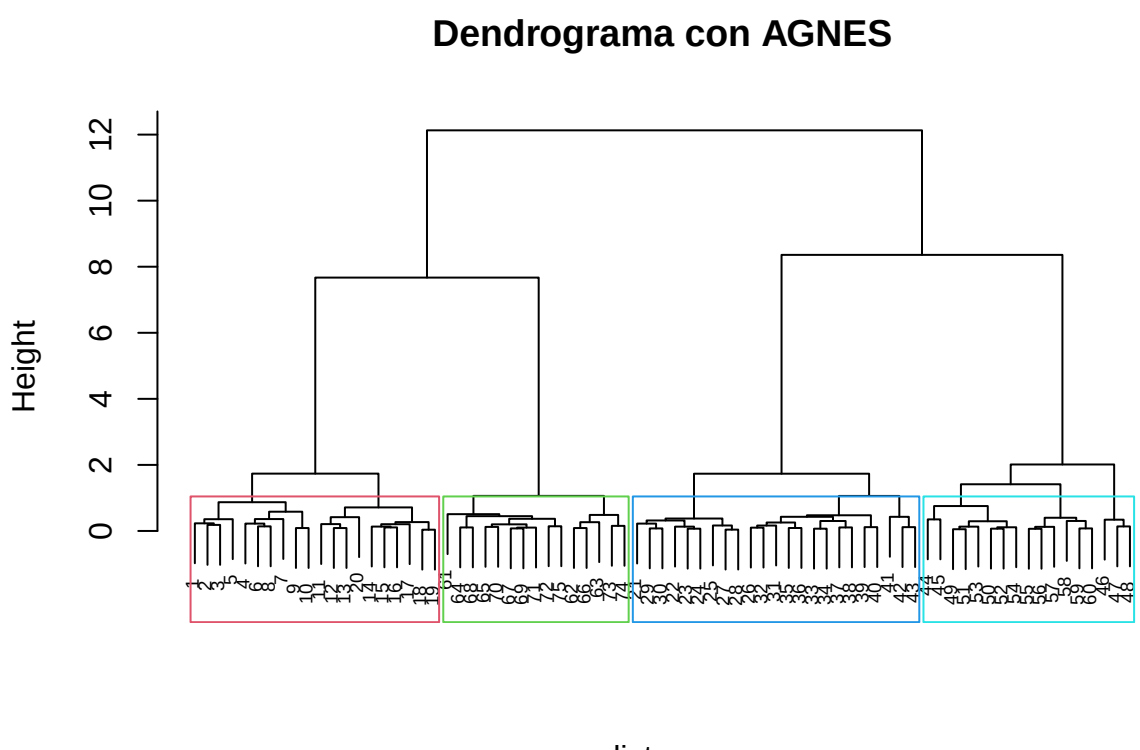
Hagamos el modelo con éste método y analicemos su dendograma correspondiente.

```
agnes <- agnes(distance, method = "ward")
pltree(agnes, cex = 0.6, hang = -1, main = "Dendrograma con AGNES")
```



El dendograma nos está gritando que lo cortemos en 4. Cortémoslo así y veamos los resultados en una gráfica como en K-Medias.

```
plot(agnes, cex = 0.6, which.plots = 2, main = "Dendrograma con AGNES")
rect.hclust(agnes, k = 4, border = 2:5)
```




```
clusters <- cutree(agnes, k = 4)
fviz_cluster(list(data = datos, cluster = clusters))
```

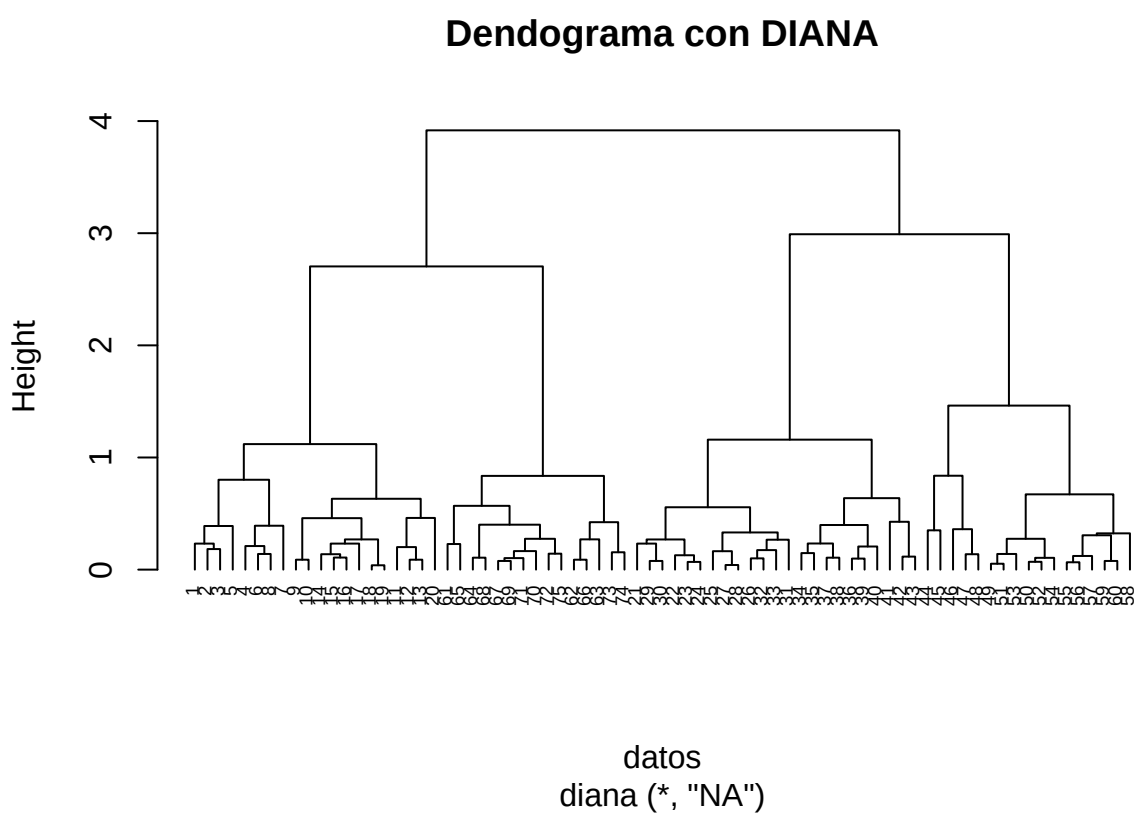
Jerárquico (Divisivo)

Ahora hagamos un modelo divisivo. Veamos su coeficiente divisivo y su respectivo dendograma.

```
diana <- diana(datos)
diana$dc
```

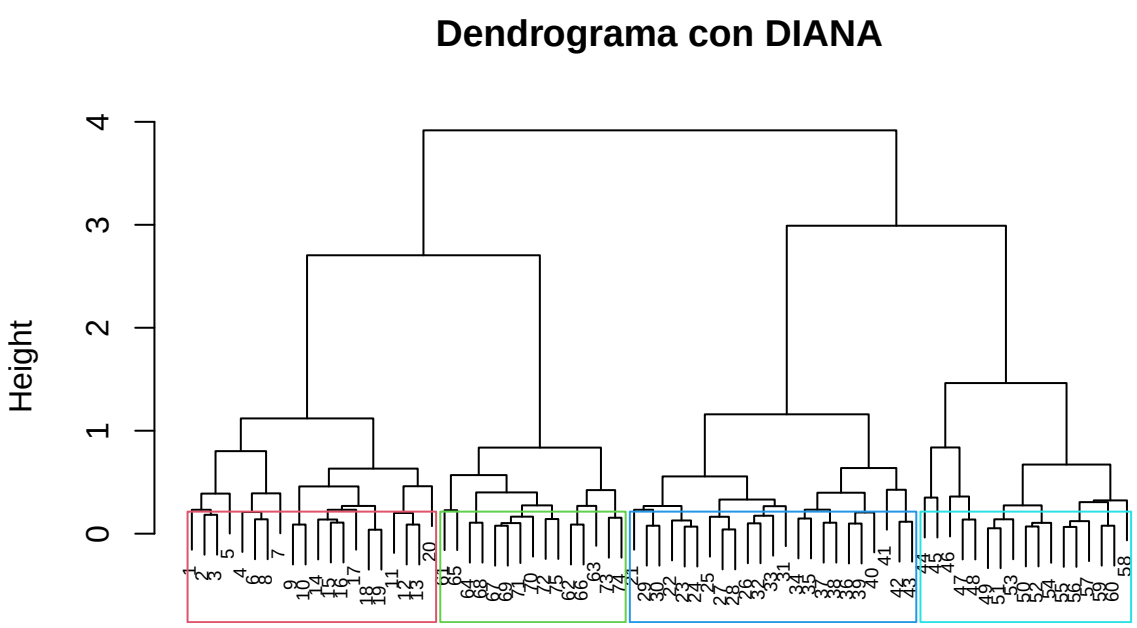
```
## [1] 0.9613788
```

```
pltree(diana, cex = 0.6, hang = -1, main = "Dendograma con DIANA")
```



Tenemos un buen coeficiente divisivo, indicio de buenas estructuras de clusters, y el dendograma nos vuelve a indicar que cortemos en 4. Hagámoslo y grafiquemos.

```
plot(diana, cex = 0.6, which.plots = 2, main = "Dendrograma con DIANA")
rect.hclust(diana, k = 4, border = 2:5)
```




```
clusters <- cutree(diana, k = 4)
fviz_cluster(list(data = datos, cluster = clusters))
```

Clasificación Basada en Modelos

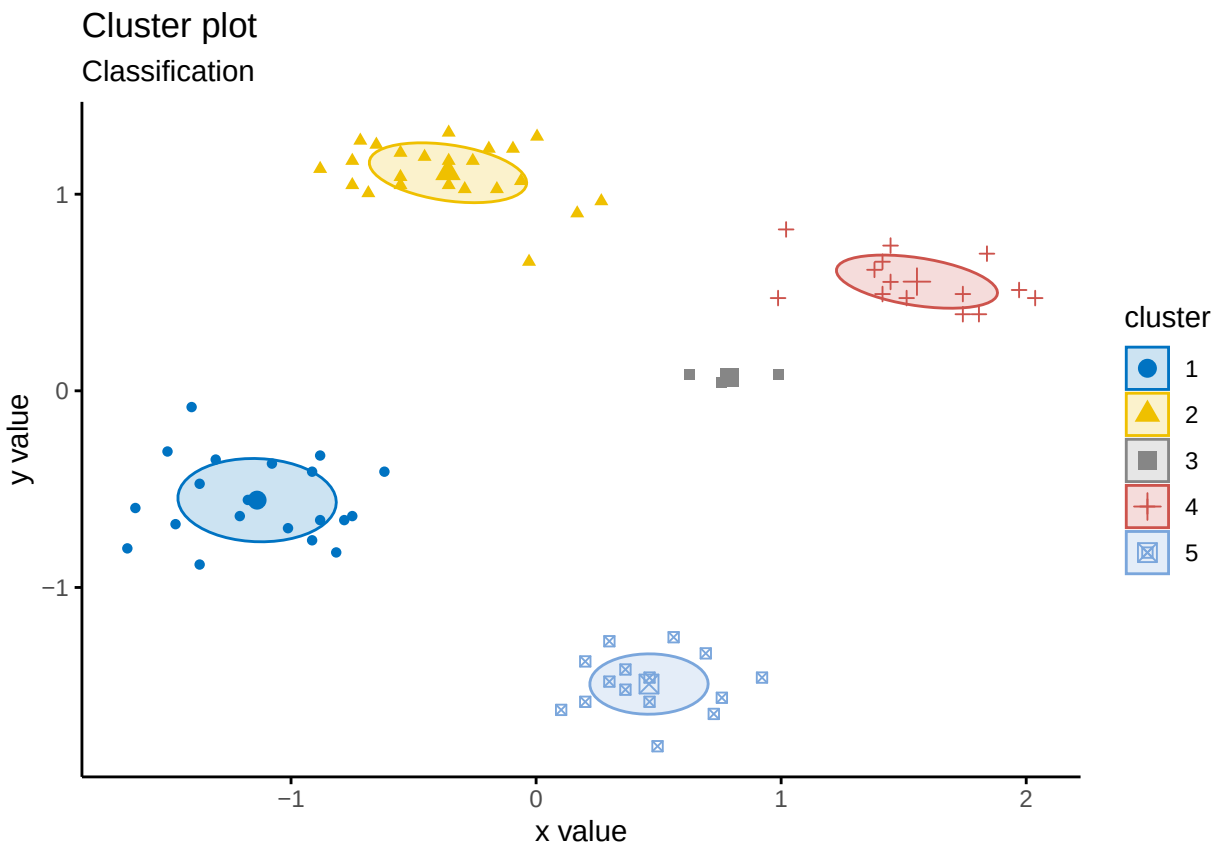
Hagamos un ajuste basado en modelos para nuestros datos escalados y veamos un poco la salida.

```
mc <- Mclust(datos)
summary(mc)
```

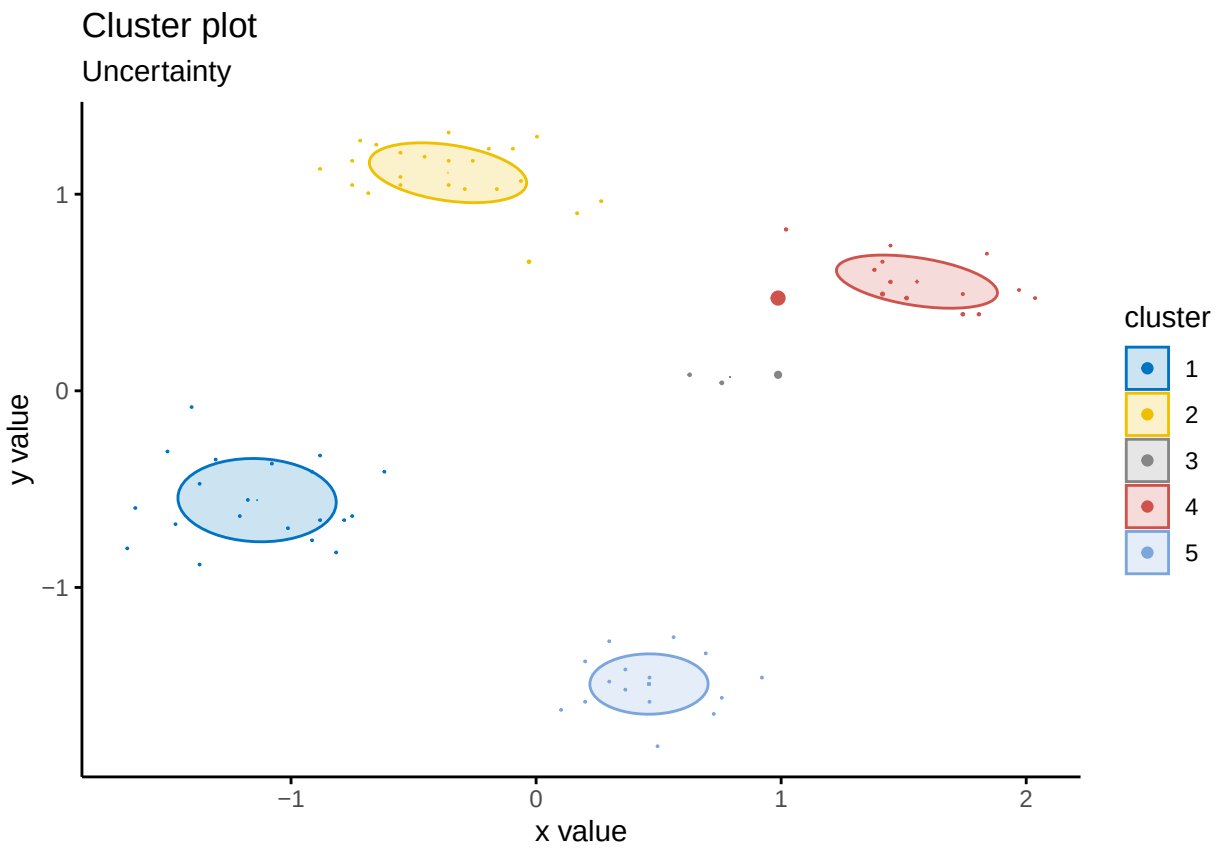
```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust EEI (diagonal, equal volume and shape) model with 5 components:
##
##  log-likelihood  n df       BIC       ICL
##       -91.26485 75 16 -251.6095 -251.7486
##
## Clustering table:
##  1  2  3  4  5
## 20 23  3 14 15
```

Podemos ver que nos arrojó un modelo orientado en la identidad y con formas y volúmenes iguales.
A diferencia de los ajustes anteriores, este basado en modelos nos indica que hay 5 clusters. El grupo número 3 tiene tres tuplas únicamente, sospechoso.
Grafiquemos los grupos para contrastarlos con los anteriores ajustes.

```
fviz_mclust(mc, "classification", geom = "point",
            pointsize = 1.5, palette = "jco")
```




```
fviz_mclust(mc, "uncertainty", palette = "jco")
```



En esta ocasión, uno de los grupos que teníamos fue seccionado en dos, los clusters 3 y 4 en estos basados en modelos. Es muy raro decir que el grupo 3 es un cluster, ya que este cluster sólo tiene tres elementos. Además que en la gráfica de incertidumbre, hay bastante confusión entre un rojo y un gris. Pareciera que para este método tampoco sería tan descaballo pensar en 4 clusters.

Conclusiones

A pesar de que el método basado en modelos y la gráfica de siluetas nos parecían indicar que la opción sería tomar 5 grupos, todos los demás modelos, tanto el aglomerativo como el divisivo, el método del codo, el estadístico Gap, y nuestra intuición nos indican que la mejor opción es concluir que hay 4 grupos.

Las gráficas de todos los métodos que concluyeron que hay 4 grupos son prácticamente idénticas, así que no hay mucha confusión de en qué grupo meter a las observaciones.

*Cabe recalcar que estos análisis fueron hechos con la métrica euclidiana. Podemos realizar el proceso con diferentes métricas si no obtuviéramos un resultado convincente. Sin embargo creemos que las conclusiones son contundentes y no es necesario utilizar otra métrica.

Ejercicio 4

Comenzamos nuevamente cargando las librerías y los datos necesarios, así como renombrando las variables del dataframe.

```
library(tidyverse)
library(caret)
library(MASS)
library(pROC)
```

```r
datos<-read.table("wais.txt")

datos<-datos[,c(1,3,4,5,6)] #Ignoramos la segunda columna porque funciona sólo como ID
colnames(datos)<-c("grupo", "information", "similarities", "arithmetic","p. completion")

datos$grupo<-as.factor(datos$grupo)
levels(datos$grupo)<-c("gpo 1", "gpo 2")
```

Ahora dividiremos los datos en una proporción 70-30 para el training-test.

```r
set.seed(100)
training.samples <- datos$grupo %>%
  createDataPartition(p = 0.7, list = FALSE)
train.data <- datos[training.samples, ]
test.data <- datos[-training.samples, ]
```

También normalizamos los datos (las variables categóricas se ingoran automáticamente) y transformamos los datos para la creación del modelo.

```r
preproc.param <- train.data %>%
  preProcess(method = c("center", "scale"))

train.transformed <- preproc.param %>% predict(train.data)
test.transformed <- preproc.param %>% predict(test.data)
```

Ahora pasemos al ajuste de modelos, tanto lineal como cuadrático.

Discriminante Lineal

Ajustemos el modelo y analicemos un poco la salida.

```r
lda <- lda(grupo~., data = train.transformed)
lda
```

```
## Call:
## lda(grupo ~ ., data = train.transformed)
##
## Prior probabilities of groups:
##     gpo 1     gpo 2
## 0.7428571 0.2571429
##
## Group means:
##       information similarities arithmetic `p. completion`
## gpo 1   0.2648016    0.2697613  0.2334984       0.2863384
## gpo 2  -0.7649823   -0.7793104 -0.6745510      -0.8271997
##
## Coefficients of linear discriminants:
##                          LD1
## information     -0.02094728
## similarities    -0.49168453
## arithmetic      -0.24782160
## `p. completion` -0.67797235
```

Como podemos ver hay mucha más probabilidad de pertenecer al grupo 1. Quizás una pista de esto es debido a que de entrada nosotros sabíamos que había más individuos del grupo 1 sumado a que los coeficientes del discriminante lineal son negativos en su totalidad.

No es obligatorio esto último (pues transformamos los datos), pero al hacer combinaciones lineales con las muestras de prueba es más probable que estas sean menores a 1.

Desafortunadamente para nuestro modelo las distribuciones de las clases no se ven muy normal que digamos, como podemos ver a continuación.

Sin embargo veamos los pronósticos del modelo y veamos qué tal lo hizo con el grupo de prueba, así como la precisión general del modelo.

```
predictions <- predict(lda, test.transformed)
table(predictions$class, test.transformed[,1])
```

```
##
##         gpo 1 gpo 2
##   gpo 1    11     1
##   gpo 2     0     2
```

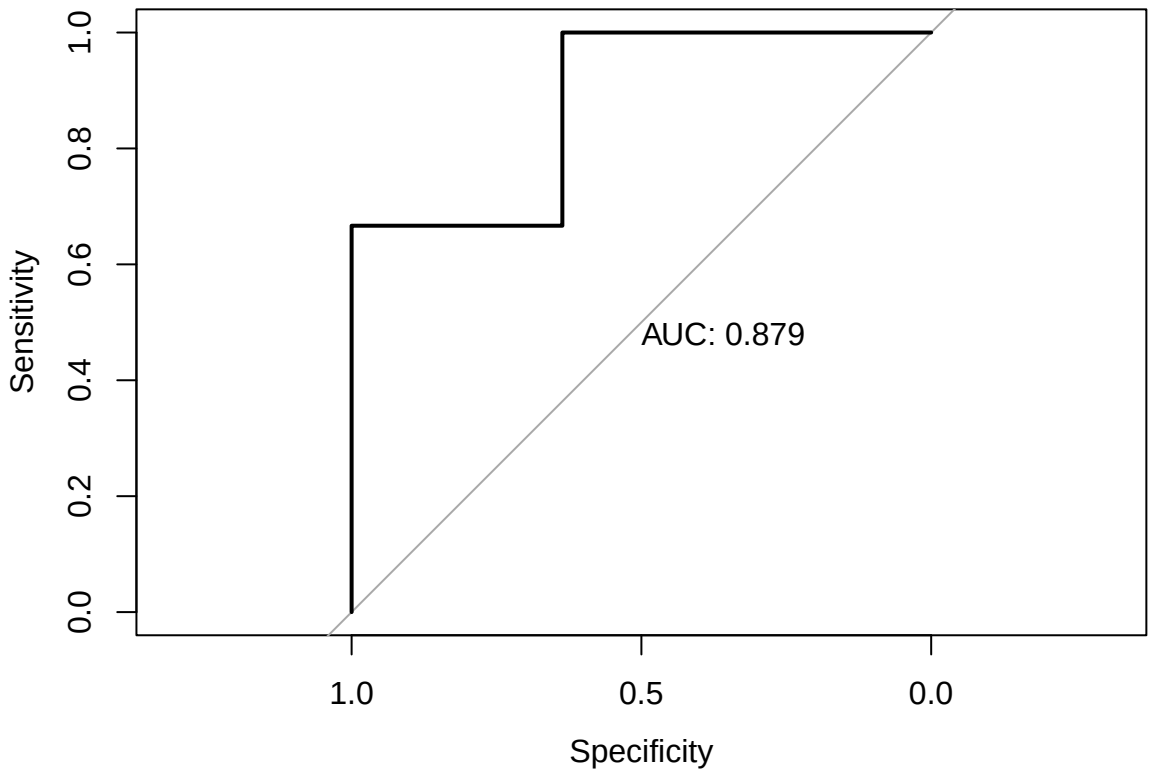
```
mean(predictions$class==test.transformed$grupo)
```

```
## [1] 0.9285714
```

Tuvimos un único error con este modelo y una precisión del 93%, por lo que podemos decir que estamos ante un buen modelo.

Por último grafiquemos también su curva ROC.

```
res.roc <- roc(test.data$grupo, predictions$posterior[,1])
plot.roc(res.roc, print.auc = TRUE)
```



También tenemos una buena curva ROC. Consideramos que este es un buen modelo.

Discriminante Cuadrático

Ajustemos el modelo y veamos un poco la salida

```
qda <- qda(grupo~., data = train.transformed)
qda
```

```
## Call:
## qda(grupo ~ ., data = train.transformed)
##
## Prior probabilities of groups:
##     gpo 1     gpo 2
## 0.7428571 0.2571429
##
## Group means:
##       information similarities arithmetic `p. completion`
## gpo 1   0.2648016    0.2697613  0.2334984       0.2863384
## gpo 2  -0.7649823   -0.7793104 -0.6745510      -0.8271997
```

Tenemos probablidades idénticas al del modelo LDA anterior. Pero veamos su desempeño prediciendo.

```
predictions <- predict(qda, test.transformed)
table(predictions$class, test.transformed[,1])
```

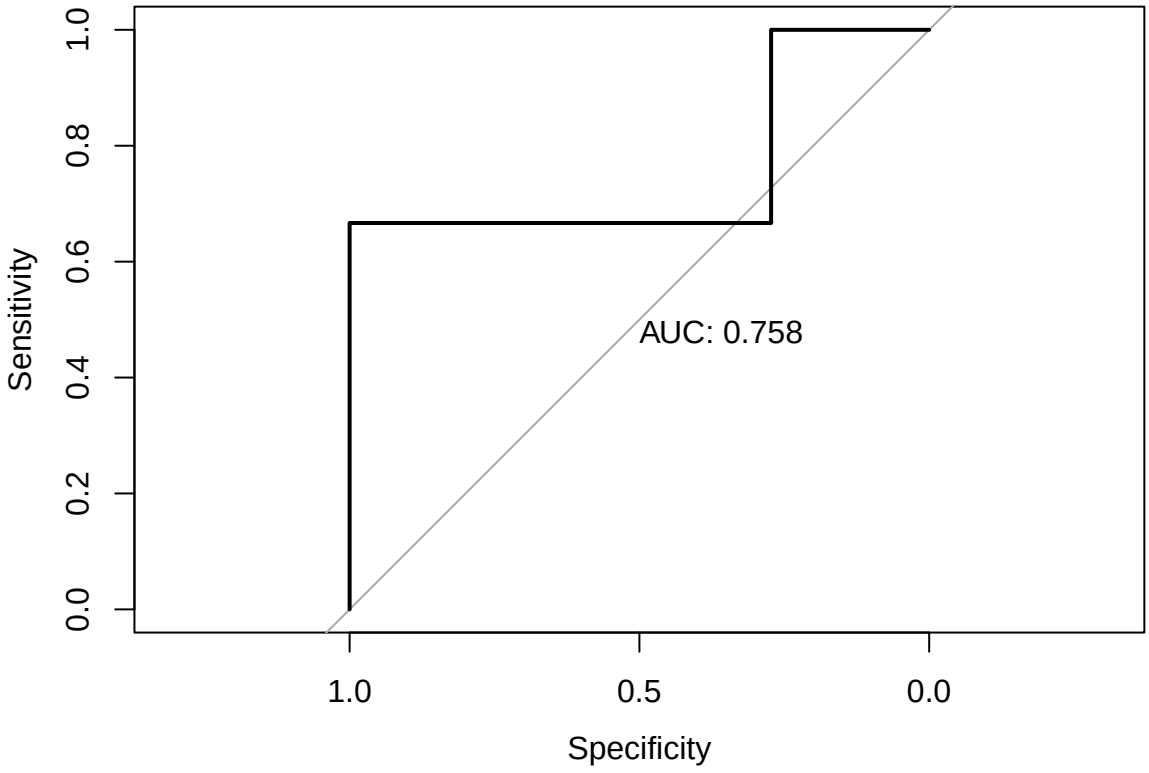
```
##
##         gpo 1 gpo 2
##   gpo 1     8     1
##   gpo 2     3     2
```

```
mean(predictions$class == test.transformed$grupo)
```

```
## [1] 0.7142857
```

En esta ocasión, el modelo cuadrático lo hizo muy mal en comparación del lineal, por lo que optamos por quedarnos con el lineal para este caso. (Graficamos por último su curva ROC, que también es bastante mala en comparación con el anterior AUC)

```
res.roc <- roc(test.data$grupo, predictions$posterior[,1])
plot.roc(res.roc, print.auc = TRUE)
```


Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
AUC: 0.758
1.0
0.5
0.0
Specificity