Clustering jerárquico: estudio y aplicaciones

Francesc López - Marc Munar - Mateu Morro

14 de enero de 2019

Introducción

Se parte del objetivo fundamental de estudiar diferentes algoritmos de clustering. Así, tal y como se ha visto en clase, el algoritmo usual en el análisis de datos es el de k-means, teniendo este ciertas limitaciones ya que depende de la configuración inicial de los parámetros. Entonces, es en este punto donde se propone la agrupación jerárquica, y su representación gráfica por dendogramas, que resulta una alternativa al clustering para agrupar objetos en función de su similitud. A diferencia de la agrupación en clusters, la agrupación jerárquica no requiere que se especifique previamente el número de agrupaciones que se producirán. Hay dos tipos:

- Agrupación aglomerada (AGNES): Cada observación se considera inicialmente como un grupo propio, de esta manera, los grupos más similares se fusionan sucesivamente hasta que haya un solo cluster grande.
- Agrupación por división (DIANA): Es la inversa a la agrupación aglomerada. Todos los objetos están incluidos en un grupo y éste se va dividiendo sucesivamente hasta que todas las observaciones están en su propio grupo.

Debido a la generalidad que representa la agrupación de diferentes objetos en su similitud, se debe recurrir a las diferentes métricas matemáticas (pero que en principio solo se usará la euclidiana), diferentes métodos de agrupamiento y finalmente realizar la comparación entre los diferentes métodos.

Una vez realizado el estudio de los agrupamientos, se deberá realizar la visualización de esos resultados. Se ha optado, por lo tanto, por los paquetes `ggplot`, `factoextra` y `cluster`. Cabe destacar que el paquete `factoextra` realiza los gráficos usando la librería `ggplot`.

Finalmente, y para poder ejemplificar los métodos que se vayan estudiando, se utilizarán los datos `USArrests` de R, que representan los tantos por 100000 habitantes de diferentes estados de Estados Unidos y diferentes crímenes en el año 1973.

Agrupación algomerada

El agrupamiento aglomerado funciona de forma ascendente, es decir, inicialmente cada objeto es considerado como un grupo de un solo elemento (hoja). En cada paso del algoritmo, los dos grupos que son los más similares se combinan en un nuevo grupo más grande (nodos). Este procedimiento se repite de forma recursiva hasta que todos los puntos son miembros de un solo gran tamaño (raíz).

El inverso de la agrupación aglomerativa es la agrupación por división y funciona de manera descendente. Comienza con la raíz, en la que todos los objetos se incluyen en un solo cluster. En cada paso de la iteración, el cluster más heterogéneo se divide en dos iterándose el proceso hasta que todos los objetos están en su propio grupo. A continuación, se exponen los pasos a seguir.

Estructura y preparación de los datos

Como se ha comentado, se realiza el estudio sobre el dataset `USArrests`. Los datos vienen estructurados de la siguiente manera:

- Las filas contienen las observaciones de los diferentes estados, habiendo en total 50 estados diferentes.
- Las columnas representan las siguientes variables:
  - `murder`: Proporción de asesinatos (por 100000).
  - `assault`: Proporción de asaltos (por 100000).
  - `rape`: Proporción de violaciones (por 100000).
  - `urbanPop`: Proporción de la población que vive en zona urbana.

Después de cargar los datos, se estandarizan las variables para que sean comparables.

```
data("USArrests")
df <- scale(USArrests)
head(df,nrow=6)
```

```
##                Murder   Assault   UrbanPop         Rape
## Alabama    1.24256408 0.7828393 -0.5209066 -0.003416473
## Alaska     0.50786248 1.1068225 -1.2117642  2.484202941
## Arizona    0.07163341 1.4788032  0.9989801  1.042878388
## Arkansas   0.23234938 0.2308680 -1.0735927 -0.184916602
## California 0.27826823 1.2628144  1.7589234  2.067820292
## Colorado   0.02571456 0.3988593  0.8608085  1.864967207
```

Análisis de la similitud

Para poder calcular la semejanza entre las observaciones, existen diversos métodos para calcular la (des)similitud de la información, siendo por defecto la distancia euclidiana la que se calcula en la función `dist()`. No obstante, es posible cambiar la métrica cambiando el parámetro `method`.

Se guarda la matriz de distancias en la variable `res.dist`.

```
res.dist <- dist(df, method = "euclidean")
```

Convertimos la matriz de distancias, que por defecto es un objeto propio de R, en una matriz usando la función `as.matrix()`. Dicha matriz contiene, en la fila i y columna j la distancia entre el objeto i-ésimo y el j-ésimo. Se muestra a continuación una previsualización.

```
as.matrix(res.dist)[1:4, 1:4]
```

```
##           Alabama   Alaska  Arizona Arkansas
## Alabama  0.000000 2.703754 2.293520 1.289810
## Alaska   2.703754 0.000000 2.700643 2.826039
## Arizona  2.293520 2.700643 0.000000 2.717758
## Arkansas 1.289810 2.826039 2.717758 0.000000
```

Cálculo de los clusters

Una vez se ha calculado la matriz de distancias, es posible agrupar los objetos en función de su semejanza. Dicho proceso se realiza por pares de grupos, creando en cada iteración grupos más grandes, y es iterado hasta que todos los objetos en el conjunto de datos original estén vinculados en una jerarquía de árbol. La función de R `hclust()` se puede usar para crear el árbol jerárquico.

```
res.hc <- hclust (d = res.dist, method = "average")
```

A continuación se explican los parámetros de la función usada:

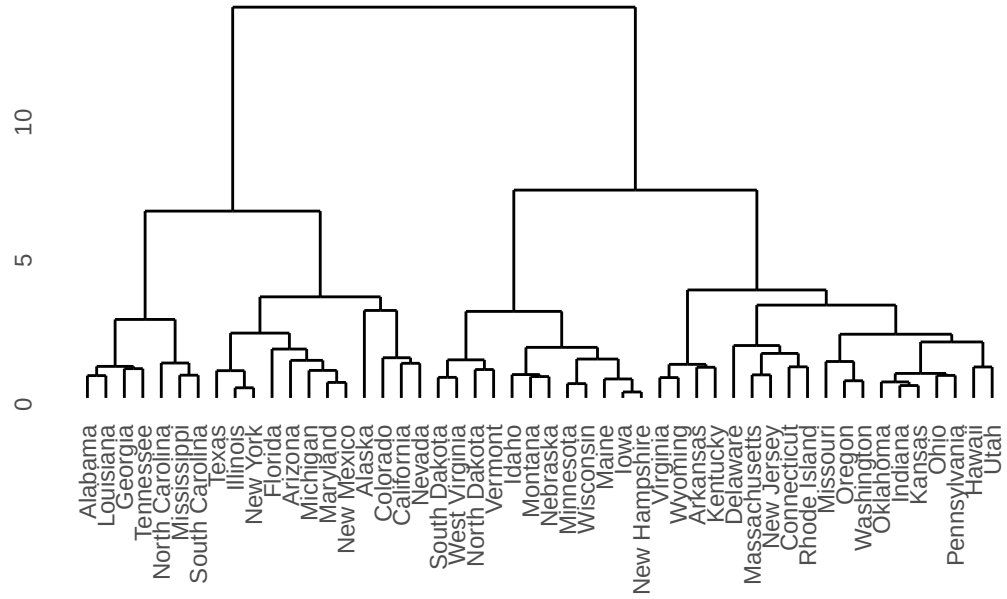


Figura 1: Dendograma del árbol jerárquico

- `d`: una estructura de disimilitud producida por la función `dist()`.
- `method`: el método de aglomeración (enlace) que se utiliza para calcular la distancia entre grupos. Los valores permitidos son uno de `ward.D`, `ward.D2`, `single`, `complete`, `average`, `mcquitty`, `median` o `centroid`.
    - `complete`: La distancia entre dos grupos se define como el valor máximo de todas las distancias por pares. Tiende a producir clusters más compactos.
    - `single`: La distancia entre dos grupos es el valor mínimo de todas las distancias por pares. Tiende a producir grupos largos y "sueltos".
    - `average`: La distancia entre los dos grupos es la distancia promedio entre los elementos del grupo 1 y del grupo 2.
    - `centroid`: La distancia entre dos grupos es la distancia entre el centroide del grupo 1 y el centroide del grupo 2.
    - `ward`: Minimiza las variaciones totales dentro del grupo. En cada paso, el par de grupos con una distancia mínima entre grupos se fusionan.

La vinculación completa y el método de Ward son generalmente preferidos.

Verificación del método

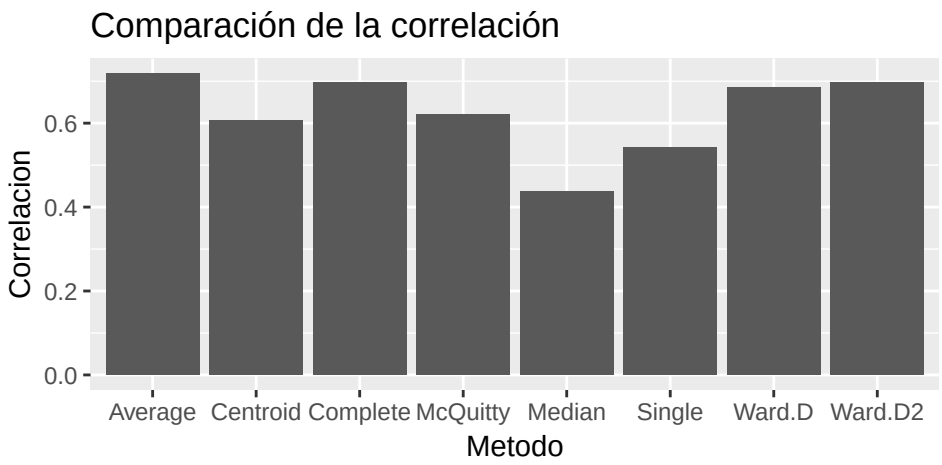
Para poder representar el árbol jerárquico generado por la función `hclust()` de la sección anterior, se usará la función `ggdendrogram()` de la librería `ggdendro`.

```
ggdendrogram(res.hc)
```

Cuanto mayor es la altura de la fusión, menos similares son los objetos. Esta altura se conoce como la distancia copenética entre los dos objetos. Las conclusiones sobre la proximidad de dos objetos solo se pueden extraer basándose en la altura donde se fusionan las ramas que contienen esos dos objetos. Entonces, no se puede usar la proximidad de dos objetos a lo largo del eje horizontal como criterio de su similitud.

Una cuestión natural que surge es la de comprobar si efectivamente la agrupación realizada es correcta o no. Después de vincular los objetos de un conjunto de datos en un árbol de clusters jerárquico, una manera para ver si las alturas en el árbol reflejan las distancias originales es calcular la correlación entre la matriz de distancias copenéticas (usando diferentes métodos) y la matriz de distancias originales, interpretándose la correlación de forma usual.

```
cor.average <- cor(res.dist, cophenetic(hclust(res.dist, method = "average")))
cor.ward_d <- cor(res.dist, cophenetic(hclust(res.dist, method = "ward.D")))
cor.ward_d2 <- cor(res.dist, cophenetic(hclust(res.dist, method = "ward.D2")))
cor.single <- cor(res.dist, cophenetic(hclust(res.dist, method = "single")))
cor.complete <- cor(res.dist, cophenetic(hclust(res.dist, method = "complete")))
cor.mcquitty <- cor(res.dist, cophenetic(hclust(res.dist, method = "mcquitty")))
cor.median <- cor(res.dist, cophenetic(hclust(res.dist, method = "median")))
cor.centroid <- cor(res.dist, cophenetic(hclust(res.dist, method = "centroid")))

correlation_data <- data.frame(method = c("Average", "Ward.D", "Ward.D2", "Single",
                                          "Complete", "McQuitty", "Median", "Centroid"),
                               correlation = c(cor.average, cor.ward_d, cor.ward_d2,
                                               cor.single, cor.complete, cor.mcquitty,
                                               cor.median, cor.centroid))

ggplot(data=correlation_data, aes(x=method, y=correlation)) +
  geom_bar(stat="identity")+
  labs(title = "Comparación de la correlación")+
  ylab("Correlacion")+xlab("Metodo")
```



Por consiguiente, se considera el método `average` como el que mejor agrupa los datos. Para poder identificar subgrupos, se debe cortar el dendrograma a una cierta altura.

Segmentación del dendograma en diferentes grupos

Tal y como puede observarse en la figura 1, no puede determinarse a priori cuantos clusters se han realizado. Esto es, uno de los problemas de la agrupación jerárquica es que en función de la altura existen diferentes grupos. El proceso a realizar es cortar el dendograma para poder formar grupos, y se usará la función `cutree()` para cortar el árbol jerárquico a una altura determinada y dividir datos en grupos. A continuación, se consideran cuatro grupos para realizar el corte:

```
grp <- cutree(res.hc, k = 4 )
```

Para poder mirar cuantos estados tiene cada cluster, se usa la instrucción siguiente:

```
table(grp)
```

```
## grp
##  1  2  3  4
##  7  1 12 30
```

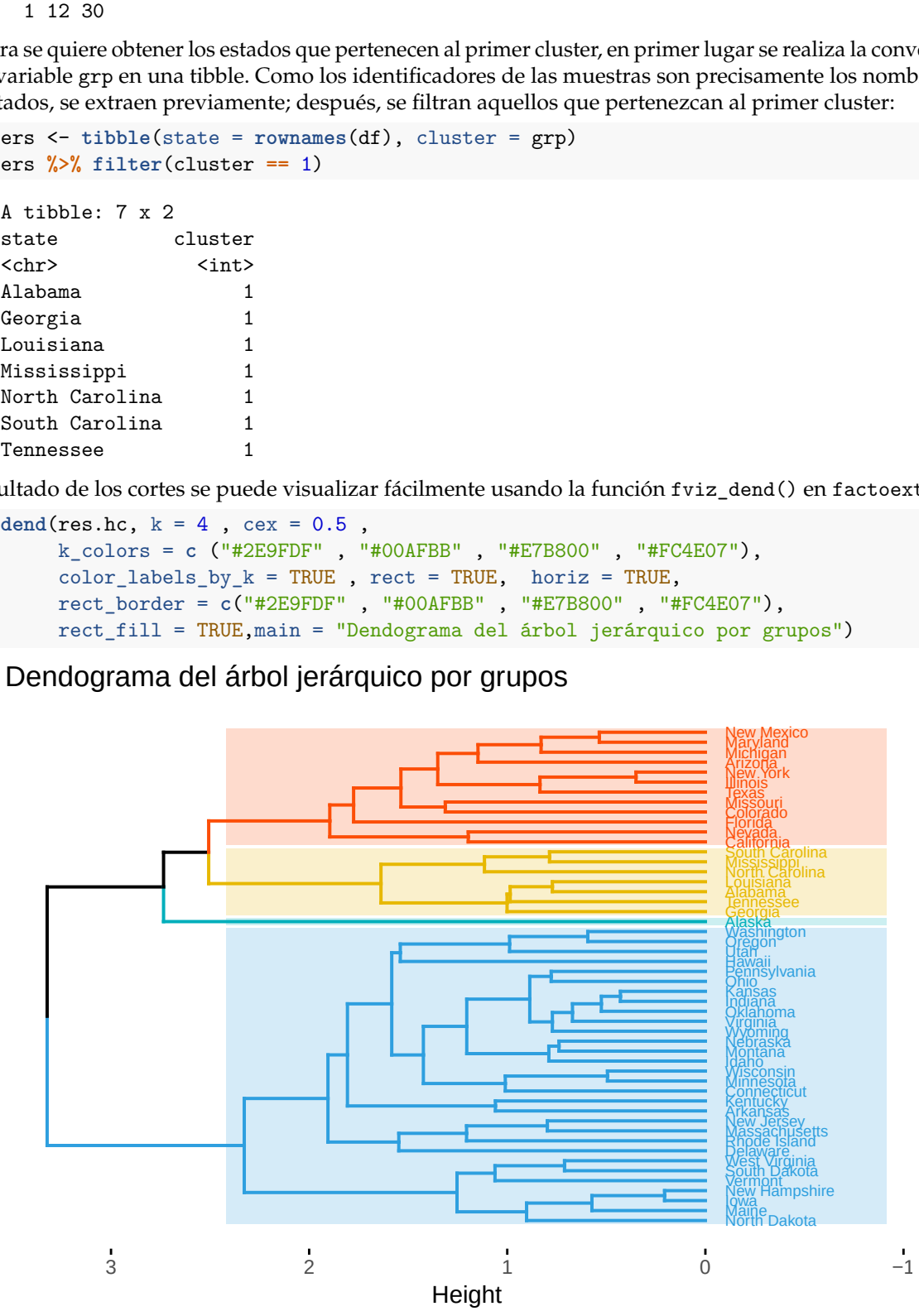
Si ahora se quiere obtener los estados que pertenecen al primer cluster, en primer lugar se realiza la conversión de la variable `grp` en una tibble. Como los identificadores de las muestras son precisamente los nombres de los estados, se extraen previamente; después, se filtran aquellos que pertenezcan al primer cluster:

```
clusters <- tibble(state = rownames(df), cluster = grp)
clusters %>% filter(cluster == 1)
```

```
## # A tibble: 7 x 2
##   state          cluster
##   <chr>            <int>
## 1 Alabama              1
## 2 Georgia              1
## 3 Louisiana            1
## 4 Mississippi          1
## 5 North Carolina       1
## 6 South Carolina       1
## 7 Tennessee            1
```

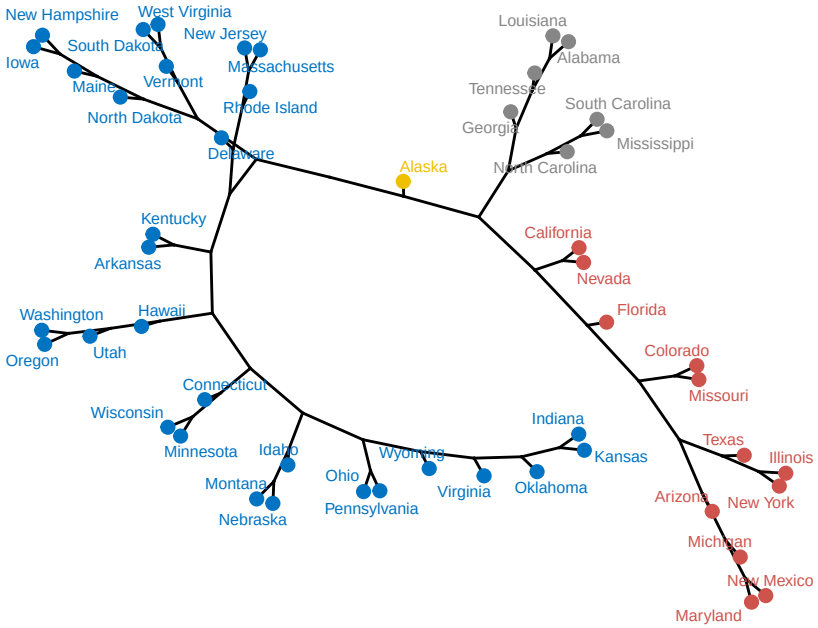
El resultado de los cortes se puede visualizar fácilmente usando la función `fviz_dend()` en `factoextra`:

```
fviz_dend(res.hc, k = 4 , cex = 0.5 ,
          k_colors = c ("#2E9FDF" , "#00AFBB" , "#E7B800" , "#FC4E07"),
          color_labels_by_k = TRUE , rect = TRUE,  horiz = TRUE,
          rect_border = c("#2E9FDF" , "#00AFBB" , "#E7B800" , "#FC4E07"),
          rect_fill = TRUE,main = "Dendograma del árbol jerárquico por grupos")
```



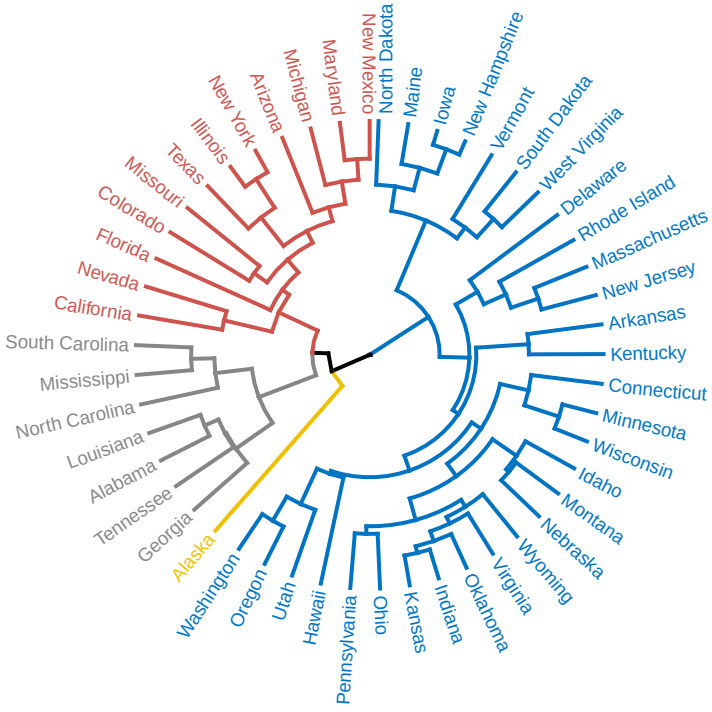
Se puede corroborar gráficamente como los estados del primer cluster se corresponden con el resultado del filtrado anterior. Además, el gráfico que se acaba de mostrar presenta diversas maneras de mostrarse: sin color de fondo, transpuesto, circular, en forma ramificada, etc. Por simplicidad, se mostrarán a continuación los dos últimos. Así, para ver en forma ramificada (llamado también árbol filogenético):

```
fviz_dend(res.hc, k = 4, k_colors = "jco", cex = 0.5,
type = "phylogenic", repel = TRUE)
```



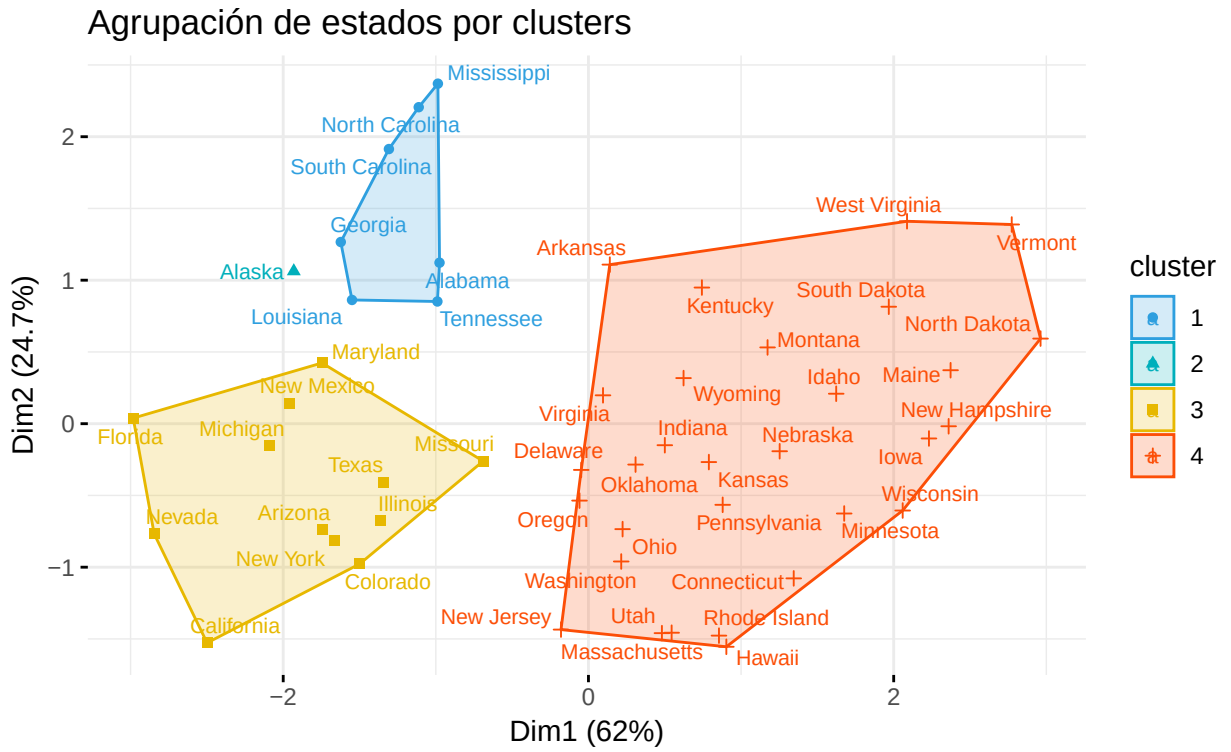
Y después la visualización circular:

```
fviz_dend(res.hc, cex = 0.5, k = 4,
k_colors = "jco", type = "circular")
```



La misma información también se puede visualizar mediante un diagrama de dispersión, donde se muestran los estados y los diferentes clusters a los que pertenecen:

```
fviz_cluster(list(data = df, cluster = grp),
  palette = c("#2E9FDF", "#00AFBB", "#E7B800", "#FC4E07"),
  ellipse.type = "convex", labelsize = 8,
  repel = TRUE,
  show.clust.cent = FALSE, ggtheme = theme_minimal(),
  main = "Agrupación de estados por clusters")
```

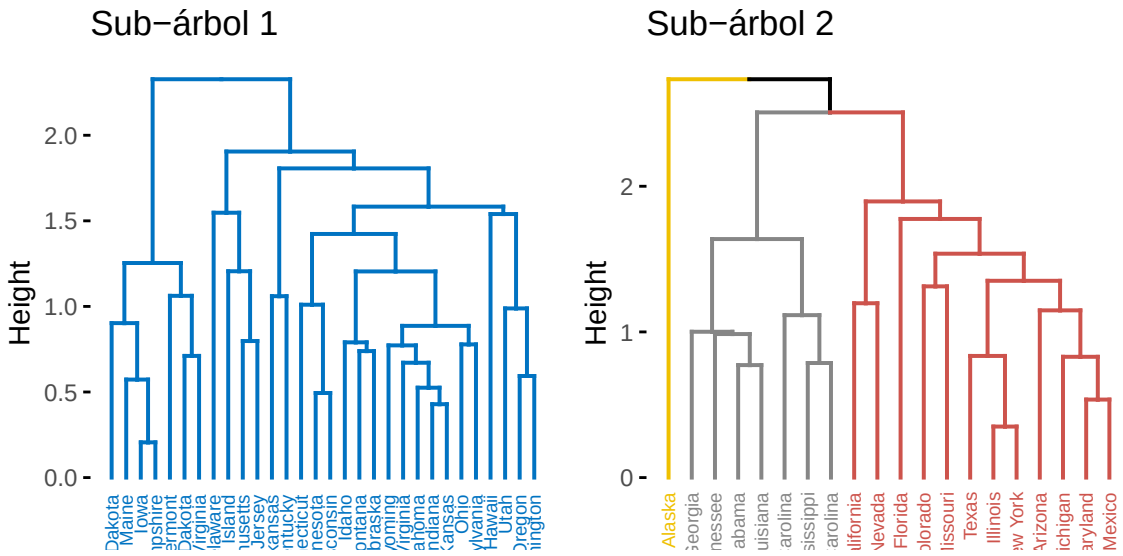


También se encuentra la opción de mostrar el dendograma en sub-árboles. En primer lugar, se declara la estructura que va a permitir, posteriormente, obtener los dos sub-árboles.

```
dend_plot <- fviz_dend(res.hc, k = 4,
  cex = 0.5,
  k_colors = "jco"
)
dend_data <- attr(dend_plot, "dendrogram")
dend_cuts <- cut(dend_data, h = 10)
```

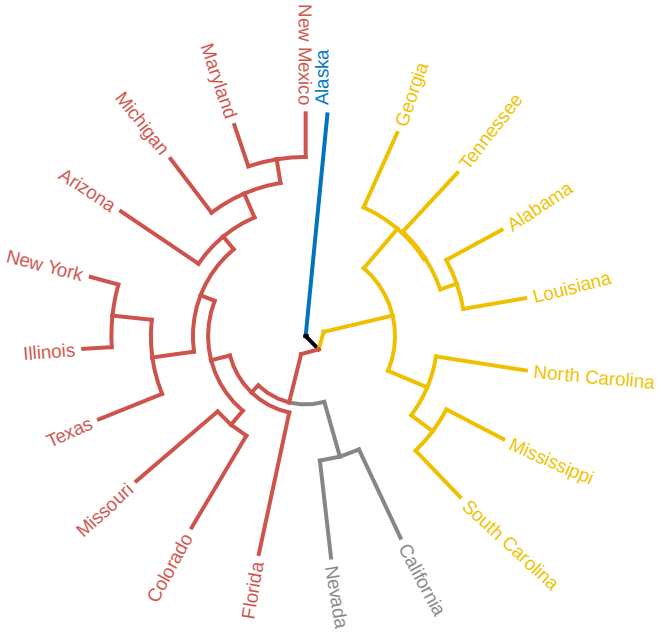
Ahora, a partir de lo calculado, se extraen las dos ramas:

```
p1 <- fviz_dend(dend_cuts$lower[[1]], main = "Sub-árbol 1")
p2 <- fviz_dend(dend_cuts$lower[[2]], main = "Sub-árbol 2")

multiplot(p1, p2, cols=2)
```



De forma análoga a lo calculado anteriormente, se puede volver a realizar el gráfico circular de los datos partiendo de un sub-árbol. Por ejemplo, si se realiza del sub-árbol derecho:

```
fviz_dend(dend_cuts$lower[[2]], cex = 0.5, k = 4,
  k_colors = "jco", type = "circular")
```

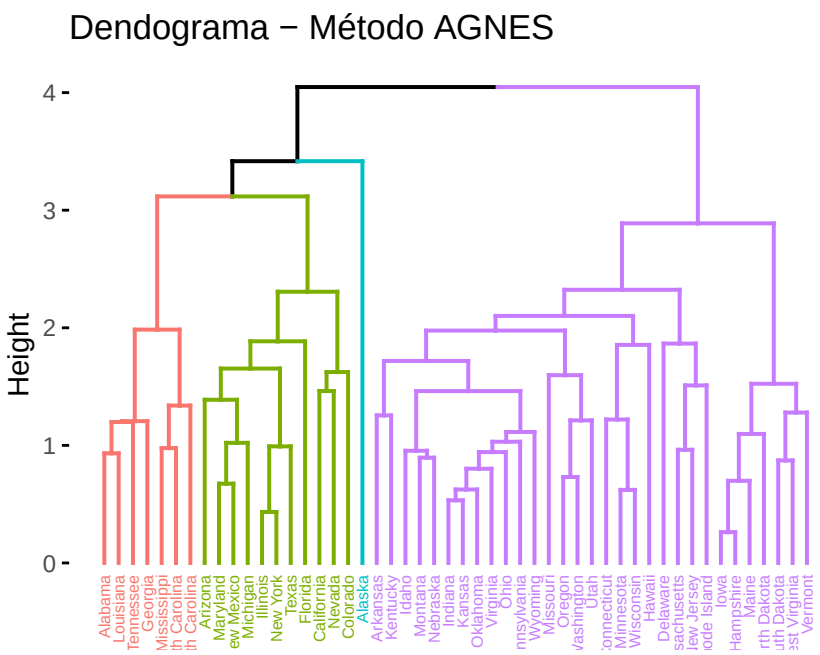


Automatización del clustering en R

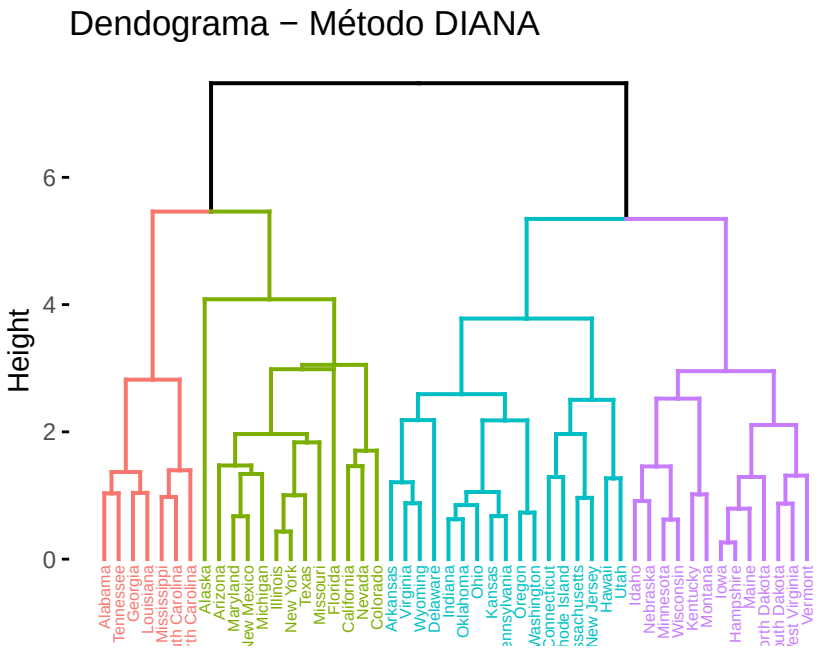
El paquete `cluster` de R proporciona las funciones `agnes()` y `diana()` para calcular el agrupamiento aglomerado y divisivo. De esta manera, no hace falta utilizar las funciones que se han ido usando, como `scale()`, `dist()` y `hclust()`.

```
res.agnes <- agnes(x = USArrests, # data matrix
stand = TRUE,
metric = "euclidean",
method = "average")
```

```
res.diana <- diana(x = USArrests,
stand = TRUE,
metric = "euclidean" )

fviz_dend(res.agnes, cex = 0.4, k = 4, main = "Dendograma - Método AGNES")
```




```
fviz_dend(res.diana, cex = 0.4, k = 4, main = "Dendograma - Método DIANA")
```



Aplicación del clustering jerárquico en genética

Una de las aplicaciones del clustering jerárquico es en genética, para saber si hay genes que tienen patrones comunes. Aunque haya diferentes distancias para poder realizar el clustering, dentro del estudio de la genética se destaca la distancia dada por la transformación del logaritmo en base 2, la distancia por correlación y agrupación de enlaces completos.

Además, la correlación de Pearson es bastante sensible a los valores atípicos. Por lo tanto, una opción alternativa es utilizar la correlación de Spearman.

Comparación de dendogramas

En la sección anterior se han estudiado diferentes métodos para realizar la agrupación. En concreto, se ha usado el método `average` para poder realizar el clustering. Ahora bien, surge la cuestión de poder determinar si dos dendogramas, obtenidos de diferentes métodos y la misma métrica, han realizado agrupaciones semejantes. En caso de no ser semejantes, será interesante obtener un coeficiente que mida el grado de similitud de los dendogramas.

Para que la representación gráfica sea más visual y legible, del conjunto de datos `df` se extraerá un subconjunto de 10 muestras aleatorias.

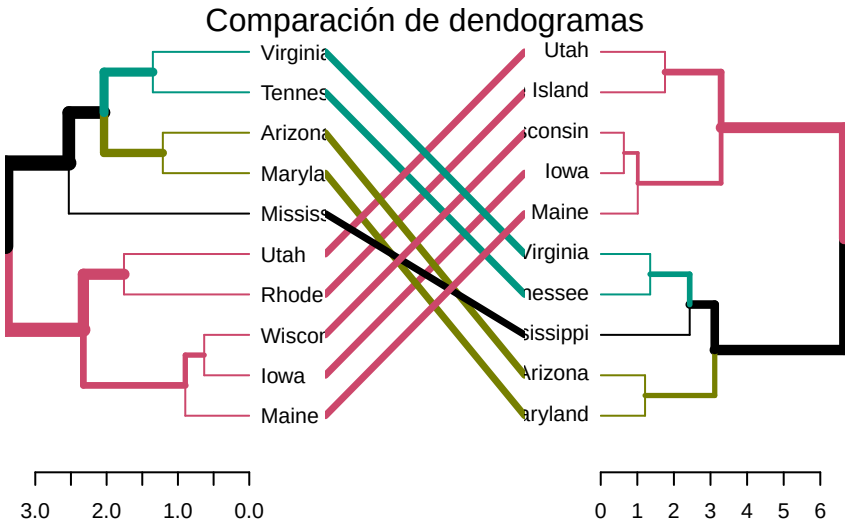
```
set.seed (123)
subset <- df [sample(1:50, 10),]
```

A partir del subconjunto calculado, se calculan las agrupaciones usando los métodos `average` y `ward.D2`, ambos partiendo de la matriz de distancias con la métrica euclidiana. Finalmente, se crean los dos dendogramas y se guardan en una lista que los contenga:

```
res.dist <- dist(subset, method = "euclidean")

hc1 <- hclust(res.dist, method = "average")
hc2 <- hclust(res.dist, method = "ward.D2")

dend1 <- as.dendrogram (hc1)
dend2 <- as.dendrogram (hc2)

dend_list <- dendlist(dend1, dend2)
```

Para comparar visualmente dos dendrogramas, usaremos la función `tanglegram()` (del paquete dendextend), que traza los dos dendrogramas, lado a lado, con sus etiquetas conectadas por líneas. A continuación, se realiza la visualización:

```
tanglegram(dend1, dend2,
highlight_distinct_edges = FALSE,
common_subtrees_color_lines = TRUE,
common_subtrees_color_branches = TRUE,
main = "Comparación de dendogramas", cex_main = 1)
```



Como se ha comentado, se desea obtener un coeficiente que mida la calidad de la alineación de ambos árboles. Para ello, se recurre a la función `entanglement()`, que devuelve el coeficiente de enredo. El resultado es una medida entre 1 y 0, siendo los valores más próximos a 1 aquellos que representan una mejor alineación. Ahora, se calcula el coeficiente de enredo para ambos dendogramas:

```
entanglement(dend_list)
```

```
## [1] 0.8971243
```

La función `cor.dendlist()` la matriz de correlación entre una lista de árboles. Dispone de distintos métodos, entre los cuales destacamos `cophenetic` que será el que se usará, aunque también existen otros. El resultado, como se ha comentado, es una matriz de correlación, interpretándose de forma usual. Para calcular la matriz de correlación copenética entre una lista de árboles, se utiliza la siguiente instrucción:

```
cor.dendlist(dend_list, method = "cophenetic")
```

```
##           [,1]      [,2]
## [1,] 1.0000000 0.9646883
## [2,] 0.9646883 1.0000000
```

Para obtener directamente el coeficiente, sin tener que usar una matriz, se calcula de la siguiente manera:

```
cor_cophenetic(dend1, dend2)
```

```
## [1] 0.9646883
```

En la sección anterior se ha comentado como existen diferentes métodos para poder realizar las agrupaciones. Entonces, el objetivo actual es crear diferentes dendogramas usando los diferentes métodos, y compararlos entre ellos.

```
dend1 <- subset %>% dist %>% hclust("complete") %>% as.dendrogram
dend2 <- subset %>% dist %>% hclust("single") %>% as.dendrogram
dend3 <- subset %>% dist %>% hclust("average") %>% as.dendrogram
dend4 <- subset %>% dist %>% hclust("centroid") %>% as.dendrogram

dend_list <- dendlist("Complete" = dend1, "Single" = dend2,
"Average" = dend3, "Centroid" = dend4)
cors <- cor.dendlist(dend_list)

round(cors, 2)
```

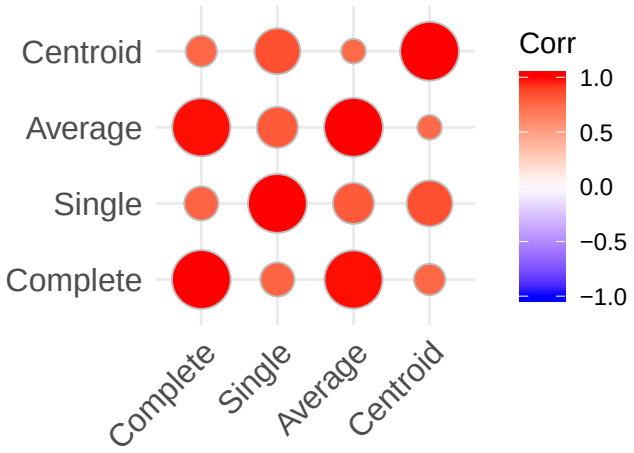
```
##          Complete Single Average Centroid
```

```
## Complete     1.00    0.76    0.99      0.75
## Single       0.76    1.00    0.80      0.84
## Average      0.99    0.80    1.00      0.74
## Centroid     0.75    0.84    0.74      1.00
```

El resultado obtenido permite obtener dos conclusiones: los métodos `average` y `complete` son los dos métodos diferentes que más correlación presentan, mientras que los métodos `average` y `centroid` son los que presentan una correlación menor.

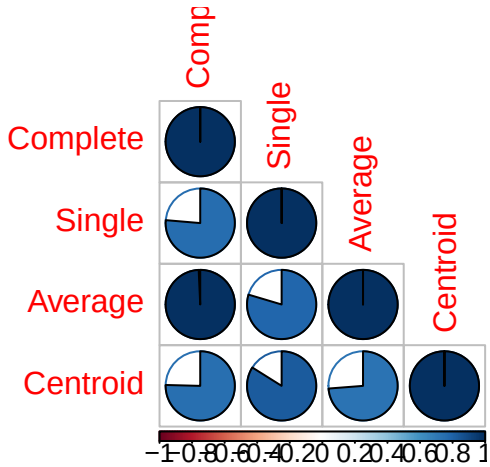
Ahora, se representan esas correlaciones con el gráfico usual:

```
ggcorrplot(cors, method = "circle")
```



Utilizando la librería `corrplot` se obtiene también el gráfico de correlación, mostrando la correlación de dos maneras diferentes: por la intensidad del color y la apertura del sector circular:

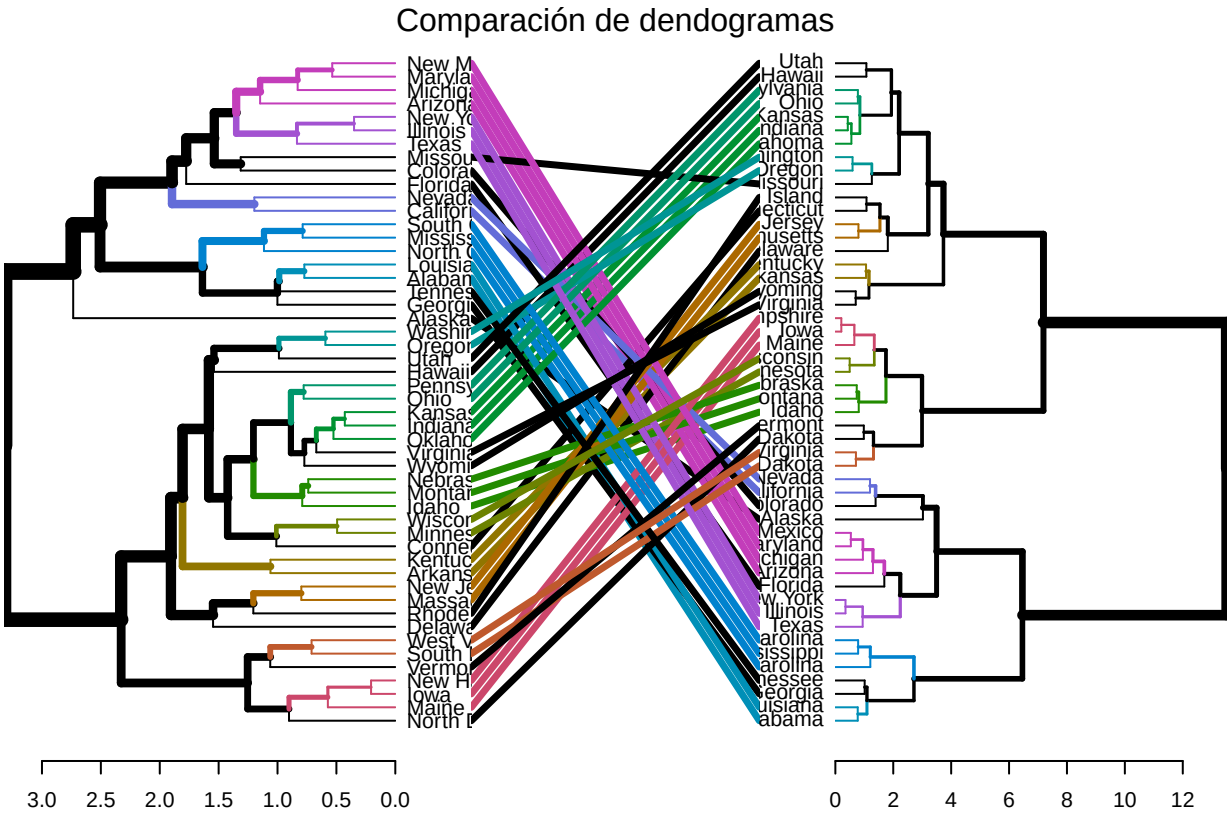
```
corrplot(cors, "pie", "lower")
```



Se debe destacar que la comparación de los dendogramas se ha realizado con un subconjunto de los datos. Si ahora se realiza el mismo proceso con todos los datos disponibles.

```
res.dist <- dist(df, method = "euclidean")

hc1 <- hclust(res.dist, method = "average")
hc2 <- hclust(res.dist, method = "ward.D2")

dend1 <- as.dendrogram (hc1)
```

```
dend2 <- as.dendrogram (hc2)
dend_list <- dendlist(dend1, dend2)

tanglegram(dend1, dend2,
highlight_distinct_edges = FALSE,
common_subtrees_color_lines = TRUE,
common_subtrees_color_branches = TRUE,
main = "Comparación de dendogramas", cex_main = 1)
```




```
entanglement(dend_list)
```

```
## [1] 0.8342094
```

Si se toman todos los datos sigue habiendo una alta correlación, y es parecida a la anterior, ya que la correlación obtenida para el subconjunto de datos era 0.897. A continuación, se calcula matriz de correlaciones entre los dendogramas:

```
cor.dendlist(dend_list, method = "cophenetic")
```

```
##          [,1]     [,2]
## [1,] 1.000000 0.843143
## [2,] 0.843143 1.000000
```

```
cor_cophenetic(dend1, dend2)
```

```
## [1] 0.843143
```

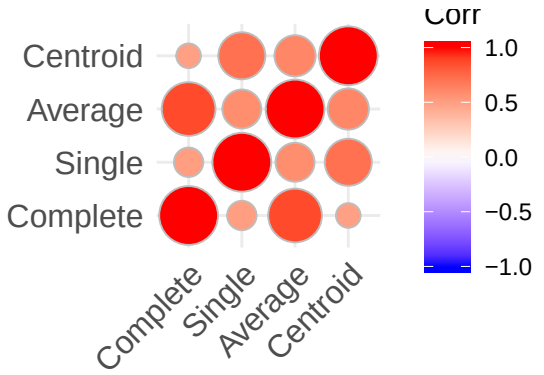
Nótese como aquí sí se puede apreciar una diferencia. Para los dos dendogramas anteriores el resultado de la correlación era 0.964, en contraposición al valor 0.84.

```
dend1 <- df %>% dist %>% hclust("complete") %>% as.dendrogram
dend2 <- df %>% dist %>% hclust("single") %>% as.dendrogram
dend3 <- df %>% dist %>% hclust("average") %>% as.dendrogram
dend4 <- df %>% dist %>% hclust("centroid") %>% as.dendrogram

dend_list <- dendlist("Complete" = dend1, "Single" = dend2,
"Average" = dend3, "Centroid" = dend4)
cors <- cor.dendlist(dend_list)

round(cors, 2)
```

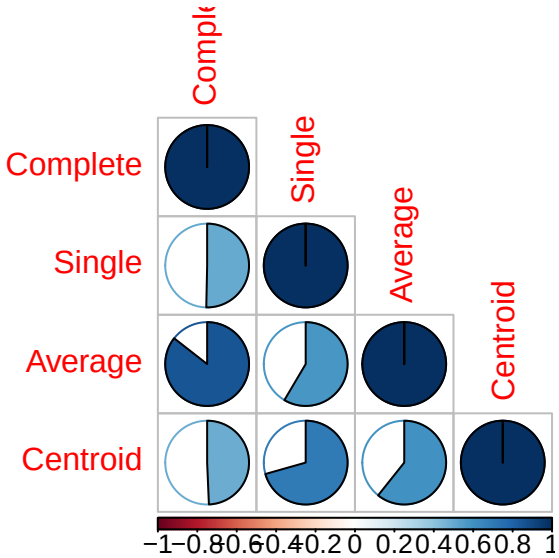
```
##          Complete Single Average Centroid
## Complete     1.00   0.50    0.86     0.49
## Single       0.50   1.00    0.58     0.71
## Average      0.86   0.58    1.00     0.61
## Centroid     0.49   0.71    0.61     1.00
```

Es en la matriz de correlaciones entre los diferentes métodos donde se observan más cambios. Los métodos más correlacionados siguen siendo `average` y `complete`, pero los menos correlacionados han sido `complete` y `centroid`. A continuación, se realizan los gráficos de las correlaciones:

```
ggcorrplot(cors, method = "circle")
```




```
corrplot(cors, "pie", "lower")
```



Conclusiones

Tal y como se ha comentado, al realizar agrupamientos siempre se suele recurrir, como primer método, al conocido *k*-means, que depende del número de grupos iniciales que se establecen. Ahora bien, se ha enfocado el estudio de este trabajo a estudiar métodos alternativos que realizan las agrupaciones de forma automática, es decir, agrupan por similitud sin depender de condiciones iniciales o el estado inicial de los datos. No obstante, no devuelven clusters como tal, sino que se deben establecer a mano los grupos finales que se quieren obtener.

Acto seguido, se han visto las diferentes maneras para comparar los dendogramas ya que para realizar las agrupaciones existen diferentes métodos que no devuelven el mismo resultado. Así, para el subconjunto extraído de los datos iniciales, se ha podido determinar que los métodos más afines son `complete` y `average`, y los que menos `average` y `centroid`. Al realizar la comparación con el conjunto de datos completo, se ha obtenido que `complete` y `average` seguían siendo los métodos más correlacionados. No obstante, los métodos menos correlacionados han pasado a ser `complete` y `centroid`.

Para poder llevar a cabo el trabajo el trabajo se han utilizado diferentes librerías. En efecto, las librerías `ggplot2` y `tidyverse` se han utilizado para el gráfico y el tratamiento de los datos, respectivamente. De la librería `ggplot2` se destaca `ggcorrplot`, usada para graficar la matriz de correlaciones y obtener una representación visual de la comparación de los diferentes métodos. Acto seguido, la librería `cluster` se ha usado para automatizar las agrupaciones AGNES y DIANA. Finalmente, para realizar los dendogramas y las comparaciones entre ellos, se han usado `dendextend` y `ggdendro`; se debe destacar que ambos paquetes usan `ggplot2` para realizar los gráficos.

Finalmente, se ha seguido la estructura de los capítulos 7, 8 y 9 del libro Multivariate Analysis I, del autor Alboukadel Kassambara.