Clustering

Joaquín Amat Rodrigo j.amatrodrigo@gmail.com

Agosto, 2017

Índice

Introducción 2
Medidas de distancia 2
  Distancia euclídea 2
  Correlación 4
  Escala de las variables 5
K-means clustering 6
  Ejemplo 8
Hierarchical Clustering 13
  Algoritmo 13
  Dendograma 15
  Ejemplo 18
Limitaciones del clustering 22
Ejemplo: Clasificar tumores por su perfil genético 23
Bibliografía 29

Versión PDF: https://github.com/JoaquinAmatRodrigo/Estadistica-con-R

Introducción

El término *clustering* hace referencia a un amplio abanico de técnicas *unsupervised* cuya finalidad es encontrar grupos (*clusters*) dentro de un conjunto de observaciones. Las particiones se establecen de forma que las observaciones que están dentro de un mismo grupo son similares entre ellas y distintas a las observaciones de otros grupos. Se trata de un método *unsupervised* ya que el proceso ignora la variable respuesta que indique a que grupo pertenece realmente cada observación (si es que existe tal variable). La agrupación se realiza teniendo en cuenta únicamente el valor de los predictores.

Dada la popularidad del *clustering* en disciplinas muy distintas, se han desarrollado multitud de variantes y adaptaciones de sus métodos y algoritmos. Sin embargo, todas ellas tienen una cosa en común, para poder llevar a cabo las agrupaciones se necesita definir y cuantificar la similitud (o diferencia) entre las observaciones.

Medidas de distancia

El término distancia se emplea dentro del contexto del *clustering* como cuantificación de la similitud entre observaciones. Si se representan las observaciones en un espacio p dimensional, siendo p el número de variables asociadas a cada observación, cuando más se asemejen dos observaciones más próximas estarán, de ahí que se emplee el término distancia. La característica que hace del *clustering* un método adaptable a escenarios muy diversos es que puede emplear cualquier tipo de distancia, lo que permite al investigador escoger la más adecuada para el estudio en cuestión. A continuación se describen algunas de las más utilizadas.

Distancia euclídea

La distancia euclídea entre dos puntos p y q se define como la longitud del segmento que une ambos puntos. En coordenadas cartesianas, la distancia euclídea se calcula empleando el teorema de Pitágoras. Por ejemplo, en un espacio de dos dimensiones en el que cada punto está definido por las coordenadas (x, y), la distancia euclídea entre p y q viene dada por la ecuación:

$$d(p,q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}$$

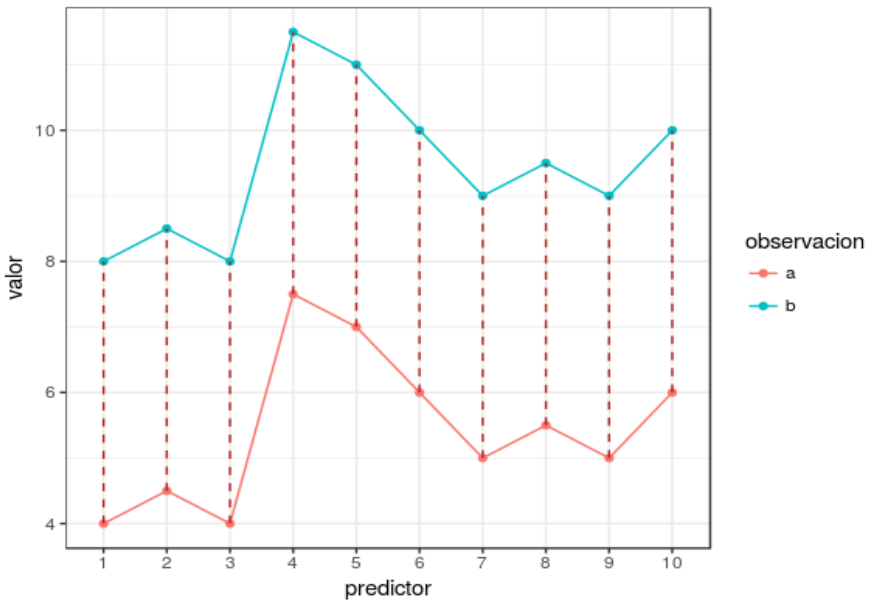
Esta ecuación puede generalizarse para un espacio euclídeo n-dimensional donde cada punto está definido por un vector de n coordenadas: $p = (p_1, p_2, p_3, \ldots, p_n)$ y $q = (q_1, q_2, q_3, \ldots, q_n)$.

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \ldots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$$

Una forma de dar mayor peso a aquellas observaciones que están más alejadas es emplear la distancia euclídea al cuadrado. En el caso del *clustering*, donde se busca agrupar observaciones que minimicen la distancia, esto se traduce en mayor penalización para aquellas observaciones que están más distantes.

La siguiente imagen muestra el perfil de dos observaciones definidas por 10 variables (espacio con 10 dimensiones).

```
library(ggplot2)
observacion_a <- c(4, 4.5, 4, 7.5, 7, 6, 5, 5.5, 5, 6)
observacion_b <- c(4, 4.5, 4, 7.5, 7, 6, 5, 5.5, 5, 6) + 4
datos <- data.frame(observacion = rep(c("a", "b"), each = 10),
                    valor = c(observacion_a, observacion_b),
                    predictor = 1:10)
ggplot(data = datos, aes(x = as.factor(predictor), y = valor,
                         colour = observacion)) +
  geom_path(aes(group = observacion)) +
  geom_point() +
  geom_line(aes(group = predictor), colour = "firebrick", linetype = "dashed") +
  labs(x = "predictor") +
  theme_bw()
```



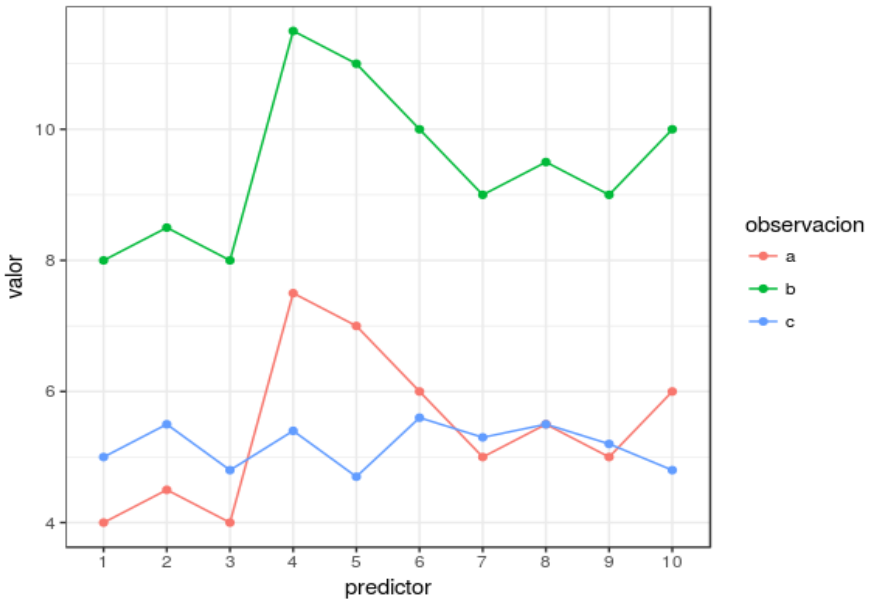
La distancia euclídea entre las dos observaciones equivale a la raíz cuadrada de la suma de las longitudes de los segmentos rojos que unen cada par de puntos. Tiene en cuenta por lo tanto el desplazamiento individual de cada uno de los predictores.

Correlación

La correlación es una medida de distancia muy útil cuando la definición de similitud se hace en términos de patrón o forma y no de desplazamiento o magnitud. ¿Qué quiere decir esto? En la imagen anterior, las dos observaciones tienen exactamente el mismo patrón, la única diferencia es que una de ellas está desplazada 4 unidades por encima de la otra. Si se emplea como medida de similitud el valor de la correlación, ambas observaciones se consideran idénticas.

En la siguiente imagen se muestra el perfil de 3 observaciones. Acorde a la distancia euclídea, las observaciones *b* y *c* son las más similares, mientras que acorde a la correlación, las observaciones más similares son *a* y *b*.

```
library(ggplot2)
observacion_a <- c(4, 4.5, 4, 7.5, 7, 6, 5, 5.5, 5, 6)
observacion_b <- c(4, 4.5, 4, 7.5, 7, 6, 5, 5.5, 5, 6) + 4
observacion_c <- c(5, 5.5, 4.8, 5.4, 4.7, 5.6, 5.3, 5.5, 5.2, 4.8)
datos <- data.frame(observacion = rep(c("a", "b", "c"), each = 10),
                    valor = c(observacion_a, observacion_b, observacion_c),
                    predictor = 1:10)
ggplot(data=datos, aes(x = as.factor(predictor), y=valor, colour = observacion)) +
  geom_path(aes(group = observacion)) +
  geom_point() + labs(x = "predictor") +  theme_bw()
```

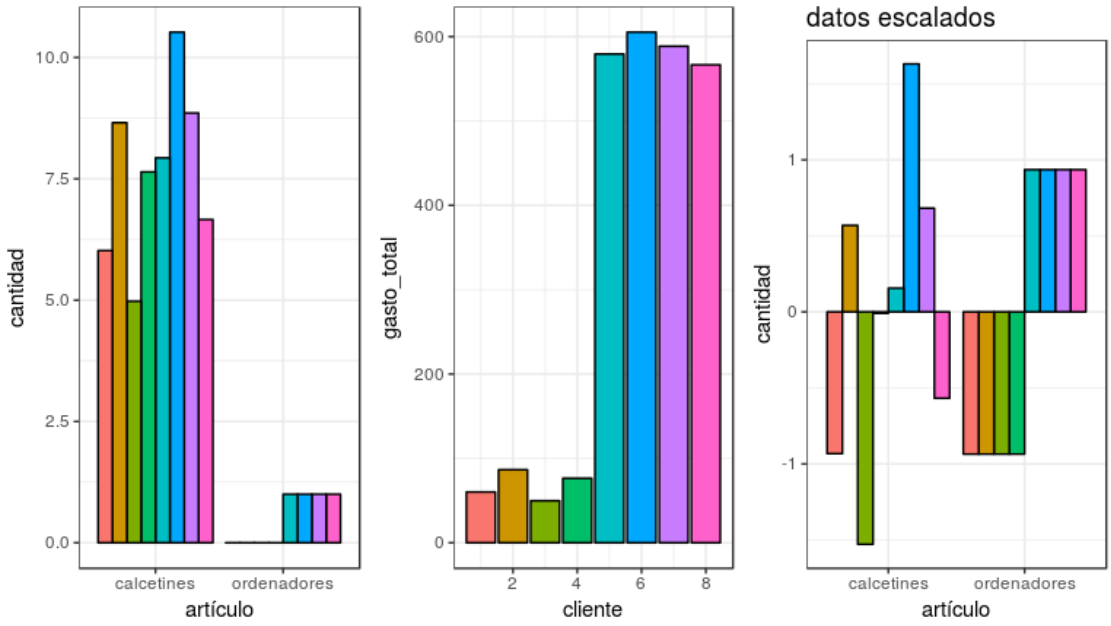


Este ejemplo pone de manifiesto que no existe una única medida de distancia que sea mejor que las demás, sino que, dependiendo del contexto, una será más adecuada que otra.

Escala de las variables

Al igual que en otros método estadísticos (*PCA*, *ridge regression*, *lasso*...), la escala en la que se miden las variables y la magnitud de su varianza puede afectar en gran medida a los resultados obtenidos en *clustering*. Si una variable tiene una escala mucho mayor que el resto, determinará en gran medida el valor de distancia/similitud obtenido al comparar observaciones dirigiendo así la agrupación final. Escalar y centrar las variables de forma que todas ellas tengan media 0 y desviación estándar 1 antes de calcular la matriz de distancias, asegura que todas las variables tengan el mismo peso cuando se realice el *clustering*.

Para ilustrar este hecho, supóngase que una tienda online quiere clasificar a los compradores en función de los artículos que adquieren, por ejemplo, calcetines y ordenadores. La siguiente imagen muestra el número de artículos comprados por 8 clientes a lo largo de un año, junto con el gasto total de cada uno.



Si se intenta agrupar a los clientes por el número de artículos comprados, dado que los calcetines se compran con mucha más frecuencia que los ordenadores, van a tener más peso al crear los *clusters*. Por el contrario, si la agrupación se hace en base al gasto total de los clientes, como los ordenadores son mucho más caros, van a determinar en gran medida la clasificación. Escalando y centrando las variables se consigue igualar la influencia de calcetines y ordenadores.

K-means clustering

El método *K-means clustering* agrupa las observaciones en *K clusters* distintos, donde el número *K* de *clusters* lo determina el investigador. Para lograrlo sigue un proceso matemático bastante intuitivo y simple. Considérense C_1,..., C_K como los sets formados por los índices de las observaciones que forman cada uno de los *clusters*. Por ejemplo, el set C_1 contiene los índices de las observaciones agrupadas en el *cluster* 1. La nomenclatura empleada para indicar que la observación *i* pertenece al *cluster k* es: $i \in C_k$.

Todos los sets satisfacen dos propiedades:

- $C_1 \cup C_2 \cup \ldots \cup C_K = \{1, \ldots, n\}$. Significa que toda observación pertenece al menos a uno de los *K clusters*.
- $C_k \cap C_{k'} = \emptyset$ para todo $k \neq k'$. Implica que los *clusters* no solapan, ninguna observación pertenece a más de un *cluster*.

La idea detrás de este método es encontrar los *K* mejores *clusters*, entendiendo como *mejor cluster* aquel cuya varianza interna sea lo más pequeña posible. Se trata por lo tanto de un problema de optimización, en el que se tienen que repartir las observaciones en *k clusters* de forma que la suma de las varianzas internas de todos ellos sea lo menor posible. Para poder solucionar este problema es necesario definir una forma de cuantificar la varianza interna. Existen múltiples métodos de hacerlo, uno de los más comúnmente empleados define la varianza interna de un *cluster* ($W(C_k)$) como la suma de las distancias euclídeas al cuadrado entre todos los pares de observaciones que forman el *cluster*, dividida entre el número de observaciones del *cluster*. Esto equivale al valor promedio de la distancia euclídea cuadrada entre las observaciones del *cluster*.

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2$$

Minimizar la ecuación anterior de forma exacta es un proceso muy complejo debido a la inmensa cantidad de formas en las que *n* observaciones se pueden dividir en *k* grupos. Sin embargo, es posible obtener una solución que, aun no siendo la mejor de entre todas las posibles, es muy buena. El algoritmo empleado para ello es:

1. Asignar aleatoriamente un número entre 1 y *K* a cada observación. Esto sirve como asignación inicial aleatoria de las observaciones a los *clusters*.

2. Iterar los siguientes pasos hasta que la asignación de las observaciones a los *clusters* no cambie o se alcance un número máximo de iteraciones establecido por el usuario.

   – Para cada uno de los *clusters* calcular su centroide. Entendiendo por centroide la posición definida por la media de cada una de las dimensiones (variables) de las observaciones que forman el *cluster*. Aunque no es siempre equivalente, puede entenderse como el centro de gravedad.

   – Asignar cada observación al *cluster* cuyo centroide está más próximo.

Este algoritmo garantiza que en cada paso se reduzca la varianza total de los grupos hasta alcanzar el *óptimo local*. La siguiente imagen muestra cómo van cambiando las asignaciones de las observaciones a medida que se ejecuta cada paso del algoritmo.

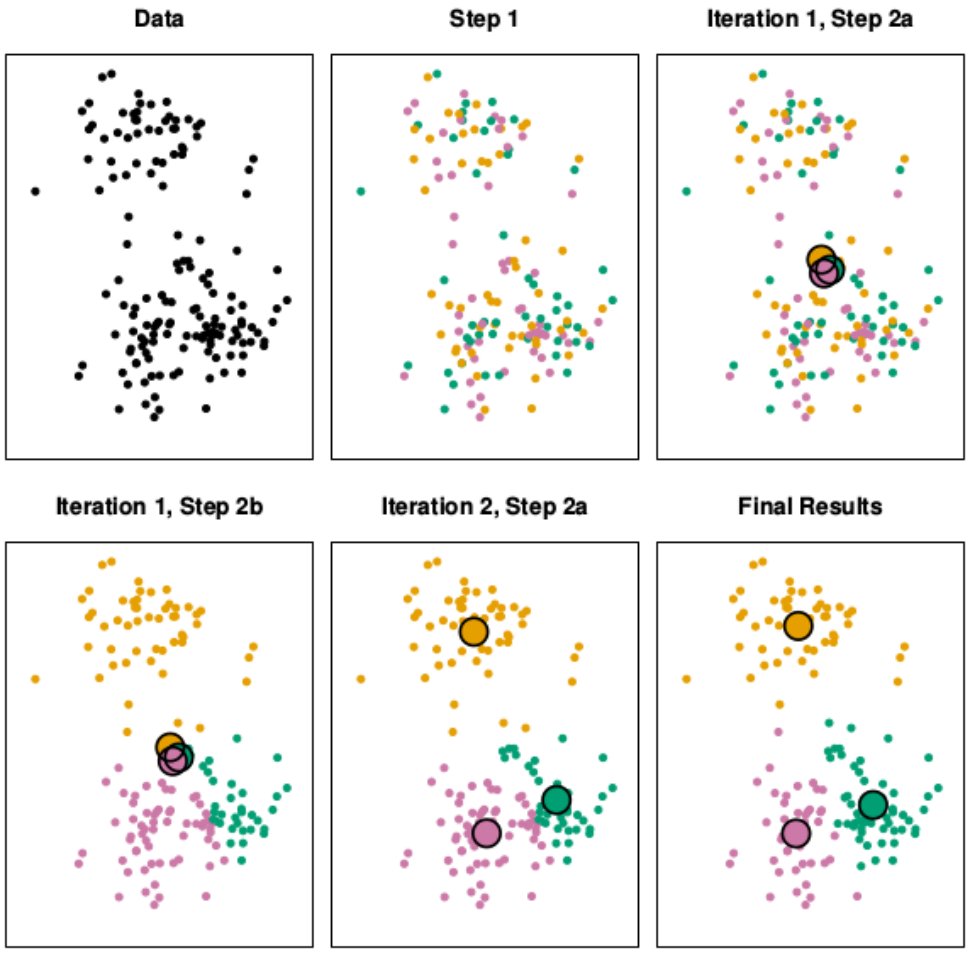


Imagen obtenida del libro ISLR

Debido a que el algoritmo de *K-means clustering* no evalúa todas las posibles distribuciones de las observaciones sino solo parte de ellas, los resultados obtenidos dependen de la asignación aleatoria inicial (paso 1). Por esta razón es importante ejecutar el algoritmo varias veces, cada una con un asignación aleatoria inicial distinta, y seleccionar aquella que haya conseguido un menor valor de varianza total.

Ejemplo

Los siguientes datos simulados contienen observaciones que pertenecen a cuatro grupos distintos. Se pretende aplicar K-means-clustering con el fin de identificarlos.

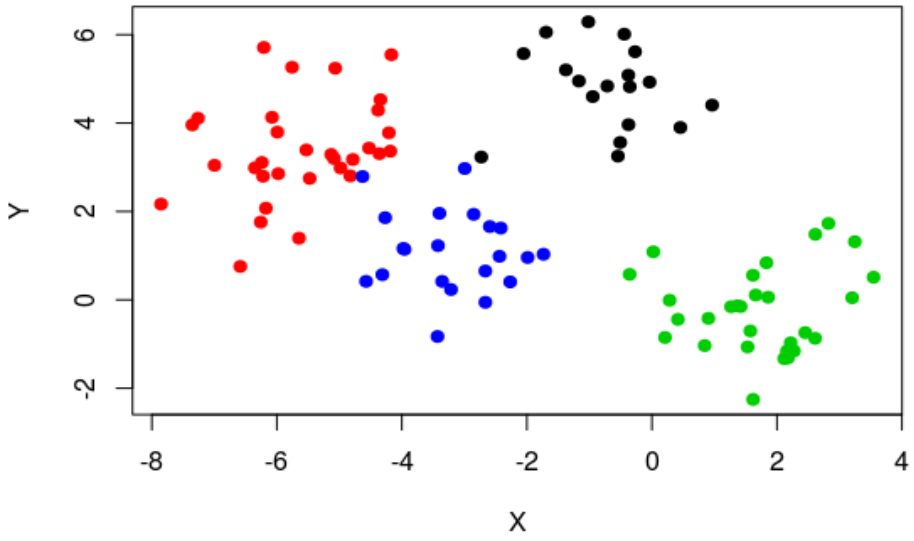
```
set.seed(101)
# Se simulan datos aleatorios con dos dimensiones
datos <- matrix(rnorm(n = 100*2), nrow = 100, ncol = 2,
                dimnames = list(NULL,c("x", "y")))

# Se determina la media que va a tener cada grupo en cada una de las dos
# dimensiones. En total 2*4 medias. Este valor se va a utilizar para
# separar cada grupo de los demás.
media_grupos <- matrix(rnorm(n = 8, mean = 0, sd = 4), nrow = 4, ncol = 2,
                       dimnames = list(NULL, c("media_x", "media_y")))
media_grupos <- cbind(grupo = 1:4, media_grupos)

# Se genera un vector que asigne aleatoriamente cada observación a uno de
# los 4 grupos
grupo <- sample(x = 1:4, size = 100, replace = TRUE)
datos <- cbind(datos, grupo)

# Se incrementa el valor de cada observación con la media correspondiente al grupo
# asignado.
datos <- merge(datos, media_grupos, by = "grupo")
datos[,"x"] <- datos[, "x"] + datos[, "media_x"]
datos[,"y"] <- datos[, "y"] + datos[, "media_y"]

plot(x = datos[,"x"], y = datos[,"y"], col = datos[,"grupo"], pch = 19,
     xlab = "X", ylab = "Y")
```



La función `kmeans()` del paquete `stats` realiza *k-mean-clustering*. Entre sus argumentos destacan: `centers`, que determina el número *K* de *clusters* que se van a generar y `nstart`, que determina el número de veces que se va a repetir el proceso, cada vez con una asignación aleatoria inicial distinta. Es recomendable que este último valor sea alto, entre 20-50, para no obtener resultados malos debido a una iniciación poco afortunada del proceso.

Como los datos se han simulado considerando que todas las dimensiones tienen aproximadamente la misma magnitud, no es necesario escalarlos ni centrarlos.

```
set.seed(101)
km_clusters <- kmeans(x = datos[, c("x", "y")], centers = 4, nstart = 50)
km_clusters
```

```
## K-means clustering with 4 clusters of sizes 32, 17, 30, 21
##
## Cluster means:
##            x          y
## 1 -5.5818142  3.3684991
## 2 -0.6148368  4.8861032
## 3  1.7226318 -0.2584919
## 4 -3.1068542  1.1213302
##
## Clustering vector:
##   [1] 2 2 2 2 2 2 2 2 2 2 4 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [36] 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [71] 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 1 4 4 4 4 4 4 4 4 4 4
##
```

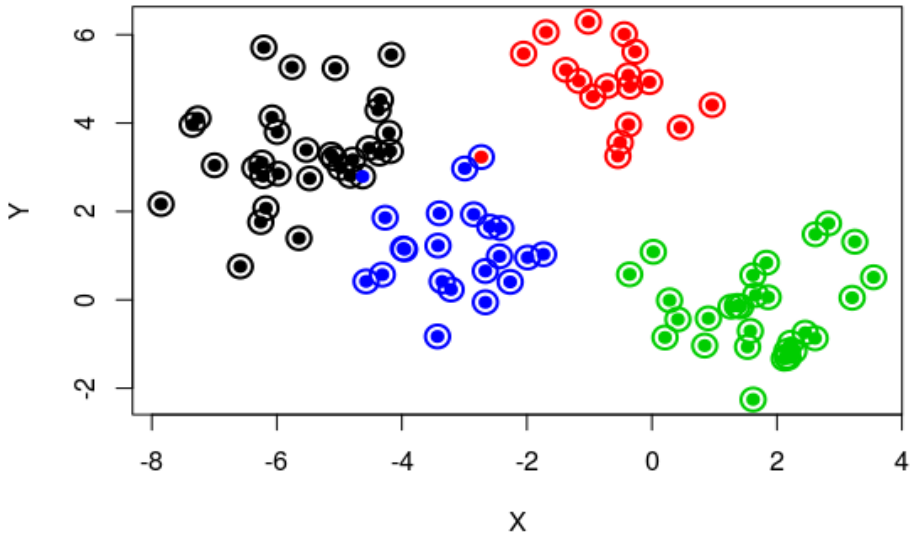
```
## Within cluster sum of squares by cluster:
## [1] 71.98228 21.04952 54.48008 30.82790
##  (between_SS / total_SS =  87.6 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

El objeto devuelto por la función `kmeans()` contiene entre otros datos: la media de cada una de las variables para cada *cluster*, un vector indicando a que *cluster* se ha asignado cada observación, la suma de cuadrados interna de cada *cluster* y el ratio de la suma de cuadrados entre *clusters* y la suma de cuadrados totales. Este último término es equivalente al R^2 de los modelos de regresión, indica el porcentaje de varianza explicada por el modelo respecto al total de varianza observada. Puede utilizarse para evaluar el *clustering* obtenido, pero, al igual que ocurre con R^2 al incrementar el número de predictores, el ratio *between_SS / total_SS* aumenta con el número de *clusters* creados.

Al tratarse de una simulación, se conoce el número real de grupos (4) y a cuál de ellos pertenece cada observación. Esto no sucede en la mayoría de casos prácticos, pero es útil para evaluar cómo de bueno es el método de *k-means-clustering* clasificando las observaciones.

```
# Se representan circunferencias con las asignaciones hechas por k-means-clustering
datos <- cbind(cluster = km_clusters$cluster, datos)
plot(x = datos[,"x"], y = datos[,"y"], col = km_clusters$cluster, pch = 1,
     cex = 2, lwd = 2, xlab = "X", ylab = "Y")

# Se rellenan las circunferencias con puntos del color real del grupo al
# que pertenecen las observaciones. Es necesario hacer coincidir los
# colores con el mismo orden que el devuelto por la función kmeans() ya
# que el clustering no asigna variable respuesta, solo agrupa las
# observaciones.
points(x = datos[,"x"], y = datos[,"y"], col = c(2,1,3,4)[datos[, "grupo"]],
       pch = 19)
```



El gráfico muestra que solo dos observaciones se han agrupado incorrectamente. Este tipo de visualización es muy útil e informativa, sin embargo solo es posible cuando se trabaja con dos dimensiones. Si los datos contienen más de dos variables (dimensiones), una posible solución es utilizar las dos primeras componentes principales obtenidas en un *PCA* previo.

El número de aciertos y errores puede representarse también en modo de matriz de confusión. A la hora de interpretarlas es importante recordar que el *clustering* asigna las observaciones a *clusters* cuyo identificador no tiene que por qué coincidir con la nomenclatura empleada para los grupos reales. Por ejemplo, el grupo 2 puede haberse identificado como *cluster 1* (tal como ocurre en este ejemplo). Así pues, por cada fila de la matriz cabe esperar un valor alto (coincidencias) para una de las posiciones y valores bajos en las otras (errores de clasificación).

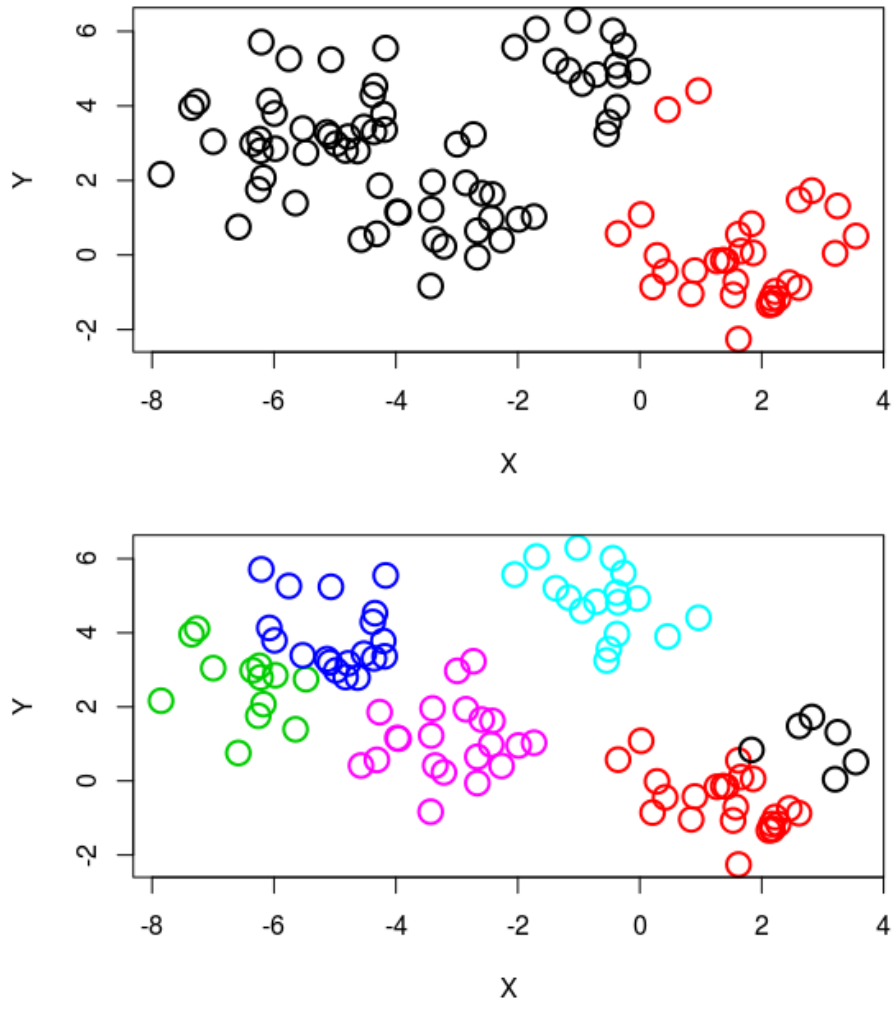
```
table(km_clusters$cluster, datos[, "grupo"],
      dnn = list("cluster", "grupo real"))
```

```
##         grupo real
## cluster  1  2  3  4
##       1  0 31  0  1
##       2 17  0  0  0
##       3  0  0 30  0
##       4  1  0  0 20
```

En este análisis, solo dos de las 100 observaciones se han agrupado erróneamente. De nuevo repetir que, en la realidad, no se suelen conocer los verdaderos grupos en los que se dividen las observaciones, de lo contrario no se necesitaría aplicar *clustering*.

Supóngase ahora que se trata de un caso real en el que se desconoce el número de grupos en los que se subdividen las observaciones. El investigador tendría que probar con diferentes valores de K y decidir cuál parece más razonable. A continuación se muestran los resultados para $K = 2$ y $K = 6$.

```
par(mfrow = c(2,1))
# Resultados para K = 2
set.seed(101)
km_clusters_2 <- kmeans(x = datos[, c("x", "y")], centers = 2, nstart = 50)
datos <- cbind(cluster = km_clusters_2$cluster, datos)
plot(x = datos[,"x"], y = datos[,"y"], col = km_clusters_2$cluster,
     pch = 1, cex = 2, lwd = 2, xlab = "X", ylab = "Y")
# Resultados para K = 6
set.seed(101)
km_clusters_6 <- kmeans(x = datos[, c("x", "y")], centers = 6, nstart = 50)
datos <- cbind(cluster = km_clusters_6$cluster, datos)
plot(x = datos[,"x"], y = datos[,"y"], col = km_clusters_6$cluster,
     pch = 1, cex = 2, lwd = 2, xlab = "X", ylab = "Y")
```



Al observar los resultado obtenidos para $K = 2$, es intuitivo pensar que el grupo que se encuentra entorno a las coordenadas $x = -1, y = 6$ (mayoritariamente considerado como negro) debería ser un grupo separado. Para $K = 6$ no parece muy razonable la separación de los grupos rojo y negro. Este ejemplo muestra la principal limitación del método de *k-means-clustering*, el hecho de tener que escoger de antemano el número de *clusters* que se generan.

Hierarchical Clustering

El *hierarchical clustering* es un método alternativo al *K-means-clustering* que presenta dos ventajas: no requiere que se especifique el número de *clusters* y sus resultados pueden representarse de forma muy intuitiva en una estructura de árbol llamada dendograma. Este método se subdivide a su vez en distintos tipos dependiendo de la estrategia seguida para construir el dendograma. En este documento se describe el método *bottom-up* o *agglomerative*, en el que el agrupamiento se inicia en la base del árbol, donde cada observación es un grupo individual, y se van combinado a medida que la estructura crece hasta converger en una única "rama" central.

Algoritmo

La estructura resultante de un *hierarchical clustering agglomerative* se obtiene mediante un algoritmo sencillo.

1. El proceso se inicia considerando cada una de las observaciones como un *cluster* individual formando la base del dendograma.
2. Se inicia un proceso iterativo hasta que todas las observaciones pertenecen a un único *cluster*:
   - Se calcula la distancia entre cada posible par de los *n clusters*. El investigador debe determinar el tipo de medida emplea para cuantificar la similitud entre observaciones o grupos (distancia y *linkage*).
   - Los dos *clusters* más similares se fusionan, de forma que quedan *n*-1 *clusters*.

La siguiente imagen muestra cómo se van sucediendo las agrupaciones a medida que avanzan las primeras iteraciones del algoritmo.

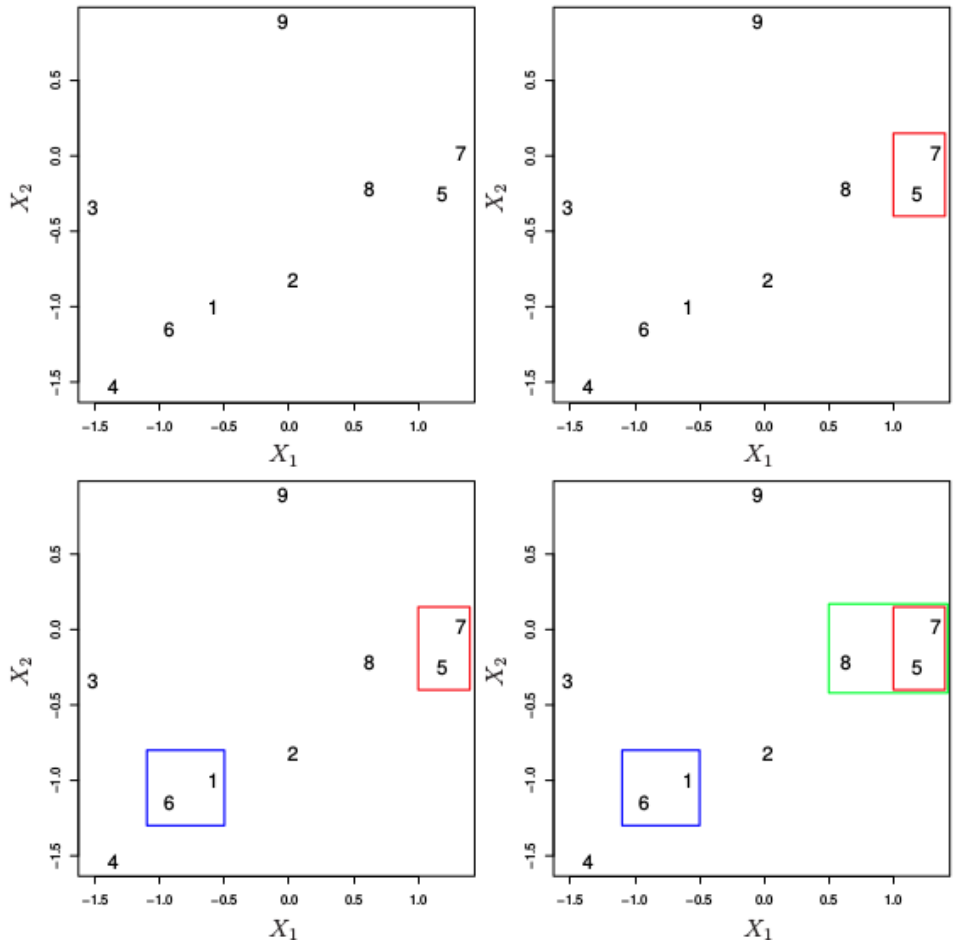


Imagen obtenida del libro ISLR

Para que el proceso de agrupamiento pueda llevarse a cabo tal como indica el algoritmo anterior, es necesario definir cómo se cuantifica la similitud entre dos *clusters*. Es decir, se tiene que extender el concepto de distancia entre pares de observaciones para que sea aplicable a pares de grupos, cada uno formado por varias observaciones. A este proceso se le conoce como *linkage*. A continuación se describen los 4 tipos de *linkage* más empleados y sus definiciones.

- **Completo**: Se calcula la distancia (similitud) entre todos los posibles pares formados por una observación del *cluster A* y una del *cluster B*. La mayor de todas ellas se selecciona como medida de similitud entre los dos *clusters*. Se trata de la medida más conservadora (*maximal intercluster dissimilarity*).

- **Single**: Se calcula la distancia (similitud) entre todos los posibles pares formados por una observación del *cluster A* y una del *cluster B*. La menor de todas ellas se selecciona como

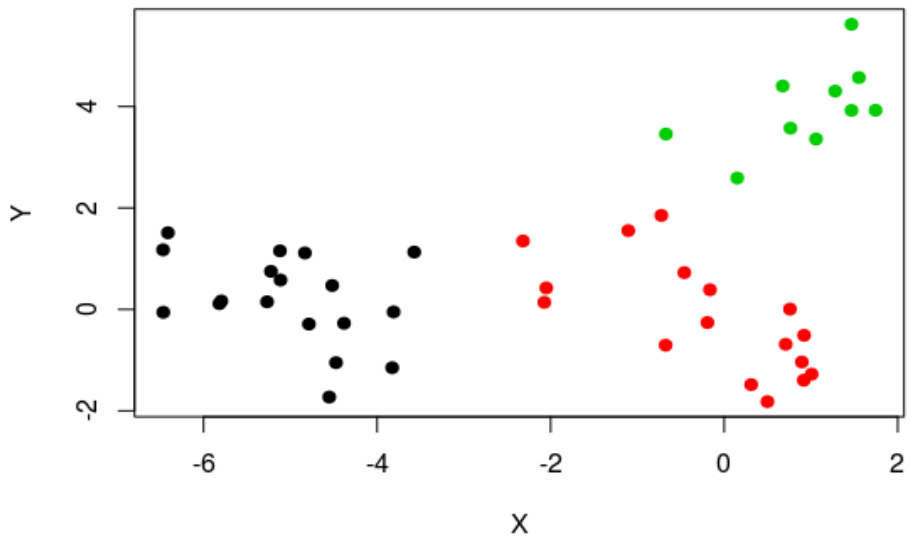
medida de similitud entre los dos *clusters*. Se trata de la medida menos conservadora (*minimal intercluster dissimilarity*).

- **Average**: Se calcula la distancia (similitud) entre todos los posibles pares formados por una observación del *cluster A* y una del *cluster B*. El valor promedio de todas ellas se selecciona como medida de similitud entre los dos *clusters* (*mean intercluster dissimilarity*).
- **Centroide**: Se calcula el centroide de cada uno de los *clusters* y se selecciona la distancia entre centroides como medida de similitud entre los dos *clusters*.

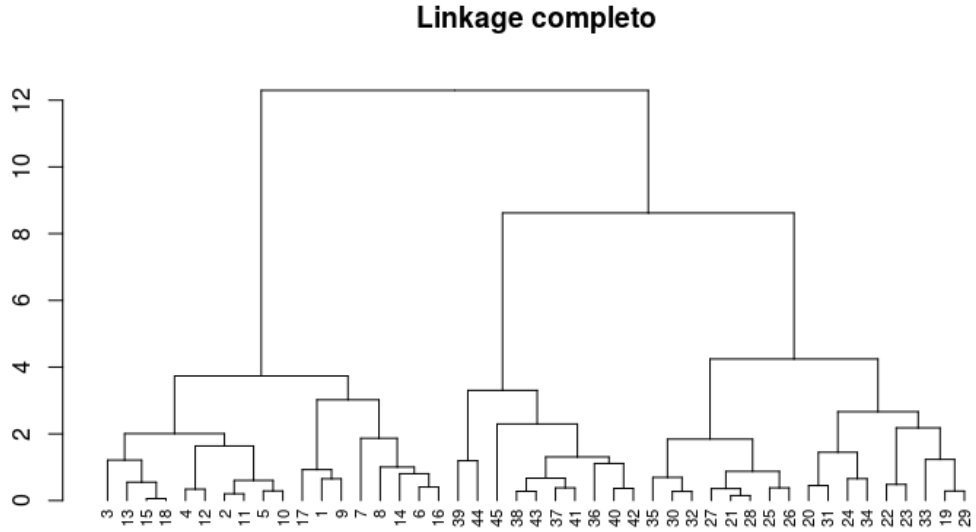
Los métodos de *linkage* completo y average suelen ser los preferidos por los analistas debido a que generan dendogramas más compensados. Sin embargo, no se puede determinar que uno sea mejor que otro, ya que depende del caso de estudio en cuestión. Por ejemplo, en genómica se emplea con frecuencia el método de *centroides*. Junto con los resultados de un proceso de *hierarchical clustering* siempre hay que indicar que distancia se ha empleado así como el tipo de *linkage*, ya que dependiendo de estos los resultados pueden variar en gran medida.

Dendograma

Supóngase que se dispone de 45 observaciones en un espacio de dos dimensiones que pertenecen a 3 grupos. Aunque se ha coloreado de forma distinta cada uno de los grupos para facilitar la comprensión de la idea, se va a suponer que se desconoce esta información y que se desea aplicar el método de *hierarchical clustering* para intentar reconocer los grupos.

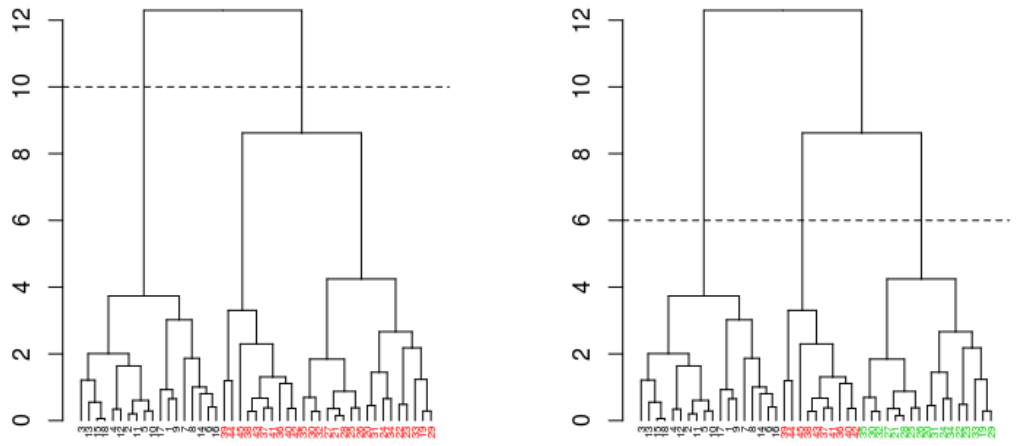


Al aplicar *hierarchical clustering* empleando como medida de similitud la distancia euclídea y *linkage* completo, se obtiene el siguiente dendograma.



En la base del dendograma, cada observación forma una terminación individual conocida como hoja o *leaf* del árbol. A medida que se asciende por la estructura, pares de hojas se fusionan formando las primeras ramas. Estas uniones se corresponden con los pares de observaciones más similares. También ocurre que ramas se fusionan con otras ramas o con hojas. Cuanto más temprana (más próxima a la base del dendograma) ocurre una fusión, mayor es la similitud existente entre los grupos de observaciones. Esto significa que, para cualquier par de observaciones, se puede identificar el punto del árbol en el que las ramas que contienen dichas observaciones se fusionan. La altura a la que esto ocurre (eje vertical) indica cómo de similares/diferentes son las dos observaciones. Los dendogramas, por lo tanto, se deben interpretar únicamente en base al eje vertical y no por las posiciones que ocupan las observaciones en el eje horizontal, esto último es simplemente por estética y puede variar de un programa a otro. Por ejemplo, la observación 11 es la más similar a la número 2 ya que es la primera fusión que recibe la observación 2 (y viceversa). Podría resultar tentador decir que la observación 12, situada inmediatamente a la derecha de la 2, es también muy similar, sin embargo, las observaciones 5 y 10 son más similares a al 2 a pesar de que se encuentran más alejadas en el eje horizontal. Del mismo modo, no es correcto decir que la observación 12 es más similar a la observación 2 de lo que lo es la 4 por el hecho de que está más próxima en en el eje horizontal. Prestando atención a la altura en que las respectivas ramas se unen, la única conclusión válida es que la similitud entre los pares 4-2 y 12-2 es la misma.

Además de interpretar la similitud entre observaciones en un dendograma, se tiene que poder identificar el número de *clusters* creados y qué observaciones forman parte de cada uno. Para ello se realiza un corte horizontal a una determinada altura del dendograma, el número de ramas que sobrepasan (en sentido ascendente) dicho corte se corresponde con el número de *clusters*. Por ejemplo, si realiza el corte a la altura de 10, se obtienen dos *clusters*, mientras que si se hace a la altura de 6 se obtienen 3. La altura de corte tiene por lo tanto la misma función que el valor *K* en *k-means-clustering*: controla el número de *clusters* obtenidos.



Dos propiedades adicionales se derivan de la forma en que se generan los *clusters* en el método de *hierarchical clustering*:

- Dada la longitud variable de las ramas, siempre existe un intervalo de altura para el que cualquier corte da lugar al mismo número de *clusters*. En el ejemplo anterior, todos los cortes entre las alturas 4 y 8 tienen como resultado los mismos 3 *clusters*.
- Con un solo dendograma se dispone de la flexibilidad para generar cualquier número de *clusters* desde 1 a *n*. La selección del número óptimo suele hacerse de forma visual, tratando de identificar las ramas principales en base a la altura a la que ocurren las uniones. En el ejemplo expuesto es razonable elegir entre 2 y 3 *clusters*.

Ejemplo

Los siguientes datos simulados contienen observaciones que pertenecen a cuatro grupos distintos. Se pretende aplicar hierarchical clustering aglomerativo con el fin de identificarlos.

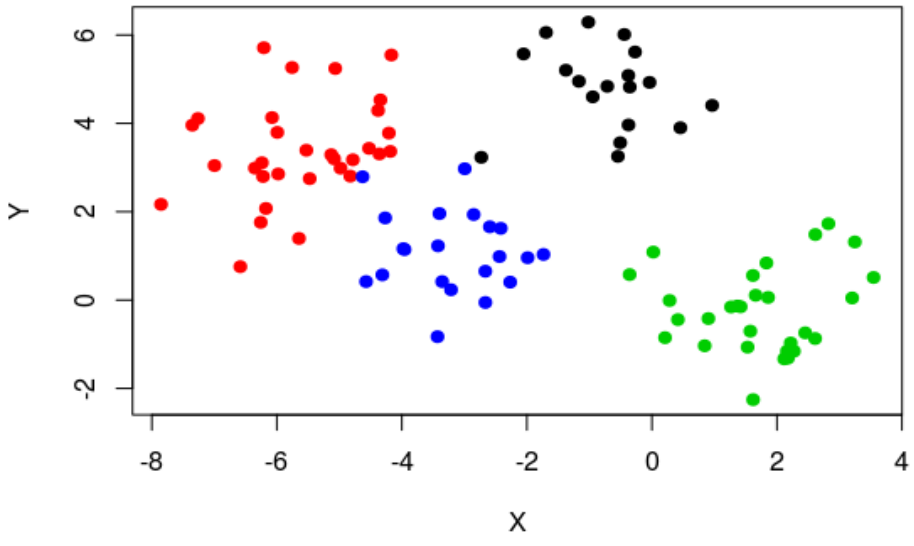
```
set.seed(101)
# Se simulan datos aleatorios con dos dimensiones
datos <- matrix(rnorm(n = 100*2), nrow = 100, ncol = 2,
                dimnames = list(NULL,c("x", "y")))

# Se determina la media que va a tener cada grupo en cada una de las dos
# dimensiones. En total 2*4 medias. Este valor se va a utilizar para
# separar cada grupo de los demás.
media_grupos <- matrix(rnorm(n = 8, mean = 0, sd = 4), nrow = 4, ncol = 2,
                       dimnames = list(NULL, c("media_x", "media_y")))
media_grupos <- cbind(grupo = 1:4, media_grupos)

# Se genera un vector que asigne aleatoriamente cada observación a uno de
# los 4 grupos
grupo <- sample(x = 1:4, size = 100, replace = TRUE)
datos <- cbind(datos, grupo)

# Se incrementa el valor de cada observación con la media correspondiente al grupo
# asignado.
datos <- merge(datos, media_grupos, by = "grupo")
datos[,"x"] <- datos[, "x"] + datos[, "media_x"]
datos[,"y"] <- datos[, "y"] + datos[, "media_y"]

plot(x = datos[,"x"], y = datos[,"y"], col = datos[,"grupo"], pch = 19,
     xlab = "X", ylab = "Y")
```

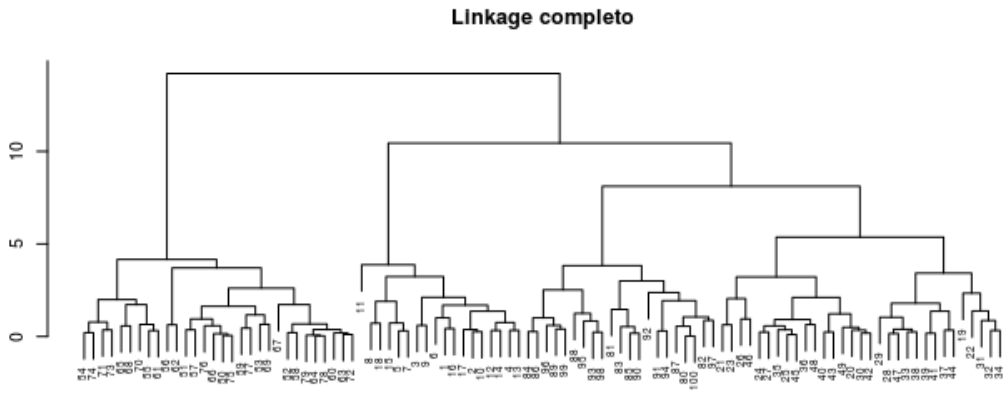


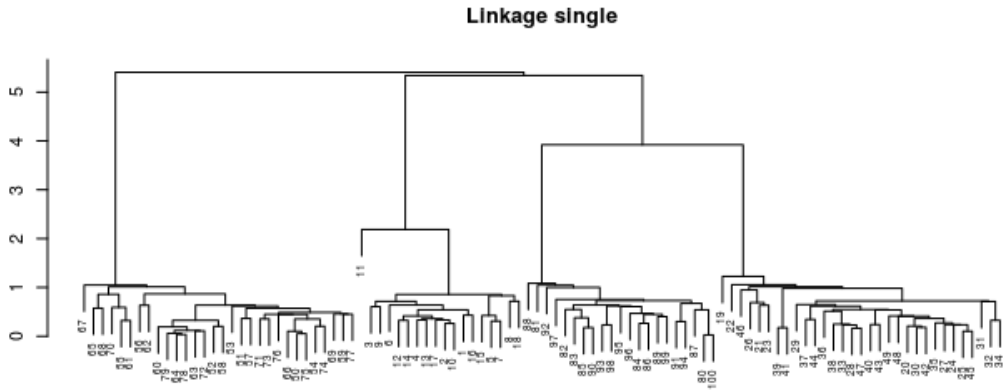
Al aplicar un *hierarchical clustering* se tiene que escoger una medida de distancia/similitud y un tipo de *linkage*. En este caso, se emplea la función `hclust()` indicando la distancia euclídea como medida de similitud y se comparan los *linkages* completo, single y average. Dado que los datos se han simulado considerando que las dos dimensiones tienen aproximadamente la misma magnitud, no es necesario escalarlos ni centrarlos.

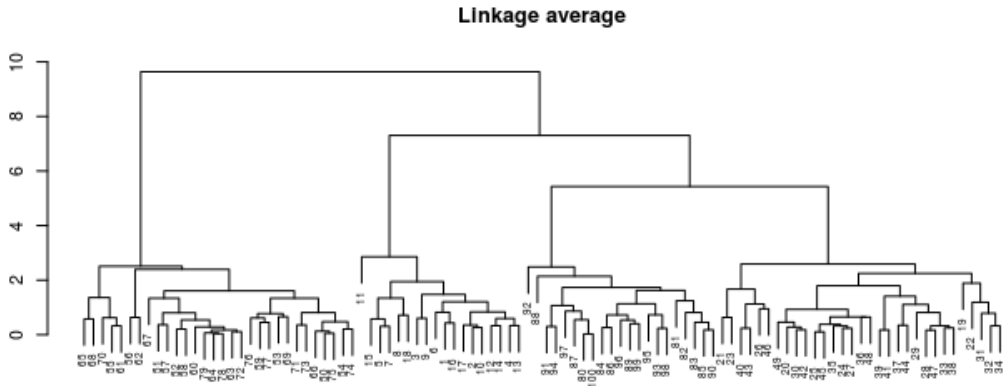
```
matriz_distancias <- dist(x = datos, method = "euclidean")
set.seed(567)
h_cluster_completo <- hclust(d = matriz_distancias, method = "complete")
h_cluster_single <- hclust(d = matriz_distancias, method = "single")
h_cluster_average <- hclust(d = matriz_distancias, method = "average")
```

Los objetos devueltos por `hclust()` pueden representarse en forma de dendograma con la función `plot()`.

```
par(mfrow = c(3,1))
plot(x = h_cluster_completo, cex = 0.6, xlab = "", ylab = "", sub = "",
     main = "Linkage completo")
plot(x = h_cluster_single, cex = 0.6, xlab = "", ylab = "", sub = "",
     main = "Linkage single")
plot(x = h_cluster_average, cex = 0.6, xlab = "", ylab = "", sub = "",
    main = "Linkage average")
```







El conocer que existen 4 grupos en la población permite evaluar que *linkage* consigue los mejores resultados. En este caso los tres tipos empleados identifican claramente 4 *clusters*, si bien esto no significa que en los 3 dendogramas los *clusters* estén formados por exactamente las mismas observaciones.

Una vez creado el dendograma, se tiene que decidir a qué altura se corta para generar los *clusters*. La función `cutree()` recibe como *input* un dendograma y devuelve el *cluster* al que se ha asignado cada observación dependiendo del número de *clusters* especificado (argumento `k`) o la altura de corte indicada (argumento `h`).

```
cutree(h_cluster_completo, k = 4)
```

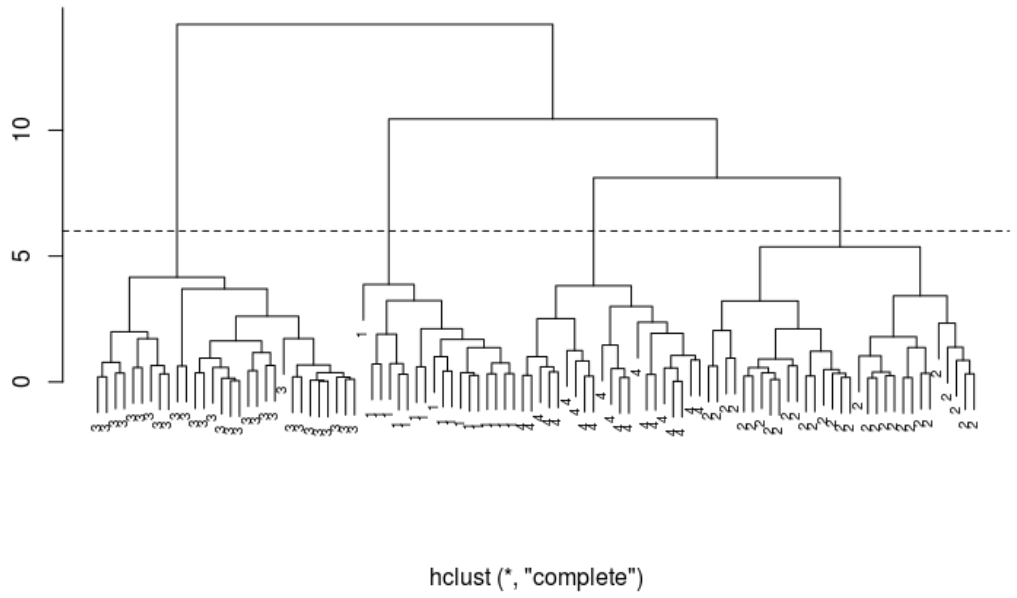
```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [36] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [71] 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
```

```
cutree(h_cluster_completo, h = 6)
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [36] 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [71] 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
```

Una forma visual de comprobar los errores en las asignaciones es indicando en el argumento `labels` el grupo real al que pertenece cada observación. Si la agrupación resultante coincide con a los grupos reales, entonces dentro de cada *clusters* las *labels* serán las mismas.

```
plot(x = h_cluster_completo, cex = 0.6, main = "", xlab = "", ylab = "",
    labels = datos[, "grupo"])
abline(h = 6, lty = 2)
```



hclust (*, "complete")

```
table(cutree(h_cluster_completo, h = 6), datos[, "grupo"])
```

```
##
##      1  2  3  4
##   1 18  0  0  0
##   2  0 31  0  0
##   3  0  0 30  0
##   4  0  0  0 21
```

El método de *hierarchical clustering aglomerativo* con *linkage* completo ha sido capaz de agrupar correctamente todas las observaciones.

Limitaciones del clustering

El *clustering* puede ser una herramienta muy útil para encontrar agrupaciones en los datos, sobre todo a medida que el volumen de los mismos aumenta. Sin embargo, es importante recordar algunas de sus limitaciones o problemas que pueden surgir al aplicarlo.

- **Pequeñas decisiones pueden tener grandes consecuencias:** A la hora de utilizar los métodos de *clustering* se tienen que tomar decisiones que influyen en gran medida en los resultados obtenidos. No existe una única respuesta correcta, por lo que en la práctica, se prueban diferentes opciones.
  - Escalado y centrado de las variables
  - Qué medida de distancia/similitud emplear
  - Número de *clusters* en *k-mean-clustering*
  - Tipo de *linkage* empleado en *hierarchical clustering*
  - A que altura establecer el corte de un dendograma
- **Validación de los clusters obtenidos:** Siempre que se aplique *clustering* a unos datos se van a encontrar agrupaciones. Sin embargo, el verdadero interés está en saber si los *clusters* obtenidos se corresponden con grupos que existen realmente en la población. Por ejemplo, si se aplica el mismo método a una segunda muestra de la misma población ¿Se obtendrían los mismos grupos? Existen algunas técnicas que asignan un *p-value* a cada *cluster* para saber si hay evidencias de que la agrupación no se debe al azar.
- **Falta de robustez:** Los métodos de *k-means-clustering* e *hierarchical clustering* asignan obligatoriamente cada observación a un grupo. Si existe en la muestra algún *outlier*, a pesar de que realmente no pertenezca a ningún grupo, el algoritmo lo asignará a uno de ellos provocando una distorsión significativa del *cluster* en cuestión.

Ejemplo: Clasificar tumores por su perfil genético

El set de datos NCI60 contiene información genética de 64 líneas celulares cancerígenas. Para cada una de ellas se ha cuantificado la expresión de 6830 genes mediante tecnología microarray. Los investigadores conocen el tipo de cáncer (histopatología) al que pertenece cada línea celular y quieren utilizar esta información para evaluar si el método de clustering (hierarchical clustering) es capaz de agrupar correctamente las líneas empleando los niveles de expresión génica.

Los métodos de *clustering* son *unsupervised*, lo que significa que al aplicarlos no se hace uso de la variable respuesta, en este caso el tipo de cáncer. Una vez obtenidos los resultados se añade esta información para determinar si es posible agrupar a las líneas celulares empleando su perfil de expresión.

```
library(ISLR)
data(NCI60)
str(NCI60)
```

```
## List of 2
##  $ data: num [1:64, 1:6830] 0.3 0.68 0.94 0.28 0.485 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:64] "V1" "V2" "V3" "V4" ...
##   .. ..$ : chr [1:6830] "1" "2" "3" "4" ...
##  $ labs: chr [1:64] "CNS" "CNS" "CNS" "RENAL" ...
```

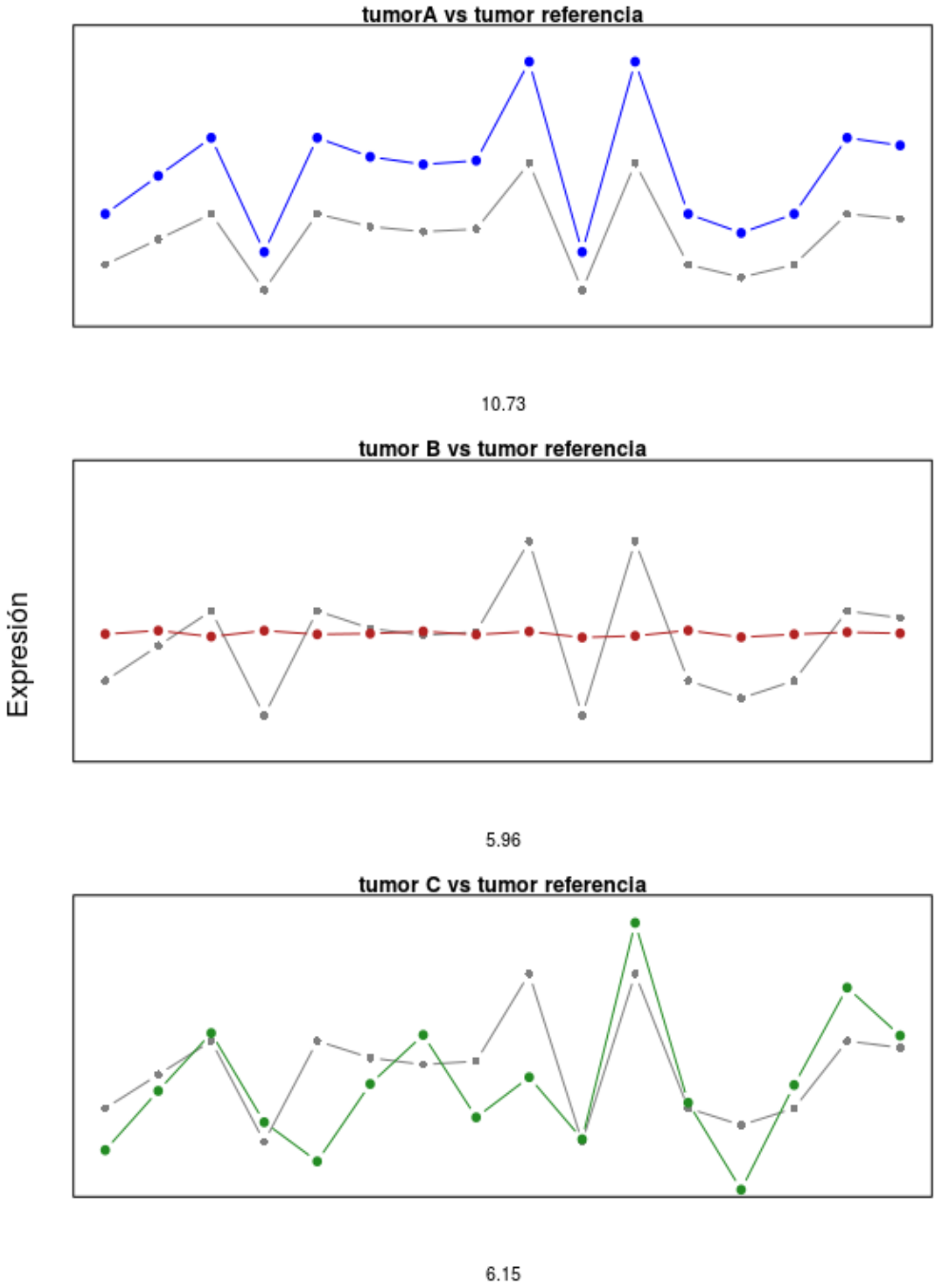
```
# Los datos están almacenados en forma de lista, un elemento contiene los niveles
# de expresión y otro el tipo de cáncer
expresion <- NCI60$data
tipo_cancer <- NCI60$labs
```

Una exploración inicial de los datos permite conocer el número de líneas celulares que hay de cada tipo de cáncer.

```
table(tipo_cancer)
```

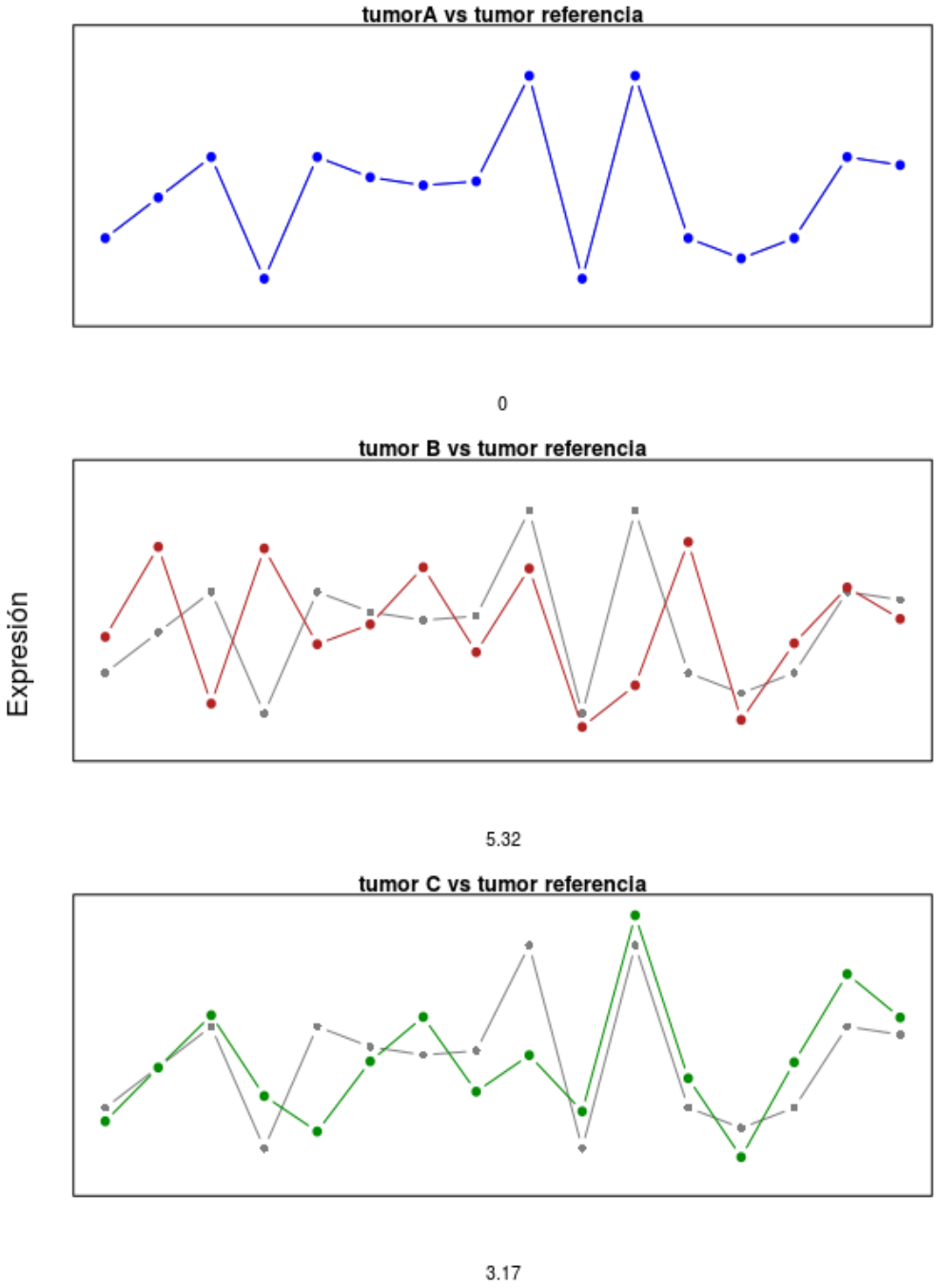
```
## tipo_cancer
##      BREAST         CNS       COLON K562A-repro K562B-repro    LEUKEMIA 
##           7           5           7           1           1           6 
## MCF7A-repro MCF7D-repro    MELANOMA       NSCLC     OVARIAN    PROSTATE 
##           1           1           8           9           6           2 
##       RENAL     UNKNOWN 
##           9           1
```

El siguiente paso antes de aplicar el método de *clustering* es decidir cómo se va a cuantificar la similitud. Los perfiles de expresión génica son un claro ejemplo de que es necesario comprender el problema en cuestión para hacer la elección adecuada. Para ilustrarlo se simula el perfil de 16 genes en 4 tumores (1 de referencia contra el que se comparan los otros 3).



Debajo de cada gráfico se indica la distancia euclídea entre cada par de tumores. Acorde a esta medida, el tumor menos parecido al de referencia es el A y el más parecido el B. Sin

embargo, analizando los perfiles con detenimiento puede observarse que el tumor A tiene exactamente el mismo perfil que el tumor de referencia pero desplazado unas unidades, mientras que el tumor B tiene un perfil totalmente distinto. Para evitar que las diferencias en magnitud determinen la similitud, se convierten los valores de expresión en *Z-factors* de forma que tengan media 0 y desviación estándar 1.

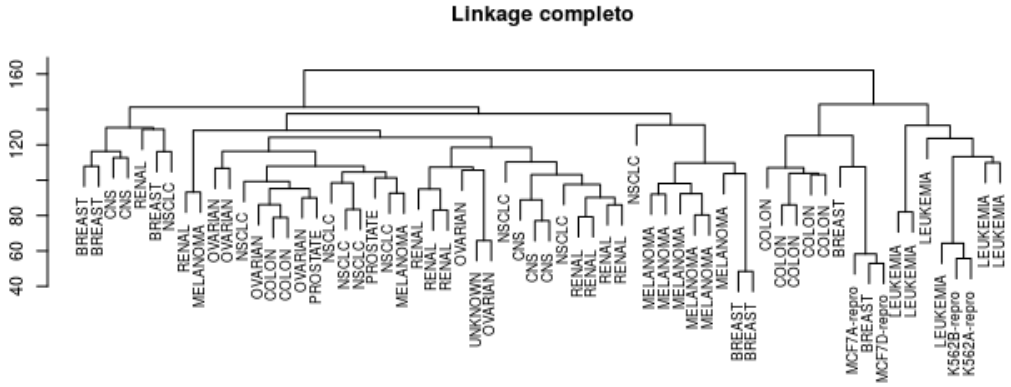


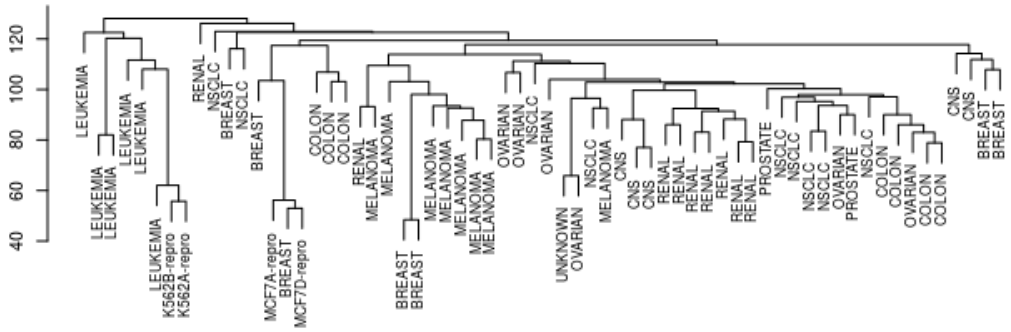
Una vez aplicado el escalado, las distancias obtenidas en las comparaciones tienen más sentido dentro del contexto de los perfiles de expresión génica. La similitud entre el tumores A y el de referencia es total (distancia 0), y el tumor B pasa a ser el más distinto. El mismo resultado se hubiese obtenido empleando como medida de similitud la correlación lineal de Pearson.

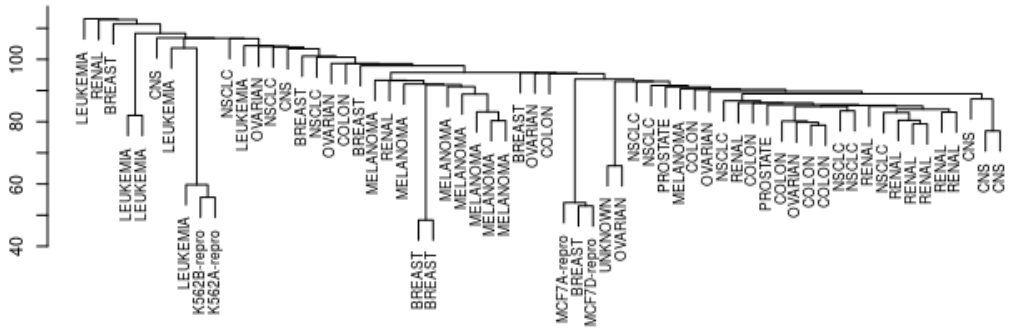
Se procede por lo tanto a escalar la matriz de expresión de NCI60 y se aplica *hierarchical clustering* con *linkage* completo, average y single.

```
expresion <- scale(expresion, center = TRUE, scale = TRUE)
matriz_distancias <- dist(x = expresion, method = "euclidean")
hc_completo <- hclust(d = matriz_distancias, method = "complete")
hc_average  <- hclust(d = matriz_distancias, method = "average")
hc_single   <- hclust(d = matriz_distancias, method = "single")


par(mfrow = c(3, 1))
plot(hc_completo, labels = tipo_cancer, ylab = "", xlab = "", sub = "",
     main = "Linkage completo", cex = 0.8)
plot(hc_average, labels = tipo_cancer, ylab = "", xlab = "", sub = "",
     main = "Linkage average", cex = 0.8)
plot(hc_single, labels = tipo_cancer, ylab = "", xlab = "", sub = "",
     main = "Linkage single", cex = 0.8)
```


Linkage completo
160
120
80
60
40
BREAST
BREAST
CNS
CNS
RENAL
BREAST
NSCLC
RENAL
MELANOMA
OVARIAN
OVARIAN
NSCLC
OVARIAN
COLON
COLON
OVARIAN
PROSTATE
NSCLC
NSCLC
NSCLC
PROSTATE
NSCLC
MELANOMA
RENAL
RENAL
RENAL
OVARIAN
UNKNOWN
OVARIAN
NSCLC
CNS
CNS
CNS
NSCLC
RENAL
RENAL
RENAL
RENAL
NSCLC
MELANOMA
MELANOMA
MELANOMA
MELANOMA
MELANOMA
MELANOMA
BREAST
BREAST
COLON
COLON
COLON
COLON
COLON
BREAST
MCF7A-repro
BREAST
MCF7D-repro
LEUKEMIA
LEUKEMIA
LEUKEMIA
LEUKEMIA
K562B-repro
K562A-repro
LEUKEMIA
LEUKEMIA


Linkage average
120
100
80
60
40
LEUKEMIA
LEUKEMIA
LEUKEMIA
LEUKEMIA
LEUKEMIA
LEUKEMIA
K562B-repro
K562A-repro
RENAL
NSCLC
BREAST
NSCLC
BREAST
MCF7A-repro
BREAST
MCF7D-repro
COLON
COLON
COLON
RENAL
MELANOMA
MELANOMA
BREAST
BREAST
MELANOMA
MELANOMA
MELANOMA
MELANOMA
MELANOMA
MELANOMA
OVARIAN
OVARIAN
NSCLC
OVARIAN
UNKNOWN
OVARIAN
NSCLC
MELANOMA
CNS
CNS
CNS
RENAL
RENAL
RENAL
RENAL
RENAL
RENAL
PROSTATE
NSCLC
NSCLC
NSCLC
NSCLC
OVARIAN
PROSTATE
NSCLC
COLON
COLON
COLON
OVARIAN
COLON
COLON
CNS
CNS
BREAST
BREAST


Linkage single
100
80
60
40
LEUKEMIA
RENAL
BREAST
LEUKEMIA
LEUKEMIA
CNS
LEUKEMIA
LEUKEMIA
K562B-repro
K562A-repro
NSCLC
LEUKEMIA
OVARIAN
NSCLC
CNS
BREAST
NSCLC
OVARIAN
COLON
BREAST
MELANOMA
RENAL
MELANOMA
BREAST
BREAST
MELANOMA
MELANOMA
MELANOMA
MELANOMA
MELANOMA
MELANOMA
BREAST
OVARIAN
COLON
MCF7A-repro
BREAST
MCF7D-repro
UNKNOWN
OVARIAN
NSCLC
NSCLC
PROSTATE
MELANOMA
COLON
OVARIAN
NSCLC
RENAL
COLON
PROSTATE
COLON
OVARIAN
COLON
COLON
NSCLC
NSCLC
RENAL
NSCLC
RENAL
RENAL
RENAL
RENAL
RENAL
CNS
CNS
CNS

La elección del tipo de *linkage* influye de forma notable en los dendogramas obtenidos. Por lo general, *single linkage* tiende a formar *clusters* muy grandes en los que las observaciones individuales se unen una a una. El completo y average suele generar dendogramas más compensados con *clusters* más definidos, tal como ocurre en este ejemplo.

A pesar de que la agrupación no es perfecta, los *clusters* tienden a segregar bastante bien las líneas celulares procedentes de *leukemia* , *melanoma* y *renal*. Véase los aciertos cuando el dendograma se corta a una altura tal que genera 4 *clusters*.

```
clusters <- cutree(tree = hc_completo, k = 4)
table(clusters, tipo_cancer, dnn = list("clusters", "tipo de cáncer"))
```

```
##         tipo de cáncer
## clusters BREAST CNS COLON K562A-repro K562B-repro LEUKEMIA MCF7A-repro
##        1      2   3     2           0           0        0           0
##        2      3   2     0           0           0        0           0
##        3      0   0     0           1           1        6           0
##        4      2   0     5           0           0        0           1
##         tipo de cáncer
## clusters MCF7D-repro MELANOMA NSCLC OVARIAN PROSTATE RENAL UNKNOWN
##        1           0        8     8       6        2     8       1
##        2           0        0     1       0        0     1       0
##        3           0        0     0       0        0     0       0
##        4           1        0     0       0        0     0       0
```

Todas las líneas celulares de *leukemia* caen en el *cluster 3*, todas las de *melanoma* y *ovarian* en el 1. Las líneas celulares de *breast* son las que más distribuidas (heterogéneas en su perfil genético) ya que están presentes en los *clusters* 1, 2 y 4.

Nota: de los 6830 genes medidos, muchos pueden estar aportando información redundante o puede que no varíen lo suficiente como para contribuir al modelo. Con el fin de eliminar todo ese ruido y mejorar los resultados del *clustering*, es aconsejable filtrar aquellos genes cuya expresión no supere un mínimo de varianza. Del mismo modo, puede ser útil evaluar si aplicando un *Principal Component Analysis* se consigue capturar la mayor parte de la información en unas pocas componentes y utilizarlas como variables de *clustering*.

Bibliografía

Introduction to Statistical Learning, Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani

Points of Significance: Clustering, Nature Methods, Martin Krzywinski & Naomi Altman