Tema 3. Análisis de Conglomerados

Análisis de Datos Multivariantes

Grado en Matemáticas,

Doble Grado en Matemáticas y Estadística

Juan Manuel Muñoz Pichardo, Rafael Pino Mejías

Departamento de Estadística e I.O.

Universidad de Sevilla

1. Introducción. Medidas de similitud.

Dado un conjunto de *n* objetos individuales (personas, ciudades, animales, plantas, genes, tejidos, etc.), cada uno de los cuales viene descrito por un conjunto de *p* características o variables, se trata de encontrar una partición de los *n* casos en un número *k*<*n* de clases, llamadas conglomerados (*clusters*).
Se pretende que los integrantes de cada conglomerado sean lo más similares posibles, mientras que los casos de diferentes conglomerados deben ser muy diferentes.
En algunas técnicas se debe prefijar el número *k*, en otros métodos se determina a partir de los resultados proporcionados por el algoritmo.

Estos métodos han recibido una gran variedad de nombres, dependiendo del área de aplicación. En Biología Taxonomía Numérica se usa. En Psicología se emplea el término Q-análisis. En Inteligencia Artificial se usa el término de Aprendizaje no supervisado. En otras áreas se emplea Agrupación y agrupamiento. En la Estadística, el término más empleado es Análisis *Cluster* o Análisis de Conglomerados.

La búsqueda exhaustiva de todas las particiones suele ser prohibitiva:

$$S(n,k) = \frac{1}{k!}\sum_{i=1}^{k}(-1)^{k-1}\binom{k}{i}i^n$$

S(20,4)=45.232.115.901
S(100,5)=10^{68}

Medidas de distancia (similitud, disimilaridad):

La primera decisión a tomar es cómo medir la distancia o bien la similitud entre dos casos cualesquiera. En R se puede utilizar la función *dist,* que devuelve una matriz de distancias.

Función dist:

Uso: dist(x, method = "euclidean", diag = FALSE, upper = FALSE, p = 2)

method: medida a utilizar. Puede ser "euclidean", "maximum", "manhattan", "minkowski", "binary" o "canberra".

Variables cuantitativas

$$d_{euc}(X,Y) = \sqrt{\sum_i (X_i - Y_i)^2} \qquad d_{\max}(X,Y) = máx|X_i - Y_i|$$

$$d_{manh}(X,Y) = \sum_i |X_i - Y_i| \qquad d_{mkows,p}(X,Y) = \left(\sum_i |X_i - Y_i|^p \right)^{\frac{1}{p}}$$

Valores no negativos, frecuencias

$$d_{canb}(X,Y) = \sum_i \frac{|X_i - Y_i|}{|X_i + Y_i|}$$

Valores binarios (0,1): proporción de variables donde un solo caso es 1 respecto del total de casos donde al menos una variable es 1.

$$d_{bin}(X,Y) = \frac{\sum_i (X_i + Y_i)}{\sum_i X_i + \sum_i Y_i - \sum_i X_i Y_i}$$

```
> mamiferos
          agua proteina grasa lactosa
CABALLO   90.1      2.6   1.0      6.9
BURRO     90.3      1.7   1.4      6.2
BALLENA   64.8     11.1  21.2      1.6
.....................................
DELFIN    44.9     18.6  34.9      0.9
```

mamiferos
D<- dist(mamiferos)
round(D,1)

```
> round(D,1)
        CABALLO BURRO BALLENA CEBRA COBAYA RATA OVEJA RENO MULO CERDO CAMELLO BUFALO ZORRO CONEJO LLAMA CIERVO HIPOPOT. BISONTE GATO PERRO FOCA
BURRO       1.2
BALLENA    33.9  33.9
CEBRA       5.7   5.6   28.4
COBAYA     12.1  12.2   22.4   7.1
RATA       22.4  22.5   11.8  17.1  11.0
OVEJA      10.4  10.6   23.6   5.2   2.8  12.0
RENO       33.1  33.2    1.3  27.7  21.8  11.0  22.8
MULO        1.7   0.9   33.3   4.9  11.5  21.9   9.9  32.6
CERDO      10.0  10.3   24.6   5.6   2.5  12.9   2.4  23.9   9.6
CAMELLO     4.1   4.0   30.2   2.2   8.2  18.7   6.8  29.4   3.3   6.4
BUFALO     11.3  11.4   22.6   5.9   2.6  11.3   1.5  21.9  10.7   3.3   7.6
ZORRO      10.8  11.0   23.4   6.0   2.7  11.7   1.2  22.6  10.4   1.9   7.3   2.2
CONEJO     24.9  25.1   10.5  19.7  13.1   3.6  14.6   9.8  24.4  15.1  21.2  13.9  14.1
LLAMA       4.6   4.8   29.4   1.9   7.6  17.8   5.8  28.6   4.2   5.6   1.5   6.8   6.3  20.3
CIERVO     31.9  31.9    2.2  26.4  20.5   9.8  21.5   1.3  31.3  22.6  28.2  20.6  21.4   8.8  27.3
HIPOPOT.    4.8   3.8   32.4   4.9  11.3  21.5  10.0  31.8   3.3  10.0   4.1  10.4  10.8  24.1   5.4  30.5
BISONTE     4.1   4.6   30.4   3.7   8.4  18.7   6.9  29.7   4.2   6.1   2.5   8.0   7.0  21.1   1.8  28.4   6.3
GATO       12.8  13.2   22.7   8.6   3.3  11.2   4.5  22.0  12.6   3.5   9.5   4.5   3.6  12.8   8.6  20.7  13.1   8.9
PERRO      18.0  18.2   16.6  12.8   6.4   4.9   7.7  15.8  17.5   8.2  14.3   7.1   7.2   6.9  13.4  14.6  17.4  14.2   6.4
FOCA       60.7  60.6   27.9  55.1  49.8  39.5  50.7  28.6  60.1  52.0  57.1  49.7  50.8  38.3  56.4  29.7  58.7  57.6  50.3  44.3
DELFIN     59.0  59.1   25.3  53.6  47.6  36.8  48.7  26.0  58.5  49.6  55.4  47.8  48.5  34.8  54.5  27.2  57.7  55.5  47.4  41.5  11.5
```

Función *daisy* de la librería cluster:

Para situaciones donde se tiene una mezcla de variables de diferente tipo.

```
daisy(x, metric = c("euclidean", "manhattan", "gower"), stand = FALSE, type = list(), weights = rep.int(1, p))
```

- *metric*= medida de distancia, incluye el coeficiente general de similitud de Gower, que admite variables de diferente naturaleza. (Ver página siguiente).
- stand=TRUE para tipificar, se supone todas las variables cuantitativas.
- list: para especificar algunos o todos los tipos de las variables en x. Puede contener los siguientes valores: "*ordratio*" (variable de escala razón que se quiere tratar como ordinal), "*logratio*" (variable de escala razón que se debe transformar log.), "*asymm*" (binaria asimétrica) and "*symm*" (binaria simétrica).

(En una variable asimétrica binaria la categoría 1 es más importante que la categoría 0).

```
library(cluster)
data(flower)
flower
D <- daisy(flower,type = list(asymm = c("V1", "V3"), symm= 2, ordratio= 7, logratio= 8))
D
```

Coeficiente general de disimilaridad de Gower

$$d(i,j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^{p} \delta_{ij}^{(f)}} \qquad \in [0,1]$$

where

$d_{ij}^{(f)}$ = contribution of variable f to $d(i,j)$, which depends on its type:

- f binary or nominal: $d_{ij}^{(f)} = 0$ if $x_{if} = x_{jf}$, and $d_{ij}^{(f)} = 1$ otherwise,
- f interval-scaled: $d_{ij}^{(f)} = \frac{|x_{if} - x_{jf}|}{\max_h x_{hf} - \min_h x_{hf}}$,
- f ordinal or ratio-scaled: compute ranks r_{if} and $z_{if} = \frac{r_{if}-1}{\max_h r_{hf}-1}$ and treat these z_{if} as interval-scaled,

and

$\delta_{ij}^{(f)}$ = weight of variable f:

- $\delta_{ij}^{(f)} = 0$ if x_{if} or x_{jf} is missing,
- $\delta_{ij}^{(f)} = 0$ if $x_{if} = x_{jf} = 0$ and variable f is asymmetric binary,
- $\delta_{ij}^{(f)} = 1$ otherwise.

```
> flower
   V1 V2 V3 V4 V5 V6  V7 V8
1   0  1  1  4  3 15  25 15
2   1  0  0  2  1  3 150 50
3   0  1  0  3  3  1 150 50
.........................................................................
18  0  0  1  2  1  5  25 10
```

```
> round(D,1)
Dissimilarities :
     1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
2  0.9
3  0.6 0.6
4  0.4 0.5 0.6
5  0.5 0.7 0.5 0.8
6  0.3 0.7 0.4 0.5 0.5
7  0.3 0.7 0.7 0.4 0.6 0.3
8  0.5 0.5 0.7 0.3 0.6 0.7 0.5
9  0.6 0.5 0.5 0.8 0.4 0.6 0.7 0.7
10 0.6 0.5 0.5 0.5 0.5 0.4 0.7 0.6 0.5
11 0.4 0.7 0.6 0.7 0.4 0.5 0.6 0.5 0.5 0.4
12 0.5 0.6 0.6 0.5 0.5 0.5 0.7 0.4 0.4 0.4 0.3
13 0.5 0.6 0.7 0.8 0.4 0.6 0.6 0.6 0.3 0.5 0.4 0.3
14 0.3 0.6 0.6 0.4 0.6 0.4 0.5 0.5 0.5 0.3 0.3 0.2 0.4
15 0.6 0.4 0.6 0.6 0.5 0.6 0.4 0.5 0.3 0.5 0.5 0.4 0.4 0.4
16 0.7 0.4 0.7 0.3 0.8 0.5 0.5 0.6 0.7 0.4 0.7 0.6 0.7 0.4 0.4
17 0.8 0.2 0.6 0.4 0.8 0.6 0.6 0.4 0.7 0.3 0.7 0.5 0.6 0.5 0.4 0.2
18 0.5 0.5 0.8 0.5 0.6 0.8 0.6 0.2 0.5 0.8 0.5 0.6 0.6 0.6 0.5 0.8 0.6

Metric :  mixed ;  Types = A, S, A, N, O, O, T, I
Number of objects : 18
```

Principales Métodos de Análisis Conglomerados

- Métodos Jerárquicos Aglomerativos.
  - Comienza con tantos *clusters* como casos, se van uniendo hasta que todos los casos formen un cluster.
- Métodos Jerárquicos Divisivos.
  - Comienza con un *cluster* formado por todos los casos, se va subdividiendo hasta que cada caso sea un conglomerado.
- Métodos de Partición.
  - Se fija el número de conglomerados y una partición inicial, y se repite un procedimiento que en cada iteración va reasignando los casos.
- Modelos de Mixturas
  - Se supone que la muestra analizada corresponde a una mezcla o mixtura de varias distribuciones.

2. Método Jerárquico Aglomerativo.

1. Calcular la matriz de distancias entre los casos, en este momento cada caso define un conglomerado de tamaño 1
2. Identificar los dos elementos más cercanos y agruparlos en un nuevo conglomerado.
3. Recalcular la matriz de distancias entre los conglomerados .
4. Repetir los pasos 2 y 3 hasta que todos los elementos se incluyan en un solo conglomerado.

¿Cómo calcular la distancia entre conglomerados (paso 3)?

Función hclust :

hclust(D, method= "complete")

➢ *method= puede ser alguno de los siguiente, o una abreviación no ambigua: "ward", "single", "complete", "average", "mcquitty", "median" or "centroid".*

$d_{AB} = \min(d_{ij})$ *single*

$d_{AB} = \max(d_{ij})$ *complete*

$d_{AB} = \frac{1}{n_A n_B} \sum_{i \in A} \sum_{j \in B} d_{ij}$ *average*

$d_{AB} = d(\overline{A}, \overline{B})$ *centroid*

$d_{AB} = d(Med_A, Med_B)$ *median*

$d_{A \cup B, C} = \frac{d_{A,C} + d_{B,C}}{2}$ *mcquitty*

Ward: Ward argumentó que los conglomerados debían constituirse de tal manera, que al fundirse dos elementos, la pérdida de información fuera mínima. En este contexto, la cantidad de información se cuantifica como la suma de las distancias al cuadrado de cada elemento respecto al centroide del conglomerado al que pertenece. En cada paso se unen aquellos conglomerados que dan lugar a un menor incremento de la SCE, definido para *k* conglomerados como sigue, siendo *W* la matriz de sumas de cuadrados y productos dentro de los grupos, cuyos centroides se denotan por $\underline{C}_g$:

$$SCE = \sum_{g=1}^{k} \sum_{i \in g} \| \underline{x}_i - \underline{C}_g \|^2 = \sum_{g=1}^{k} \sum_{i \in g} (\underline{x}_i - \underline{C}_g)^t (\underline{x}_i - \underline{C}_g) = tr(W)$$

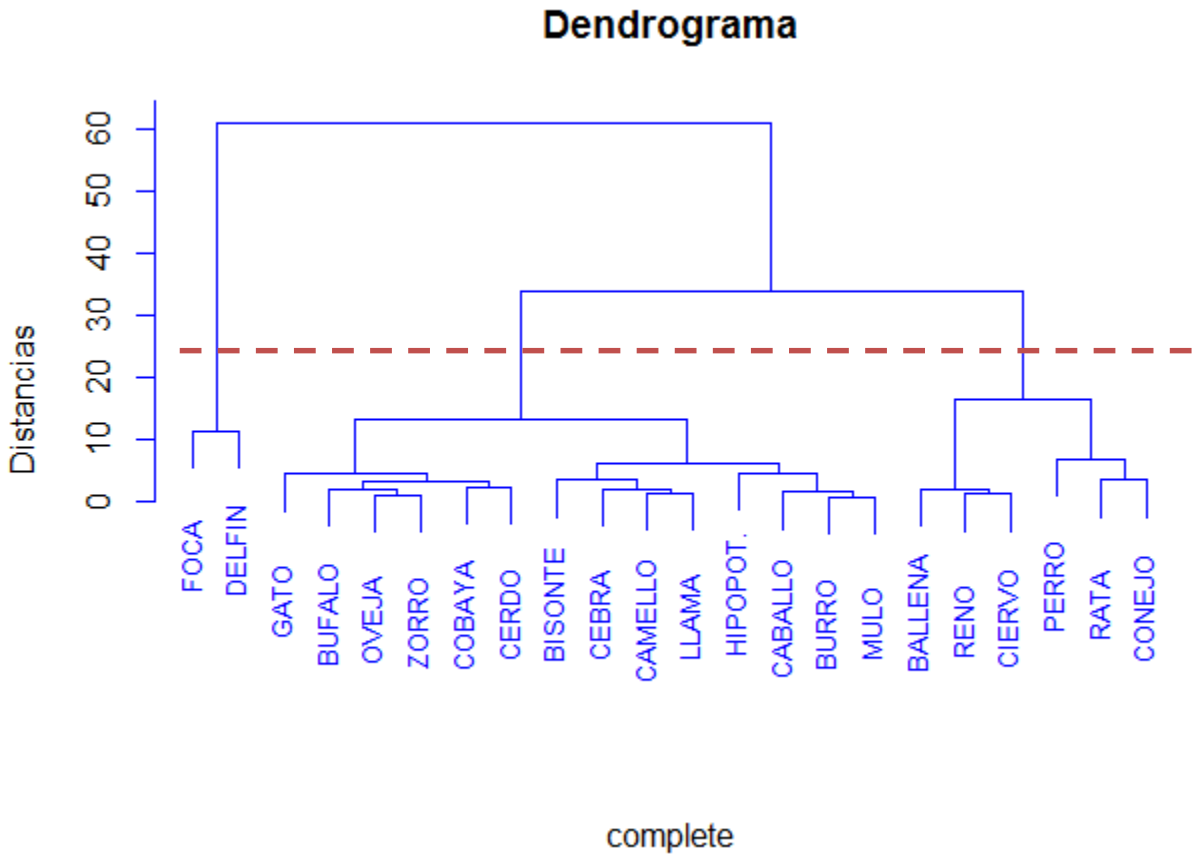
No se puede decir de antemano cuál es el mejor método, se recomienda probar con varios criterios y diferentes números de conglomerados y tener en cuenta la posible interpretación.

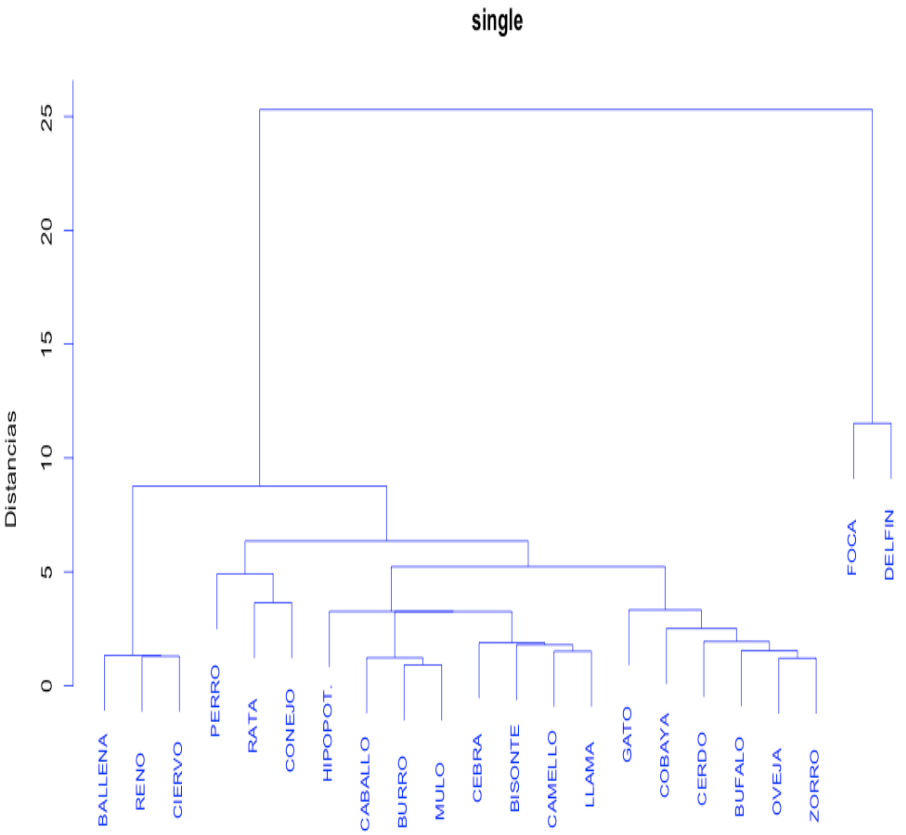
Algunas características:

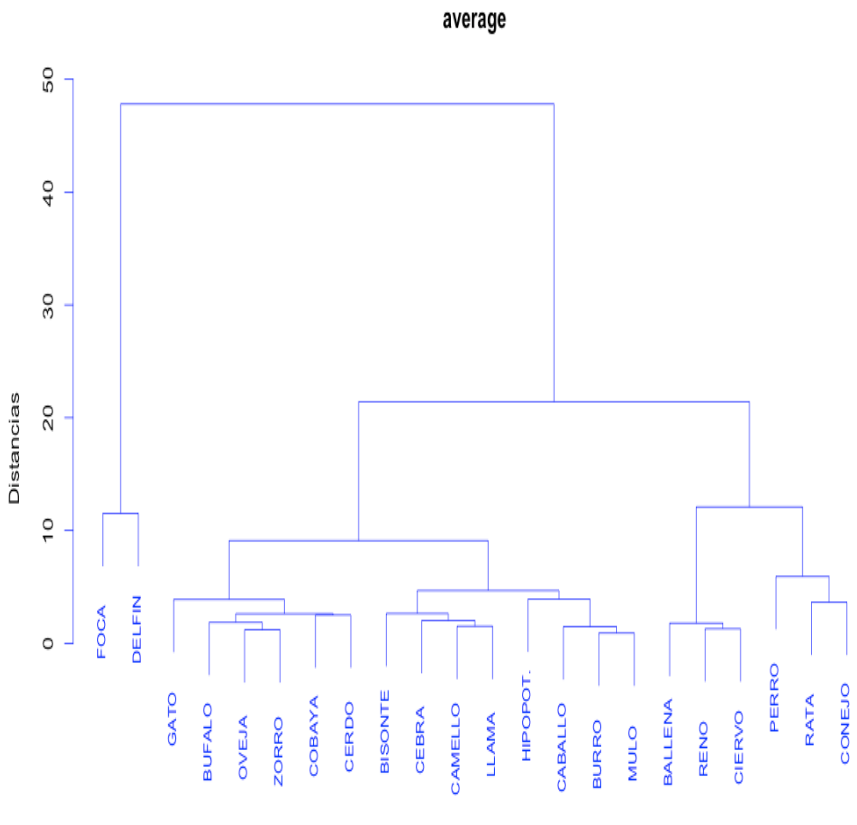
- Ward: intenta determinar conglomerados esféricos.
- Complete: busca *clusters* formados por elementos muy similares.
- Single: sensible a posibles encadenamientos (amigos de los amigos); está muy relacionado con el mínimo árbol de unión.
- Los demás métodos presentan características entre Single y Complete.
- Sin embargo, "median" y "centroid" no producen una medida de distancia monótona, de forma que en el dendrograma se pueden producir inversiones (distancias de unión que son menores en una etapa que en las anteriores), lo que dificulta notablemente la interpretación.

```
D<- dist(mamiferos)
conglo<- hclust(D)
plot(conglo, cex=0.8,labels=mamiferos$ANIMAL,main="dendrograma",
xlab="Completo",sub="",ylab="Distancias",col="blue")
cbind(conglo$merge,conglo$height) #Lista de uniones que se van produciendo
```

```
      [,1] [,2]   [,3]
 [1,]   -2   -9  0.91
 [2,]   -7  -13  1.20
 [3,]   -8  -16  1.29
 [4,]  -11  -15  1.51
 [5,]   -1    1  1.72
 [6,]   -4    4  2.17
 [7,]  -12    2  2.19
 [8,]   -3    3  2.22
 [9,]   -5  -10  2.51
[10,]    7    9  3.28
[11,]   -6  -14  3.64
[12,]  -18    6  3.67
[13,]  -19   10  4.53
[14,]  -17    5  4.75
[15,]   12   14  6.28
[16,]  -20   11  6.94
[17,]  -21  -22 11.52
[18,]   13   15 13.17
[19,]    8   16 16.56
[20,]   18   19 33.94
[21,]   17   20 60.73
```

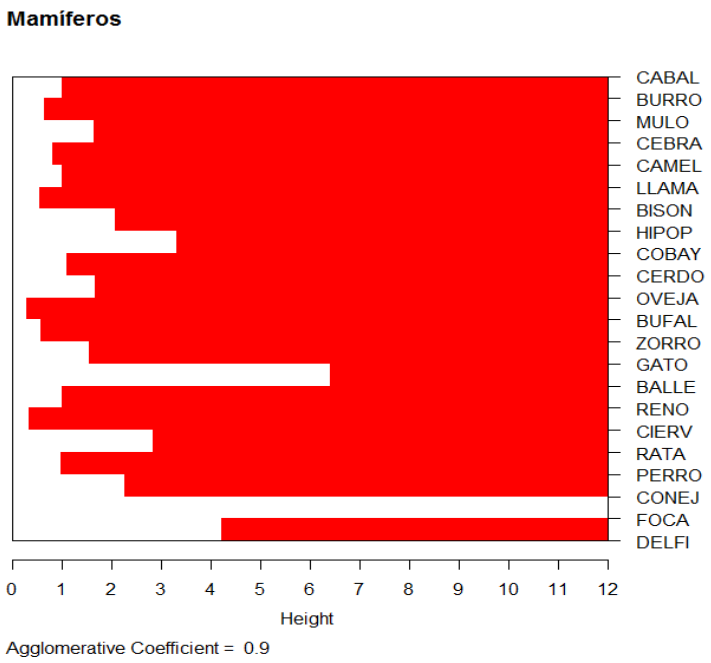


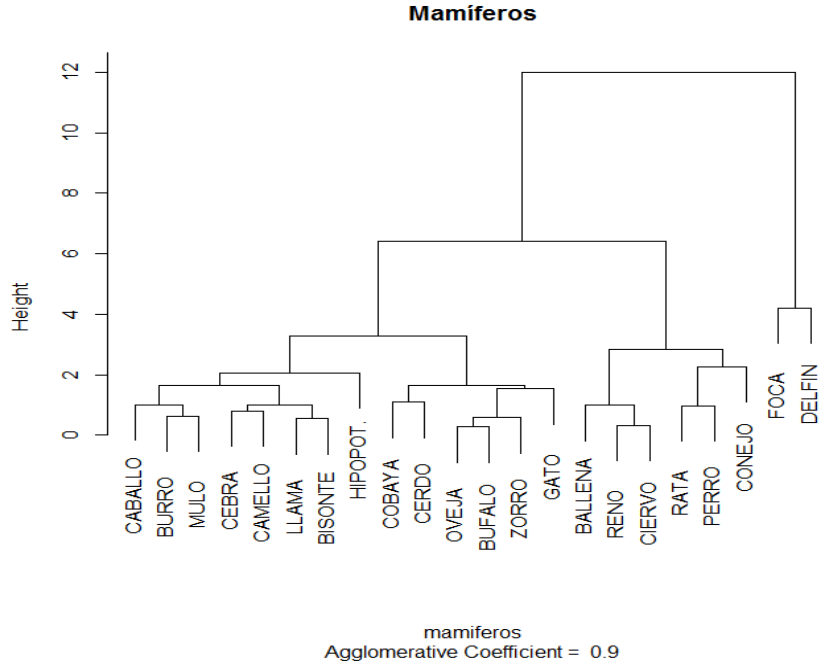

single
Distancias
0
5
10
15
20
25
BALLENA
RENO
CIERVO
PERRO
RATA
CONEJO
HIPOPOT.
CABALLO
BURRO
MULO
CEBRA
BISONTE
CAMELLO
LLAMA
GATO
COBAYA
CERDO
BUFALO
OVEJA
ZORRO
FOCA
DELFIN


average
Distancias
0
10
20
30
40
50
FOCA
DELFIN
GATO
BUFALO
OVEJA
ZORRO
COBAYA
CERDO
BISONTE
CEBRA
CAMELLO
LLAMA
HIPOPOT.
CABALLO
BURRO
MULO
BALLENA
RENO
CIERVO
PERRO
RATA
CONEJO

Librería cluster, función *agnes*:

```
library(cluster)
agnclus <- agnes(mamiferos, metric = "manhattan", stand = TRUE)
summary(agnclus)
plot(agnclus)   #Gráfico banner, dendrograma y coeficiente de aglomeración
```





Coeficiente de aglomeración: $AC = \frac{1}{n}\sum_{i=1}^{n}(1-d_i)$

d_i para cada caso, es el cociente entre la disimilaridad al primer cluster al que se unió y la disimilaridad en la última unión del algoritmo. Mientras más cercano sea AC a 1, mejor será el dendrograma obtenido. Es también la anchura media del gráfico de banderas (o proporción rellena de dicho gráfico).
Crece con el número de casos, no sirve para comparar análisis con distintos tamaños muestrales.

3. Métodos de Partición.

Se fija a priori el número deseado de conglomerados, k. Destaca en primer lugar el método de los centros móviles, también llamado algoritmo de las k-medias.

Algoritmo de k-medias:

1.- Selección inicial de *k* centros.

2.- Cada objeto se asigna al conglomerado cuyo centro esté más próximo.

3.- Recalcular los centros de los conglomerados.

4.- Repetir el paso 2 hasta que no se produzcan más reasignaciones.

Función kmeans:

kmeans(x, centers, iter.max = 10, nstart = 1, algorithm = c("Hartigan-Wong", "Lloyd", "Forgy", "MacQueen"))

- *centers= el número deseado de conglomerados o un conjunto inicial de centros*
- *iter.max=máximo número de iteraciones*
- *nstart=número de conjuntos de centros iniciales que se generan aleatoriamente (se toma el de menor suma de las distancias cuadradas de los puntos a su centro).*
- *algorithm=algoritmo concreto, se recomienda el primero, por defecto es el usado.*

Algoritmo detallado de los centros móviles (o k-medias)

- Construir *k* centros iniciales, sean

$$W_1^{(0)},\ldots,W_k^{(0)}$$

- Construir $C_1^{(0)},\ldots,C_k^{(0)}$: $C_j^{(0)}=\{X_i / \left\|X_i-W_j^{(0)}\right\| \le \left\|X_i-W_l^{(0)}\right\| \ \forall l \neq j\}$
- m=0
- Repetir
  - m=m+1
  - Calcular los centroides de la anterior partición, obteniendo así

$$W_1^{(m)},\ldots,W_k^{(m)}$$

  - Construir

$$C_1^{(m)},\ldots,C_k^{(m)}: \quad C_j^{(m)}=\{X_i / \left\|X_i-W_j^{(m)}\right\| \le \left\|X_i-W_l^{(m)}\right\| \ \forall l \neq j\}$$

- Hasta una condición de terminación, por ejemplo

$$C_l^{(m)}=C_l^{(m-1)} \quad \forall l=1,\ldots,k$$

Definición: En el contexto del algoritmo de los centros móviles, se define el error cuadrático medio o función distorsión en la etapa *m*:

$$ECM(m) = \frac{1}{2}\sum_{j=1}^{k}\sum_{\{i/X_i \in C_j^{(m)}\}} \left\| X_i - W_j^{(m)} \right\|^2$$

Proposición: Se verifica que el error cuadrático medio es una función monótona no creciente:

$$ECM(m) \geq ECM(m+1) \quad \forall m = 0,1,2,\ldots$$

Este algoritmo conduce a óptimos locales del problema de minimización de esta función objetivo.

La forma de esta función objetivo conlleva de forma implícita la suposición de que cada *cluster* tiene una distribución normal esférica.

Otros métodos, como el de los *k*-mediodes, son más robustos ante desviaciones de esta hipótesis.

Una aplicación de k-medias a la compresión de imágenes

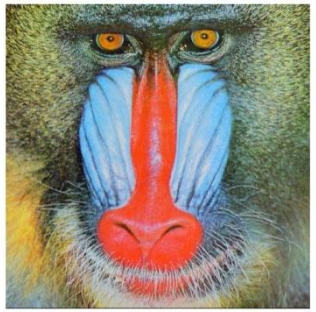
- Partimos de una imagen RGB, con resolución A filas x B columnas, cada pixel es un punto tridimensional según las componentes roja (R), verde (G), azul (B).
- Aplicar k-medias a la matriz de AxB filas y 3 columnas.
- Reemplazar cada pixel original por el centroide del cluster correspondiente.
- Tamaño original, suponiendo 8 bits para representar cada valor R, G, B: 24AB bits.
- Imagen comprimida: $AB([\log_2 k]+1)+24k$ (Identificación del centroide + Representación binaria de los centroides).
- En una imagen de 512x512 pixels, k=10: 6.291.456 bits frente a 1.048.816 bits (16.67%).

k=2

k=5

k=10

Original



Métodos de Partición

Algoritmo de "k-medioides".

Se buscan k puntos representativos de k conglomerados, llamados medioides, de modo que la suma de las distancias de los puntos a su medioide más cercano sea mínima (**función objetivo**):

$$\underset{m_1,\ldots,m_k}{Min} \sum_{i=1}^{n} \min_{t=1,\ldots k} d(i,m_t)$$

Cada caso es asignado al medioide más cercano.

Paso 1. Construir *k* medioides iniciales :

$$m_1 = \arg\min_{j=1,\ldots,n} \sum_{i=1}^{n} d(i,m_j)$$

Los restantes m_j son los puntos que decrecen la función objetivo lo máximo posible.

Paso 2. Repetir hasta la convergencia:

Considerar todos los pares de casos (*i,j*) siendo *i* un medioide y *j* un punto no medioide, e intercambiar *i* con *j* para el par que produce una mayor disminución de la función objetivo.

Este método no necesita la matriz de casos por variables, basta con la matriz de distancias.

Es un método más robusto (menos sensible a casos atípicos) que el método de *k*-medias.

```
library(cluster)
data(agriculture)
?agriculture   #x per capita GNP; y  percentage in agriculture
plot(agriculture,type="n")
text(agriculture,labels=row.names(agriculture))
agric.pam<- pam(agriculture,2)
points(agric.pam$medoids,col=c("red","blue"),lwd=3)
summary(agric.pam)
plot(agric.pam)
```

```
> agriculture
       x    y
B   16.8  2.7
DK  21.3  5.7
D   18.7  3.5
GR   5.9 22.2
E   11.4 10.9
F   17.8  6.0
IRL 10.9 14.0
I   16.6  8.5
L   21.0  3.5
NL  16.4  4.3
P    7.8 17.4
UK  14.0  2.3
```

```
Medoids:
  ID    x    y
D  3 18.7  3.5
P 11  7.8 17.4
Clustering vector:
  B  DK   D  GR   E   F IRL   I   L  NL   P  UK
  1   1   1   2   2   1   2   1   1   1   2   1
Objective function:
   build     swap      #Paso 1 y final del paso 2
3.429317 3.360610

Numerical information per cluster:
     size max_diss  av_diss diameter separation
[1,]    8 5.423099 2.891691  8.05295   5.727128
[2,]    4 7.430343 4.298448 12.56742   5.727128

Isolated clusters:
 L-clusters: character(0)
 L*-clusters: character(0)
```

clusinfo: cada fila de esta matriz se refiere a *un cluster*, y contiene:
tamaño del cluster;
disimilaridad máxima y media entre los casos del cluster y su medioide;
diámetro= máxima disimilaridad entre dos casos del *cluster*;
separación=mínima distancia entre un caso del *cluster* y un caso de otro *cluster*.

isolation: para cada cluster se indica si es aislado (L- o L*-clusters) no.
Un cluster es un L*-cluster si su diámetro es menor que su separación.
Un cluster es un L-cluster si para cada caso *i* del cluster la máxima disimilaridad entre *i* y otro caso del cluster es menor que la mínima distancia entre *i* y cualquier observación de otro cluster.
Todo L*-cluster es un L-cluster.

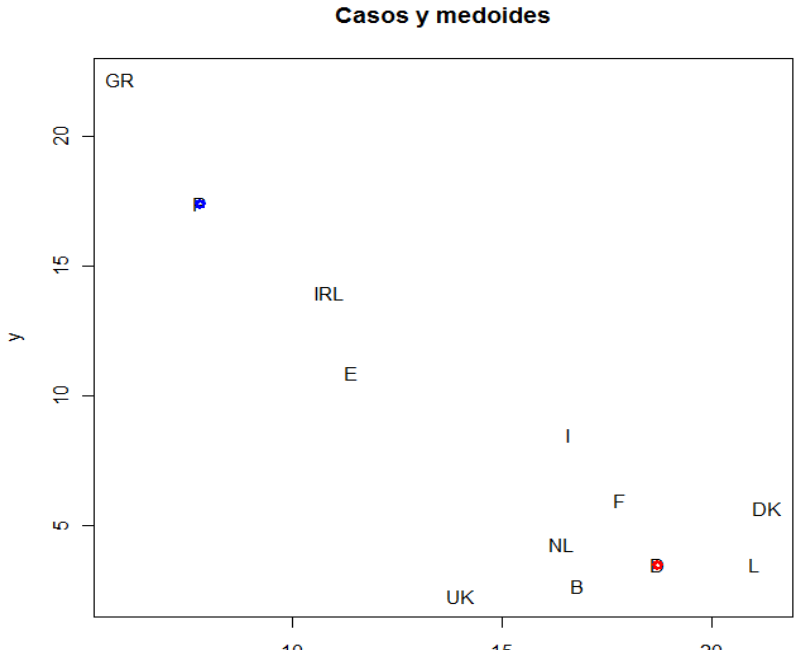
```
Silhouette plot information:
    cluster neighbor sil_width
D         1        2 0.7928036
B         1        2 0.7657461
NL        1        2 0.7607978
F         1        2 0.7459605
L         1        2 0.7442013
DK        1        2 0.7080748
UK        1        2 0.6314596
I         1        2 0.5137157
P         2        1 0.6516167
GR        2        1 0.5745566
IRL       2        1 0.5153727
E         2        1 0.1725640

Average silhouette width per cluster:
[1] 0.7078449 0.4785275
Average silhouette width of total data set:
[1] 0.6314058

66 dissimilarities, summarized :
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 1.6492  4.3569  7.9869  9.5936 13.2500 24.0350
Metric :  euclidean
Number of objects : 12
```

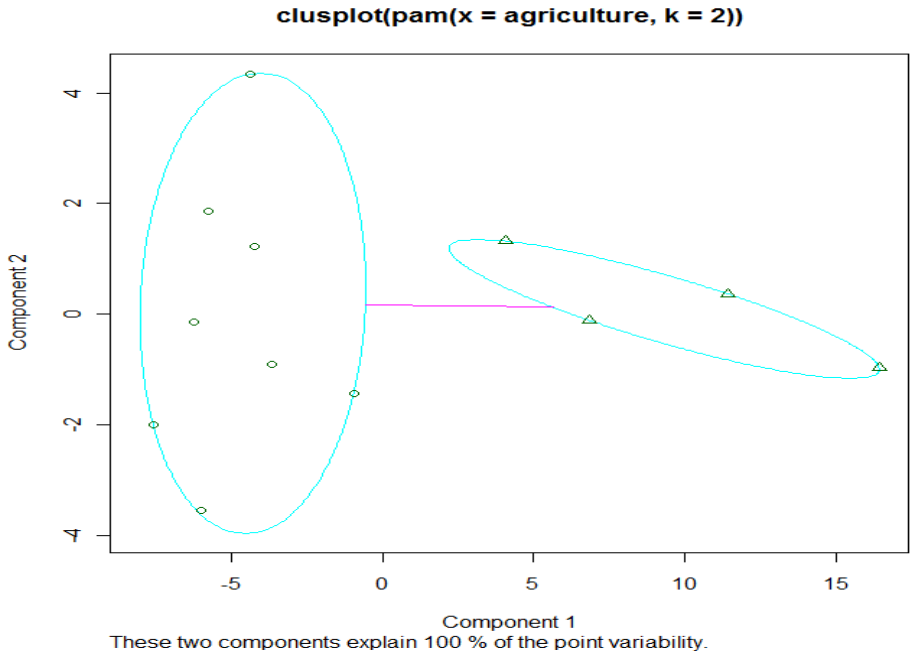
silinfo: una lista con la información sobre la *silueta* , solo cuando el número de clusters no es trivial, i.e., *1 < k < n. Es una matriz n* x 3, tal y como la devuelve silhouette(), que proporciona para cada caso *i*:
El *cluster* al que pertenece *i*;
neighbor cluster de *i*: el cluster, que no contiene a *i*, para el que la distancia media entre sus casos e *i* es mínima;
silhouette width s(i) del caso i.
s(i)= (b(i)-a(i))/max{a(i),b(i)} (toma valores entre -1 y 1)
a(i)=distancia media de *i* a los casos de su cluster
b(i)= mínimo de las distancias de *i* a otros clusters (calculada como media de las distancias de i a cada punto del cluster)

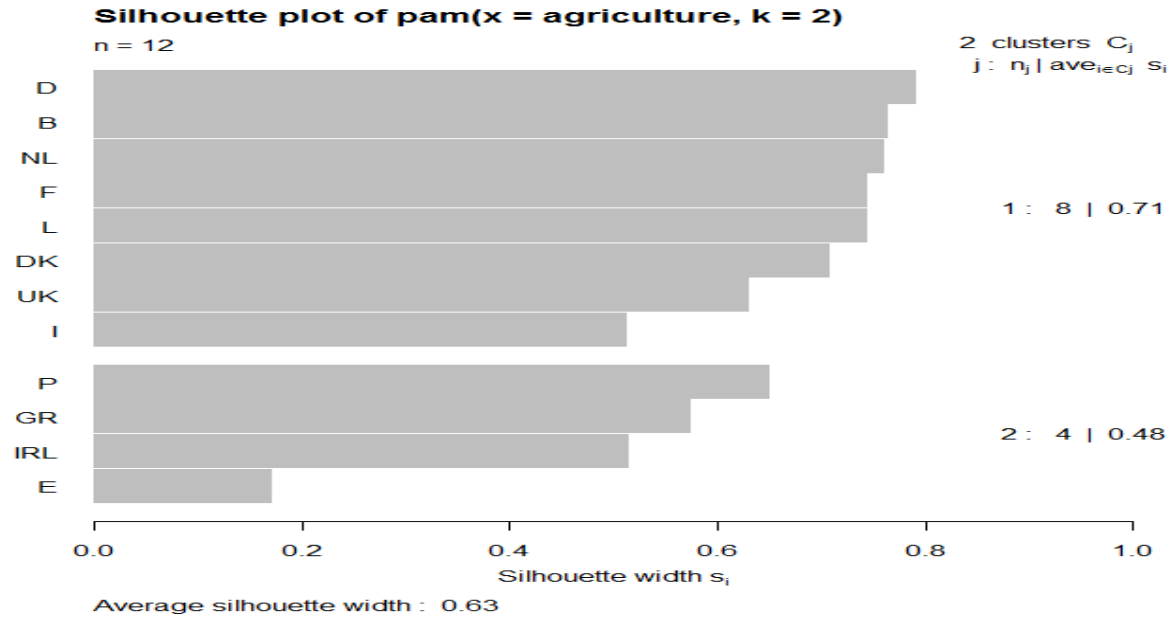
clus.avg.widths es la anchura media de silueta del cluster. *avg.width* es la anchura media de silueta del conjunto de datos.

s(i)~1: objeto bien clasificado
s(i)~0: objeto intermedio entre dos clusters
s(i)~-1: objeto mal clasificado (más cerca de otro cluster)

avg.width
0.71-1.00: Estructura fuerte
0.51-0.70: Estructura razonable
0.26-0.50: Estructura débil, probar otros métodos
0-0.25: No se ha encontrado ninguna estructura


Casos y medoides
GR
IRL
E
I
F
DK
NL
L
B
UK
y


clusplot(pam(x = agriculture, k = 2))
Component 2
Component 1
These two components explain 100 % of the point variability.


Silhouette plot of pam(x = agriculture, k = 2)
n = 12
2 clusters Cj
j : nj | avei∈cj si
D
B
NL
F
L
DK
UK
I
P
GR
IRL
E
1 : 8 | 0.71
2 : 4 | 0.48
Silhouette width si
Average silhouette width : 0.63

Función *clara* de la librería cluster:

para *n* grande, *pam* puede ser inviable al tener que almacenar la matriz de distancias completa.

La función clara trabaja con la matriz de casos y variables, y se queda con la mejor partición tras seleccionar aleatoriamente un número de muestras igual al parámetro *samples*, de tamaño *sampsize:*

Algoritmo:

Repetir "samples" veces

{ Extraer aleatoriamente una submuestra de tamaño "sampsize"

Aplicar *pam* a dicha submuestra

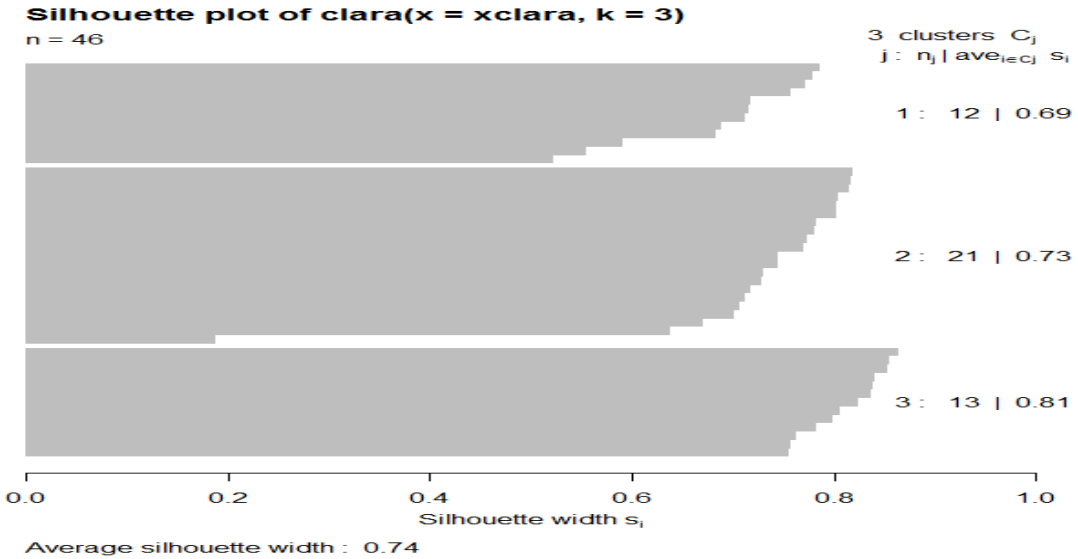
Calcular la función objetivo $\underset{m_1,\ldots m_k}{Min} \sum_{i=1}^{n} \min_{t=1,\ldots k} d(i,m_t)$

}

Elegir la partición de menor función objetivo.

(con los medioides de dicha partición se realiza el análisis de conglomerados de todos los casos)

```
data(xclara)
clx3 <- clara(xclara, 3)
plot(clx3)
```



Función *fanny* de la librería cluster:

Esta función ofrece un método *fuzzy*.

Para cada caso se calcula un *coeficiente de pertenencia* a cada cluster.

Se trata de minimizar una suma ponderada de distancias, donde los coeficientes $\{u_{iv}\}$ son los pesos para dichas distancias.

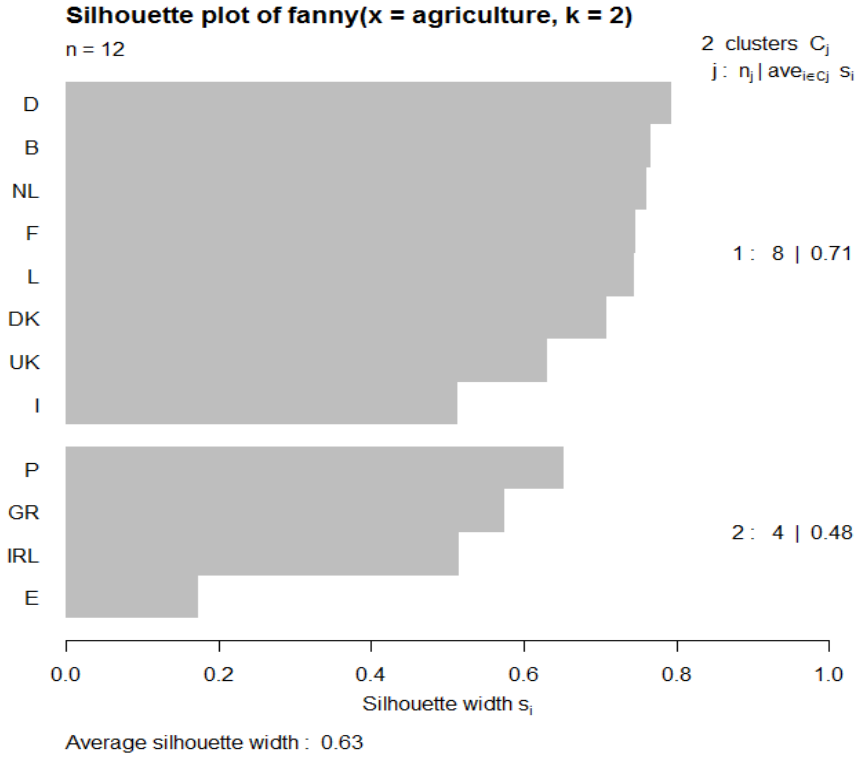
Asignando cada caso al conglomerado de mayor coeficiente se tiene una partición.

$$\underset{u_{iv}}{Min} \sum_{v=1}^{k} \frac{\sum_{i,j=1}^{n} u_{iv}^2 u_{jv}^2 d(i,j)}{2\sum_{j=1}^{n} u_{jv}^2}$$

$$s.a. \begin{cases} u_{iv} \geq 0, \ \forall i = 1,\ldots,n; v = 1,\ldots,k \\ \sum_{v=1}^{k} u_{iv} = 1, \ \forall i = 1,\ldots,n \end{cases}$$

```
agric.fan<- fanny(agriculture,2)
summary(agric.fan)
plot(agric.fan)
```

```
Membership coefficient
 (in %, rounded):
    [,1] [,2]
B     90   10
DK    85   15
D     93    7
GR    18   82
E     31   69
F     90   10
IRL   15   85
I     73   27
L     87   13
NL    91    9
P     10   90
UK    80   20
```



Otros métodos:

Divisivo:

función *diana* de la librería *cluster*.

mona, para variables binarias.

4. Análisis de Conglomerados basado en modelos de mixturas.

La función *Mclust* de la librería *mclust* elige de manera automática la mejor mixtura de K poblaciones normales multivariantes, entre cierto número de formas de mixturas.

La forma de la mixtura y el valor de K se seleccionan con la ayuda del Criterio de Información Bayesiano BIC (el más alto):

$$BIC = 2\log L(x,\hat{\theta}) - M\log(n)$$

M es el número de parámetros independientes estimados.

Definición. Se dice que un vector aleatorio *X* sigue una **distribución mixtura finita** si su función de densidad (o probabilidad) verifica

$$f(x) = \sum_{k=1}^{K} \pi_k f_k(x)$$

$$\pi_k > 0 \quad \forall\, k = 1,2,\ldots,K; \quad \sum_{k=1}^{K} \pi_k = 1; \qquad f_k(x) \geq 0 \quad \forall\, x \in \aleph \quad \int_{\aleph} f_k(x)dx = 1 \;\forall\, k = 1,2,\ldots,K$$

$\{\pi_k\}_{k=1}^{K}$ Coeficientes (pesos) de la mixtura, representan las probabilidades a priori para cada una de las k distribuciones mezcladas.

Mixtura de normales *p*-variantes:

$$f(x) = \sum_{k=1}^{K} \pi_k N_p(x;\mu_k,\Sigma_k)$$

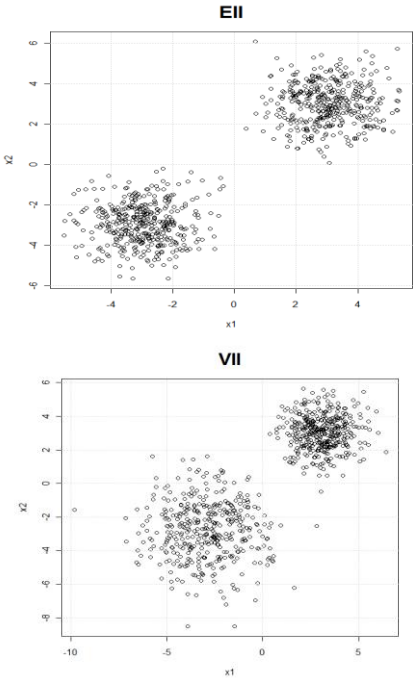
La forma de cada una de las componentes depende de la matriz de dispersión Σ_k. En *Mclust* se considera la siguiente parametrización, basada en la descomposición espectral:

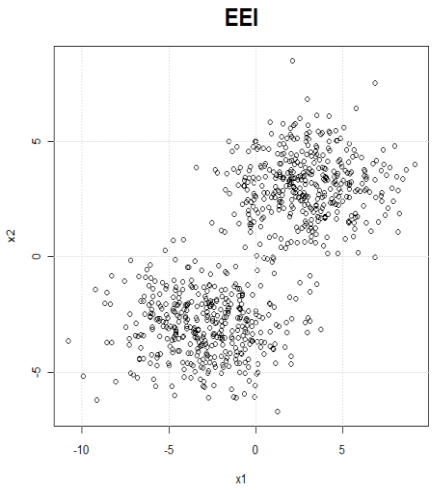
$$\Sigma_k = \lambda_k D_k A_k D_k^t$$

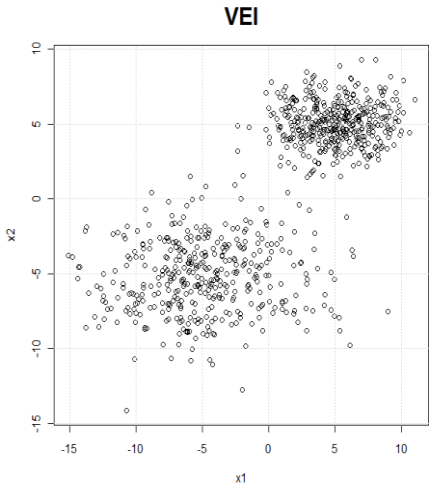
La orientación depende de D_k, la forma (elipse más o menos estrecha, o circular), depende de A_k, mientras que el volumen depende de λ_k.

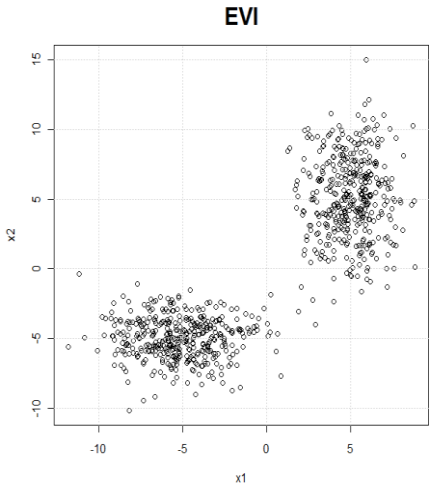
La siguiente tabla recoge los modelos de mixtura disponibles. HC indica que ese modelo puede ser usado en un procedimiento jerárquico aglomerativo que explota las características de la normal multivariante. EM indica que los parámetros se estiman mediante el algoritmo EM. Los dos primeros modelos son mixturas univariantes.

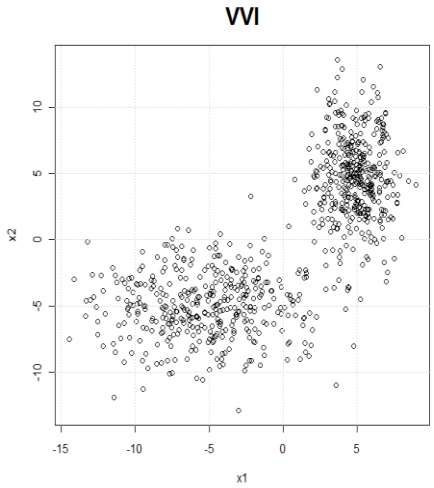
identifier	Model	HC	EM	Distribution	Volume	Shape	Orientation
E		•	•	(univariate)	equal		
V		•	•	(univariate)	variable		
EII	λI	•	•	Spherical	equal	equal	NA
VII	$\lambda_k I$	•	•	Spherical	variable	equal	NA
EEI	λA		•	Diagonal	equal	equal	coordinate axes
VEI	$\lambda_k A$		•	Diagonal	variable	equal	coordinate axes
EVI	λA_k		•	Diagonal	equal	variable	coordinate axes
VVI	$\lambda_k A_k$		•	Diagonal	variable	variable	coordinate axes
EEE	$\lambda D A D^T$	•	•	Ellipsoidal	equal	equal	equal
EEV	$\lambda D_k A D_k^T$		•	Ellipsoidal	equal	equal	variable
VEV	$\lambda_k D_k A D_k^T$		•	Ellipsoidal	variable	equal	variable
VVV	$\lambda_k D_k A_k D_k^T$	•	•	Ellipsoidal	variable	variable	variable

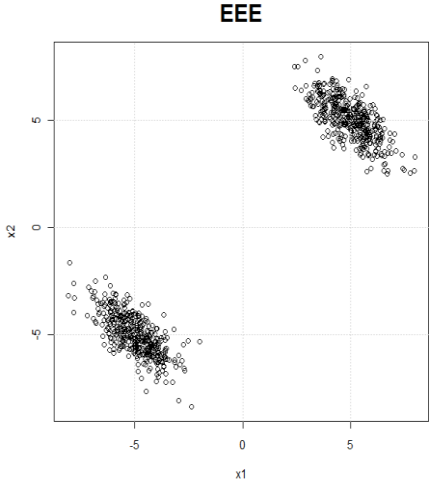


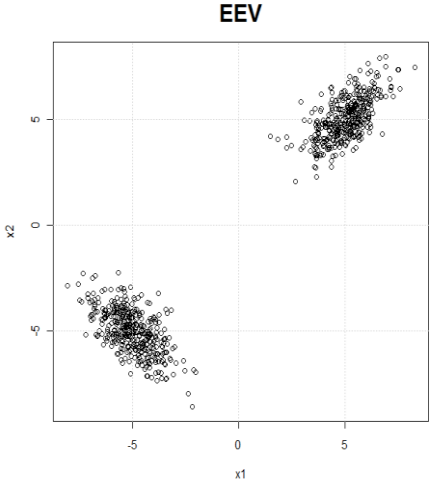

EEI
x1
x2

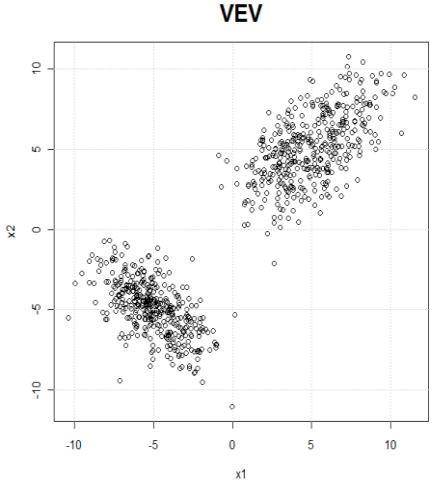

VEI
x1
x2

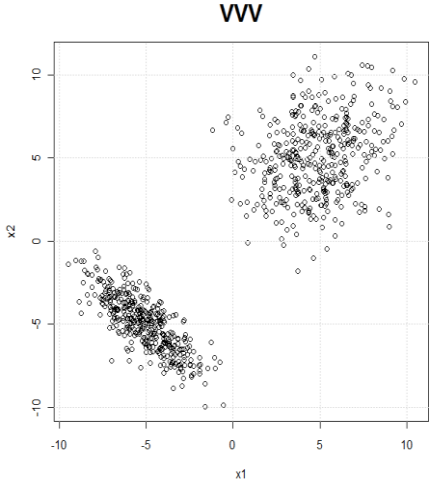

EVI
x1
x2


VVI
x1
x2


EEE
x1
x2


EEV
x1
x2


VEV
x1
x2


VVV
x1
x2

Algoritmo EM

Para aplicar el algoritmo EM, el problema se transforma en otro con valores perdidos.

Datos observados

$$X = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

Datos Completos

$$Y = \begin{bmatrix} x_1 & z_{11} & \cdots & z_{1K} \\ \vdots & \vdots & \vdots & \vdots \\ x_n & z_{n1} & \cdots & z_{nK} \end{bmatrix}$$

Variables latentes (no observadas)

$$z_{ij} = \begin{cases} 1 & x_i \sim N_p(\mu_j, \Sigma_j^2) \\ 0 & c.c. \end{cases}$$

Log-verosimilitud datos observados:

$$\ln L(X/\pi,\mu,\Sigma) = \sum_{i=1}^{n} \ln \sum_{k=1}^{K} \pi_k N_p(x_i;\mu_k,\Sigma_k)$$

Log-verosimilitud datos completos:

$$L_c(Y/\pi,\mu,\Sigma) = \prod_{i=1}^{n}\prod_{k=1}^{K} \pi_k^{z_{ik}} N_p(x_i;\mu_k,\Sigma_k)^{z_{ik}}$$

$$\ln L_c(Y/\pi,\mu,\Sigma) = \sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik}\{\ln \pi_k + \ln N_p(x_i;\mu_k,\Sigma_k)\}$$

- EM: “Expectation-Maximization”: orientado al cálculo de estimadores de máxima verosimilitud a partir de conjuntos de datos incompletos o bien en problemas donde hay unas variables latentes que pueden ser tratadas como desconocidas
- Inicial y principal contribución: Dempster, Laird y Rubin (1977) (DLR)
- Aplicaciones muy diversas: mixturas, modelos log-lineales, tratamiento de imágenes, etc
- Idea principal: en el paso E se construye un conjunto de datos completo, para el cual se pueden calcular los EMV en el paso M, y el proceso se repite cierto número de veces hasta detectar convergencia
- Datos:
  - x: datos incompletos (solo los valores observados, a consecuencia de algún patrón o esquema de valores perdidos M)
  - $y=(x,z)$: datos completos (incluye valores no observados z)

Ejemplo de datos incompletos *x*

(Muestra de tamaño $n=m+m_1+m_2$ de una distribución bivariante *W*)

Caso	W_1	W_2
1	w_{11}	w_{12}
.....	.....	.....
m	w_{m1}	w_{m2}
m+1	?	$w_{m+1,2}$
.....	....	.....
$m+m_1$	?	$w_{m+m_1,2}$
$m+m_1+1$	$w_{m+m_1+1,1}$	?
.....	.....	....
$m+m_1+m_2$	$w_{m+m_1+m_2,1}$	?

Z

$$Y(x)=\left\{\begin{array}{l} y/y=\{(w_{i1},w_{i2}),i=1,\ldots,m\}\cup\{(a_i,w_{i2}),a_i\in R,i=m+1,\ldots,n+m_1\}\cup \\ \{(w_{i1},b_i),b_i\in R,i=m+m_1+1,\ldots,m+m_1+m_2\} \end{array}\right\}$$

Dado el conjunto de datos observados *x*, se pretende maximizar la función de verosimilitud $L(\theta/x)$.

En general es más fácil trabajar con las funciones de densidad o probabilidad de y/θ o de $z/(x,\theta)$.

La densidad de *x* corresponde a la densidad marginal de la distribución de *y*:

$$L(\theta) = f_X(x/\theta) = \int_{Y(x)} f_Y(y/\theta)dy$$

La densidad de *y*: $L_c(\theta) = f_Y(y/\theta) = f_{(X,Z)}((x,z)/\theta) = f_X(x/\theta)\, f_{Z/X}(z/x,\theta)$

Por tanto la densidad de z/x: $f_{Z/X}(z/x,\theta) = \dfrac{f_Y(y/\theta)}{f_X(x/\theta)}$

Ejemplo: Si los datos de la transparencia anterior provienen de una normal bivariante:

$$\ln L(\theta) = -n\ln(2\pi) - \frac{1}{2}m\ln|\Sigma| - \frac{1}{2}\sum_{j=1}^{m}(w_j-\mu)'\Sigma^{-1}(w_j-\mu) - \frac{1}{2}\sum_{i=1}^{2}m_i\ln\sigma_i^2 - \frac{1}{2}\left\{\frac{1}{\sigma_1^2}\sum_{j=m+m_1+1}^{n}(w_{j1}-\mu_1)^2 + \frac{1}{\sigma_2^2}\sum_{j=m+1}^{m+m_1}(w_{j2}-\mu_2)^2\right\}$$

$$\ln L_c(\theta) = -n\ln(2\pi) - \frac{1}{2}n\ln|\Sigma| - \frac{1}{2}\sum_{j=1}^{n}(w_j-\mu)'\Sigma^{-1}(w_j-\mu) = -n\ln(2\pi) - \frac{1}{2}n\ln\gamma - \frac{1}{2\gamma}\left[\begin{array}{l}\sigma_2^2T_{11}+\sigma_1^2T_{22}-2\sigma_{12}T_{12}\\ -2\{T_1(\mu_1\sigma_2^2-\mu_2\sigma_{12})+T_2(\mu_2\sigma_1^2-\mu_1\sigma_{12})\}+\\ n\{\mu_1^2\sigma_2^2+\mu_2^2\sigma_1^2-2\mu_1\mu_2\sigma_{12}\}\end{array}\right]$$

$$T_i = \sum_{j=1}^{n}w_{ji}\ \ i=1,2;\ T_{hi} = \sum_{j=1}^{n}w_{jh}w_{ji}\ \ i=1,2;\ \gamma = \sigma_1^2\sigma_2^2(1-\rho^2);\ \rho = \sigma_{12}/\sigma_1\sigma_2$$

Algoritmo EM:

Es un algoritmo iterativo que intenta maximizar la función de verosimilitud $L(\theta/x)$. Se parte de un estimador inicial $\theta^{(0)}$, y en cada etapa se calcula un nuevo estimador $\theta^{(t)}$, t=0,1,2,…

Sea $Q(\theta/\theta^{(t)})$ el valor esperado de la función de verosimilitud conjunta de los datos completos, donde se tiene en cuenta que z es la parte aleatoria de y, mientras que x es fijo:

$$Q(\theta/\theta^{(t)}) = E\{\log f_Y(y/\theta)/x, \theta^{(t)}\} = \int \log\left(f_X(x/\theta) f_{Z/X}(z/x, \theta^{(t)})\right) dz$$

El algoritmo va repitiendo alternativamente los pasos E y M:

Paso E: Calcular $Q(\theta/\theta^{(t)})$

Paso M: Obtener $\theta^{(t+1)} = \arg \underset{\theta}{Max} Q(\theta/\theta^{(t)})$

Los pasos E y M se repiten hasta verificar alguna condición de terminación, por ejemplo

$$\left\|\theta^{(t+1)} - \theta^{(t)}\right\| < \varepsilon \qquad \frac{\left\|\theta^{(t+1)} - \theta^{(t)}\right\|}{\left\|\theta^{(t)}\right\|} < \varepsilon \qquad \left|Q(\theta^{(t+1)}/\theta^{(t)}) - Q(\theta^{(t)}/\theta^{(t-1)})\right| < \varepsilon$$

Algoritmo EM para la mixtura de normales multivariantes

1. Inicializar los estimadores de las medias, matrices de dispersión y pesos de las componentes.

2. Paso E tras *m* iteraciones. Calcular las "responsabilidades".

$$z_{ik}^{m} = \hat{E}[z_{ik}] = \hat{P}[z_{ik} = 1] = \frac{\pi_k^{(m)} N_p(x_i; \mu_k^{(m)}, \Sigma_k^{(m)})}{\sum_{j=1}^{K} \pi_j^{(m)} N_p(x_i; \mu_j^{(m)}, \Sigma_j^{(m)})}$$

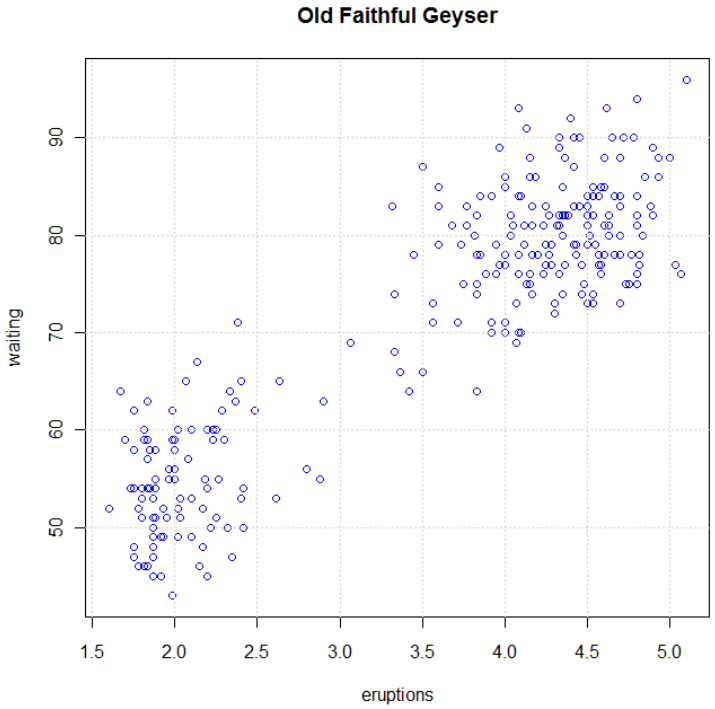
3. Paso M tras *m* iteraciones. Reestimar los parámetros:

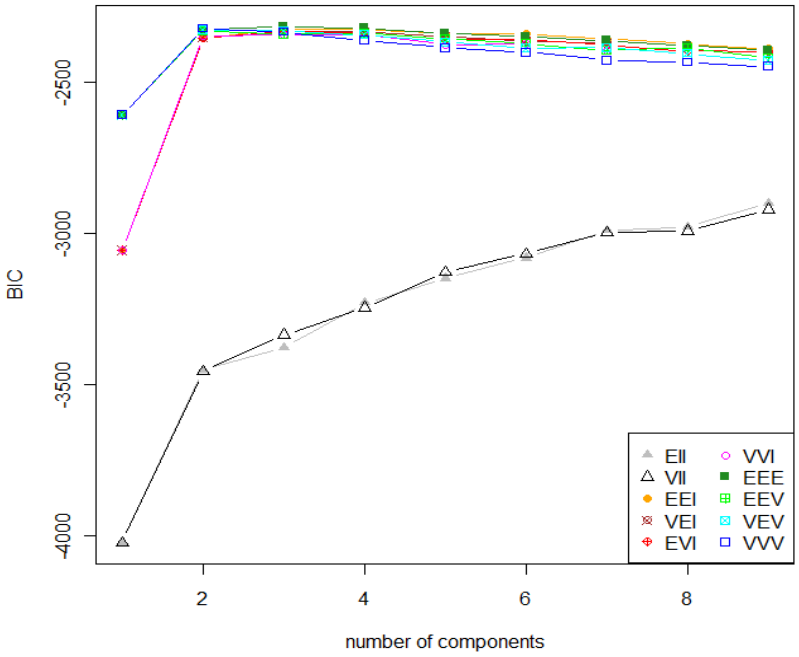
$$\mu_k^{(m+1)} = \frac{1}{\sum_{i=1}^{n} z_{ik}^{(m)}} \sum_{i=1}^{n} z_{ik}^{(m)} x_i \qquad \Sigma_k^{(m+1)} = \frac{1}{\sum_{i=1}^{n} z_{ik}^{(m)}} \sum_{i=1}^{n} z_{ik}^{(m)} \left(x_i - \mu_k^{(m+1)}\right)\left(x_i - \mu_k^{(m+1)}\right)^t \qquad \pi_k^{(m+1)} = \frac{1}{n} \sum_{i=1}^{n} z_{ik}^{(m)}$$

4. Comprobar si se cumple el criterio de convergencia. En caso contrario, ir a 2.

```
library(mclust)
data(faithful)
?faithful
plot(faithful,main="Old Faithful Geyser",col="blue")
grid()
faithMclust <- Mclust(faithful)
faithMclust
plot(faithMclust,data=faithful)
```

```
>faithMclust
 best model: elliposidal, equal variance with 3 components
```





A partir de las "responsabilidades" finales z_{ik}, se puede asignar cada caso a un conglomerado.

La **incertidumbre** es uno menos la responsabilidad de la clase asignada, es más alta en puntos intermedios entre dos o más conglomerados.