Clustering

Mathias Bourel

IMERL - Facultad de Ingeniería, Universidad de la República, Uruguay

May 22, 2019

Plan

Introducción

El objetivo del análisis de clusters es el agrupar elementos (datos, individuos, variables, etc.) en grupos (clusters) homogéneos, en función de las similitudes o diferencias entre ellos. Se arma grupos de observaciones que sean lo más parecidas entre ellas pero también lo más diferentes que observaciones de los otros grupos.

Introducción

El objetivo del análisis de clusters es el agrupar elementos (datos, individuos, variables, etc.) en grupos (clusters) homogéneos, en función de las similitudes o diferencias entre ellos. Se arma grupos de observaciones que sean lo más parecidas entre ellas pero también lo más diferentes que observaciones de los otros grupos.

El análisis de cluster es un método de *aprendizaje no supervisado* donde se dispone únicamente de valores de X y no hay etiquetas de clase que identifiquen las observaciones.

Introducción

El objetivo del análisis de clusters es el agrupar elementos (datos, individuos, variables, etc.) en grupos (clusters) homogéneos, en función de las similitudes o diferencias entre ellos. Se arma grupos de observaciones que sean lo más parecidas entre ellas pero también lo más diferentes que observaciones de los otros grupos.

El análisis de cluster es un método de *aprendizaje no supervisado* donde se dispone únicamente de valores de X y no hay etiquetas de clase que identifiquen las observaciones.

A diferencia de los problemas de clasificación, la (posible) estructura de los grupos es desconocida a priori, incluyendo el número de clases o clusters.

- Posibles aplicaciones son:
  - en marketing, para segmentar el mercado en pequeños grupos homogéneos donde realizar campañas publicitarias espécificas;
  - en biología, para dividir organismos en estructuras jerárquicas con el propósito de describir la diversidad biológica;
  - en medicina, para diseñar tratamientos específicos para distintos grupos de riesgo;
  - en psicología, para clasificar individuos en distintos tipos de personalidad, etc.
- Las técnicas de clustering esencialmente intentan extender y formalizar lo que los seres humanos observan muy bien en dos o tres dimensiones.

Introducción

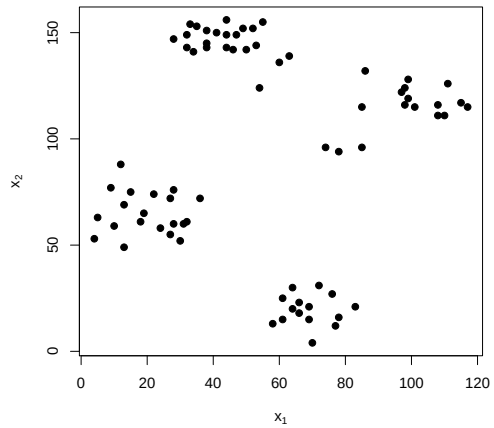


Figure: Ejemplo en dos dimensiones donde los grupos parecen "naturales".

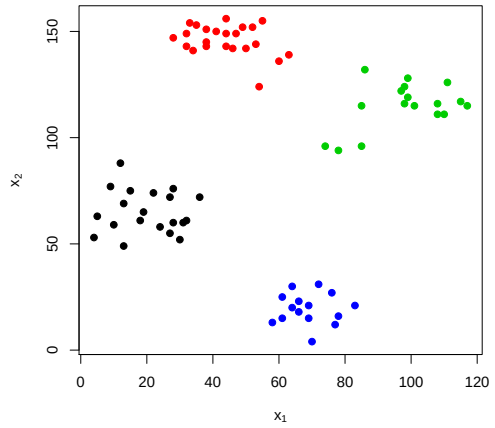


Figure: Grupos obtenidos mediante un algoritmo de clustering (PAM).

Introducción

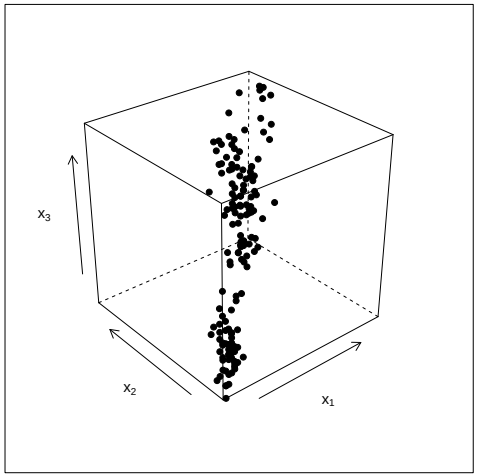


Figure: Ejemplo en tres dimensiones sobre los datos `iris`.

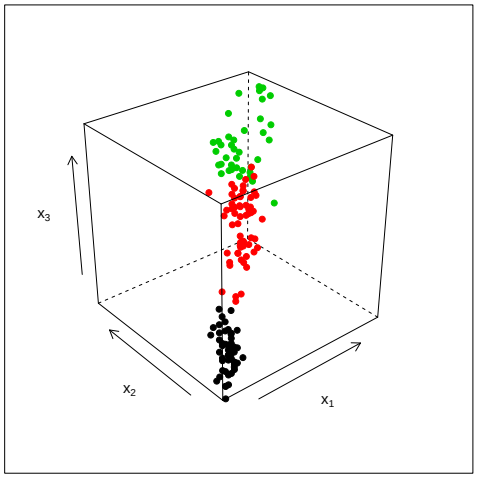


Figure: Grupos obtenidos mediante un algoritmo de clustering (PAM).

Introducción

Primeras aproximaciones:

- Encontrar grupos consiste en proyectar los datos en dos o tres dimensiones de forma tal que la estructura global se mantenga lo mejor posible y los grupos puedan ser identificados visualmente. Esto es lo que se hace con el *análisis de componentes principales* (ACP).
- Otra posibilidad consiste en representar gráficamente los datos multivariados de forma tal que puedan ser agrupados manualmente.

- Un ejemplo está dado por las *caras de Chernoff*, teniendo en cuenta que los humanos están por lo general bien entrenados para reconocer parecidos en los rostros (Dillon y Goldstein, 1984; Varmuza y Filmoser, 2009).

Introducción

$x_1 = (156, 592, 992, 263, 178, 553, 639, 75)$
$x_2 = (968, 258, 357, 957, 539, 8, 618, 132)$
$x_3 = (468, 632, 545, 834, 1, 340, 146, 329)$
$x_4 = (775, 517, 239, 113, 307, 665, 799, 763)$
$x_5 = (407, 864, 708, 519, 73, 384, 562, 740)$
$x_6 = (537, 595, 678, 221, 563, 218, 171, 654)$
$x_7 = (206, 724, 709, 570, 304, 812, 892, 870)$
$x_8 = (186, 785, 771, 674, 735, 951, 761, 29)$
$x_9 = (774, 881, 722, 613, 204, 886, 636, 451)$
$x_{10} = (193, 91, 608, 177, 751, 147, 921, 497)$
$x_{11} = (430, 560, 238, 956, 188, 924, 437, 891)$
$x_{12} = (3, 974, 528, 67, 469, 673, 311, 565)$
$x_{13} = (825, 918, 225, 949, 482, 872, 182, 550)$
$x_{14} = (820, 213, 247, 581, 78, 362, 190, 292)$
$x_{15} = (944, 464, 910, 159, 248, 371, 796, 166)$

$x_{16} = (935, 44, 817, 137, 927, 556, 358, 191)$

Alguna representación visual?

Introducción

$x_1 = (156, 592, 992, 263, 178, 553, 639, 75)$
$x_2 = (968, 258, 357, 957, 539, 8, 618, 132)$
$x_3 = (468, 632, 545, 834, 1, 340, 146, 329)$
$x_4 = (775, 517, 239, 113, 307, 665, 799, 763)$
$x_5 = (407, 864, 708, 519, 73, 384, 562, 740)$
$x_6 = (537, 595, 678, 221, 563, 218, 171, 654)$
$x_7 = (206, 724, 709, 570, 304, 812, 892, 870)$
$x_8 = (186, 785, 771, 674, 735, 951, 761, 29)$
$x_9 = (774, 881, 722, 613, 204, 886, 636, 451)$
$x_{10} = (193, 91, 608, 177, 751, 147, 921, 497)$
$x_{11} = (430, 560, 238, 956, 188, 924, 437, 891)$
$x_{12} = (3, 974, 528, 67, 469, 673, 311, 565)$
$x_{13} = (825, 918, 225, 949, 482, 872, 182, 550)$
$x_{14} = (820, 213, 247, 581, 78, 362, 190, 292)$
$x_{15} = (944, 464, 910, 159, 248, 371, 796, 166)$

$x_{16} = (935, 44, 817, 137, 927, 556, 358, 191)$

Alguna representación visual?

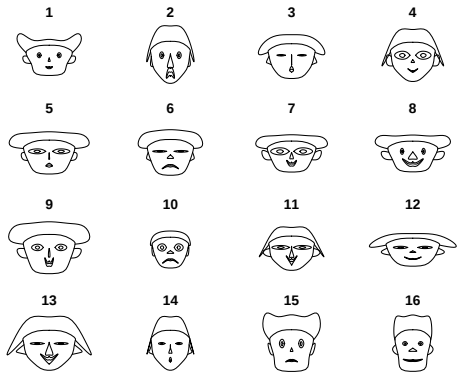
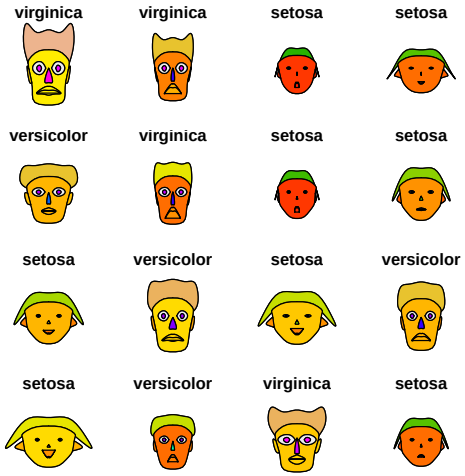


Figure: Ejemplo de representación de datos multivariados ($p = 8$) mediante *caras de Chernoff*.

Introducción

```
>library("aplpack")
>iris_sample<-iris[sample(1:dim(iris)[1],
size=16,replace=F),]
>faces(iris_sample[1:4],face.type=1,
labels=iris_sample$Species)
```



Introducción

- Describiremos técnicas "más automáticas" de clustering dado que son las más apropiadas para trabajar problemas en *grandes dimensiones* (número elevado de observaciones y/o variables).

- Nos concentraremos especialmente en estas técnicas:
  - de *particionamiento* (K-medias, PAM),
  - *jerárquicas* (aglomerativas, divisivas, Ward),
  - *basadas en modelos*,
  - *Spectral Clustering*.
- Un aspecto central en todas ellas es la noción de *similaridad* o *disimilaridad* entre los objetos a ser agrupados. Para ello es necesario en muchos casos utilizar medidas de distancia entre los datos.

Plan

Distancias.

- Una distancia $d : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ cumple:
  1. $d(\mathrm{x}, \mathrm{y}) \geq 0, \forall\, \mathrm{x}, \mathrm{y} \in \mathbb{R}^p \;\; ; \;\; d(\mathrm{x}, \mathrm{y}) = 0 \Leftrightarrow \mathrm{x} = \mathrm{y}$
  2. $d(\mathrm{y}, \mathrm{x}) = d(\mathrm{x}, \mathrm{y}) \, \forall\, \mathrm{x}, \mathrm{y} \in \mathbb{R}^p$
  3. $d(\mathrm{x}, \mathrm{y}) \leq d(\mathrm{x}, \mathrm{z}) + d(\mathrm{z}, \mathrm{y}) \, \forall\, \mathrm{x}, \mathrm{y}, \mathrm{z} \in \mathbb{R}^p$.
- Para variables cuantitativas, las distancias entre $\mathrm{x}_i = (x_{i1}, \ldots, x_{ip})$ y $\mathrm{x}_j = (x_{j1}, \ldots, x_{jp}) \in \mathbb{R}^p$ más utilizadas son:
  - *Euclídea* o L_2: $\mathrm{d}(\mathrm{x}_i, \mathrm{x}_j) = \left(\sum_{k=1}^{p} (x_{ik} - x_{jk})^2\right)^{1/2}$,
  - *Manhattan* o L_1: $\mathrm{d}(\mathrm{x}_i, \mathrm{x}_j) = \sum_{k=1}^{p} |x_{ik} - x_{jk}|$,
  - *Minkowski* o L_q $(q \geq 1)$: $\mathrm{d}(\mathrm{x}_i, \mathrm{x}_j) = \left(\sum_{k=1}^{p} (x_{ik} - x_{jk})^q\right)^{1/q}$
- La distancia euclídea es la más utilizada en la práctica. La distancia L_1 es más robusta frente a la presencia de datos atípicos (*outliers*). La distancia de Minkowski es una generalización de las otras dos.
- Estas medidas de distancia no son invariantes frente a cambios de escala de las variables. Por eso, buscaremos estandarizar los datos previamente (media cero y variancia 1).
- Una disimilaridad es una función $d : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ que verifica las propiedades 1 y 2 de distancia.

```
> (x1 <- c(-1, 2, 3.5, sqrt(2), -5))
[1] -1.000000  2.000000  3.500000  1.414214 -5.000000
> (x2 <- c(2, 8, 6, pi, -5))
[1]  2.000000  8.000000  6.000000  3.141593 -5.000000
>
> # distancia 'euclidea' o L2
> sqrt(sum( (x1 - x2)^2 ))
[1] 7.364363

> # distancia 'Manhattan' o L1
> sum( abs(x1 - x2) )
[1] 13.22738

> # distancia 'Minkowski' o Lq
> q <- 2.5 # por ejemplo
> (sum( (abs(x1 - x2))^q))^(1/q)
[1] 6.731693
```

Distancias.

- Existen otras distancias, como la de *Mahalanobis*, para variables cuantitativas:

$$d(\mathrm{x}, \mathrm{y}) = (\mathrm{x} - \mu)' \Sigma^{-1} (\mathrm{y} - \mu)$$

  donde Σ es la matriz de varianzas-covarianzas. La misma toma en cuenta las correlaciones entre las variables. Distintas distancias a menudo dan resultados diferentes (incluso para una misma técnica).

- Para otro tipo de variables (categóricas, ordinales o nominales), se utilizan por lo general otro tipo de distancias (basadas en el número de coincidencias o discordancias, por ejemplo) (Varmuza y Filmoser, 2009).

- La elección de la medida de distancia a utilizar puede ser tan o más importante que la técnica de cluster utilizada (Hastie y otros, 2009).
- Sin embargo, esta elección no es sencilla y requiere por lo general conocimiento específico del problema a resolver.

- Dadas n observaciones $\mathrm{x}_1, \ldots, \mathrm{x}_n \in \mathbb{R}^p$, algunas técnicas de clustering requieren el cálculo de las distancias entre todo par de observaciones $(\mathrm{x}_i, \mathrm{x}_j)$.

- Esta información puede representarse mediante una matriz de distancias, de dimensión $n \times n$, $D = ((\mathrm{d}_{ij}))$, simétrica, donde $\mathrm{d}_{ij} = \mathrm{d}(\mathrm{x}_i, \mathrm{x}_j)$.

Ejemplo numérico en R

```
> X <- matrix(sample(-5:5, 200, rep = T), ncol = 20)

> dim(X)
[1] 10 20

> head(X, 5) # primeras 5 filas

     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14]
[1,]    4    2    4    5   -4   -1   -1   -2    3     2    -1    -4    -4    -1
[2,]   -1    2    2    5   -3    5   -2   -4    3    -5    -1     4     0    -3
[3,]    4   -5    1   -3   -5    2    2    5    3     4    -3     2     5    -5
[4,]   -2   -1    5    4   -5   -4    0    2   -2    -5     3    -4    -3     0
[5,]   -3    5   -3   -2   -4   -2    2   -1    5     0     2     0    -4     3

     [,15] [,16] [,17] [,18] [,19] [,20]
[1,]    -2    -4    -1    -1     2    -4
[2,]    -5     0    -3     4    -1     3
[3,]     2    -3     3     4     2    -3
[4,]     0     4    -4    -2    -2     3
[5,]    -2    -3     3    -4    -3    -5



> round(dist(X, 'euclidean', diag = T), 3)
       1      2      3      4      5      6      7      8      9     10
1   0.000
2  17.776  0.000
3  19.748 22.494  0.000
4  17.578 18.628 24.960  0.000
5  16.371 21.307 22.847 20.664  0.000
6  19.000 20.952 19.570 22.450 21.331  0.000
7  19.442 19.799 20.396 20.712 20.248 21.424  0.000
8  14.491 19.545 25.219 21.190 16.613 20.372 23.409  0.000
9  19.287 21.260 21.166 20.421 19.647 27.368 20.543 21.260  0.000
10 18.138 22.694 17.349 20.100 15.330 18.708 21.703 19.975 18.735  0.000
```

Plan

Descomposición de la inercia

Cada cluster C_k está caracterizado por:

- Su centro μ_k o su centro de gravedad: $\overline{x}_k = \frac{1}{n_k} \sum_{i \in C_k} x_i$ con $n_k = |C_k|$.
- Su inercia: $J_k = \sum_{i \in C_k} d^2(x_i, \mu_k)$. La inercia mide la concentración alrededor de μ_k. Si J_k es chica, la dispersión alrededor de $\overline{x}_k$ es chica.
- Su matriz de varianza-covarianza: $\Sigma_k = \sum_{i \in C_k} (x_i - \mu_k)(x_i - \mu_k)'$ Observar que $J_k = \text{traza}(\Sigma_k)$.

La inercia intra-cluster (within) se define como $J_w = \sum_{k=1}^{K} \sum_{i \in C_k} d^2(x_i, \mu_k) = \sum_{i \in C_k} J_k$

La inercia inter-cluster (between) se define como $J_b = \sum_{k=1}^{K} n_k d^2(\mu_k, \mu)$ y mide la separación de los centros de los clusters entre ellos. Más esta inercia es grande, más estos clusters están separados.

La matriz de covarianzas inter-cluster es $\Sigma_b = \sum_{k}^{K} (\mu_k - \mu)(\mu_k - \mu)'$ y $J_b = \text{traza}\,(\Sigma_b)$

Descomposición de la inercia

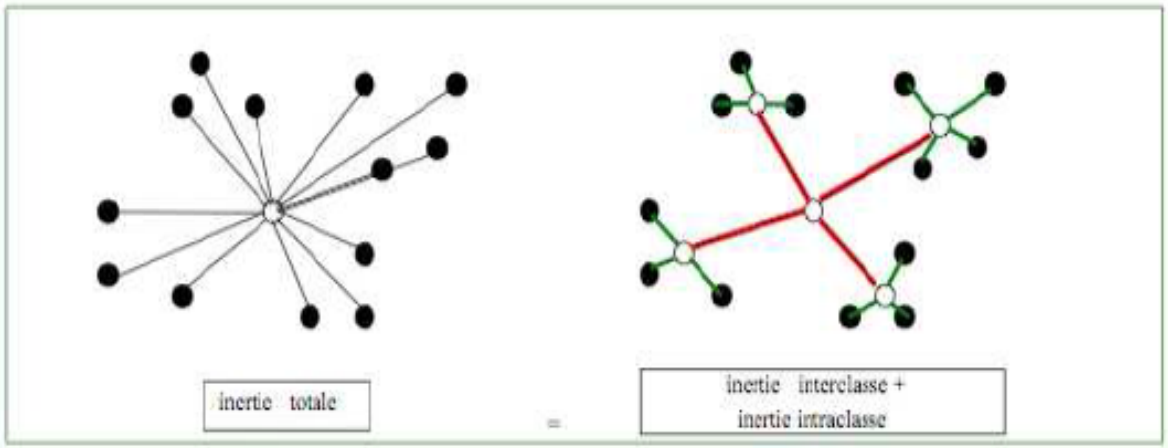
Si $\mu = \frac{1}{n}\sum_i x_i = \overline{x}$, se puede descomponer la suma de cuadrados de la siguiente manera:

$$\sum_{i=1}^{n}(x_i - \overline{x})^2 = \sum_{k=1}^{K}\sum_{i=1}^{n} z_{ik}(x_i - \overline{x_k})^2 + \sum_{k=1}^{K} n_k(\overline{x_k} - \overline{x})^2$$

$$= \sum_{k=1}^{K}\sum_{i=1}^{n_k}(x_{i(k)} - \overline{x_k})^2 + \sum_{k=1}^{K} n_k(\overline{x_k} - \overline{x})^2$$

donde $z_{ik} = \begin{cases} 1 & \text{si } x_i \in \text{ cluster } C_k \\ 0 & \text{si } x_i \notin \text{ cluster } C_k \end{cases}$

Observar que es parecido a $\text{Var}[X] = \text{E}[\text{Var}[X \mid K]] + \text{Var}[\text{E}[X \mid K]]$



Descomposición de la varianza

La cuenta anterior se obtiene de:

$$\begin{aligned}
\sum_i (x_i - \bar{x})^2 &= \sum_i \sum_k z_{ik}(x_i - \bar{x})^2 \\
&= \sum_i \sum_k z_{ik}(x_i - \bar{x}_k + \bar{x}_k - \bar{x})^2 \\
&= \sum_k \sum_i z_{ik} \left((x_i - \bar{x}_k)^2 + 2(x_i - \bar{x}_k)(\bar{x}_k - \bar{x}) + (\bar{x}_k - \bar{x})^2 \right) \\
&= \sum_k \sum_i z_{ik}(x_i - \bar{x}_k)^2 + 2 \sum_k \sum_i z_{ik}(x_i - \bar{x}_k)(\bar{x}_k - \bar{x}) + \sum_k \sum_i z_{ik}(\bar{x}_k - \bar{x})^2 \\
&= \sum_k \sum_i z_{ik}(x_i - \bar{x}_k)^2 + 2 \sum_k (\bar{x}_k - \bar{x}) \left(\sum_i z_{ik} x_i - \sum_i z_{ik} \bar{x}_k \right) + \sum_k n_k (\bar{x}_k - \bar{x} \\
&= \sum_k \sum_i z_{ik}(x_i - \bar{x}_k)^2 + 2 \sum_k (\bar{x}_k - \bar{x}) \left(n_k \bar{x}_k - n_k \bar{x}_k \right) + \sum_k n_k (\bar{x}_k - \bar{x})^2 \\
&= \sum_k \sum_i z_{ik}(x_i - \bar{x}_k)^2 + \sum_k n_k (\bar{x}_k - \bar{x})^2 .
\end{aligned}$$

Descomposición de la varianza

Ahora consideramos la suma sobre todas las variables $j = 1, \ldots, p$:

$$T = SCTotal = \sum_{i=1}^{n} \underbrace{\sum_{j=1}^{p} (x_{ij} - \overline{x}_j)^2}_{d^2(\overline{x}_i, \overline{x})}$$

$$W = SCDentro = \sum_{k=1}^{K} \sum_{i=1}^{n_k} \underbrace{\sum_{j=1}^{p} (x_{ij(k)} - \overline{x}_{kj})^2}_{d^2(x_{i(k)}, \overline{x}_k)} \qquad B = SCEntre = \sum_{k=1}^{K} n_k \underbrace{\sum_{j=1}^{p} (\overline{x}_{kj} - \overline{x}_j)^2}_{d^2(\overline{x}_k, \overline{x})}$$

Entonces

$$T = W(\mathcal{C}) + B(\mathcal{C})$$

Observar que T es constante (depende de los datos) mientras W y B dependen de los clusters que se eligen.

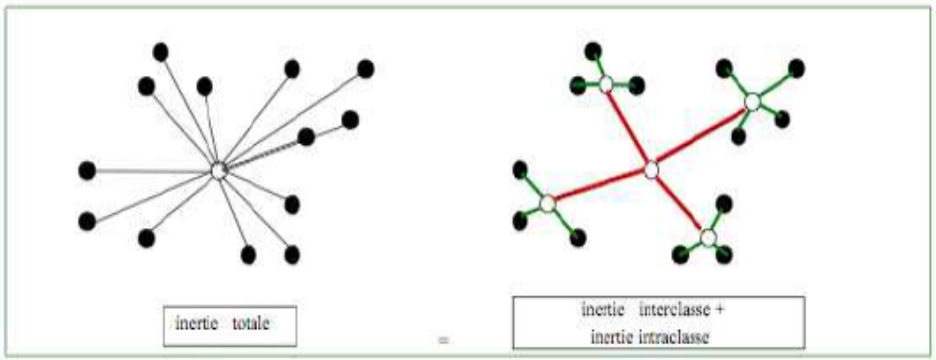
Técnicas de particionamiento

- Estas técnicas buscan particionar el conjunto de datos en un número especificado de grupos K, $1 \leq K \leq n$, minimizando algún criterio o función objetiva que indica la "bondad" (en términos del objetivo del clustering) de cada partición.

- El enfoque más utilizado consiste en buscar la partición $\mathcal{C} = \{C_1, \ldots, C_K\}$ de n individuos en K grupos de forma tal de minimizar la suma de las distancias respecto al *centroide* del grupo, μ_k,:

$$SCD(\mathcal{C}) = \sum_{k=1}^{K} \sum_{x_i \in C_k} d^2(x_i, \mu_k)$$

- Cuando la distancia utilizada es la euclidea y el centroide es el promedio de las observaciones del grupo, éste es el criterio de *mínima suma de cuadrados* dentro de los grupos, o sea $SCD(\mathcal{C}) = W(\mathcal{C})$.
- En este caso, ya que T es constante, mínimizar W es equivalente en maximizar B.



Técnicas de particionamiento

- El problema parece entonces relativamente simple: considerar todas las particiones posibles de n individuos en K grupos y seleccionar aquella con el valor más bajo de $SCD(\mathcal{C})$.

- Desafortunadamente, en la práctica esta solución no es viable. El número total de particiones a considerar, $S(n, K)$ (equivalente a buscar la cantidad de funciones sobreyectivas de un conjunto de n elementos en otro de K elementos), es por lo general tan elevado que la enumeración completa es imposible incluso para la computadora más rápida!

- Se puede demostrar que:

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

(Hastie y otros, 2009).

n	K	$S(n, K)$
15	3	2375101
20	4	$\approx 4.52 \times 10^{10}$
25	8	$\approx 6.9 \times 10^{17}$
100	5	$\approx 6.57 \times 10^{67}$

Table: Número de posibles particiones según la cantidad de observaciones n y grupos K.

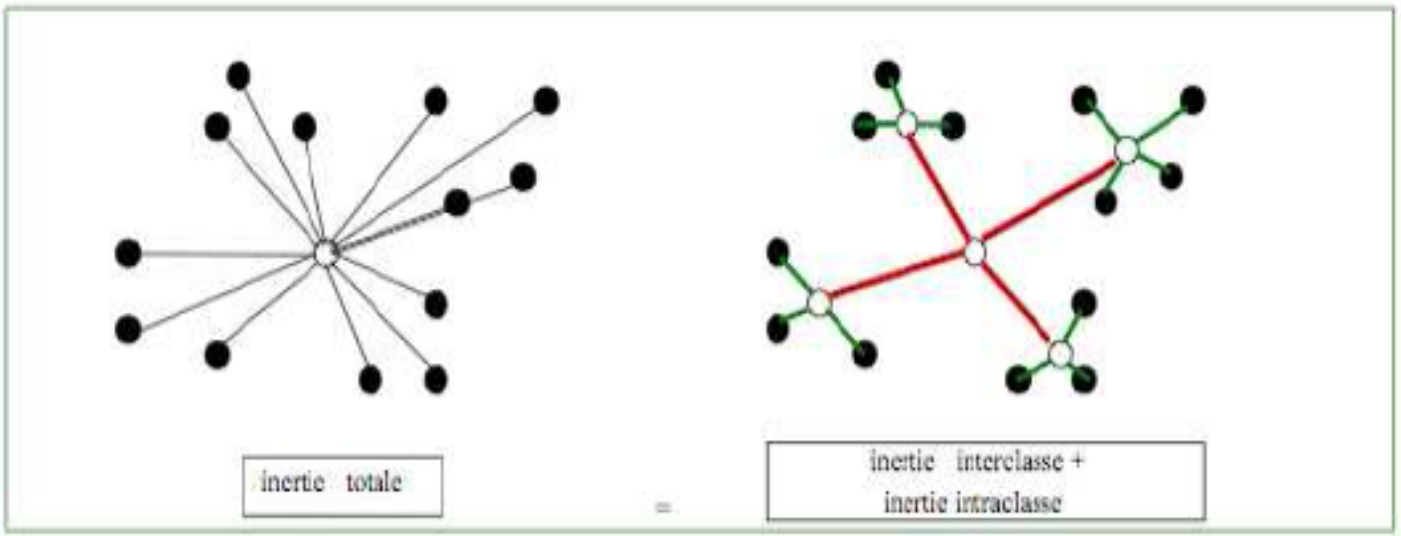
Técnicas de particionamiento

Inercia total: $I_T = \sum_{i=1}^{n} d^2(\mathbf{x}_i, \bar{\mathbf{x}}) \left(= 2n \sum_{i,i'=1}^{n} d^2(\mathbf{x}_i, \mathbf{x}_{i'}) \right)$

Inercia intra clase (within): $I_W(\mathcal{C}) = \sum_{k=1}^{K} n_k \sum_{i \in C_k} d^2(\mathbf{x}_i, \bar{\mathbf{x}}_k) \left(= 2 \sum_{k=1}^{K} \sum_{i,i' \in C_k} d^2(\mathbf{x}_i, \mathbf{x}_{i'}) \right)$

Inercia entre clases (between): $I_B(\mathcal{C}) = \sum_{k=1}^{K} n_k d^2(\bar{\mathbf{x}}_k, \bar{\mathbf{x}}) \left(= 2 \sum_{k=1}^{K} \sum_{i \in C_k} \sum_{i' \notin C_k} d^2(\mathbf{x}_i, \mathbf{x}_{i'}) \right)$

$$I_T = I_W(\mathcal{C}) + I_B(\mathcal{C})$$



Técnicas de particionamiento

- Estas estrategias están basadas en algoritmos iterativos del tipo "greedy".
- Se especifica (o sortea) una partición inicial y en cada iteración se modifica la asignación de grupos de forma tal que el valor de la función objetivo disminuya.
- El objetivo es identificar y explorar un pequeño subconjunto de las particiones posibles que contenga la mejor o, al menos, una buena partición subóptima.
- Estos algoritmos si bien pueden ser muy eficientes computacionalmente, no garantizan encontrar el óptimo global de $SCD(\mathcal{C})$ y pueden ser dependientes de la partición inicial elegida.
- La dificultad del problema y de encontrar algoritmos generales que sean eficientes, así como la gran disponibilidad de datos que existe actualmente, explican la cantidad de algoritmos que existen y de líneas de investigación abiertas.

Técnicas de particionamiento: K-medias

- El algoritmo de K-medias es un algoritmo particional y fue propuesto en los '50 (Jain, 2009)
- A pesar de que su primera aparición es desde hace más de 50 años sigue siendo de los algoritmos más utilizados para clustering por su facilidad de implementación, simpleza y buenos resultados empíricos.

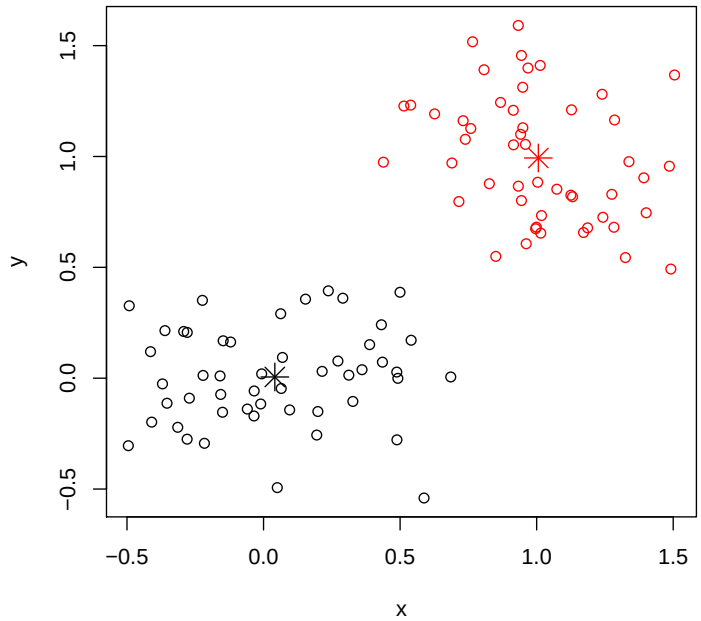
- Initialización : Se elige aleatoriamente K puntos del espacio (serán los centros de los grupos) o una partición inicial aleatoria con K clusters (son equivalentes).
- Se iteran las dos etapas siguientes hasta que se cumpla que el criterio de minimización de la inercia intraclases se estabilice o hasta una cantidad de iteraciones fijas:
  1. Todos los individuos están asignados a una clase cuyo centro es el más cercano (respecto de la distancia elegida). Se construyen entonces K clases.
  2. Se calculan los baricentros (centroides) de las clases creadas que son los nuevos centros que se consideraran en la etapa siguiente.

Figure: Algoritmo K-means

El algoritmo K-medias requiere del usuario los siguientes parámetros: número de clusters, inicialización de los clusters (centros), distancia (en general la distancia euclídea, es la distancia que usa R).

Técnicas de particionamiento: K-medias

```
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
matrix(rnorm(100, mean = 1, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")
(cl <- kmeans(x, 2,iter.max=2))
plot(x, col = cl$cluster)
points(cl$centers, col = 1:2, pch = 8, cex = 2)
```



Grupos obtenidos en la iteración 1 de K-medias

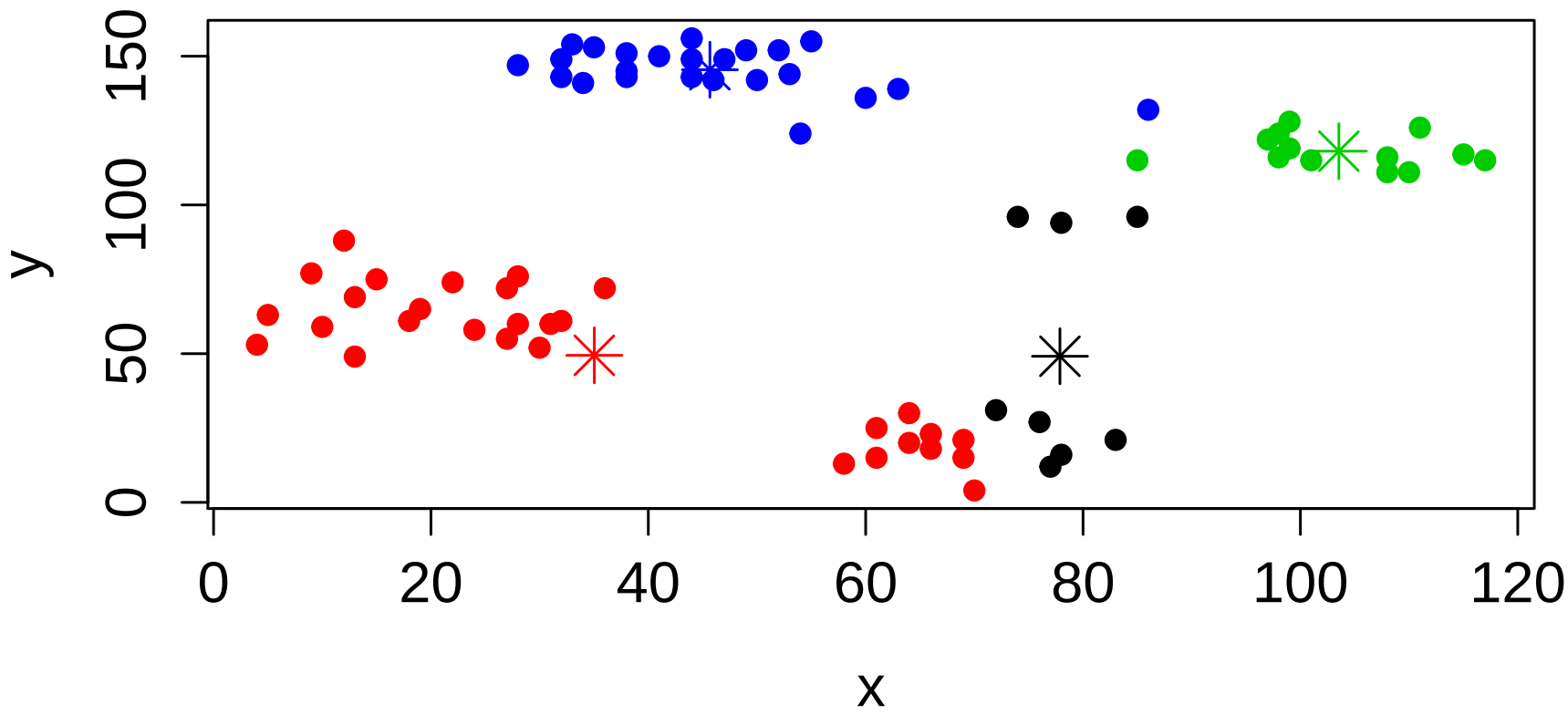


Figure: Grupos obtenidos en la iteración 1 de K-medias.

Grupos obtenidos en la iteración 2 de K-medias

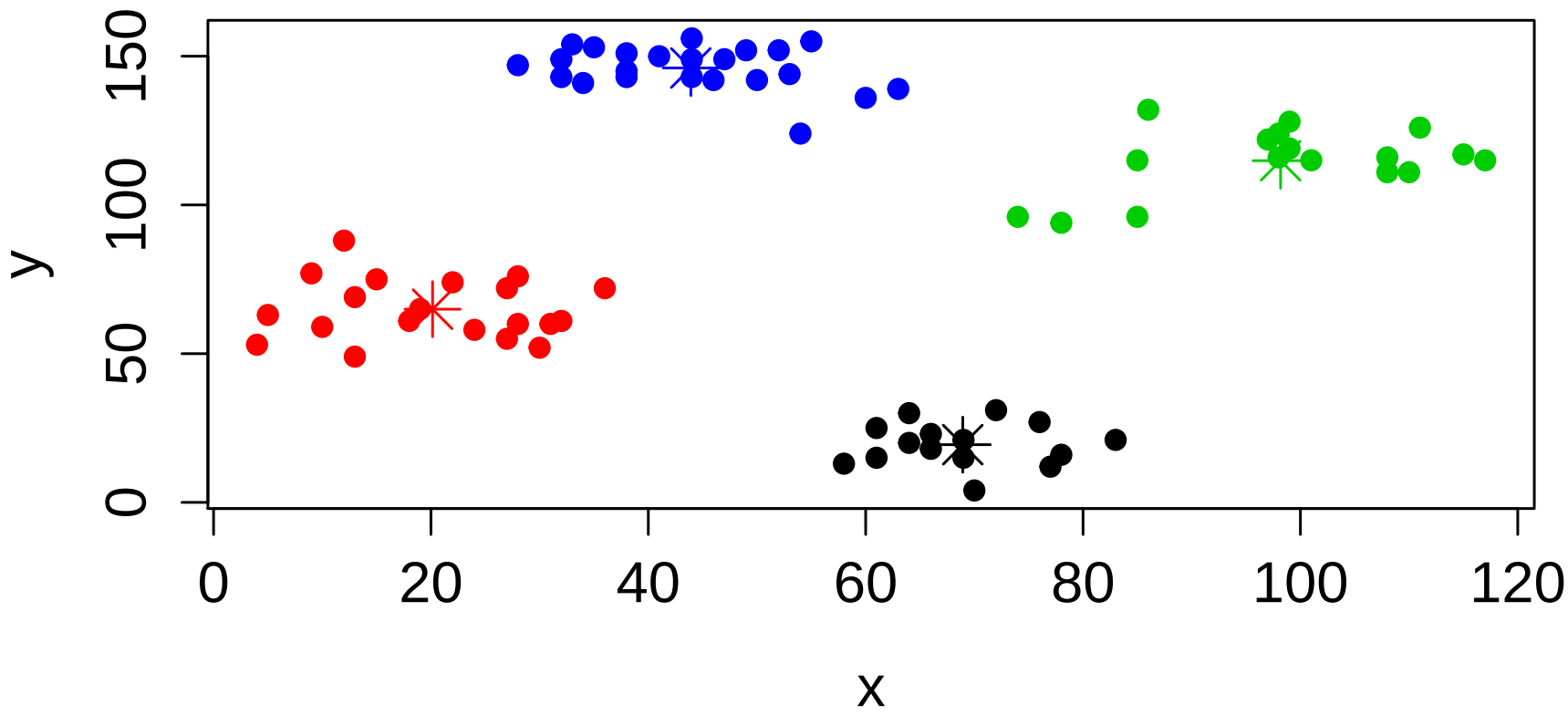


Figure: Grupos obtenidos en la iteración 2 de K-medias.

Grupos obtenidos en la iteración 10 de K-medias

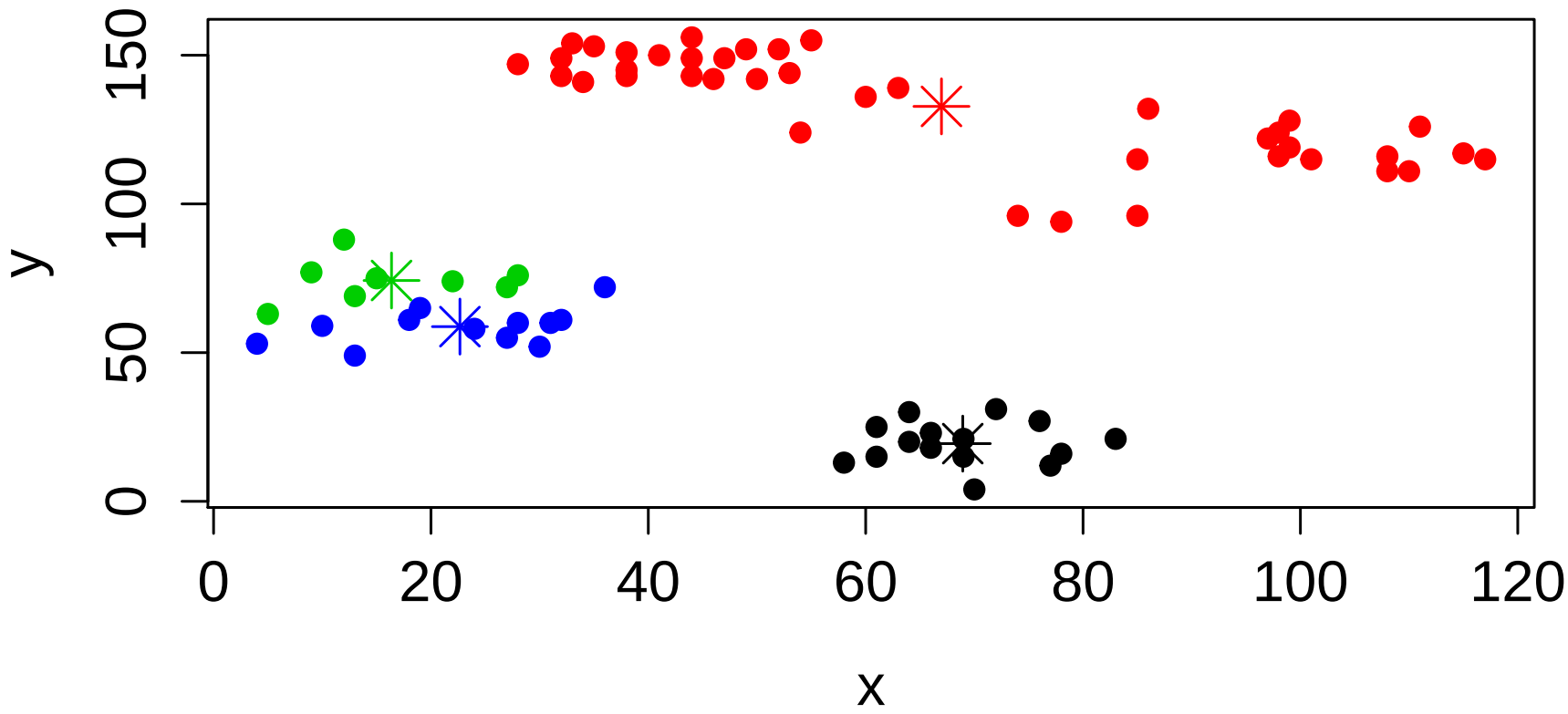


Figure: Grupos obtenidos en la iteración 10 de K-medias. Los grupos se estabilizan en un óptimo local.

K-means

La idea de K-means es que dado un conjunto de datos $\mathbf{x}_1, \ldots, \mathbf{x}_n$ un buen reagrupamiento es aquel que particiona las n observaciones en K conjuntos $C_1, \ldots, C_k$ de manera a minimizar la varianza intragrupos $W(\mathcal{C})$. Formalmente:

$$\mathcal{C}^* = \arg\min_{\mathcal{C}} \underbrace{\sum_{k=1}^{K} \sum_{\mathbf{x} \in C_k} \|\mathbf{x} - \overline{\boldsymbol{x}}_k\|^2}_{J_w} = \arg\min_{\mathcal{C}} \sum_{k=1}^{K} |C_k| \operatorname{Var} C_k$$

donde $\overline{\boldsymbol{x}}_k$ es el promedio de los puntos de C_k. Esto es equivalente en minimizar la suma de cuadrados de los pares de observaciones de un mismo cluster

$$\arg\min_{\mathcal{C}} \sum_{k=1}^{K} \frac{1}{2|C_k|} \sum_{\mathbf{x}_i, \mathbf{x}_{i'} \in C_k} \|\mathbf{x}_i - \mathbf{x}_{i'}\|^2$$

En efecto si $n_k = |C_k|$ se prueba que

$$\sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 = 2n_k \sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - \overline{x}_{kj})^2$$

donde $\overline{x}_{kj} = \frac{1}{n_k} \sum_{i \in C_k} x_{ij}$ es el promedio de la variable j en el cluster C_k.

Esto es porque, fijado la variable j :

$$\sum_{i,i' \in C_k} (x_{ij} - x_{i'j})^2 = \sum_{i \in C_k} \sum_{i' \in C_k} (x_{ij} - x_{i'j})^2 = 2 \sum_{i \in C_k} \sum_{i' \in C_k} (x_{ij} - \overline{x}_{kj})^2 = 2 \sum_{i \in C_k} n_k (x_{ij} - \overline{x}_{kj})^2$$

K-means

Given an initial set of K-means $\overline{\boldsymbol{x}}_1, \overline{\boldsymbol{x}}_2, \ldots, \overline{\boldsymbol{x}}_K$, the algorithm proceeds by alternating between two steps:

1. **Assignment step** Assign each observation to the cluster whose mean has the least squared euclidean distance], this is intuitively the "nearest" mean. Mathematically, this means partitioning the observations according to the groups generated by the means. $C_k^{(t)} = \left\{\mathbf{x}_i : \left\|\mathbf{x}_i - \overline{\boldsymbol{x}}_k^{(t)}\right\|^2 \leq \left\|\mathbf{x}_i - \overline{\boldsymbol{x}}_{k'}^{(t)}\right\|^2 \ \forall k', 1 \leq k' \leq k\right\}$
2. **Update step** Calculate the new means to be the centroids of the observations in the new clusters. $\overline{\boldsymbol{x}}_k^{(t+1)} = \frac{1}{|C_k^{(t)}|} \sum_{\mathbf{x}_i \in C_k^{(t)}} \mathbf{x}_i$

The algorithm has converged when the assignments no longer change. There is no guarantee that the optimum is found using this algorithm.

The method divides the space by several hyperplanes

Ejemplo tomado del ISLR

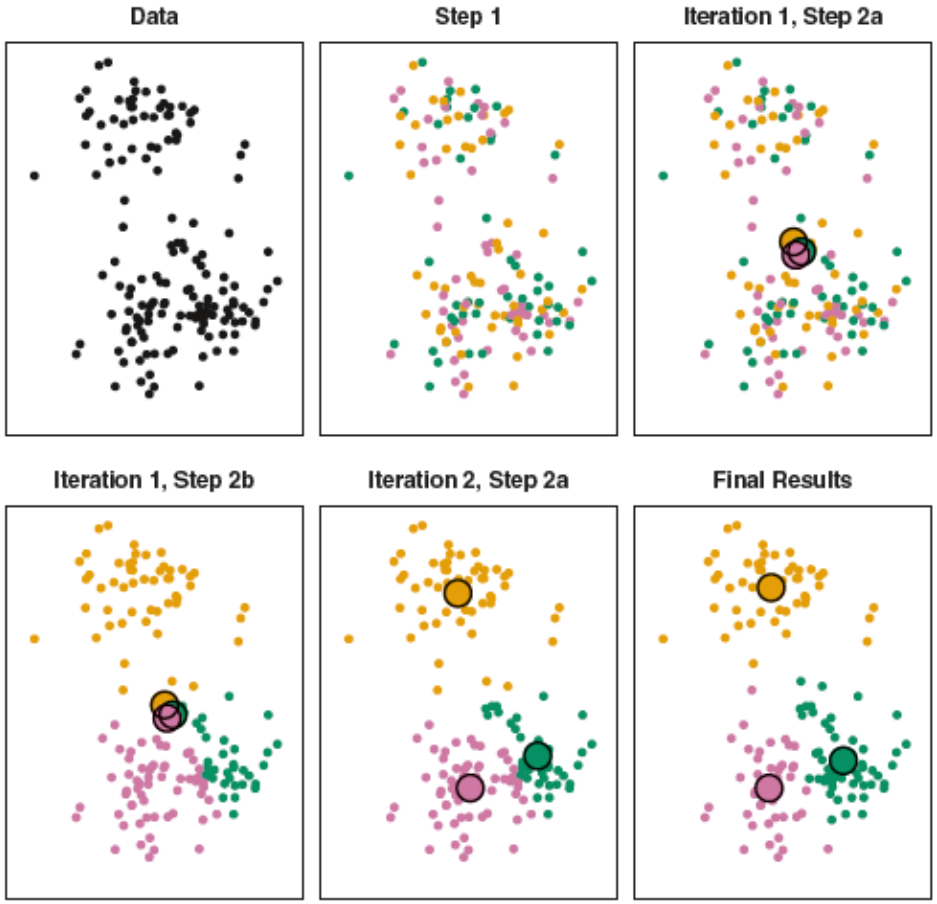


FIGURE 10.6. *The progress of the K-means algorithm on the example of Figure 10.5 with K=3.* Top left: *the observations are shown.* Top center: *in Step 1 of the algorithm, each observation is randomly assigned to a cluster.* Top right: *in Step 2(a), the cluster centroids are computed. These are shown as large colored disks. Initially the centroids are almost completely overlapping because the initial cluster assignments were chosen at random.* Bottom left: *in Step 2(b), each observation is assigned to the nearest centroid.* Bottom center: *Step 2(a) is once again performed, leading to new cluster centroids.* Bottom right: *the results obtained after ten iterations.*

K-Means

- A cada iteración el criterio $J_w = \sum_{k=1}^{K} \sum_{\mathbf{x} \in C_k} \|\mathbf{x} - \overline{\boldsymbol{x}}_k\|^2$ disminuye.
- El algoritmo converge rapidamente al menos hacia un mínimo local de J_w.
- Algunos inconvenientes de K-medias y los métodos particionales:
  - las variables son cuantitativas (por lo general).
  - hay que elegir el número de clusters al principio
  - son sensibles a la inicialización: inicializaciones distintas pueden dar clusters distintos .

- Algunas soluciones:
  - para el número de clusters: visualización, consideraciones sobre el problema, criterios de penalización (penalizan la cantidad de clusters, podemos citar por ejemplos AIC o BIC)
  - inicialización: como el algoritmo puede converger a mínimos locales, una solución es correr el algoritmo con diferentes particiones iniciales aleatorias y elegir el resultado con menor función objetivo.
    Otra posiblidad es reagrupar las observaciones x_i que están siempre en el mismo cluster.

K-means: varias inicializaciones

Multiples inicializaciones.

Because the K-means algorithm finds a local rather than a global optimum, the results obtained will depend on the initial (random) cluster assignment of each observation in Step 1 of Algorithm 10.1. For this reason, it is important to run the algorithm multiple times from different random initial configurations. Then one selects the best solution, i.e. that for which the objective (10.11) is smallest. Figure 10.7 shows the local optima obtained by running K-means clustering six times using six different initial cluster assignments, using the toy data from Figure 10.5. In this case, the best clustering is the one with an objective value of 235.8.

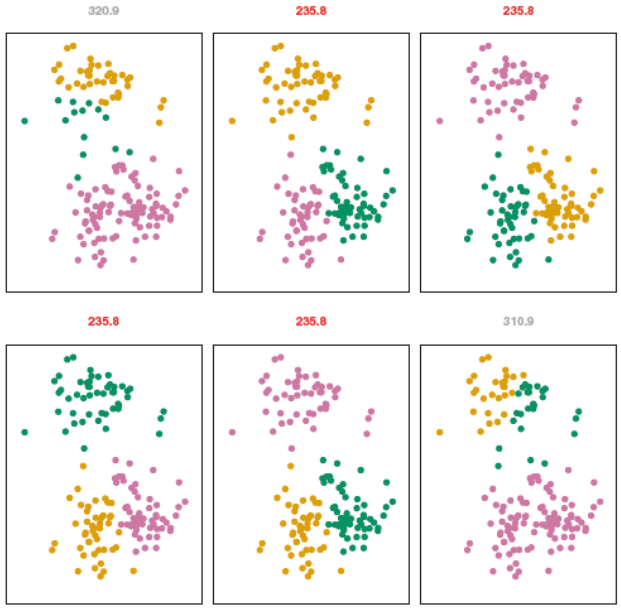


FIGURE 10.7. *K-means clustering performed six times on the data from Figure 10.5 with $K = 3$, each time with a different random assignment of the observations in Step 1 of the K-means algorithm. Above each plot is the value of the objective (10.11). Three different local optima were obtained, one of which resulted in a smaller value of the objective and provides better separation between the clusters. Those labeled in red all achieved the same best solution, with an objective value of 235.8.*

K-means: Criterio de parada

Buscar un codo en la gráfica del criterio (within groups sum of squares)

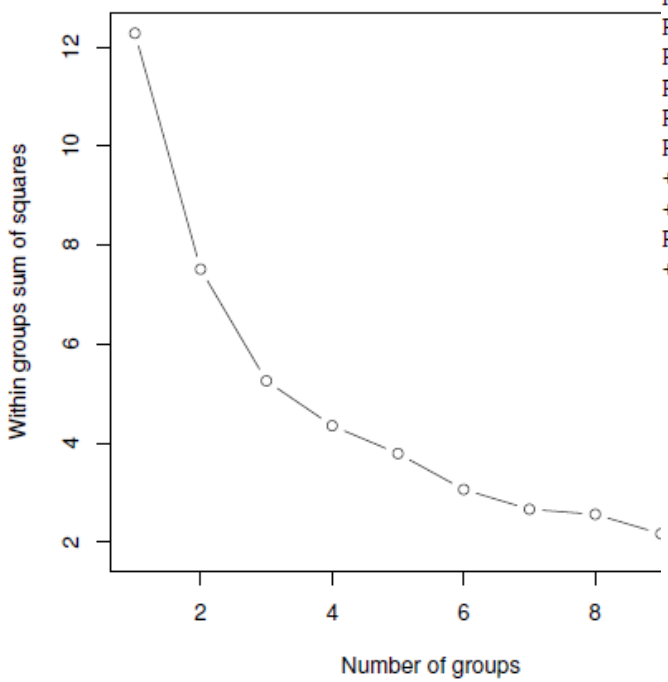


Figure 18.7 Within-cluster sum of squares for different numbe
the exoplanet data.

```
R> rge <- apply(planets, 2, max) - apply(planets, 2, min)
R> planet.dat <- sweep(planets, 2, rge, FUN = "/")
R> n <- nrow(planet.dat)
R> wss <- rep(0, 10)
R> wss[1] <- (n - 1) * sum(apply(planet.dat, 2, var))
R> for (i in 2:10)
+      wss[i] <- sum(kmeans(planet.dat,
+                           centers = i)$withinss)
R> plot(1:10, wss, type = "b", xlab = "Number of groups",
+       ylab = "Within groups sum of squares")
```

Una variante robusta: PAM

- Al igual que K-medias, el algoritmo de *partitioning around medoids* (PAM) necesita una configuración (partición) inicial y un número preespecificado de grupos.
- Por otro lado, PAM busca los K "individuos representativos" (o *medoides*) entre el conjunto de observaciones (mientras que K-medias utiliza los promedios del grupo), que minimizan la suma de las disimilaridades al resto de los integrantes. No se necesita de la distancia euclidea.
- En general PAM es más robusto que K-medias y requiere como argumento de entrada solamente la matriz de disimilaridades entre observaciones y no los datos originales.
- Como contrapartida es más intensivo computacionalmente, debido principalmente a la búsqueda de *medoides* (Izenman, 2008).

Plan

Técnicas jerárquicas

- Los resultados de aplicar las técnicas de particionamiento (K-medias o PAM por ejemplo), dependen de la elección del número de clusters y de una configuración inicial.

- En cambio, los métodos de *clustering jerárquico* no requieren estas especificaciones.

- En su lugar, estas técnicas necesitan que el usuario especifique una *medida de disimilaridad entre grupos* (disjuntos), basada en las disimilaridades entre las observaciones de los grupos.
- Como su nombre sugiere, estas técnicas producen representaciones jerárquicas en las cuales los clusters en cada nivel de la jerarquía son creados uniendo clusters del siguiente nivel inferior (y por ende podemos reconstruir la "historia").

- En el nivel más bajo cada cluster contiene una única observación y en el más alto hay un sólo cluster con todas las observaciones.

- Las estrategias para el clustering jerárquicos se dividen en dos paradigmas básicos: *aglomerativos* y *divisivos*.
- Son computacionalmente más complejos que K-means (tipicamente $O(n^2 log(n))$ vs. $O(n)$) y conviene aplicarlos en muestras chicas.

Medidas de disimilaridad

Las medidas de disimilaridad más comunes entre clusters son:

- *single linkage* o vecino más cercano:

$$d_{min}(C_1, C_2) = \min_{x_i \in c_1, x_j \in C_2} d(x_i, x_j)$$

- *complete linkage* o vecino más lejano:

$$d_{max}(C_1, C_2) = \max_{x_i \in C_1, x_j \in C_2} d(x_i, x_j)$$

- *average linkage*:

$$d_{media}(C_1, C_2) = \frac{1}{|C_1|C_2|} \sum_{x_i \in C_1, x_j \in C_2} d(x_i, x_j)$$

- *método centroide*:

$$d_{cent}(C_1, C_2) = d(\mu_1, \mu_2)$$

donde μ_1 y μ_2 son los centroides de C_1 y C_2 respectivamente

- *de Ward*:

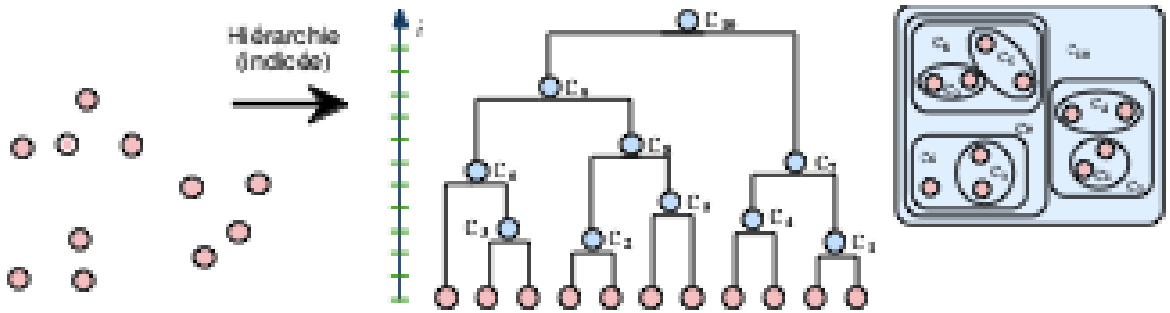
$$d_{ward}(C_1, C_2) = \sqrt{\frac{|C_1||C_2|}{|C_1| + |C_2|}} d(\mu_1, \mu_2)$$

Medidas de disimilaridad

- Ninguna de estas medidas es *uniformemente* mejor que las otras para todos los problemas de clustering (Izenman, 2008).
- El vecino más cercano a menudo produce largas "cadenas" de clusters, unidas por unos pocos puntos cercanos (lo cual podra no ser deseado), mientras que el vecino más lejano tiende a producir muchos clusters, pequeños y compactos.
- La distancia promedio es dependiente del tamaño de los clusters, a diferencia de las otras dos.
- Si los datos presentan una estructura de grupos bien diferenciados, entonces los cuatro métodos tenderán a dar los mismos resultados.

Métodos aglomerativos

- Estos métodos comienzan con cada observación representando un solo cluster.
- En cada uno de los siguientes pasos los dos cluster más cercanos (menos disímiles) son unidos en un único grupo, produciendo un cluster menos que en el nivel inmediato anterior.
- El resultado es un árbol (dendograma) donde las hojas son las muestras originales y representa a las fusiones sucesivas.
- Por lo tanto, es necesario definir una medida de disimilaridad entre clusters o grupos de observaciones.
- Distintas medidas dan lugar a variantes que pueden producir resultados diferentes.



Métodos divisivos

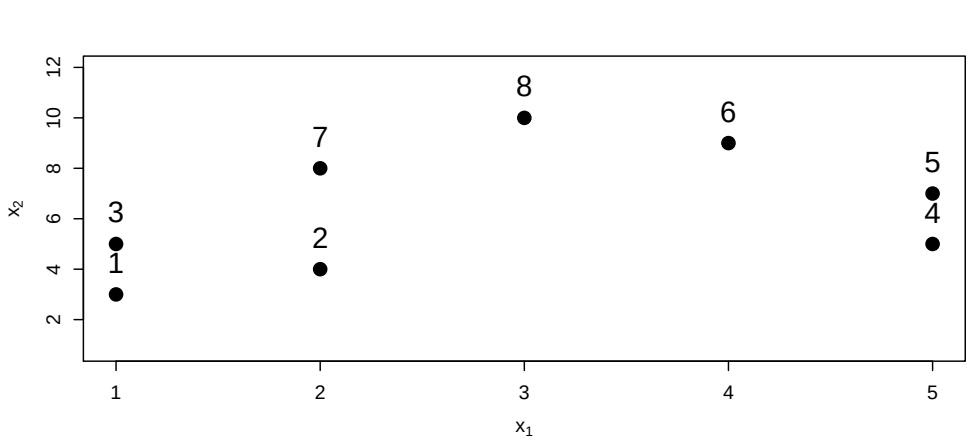
- Los algoritmos de clustering divisivos comienzan con un único cluster con todas las observaciones y recursivamente dividen uno de los clusters existentes en dos clusters "hijos" hasta obtener tantos grupos como observaciones.
- Dividir un cluster es computacionalmente más demandante que unir dos, dado que no solo se debe encontrar el cluster a ser dividido sino que también las observaciones que formarn los dos nuevos grupos deben ser identificadas.
- Por este motivo, los métodos divisivos son menos utilizados en la práctica (Varmuza y Filmoser, 2009).
- Los resultados del clustering jerárquico suelen ser representados mediante un diagrama de árbol jerárquico, conocido como *dendograma*.
- Grupos u observaciones que son más similares son combinados a bajas alturas, mientras que los más dismiles lo hacen a alturas grandes.
- Una partición de los datos en un número dado de grupos puede obtenerse "cortando" el dendograma en un nivel apropiado de altura.

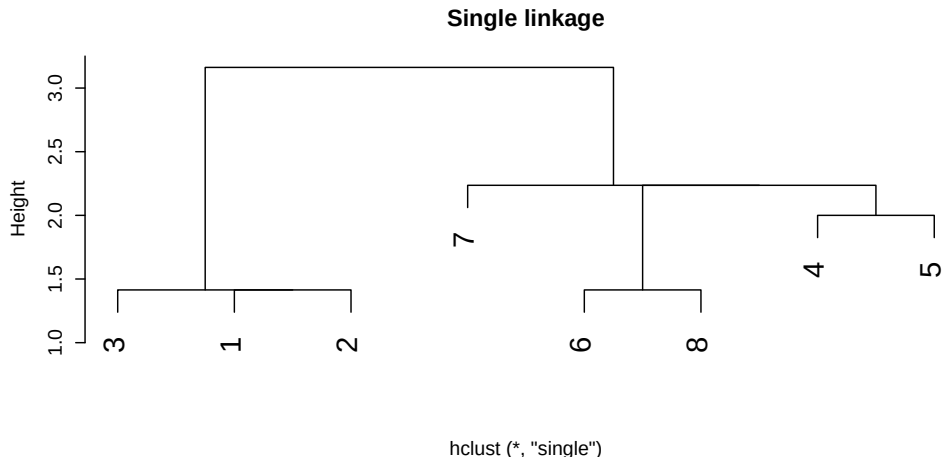
Ejemplo numérico en R

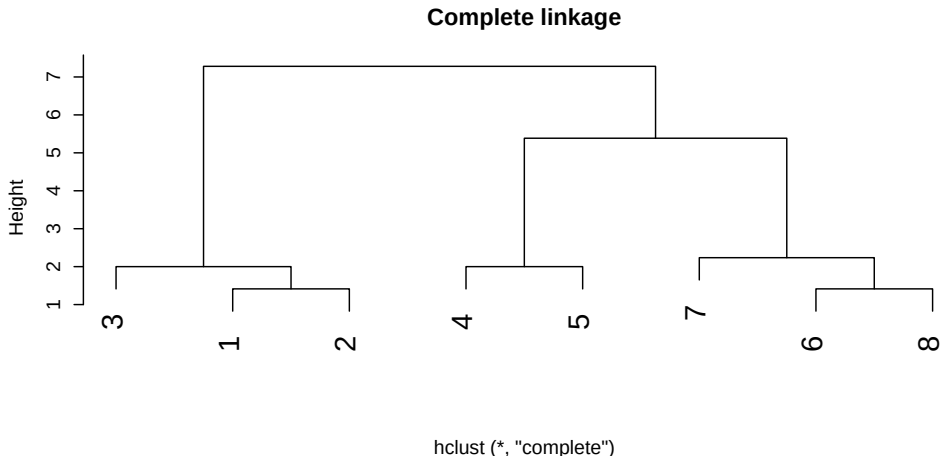
```
X <- rbind(c(1, 3), c(2, 4), c(1, 5), c(5, 5), c(5, 7), c(4, 9), c(2, 8), c(3, 10))
> X
     [,1] [,2]
[1,]    1    3
[2,]    2    4
[3,]    1    5
[4,]    5    5
[5,]    5    7
[6,]    4    9
[7,]    2    8
[8,]    3   10
> (D <- dist(X))
         1        2        3        4        5        6        7
2 1.414214
3 2.000000 1.414214
4 4.472136 3.162278 4.000000
5 5.656854 4.242641 4.472136 2.000000
6 6.708204 5.385165 5.000000 4.123106 2.236068
7 5.099020 4.000000 3.162278 4.242641 3.162278 2.236068
8 7.280110 6.082763 5.385165 5.385165 3.605551 1.414214 2.236068
```

```
> par(mfrow = c(2, 2), mex =  .8)
> plot(X, ylim = c(.8 * min(X[, 1]), 1.2 * max(X[, 2])), pch = 19, cex = 1.5,
+ xlab = expression(x[1]), ylab = expression(x[2]))
> text(X[, 1], 1.25 + X[, 2], 1:8, cex = 1.5)
> plot(hclust(dist(X), 'single'), main = 'Single linkage', xlab = '', cex = 1.5)
> plot(hclust(dist(X), 'complete'), main = 'Complete linkage', xlab = '', cex = 1.5)
> plot(hclust(dist(X), 'average'), main = 'Average linkage', xlab = '', cex = 1.5)
```

Ejemplo en R







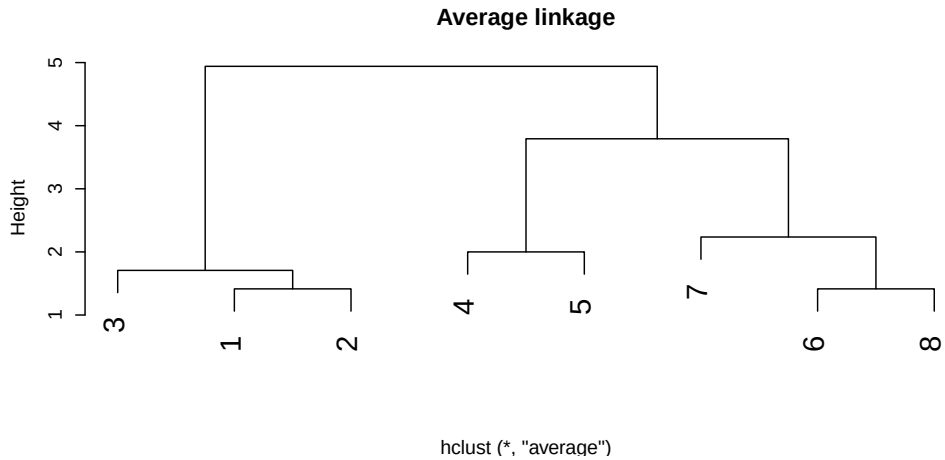


Figure: Pequeño ejemplo de clustering jerárquico aglomerativo.

Método Ward (de mínima varianza)

Este método busca optimizar en cada etapa la dispersión de las clases de la partición obtenida por agregación de dos objetos. La unión entre dos grupos se hace de manera a unir los dos clusters para los cuales se tenga el menor incremento en el valor total de la suma de los cuadrados de las diferencias, dentro de cada cluster, de cada individuo al centroide del cluster.

El objetivo es encontrar en cada etapa aquellos dos clusters cuya unión proporcione el menor incremento en la suma total de errores, $E = \sum_{k=1}^{h} E_k$ donde E_k denota a la suma de cuadrados de los errores del cluster k, o sea, la distancia euclídea al cuadrado entre cada individuo del cluster k a su centroide.

El método empieza con n clusters ($E_i = 0$ para todo $i = 1, \ldots, n$ y por lo tanto $E = 0$). Si se unen los clusters C_p y C_q para formar un nuevo cluster C_t, entonces el incremento de E es:

$$\Delta E_{pq} = E_t - E_p - E_q = \frac{n_p n_q}{n_p + n_q} ||\mu_p - \mu_q||^2$$

siendo n_p y n_q las cantidades de observaciones en E_p y E_q. Entonces el menor incremento de los errores cuadráticos es proporcional a la distancia euclídea al cuadrado de los centroides de los clusters unidos. En cada paso del algorítmo se calcula la expresión planteada anteriormente y se construye la tabla de disimilaridades. Se va uniendo de a dos grupos teniendo en cuenta cuál unión implica un menor incremento de la variación intragrupos.

Técnicas jerrquicas, algunos comentarios

- `https://www.ugr.es/~gallardo/pdf/cluster-3.pdf`
- Las fusiones o divisiones, una vez que son hechas, son irreversibles. De manera que, por ejemplo, cuando un algoritmo aglomerativo puso dos observaciones en un mismo grupo ellas no podrán aparecer en diferentes grupos en etapas posteriores.
- Estos métodos imponen una estructura jerárquica independientemente de que la misma exista o no en los datos.
- Aunque se los ha utilizado en diversas áreas, es en las aplicaciones biológicas donde son más relevantes y justificados (Everitt y Hothorn, 2008).

Reglas de detención

1. Mirar el dendograma
2. El R^2. En regresión lineal, se define como $R^2 = 1 - \frac{SCResiduos}{SCT}$. En Clustering

$$R^2 = 1 - \frac{SCDentro}{SCTotal} = 1 - \frac{W}{T} = 1 - \frac{\sum_{k=1}^{K}\sum_{i=1}^{n_k}\sum_{j=1}^{p}(x_{ij(k)} - \overline{x}_{kj})^2}{\sum_{i=1}^{n}\sum_{j=1}^{p}(x_{ij} - \overline{x}_j)^2}$$

Si hay un sólo grupo entonces $R^2 = 0$ y si hay n grupos $R^2 = 1$, Cortamos con la cantidad de grupos cuando no se produce un cambio demasiado significativo con el R^2 de un grupo al otro.

Reglas de detención

3 El pseudo F:

$$\frac{B/(k-1)}{W/(n-k)} = \frac{R^2/(k-1)}{(1-R^2)/(n-k)} = F_{p(k-1),p(n-k)}$$

donde

- B es la suma de las variaciones entre los grupos.
- W es la suma de las variaciones dentro de los grupos.
- k es el número de grupos, n la cantidad de observaciones y p la cantidad de variables.

El pseudo F no es una variable aleatoria que se distribuye como F pues no hay supuestos de normalidad. La regla es la siguiente:

- Si el indicador crece monótonamente con la cantidad de grupos, entonces no hay una estructura clara.
- Si el indicador disminuye al crecer la cantidad de grupos, tampoco, pero se puede decir que existe una estructura jerárquica.
- Si el indicador llega a un m'áximo y luego decrece, entonces la población presenta un número de grupos definidos en este máximo.

Reglas de detención

4 El pseudo t^2. Este indicador se calcula como:

$$\frac{W_{GL} - W_G - W_L}{(W_G + W_L)/(n_G + n_L - 2)}$$

siendo W_{GL} la variance dentro de los grupos G y L, y tampoco tiene distrbución t^2 de Student. Se trata de determinar si la disminución en la variación intragrupos al pasar de k a $k+1$ grupos es significativa o no.

Se analizan los valores pseudo t^2 de arriba hacia abajo mirando de 1 grupo a n en la cantidad de grupos. Si con k grupos el indicador presenta valores muy grandes respecto a los que presenta en $k+1$, nos quedamos con $k+1$ grupos.
Se analiza también de arriba hacia abajo: partiendo de los n individuos hasta un único grupo final. Si en el paso $k+1$ el indicador tiene un valor muy chico con respecto al de paso k es conveniente quedarse con $k+1$ grupos.

Plan

Clustering basado en modelos

- En las técnicas de clustering presentadas anteriormente no se hace ningún supuesto acerca de la distribución de los datos (excepto que se asume una estructura de grupos).

- En cambio, en las técnicas de clustering *basado en modelos* se asume un modelo estadístico para los clusters, donde cada uno de ellos puede ser representado por una distribución de probabilidad multivariada.

- La distribución conjunta de los datos se dice que es una *mezcla* de distribuciones con componentes dadas por cada uno de los clusters:

$$f(x|\theta) = \sum_{j=1}^{K} \pi_j f(x|\theta_j)$$

con $\pi_j \geq 0$ y $\sum_{j=1}^{K} \pi_j = 1$.

- Luego, el algoritmo debe buscar la estimación de los parámetros que "mejor" se adapten a los datos y al modelo propuesto (*máxima verosimilitud*), así como el grado de pertenencia de cada observación a los distintos grupos.

Clustering basado en modelos

- La estimación de los parámetros del modelo asumido está basada en el algoritmo EM (*Expectation Maximization*).
- Este algoritmo busca maximizar la verosimilitud $f(x|\theta)$ alternando entre un paso de *esperanza* (donde las pertenencias a los grupos son estimadas) y un paso de *maximización* (donde los parámetros del modelo son estimados).
- En el modelo más simple se asume que cada uno de los K clusters están representados por distribuciones normales (gaussianas) multivariadas con diferentes medias pero iguales matrices de covarianzas de la forma $\sigma^2 \mathbf{I}$. Este supuesto produce clusters *esféricos* y de igual tamaño.
- En el modelo más general, las matrices de covarianza Σ_j, no tienen porque ser diagonales lo cual permite modelar clusters *elípticos*.

Clustering basado en modelos para dos clases

Supongamos que trabajamos con una población P con densidad f que se puede subdividir en dos subpoblaciones P_1 y P_2 y sea α la proporción de P en P_2. Entonces la probabilidad de observar x es

$$p(x) = \mathbb{P}(1)p(x|P_1) + \mathbb{P}(2)p(x|P_2)$$

$$f(x)\Delta x = \mathbb{P}(1)f_1(x)\Delta x + \mathbb{P}(2)f_2(x)\Delta x$$

y simplificando por Δx se tiene que

$$f(x) = (1-\alpha)f_1(x) + \alpha f_2(x) \text{ mezcla de dos densidades}$$

Entonces utilizando la formula de Bayes:

$$\mathbb{P}(P_1|x) = \frac{\mathbb{P}(P_1)p(x|P_1)}{p(x)} = \frac{(1-\alpha)f_1(x)}{f(x)} \qquad \mathbb{P}(P_2|x) = \frac{\mathbb{P}(P_2)p(x|P_2)}{p(x)} = \frac{\alpha f_2(x)}{f(x)}$$

Decidimos P_1 si

$$\mathbb{P}(P_1|x) > \mathbb{P}(P_2|x)$$

y si suponemos que $\alpha = \frac{1}{2}$, entonces dedidimos P_1 si

$$f_1(x) > f_2(x)$$

EM para dos clases

Supongamos en la clase 1 los datos tienen distribución $\mathcal{N}(\mu_1, \Sigma_1)$ y en la clase 2 distribución $\mathcal{N}(\mu_2, \Sigma_2)$.
Veamos como estimar el conjunto de parámetros $\theta = \{\mu_1, \mu_2, \Sigma_1, \Sigma_2, \alpha\}$. Suponemos que $x_1, \ldots, x_n$ son i.i.d. con densidad $f(X, \theta)$. La logverosimilitud es:

$$\log(\mathcal{L}(\theta|x_1, \ldots, x_n)) = \sum_{i=1}^{n} \log(f(X = x_i|\theta))$$

$$\sum_{i=1}^{n} \log(f(X = x_i|\theta)) = \sum_{i=1}^{n} \log((1-\alpha)\mathcal{N}(x_i; \mu_1, \Sigma_1) + \alpha\mathcal{N}(x_i; \mu_2, \Sigma_2))$$

Uno de los problemas es que no conocemos el cluster en el que pertenecen $x_1, \ldots, x_n$. Consideramos $z_i = 1$ si x_i pertenece al cluster 1 y $z_i = 0$ si x_i pertenece al cluster 2. Definimos entonces la log verosimilitud completada como:

$$\begin{aligned}
\mathcal{L}_c(\theta|\{(x_i, z_i)_{i=1}^n\}) &= \sum_{i=1}^{n} \log p(x_i, z_i|\theta) = \sum_{i=1}^{n} \log p(x_i|z_i, \theta)p(z_i|\theta) \\
&= \sum_{i=1}^{n} \log\left((\mathcal{N}(x_i, \mu_1, \Sigma_1)(1-\alpha))^{z_i}(\mathcal{N}(x_i, \mu_2, \Sigma_2)(\alpha))^{1-z_i}\right) \\
&= \sum_{i=1}^{n} z_i \log\big((1-\alpha)\mathcal{N}(x_i, \mu_1, \Sigma_1)\big) + (1-z_i)\log\big(\alpha\mathcal{N}(x_i, \mu_2, \Sigma_2)\big)
\end{aligned}$$

EM para dos clases

Como no conocemos los z_i, supongamos que son aleatorias, y tomamos la esperanza condicional de $\mathcal{L}_c$, ya que intuitivamente, si hacemos un tiraje aleatorio de la clase de x_i muchas veces, la cantidad de veces que x_i cae en la clase $\mathcal{C}_1$ se cuantifica por $\mathbb{E}(z_i|x_i) = \mathbb{P}(z_i = 1|x_i) = \gamma_i$.

$$\begin{aligned}\mathbb{E}\left(\mathcal{L}_c|x_1,\ldots,x_n,\theta\right) = \sum_{i=1}^{n} &\mathbb{E}(z_i|x_i)\log\big((1-\alpha)\mathcal{N}(x_i,\mu_1,\Sigma_1)\big) \\ &+(1-\mathbb{E}(z_i|x_i))\log\big(\alpha\mathcal{N}(x_i,\mu_2,\Sigma_2)\big)\end{aligned}$$

Por otro lado:

$$\gamma_i = \mathbb{P}(z_i = 1|x_i) = \frac{\mathbb{P}(z_i=1)f(x_i|z_i=1)}{f(x_i)} = \frac{(1-\alpha)\mathcal{N}(x_i,\mu_1,\Sigma_1)}{(1-\alpha)\mathcal{N}(x_i,\mu_1,\Sigma_1)+\alpha\mathcal{N}(x_i,\mu_2,\Sigma_2)}$$

Todas estas estimaciones dependen una de las otras, o sea para estimar $\gamma_1,\gamma_2,\ldots,\gamma_n$ se necesitan las estimaciones de $\mu_1,\mu_2,\Sigma_1,\Sigma_2$ y α y al revéz, para estimar $\mu_1,\mu_2,\Sigma_1,\Sigma_2$ y α necesitamos de las estimaciones de $\gamma_1,\gamma_2,\ldots,\gamma_n$.

EM para dos clases

1. Elegimos valores iniciales para $\hat{\mu}_1, \hat{\mu}_2, \hat{\Sigma}_1, \hat{\Sigma}_2$ y $\hat{\alpha}$
2. Etapa E (Expectation) Para cada $i = 1, \ldots, n$, estimo $\gamma_i = \mathbb{P}(z_i = 1 | x_i)$ (que es una esperanza) como

$$\hat{\gamma}_i = \frac{\hat{\alpha}\mathcal{N}(x_i, \hat{\mu}_2, \hat{\Sigma}_2)}{(1-\hat{\alpha})\mathcal{N}(x_i, \hat{\mu}_1, \hat{\Sigma}_1) + \hat{\alpha}\mathcal{N}(x_i, \hat{\mu}_2, \hat{\Sigma}_2)}$$

   a partir de una estimación de los parámetros.
3. Etapa M (Maximization) Teniendo las estimaciones $\hat{\gamma}_1, \hat{\gamma}_2, \ldots, \hat{\gamma}_n$, se procede a una segunda etapa, de maximización de la verosimilitud

$$\sum_{i=1}^{n} \hat{\gamma}_i \log\big((1-\alpha)\mathcal{N}(x_i, \mu_1, \Sigma_1)\big) + (1-\hat{\gamma}_i)\log\big(\alpha\mathcal{N}(x_i, \mu_2, \Sigma_2)\big)$$

   y se obtiene (con calculos simples):

$$\hat{\mu}_1 = \frac{\sum_{i=1}^{n}(1-\hat{\gamma}_i)x_i}{\sum_{i=1}^{n}(1-\hat{\gamma}_i)} \quad \hat{\mu}_2 = \frac{\sum_{i=1}^{n}\hat{\gamma}_i x_i}{\sum_{i=1}^{n}\hat{\gamma}_i} \quad \hat{\alpha} = \frac{\sum_{i=1}^{n}\hat{\gamma}_i}{n} \quad \hat{\Sigma}_j = \frac{\sum_{i=1}^{n}\hat{\gamma}_i(x_i-\mu_j)(x_i-\mu_j)'}{\sum_{i=1}^{n}\hat{\gamma}_i} \; j = 1, 2$$

4. Repetimos los pasos 2 y 3 hasta convergencia: fijado un umbral ϵ, pediremos que la diferencia entre una estimación del conjunto de parámetros de una etapa y la de la etapa siguiente sea menor que este umbral ($||\theta^{(t)} - \theta^{(t-1)}||_1 < \epsilon$).

Comparación con k-means

	k-means	Cluster basado en modelos
Modelo	-	Mezcla: $f(X) = \sum_{j=1}^{K} \pi_j f(X\|Z = j)$
Parámetros a estimar	Centro de los clusters μ_k	Probilidades a priori π_k y parámetros de $f(X\|Z = k)$
Criterio	Varianza intra-clases	Logverosimilitud
Regla para la afectación	Cluster más cercano $\underset{k}{\text{Argmin}}\, d(x, \mu_k)$	Proba a posteriori $\underset{k}{\text{Argmax}}\, \mathbb{P}(Z = k\|X = x)$

Plan

Spectral Clustering

(A Tutorial on Spectral Clustering, U. Von Luxburg 2007)
Traditional methods as k-means use elliptical metrics and may not detected non-convex clusters. Spectral Clustering converts a clustering problem into a graph partition problem and outperforms many traditional algorithms.

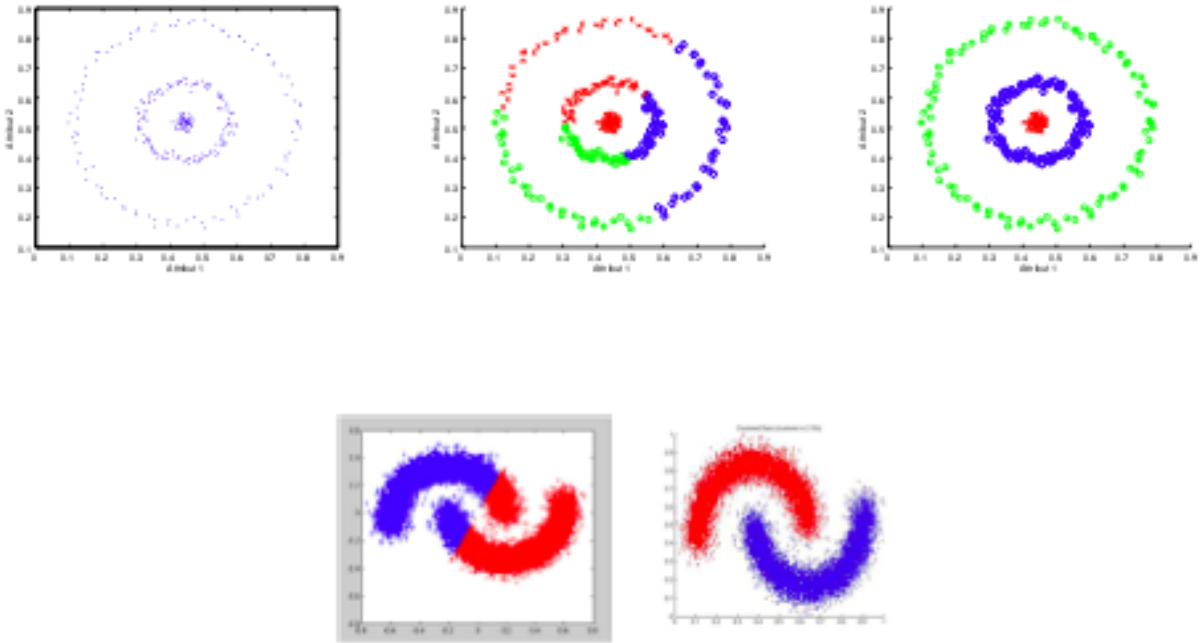


Figure: First row at left: the data, at center the clustering obtained by k-means and at right the clustering obtained by spectral clustering. Second row: Banana data set with k-means and with spectral clustering

1. Construct a similarity graph, as a K-NN graph or an epsilon-graph or a similarity graph $\left(s_{ij} = s(x_i, x_j) = exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right)\right)$, from all the data points:

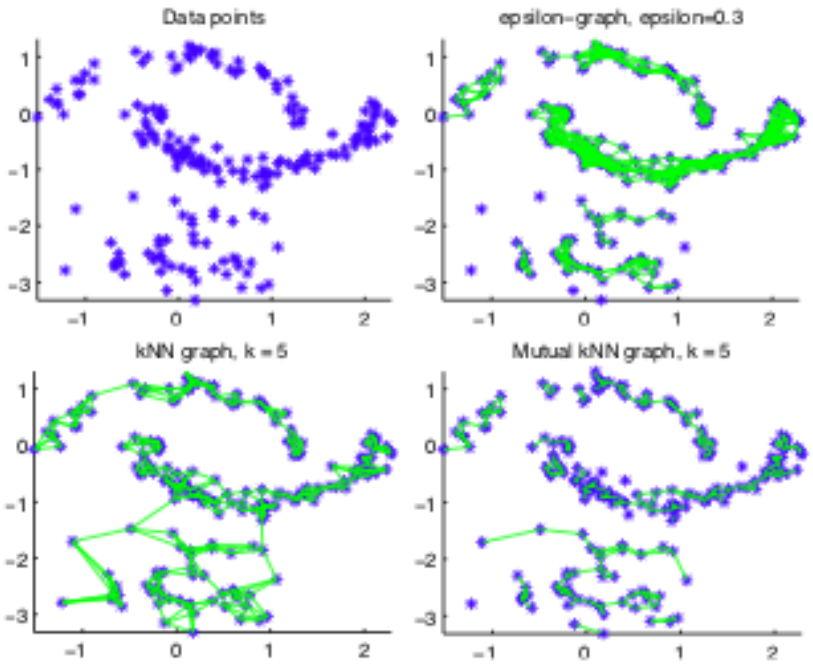


Figure: Different similarity graphs (Von Luxburg, 2007)

2. Embed the data points in a low dimensional space (*spectral embedding*) in which the cluster are more obvious, with the use of the eigenvectors of the graph Laplacian.
3. Apply a classical cluster algorithm such k-means in this new space to partition the embedding.

Spectral Clustering

Step 1: In all case we have to construct the weight matrix $W = ((W_{ij}))$ between the vertex of the graph. This matrix is symmetric.

- in the case of a k-NN graph $w_{ij} = 1$ if v_i is one of the k closest points to v_j and 0 otherelse.
- in the case of an ϵ-graph $w_{ij} = 1$ if $||v_i - v_j||_2 < \epsilon$
- In a similarity graph: the vertices $v_1, \ldots, v_n$ of the graph represent the observations $x_1, \ldots, x_n$. Two vertices are connected if the similarity s_{ij} between x_i and x_j is positive. We want to obtain a partition of the graph such that:
  - edges between different groups have a very low weight (points in different clusters are very different)
  - edges within a group have high weight (points of the same cluster are similar to each other).

  The goal of the construction of the similarity graph is to represent the local neighborhood relationships between the data points. In this case $w_{ij} = s_{ij} = exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right)$. Also, we can set some s_{ij} to 0 if $s_{ij} < \delta$ where δ is a threshold or we apply a k-nearest neighboor filter to build a representation of a graph connecting just the closest dataset points.

Spectral Clustering

Step 2: Then we compute:

- a degree matrix D where each diagonal value is the degree of the respective vertex and all other positions are zero.
- the Laplacian matrix: $L = D - W$. This matrix:
  - For all vector $f \in \mathbb{R}^n$, $f'Lf = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2$
  - each row sums 0, is symmetric, positive semi-definite
  - has n non-negative real valued eigenvalues: $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.
  - The vector $\mathbf{1} = (1, \ldots, 1)'$ is an eigenvector associated to the eigenvalue 0. Moreover, multiplicity k of eigenvalue 0 indicated the quantity of different components $A_1, \ldots, A_k$ of graph G and if S_0 is the eigensubspace associated to 0, then $S_0 = [\mathbf{1}_{A_1}, \mathbf{1}_{A_2}, \ldots, \mathbf{1}_{A_k}]$ where $\mathbf{1}_{A_k}$ is a vector with all components 0 except $\#A_k$ with value 1.

Spectral Clustering: algorithm in general case

Input: n data points $v_1, \ldots, v_n$, adjacency matrix parameter, and number of clusters K

Output: K cluster assignments

- Construct the adjacency matrix W and graph Laplacian $L = D - W$ where D is the degree matrix
- Compute the SVD of L and retrieve K the eigenvectors $Y = [y_1 | \ldots | y_K]$ associated with the smallest K eigenvalues of L.
- Use K-means on the rows of Y to determine cluster assignments.

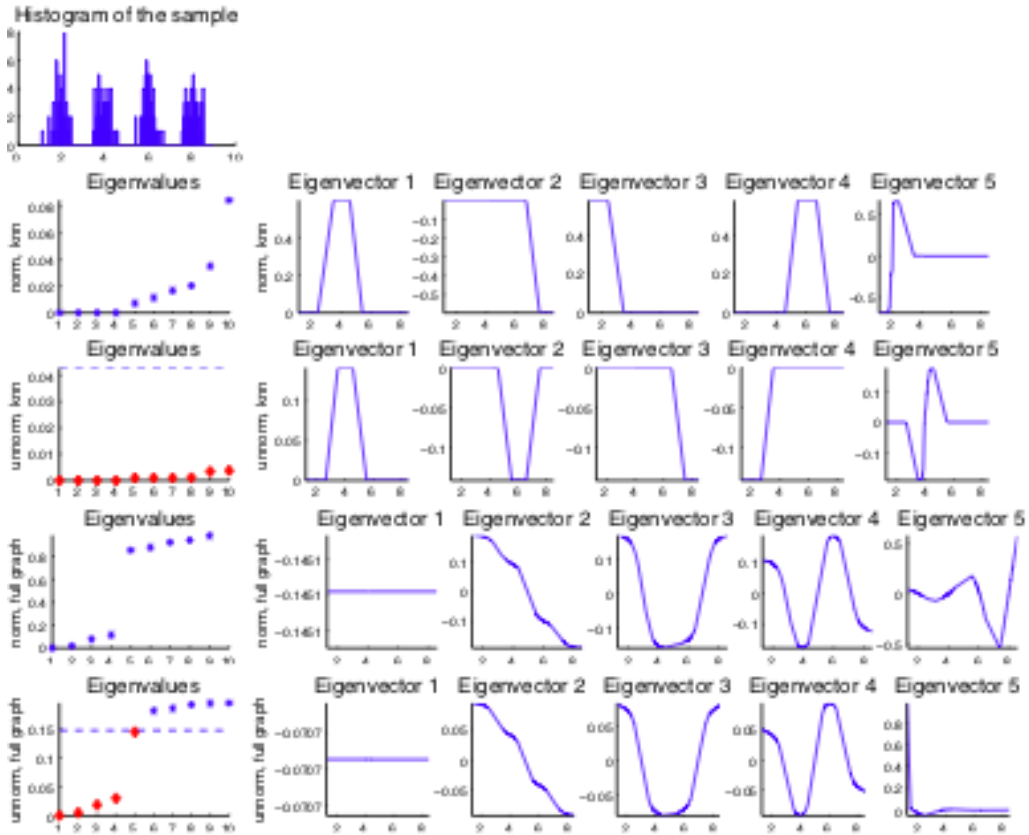
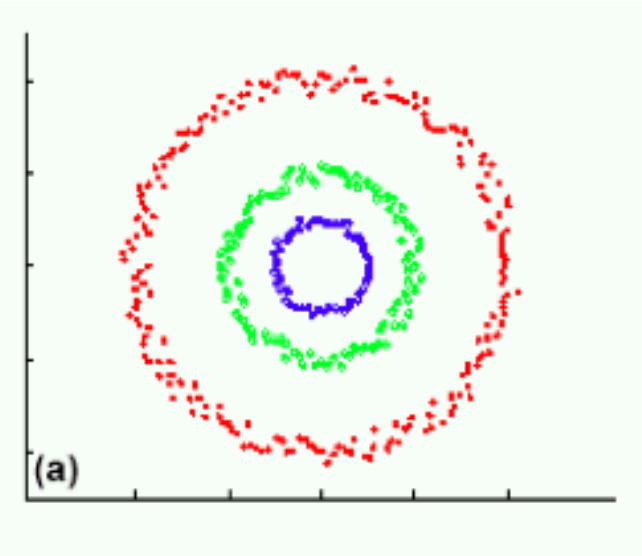


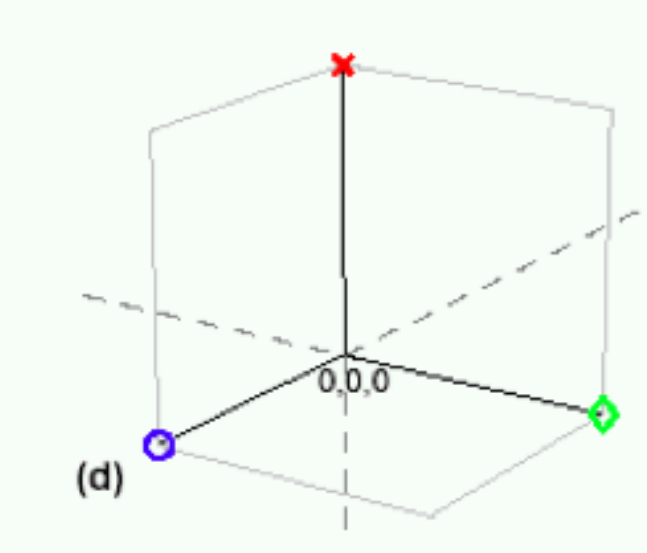
Figure 1: Toy example for spectral clustering where the data points have been drawn from a mixture of four Gaussians on $\mathbb{R}$. Left upper corner: histogram of the data. First and second row: eigenvalues and eigenvectors of L_{rw} and L based on the k-nearest neighbor graph. Third and fourth row: eigenvalues and eigenvectors of L_{rw} and L based on the fully connected graph. For all plots, we used the Gaussian kernel with $\sigma = 1$ as similarity function. See text for more details.

In second case (fully connected graph), the first eigenvector is the all ones vector 1 (as the graph is connected). The 2nd eigenvector thresholded at 0 separates the first two clusters from the last two. k-means clustering of the 4 eigenvectors identifies all clusters.

Original data



Projected data



Graph has 3 connected components – first three eigenvectors are constant (all ones) on each component.

Un ejemplo:

$$W = \begin{pmatrix} 1 & 1.5 & 0 & 0 & 0 \\ 1.5 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0.9 \\ 0 & 0 & 0 & 1 & 2.1 \\ 0 & 0 & 0.9 & 2.1 & 1 \end{pmatrix} \qquad W_1 = \begin{pmatrix} 1 & 0 & 0 & 1.5 & 0 \\ 0 & 1 & 0 & 0 & 2.1 \\ 0 & 0 & 1 & 0 & 0.9 \\ 1.5 & 0 & 0 & 1 & 0 \\ 0 & 2.1 & 0.9 & 0 & 1 \end{pmatrix}$$

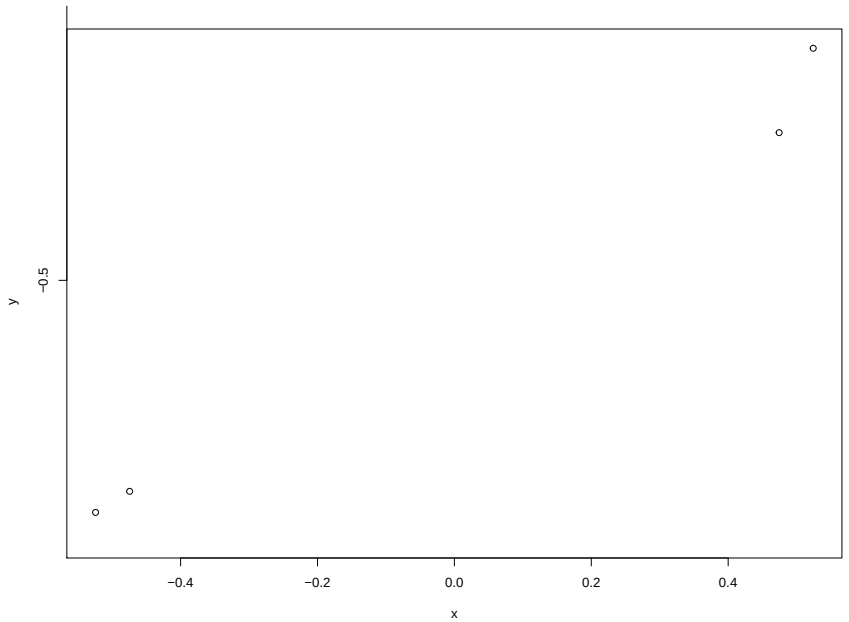
```
W=matrix(c(1,1.5,0,0,0,1.5,1,0,0,0,0,0,1,0,0.9,0,0,0,1,2.1,0,0,0.9,2.1,1),5,5)
D=diag(apply(W,FUN=sum,1))
L=D-W
eigen(L)

eigen() decomposition
$values
[1]   4.824829e+00   3.000000e+00   1.175171e+00   1.776357e-15 -2.743431e-307

$vectors
           [,1]       [,2]       [,3]      [,4]      [,5]
[1,]  0.0000000 -0.7071068  0.0000000 0.7071068 0.0000000
[2,]  0.0000000  0.7071068  0.0000000 0.7071068 0.0000000
[3,]  0.1787042  0.0000000 -0.7967004 0.0000000 0.5773503
[4,]  0.6006107  0.0000000  0.5531126 0.0000000 0.5773503
[5,] -0.7793149  0.0000000  0.2435878 0.0000000 0.5773503

W1=matrix(c(1,0,0,1.5,0,0,1,0,0,2.1,0,0,1,0,0.9,1.5,0,0,1,0,0,2.1,0.9,0,1),5,5)
D1=diag(apply(W1,FUN=sum,1))
L1=D1-W1
eigen(L1)

eigen() decomposition
$values
[1]   4.824829e+00   3.000000e+00   1.175171e+00   1.776357e-15 -2.743431e-307

$vectors
           [,1]       [,2]       [,3]      [,4]      [,5]
[1,]  0.0000000 -0.7071068  0.0000000 0.7071068 0.0000000
[2,]  0.6006107  0.0000000  0.5531126 0.0000000 0.5773503
[3,]  0.1787042  0.0000000 -0.7967004 0.0000000 0.5773503
[4,]  0.0000000  0.7071068  0.0000000 0.7071068 0.0000000
[5,] -0.7793149  0.0000000  0.2435878 0.0000000 0.5773503
```

conduce al mismo resultado

$$W = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix} \Rightarrow y_1 = \begin{pmatrix} 0.71 \\ 0.71 \\ 0 \\ 0 \end{pmatrix} \quad y_2 = \begin{pmatrix} 0 \\ 0 \\ 0.71 \\ 0.71 \end{pmatrix}$$ Acá los grupos son claros

$$W = \begin{pmatrix} 1 & 1 & 0.2 & 0 \\ 1 & 1 & 0 & 0.1 \\ 0.2 & 0 & 1 & 1 \\ 0 & 0.1 & 1 & 1 \end{pmatrix} \Rightarrow y_1 = \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} \quad y_2 = \begin{pmatrix} 0.47 \\ 0.52 \\ -0.47 \\ -0.52 \end{pmatrix}$$

En el segundo caso aparece el vector de 1's ya que el grafo es conexo, y el signo de las componentes de y_2 nos da los grupos.



Plan

Comparación de particiones

- Otro de los problemas que surge es el de validación de los clusters obtenidos. Por ejemplo el algoritmo de K-medias siempre encuentra clusters, no importa si los hay o no.

- Algunos criterios de validez de los clusters:
  - criterios internos: solo se basan en los datos y en el algoritmo usado. Una medida interna es la noción de estabilidad de los clusters, donde se mide la variablidad de los clusters al aplicar el algoritmo a diferentes submuestras de los datos;
  - criterios relativos: se comparan diferentes estructuras, por ejemplo se hacen clusters con diferentes algoritmos y se elige cuales son mejores en algún sentido;
  - criterios externos: se valida en base a información a priori de etiquetas de los datos (aunque en el caso de existir etiquetas se podrán usar métodos supervisados).

Comparación de particiones

Una forma intuitiva de comparar particiones es contando los pares de observaciones que son clasificados de la misma forma en ambas particiones.
El conjunto de todos los pares (no ordenados) de $\mathscr{L}$ es la unión disjunta de los siguientes conjuntos:

- S_a=\{pares que están en el mismo *cluster* en $\mathscr{C}$ y $\mathscr{C}'$\}
- S_b=\{pares que están en distintos *clusters* en $\mathscr{C}$ y $\mathscr{C}'$\}
- S_c=\{pares que están en el mismo *cluster* en $\mathscr{C}$ y distinto *cluster* en $\mathscr{C}'$\}
- S_d=\{pares que están en distintos *clusters* en $\mathscr{C}$ y en el mismo *cluster* en $\mathscr{C}'$\}

Se denotan como a, b, c y d a los cardinales de S_a, S_b, S_c y S_d, respectivamente y es claro que:

- S_a, S_b, S_c y S_d son disjuntos dos a dos
- $a + b + c + d = n(n-1)/2 = \binom{n}{2}$.

Comparación de particiones

Para un conjunto de datos con $n = 8$ observaciones, se definen las particiones con 3 clases:
$\mathscr{C} = \{C_1 = (x_1, x_2, x_3), C_2 = (x_4, x_5), C_3 = (x_6, x_7, x_8)\}$
$\mathscr{C}' = \{C_1' = (x_6, x_7), C_2' = (x_1, x_2, x_3, x_4, x_5), C_3' = (x_8)\}$
A partir de la comparación, dichos elementos se clasifican en los siguientes conjuntos definidos anteriormente:

- $S_a = \{(x_1, x_2), (x_1, x_3), (x_2, x_3), (x_4, x_5), (x_6, x_7)\}$
- $S_b = \{(x_1, x_6), (x_1, x_7), (x_1, x_8), (x_2, x_6), (x_2, x_7), (x_2, x_8), (x_3, x_6), (x_3, x_7), (x_3, x_8), (x_4, x_6), (x_4, x_7), (x_4, x_8), (x_5, x_6), (x_5, x_7), (x_5, x_8)\}$
- $S_c = \{(x_6, x_8), (x_7, x_8)\}$
- $S_d = \{(x_1, x_4), (x_2, x_4), (x_3, x_4), (x_1, x_5), (x_2, x_5), (x_3, x_5)\}$

Por lo cual el conteo de los $n(n-1)/2 = 28$ pares tiene como resultado las cantidades $a = 5, b = 15, c = 2$ y $d = 6$.

Algunos indices de comparación

1. Indice de Rand: $R(\mathscr{C},\mathscr{C}') = \frac{a+b}{a+b+c+d} = \frac{2(a+b)}{n(n-1)}$
2. Indice de Rand Ajustado: $R_{adj}(\mathscr{C},\mathscr{C}') = \frac{a-((a+d)(a+c)/(a+b+c+d))}{\frac{(a+d)+(a+c)}{2} - \frac{(a+d)(a+c)}{a+b+c+d}}$
3. Indice de Fowlkes-Mallows: $FM(\mathscr{C},\mathscr{C}') = \frac{a}{\sqrt{(a+c)(a+d)}}$
4. Indice de Jaccard: $J(\mathscr{C},\mathscr{C}') = \frac{a}{a+c+d}$

Con los datos del ejemplo:

$$R(\mathscr{C},\mathscr{C}') = 0.72 \quad R_{adj}(\mathscr{C},\mathscr{C}') = 0.36 \quad FM(\mathscr{C},\mathscr{C}') = 0.57 \quad J(\mathscr{C},\mathscr{C}') = 0.38$$

1. Información mutua: $NMI_1(\mathscr{C},\mathscr{C}') = \frac{I(\mathscr{C},\mathscr{C}')}{\sqrt{H(\mathscr{C})H(\mathscr{C}')}}$
2. Minimo Error de Clasificación: $MCE(\mathcal{C},\mathcal{C}') = \min_{\sigma\in S_n} \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\{y_i \neq \sigma(\hat{y}_i)\}}$

Con los datos del ejemplos:

$$NMI_1(\mathscr{C},\mathscr{C}') = 0.670 \quad MCE(\mathcal{C},\mathcal{C}') = 0.375$$

Ejemplo

```
library(clv)
data(iris)
iris.data <- iris[,1:4]

# cluster data
pam.mod <- pam(iris.data,3) # create three clusters
v.pred <- as.integer(pam.mod$clustering) # get cluster ids associated to gi
v.real <- as.integer(iris$Species) # get also real cluster ids

# compare true clustering with those given by the algorithm
# 1. optimal solution:

# use only once std.ext function
std <- std.ext(v.pred, v.real)
# to compute three indicies based on std.ext result
rand1 <- clv.Rand(std)
jaccard1 <- clv.Jaccard(std)
folk.mal1 <- clv.Folkes.Mallows(std)
```