Clustering

Eduardo Morales, Hugo Jair Escalante

INAOE

Clustering

- *Clustering* es el proceso de agrupar datos en clases o *clusters* de tal forma que los objetos de un *cluster* tengan una similaridad alta entre ellos, y baja (sean muy diferentes) con objetos de otros *clusters*.
- La medida de similaridad está basada en los atributos que describen a los objetos.
- Los grupos pueden ser exclusivos, con traslapes, probabilísticos, jerárquicos.
- *Clustering* puede ser aplicado, por ejemplo, para caracterizar clientes, formar taxonomías, clasificar documentos, etc.

Retos

- Escalabilidad: Normalmente corren con pocos datos.
- Capacidad de manejar diferentes tipos de atributos: Numéricos (lo más común), binarios, nominales, ordinales, etc.
- *Clusters* de formas arbitrarias: Los basados en distancias numéricas tienden a encontrar *cluster* esféricos.
- Requerimientos mínimos para especificar parámetros, como el número de *clusters*.
- Manejo de ruido: Muchos son sensibles a datos erróneos.
- Independientes del orden de los datos.
- Poder funcionar eficientemente con alta dimensionalidad.
- Capacidad de añadir restricciones.
- Que los *clusters* sean interpretables y utilizables.

Clustering

- La medida de similaridad se define normalmente por proximidad en un espacio multidimensional.
- Para datos numéricos, usualmente se pasa primero por un proceso de estandarización.
- La medida *z* (*z-score*) elimina las unidades de los datos:

$$z_{if} = \frac{x_{if} - \mu_f}{\sigma_f}$$

Clustering

- Donde, σ_f es la desviación media absoluta de la variable f, μ_f es su media y x_{if} es el i-ésimo valor de f.

$$\sigma_f = \frac{1}{n}(|x_{1f} - \mu_f| + |x_{2f} - \mu_f| + \ldots + |x_{nf} - \mu_f|)$$

$$\mu_f = \frac{1}{n}(x_{1f} + x_{2f} + \ldots + x_{nf})$$

Medidas de similaridad

Existen medidas para:

- Variables numéricas
- Variables binarias
- Variables nominales
- Variables ordinales
- Variables escalares no lineales
- Variables mixtas

Variables numéricas (lineales)

- Euclideana:

$$d(i,j) = \sqrt{|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \ldots + |x_{in} - x_{jn}|^2}$$

- Manhattan:

$$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \ldots + |x_{in} - x_{jn}|$$

- Minkowski (Si $q = 1$ es Manhattan y si $q = 2$ es Euclideana)

$$d(i,j) = (|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + \ldots + |x_{in} - x_{jn}|^q)^{1/q}$$

- Distancia Pesada (e.g., Euclideana):

$$d(i,j) = \sqrt{w_1|x_{i1} - x_{j1}|^2 + w_2|x_{i2} - x_{j2}|^2 + \ldots + w_n|x_{in} - x_{jn}|^2}$$

Propiedades de distancias: (i) $d(i,j) \geq 0$, (ii) $d(i,i) = 0$, (iii) $d(i,j) = d(j,i)$, y (iv) $d(i,j) \leq d(i,h) + d(h,j)$.

Variables Binarias (0,1)

- Simétricas (ambos valores tienen el mismo peso):

$$d(i,j) = \frac{r+s}{q+r+s+t}$$

donde:

  - q = número de valores que son 1 en las dos
  - r = número de valores que son 1 en i y 0 en j
  - s = número de valores que son 0 en i y 1 en j
  - t = número de valores que son 0 en las dos.

- No-simétricas (el más importante y más raro vale 1), conocido como el coeficiente Jaccard:

$$d(i,j) = \frac{r+s}{q+r+s}$$

Variables nominales

- Por ejemplo, “color”:

$$d(i,j) = \frac{p - m}{p}$$

  donde: m = número de valores iguales, p = número total de casos.
- Se pueden incluir pesos para darle más importancia a m.
- Se pueden crear nuevas variables binarias asimétricas a partir de las nominales (e.g., es amarillo o no).

Variables ordinales

- Son como las nominales pero con un orden relevante. El orden es importante, pero no la magnitud.
- Pasos:
  1. Cambia el valor de cada variable por un ranqueo $r_{if} \in \{1, \ldots, M_f\}$, donde M_f es el índice del valor más alto de la variable
  2. Mapea el ranqueo entre 0 y 1 para darle igual peso

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

  3. Usa cualquiera de las medidas numéricas anteriores.

Variables escalares no lineales

- Son variables que siguen una escala no lineal, por ejemplo, una escala exponencial
- Posibilidades:
  1. Tratalas como numérica normal.
  2. Obten su logaritmo (o algúna otra transformación) antes para convertirlas en lineales.
  3. Consideralas como variables ordinales.

Variables mixtas

- Una posibilidad es escalar todas las variables a un intervalo común (entre 0 y 1):

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

donde:
$\delta_{ij}^{(f)} = 0$ si x_{if} o x_{jf} se desconocen o si los dos valores son 0 y la variable es asimétrica binaria. En caso contrario vale 1.
$d_{ij}^{(f)}$ depende del tipo:

Variables mixtas

- Si f es binaria o nominal: $d_{ij}^{(f)} = 0$ si $x_{if} = x_{jf}$, si no, $d_{ij}^{(f)} = 1$.
- Si f es numérica lineal: $d_{ij}^{(f)} = \frac{|x_{if} - x_{jf}|}{max_h x_{hf} - min_h x_{hf}}$
- Si f es ordinal o numérica no lineal: calcula los índices r_{if} y $z_{if} = \frac{r_{if}-1}{M_f-1}$ y toma a z_{if} como numérica lineal.

HVDM

- Otra medida popular entre dos vectores $\vec{x}, \vec{y}$ es HVDM (*Heterogeneous Value Difference Metric*) definida como:

$$HVDM(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{m} d_a^2(x_i, y_i)}$$

donde m es el número de atributos y

$$d_a(x, y) = \begin{cases} 1 & \text{si } x \text{ o } y \text{ son desconocidas} \\ norm\_vdm_a(x, y) & \text{si } a \text{ es nominal} \\ norm\_diff_a(x, y) & \text{si } a \text{ es numérico} \end{cases}$$

HVDM

- Como el 95% de los valores de una distribución normal están dentro de dos desviaciones estándar de la media, la distancia se divide entre 4 σ_a

$$norm\_diff_a(x, y) = \frac{|x - y|}{4\sigma_a}$$

- Originalmente se propusieron 3 medidas:

$$N1 : norm\_vdm1_a(x, y) = \sum_{c=1} C \left| \frac{N_{a,x,c}}{N_{a,x}} - \frac{N_{a,x,c}}{N_{a,x}} \right|$$

$N_{a,x}$ = número de instancias con valor x en el atributo a
$N_{a,x,c}$ = número de instancias con valor x en el atributo a y clase c
C = número de clases

HVDM

- N2: Equivalente a Euclideana y Manhattan (N1)

$$N2 : norm\_vdm2_a(x, y) = \sqrt{\sum_{c=1} C \left| \frac{N_{a,x,c}}{N_{a,x}} - \frac{N_{a,x,c}}{N_{a,x}} \right|^2}$$

- N3: Original

$$N3 : norm\_vdm3_a(x, y) = \sqrt{C * \sum_{c=1} C \left| \frac{N_{a,x,c}}{N_{a,x}} - \frac{N_{a,x,c}}{N_{a,x}} \right|^2}$$

Algoritmos de Clustering

Existe una gran cantidad de algoritmos de *clustering* (sólo vamos a ver algunos). En particular existen diferentes algoritmos basados en:

- Paticiones
- Jerárquicos
- Densidades
- Rejillas
- Modelos
- Teoría de grafos
- Búsqueda combinatoria
- Técnicas Fuzzy
- Redes neuronales
- Kernels
- Para datos secuenciales
- Para grandes conjuntos de datos

Métodos basados en paticiones

- Construyen *k* particiones de los datos, donde cada partición representa un grupo o *cluster*
- Cada grupo tiene al menos un elemento y cada elemento pertenece a un solo grupo.
- Estos métodos, crean una partición inicial e iteran hasta un criterio de paro
- Los más populares son *k*-medias y *k*-medianas (otros: CLARA y CLARANS).

Métodos Jerárquicos

- Crean descomposiciones jerárquicas
- Existen dos tipos:
  1. El método aglomerativo o *bottom-up*, empieza con un grupo por cada objeto y une los grupos más parecidos hasta llegar a un solo grupo u otro criterio de paro (e.g., AGNES, BIRCH, CURE, ROCK).
  2. El método divisorio o *top-down*, empieza con un solo grupo y lo divide en grupos más pequeños hasta llegar a grupos de un solo elemento u otro criterio de paro (e.g., DIANA, MONA).

Otros Métodos

- Métodos basados en densidades: Se agrupan objetos mientras su densidad (número de objetos) en la "vecindad" esté dentro de un cierto umbral (e.g., DBSCAN, DENCLUE).
- Métodos basados en rejillas: Se divide el espacio en rejillas a diferentes niveles (e.g, STING, CLIQUE).
- Métodos basados en modelos: Se encuentra un modelo para cada *cluster* que mejor ajuste los datos de ese grupo (e.g., COBWEB, AutoClass).

Otros Métodos

- Métodos basados en teoría de grafos: Utilizan representaciones basadas en grafos (e.g., Chameleon, *Delaunay triangulation graph* (DTG), *highly connected subgraphs* (HCS), *clustering identification via connectivity kernels* (CLICK), *cluster affinity search technique* (CAST))
- Técnicas basadas en Búsqueda Combinatoria (e.g., *Genetically guided algorithm* (GGA), TS *clustering*, SA *clustering*)
- Técnicas Fuzzy (e.g., Fuzzy c-means (FCM), *mountain method* (MM), *possibilistic c-means clustering algorithm* (PCM), *fuzzy c-shells* (FCS))

Otros Métodos

- Técnicas basadas en Redes Neuronales (e.g., *Learning vector quantization* (LVQ), *self-organizing feature map* (SOFM), ART, *simplified ART* (SART), *hyperellipsoidal clustering network* (HEC), *self-splittting competitive learning network* (SPLL))
- Técnicas basadas en Kernels (e.g. *Kernel K-means*, *support vector clustering* (SVC))
- Técnicas para Datos Secuenciales (e.g. Similaridad secuencial, *clustering* secuencial indirecto, *clustering* secuencial estadístico)
- Técnicas para grandes conjuntos de datos (e.g., CLARA, CURE, CLARANS, BIRCH, DBSCAN, DENCLUE, WaveCluster, FC, ART)

k-Means

- Es de los más conocidos y usados
- Toma como parámetro *k* que es el número de *clusters* que forma.
- Pasos:
  1. Selecciona *k* elementos aleatoriamente, los cuales representan el centro o media de cada *cluster*.
  2. A cada objeto restante se le asigna el *cluster* con el cual más se parece, basándose en una distancia entre el objeto y la media del *cluster*
  3. Despúes calcula la nueva media del *cluster* e itera hasta no cambiar de medias.

Algoritmo de k-means

Selecciona *k* objetos aleatoriamente
repeat
    Re(asigna) cada objeto al *cluster* más similar
        con el valor medio
    Actualiza el valor de las medias de los *clusters*
until no hay cambio

k-means

- Normalmente se utiliza un medida de similaridad basada en el error cuadrático:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2$$

  donde: p representa al objeto y m_i a la media del *cluster* C_i (ambos son objetos multidimensionales).
- *k-means* es susceptible a valores extremos porque distorcionan la distribución de los datos.
- Tambíen se pueden utilizar las modas (*k-modes*) para agrupar objetos categóricos.

k-Medianas

- Otra posibilidad es usar medianas (*k-medoids*) para agrupar con base al objeto más representativo del *cluster*
- La idea básica es encontrar un objeto representativo
- La estrategia es reemplazar una de las medianas por otro objeto en forma aleatoria y medir si la calidad de los *clusters* resultantes mejoran
- La calidad se evalúa con base en una función de costo que mide la disimilaridad promedio entre un objeto y la mediana en su *cluster*.

k-Medianas

- Para ver si un objeto aleatorio es un buen reemplazo de la mediana actual, se consideran todos los objetos que no sean medianas y se analiza la re-distribución de los objetos a partir de la cual se calcula un costo basado, por ejemplo, en el error cuadrático
- Esto se repite hasta que no exista mejora.
- Cómo en muchos de los métodos vistos, no garantiza encontrar el mínimo global, por lo que se recomienda correr varias veces el algoritmo con diferentes valores iniciales.
- Otra variante es hacer un *k-means* jerárquico, en donde se empieza con $k = 2$ y se continua formando *clusters* sucesivos en cada rama.
- Si queremos escalarlo a grandes bases de datos, podemos tomar únicamente muestras de los datos.

Cobweb

- Crea un *cluster* jerárquico con un árbol de clasificación.
- En un árbol de clasificación cada nodo es un concepto que tiene una descipción probabilística de ese concepto que resume los objetos clasificados bajo ese nodo.
- La descripción probabilística incluye la probabilidad del concepto ($P(C_i)$) y las probabilidades condicionales de pares atributos-valor dado el concepto ($P(A_i = V_{ij}|C_k)$).

Cobweb

- Utiliza una medida llamada *utilidad de la categoria* para construir el árbol:

$$CU = \frac{\sum_{k=1}^{n} P(C_k) \left[\sum_i \sum_j P(A_i = V_{ij} | C_k)^2 - \sum_i \sum_j P(A_i = V_{ij})^2 \right]}{n}$$

donde: n es el número de clases en un nivel del árbol.

- La utilidad de la categoria mide el valor esperado de valores de atributos que pueden ser adivinados a partir de la partición sobre los valores que se pueden adivinar sin esa partición.

Cobweb

- Si la partición no ayuda en esto, entonces no es buena partición.
- Entre más grande es la proporción de elementos de la clase que tienen ese atributo-valor, ese atributo-valor es más predictivo sobre la clase.
- COBWEB desciende el árbol buscando el mejor lugar o nodo para cada objeto
- Esto se basa en poner el objeto en cada nodo y en un nodo nuevo y medir en cual se tiene la mayor ganancia de utilidad de categoría.

Cobweb

- COBWEB también considera en cada iteración unir los dos mejores nodos evaluados y dividir el mejor nodo evaluado
- Esto es, cada vez que se considera un lugar en un nivel para un nuevo objeto, se consideran los dos mejores objetos (de mayor utilidad) y se considera juntarlos.
- El caso contrario, sucede una vez que se encuentra el mejor lugar para un nuevo objeto, pero el unir nodos no resulta beneficioso, entonces se considera dividir ese nodo.

Cobweb

- COBWEB depende del orden de los objetos, por lo que a veces es conveniente probarlo con objetos en diferente orden.
- La división entre el número de *cluster* sirve para incentivar tener *clusters* con más de un elemento.
- COBWEB supone que la distribución de probabilidad de los atributos es independiente de las demás.

Classit

- Cobweb se puede extender a valores numéricos usando gaussianas (CLASSIT).

$$f(a) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(a-\mu)^2}{2\sigma^2}}$$

- El equivalente a la sumatoria de probabilidades es:

$$\sum_j P(A_i = V_{ij})^2 \sim \int f(a_i)^2 da_i = \frac{1}{2\sqrt{\pi}\sigma_i}$$

- Ahora se estima la desviación estandar del atributo numérico con los datos en el *cluster* y en los datos para todos los *clusters*:

$$CU = \frac{1}{k} \sum_{k=1}^{n} P(C_k) \frac{1}{2\sqrt{\pi}} \sum_i \left(\frac{1}{\sigma_{ik}} - \frac{1}{\sigma_i} \right)$$

Classit

- Si la desviación estandar es cero el valor de utilidad se vuelve infinito, por lo que se impone un valor de varianza mínimo en cada atributo (*acuity*).
- El otro parámetro que se usa en COBWEB es el de corte (*cutoff*), que básicamente se usa para parar la generación de nuevos nodos.

Clustering basado en probabilidades

- Desde el punto de vista bayesiano, lo que buscamos es el grupo de *clusters* más probables dados los datos.
- Ahora los objetos tienen cierta probabilidad de pertenecer a un grupo o *cluster*.
- La base de un *clustering* probilístico está basado en un modelo estadístico llamado *finite mixtures* (mezcla de distribuciones).
- Una mezcla es un conjunto de k distribuciones, representando k *clusters*.

Clustering basado en probabilidades

- Cada distribución nos da la probabilidad de que un objeto tenga un conjunto particular de pares atributo-valor si se supiera que fuera miembro de ese *cluster*.
- La mezcla más sencilla es cuando tenemos puros atributos numéricos con distribuciones gaussianas con diferentes medias y varianzas.
- La idea es, dado un conjunto de datos, determinar las k distribuciones normales (medias y varianzas) y las probabilidades particulares de cada distribución (pueden ser diferentes).

Mezcla de Gaussianas

- Por ejemplo, si tuvieramos dos distribuciones A y B con μ_A, σ_A y μ_B, σ_B, y P_A ($P_A + P_B = 1$), podemos generar un conjunto de datos.
- Si supieramos de qué distribución salió cada dato, es fácil calcular su media y varianza, y las P_A y P_B.

$$\mu = \frac{x_1 + x_2 + \ldots x_n}{n}$$

$$\sigma^2 = \frac{(x_1 - \mu)^2 + (x_2 - \mu)^2 + \ldots (x_n - \mu)^2}{n - 1}$$

Mezcla de Gaussianas

- Calcular la probabilidad de que un objeto (x) pertenezca a un *cluster* (e.g., A), es:

$$P(A|x) = \frac{P(x|A)P(A)}{P(x)} = \frac{f(x;\mu_A,\sigma_A)P_A}{P(x)}$$

donde $f(x;\mu_A,\sigma_A)$ es una distribución normal:

$$f(x;\mu_A,\sigma_A) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Podemos ignorar $P(x)$ y al final normalizar.

Algoritmo EM

- El problema es que no sabemos de qué distribución viene cada dato y no concemos los parámetros de las distribuciones.
- El algoritmo EM (*Expectation Maximization*) empieza adivinando los parámetros de las distribuciones y los usa para calcular las probabilidades de que cada objeto pertenezca a un *cluster*
- Usa esas probabilidades para re-estimar los parámetros de las probabilidades, hasta converger (se puede empezar adivinando las probabilidades de que un objeto pertenezca a una clase).

Algoritmo EM

- El cálculo de las probabilidades de las clases o los valores esperados de las clases es la parte de *expectation*.
- El paso de calcular los valores de los parámetros de las distribuciones, es *maximization*, maximar la verosimilitud de las distribuciones dados los datos.
- Para estimar los parámetros, tenemos que considerar que tenemos únicamente las probabilidades de pertenecer a cada *cluster* y no los *clusters* en si.

Algoritmo EM

- Estas probabilidades actúan como pesos:

$$\mu_A = \frac{w_1 x_1 + w_2 x_2 + \ldots w_n x_n}{w_1 + w_2 + \ldots w_n}$$

$$\sigma_A^2 = \frac{w_1 (x_1 - \mu)^2 + w_2 (x_2 - \mu)^2 + \ldots w_n (x_n - \mu)^2}{w_1 + w_2 + \ldots w_n}$$

donde w_i es la probabilidad de que el objeto i pertenezca al *cluster A* y se suma sobre todos los objetos (no sólo los de A).

Algoritmo EM

- El algoritmo tiende a converger pero nunca llega a un punto fijo.
- Podemos ver que tanto se acerca calculando la versorimilitud general de los datos con esos parámetros, multiplicando las probabilidades de los objetos individuales (i):

$$\prod_i (P_A P(x_i|A) + P_B P(x_i|B))$$

- Esta medida crece en cada iteración, y se itera hasta que el crecimiento es despreciable.
- Aunque EM garantiza convergencia, esta puede ser a un máximo local, por lo que se recomienda repetir el proceso varias veces.

Extensiones

- Extender a más de dos distribuciones es prácticamente directo.
- Extenderlo a instancias con atributos múltiples, si se supone independencia de los atributos, se puede hacer multiplicando las probabilidades para obtener una distribución de probabilidad conjunta.
- Si existen dos atributos correlacionados, se pueden analizar con una distribución normal bi-variable en donde se utiliza una matriz de covarianza
- El número de parámetros crece al cuadrado del número de atributos que se consideren correlacionados entre sí

Extensiones

- Se pueden especificar diferentes distribuciones (cada una con sus propios parámetros) para diferentes tipos de datos.
- Se puede penalizar el modelo que instroduzca parámetros y el que defina un número mayor de *clusters*.

AutoClass

- Realiza un descubrimiento automático de clases en datos (P. Cheeseman, J. Stutz).
- Una vez que las clases han sido identificadas, éstas pueden servir para clasificar nuevos datos.
- La idea es encontrar la hipótesis más probable, dados los datos e información *a priori*.

AutoClass

- Normalmente se busca un balance entre qué tan bien se ajustan los datos a las clases y complejidad de las clases (casos extremos, una clase por dato o una sola clase para todos los datos).
- En AutoClass los datos se pueden representar por valores discretos, enteros y reales.
- El modelo es una mezcla finita de distribuciones de probabilidad, cada una con su conjunto de parámetros.

AutoClass

- Para cada dato se asigna una probabilidad de pertenencia a una clase (o un peso).
- Dado un conjunto de datos se busca:
  1. Los valores más probables (MAP) de los parámetros (para las distribuciones y clases dadas), dada una distribución de probabilidad.
  2. La distribución de probabilidad más probable (número de clases y modelos alternativos), independientemente de los parámetros.

AutoClass

- Se supone que los datos son condicionalmente independientes dada la clase, por lo que la distribución conjunta de los datos es el producto de las probabilidades individuales.
- Cada dato pertenece a una y solo una clase (de un conjunto disjunto de clases) con probabilidad $P(X_i \in C_j \mid \vec{V}_c, T_c)$, donde $\vec{V}_c$ es el vector de parámetros de la distribución y T_c es la distribución particular).
- Las clases representan una partición discreta de los datos y por lo tanto la distribución más apropiada es una distribución Bernoulli o binomial

AutoClass

AutoClass trata de encontrar los parámetros de las distribuciones de probabilidad dependiendo del tipo de valores de las variables:

- Discretos: Bernoulli
- Reales: Gaussianas
- Reales - Escalares (e.g., edad, peso): log-Gaussianas
- Enteros: Poisson

AutoClass

- En general se debe de establecer cuantas clases y correr el proceso.
- Al correrlo, existen muchos máximos locales, por lo que hay que correr el proceso varias veces a partir de diferentes valores iniciales para los parámetros.

¿Cuántos *Clusters*?

- Para algunas aplicaciones es fácil determinar el número de *clusters*, ”K”, de acuerdo al conocimiento del dominio.
- Para la mayoría de los casos, ”K” se desconoce y se estima a partir de los datos.
- Muchos algoritmos de *clustering* requieren a ”K” como parámetro de entrada y la calidad de los resultados está fuértemente ligada a este valor.

¿Cuántos *Clusters*?

- Una división con muchos *clusters* complica los resultados porque los hace difíciles de interpretar y analizar.
- Una división con muy pocos *clusters* lleva a una pérdida de información y puede llevar a tomar malas decisiones.
- Al problema de determinar el número de *clusters* se le conoce como “el problema fundamental de la validez del *cluster*”

Número de Clusters

Algunos métodos que se han utilizado para encontrar el número adecuado de *clusters* son:

- Visualización del conjunto de datos, lo que funciona bien para dos dimensiones pero generalmente nuestros conjuntos de datos son mucho más complicados.
- Construcción de índices (o reglas de paro). En este caso se utilizan índices para enfatizar la compactés intra-*cluster* e isolación inter-*cluster* considerando efectos tales como: el error cuadrático, propiedades geométricas o estadísticas de los datos, el número de patrones, la disimilaridad o similaridad, número de *clusters*.

Número de Clusters

- Optimización de alguna función de criterio bajo el marco del modelo de mezcla de probabilidades.
- En este caso se utiliza el algoritmo EM (usualmente), para encontrar el valor de ”K” que maximize o minimize el criterio definido como óptimo.
  - Criterio de Información de Akaike (AIC).
  - Criterio de Inferencia Bayesiana.
- Otros métodos heurísticos basados en una variedad de técnicas y teorías.

X-means

Variante de K-means para determinar automáticamente el número de clusters

Algoritmo X-Means (rango de K : $[K_{min} \ldots K_{max}]$)

1. Mejora parámetros (corre K-means)
2. Mejora estructura (ver abajo)
3. Si $K > K_{max}$ termina y regresa el modelo mejor evaluado, si no regresa a 1

Mejora Estructura

- Divide cada cluster en 2 (a una distancia proporcional a su tamaño a lo largo de un vector aleatorio)
- Corre un K-means local, con K=2
- Evalúa si la medida mejora (con 2 clusters) o no (cluster original)

Medida

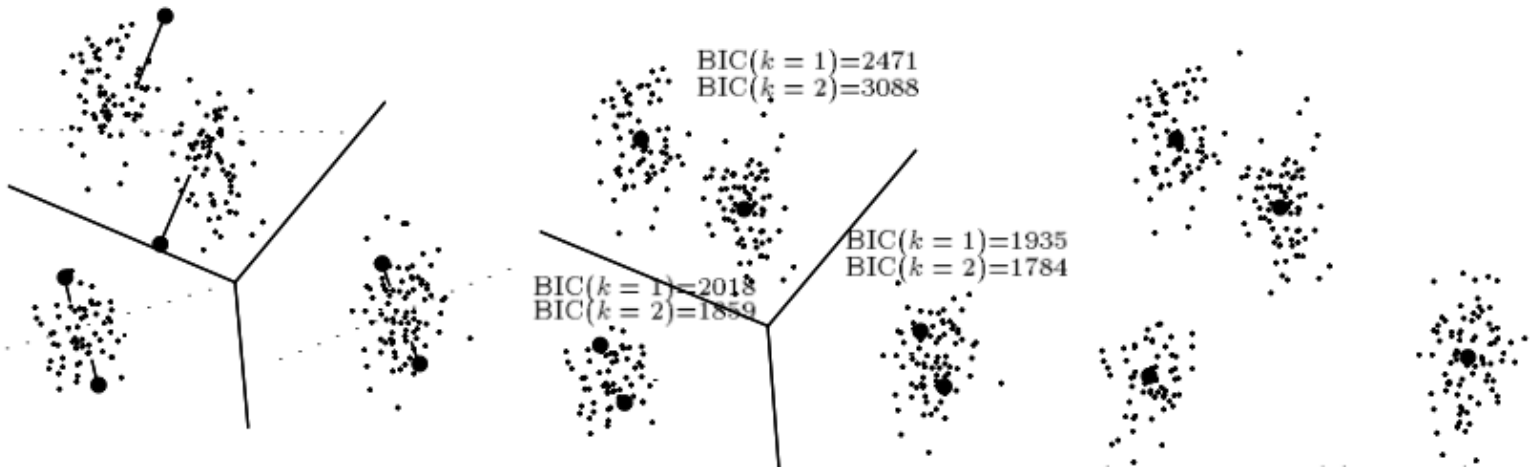
$$BIC(M_j) = \hat{l}(D_j) - \frac{p_j}{2} \cdot logR$$

$$\hat{l}(D_n) = -\frac{R_n}{2} log(2\pi) - \frac{R_n \cdot M}{2} log(\hat{\sigma}^2) - \frac{R_n - K}{2} + R_n logR_n - R_n logR$$

$$\hat{\sigma}^2 = \frac{1}{R-K} \sum_i (x_i - \mu_{(i)})^2$$

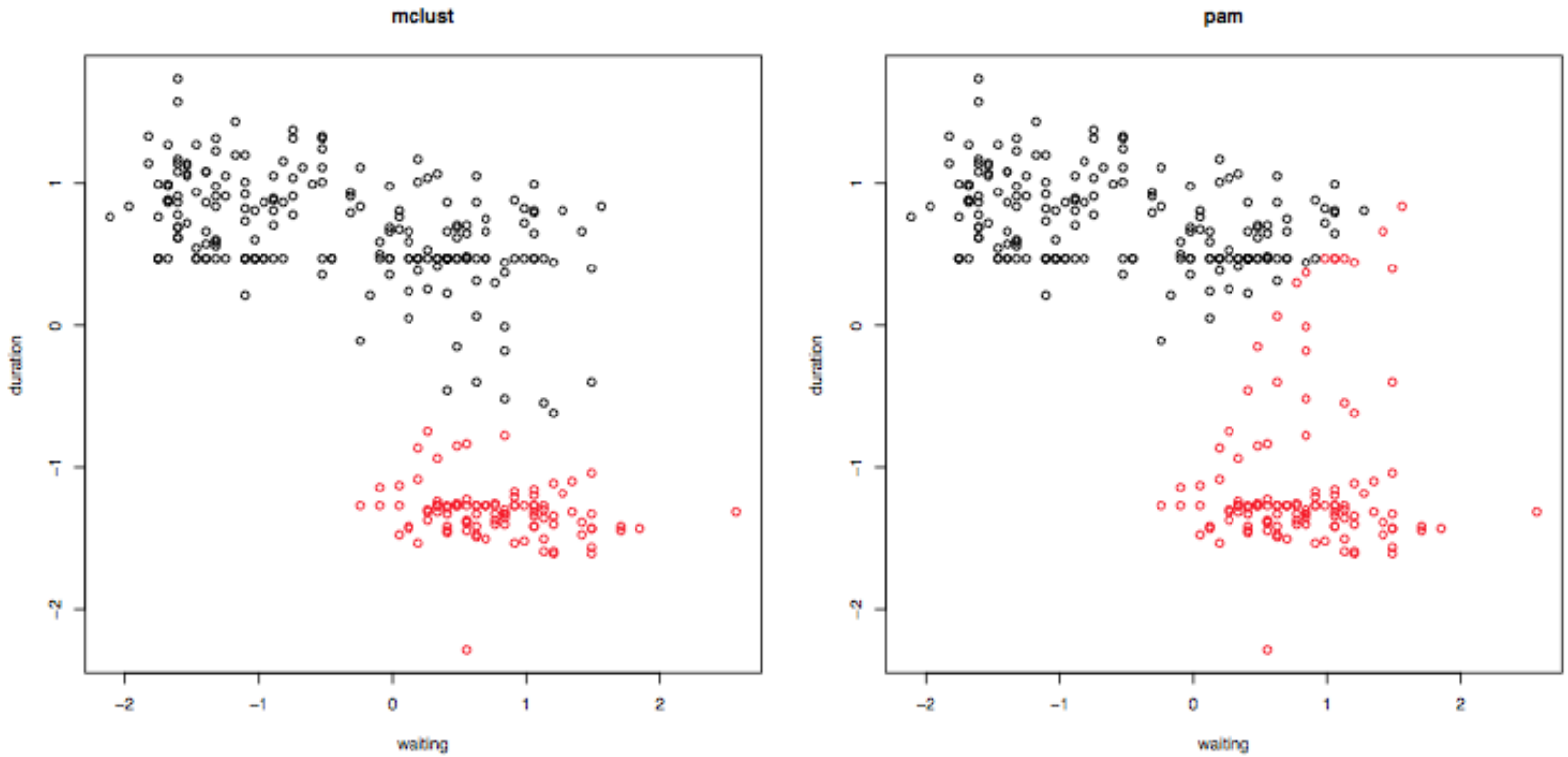
p_j = número de parámetros = $K - 1 + (M \cdot K) + 1$
$R = |D|$ y $R_i = |D_i|$
M = número de dimensiones

X-means

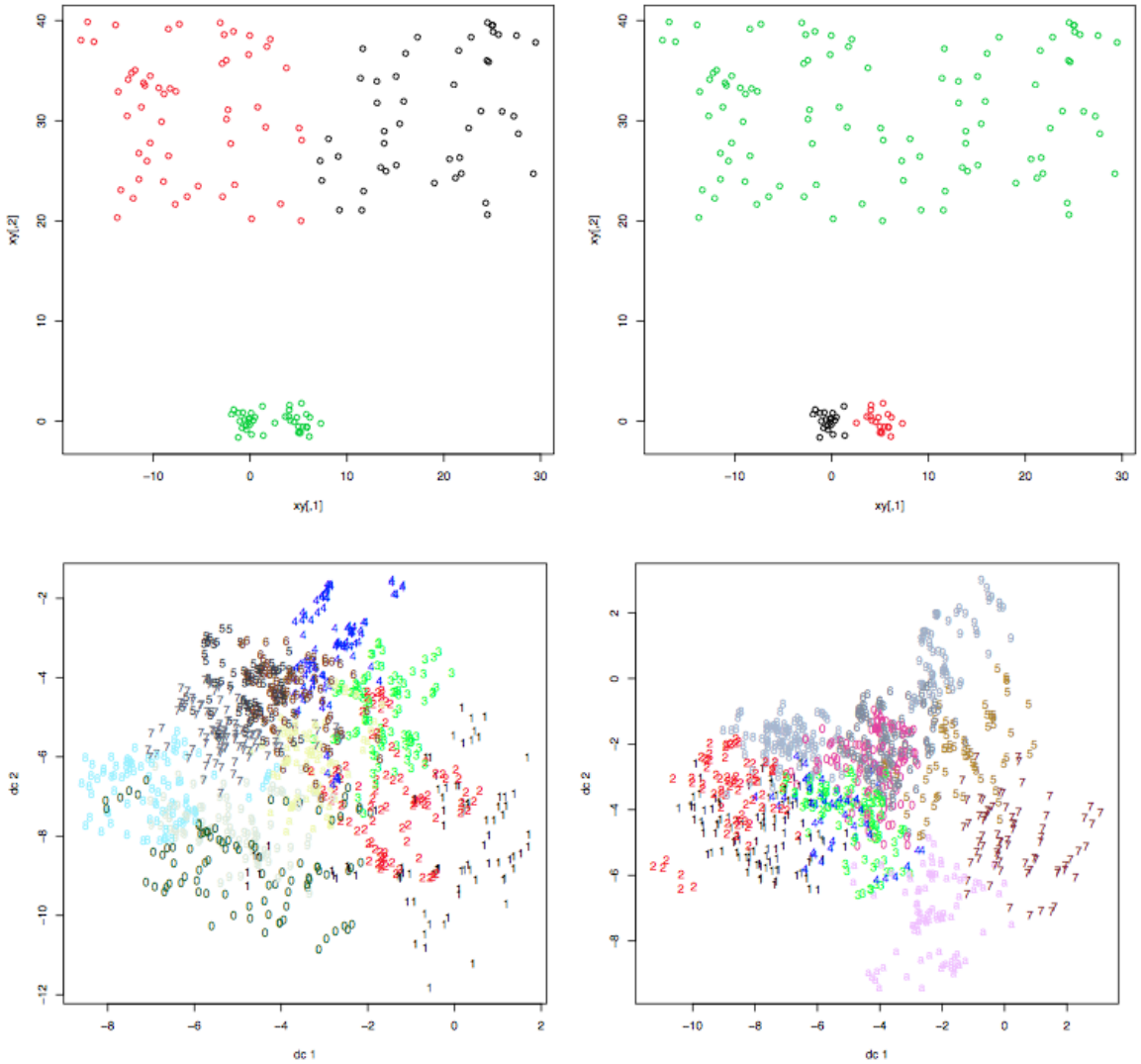


Medidas de Calidad

- La evaluación de los *clusters* también es complicada
- ¿Cuál agrupación es mejor?



¿Cuál agrupación es mejor?



Medidas de Calidad

Existen dos clases:

- Evaluación interna: Entre los grupos generados
- Evaluación externa: Contra grupos conocidos

La evaluación final generalmente la realiza una persona

Índice Davies-Bouldin (interna)

$$DB = \frac{1}{n}\sum_{i=1}^{n} max_{j\neq i}\left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)}\right)$$

- n = número de *clusters*
- c_x = centroide del cluster x
- σ_x = distancia promedio de todos los elementos en el *cluster* x a su centroide c_x y $d(c_i, c_j)$ es la distancia entre centroides c_i y c_j.
- El algoritmo que produce el valor menor entre todos los *clusters* es el mejor

Índuce Dunn (interna)

- Busca indentificar *clusters* densos y claramente separados.

$$D = \frac{min_{1 \leq i < j \leq n} d(i,j)}{max_{1 \leq k \leq n} d'(k)}$$

- $d(i,j)$ = distancia entre *clusters* i y j (intra-clusters), que puede ser la distancia entre centroides.
- $d'(k)$ = distancia intra-cluster de cluster k. Puede ser la distancia máxima entre pares de elementos del cluster.
- Grupos con valores mayores del índice son mejores

Coeficiente de Silueta (interna)

- El coeficiente de silueta (*silhouette*) contrasta la distancia promedio de elementos en el mismo *cluster* con la distancia promedio de elementos en otros *clusters*
- Elementos con alto valor se consideran bien agrupados, mientras que objetos con medidas bajas se consideran *outliers*
- Funciona bien para k-means

Purity (externa)

- La pureza mide en qué medida los *clusters* contienen una sola clase

$$\frac{1}{N}\sum_{m\in M} max_{d\in D}|m\cap d|$$

- M = clusters
- D = clases
- N = datos

Índice Rand (externa)

- Mide que tan parecidos son los *clusters* a las clases

$$RI = \frac{TP + TN}{TP + FP + TN + FN}$$

- TP = true positives
- FP = false positives
- TN = true negatives
- FN = false negatives

F-Measure (externa)

- Puede balancear los falsos negativos usando precisión (P) y recuerdo (R)

$$F_\beta = \frac{(\beta^2 + 1) \cdot P \cdot R}{\beta^2 \cdot P + R}$$

$$P = \frac{TP}{TP + FP}$$

$$R = \frac{TP}{TP + FN}$$

Índice de Jaccard (externo)

- Mide la similaridad entre dos grupos.
- Los elementos comunes entre los dos grupos entre los elementos de los dos grupos

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{TP}{TP + FP + FN}$$

Otros (externos)

Índice Dice:

$$D(A,B) = \frac{2TP}{2TP + FP + FN}$$

Índice Fowlkes-Mallows: La media geométrica de precisión y recuerdo (también conocida como $G - measure$ (F-Measure es la media armónica)

$$FM = \sqrt{\frac{TP}{TP + FP} \cdot \frac{TP}{TP + FN}}$$