



# Ciencia de Datos

## Clustering

**Dr. José Ramón Iglesias**

DSP-ASIC BUILDER GROUP  
Director Semillero TRIAC  
Ingeniería Electrónica  
Universidad Popular del Cesar

# **Clustering Conglomerados**

**“Clasificación” NO  
supervisada**

# Mapa de Ruta

1. Intuición general de clustering
2. Conocimiento de los Datos e Información Relevante al problema
3. Importancia del conocimiento de dominio
4. Similaridad y/o Distancia entre datos
5. Algoritmos de agrupamiento
6. Evaluación de resultados: Visualización, Medidas y relevancia: utilidad o impacto

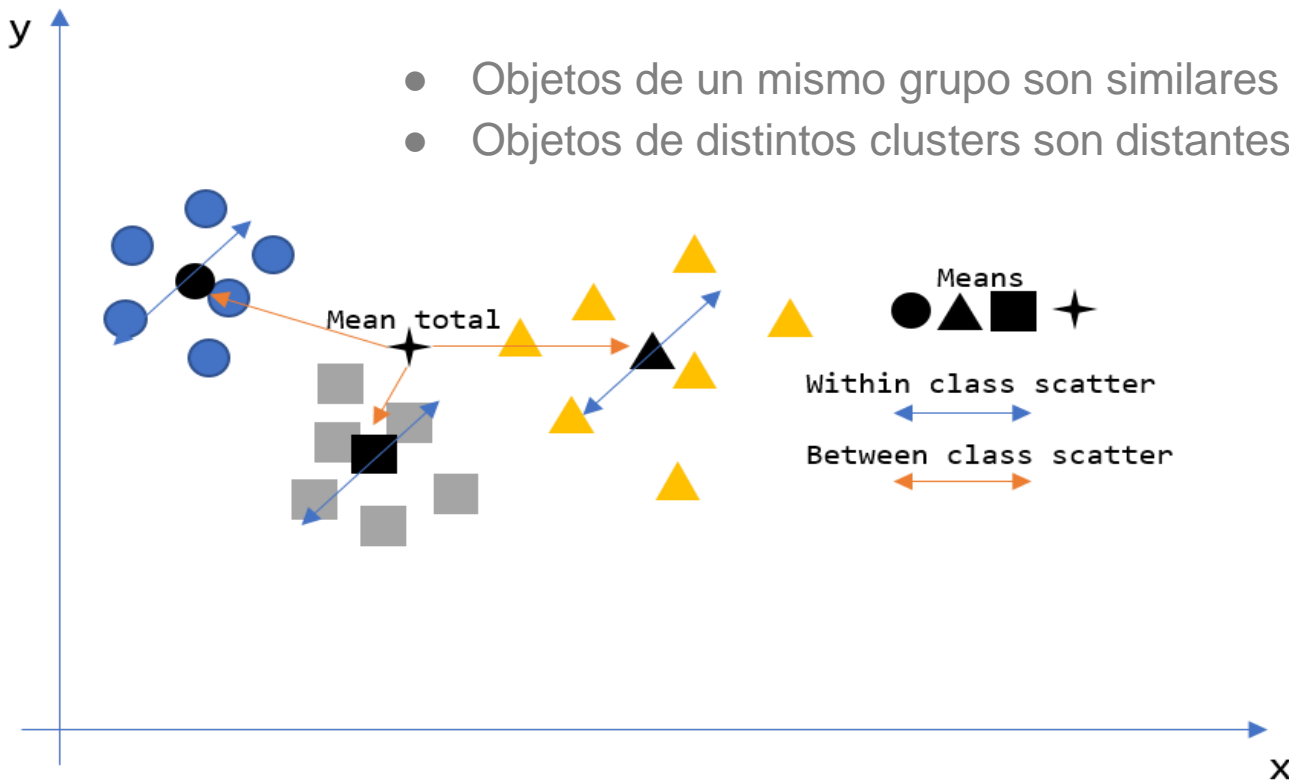
# Clustering o agrupamiento

Agrupar objetos semejantes, parecidos

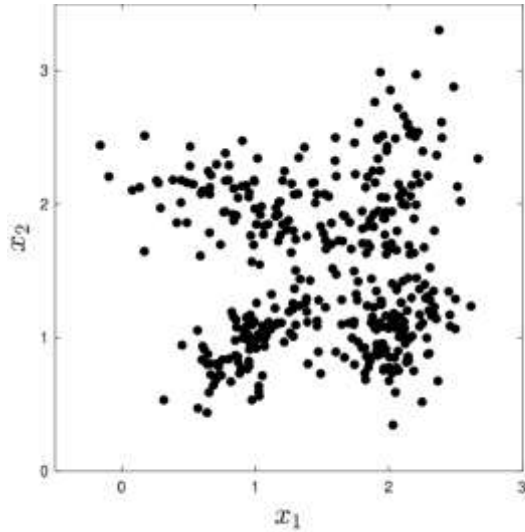
- Entrada:  $n$  objetos o individuos en un espacio  $m$ -dimensional:  
 $X$  en  $\mathbb{R}^{n \times m}$ , cada fila representa un objeto (vector con  $m$  valores)
- Salida: una **solución** con conglomerados (**clusters**) de objetos semejantes (semejantes  $\rightarrow$  cercanos en el espacio o similares)
  - Se minimiza la distancia entre los objetos de un mismo conglomerado
  - Se maximiza la distancia entre los objetos de distintos conglomerados

# Cómo funciona clustering

- Objetos de un mismo grupo son similares
- Objetos de distintos clusters son distantes o distintos



# Mis Datos...



(a)



- ❖ ¿Cómo es el espacio? ¿Cómo represento mis problemas?
- ❖ ¿Cómo se calcula la distancia (semejanza) en este espacio?

# Pre-procesamiento

Para atributos continuos:

- Para evitar que unas variables dominen sobre otras los valores de los atributos se escalan o estandarizan a priori
- `sklearn.preprocessing`: Preprocessing and Normalization
  - **StandardScaler** (z-score), c/variable o columna de media 0 y varianza 1
  - **MinMaxScaler**, por variable
- En algunas aplicaciones conviene normalizar por fila
  - **normalize** (por defecto por fila, c/vector de norma 1, unitario)

Notar que estamos haciendo una transformación de los datos-> Embedding ! Eh??!!

# Pre-procesamiento

- ❖ Atributos categoricos
  - encoding mediante transformaciones
  - <https://scikit-learn.org/stable/modules/preprocessing.html#preprocessing-categorical-features>

Transformación de los datos-> Embedding



# Distancias: datos continuos

- ❖ Euclídea
- ❖ Distancia de Manhattan
- ❖ Distancia de mahalanobis
- ❖ “distancia” del Coseno → primero normalizado por longitud de cada vector/fila,
- ❖ Similitud del coseno: considera el producto punto entre vectores, es alta cuando están alineados

# Distancia Euclídea

$$\underline{x}=(x_1,x_2,\dots,x_m)$$

$$\underline{y}=(y_1,y_2,\dots,y_m)$$

$$\begin{aligned}d(\underline{x},\underline{y})&=[(\underline{x}-\underline{y})(\underline{x}-\underline{y})]^{(\frac{1}{2})}\\&=[(x_1-y_1)^2+(x_2-y_2)^2+\dots+(x_m-y_m)^2]^{(\frac{1}{2})}\end{aligned}$$

$$\underline{x}=(2,2,1)$$

$$\underline{y}=(1,2,1)$$

$$\underline{w}=(8,8,4)$$

$$d(\underline{x},\underline{y})=1=[(2-1)^2+(2-2)^2+(1-1)^2]^{(\frac{1}{2})}$$

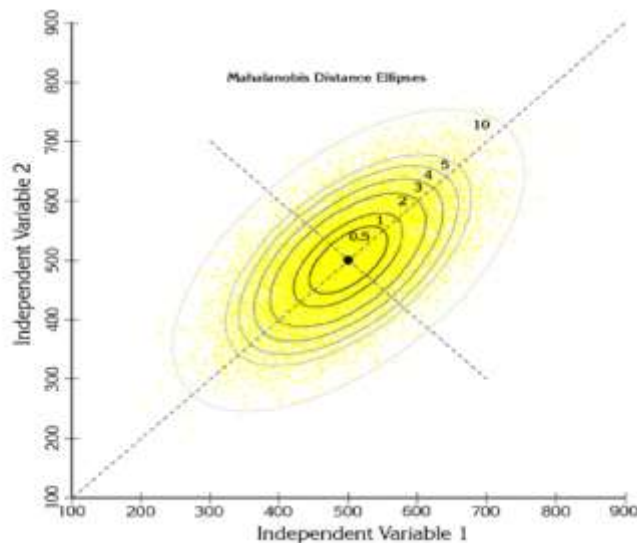
$$d(\underline{x},\underline{w})=9=[(8-2)^2+(8-2)^2+(4-1)^2]^{(\frac{1}{2})}$$

# Distancias: datos continuos

## Distancia de Mahalanobis

$$d(\vec{x}, \vec{y}) = \sqrt{(\vec{x} - \vec{y})^T \Sigma^{-1} (\vec{x} - \vec{y})}.$$

- Considera las correlaciones entre variables.
- No depende de la escala de medida.



# Distancias: datos continuos

## Distancia de Minkowski

$$d_r(x, y) = \left( \sum_{j=1}^J |x_j - y_j|^r \right)^{\frac{1}{r}}, \quad r \geq 1$$

- Distancia de Manhattan (r=1) / city block / taxicab

$$d_1(x, y) = \sum_{j=1}^J |x_j - y_j|$$

- Distancia euclídea (r=2):

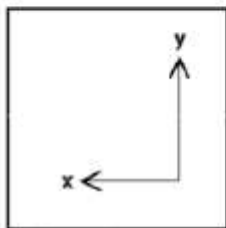
$$d_2(x, y) = \sqrt{\sum_{j=1}^J (x_j - y_j)^2}$$

- Distancia de Chebyshev (r→∞) / dominio / chessboard

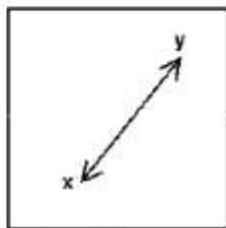
$$d_\infty(x, y) = \max_{j=1..J} |x_j - y_j|$$

# Distancias: datos continuos

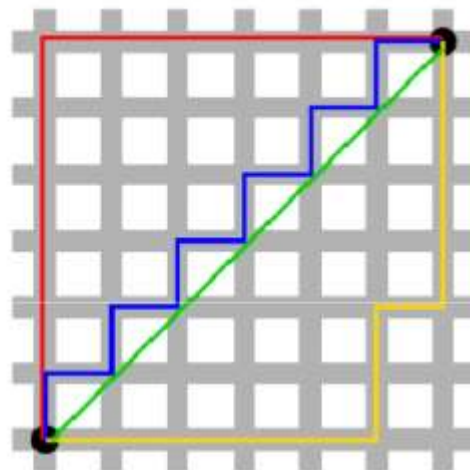
## Distancia de Minkowski



Manhattan



Euclidean



- Distancia de Manhattan = 12
- Distancia euclídea  $\approx 8.5$
- Distancia de Chebyshev = 6

(roja, azul o amarilla)

(verde - continua)


(verde - discreta)

# Distancias: datos continuos

## Distancia de Chebyshev

$$d_{\infty}(x, y) = \max_{j=1..J} |x_j - y_j|$$

También conocida como distancia de tablero de ajedrez (chessboard distance):  
Número de movimientos que el rey ha de hacer para llegar de una casilla a otra en un tablero de ajedrez.

	a	b	c	d	e	f	g	h
8	5	4	3	2	2	2	2	2
7	5	4	3	2	1	1	1	2
6	5	4	3	2	1		1	2
5	5	4	3	2	1	1	1	2
4	5	4	3	2	2	2	2	2
3	5	4	3	3	3	3	3	3
2	5	4	4	4	4	4	4	4
1	5	5	5	5	5	5	5	5
	a	b	c	d	e	f	g	h

# Distancias: datos discretos

## Distancia de edición = Distancia de Levenshtein

Número de operaciones necesario  
para transformar una cadena en otra.

$d(\text{"data mining"}, \text{"data minino"}) = 1$

$d(\text{"efecto"}, \text{"defecto"}) = 1$

$d(\text{"poda"}, \text{"boda"}) = 1$

$d(\text{"night"}, \text{"natch"}) = d(\text{"natch"}, \text{"noche"}) = 2$

Aplicaciones: Correctores ortográficos, reconocimiento de voz,  
detección de plagios, análisis de ADN...

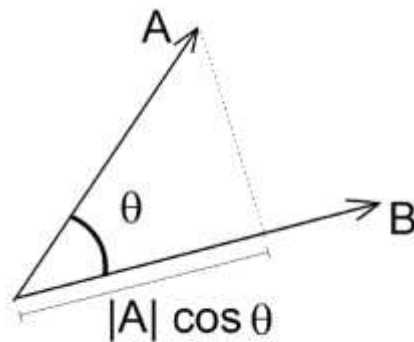
Para datos binarios: Distancia de Hamming

# Similaridades

## Medidas de correlación

- Producto escalar

$$S.(x, y) = x \cdot y = \sum_{j=1}^J x_j y_j$$



- "Cosine similarity"

$$\cos(\vec{x}, \vec{y}) = \sum_i \frac{x_i \cdot y_i}{\sqrt{\sum_i x_i^2} \cdot \sqrt{\sum_i y_i^2}}$$

- Coeficiente de Tanimoto

$$s(\vec{X}, \vec{Y}) = \frac{\vec{X}^t \cdot \vec{Y}}{\vec{X}^t \cdot \vec{X} + \vec{Y}^t \cdot \vec{Y} - \vec{X}^t \cdot \vec{Y}},$$



# Similaridad vs Distancia

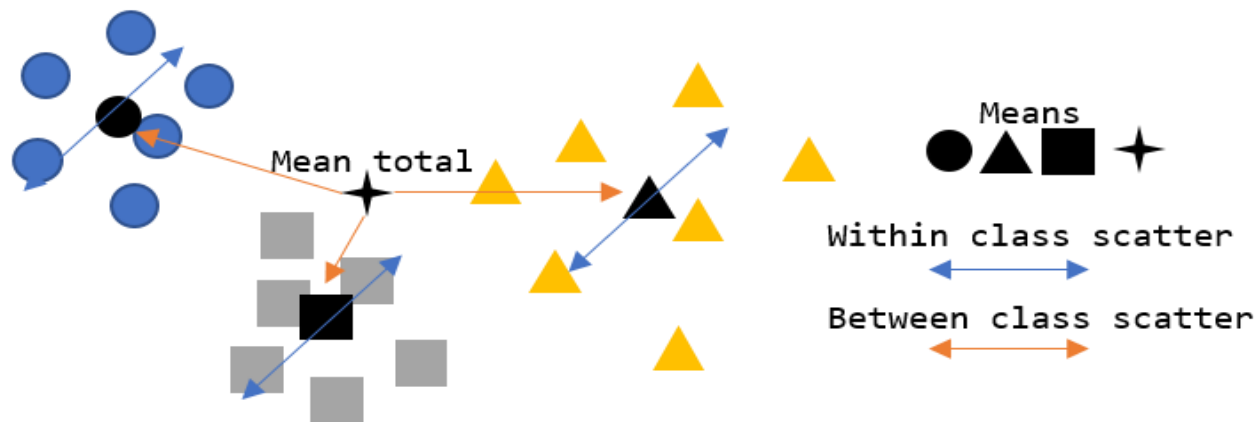
❖ ¿Cuál es la diferencia?

# Algoritmos de clustering

- ❖ k-Medias, PAM/CLARA/CLARANS
- ❖ Mezcla de gaussianas (GMM) , MeanShift
- ❖ DBSCAN, Optics, DenClue
- ❖ Clustering jerárquico
  - Ward, Diana/Agnes, BIRCH, CURE, Chameleon, ROCK

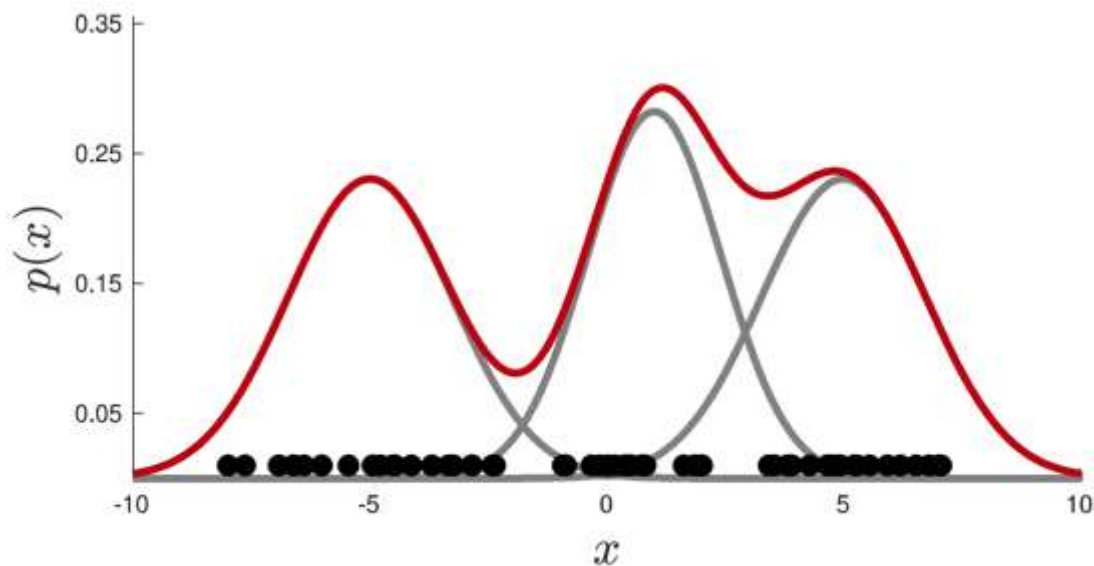
# K-means

- Particiona usando distancia a k centroides (k-medias)

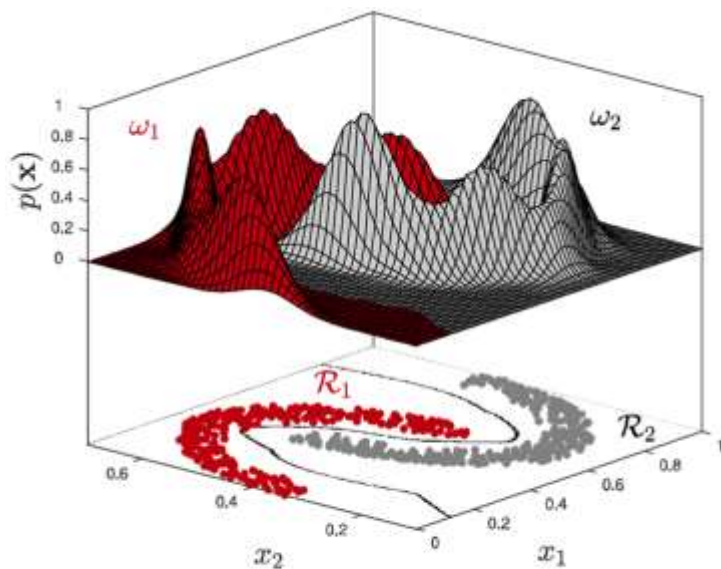


# Mezcla de Gaussianas (GMM)

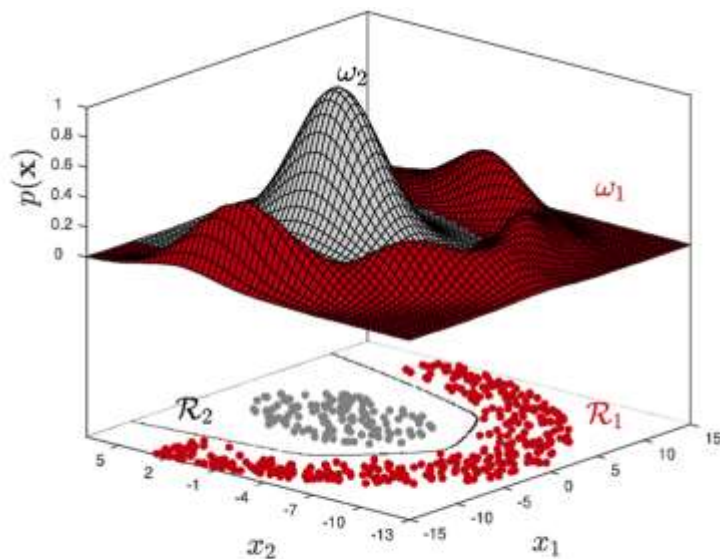
- ❖ Supongamos tener alguna información
  - Datos numéricos (reales),
  - producidos por una densidad mezcla de Gaussianas,



# Mezcla de Gaussianas



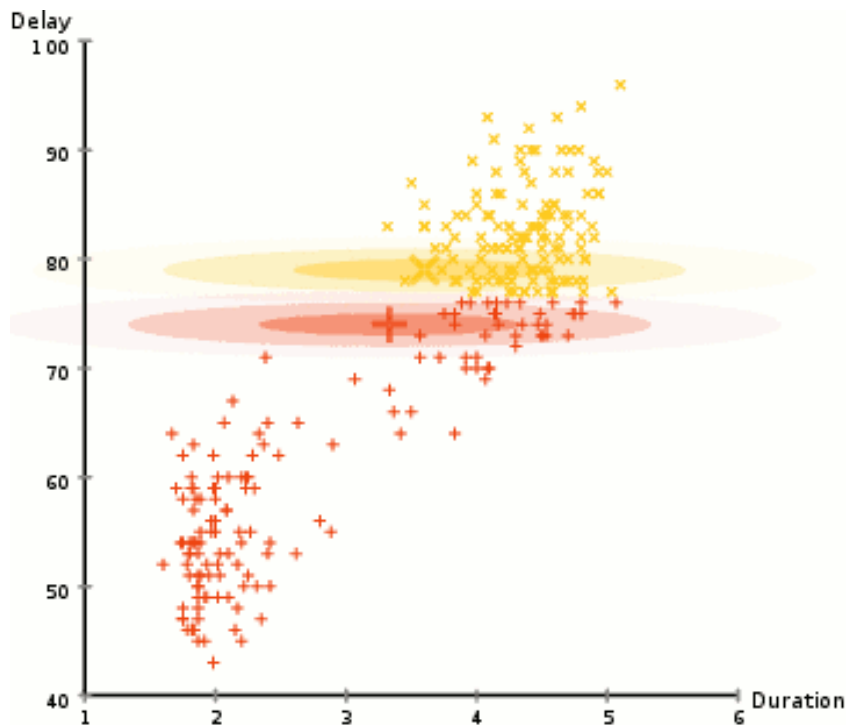
(a)



(b)

# Como funciona el GMM?

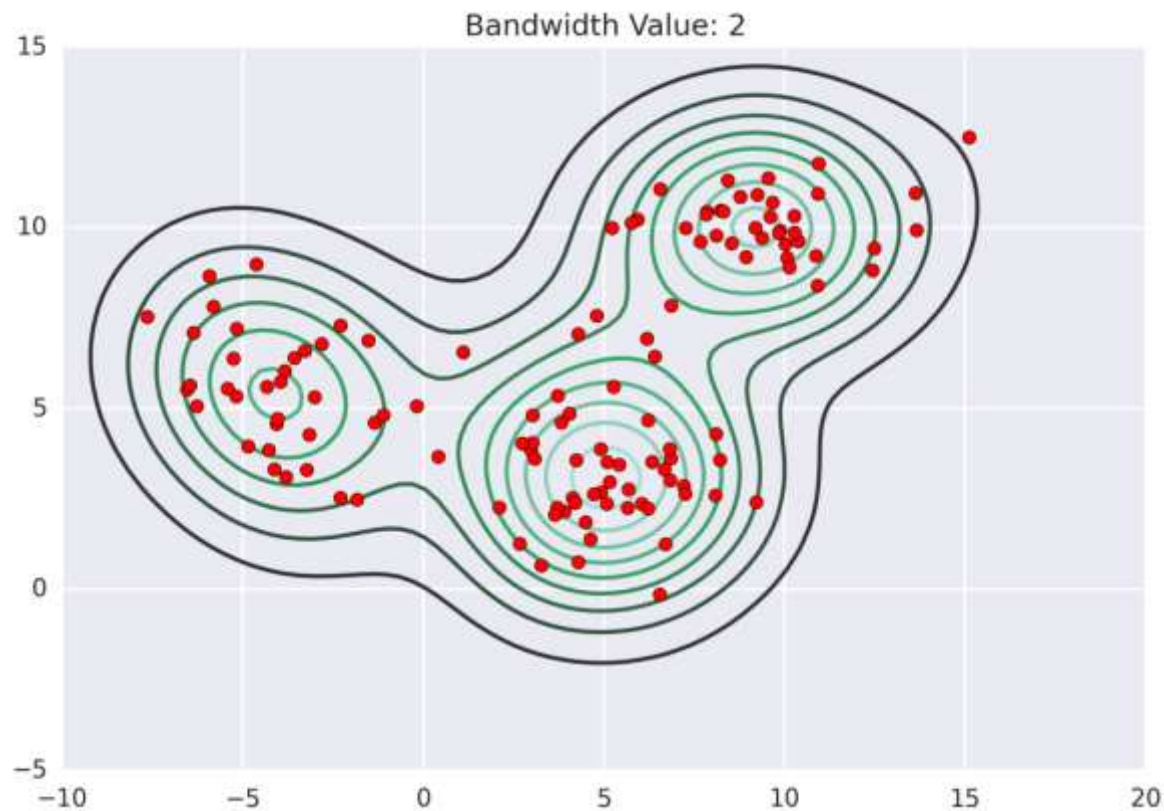
- ❖ Comenzamos con una partición aleatoria de la cual se sacan los parámetros de inicio y desde allí se itera



# Mean Shift Algorithm

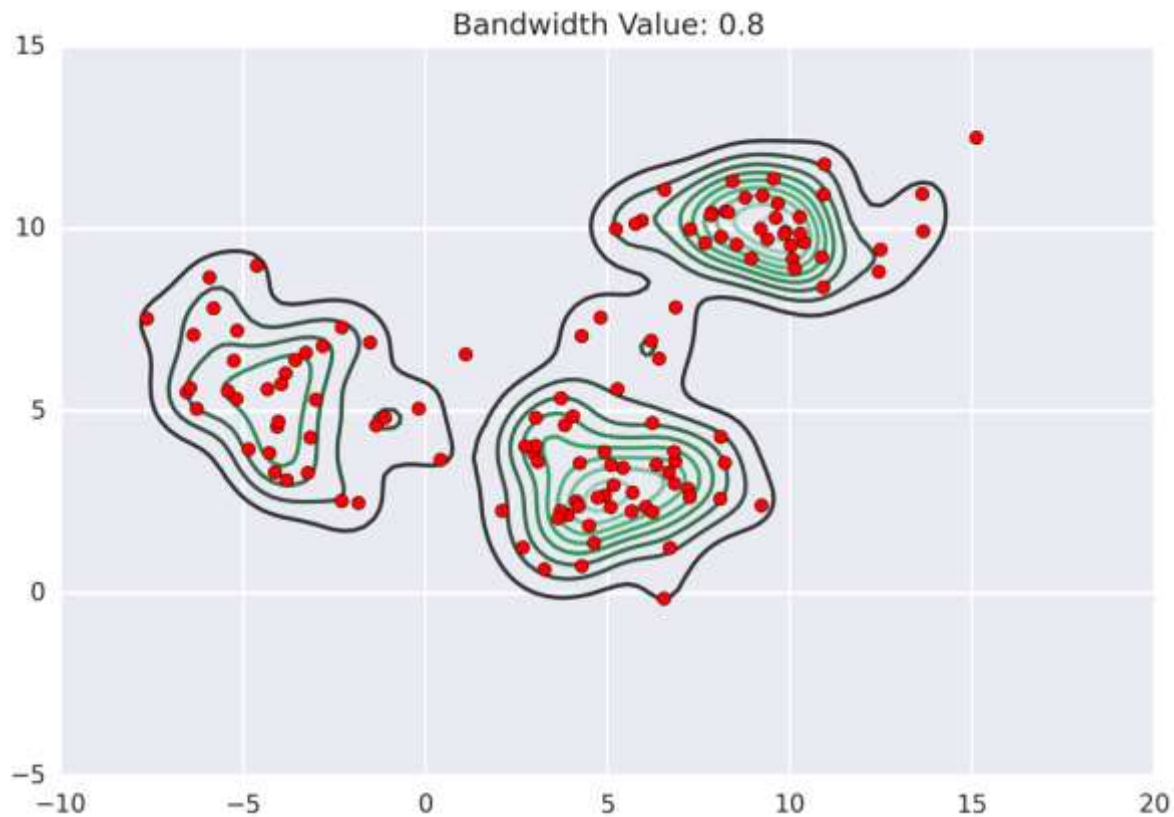
- ❖ Parámetro **bandwidth** debe ser fijado a priori, le define un entorno de influencia o cercanía a cada dato y media móvil.
  - el algoritmo se define inicializado con cada dato, considera el entorno de cercanía de este y calcula la media de los datos de ese entorno, el dato se cambia por esa media y se repite el algoritmo de forma iterativa hasta converger.
- ❖ El parámetro también se utiliza para definir un kernel de densidad
  - Intuitivamente se suman esas densidades y se encuentran los centros o regiones más densos y la cantidad de estas.

# Mean Shift Algorithm





# Mean Shift Algorithm



# Mean Shift Algorithm

- ❖ Parámetro bandwidth fijado a priori.
- ❖ Puede ser estimado utilizando la teoría no paramétrica, dependiendo de que kernel se use.
- ❖ Note\_fig4.ipynb tiene un ejemplo de Mean Shift automático
- ❖ (No tiene sentido usar medidas de bondad de ajuste BIC o AIC pues no se está fijando un modelo paramétrico)

# Notebook

Demo CIED04\_clustering\_2\_fifa2019.ipynb

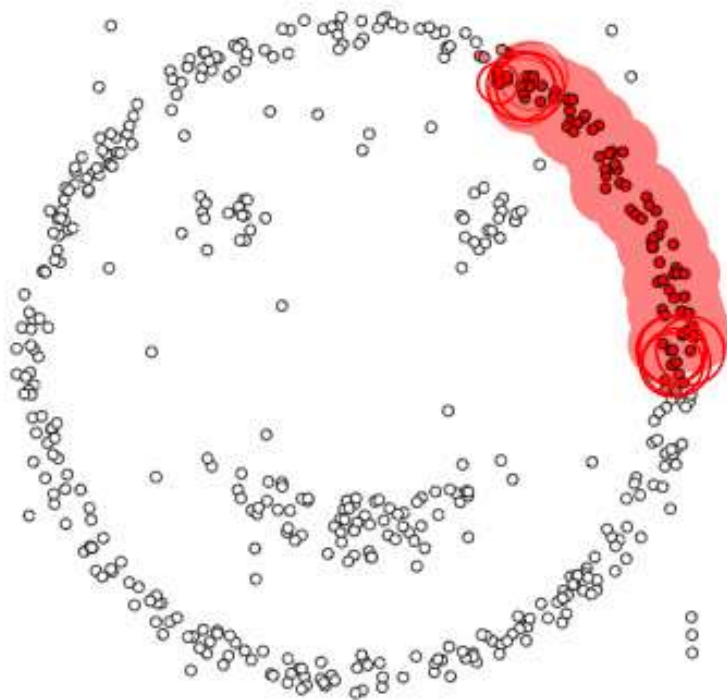
# Dbscan

- ❖ Hay dos parámetros en este algoritmo: **min\_samples** y **eps**, donde min\_samples son los datos mínimos requeridos en un entorno de radio eps, los cuales definen la noción de zona densa.
- ❖ Un cluster es un conjunto de puntos de núcleo cercanos unos con otros, junto a un conjunto de puntos no núcleo que están cercanos a algún punto de núcleo.

# Dbscan

- ❖ El algoritmo empieza con una muestra aleatoria y encuentra todos los puntos en el entorno de radio  $\epsilon$ .
- ❖ Si el número de puntos es mayor a  $\text{min\_number}$  se etiqueta ese punto como un punto de núcleo, si no es un punto outlier.

# DbSCAN



Restart

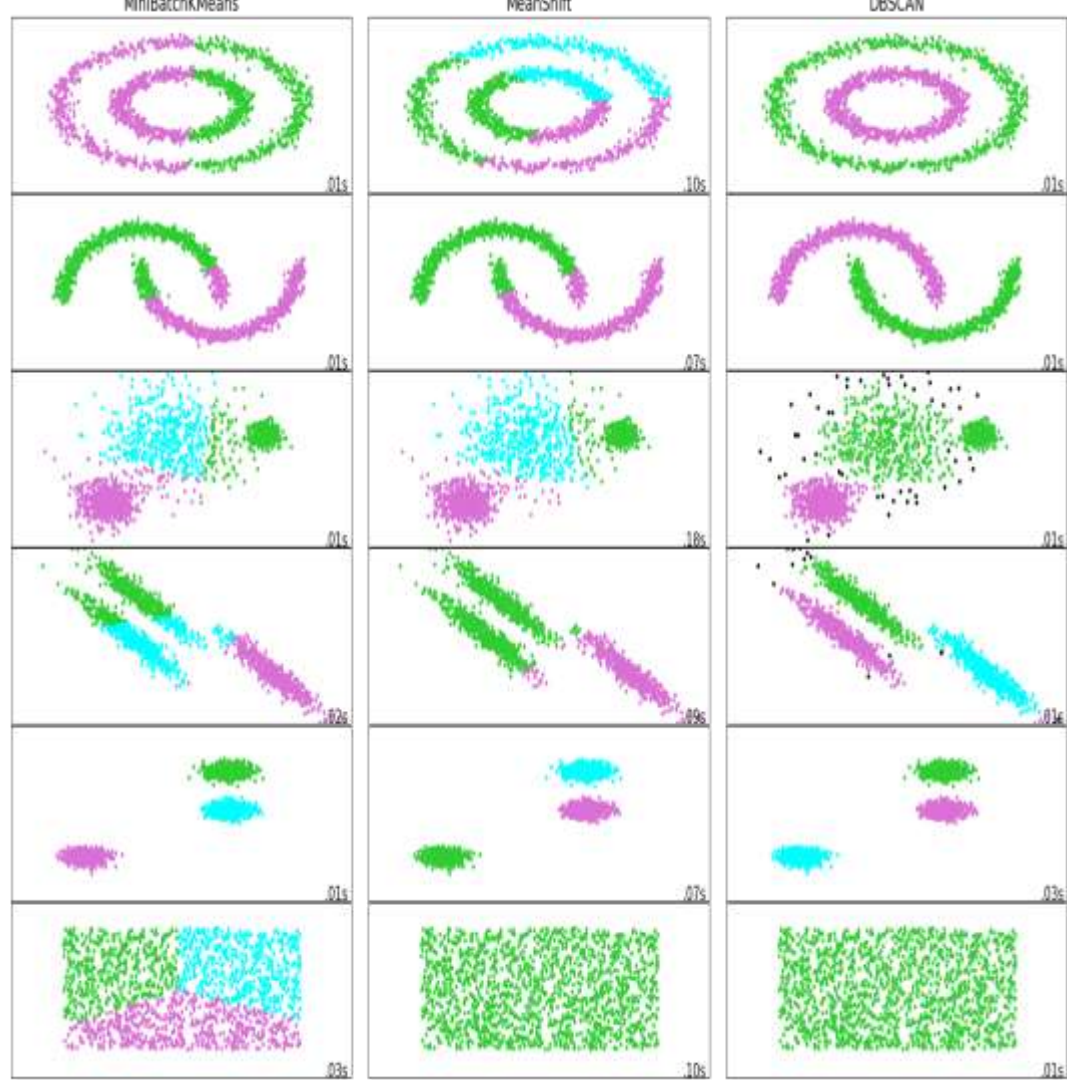


Pause

# Dbscan

- ❖ Todos los puntos del entorno se etiquetan como puntos no núcleo de cluster. Se realiza el mismo procedimiento para cada uno de ellos, cambiando a punto con su etiqueta y agregando nuevos puntos, o marcando outliers.
- ❖ Si no hay más puntos en un entorno  $\epsilon$  de cada punto del cluster, se salta a otro punto aleatoriamente y se continúa hasta que todo punto es bien un punto de cluster o un outlier.

**note\_fig8.ipynb**





# Clustering jerárquico

Si no queremos especificar k...

Algoritmos jerárquicos que generan una

taxonomía jerárquica de clusters (dendrograma)

- Interpretación más rica
- Más difícil de interpretar
- El corte del árbol tiene que ser ortogonal

