

Aprendizaje No Supervisado

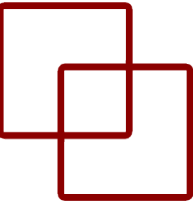


Dr. Manuel Castillo-Cara

www.manuelcastillo.eu

**Departamento de Inteligencia Artificial
Escuela Técnica Superior de Ingeniería Informática
Universidad Nacional de Educación a Distancia (UNED)**

Índice

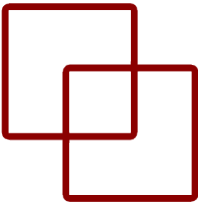


- Aprendizaje No Supervisado.
- Algoritmos ULAS.
- Algoritmo k-Means.
- Clústering jerárquico.
- Métodos basados en densidad.

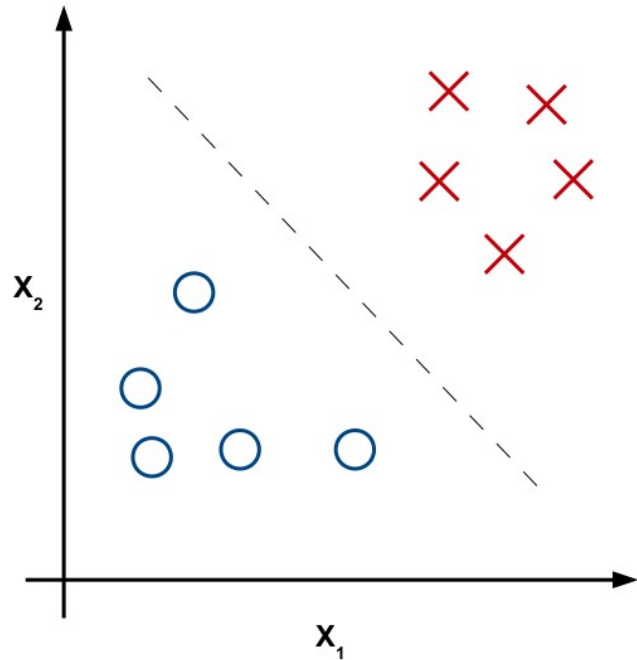


Aprendizaje No Supervisado

1. SLAs Vs. ULAs



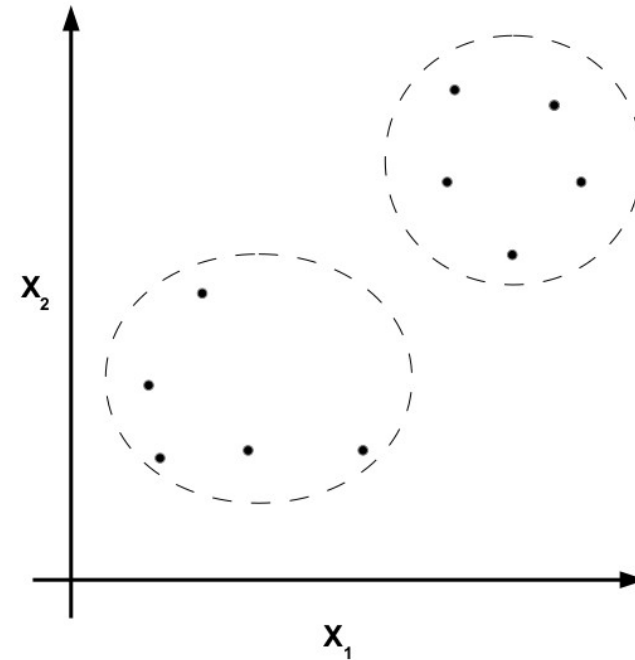
Aprendizaje supervisado



Training

$$\{ (\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \dots, (\mathbf{x}^{(m)}, y^{(m)}) \}$$

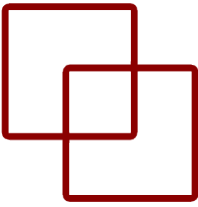
Aprendizaje no supervisado



Training

$$\{ (\mathbf{x}^{(1)}), (\mathbf{x}^{(2)}), \dots, (\mathbf{x}^{(m)}) \}$$

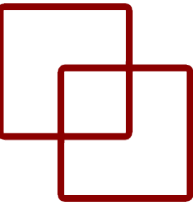
Tenemos que descubrir y



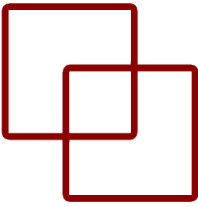
2. Clustering (I)

- El *clustering* (agrupamiento) consiste en dividir un conjunto de elementos heterogéneos en **clústers** o grupos homogéneos.
- Se considera un paradigma de **clasificación no supervisada**, ya que asigna una clase a cada elemento (clúster al que pertenece), pero dichas clases no son conocidas durante el proceso de aprendizaje de el modelo.
 - Tenemos que descubrir y

2. Clustering (II)

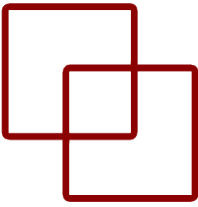


- El objetivo del agrupamiento es encontrar una división de los datos en la que se dé:
 - **Alta similaridad intra-cluster** (entre los elementos de un mismo clúster).
 - **Baja similaridad inter-cluster** (entre los elementos de distinto clústers).



2. Clustering (II)

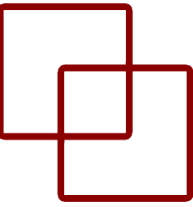
- El objetivo del agrupamiento es encontrar una división de los datos en la que se dé:
 - **Alta similaridad intra-cluster** (entre los elementos de un mismo clúster).
 - **Baja similaridad inter-cluster** (entre los elementos de distinto clústers).
- La similaridad se suele (veremos también que puede considerarse la conectividad) basar en una medida de **distancia**. Esta medida depende de la representación de los datos (discretos, numéricos, booleanos, etc).



2. Clustering (II)

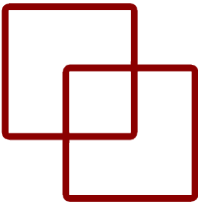
- El objetivo del agrupamiento es encontrar una división de los datos en la que se dé:
 - **Alta similaridad intra-cluster** (entre los elementos de un mismo clúster).
 - **Baja similaridad inter-cluster** (entre los elementos de distinto clústers).
- La similaridad se suele (veremos también que puede considerarse la conectividad) basar en una medida de **distancia**. Esta medida depende de la representación de los datos (discretos, numéricos, booleanos, etc).
- Además de la similaridad, se suelen usar medidas para medir la **calidad del clúster**, y que contemplan tanto la similaridad/similaridad intra/inter clúster, como el número de clústers.

3. Aplicaciones clustering



- **Biología:** Detección y agrupación de secuencias similares de genes.
- **Marketing:** descubrimiento de distintos grupos de clientes para personalización de ofertas, etc.
- **Análisis de redes sociales:** Detección de comunidades.
- **Búsqueda de información:** Agrupación de documentos o noticias similares.
- etc.

4. Métodos de clustering



- **Basados en centroides**

- Construyen distintas particiones y las evalúan en función de algún criterio (generalmente basados en distancia).
- k -means, k -medoids, etc.

- **Jerárquicos**

- Crean una descomposición jerárquica del conjunto de datos (objetos) usando algún criterio.
- Diana, Agnes, BIRCH, ROCK, etc.

- **Métodos basados en densidad**

- Se basan en conectividad y en funciones de densidad de puntos en el espacio.
- DBSCAN, OPTICS, DenClue, etc.

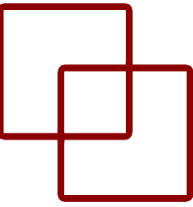
- **Basados en modelos**

- Se propone un modelo hipótesis para cada uno de los clústers y se trata de encontrar el mejor ajuste de ese modelo a otros.
- EM, COBWEB

- Otros métodos.



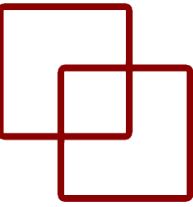
Algoritmos ULAS



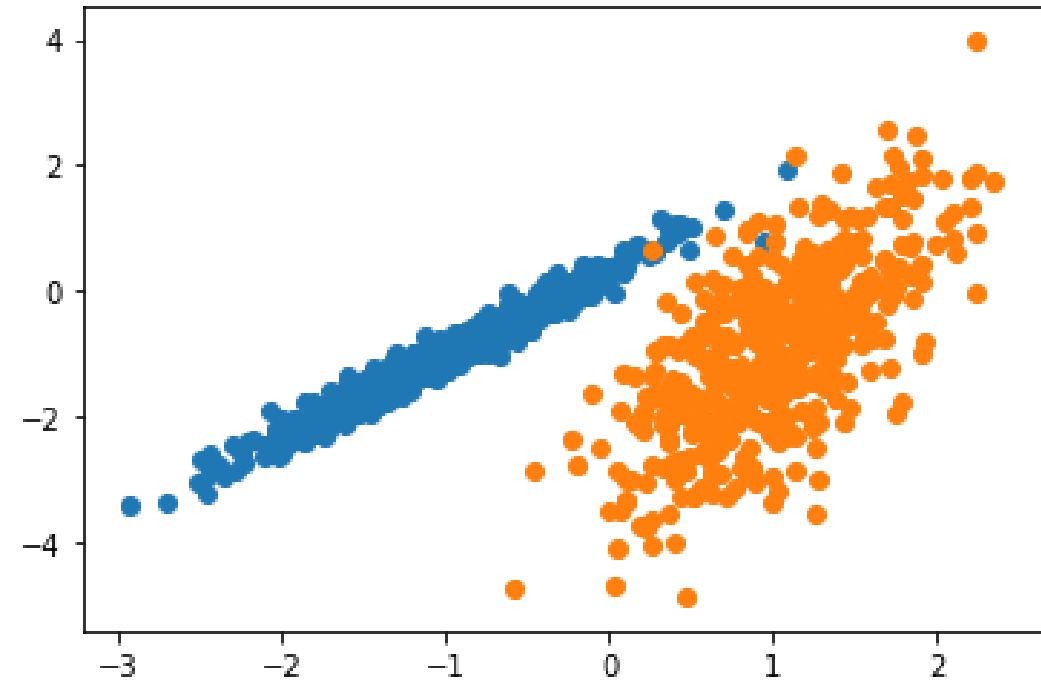
1. Algoritmos (I)

- Existen muchos algoritmos de clustering para elegir.
- No existe el mejor algoritmo de clustering para todos los casos.
- En cambio, es una buena idea explorar una variedad de algoritmos de clustering y diferentes configuraciones para cada algoritmo.
- Trata de encontrar grupos naturales en el espacio de características de los datos de entrada.
- Veamos una descripción sencilla de algunos de estos algoritmos según su naturaleza.

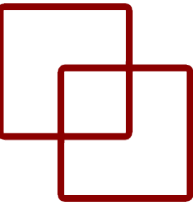
1. Algoritmos (II)



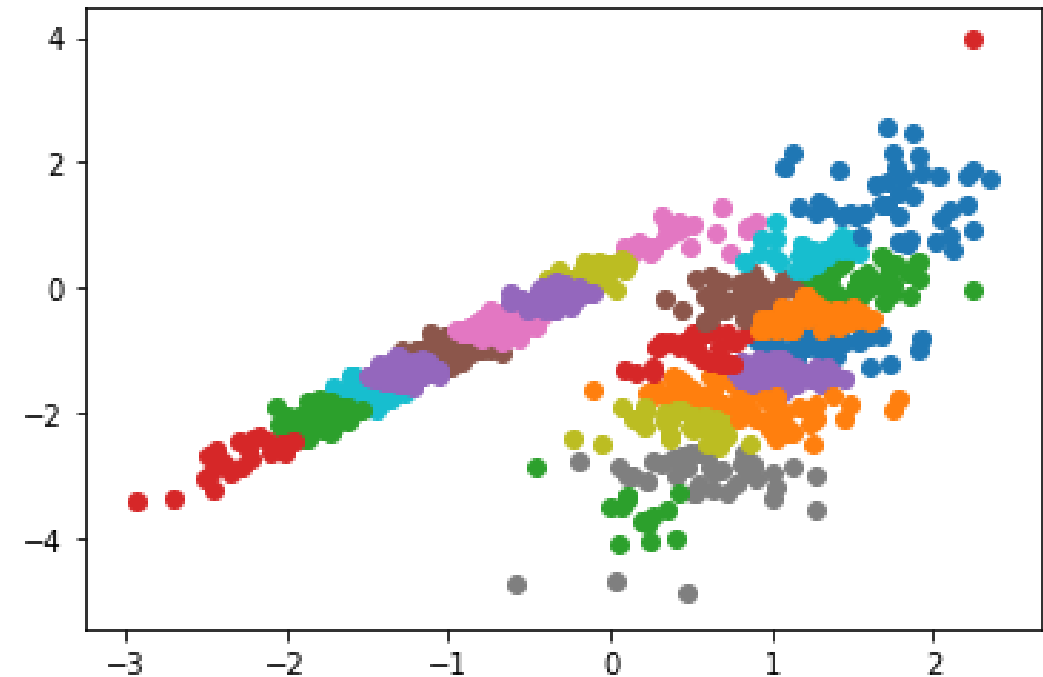
- El conjunto de datos tendrá 1,000 ejemplos, con dos características de entrada y un clúster por clase.
- Los grupos son visualmente obvios en dos dimensiones para que podamos trazar los datos con un diagrama de dispersión y colorear los puntos en el diagrama por el grupo asignado.
 - Esto ayudará a ver, al menos en el problema de la prueba, qué tan bien se identificaron los grupos.
- Los grupos en este problema de prueba se basan en un gaussiano multivariado, y no todos los algoritmos de agrupamiento serán efectivos para identificar estos tipos de grupos.
- Como tal, los resultados en este tutorial no deben usarse como base para comparar los métodos en general.



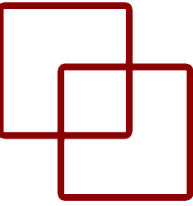
2. Affinity Propagation



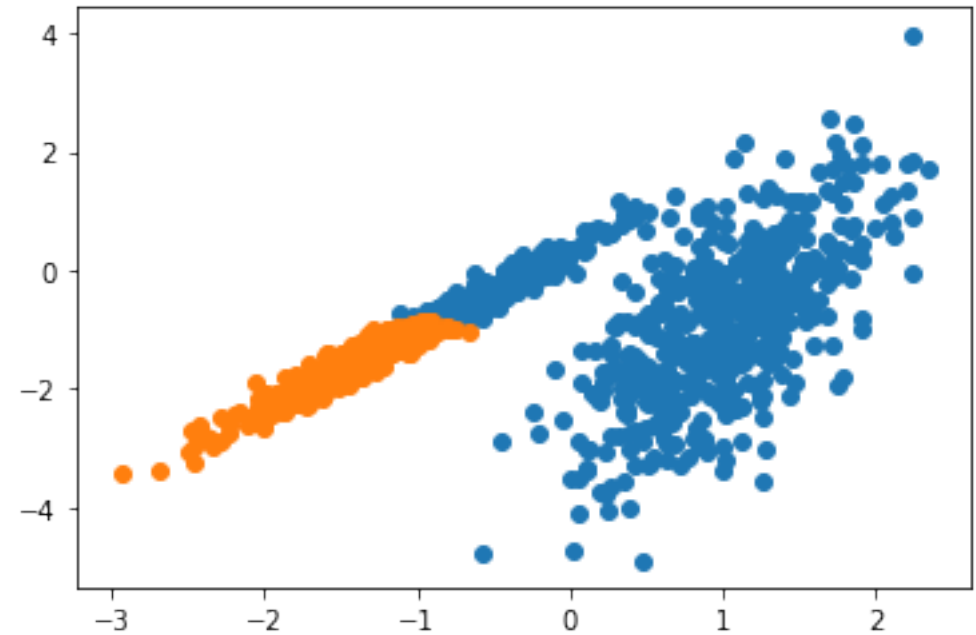
- Affinity Propagation implica encontrar un conjunto de ejemplos que mejor resuman los datos.
- Toma como entrada medidas de similitud entre pares de puntos de datos.
- Los mensajes de valor real se intercambian entre puntos de datos hasta que emerge gradualmente un conjunto de ejemplos de alta calidad y grupos correspondientes.
- Podemos observar que este algoritmo no puede lograr un buen resultado.



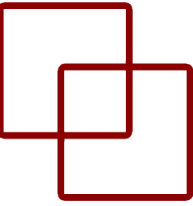
3. Agglomerative



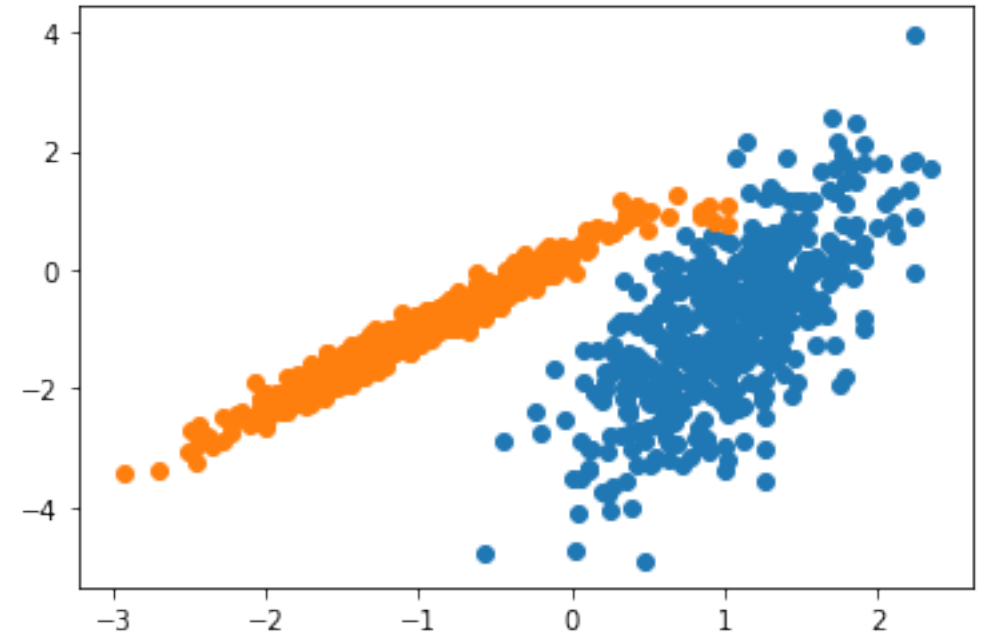
- Agglomerative implica fusionar ejemplos hasta que se alcanza el número deseado de agrupaciones.
- La agrupación jerárquica es un método de análisis de agrupación que busca construir una jerarquía de agrupaciones.
- Son muy lentos y requieren una capacidad computacional alta,
 - Tiene una complejidad de $O(n)^3$, requiriendo $O(n)^2$ de memoria.
- Podemos observar una agrupación razonable.



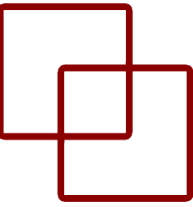
4. BIRCH



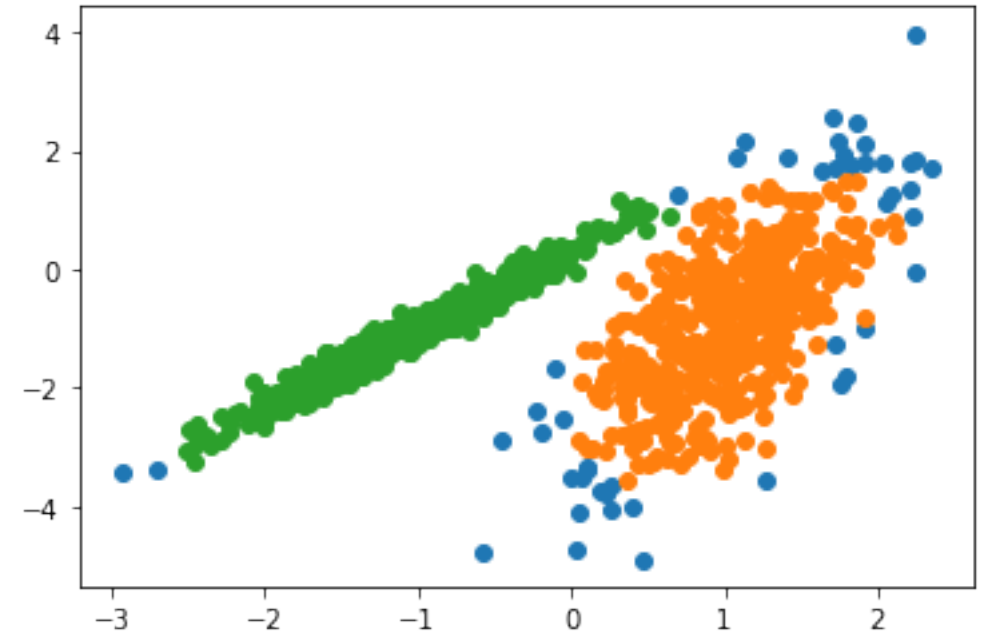
- Es un algoritmo de minería de datos sin supervisión que se utiliza para realizar agrupamientos jerárquicos en conjuntos de datos particularmente grandes.
- Una ventaja de BIRCH es su capacidad de agrupar de forma incremental y dinámica los puntos de datos métricos multidimensionales entrantes en un intento de producir la agrupación de la mejor calidad para un conjunto dado de recursos (limitaciones de memoria y tiempo).
- En la mayoría de los casos, BIRCH solo requiere un solo escaneo de la base de datos.
- Podemos observar una excelente agrupación.



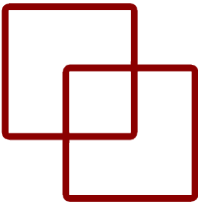
5. DBSCAN



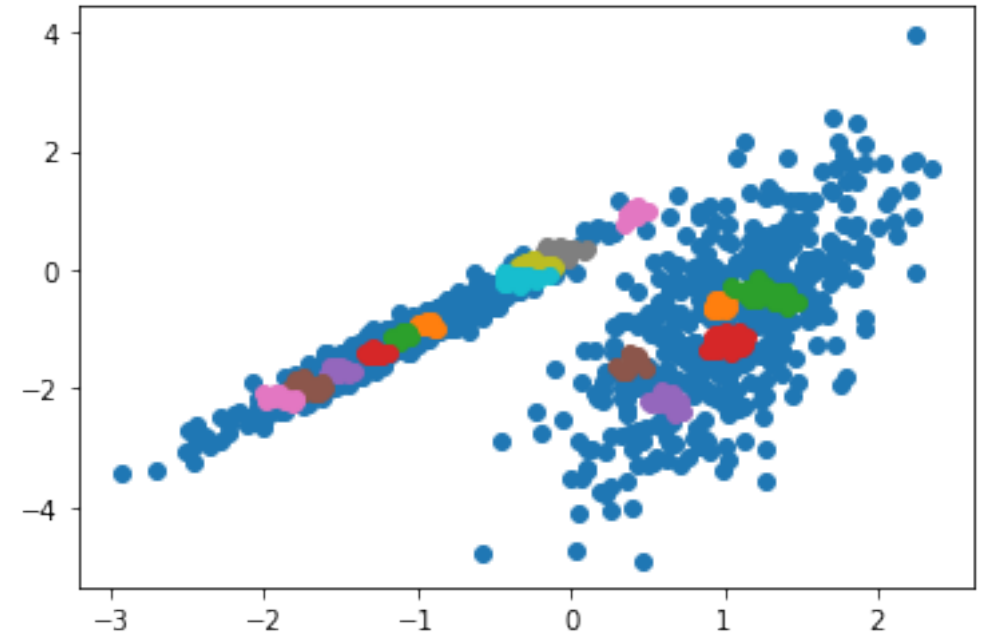
- Es un algoritmo no paramétrico de agrupamiento basado en la densidad:
 - Dado un conjunto de puntos en algún espacio, agrupa los puntos que están muy juntos, marcando como puntos atípicos que se encuentran solos en regiones de baja densidad.
- Es uno de los algoritmos de agrupamiento más comunes y también más citado en la literatura científica.
- Implica encontrar áreas de alta densidad en el dominio y expandir esas áreas del espacio de características a su alrededor como clústeres.
- Podemos observar una agrupación razonable, aunque se requiere más ajuste.



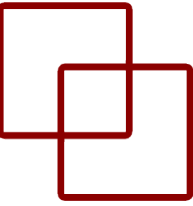
6. OPTICS



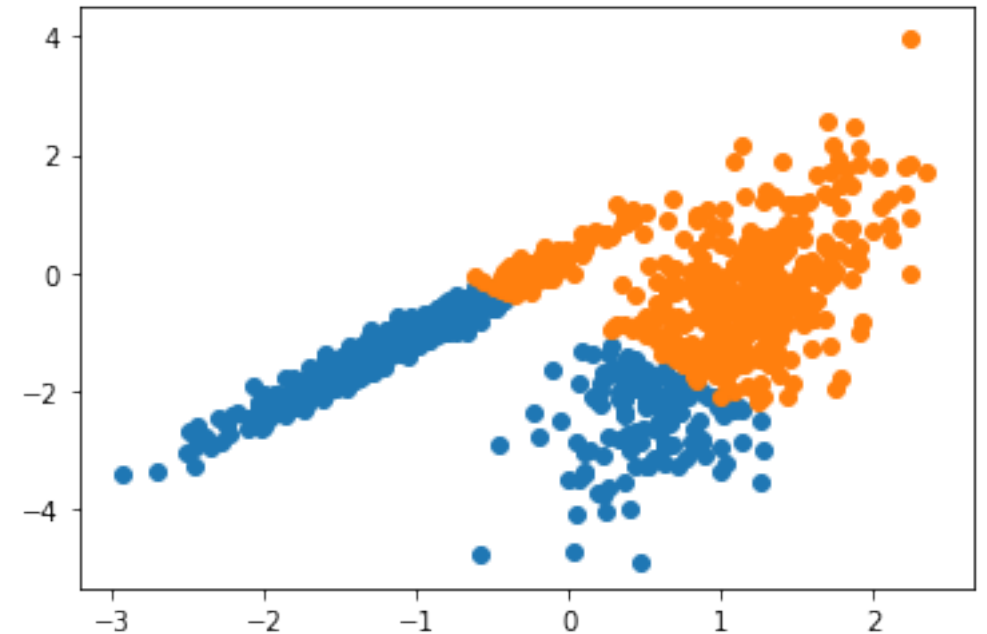
- Es una versión modificada de DBSCAN, pero aborda una de las principales debilidades de DBSCAN:
 - El problema de detectar grupos significativos en datos de densidad variable.
 - Para hacerlo, los puntos de la base de datos se ordenan (linealmente) de modo que los puntos espacialmente más cercanos se conviertan en vecinos en el orden.
 - Además, se almacena una distancia especial para cada punto que representa la densidad que se debe aceptar para un grupo para que ambos puntos pertenezcan al mismo grupo.
 - Esto se representa como un dendrograma.
- Podemos observar que no pude lograr un resultado razonable en este conjunto de datos.



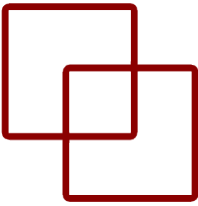
7. *k*-Means



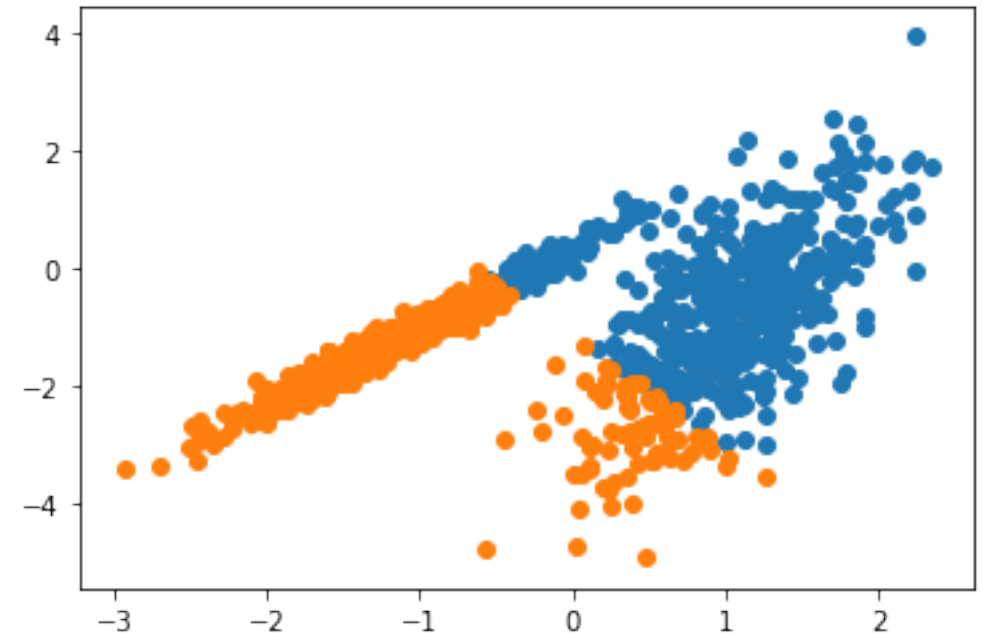
- Puede ser el algoritmo de agrupación más conocido e implica asignar ejemplos a los grupos en un esfuerzo por minimizar la variación dentro de cada grupo.
- El objetivo principal es describir un proceso para dividir una población N-dimensional en k conjuntos sobre la base de una muestra.
- Podemos observar una agrupación razonable, aunque la varianza desigual en cada dimensión hace que el método sea menos adecuado para este conjunto de datos.



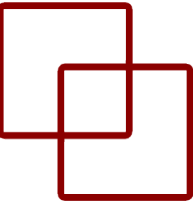
8. Mini-Batch k -Means



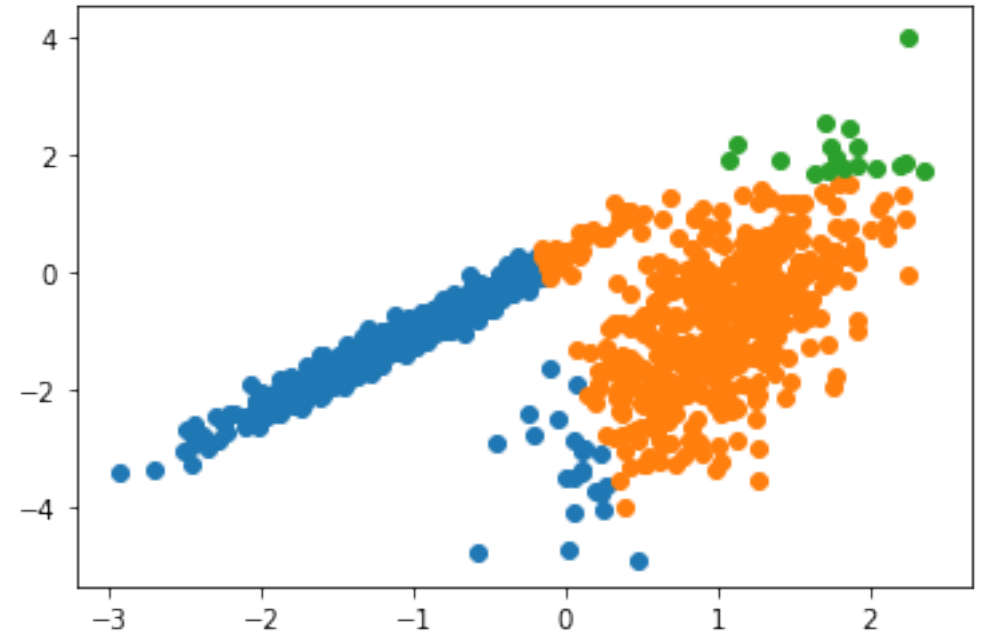
- Es una versión modificada de k -Means que realiza actualizaciones a los centroides del clúster utilizando mini-lotes de muestras en lugar de todo el conjunto de datos
 - Lo hace más rápido para grandes conjuntos de datos,
 - Quizás más robusto al ruido estadístico; y
 - Proporciona soluciones significativamente mejores que el descenso de gradiente estocástico en línea.
- Podemos observar un resultado equivalente al algoritmo estándar k -Means.



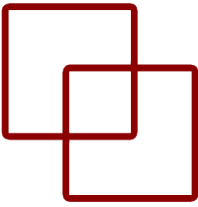
9. Mean Shift



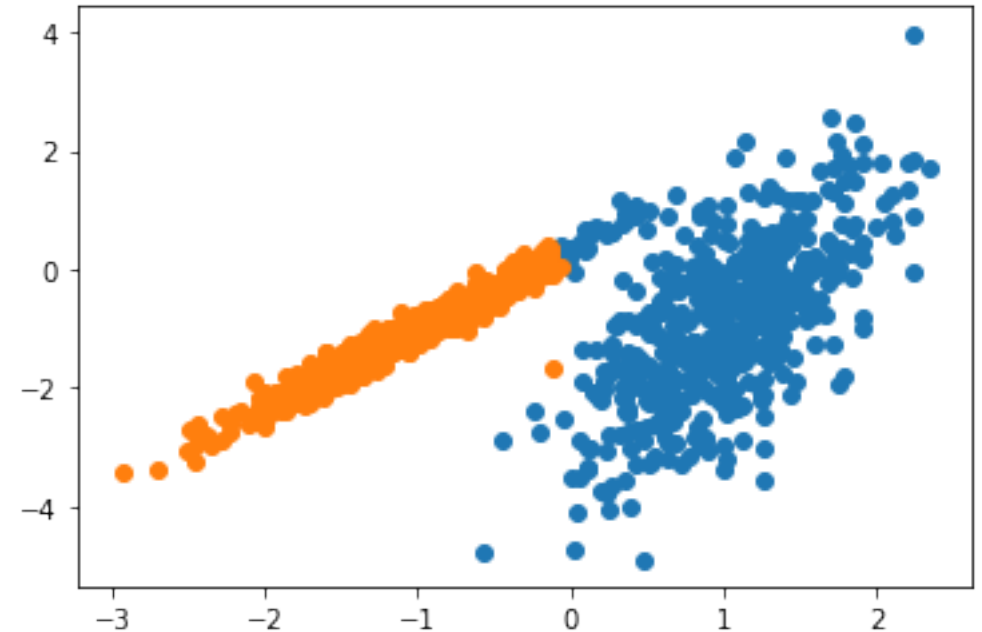
- Es una técnica de análisis de espacio de características no paramétrica para localizar los máximos de una función de densidad.
- Los dominios de aplicación incluyen análisis de conglomerados en visión por computadora y procesamiento de imágenes.
- La agrupación Mean Shift implica encontrar y adaptar los centroides en función de la densidad de ejemplos en el espacio de características.
- Podemos observar un conjunto razonable de clústeres en los datos.



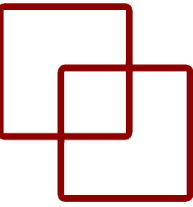
10. Spectral Clustering



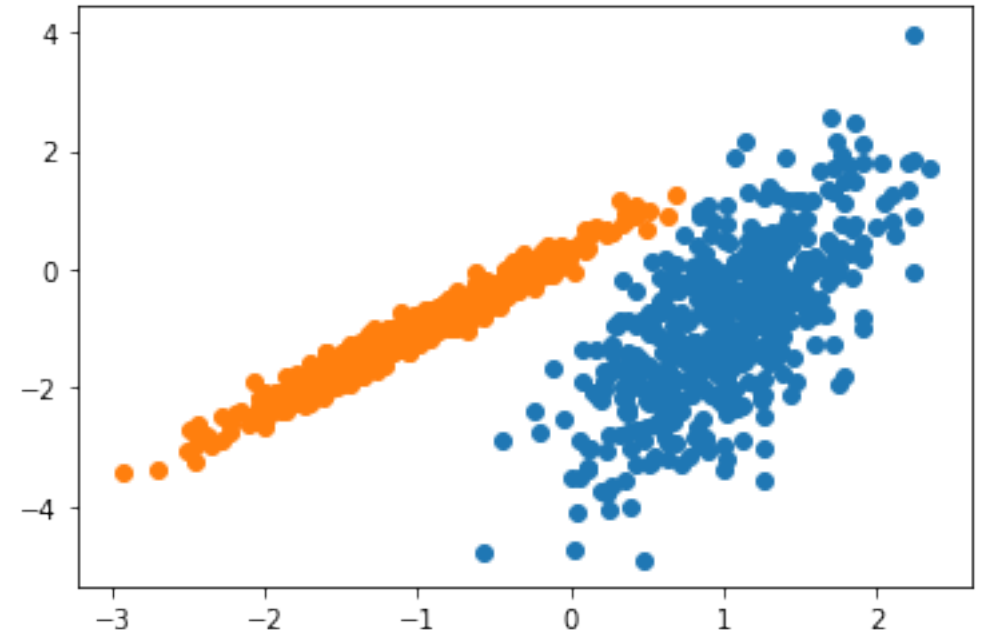
- Spectral Clustering es una clase general de métodos de agrupación, extraída del álgebra lineal.
- Usa los vectores propios superiores de una matriz derivada de la distancia entre puntos.
- Podemos observar quun resultado razonable en este conjunto de datos.



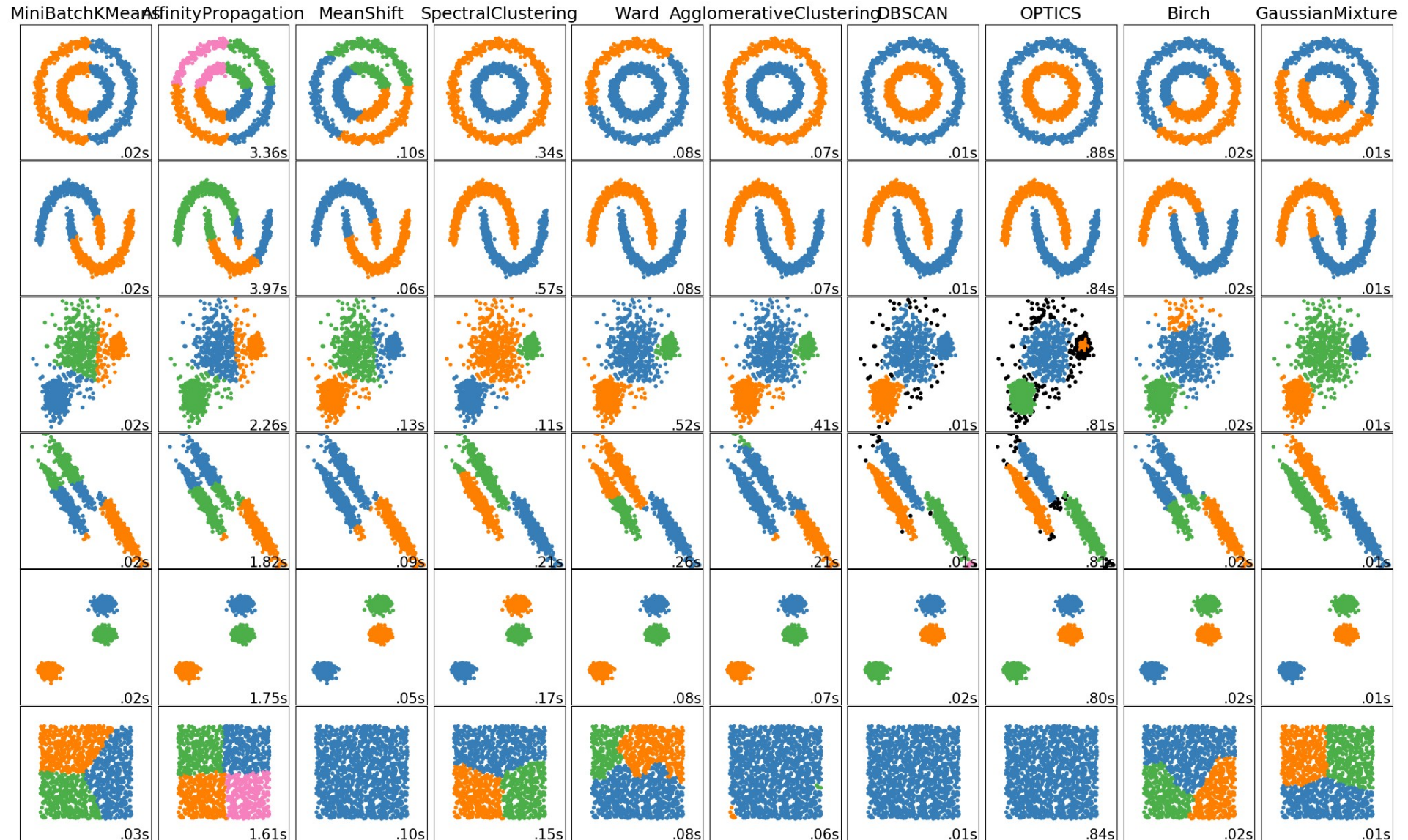
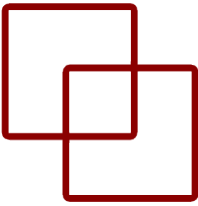
11. Gaussian Mixture Model



- Resume una función de densidad de probabilidad multivariada con una mezcla de distribuciones de probabilidad gaussianas como su nombre indica.
- Podemos ver que los grupos se identificaron perfectamente.
 - No es sorprendente dado que el conjunto de datos se generó como una mezcla de gaussianos.



12. Algoritmos (II)

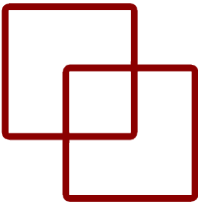
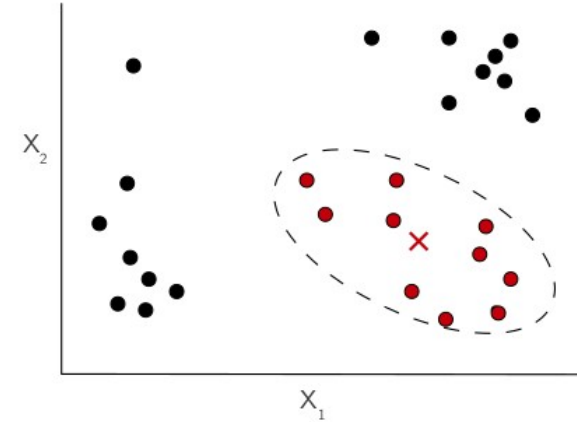


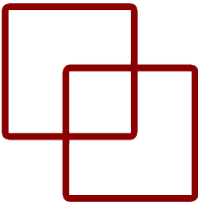


Algoritmo *k*-means

1. Definición

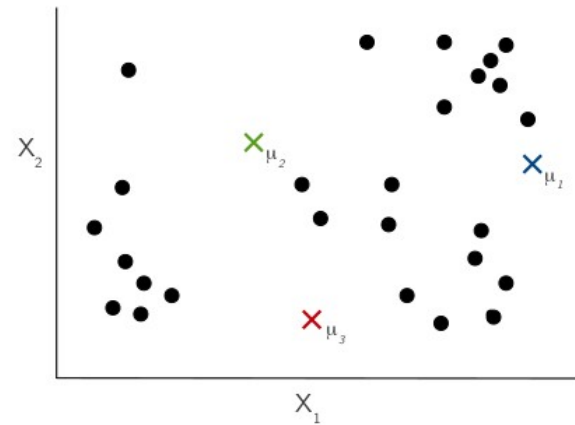
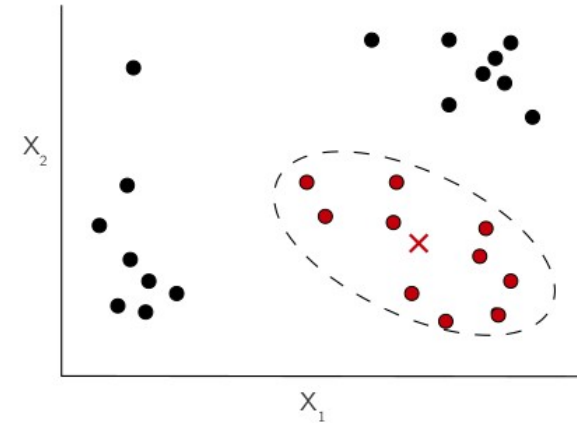
- El centroide de un cluster se obtiene como el punto medio de sus elementos.



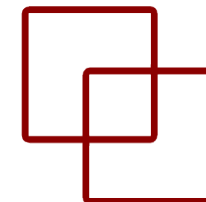


1. Definición

- El centroide de un cluster se obtiene como el punto medio de sus elementos.

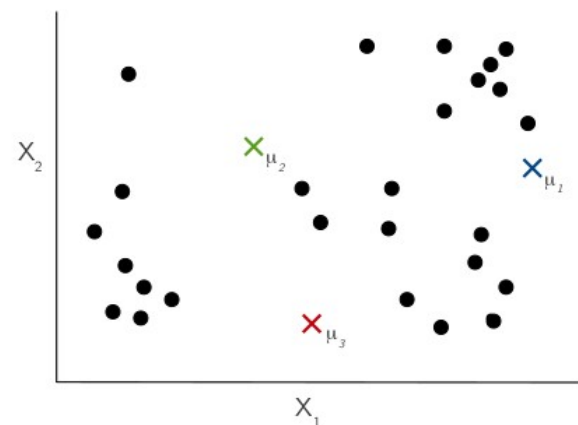
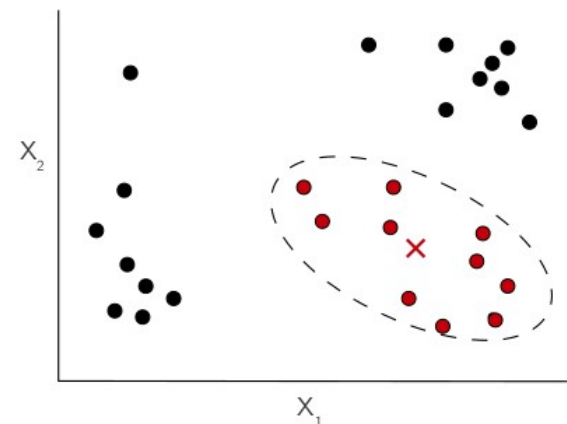


- El algoritmo k -means parte de k puntos generados aleatoriamente.



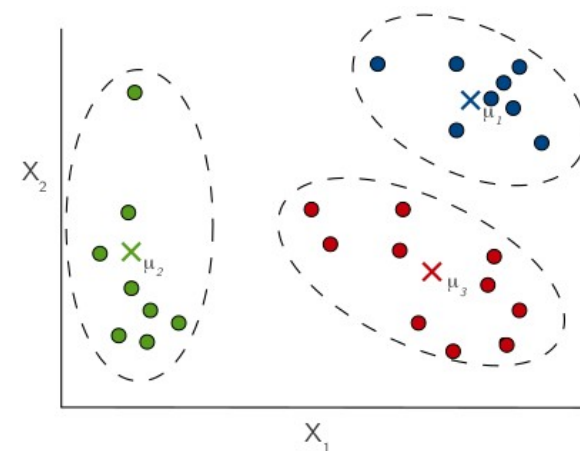
1. Definición

- El centroide de un cluster se obtiene como el punto medio de sus elementos.

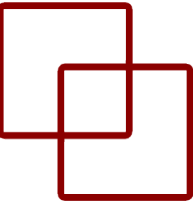


- El algoritmo k -means parte de k puntos generados aleatoriamente.

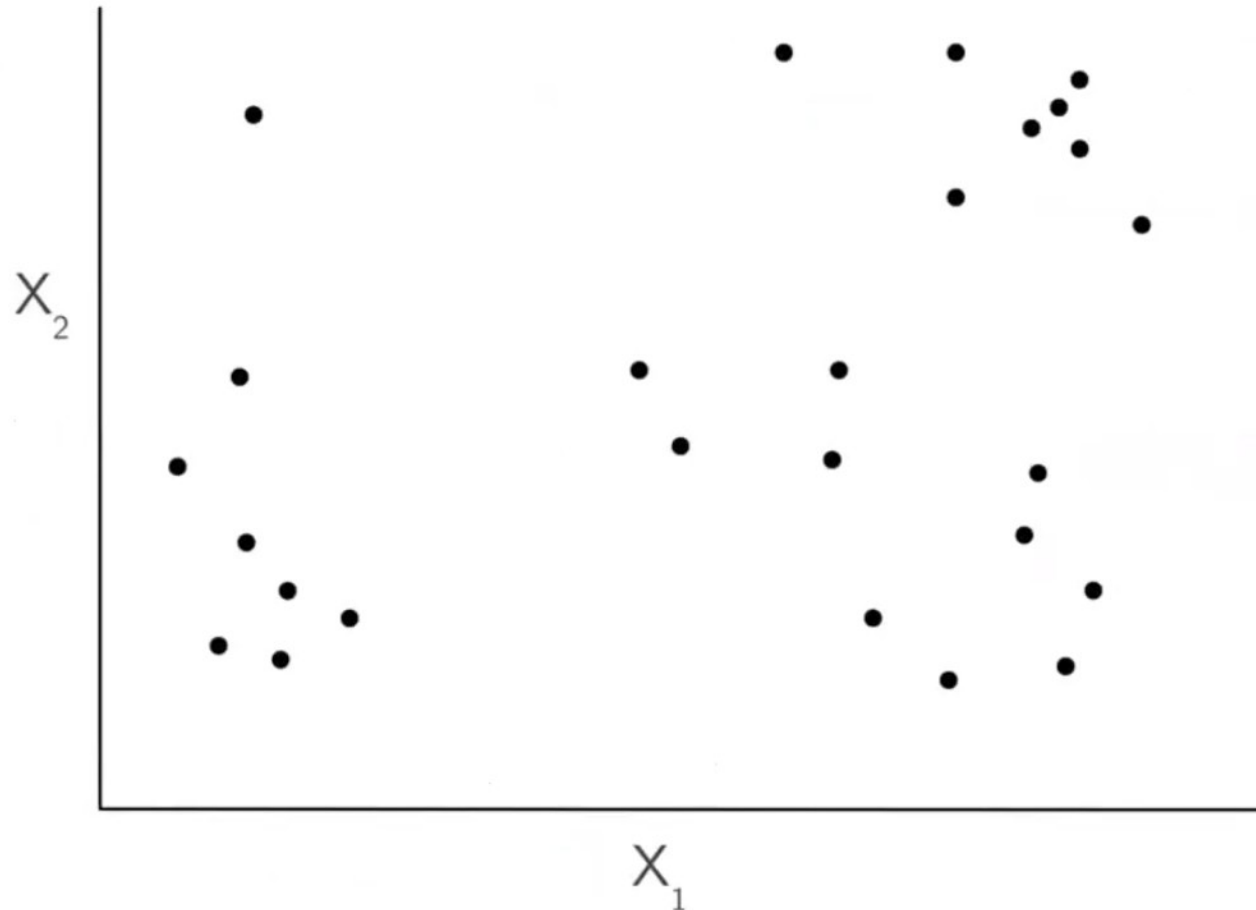
- Devuelve las coordenadas de los centroides de cada uno de los k clústers (μ_k), y el clúster al que pertenece cada ejemplo $x^{(i)}$ ($c^{(i)}$).



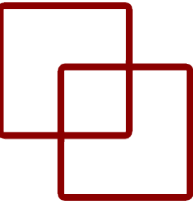
2. Ejemplo ($k=3$)



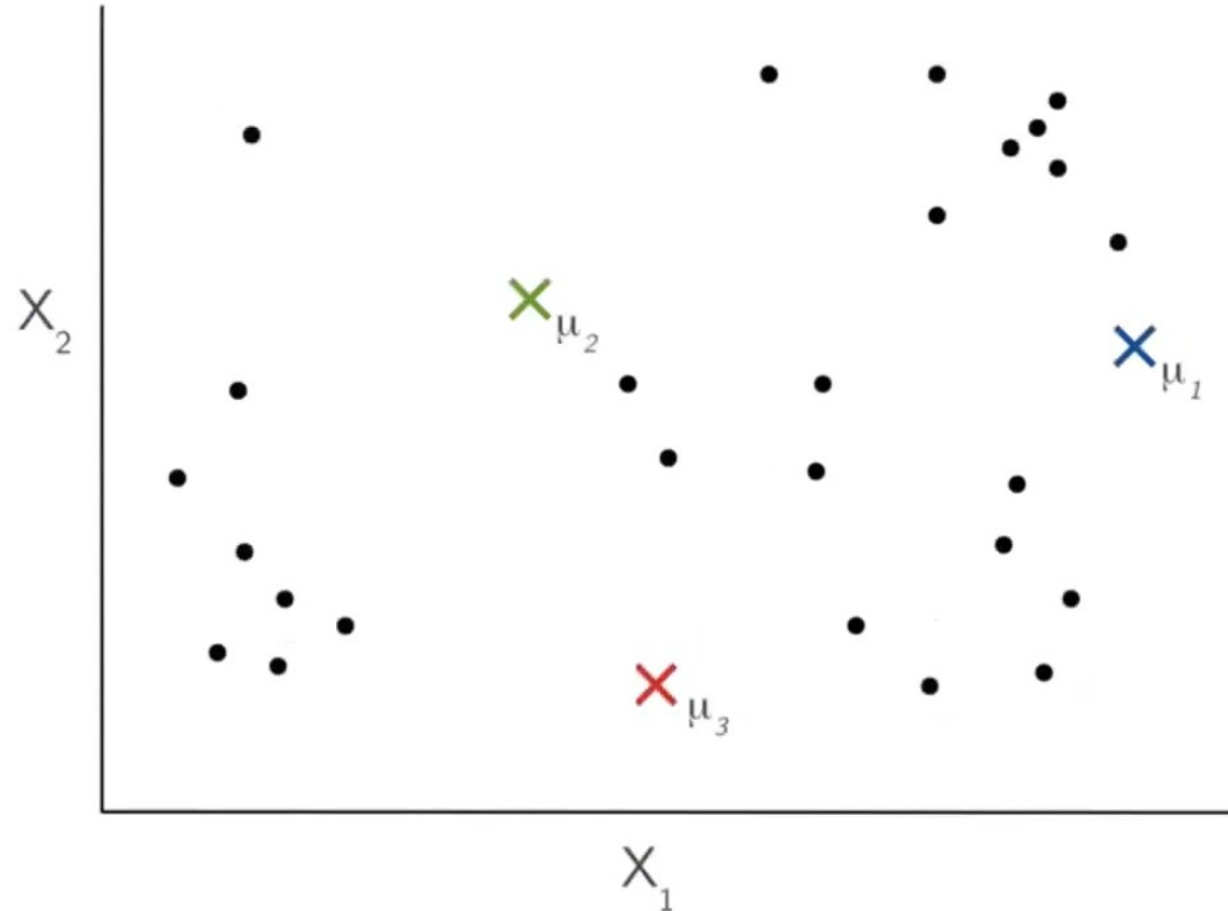
- Tenemos la representación de puntos

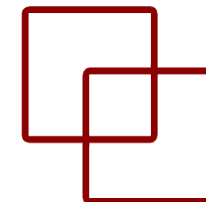


2. Ejemplo ($k=3$)



- Calculamos los k centroides aleatoriamente



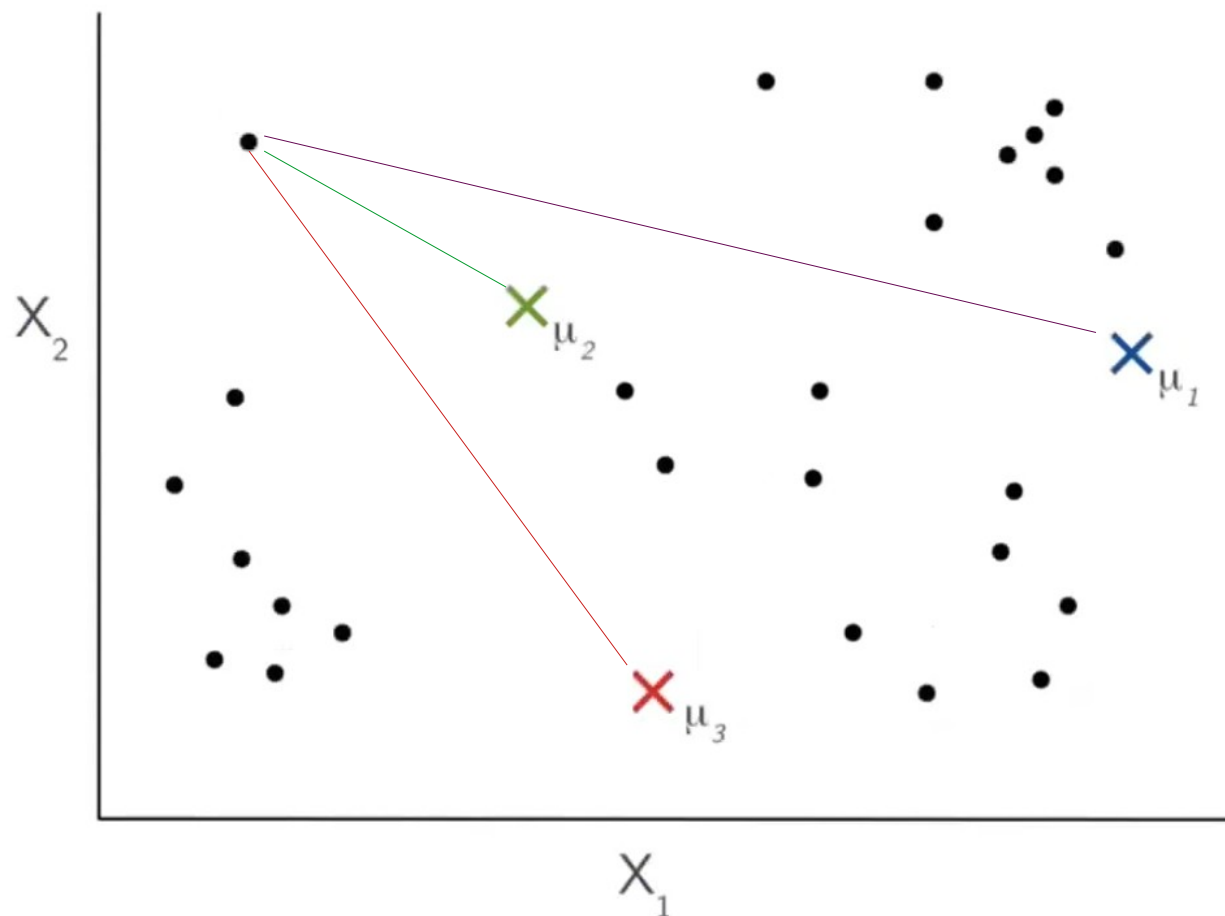


2. Ejemplo ($k=3$)

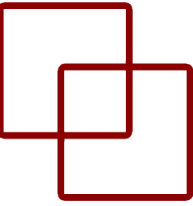
- Calculamos los k centroides aleatoriamente

Una vez calculados los centroides calcula la distancia de cada punto a ellos para agregarlos al que tenga menor distancia.

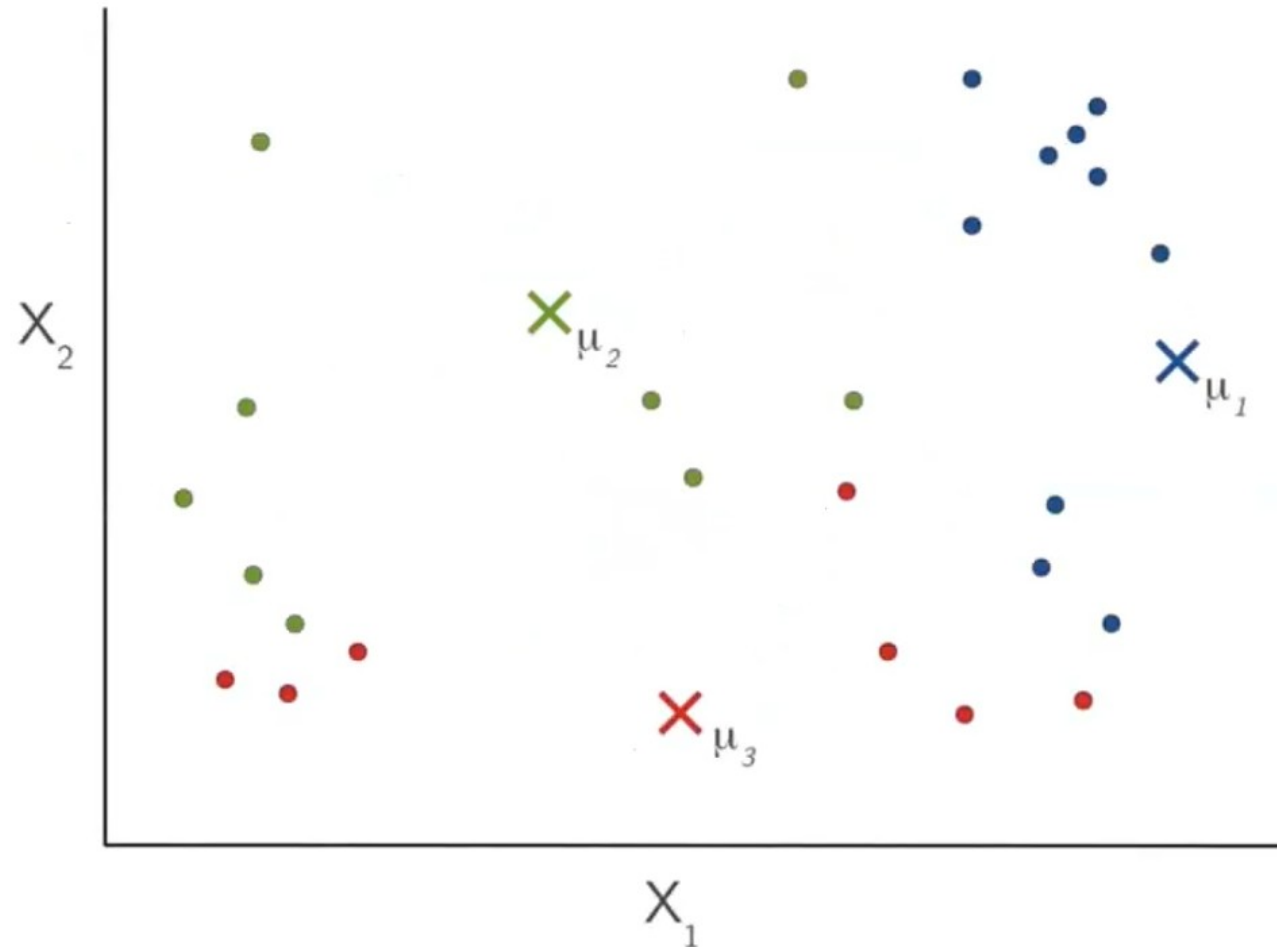
En este caso ese punto pertenece a los verdes

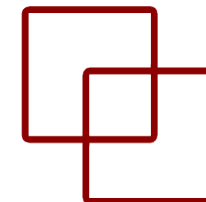


2. Ejemplo ($k=3$)



- Se crean los cluster asociando cada punto al centroide más cercano

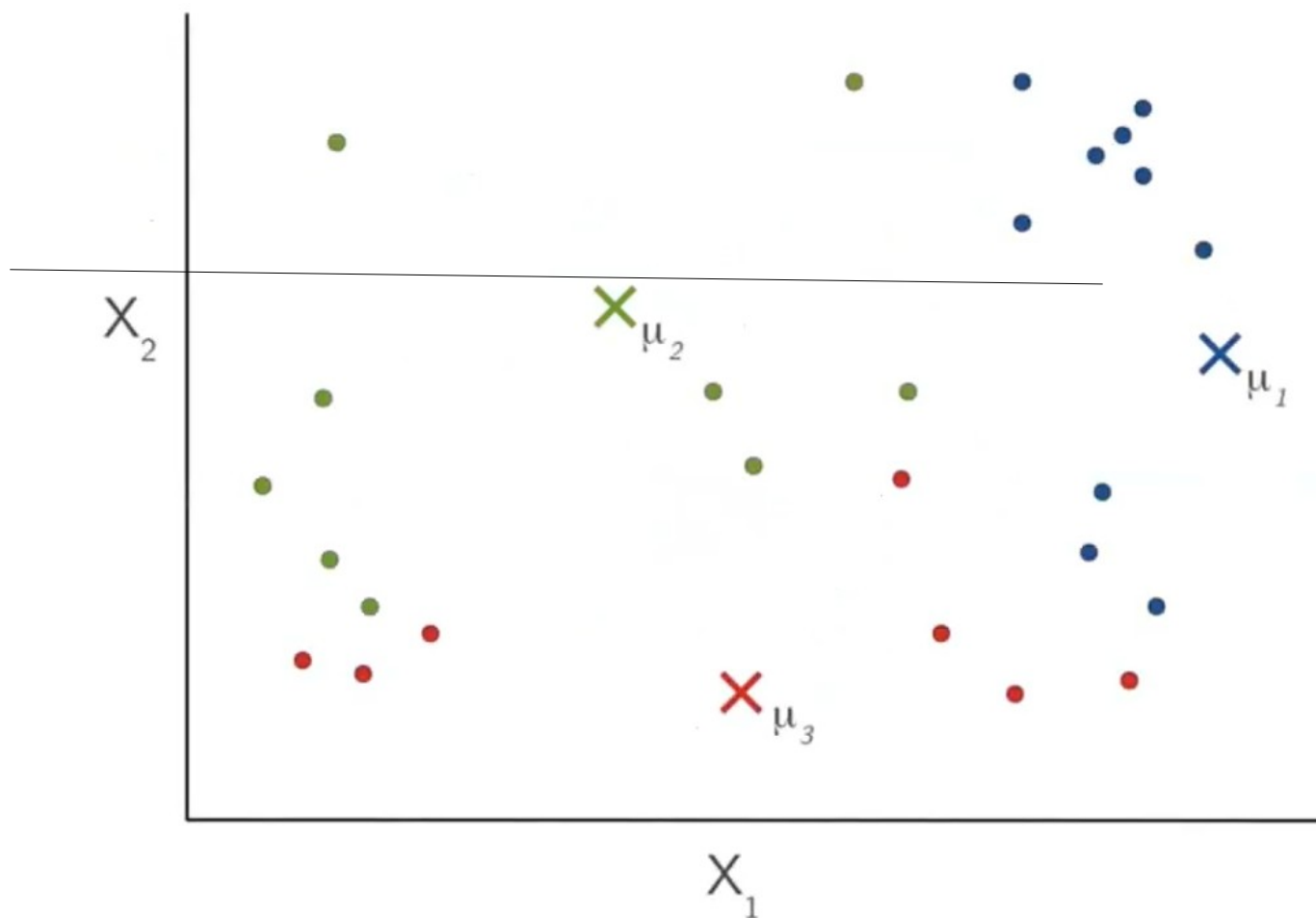




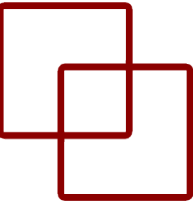
2. Ejemplo ($k=3$)

- Se crean los cluster asociando cada punto al centroide más cercano

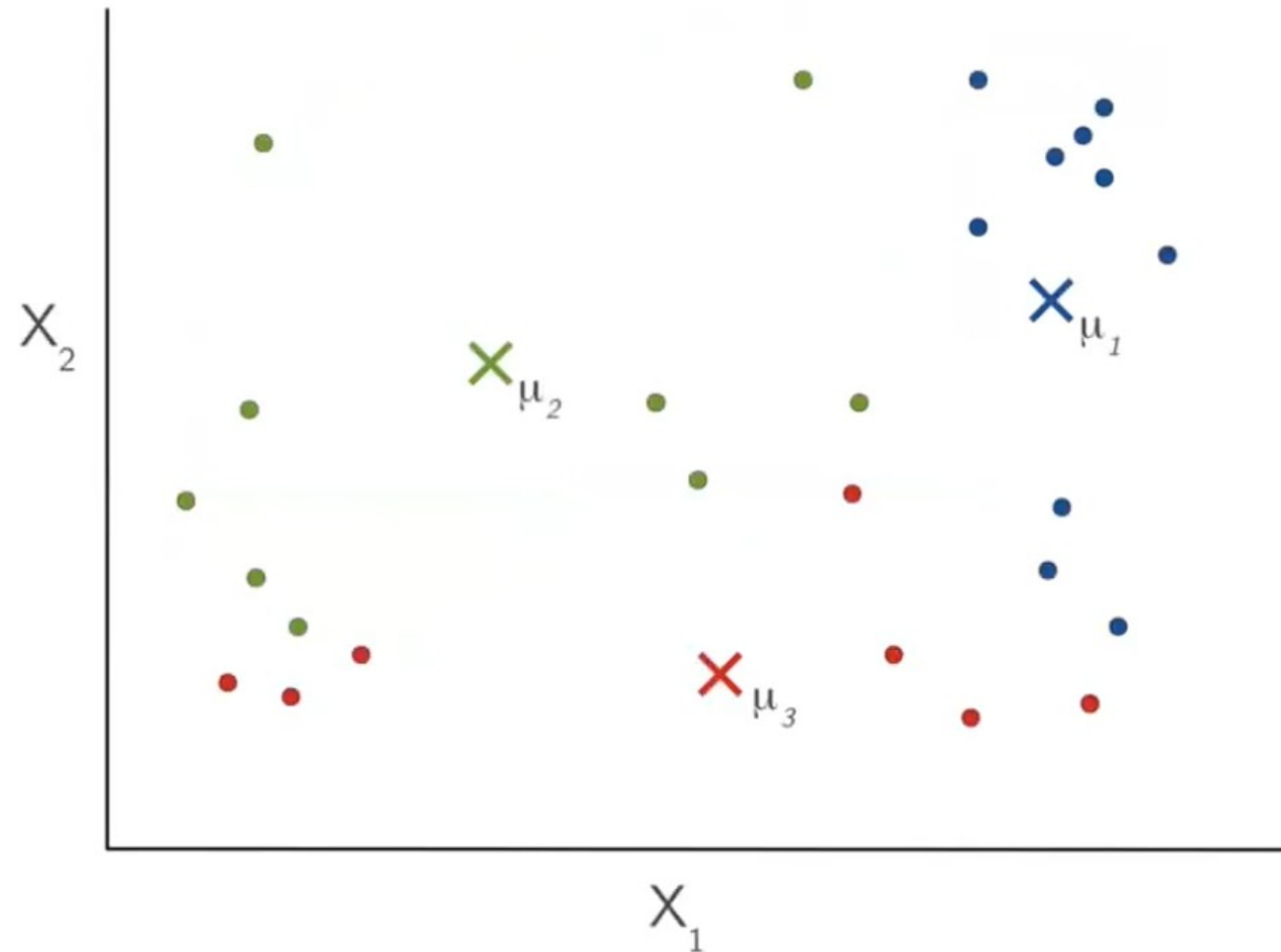
En una segunda iteración busca el punto medio para el centroide de cada cluster



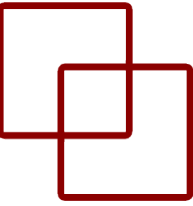
2. Ejemplo ($k=3$)



- Se recalculan los centroides a partir de los puntos que componen cada clúster

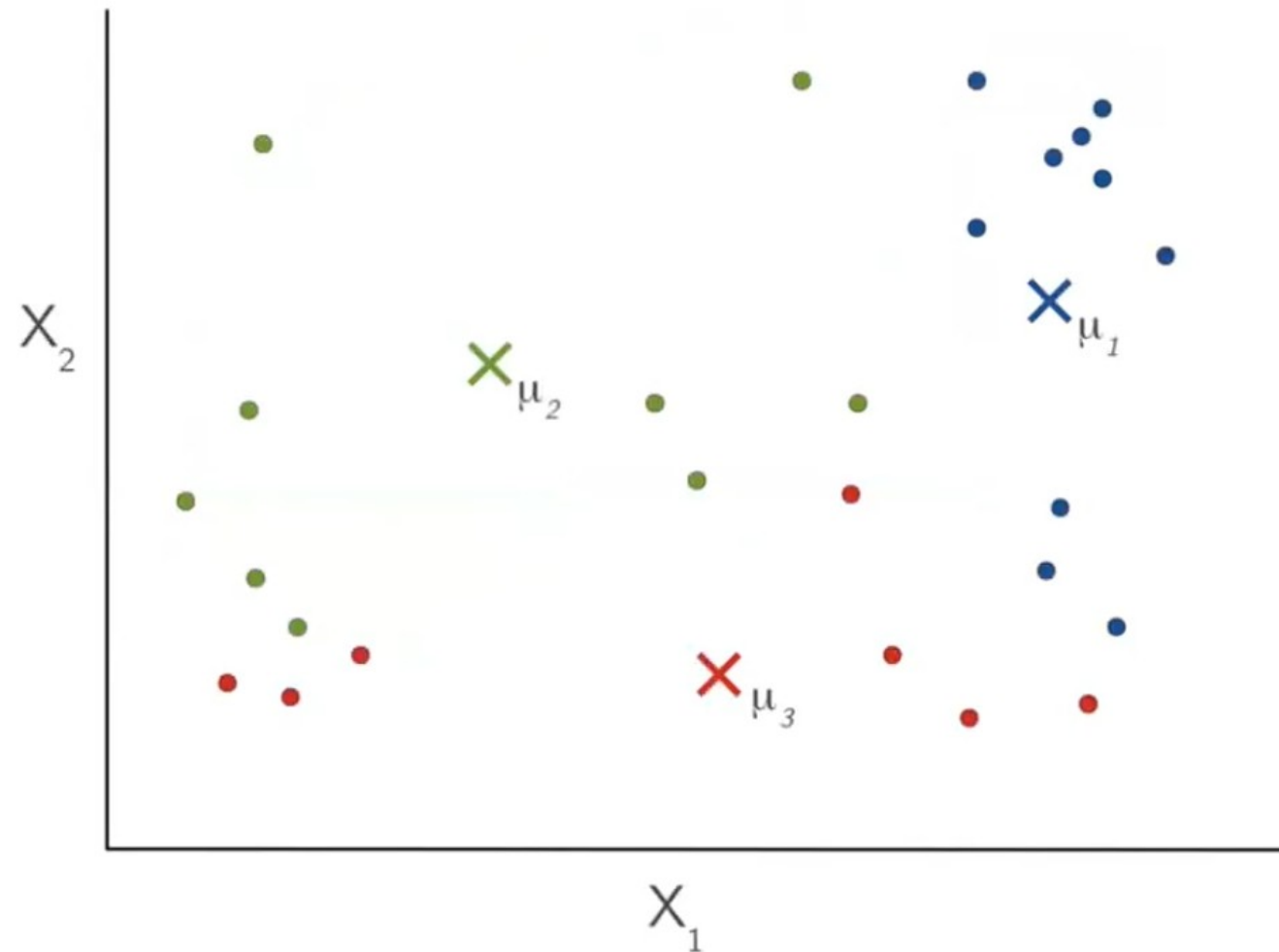


2. Ejemplo ($k=3$)

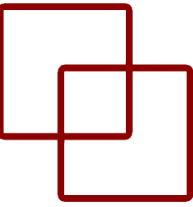


- Se recalculan los centroides a partir de los puntos que componen cada clúster

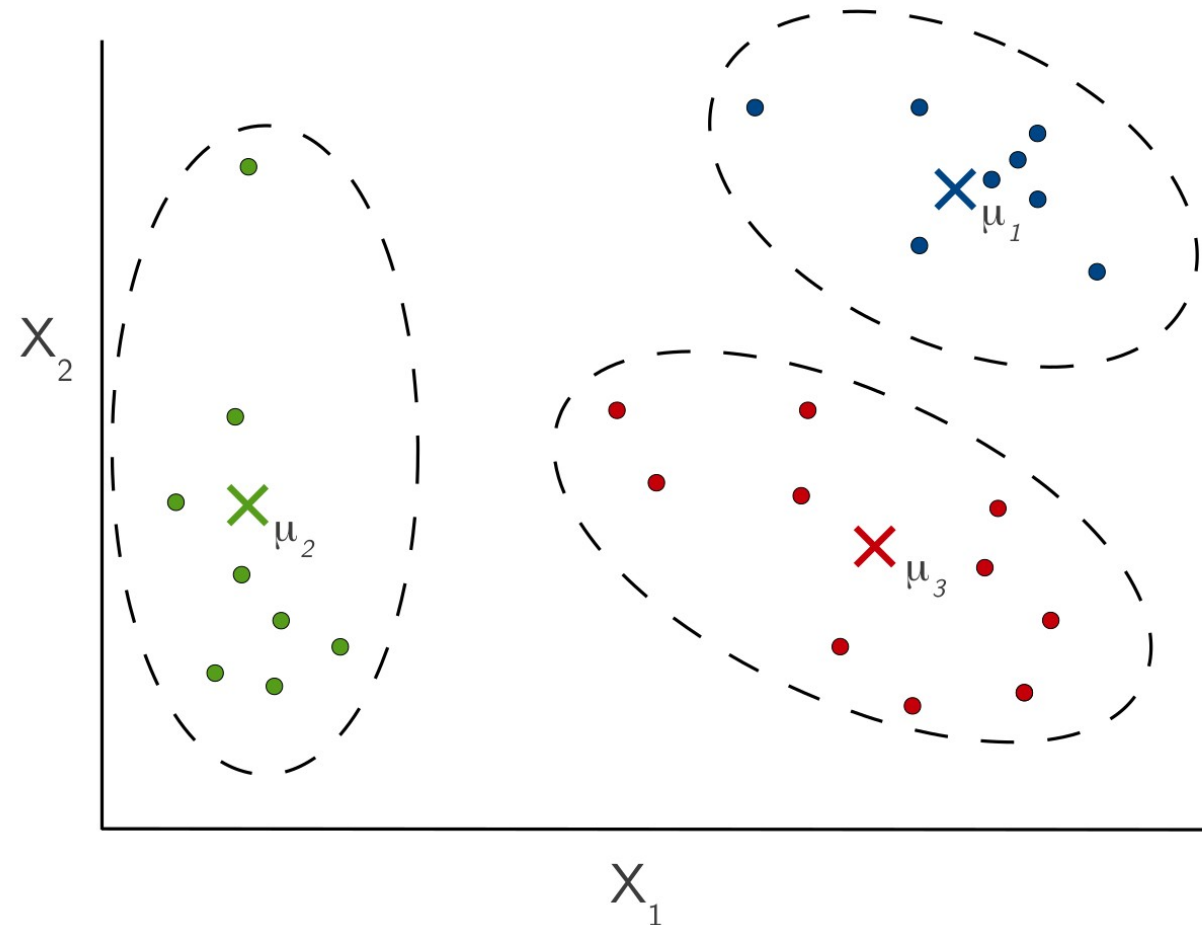
Se realiza el mismo proceso iterativamente



2. Ejemplo ($k=3$)

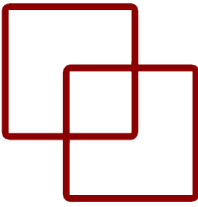


- Hasta que el algoritmo ha convergido (cuando no cambian los centroides)



3. Algoritmo *k*-means

pseudocódigo



- ✓ Entrada:
 - ✗ número de clusters: K
 - ✗ conjunto de entrenamiento: $\{(x^{(1)}), (x^{(2)}), \dots, (x^{(m)})\}$
- ✓ Salida:
 - ✗ $\{(c^{(1)}), \dots, (c^{(m)})\}$, donde $c^{(i)} \in \{1, \dots, K\}$ es el cluster asignado a $x^{(i)}$
 - ✗ El centroide de cada cluster: $\{\mu_1, \dots, \mu_K\}$

K-medias

inicializar aleatoriamente los K centroides $\mu_1, \dots, \mu_K \in \mathbb{R}^n$

repetir {

 para $i = 1$ hasta m

$c^{(i)} :=$ índice del centroide más cercano a $x^{(i)}$

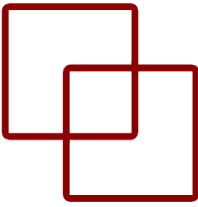
 para $k = 1$ hasta K

$\mu_k :=$ media de los puntos asignados al cluster k

} hasta que no cambie μ_1, \dots, μ_K

3. Algoritmo *k*-means

pseudocódigo



- ✓ Entrada:
 - ✗ número de clusters: K
 - ✗ conjunto de entrenamiento: $\{(x^{(1)}), (x^{(2)}), \dots, (x^{(m)})\}$
- ✓ Salida:
 - ✗ $\{(c^{(1)}), \dots, (c^{(m)})\}$, donde $c^{(i)} \in \{1, \dots, K\}$ es el cluster asignado a $x^{(i)}$
 - ✗ El centroide de cada cluster: $\{\mu_1, \dots, \mu_K\}$

K-medias

inicializar aleatoriamente los K centroides $\mu_1, \dots, \mu_K \in \mathbb{R}^n$

repetir {

para $i = 1$ hasta m

$c^{(i)} :=$ índice del centroide más cercano a $x^{(i)}$

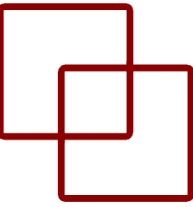
para $k = 1$ hasta K

$\mu_k :=$ media de los puntos asignados al cluster k

} hasta que no cambie μ_1, \dots, μ_K

Fase de
asignación
Fase de
Centroides

4. Función de coste y optimización



- $c^{(i)} \in \{1, \dots, K\}$ = índice del clúster al que se ha asignado $x^{(i)}$
- $\mu_k \in \mathbb{R}^n$ = centroide del cluster k .
- $\mu_{c^{(i)}}$ = centroide del cluster al que ha sido asignado $x^{(i)}$.
- Función de coste:

$$J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K) = \frac{1}{m} \sum_{i=1}^m \|x^{(i)} - \mu_{c^{(i)}}\|^2$$

- Objetivo de la optimización:

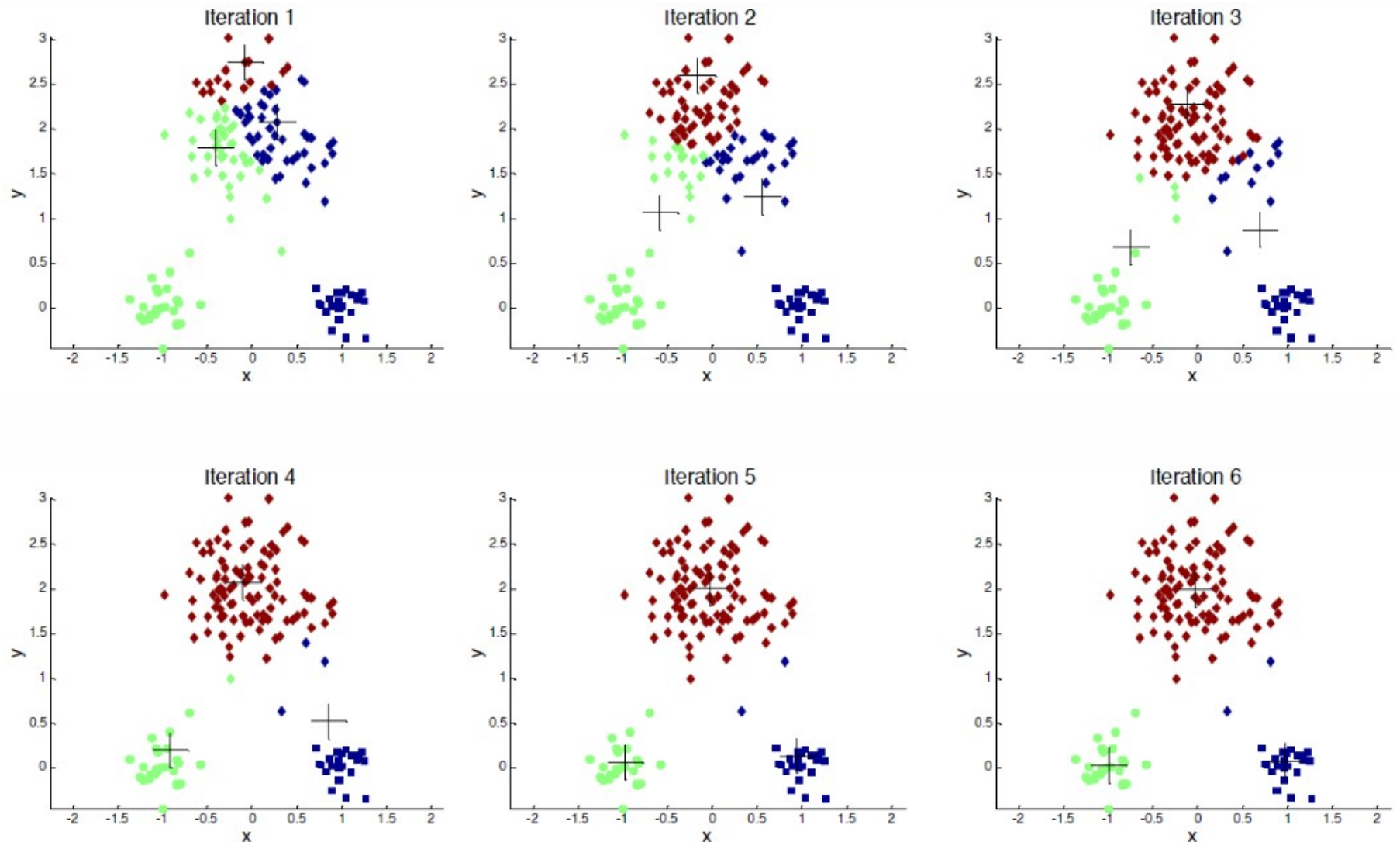
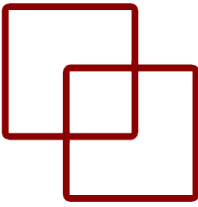
$$\min_{(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)} J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)$$

Objetivo: Minimizar la distancia entre cada punto y su centroide

k-means converge a un **óptimo local** de la función de coste

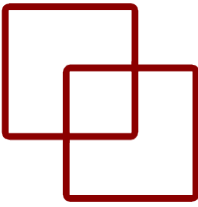
4. Función de coste y optimización

Convergencia a óptimos locales (Pier Luca Lanzi)

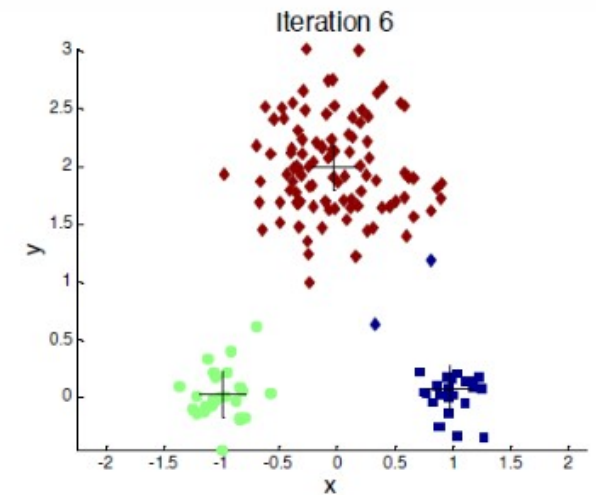
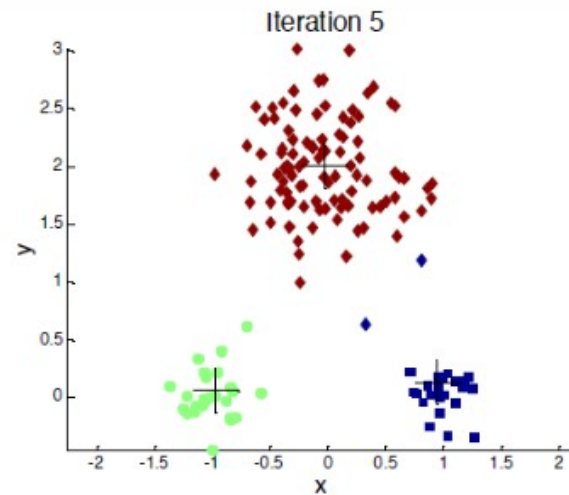
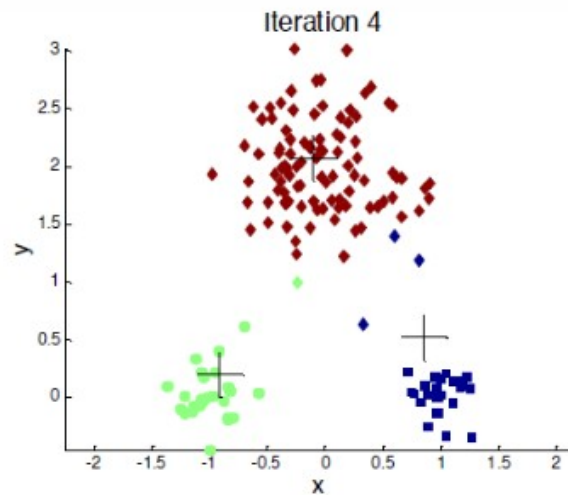
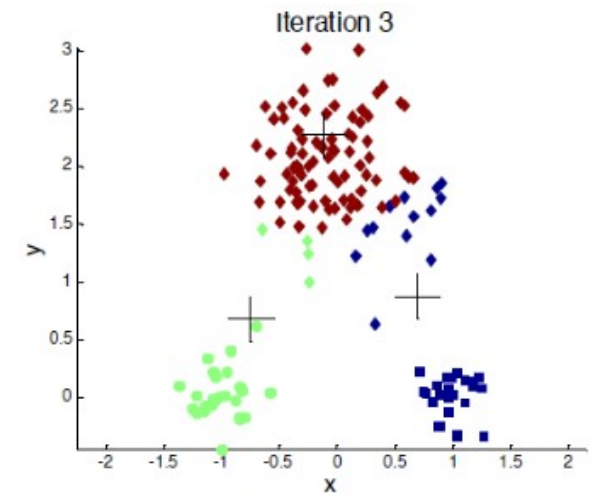
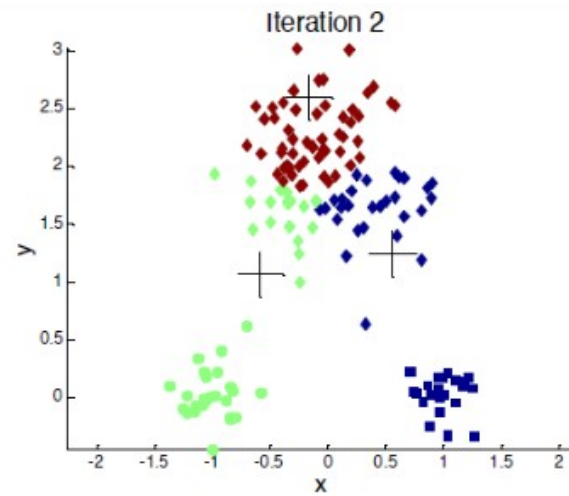
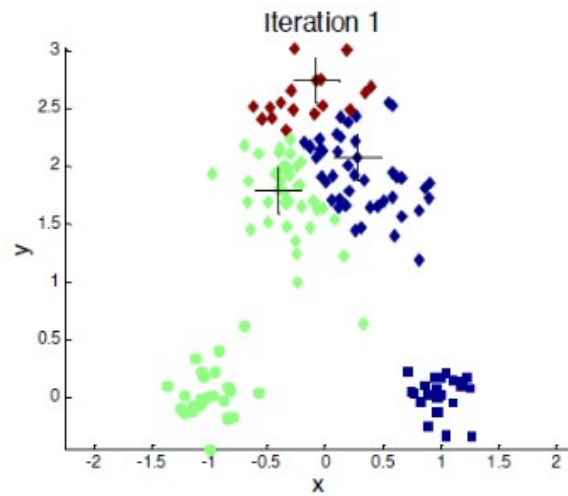


4. Función de coste y optimización

Convergencia a óptimos locales

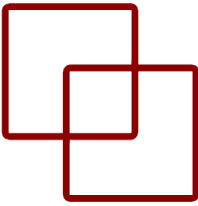


Vemos una convergencia óptima ya que ha escogido unos puntos iniciales bien posicionados



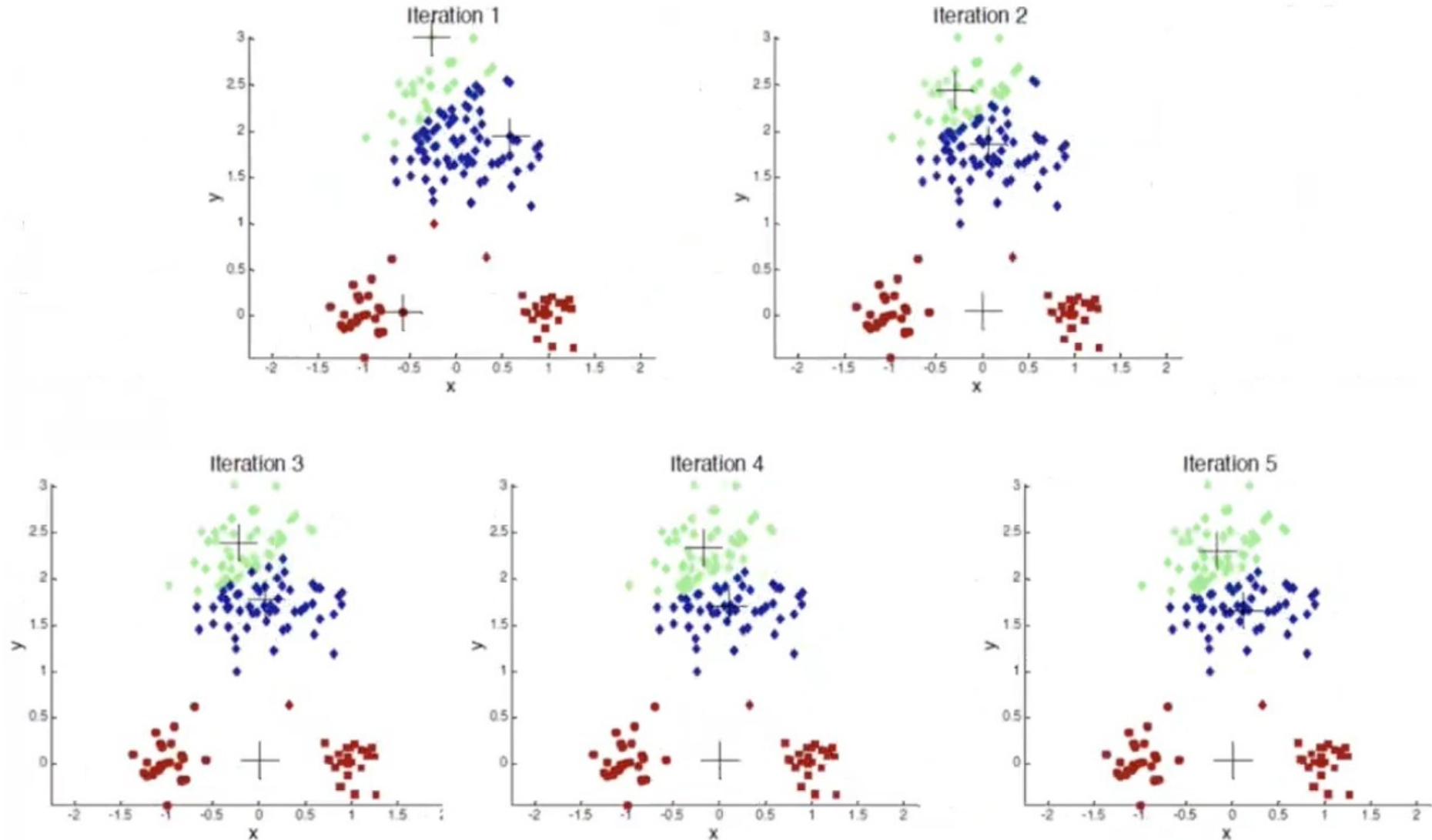
4. Función de coste y optimización

Convergencia a óptimos locales



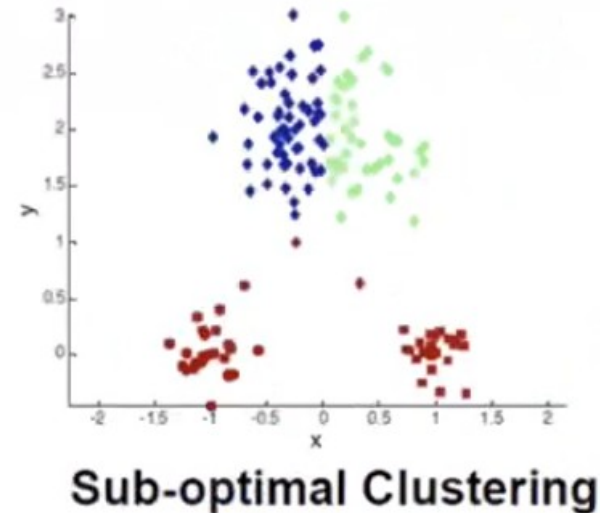
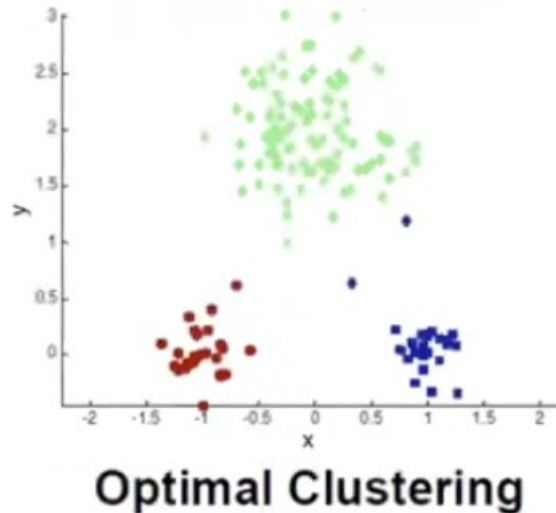
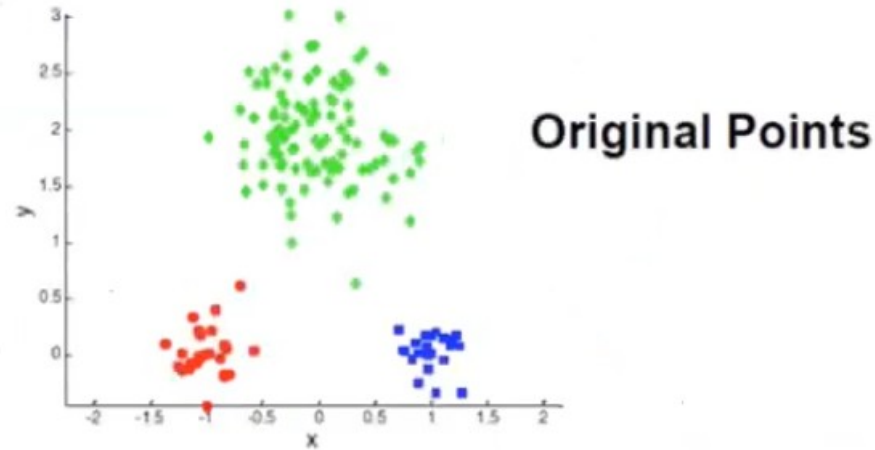
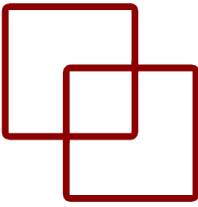
Aquí observamos unos malos clúster por no escoger unos buenos puntos iniciales.

Por tanto, depende de los puntos de inicio



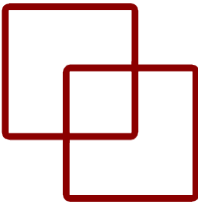
4. Función de coste y optimización

Convergencia a óptimos locales



4. Función de coste y optimización

Inicializaciones aleatorias



- Para obtener una mejor solución, k -means puede ejecutarse varias veces con distintas inicializaciones
- Un modo de hacer la inicialización aleatoria es tomar k elementos del conjunto de entrenamiento y utilizarlos como centroides.
- Demos:
 - <http://shabal.in/visuals/kmeans/6.html>
 - <http://stanford.edu/class/ee103/visualizations/kmeans/kmeans.html>

K-medias

inicializar aleatoriamente los K centroides $\mu_1, \dots, \mu_K \in \mathbb{R}^n$

para $i = 1$ hasta 100

 Inicializar aleatoriamente los k centroides

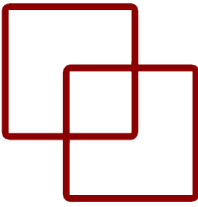
 Ejecutar k -medias y obtener $(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)$

 Calcular $J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)$

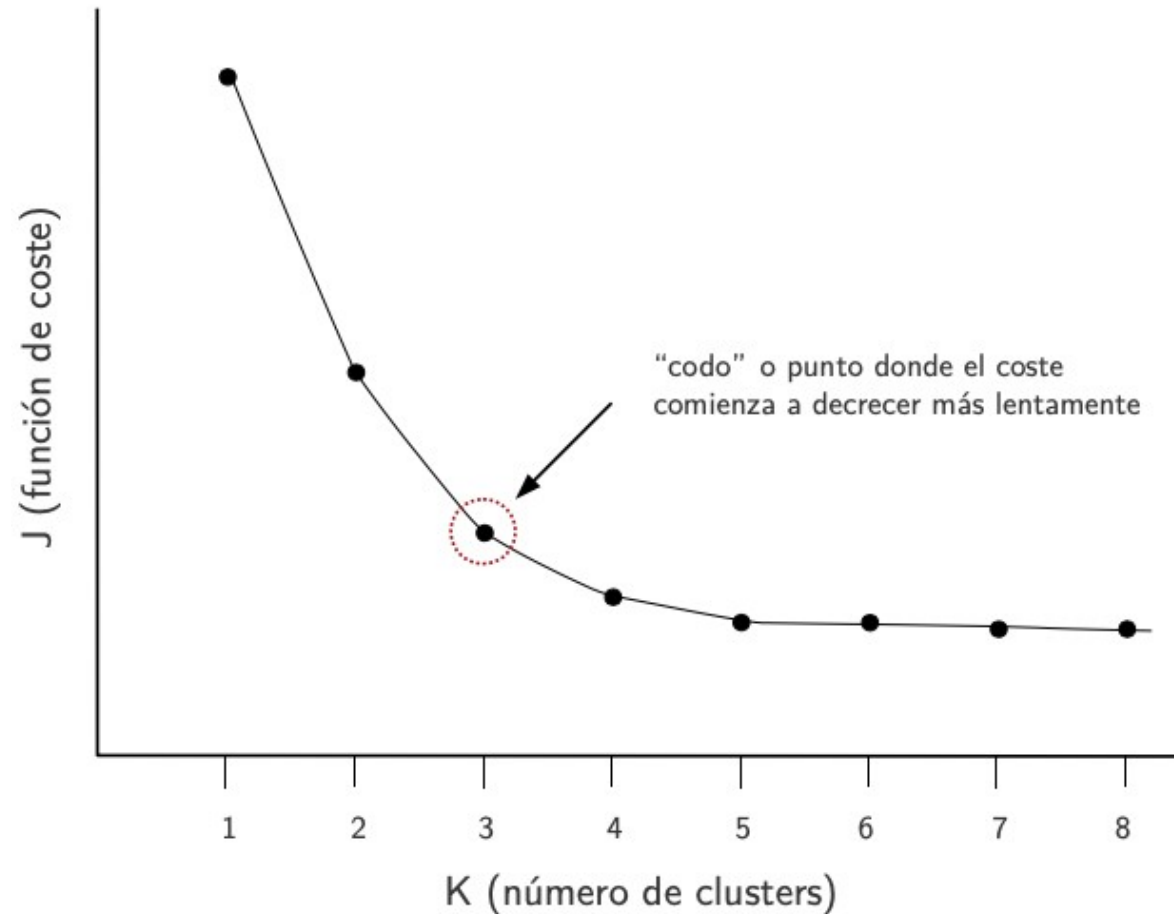
escoger el resultado de la ejecución que minimize $J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)$

5. Elección del número de clústers

¿Cual es el número idóneo de clústers?

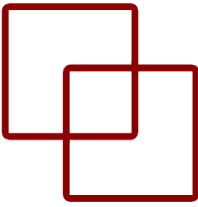


- Demo: <http://cs.joensuu.fi/sipu/clustering/animador/>

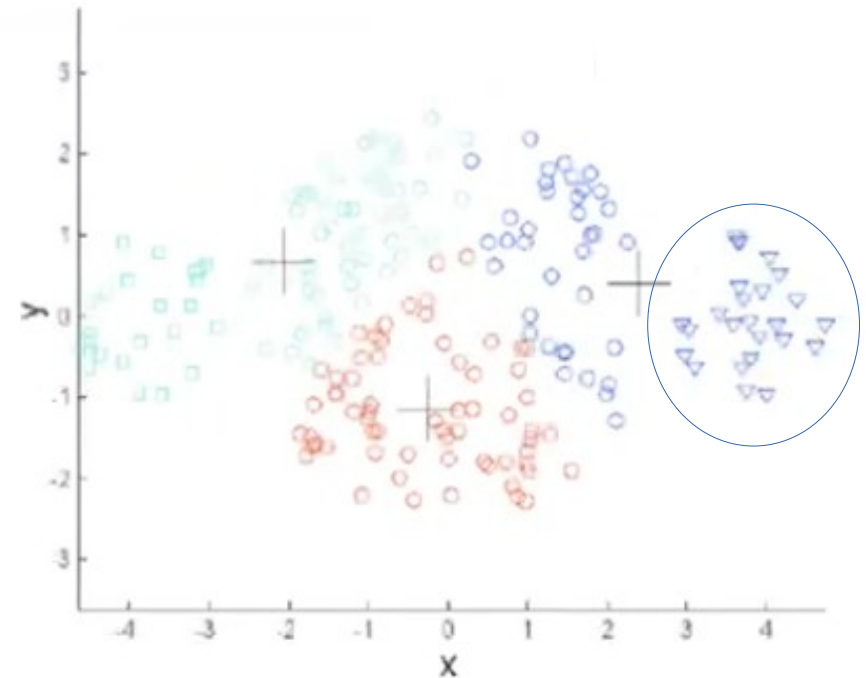
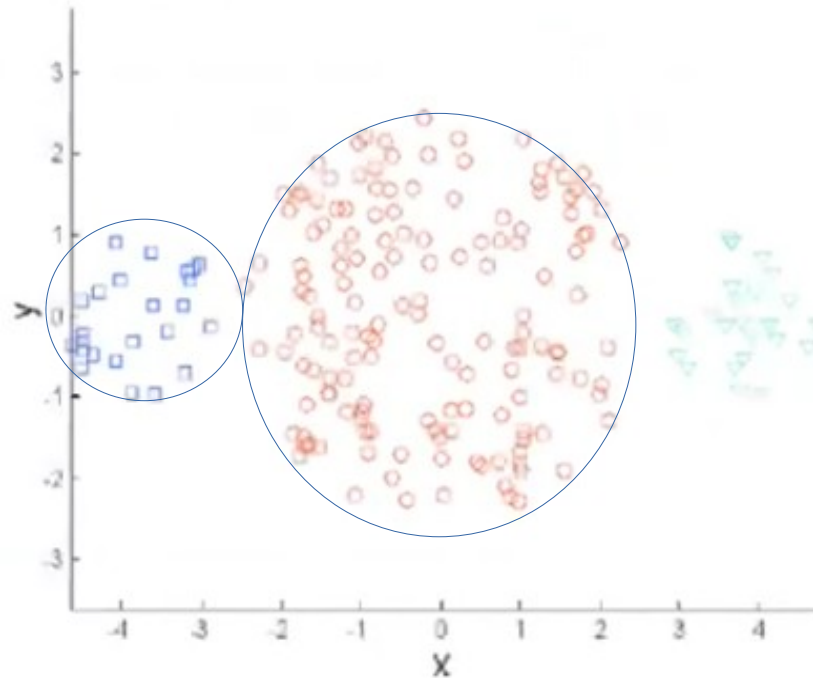


6. Limitaciones de k -means

Limitaciones

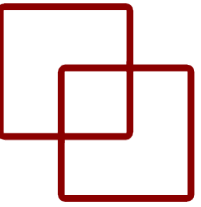


- k -medias no funciona bien cuando los datos tienen mucho ruido
- No es adecuado tampoco para conjuntos de datos en los que los clusters o grupos originales presentan algunas características:
 - **Diferentes tamaños**
 - Diferentes densidades
 - Forma no globular

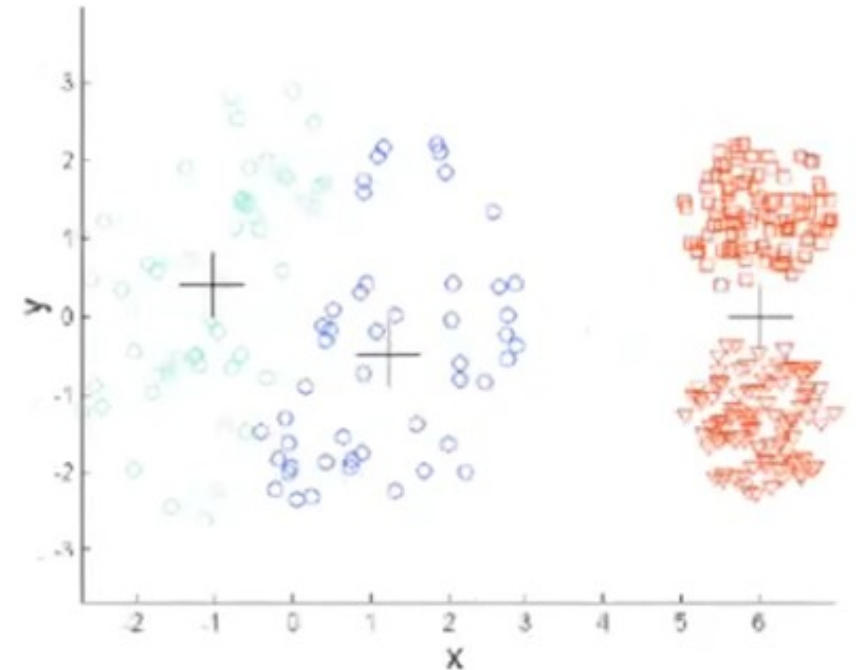
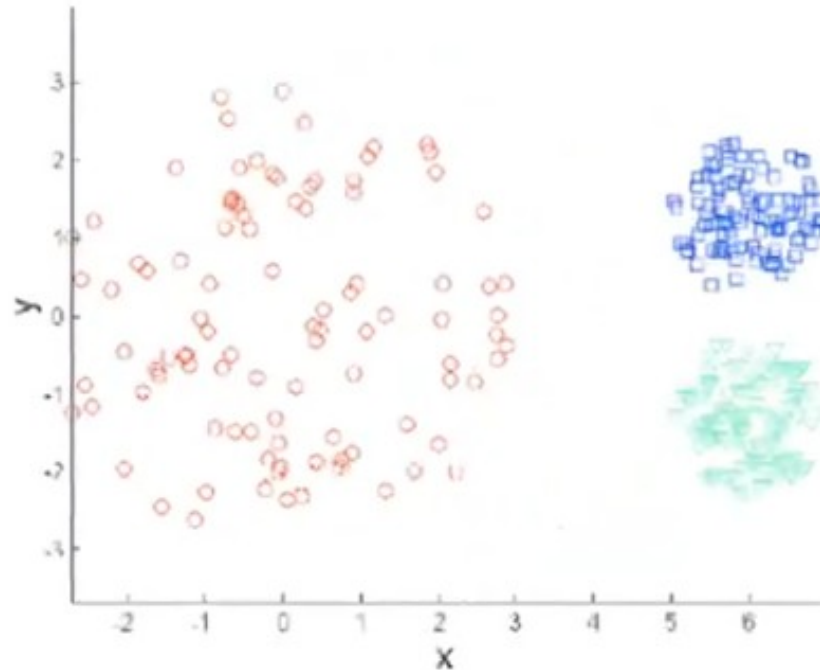


6. Limitaciones de k -means

Limitaciones

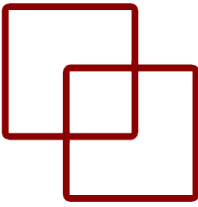


- k -medias no funciona bien cuando los datos tienen mucho ruido
- No es adecuado tampoco para conjuntos de datos en los que los clusters o grupos originales presentan algunas características:
 - Diferentes tamaños
 - **Diferentes densidades**
 - Forma no globular

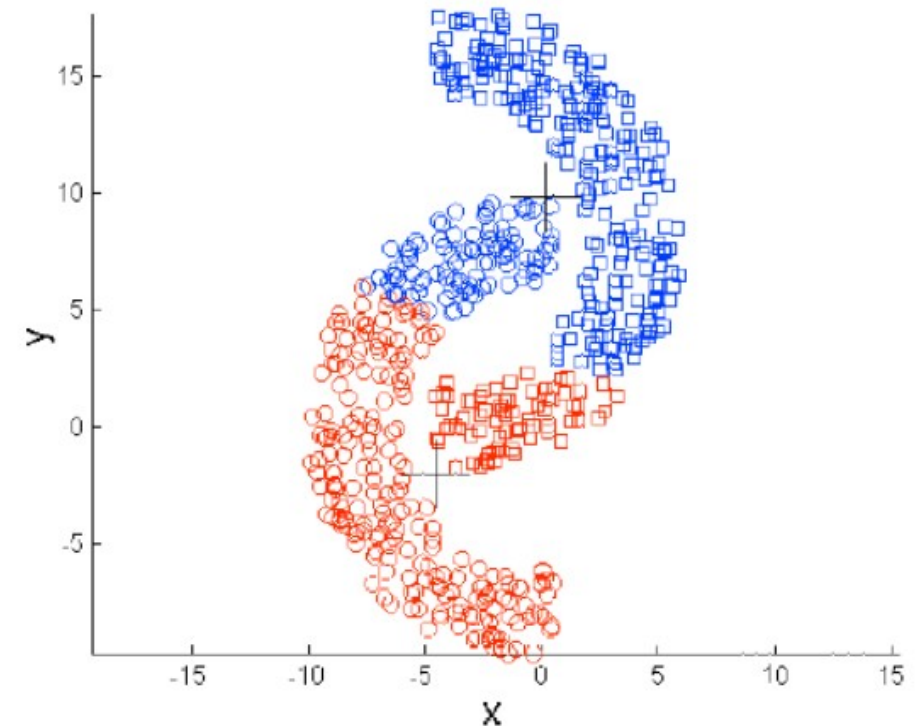
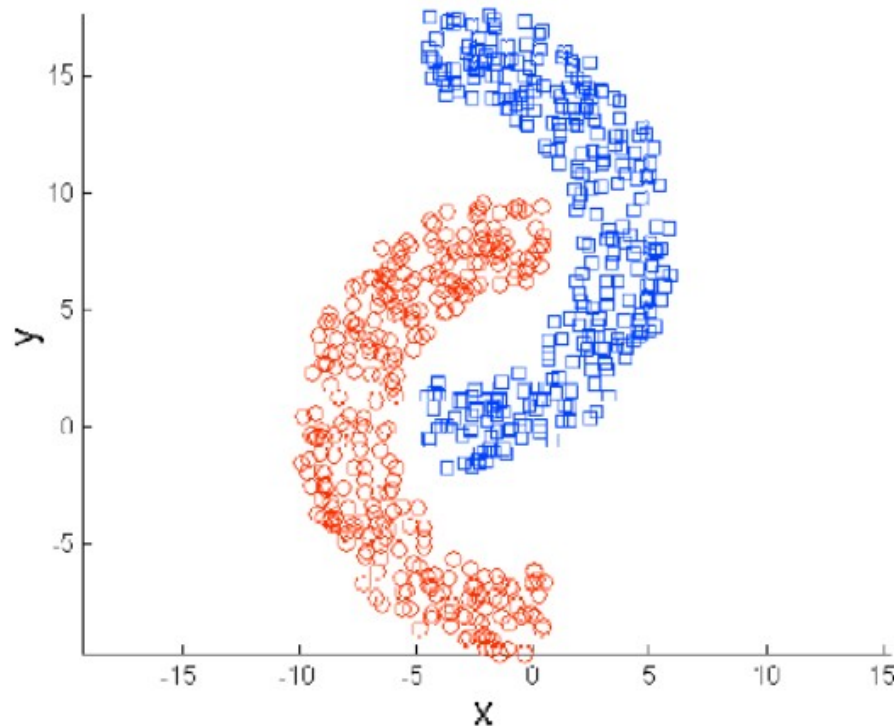


6. Limitaciones de k -means

Limitaciones

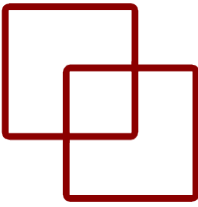


- k -medias no funciona bien cuando los datos tienen mucho ruido
- No es adecuado tampoco para conjuntos de datos en los que los clusters o grupos originales presentan algunas características:
 - Diferentes densidades
 - Diferentes tamaños
 - **Forma no globular**



6. Limitaciones de k -means

Normalización



Es muy importante que todas las variables estén en la misma escala

- Ejemplo

$$\mathbf{x}^{(1)} = (5, 0.2, 0.9, 0.5, 0.1)$$

$$\mathbf{x}^{(2)} = (7, 0.2, 0.9, 0.5, 0.1)$$

$$\mathbf{x}^{(3)} = (6, 0.8, 0.1, 0.2, 0.9)$$

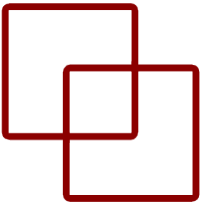
$$||\mathbf{x}^{(1)} - \mathbf{x}^{(2)}||^2 = 4 + 0 + 0 + 0 + 0 = 4$$

$$||\mathbf{x}^{(1)} - \mathbf{x}^{(3)}||^2 = 1 + 0.36 + 0.64 + 0.09 + 0.64 = 2.73$$

- $\mathbf{x}^{(1)}$ y $\mathbf{x}^{(2)}$ son instancias muy similares. Sin embargo, la distancia entre $\mathbf{x}^{(1)}$ y $\mathbf{x}^{(3)}$ es mucho menor. Al no normalizar, la primera variable pesa mucho más en el cálculo de la distancia.

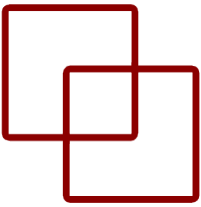
6. Ejemplos de k -means

Compresión y segmentación de imágenes (I)



6. Ejemplos de k -means

Compresión y segmentación de imágenes (II)



$K = 2$



$K = 3$



$K = 10$

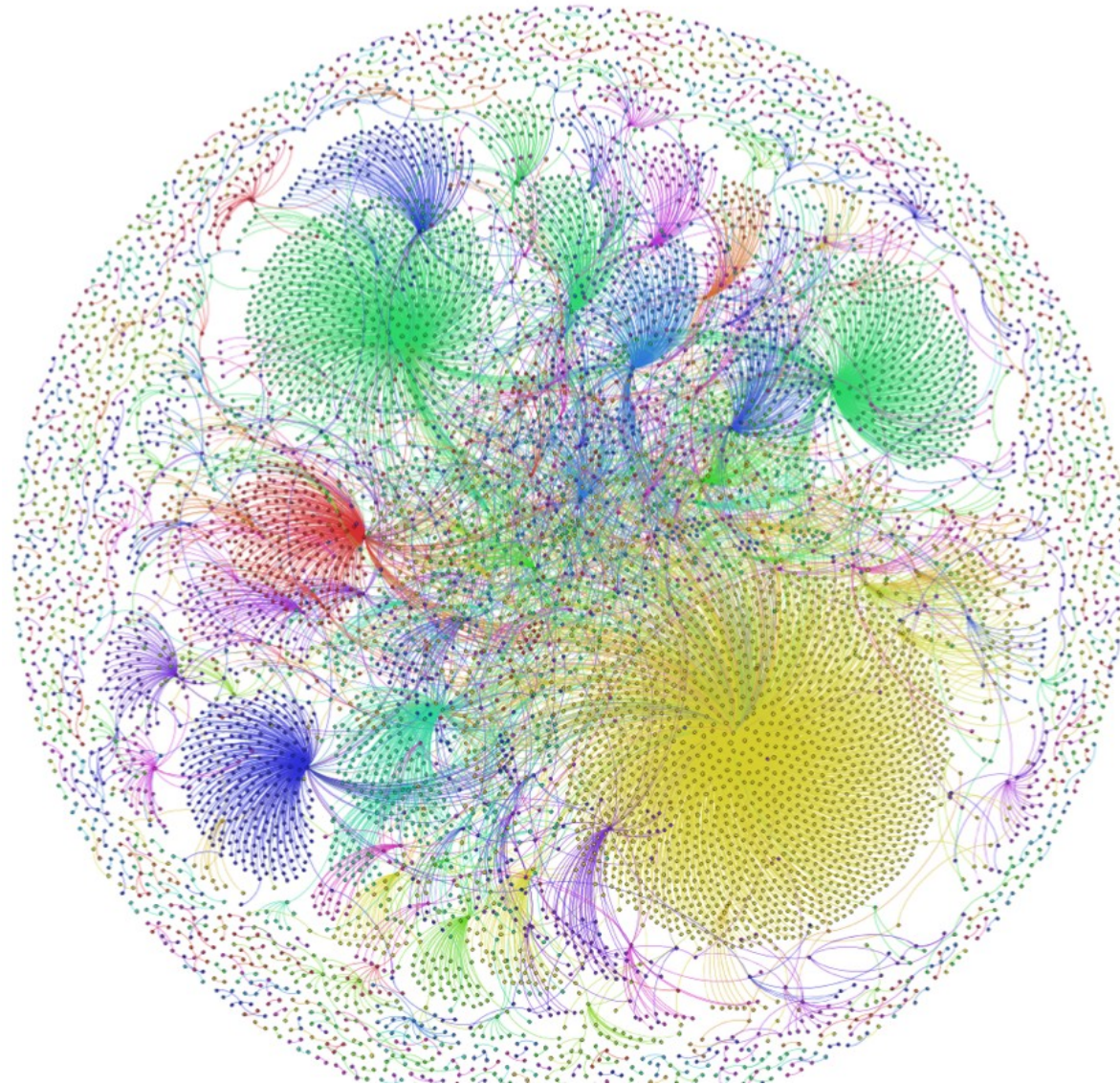
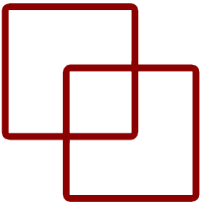


Original image



6. Ejemplos de *k*-means

Detección de comunidades en redes sociales (I)



6. Ejemplos de k -means

Detección de comunidades en redes sociales (I)

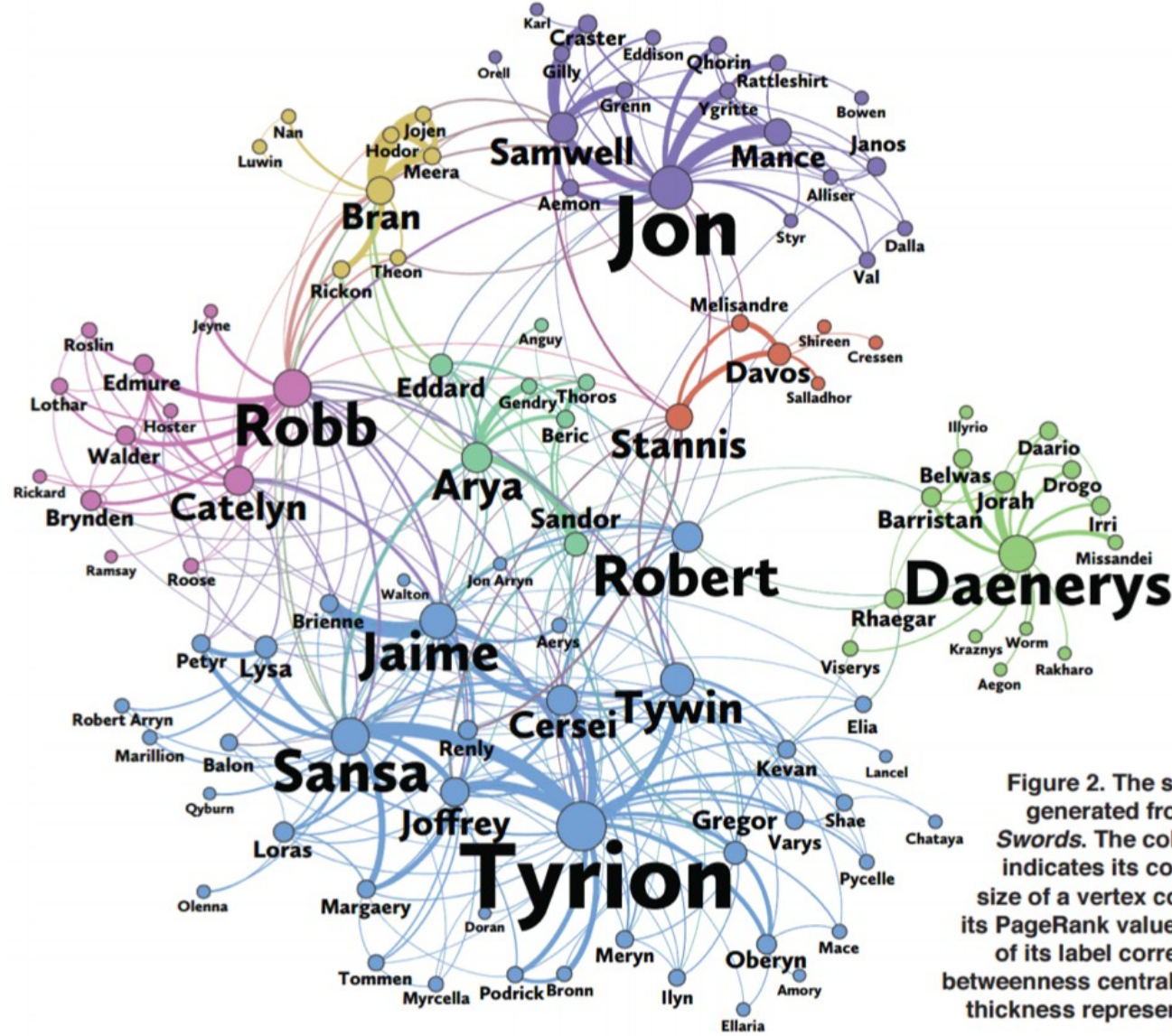
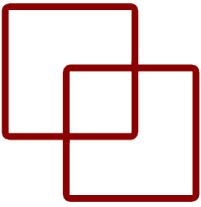
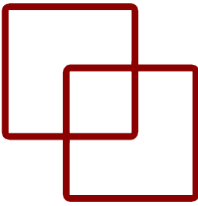


Figure 2. The social network generated from *A Storm of Swords*. The color of a vertex indicates its community. The size of a vertex corresponds to its PageRank value, and the size of its label corresponds to its betweenness centrality. An edge's thickness represents its weight.

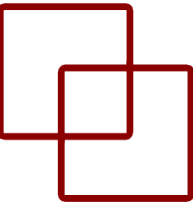


7. *k*-means en scikit-learn

- Se implementa en la clase `sklearn.cluster.KMeans`.
- Los parámetros más importantes son:
 - `n_clusters`. *k*, es decir, el número de clústers.
 - `max_iter`. Máximo número de iteraciones.
 - `init`. Inicialización. Por defecto usa *k-means++*.
- Los atributos de interés, disponibles una vez ejecutado el algoritmo, son:
 - `cluster_centers_`. Coordenadas de los centroides de cada clúster
 - `labels_`. Etiqueta de cada elemento (clúster al que pertenece).
 - `inertia_`. Coste asociado a ese agrupamiento.

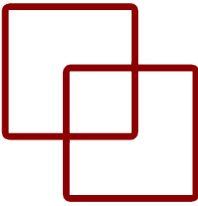


Clústering jerárquico



1. Definición

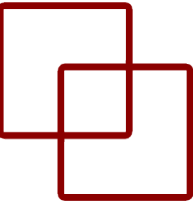
- El clustering jerárquico no requiere que se fije a priori el número de clústers.
- Progresivamente une los dos clústers más cercanos en un solo clúster.
 - También puede proceder al revés, **clustering divisivo**, dividiendo clústeres, pero lo que se usa con mucha más frecuencia es el clustering aglomerativo.
- En muchos casos, el árbol (dendograma) que produce la secuencia de fusiones representa taxonomías.



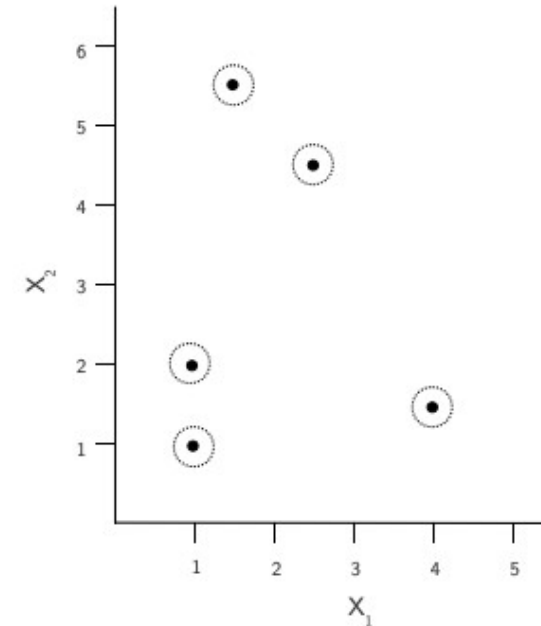
1. Definición

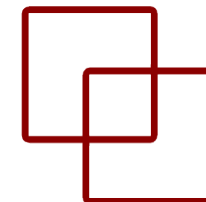
- El clustering jerárquico no requiere que se fije a priori el número de clústers.
- Progresivamente une los dos clústers más cercanos en un solo clúster.
 - También puede proceder al revés, **clustering divisivo**, dividiendo clústeres, pero lo que se usa con mucha más frecuencia es el clustering aglomerativo.
- En muchos casos, el árbol (dendograma) que produce la secuencia de fusiones representa taxonomías.
- Es necesario usar una medida de similaridad para determinar qué clústeres se funden/separan en cada paso. Dados dos clústeres, esta distancia puede ser:
 - Mínima distancia entre un elemento de un clúster y un elemento del otro clúster
 - Máxima distancia entre un elemento de un clúster y un elemento del otro clúster
 - Distancia entre los centroides de los clústeres
 - etc.

2. Clustering jerárquico



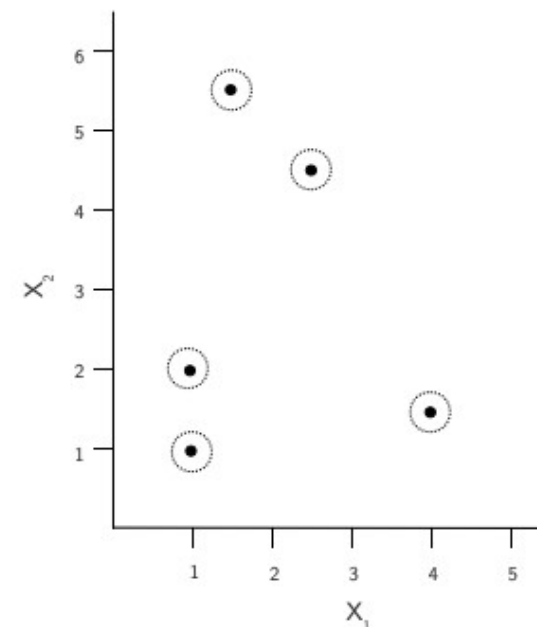
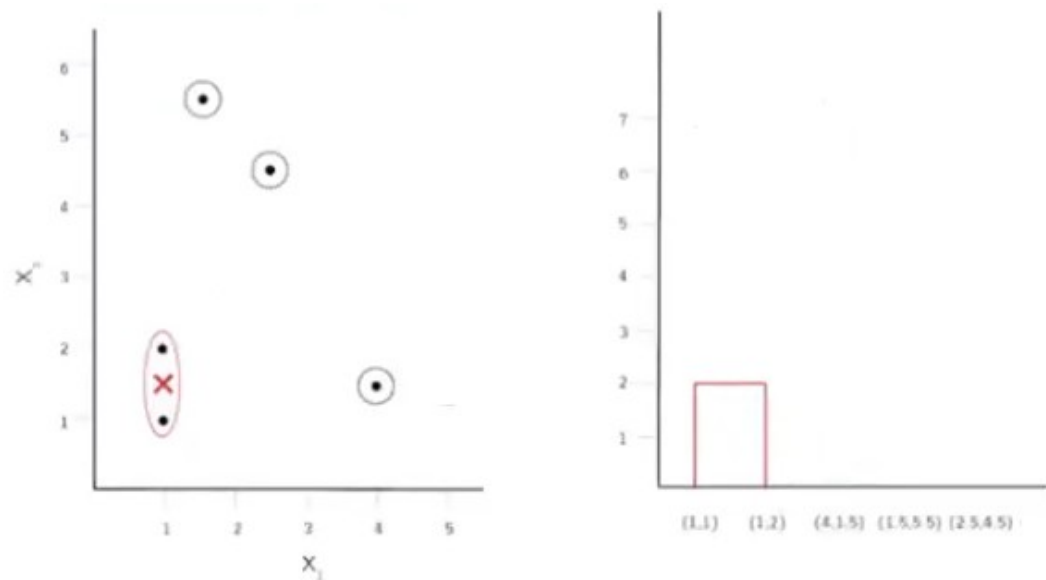
- Se parte de tantos clústers como elementos hay en el conjunto de entrenamiento (m). En este caso 5.



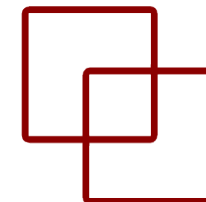


2. Clustering jerárquico

- Se parte de tantos clústers como elementos hay en el conjunto de entrenamiento (m). En este caso 5.

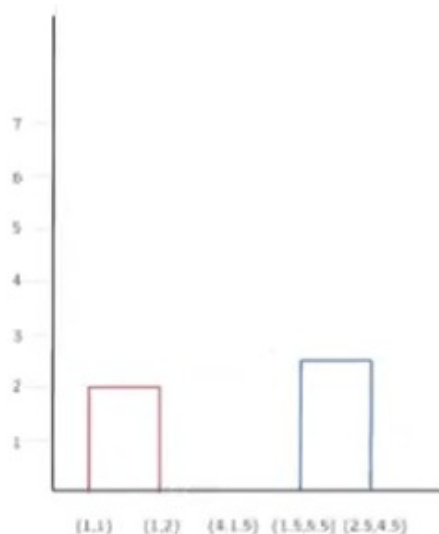
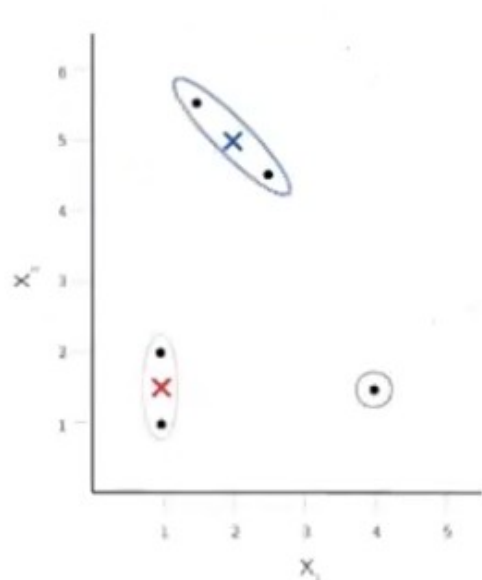
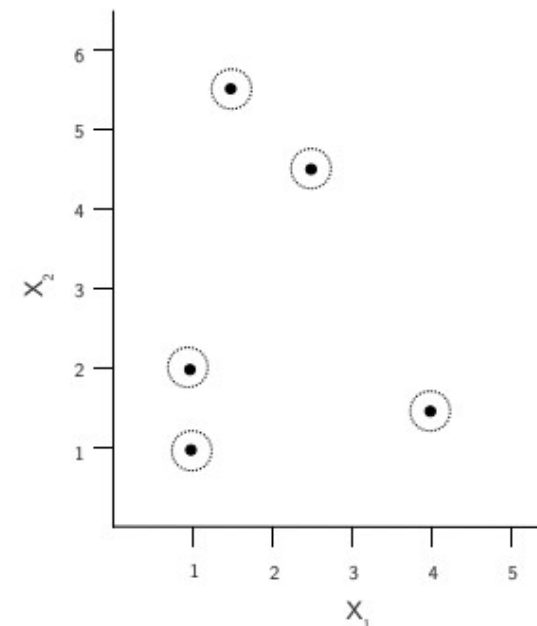


- En cada paso se une el par de clusters más cercanos entre sí y disminuye el número de clusters



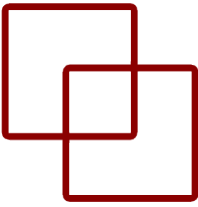
2. Clustering jerárquico

- Se parte de tantos clústers como elementos hay en el conjunto de entrenamiento (m). En este caso 5.

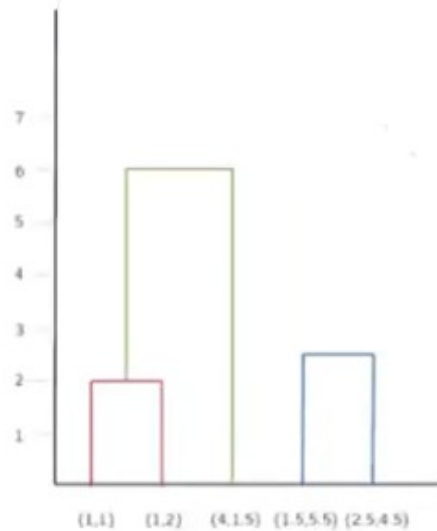
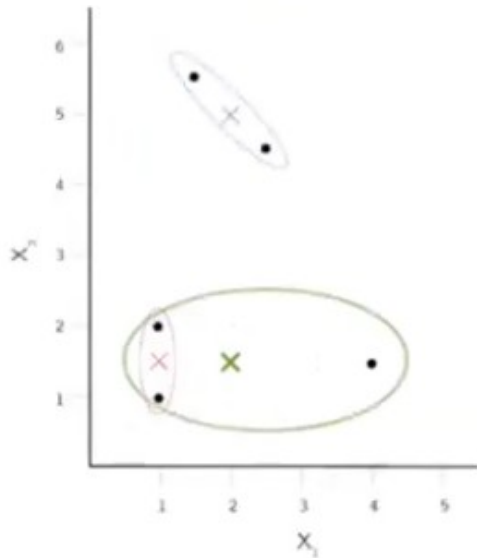
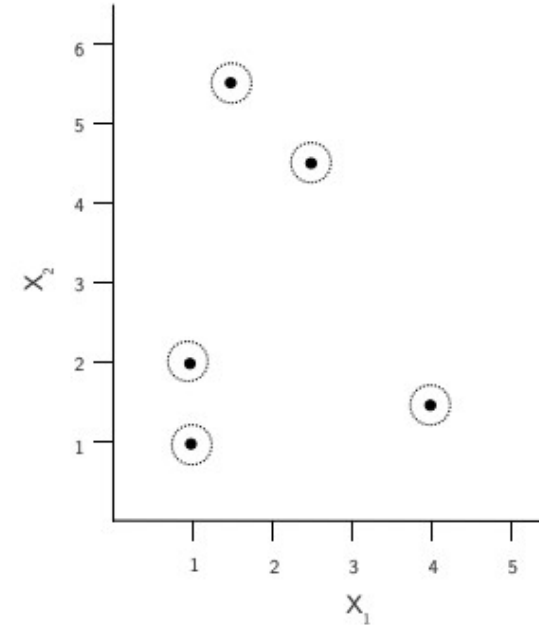


- En cada paso se une el par de clusters más cercanos entre sí y disminuye el número de clusters.
- Para medir las distancia calcula el centroide entre puntos

2. Clustering jerárquico

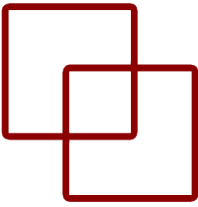


- Se parte de tantos clústers como elementos hay en el conjunto de entrenamiento (m). En este caso 5.

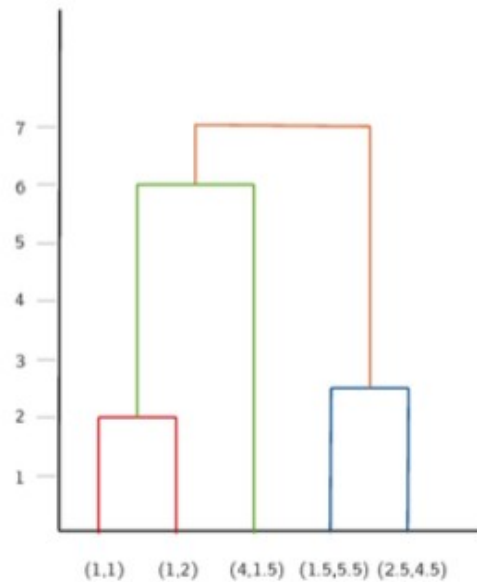
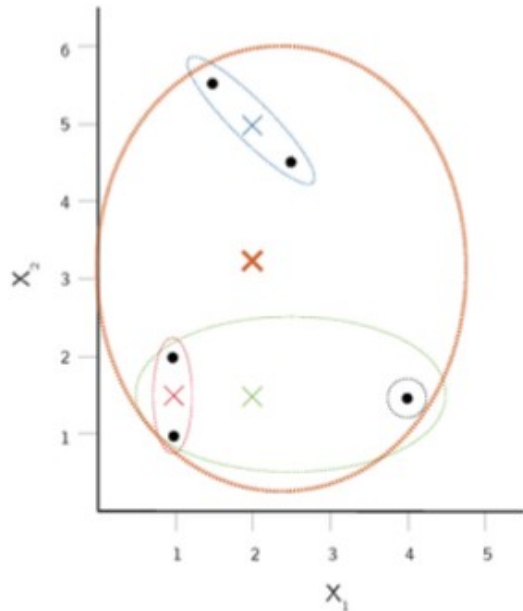
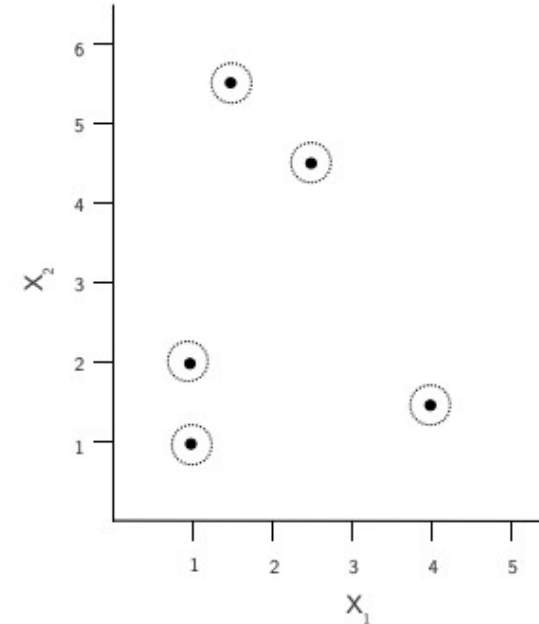


- En cada paso se une el par de clusters más cercanos entre sí y disminuye el número de clusters

2. Clustering jerárquico

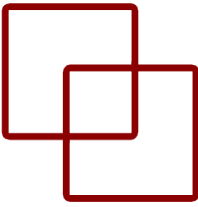


- Se parte de tantos clústers como elementos hay en el conjunto de entrenamiento (m). En este caso 5.



- En cada paso se une el par de clusters más cercanos entre sí y disminuye el número de clusters.
- Este proceso es el dendrograma

3. Algoritmo de clustering jerárquico Aglomerativo



✓ Entrada:

✗ conjunto de entrenamiento: $\{(x^{(1)}), (x^{(2)}), \dots, (x^{(m)})\}$

✓ Salida:

✗ $\{(c^{(1)}), \dots, (c^{(m)})\}$, donde $c^{(i)} \in \{1, \dots, K\}$ es el cluster asignado a $x^{(i)}$

AGNES

inicializar $\mu_i = x^{(i)}$

centroides $\leftarrow \{\mu_1, \dots, \mu_m\}$

repetir {

$k, l \leftarrow \min_{\substack{k, l \\ \mu_k, \mu_l \in \text{centroides}}} \text{distancia}(\mu_k, \mu_l)$

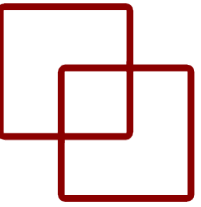
unir los clusters k y l en un nuevo cluster k'

calcular $\mu_{k'}$ y añadirlo a *centroides*

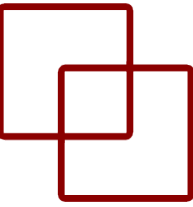
borrar μ_k, μ_l de *centroides*

} hasta criterio de parada

4. Criterios de parada



- En clustering jerárquico se pueden utilizar dos criterios de parada:
 1. **Un umbral de distancia.** De modo que cuando la distancia entre los dos clusters que corresponde fundir supera ese umbral, el algoritmo se detiene.
 2. **El número de clusters.**

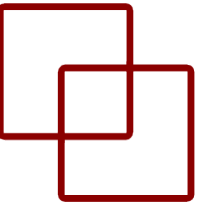


4. Criterios de parada

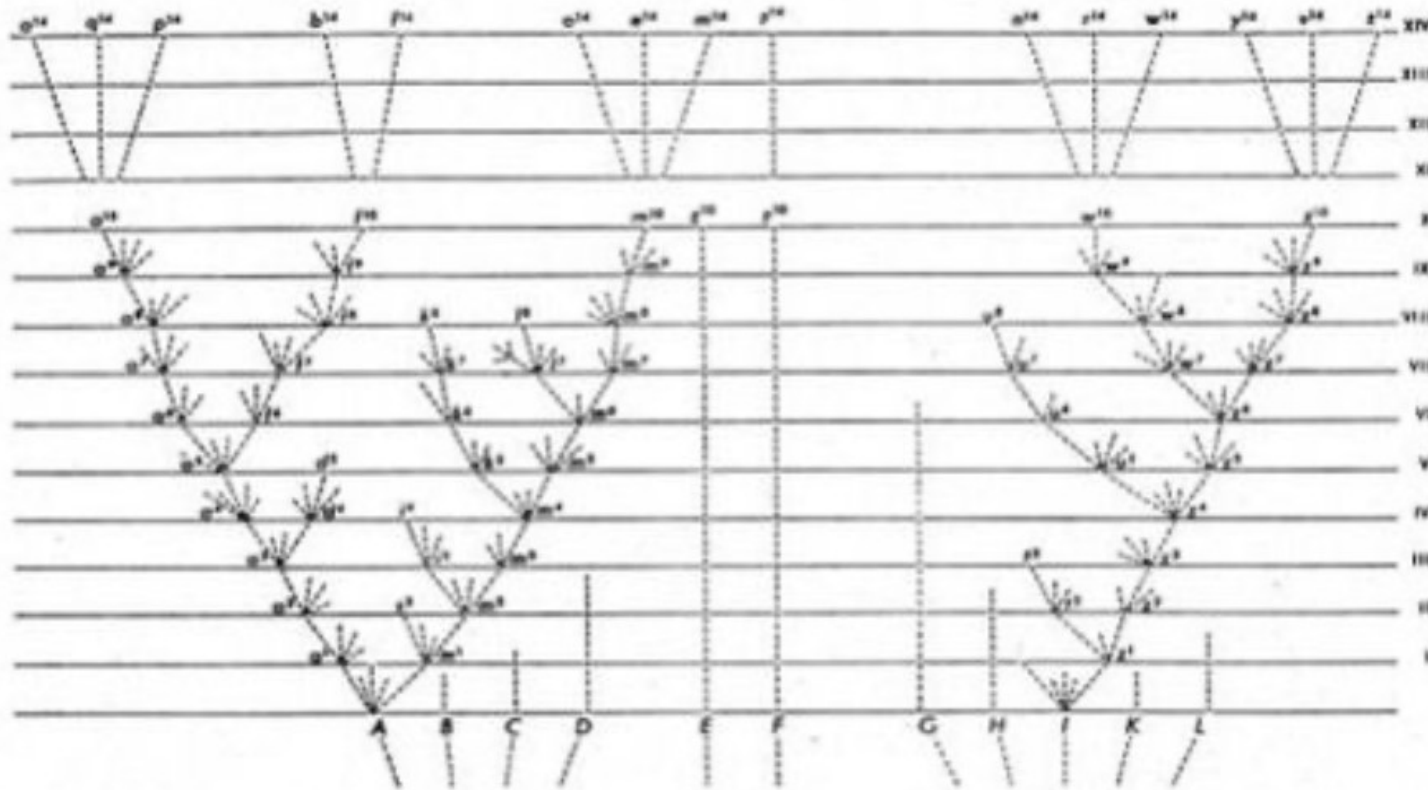
- En clustering jerárquico se pueden utilizar dos criterios de parada:
 1. **Un umbral de distancia.** De modo que cuando la distancia entre los dos clusters que corresponde fundir supera ese umbral, el algoritmo se detiene.
 2. **El número de clusters.**
- Sin embargo, una de las ventajas de este algoritmo es que se puede hacer un análisis del número de clusters a posteriori.

5. Ejemplo de aplicación

Árboles filogenéticos (I)

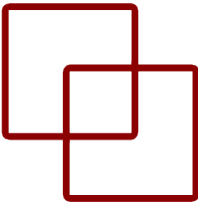


- Un **árbol filogenético** es un árbol que muestra las relaciones evolutivas entre varias especies u otras entidades que se cree que tienen una ascendencia común.
- Un árbol filogenético es una forma de **dendograma**.

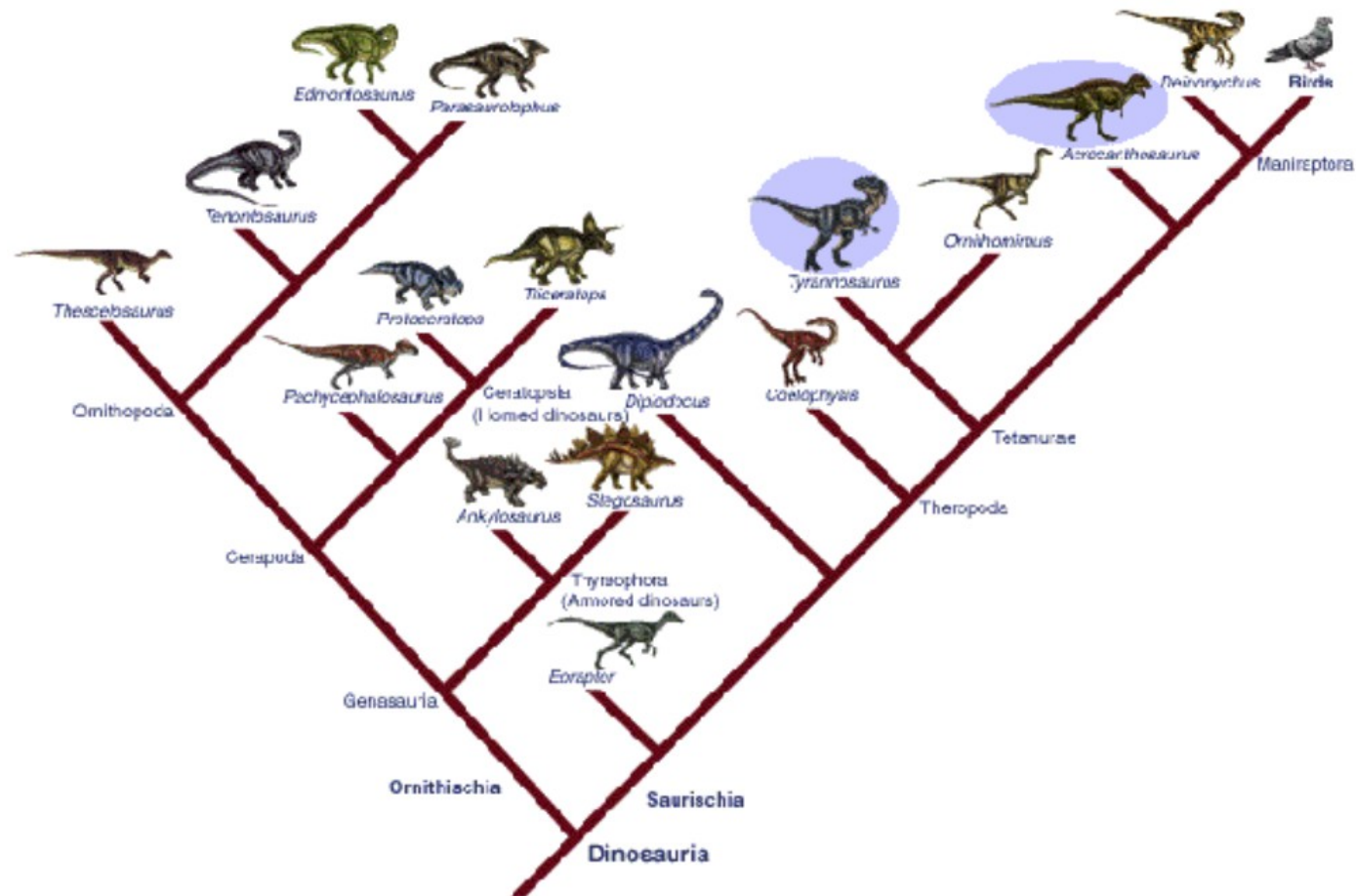


5. Ejemplo de aplicación

Árboles filogenéticos (II)

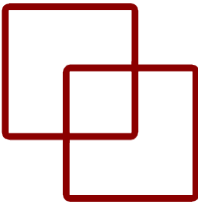


- Muy útiles para representar jerarquías de organismos a partir de su ADN.



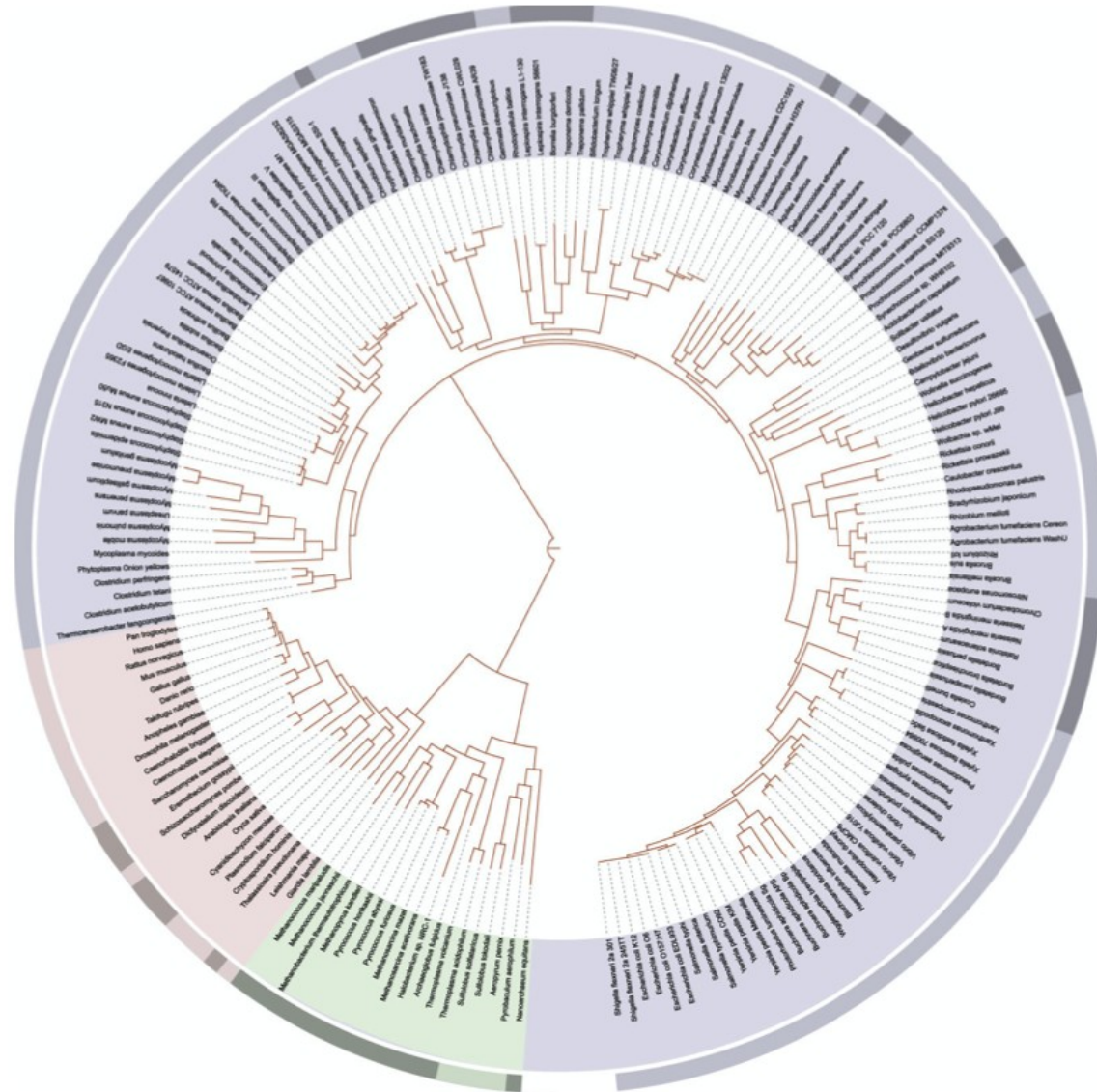
5. Ejemplo de aplicación

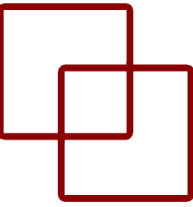
Árboles filogenéticos (III)



- Como son en realidad...

Árbol reconstruido de forma automática mediante la concatenación de 36 alineamientos de genes presentes en 196 especies



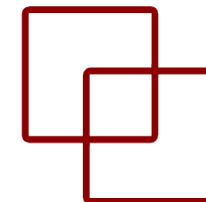


6. Scikit-learn

- Se implementa en la clase `sklearn.cluster.AgglomerativeClustering`. Los parámetros más importantes son:
 - `n_clusters`. k , es decir, el número de clusters.
 - `linkage`. Medida de distancia entre clusters. Por defecto es `'ward'`, que minimiza la varianza de los clusters fundidos.
- Los atributos de interés, disponibles una vez ejecutado el algoritmo, son:
 - `labels_`. Etiqueta de cada elemento (cluster al que pertenece).
 - `children_`. Representación del árbol resultante.



Métodos basados en densidad



1. Definición

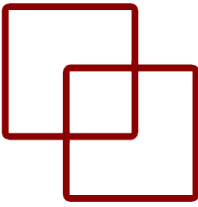
- Método basado en distancia



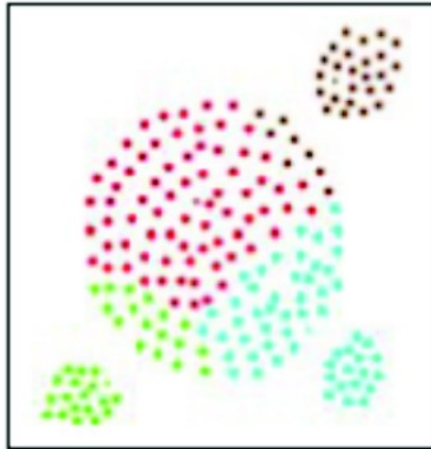
- Método basado en conectividad



1. Definición

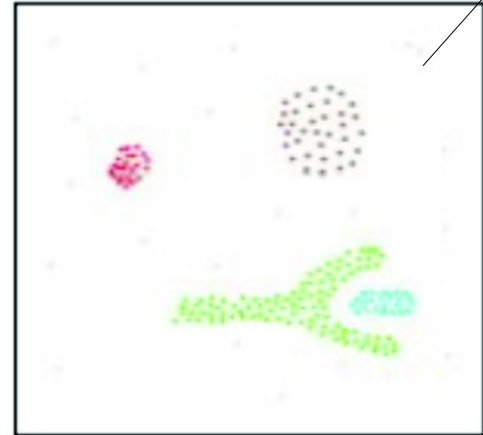
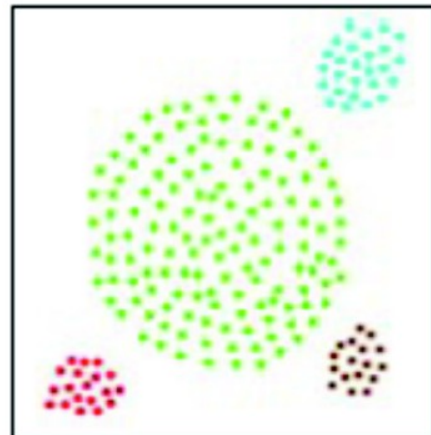


- Método basado en distancia

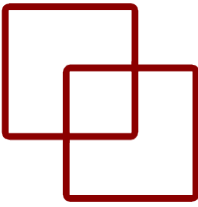


Discretiza
outliers

- Método basado en conectividad



2. Clustering basado en densidad

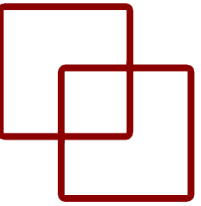


- Características principales:
 - Pueden descubrir clusters con formas arbitrarias
 - Pueden manejar ruido
 - Aprenden en una única pasada/iteración
 - Necesitan parámetros de densidad como criterio de terminación
- Algunas propuestas:
 - *DBSCAN, OPTICS, DENCLUE, CLIQUE*

3. DBSCAN

Tipos de puntos

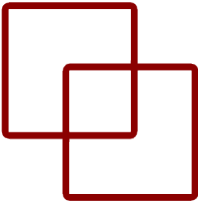
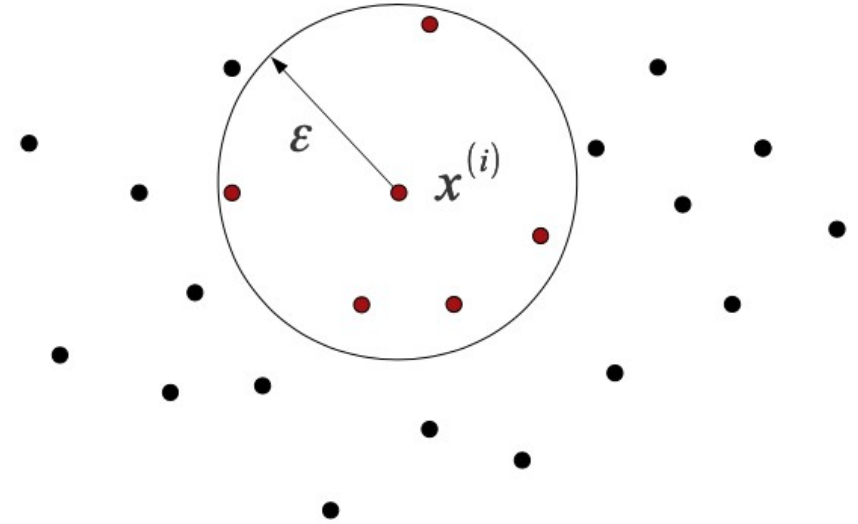
- *Density-Based Spatial Clustering of Application with Noise (DBSCAN)*



3. DBSCAN

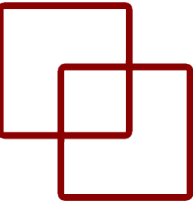
Tipos de puntos

- *Density-Based Spatial Clustering of Application with Noise*
- El conjunto de elementos comprendidos dentro de un radio ϵ del elemento $x^{(i)}$ se conoce como **ϵ -vecindario** del elemento $x^{(i)}$

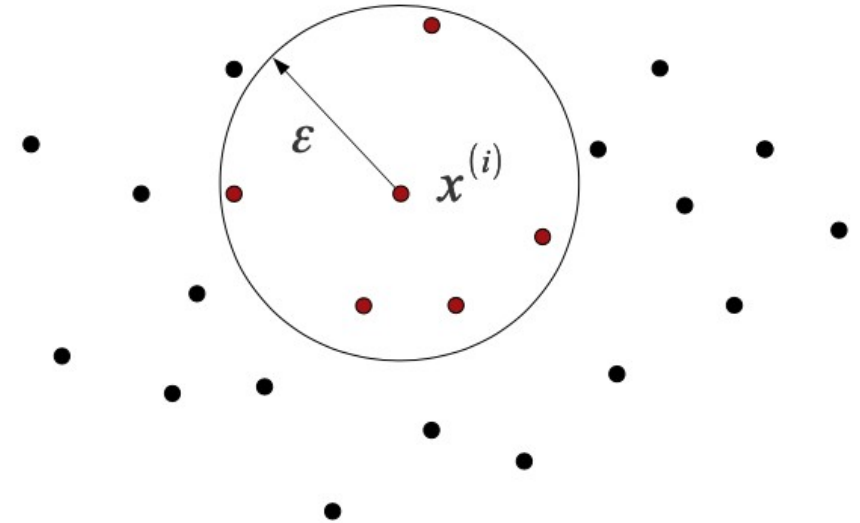


3. DBSCAN

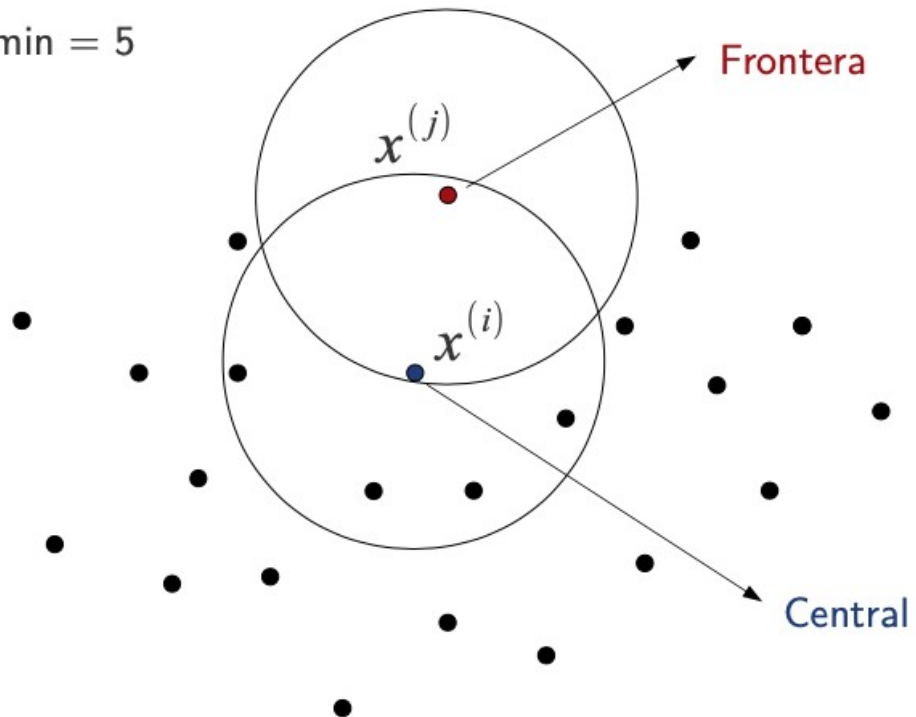
Tipos de puntos



- *Density-Based Spatial Clustering of Application with Noise*
- El conjunto de elementos comprendidos dentro de un radio ϵ del elemento $x^{(i)}$ se conoce como **ϵ -vecindario** del elemento $x^{(i)}$



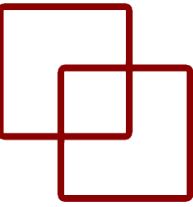
min = 5



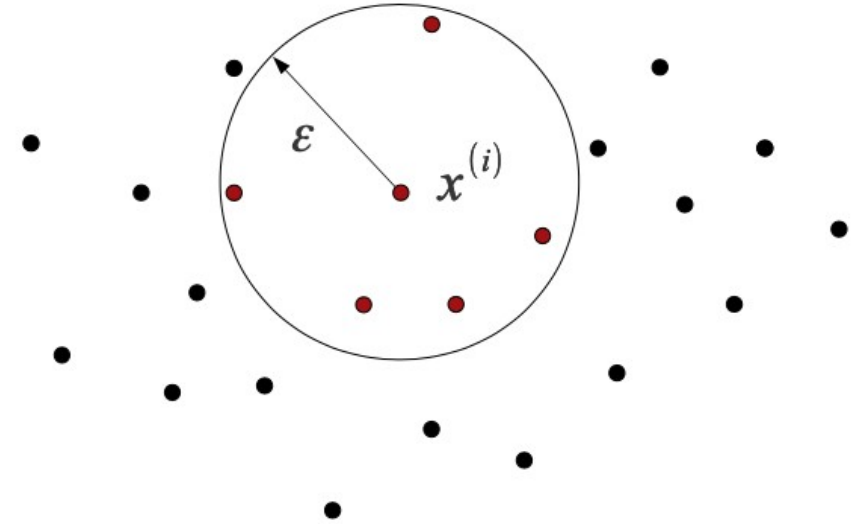
- Un **punto núcleo** es un elemento cuyo ϵ -vecindario contiene al menos otros *min* elementos.
- Un **punto frontera** tiene menos de *min* elementos en su ϵ -vecindario, pero está en el vecindario de un punto núcleo.
- El resto de puntos se conocen como **ruido**.

3. DBSCAN

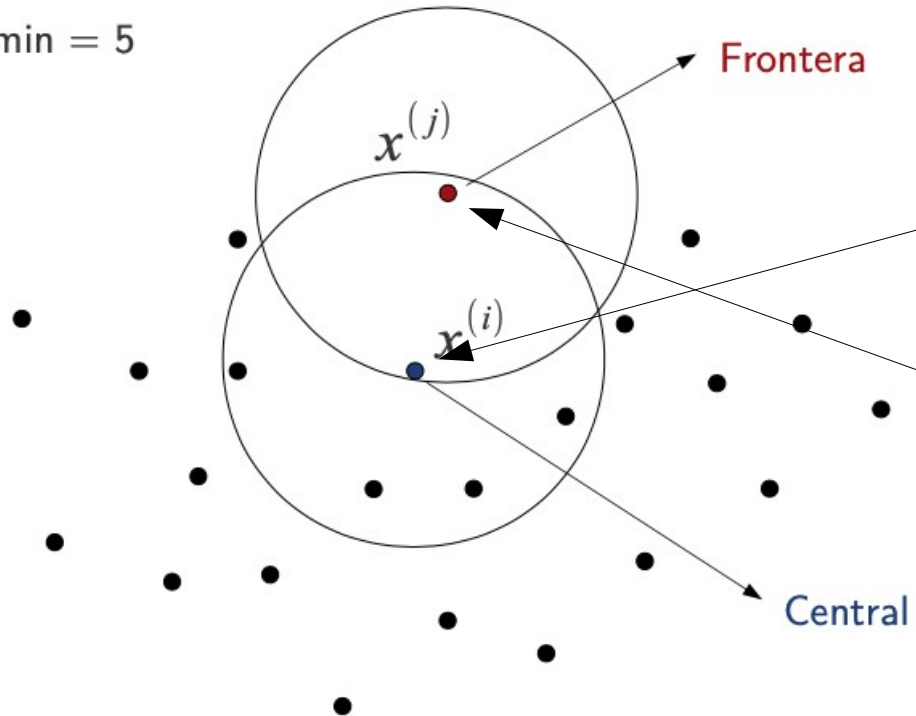
Tipos de puntos



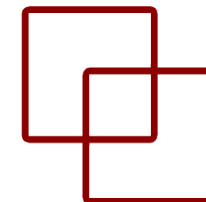
- *Density-Based Spatial Clustering of Application with Noise*
- El conjunto de elementos comprendidos dentro de un radio ϵ del elemento $x^{(i)}$ se conoce como **ϵ -vecindario** del elemento $x^{(i)}$



min = 5

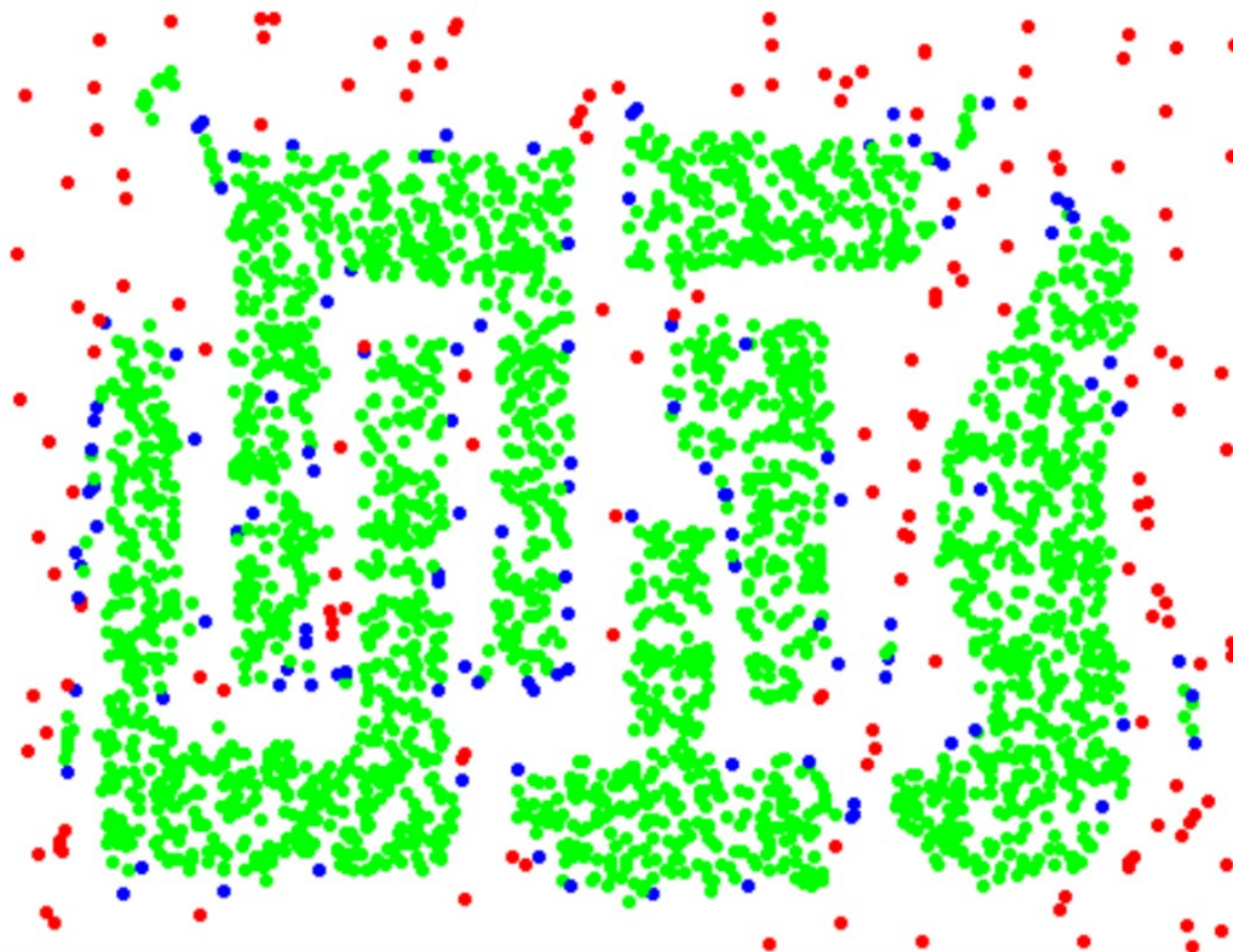


- Un **punto núcleo** es un elemento cuyo ϵ -vecindario contiene al menos otros *min* elementos.
- Un **punto frontera** tiene menos de *min* elementos en su ϵ -vecindario, pero está en el vecindario de un punto núcleo.
- El resto de puntos se conocen como **ruido**.



3. DBSCAN

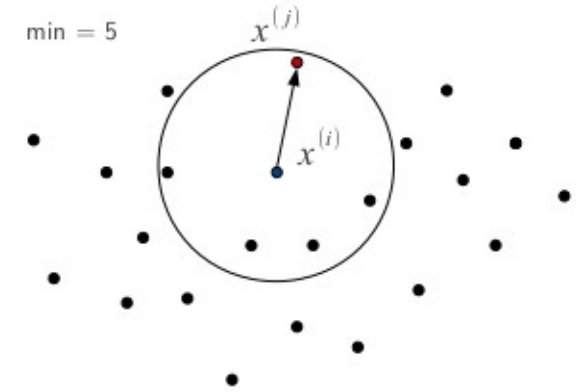
Ejemplo con puntos **núcleo**, frontera y **ruido**



3. DBSCAN

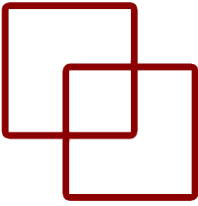
Conectividad

Un elemento $x^{(j)}$ es **directamente alcanzable por densidad** desde el elemento $x^{(i)}$ si $x^{(j)}$ está en el ε -vecindario de $x^{(i)}$ y $x^{(i)}$ es un punto núcleo. Si $x^{(j)}$ es un punto frontera entonces $x^{(i)}$ no es directamente alcanzable por densidad desde $x^{(j)}$.

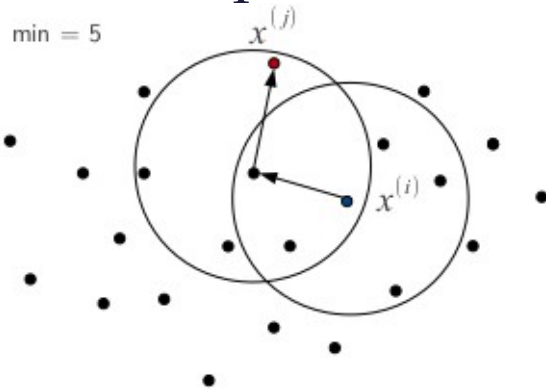
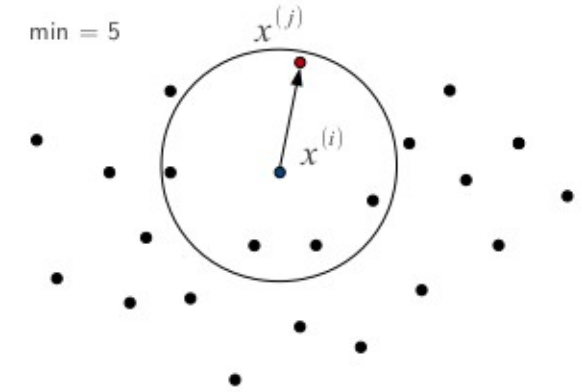


3. DBSCAN

Conectividad



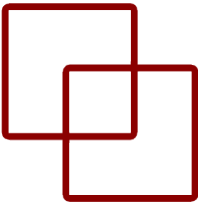
Un elemento $x^{(j)}$ es **directamente alcanzable por densidad** desde el elemento $x^{(i)}$ si $x^{(j)}$ está en el ε -vecindario de $x^{(i)}$ y $x^{(i)}$ es un punto núcleo. Si $x^{(j)}$ es un punto frontera entonces $x^{(i)}$ no es directamente alcanzable por densidad desde $x^{(j)}$.



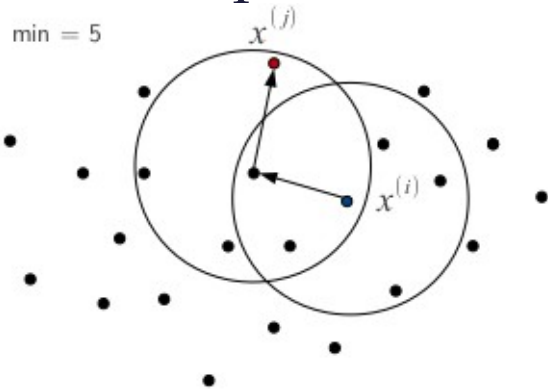
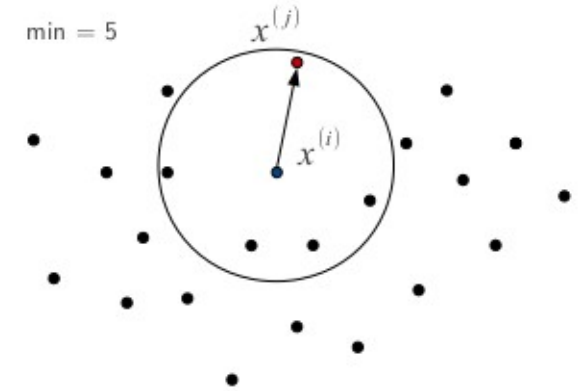
Un objeto $x^{(j)}$ es **alcanzable por densidad** desde un objeto $x^{(i)}$ si hay una cadena de objetos p_1, \dots, p_n donde $p_1 = x^{(i)}$ y $p_n = x^{(j)}$ tal que $p_k + 1$ es **directamente alcanzable por densidad** desde p_k .

3. DBSCAN

Conectividad

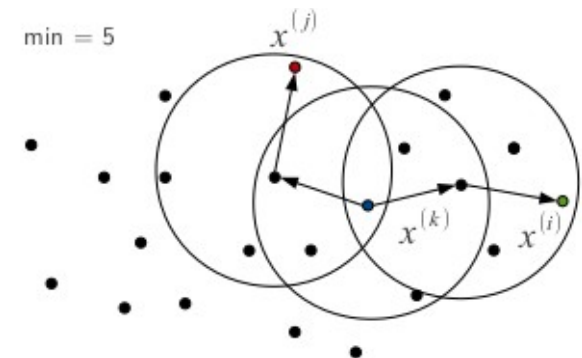


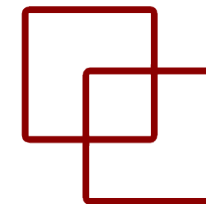
Un elemento $x^{(j)}$ es **directamente alcanzable por densidad** desde el elemento $x^{(i)}$ si $x^{(j)}$ está en el ε -vecindario de $x^{(i)}$ y $x^{(i)}$ es un punto núcleo. Si $x^{(j)}$ es un punto frontera entonces $x^{(i)}$ no es directamente alcanzable por densidad desde $x^{(j)}$.



Un objeto $x^{(j)}$ es **alcanzable por densidad** desde un objeto $x^{(i)}$ si hay una cadena de objetos p_1, \dots, p_n donde $p_1 = x^{(i)}$ y $p_n = x^{(j)}$ tal que $p_k + 1$ es **directamente alcanzable por densidad** desde p_k .

- Un objeto $x^{(i)}$ está **conectado por densidad** con $x^{(j)}$ con respecto a ε y min si hay un objeto $x^{(k)}$ tal que tanto $x^{(i)}$ como $x^{(j)}$ son alcanzables por densidad desde $x^{(k)}$.





3. DBSCAN

Algoritmo

- DBSCAN se basa dos propiedades:
 - Si $x^{(i)}$ pertenece al cluster k y $x^{(j)}$ es **alcanzable por densidad** desde $x^{(i)}$, entonces $x^{(j)}$ es también parte del cluster k .
 - Dos elementos $x^{(i)}$ y $x^{(j)}$ pertenecen al mismo cluster k , si están **conectados por densidad**.

Algoritmo DBSCAN

repetir {

Tomar un punto $x^{(i)}$ que no pertenezca a ningún cluster.

$c^{(i)} \leftarrow$ nuevo numero de cluster.

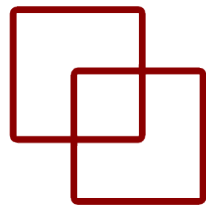
$A^{(i)} \leftarrow$ nodos alcanzables por densidad desde $x^{(i)}$

$c^{(j)} = c^{(i)}, \forall x^{(j)} \in A^{(i)}$

} hasta que no queden puntos sin asignar

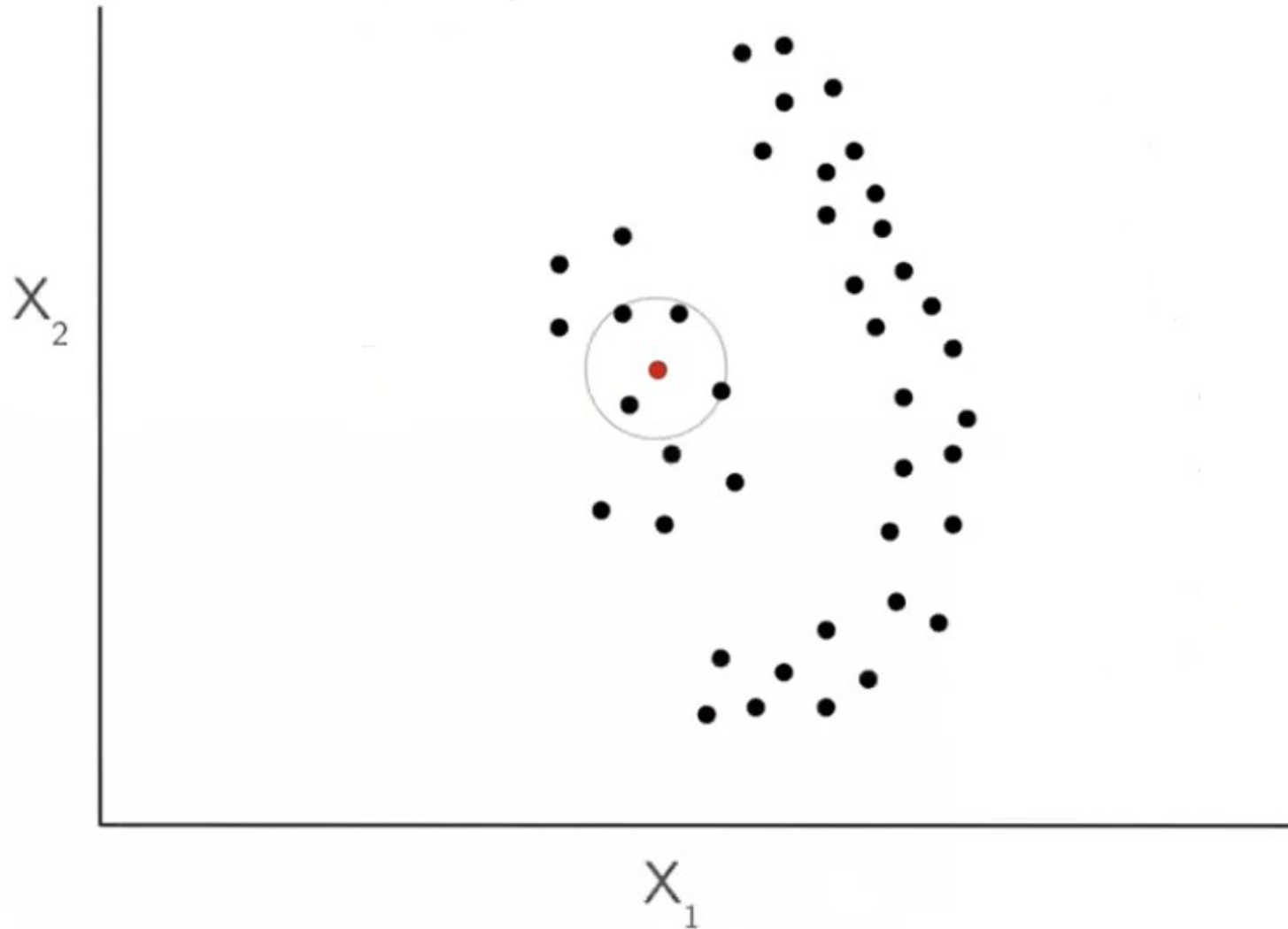
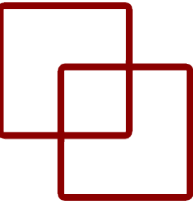
3. DBSCAN

Ejemplo DBSCAN (m=2)



3. DBSCAN

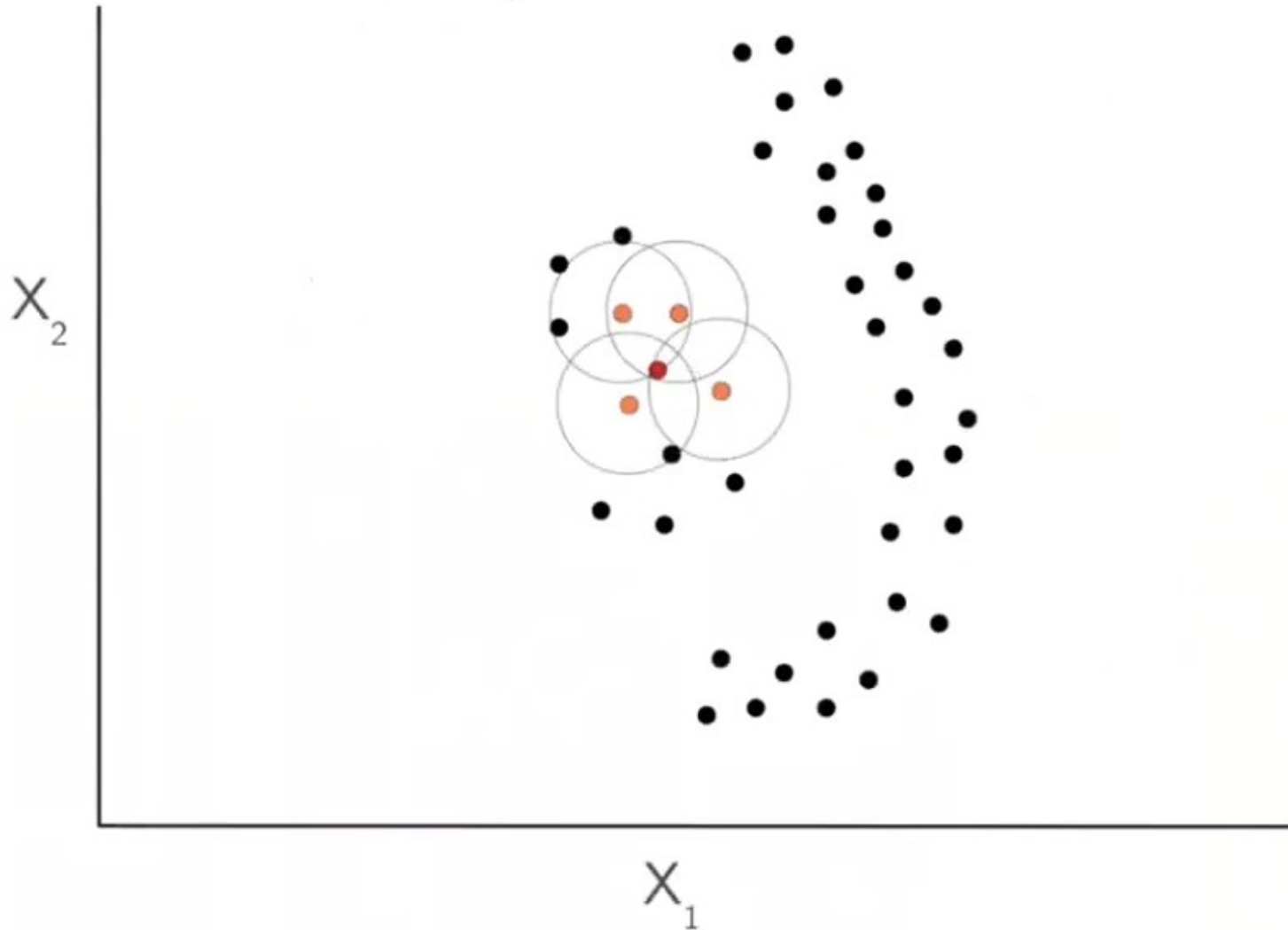
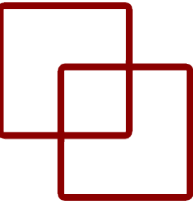
Ejemplo DBSCAN (m=2)



Definimos un punto,
hacemos su radio y
marcamos los puntos
conectados

3. DBSCAN

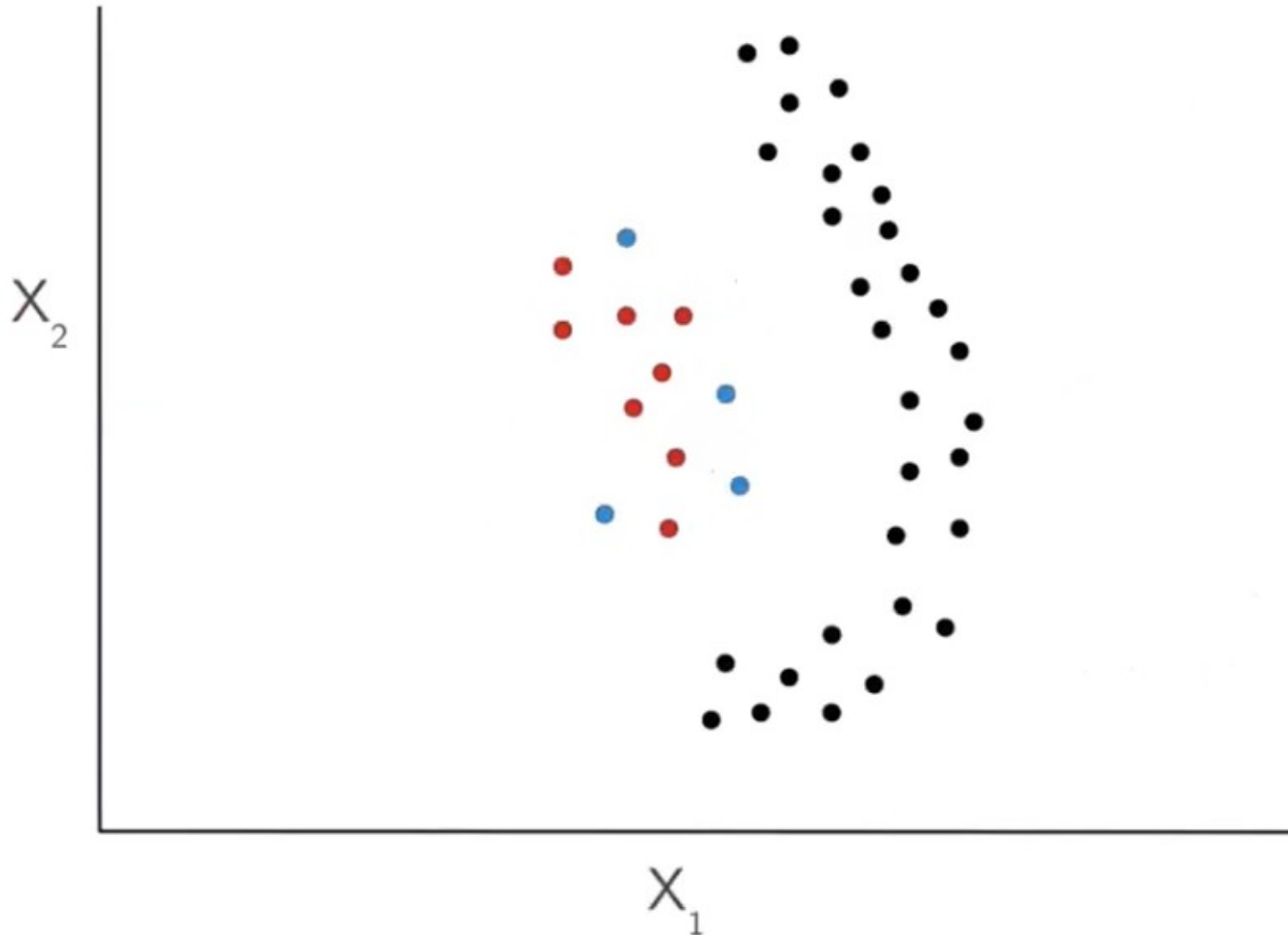
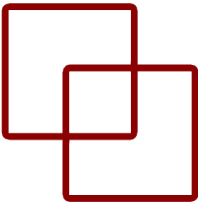
Ejemplo DBSCAN (m=2)



Se mira el vecindario
de todos ellos

3. DBSCAN

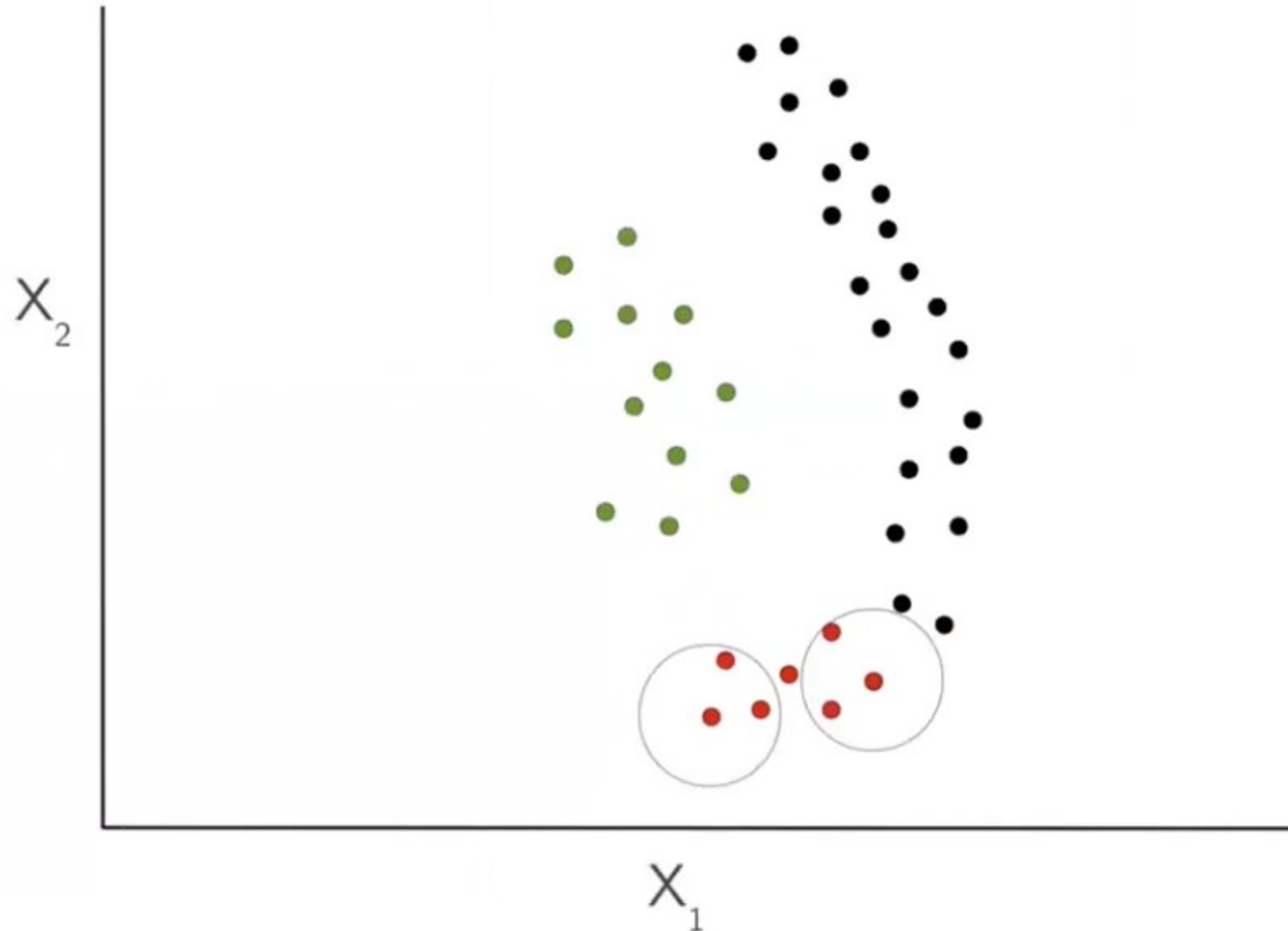
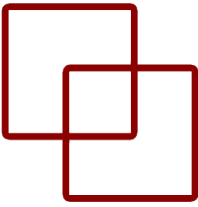
Ejemplo DBSCAN (m=2)



Al final se marcan los puntos conectados con los puntos nucleo y frontera

3. DBSCAN

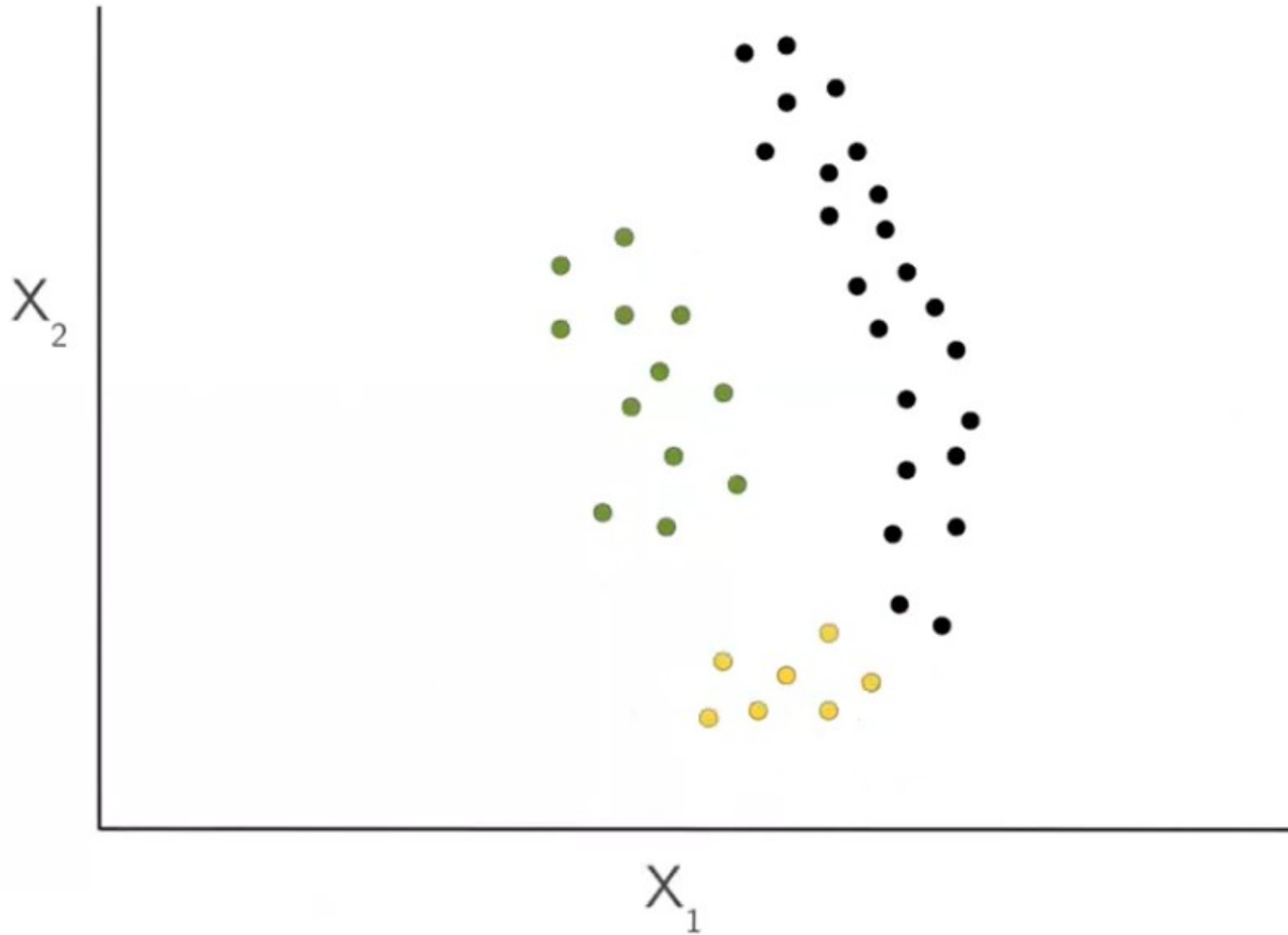
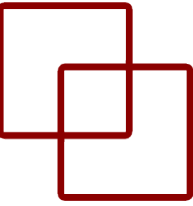
Ejemplo DBSCAN (m=2)



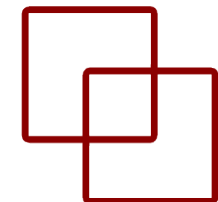
Mismo proceso con otro clúster.

3. DBSCAN

Ejemplo DBSCAN (m=2)

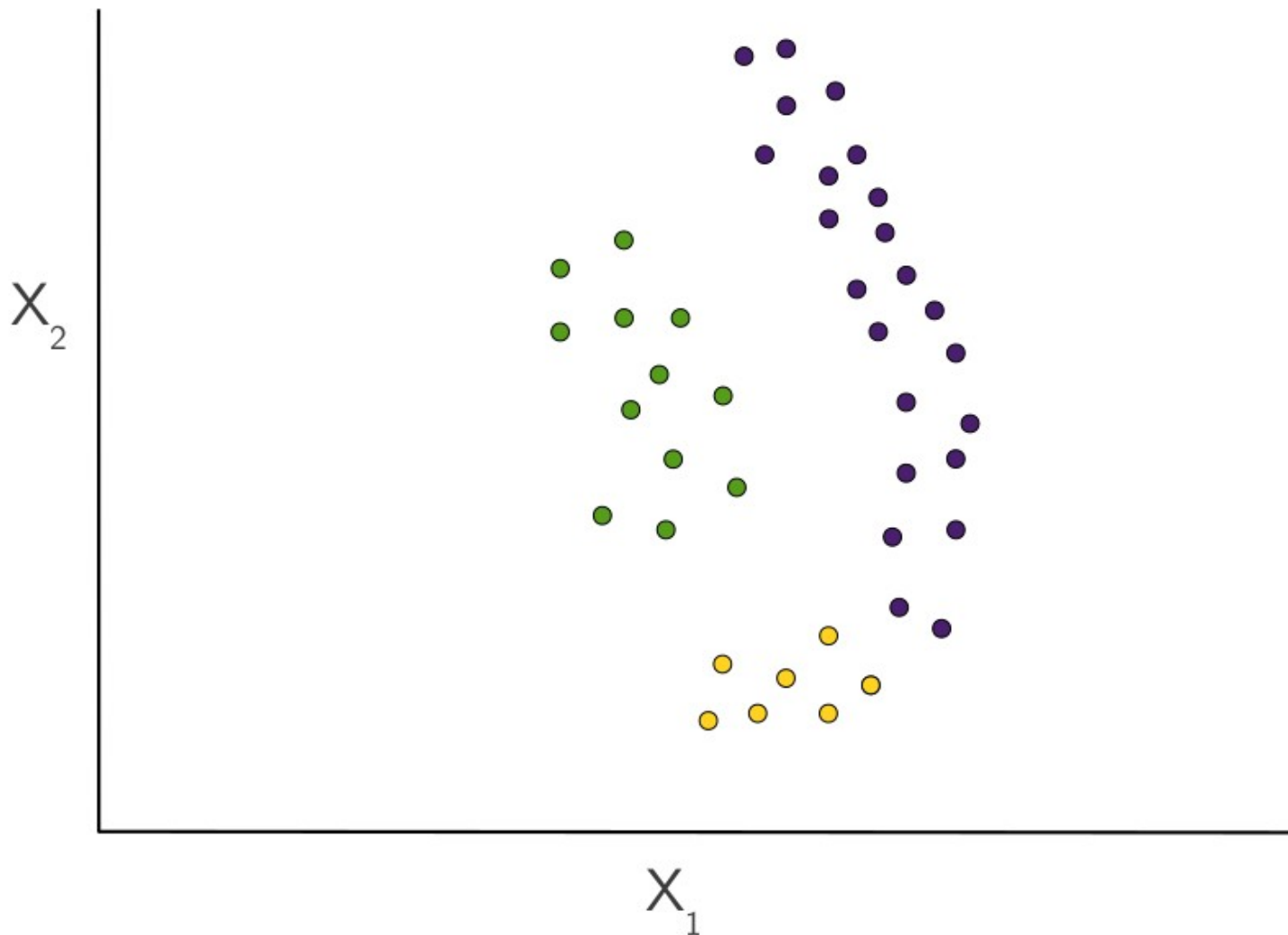


Se forma el siguiente clúster

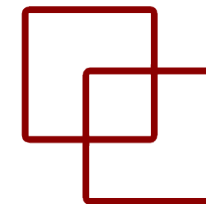


3. DBSCAN

Ejemplo DBSCAN (m=2)



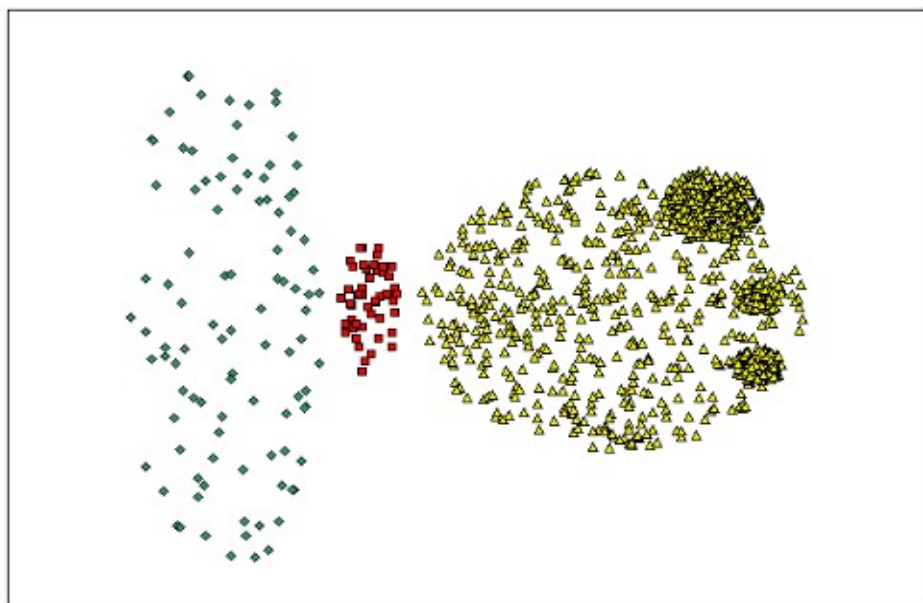
Otra vez el proceso para el último clúster



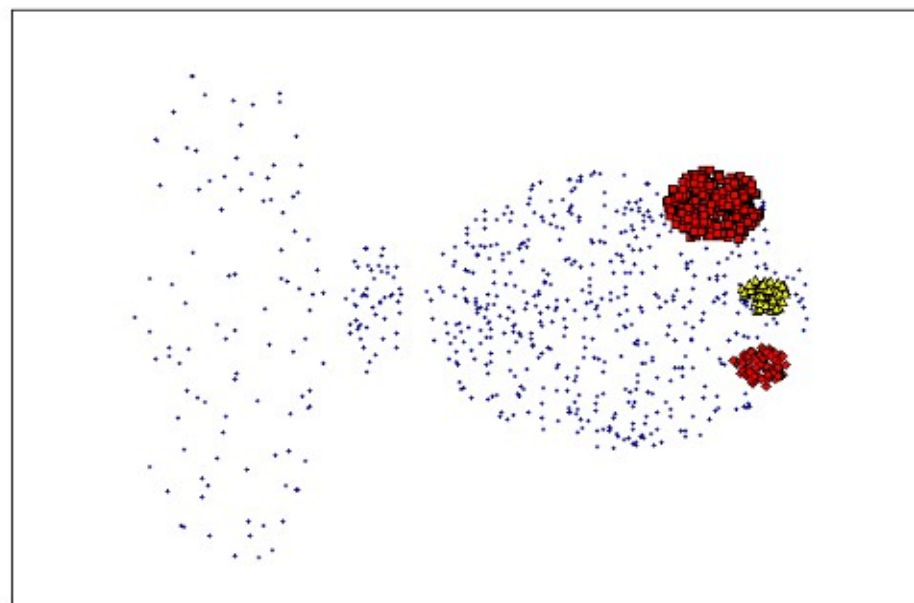
4. Limitaciones del algoritmo

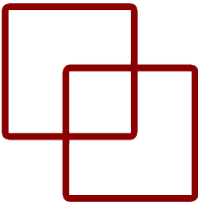
- DBSCAN no funciona bien cuando los clusters tienen diferente densidad.
 - La salida con un poco de diferencia de radio es demasiado grande

$min = 4, \epsilon = 9.75$



$min = 4, \epsilon = 9.92$





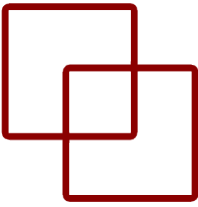
5. DBSCAN: Implementación

scikit-learn

- Se implementa en la clase `sklearn.cluster.DBSCAN`.
- Los parámetros más importantes son:
 - *eps*. La distancia máxima entre dos puntos de un cluster.
 - *min_samples*. Número mínimo de puntos en el vecindario de otro para que éste último sea considerado un punto núcleo.
- Los atributos de interés, disponibles una vez ejecutado el algoritmo, son:
 - *labels_*. Etiqueta de cada elemento (cluster al que pertenece). A los puntos considerados ruido se les asigna el valor -1 .
 - *core_sample_indices_*. Índices de los elementos considerados núcleo.

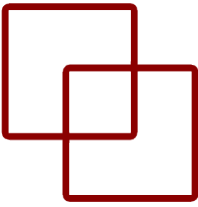


Número óptimo de clústers



1. Introducción

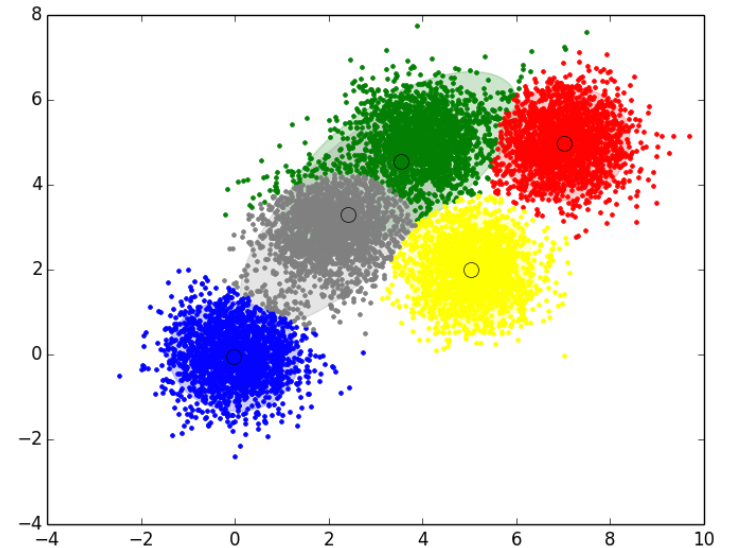
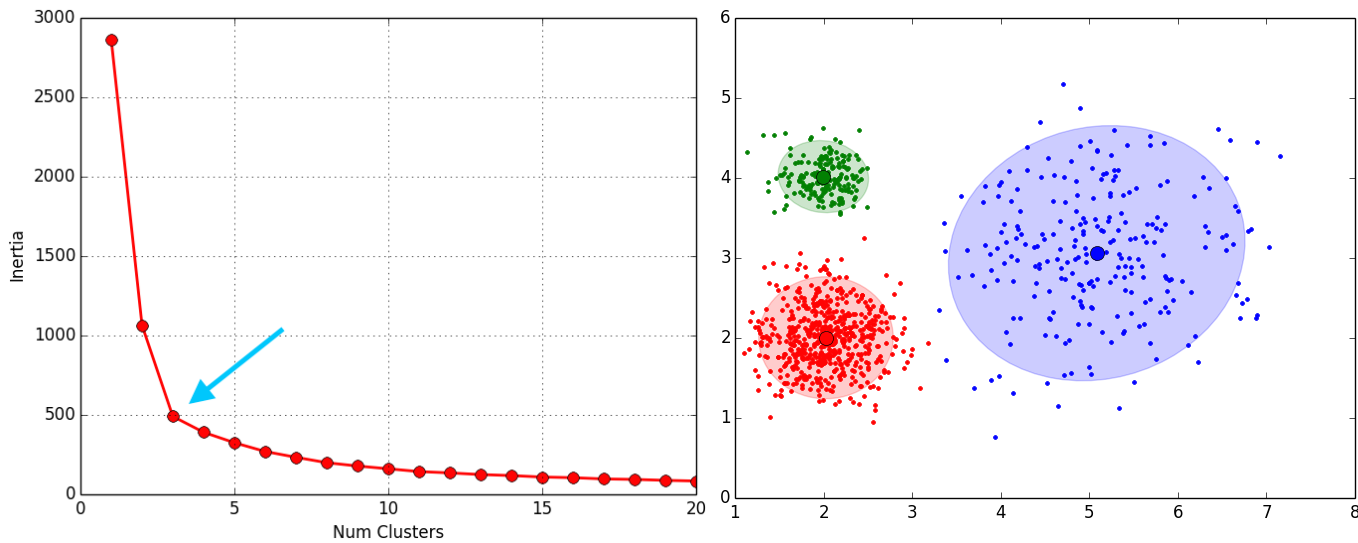
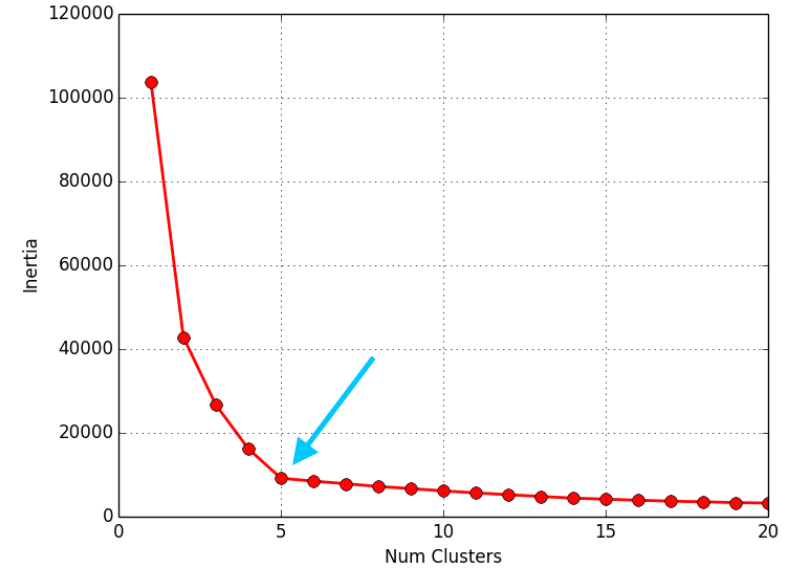
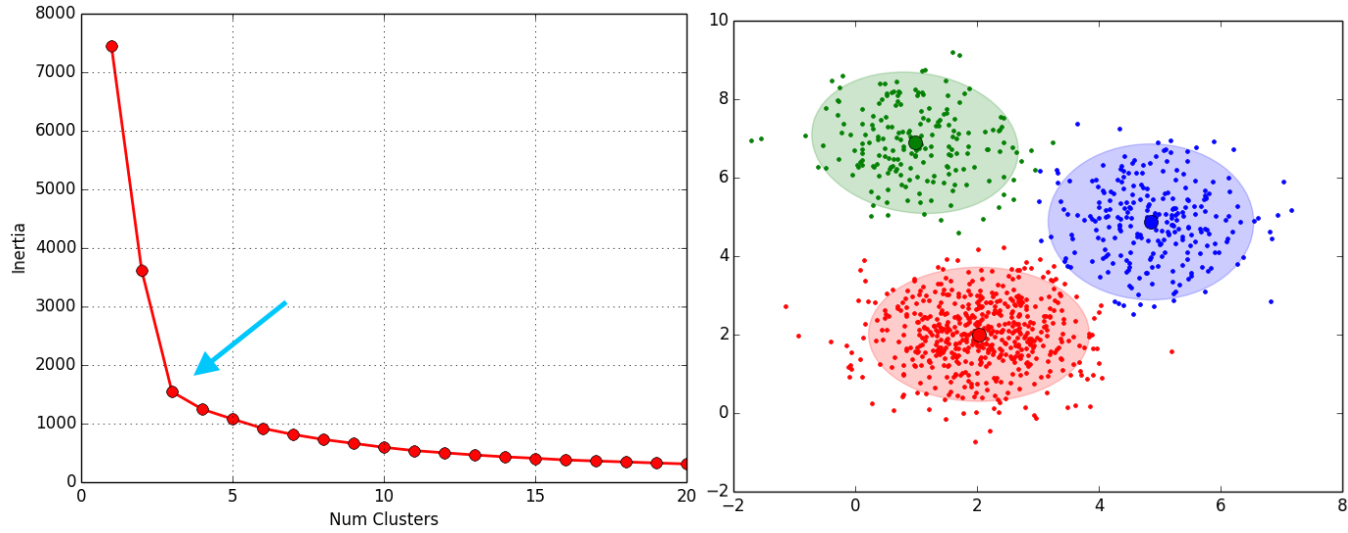
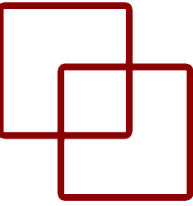
- Problema: elección del número de Clusters.
- Una mala elección da a lugar a:
 - Agrupaciones de datos muy heterogéneos (pocos Clusters);
 - O datos, que siendo muy similares unos a otros se agrupan en Clusters diferentes (muchos Clusters).
- Técnicas: El método del codo (*elbow*), el criterio de Calinsky, el Affinity Propagation (AP), el GAP (también con su versión estadística), Dendrogramas, etc.

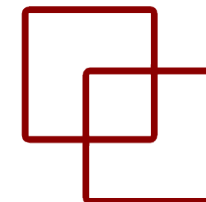


2. Método del codo (I)

- Utiliza los valores de la inercia obtenidos tras aplicar el algoritmo a diferentes números de Clusters (desde 1 a N Clusters),
 - Siendo la inercia la suma de las distancias al cuadrado de cada objeto del Cluster a su centroide.
- Se representa una gráfica lineal la inercia respecto del número de Clusters.
- El punto en el que se observa ese cambio brusco (semejante a un codo) en la inercia nos dirá el número óptimo de Clusters

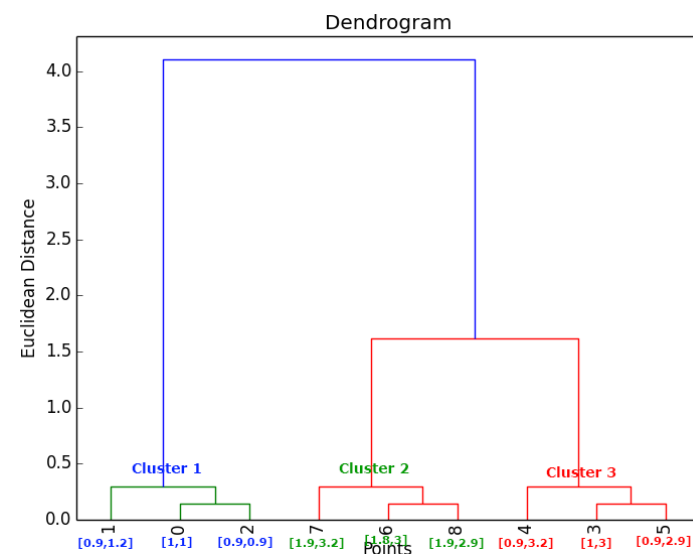
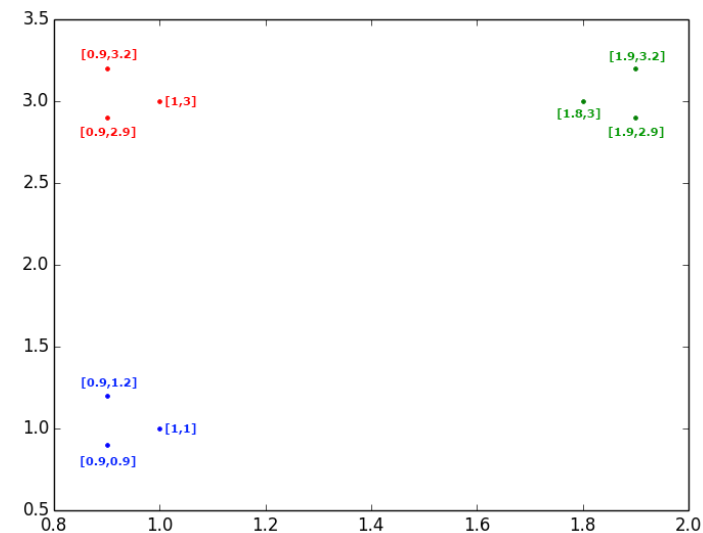
2. Método del codo (II)



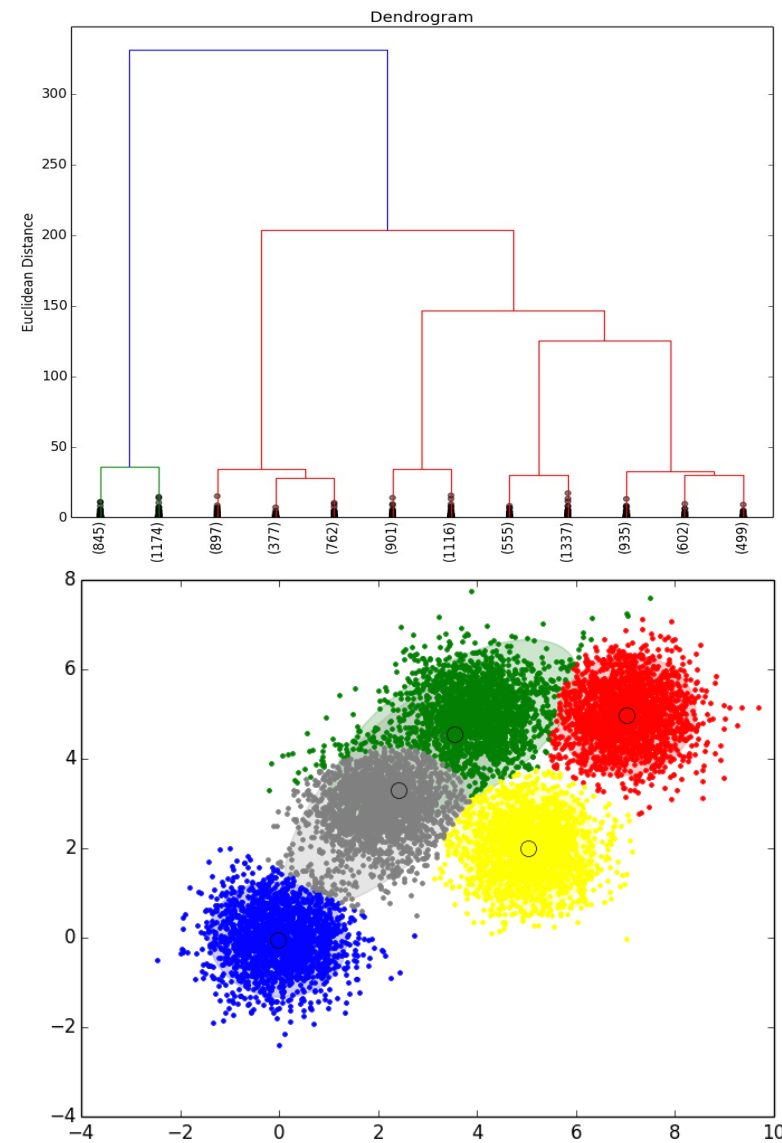
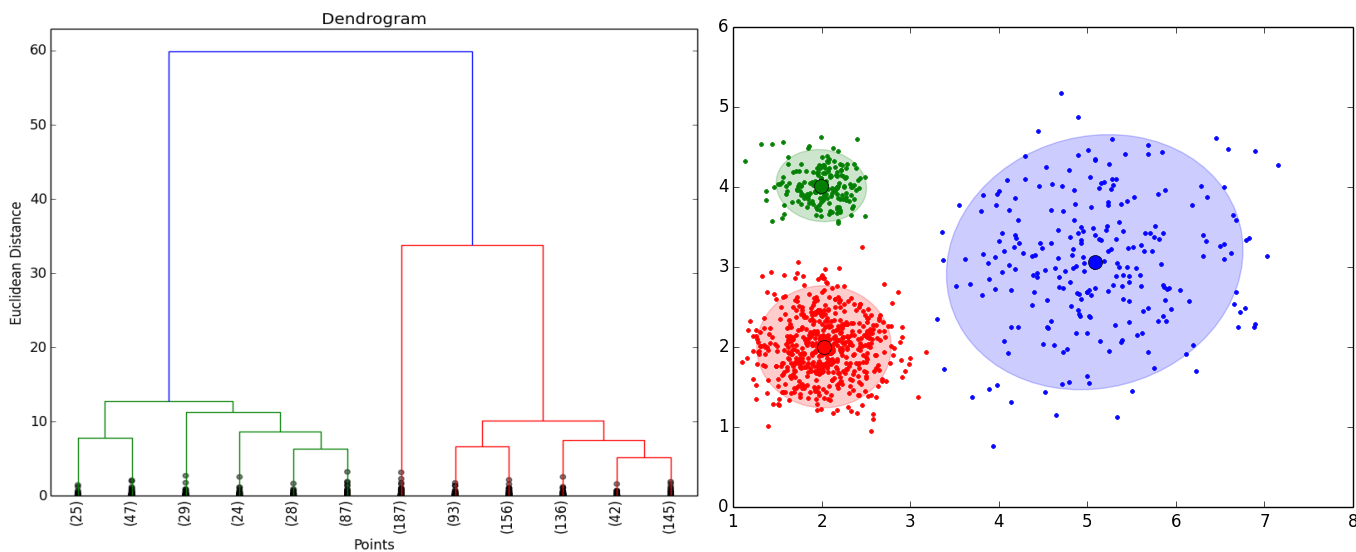
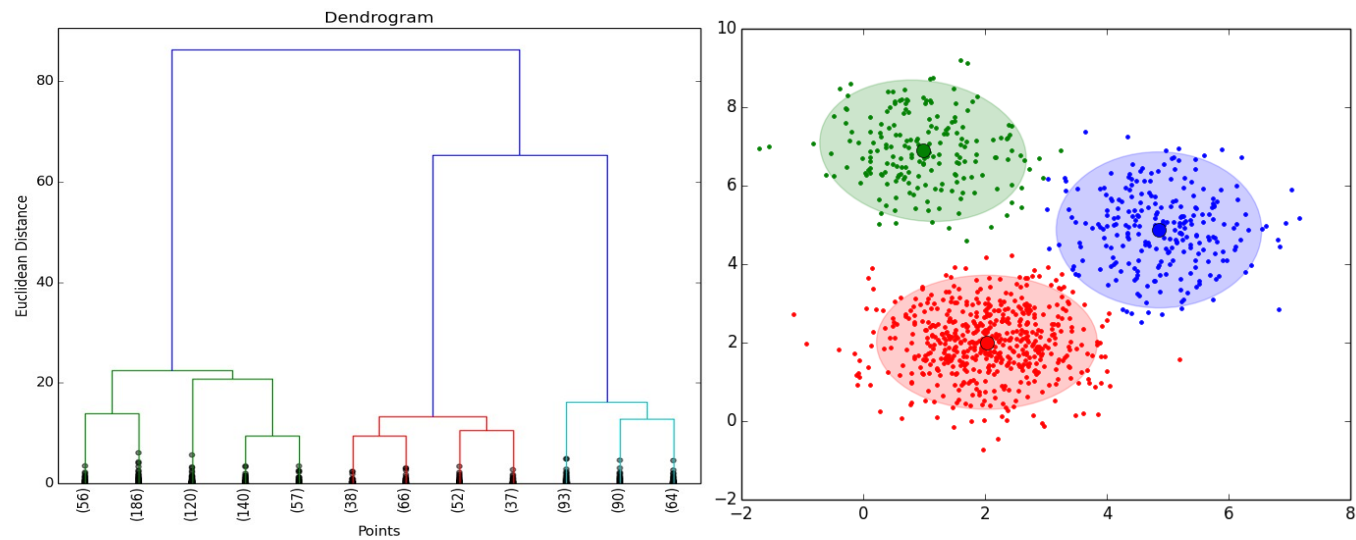
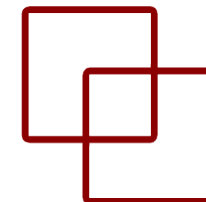


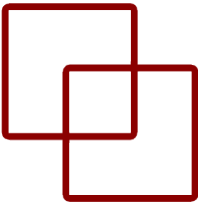
3. Dendograma (I)

- Es un tipo de representación gráfica en forma de árbol que organiza y agrupa los datos en subcategorías según su similitud:
 - Sobre una medida de distancia, que normalmente es distancia Euclídea.
- En resumen lo que se hace es:
 - Calcular la distancia entre todos los puntos,
 - Cogemos la menor distancia,
 - Agrupamos esos dos puntos, y
 - Volvemos a calcular la distancia entre todos los puntos o entre todos los grupos ya formados y puntos.



3. Dendrograma (II)

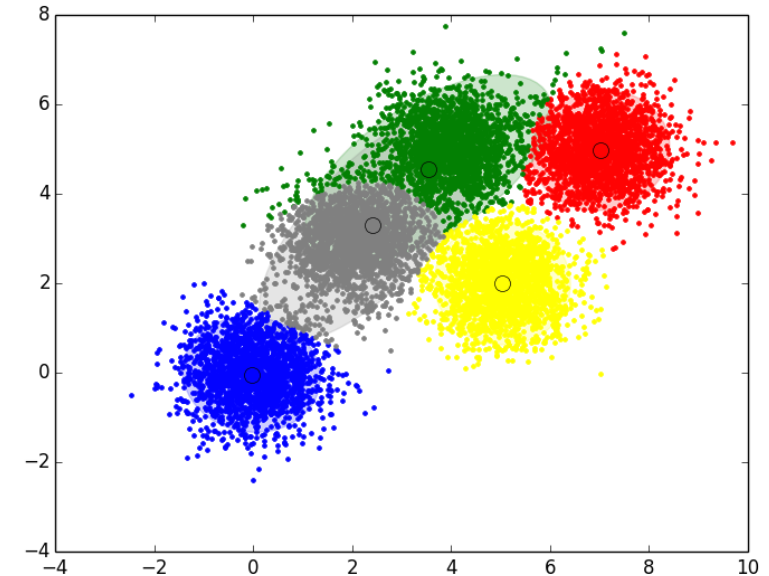
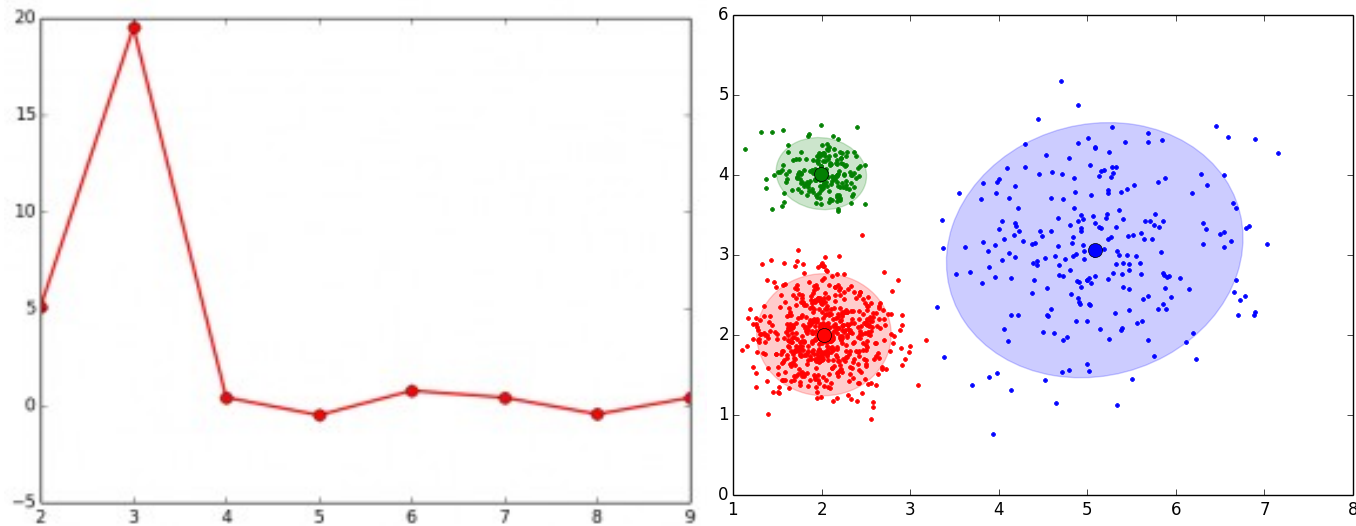
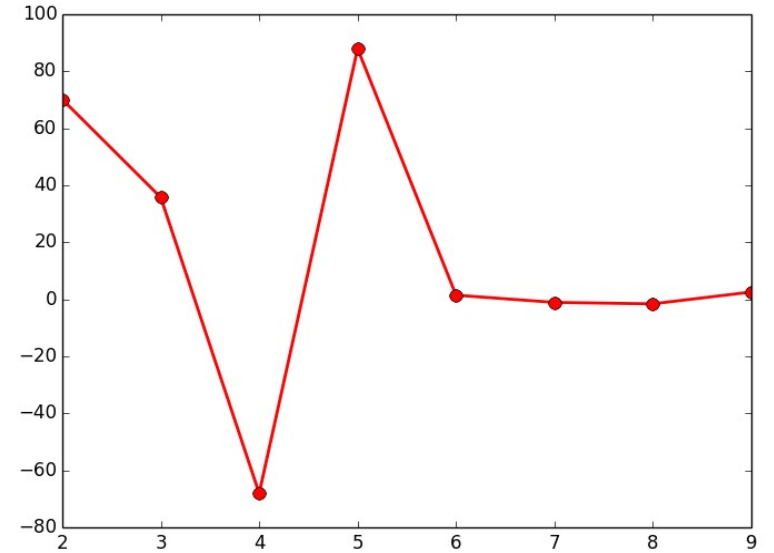
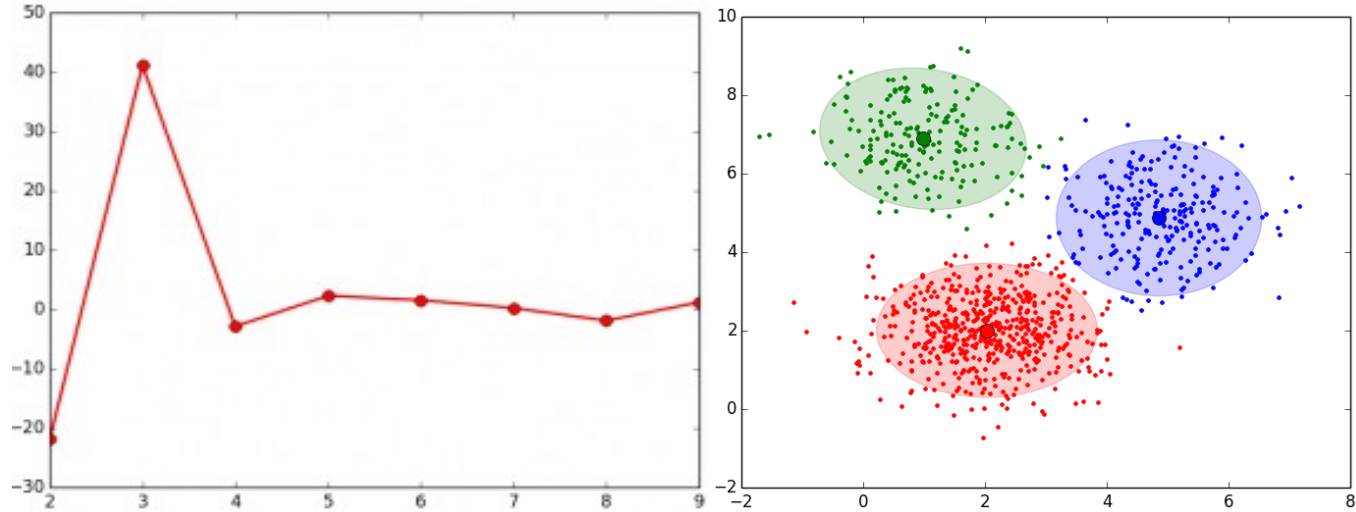
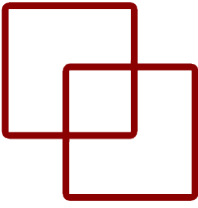


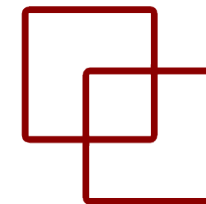


4. GAP (I)

- GAP (brecha) es similar al método del codo
- Su finalidad es la de encontrar la mayor diferencia o distancia que hay entre los diferentes grupos de objetos que vamos formando para representarlos en un dendrograma.
- Para ello, va escogiendo las distancias que hay de cada uno de los enlaces que forman el dendrograma y vemos cual es la mayor diferencia que hay entre cada uno de estos enlaces.

3. Dendrograma (II)



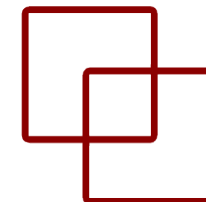


5. Silhouette (I)

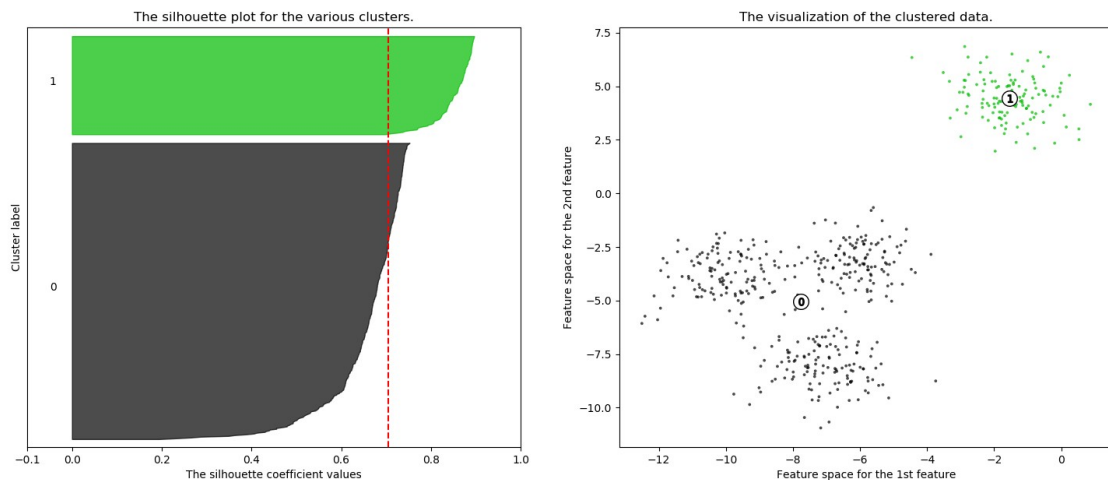
- Estudia la distancia de separación entre los grupos resultantes.
- El diagrama de Silhouette muestra una medida de cuán cerca está cada punto de un grupo a los puntos en los grupos vecinos.
- Esta medida tiene un rango de $[-1, 1]$.
 - Cerca de +1 indican que la muestra está muy lejos de los grupos vecinos.
 - Un valor de 0 indica que la muestra está dentro o muy cerca del límite de decisión entre dos grupos vecinos.
 - Los valores negativos indican que esas muestras podrían haberse asignado al grupo incorrecto.

```
For n_clusters = 2 The average silhouette_score is : 0.7049787496083262
For n_clusters = 3 The average silhouette_score is : 0.5882004012129721
For n_clusters = 4 The average silhouette_score is : 0.6505186632729437
For n_clusters = 5 The average silhouette_score is : 0.5745566973301872
For n_clusters = 6 The average silhouette_score is : 0.43902711183132426
```

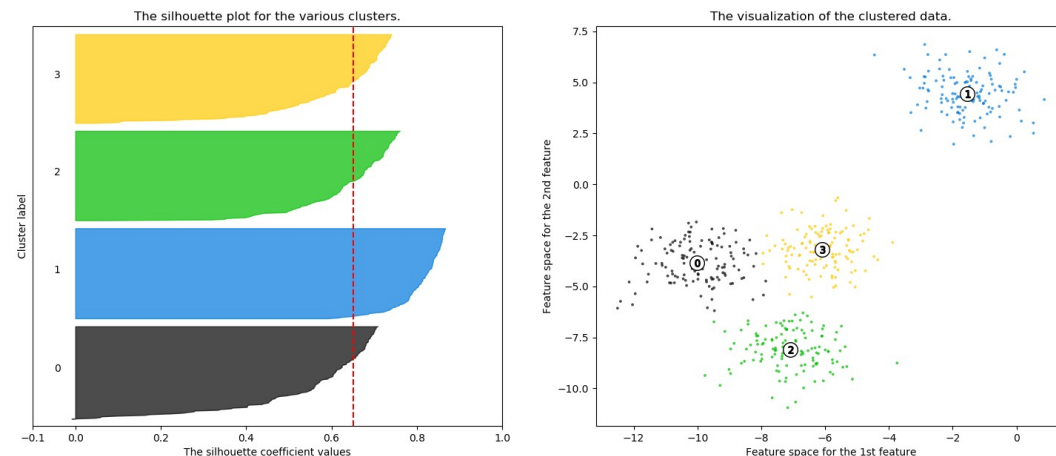
5. Silhouette (II)



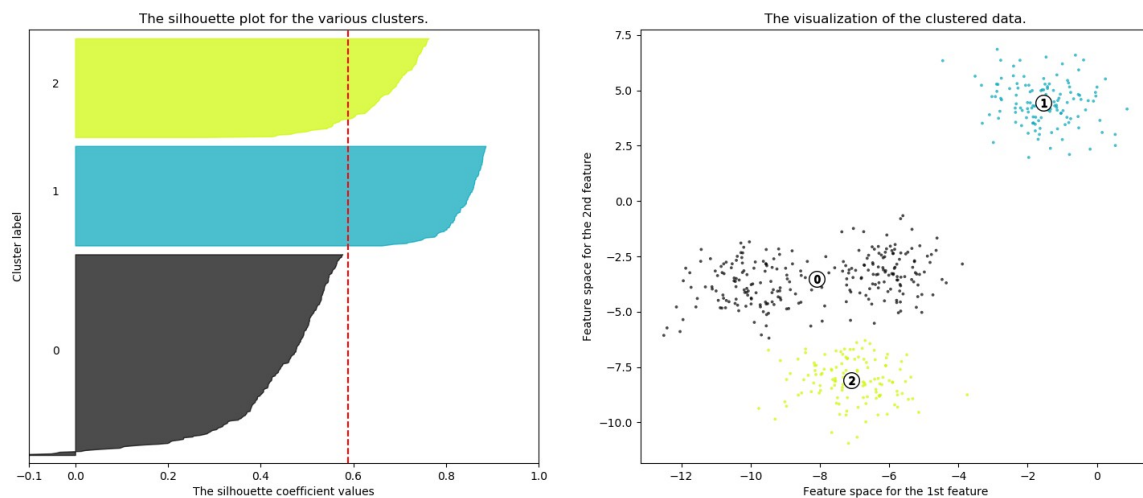
Silhouette analysis for KMeans clustering on sample data with $n_clusters = 2$



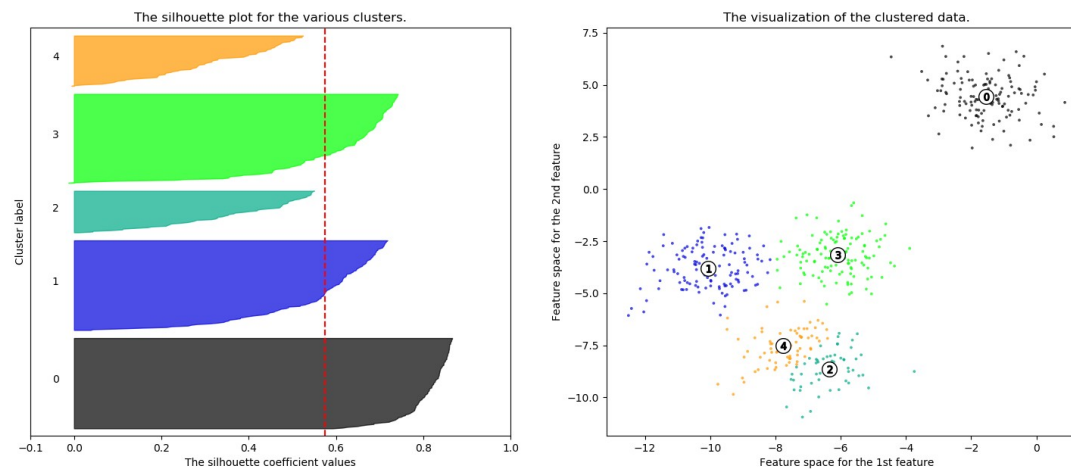
Silhouette analysis for KMeans clustering on sample data with $n_clusters = 4$



Silhouette analysis for KMeans clustering on sample data with $n_clusters = 3$



Silhouette analysis for KMeans clustering on sample data with $n_clusters = 5$



¡GRACIAS!



Dr. Manuel Castillo-Cara

www.manuelcastillo.eu

**Departamento de Inteligencia Artificial
Escuela Técnica Superior de Ingeniería Informática
Universidad Nacional de Educación a Distancia (UNED)**