

# Clustering

## Introducción y Algoritmos

# Recordatorio: aprendizaje supervisado

- Conjunto de entrenamiento  $D_n = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$
- $X = \{x_1, x_2, \dots, x_n\} \in X$ : atributos o *features*
- $Y = \{y_1, y_2, \dots, y_n\} \in Y$ : etiquetas o *labels*
- Sabemos que  $(X, Y) \sim P$ , pero  $P$  es desconocida
- Objetivo: construir una función  $f: X \rightarrow Y$ , que se parezca a  $P$

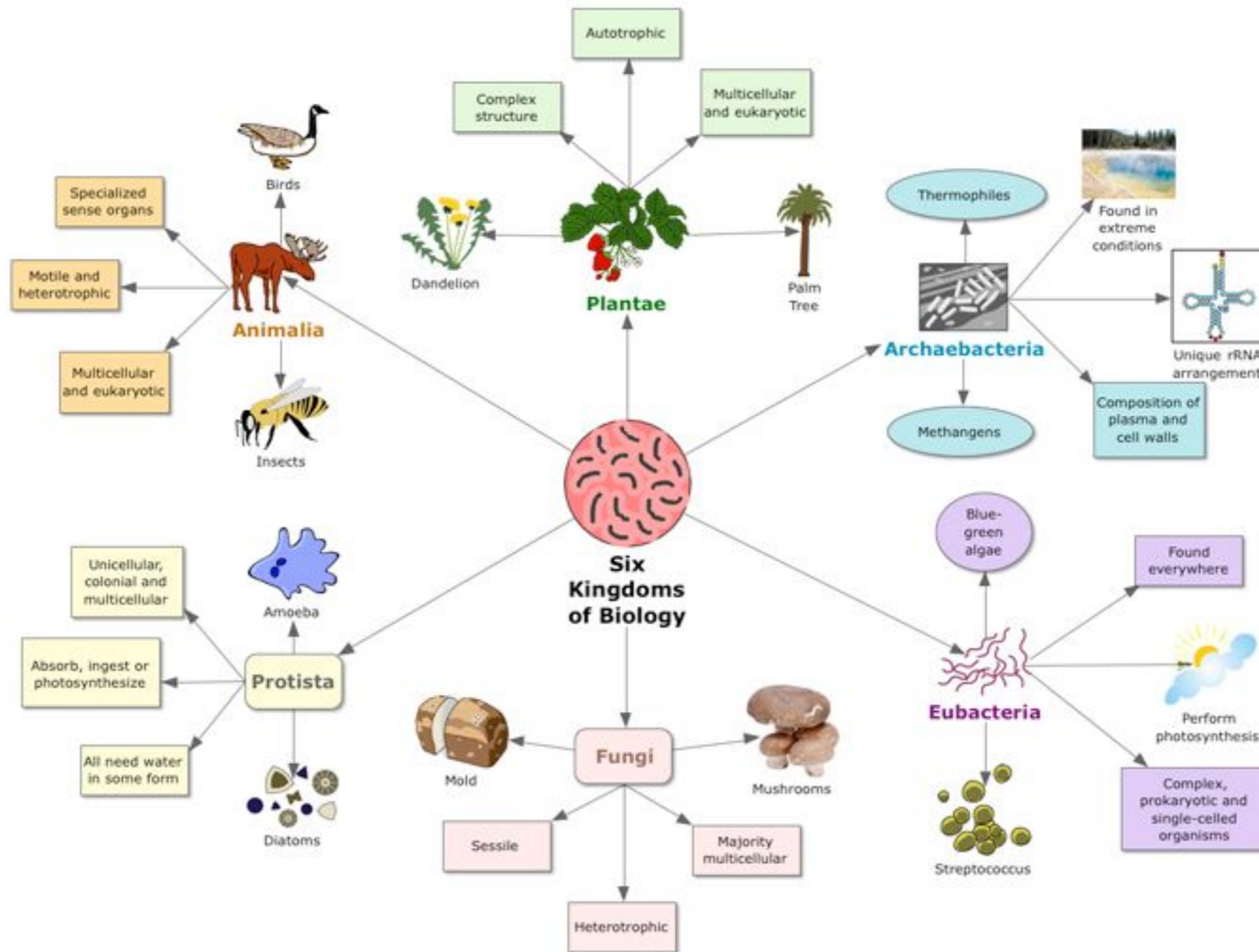
# ¿Qué pasa cuando no tenemos etiquetas?

- Ahora tenemos sólo  $D_n = \{x_1, x_2, \dots x_n\}$
- Ya no tenemos Y
- Todos los métodos que hemos estudiado hasta ahora dependen de una etiqueta, de algo a lo que nos queremos “parecer”
- ...pero aún podemos hacer algo :)
- **Aprendizaje no supervisado**

# Aprendizaje no supervisado

- Conseguir etiquetas suele ser un proceso costoso (ya sea en dinero y/o tiempo)
- A veces, ni siquiera se sabe cómo etiquetarlos (¿cuántas etiquetas hay?)
- Por otro lado, datos no etiquetados son más fáciles de conseguir
- Aún así, sobre los datos no etiquetados podemos aprender (o descubrir) cosas
- Una pregunta interesante es si podemos encontrar subgrupos dentro de los datos -> clustering

# Inspiración histórica (taxonomía)



# ¿Qué es el clustering?

- Conjunto de técnicas para encontrar subgrupos de objetos tal que los objetos en un grupo sean similares (o relacionados) entre sí y que sean diferentes (o no relacionados) a los objetos en otros grupos.
- ¿Qué significa que sean *diferentes* o *parecidos*?

# ¿Cuándo y para qué usar clustering?

- Queremos hacer clustering
  - Para entender los datos
  - Para ayudar en otra tarea (utilidad)

# Clustering para entender los datos

## CLINICAL CANCER RESEARCH

ABOUT ▾ ARTICLES ▾ FOR AUTHORS ▾ ALERTS NEWS CANCER HALLMARKS WEBINARS

**Volume 10, Issue 18**  
15 September 2004



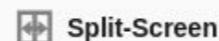
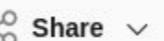
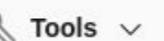
RESEARCH ARTICLES | SEPTEMBER 24 2004

### Hierarchical Clustering Analysis of Tissue Microarray Immunostaining Data Identifies Prognostically Significant Groups of Breast Carcinoma FREE

Nikita A. Makretsov; David G. Huntsman; Torsten O. Nielsen; Erika Yorida; Michael Peacock; Maggie C. U. Cheang; Sandra E. Dunn; Malcolm Hayes; Matt van de Rijn; Chris Bajdik; C. Blake Gilks

 Check for updates

 Author & Article Information  
*Clin Cancer Res* (2004) 10 (18): 6143–6151.  
<https://doi.org/10.1158/1078-0432.CCR-04-0429> Article history

 Split-Screen  Views ▾  PDF  Share ▾  Tools ▾  Versions ▾

### Abstract

Prognostically relevant cluster groups, based on gene expression profiles, have been recently identified for breast cancers, lung cancers, and lymphoma. Our aim was to determine whether hierarchical clustering analysis of multiple immunomarkers (protein expression profiles) improves prognostication in patients with invasive breast cancer. A cohort of 438 sequential cases of invasive breast cancer with median follow-up of 15.4 years was selected for tissue microarray construction. A total of 31 biomarkers were tested by immunohistochemistry on these tissue arrays. The prognostic significance of individual markers was assessed by using Kaplan-Meier survival estimates and log-rank tests. Seventeen of 31 markers showed prognostic significance in univariate analysis ( $P \leq 0.05$ ) and 4 markers showed a trend toward significance ( $P \leq 0.2$ ). Unsupervised hierarchical clustering analysis was done by using these 21 immunomarkers, and this resulted in identification of three cluster groups with significant differences in clinical outcome.  $\chi^2$  analysis showed that expression of 11 markers significantly correlated with membership in one of the three cluster groups. Unsupervised

< Previous Article Next Article >

#### Article Contents

- Abstract
- INTRODUCTION
- MATERIALS AND METHODS
- RESULTS
- DISCUSSION
- Acknowledgments
- References

# Clustering para entender los datos

 Computers & Operations Research  
Volume 29, Issue 11, September 2002, Pages 1475-1493 

## Integration of self-organizing feature map and K-means algorithm for market segmentation

R.J. Kuo <sup>a</sup>  , L.M. Ho <sup>b</sup>, C.M. Hu <sup>c</sup>

Show more ▾

+ Add to Mendeley  Share  Cite

[https://doi.org/10.1016/S0305-0548\(01\)00043-0](https://doi.org/10.1016/S0305-0548(01)00043-0) [Get rights and content](#)

---

### Abstract

Cluster analysis is a common tool for market segmentation. Conventional research usually employs the multivariate analysis procedures. In recent years, due to their high performance in engineering, artificial neural networks have also been applied in the area of management. Thus, this study aims to compare three clustering methods: (1) the conventional two-stage method, (2) the self-organizing feature maps and (3) our proposed two-stage method, via both simulated and real-world data. The proposed two-stage method is a combination of the self-organizing feature maps and the K-means method. The simulation results indicate that the proposed scheme is slightly better than the conventional two-stage method with respect to the rate of misclassification, and the real-world data on the basis of Wilk's Lambda and discriminant analysis.

# Clustering para entender los datos

Conferences > 2016 International Conference...



## Document clustering: TF-IDF approach

Publisher: IEEE

Cite This



Prafulla Bafna ; Dhanya Pramod ; Anagha Vaidya All Authors

117

Cites in  
Papers

4128

Full  
Text Views



### Abstract

### Document Sections

I. Introduction

II. Multistep Algorithm

III. Experimental Set Up

IV. Conclusion

### Abstract:

Recent advances in computer and technology resulted into ever increasing set of documents. The need is to classify the set of documents according to the type. Laying related documents together is expedient for decision making. Researchers who perform interdisciplinary research acquire repositories on different topics. Classifying the repositories according to the topic is a real need to analyze the research papers. Experiments are tried on different real and artificial datasets such as NEWS 20, Reuters, emails, research papers on different topics. Term Frequency-Inverse Document Frequency algorithm is used along with fuzzy K-means and hierarchical algorithm. Initially experiment is being carried out on small dataset and performed cluster analysis. The best algorithm is applied on the extended dataset. Along with different clusters of the related documents the resulted silhouette coefficient, entropy and F-measure trend are presented to show algorithm behavior for each data set.

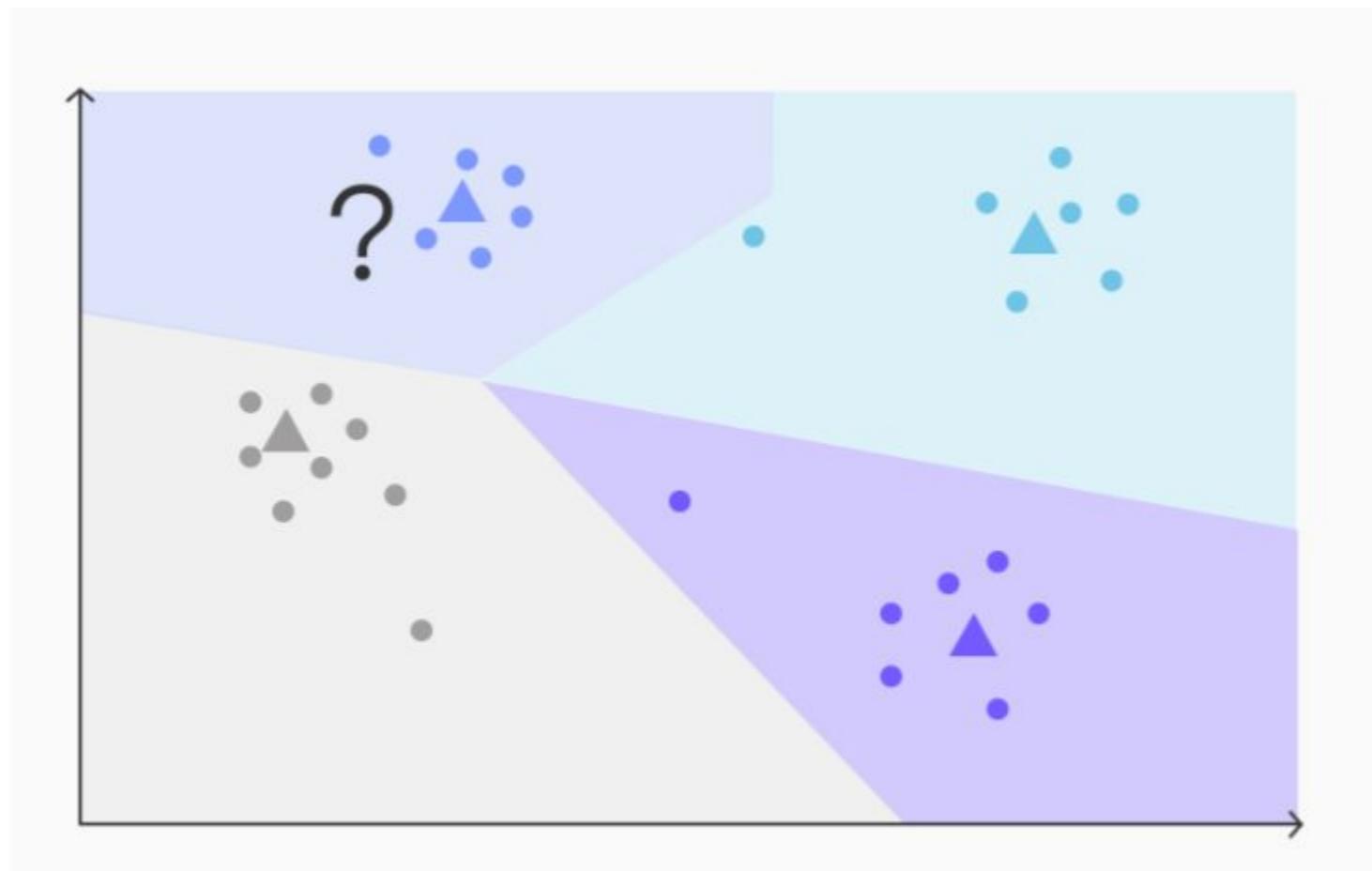
### Authors

Published in: 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)

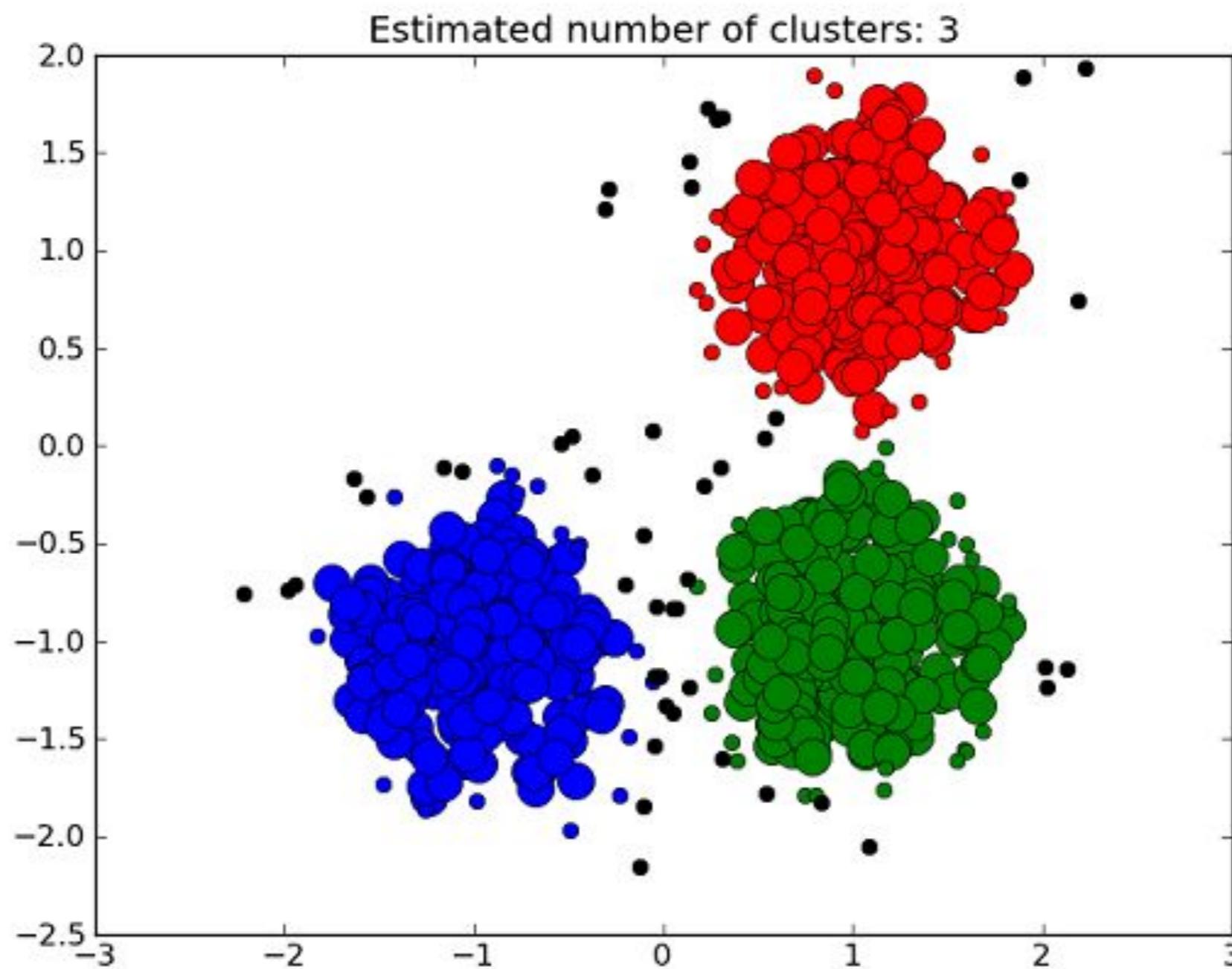
### Figures

# Clustering para ser usado en otra tarea

- Compresión
- Búsqueda eficiente de vecinos más cercanos (*ivfflat* index en pgvector)



# Busca capturar agrupaciones naturales en los datos



# Análisis de clusters es una tarea esencial para muchas aplicaciones

Por ej:

- Encontrar clusters naturales y describir sus propiedades (data understanding)
- Encontrar agrupamientos útiles (data class identification)
- Encontrar representantes para grupos homogéneos (data reduction)
- Encontrar objetos inusuales (outliers detection)
- Encontrar perturbaciones aleatorios de los datos (noise detection)

# Aplicaciones

- WWW (e.j. clasificación de documentos en buscadores)
- Reconocimiento de patrones (e.j. agrupar series de tiempo)
- Procesamiento de imágenes
- Recomendación
- etc.

source:

<http://www.cs.put.poznan.pl/jstefanowski/sed/DM-7clusteringnew.pdf>

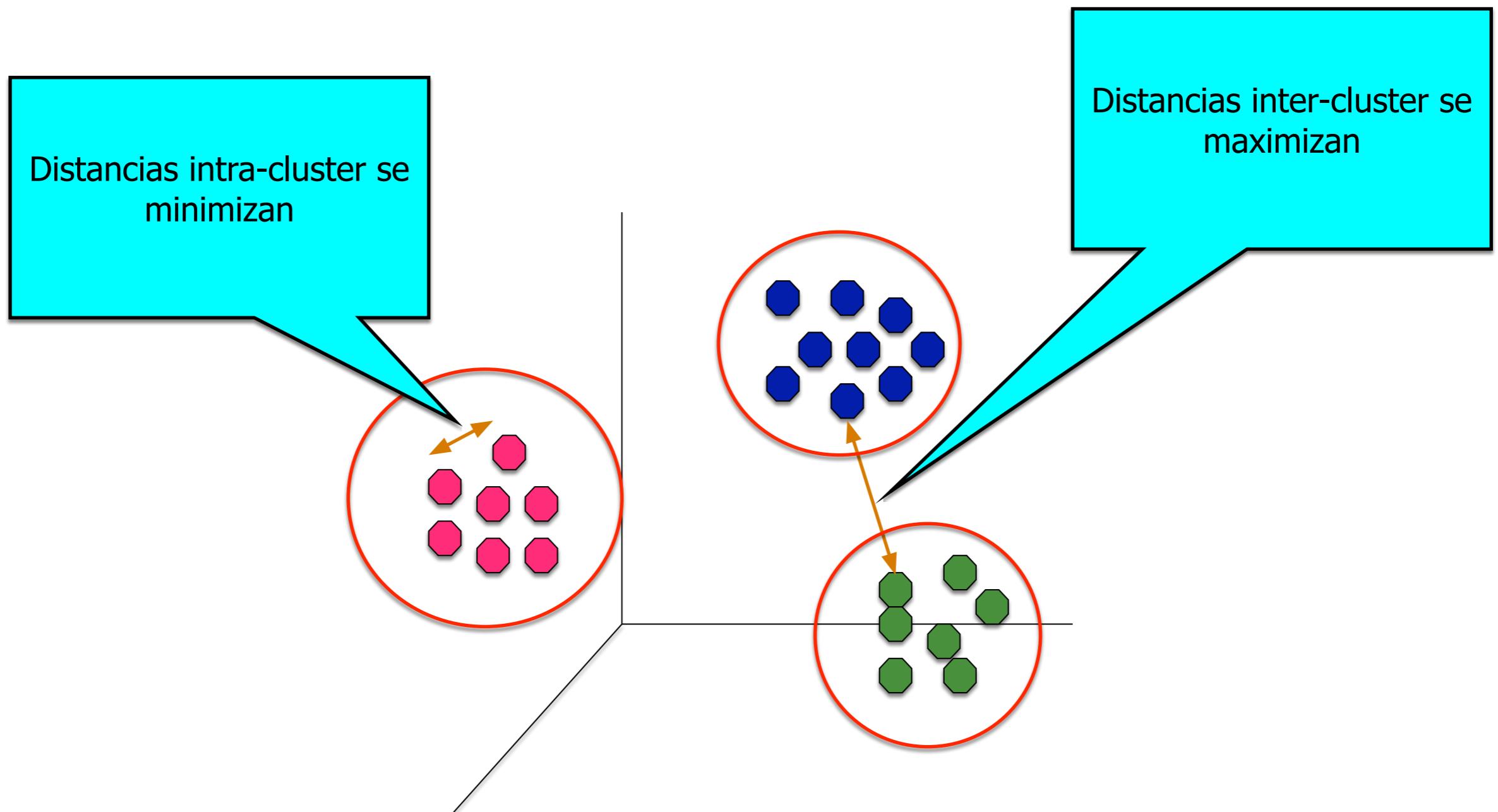
# más aplicaciones

- Imagenes
- Web
- Películas
- Marketing

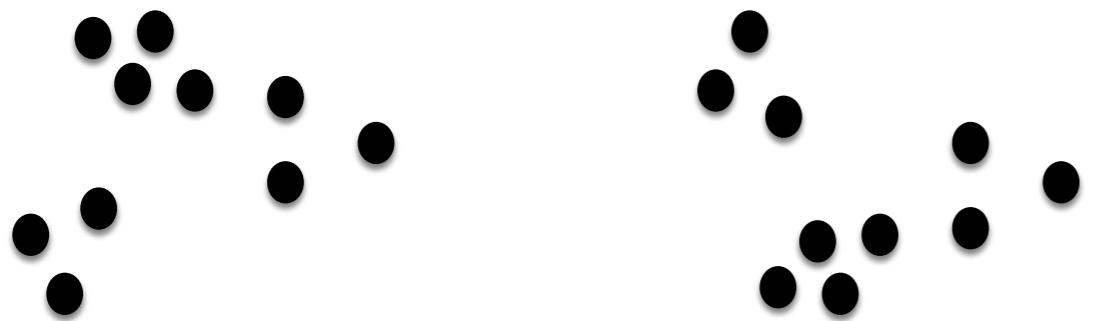
# Formulación del problema

- Dado un conjunto de datos no etiquetados, queremos asignarlos en clusters (grupos, clases) no previamente definidos.

# ¿Qué es análisis de clusters?

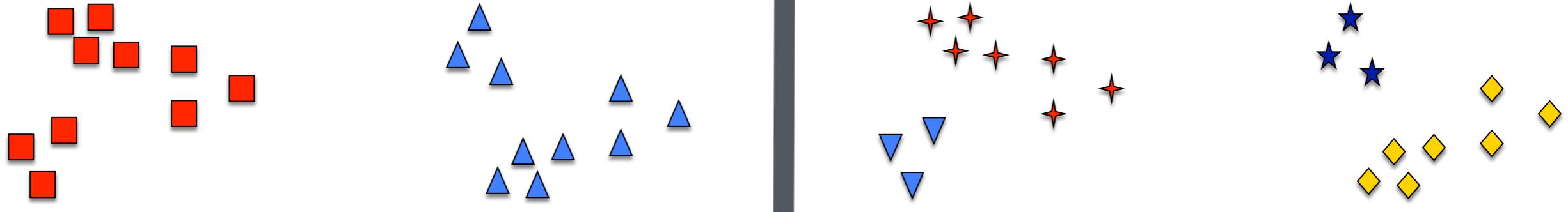
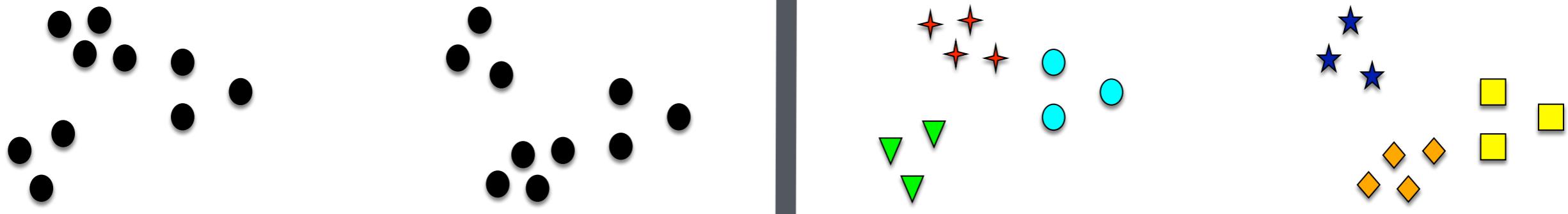


# La noción de cluster puede ser ambigua



*¿Cuántos clusters?*

# La noción de cluster puede ser ambigua

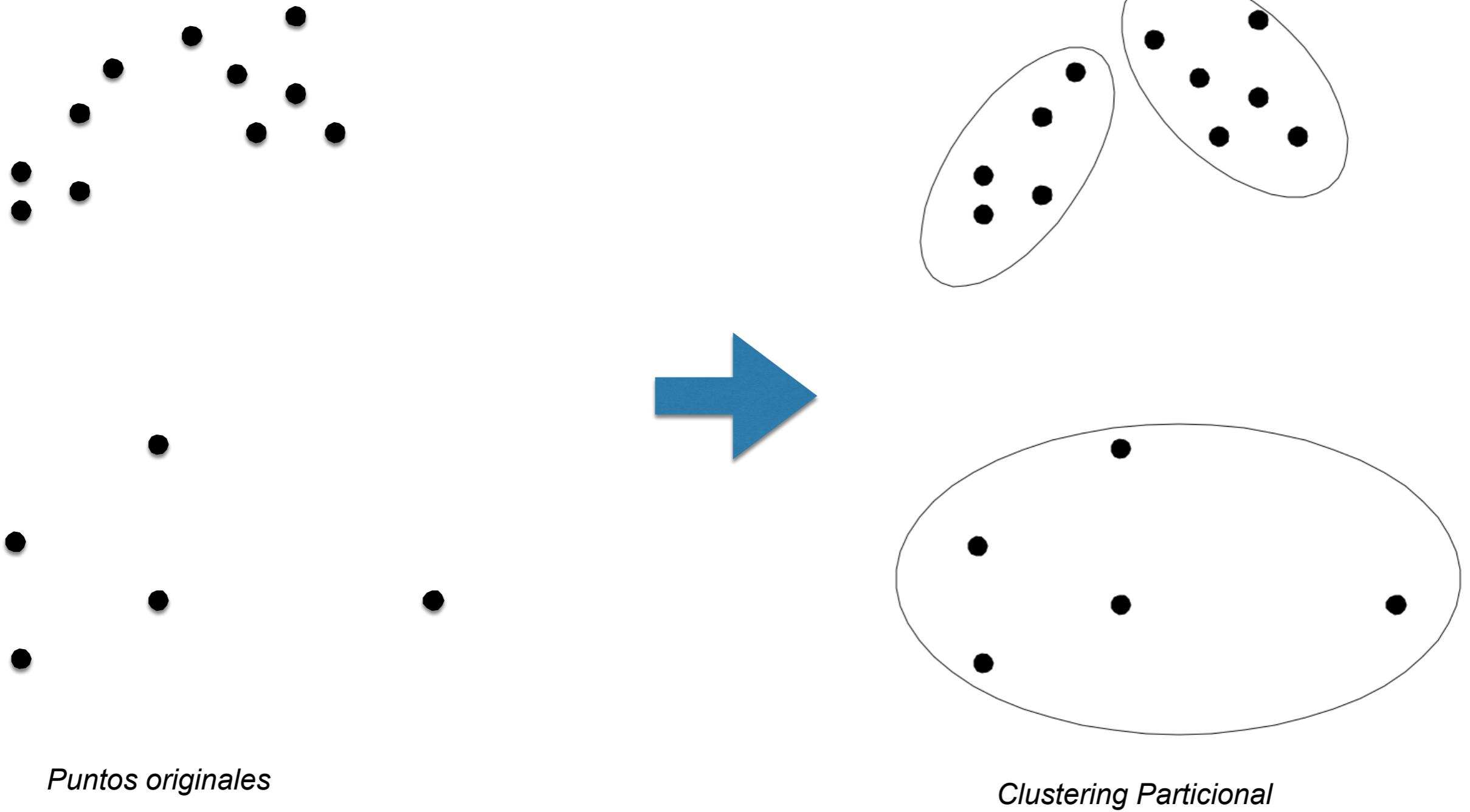


*Cuatro Clusters*

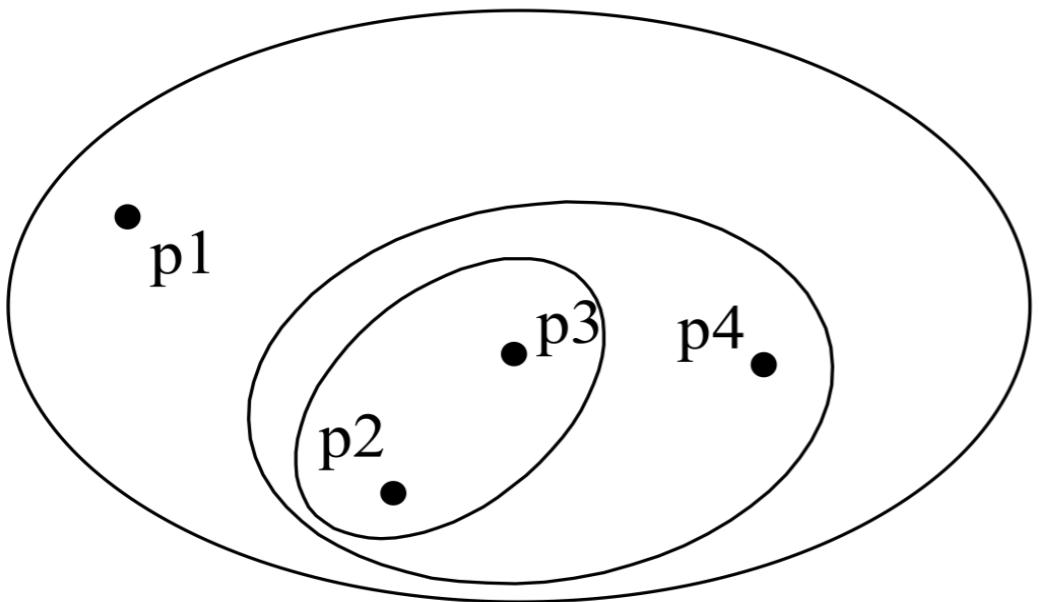
# Tipos de clustering

- Clustering Particional
  - Divide los datos en subconjuntos sin superposición (clusters), tal que cada dato está en un solo subconjunto
- Clustering Jerárquico
  - Un conjunto de clusters anidados, organizados como un árbol.
- Clustering Probabilístico o Difuso
  - Cada objeto pertenece a cada cluster con un peso de pertenencia entre 0 y 1.

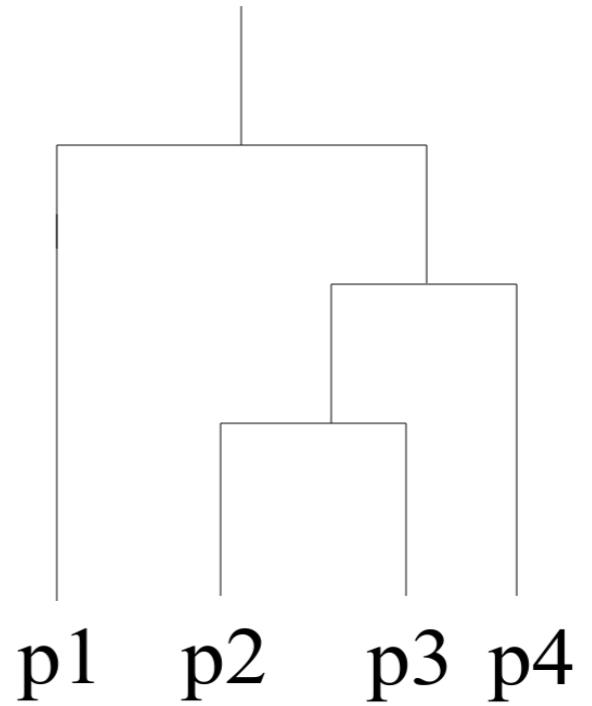
# Clustering particional



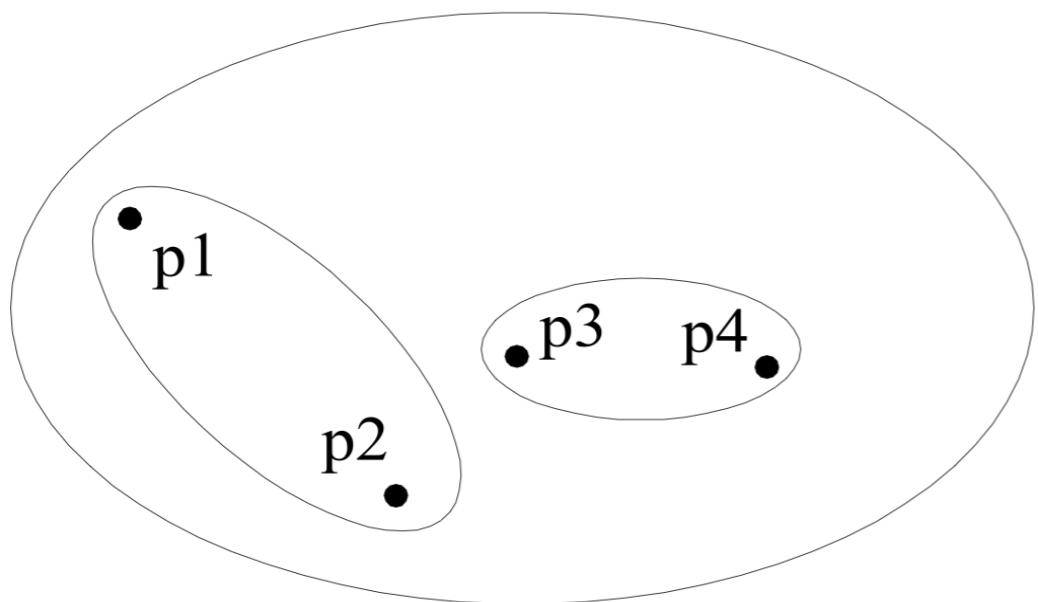
# Clustering jerárquico



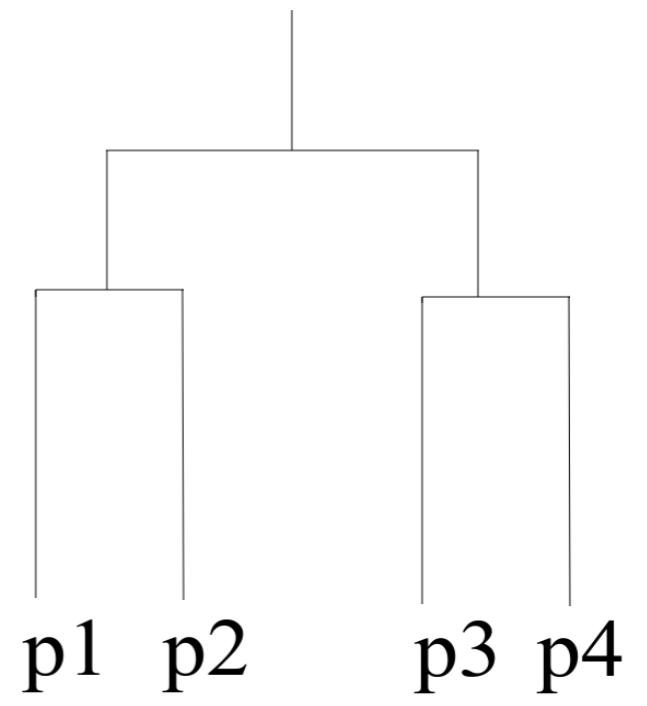
*Clustering Jerárquico tradicional*



*Dendograma tradicional*



*Clustering Jerárquico no tradicional*



*Dendograma no tradicional*

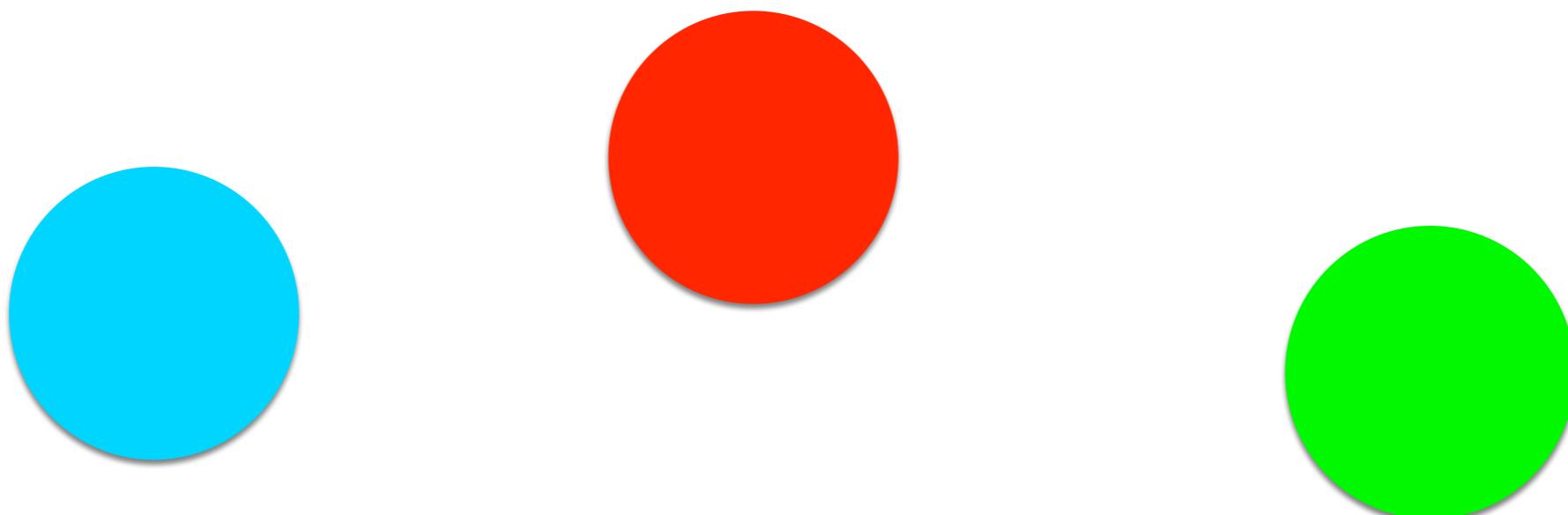
# Tipos de clusters

Existen distintas nociones de clusters que pueden ser consideradas:

- Bien separados
- Basados en un centro
- Contiguos
- Basados en densidad
- Propiedad o Conceptual

# Clusters bien separados

- Un cluster es un conjunto de puntos, tal que: cualquier punto en un cluster está más cerca (es más similar) **a cualquier otro punto en el mismo cluster** que a cualquier punto fuera de este.



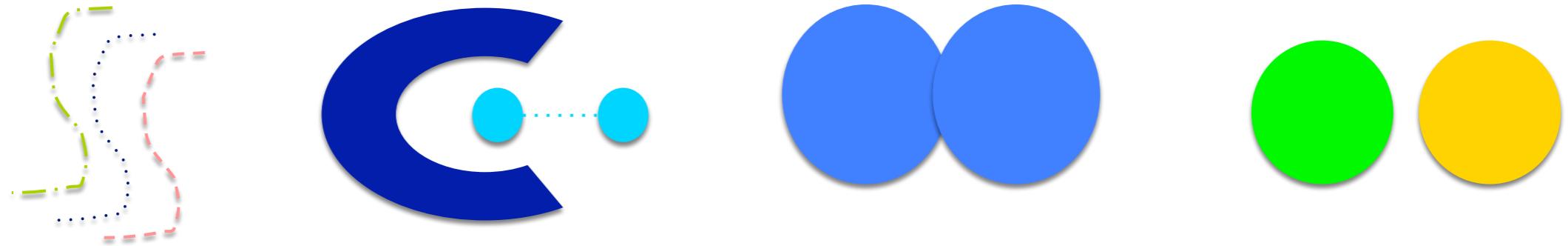
# Clusters basados en un centro

- Un cluster es un conjunto de objetos, tal que: un objeto dentro del cluster está más cerca (es más similar) **al centro de este cluster** que al centro de cualquier otro.
- El centro de un cluster puede ser el centroide, el promedio de todos los puntos en el cluster, o el medioide, el punto más “representativo” del cluster



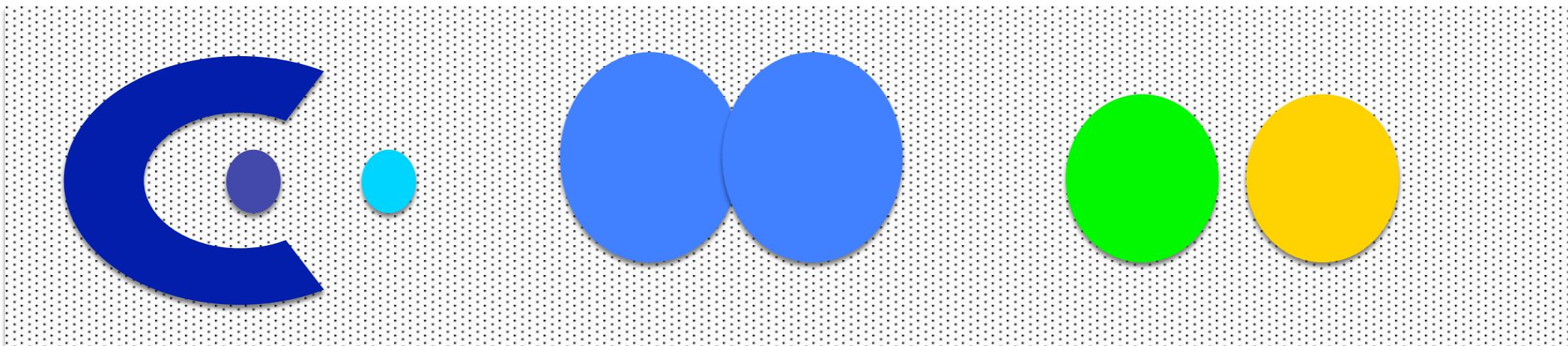
# Clusters contiguos (vecino más cercano o transitivo)

- Un cluster es un conjunto de puntos, tal que: un punto en un cluster está más cerca (es más similar) **a uno o más puntos en el cluster** que a cualquier punto no en el cluster



# Clusters basados en densidad

- Un cluster es una región densa de puntos, separada por regiones de baja densidad de otras regiones de alta densidad
- Usado cuando los clusters son irregulares o están entrelazados, y cuando hay ruido y outliers



# Algoritmos de Clustering

- K-means
- Clustering jerárquico aglomerativo
- DBSCAN

# Algoritmos de Clustering

- **K-means**
- Clustering jerárquico aglomerativo
- DBSCAN

# K-means

- Separa los datos en K clusters, sin sobreposición (particional)
- Idea base: un buen *clustering* es aquel donde la variación intra-cluster es pequeña.
  - Variación intra-clúster: suma de distancias (al cuadrado) entre los distintos elementos del cluster, dividida por número de elementos en el cluster
- Queremos minimizar esa cantidad -> no es fácil!
  - Existen  $O(K^n)$  formas de asignar n elementos en K grupos
- Debemos encontrar un algoritmo que permita encontrar un mínimo (aunque sea local)

# K-means

## Algoritmo de K-Means (abstracto)

- Input: Un dataset de atributos numéricos.
- Párametro: número de clusters K
- Se asignan K centroides iniciales (aleatorios)
- Se realizan dos operaciones iterativamente:  
**asignar y recalcular centroides**
  - a. Asignar: cada punto es asignado a su centroide más cercano.
  - b. Recacular centroides: se recalculan los centroides promediando sus puntos.
- Iterar hasta converger.

# Centroide

- Es el punto medio de los elementos de un cluster.
- No necesariamente es un punto del cluster

Ejemplo: Sean 3 vectores de tres dimensiones:

$$\vec{x}_1 = [6, 4, 3], \vec{x}_2 = [4, 5, 1], \vec{x}_3 = [2, -3, 5]$$

El centroide de estos vectores es:

$$c(\vec{x}_1, \vec{x}_2, \vec{x}_3) = [(6 + 4 + 2)/3, (4 + 5 - 3)/3, (3 + 1 + 5)/3] = [4, 2, 3]$$

# Algoritmo

---

## **Algorithm 1** Basic K-means Algorithm.

---

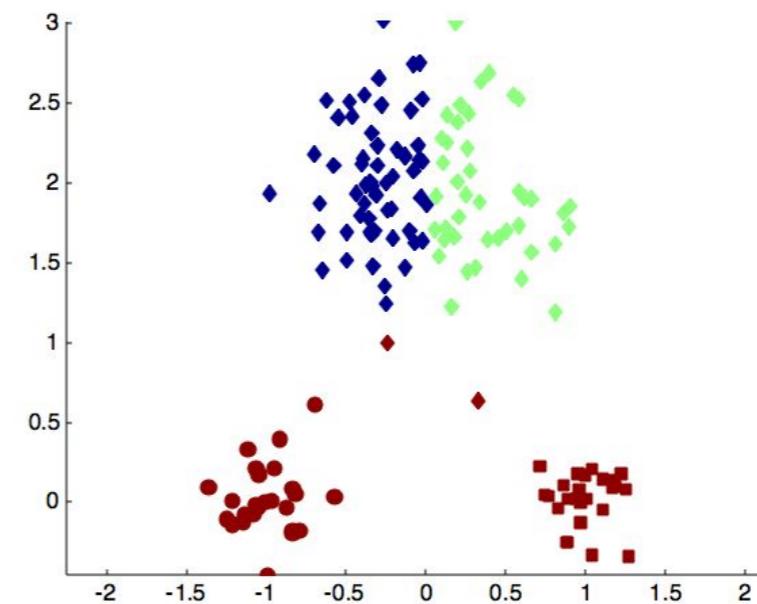
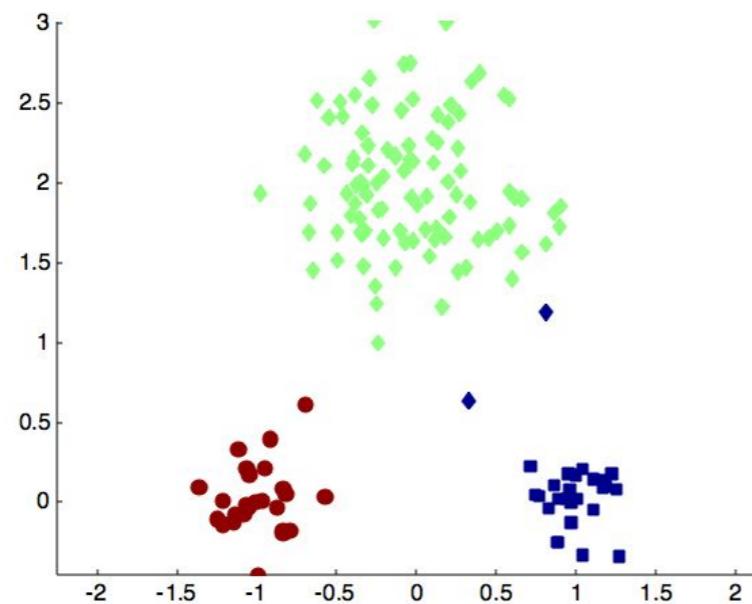
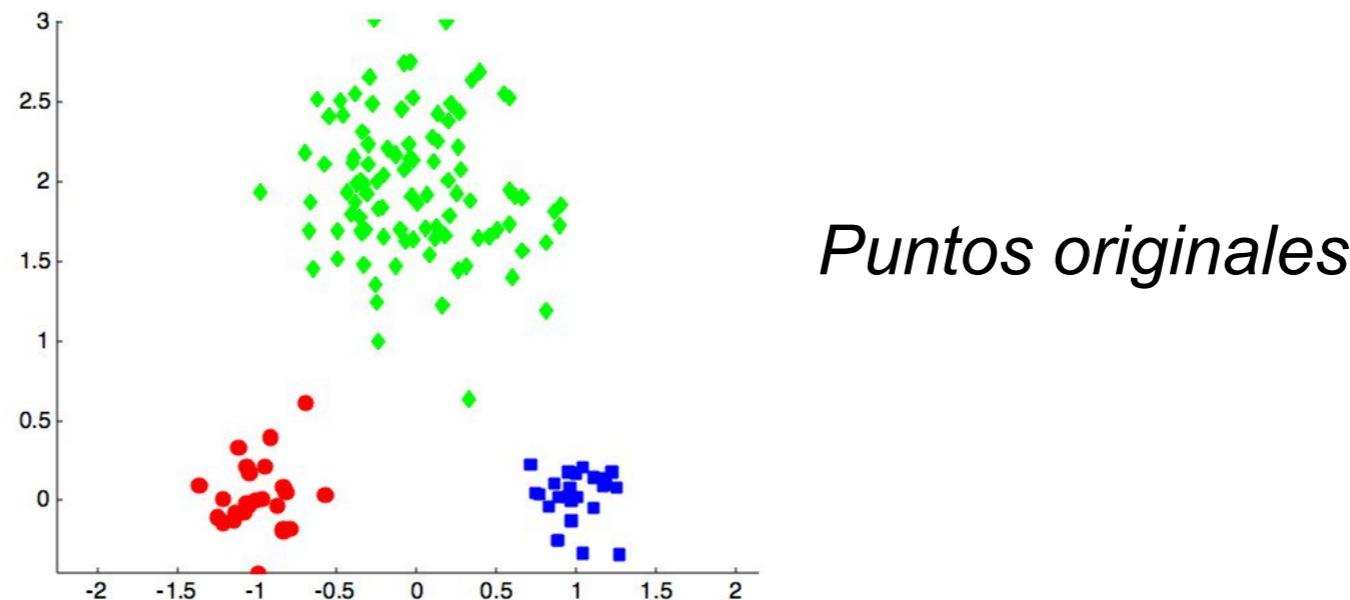
- 1: Select  $K$  points as the initial centroids.
  - 2: **repeat**
  - 3:   Form  $K$  clusters by assigning all points to the closest centroid.
  - 4:   Recompute the centroid of each cluster.
  - 5: **until** The centroids don't change
-

# K-means

## Detalles del algoritmo:

- Centroides iniciales: aleatorios
  - Clusters varían dependiendo de la elección
- “Cercanía” se mide con alguna distancia (generalmente usamos distancia euclídea para variables numéricas)
- K-means converge para distancias “usuales”
- En general la convergencia sucede con pocas iteraciones
  - Iterar hasta que cambien “pocos” puntos de cluster
- Complejidad es  $O(n * K * I * d)$ 
  - $n$  puntos,  $K$  centros,  $I$  iteraciones,  $d$  dimensiones
- Visualización:  
<https://www.youtube.com/watch?v=5I3Ei69I40s>

# K-means no asegura encontrar los clusters óptimos



# K-means: escogiendo los centroides iniciales

- Escoger los centroides iniciales es una pieza clave en K-means.
- Enfoque tradicional: inicializar los centroides aleatoriamente.
- Cuando los centroides iniciales son escogidos aleatoriamente, diferentes ejecuciones de k-means producen distintos valores de SSE.
- Solución simple: correr K-means varias veces variando la semilla aleatoria y quedarse con el modelo de menor SSE.
  - ¡Esto último **no garantiza** que encontraremos los clusters óptimos!
- Ejemplo:  
<https://www.youtube.com/watch?v=9nKfViAfajY>

# Sum of Squared Error (SSE)

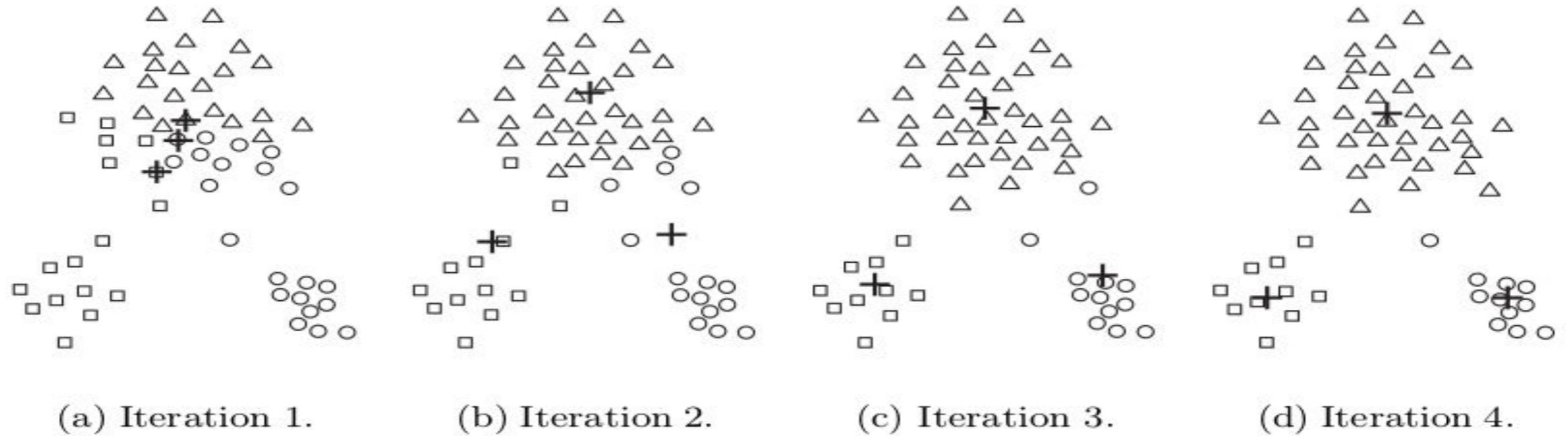
- Medida más común para evaluar clusters
- Suma de las distancias cuadradas de cada punto al centroide de su cluster asignado.

$$\text{SSE} = \sum_{i=1}^K \sum_{\mathbf{x} \in C_i} \text{dist}(\mathbf{c}_i, \mathbf{x})^2$$

- Dados 2 clusterings distintos, se escoge el que tiene menor error

# Comparando distintas inicializaciones

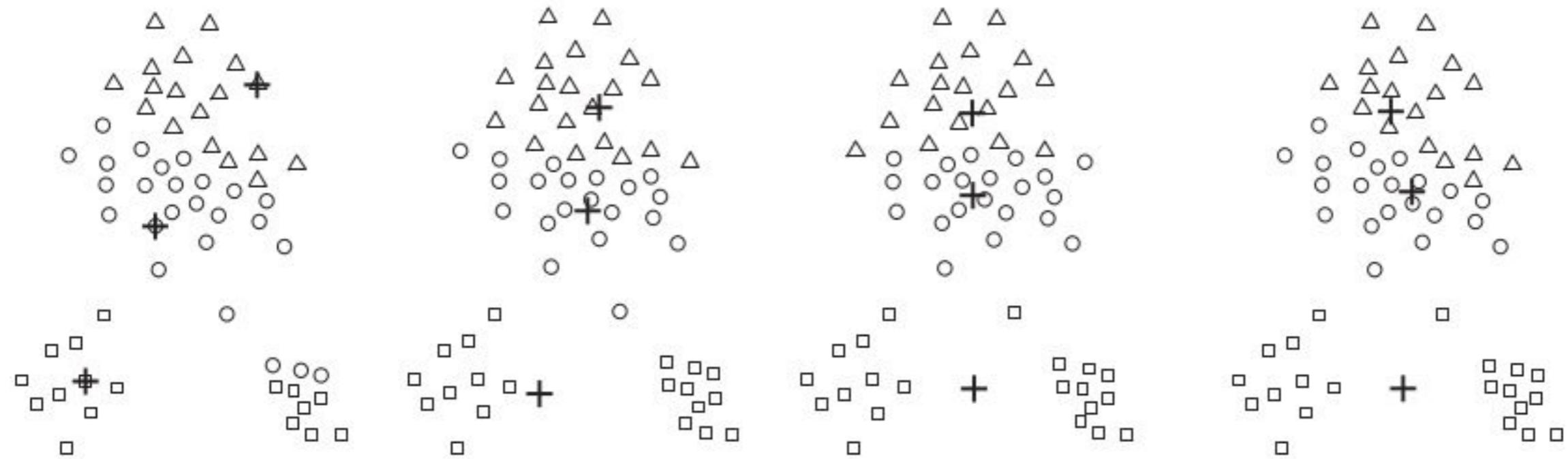
Caso 1:



Aunque los centroides iniciales están todos en único cluster, igual se converge a los clusters deseados.

# Comparando distintas inicializaciones

Caso 2:



(a) Iteration 1.

(b) Iteration 2.

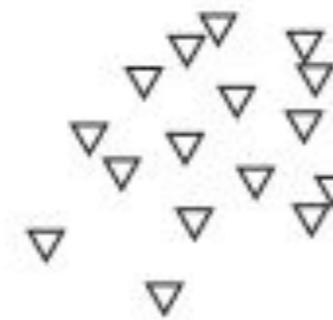
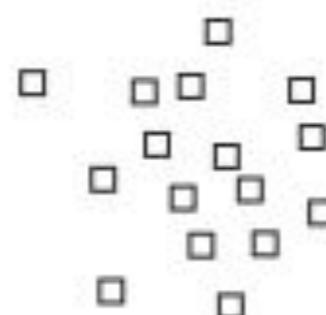
(c) Iteration 3.

(d) Iteration 4.

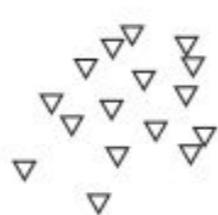
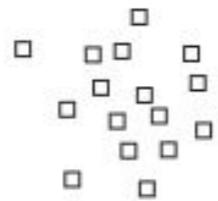
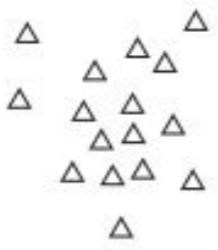
Aunque los centroides iniciales parecen estar mejor repartidos, llegamos a una solución peor que en el caso anterior.

# Otro ejemplo

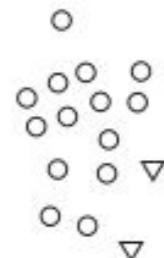
- Tenemos datos donde hay dos pares de clusters (el par izquierdo y el par derecho).
- Los clusters de cada par están más cerca entre sí que de los clusters del otro par.



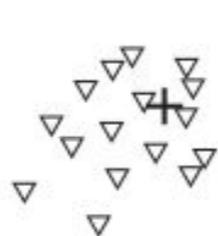
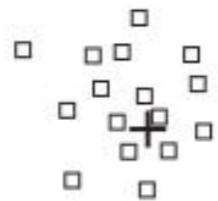
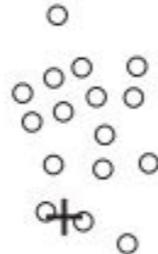
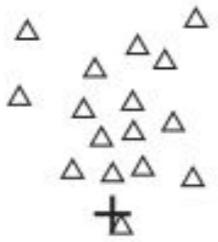
# Caso 1



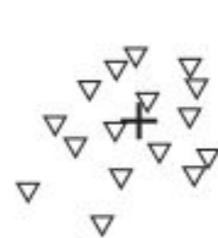
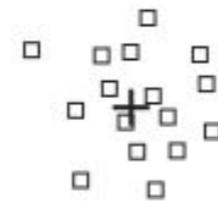
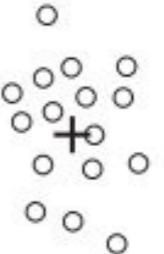
(a) Initial points.



(b) Iteration 1.



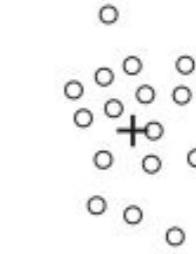
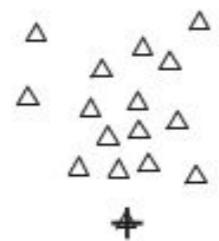
(c) Iteration 2.



(d) Iteration 3.

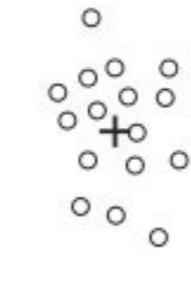
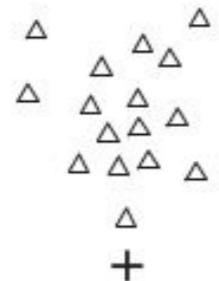
Si empezamos con dos centroides iniciales en cada par, incluso si los pares de centroides están en el mismo cluster, los centroides se redistribuyen para encontrar los clusters reales.

# Caso 2



(a) Iteration 1.

(b) Iteration 2.



(c) Iteration 3.

(d) Iteration 4.

Por otro lado, si un par de clusters recibe sólo un centroide inicial y el otro par recibe 3, entonces 2 de los clusters reales se combinarán y un cluster será dividido.

# K-means

- Idealmente nos gustaría partir con un centroide por cluster “real”.
- Si hay K clusters “reales”, probabilidad de escoger un centroide por cluster es baja
- Asumiendo n elementos en cada cluster:

$$P = \frac{\text{# formas de escoger un centroide de cada cluster}}{\text{# formas de escoger K centroides}} = \frac{K!n^K}{(Kn)^K} = \frac{K!}{K^K}$$

Ej.: si  $K = 10$ , entonces  $P = 10!/10^{10} = 0.00036$

# K-means: Manejando clusters vacíos

- Algoritmo K-means puede retornar clusters vacíos si no se le asignan puntos al cluster en el paso de asignación.
- Estrategias para encontrar centroide de reemplazo:
  - Escoger el punto más lejano a todos los centroides como nuevo centroide (punto que contribuye más al SSE)
  - Escoger un punto aleatorio del cluster con mayor SSE.
    - Esto generalmente dividirá ese cluster y reducirá el SSE total.

# K-means: Preprocesamiento

- Normalizar los datos: que todos los atributos aporten lo mismo a las distancias.
- Eliminar outliers: outliers producen centroides no representativos con alto SSE.

# K-means: Postprocesamiento

- Aumentar el número de clusters es la solución trivial para bajar el valor del SSE.
- Eso no es lo queremos => tener K igual al número de datos nos daría un SSE de cero.

Bajar el SSE de forma más inteligente:

- Eliminar clusters pequeños que puedan representar outliers
- Dividir clusters “sueltos” (con alto SSE)
- Mezclar clusters cercanos y con bajo SSE.

# Bisecting K-means

- Extensión simple de K-means
- Idea: Dividir el conjunto de todos los puntos en dos clusters, escoger uno de los dos para ser dividido, e iterar hasta producir K clusters.
- Cada división se obtiene ejecutando K-means (con k=2)

---

**Algorithm 7.3** Bisecting K-means algorithm.

---

- 1: Initialize the list of clusters to contain the cluster consisting of all points.
- 2: **repeat**
- 3:   Remove a cluster from the list of clusters.
- 4:   {Perform several “trial” bisections of the chosen cluster.}
- 5:   **for**  $i = 1$  to *number of trials* **do**
- 6:     Bisect the selected cluster using basic K-means.
- 7:   **end for**
- 8:   Select the two clusters from the bisection with the lowest total SSE.
- 9:   Add these two clusters to the list of clusters.
- 10: **until** The list of clusters contains  $K$  clusters.

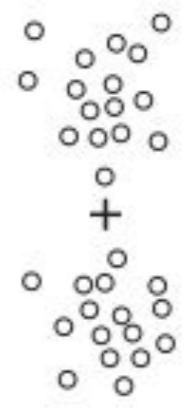
---

# Bisecting K-means

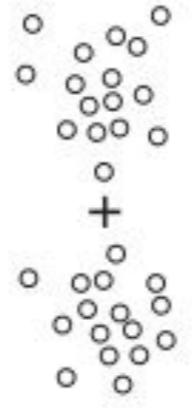
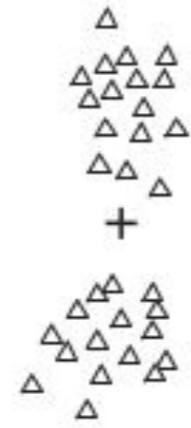
¿Cómo escojo los clusters a dividir (Paso 3 del algoritmo)?

- Opción 1: Escoger el cluster más grande en cada paso.
- Opción 2: Escoger el cluster con mayor SSE.
- Opción 3: Estrategia híbrida entre las dos anteriores

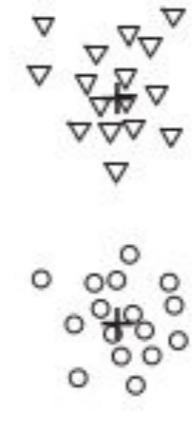
# Bisecting K-means con datos del ejemplo anterior



(a) Iteration 1.



(b) Iteration 2.



(c) Iteration 3.



Iteración 1: se encuentran dos pares de clusters.

Iteración 2: El par de clusters de la derecha es dividido.

Iteración 3: El par de clusters de la izquierda es dividido.

# Bisecting K-means

Bisecting K-means tiene menos problemas de inicialización que K-means.

- Esto es porque realiza varios intentos de biseción y toma la biseción de menor SSE.
- Además sólo se consideran dos centroides en cada paso.

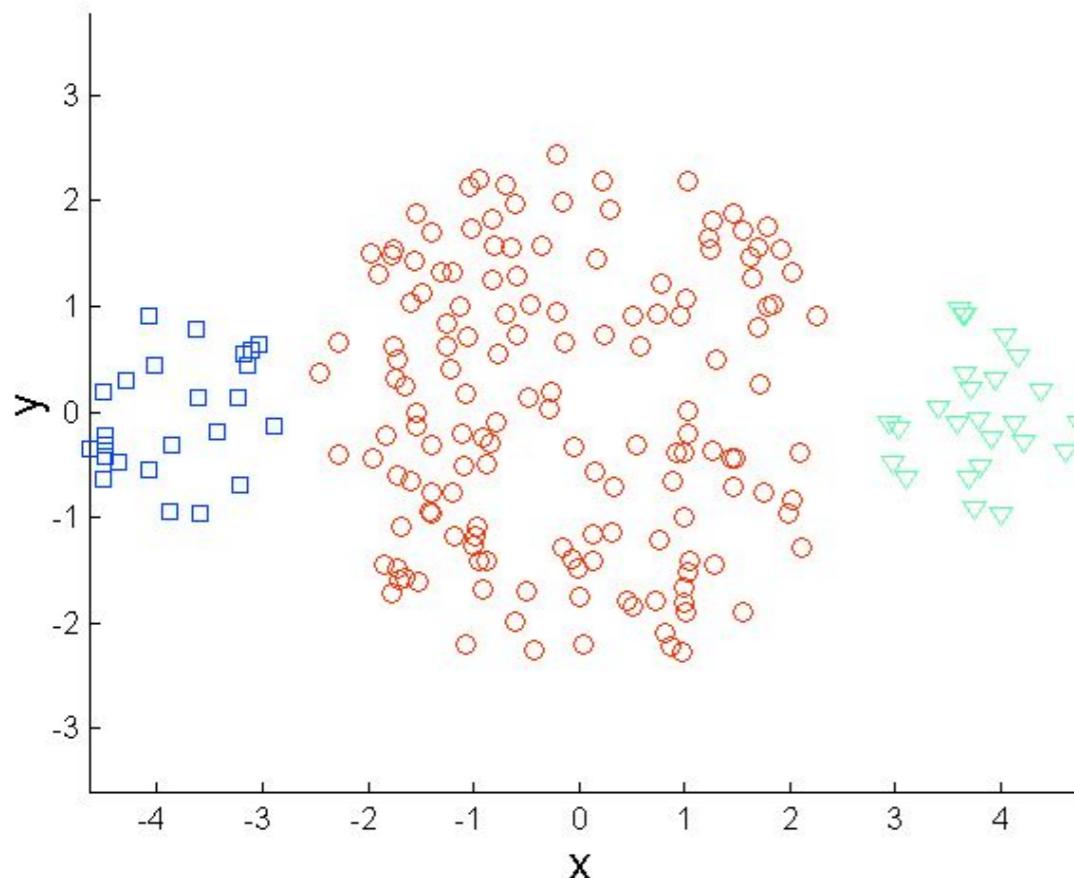
Si registramos la secuencia de clusters bisectados podemos producir un clustering jerárquico.

# K-means

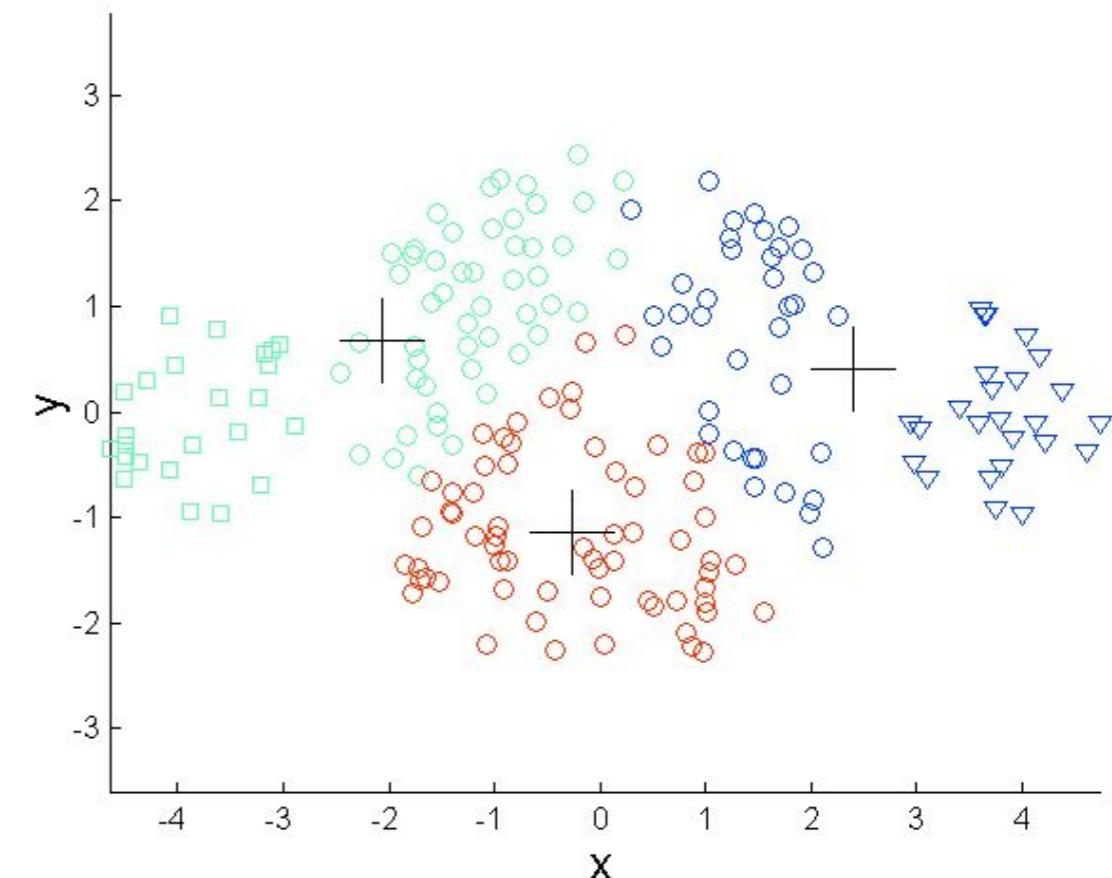
- Limitaciones de K-means
  - Clusters de diferente tamaño
  - Clusters de diferentes densidades
  - Clusters con formas no esféricas
- K-means no es robusto a outliers

# K-means

- Ejemplo: tamaños diferentes



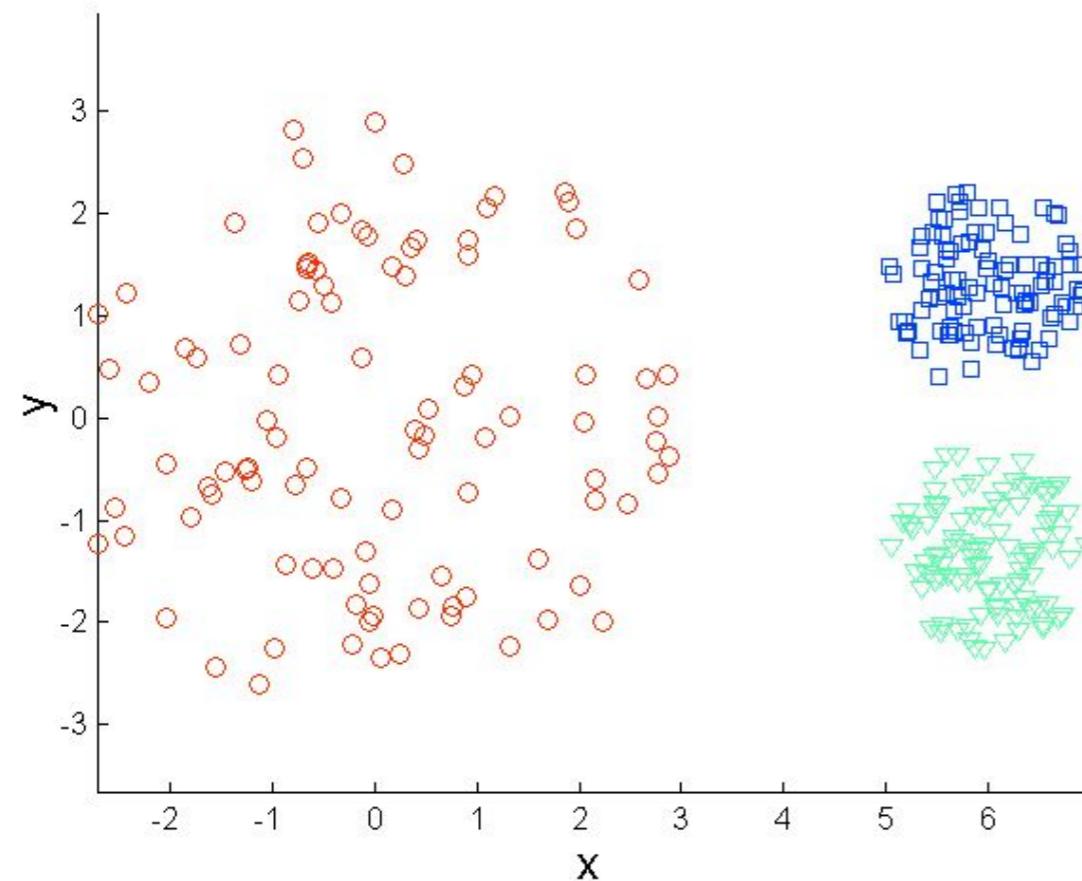
*Puntos originales*



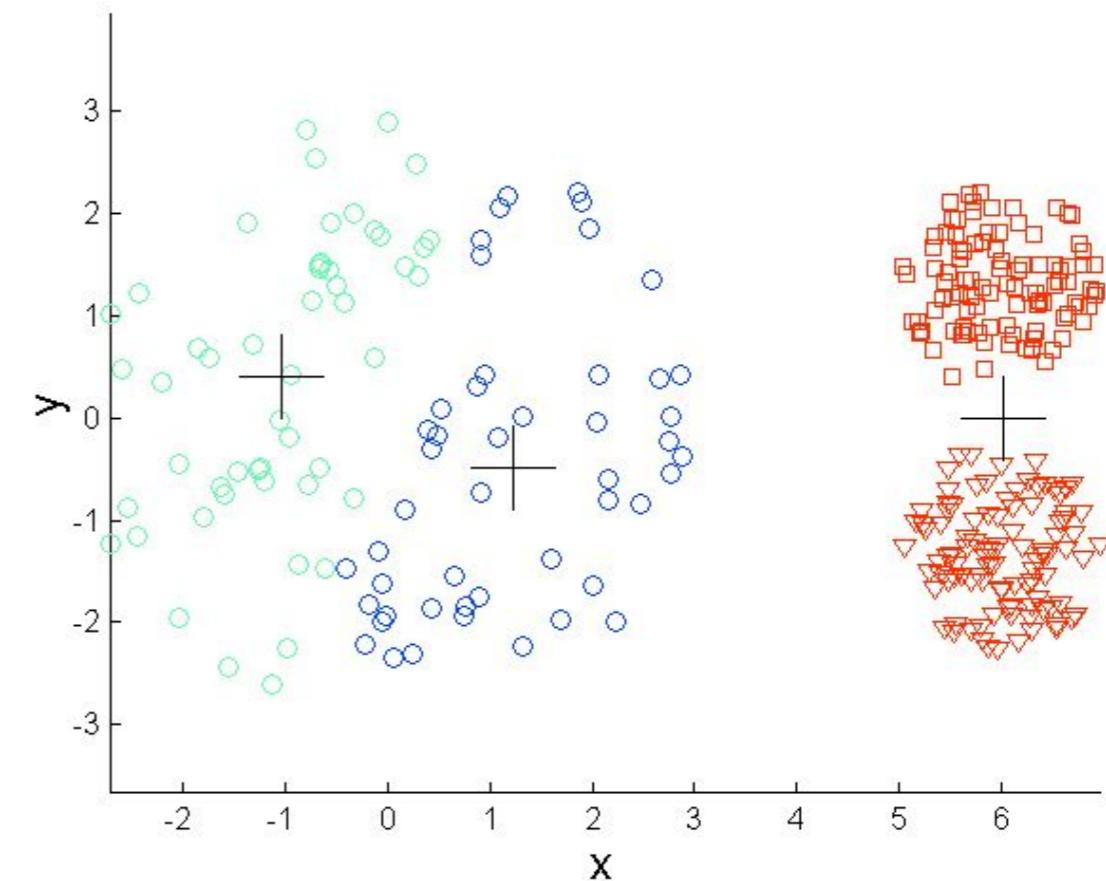
*K-means (tres clusters)*

# K-means

- Ejemplo: densidades diferentes



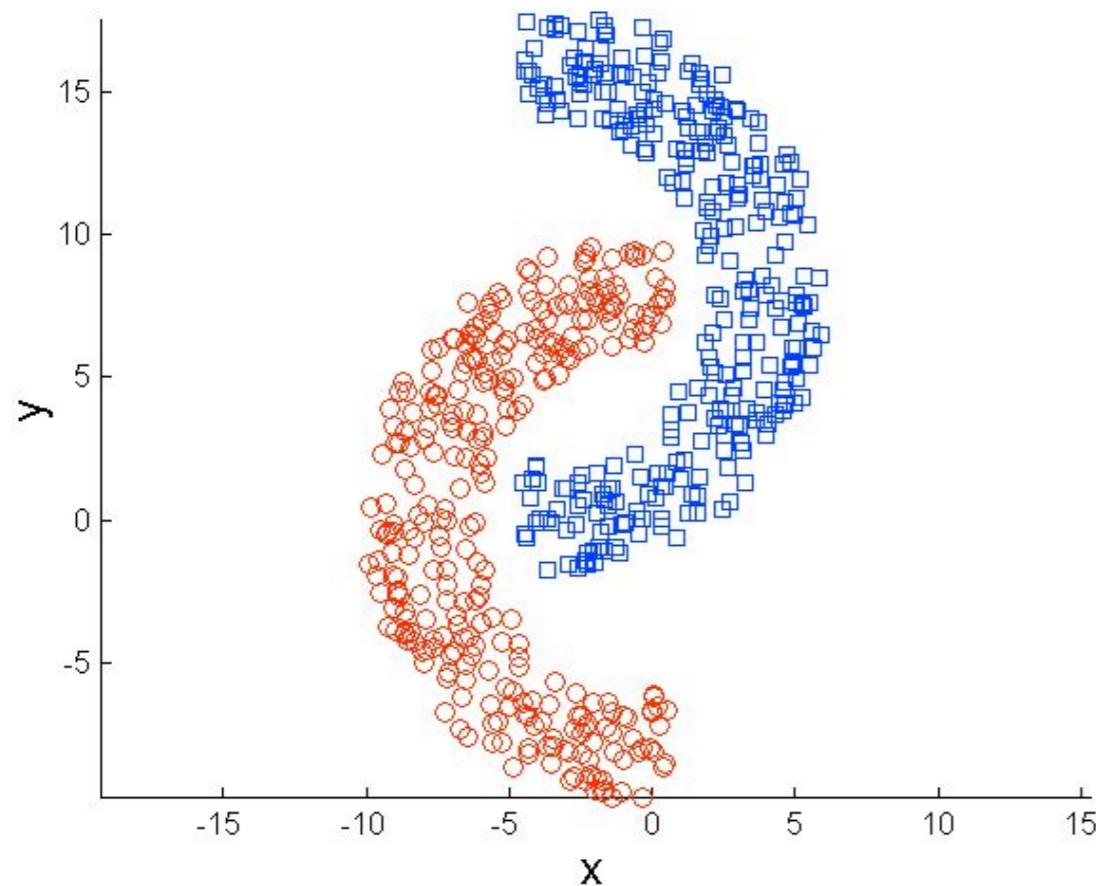
*Puntos originales*



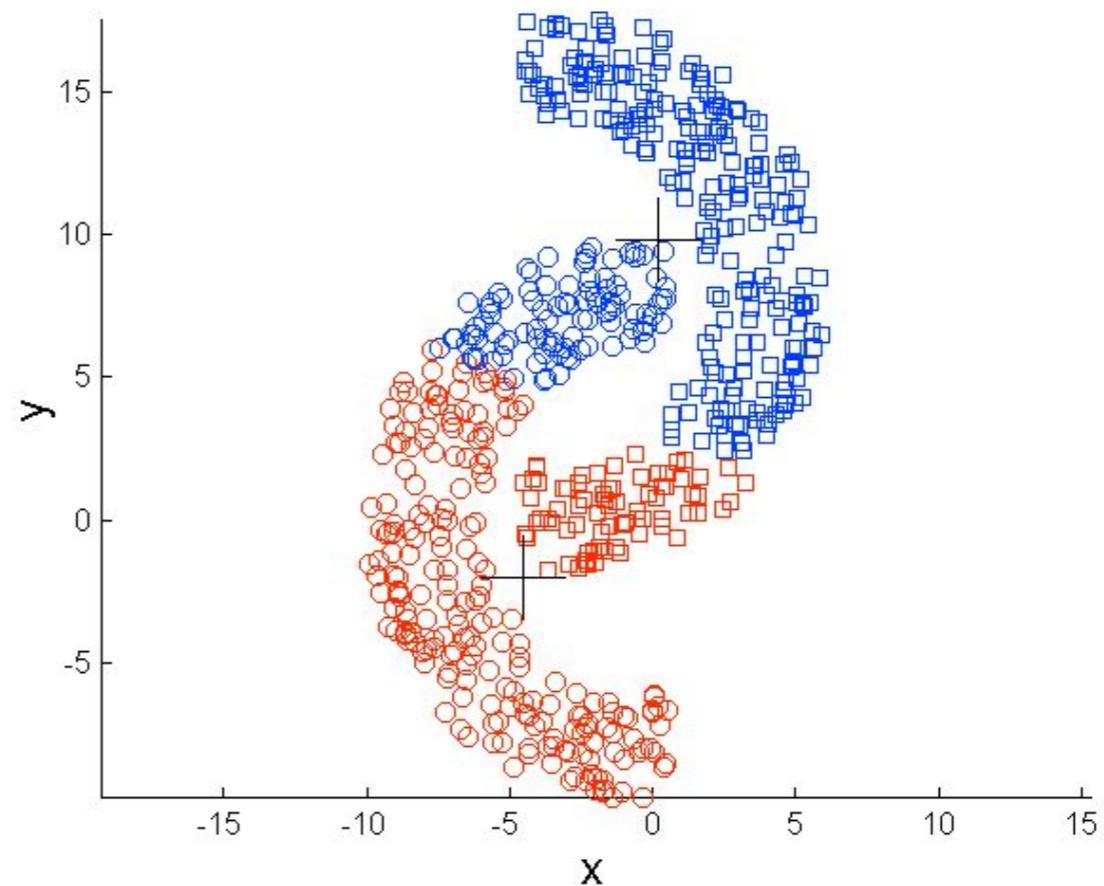
*K-means (tres clusters)*

# K-means

- Ejemplo: formas no esféricas



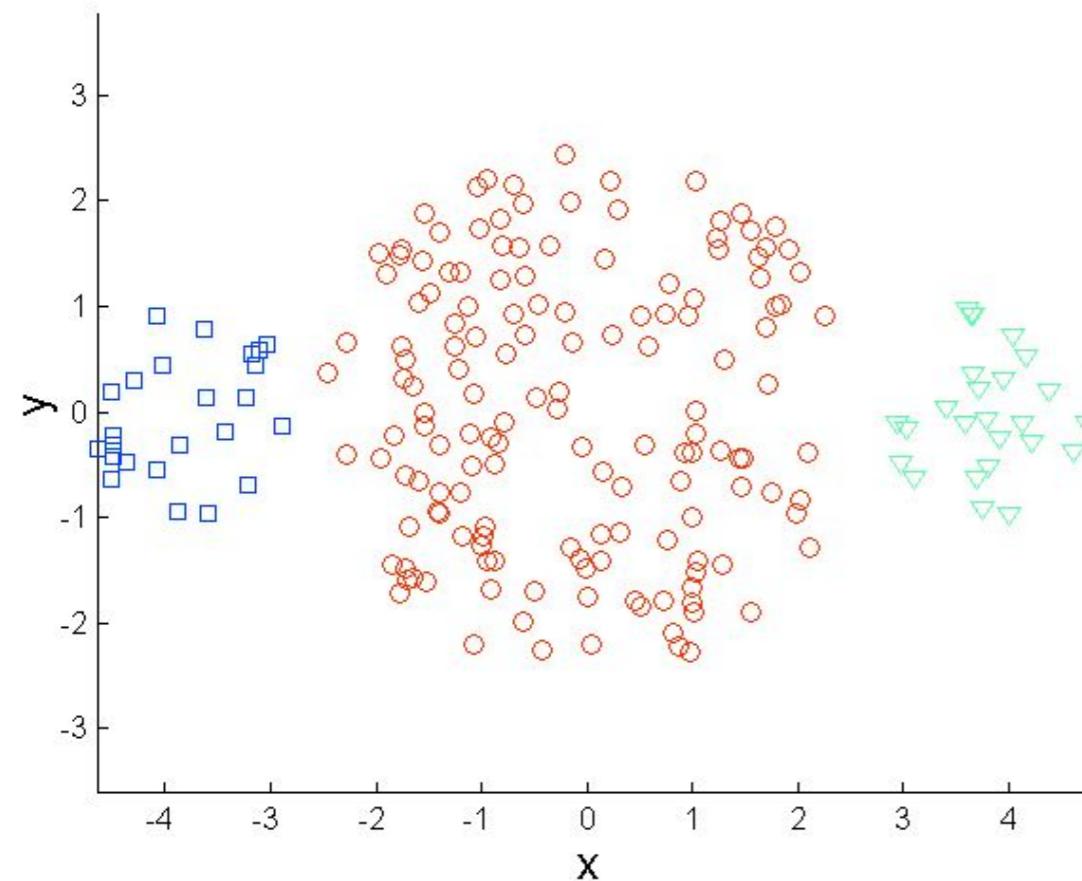
*Puntos originales*



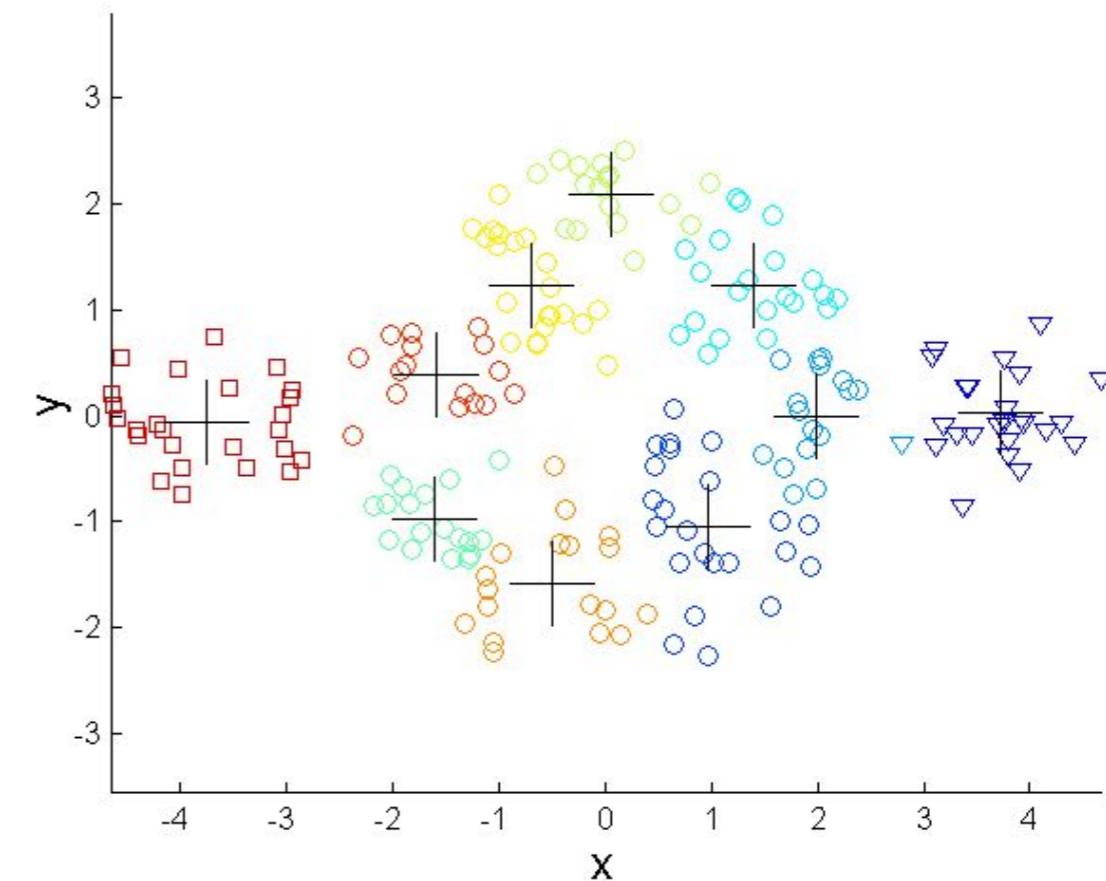
*K-means (dos clusters)*

# K-means

- Solución: usar K alto, luego mezclar clusters



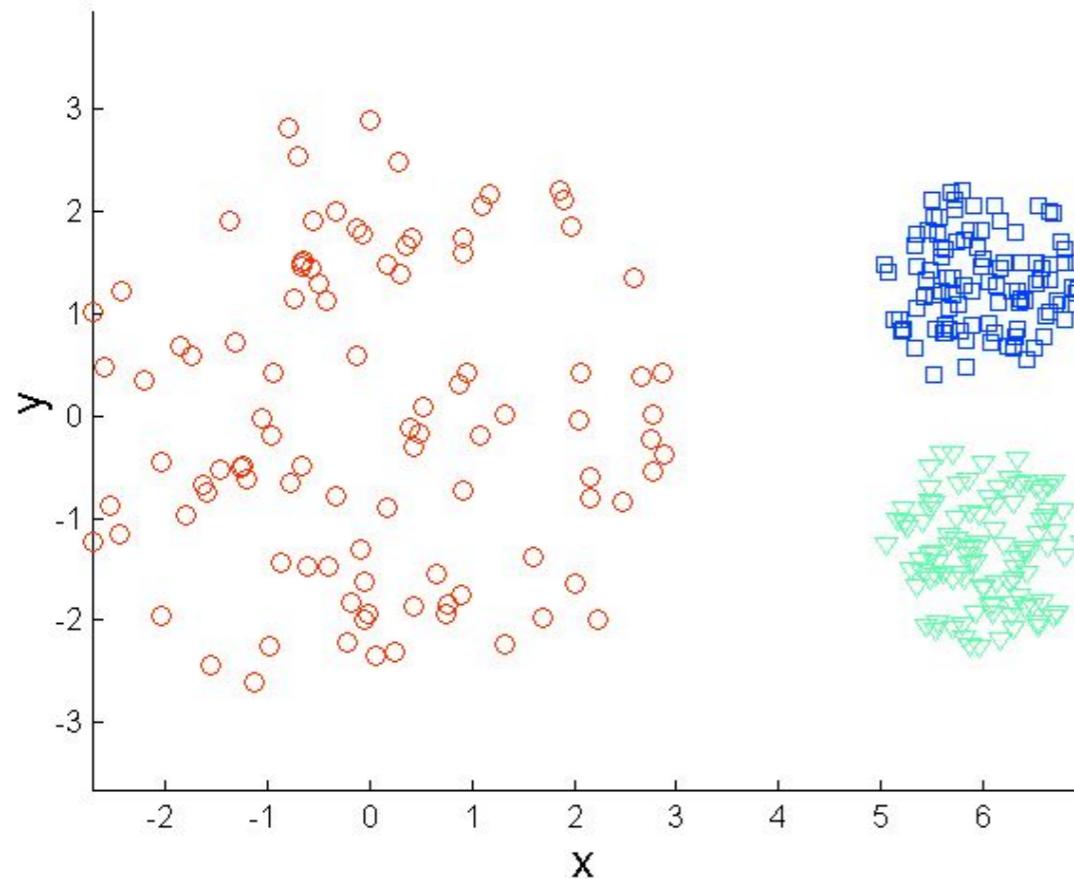
*Puntos originales*



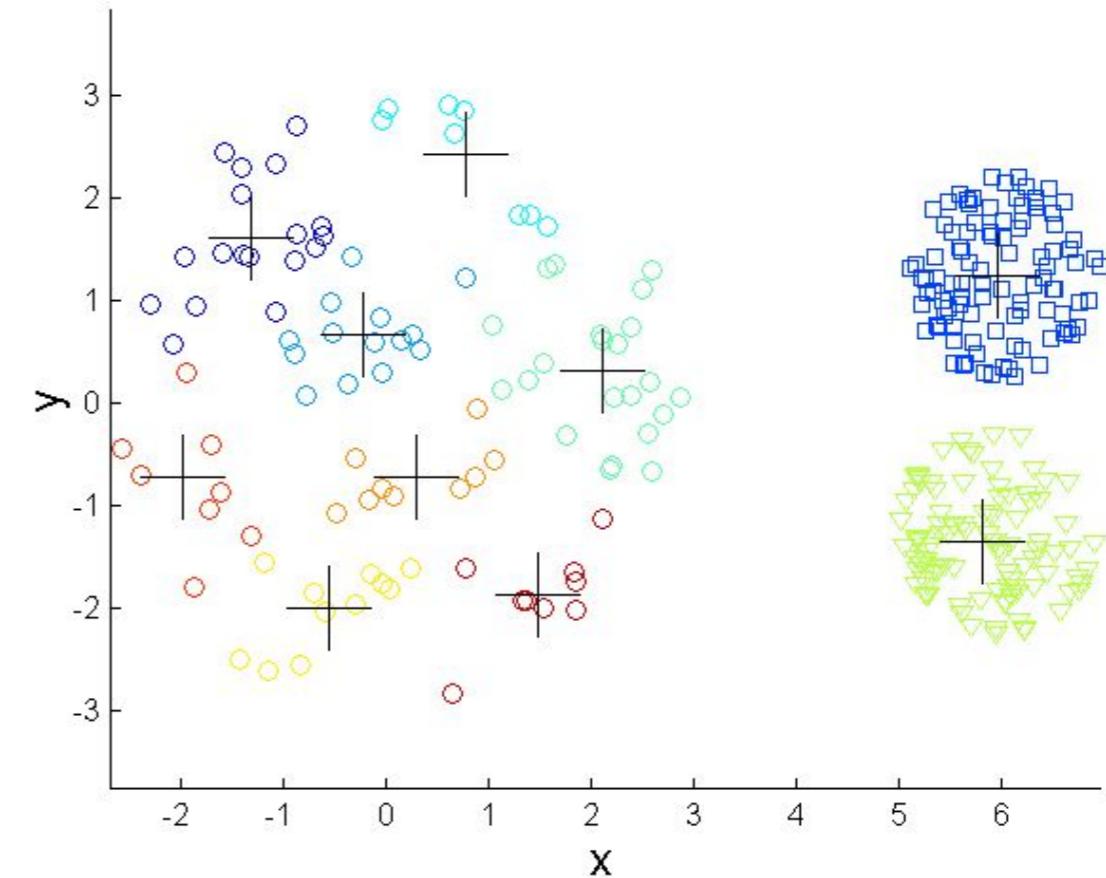
*K-means clusters*

# K-means

- Solución: usar K alto, luego mezclar clusters



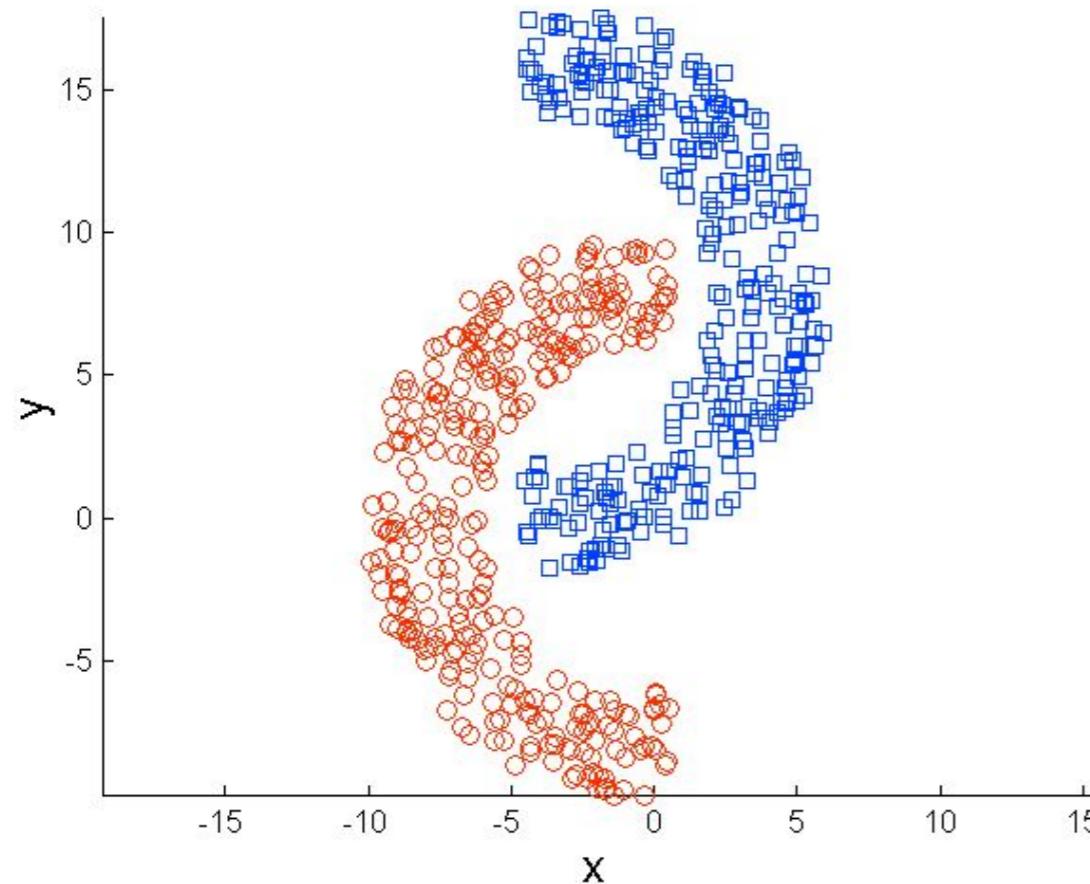
*Puntos originales*



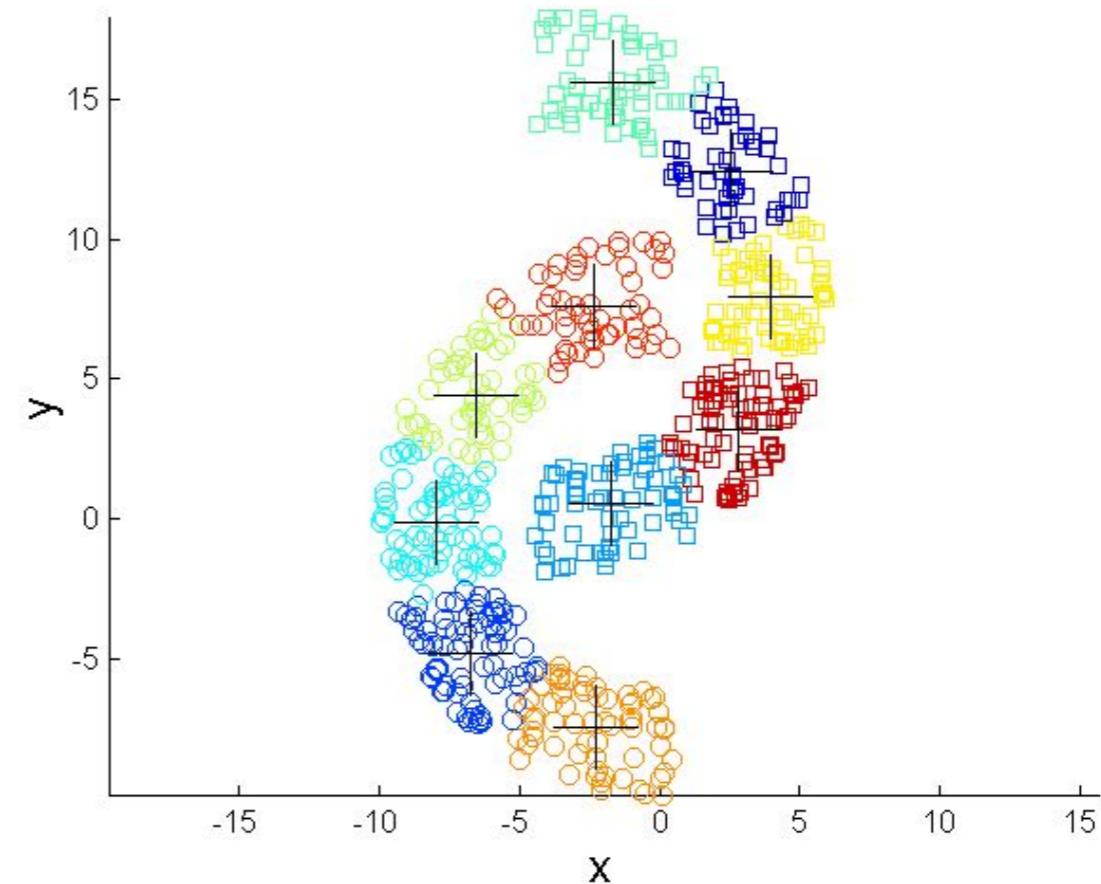
*K-means clusters*

# K-means

- Solución: usar K alto, luego mezclar clusters



*Puntos originales*



*K-means clusters*

# Algoritmos de Clustering

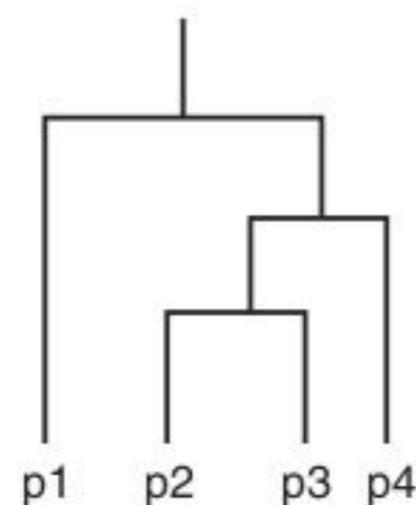
- K-means
- **Clustering jerárquico aglomerativo**
- DBSCAN

# Clustering Jerárquico Aglomerativo

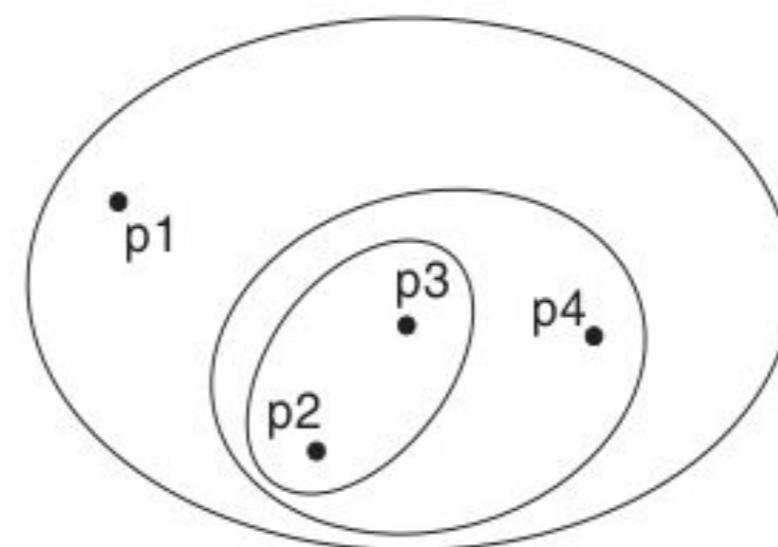
Produce un conjunto de clusters anidados organizados en un árbol jerárquico. Es una técnica antigua.

Visualizaciones:

- a) **Dendograma**: árbol que muestra las relaciones cluster-subcluster y el orden en que los clusters fueron mezclados o divididos.
- b) **Diagrama de clusters anidados**: sólo para puntos 2-dimensionales.



(a) Dendrogram.



(b) Nested cluster diagram.

# Clustering Jerárquico Aglomerativo

- **Fortalezas**
  - No tiene que suponer un número a priori de clusters
    - Se puede obtener cualquier número de clusters deseado “cortando” el dendograma en el nivel apropiado
  - Clusters pueden corresponder a taxonomía
    - Ejemplos en biología.

# Clustering Jerárquico Aglomerativo

- Tipos principales de clustering jerárquico
  - Aglomerativo
    - Empezar con cada punto como cluster individual
    - En cada paso, mezclar el par de clusters más cercano hasta que quede sólo un cluster (o  $k$  clusters)
  - Divisivo
    - Empezar con un cluster que contenga todos los puntos
    - En cada paso, dividir un cluster en dos hasta que todo cluster contenga un solo punto (o haya  $k$  clusters)
- Requieren una definición de proximidad entre clusters.

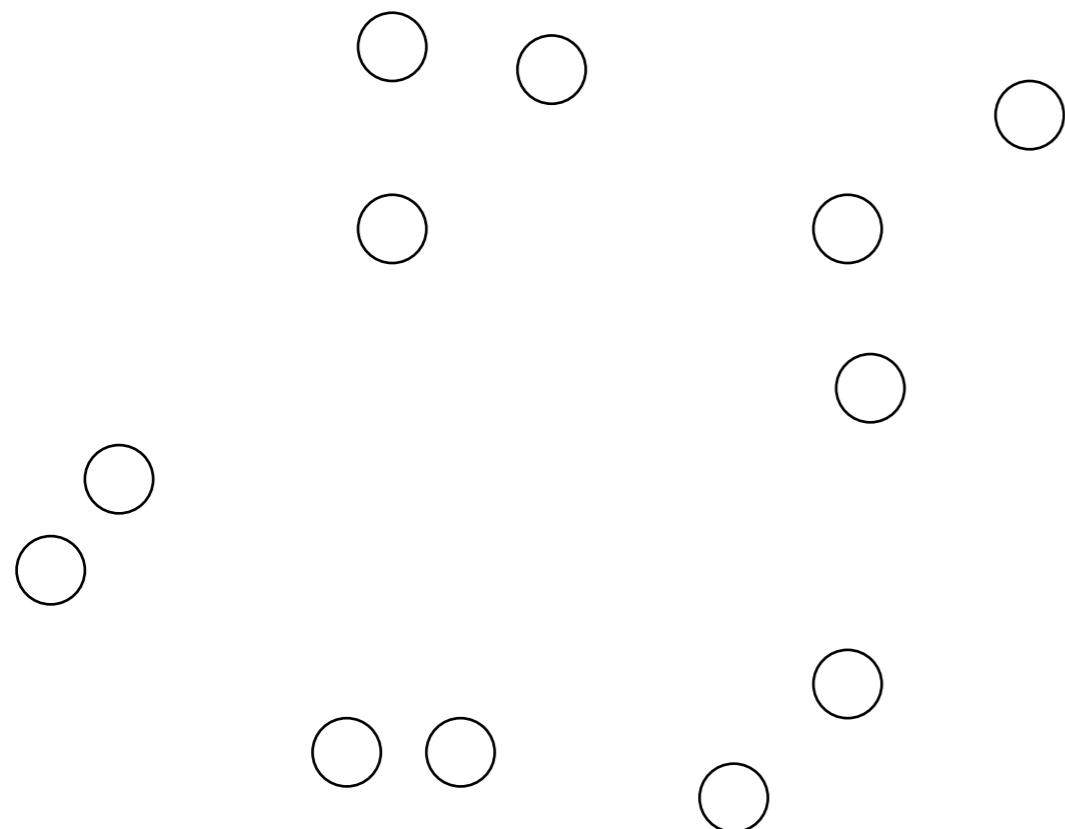
# Clustering Jerárquico Aglomerativo

- **Algoritmo básico (aglomerativo)**

1. Calcular matriz de distancias
2. Sea cada punto un cluster
3. **Repetir**
4. Mezclar par de clusters más cercano
5. Actualizar matriz de distancias
6. **Hasta** que quede sólo un cluster

# Clustering Jerárquico Aglomerativo

- Situación inicial: empezar con clusters de puntos individuales y la matriz de distancias



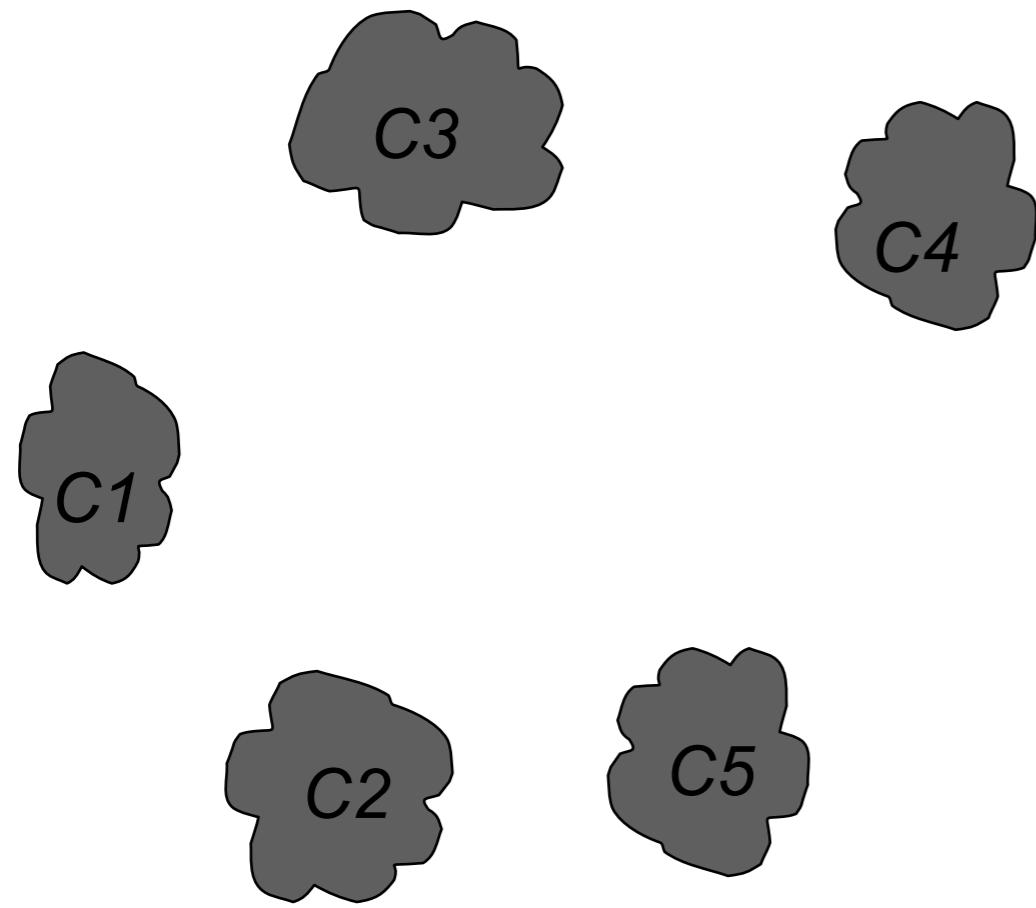
	$p1$	$p2$	$p3$	$p4$	$p5$	$\dots$
$p1$						
$p2$						
$p3$						
$p4$						
$p5$						
$\vdots$						

*Matriz de distancias*

$\bullet$   $p1$   $p2$   $p3$   $\dots$   $p9$   $p10$   $p11$   $p12$

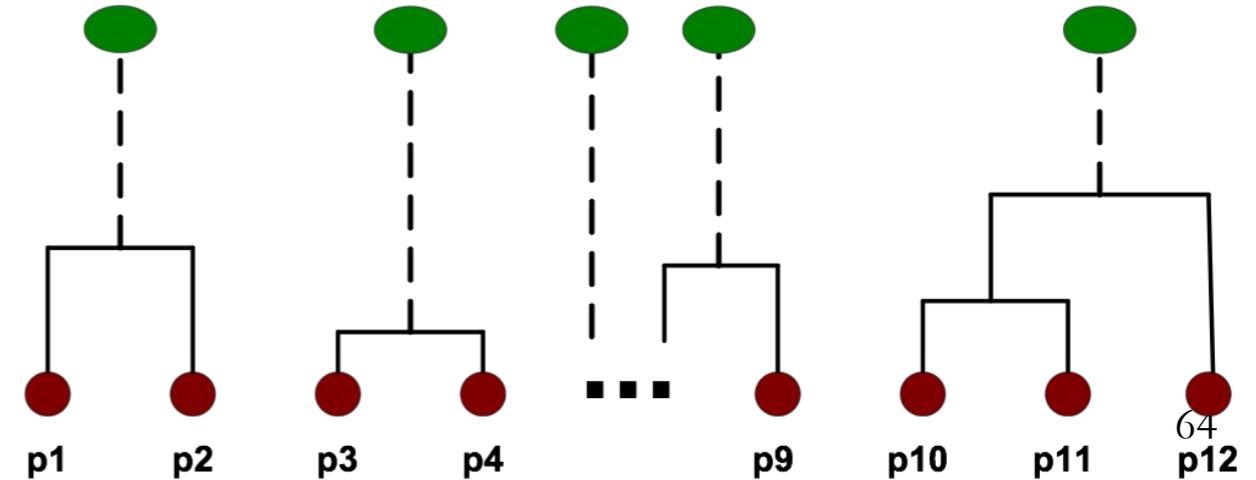
# Clustering Jerárquico Aglomerativo

- Después de un par de iteraciones...



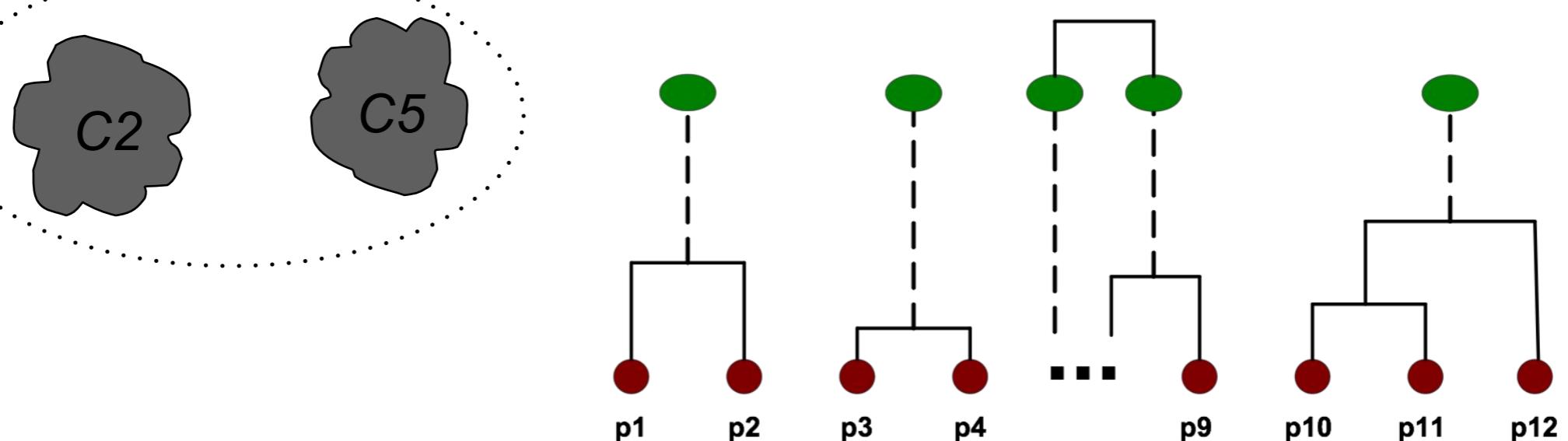
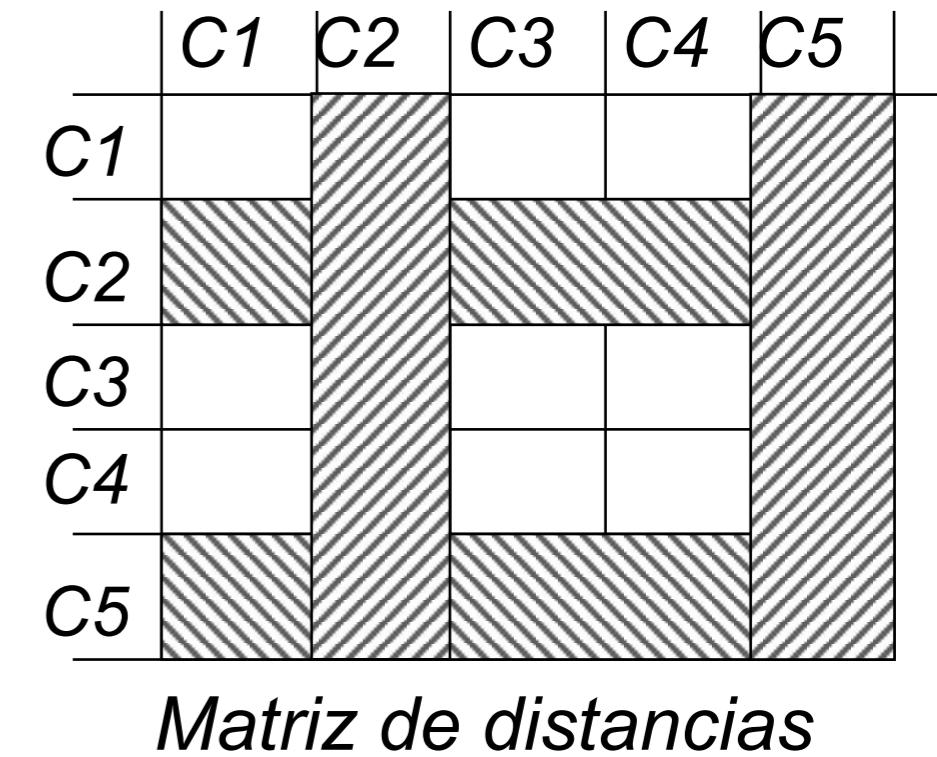
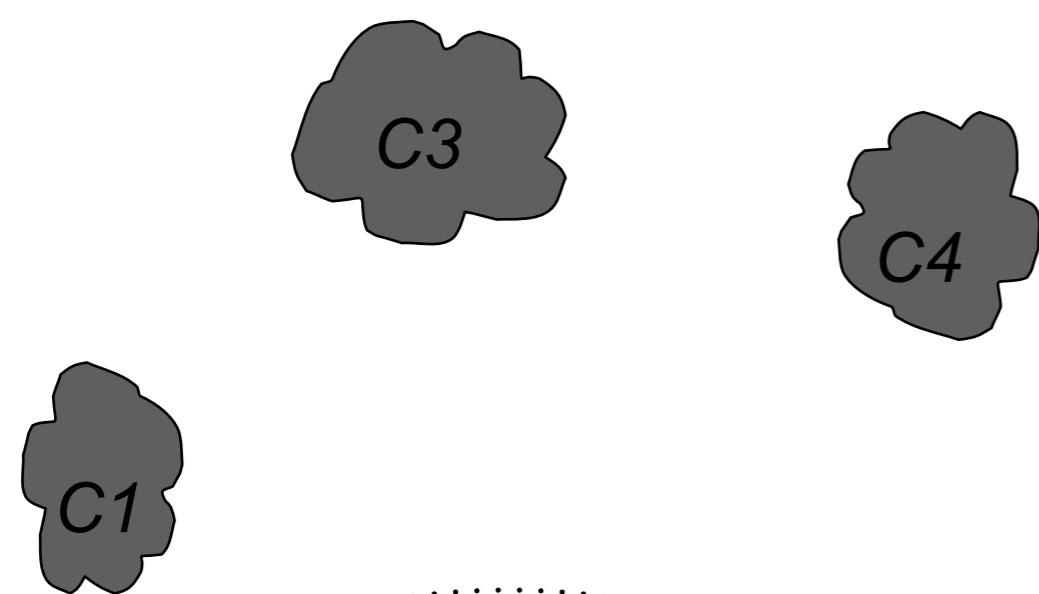
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

*Matriz de distancias*



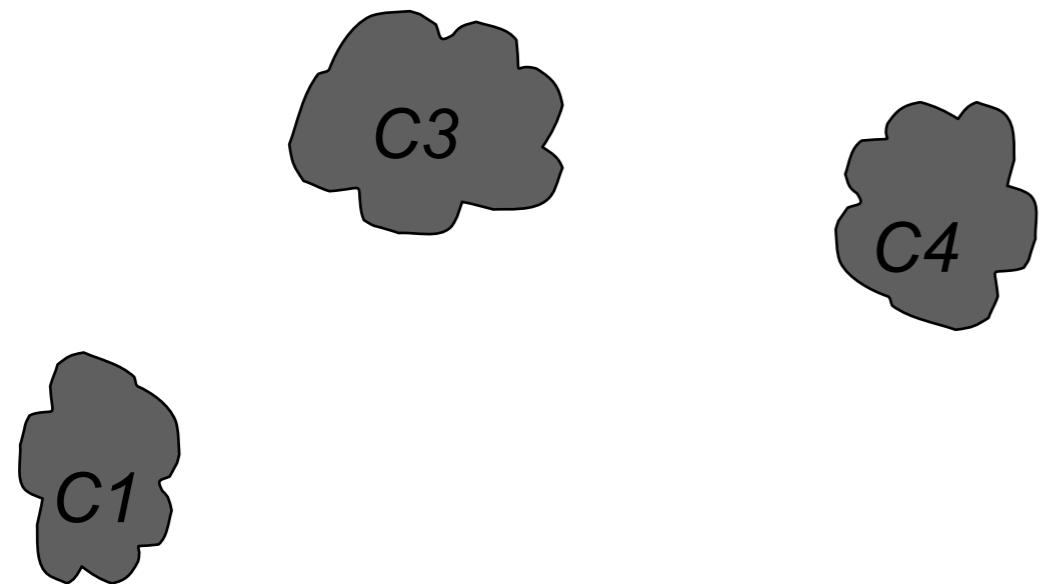
# Clustering Jerárquico Aglomerativo

- ... mezclar clusters más cercano (C2 y C5) y actualizar matriz de distancias



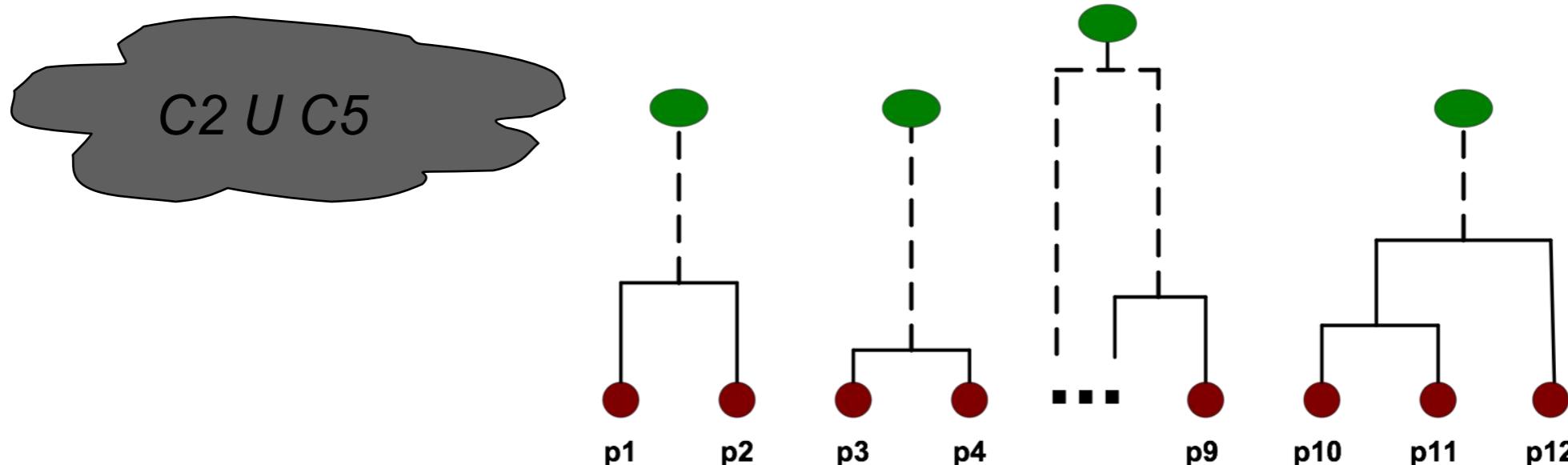
# Clustering Jerárquico Aglomerativo

- ¿Cómo actualizar matriz de distancias?



		$C_2$	$U$	$C_1$	$C_5$	$C_3$	$C_4$
$C_1$				?			
$C_2 \cup C_5$			?	?	?	?	?
$C_3$				?			
$C_4$				?			

Matriz de distancias

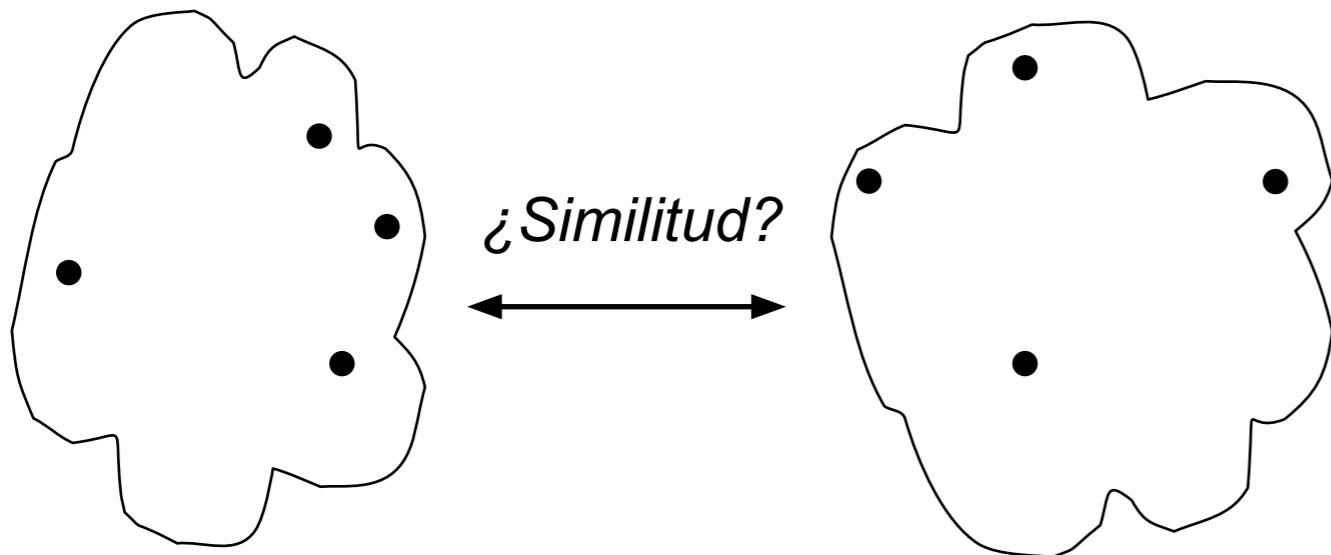


# Clustering Jerárquico Aglomerativo

- Operación clave: cálculo de la distancia entre clusters
  - Diferentes formas de hacerlo distinguen a los diferentes algoritmos
- Intuición: Sabemos como calcular la distancia entre dos puntos, pero ¿cómo calculamos la distancia entre dos clusters? o ¿entre un punto y un cluster?

# Clustering Jerárquico Aglomerativo

- ¿Cómo definir distancias entre clusters?



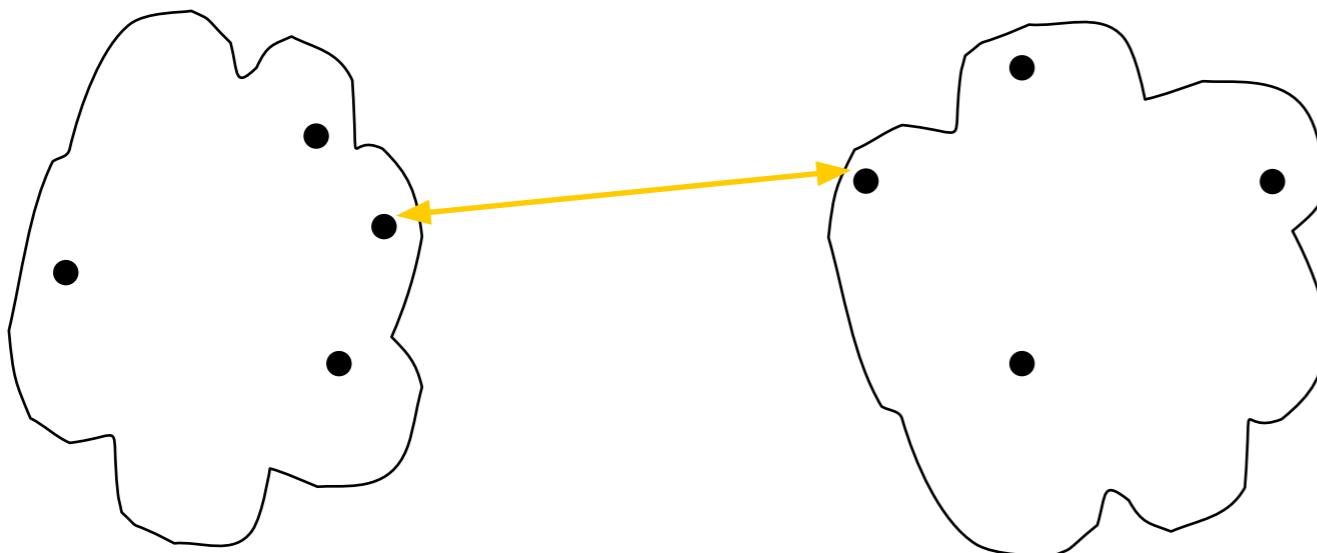
- MIN (single link)
- MAX (complete link)
- Promedio del grupo
- Distancia entre centroides

	$p_1$	$p_2$	$p_3$	$p_4$	$p_5$	...
$p_1$						
$p_2$						
$p_3$						
$p_4$						
$p_5$						
.						

*Matriz de distancias*

# Clustering Jerárquico Aglomerativo

- ¿Cómo definir distancias entre clusters?



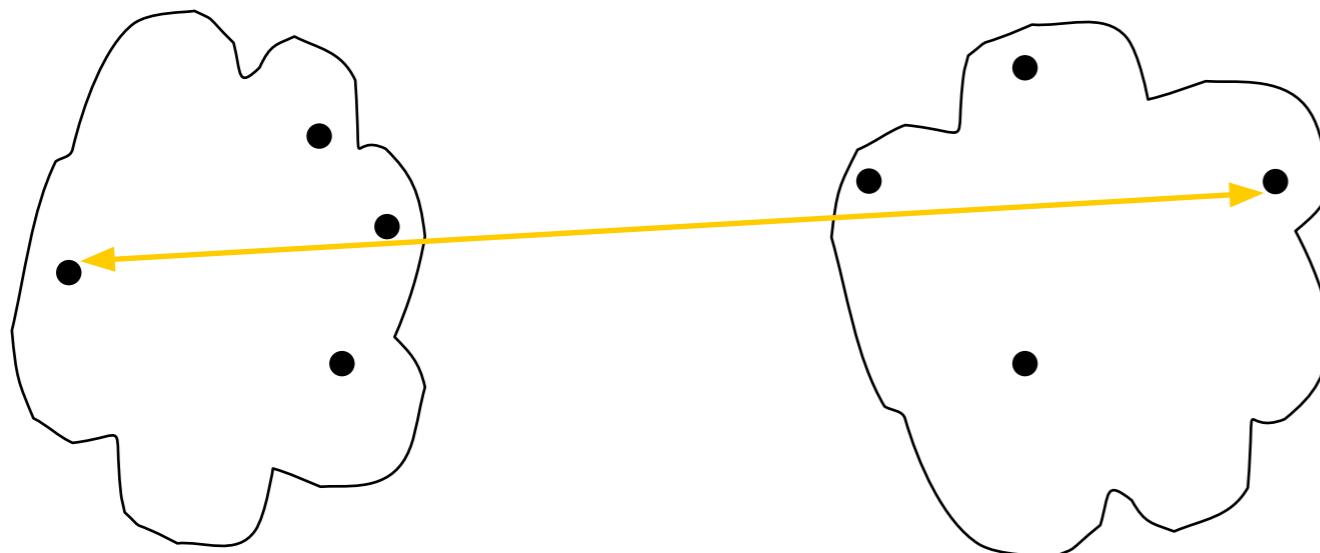
- MIN (single link)
  - Considero los dos puntos más cercanos entre sí (cada uno de un cluster distinto)

	$p_1$	$p_2$	$p_3$	$p_4$	$p_5$	$\dots$
$p_1$						
$p_2$						
$p_3$						
$p_4$						
$p_5$						
.						

*Matriz de distancias*

# Clustering Jerárquico Aglomerativo

- ¿Cómo definir distancias entre clusters?



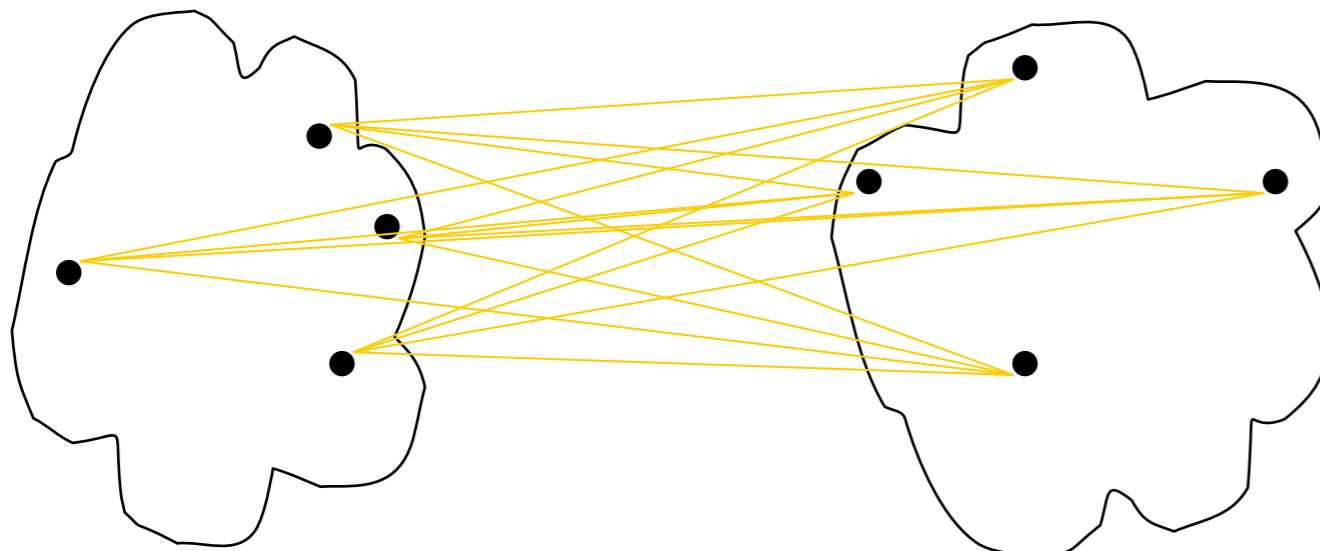
- MAX (complete link)
  - Considero los dos puntos más lejanos entre sí (cada uno de un cluster distinto)

	$p_1$	$p_2$	$p_3$	$p_4$	$p_5$	$\dots$
$p_1$						
$p_2$						
$p_3$						
$p_4$						
$p_5$						
.						

*Matriz de distancias*

# Clustering Jerárquico Aglomerativo

- ¿Cómo definir distancias entre clusters?



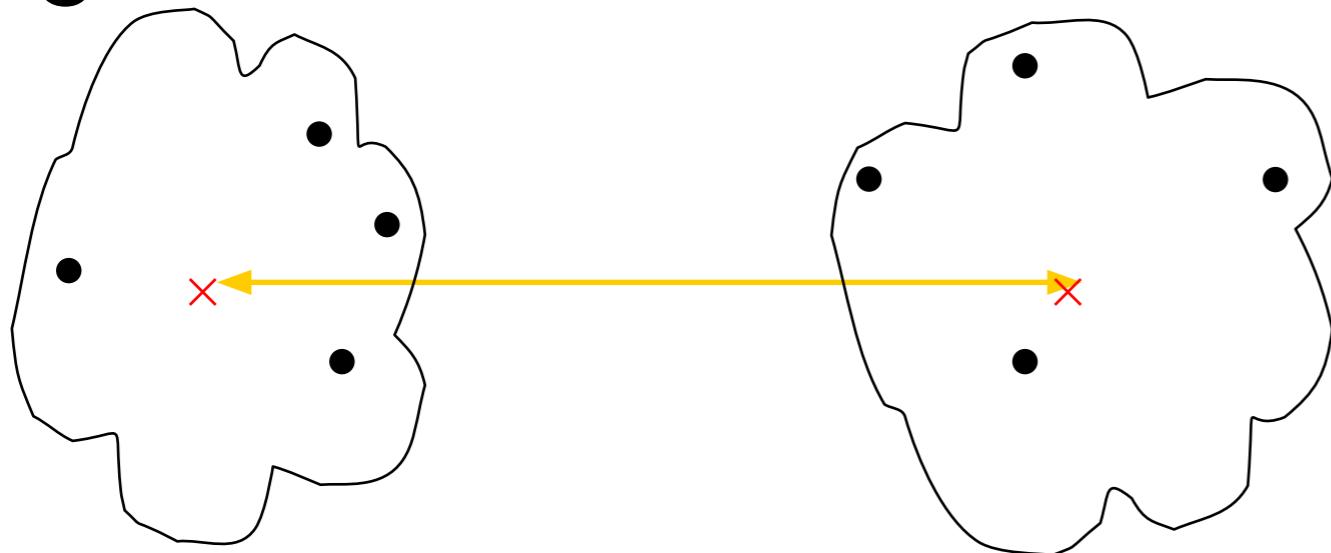
- Promedio del grupo
  - Distancia promedio de todos los pares de puntos (cada par tiene un punto por cluster)

	$p_1$	$p_2$	$p_3$	$p_4$	$p_5$	$\dots$
$p_1$						
$p_2$						
$p_3$						
$p_4$						
$p_5$						
.						

*Matriz de distancias*

# Clustering Jerárquico Aglomerativo

- ¿Cómo definir distancias entre clusters?



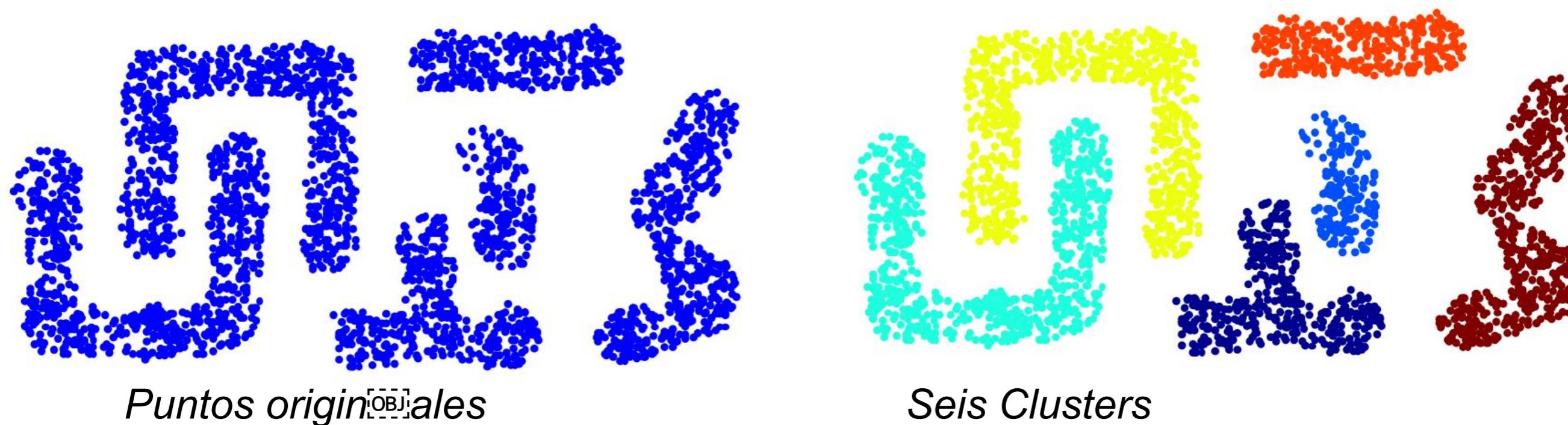
- Distancia entre centroides
  - distancia entre los centroides de cada grupo

	$p_1$	$p_2$	$p_3$	$p_4$	$p_5$	$\dots$
$p_1$						
$p_2$						
$p_3$						
$p_4$						
$p_5$						
.						

*Matriz de distancias*

# Clustering Jerárquico Aglomerativo

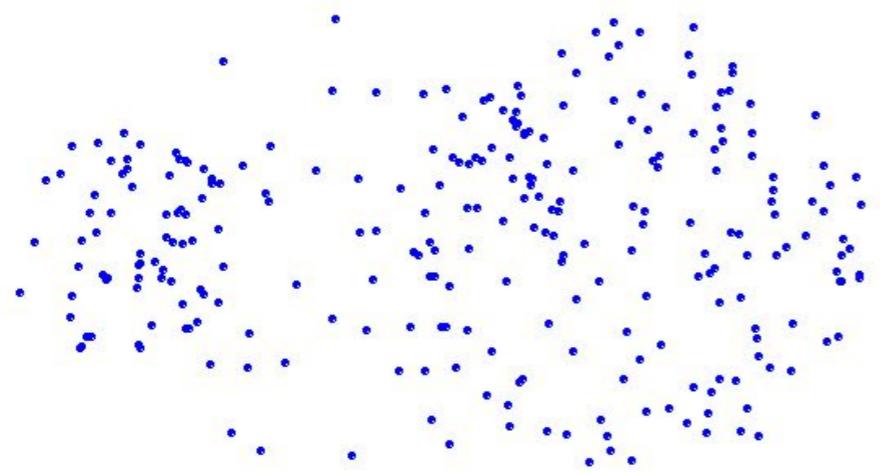
- Fortaleza de distancia MIN



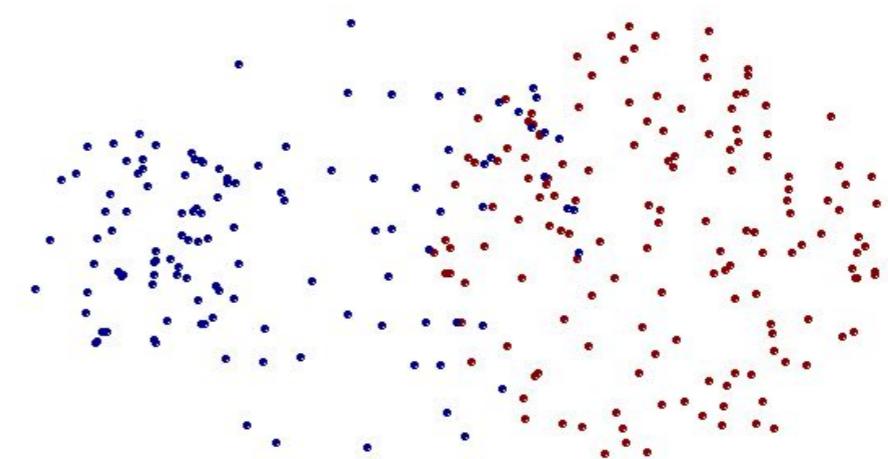
- Puede manejar formas no-elípticas

# Clustering Jerárquico Aglomerativo

- Limitaciones de distancia MIN



*Puntos originales*

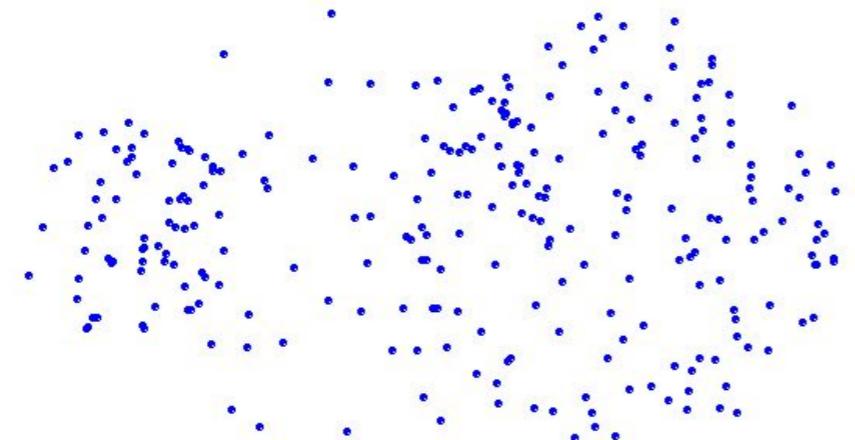


*Dos Clusters*

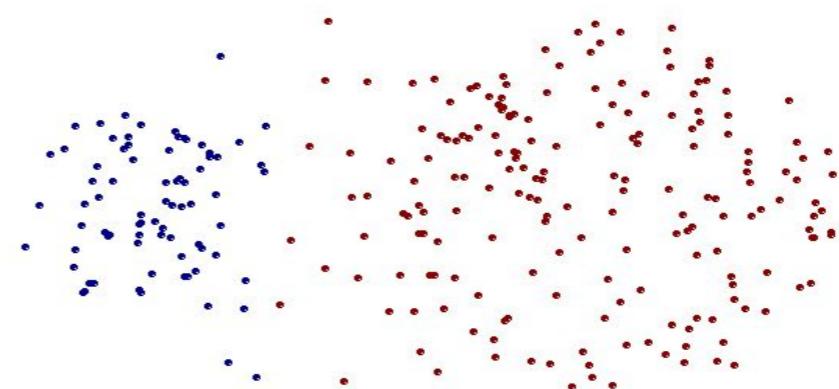
- *Sensible a ruido y outliers*

# Clustering Jerárquico Aglomerativo

- Fortaleza de MAX



*Puntos originales*

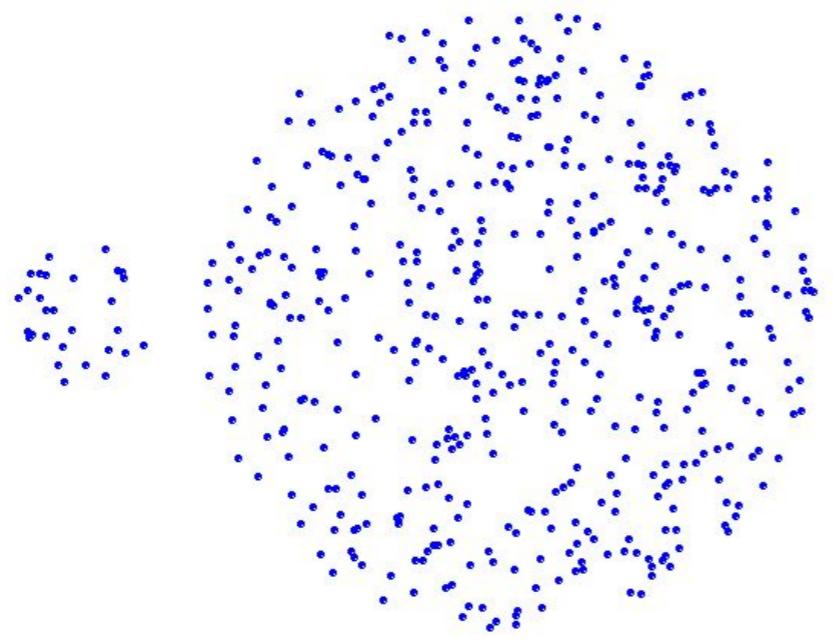


*Dos Clusters*

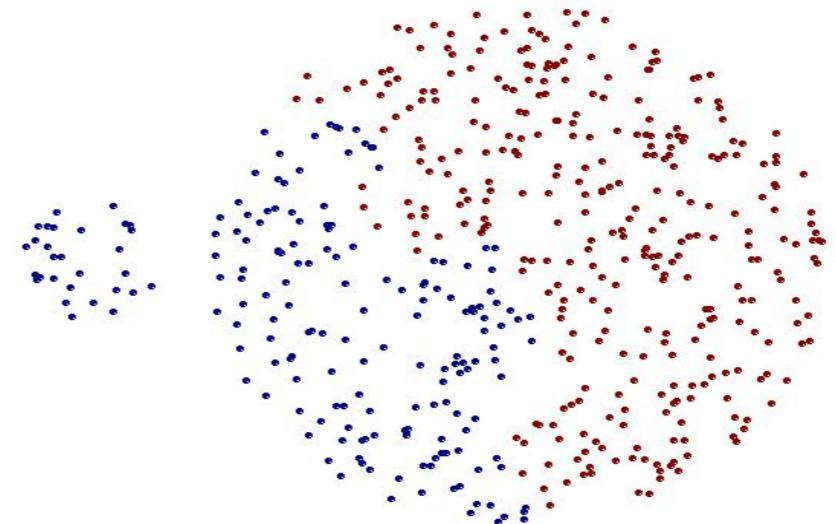
- *Menos susceptible a ruido y outliers*

# Clustering Jerárquico Aglomerativo

- Limitaciones de MAX



*Puntos originales*



*Dos Clusters*

- *Tiende a quebrar clusters grandes*
- *Sesgado a clusters esféricos*

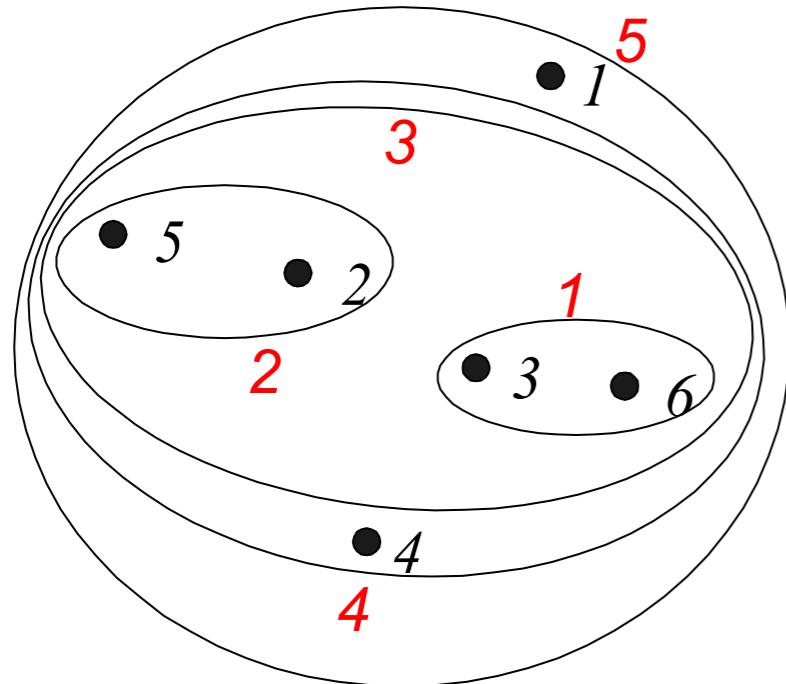
# Clustering Jerárquico Aglomerativo

- Distancia promedio de grupo
  - Compromiso entre MIN y MAX
  - Fortalezas
    - Menos susceptible a ruido y outliers
  - Limitaciones
    - Sesgado a clusters esféricos

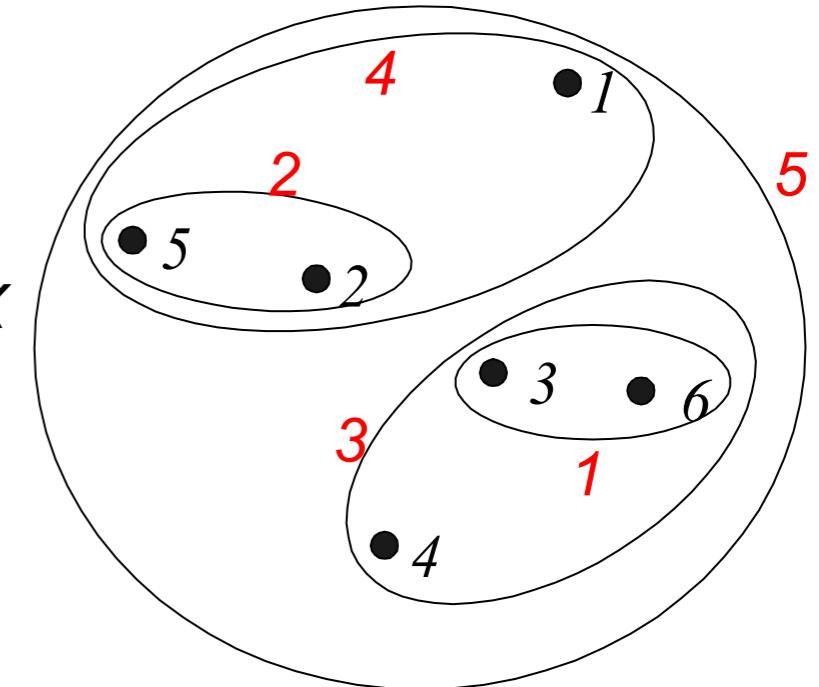
# Clustering Jerárquico Aglomerativo

- Método de Ward
  - Similitud entre clusters se basa en el incremento del SSE cuando se mezclan dos clusters
    - Similar a distancia promedio de grupo si la distancia entre puntos es distancia cuadrada
  - Menos susceptible a ruido y outliers
  - Sesgado a clusters esféricos

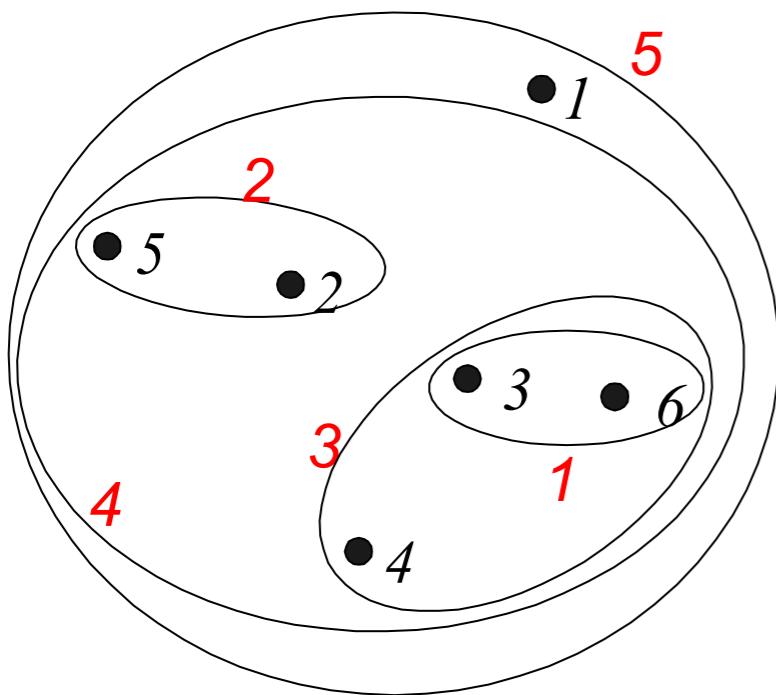
# Clustering Jerárquico Aglomerativo



*MIN*

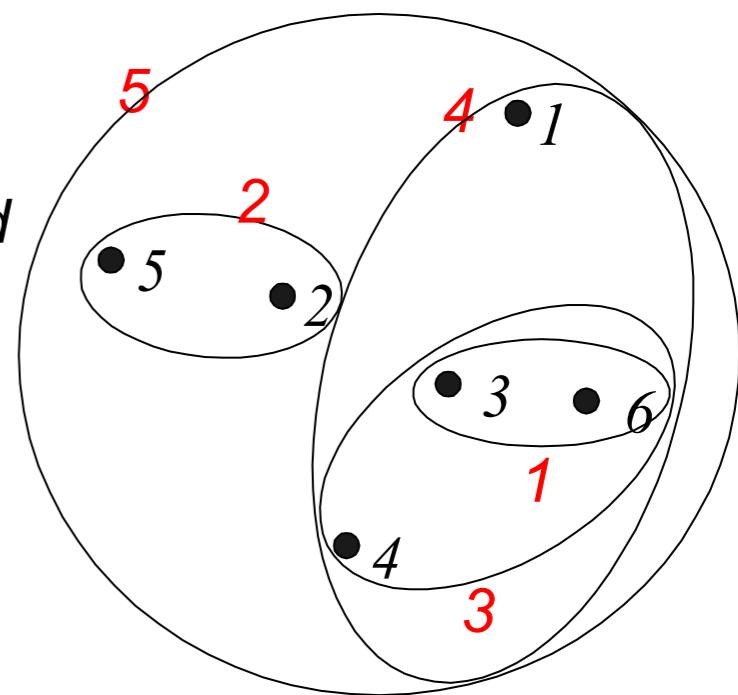


*MAX*



*Promedio de  
grupo*

*Método de Ward*



# Clustering Jerárquico Aglomerativo

- Requerimientos de tiempo y espacio
  - Espacio:  $O(N^2)$  para guardar matriz de distancias
    - $N$ : número de puntos
  - Tiempo:  $O(N^3)$  en muchos casos
    - Para  $N$  pasos, se debe actualizar matriz de similitud en cada paso
    - Complejidad puede reducirse a  $O(N^2 \log N)$  usando listas ordenadas o heaps

# Clustering Jerárquico Aglomerativo

- Problemas y limitaciones
  - Una vez se ha decidido unir dos clusters, no se puede deshacer
  - No hay una función objetivo que sea directamente minimizada
  - Problemas de los diferentes esquemas:
    - Sensibles a ruido y outliers
    - Dificultad para manejar clusters de distinto tamaño
    - Pueden romper clusters grandes

# Algoritmos de Clustering

- K-means
- Clustering jerárquico aglomerativo
- **DBSCAN**

# DBSCAN

Algoritmo de clustering basado en densidad

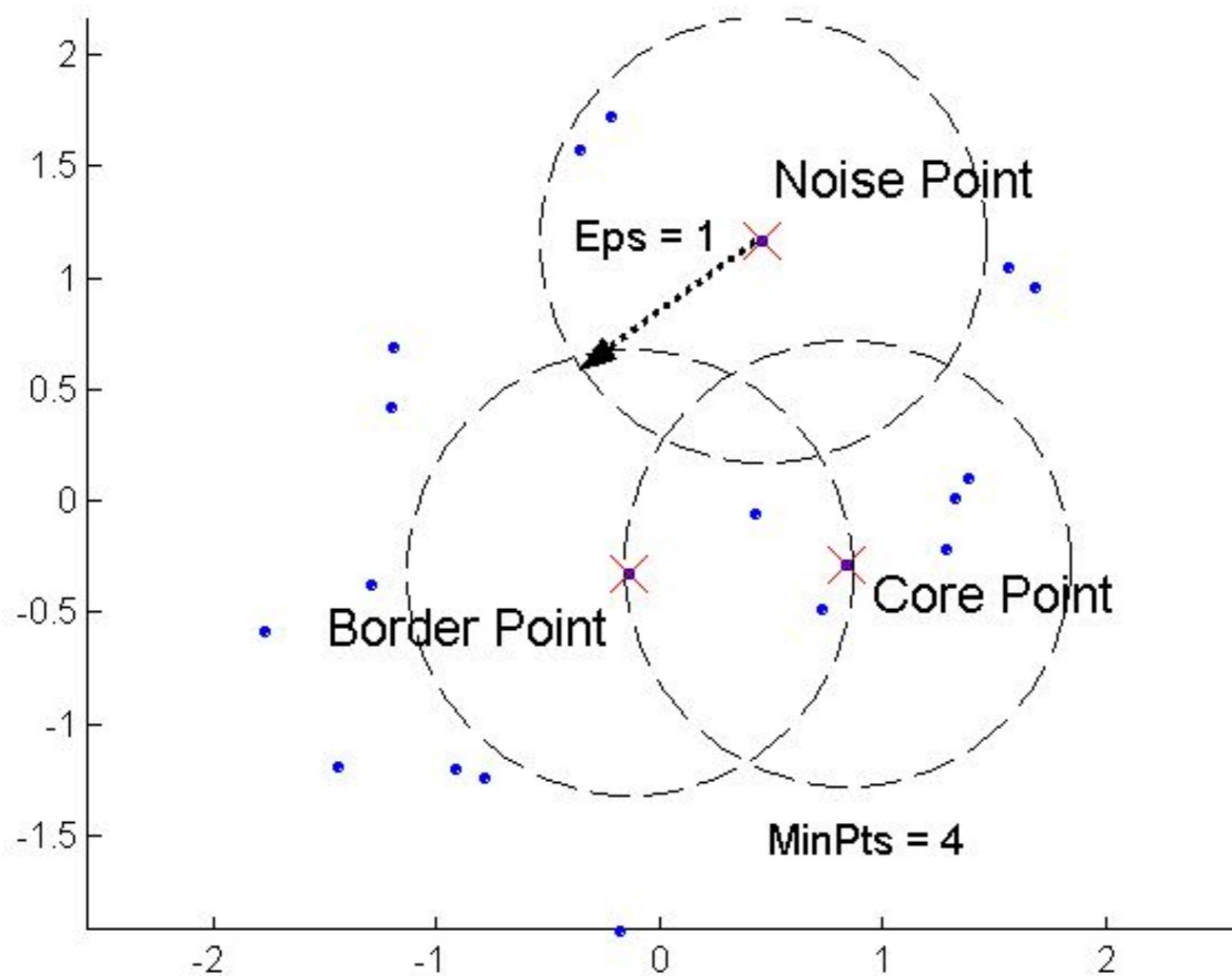
**Idea:** encontrar regiones de alta densidad de puntos separado por regiones de baja densidad.

- Las regiones densas corresponden a los clusters.
- La densidad de un punto es el número de puntos que tiene dentro de un radio dado.

# DBSCAN

- Parámetros:
  - 1) **Eps**: radio especificado
  - 2) **MinPts**: número mínimo de puntos en una región.
- Tipos de punto:
  - Punto “core”: punto con más puntos que MinPts a distancia Eps
    - Éstos son los puntos dentro del cluster
  - Punto “border”: tiene menos que MinPts puntos en el radio Eps, pero está en la vecindad de un punto core.
  - Punto “noise”: cualquier punto que no sea core ni border.

# DBSCAN



# DBSCAN

- Cualquier par de puntos core que tengan una distancia entre sí menor que Eps son asignados al mismo cluster.
- Cualquier punto border que esté a una distancia menor que Eps de un punto core **pc** se le asigna el cluster de **pc**.
  - Hay que definir una estrategia cuando el punto border está cerca de dos puntos core de distinto cluster.
- Eliminar los puntos de ruido.

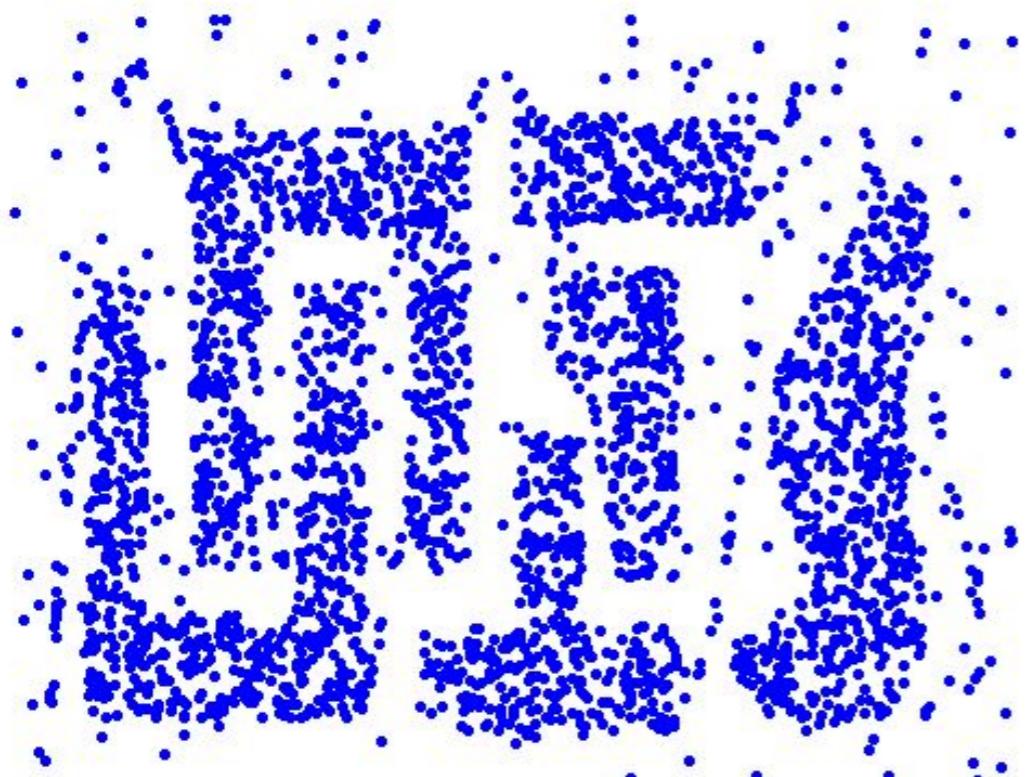
# DBSCAN

---

**Algorithm 7.5** DBSCAN algorithm.

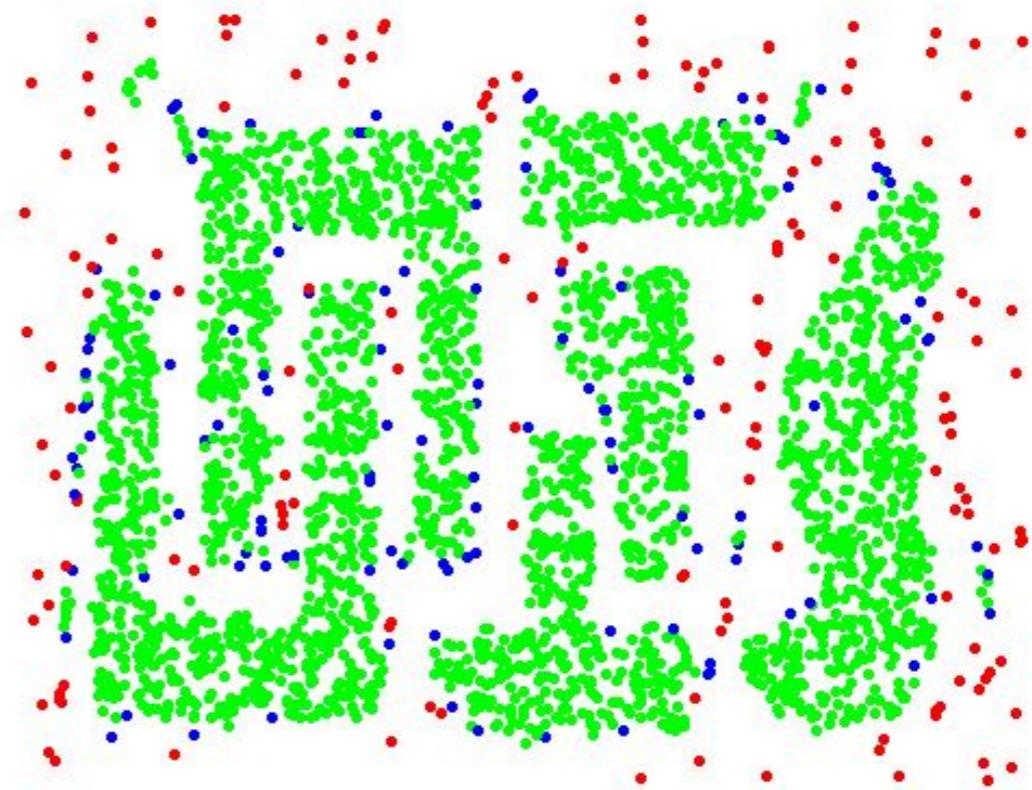
- 
- 1: Label all points as core, border, or noise points.
  - 2: Eliminate noise points.
  - 3: Put an edge between all core points within a distance  $Eps$  of each other.
  - 4: Make each group of connected core points into a separate cluster.
  - 5: Assign each border point to one of the clusters of its associated core points.
-

# DBSCAN



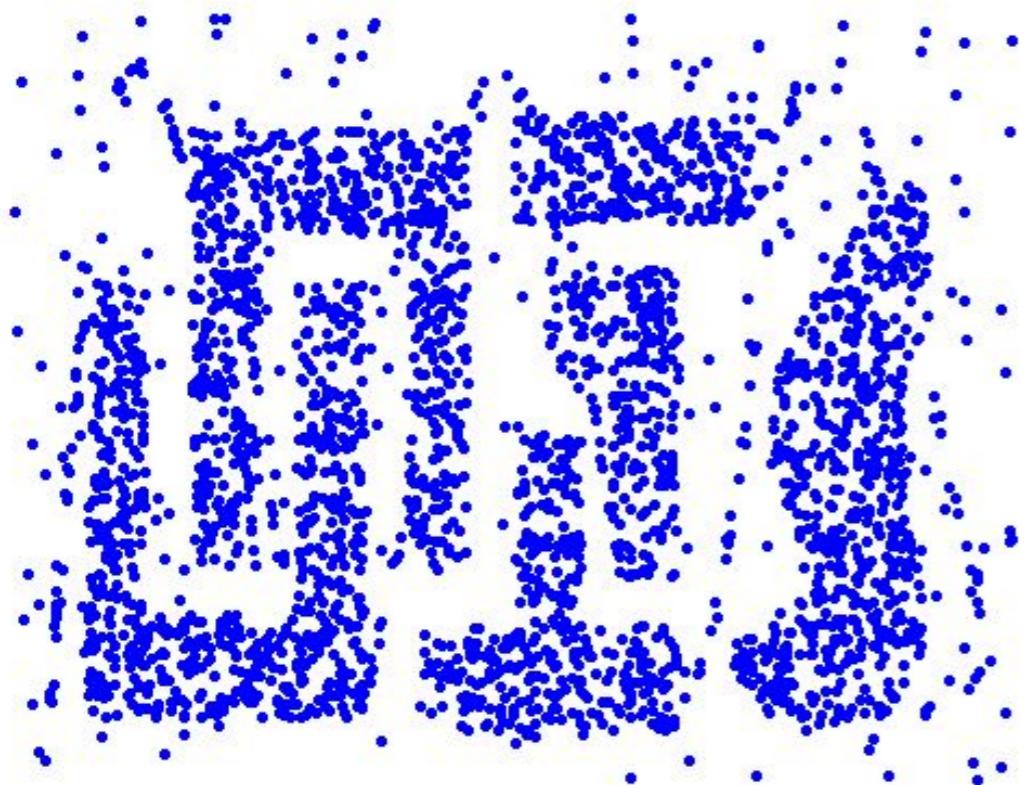
*Puntos originales*

$Eps = 10, MinPts = 4$

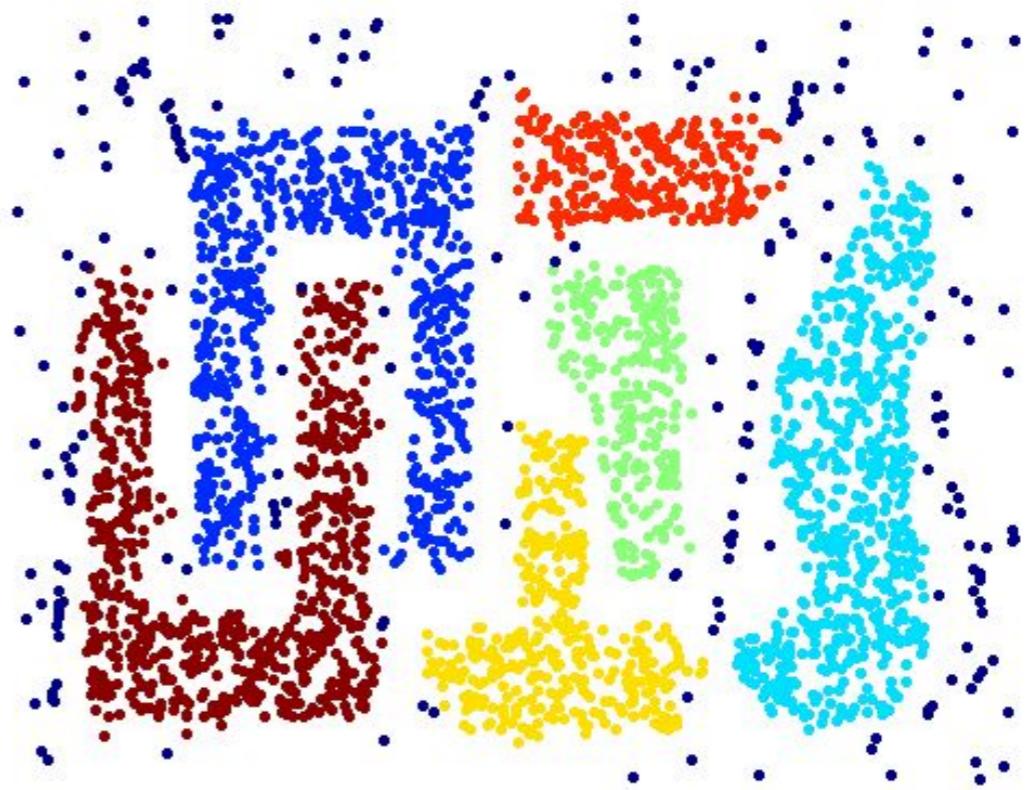


*Tipos de punto: core,  
border y noise*

# DBSCAN



*Puntos originales*

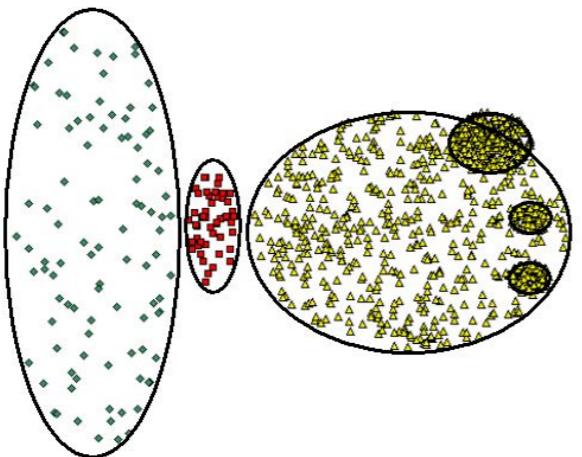


*Clusters*

- *Resistente a ruido*
- *Puede encontrar clusters de diferentes formas y tamaños*

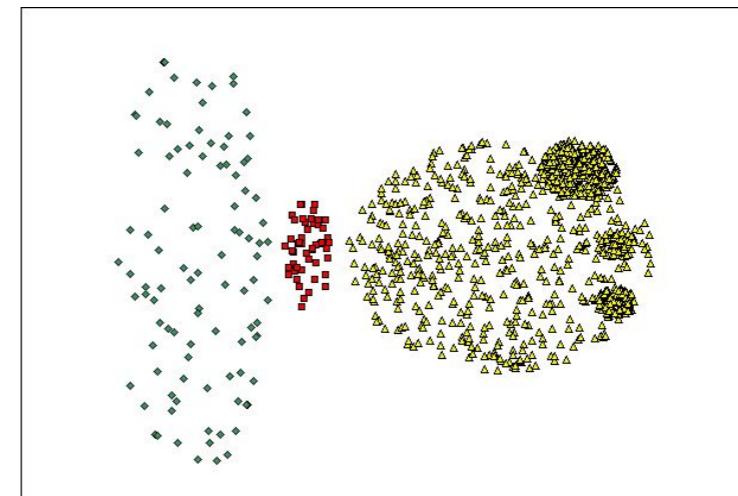
# DBSCAN

- No funciona bien en:

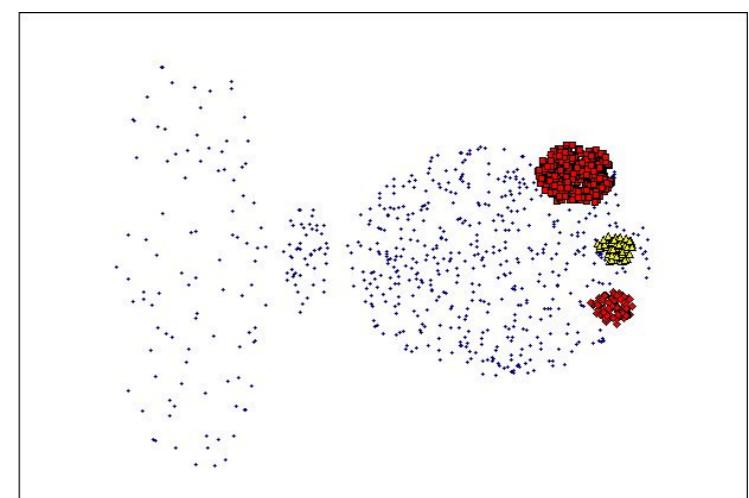


*Puntos originales*

- *Densidades variables:*  
*clusters de baja densidad son confundidos con ruido.*
- *Datos de alta dimensionalidad: definición de densidad se vuelve compleja.*



$(MinPts=4,$   
 $Eps=9.92)$



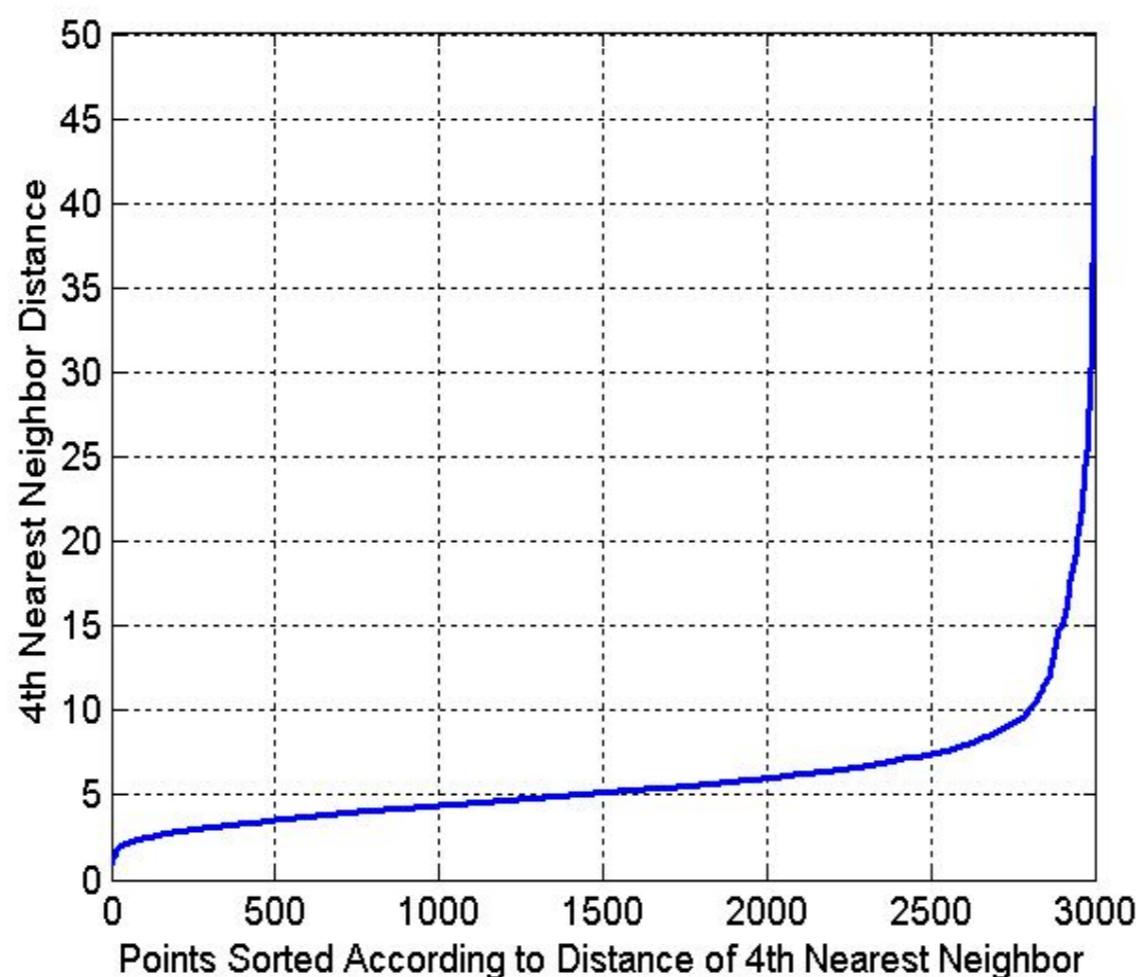
$(MinPts=4,$   
 $Eps=9.75).$

# Método de la Rodilla para Determinar Eps y MinPts

- Análisis del Comportamiento de k-dist
  - Evaluamos la distancia de un punto a su k-ésimo vecino más cercano (k-dist).
  - Para puntos dentro de un clúster, k-dist será pequeño siempre y cuando k no exceda el tamaño del clúster.
  - Para puntos fuera de un clúster (puntos de ruido), k-dist será alto.

# Técnica de la Rodilla

- Calcular k-dist para todos los puntos con un valor fijo de k.
- Ordenar los valores de k-dist de manera creciente.
- Graficar k-dist vs. la cantidad de puntos.



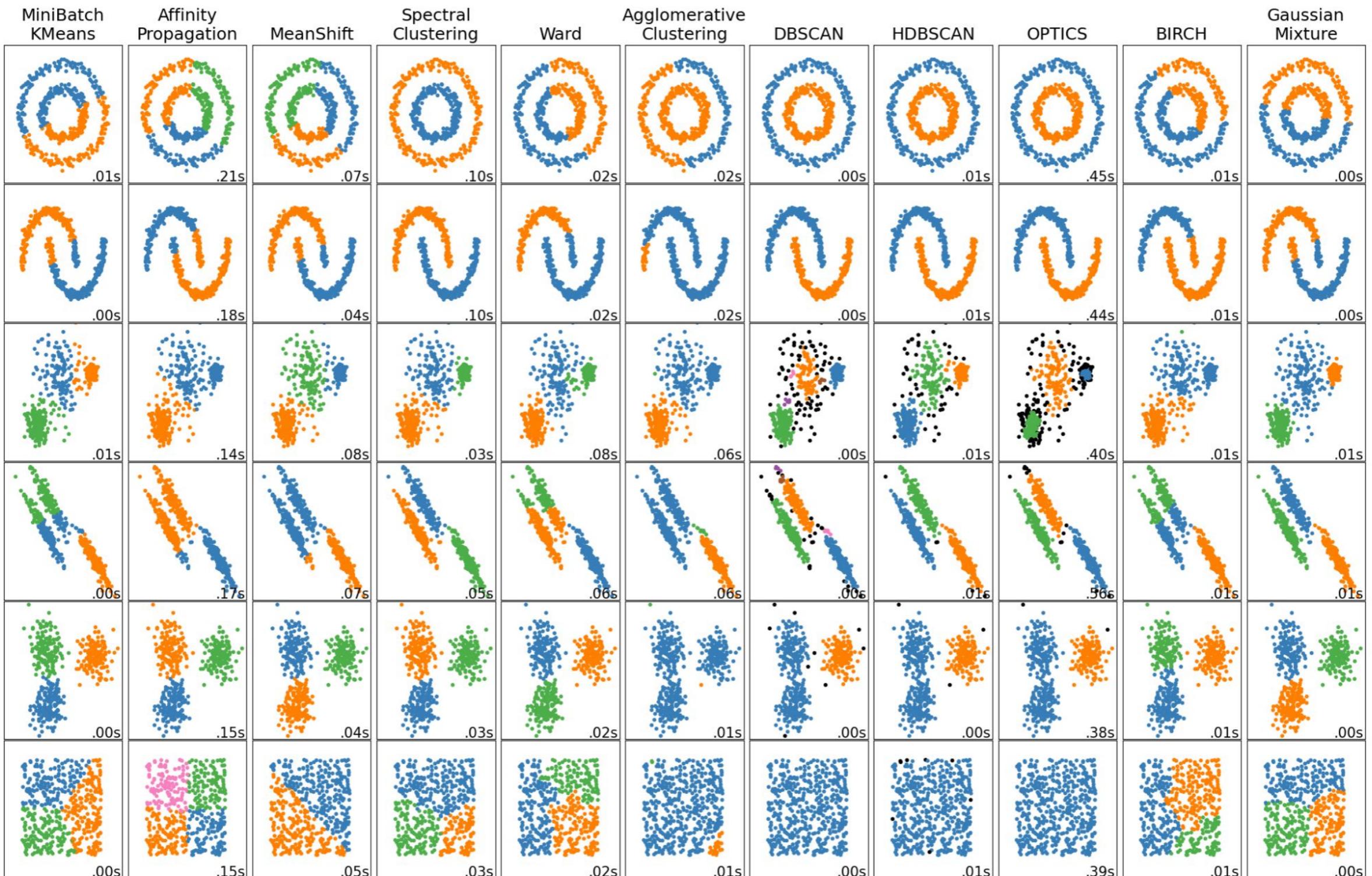
# Técnica de la Rodilla

- Identificar el "salto" en la gráfica de k-dist (10 en la figura)
- El punto donde ocurre el salto se utiliza para determinar el valor de Eps.
- MinPts es el valor de k utilizado en el análisis de k-dist.

# Otras técnicas

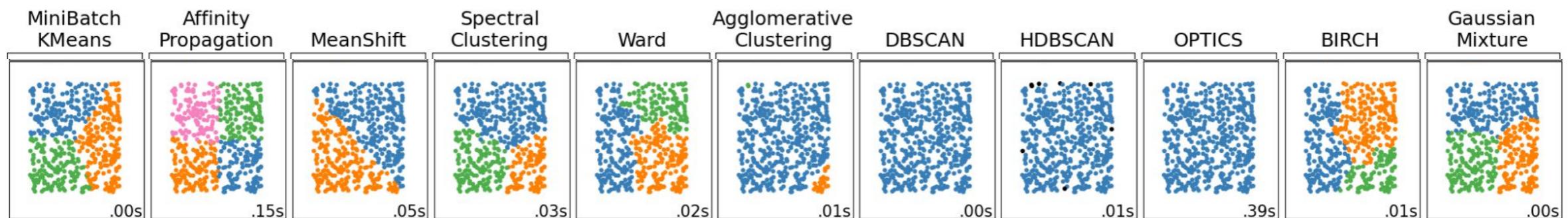
- Mapas auto-organizados (SOM) o redes de Kohonen
- Fuzzy C-means
- Mezcla de Gaussianas y algoritmo EM

# Otras técnicas



Fuente: [2.3. Clustering — scikit-learn 1.5.2 documentation](#)

# ¿Cómo saber si lo hacemos bien?



- Con datos aleatorios, sin ninguna estructura, igual podemos generar clústers
- Debemos validar de algún modo
- **Próxima clase**



**dcc**

CIENCIAS DE LA COMPUTACIÓN  
UNIVERSIDAD DE CHILE

[www.dcc.uchile.cl](http://www.dcc.uchile.cl)

f o in t / DCCUCHILE