



APRENDIZAJE NO SUPERVISADO

APRENDIZAJE DE MÁQUINA - CEIA - FIUBA

Dr. Ing. Facundo Adrián Lucianna

Esp. Lic. María Carina Roldán

REPASO CLASE ANTERIOR

- Calibración de modelos
- Medición de la calibración
 - Reliability Diagram o Curva de calibración
 - Brier Score
- Métodos de calibración
 - Platt Scaling
 - Regresión isotónica

CALIBRACIÓN DE MODELOS

¿Qué es la calibración?

Pensemos en una clasificación binaria.

Decimos que el modelo está calibrado si: “Cuando el modelo predice ‘0.7’, el evento realmente ocurre el 70% de las veces.”

La **calibración NO** evalúa si el modelo clasifica bien. La calibración evalúa si **las probabilidades son confiables**.

CALIBRACIÓN DE MODELOS

La **calibración** mide si las probabilidades reflejan frecuencias reales.

Si predice 0,7 ¿ocurre realmente el 70% de las veces?

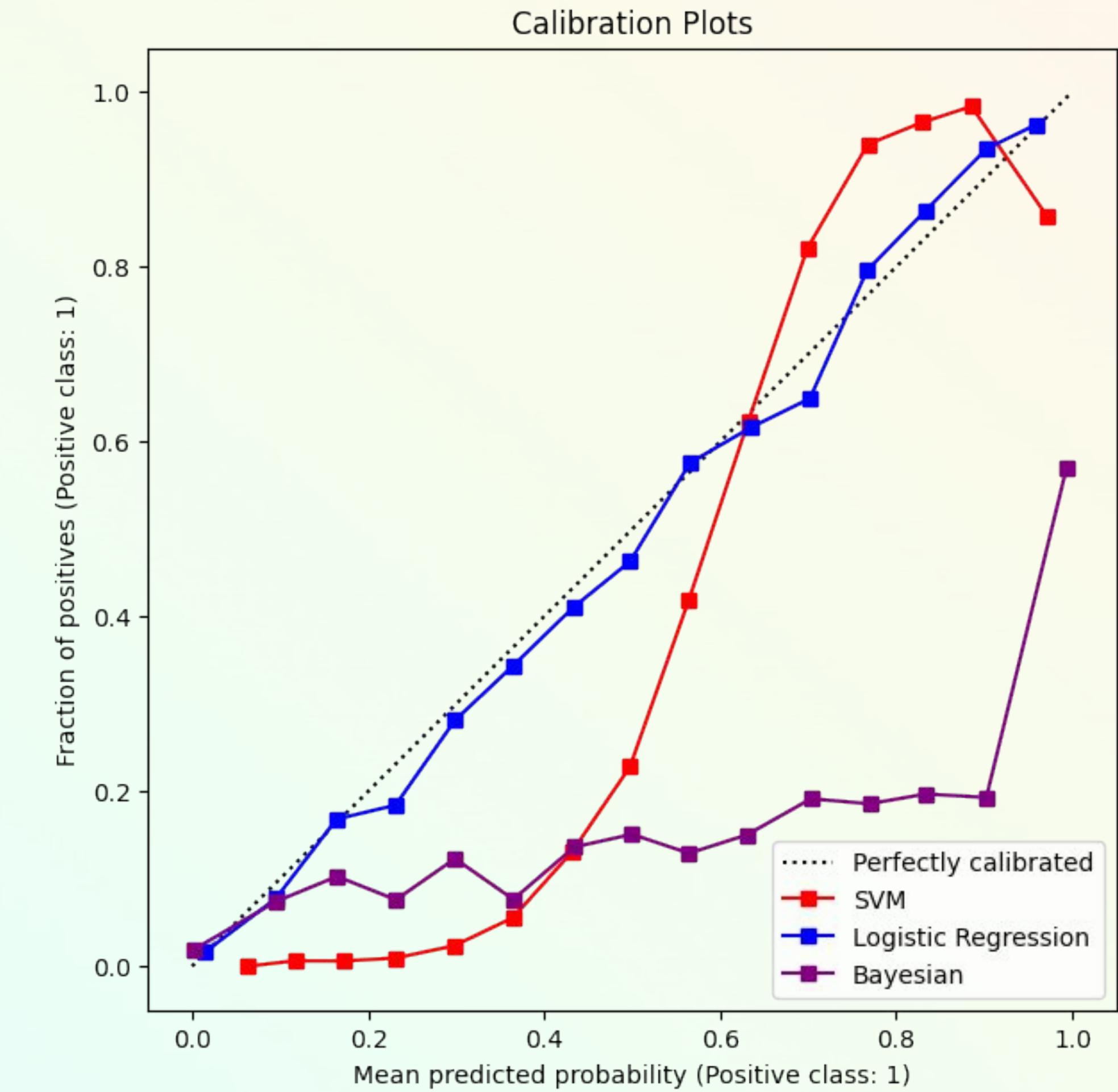
Un modelo con buena discriminación ordena bien, pero no necesariamente está calibrado, es decir, que ordene bien no significa que prediga probabilidades confiables.

Ejemplo: Predice 0,99 para los positivos, pero si los verdaderos positivos ocurren solo el 40% de las veces, está ***mal calibrado***.

MEDICIÓN DE LA CALIBRACIÓN

Para medir la calibración tenemos principalmente dos herramientas.

Reliability Diagram o Curva de calibración: es un gráfico que compara la probabilidad predicha y la frecuencia real. Si la línea está sobre la **diagonal**, significa que la calibración es perfecta. Las desviaciones indican **sobreestimación** o **subestimación**.



MEDICIÓN DE LA CALIBRACIÓN

Para medir la calibración tenemos principalmente dos herramientas.

Brier Score: Es una métrica que mide qué tan lejos están las probabilidades de la realidad. Va de 0 a 1, donde 0 es la calibración y precisión perfecta y los valores más altos indican peor desempeño.

$$\text{Brier Score} = \frac{1}{N} \sum_{i=1}^N (p_i - y_i)^2$$

- N: Número total de observaciones en el set de datos.
- p_i : La probabilidad predicha por el modelo para la instancia i.
- y_i : El resultado real (label) de la instancia i.

No es buena idea comparar el Brier Score entre modelos diferentes dado que si cambia la capacidad de discriminación, tendremos valores diferentes.

MÉTODOS DE CALIBRACIÓN

Cómo solucionamos los casos de modelos mal calibrados?

Pues construyendo otro modelo que mapee las probabilidades. Un modelo de una sola dimensión:

$$f_{\text{calib}}(s(X)) \approx P(y)$$

Donde: $s(X)$ es el score dado por el modelo. Es decir, podemos usarlos en modelos que no dan salida probabilísticas.

Para entrenar a f_{calib} se puede:

- Usar el set de entrenamiento, pero vamos a overfittear.
- Usar un set distinto al de entrenamiento y evaluación.
- Se puede entrenar usando **validación cruzada**. Más lento pero asegura evitar overfitting.

MÉTODOS DE CALIBRACIÓN

Platt Scaling

Que pasa si ajustamos una regresión logística a la salida del modelo? Transformamos la salida cruda del modelo (score) en una probabilidad asumiendo que la distribución tiene forma de Sigmoide (S).

Entrenamos una regresión logística auxiliar para encontrar los parámetros A (pendiente) y B (sesgo)

$$P(y = 1|X) = \frac{1}{1 + \exp(A \cdot s(X) + B)}$$

Se optimiza por Máxima Verosimilitud (minimizando la Log Loss), igual que una regresión logística estándar.

Caso de Uso

- Funciona excelentemente en modelos de "máximo margen" como SVM.
- Convierte distancias arbitrarias en una probabilidad acotada [0,1].

MÉTODOS DE CALIBRACIÓN

Regresión isotónica

En este caso no asumimos ninguna forma funcional (como una sigmoide). Solo asumimos que si el score del modelo sube, la probabilidad real no debe bajar. Es decir, usamos un modelo **no paramétrico**.

Aprende una función escalonada arbitraria que debe cumplir una única regla: ser monótonamente creciente (nunca baja, solo sube o se mantiene).

- Zonas constantes: Agrupa rangos de scores donde la probabilidad empírica es similar.
- Saltos: Cambia de nivel para ajustarse a los datos.

Para entrenarse se usa el algoritmo **Pool Adjacent Violators Algorithm**: Si un punto tiene una probabilidad menor que el anterior (violando la monotonía), los agrupa y promedia hasta que la curva sea creciente. Se busca minimizar:

$$\text{Minimizar: } \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad \text{Sujeto a: } \hat{y}_1 \leq \hat{y}_2 \leq \dots \leq \hat{y}_n$$

APRENDIZAJE NO SUPERVISADO

APRENDIZAJE NO SUPERVISADO

Como vimos en la clase 1, en aprendizaje no supervisado tenemos un dataset de n observaciones con p atributos, pero no tenemos una variable Y de respuesta.

Esto nos genera un desafío...

APRENDIZAJE NO SUPERVISADO

Como vimos en la clase 1, en aprendizaje no supervisado tenemos un dataset de n observaciones con p atributos, pero no tenemos una variable Y de respuesta.

Esto nos genera un desafío...

No tenemos un **objetivo simple** para el análisis.

Es difícil evaluar los resultados obtenidos, ya que no existe un mecanismo aceptado para realizar una **validación cruzada** o **validar** los resultados en un conjunto de datos independiente.

APRENDIZAJE NO SUPERVISADO

Como vimos en la clase 1, en aprendizaje no supervisado tenemos un dataset de n observaciones con p atributos, pero no tenemos una variable Y de respuesta.

Esto nos genera un desafío...

No tenemos un **objetivo simple** para el análisis.

Es difícil evaluar los resultados obtenidos, ya que no existe un mecanismo aceptado para realizar una **validación cruzada** o **validar** los resultados en un conjunto de datos independiente.

Esto se debe justamente a que no tenemos con qué comparar.

APRENDIZAJE NO SUPERVISADO

Entonces, para que usaríamos este tipo de aprendizaje?

Lo que buscamos en aprendizaje no supervisado es descubrir **una estructura sobre la base del conjunto de datos**. A diferencia de los problemas de aprendizaje supervisado, que buscamos predecir un resultado.

En un caso particular, es cuando queremos encontrar en los datos conjuntos que responden de una forma similar, es decir agrupamientos que *a priori* no son evidentes.

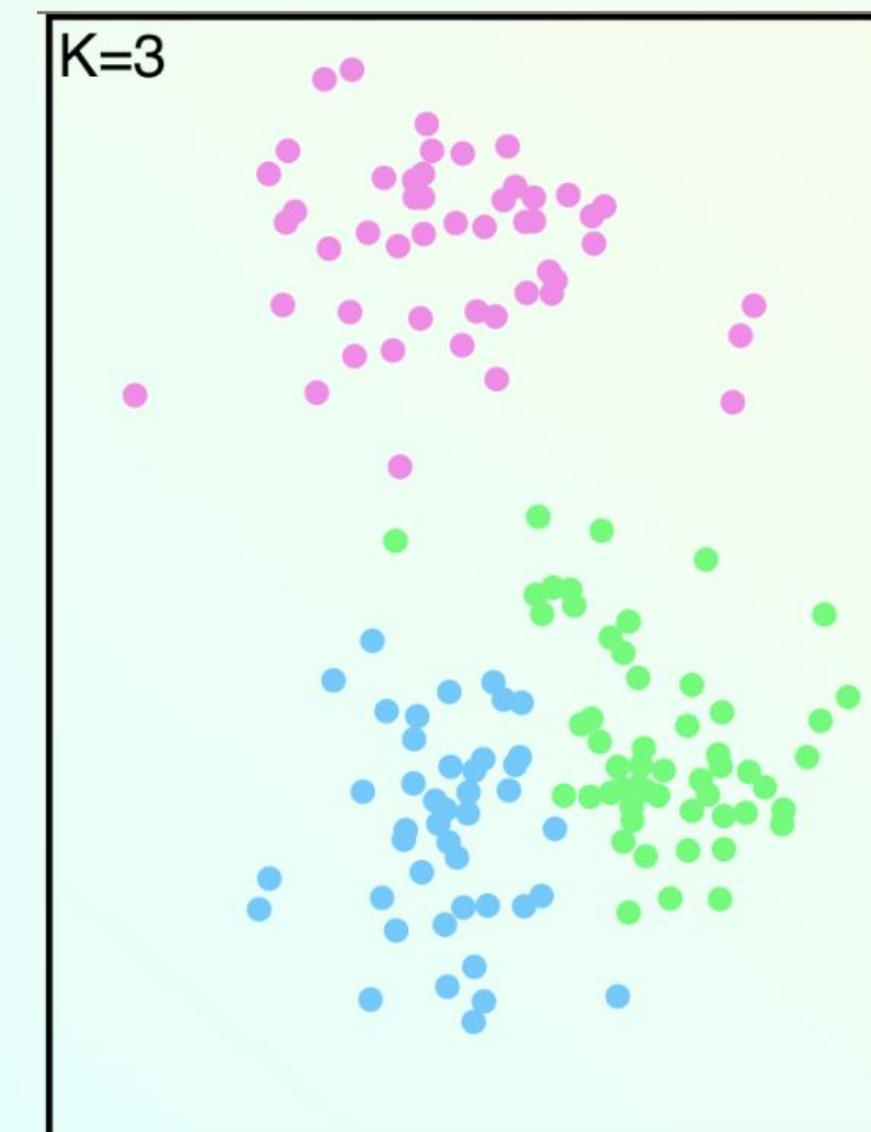
Por ejemplo, un sitio de compras podría intentar identificar grupos de compradores con historiales de navegación y compras similares, así como artículos que sean de particular interés para los compradores dentro de cada grupo.

CLUSTERING

CLUSTERING

Clustering o agrupación se refiere a un conjunto amplio de técnicas para encontrar subgrupos o clusters en un dataset.

Cuando agrupamos las observaciones, buscamos dividirlas en grupos distintos de modo que las observaciones dentro de cada grupo sean **similares entre sí**, mientras que las observaciones en otro grupo sean bastante **diferentes** de las del primero.



CLUSTERING

Vamos a ver tres algoritmos de clustering de los más famosos entre los múltiples que existen:

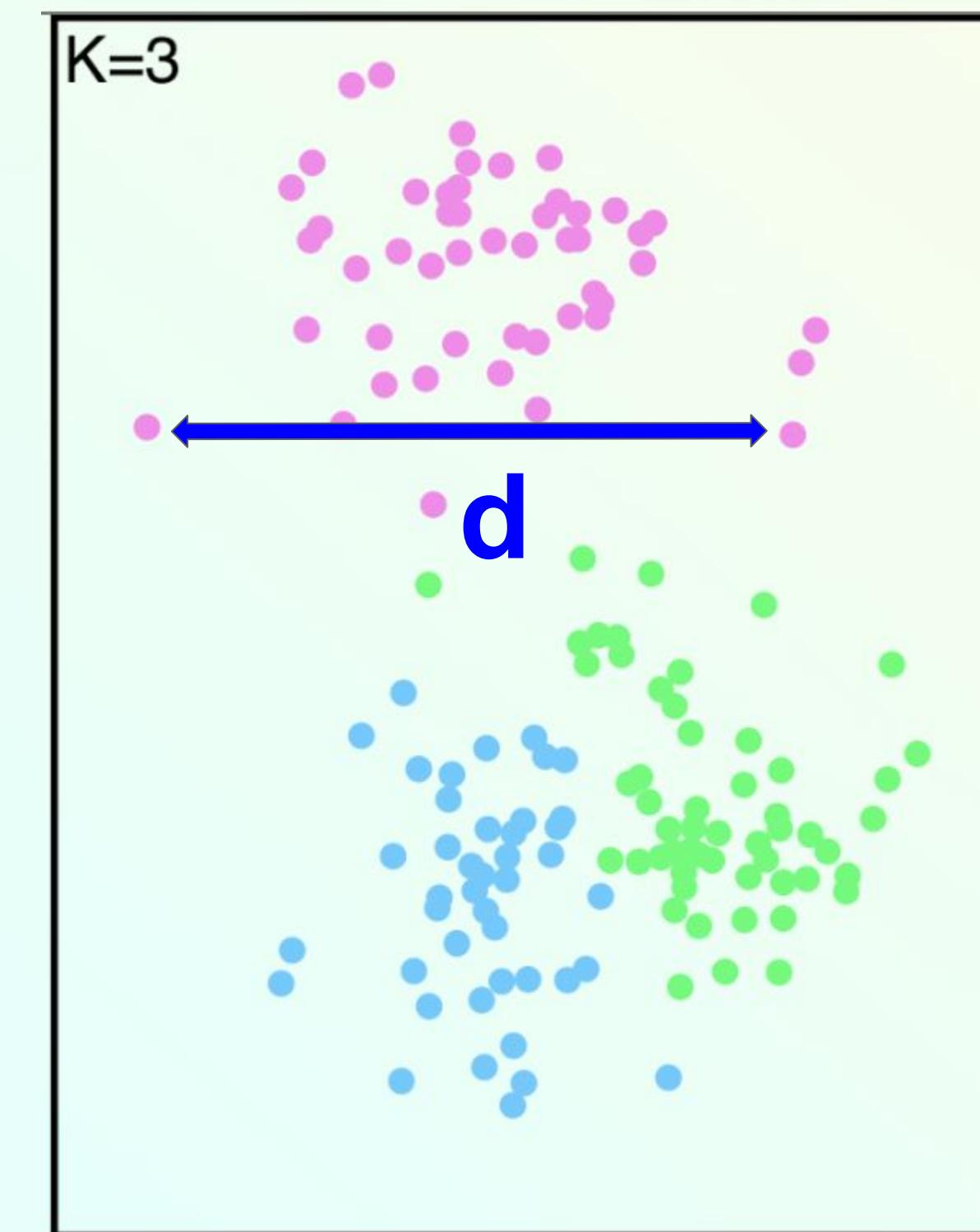
- **K-Means**
- **Hierarchical Clustering**
- **Modelo de Mixtura Gaussiana**

Pero antes de ver los algoritmos, hay que definir qué es similar o qué es diferente. Esto suele ser una consideración específica de un dominio que debe realizarse basándose en el conocimiento de los datos que se están estudiando.

MEDIDAS DE DISTANCIA O SIMILITUD

MEDIDAS DE DISTANCIA O SIMILITUD

La **distancia** es la medida que cuantifica el grado de similitud o diferencia entre dos observaciones individuales dentro del conjunto de datos.

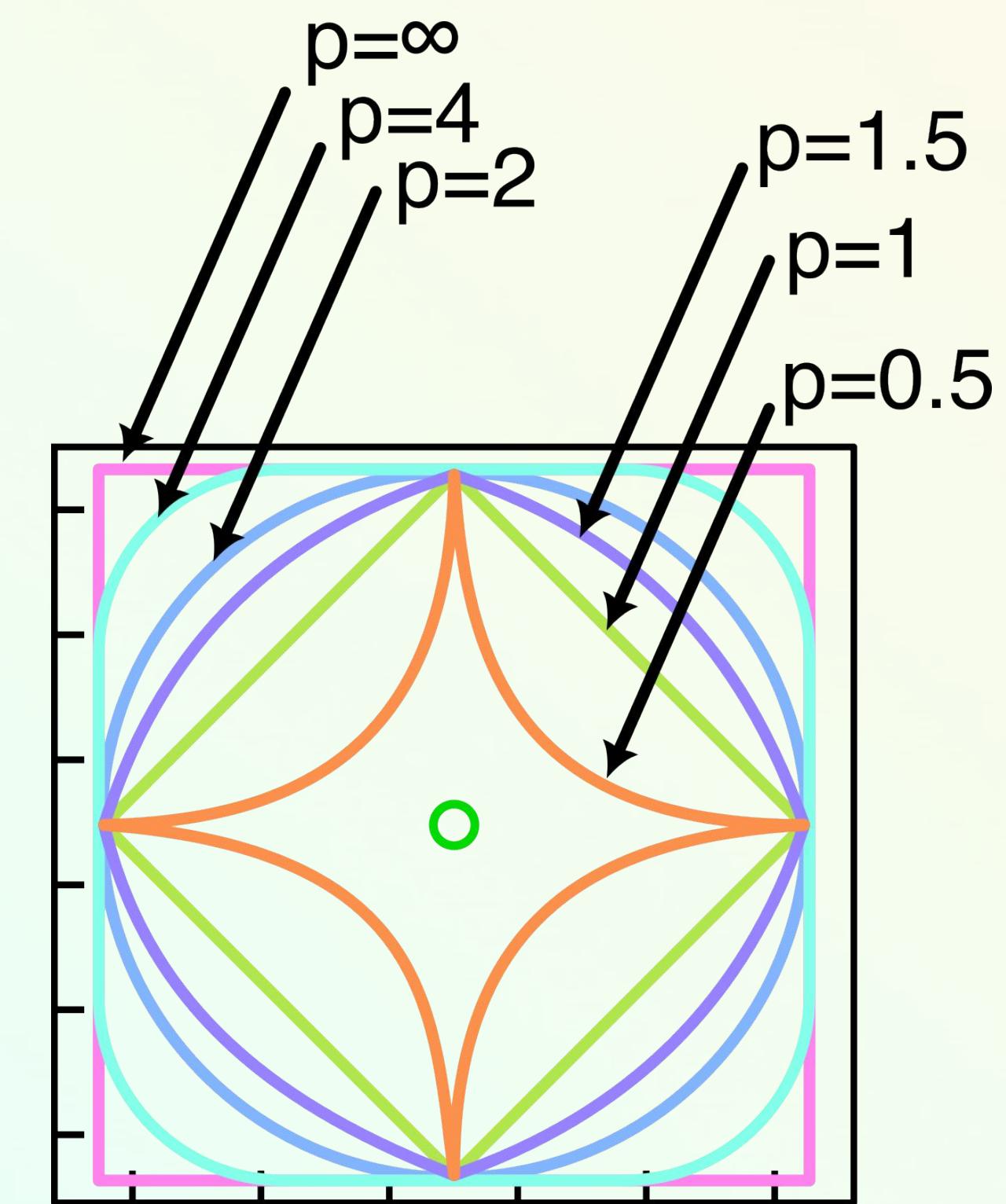


MEDIDAS DE DISTANCIA O SIMILITUD

En **kNN** vimos las medidas de distancias, en clustering, se utilizan las mismas medidas para indicar que tan diferente o similares son.

- **Distancia de Manhattan**
- **Distancia euclidiana**
- **Distancia de Chebyshev**
- **Distancia de Minkowski**

$$d_p(a, b) = \left(\sum_{i=1}^n (a_i - b_i)^p \right)^{1/p}$$



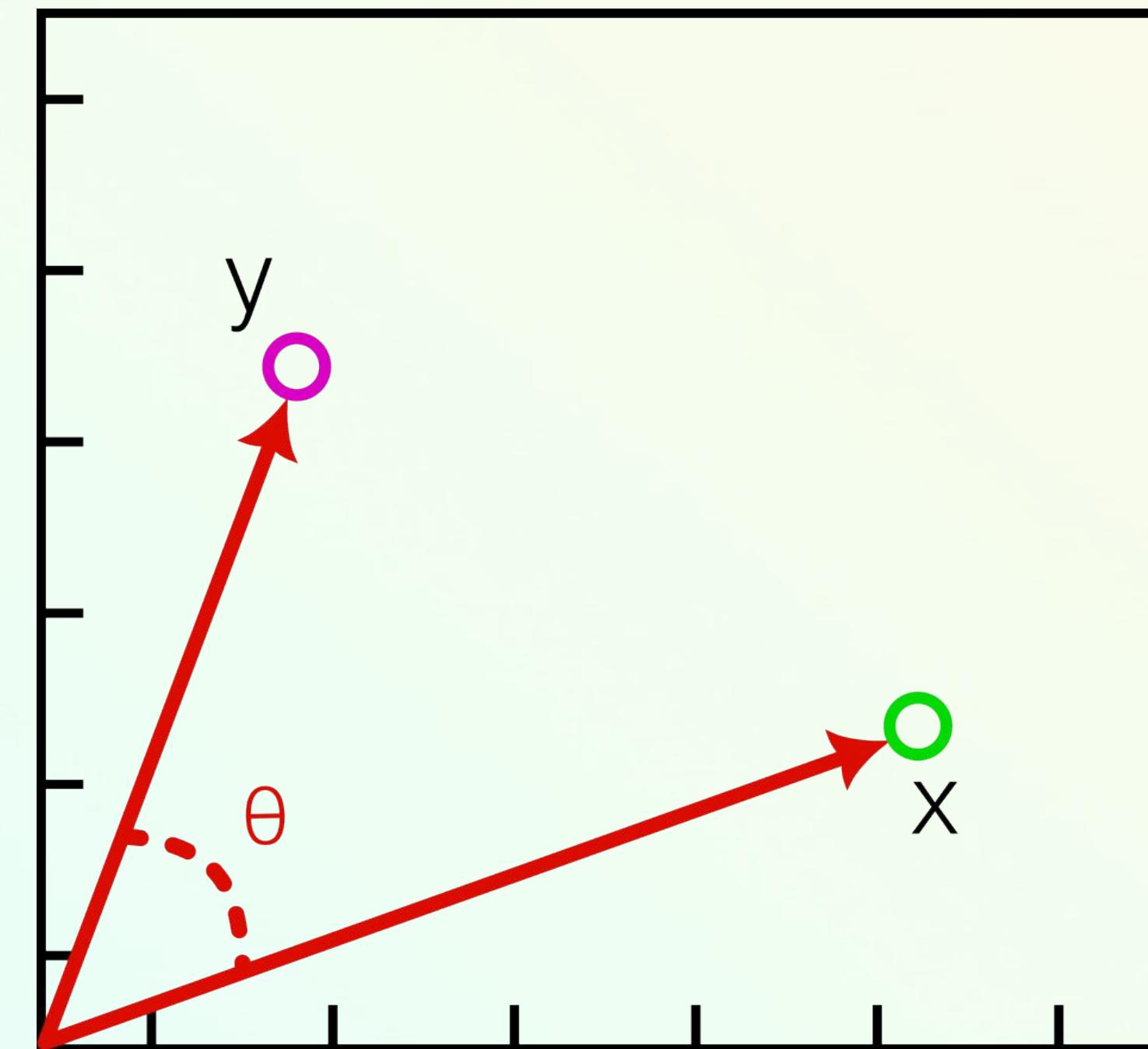
MEDIDAS DE DISTANCIA O SIMILITUD

En **kNN** vimos las medidas de distancias, en clustering, se utilizan las mismas medidas para indicar que tan diferente o similares son.

- **Distancia Coseno**

$$d_c(a, b) = 1 - S_c(a, b)$$

$$d_c(a, b) = \cos(\theta) = \frac{a \cdot b}{\|a\| \|b\|}$$

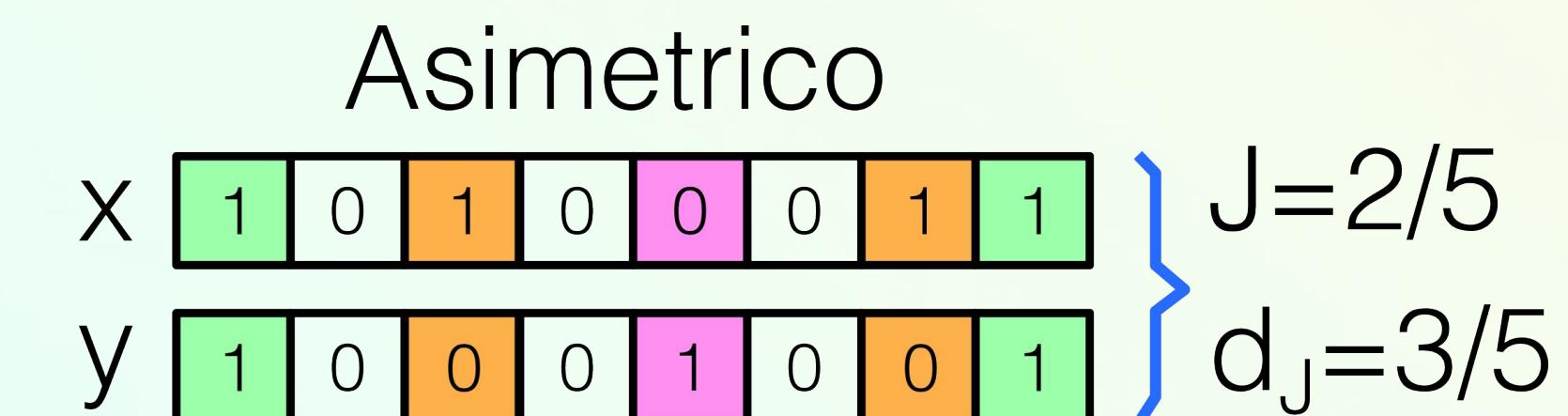


MEDIDAS DE DISTANCIA O SIMILITUD

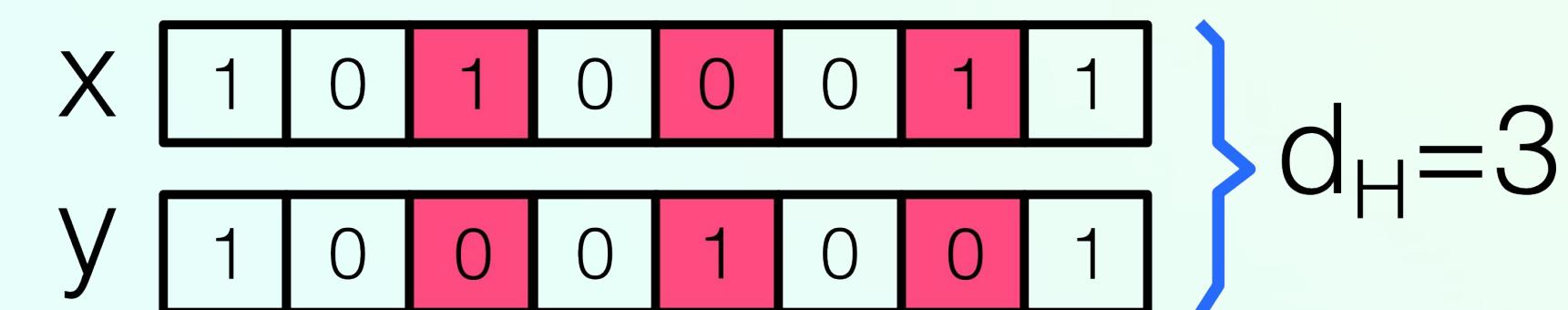
En **kNN** vimos las medidas de distancias, en clustering, se utilizan las mismas medidas para indicar que tan diferente o similares son.

- **Similitud de Jaccard**

$$J(a, b) = \frac{|a \cap b|}{|a \cup b|}$$



- **Distancia de Hamming**

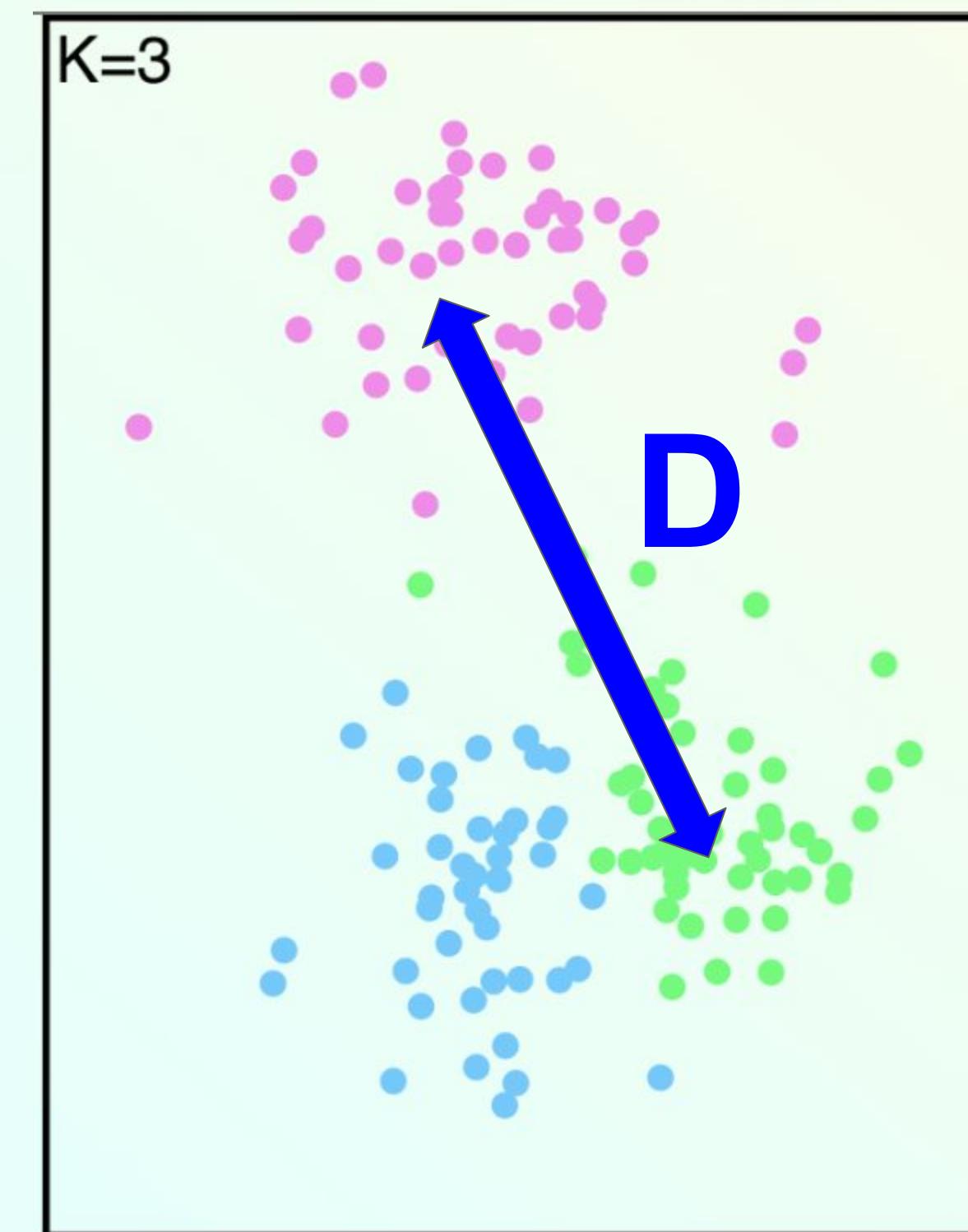


LINKAGE

*o métodos para calcular
distâncias inter-cluster*

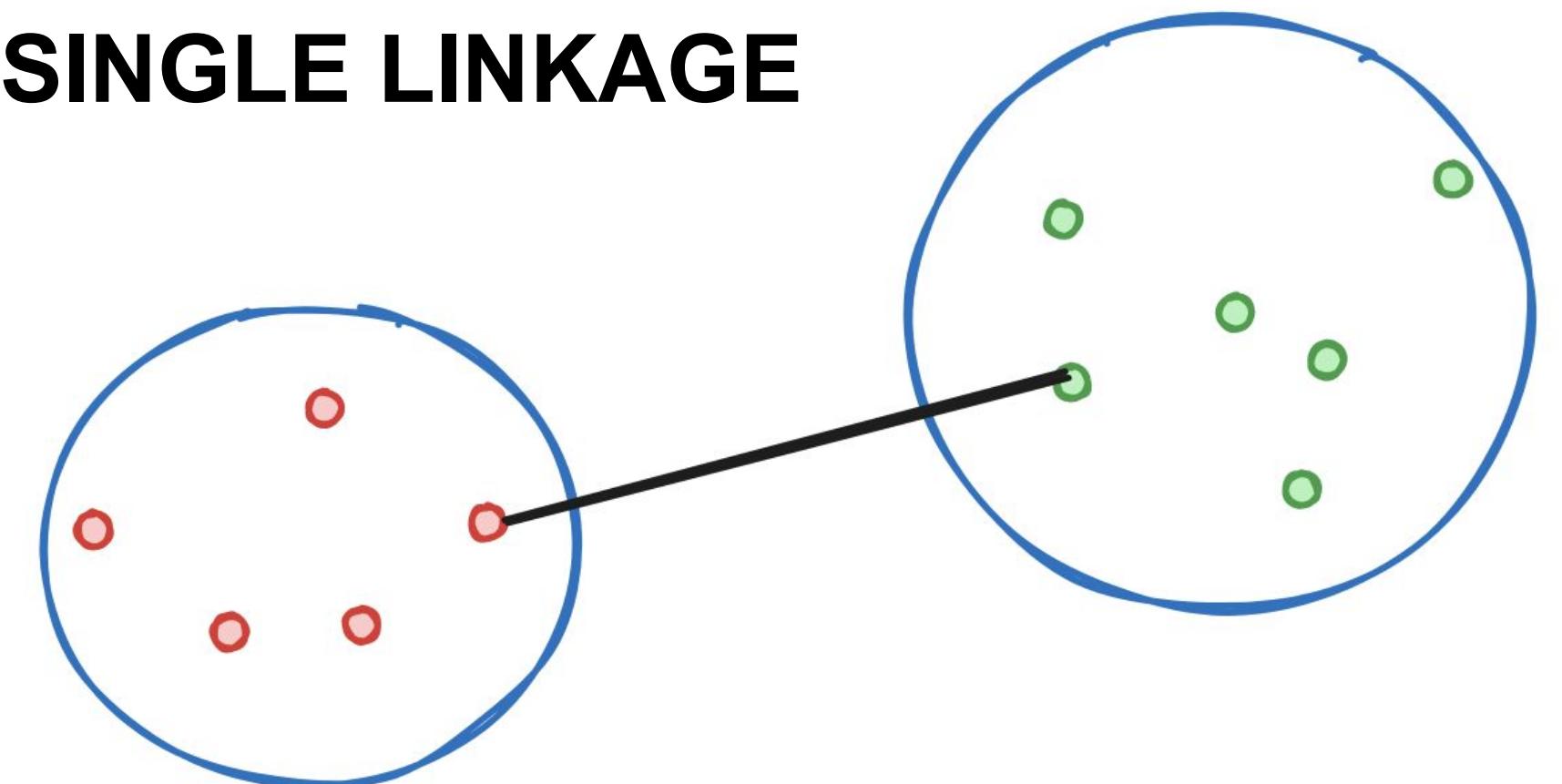
LINKAGE

El **linkage** es el método o criterio que define cómo se calcula la distancia entre dos clusters, a partir de las distancias individuales entre sus elementos.

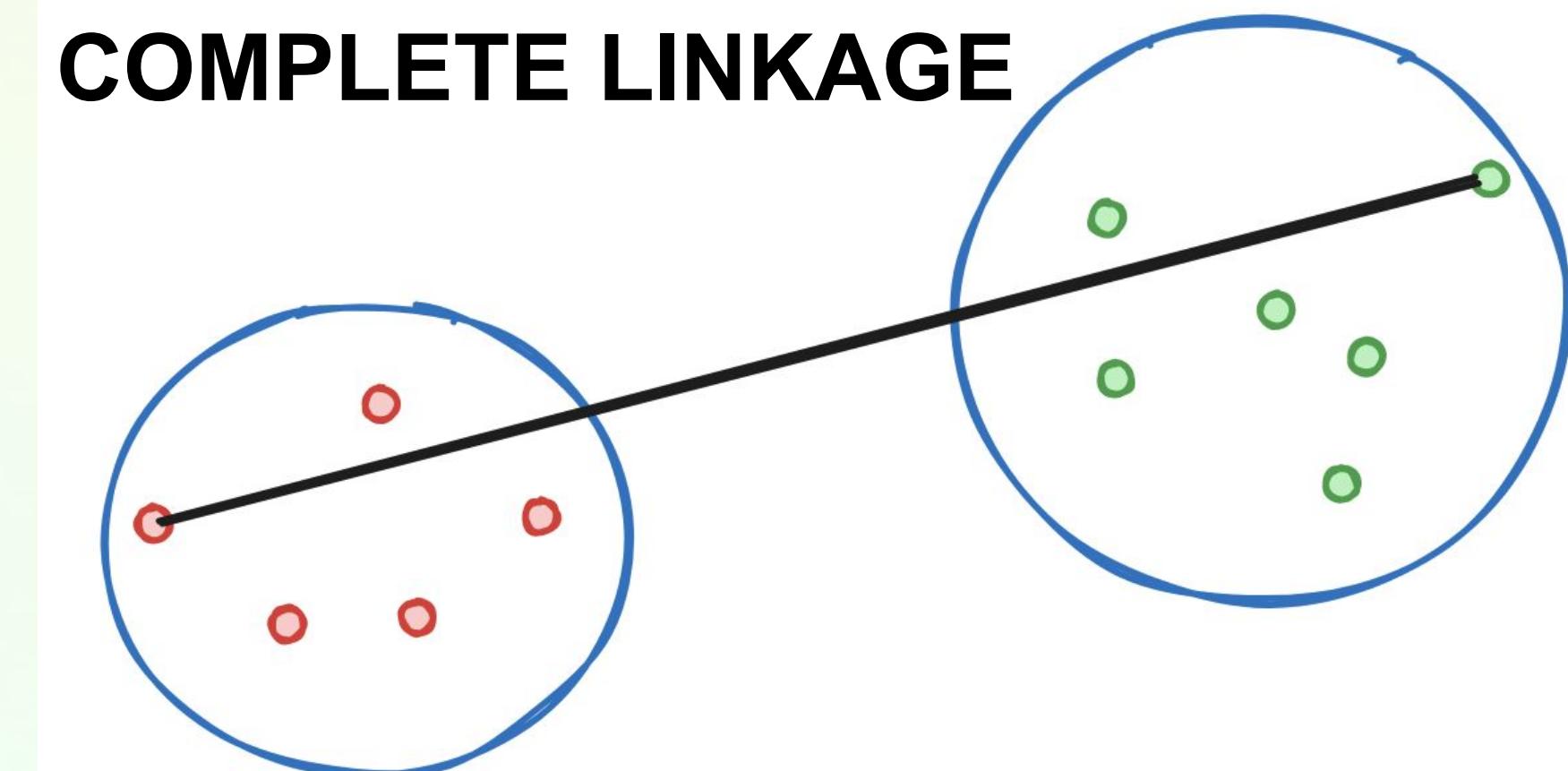


TIPOS DE LINKAGE

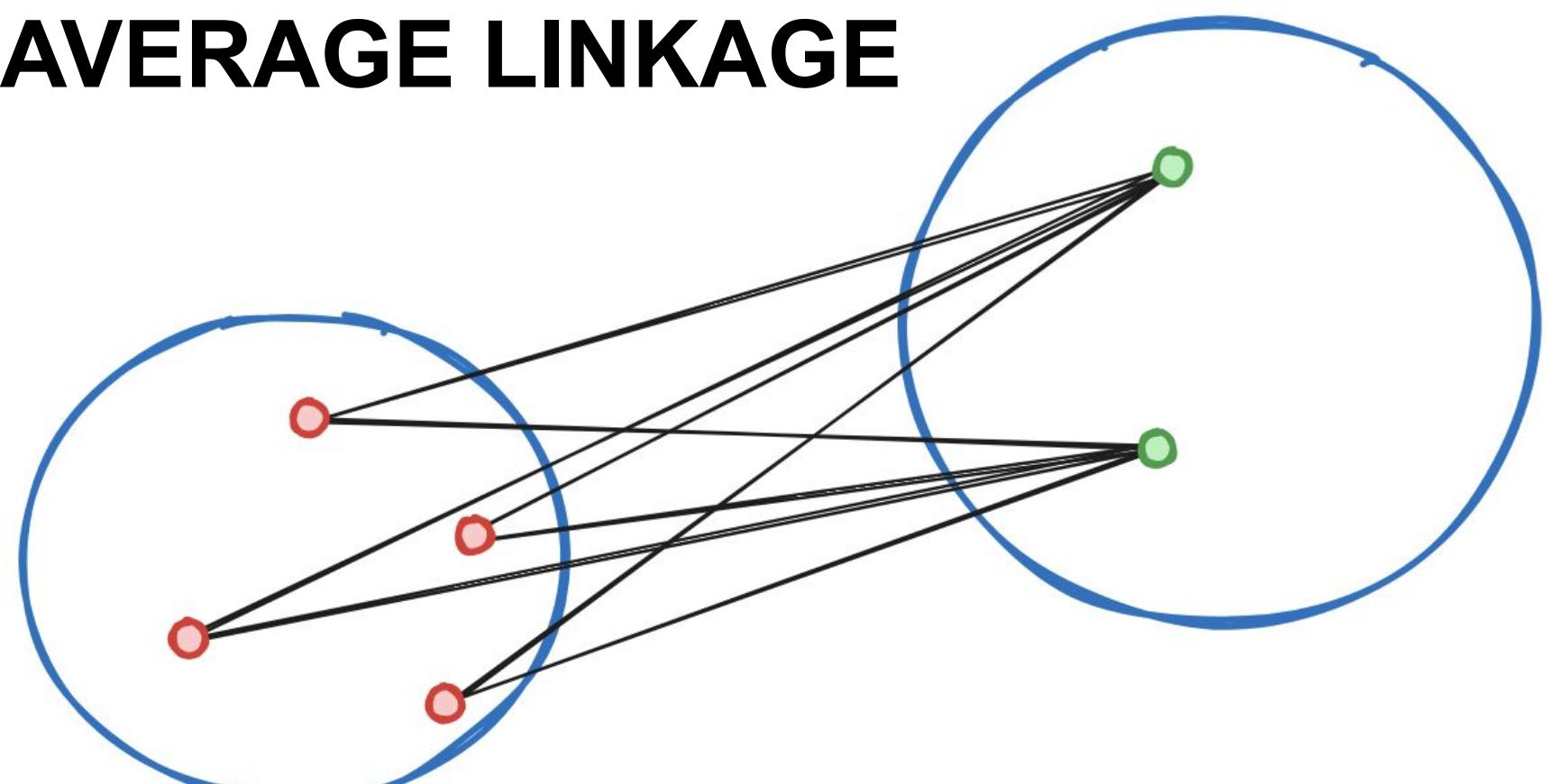
SINGLE LINKAGE



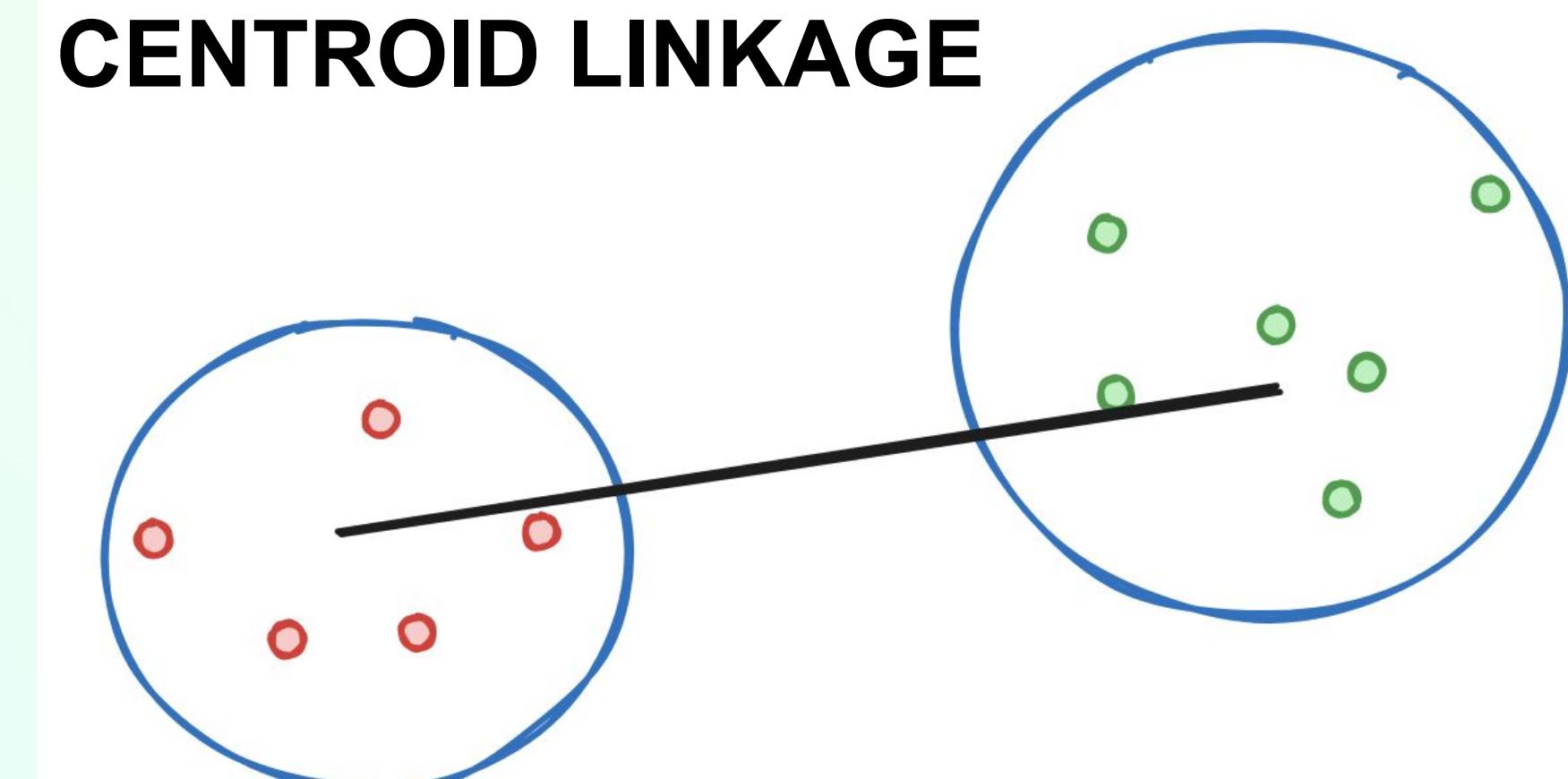
COMPLETE LINKAGE



AVERAGE LINKAGE



CENTROID LINKAGE



TIPOS DE LINKAGE

Los tipos más comunes de **linkage** son:

- **Single linkage**: utiliza la distancia **mínima** entre todos los pares de puntos de los dos grupos.
- **Complete linkage**: utiliza la distancia **máxima** entre todos los pares de puntos. Es un método *greedy*, tiende a formar clusters compactos y bien separados.
- **Average linkage**: utiliza el **promedio** de todas las distancias. Representa un compromiso entre los dos métodos anteriores.
- **Centroid linkage**: utiliza la distancia entre los **centroides** de los grupos.
- **Método Ward**: no se basa en distancias sino en la suma de los cuadrados dentro de cada cluster (**SSE**).

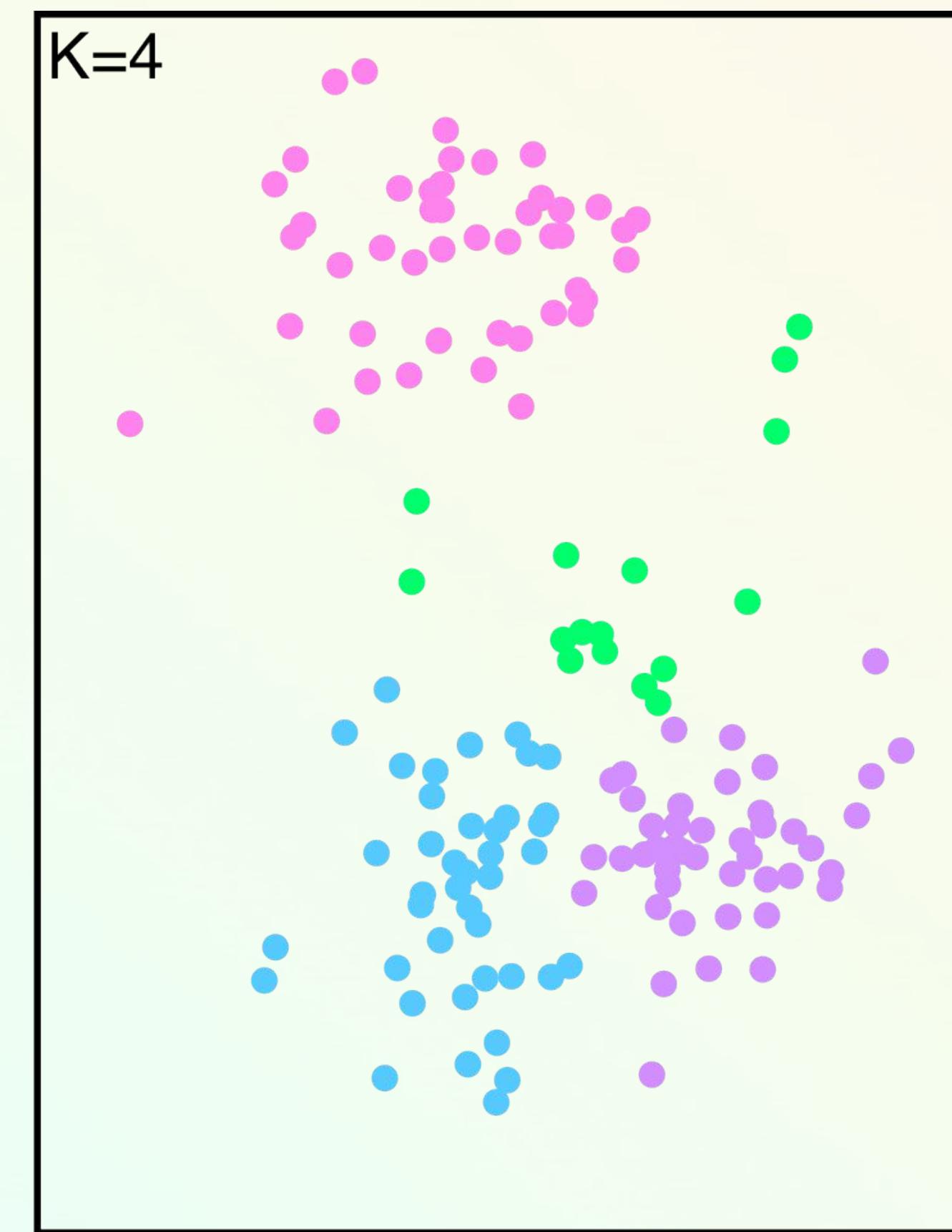
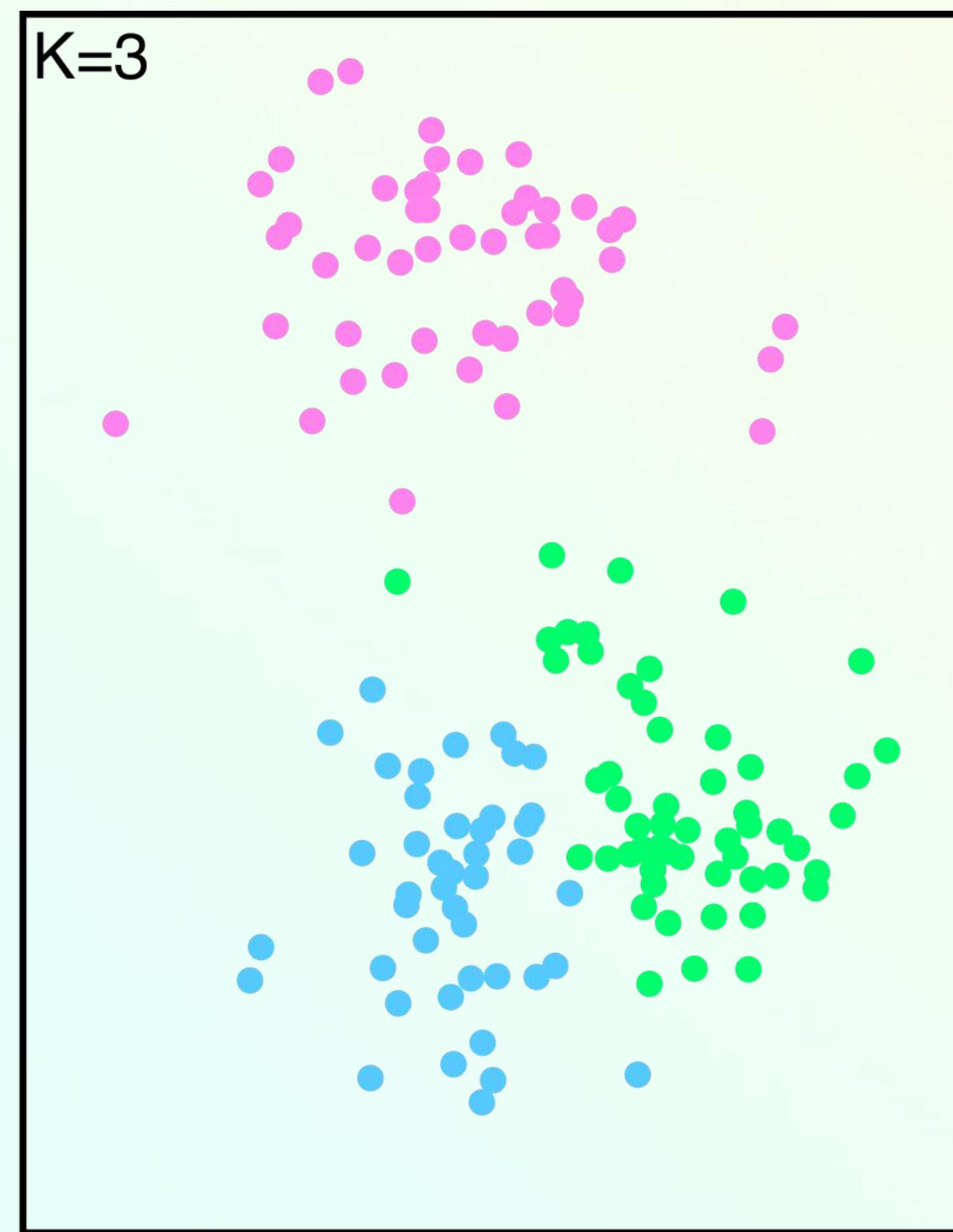
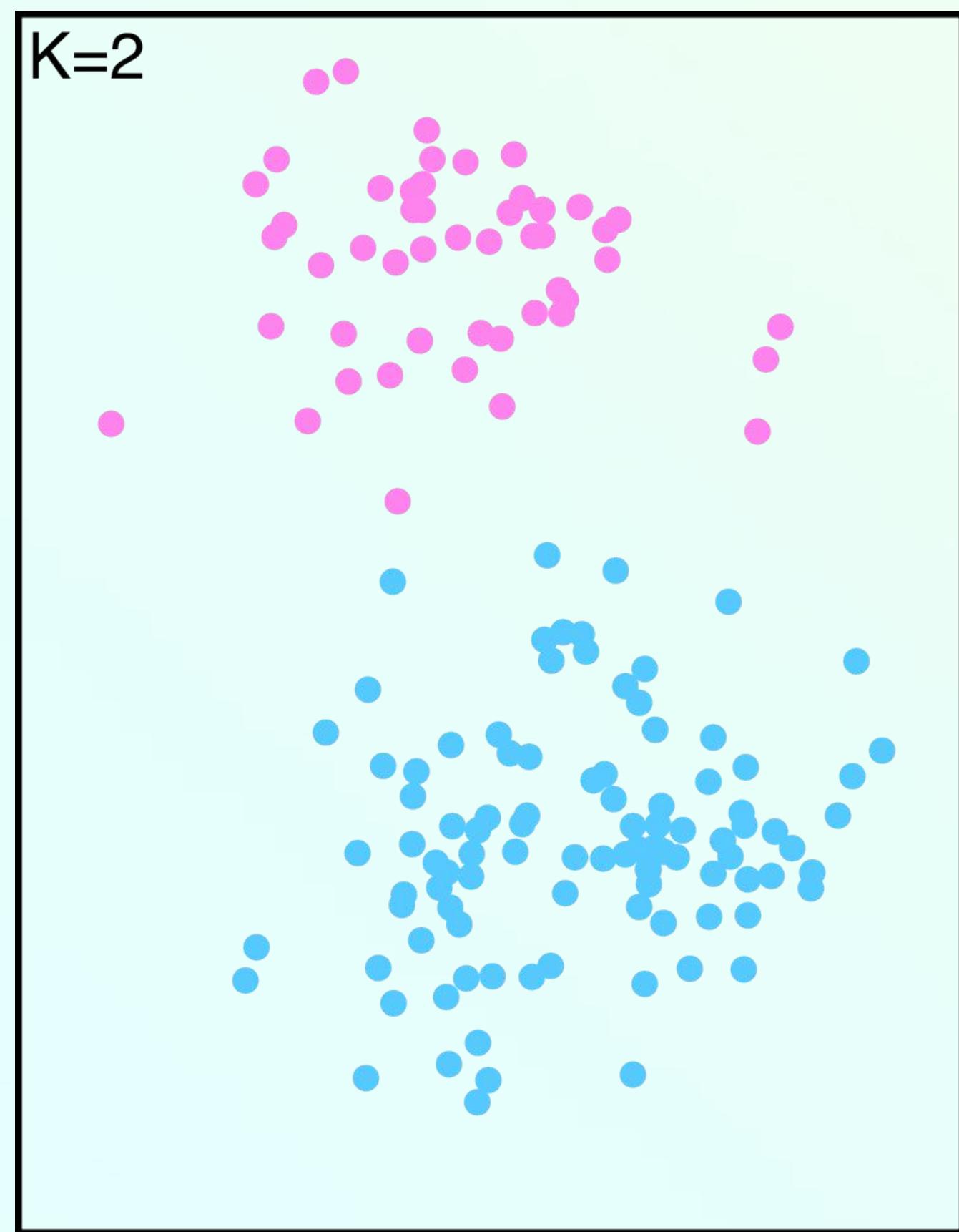
K-MEANS CLUSTERING

K-MEANS

K-means clustering es un enfoque simple para dividir un conjunto de datos en K grupos distintos y no superpuestos.

Para realizar la agrupación K-means, primero debemos especificar el número deseado de agrupaciones K y luego, el algoritmo asignará cada observación a exactamente uno de los K grupos.

K-MEANS



K-MEANS

Pongamos algunas notaciones, sean C_1, C_2, \dots, C_K son conjuntos que contienen los índices de las observaciones del dataset en cada cluster. Y se satisface las siguientes propiedades:

- Cada observación pertenece al menos a uno de los K clusters

$$C_1 \cup C_2 \cup \dots \cup C_K = \{1, 2, \dots, n\}$$

- Los cluster no se superponen, es decir, una observación pertenece a un solo cluster.

$$C_k \cap C_{k'} = \emptyset \quad \forall k \neq k'$$

K-MEANS

La idea detrás del algoritmo es que un buen agrupamiento es uno en donde la variación dentro del cluster es la menor posible.

La variación dentro de un cluster C_K es una medida $W(C_k)$ de la cantidad en la que las observaciones dentro de un cluster difieren entre sí. Lo que buscamos hacer es:

$$\underset{C_1, \dots, C_K}{\text{minimizar}} \left\{ \sum_{k=1}^K W(C_k) \right\}$$

K-MEANS

Para poder resolver este problema, debemos definir a $W(C_K)$. Hay muchas métricas posibles, pero la más común es la distancia Euclíadiana cuadrada o Suma de Cuadrados Intracluster:

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^p (x_{ij} - x_{i'j})^2$$

$|C_k|$ es el número de observaciones en el cluster k.

Ahora, resolver esta minimización es un problema difícil de resolver dado a la inmensa cantidad de cálculos.

K-MEANS

Una forma de resolver esto de una forma más eficiente es calcular el centroide del cluster y calcular la distancia con respecto a ese punto.

El centroide de un cluster se calcula de la siguiente forma:

$$\text{Centroide} = \frac{1}{|C_k|} \sum_{i \in C_k} x_i$$

Entonces, ahora podemos calcular la distancia intra-cluster más sencillamente:

$$W(C_k) = \sum_{i \in C_k} \sum_{j=1}^p (x_{ij} - \bar{x}_j)^2$$

K-MEANS

Y ahora buscamos minimizar la suma de las distancias intra-cluster que es lo que se conoce como **inercia**:

$$Inercia = \sum_{k=1}^K W(C_k)$$

Y ahora sí, el problema de minimización es más eficiente, minimizar $\left\{ \sum_{k=1}^K W(C_k) \right\}_{C_1, \dots, C_K}$

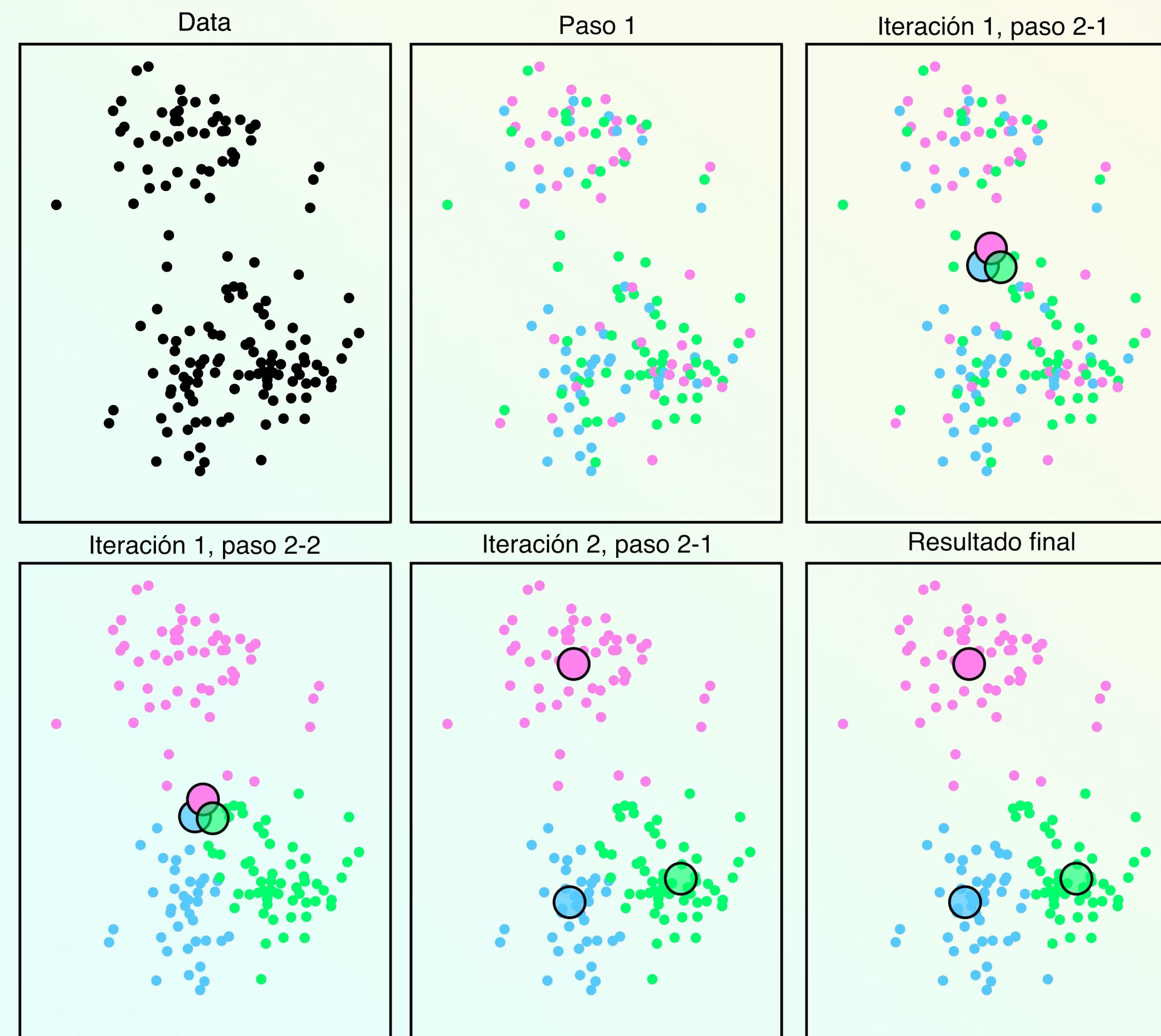
El proceso es iterativo, y se va minimizando de a pasos hasta que llegue a un valor estable, siempre en dirección que reduce la inercia.

K-MEANS

El algoritmo consiste en los siguientes pasos:

1. Asignar **aleatoriamente**, de 1 a K, a cada observación, a modo de inicialización.
2. Iterar hasta que la asignación de cluster no cambia más:
 1. Para cada uno de los K clusters, computar el **centroide** del cluster.
 2. Asignar cada observación al cluster cuyo centroide se encuentre más cercano (mediante la distancia Euclíadiana).
 3. Calcular el valor de inercia.

K-MEANS

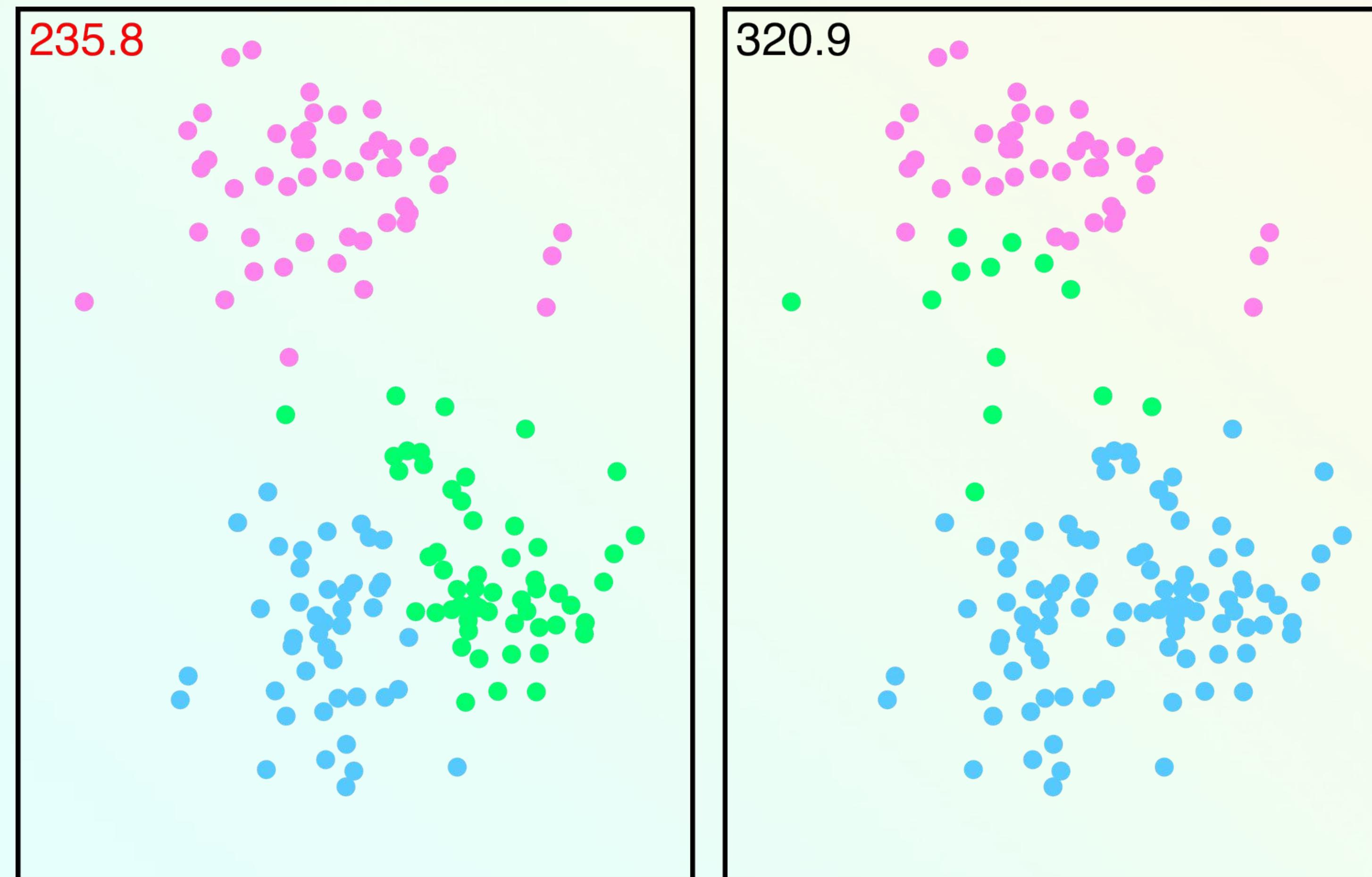


K-MEANS

Este algoritmo garantiza que se va a lograr la minimización de la inercia en cada paso. Además, dado que cada paso de iteración siempre disminuye, el algoritmo se ejecuta hasta que llegue a un mínimo local (o se alcance un número máximo de iteraciones establecido).

Dado que los centroides iniciales se eligen de forma aleatoria, distintas ejecuciones pueden converger a distintos mínimos locales. Esto significa que cada vez que se ejecuta este algoritmo, se puede obtener un resultado diferente. Por esta razón es recomendable ejecutarlo múltiples veces y finalmente seleccionar la **mejor** solución (La que tenga menor inercia).

K-MEANS



K-MEANS

Un problema que nos presenta K-means es determinar el número óptimo de clusters.

En la práctica, existen varios métodos para elegir el número óptimo de clusters en K-means (y también en otros algoritmos de clustering):

- Método del codo
- Índice de la silueta
- Fuerza de predicción
- otros ...

MÉTODO DEL CODO

MÉTODO DEL CODO

El **método del codo (Elbow Method)** es una técnica utilizada para determinar el número óptimo de clusters en un conjunto de datos. Se basa en ver variación de una métrica de calidad del clustering (generalmente la inercia) a medida que se varía el número de clusters.

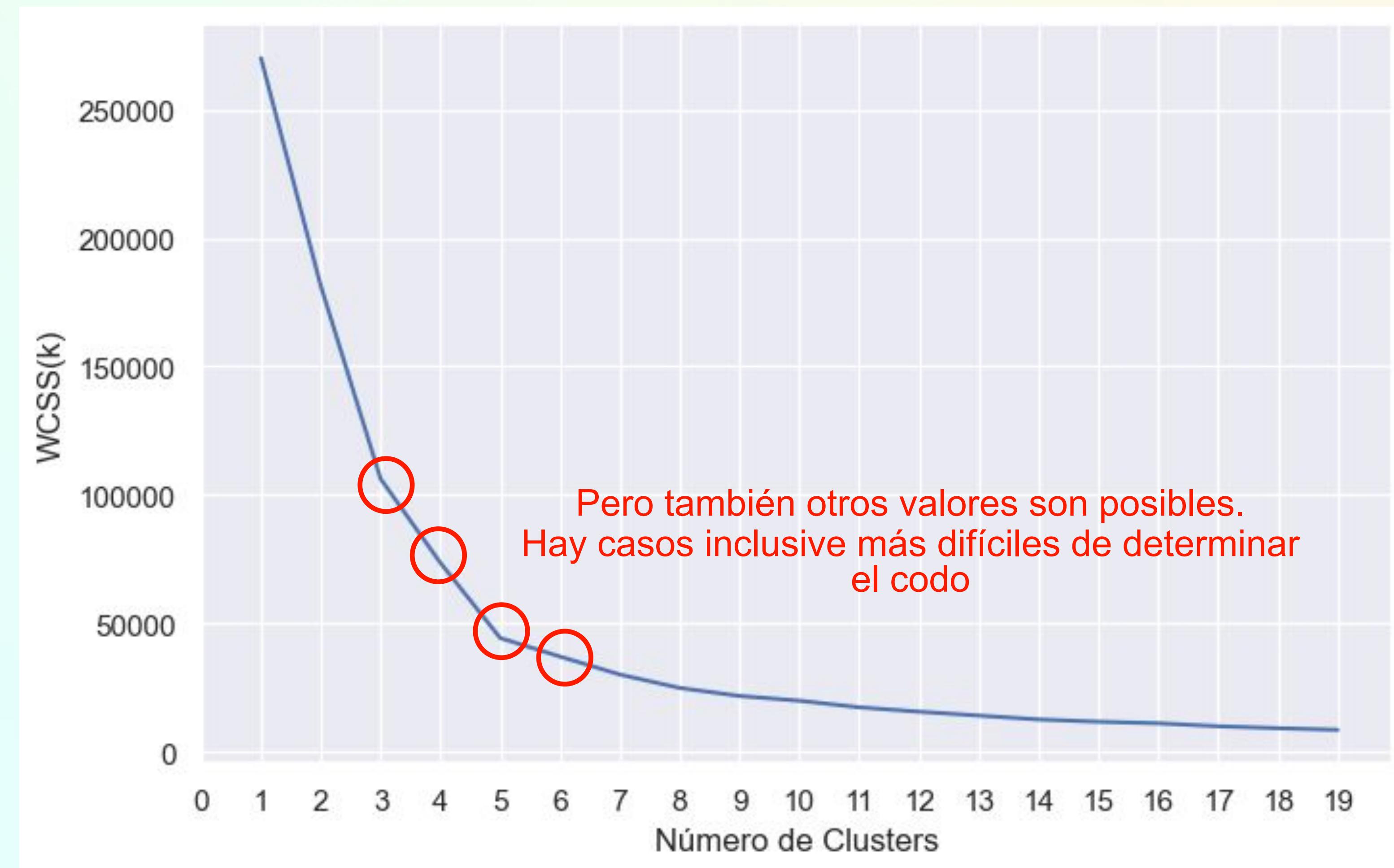
MÉTODO DEL CODO

1. Se ejecuta el algoritmo de clustering (por ejemplo, k-means) en el conjunto de datos para diferentes valores de k (número de clusters).
2. Para cada valor de k , se calcula una métrica de calidad del clustering, generalmente es la suma de cuadrados dentro del cluster (inercia).
3. Luego, se traza un gráfico que muestra los valores de k en el eje x y los valores de la métrica (inercia) en el eje y.
4. Se observa el gráfico y se busca un punto de **codo**. El punto de codo es aquel donde la métrica (inercia) comienza a disminuir a una tasa más lenta. En otras palabras, es el punto donde agregar más clusters ya no mejora significativamente la calidad del clustering.

MÉTODO DEL CODO



MÉTODO DEL CODO



ÍNDICE DE LA SILUETA

ÍNDICE DE LA SILUETA

Otra métrica que podemos ver es el puntaje de la silueta. Es una métrica que combina la cohesión y la separación para evaluar la calidad de los clusters y proporcionar información sobre la "bondad" de la agrupación. Para cada punto en un cluster, se calcula:

$$a_i = \frac{1}{|C_I| - 1} \sum_{j \in C_I, i \neq j} d(i, j)$$

Cohesión de i con sus
co-miembros

$$b_i = \min_{J \neq I} \frac{1}{|C_J|} \sum_{j \in C_J} d(i, j)$$

Separación de i con otros
clusters

ÍNDICE DE LA SILUETA

Métricas relacionadas al índice de la silueta son:

- **Cohesión**: Es una medida de cuán cerca están los puntos dentro de un mismo cluster.
- **Separación**: Es una medida de cuán lejos están los clusters entre sí.

Estas métricas se podrían calcular como métricas independientes para ver la calidad del modelo.

ÍNDICE DE LA SILUETA

El índice de silueta se calcula para cada punto como:

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

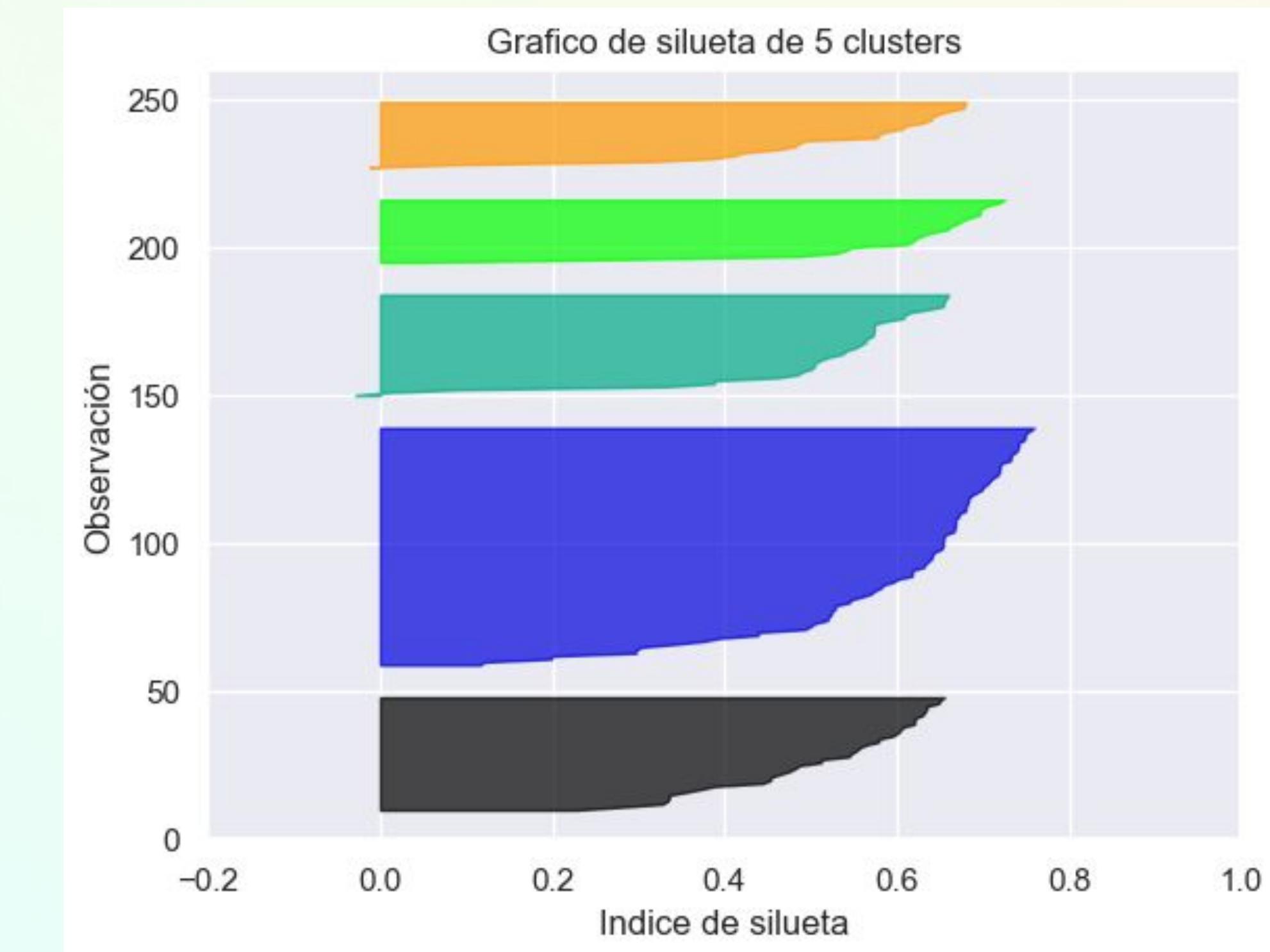
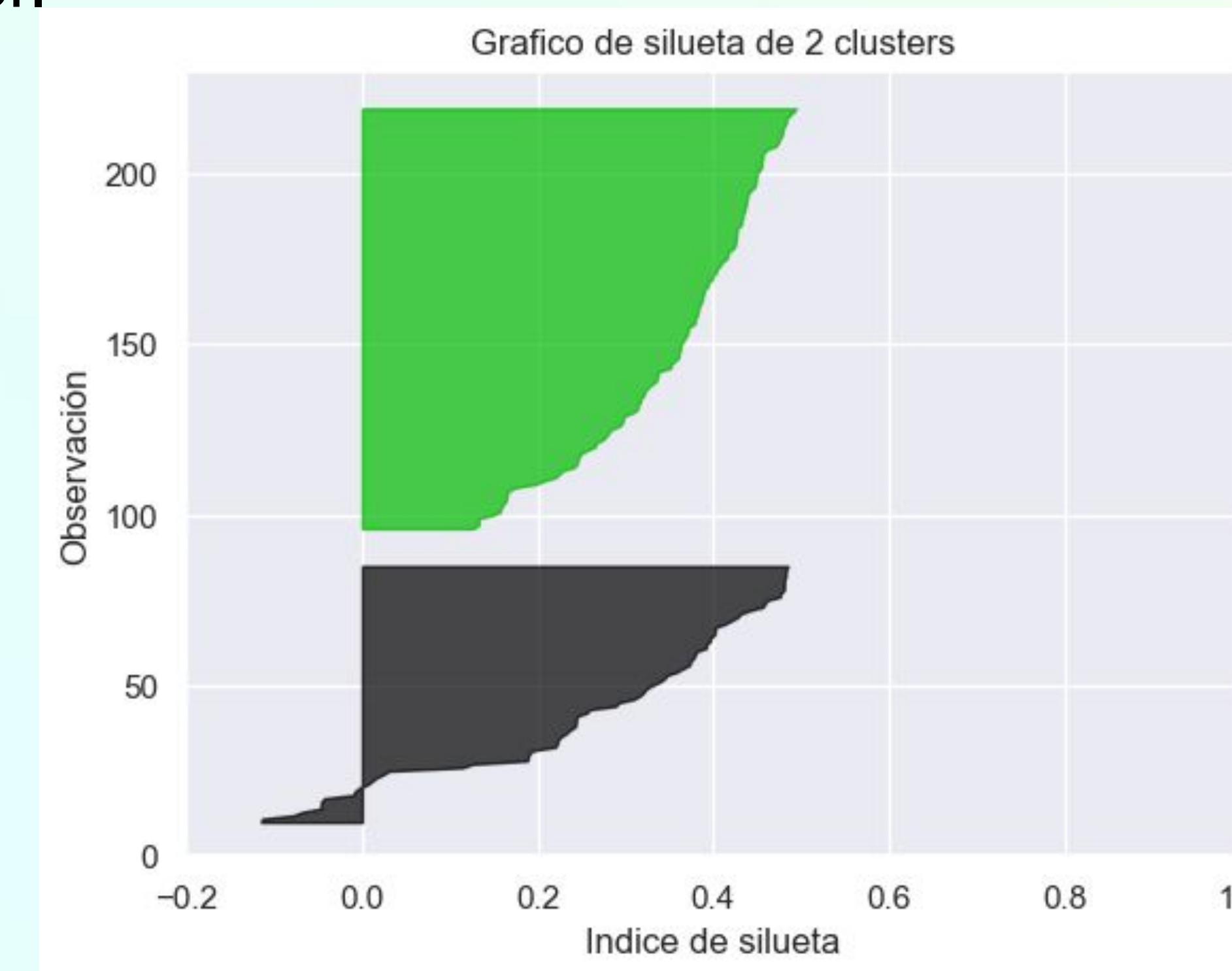
Los valores del índice de silueta varían entre -1 y 1, donde:

- Cercano a 1 indica que una observación es muy similar a su cluster y disímil de los otros
- 0 indica que prácticamente los cluster se superponen y es difícil determinar la pertenencia.
- -1 significa que la observación tendría más sentido que esté asignada a otro cluster.

Dado la dificultad de calcular esto, $O(n^2)$, en realidad se calcula con respecto a los centroides de los clusters.

ÍNDICE DE LA SILUETA

Este índice nos permite graficar los clusters (incluso cuando se tienen más de dos dimensiones), ver el tamaño de cada uno y evaluar qué tan segura es la pertenencia de cada observación a su cluster.



ÍNDICE DE LA SILUETA

Pero la forma que más se usa es para establecer si la cantidad de clusters es apropiada. Para ello se calcula el promedio de todas las observaciones para indicarnos qué tan apropiadamente están los datos agrupados.



FUERZA DE PREDICIÓN

FUERZA DE PREDICCIÓN

Otro método que creció en popularidad es el llamado **Fuerza de predicción**.

La idea se toma prestado del caso de un clasificador de aprendizaje supervisado, en el cual se minimiza un error de predicción, y además permite obtener estimaciones de observaciones individuales.

Este método se enfoca en el error de predicción en vez de la suma de los cuadrados intra-cluster o la cohesión o separación de los clusters.

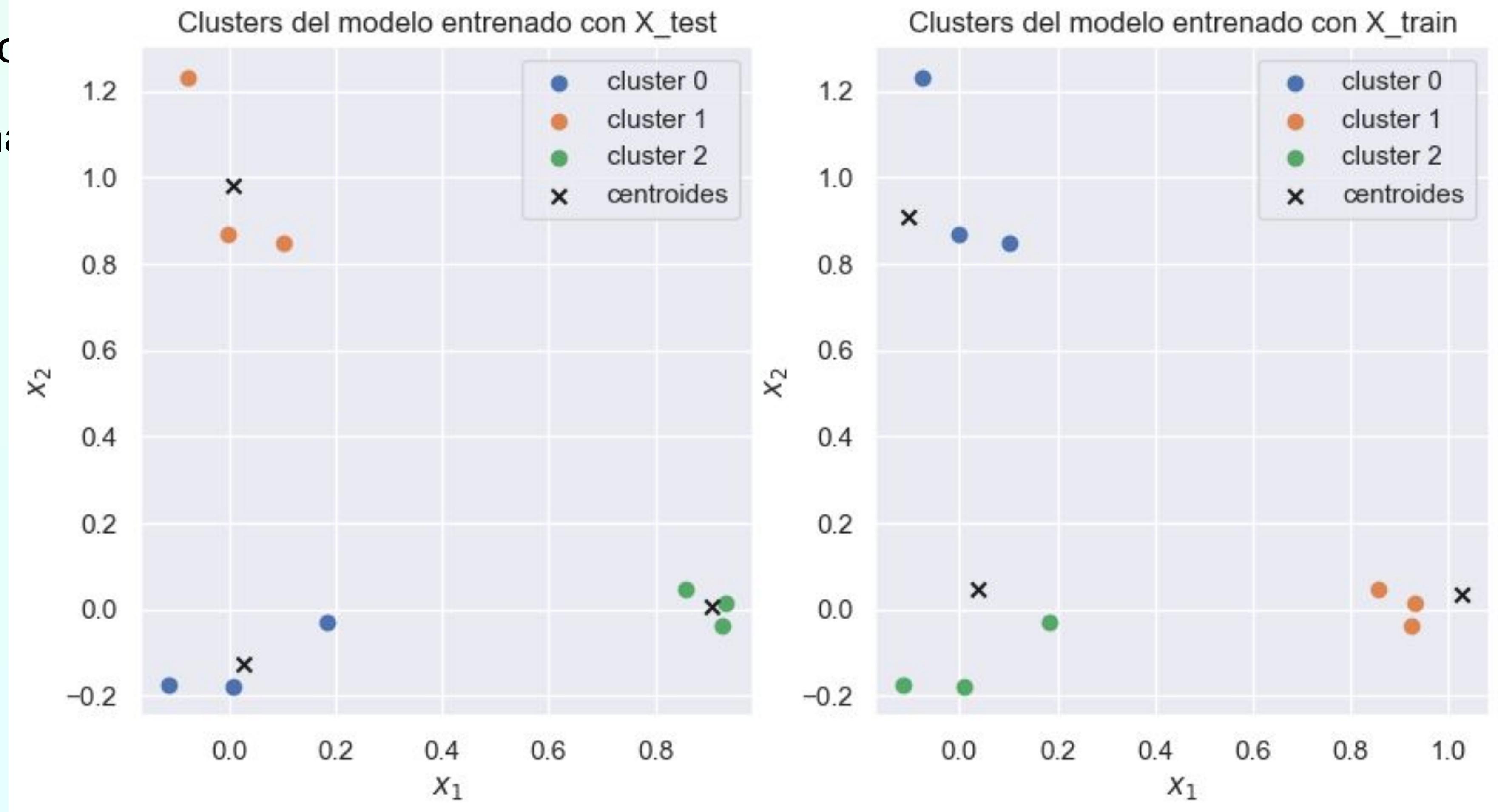
FUERZA DE PREDICCIÓN

¿Cómo hacemos para obtener un error de predicción en un modelo de clustering?

En su forma más simple, el dataset se separa en dos, X_{train} y X_{test} , y para cada uno de ellos se entrena un modelo de clustering y se evalúa a ambos modelos con X_{test} .

FUERZA DE PREDICCIÓN

¿Cómo hace
En su forma
entrena un



FUERZA DE PREDICCIÓN

Entonces para el modelo entrenado con X_{test} , obtenemos la matriz de co-miembros para algún cluster:

	0	1	2	3	4	5	6	7	8
0		1	1						
1	1			1					
2	1	1							
3									
4									
5									
6									
7									
8									

FUERZA DE PREDICCIÓN

Y luego vemos si en el modelo entrenado con X_{train} si estas conexiones se siguen manteniendo:

	0	1	2	3	4	5	6	7	8
0		1	0						
1	1		1						
2	0	1							
3									
4									
5									
6									
7									
8									

FUERZA DE PREDICCIÓN

Finalmente se calcula la proporción de conexiones para el cluster:

$$FP_k = \frac{\text{Conexiones que se mantuvieron}}{|C_I|(|C_I| - 1)}$$

Y el valor para medir la calidad de la cantidad de clusters, se obtiene el mínimo valor de todos los clusters:

$$FP = \min(FP_k)$$

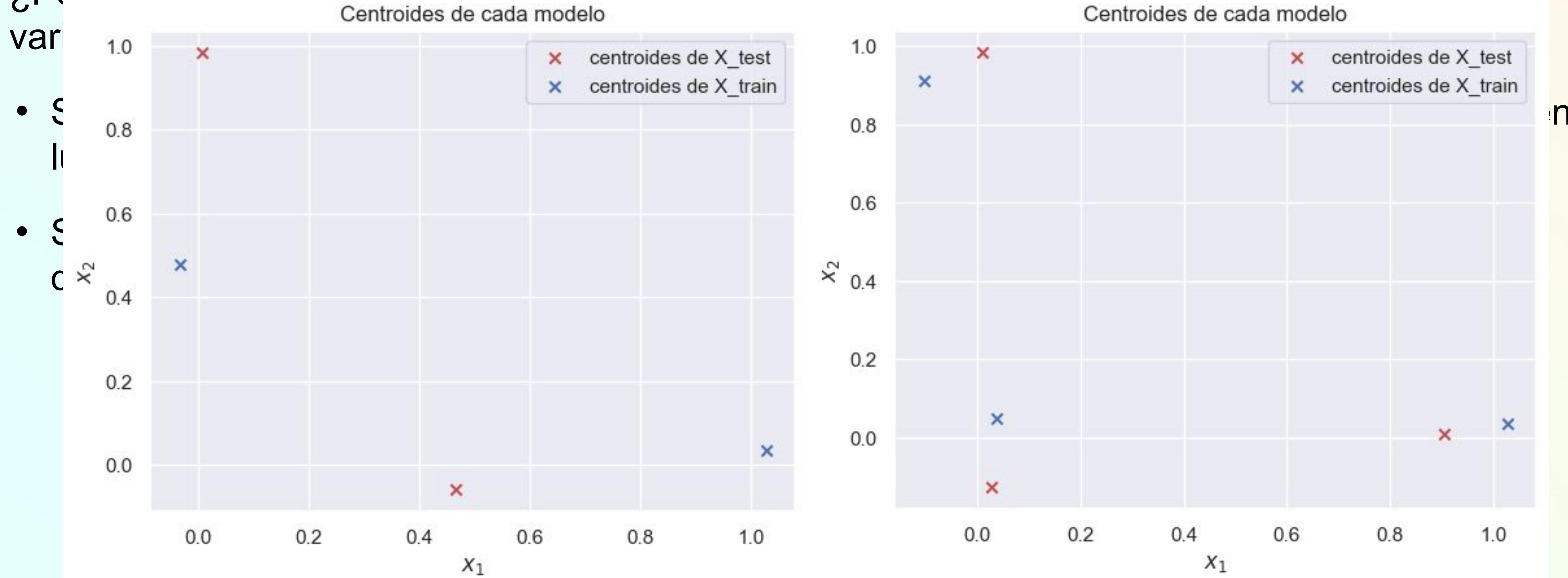
FUERZA DE PREDICCIÓN

¿Por qué este método es tan popular? Esto se debe a que nos permite medir el error de variabilidad:

- Si el número de clusters elegido es apropiado, los centroides de los dos datasets estarán en lugares cercanos con un error de variabilidad bajo.
- Si la cantidad es inapropiada, los centroides van a estar en lugares muy diferentes, y van a depender de cómo se separó el dataset. Es decir, la variabilidad es alta.

FUERZA DE PREDICCIÓN

¿Por qué este método es tan popular? Esto se debe a que nos permite medir el error de variación.



FUERZA DE PREDICCIÓN

¿Por qué este método es tan popular?

Además, permite extender el método para usarse con validación cruzada.

VAMOS A PRACTICAR UN POCO...

MODELO DE MIXTURA GAUSSIANA (GMM)

MODELO DE MIXTURA GAUSSIANA

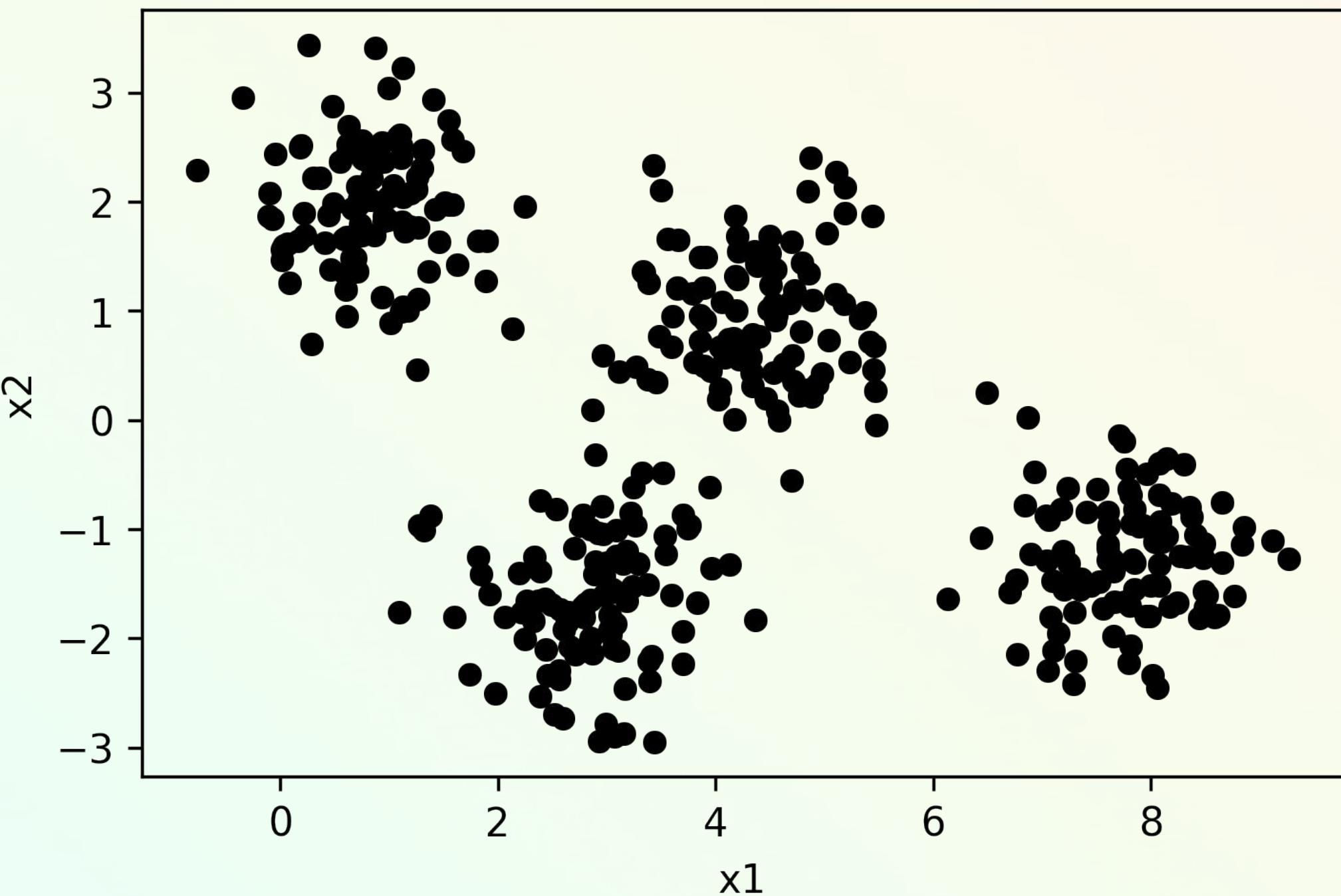
K-means es simple y relativamente fácil de entender, pero su simplicidad plantea desafíos prácticos en su aplicación. **La naturaleza no probabilística** de K-means y el uso de simplemente una distancia desde el centro del grupo para asignar la membresía del grupo conduce a un rendimiento deficiente en muchas situaciones del mundo real.

Veamos entonces a los modelos de mixtura gaussiana (GMM), que pueden verse como una extensión de las ideas detrás de K-means, pero también pueden ser una herramienta poderosa para la estimación más allá de la simple agrupación.

MODELO DE MIXTURA GAUSSIANA

Analicemos las debilidades de k-means:

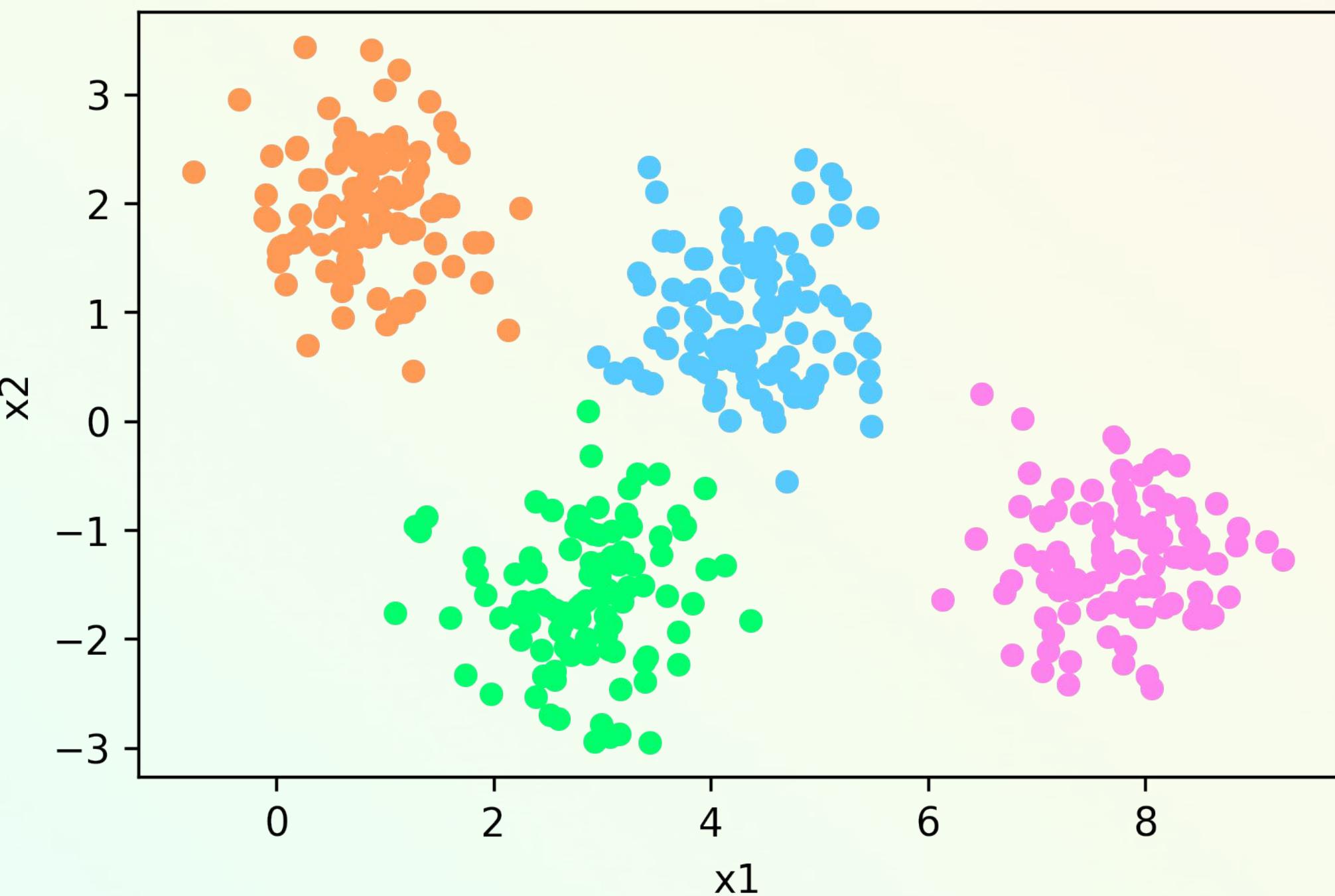
Si tenemos bloques separados de datos, el algoritmo puede agruparlos rápidamente.



MODELO DE MIXTURA GAUSSIANA

Analicemos las debilidades de k-means:

Si tenemos bloques separados de datos, el algoritmo puede agruparlos rápidamente.



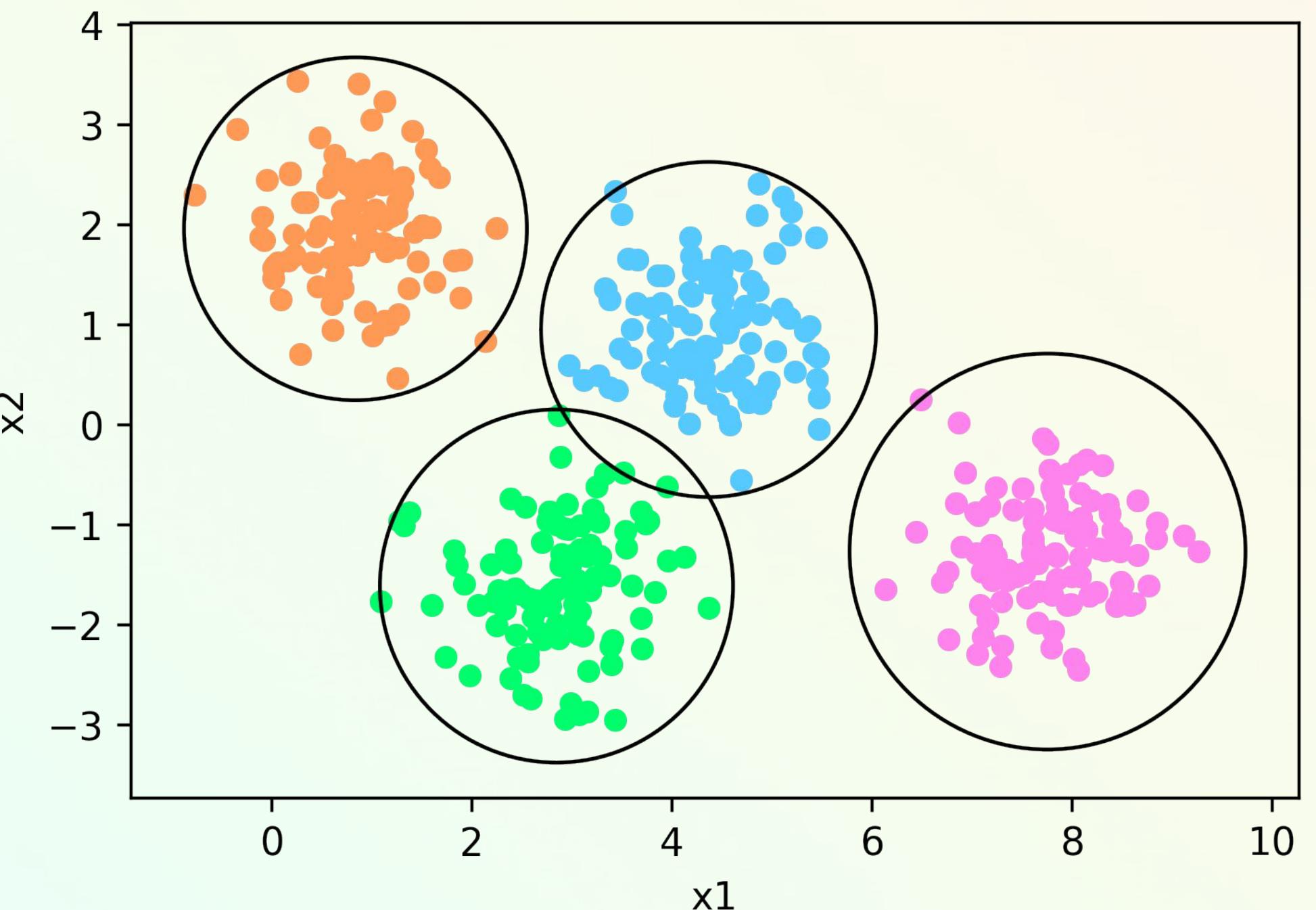
MODELO DE MIXTURA GAUSSIANA

Analicemos las debilidades de k-means:

Si tenemos bloques separados de datos, el algoritmo puede agruparlos rápidamente.

Hay etiquetados de cluster en observaciones que son más segura que otras, las menos seguras están es el espacio entre el cluster azul y el verde.

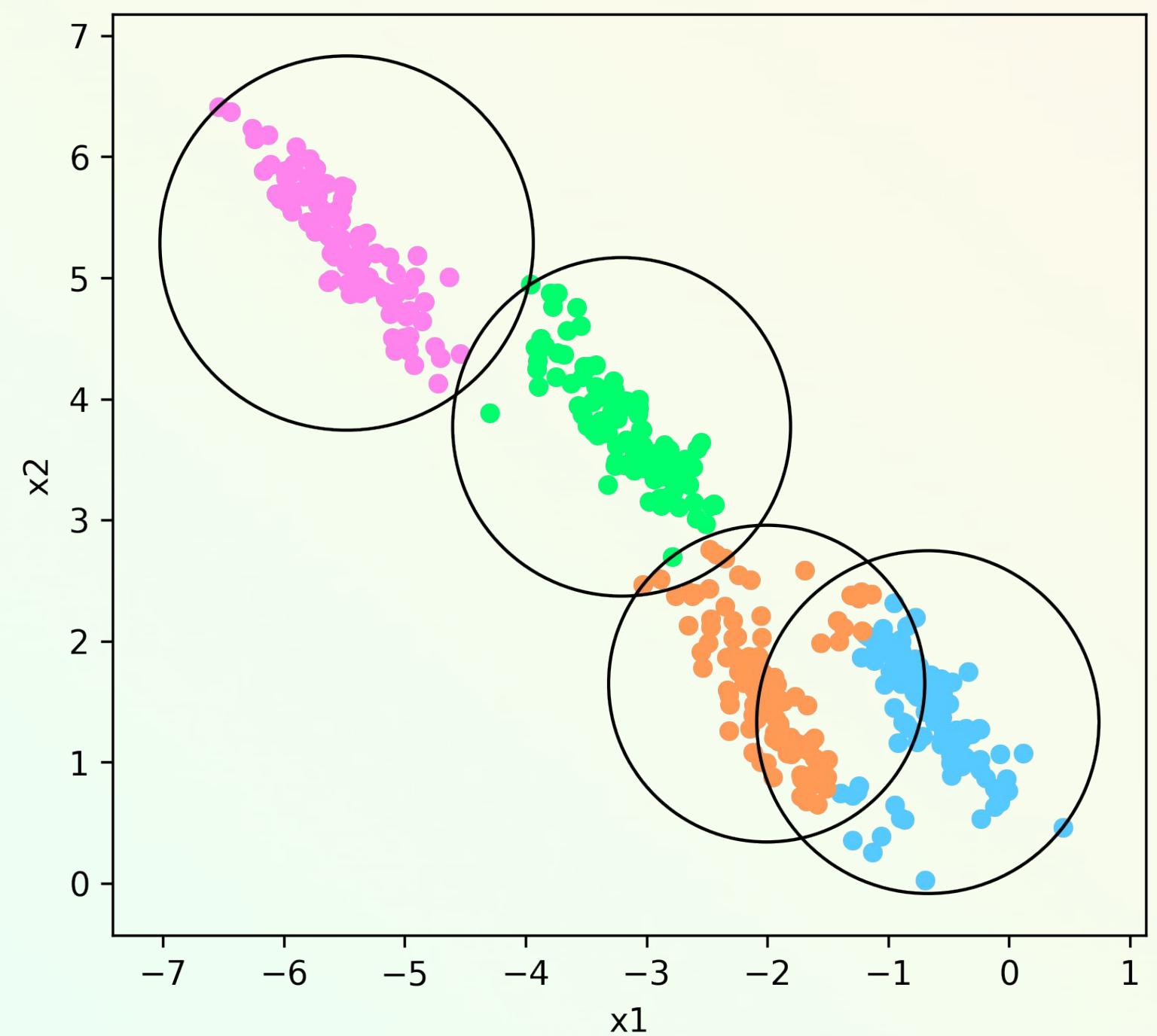
K-means **no tiene una medida de probabilidad o de incertidumbre.**



MODELO DE MIXTURA GAUSSIANA

Analicemos las debilidades de k-means:

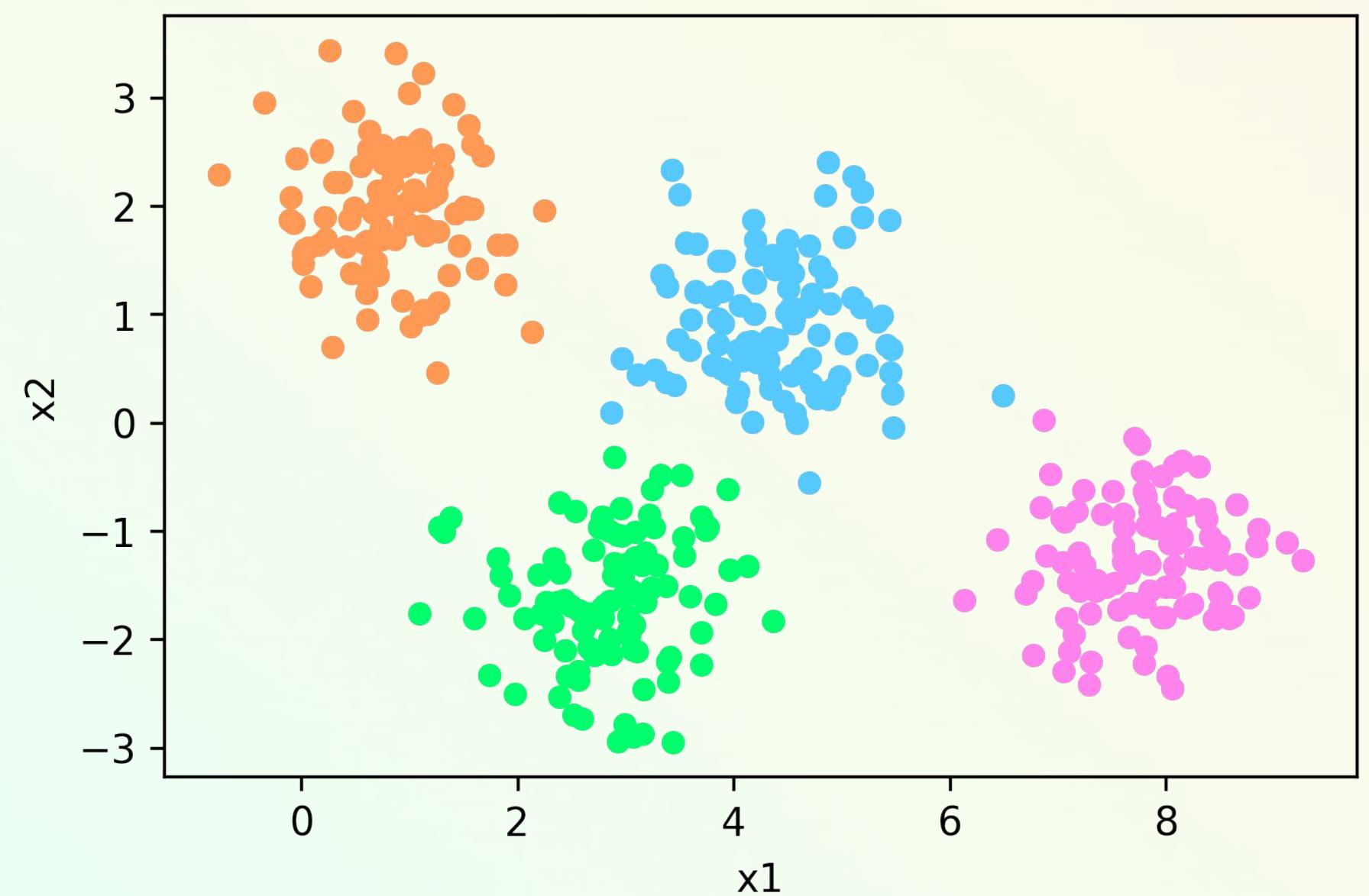
Un punto importante es que los clusters deben ser circulares... **k-means no es suficiente flexible para adaptarse a este tipo de casos.**



MODELO DE MIXTURA GAUSSIANA

Un *Gaussian Mixture Model* (GMM) intenta encontrar una mezcla de distribuciones de probabilidad gaussianas multidimensionales que modelen mejor cualquier conjunto de datos de entrada.

En el caso más simple, los GMM se pueden utilizar para encontrar conglomerados de la misma manera que k-means.

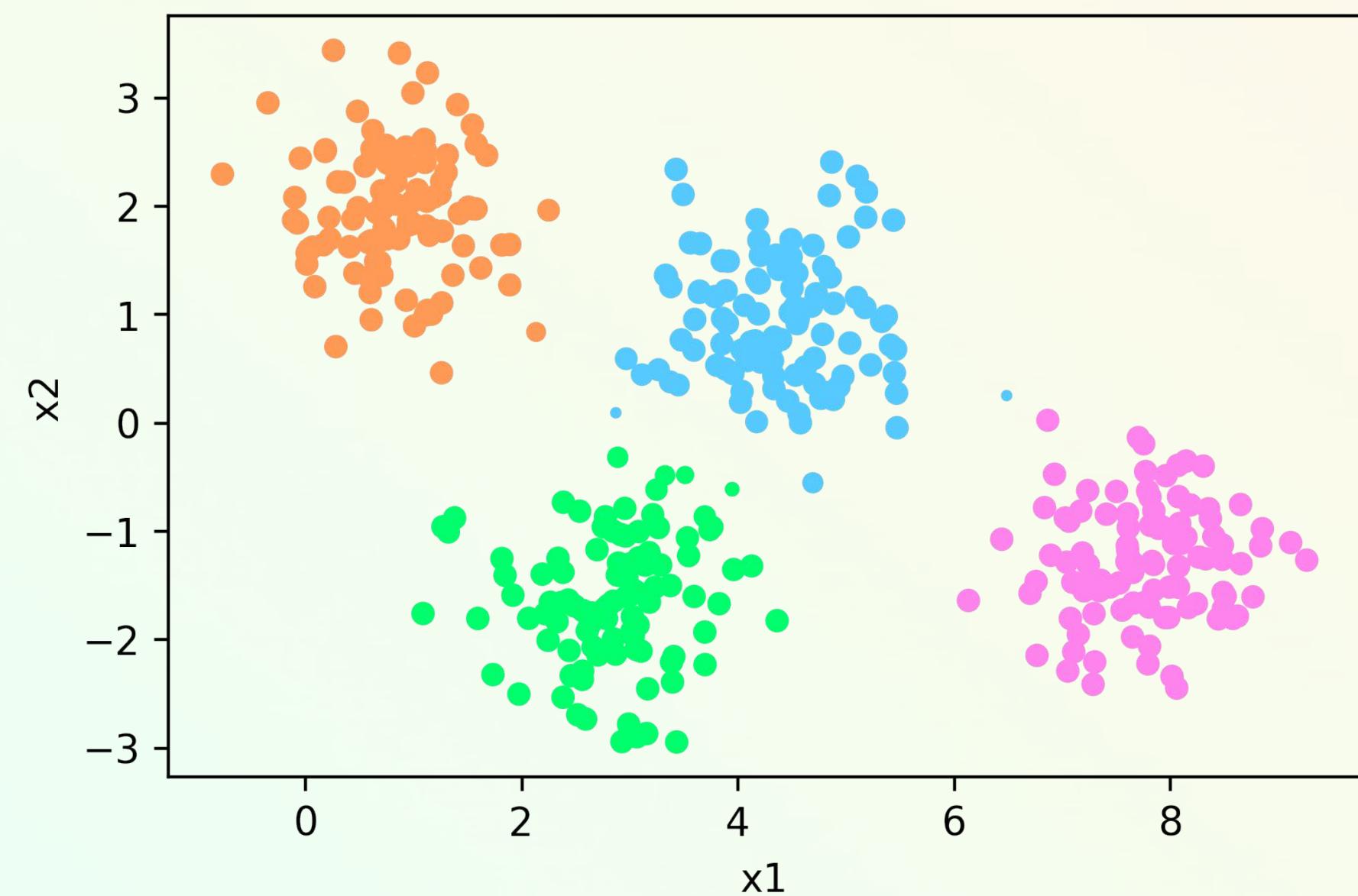


MODELO DE MIXTURA GAUSSIANA

Un *Gaussian Mixture Model* (GMM) intenta encontrar una mezcla de distribuciones de probabilidad gaussianas multidimensionales que modelen mejor cualquier conjunto de datos de entrada.

En el caso más simple, los GMM se pueden utilizar para encontrar conglomerados de la misma manera que k-means.

Pero debido a que GMM contiene un modelo probabilístico, también es posible encontrar asignaciones probabilísticas.

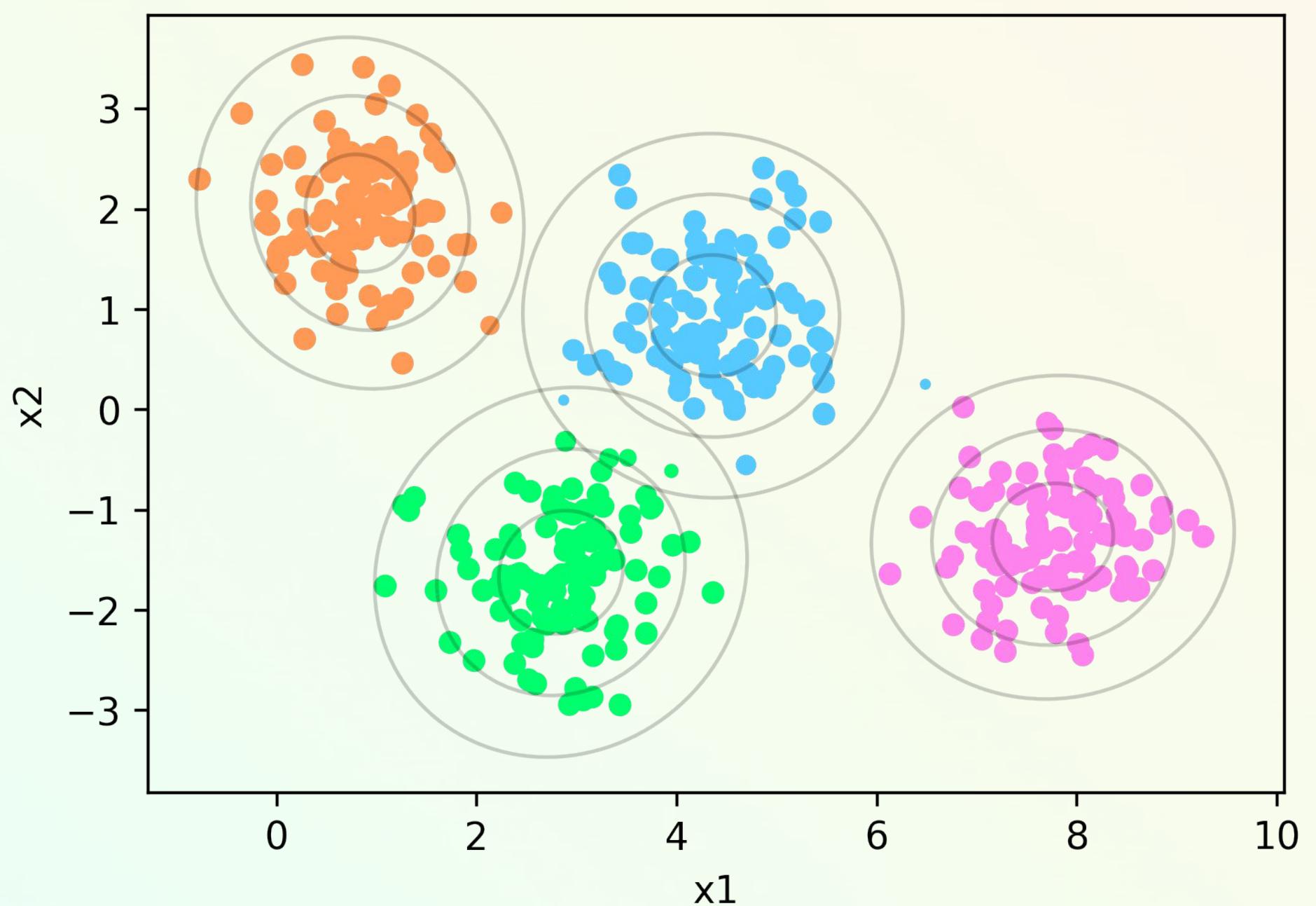


MODELO DE MIXTURA GAUSSIANA

Un *Gaussian Mixture Model* (GMM) intenta encontrar una mezcla de distribuciones de probabilidad gaussianas multidimensionales que modelen mejor cualquier conjunto de datos de entrada.

En el caso más simple, los GMM se pueden utilizar para encontrar conglomerados de la misma manera que k-means.

Pero debido a que GMM contiene un modelo probabilístico, también es posible encontrar asignaciones probabilísticas.

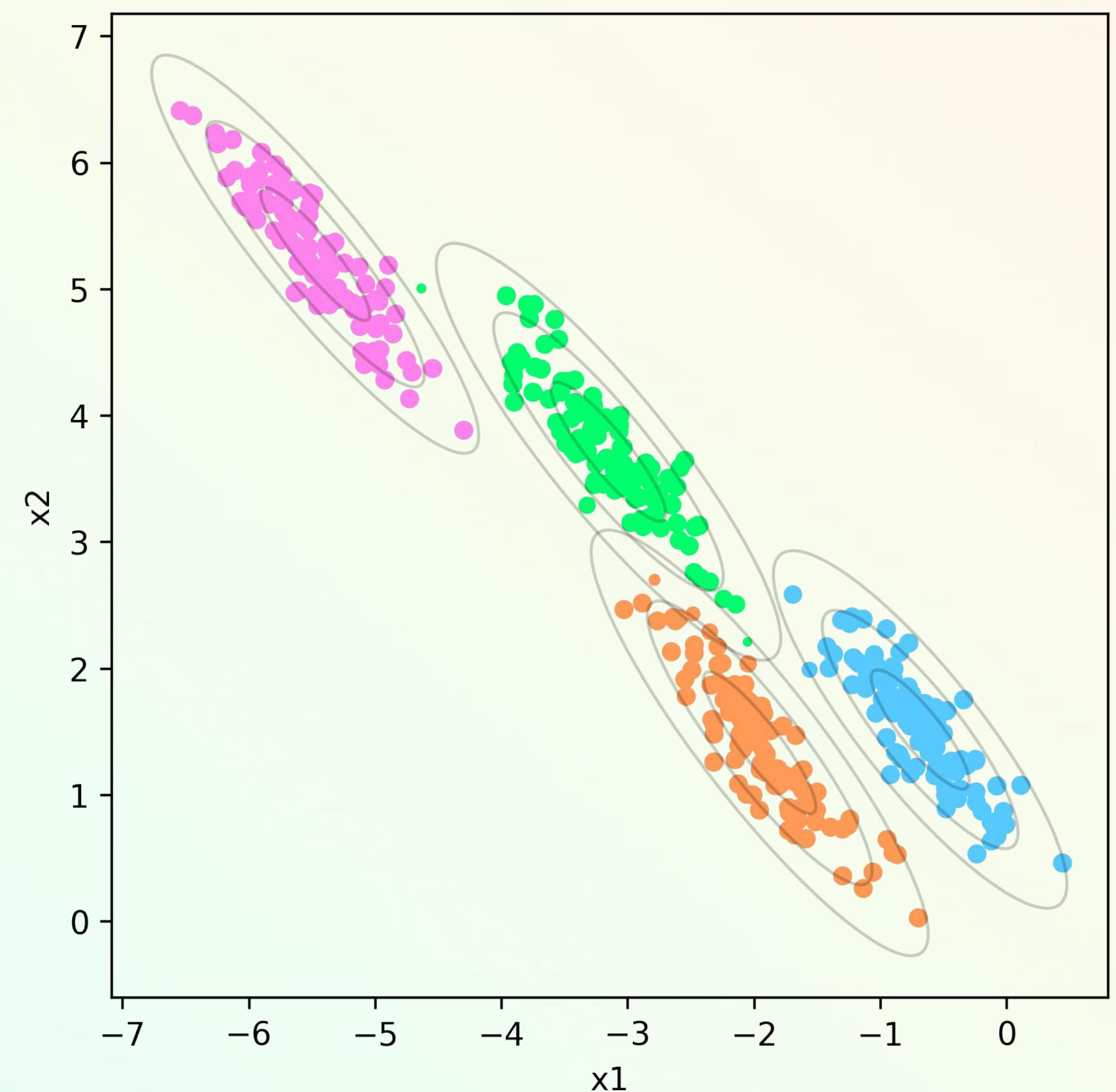


MODELO DE MIXTURA GAUSSIANA

Un *Gaussian Mixture Model* (GMM) intenta encontrar una mezcla de distribuciones de probabilidad gaussianas multidimensionales que modelen mejor cualquier conjunto de datos de entrada.

En el caso más simple, los GMM se pueden utilizar para encontrar conglomerados de la misma manera que k-means.

Pero debido a que GMM contiene un modelo probabilístico, también es posible encontrar asignaciones probabilísticas.



MODELO DE MIXTURA GAUSSIANA

Este modelo tiene la siguiente expresión:

$$f_X = \sum_{j=1}^k \phi_j f_{\mu_j \Sigma_j}$$

Donde $f_{\mu_j \Sigma_j}$ es una distribución normal multivariada con media μ_j y covarianza Σ_j .

Este modelo representa una suma pesada de k distribución normal multivariada cada una con peso ϕ_j .

Los valores de μ_j , Σ_j y ϕ_j son los parámetros por entrenar. Este entrenamiento se obtiene mediante el **algoritmo de máxima expectación (EM)** para optimizar con el **criterio de máxima verosimilitud**.

MODELO DE MIXTURA GAUSSIANA

Para simplificar, tomemos un caso de data unidimensional y dos clusters ($k=2$):

$$f(x | \mu_1 \sigma_1^2) \leftarrow \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-(x-\mu_1)^2 / 2\sigma_1^2} \quad \text{y} \quad f(x | \mu_2 \sigma_2^2) \leftarrow \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-(x-\mu_2)^2 / 2\sigma_2^2}$$

EM trabaja de la siguiente forma. A principio inicializa al azar los parámetros $\mu_1, \sigma_1^2, \mu_2, \sigma_2^2$ y hace que $\phi_1 = \phi_2 = \frac{1}{k}$.

MODELO DE MIXTURA GAUSSIANA

En cada iteración de EM los siguientes cuatro pasos se ejecutan:

1. Para cada observación $i=1,\dots,N$, se calcula la probabilidad:

$$f(x | \mu_1 \sigma_1^2) \leftarrow \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} \text{ y } f(x | \mu_2 \sigma_2^2) \leftarrow \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}}$$

2. Usando la regla de Bayes, para cada observación, calcula la probabilidad $b_i^{(j)}$ que la observación pertenece al cluster $j \in \{1,2\}$ (la probabilidad de que la observación sea extraída de la distribución normal j):

$$b_i^{(j)} \leftarrow \frac{f(x_i | \mu_j \sigma_j^2) \phi_j}{f(x_i | \mu_1 \sigma_1^2) \phi_1 + f(x_i | \mu_2 \sigma_2^2) \phi_2}$$

El parámetro ϕ_j refleja la probabilidad de que la distribución gaussiana j con parámetros μ_j y σ_j^2 haya producido la observación. Por eso iniciamos $\phi_j = \frac{1}{2}$ es la probabilidad a priori.

MODELO DE MIXTURA GAUSSIANA

En cada iteración de EM los siguientes cuatro pasos se ejecutan:

3. Computa nuevos valores de μ_j y σ_j^2 :

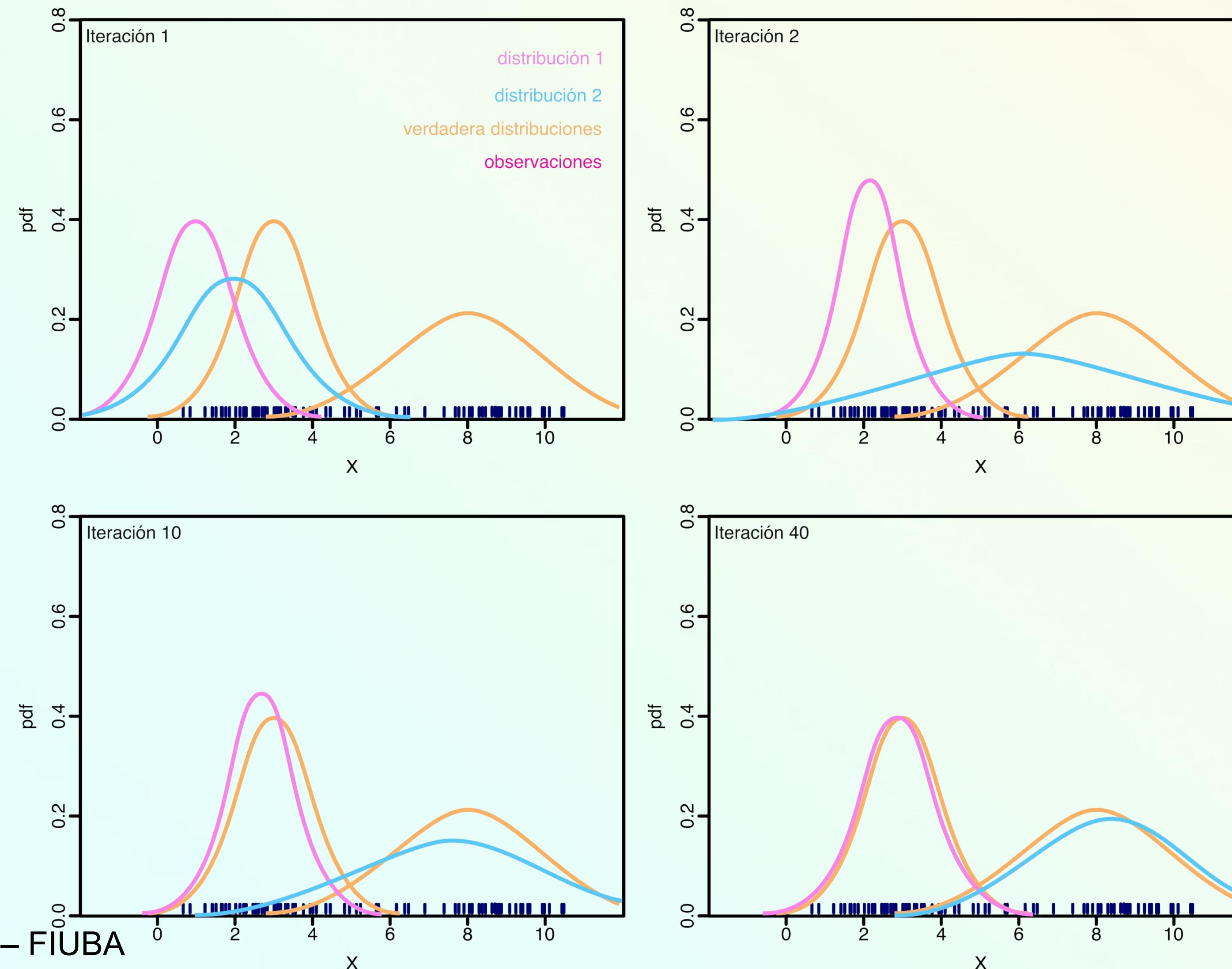
$$\mu_j \leftarrow \frac{\sum_{i=1}^N b_i^{(j)} x_i}{\sum_{i=1}^N b_i^{(j)}} \text{ y } \sigma_j^2 \leftarrow \frac{\sum_{i=1}^N b_i^{(j)} (x_i - \mu_j)^2}{\sum_{i=1}^N b_i^{(j)}}$$

4. Actualiza ϕ_j

$$\phi_j \leftarrow \frac{1}{N} \sum_{i=1}^N b_i^{(j)}$$

Este algoritmo continúa hasta que los valores de μ_j y σ_j^2 no cambian más (se establece un valor umbral).

MODELO DE MIXTURA GAUSSIANA

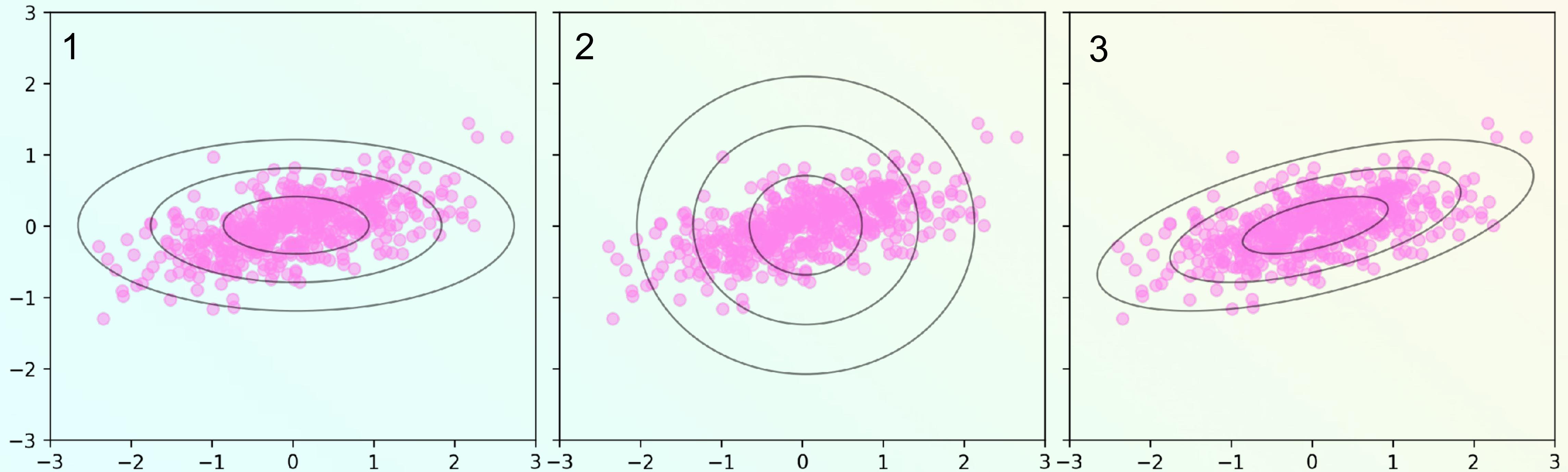


MODELO DE MIXTURA GAUSSIANA

Un detalle importante a tener el caso multidimensional es que tenemos la matriz de covarianzas y a esta la podemos definir de diferentes formas (que terminarán siendo un hiper-parámetro más):

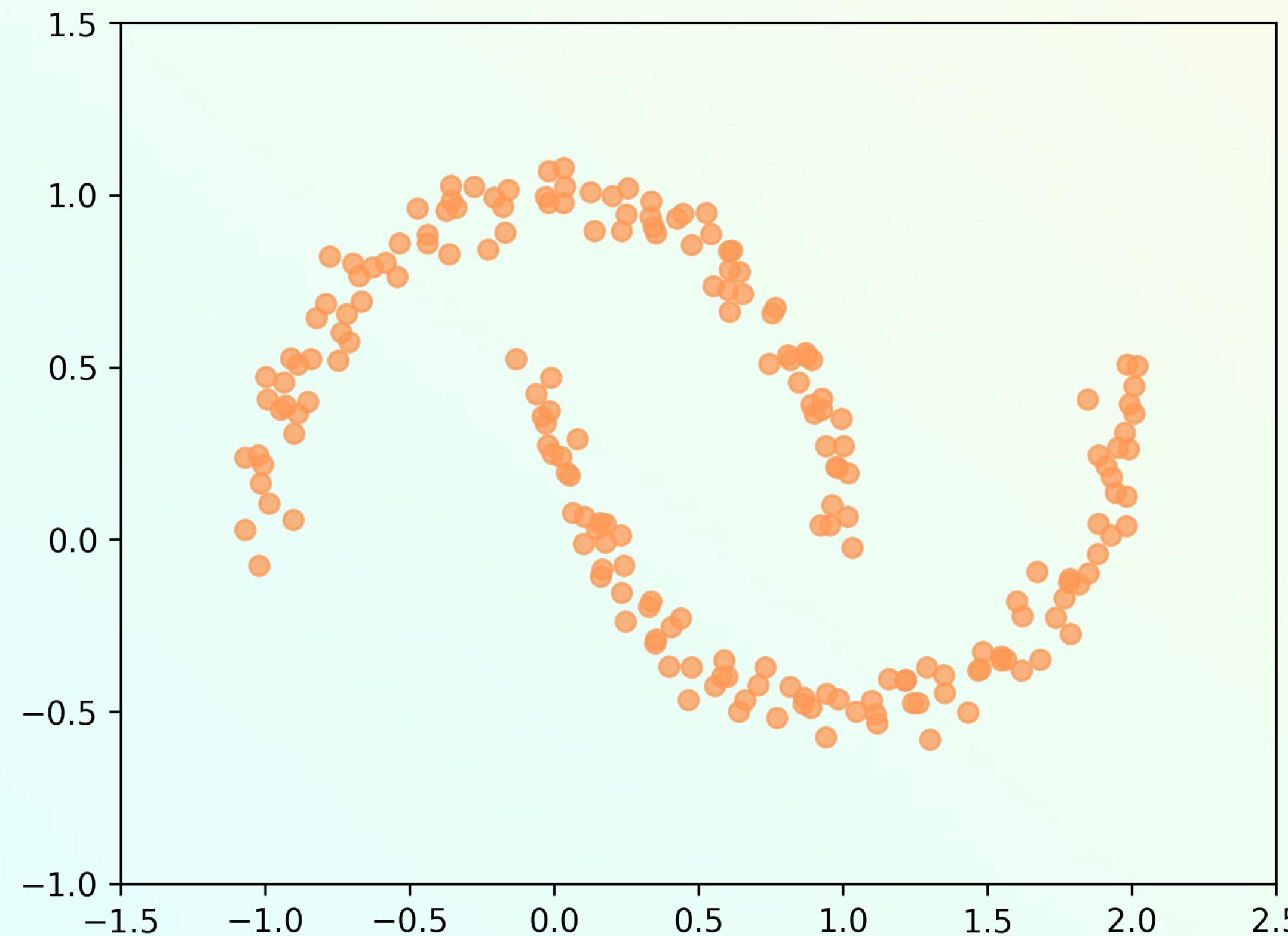
1. El tamaño del cluster va a ser independiente para cada dimensión.
2. El tamaño del cluster es igual en todas dimensiones (establece clusters similares a los de k-means).
3. Hay covarianza entre los atributos haciendo que la distribución se oriente en cualquier sentido. Es el más caro de calcular computacionalmente.

MODELO DE MIXTURA GAUSSIANA



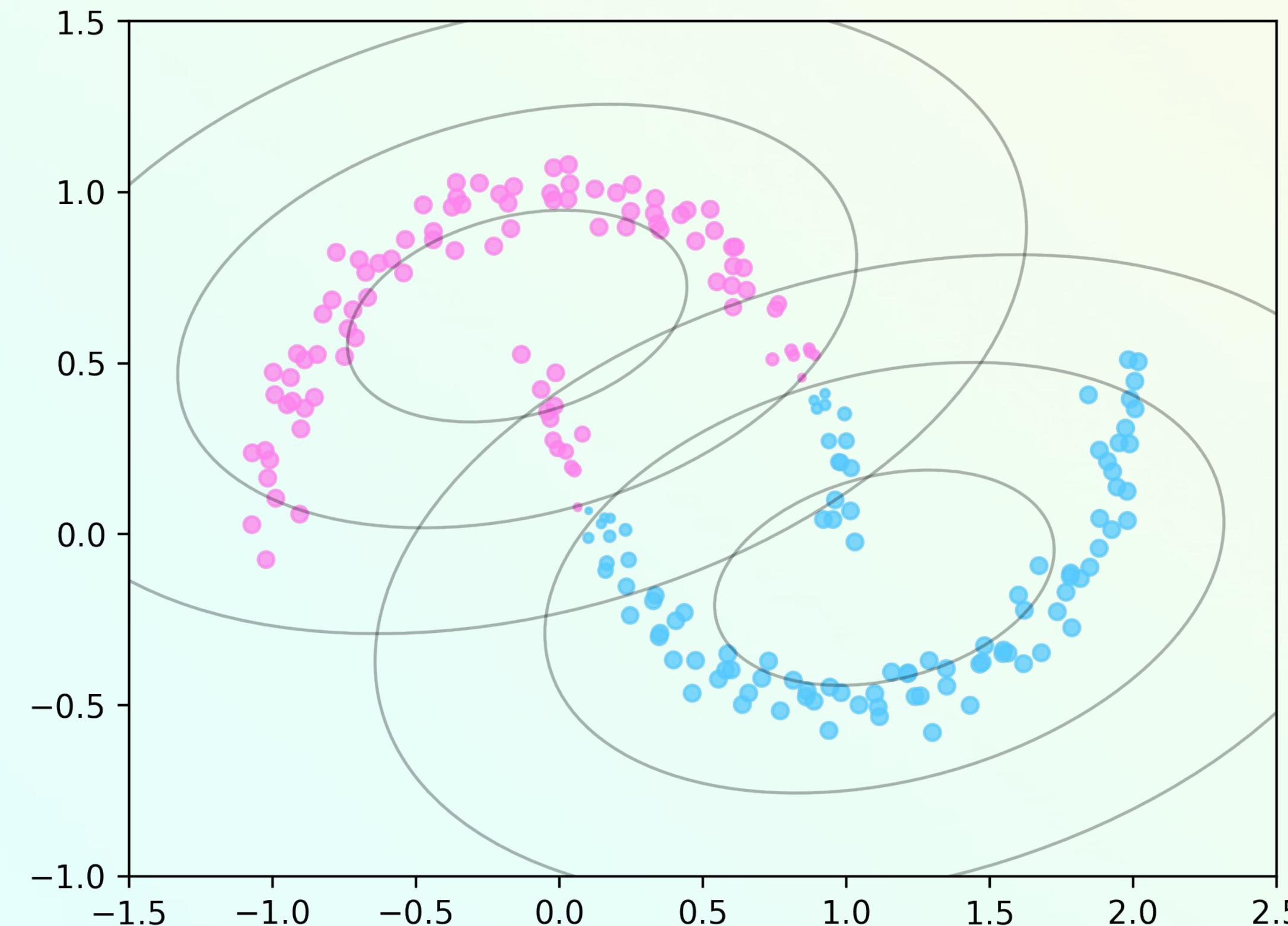
MODELO DE MIXTURA GAUSSIANA

Estos modelos pueden servirnos no solo para armar clusters, sino para **estimar densidades**.



MODELO DE MIXTURA GAUSSIANA

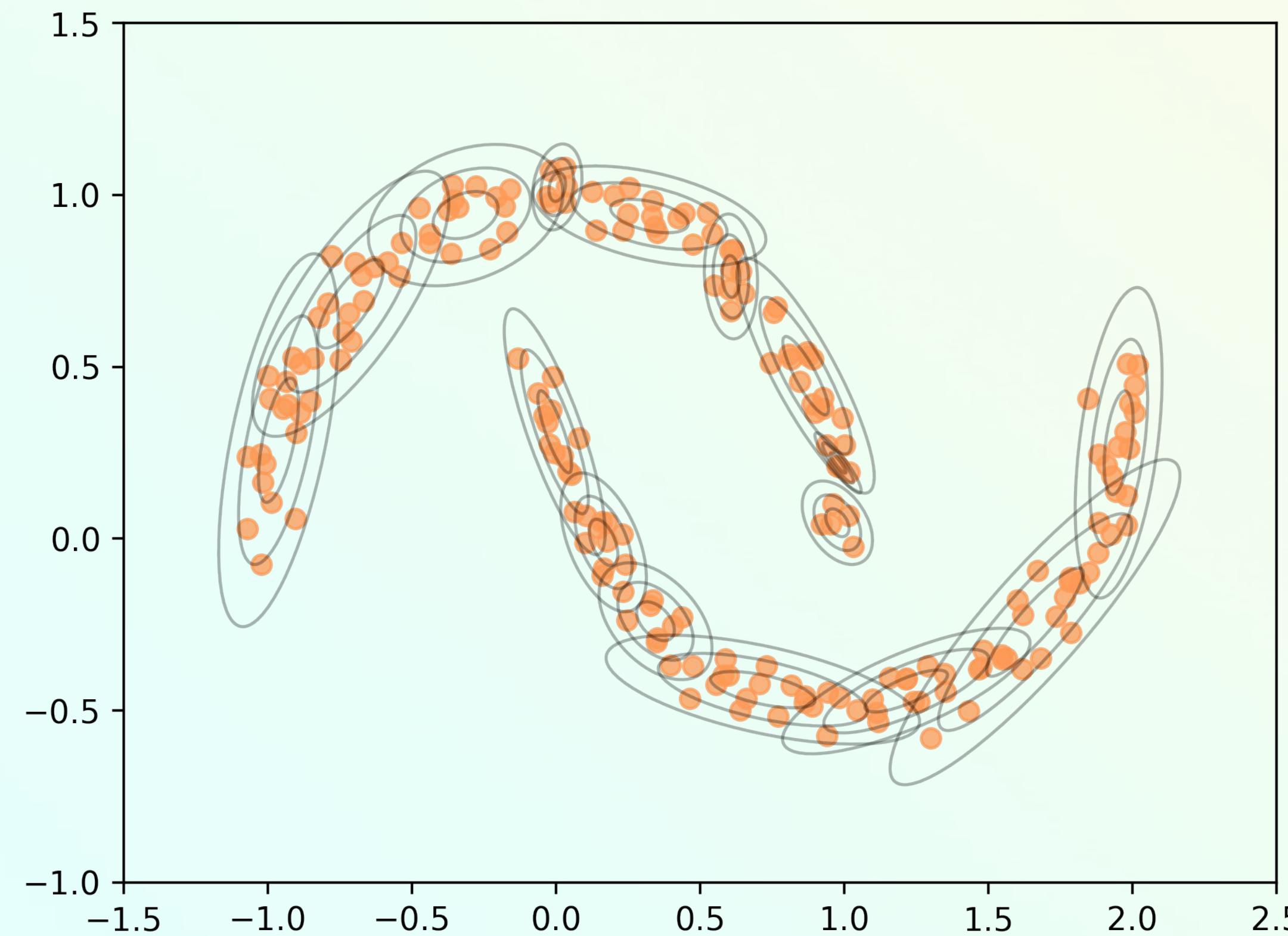
Estos modelos pueden servirnos no solo para armar clusters, sino para **estimar densidades**.



En este caso no genera buenos clusters.

MODELO DE MIXTURA GAUSSIANA

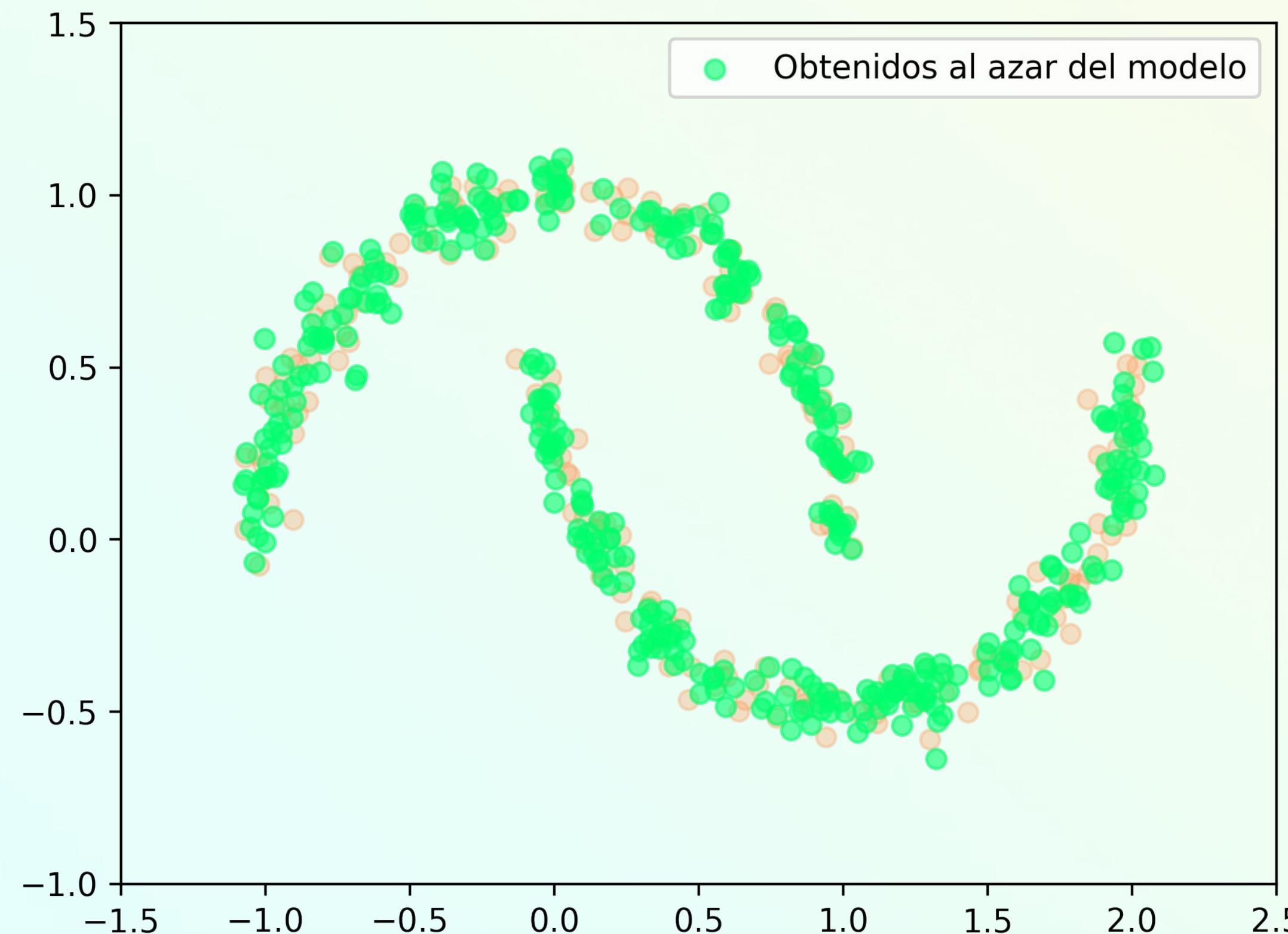
Estos modelos pueden servirnos no solo para armar clusters, sino para **estimar densidades**.



Pero podemos usarlo para modelar la distribución (i.e. con 16 componentes)

MODELO DE MIXTURA GAUSSIANA

Estos modelos pueden servirnos no solo para armar clusters, sino para **estimar densidades**.



Y una vez obtenido el modelo, podemos generar nuevas muestras al azar de nuestra distribución.

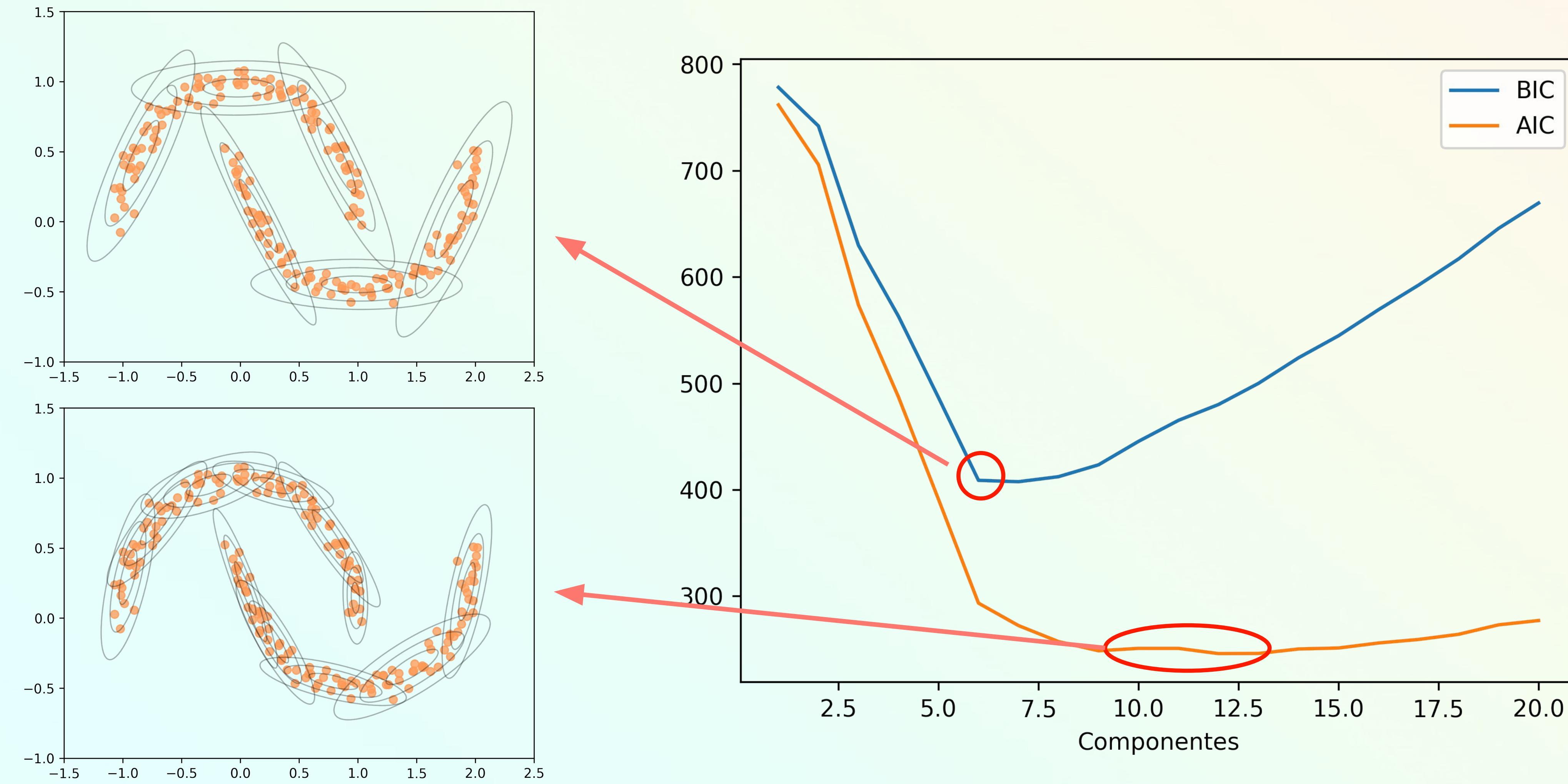
MODELO DE MIXTURA GAUSSIANA

Similar al caso de k-means, establecer la cantidad de clusters es un paso inicial para aplicar este modelo. Podemos resolverlo con las técnicas que vimos, particularmente con **Fuerza de predicción**.

Para el caso de estimación de densidad se puede hacer uso del criterio de información de Akaike (AIC) o el criterio de información Bayesiana (BIC).

Se ajustan varios modelos GMM con distintos valores de K (por ejemplo, de 1 a 10), se calculan el AIC o el BIC para cada uno, y el mejor K es el que minimiza esos valores.

MODELO DE MIXTURA GAUSSIANA



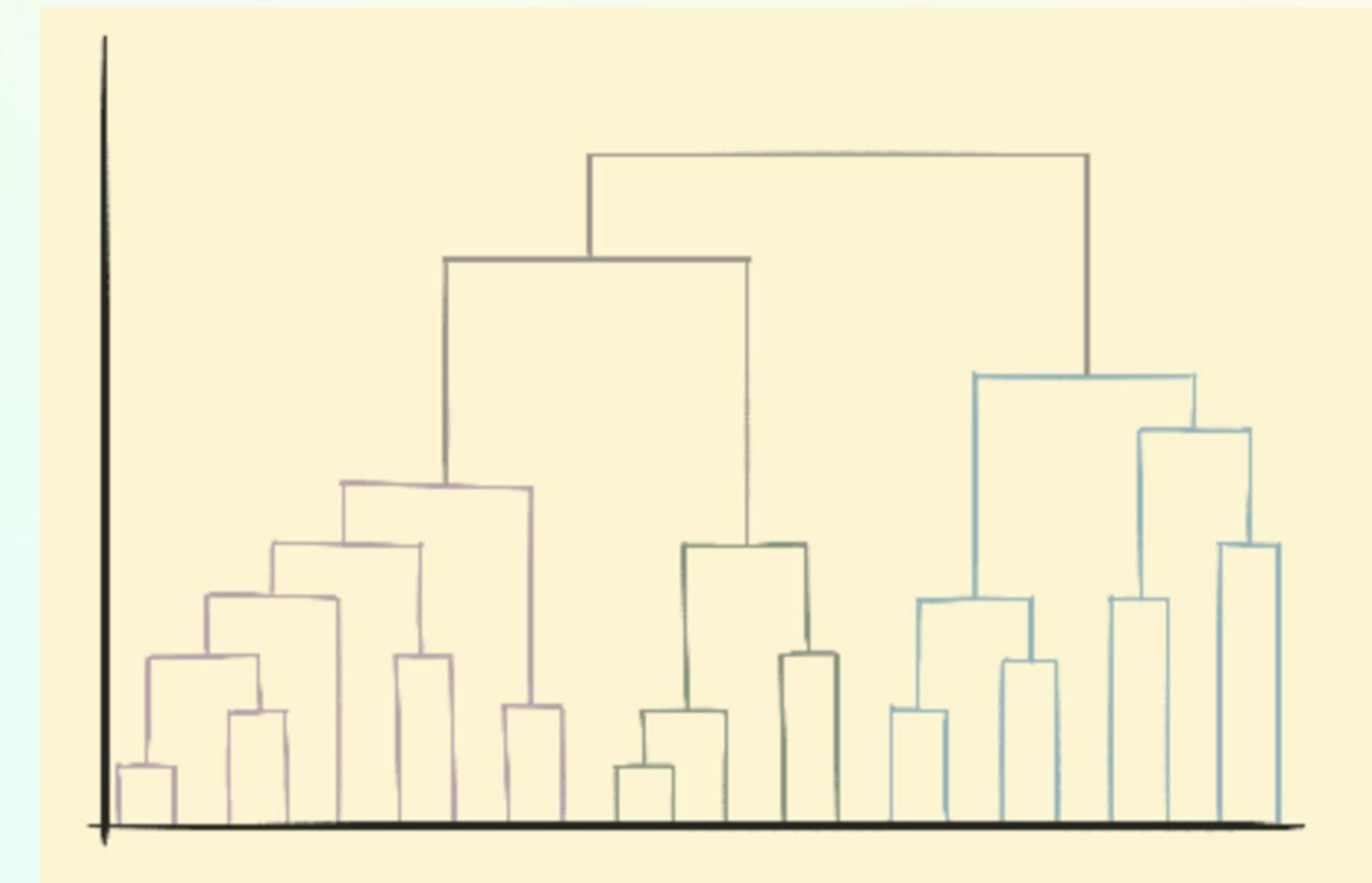
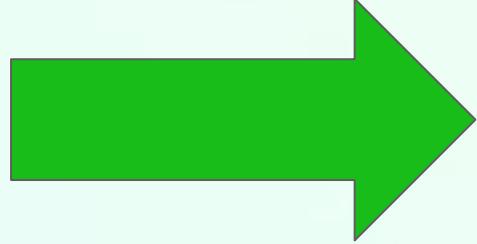
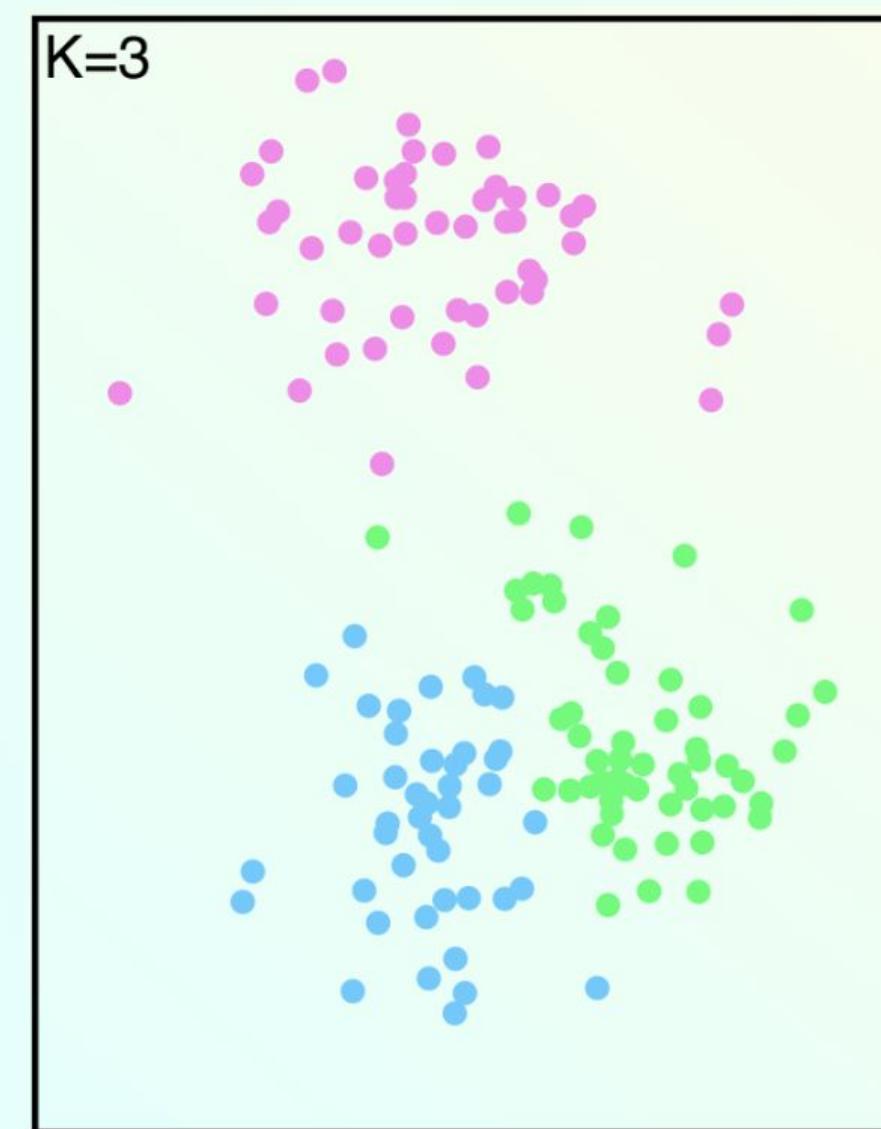
VAMOS A PRACTICAR UN POCO...

HIERARCHICAL CLUSTERING

HIERARCHICAL CLUSTERING

Una (potencial) desventaja de los modelos anteriores es que requieren que especifiquemos el número de clusters.

El clustering jerárquico es una alternativa que no tiene ese prerequisito.



HIERARCHICAL CLUSTERING

El **clustering jerárquico** es un algoritmo de aprendizaje automático no supervisado utilizado para agrupar datos en función de su similitud.

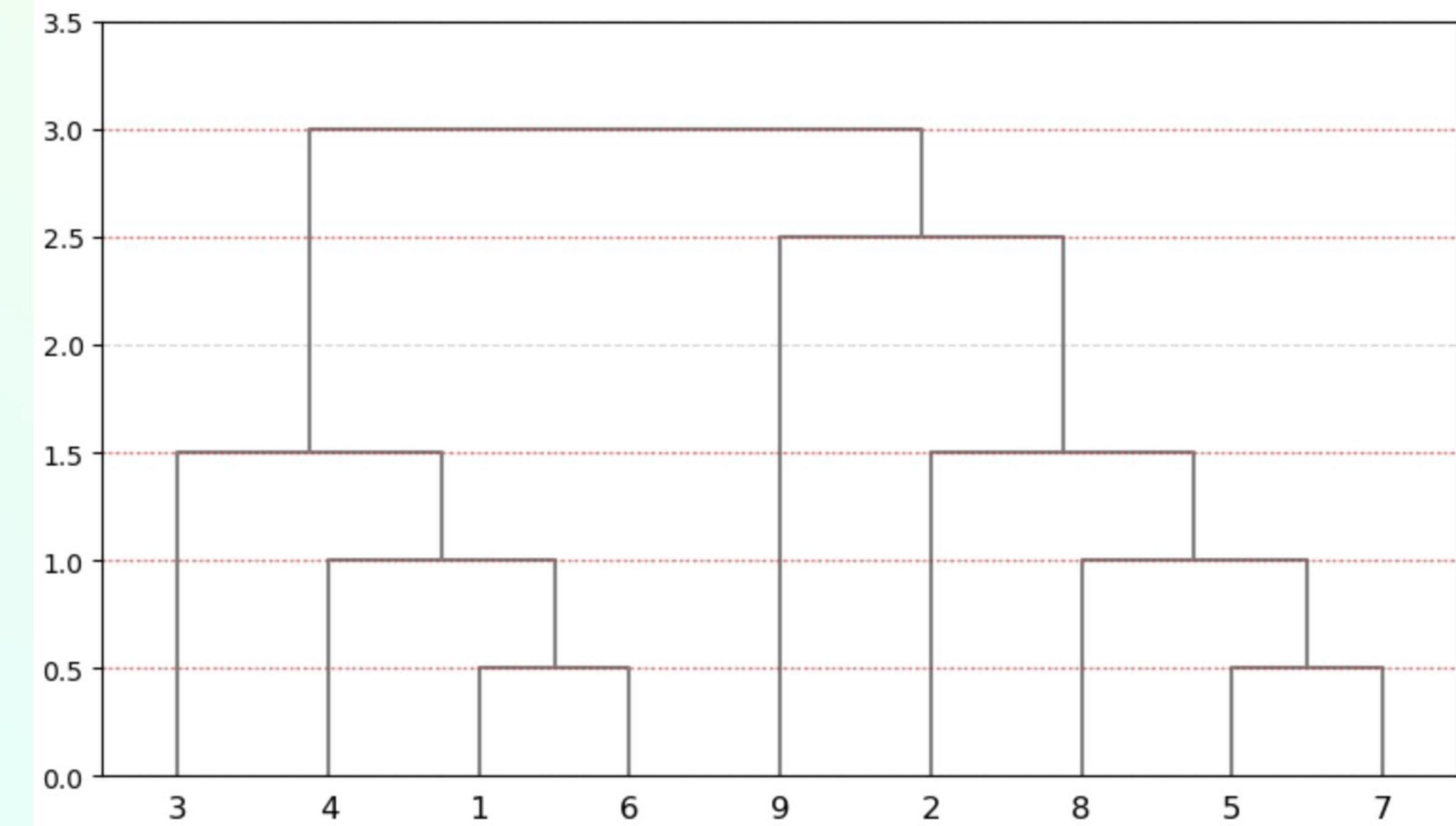
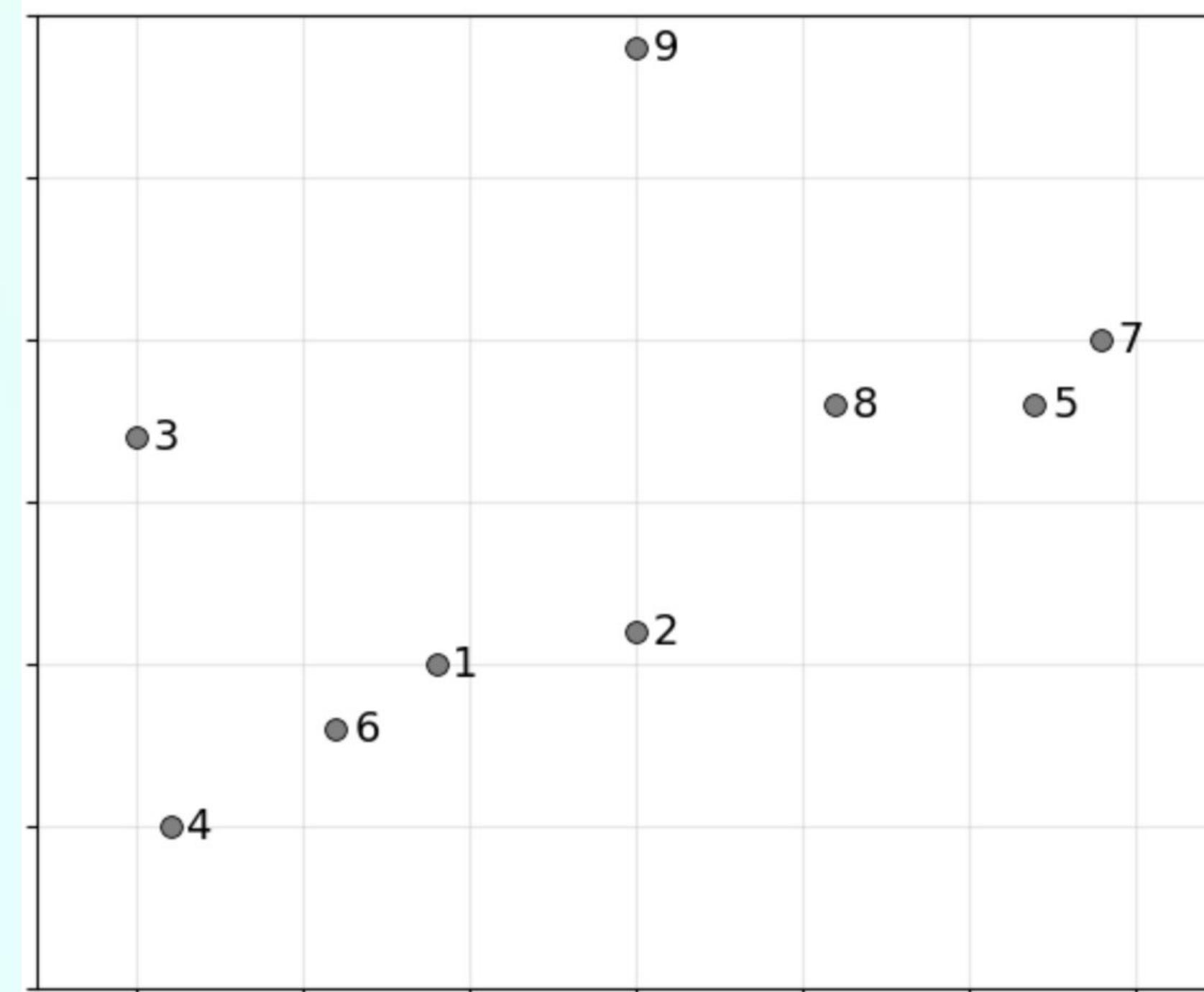
Su objetivo es organizar las observaciones en una estructura jerárquica tipo árbol, donde los elementos más parecidos se agrupan primero y los menos similares se combinan en niveles superiores.

Este método permite visualizar las relaciones entre los datos mediante un **dendrograma** y explorar diferentes números de clusters sin necesidad de definirlos previamente, lo que facilita la interpretación y el descubrimiento de patrones en los datos.

HIERARCHICAL CLUSTERING

Dendrogramas

Un dendrograma es una representación visual de los datos y la jerarquía de clusters a la que pertenecen. Cada hoja es una observación.

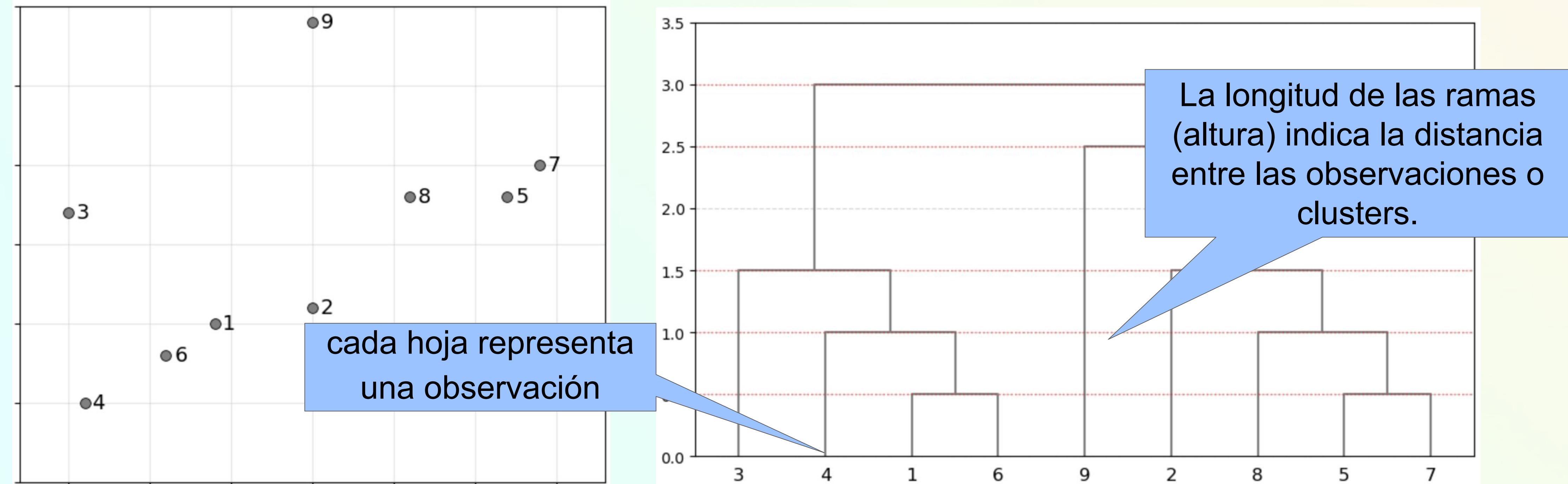


HIERARCHICAL CLUSTERING

Dendrogramas

A medida que subimos en el árbol, algunas hojas empiezan a fusionarse en ramas, y luego las ramas se fusionan con hojas o con otras ramas.

Las fusiones corresponden a observaciones que son similares entre sí.

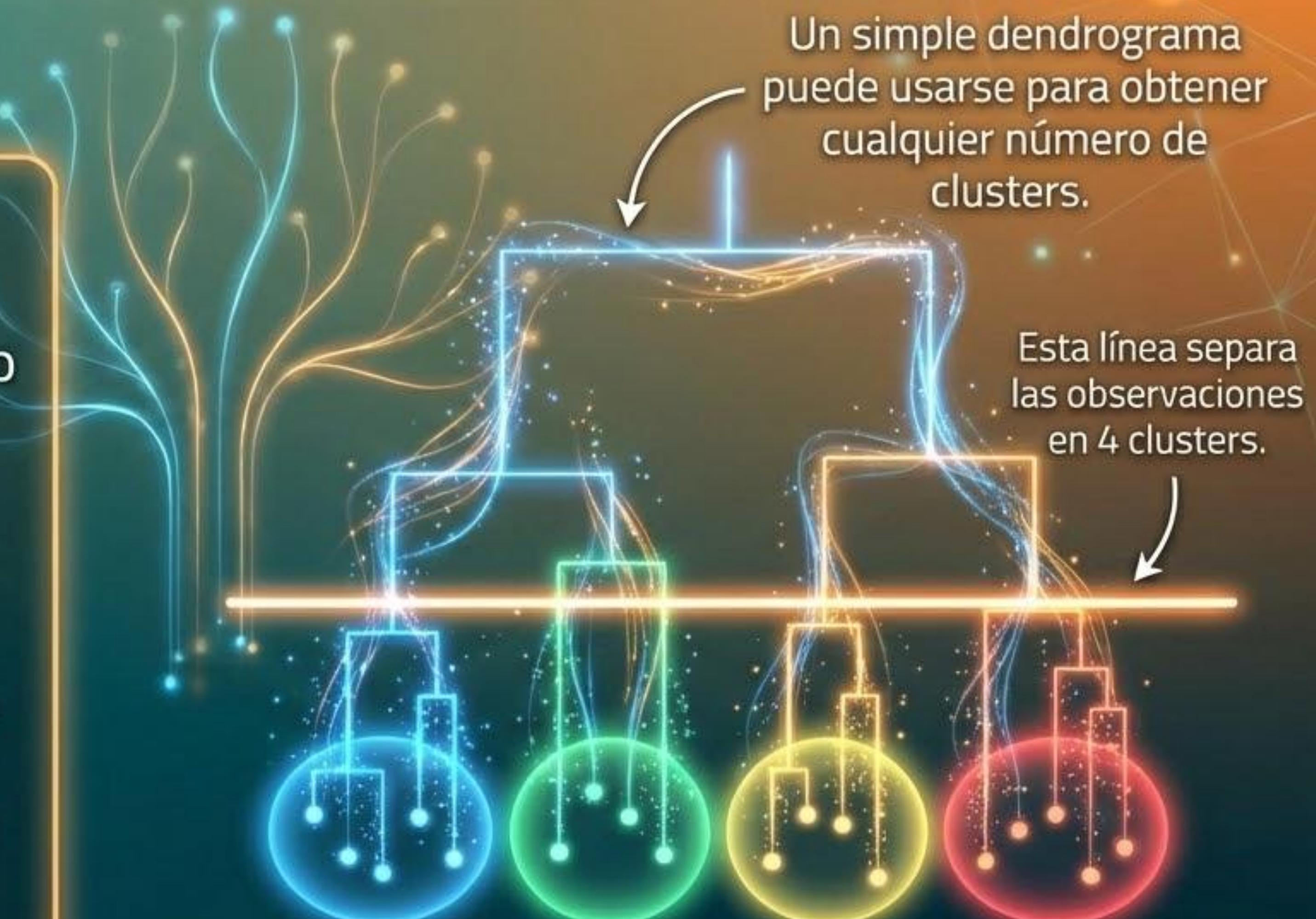


HIERARCHICAL CLUSTERING

Dendrogramas

→ Gráficamente, se pueden identificar diferentes números de clusters trazando una línea horizontal y deslizándola hacia arriba o hacia abajo.

| La altura del corte determina el número de clusters (# de líneas verticales atravesadas por el corte).



HIERARCHICAL CLUSTERING

El algoritmo que construye el clustering jerárquico consiste en combinar iterativamente **clusters similares**.

Para determinar qué tan similares son dos clusters, necesitamos tener definidos en primer lugar:

- un método para medir la **distancia** intra-cluster, es decir, la distancia entre dos observaciones de un mismo cluster. Por ejemplo, la distancia euclídea.
- un **linkage** o método para medir la distancia inter-cluster. El linkage se calcula a partir de las distancias individuales entre los elementos de los clusters.

HIERARCHICAL CLUSTERING

El algoritmo consiste en los siguientes pasos:

1. Al inicio tratar a cada observación como un cluster separado.
2. Luego iterar:
 - i. Computar la distancia (usando el linkage elegido) entre todos los pares de clusters.
 - ii. Identificar el par de clusters con menor distancia inter-cluster (es decir, los dos más similares) y fusionarlos. Guardar el valor de la distancia.
 - iii. Si todavía hay más de un cluster, repetir los pasos.

En el caso del linkage **Ward**, la distancia inter-cluster es el **aumento de la SSE** al combinarlos. Este método, como k-means, busca clusters compactos y homogéneos.

VAMOS A PRACTICAR UN POCO...

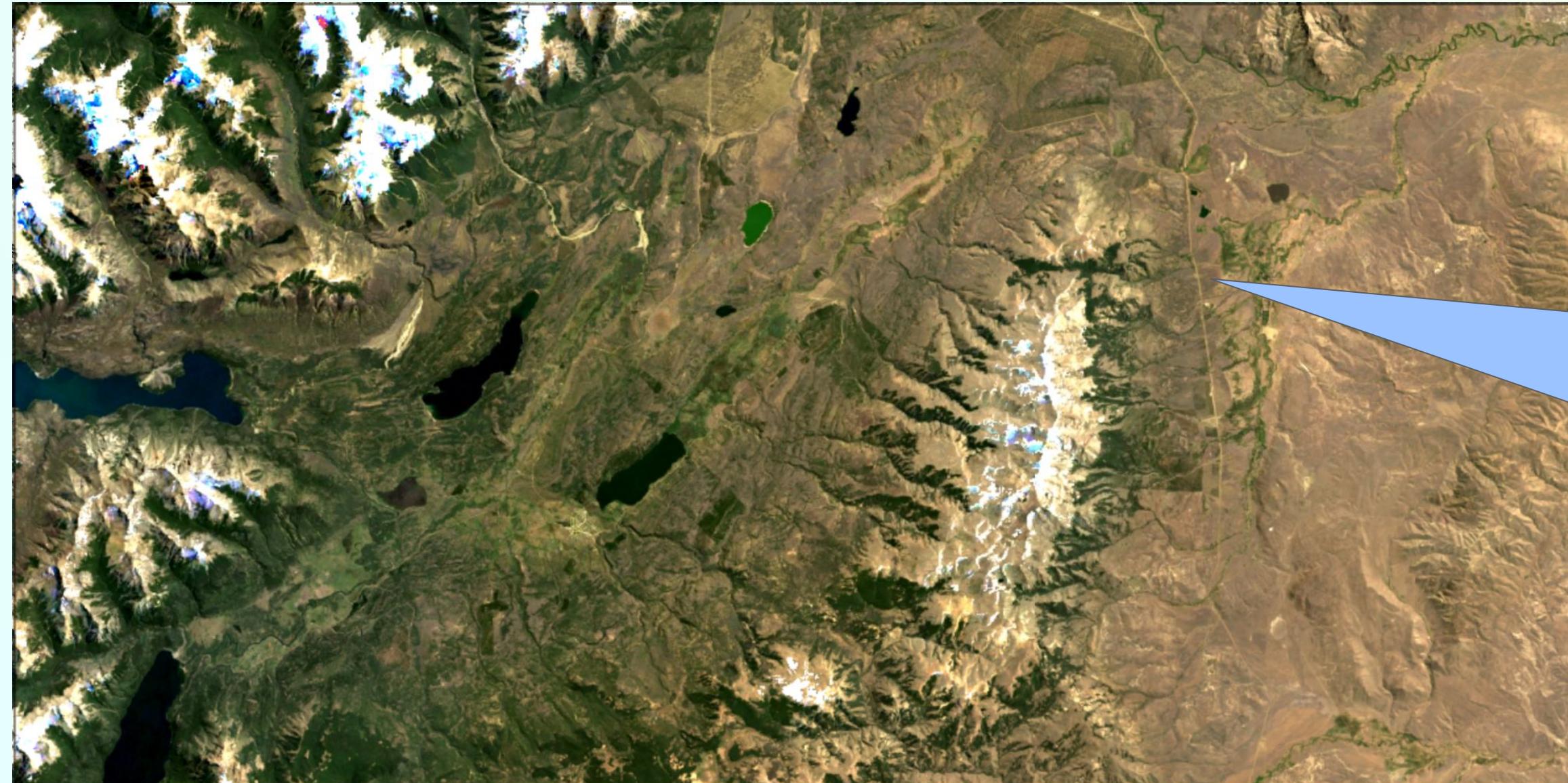
EJEMPLO PRÁCTICO

Un ejemplo de aplicación de estos algoritmos es la clasificación de tipos de coberturas de suelo: valle, montaña, cuerpos de agua, etc...



EJEMPLO PRÁCTICO

Un ejemplo de aplicación de estos algoritmos es la clasificación de tipos de coberturas de suelo: valle, montaña, cuerpos de agua, etc...

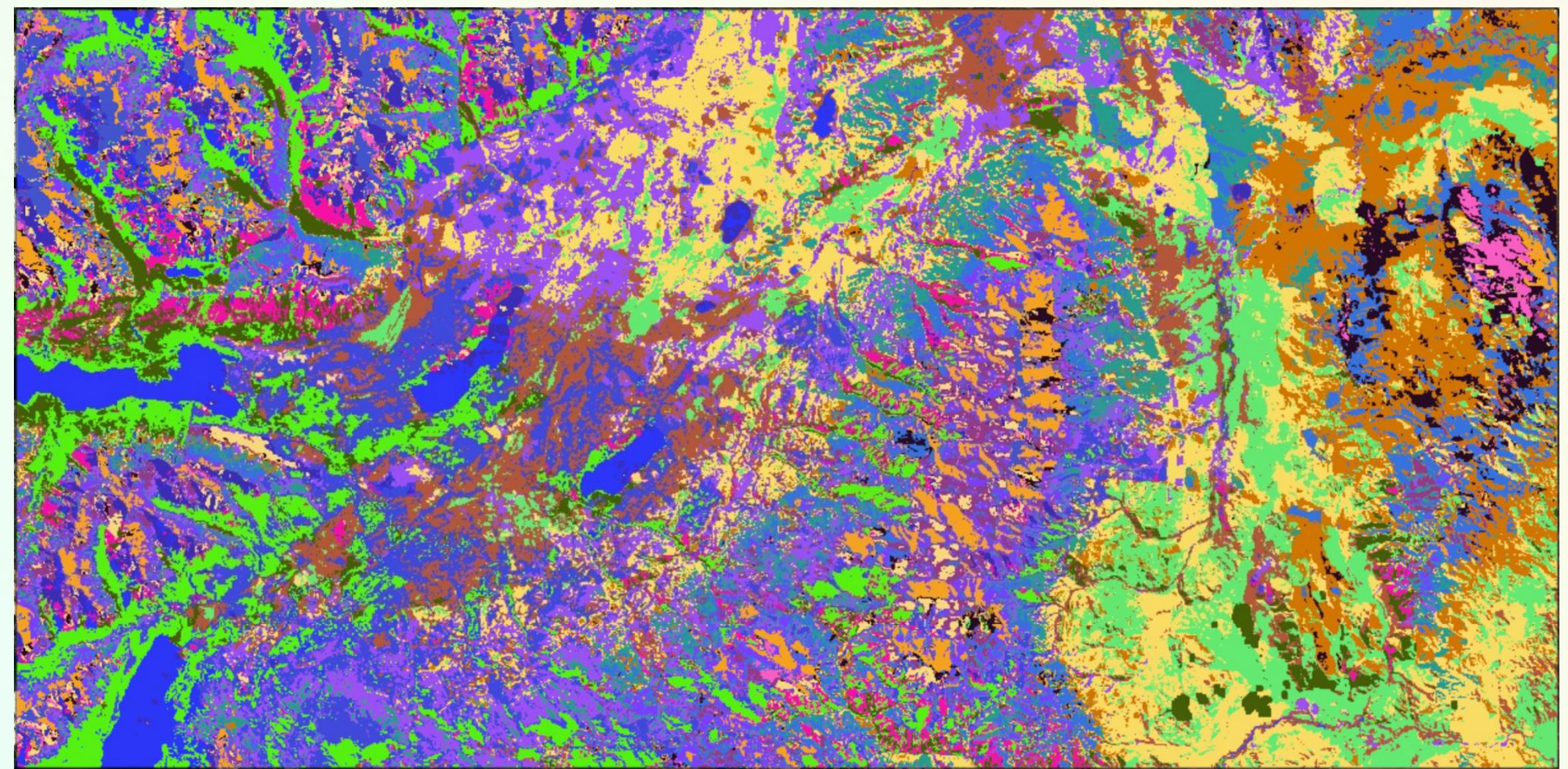


En una imagen satelital un pixel (observación) puede tener asociados muchos valores (features), entre ellos los valores del espectro visible (RGB).

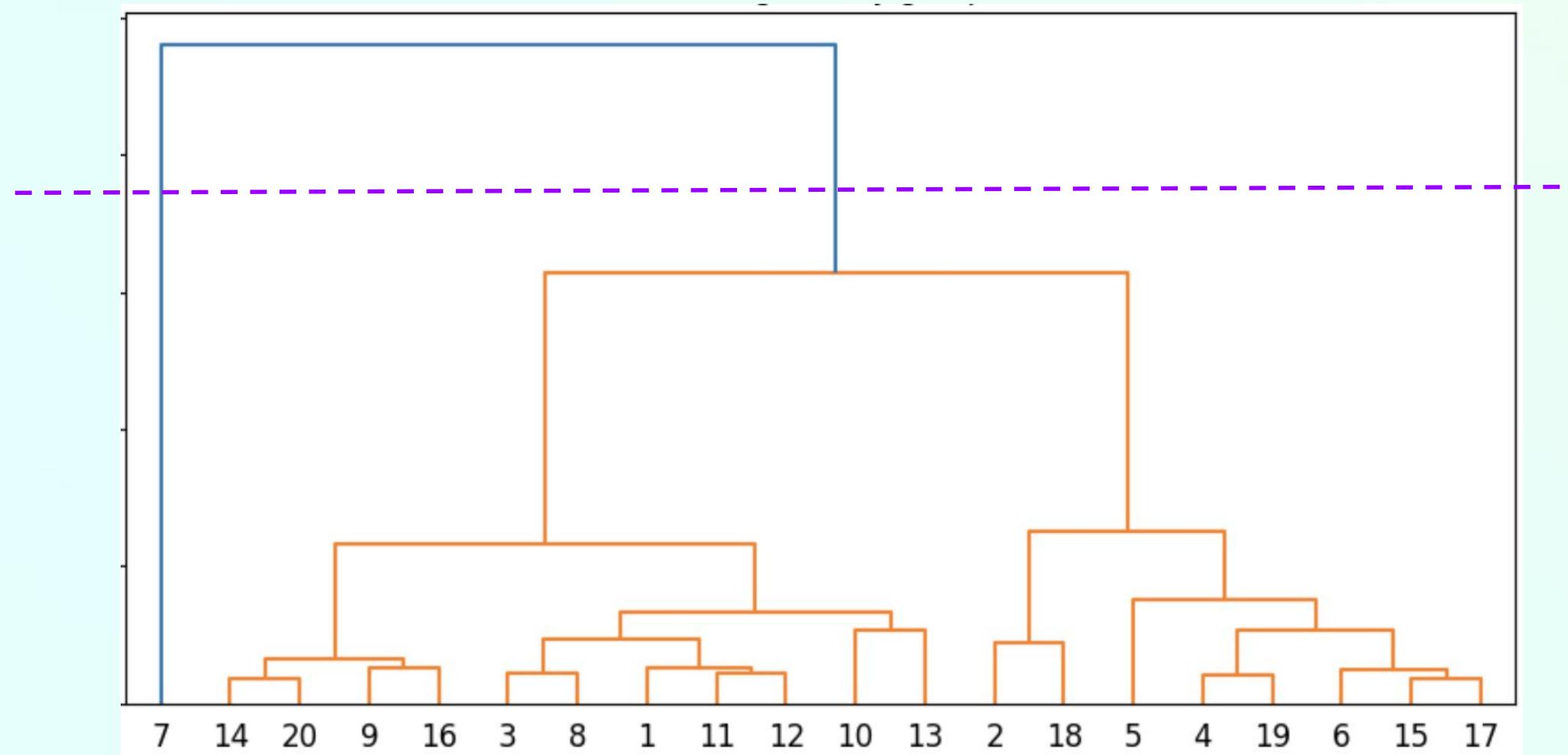
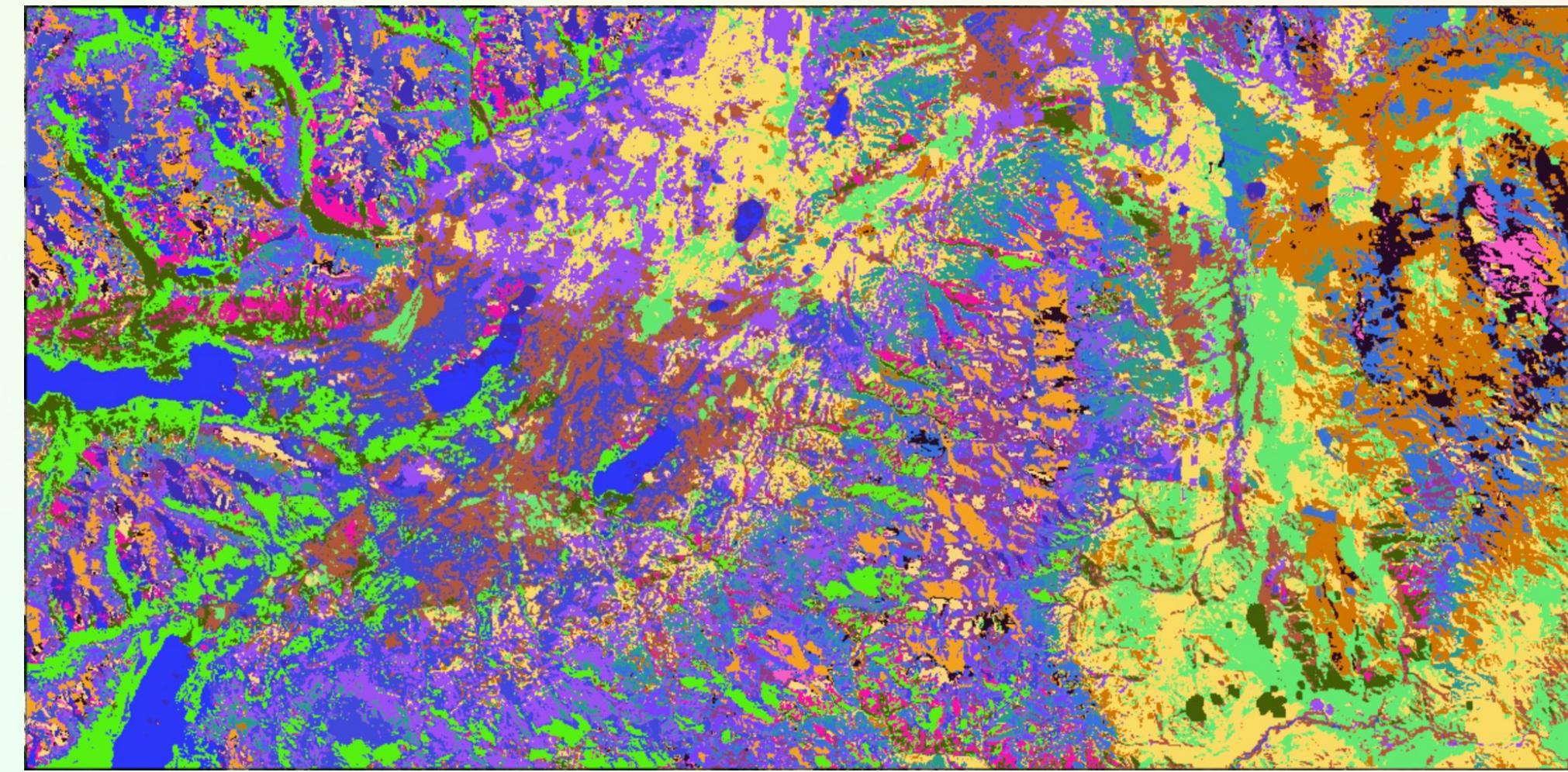
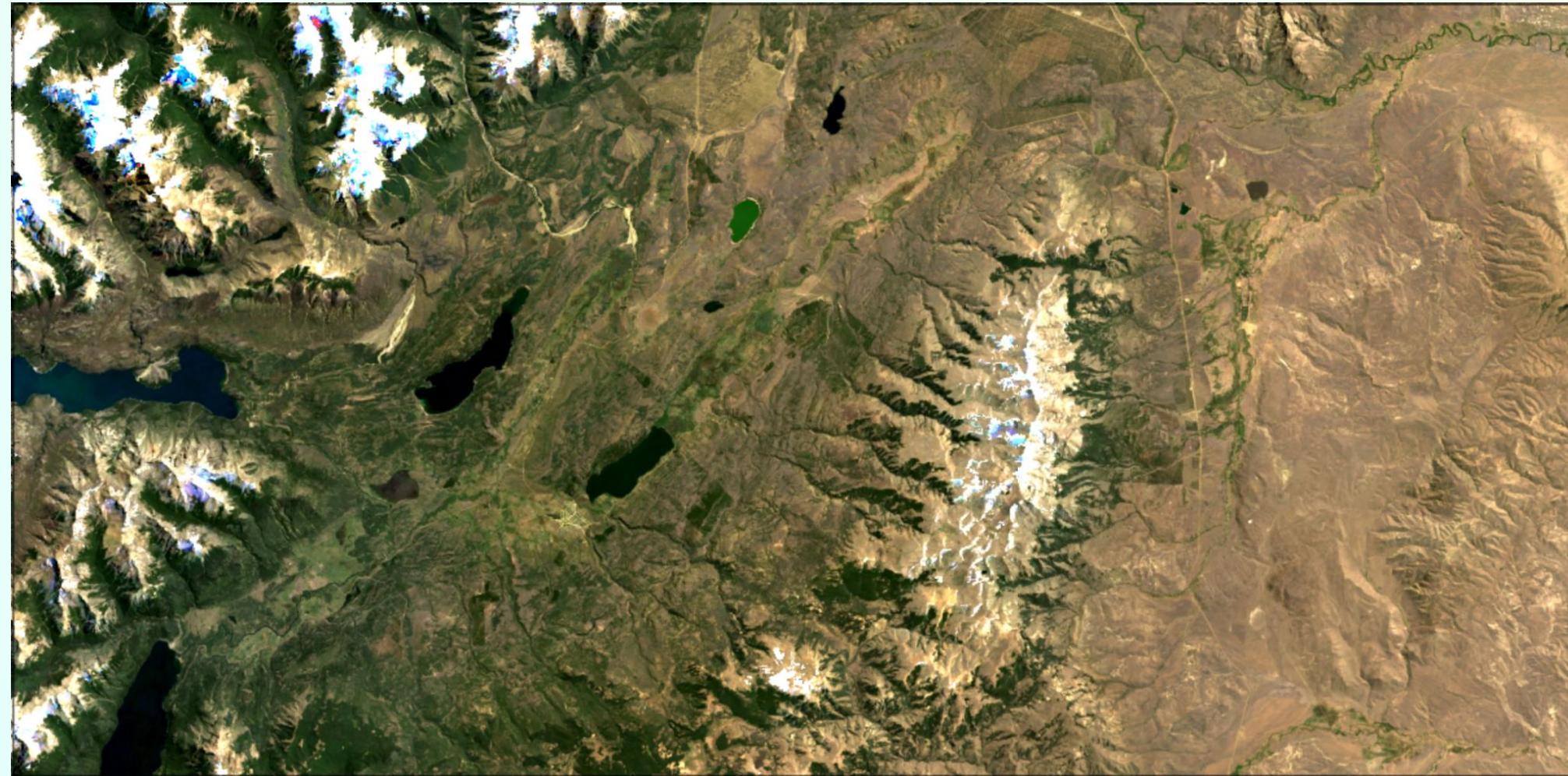
Quiero clasificar los píxeles de la imagen. Qué método puedo usar?

EJEMPLO PRÁCTICO

Puedo tomar una muestra de 1000 píxeles y con k-means hacer una primera clasificación. El ejemplo de la derecha es el resultado de hacer una clasificación con **k=20**.



Esas 20 clases son mis nuevas observaciones. Con ellas hago un clustering jerárquico.



Según el corte horizontal que haga en el dendograma, puedo obtener distinto nivel de detalle en la clasificación de los tipos de cobertura de suelo.

