

# Machine Learning en **Spark** Aprendizaje No Supervisado

*Rafael Caballero Roldán*

+ Spark ML lib

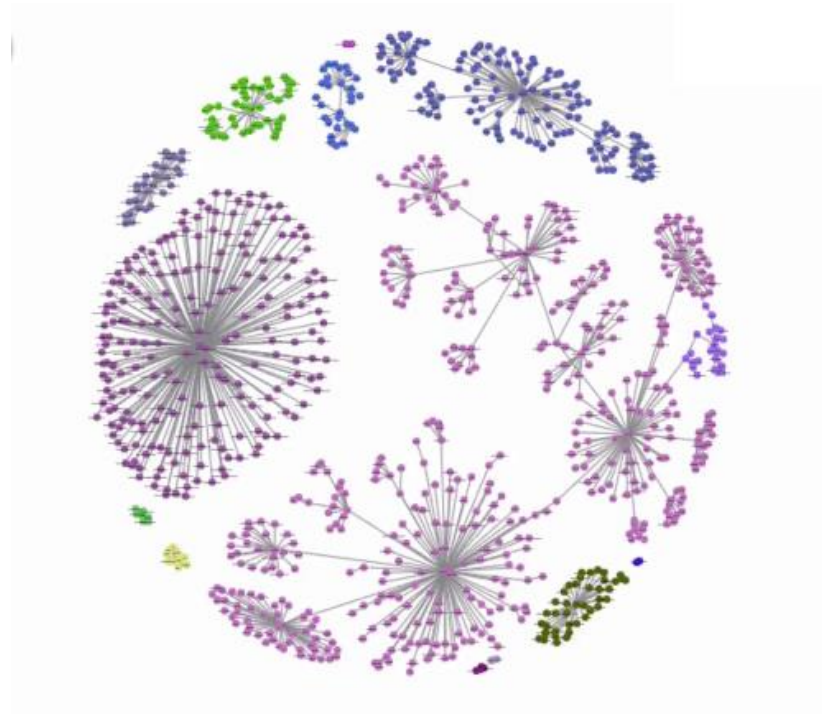
# Aprendizaje no supervisado

A veces tenemos conjuntos sin más; no hay una separación clara entre etiquetas y características.

En esos casos ML también puede ser útil, ayudándonos a reconocer estructuras, descubrir grupos, etc.

## K-Means

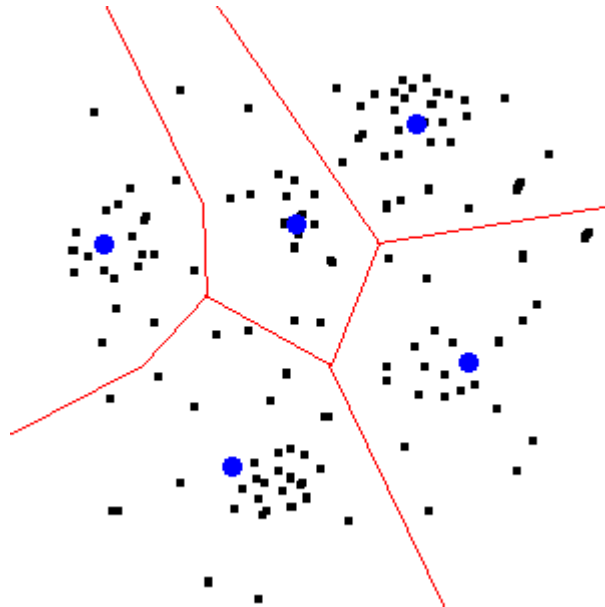
### Bisecting K-means



# Clustering: agrupación de elementos

**Objetivo de los métodos de clustering:** agrupar elementos con características similares

En el método **k-means** se utiliza un criterio geométrico para agrupar → puntos cercanos se consideran del mismo grupo



# Aplicaciones de clustering

- ✓ Segmentación de mercados: agrupar clientes con características similares
- ✓ Análisis de redes sociales: descubrir relaciones no obvias entre personas
- ✓ Astronomía: catalogar objetos celestes
- ✓ Búsqueda en internet: agrupar páginas con contenidos similares
- ✓ Visualización: evitar “aglomeración” de indicadores en mapas

# K-means: agrupación de elementos

Según por ejemplo Bishop (pattern recognition...):

*Partimos de una **variable aleatoria** en un **espacio n-dimensional** Euclídeo*


Ejemplo: facturas representados por

4 dimensiones

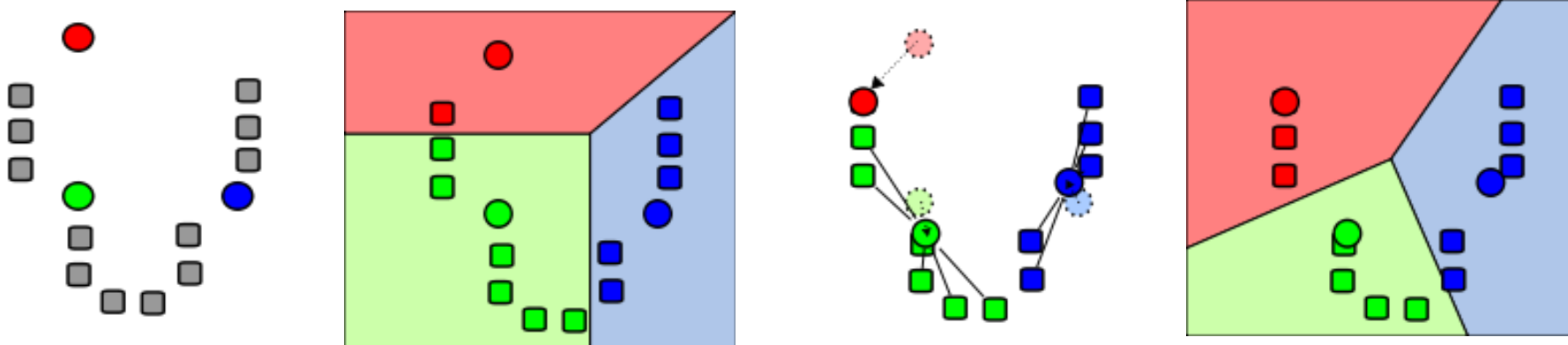
(importeTotal, númUnidades, grupodelVA, antigüedadDelCliente)

Interesa **valores genéricos** (en el ejemplo: aplicables a muchas facturas)

# K-means: algoritmo

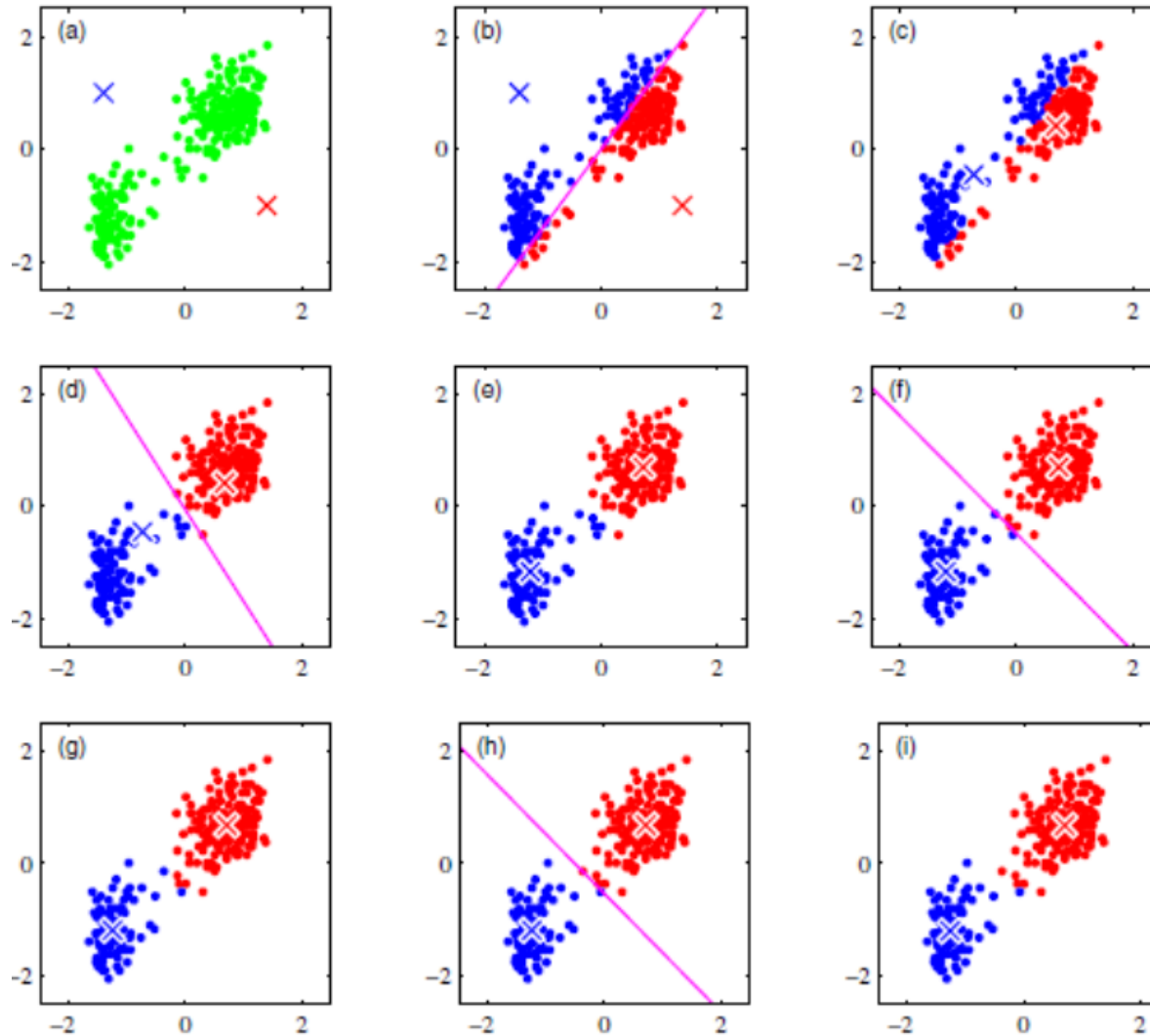
- 0) Se elige un número  $k$  de grupos inicial, y un número de iteraciones
  - 1) Se generan  $k$  puntos en el espacio  $n$  dimensional  $\rightarrow$  centroides
  - 2) Se asignan los puntos del conjunto de entrenamiento, cada uno al centroide más cercano  $\rightarrow k$  grupos
  - 3) Ahora se calcula el centro geométrico de cada centro  $\rightarrow$  nuevos centroides
  - 4) Si no se ha alcanzado el número total de iteraciones  $\rightarrow$  paso 2
- 

# K-means: ejemplo



De I, Weston.pace, CC BY-SA 3.0,  
<https://commons.wikimedia.org/w/index.php?curid=2463053>

# K-means: ejemplo



Bishop –  
Pattern recognition and Machine Learning





## K - means



Fácil de implementar

No es adecuado para  
características discretas

Resultados similares a otros métodos más  
complejos

Puede ser muy lento

Los clúster no son nunca vacíos

Para mejores resultados hay que  
ejecutarlo varias veces

Siempre acaba encontrando clúster que no  
se solapan

Difícil determinar el valor “bueno”  
para de K



## K - means



Fácil de implementar

No es adecuado para  
características discretas

Resultados similares a otros métodos más  
complejos

Puede ser muy lento

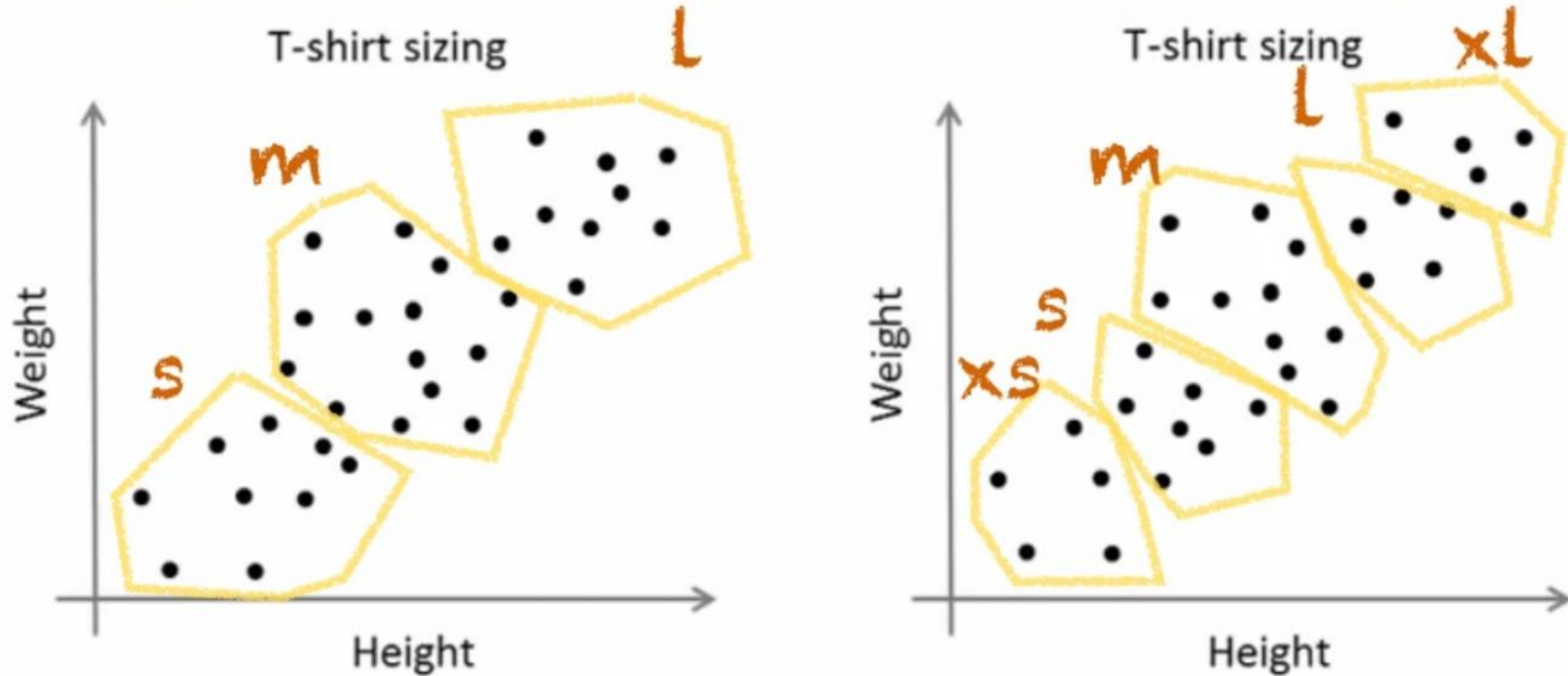
Los clúster no son nunca vacíos

Para mejores resultados hay que  
ejecutarlo varias veces

Siempre acaba encontrando clúster que no  
se solapan

Difícil determinar el valor “bueno”  
para de K

# El valor de K depende de nuestros propósitos

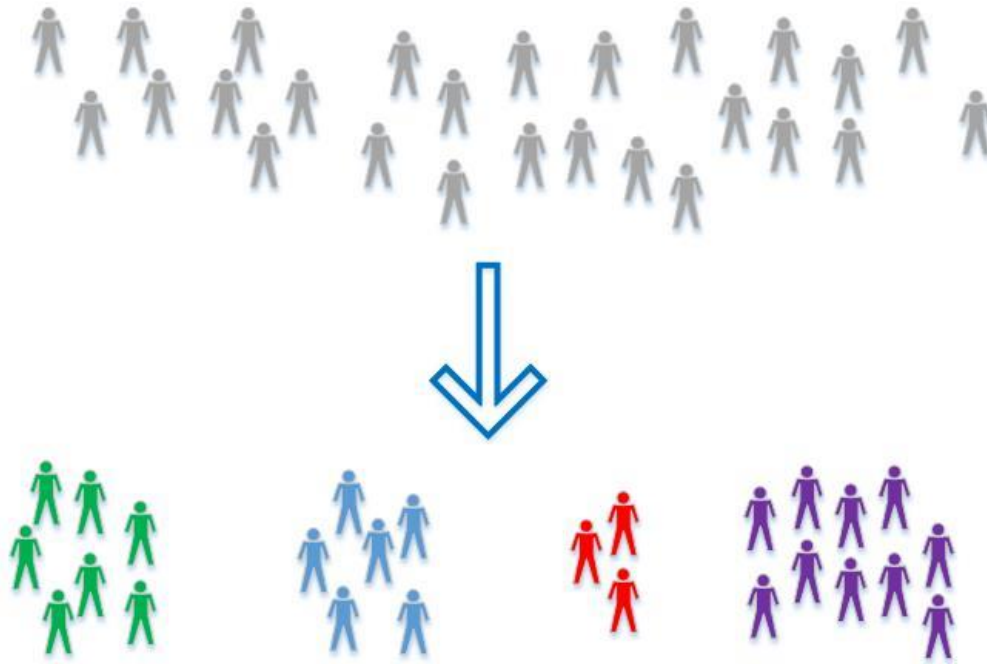


# ¿Qué k es el mejor?

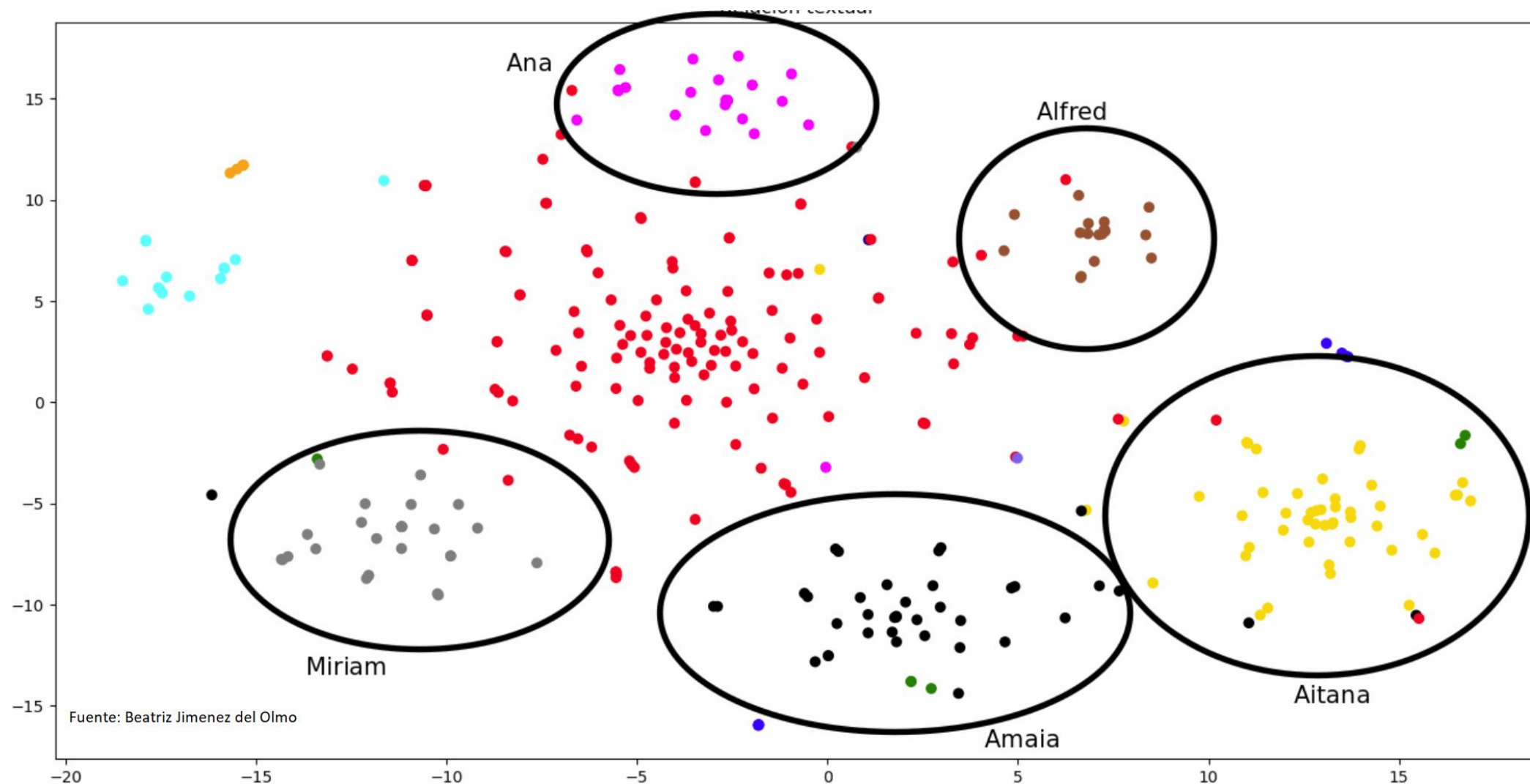
Índices:

- Silhouette
- Calinski & Harabaz
- Muchos más!!!!

- Representación  
t-SNE (100 o menos dimensiones)



# Ejemplo: aplicación k-means + t-SNE



# Parámetros de k-means en Spark

*setk* fija el número de clusters

*maxIterations* el número máximo de iteraciones antes de encontrar un resultado

*initializationMode* indica el metodo de inicialización “random” o “[kmeans||](#)”

*runs* número de veces que se ejecutará el algoritmo (luego hará la media de centros)

*initializationSteps* número de pasos iniciales en k-means ||

*epsilon* distancia de convergencia. Para cuando todos los centroides se mueven menos

*initialModel* conjunto inicial de centros de cluster. Si se pasa solo se itera una vez

# Para lograr un óptimo resultado con k-means



*estandarizar* asegurarse de que todos los valores están en la misma unidad si son unidades comparables (no metros y cm)

*escalar* Multiplicar por una constante las columnas para conseguir intervalos similares [0,1]

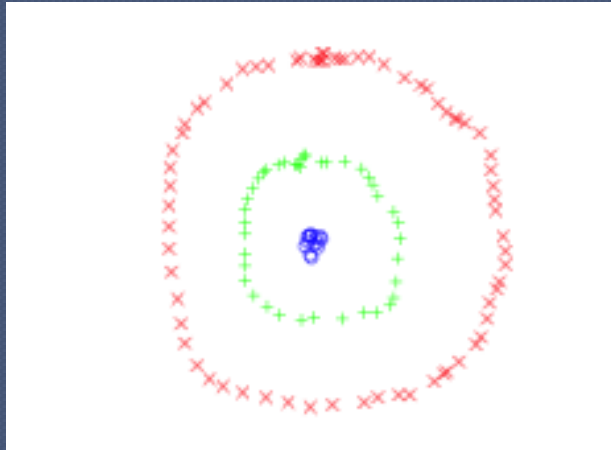
*Ejecutar varias veces* y a ser posible reordenar las filas entre cada ejecución (esto no es fácil pero el orden de las filas influye en el algoritmo)

*Última ejecución* Tomar los centroides obtenidos como media después de todas las ejecuciones y usarlo como valor inicial en una última ejecución (si se obtienen unos centros muy diferentes → tenemos un problema)

# Test

---

¿Resulta adecuado K-means para distinguir estos 3 clúster?



- A. Sí, clarísima disposición en círculo, justo para K-Means
- B. No, no corresponde a la agrupación por distancias
- C. No sabe/No contesta/Hablando por el móvil



+ Spark ML lib

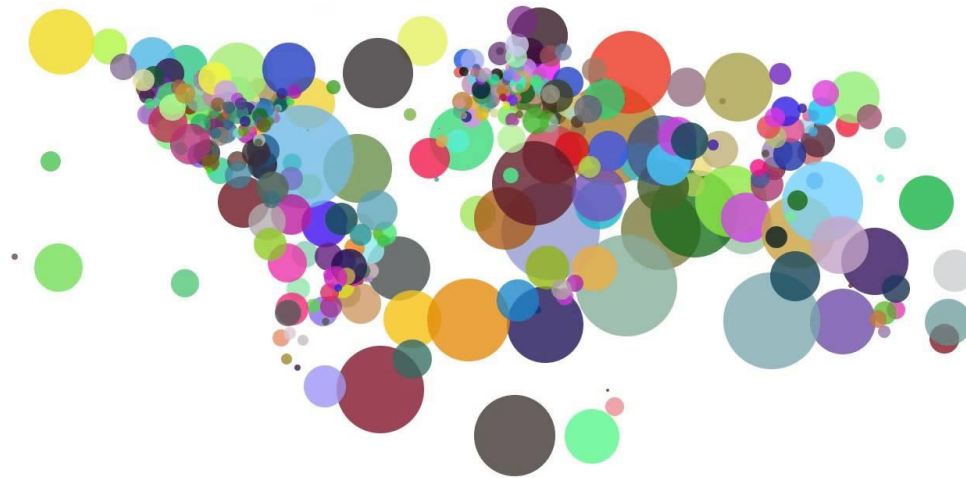
## Aprendizaje no supervisado

A veces tenemos conjuntos sin más; no hay una separación clara entre etiquetas y características.

En esos casos ML también puede ser útil, ayudándonos a reconocer estructuras, descubrir grupos, etc.

K-Means

Bisecting K-means



# Clustering jerárquico

Alternativa a K-means



Dos tipos

- Por **aglomeración**: bottom-up; cada elemento empieza en un clúster y se van uniendo
- Por **división**: top-down; se empieza con un solo clúster y se va dividiendo

Ventajas Jerárquico Versus K-means	Desventajas Jerárquico Versus K-Means
Posibilidades de elegir granularidad	Múcho más lento -> no válido para Big Data
Resultados consistentes, no factor aleat.	Peor para configuraciones esféricas
No necesita conocer K de antemano	
Válido para configuraciones no esféricas	

# Bisecting K-means

Algoritmo **jerárquico por división**, utiliza K-means para las divisiones

0) Se parte de un cluster único con todo el conjunto de valores iniciales

1) Se selecciona un clúster

2) Se divide en 2, usando k-means

3) Se repite hasta alcanzar el número de clústers o el tamaño de clúster que se desee

