

UNIVERSIDADE DA CORUÑA

APRENDIZAJE AUTOMÁTICO

APRENDIZAJE NO SUPERVISADO

APRENDIZAJE NO SUPERVISADO

- Rama del aprendizaje automático donde los algoritmos son entrenados en un conjunto de datos sin etiquetas o salidas deseadas



- **Objetivo principal**
 - Descubrir patrones y relaciones ocultas



APRENDIZAJE NO SUPERVISADO

- Ventajas importantes:
 - Capacidad de trabajar con una gran cantidad de datos sin etiquetar
 - La inmensa mayoría de los datos disponibles están sin etiquetar
 - Ejemplo: imágenes en internet
 - Descubrir patrones ocultos



APRENDIZAJE NO SUPERVISADO

- Desventajas:
 - Dificultad en la evaluación del resultado
 - Al no haber etiquetas o salidas objetivo previamente conocidas, es difícil determinar si los patrones y relaciones descubiertos por el algoritmo son útiles o no
 - Dependencia de la elección del algoritmo y los parámetros



APRENDIZAJE NO SUPERVISADO

- Principales problemas a resolver:
 - Clustering
 - Agrupar datos similares en "clusters" o grupos
 - Compresión de datos y reducción de dimensionalidad
 - Representar los datos en un espacio de características más pequeño sin perder demasiada información
 - Asociación de reglas
 - Descubrir patrones frecuentes en los datos
 - Detección de anomalías
 - Identificar datos atípicos o "anómalos"



APRENDIZAJE NO SUPERVISADO

- Clustering
 - k-means
 - Clústering jerárquico
 - DBSCAN
 - GMM
- Reducción de dimensionalidad
- Asociación de reglas
 - Apriori
- Detección de anomalías
 - k-NN
 - One-class classification
 - One-class SVM
 - Isolation Forest
- Redes de Neuronas Artificiales no supervisadas
 - Mapas autoorganizativos
 - Autoencoders



APRENDIZAJE NO SUPERVISADO

- **Clustering**
 - k-means
 - Clústering jerárquico
 - DBSCAN
 - GMM
- Reducción de dimensionalidad
- Asociación de reglas
 - Apriori
- Detección de anomalías
 - k-NN
 - One-class classification
 - One-class SVM
 - Isolation Forest
- Redes de Neuronas Artificiales no supervisadas
 - Mapas autoorganizativos
 - Autoencoders

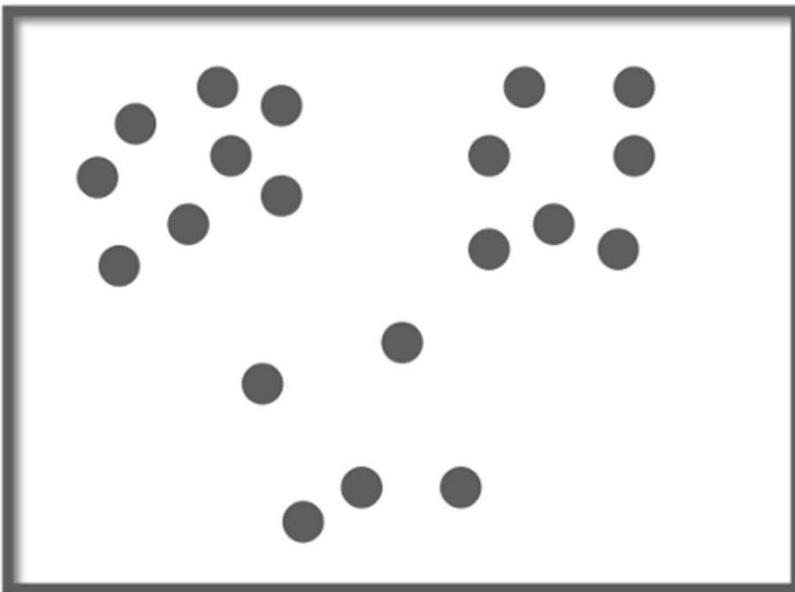


CLUSTERING

- También llamada agrupamiento o “clasificación no supervisada”
- Objetivo: analizar similitud entre los datos y agrupa los similares en un mismo cluster
 - Los objetos dentro de un cluster son considerados similares entre sí y diferentes a los objetos de otros clusters
- Se utiliza para identificar patrones y estructuras en los datos sin necesidad de etiquetas previas

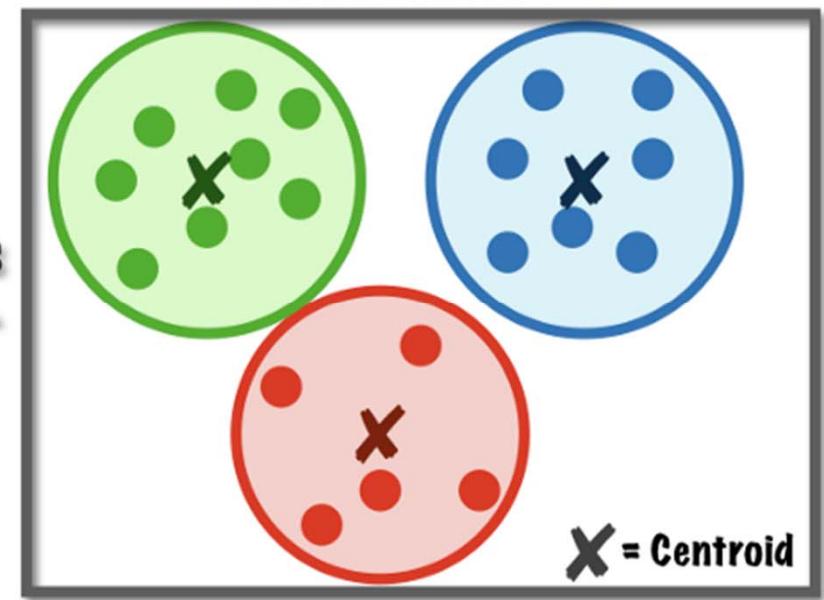
CLUSTERING

Unlabelled Data



K-means
→

Labelled Clusters

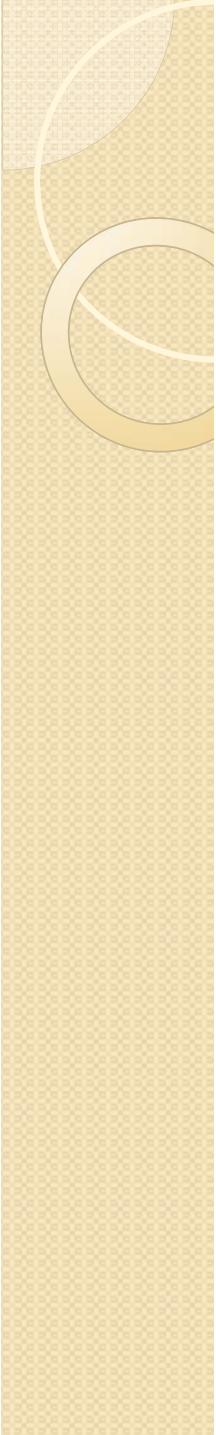


X = Centroid



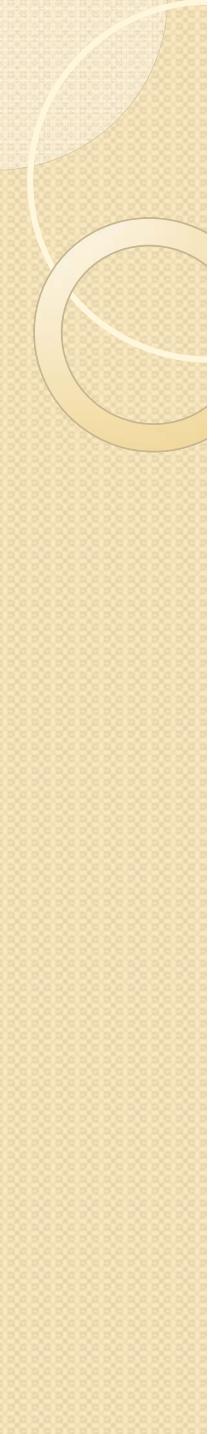
CLUSTERING

- El clustering es ampliamente utilizado en una variedad de aplicaciones, como por ejemplo:
 - Segmentación de clientes en marketing
 - Agrupación de documentos en recuperación de información
 - Detección de errores en equipos industriales
- En general, es una herramienta valiosa para explorar y comprender la estructura subyacente de los datos
 - Pero es importante tener en cuenta que los resultados pueden ser influenciados por la selección de los parámetros y el algoritmo utilizado



CLUSTERING

- Principales algoritmos:
 - k-means: divide los datos en k clusters basados en la distancia euclídea.
 - *Hierarchical Clustering*: divide los datos en grupos anidados a través de la agregación o división de los grupos existentes
 - DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*): agrupa los puntos basados en la densidad de los puntos cercanos
 - *Gaussian Mixture Model* (GMM): asigna una probabilidad a cada punto para pertenecer a uno de los clusters



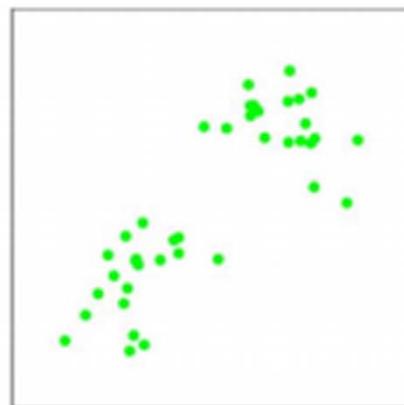
CLUSTERING

- **k-means**

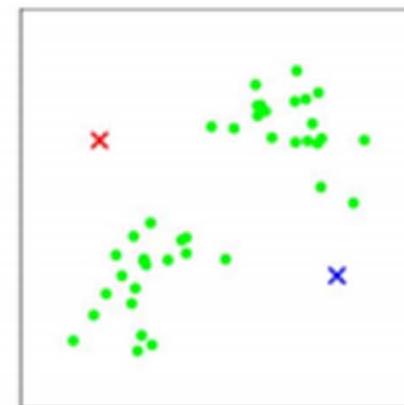
- Algoritmo de clustering iterativo que se utiliza para agrupar datos similares en k clusters
- Objetivo: encontrar los k centroides que minimicen la suma de las distancias euclidianas entre cada punto de datos y su centroide correspondiente

CLUSTERING

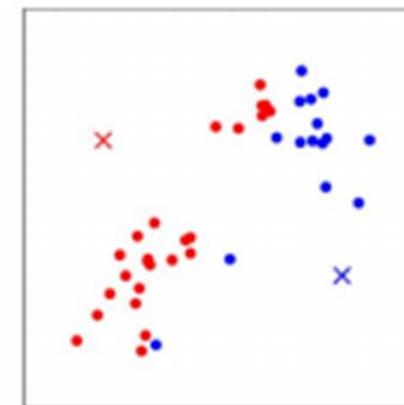
- k-means



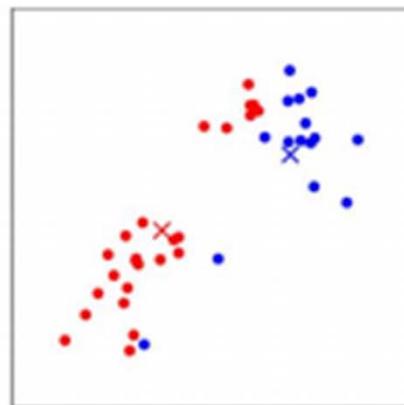
(a)



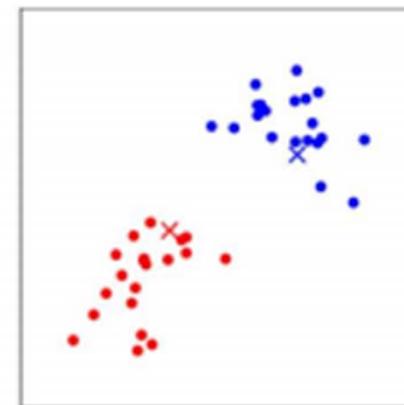
(b)



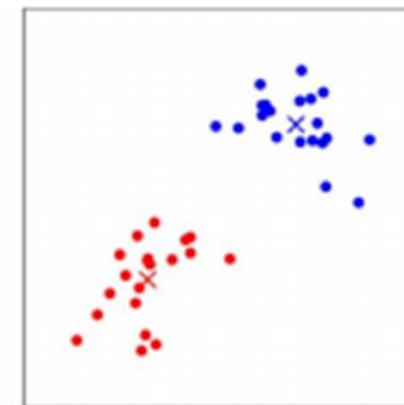
(c)



(d)



(e)



(f)



CLUSTERING

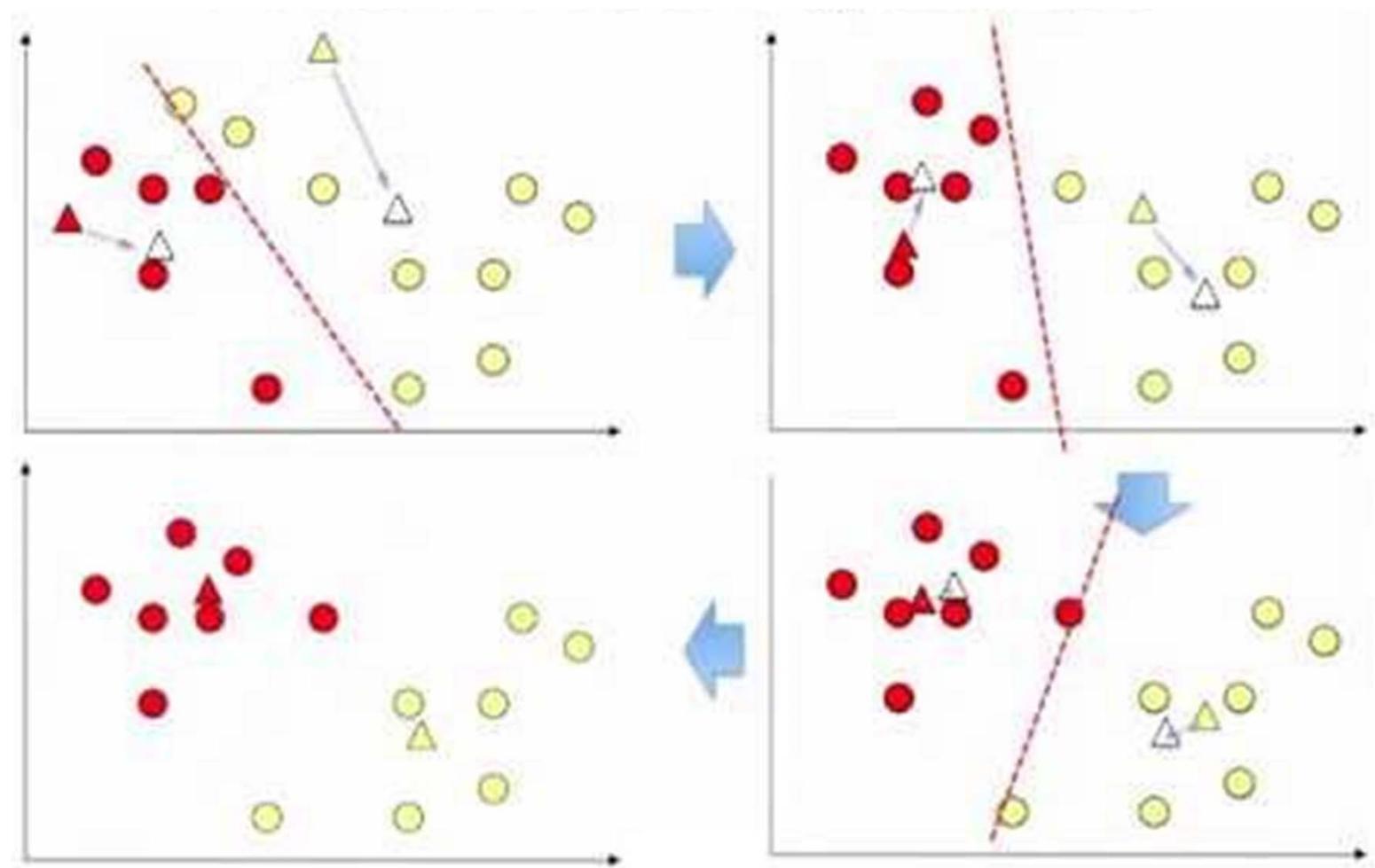
- **k-means**

- **Algoritmo:**

1. Inicializar los k centroides aleatoriamente en el espacio de características
2. Asignar cada punto de datos al centroide más cercano
3. Calcular la media de cada clúster y actualizar el centroide correspondiente
 - Media de cada clúster: valor promedio de todos los puntos de datos en el clúster
 - Una vez calculada, se actualiza la posición del centroide para la siguiente iteración del algoritmo k-means.
4. Repetir 2 y 3 hasta que los centroides no cambien o hasta que se alcance un número máximo de iteraciones

CLUSTERING

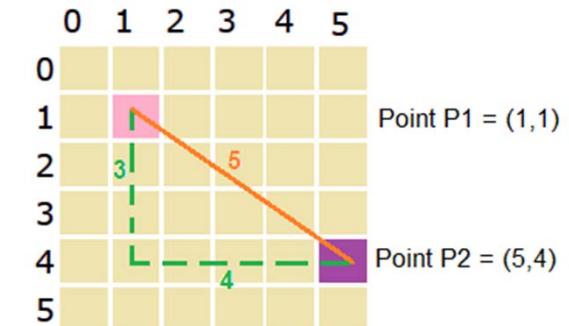
- k-means



CLUSTERING

- k-means

- Métricas de distancia
 - Distancia entre los puntos de datos y los centroides
 - Más habitual: distancia euclidiana
 - Datos numéricos
 - Otras posibles si son adecuadas para el conjunto de datos y el objetivo del análisis:
 - Distancia de Manhattan
 - Distancia de Mahalanobis
 - Distancia de coseno



$$\text{Euclidean distance} = \sqrt{(5-1)^2 + (4-1)^2} = 5$$

$$\text{Manhattan distance} = |5-1| + |4-1| = 7$$



CLUSTERING

- **k-means**
 - Métricas de distancia
 - Distancia de Manhattan:
 - Suma de las diferencias absolutas entre las coordenadas de los dos puntos
 - Medida de distancia no euclídea
 - No se basa en la distancia euclídea
 - Por lo tanto, no satisface la propiedad de triangularidad
 - La distancia de Manhattan se utiliza en situaciones en las que la distancia de viaje entre dos puntos es más importante que la distancia en línea recta entre ellos
 - Por ejemplo, en un entorno con calles y avenidas con direcciones únicas
 - Se utiliza con datos categóricos o binarios



CLUSTERING

- **k-means**

- **Métricas de distancia**

- **Distancia de Mahalanobis**
 - La distancia de Mahalanobis tiene en cuenta la estructura de la varianza-covarianza de los datos
 - Es decir, tiene en cuenta la relación lineal entre las variables y sus escalas
 - La distancia de Mahalanobis se calcula como:

$$d(x, y) = (x - y)^T C^{-1} (x - y)$$

- donde C^{-1} es la inversa de la matriz de covarianza
 - La distancia de Mahalanobis es una medida de distancia normalizada que es insensible a la escala de las variables y tiene en cuenta la relación entre ellas

CLUSTERING

- **k-means**

- **Métricas de distancia**

- **Distancia de coseno**
 - Mide la similitud angular entre dos vectores de características en un espacio vectorial y se calcula como:

$$\frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2} \sqrt{\sum_1^n b_i^2}}$$

- donde $\|\mathbf{a}\|$ y $\|\mathbf{b}\|$ son las normas de los vectores \mathbf{a} y \mathbf{b} y $(\mathbf{a} \cdot \mathbf{b})$ es el producto escalar entre ellos
 - La distancia de coseno varía de -1 a 1, con valores más cercanos a 1 indicando una mayor similitud entre los vectores y valores más cercanos a -1 indicando una mayor disimilitud
 - La distancia de coseno se utiliza comúnmente en problemas de recomendación de productos y similitud de documentos

CLUSTERING

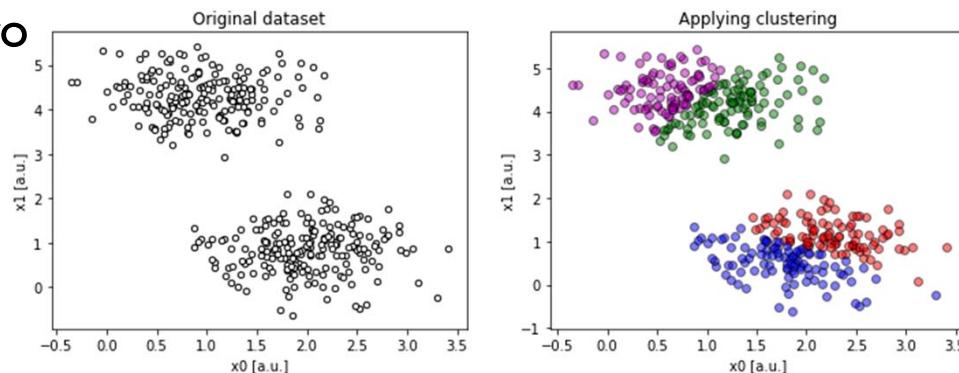
- k-means

- ¿Cuántos clústeres son necesarios?

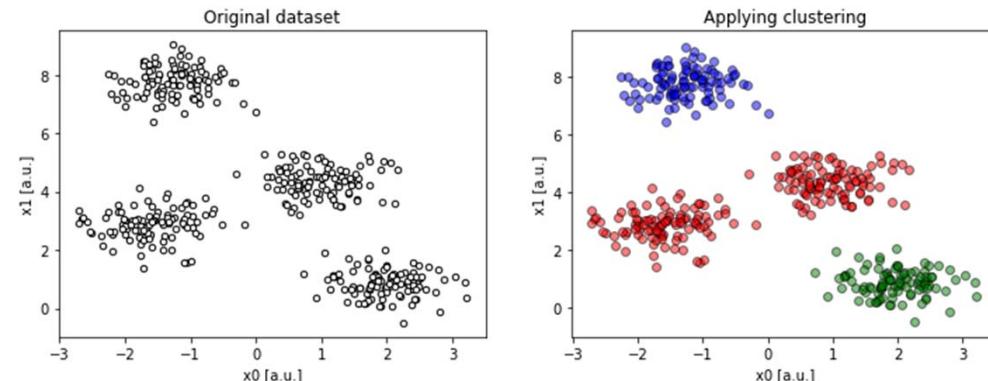
- Valor escogido por el usuario

- Hiperparámetro

- Demasiados



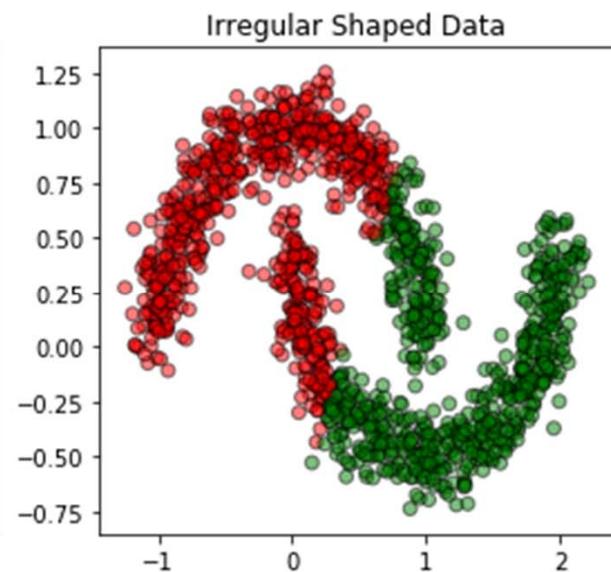
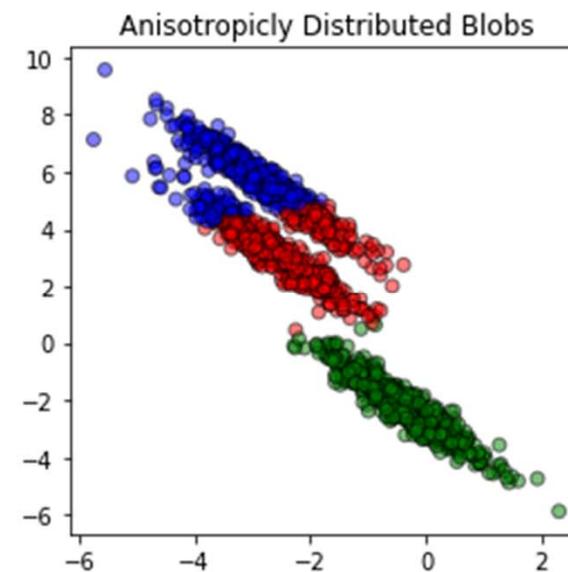
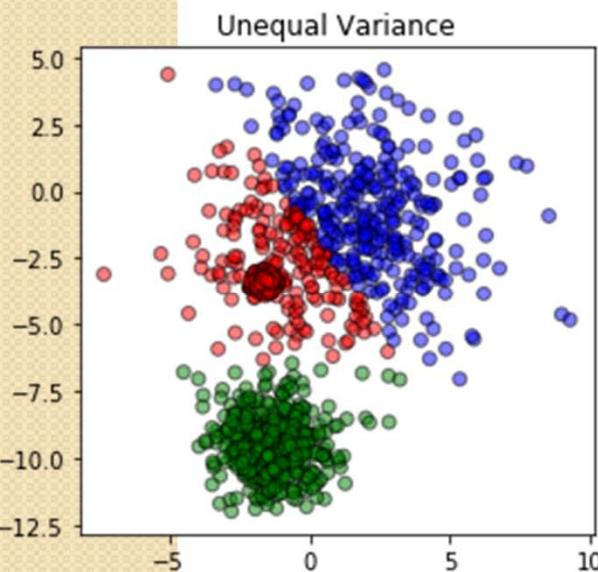
- Demasiado pocos



CLUSTERING

- **k-means**

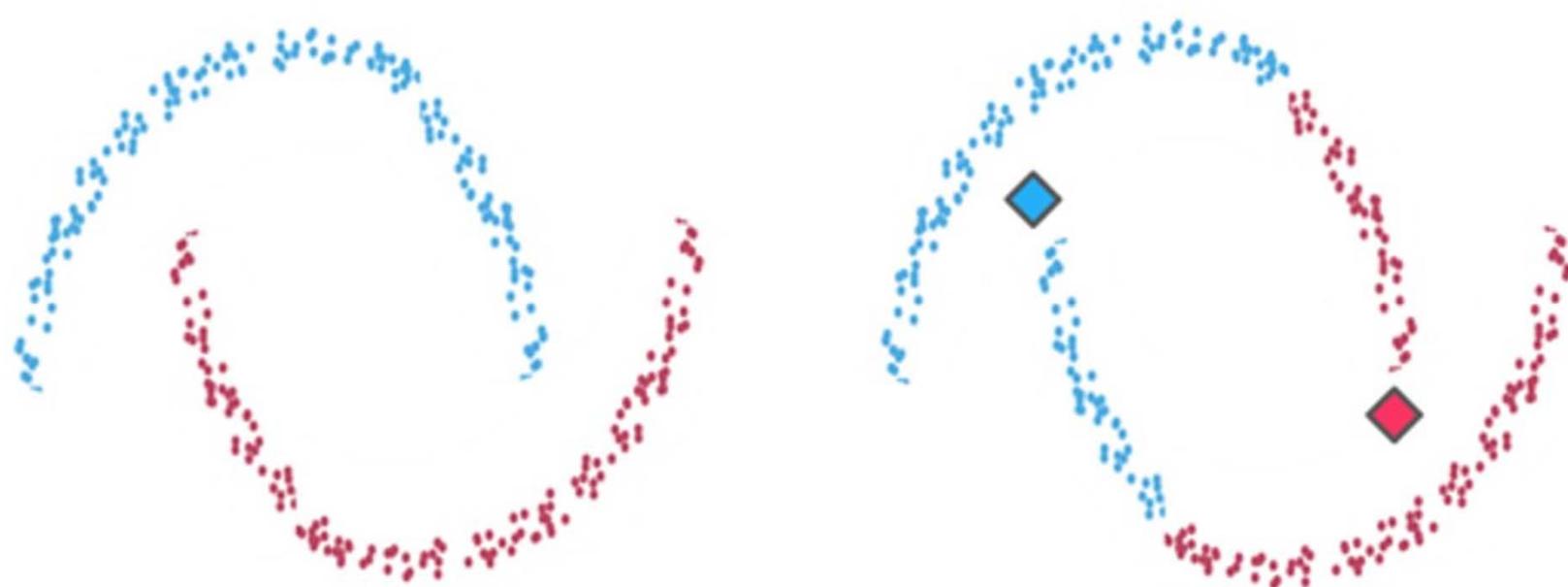
- Desventaja: Sólo funciona bien con clusters *globulares*, bien separados y con similar varianza
 - Ejemplos:



CLUSTERING

- k-means

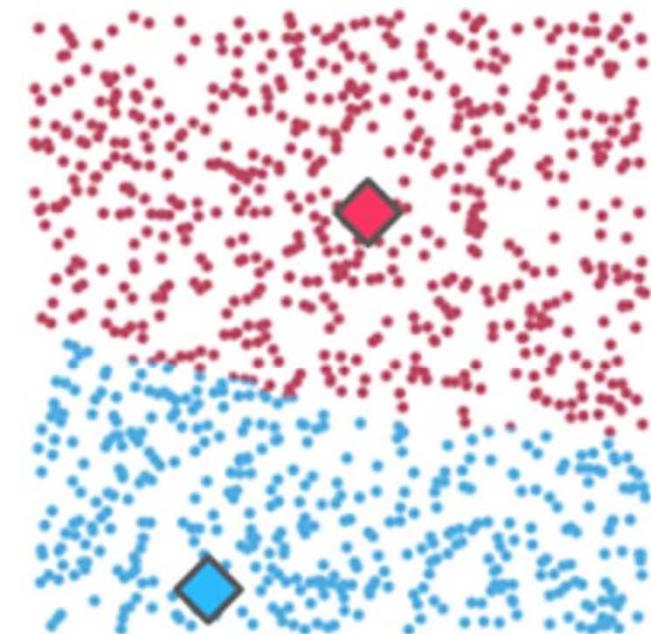
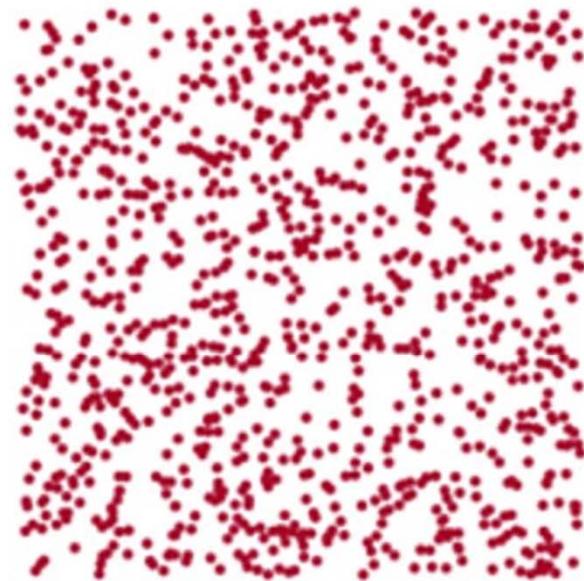
- Desventaja: Sólo funciona bien con clusters *globulares*, bien separados y con similar varianza
 - Ejemplos:



CLUSTERING

- **k-means**

- Desventaja: Sólo funciona bien con clusters *globulares*, bien separados y con similar varianza
 - Ejemplos:





CLUSTERING

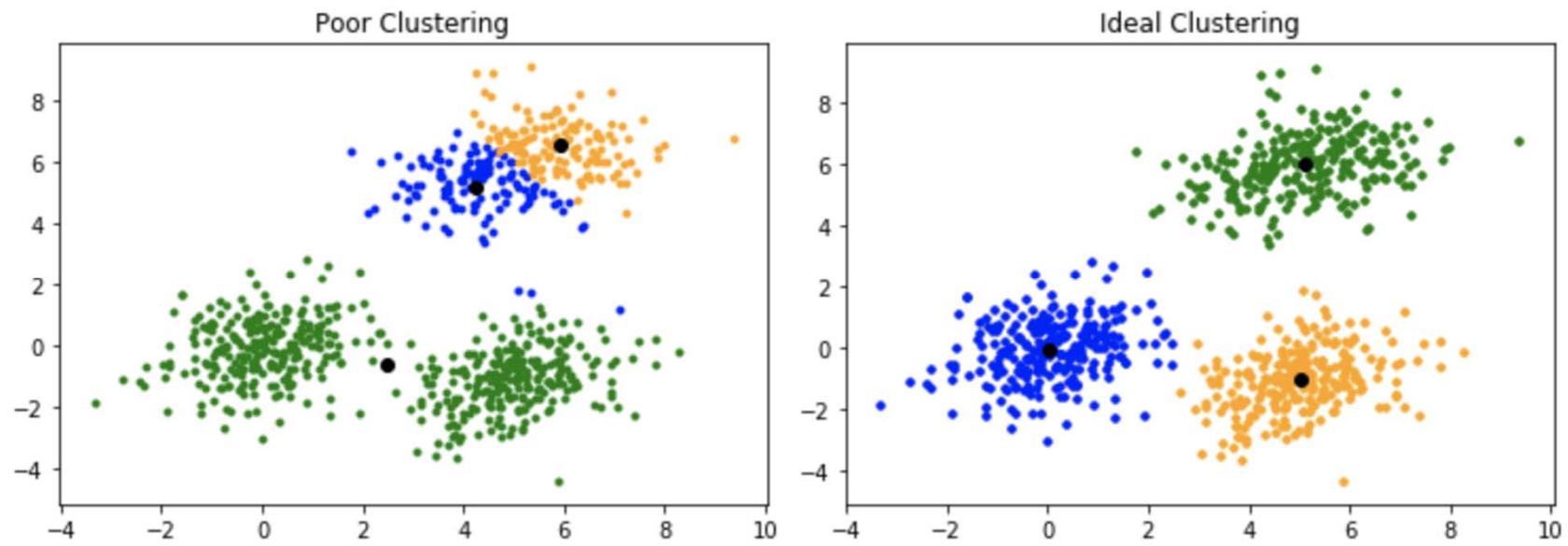
- **k-means**

- **Desventaja: Sensible a la inicialización**
 - Algunas posibles situaciones:
 - Si un centroide se inicializa en un punto alejado del conjunto de datos, podría terminar sin datos asociados al mismo
 - Más de un centroide podría ser inicializado en el mismo grupo de datos, dando como resultado varios clusters incorrectos
 - Los centroides se podrían inicializar de tal forma que sólo uno sea el más cercano a varios grupos de datos, dando como resultado un solo clúster

CLUSTERING

- **k-means**

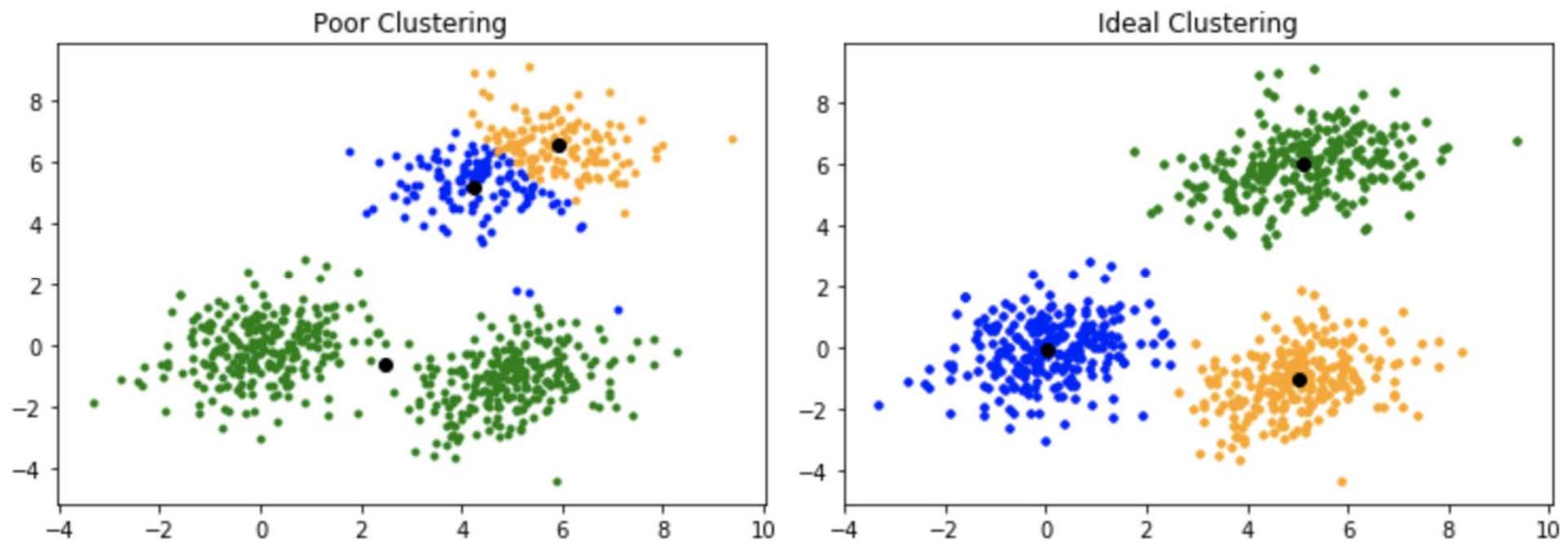
- Desventaja: Sensible a la inicialización
 - Algunas posibles situaciones:
 - Más de un centroide podría ser inicializado en el mismo grupo de datos, dando como resultado varios clusters incorrectos



CLUSTERING

- **k-means**

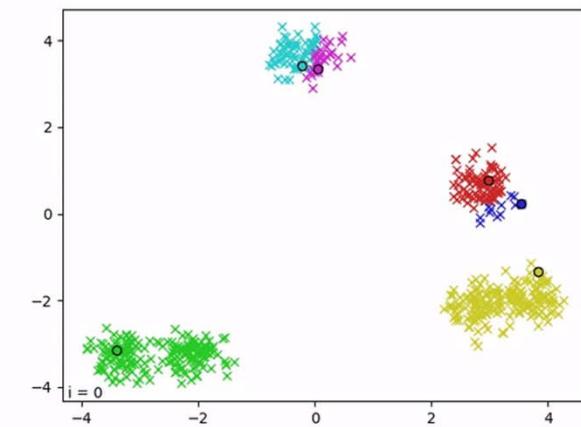
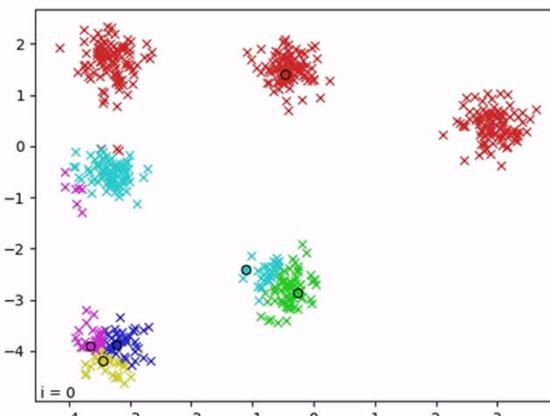
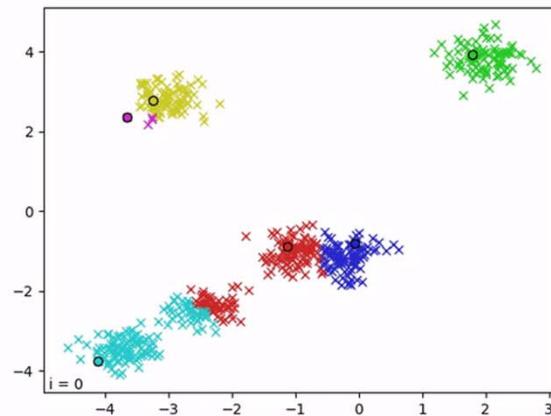
- Desventaja: Sensible a la inicialización
 - Algunas posibles situaciones:
 - Los centroides se podrían inicializar de tal forma que sólo uno sea el más cercano a varios grupos de datos, dando como resultado un solo clúster



CLUSTERING

- **k-means**

- Desventaja: Sensible a la inicialización
 - Más ejemplos:





CLUSTERING

- **k-means**
 - Desventaja: Sensible a la inicialización
 - Para solucionarlo, es común ejecutar el algoritmo varias veces con diferentes inicializaciones y seleccionar el resultado con la menor suma de las distancias cuadráticas



CLUSTERING

- **k-means**
 - **k-means++**
 - Variante de k-means
 - Mejora en la inicialización de los centroides
 - Inicializarlos de tal manera que sean más probable que estén lejos el uno del otro
 - Esto reduce la probabilidad de que los centroides iniciales sean muy similares y resulten en una solución subóptima



CLUSTERING

- **k-means**

- **k-means++**

- **Algoritmo:**

- 1. **Inicialización:**

- k-means tradicional: los k centroides se inicializan aleatoriamente en el espacio de características
 - k-means++: se elige un centroide aleatoriamente y se eligen los siguientes centroides con probabilidad proporcional a la distancia euclídea entre cada punto de datos y los centroides existentes

- 2. **Asignación:** ambos algoritmos asignan cada punto de datos al centroide más cercano y calculan la media de cada clúster



CLUSTERING

- **k-means**
 - **k-means++**
 - k-means++ es menos sensible a la inicialización y suele producir soluciones más precisas y estables que el k-means tradicional
 - Sin embargo, también es un poco más costoso computacionalmente debido a la selección de centroides con probabilidad.



CLUSTERING

- **k-means**
 - **k-medians**
 - Variante de k-means en el que, en lugar de calcular la media como centroide, se calcula la mediana
 - Cuando se actualiza el centroide, este se desplaza a la posición cuyos elementos son iguales al valor de la mediana de cada dimensión de todas las instancias asignadas al cluster
 - Esto tiene el efecto de minimizar con respecto a la métrica de distancia norma L1
 - Distancia de Manhattan
 - Por la contra, la métrica habitual en k-means es la norma L2 al cuadrado
 - Distancia Euclídea



CLUSTERING

- **k-means**

- **k-medians**

- k-means minimiza la varianza dentro del clúster, que es igual a las distancias euclídeas al centroide, al cuadrado
 - Para ello, se usa la media aritmética para calcular los centroides
 - Optimiza las desviaciones al cuadrado de la media
 - k-medians minimiza las desviaciones absolutas
 - Esto equivale a utilizar la distancia Manhattan
 - Si se desea minimizar esta distancia en lugar de las desviaciones al cuadrado, la mediana un buen estimador para la media



CLUSTERING

- **k-means**

- **k-medians**

- Para cuantificar el proceso de clustering, se pueden utilizar las siguientes métricas:

- Suma, para cada clúster, de la suma de las distancias de cada elemento del clúster al centroide
 - k-means:

$$J = \sum_{j=1}^K \sum_{i=1}^n \sqrt{(q_i - \mu_j)^2}$$

- (K: númer. clústeres, μ_j : vector promedio del cluster j)
 - k-medians:

$$J = \sum_{j=1}^K \sum_{i=1}^n |p_i - \mu_j|$$

- (K: númer. clústeres, μ_j : vector mediana del cluster j)



CLUSTERING

- **k-means**
 - **k-medians**
 - En líneas generales:
 - Si se quiere minimizar una distancia L₂: k-means
 - Por ejemplo, distancia Euclídea
 - Si se quiere minimizar una distancia L₁: k-medians
 - Por ejemplo, distancia Manhattan
 - Tanto en k-means como en k-medians los centroides pueden no coincidir con ningún punto de la muestra



CLUSTERING

- **k-means**
 - **k-medoids**
 - También llamado PAM (*Partitioning Around Medoid*)
 - Kaufman & Rousseeuw, 1987
 - En este algoritmo, en lugar de utilizar centroides como valores medios o medianas, se utilizan *medoides*, que son puntos de datos reales
 - Más cercano al concepto de mediana que al de media
 - Por lo tanto, más robusto frente a ruido y datos atípicos



CLUSTERING

- **k-means**

- **k-medoids**

- Los medoides son puntos cuya disimilitud con el resto de puntos del clúster es mínima
 - La disimilitud entre el medoide C_i y un punto P_i se calcula como la distancia:

$$|P_i - C_i|$$

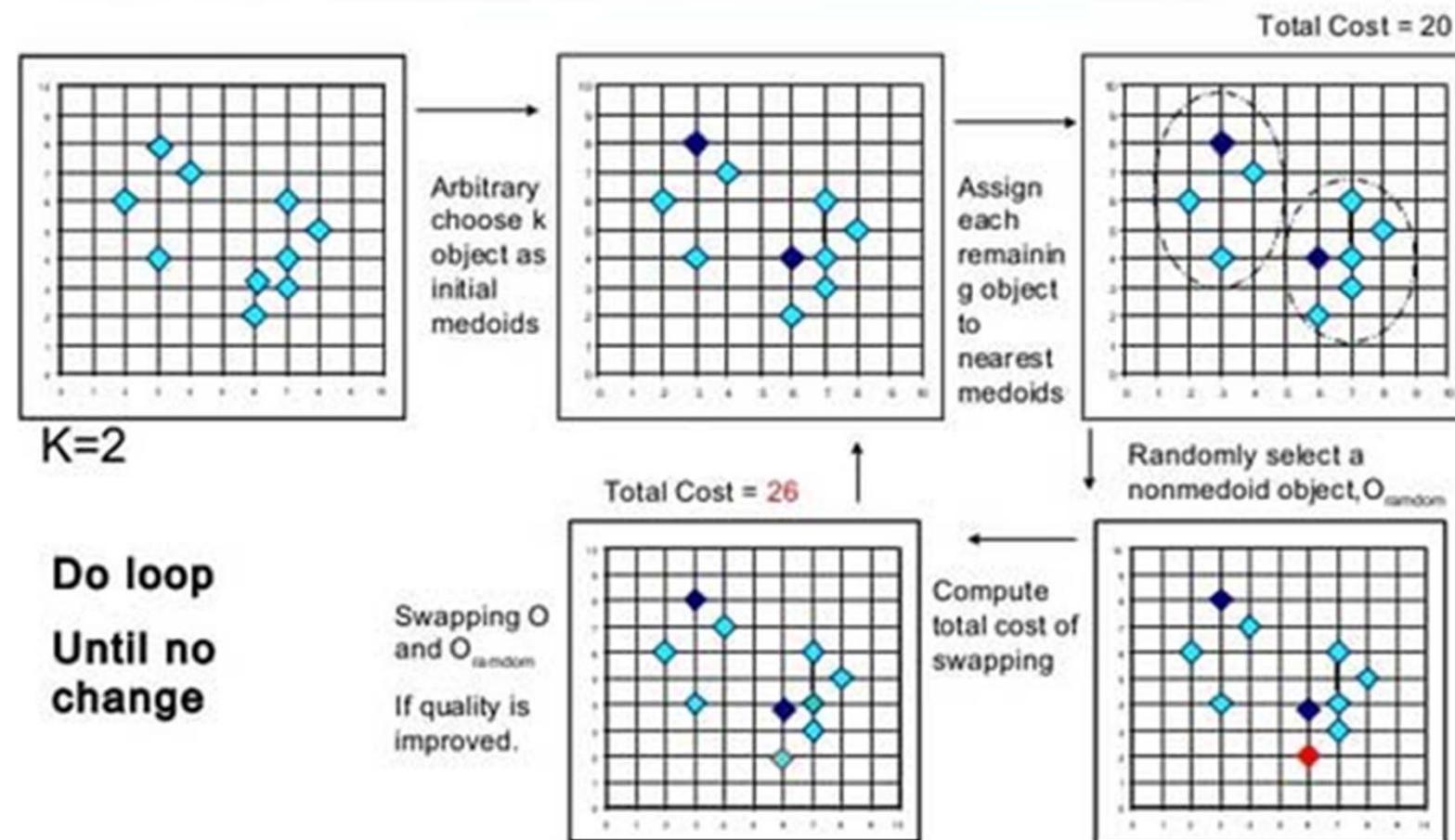
- Se puede usar cualquier medida de distancia
 - Por lo tanto, la medida del coste en este algoritmo es:

$$c = \sum_{C_i} \sum_{P_i \in C_i} |P_i - C_i|$$

- La suma, para cada clúster, de la suma, para cada punto de ese clúster, de la disimilitud entre ese punto y el medoide del clúster

CLUSTERING

- k-means
 - k-medoids





CLUSTERING

- **k-means**

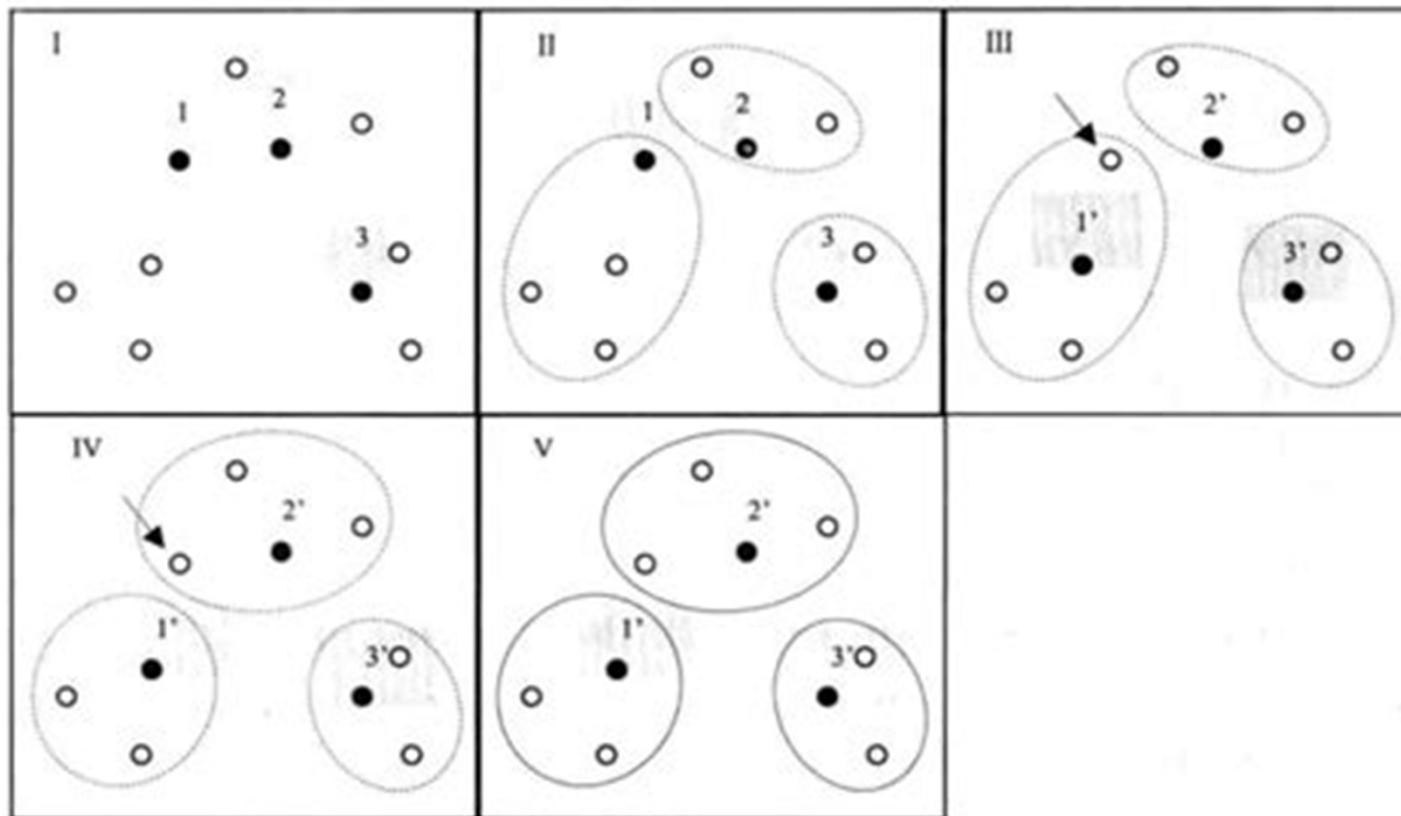
- **k-medoids**

- Proceso:

1. (BUILD) Se eligen k puntos de datos como medoides iniciales
 - De forma aleatoria o utilizando alguna heurística que minimice el coste
2. Asignar los demás puntos de datos a los medoides más cercanos
3. (SWAP) Mientras el coste disminuya:
 - Para cada medoide m, y cada punto de datos no medoide o
 - Se considera el intercambio de m y o, y se calcula el nuevo coste
 - Si hay algún cambio que decremente el coste, llevar a cabo el que produzca un mayor decremento de coste
 - Si no, el algoritmo finaliza

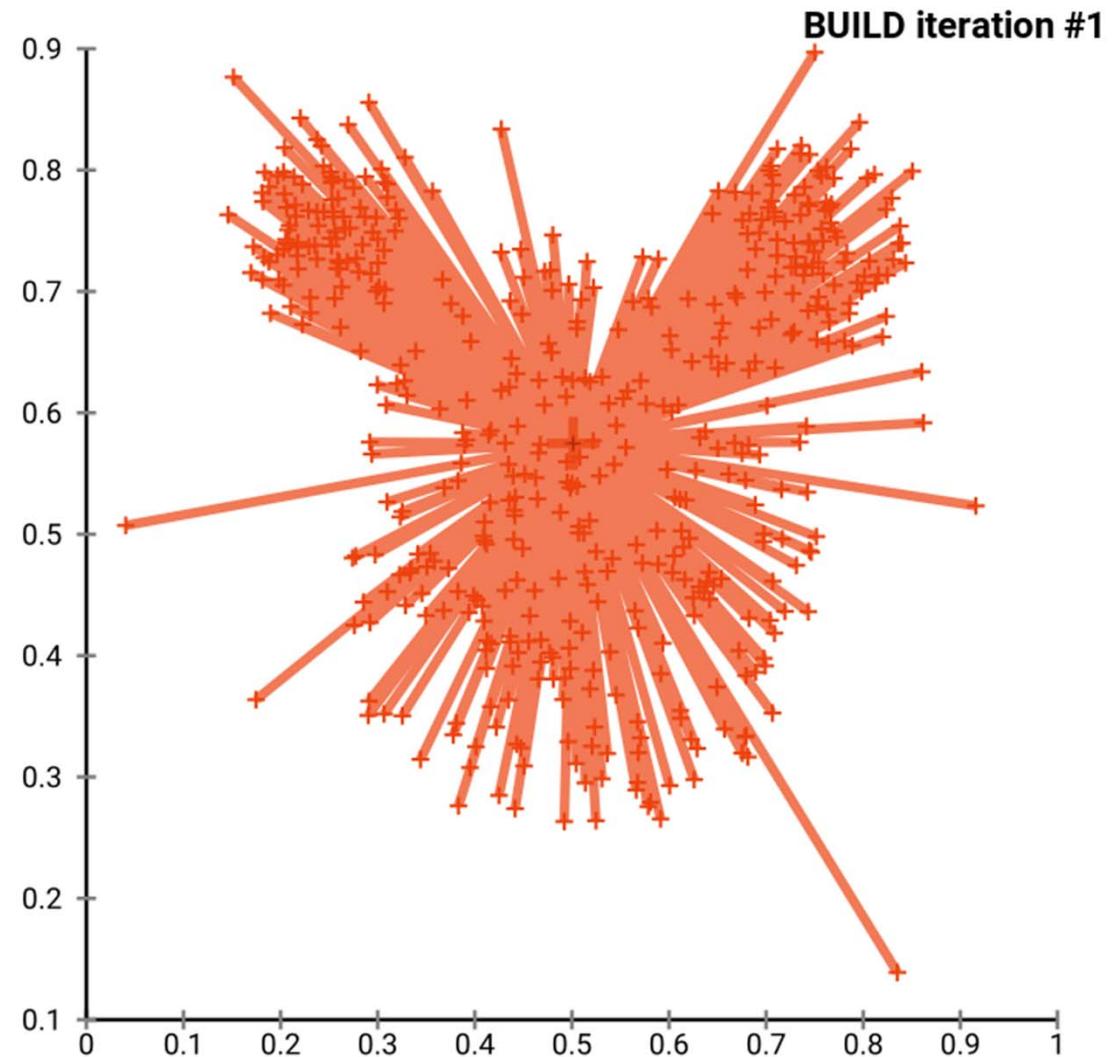
CLUSTERING

- k-means
 - k-medoids
 - Proceso:



CLUSTERING

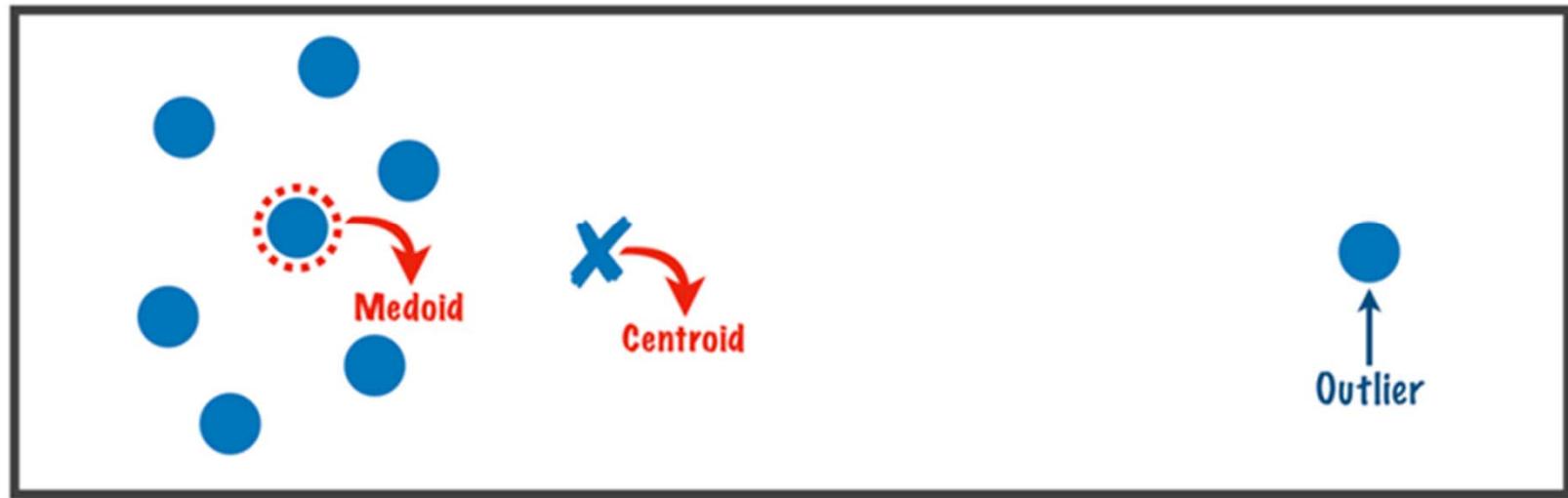
- k-means
 - k-medoids
 - Proceso:



CLUSTERING

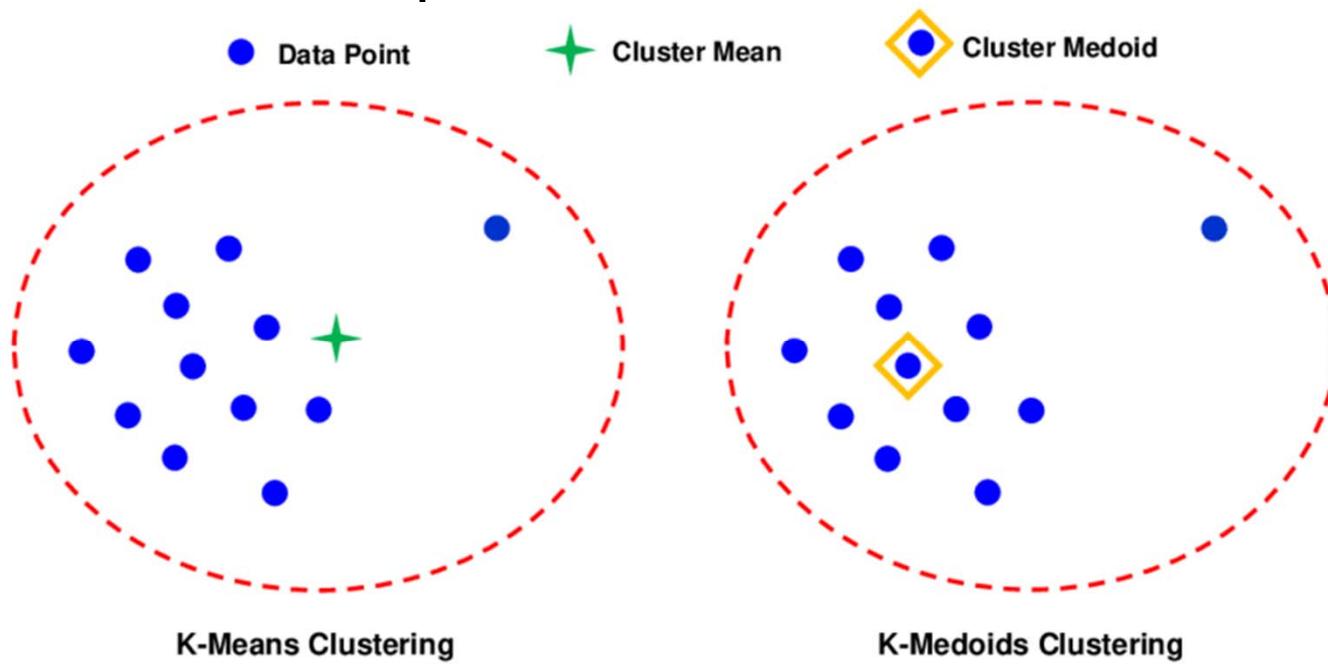
- k-means
 - k-medoids
 - k-medoids es más robusto que el k-means en términos de *outliers* y es más adecuado para conjuntos de datos con valores atípicos o distribuciones no Gaussianas

The Outlier Effect



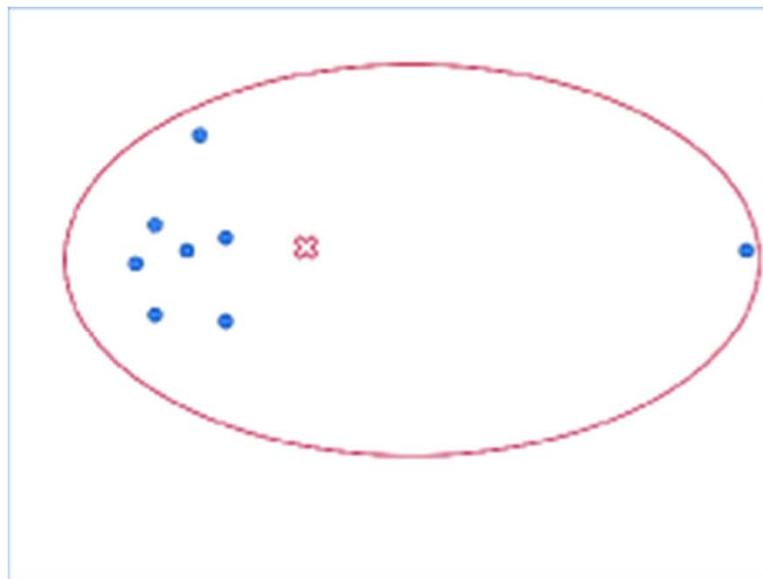
CLUSTERING

- k-means
 - k-medoids
 - k-medoids es más robusto que el k-means en términos de *outliers* y es más adecuado para conjuntos de datos con valores atípicos o distribuciones no Gaussianas

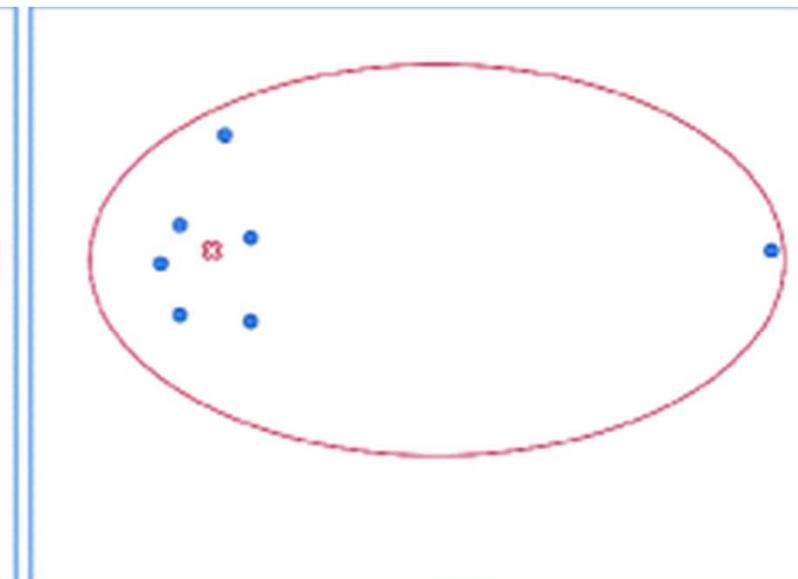


CLUSTERING

- k-means
 - k-medoids
 - k-medoids es más robusto que el k-means en términos de *outliers* y es más adecuado para conjuntos de datos con valores atípicos o distribuciones no Gaussianas



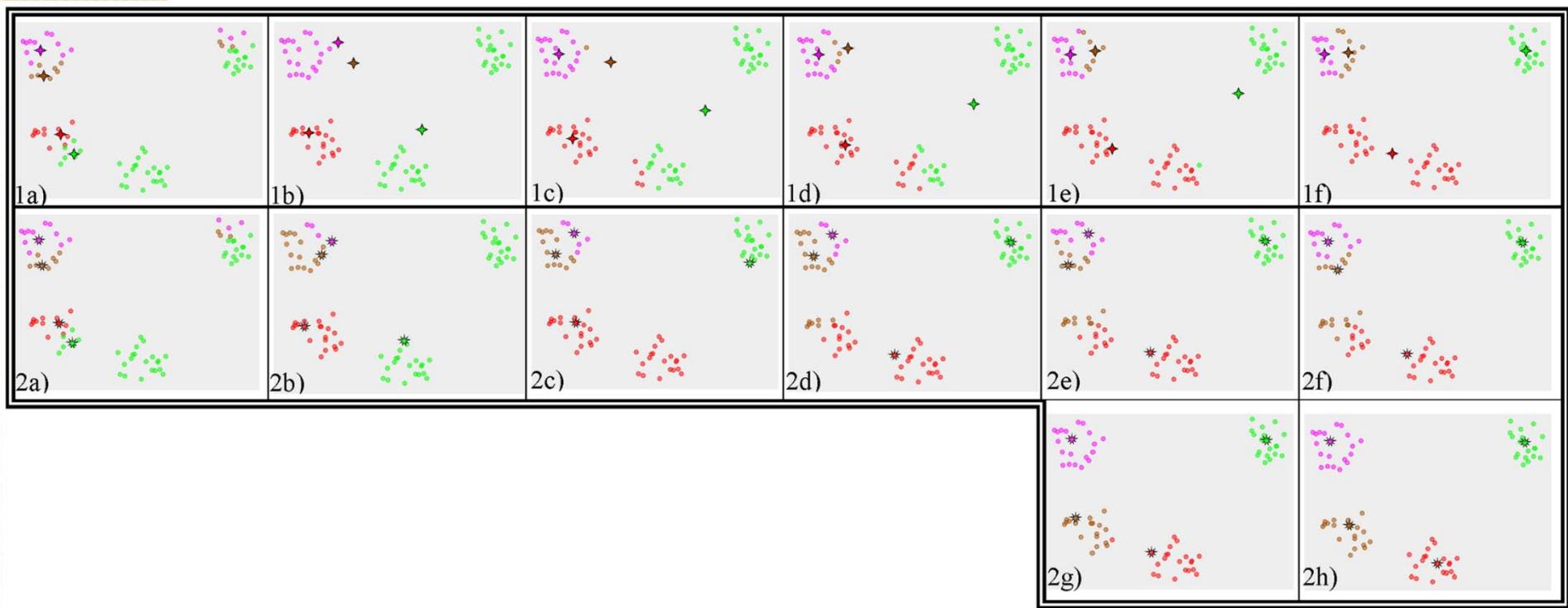
(a) Mean



(b) Medoid

CLUSTERING

- k-means
 - k-medoids
 - 1: k-means, 2: k-medoids





CLUSTERING

- **k-means**
 - **k-medoids**
 - En general, k-medoids es una alternativa al k-means que es más adecuada para conjuntos de datos con valores atípicos y distribuciones no Gaussianas
 - Ventajas:
 - Robusto frente a valores atípicos y ruido
 - Pueden utilizarse medidas de disimilitud elegidas arbitrariamente (por ejemplo, la distancia coseno) o con cualquier métrica de distancia



CLUSTERING

- **k-means**
 - **k-medoids**
 - Desventajas:
 - El algoritmo K-medoids tiene una gran complejidad temporal
 - Por lo tanto, funciona eficazmente para conjuntos de datos pequeños, pero no se adapta bien a conjuntos de datos grandes
 - En general, es mucho más lento que el k-means
 - No es adecuado para agrupar grupos de objetos no esféricos
 - De forma arbitraria



CLUSTERING

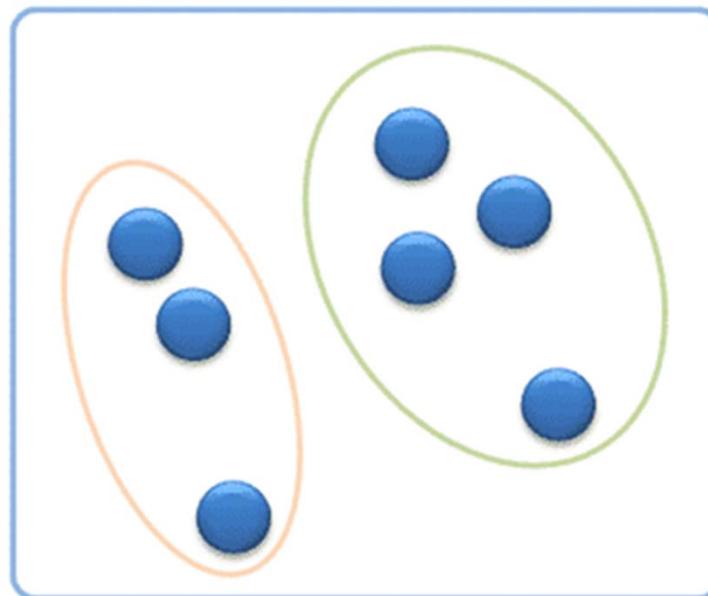
- Clustering jerárquico
 - El clustering jerárquico es un tipo de clustering no supervisado que construye un árbol de clustering o un *dendograma* a partir de los datos
 - Este árbol representa la relación jerárquica entre los clústeres, desde la fusión de clústeres individuales hasta la formación de un gran clúster único.

CLUSTERING

- Clustering jerárquico

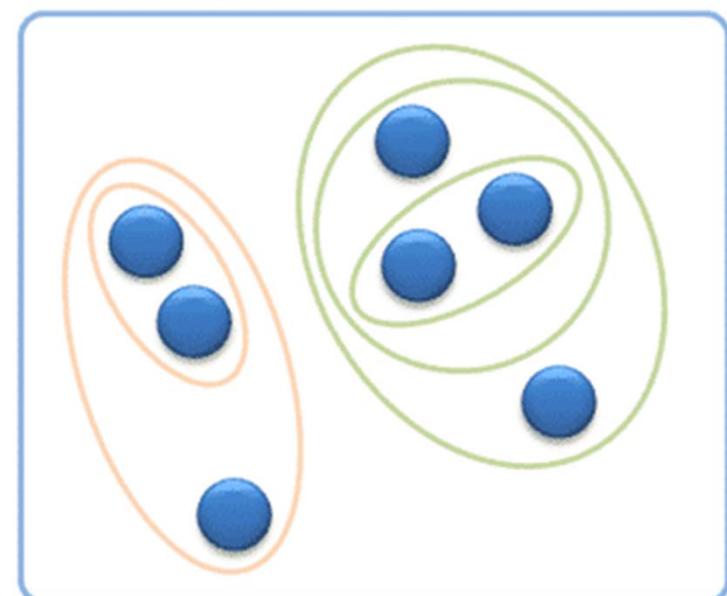
CLUSTER NO JERÁRQUICOS

PARTITIONING CLUSTER



CLUSTER JERÁRQUICOS

HIERARCHICAL CLUSTER



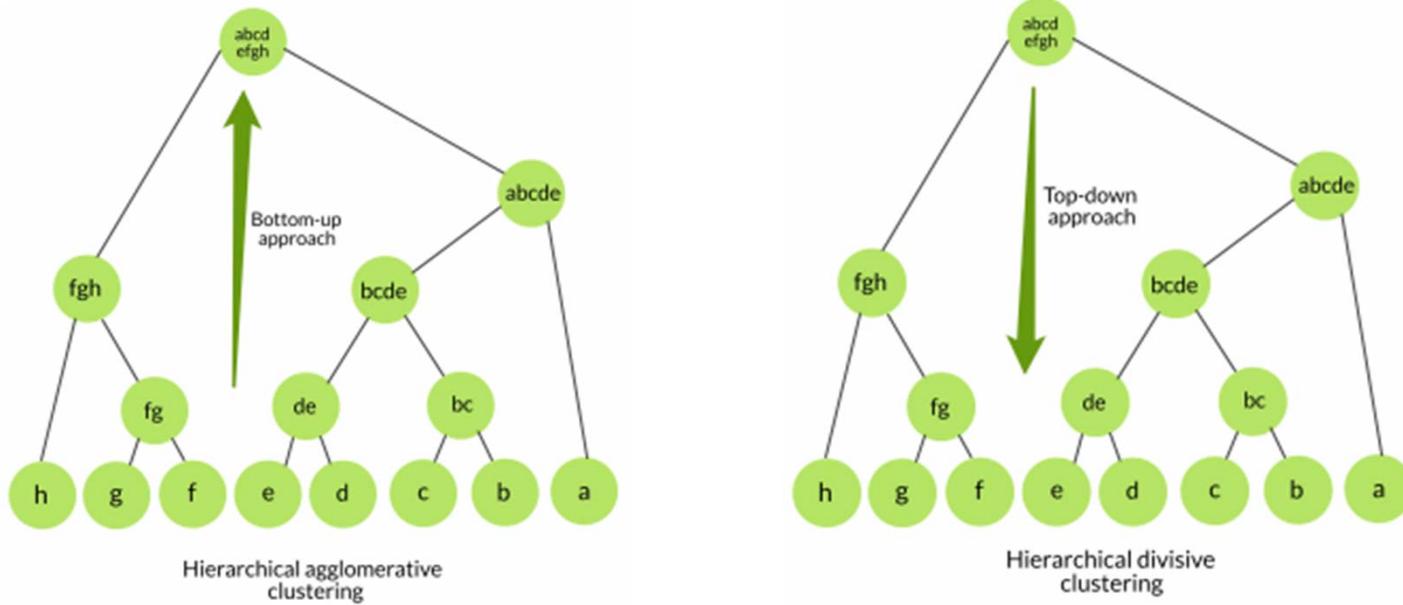


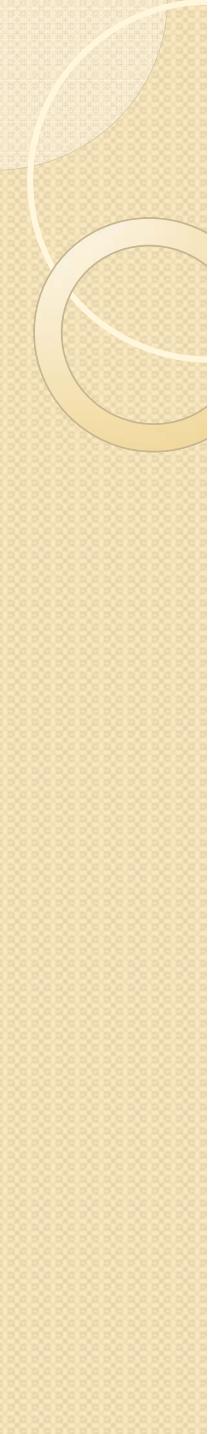
CLUSTERING

- Clustering jerárquico
 - Hay dos tipos principales de clustering jerárquico:
 - Clustering jerárquico aglomerativo
 - Comienza con cada punto de datos como un clúster individual y luego fusiona gradualmente los clusteres más cercanos hasta que todos los puntos de datos estén en un único clúster.
 - Clustering jerárquico divisivo
 - Comienza con todos los puntos de datos en un único clúster y luego divide gradualmente el clúster en clusteres más pequeños hasta que cada punto de datos esté en su propio clúster.

CLUSTERING

- Clustering jerárquico
 - Hay dos tipos principales de clustering jerárquico:
 - Clustering jerárquico aglomerativo
 - Clustering jerárquico divisivo





CLUSTERING

- Clustering jerárquico
 - Clústering Jerárquico Aglomerativo
 - En un enfoque aglomerativo o enfoque ascendente inicialmente cada muestra se trata como un solo clúster
 - Iterativamente, se van fusionando, o aglomerando, pares de clústeres hasta que todos los clústeres se hayan fusionado en uno solo
 - Se va creando así una jerarquía de clústeres



CLUSTERING

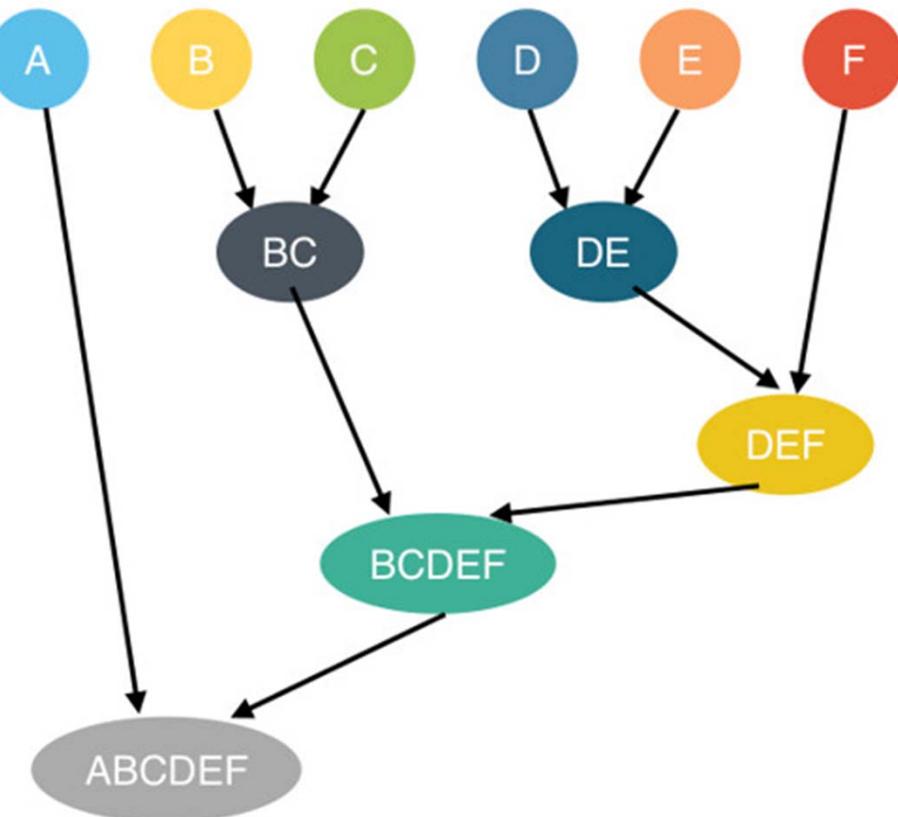
- Clustering jerárquico
 - Clústering Jerárquico Aglomerativo
 - El proceso se realiza de la siguiente manera:
 1. Se asigna cada punto de datos a un clúster individual
 2. Se calcula la distancia entre todos los clústeres y se combinan los dos clústeres más cercanos en un nuevo clúster
 3. Se actualiza la lista de clústeres y se repite el paso 2 hasta que todos los puntos de datos estén en un solo clúster
 4. Se construye la jerarquía de clústeres conectando los clústeres combinados

CLUSTERING

- Clustering jerárquico
 - Clústering Jerárquico Aglomerativo

- Ejemplo: 6 puntos: A, B, C, D, E

1. A, BC, D, E, F
2. A, BC, DE, F
3. A, BC, DEF
4. A, BCDEF
5. ABCDEF





CLUSTERING

- Clustering jerárquico
 - Clústering Jerárquico Aglomerativo
 - Adecuado para conjuntos de datos grandes y complejos, ya que permite la creación de una jerarquía de clústeres que puede ser explorada para encontrar patrones y relaciones en los datos
 - Sin embargo, puede ser más lento que otros algoritmos de clustering debido a la necesidad de calcular las distancias entre todos los clústeres en cada iteración



CLUSTERING

- Clustering jerárquico
 - Clustering jerárquico divisivo
 - Proceso inverso
 - En un agrupamiento divisivo o de arriba hacia abajo, un solo clúster de todas las muestras se divide recursivamente en dos clústeres menos similares hasta que haya un clúster para cada observación
 - Menos utilizado que el aglomerativo
 - Útil cuando se desconocen los números de clusters a priori y se desea explorar la estructura subyacente de los datos
 - Sin embargo, el proceso puede ser computacionalmente costoso y requiere una selección cuidadosa del criterio de parada



CLUSTERING

- Clustering jerárquico
 - Clústering jerárquico divisivo
 - El proceso se realiza de la siguiente manera:
 1. Se selecciona un punto de inicio, por ejemplo, el punto de datos con las características medias más alejadas de todos los demás
 2. Se divide el cluster actual en dos subgrupos utilizando cualquier técnica de clústering, como k-means o k-medoids
 3. Se repiten los pasos 1 y 2 para cada subgrupo recién formado hasta que se cumpla un criterio de parada, como el número máximo de clusters o la homogeneidad dentro de cada cluster
 - Al final, se obtiene un árbol de clustering que representa la estructura de agrupación de los datos

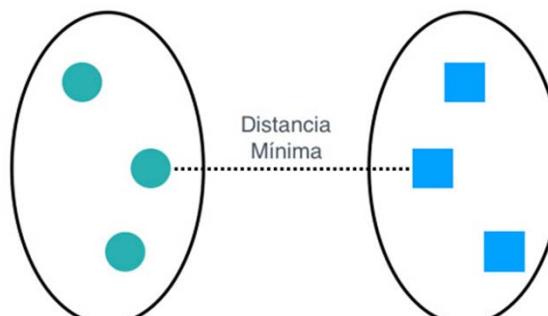
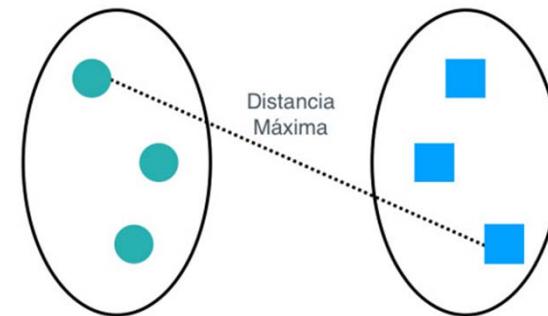


CLUSTERING

- Clustering jerárquico
 - En ambos tipos de clustering jerárquico, se utiliza una medida de distancia para calcular la similitud entre los clusteres y determinar cuándo deben fusionarse o dividirse
 - Métrica más habitual: distancia Euclídea
 - Otras distancia de Manhattan, distancia de Mahalanobis, distancia de coseno
 - Dependiendo de la naturaleza de los datos y de los objetivos
 - ¿Cómo medir distancias entre clústeres?
 - Varios métodos

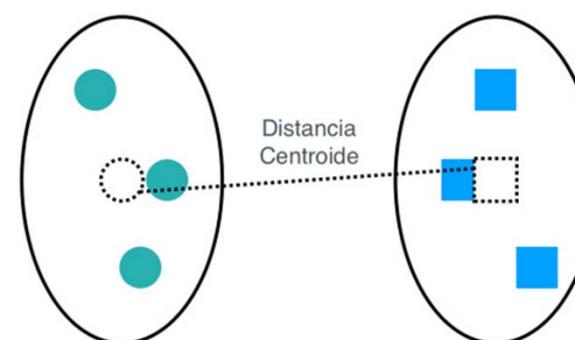
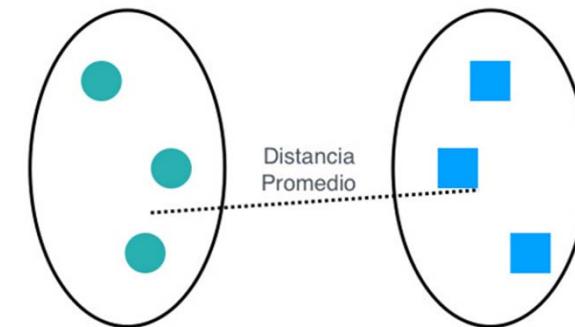
CLUSTERING

- Clustering jerárquico
 - Métodos para medir distancias entre clústeres:
 - Distancia máxima o completa:
 - Mide la distancia máxima entre cualquier par de puntos, uno en cada clúster.
 - Distancia mínima o simple:
 - Distancia mínima entre cualquier par de puntos, uno en cada clúster.



CLUSTERING

- Clustering jerárquico
 - Métodos para medir distancias entre clústeres:
 - Distancia promedio:
 - Distancia promedio entre todos los pares de puntos, uno en cada clúster.
 - Distancia de centroide:
 - Se hallan los centroides, y se calculan las distancias entre los mismos.





CLUSTERING

- Clustering jerárquico
 - Métodos para medir distancias entre clústeres:
 - Método de Ward (método de mínima varianza) (1/3):
 - Aglomerativo
 - Objetivo: reducir la varianza de los clústeres resultantes
 - Según este método, la distancia entre dos clústeres, A y B, se calcula como el incremento de la suma de los cuadrados cuando se unan, es decir:
 - Para cada clúster A y B se calcula el centroide y la suma de cuadrados de distancias de cada punto al centroide
 - Se realiza el mismo proceso para el clúster $A \cup B$
 - Se calcula la diferencia entre la suma de cuadrados de $A \cup B$ menos la suma de cuadrados de A y de B

CLUSTERING

- Clustering jerárquico

- Métodos para medir distancias entre clústeres:
 - Método de Ward (método de mínima varianza) (2/3):
 - Aglomerativo
 - Objetivo: reducir la varianza de los clústeres resultantes
 - Según este método, la distancia entre dos clústeres, A y B, se calcula como el incremento de la suma de los cuadrados cuando se unan, es decir:

$$\Delta(A, B) = \sum_{i \in A \cup B} \underbrace{\|\vec{x}_i - \vec{m}_{A \cup B}\|^2}_{\text{Distancia entre la muestra } x_i \text{ (i } \in A \cup B\text{) y el centroide (media de las muestras) de } A \cup B} - \underbrace{\sum_{i \in A} \|\vec{x}_i - \vec{m}_A\|^2}_{\text{Distancia entre la muestra } x_i \text{ (i } \in A\text{) y el centroide (media de las muestras) de A}} - \underbrace{\sum_{i \in B} \|\vec{x}_i - \vec{m}_B\|^2}_{\text{Distancia entre la muestra } x_i \text{ (i } \in B\text{) y el centroide (media de las muestras) de B}} = \frac{n_A n_B}{n_A + n_B} \|\vec{m}_A - \vec{m}_B\|^2$$

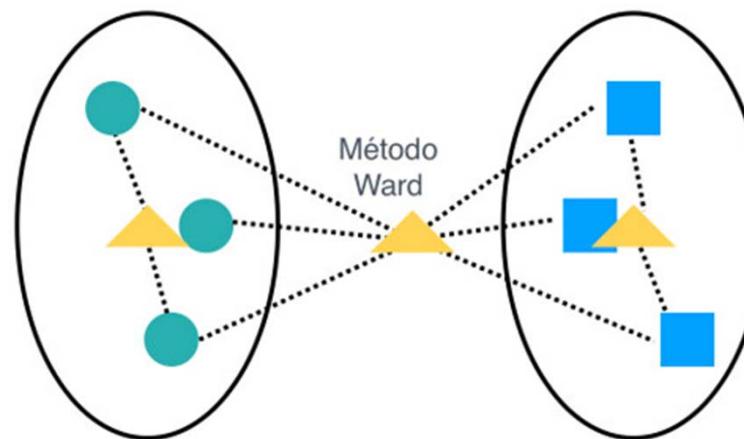
Distancia entre la muestra x_i ($i \in A \cup B$) y el centroide (media de las muestras) de $A \cup B$

Distancia entre la muestra x_i ($i \in A$) y el centroide (media de las muestras) de A

Distancia entre la muestra x_i ($i \in B$) y el centroide (media de las muestras) de B

CLUSTERING

- Clustering jerárquico
 - Métodos para medir distancias entre clústeres:
 - Método de Ward (método de mínima varianza) (3/3):
 - Por lo tanto, minimizar la distancia anterior conlleva escoger un par de clústeres que incrementen el mínimo la varianza interna después de la fusión



- Es necesario probar todas las posibles combinaciones de clústeres

CLUSTERING

- Clustering jerárquico
 - Métodos para medir distancias entre clústeres:

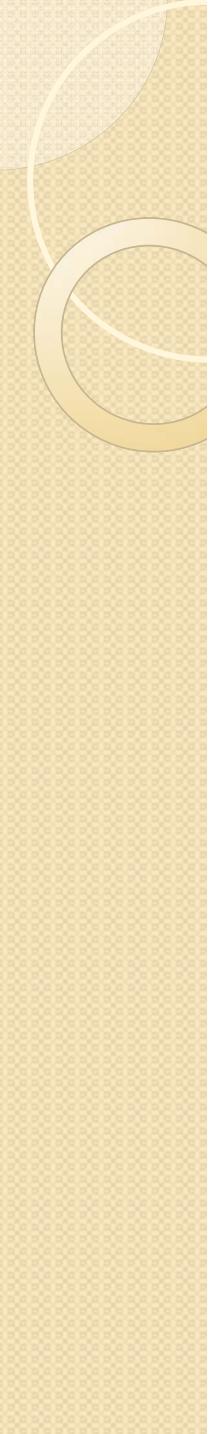
Maximum or complete-linkage clustering
Minimum or single-linkage clustering
Unweighted average linkage clustering (UPGMA)
Weighted average linkage clustering (WPGMA)
Centroid linkage clustering (UPGMC)
Median linkage clustering (WPGMC)
Versatile linkage clustering
Ward linkage, Minimum Increase of Sum of Squares (MISSQ)

Minimum Error Sum of Squares (MNSSQ)
Minimum Error Sum of Squares (MNSSQ)
Minimum Increase in Variance (MIVAR)
Minimum Variance (MNVAR)
Mini-Max linkage
Hausdorff linkage
Minimum Sum Medoid linkage
Minimum Sum Increase Medoid linkage
Medoid linkage
Minimum energy clustering



CLUSTERING

- Clustering jerárquico
 - El clustering jerárquico es útil cuando se desea explorar la estructura subyacente de los datos y se busca una solución de clustering con un número variable de clusteres.
 - Sin embargo, puede ser más lento que otros algoritmos de clustering debido a la necesidad de calcular la distancia entre todos los pares de clusteres a lo largo del proceso

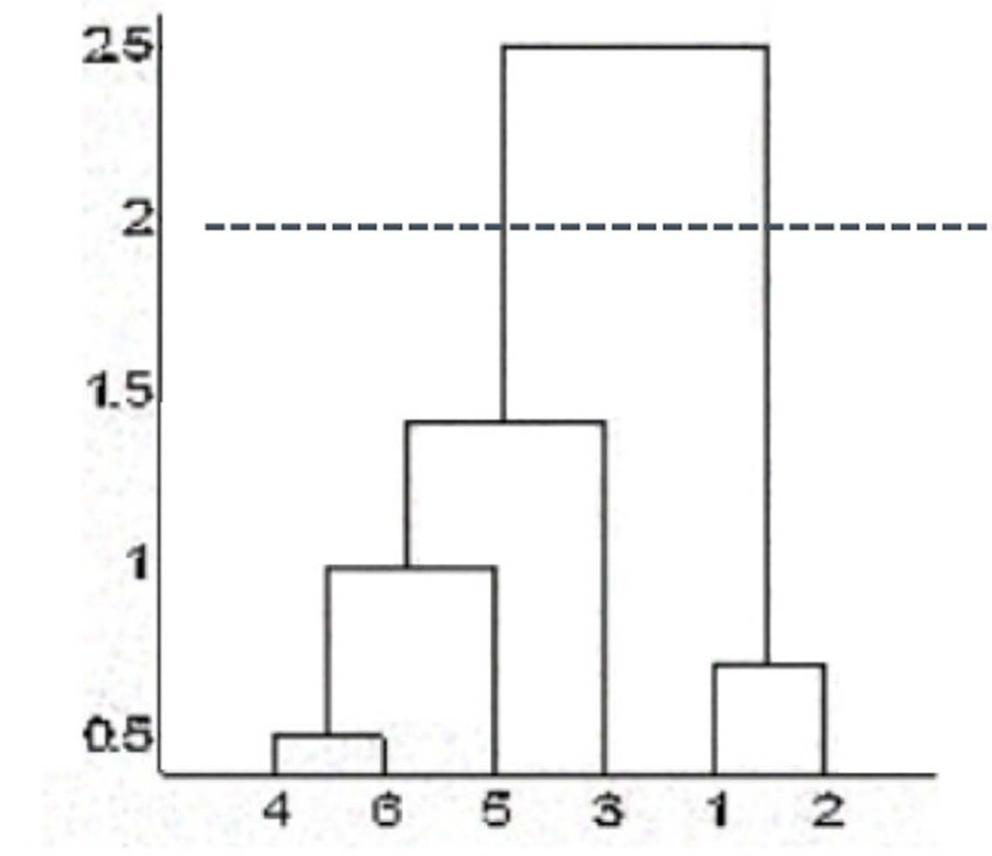


CLUSTERING

- Clustering jerárquico
 - Permite construir estructuras de árbol (dendogramas) a partir de similitudes de datos
 - Al contrario que en clustering no jerárquico, el número de clústeres no tiene que estar fijado por el usuario de antemano
 - Una vez que se tiene el dendograma, se puede “partir” por distintos puntos para analizar con números distintos de clústeres
 - Esto se logra al cortar el árbol en un cierto punto y considerar cada rama cortada como un clúster individual

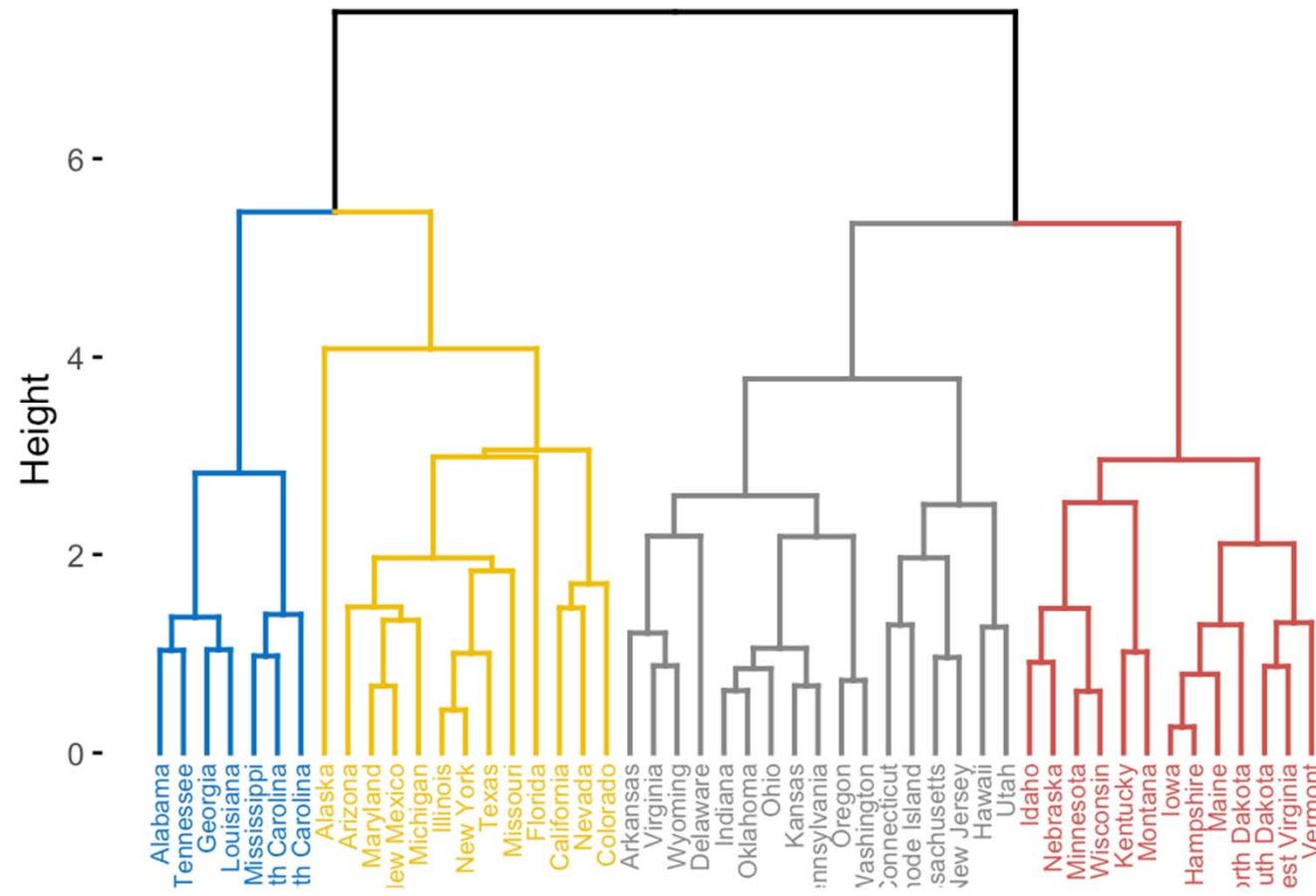
CLUSTERING

- Clustering jerárquico
 - Ejemplo de dendograma:



CLUSTERING

- Clustering jerárquico
 - Ejemplo de dendograma:





CLUSTERING

- Clustering jerárquico
 - Desventajas:
 - El clustering jerárquico es computacionalmente costoso por lo tanto no se recomienda en grandes conjuntos de datos
 - Sensible al ruido y a los valores atípicos



CLUSTERING

- DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*)
 - Algoritmo de clustering de densidad basado en la vecindad
 - Los clústeres se forman identificando áreas densas de puntos o datos en el espacio de datos, donde los puntos están estrechamente agrupados
 - Los puntos se asignan a clústeres según su vecindad, es decir, la cantidad de puntos cercanos a ellos



CLUSTERING

- DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*)
 - Parámetros:
 - eps (epsilon): La distancia máxima entre un par de puntos vecinos
 - Típicamente, distancia euclídea
 - Pero puede ser otra medida
 - Dos puntos se consideran vecinos si están separados por una distancia menor o igual que eps
 - minPts: El número mínimo de puntos necesarios para formar un cluster denso.

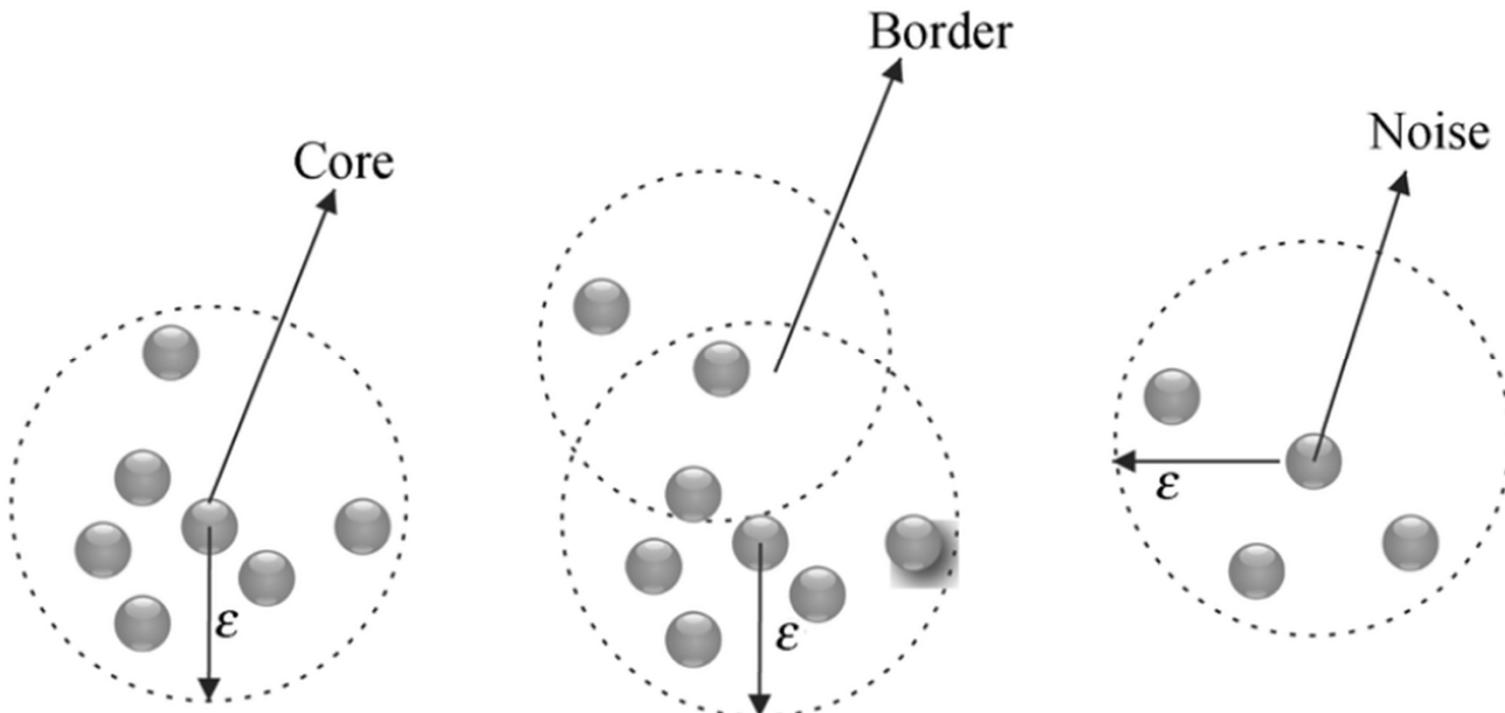


CLUSTERING

- DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*)
 - Los puntos de datos pueden ser:
 - "Core point" si tiene al menos un número específico (minPts) de puntos cercanos dentro de un radio específico (eps)
 - El punto está en una región densa
 - "Border point" si no es "core point" pero está dentro del radio eps de algún "core point"
 - El punto se encuentra en la vecindad de una región densa
 - "Noise point" si no es "core point" ni "border point"
 - Ruido o valor atípico, no asociado con ningún clúster

CLUSTERING

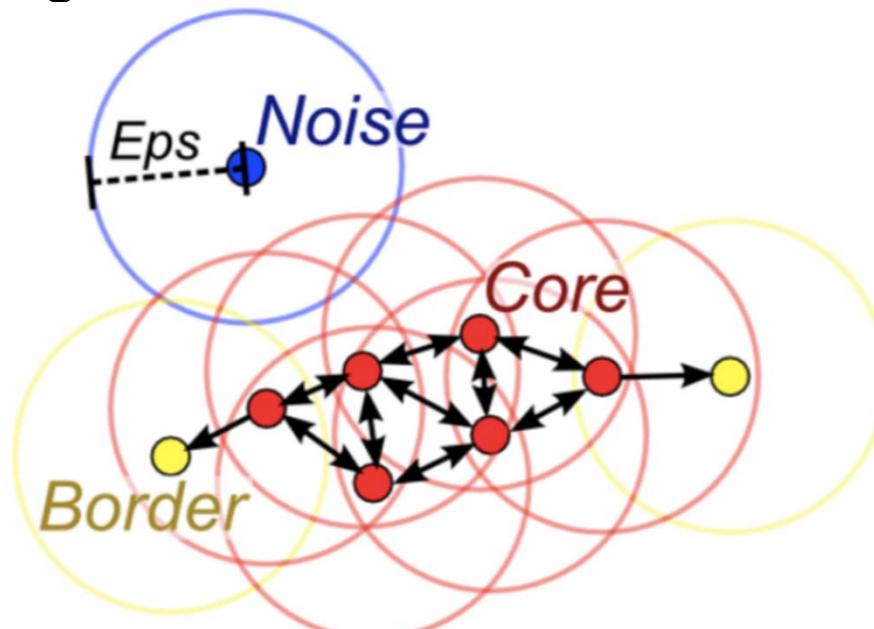
- DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*)



- eps : Radio máximo del vecindario
- minPts : Número mínimo de puntos en un vecindario

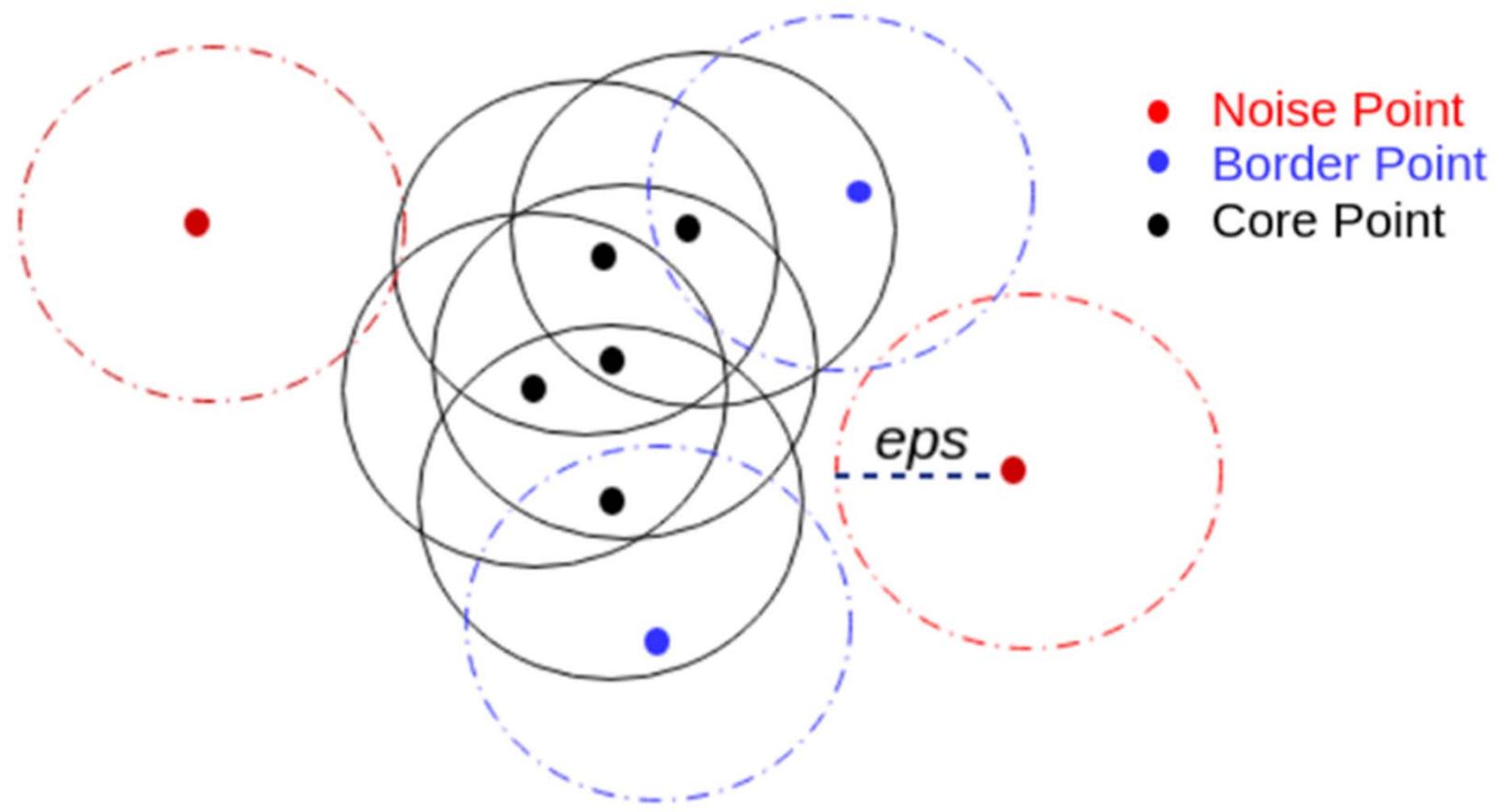
CLUSTERING

- DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*)
 - Estas definiciones permiten establecer conjuntos de puntos "conectados por vecindad":
 - Darán lugar a los clústeres



CLUSTERING

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

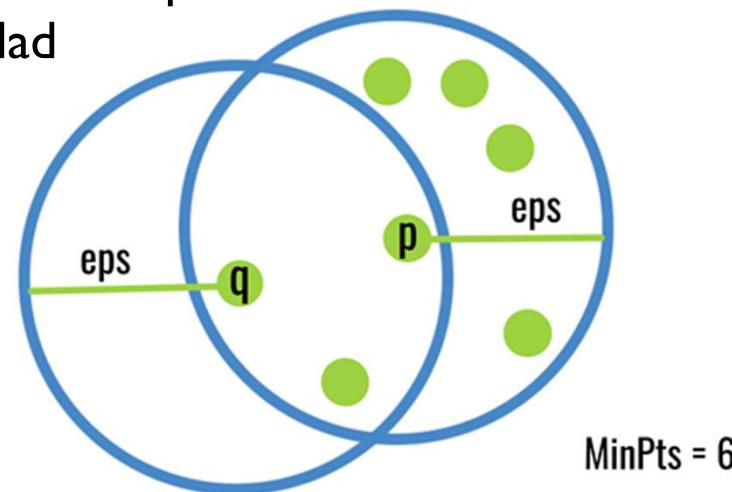


CLUSTERING

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

- Definiciones adicionales (1/3):

- *Direct density reachable*:
- Un punto se llama alcanzable por densidad directa si tiene un *core point* en su vecindad

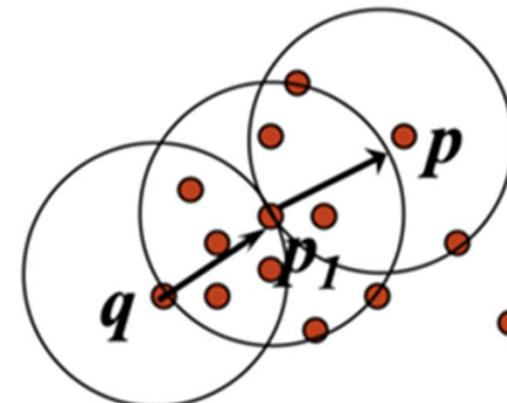
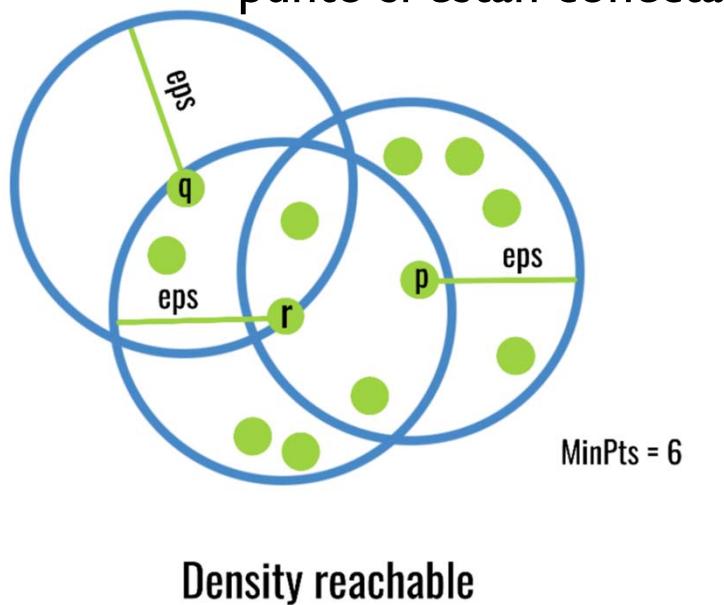


Directly density reachable

CLUSTERING

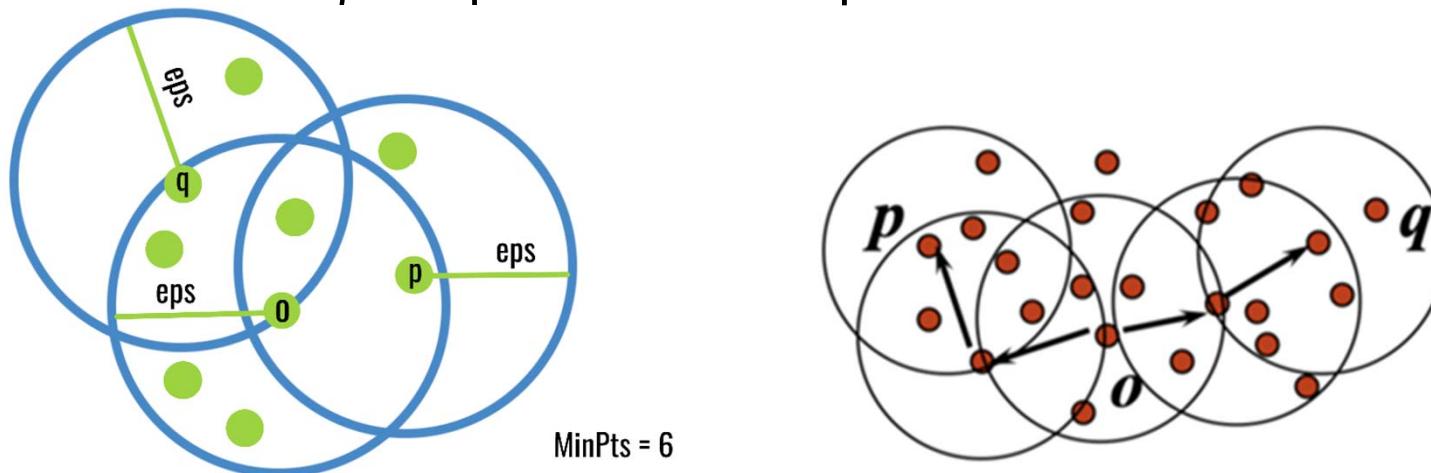
- DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*)

- Definiciones adicionales (2/3):
 - *Density Reachable*:
 - Un punto se denomina alcanzable por densidad desde otro punto si están conectados a través de una serie de *core points*



CLUSTERING

- DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*)
 - Definiciones adicionales (3/3):
 - *Density Connected*:
 - Dos puntos se denominan conectados por densidad si existe un *core point* que es alcanzable por densidad desde ambos puntos



Density connectivity



CLUSTERING

- DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*)
 - El algoritmo DBSCAN se ejecuta en tres pasos:
 1. Se selecciona un punto al azar y se busca su vecindad
 2. Si hay al menos minPts puntos dentro del radio eps, se forma un nuevo clúster alrededor de ese punto y se buscan las vecindades de los puntos dentro de ese clúster
 - Es un *core point*
 3. Se continúa buscando vecindades hasta que no se encuentre ninguna más
 4. Se selecciona otro punto no asignado y se repite el proceso desde el paso 1

CLUSTERING

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

DBSCAN



k-means



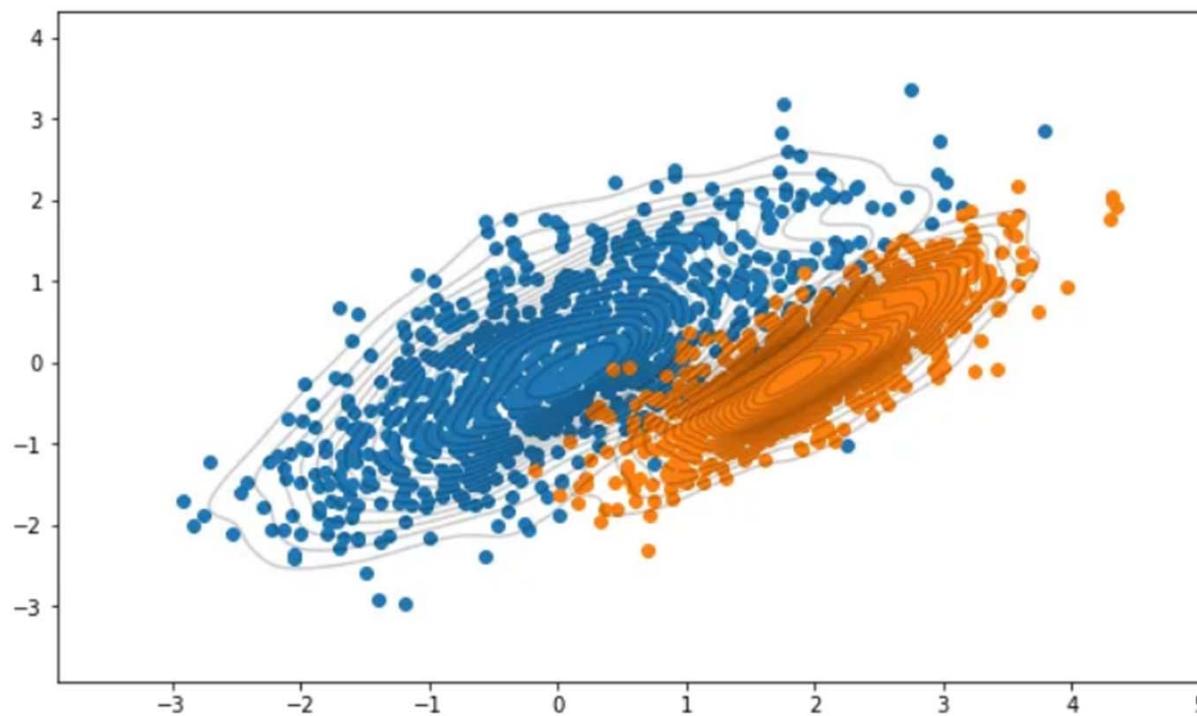


CLUSTERING

- DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*)
 - El algoritmo DBSCAN es útil para detectar clústeres de formas arbitrarias y para identificar puntos atípicos
 - Desventajas:
 - No garantiza encontrar el número exacto de clústeres
 - Puede ser sensible a los valores de eps y minPts elegidos

CLUSTERING

- GMM (*Gaussian Mixture Model*)
 - Este algoritmo supone que los datos provienen de una combinación de diferentes distribuciones Gaussianas





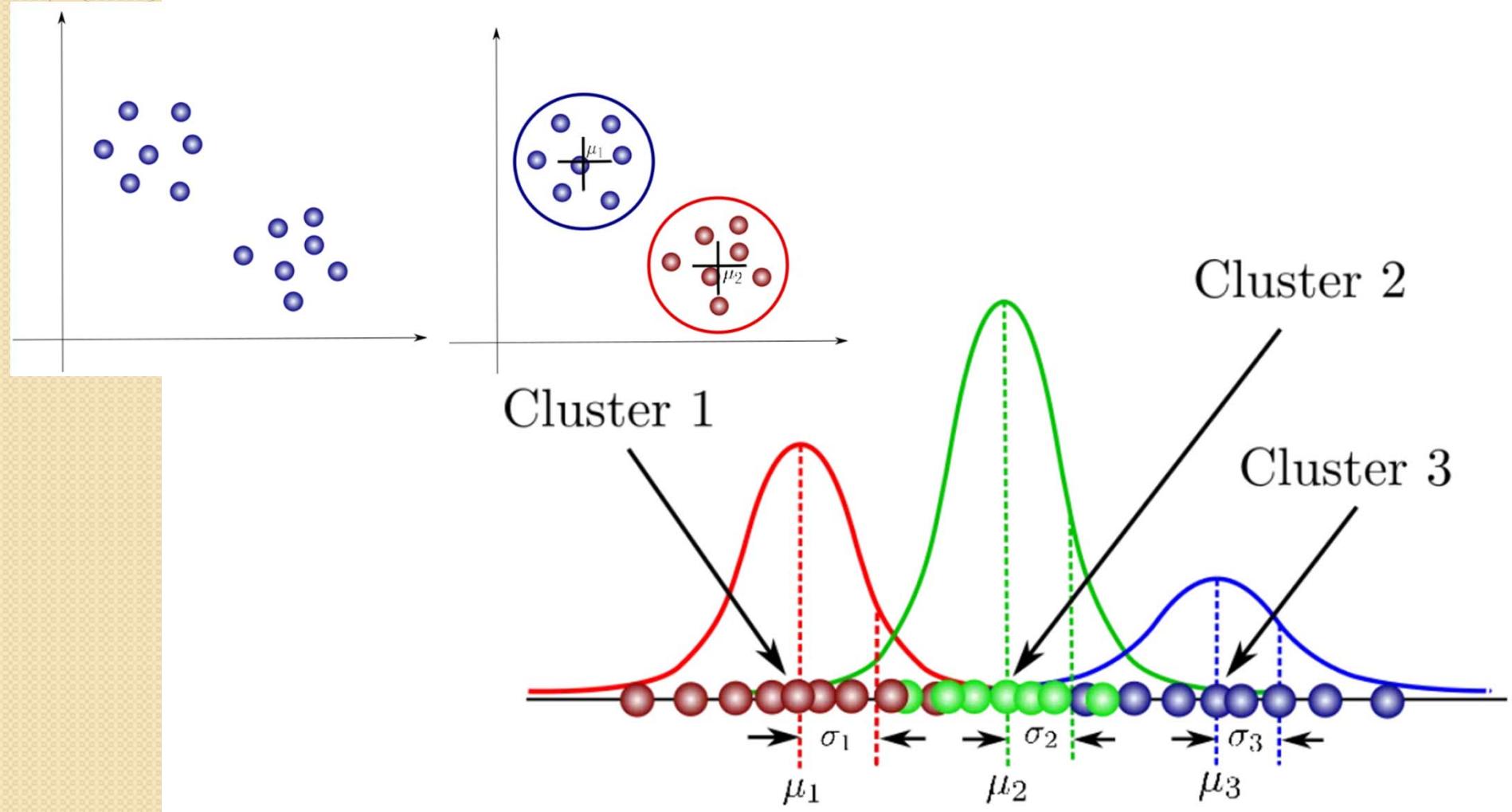
CLUSTERING

- GMM (*Gaussian Mixture Model*)
 - Este algoritmo supone que los datos provienen de una combinación de diferentes distribuciones Gaussianas
 - Parámetros que usa el algoritmo:
 - Media de cada clúster (μ_k)
 - Centro de cada clúster
 - Covarianza de cada clúster (Σ_k)
 - Ancho de cada clúster
 - Proporción de mezcla de cada clúster (π_k)
 - Tamaño de la función Gaussiana de cada clúster

$$\sum_{k=1}^K \pi_k = 1$$

CLUSTERING

- GMM (*Gaussian Mixture Model*)





CLUSTERING

- GMM (*Gaussian Mixture Model*)
 - El proceso comienza con la inicialización de los parámetros de las distribuciones Gaussianas
 - Luego, se asignan probabilidades a cada punto de datos de pertenecer a cada clúster, utilizando la función de probabilidad Gaussiana
 - A continuación, se actualizan los parámetros de cada clúster y se vuelven a asignar los puntos a clústeres
 - Este proceso se repite hasta que los parámetros de los clústeres convergen o hasta que se alcance el número máximo de iteraciones



CLUSTERING

- GMM (*Gaussian Mixture Model*)

- Algoritmo:

- 1. Inicialización de los parámetros de los clústeres:

- Se puede ejecutar el algoritmo k-means para crear unos clústeres iniciales

- 2. Asignación de puntos de datos a clústeres:

- Para cada punto de datos x_i , calcular la probabilidad $P(k|x_i)$ de pertenecer a cada clúster k utilizando la función de probabilidad Gaussiana
 - Asignar el punto de datos x_i al clúster k con la probabilidad máxima $P(k|x_i)$

- 3. Actualización de los parámetros de los clústeres:

- Para cada clúster k , actualizar la media μ_k y la covarianza Σ_k utilizando los puntos de datos asignados a ese clúster
 - Actualizar la proporción de la mezcla π_k para cada clúster k

- 4. Repetir los pasos 2 y 3 hasta que los parámetros de los clústeres converjan o se alcance el número máximo de iteraciones



CLUSTERING

- **GMM (*Gaussian Mixture Model*)**
 - GMM es un algoritmo flexible y puede ser utilizado para modelar la distribución compleja de datos, incluyendo distribuciones multivariadas y múltiples modos
 - Desventajas:
 - Necesidad de seleccionar el número correcto de clústeres
 - Sensibilidad a la elección inicial de los parámetros de los clústeres



APRENDIZAJE NO SUPERVISADO

- Clustering
 - k-means
 - Clústering jerárquico
 - DBSCAN
 - GMM
- **Reducción de dimensionalidad**
- Asociación de reglas
 - Apriori
- Detección de anomalías
 - k-NN
 - One-class classification
 - One-class SVM
 - Isolation Forest
- Redes de Neuronas Artificiales no supervisadas
 - Mapas autoorganizativos
 - Autoencoders



REDUCCIÓN DE DIMENSIONALIDAD

- Representar los datos en un espacio de características más pequeño sin perder demasiada información
- Útil para:
 - Visualizar y comprender los datos
 - Puede ser un paso importante en el pre-procesamiento de los datos antes de aplicar técnicas de aprendizaje supervisado



REDUCCIÓN DE DIMENSIONALIDAD

- Principales técnicas:
 - Análisis de componentes principales (PCA)
 - Análisis discriminante lineal (LDA)
 - Técnicas de proyección non-lineal
 - Isomap
 - t-SNE
 - Reducción de dimensionalidad basada en árboles
 - **Random Forest**
 - **XGBoost**
 - Autoencoders
 - Factorización matricial



REDUCCIÓN DE DIMENSIONALIDAD

- Random Forest y XGBoost
 - Técnicas de aprendizaje supervisado para clasificación y regresión
 - No se utiliza específicamente para hacer reducción de dimensionalidad
 - Sin embargo, se pueden utilizar con este objetivo
 - Ambas se basan en la generación de un gran número de árboles de decisión a partir de un submuestreo del dataset y del conjunto de atributos
 - Salida final: Votación de todos los árboles
 - Técnicas de **ensemble**



REDUCCIÓN DE DIMENSIONALIDAD

- Random Forest y XGBoost
 - Una vez entrenado un conjunto de árboles en un conjunto de datos de alta dimensionalidad, es posible evaluar la importancia de cada característica utilizada en el modelo
 - Las características con baja importancia pueden eliminarse del conjunto de datos, lo que puede reducir la dimensionalidad efectiva de los datos
 - Esto puede ayudar a mejorar la eficiencia y la velocidad de otros algoritmos de aprendizaje automático.



REDUCCIÓN DE DIMENSIONALIDAD

- Random Forest
 - Dos maneras de medir la importancia:
 - *Mean Decrease Gini*
 - Disminución de la impureza en los nodos del árbol donde se utiliza esa característica para separar los datos
 - *Mean Decrease Accuracy*
 - Cuánta precisión pierde el modelo al excluir cada atributo
 - Estas medidas se calculan para cada árbol del conjunto y se promedian para obtener una estimación de la importancia de cada característica



REDUCCIÓN DE DIMENSIONALIDAD

- **XGBoost (*eXtreme Gradient Boosting*)**
 - Tres manera de medir la importancia:
 - Peso:
 - Número de veces que un atributo aparece en un árbol
 - Ganancia:
 - Cantidad en la que cada atributo en una división mejora la medida de rendimiento, ponderada por el número de ejemplos
 - Medidas de rendimiento: pureza (índice de Gini), error, etc.
 - Cobertura:
 - Número de muestras afectadas por una división de ese atributo
 - Después se promedia sobre todos los árboles
 - Los tres promedios son proporcionales a la importancia de la característica



REDUCCIÓN DE DIMENSIONALIDAD

- Random Forest y XGBoost
 - Ventaja frente a PCA:
 - Los atributos resultantes son atributos del conjunto original
 - En PCA cada atributo resultante es una combinación lineal de todos los del conjunto original de atributos
 - Se han eliminado los menos importantes
 - Sin embargo, es importante tener en cuenta que la reducción de dimensionalidad mediante la eliminación de características debe realizarse con precaución
 - Se debe evaluar el impacto en la precisión del modelo antes de aplicarlo en producción



APRENDIZAJE NO SUPERVISADO

- Clustering
 - k-means
 - Clústering jerárquico
 - DBSCAN
 - GMM
- Reducción de dimensionalidad
- **Asociación de reglas**
 - Apriori
- Detección de anomalías
 - k-NN
 - One-class classification
 - One-class SVM
 - Isolation Forest
- Redes de Neuronas Artificiales no supervisadas
 - Mapas autoorganizativos
 - Autoencoders



ASOCIACIÓN DE REGLAS

- Principales técnicas:
 - Algoritmo Apriori
 - ECLAT (*Equivalence Class Clustering and bottom-up Lattice Traversal*)
 - Reglas de asociación basadas en arboles, ejemplo:
 - CART (*Classification and Regression Trees*)
 - CHAID (*Chi-squared Automatic Interaction Detection*)
 - Algoritmo FP-Growth (*Frequent Pattern Growth*)
 - Algoritmo Association Rule Mining basado en Redes Bayesianas



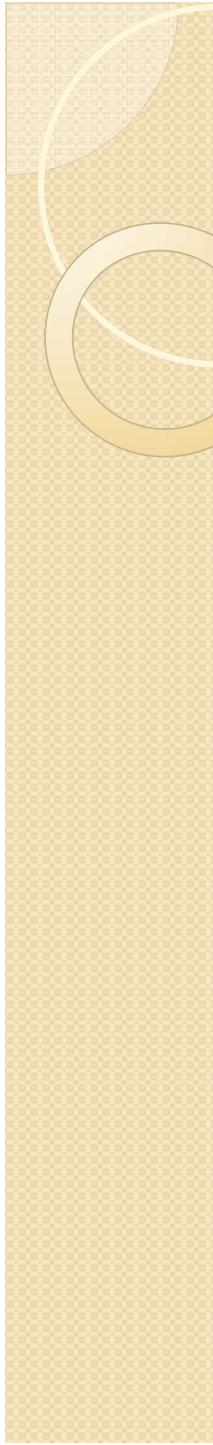
ASOCIACIÓN DE REGLAS

- Principales técnicas:
 - **Algoritmo Apriori**
 - ECLAT (*Equivalence Class Clustering and bottom-up Lattice Traversal*)
 - Reglas de asociación basadas en arboles, ejemplo:
 - CART (*Classification and Regression Trees*)
 - CHAID (*Chi-squared Automatic Interaction Detection*)
 - Algoritmo FP-Growth (*Frequent Pattern Growth*)
 - Algoritmo Association Rule Mining basado en Redes Bayesianas



ASOCIACIÓN DE REGLAS

- **Apriori:**
 - El algoritmo Apriori es un algoritmo de minería de datos utilizado para encontrar patrones frecuentes o reglas de asociación en grandes conjuntos de datos transaccionales
 - Funciona mediante la generación y prueba de hipótesis sobre los elementos frecuentes en los datos



ASOCIACIÓN DE REGLAS

- **Apriori:**
 - Ejemplo: Análisis de patrones de compras en un supermercado:
 - Los datos contienen información sobre los productos comprados por cada cliente
 - Un ejemplo de regla de asociación generada por Apriori podría ser
 - "Si un cliente compra leche, también es probable que compre pan" con un soporte del 60% y una confianza del 80%
 - En el 60% de las compras en el supermercado, los clientes compran tanto leche como pan
 - En el 80% de las compras en las que se compra leche, también se compra pan



ASOCIACIÓN DE REGLAS

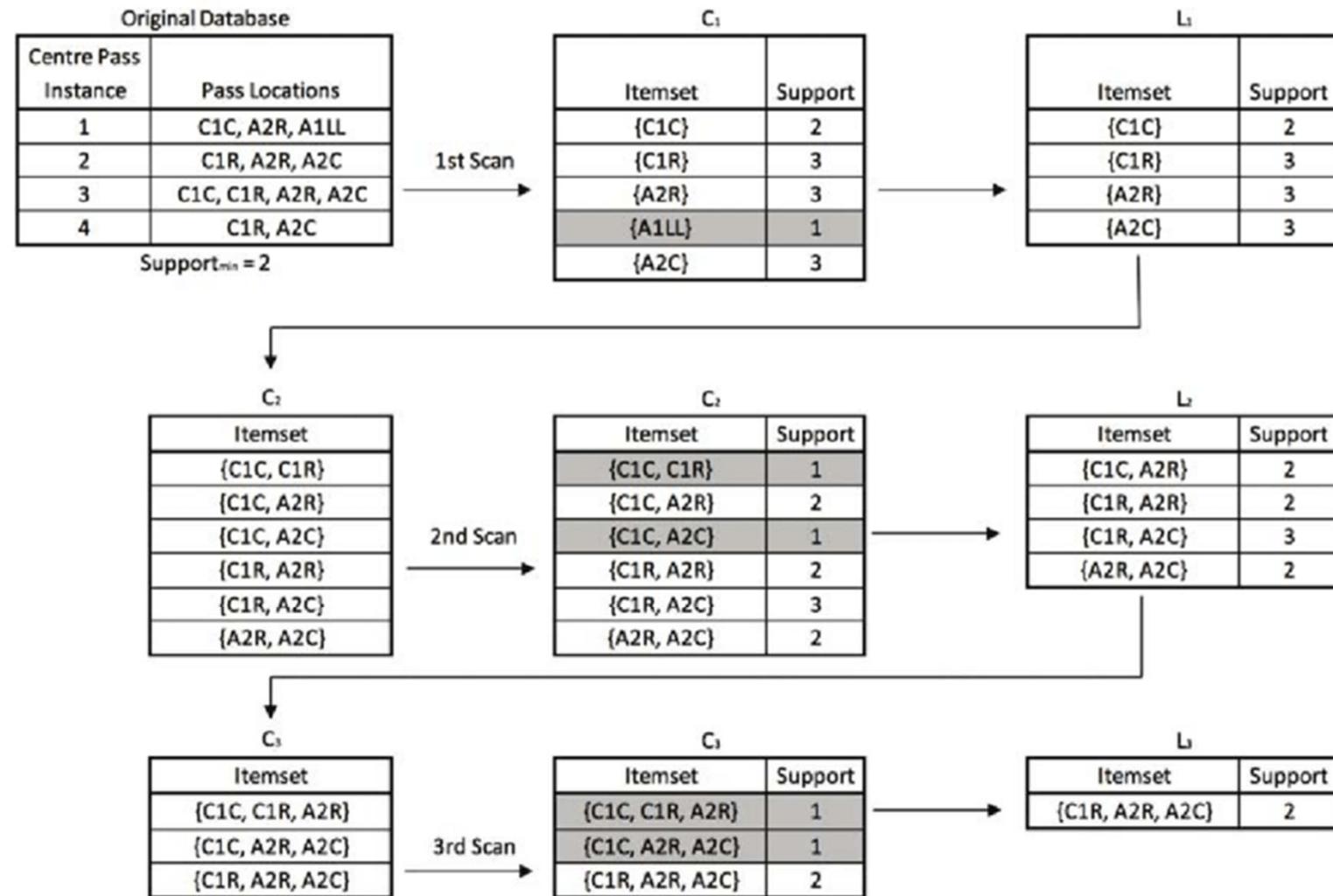
- **Apriori:**

- **Algoritmo:**

1. **$k=1$:** número de elementos en las combinaciones
2. **Calcular combinaciones de k elementos**
 - Si $k=1$, cada combinación contiene un elemento del *dataset*
 - Si no, se usan las combinaciones de la iteración anterior
3. **Calcular el soporte de las combinaciones**
 - Frecuencia conjunta de los elementos de cada combinación en el *dataset*
4. **Seleccionar las combinaciones confiables**
 - Se seleccionan aquellas reglas con un soporte superior a un umbral determinado.
5. **Incrementar k y repetir los pasos 2-4**
 - Combinaciones más complejas

ASOCIACIÓN DE REGLAS

- Apriori:





ASOCIACIÓN DE REGLAS

- **Apriori:**
 - Algoritmo eficiente en términos de tiempo de ejecución
 - Útil para encontrar patrones frecuentes en grandes conjuntos de datos transaccionales



APRENDIZAJE NO SUPERVISADO

- Clustering
 - k-means
 - Clústering jerárquico
 - DBSCAN
 - GMM
- Reducción de dimensionalidad
- Asociación de reglas
 - Apriori
- **Detección de anomalías**
 - k-NN
 - One-class classification
 - One-class SVM
 - Isolation Forest
- Redes de Neuronas Artificiales no supervisadas
 - Mapas autoorganizativos
 - Autoencoders



DETECCIÓN DE ANOMALÍAS

- Anomalía \neq dato con ruido o erróneo
- La detección de anomalías abarca una amplia gama de técnicas y algoritmos diseñados para identificar datos que son inusuales o fuera de lo común en un *dataset*
 - Estos pueden incluir datos que son raros, atípicos o simplemente diferentes a lo que se espera



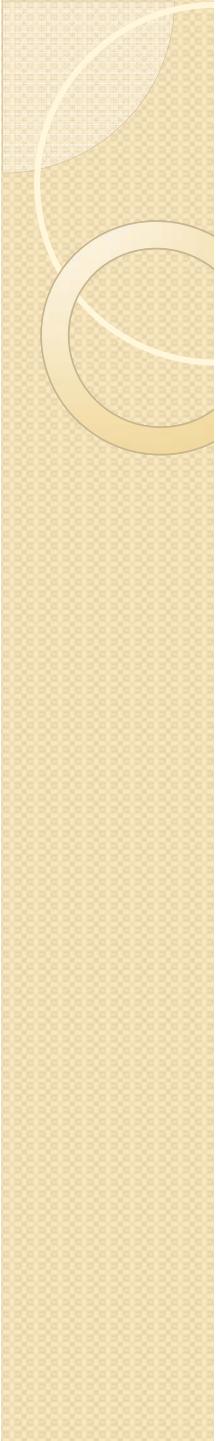
DETECCIÓN DE ANOMALÍAS

- La idea que subyace a la detección de anomalías se basa en el supuesto de que las instancias de datos válidas tienden a tener una determinada estructura
 - Una vez determinadas ciertas características de esta estructura, se pueden analizar otras instancias de datos para ver si sus rasgos exhiben una estructura común a los puntos de datos válidos y, si es así, entonces se pueden etiquetar como válidas
 - En cambio, si su estructura difiere, se etiquetan como anomalías



DETECCIÓN DE ANOMALÍAS

- Identificar anomalías puede ser el objetivo final en sí mismo
 - Por ejemplo:
 - Detección de fraudes
 - Alarmas
- También pueden aportar información adicional sobre los datos
 - Por ejemplo:
 - Descubrir atributos irrelevantes
 - Descubrir que faltan atributos necesarios



DETECCIÓN DE ANOMALÍAS

- La detección de anomalías también puede ser de gran utilidad en desarrollo de modelos de aprendizaje supervisado
 - Una vez identificadas las anomalías, pueden eliminarse de los conjuntos de datos utilizados para entrenar los modelos predictivos
 - Esto reducirá el ruido en los datos, reforzando así la precisión de los modelos predictivos



DETECCIÓN DE ANOMALÍAS

- Cuidado: pueden surgir anomalías debido a la insuficiencia de datos
 - Cuantos menos datos se tengan, más difícil será discernir patrones en los datos.
 - De ahí la importancia de garantizar un tamaño de muestra adecuado.



DETECCIÓN DE ANOMALÍAS

- Las detección de anomalías se utiliza en una amplia variedad de aplicaciones
 - Por ejemplo:
 - Seguridad cibernética
 - Monitorización del rendimiento
 - Detección de fraudes
 - Identificación de fallos en sistemas
 - etc.



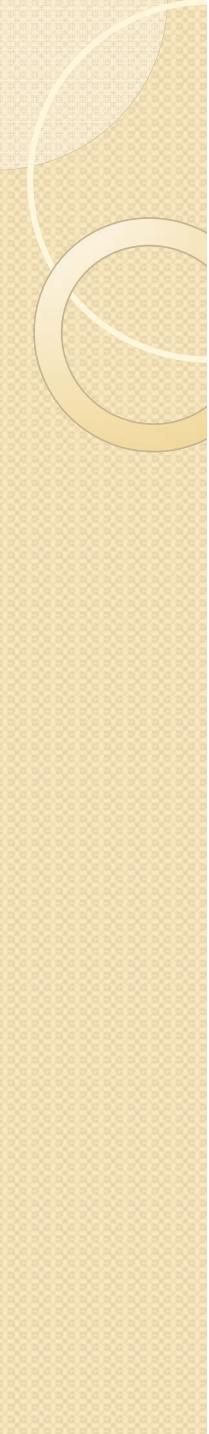
DETECCIÓN DE ANOMALÍAS

- Principales técnicas:
 - k-Nearest Neighbor (KNN)
 - One-class classification
 - Isolation Forest
 - One-Class SVM
 - Algoritmos de clústering
 - DBSCAN
 - Gaussian Mixture Models (GMM)
 - Autoencoders
 - Local Outlier Factor (LOF)



DETECCIÓN DE ANOMALÍAS

- **k-NN**
 - El algoritmo k-NN puede ser utilizado para detectar anomalías midiendo la distancia de una instancia a sus k vecinos más cercanos
 - Si la distancia promedio de una instancia a sus vecinos más cercanos es superior a un umbral, se considera una anomalía



DETECCIÓN DE ANOMALÍAS

- **k-NN**
 - Esta técnica es sencilla y rápida
 - Desventajas:
 - Necesidad de definir umbral
 - Necesidad de definir el valor de k
 - Valor alto: menos sensible a anomalías
 - Valor bajo: demasiado sensible a anomalías, con muchos falsos positivos



DETECCIÓN DE ANOMALÍAS

- One-class classification
 - La clasificación de una clase es un subconjunto de la detección de anomalías, donde solo se considera una clase normal y se detectan puntos que no pertenecen a esa clase



DETECCIÓN DE ANOMALÍAS

- One-class classification
 - ¿Aprendizaje supervisado o no supervisado?
 - Se podría argumentar que es no supervisado
 - En el aprendizaje no supervisado, el modelo se entrena con datos no etiquetados y trata de identificar patrones y estructuras en los datos sin ayuda externa
 - En la clasificación one-class, el modelo también se entrena con datos sin etiquetar, con el mismo objetivo
 - También es posible argumentar que es supervisado
 - El modelo aprende la distribución de una clase de datos y luego utiliza esta información para realizar la clasificación
 - En este sentido, la clasificación one-class se asemeja más al aprendizaje supervisado que al no supervisado.
 - En general, se considera un híbrido entre ambos



DETECCIÓN DE ANOMALÍAS

- One-class classification
 - One-class SVM
 - Isolation Forest



DETECCIÓN DE ANOMALÍAS

- One-class classification
 - One-class SVM
 - En one-class SVM, el objetivo es encontrar un hiperplano que maximice el margen y que tenga la mayor cantidad de datos de la clase positiva dentro de él
 - Los que no están dentro se consideran anomalías o fuera de la clase
 - Maximizar ancho del margen
 - Distancia entre la región de decisión y la zona donde están los patrones

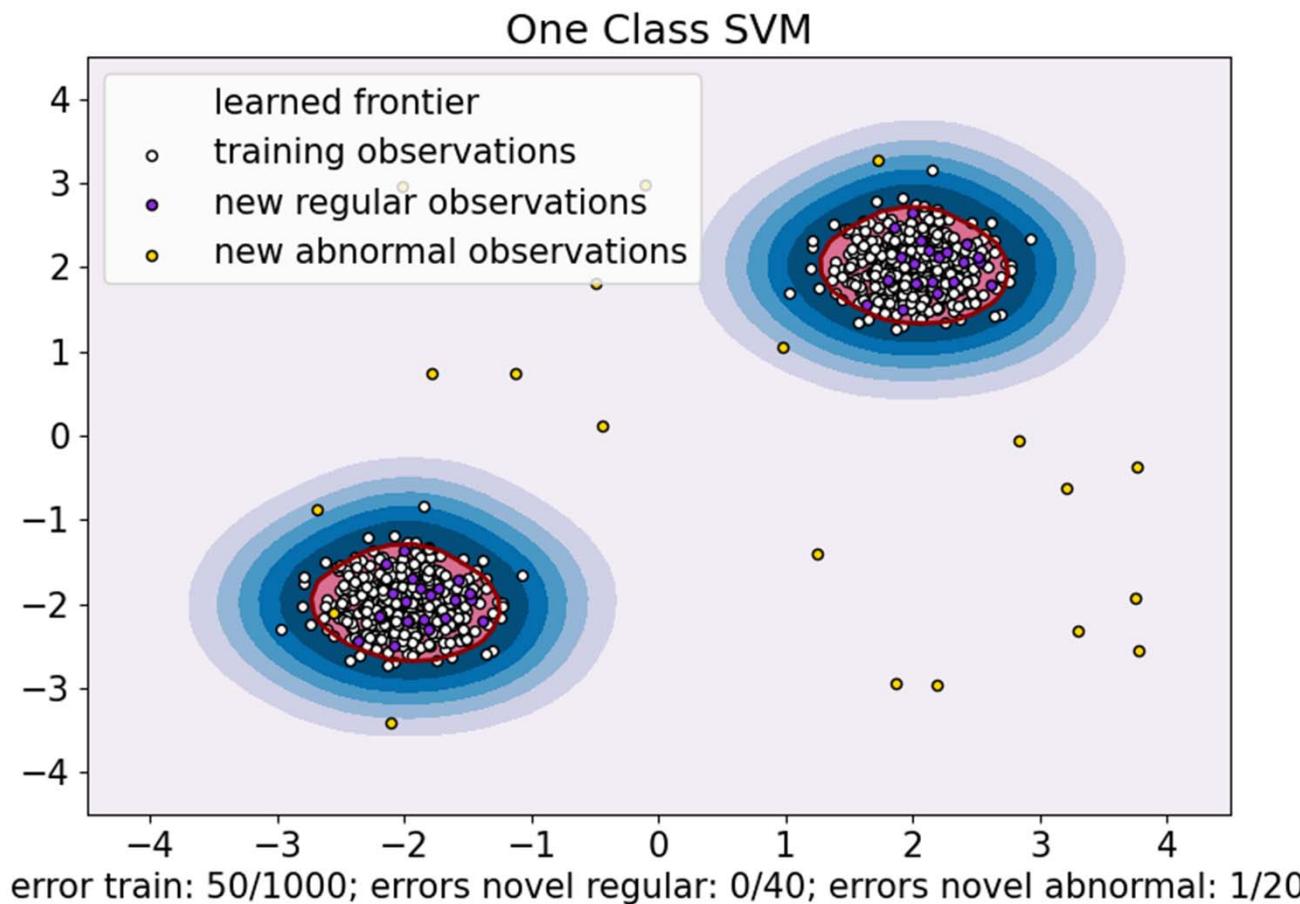


DETECCIÓN DE ANOMALÍAS

- One-class classification
 - One-class SVM
 - 2 formulaciones distintas, según si la región de decisión buscada es:
 - Hiperplano
 - Hiperesfera
 - Minimizar función cuadrática de tal forma que todos los puntos se encuentren dentro de la región de decisión (hiperplano o hiperesfera)
 - Se permite errores en esta clasificación

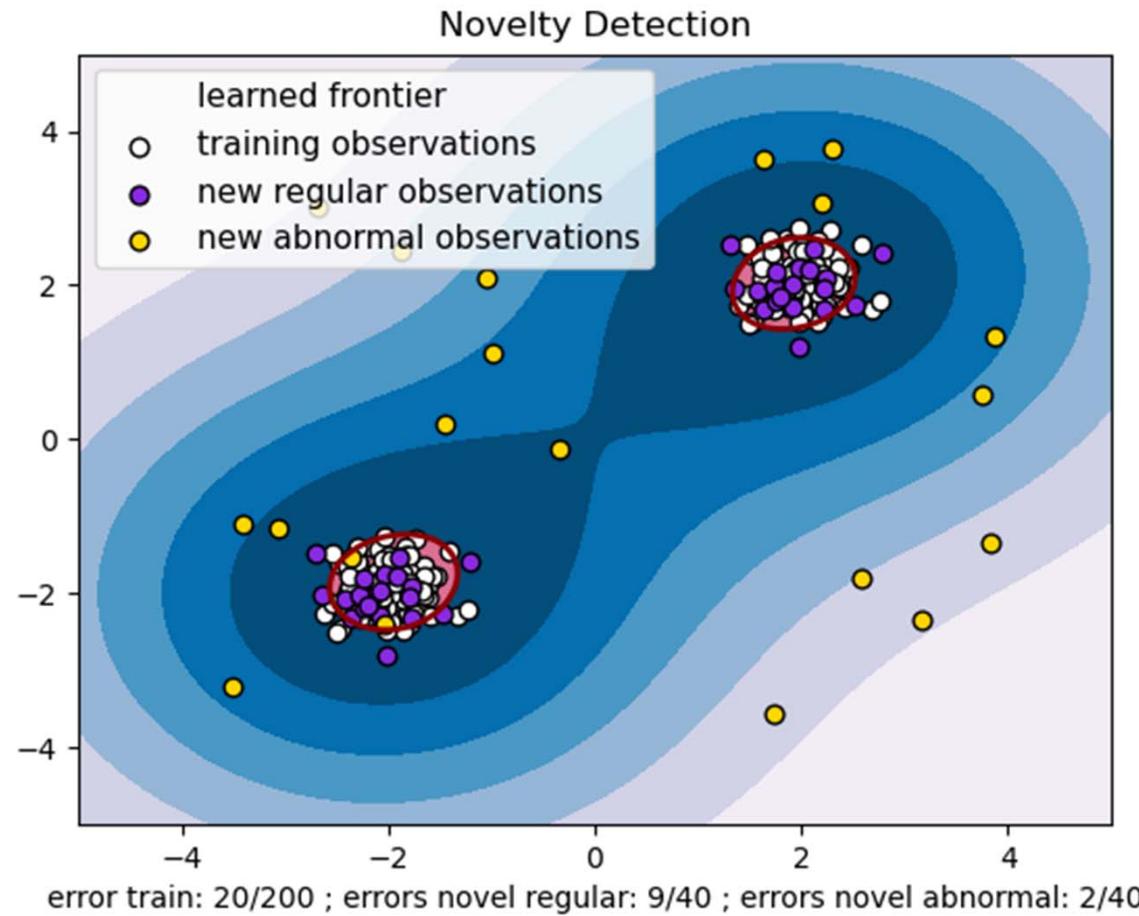
DETECCIÓN DE ANOMALÍAS

- One-class classification
 - One-class SVM



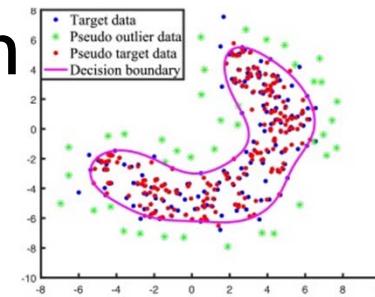
DETECCIÓN DE ANOMALÍAS

- One-class classification
 - One-class SVM

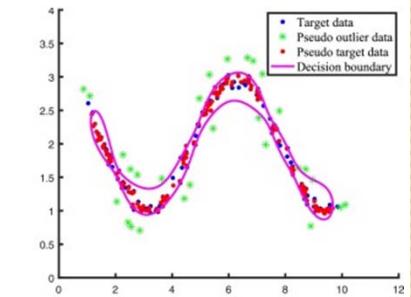


DETECCIÓN DE ANOMALÍAS

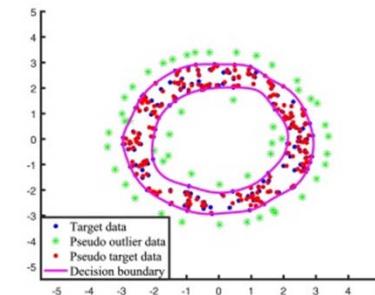
- One-class classification
 - One-class SVM
 - Ejemplo: bases de datos sintéticas



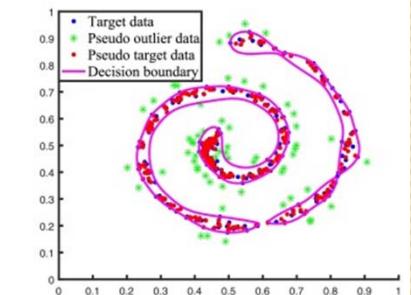
(a) Banana



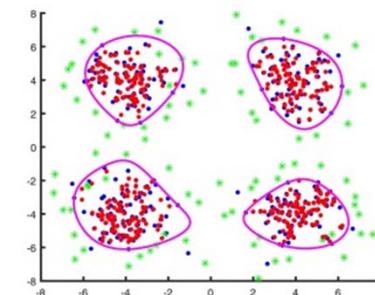
(b) Sine



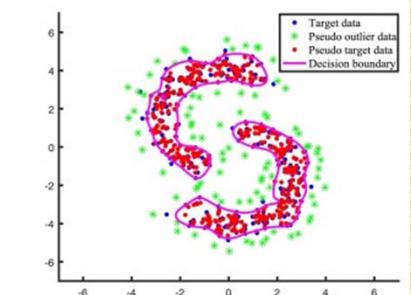
(c) Ring



(d) Spiral



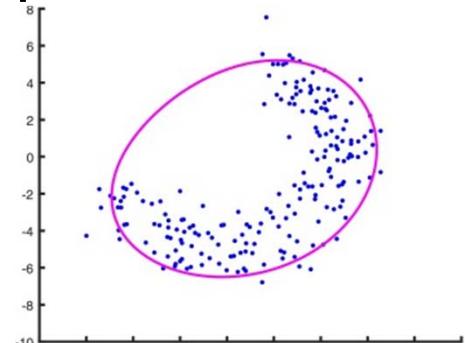
(e) Four gauss



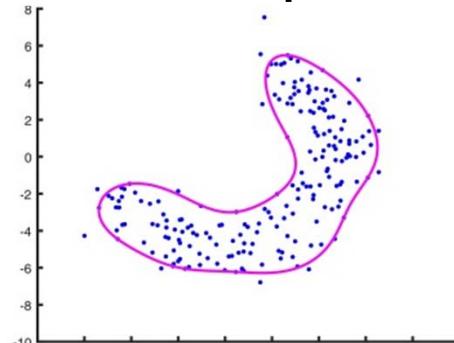
(f) Twin banana

DETECCIÓN DE ANOMALÍAS

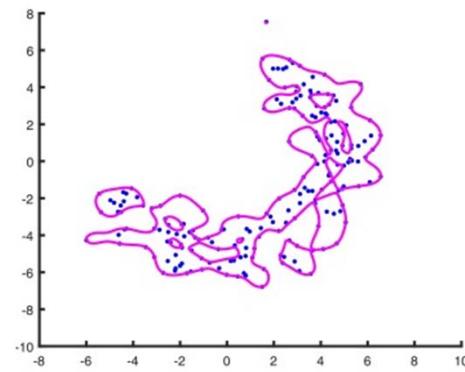
- One-class classification
 - One-class SVM
 - Ejemplo: kernel Gaussiano, distintos parámetros



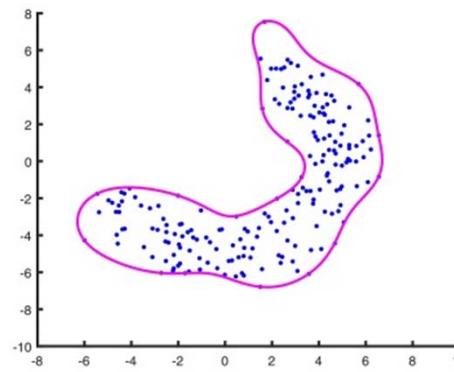
(a) $\nu = 0.1, \sigma = 10$



(b) $\nu = 0.1, \sigma = \sqrt{10}$



(c) $\nu = 0.1, \sigma = 1$



(d) $\nu = 0.01, \sigma = \sqrt{10}$

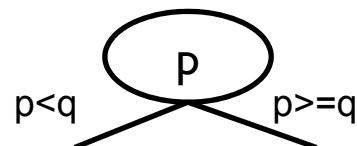


DETECCIÓN DE ANOMALÍAS

- One-class classification
 - Isolation Forest
 - Este algoritmo asume que los datos anómalos están más aislados que los datos normales en el espacio de características
 - La idea básica es que los puntos anómalos son más fáciles de identificar y aislar porque se encuentran a una mayor distancia del resto de los puntos de datos en el espacio de características
 - Para identificar los datos anómalos, Isolation Forest utiliza un enfoque de "aislar" a cada punto de datos, para lo cual se construye una serie de particiones recursivas del espacio

DETECCIÓN DE ANOMALÍAS

- One-class classification
 - Isolation Forest
 - Cada partición recursiva puede representarse mediante una estructura de árbol denominada Árbol de Aislamiento (*Isolation tree, iTree*)
 - Parecidos a árboles de decisión aleatorios
 - Cada nodo interno contiene un atributo continuo que se divide en 2 intervalos, y las ramas son esos dos distintos intervalos
 - Ejemplo: atributo p





DETECCIÓN DE ANOMALÍAS

- One-class classification
 - Isolation Forest
 - El árbol se crea de forma aleatoria
 - Para dividir un nodo, se selecciona aleatoriamente un atributo y un valor de partición entre los valores mínimo y máximo que toman las muestras de ese nodo
 - Los nodos se dividen hasta que contienen una sola instancia, o bien hasta que todas las instancias tienen los mismos valores en los atributos
 - Se podría indicar también como condición de parada una profundidad máxima



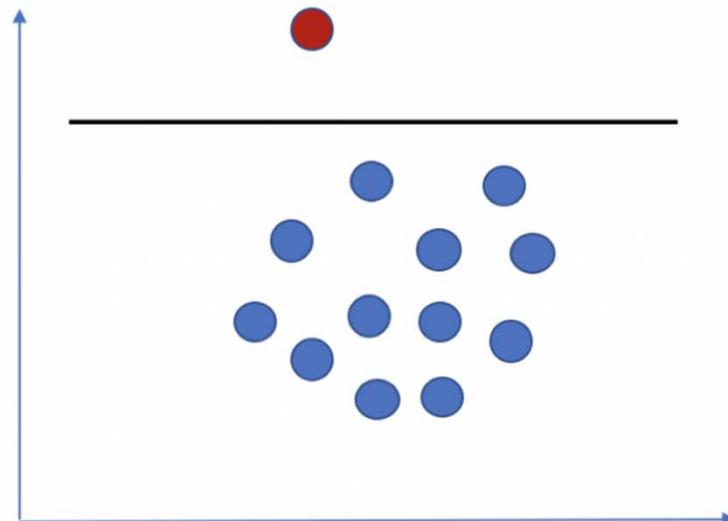
DETECCIÓN DE ANOMALÍAS

- One-class classification
 - Isolation Forest
 - Cuando el iTree está desarrollado, cada punto está aislado en una de las hojas
 - El número de particiones para aislar un punto será la longitud del camino para llegar a la hoja desde la raíz
 - Medida del aislamiento
 - La partición aleatoria produce caminos notablemente más cortos para las anomalías
 - Para aislar los puntos anómalos sólo se necesitan unas pocas condiciones
 - Para aislar los puntos normales se necesitan más condiciones
 - Por tanto, los puntos anómalos son aquellos con menor longitud de camino

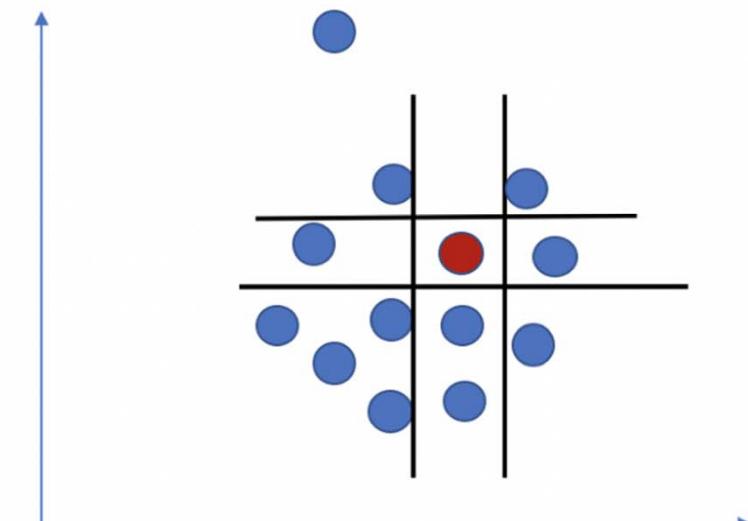
DETECCIÓN DE ANOMALÍAS

- One-class classification
 - Isolation Forest
 - Ejemplos:

Isolating an anomalous point



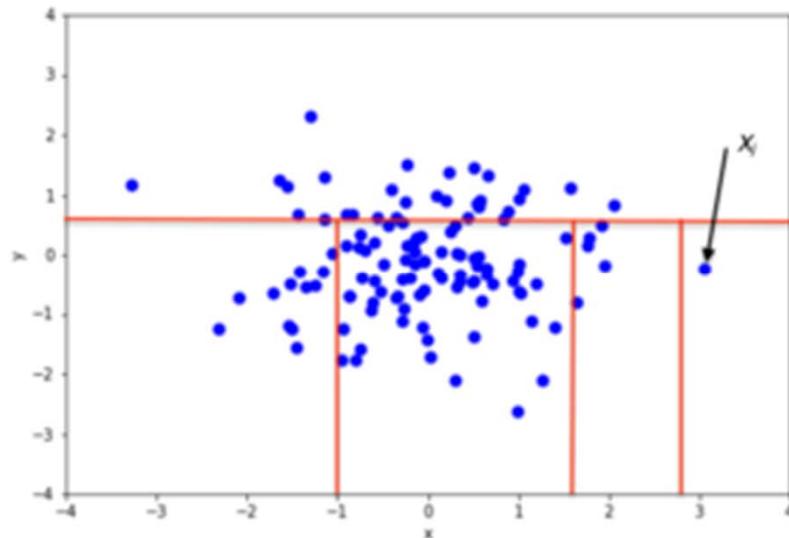
Isolating a normal point



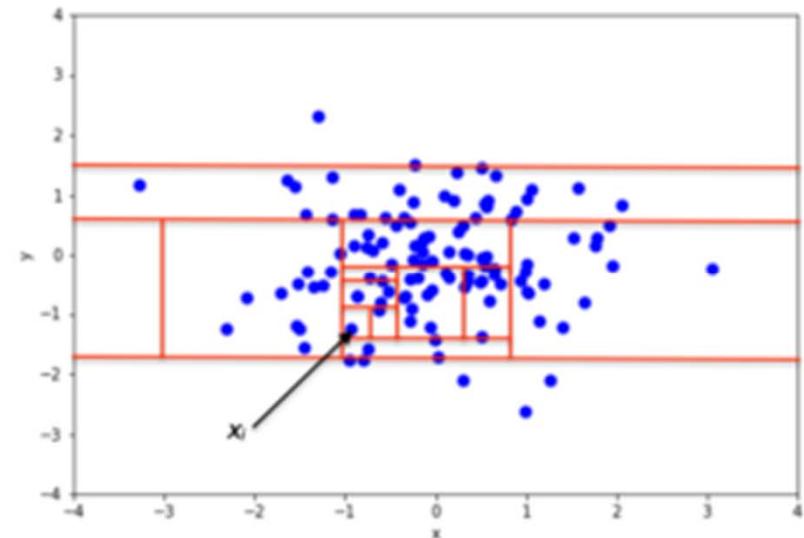
DETECCIÓN DE ANOMALÍAS

- One-class classification
 - Isolation Forest
 - Ejemplos:

Aislar un punto anómalo
(menos divisiones)



Aislar un punto no anómalo
(más divisiones)



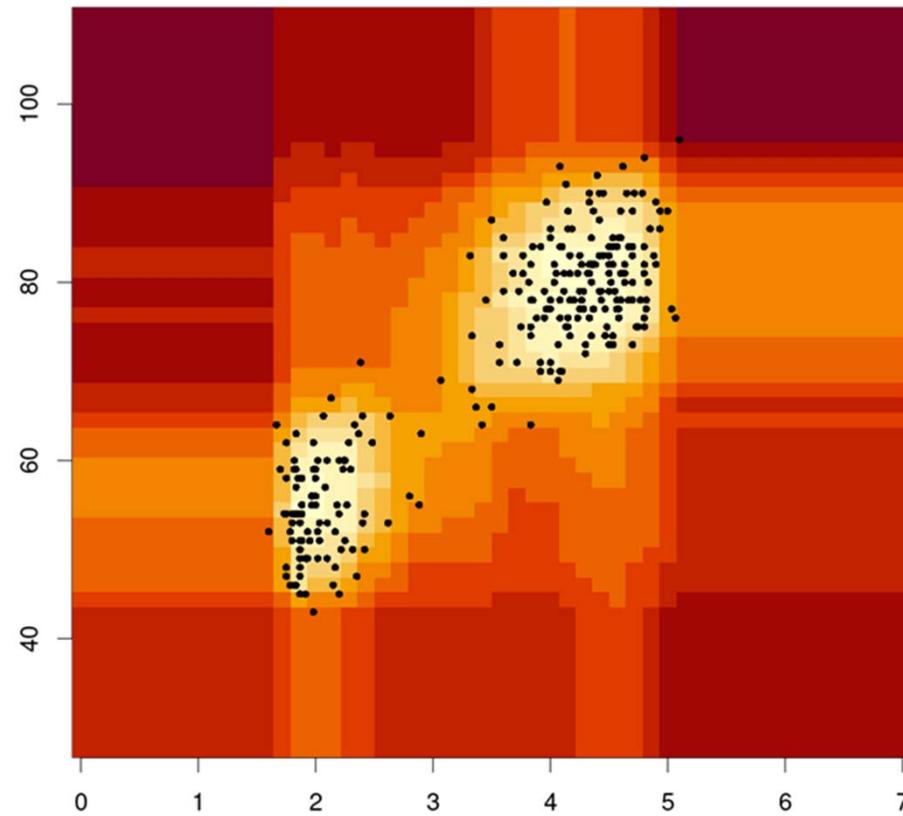


DETECCIÓN DE ANOMALÍAS

- One-class classification
 - Isolation Forest
 - Para evitar problemas debidos a la aleatoriedad del algoritmo de árboles, el proceso se repite varias veces y se calcula y normaliza la longitud media del camino.
 - Se observa que la longitud media del camino converge tras unas pocas iteraciones
 - Esta longitud del camino, promediada en un bosque de árboles aleatorios, es una medida de la normalidad
 - Por lo tanto, cuando un bosque de árboles aleatorios produce colectivamente trayectorias más cortas para determinadas muestras, es muy probable que se trate de anomalías

DETECCIÓN DE ANOMALÍAS

- One-class classification
 - Isolation Forest
 - Ejemplo: Puntuaciones normalizadas de anomalía





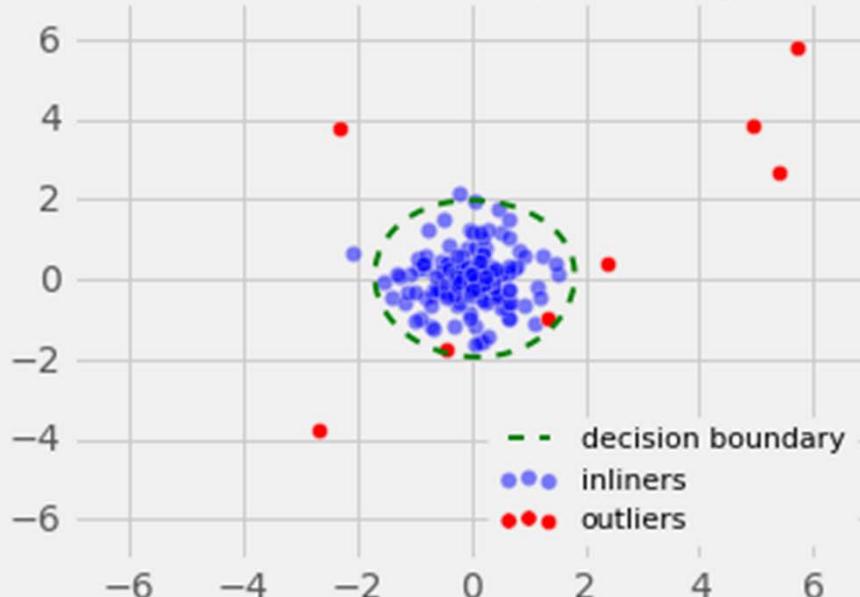
DETECCIÓN DE ANOMALÍAS

- One-class classification
 - Isolation Forest
 - En resumen, Isolation Forest es un algoritmo de detección de anomalías eficiente que funciona dividiendo recursivamente los puntos de datos en subgrupos y asignando una puntuación de anomalía basada en la profundidad a la que se encuentra el punto dentro del bosque.
 - Es un algoritmo eficiente y fácil de implementar que puede trabajar con grandes conjuntos de datos y manejar características categóricas y numéricas.

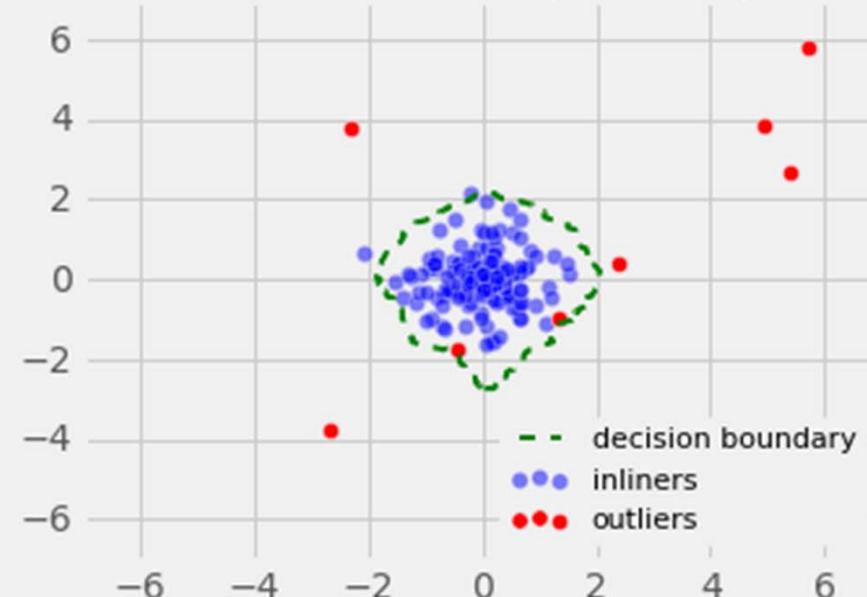
DETECCIÓN DE ANOMALÍAS

- One-class classification
 - One-class SVM
 - Isolation Forest

One-Class SVM (errors: 9)



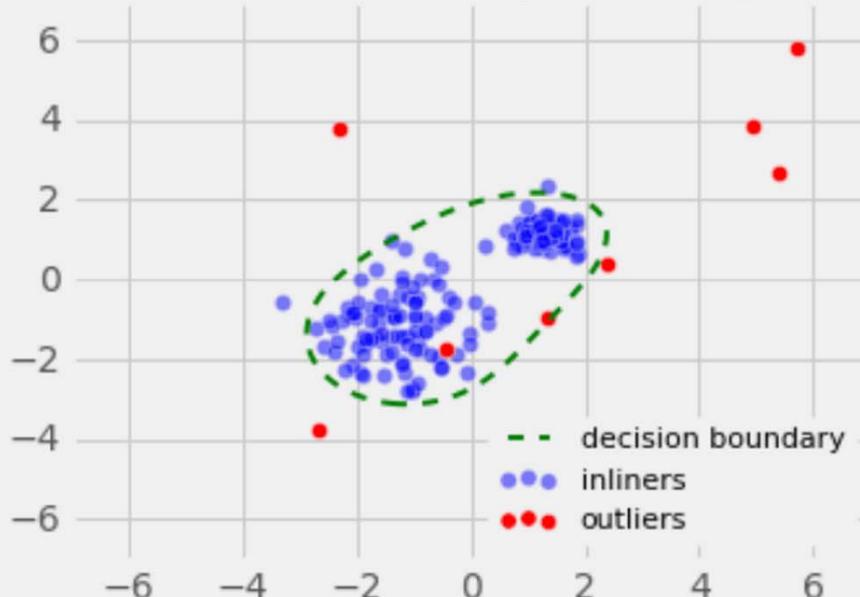
Isolation Forest (errors: 4)



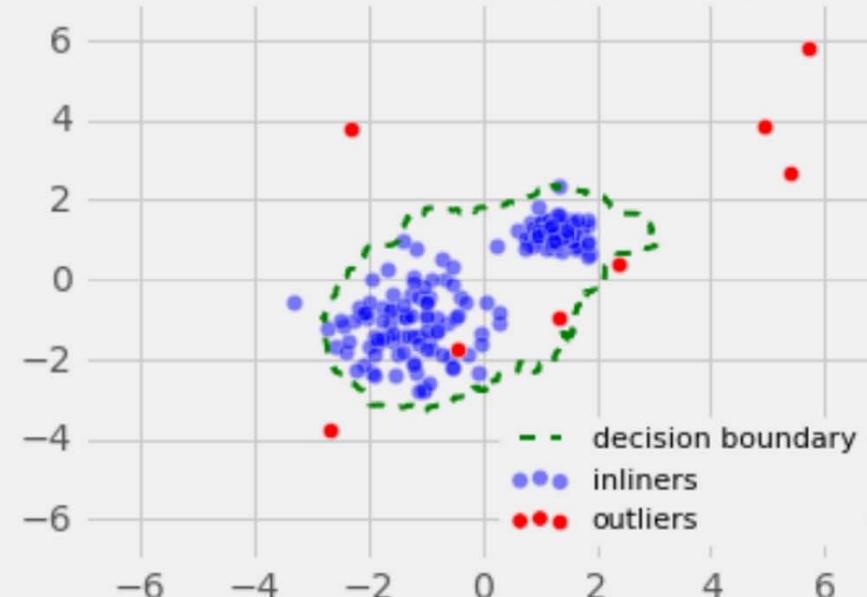
DETECCIÓN DE ANOMALÍAS

- One-class classification
 - One-class SVM
 - Isolation Forest

One-Class SVM (errors: 14)



Isolation Forest (errors: 4)





APRENDIZAJE NO SUPERVISADO

- Clustering
 - k-means
 - Clústering jerárquico
 - DBSCAN
 - GMM
- Reducción de dimensionalidad
- Asociación de reglas
 - Apriori
- Detección de anomalías
 - k-NN
 - One-class classification
 - One-class SVM
 - Isolation Forest
- **Redes de Neuronas Artificiales no supervisadas**
 - Mapas autoorganizativos
 - Autoencoders



RR.NN.AA. NO SUPERVISADAS

- Principales modelos:
 - Autoencoders
 - Redes neuronales autorganizadas (SOM)
 - Redes neuronales recurrentes generativas (RNN-G)
 - Generative Adversarial Networks (GANs)
 - Variational Autoencoders (VAEs)
 - Redes neuronales profundas de clustering (DDC)



RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Un SOM (*Self-Organizing Maps*) es “un mapa de sombras” que permite visualizar y agrupar datos multidimensionales en una representación bidimensional
 - Mapea patrones de entrada en una topología bidimensional
 - Se usa para tareas como clustering, detección de anomalías o reducción de la dimensionalidad

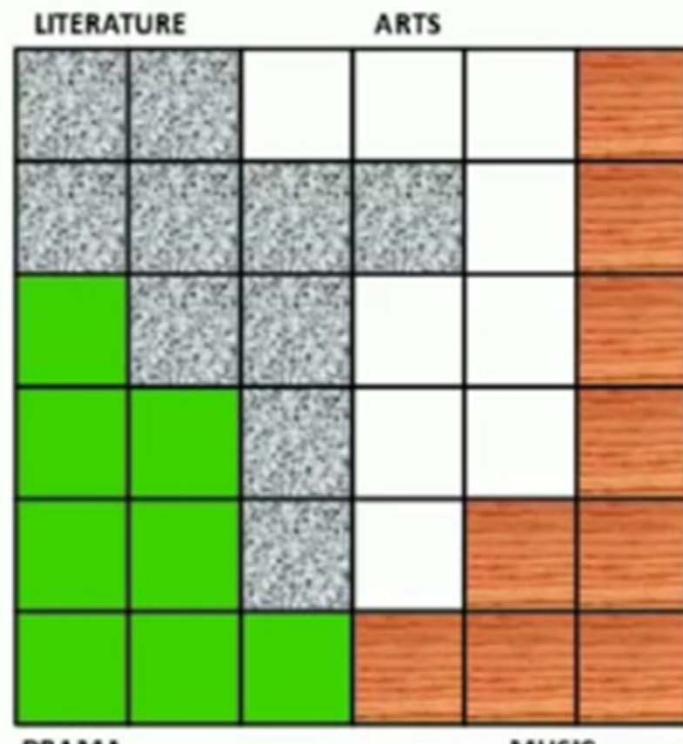


RR.NN.AA. NO SUPERVISADAS

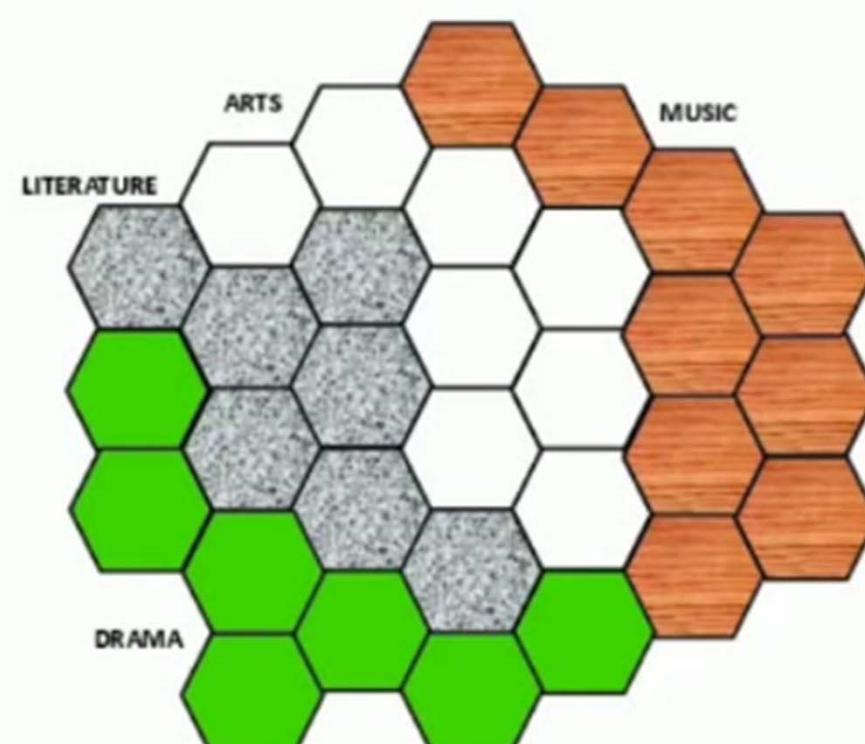
- Redes autoorganizadas:
 - La topología de un SOM es una organización en forma de malla de los nodos (neuronas) del mapa
 - Un SOM sólo tiene una capa
 - La capa de neuronas constituye un mapa bidimensional en forma de rejilla rectangular o hexagonal
 - Cada nodo del mapa (neurona) contiene un vector de pesos, que es la posición del nodo en el espacio de entradas
 - Cada nodo representa un vector de características
 - Cada neurona está situada en una posición dentro de la rejilla
 - Rejilla rectangular o hexagonal

RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:



(a) Rectangular lattice



(b) Hexagonal lattice



RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Por tanto, cada neurona proyecta datos del espacio de entradas en un punto en un espacio de menor dimensionalidad
 - Los pesos de la neurona la sitúan en el espacio de entradas
 - La posición en la rejilla indica el punto en el espacio bidimensional
 - Cada nodo representa un vector en un espacio de características reducido y se asigna a una ubicación específica en el mapa

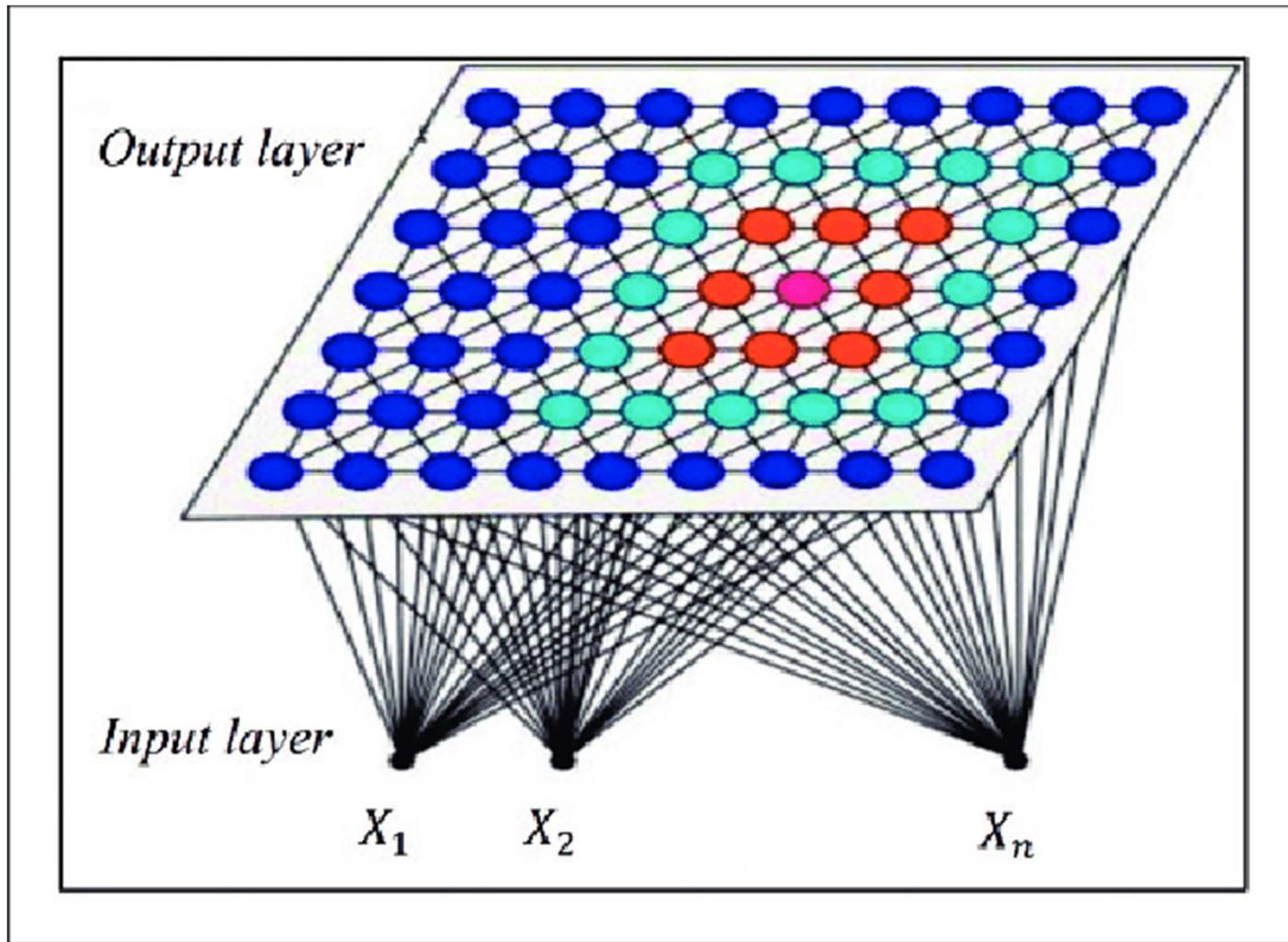


RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - La disposición de los nodos en el mapa es fija, y su organización refleja las relaciones topológicas y semánticas entre los datos de entrada
 - Los nodos cercanos en el mapa representan características similares en los datos de entrada
 - La topología permite una visualización compacta y fácil de entender de la estructura de los datos.
 - Una vez entrenado el SOM, ante un nuevo dato, se halla el nodo con la menor distancia del dato a su vector de pesos.

RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:





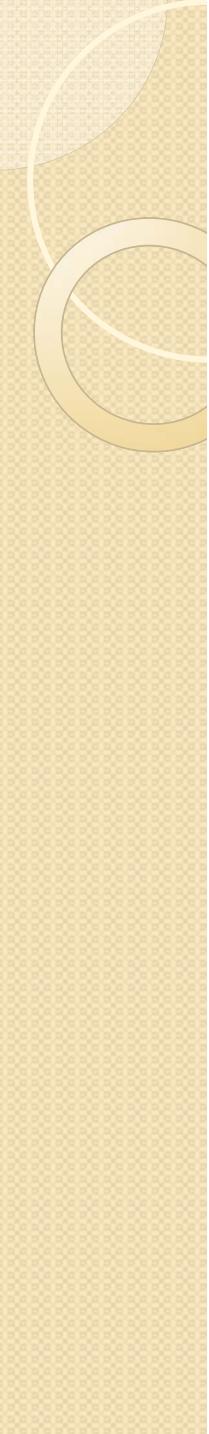
RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - La idea detrás de las SOM es que los puntos de datos similares se asignarán a neuronas cercanas en el mapa
 - Esto permite agrupar los datos en grupos y visualizarlos de manera más sencilla
 - La forma en que las neuronas se organizan y los puntos de datos se asignan a ellas depende de la similitud de las características de los puntos de datos y las neuronas.



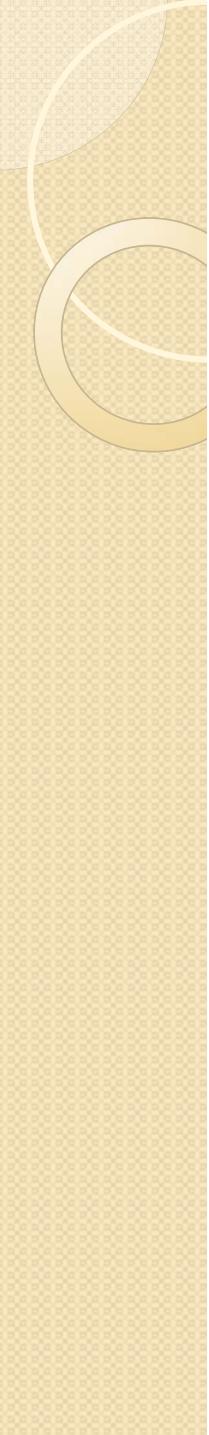
RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Un SOM es entrenado mediante **aprendizaje competitivo** en lugar de aprendizaje por corrección del error (como el algoritmo de *backpropagation*)
 - El entrenamiento consiste en mover estos vectores de pesos hacia los datos de entrada, reduciendo la distancia sin romper la topología del mapa



RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Entrenamiento:
 - I. Inicialización
 - Se inicializa la posición (pesos) de cada neurona en el mapa



RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Entrenamiento:
 1. Inicialización
 2. Repetir para cada patrón de entrada:
 - I. Se calcula la distancia de este patrón a todos los nodos
 - Neurona más cercana: *best matching unit* (BMU)

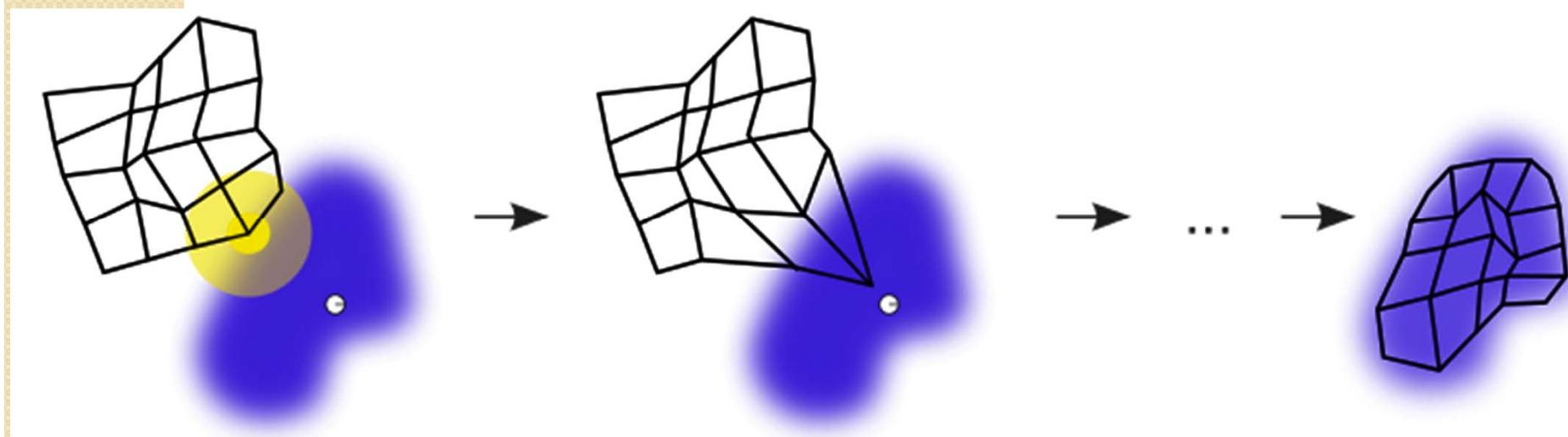


RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Entrenamiento:
 1. Inicialización
 2. Repetir para cada patrón de entrada:
 1. Se calcula la distancia de este patrón a todos los nodos
 2. Se actualizan los pesos de la neurona ganadora y las cercanas en la rejilla del SOM para acercarse al patrón de entrada.
 - La magnitud de este cambio disminuye con el tiempo y con la distancia al nodo ganador
 - Neuronas cercanas: definir cercanía
 - Sólo las adyacentes
 - Funciones gausianas, mexican-hat
 - Se disminuye el tamaño del vecindario a medida que el entrenamiento continúa

RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Entrenamiento:
 1. Inicialización
 2. Repetir para cada patrón de entrada:
 1. Se calcula la distancia de este patrón a todos los nodos
 2. Se actualizan los pesos de la neurona ganadora y las cercanas en la rejilla del SOM para acercarse al patrón de entrada.



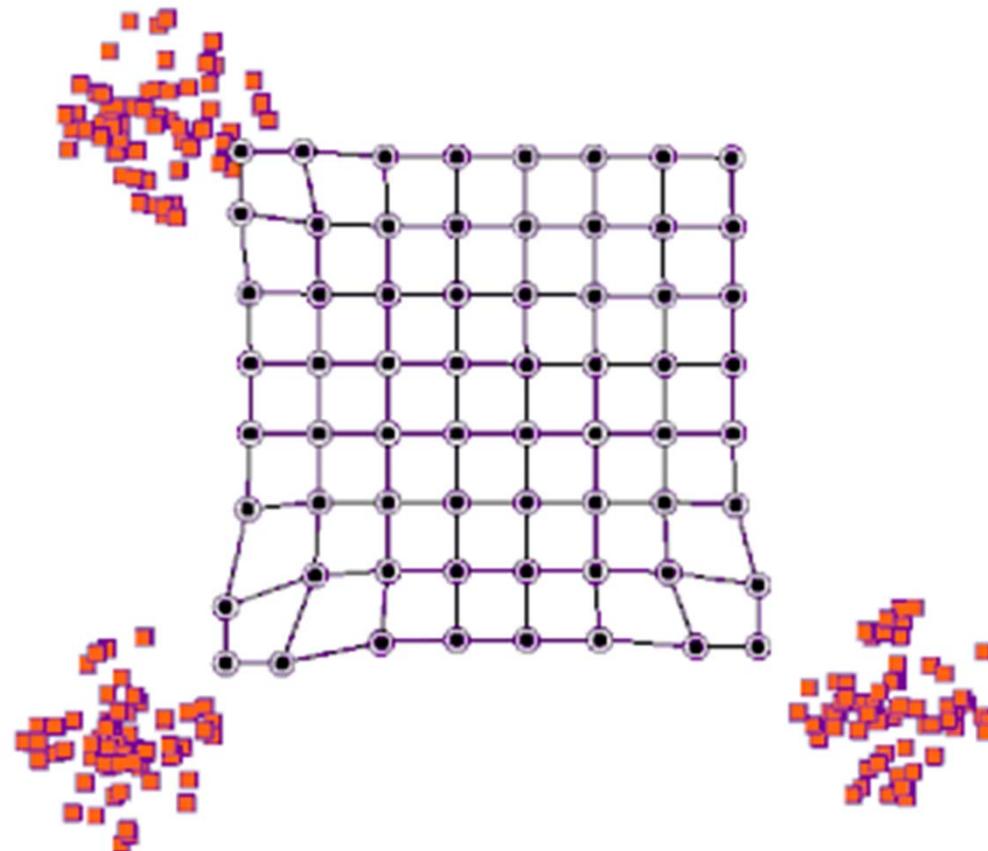


RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Entrenamiento:
 1. Inicialización
 2. Repetir para cada patrón de entrada:
 1. Se calcula la distancia de este patrón a todos los nodos
 2. Se actualizan los pesos de la neurona ganadora y las cercanas en la rejilla del SOM para acercarse al patrón de entrada.
 3. Repetir el paso 2 hasta que se alcance un criterio de parada

RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:





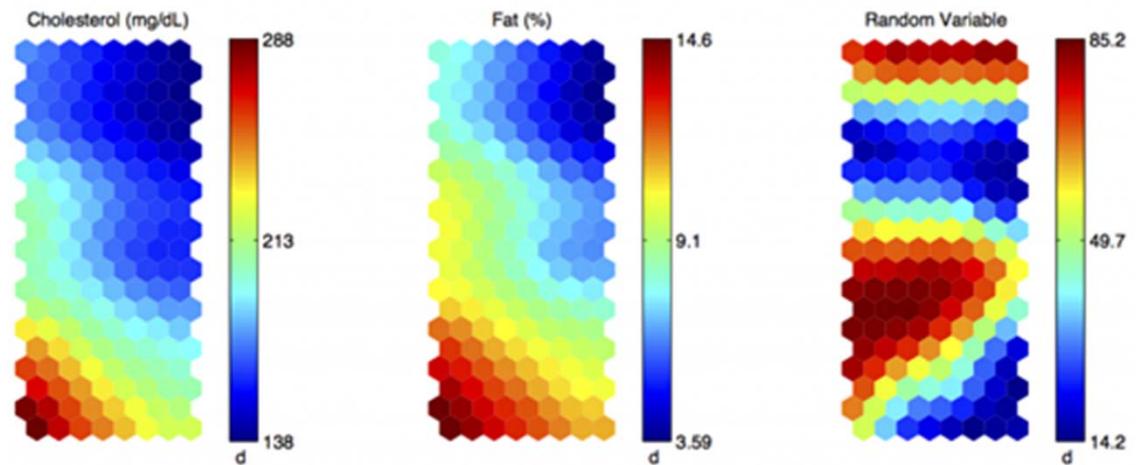
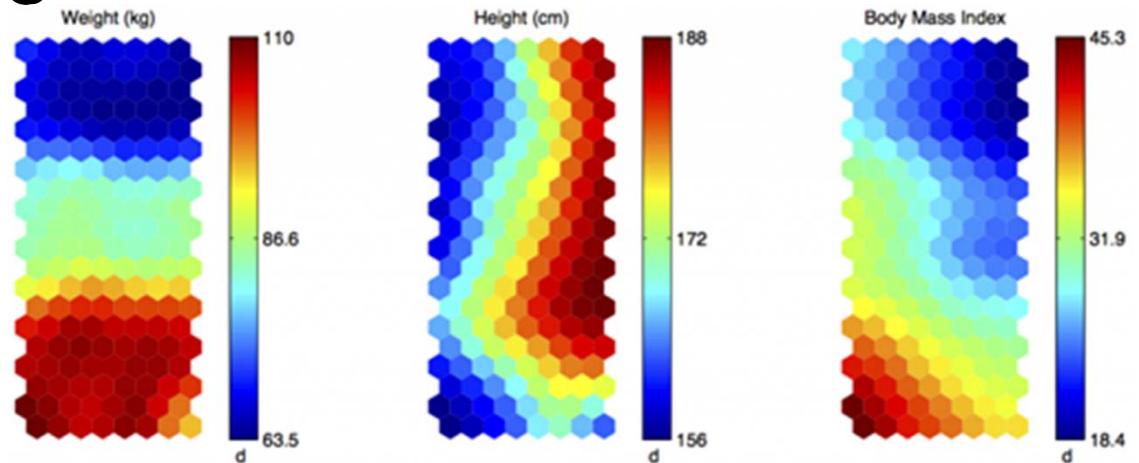
RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Ejemplo:
 - 1000 individuos, 5 variables corporales:
 - Peso, altura, índice de masa corporal, índice de colesterol, porcentaje de grasa corporal
 - Adicionalmente, se añadió un sexto atributo aleatorio
 - Valores entre 0 y 100

RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:

- Ejemplo:
 - Valores de pesos para cada atributo:



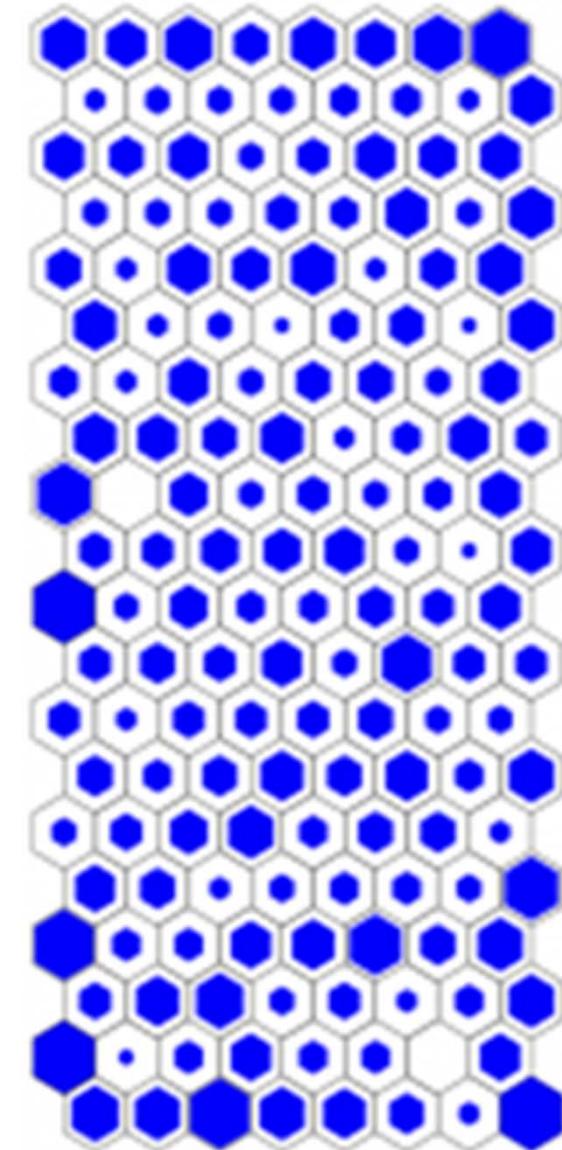


RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Ejemplo:
 - Interpretación de resultados:
 - Neurona en la esquina inferior izquierda:
 - Individuos con un peso elevado y estatura baja
 - Todos ellos tienen un índice de masa corporal muy elevado y un nivel de colesterol y un porcentaje de grasa altos
 - Por tanto, estos individuos deben considerarse de alto riesgo
 - Neurona en la esquina superior derecha:
 - Individuos con bajo peso y elevada estatura
 - Estos valores conducen a un índice de masa corporal bajo y a un porcentaje de grasa bajo, con niveles de colesterol bajos
 - No se aprecia ninguna relación entre la variable aleatoria y cualquier otra

RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Ejemplo:
 - Número de patrones en cada neurona:





RR.NN.AA. NO SUPERVISADAS

- **Redes autoorganizadas:**
 - Ejemplo: Pobreza mundial por países
 - 39 estadísticas tomadas del Banco Mundial sobre los países en 1992
 - Estos atributos describen diversos factores de la calidad de vida, como el estado de salud, la nutrición, los servicios educativos, etc.
 - Difícil comprender la interacción de estos indicadores
 - El complejo efecto conjunto de estos factores puede visualizarse disponiendo los países mediante un mapa autoorganizado.
 - Los países que tenían valores similares de los indicadores encontraban un lugar cercano en el mapa

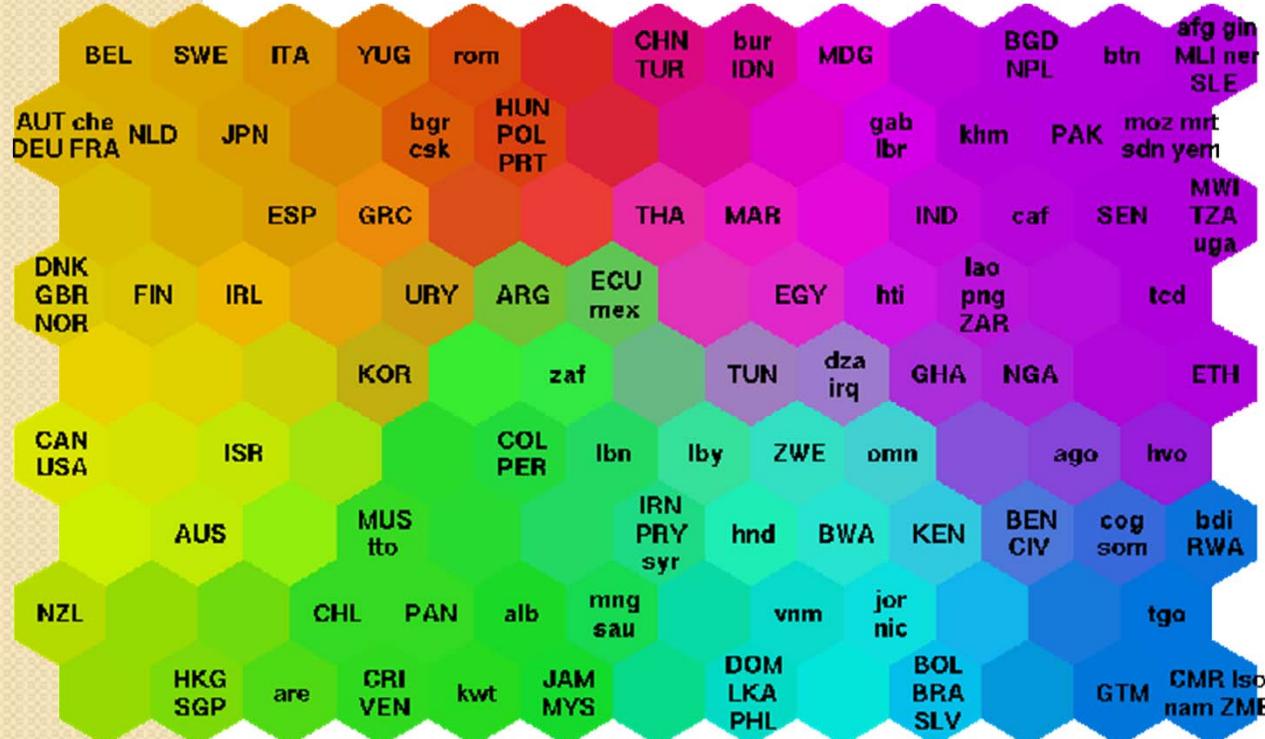


RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Ejemplo: Pobreza mundial por países
 - Las diferentes agrupaciones del mapa se codificaron automáticamente con diferentes colores, de modo que los colores cambian suavemente en la visualización del mapa
 - Como resultado de este proceso, a cada país se le asignó automáticamente un color que describe su tipo de pobreza en relación con otros países
 - De este modo, las estructuras de la pobreza en el mundo pueden visualizarse de forma sencilla: cada país del mapa geográfico se ha coloreado según su tipo de pobreza

RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Ejemplo: Pobreza mundial por países

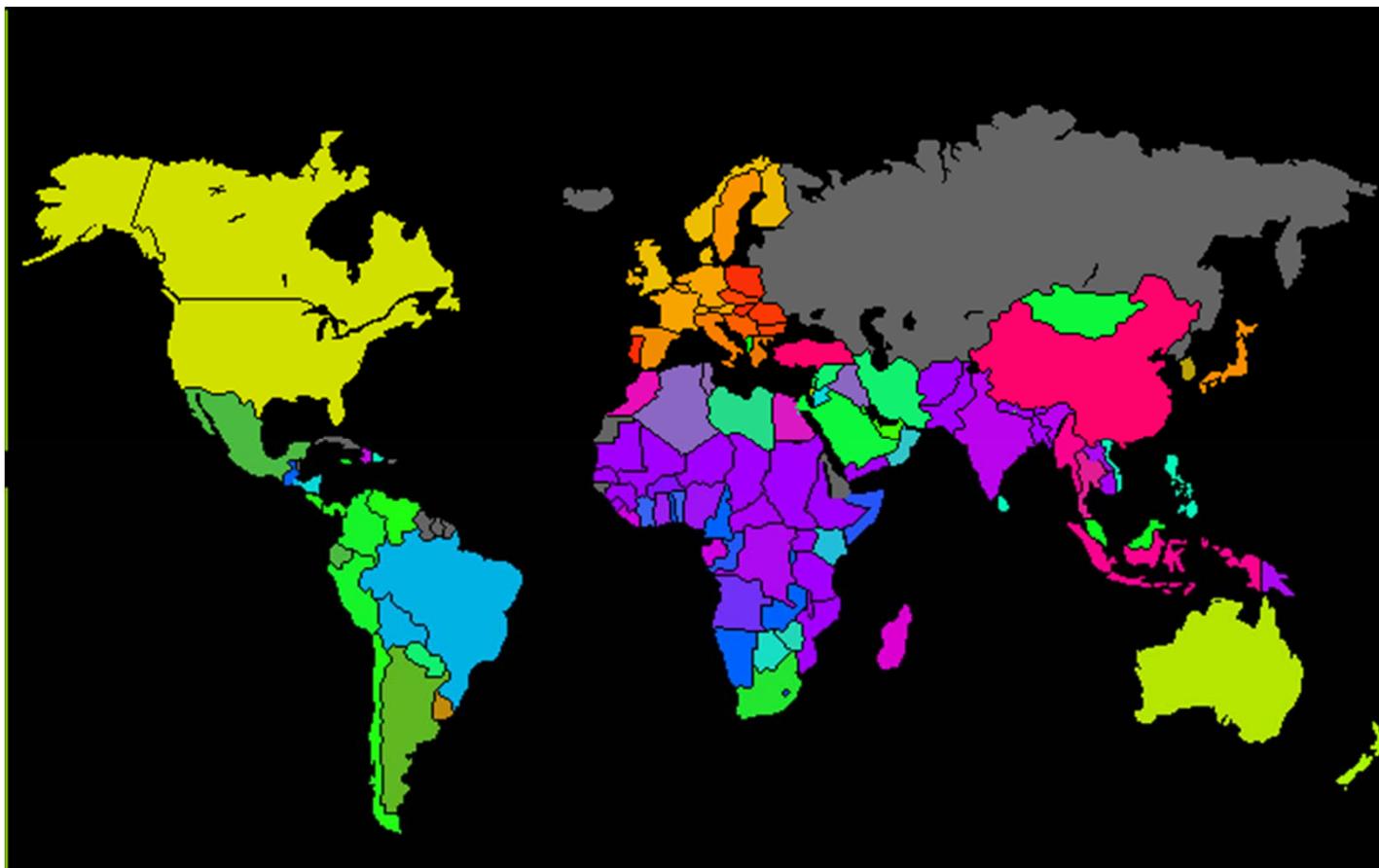


The Country Names

AFG	Afghanistan	GTM	Guatemala	NZL	New Zealand
AGO	Angola	HKG	Hong Kong	OAN	Taiwan, China
ALB	Albania	IND	Indonesia	OMN	Oman
ARE	United Arab Emirates	HTI	Haiti	PAK	Pakistan
ARG	Argentina	HUN	Hungary	PAN	Panama
AUS	Australia	IVO	Burkina Faso	PER	Peru
AUT	Austria	IDN	Indonesia	PHL	Philippines
BDI	Burundi	IDN	India	PNG	Papua New Guinea
BEL	Belgium	IRL	Ireland	POL	Poland
BEN	Benin	IRN	Iran, Islamic Rep.	PRT	Portugal
BGD	Bangladesh	IRQ	Iraq	PRY	Panama
BGR	Bulgaria	ISR	Israel	ROM	Romania
BOL	Bolivia	ITA	Italy	RWA	Rwanda
BRA	Brazil	JAM	Jamaica	SAU	Saudi Arabia
BTN	Bhutan	JOR	Jordan	SDN	Sudan
BLR	Myanmar	JPN	Japan	SEN	Senegal
BWA	Botswana	KEN	Kenya	SGP	Singapore
CAP	Central African Rep.	KHM	Cambodia	SLE	Sierra Leone
CAN	Canada	KOR	Korea, Rep.	SLV	El Salvador
CHE	Switzerland	KWT	Kuwait	SOM	Somalia
CHL	Chile	LAO	Lao PDR	SVF	Sweden
CHN	China	LBN	Lebanon	SYR	Syrian Arab Rep.
CIV	Côte d'Ivoire	LBR	Liberia	TCD	Chad
CMR	Cameroon	LBY	Liberia	TGO	Togo
COG	Congo	LKA	Sri Lanka	THA	Thailand
COL	Colombia	LSC	Lesotho	TTO	Trinidad and Tobago
CRI	Costa Rica	MAR	Morocco	TUN	Tunisia
CSK	Czechoslovakia	MDG	Madagascar	TUR	Turkey
DEU	Germany	MEX	Mexico	TZA	Tanzania
DNK	Denmark	MLI	Mali	UGA	Uganda
DOM	Dominican Rep.	MNG	Mongolia	URU	Uruguay
DZA	Algeria	MOZ	Mozambique	USA	United States
ECU	Ecuador	MRT	Mauritania	VEN	Venezuela
EGY	Egypt, Arab Rep.	MUS	Mauritius	VNM	Viet Nam
ESP	Spain	MVR	Maldives	YEM	Yemen, Rep.
ETH	Ethiopia	MYS	Malaysia	VUG	Vogadlovia
FIN	Finland	NAM	Namibia	ZAF	South Africa
FRA	France	NER	Niger	ZAR	Zaire
GAB	Gabon	NGA	Nigeria	ZMB	Zambia
GBR	United Kingdom	NIC	Nicaragua	ZWE	Zimbabwe
GHA	Ghana	NLD	Netherlands		
GIN	Guinea	NOR	Norway		
GRC	Greece	NPL	Nepal		

RR.NN.AA. NO SUPERVISADAS

- Redes autoorganizadas:
 - Ejemplo: Pobreza mundial por países





APRENDIZAJE NO SUPERVISADO

- Clustering
 - k-means
 - Clústering jerárquico
 - DBSCAN
 - GMM
- Reducción de dimensionalidad
- Asociación de reglas
 - Apriori
- Detección de anomalías
 - k-NN
 - One-class classification
 - One-class SVM
 - Isolation Forest
- Redes de Neuronas Artificiales no supervisadas
 - Mapas autoorganizativos
 - **Autoencoders**



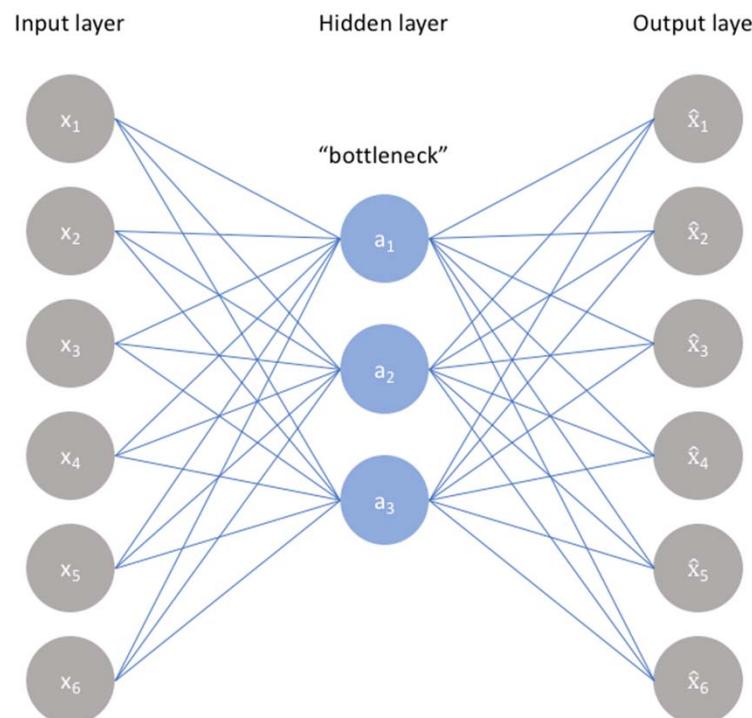
RR.NN.AA. NO SUPERVISADAS

- Autoencoders:

- Los autoencoders son redes neuronales artificiales, entrenadas de manera no supervisada
 - No supervisada en el sentido en que las entradas no están etiquetadas, ni hay salida deseada
 - En realidad, se utilizan como salidas deseadas las propias entradas y se utiliza un entrenamiento supervisado
 - *Backpropagation*
 - Capa interna de menos dimensiones que las de entrada y salida
 - Codificador
 - La capa de salida realiza la reconstrucción de los datos
 - Decodificador

RR.NN.AA. NO SUPERVISADAS

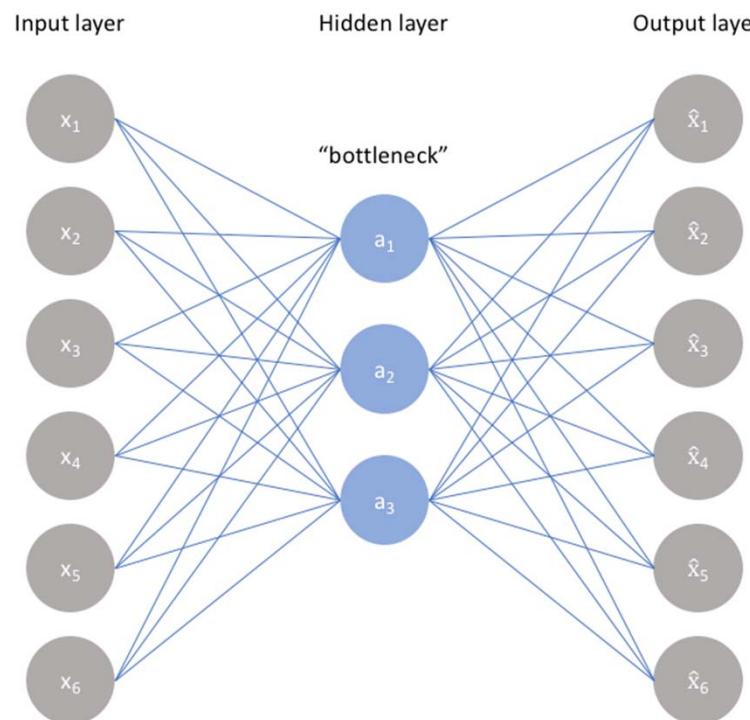
- Autoencoders:



- Si la reconstrucción de los datos se realiza, las salidas de la capa oculta serán representaciones codificadas de los datos de entrada
 - De menor dimensionalidad

RR.NN.AA. NO SUPERVISADAS

- Autoencoders:



- Salidas de la capa oculta: dimensionalidad menor
 - Reducción de dimensionalidad
 - **Espacio latente**



RR.NN.AA. NO SUPERVISADAS

- Autoencoders:
 - Objetivo del entrenamiento:
 - Minimizar el error de reconstrucción entre la entrada original y la salida del decodificador utilizando una función de pérdida como el error cuadrático medio (MSE)
 - El autoencoder aprende a capturar las características importantes de los datos de entrada en la representación latente y reconstruye la entrada original a partir de ella

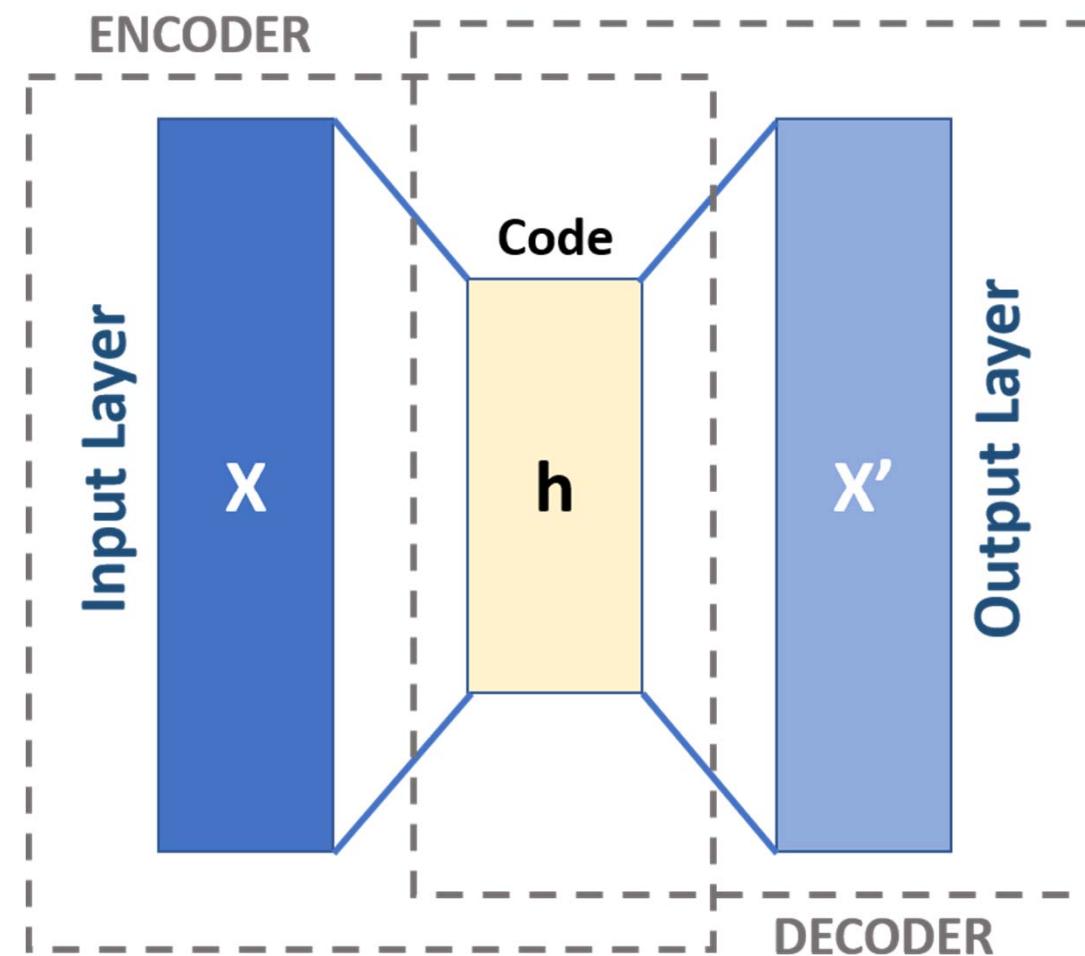


RR.NN.AA. NO SUPERVISADAS

- Autoencoders:
 - Por lo tanto, los autoencoders consisten en una arquitectura codificador-decodificador que aprende una representación latente de baja dimensión de los datos de entrada a través de la reducción de dimensionalidad no supervisada
 - Codificador: función que transforma la entrada original en una representación latente de menor dimensión
 - Decodificador: función que reconstruye la entrada original a partir de la representación latente

RR.NN.AA. NO SUPERVISADAS

- Autoencoders:





RR.NN.AA. NO SUPERVISADAS

- Autoencoders:
 - Arquitectura:
 - Al menos tres capas: entrada, oculta salida
 - Se pueden incluir capas adicionales en la parte central para aumentar la capacidad de representación
 - Deep Learning:
 - Apilar capas de autoencoders entrenados por turnos
 - *Stacked Auto-Encoders*
 - Usar distintos tipos de capas
 - Por ejemplo, convolucionales
 - Usar distintas funciones de coste
 - *Denoising autoencoder / Sparse autoencoder / Variational autoencoder*

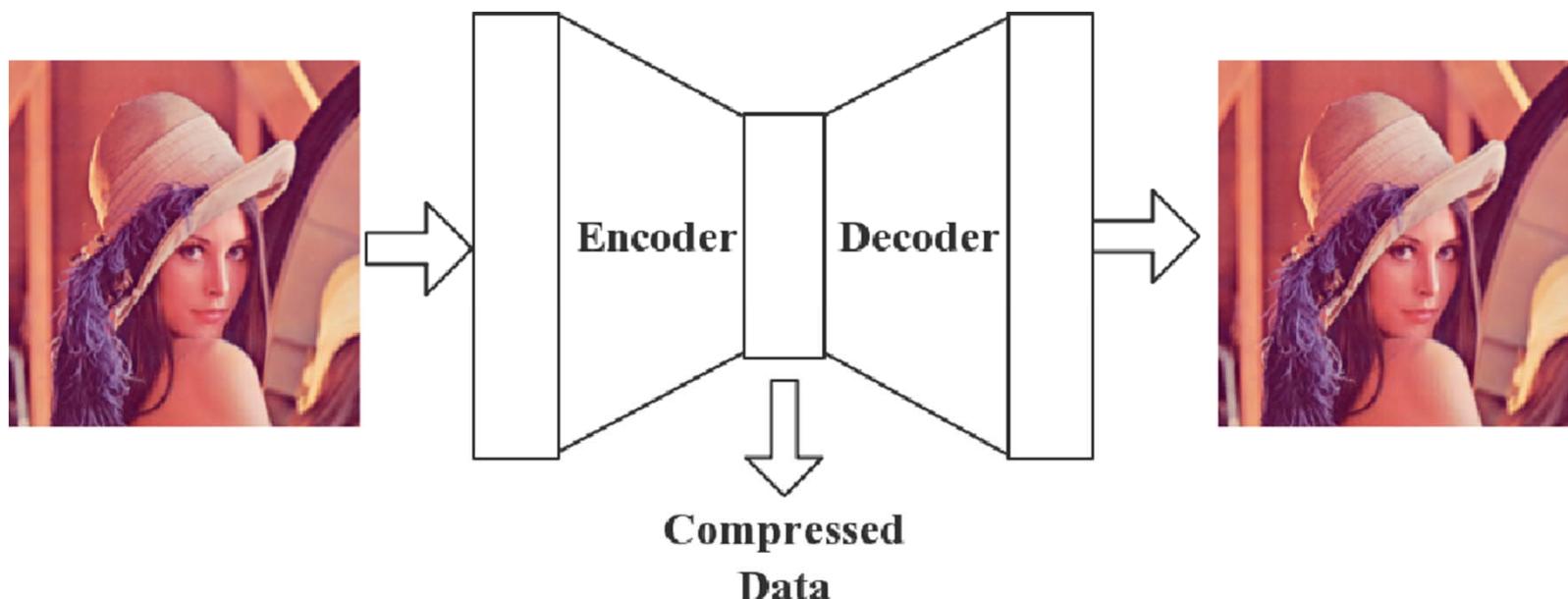


RR.NN.AA. NO SUPERVISADAS

- Autoencoders:
 - Aplicaciones (1/7):
 - Reducción de dimensionalidad
 - Reducir la dimensionalidad de los datos de entrada a una representación más compacta.
 - Compresión de datos
 - Permite reducir el tamaño de los datos de entrada sin perder información importante
 - Escenarios en los que resulta útil:
 - Almacenamiento de datos
 - Transmisión de datos
 - Privacidad y anonimización

RR.NN.AA. NO SUPERVISADAS

- Autoencoders:
 - Aplicaciones (2/7):
 - Reconstrucción de imágenes
 - Reconstruir imágenes a partir de una versión reducida de las mismas





RR.NN.AA. NO SUPERVISADAS

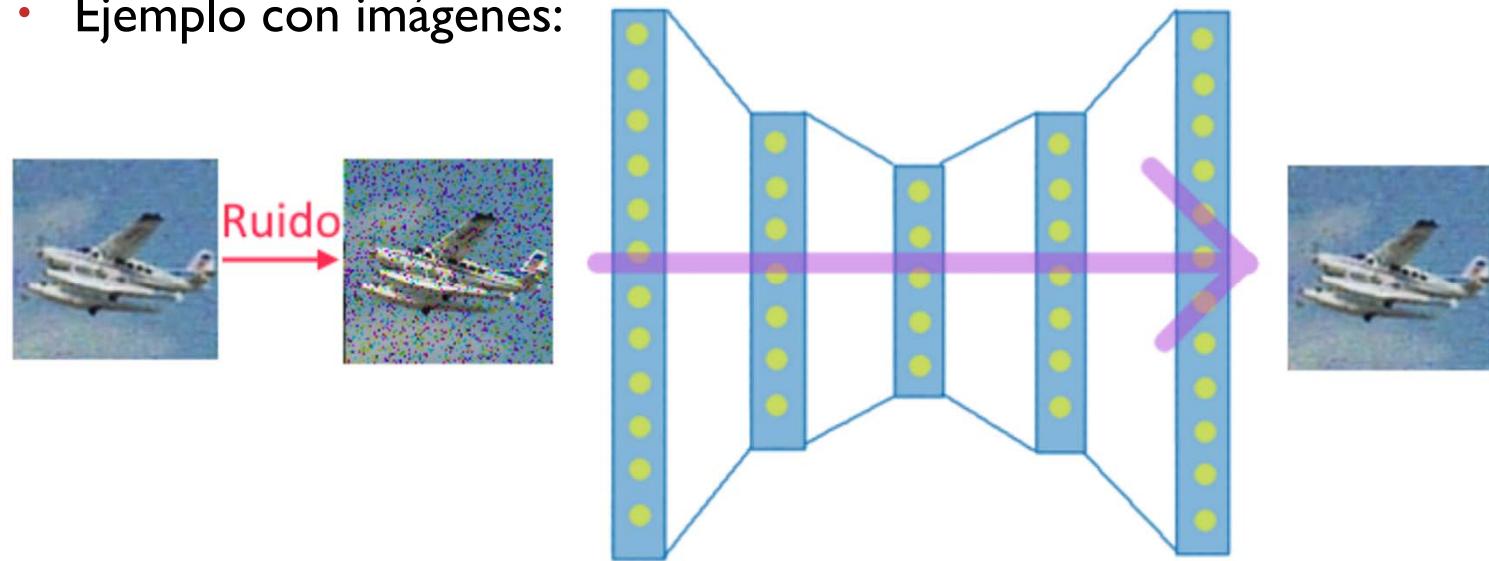
- Autoencoders:
 - Aplicaciones (3/7):
 - Generación de datos e imágenes
 - Generar nuevos datos a partir de la representación aprendida, lo que puede ser útil para la generación de datos de entrenamiento adicionales.
 - Los autoencoders pueden ser utilizados para generar nuevas imágenes a partir de un conjunto de datos de entrenamiento.
 - Para ello, se entrena el autoencoder con imágenes de entrada y se utiliza la salida como una imagen generada
 - Aplicar una imagen al codificador, aplicar algún desvío a la imagen codificada, y aplicar el resultado al decodificador

RR.NN.AA. NO SUPERVISADAS

- Autoencoders:

- Aplicaciones (4/7):

- Reducción de ruido
 - Se entrena el autoencoder con datos con ruido en la entrada, y los datos originales en la salida
 - Así, se puede aprender a reconstruir los datos sin ruido
 - Ejemplo con imágenes:



RR.NN.AA. NO SUPERVISADAS

- Autoencoders:

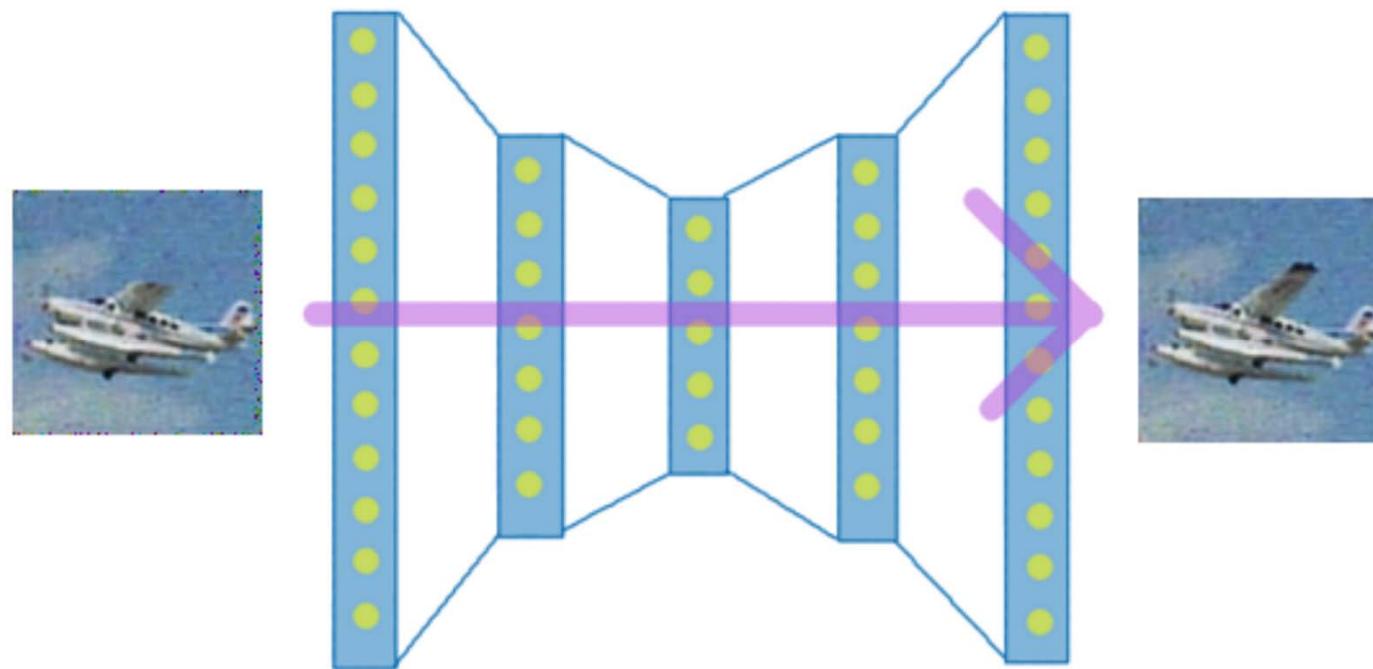
- Aplicaciones (5/7):

- Reducción de ruido
 - Se entrena el autoencoder con datos con ruido en la entrada, y los datos originales en la salida
 - Así, se puede aprender a reconstruir los datos sin ruido
 - Ejemplo con imágenes:



RR.NN.AA. NO SUPERVISADAS

- Autoencoders:
 - Aplicaciones (6/7):
 - Imputación de datos
 - Estimar datos que no han podido obtenerse
 - Similar a la reducción de ruido





RR.NN.AA. NO SUPERVISADAS

- Autoencoders:
 - Aplicaciones (7/7):
 - Clustering
 - Los clústers se definen por la similitud en los datos codificados
 - Detección de anomalías
 - Se entrena un autoencoder con los datos normales y luego se aplica a nuevos datos
 - Se mide la distancia entre la entrada y la salida reconstruida
 - Si la distancia es demasiado grande, se considera que el dato es anómalo
 - La idea es que los datos normales sean fácilmente reconstruidos por el autoencoder, mientras que los datos anómalos sean difíciles de reconstruir