
Introducción a la Minería de Datos

J. Mario García Valdez

Índice general

Otras Técnicas de Clasificación	2
Clasificadores basados en reglas	2
k vecinos más próximos	3
Naïve Bayes	4
Redes Neuronales Artificiales	7
Support Vector Machines	7
Métodos Ensamble	7
Análisis de Asociaciones	7
Definición formal del problema	8
Generación de conjuntos de artículos frecuentes	9
El principio Apriori	9
Generación de reglas	11
Representación compacta de conjuntos de artículos frecuentes	11
El algoritmo FP-Growth	12
Análisis de Clusters	14
Clustering basado en prototipos	18
Clustering basado en densidades	18
Clustering basado en grafos	18
k -medias	18
Incorrecta selección de las posiciones iniciales de los centroides	19
¿Como evaluamos la calidad de los clusters generados?	20
¿Como evitamos el error provocado por la inicialización?	21
Agrupamiento Jerárquico	21
DBSCAN	22
Bibliografía	22

Otras Técnicas de Clasificación

Clasificadores basados en reglas

En la sección anterior, el modelo utilizado para clasificar se representaba como un árbol de decisión. Los clasificadores que veremos a continuación representan al modelo como conjunto de reglas IF-THEN. Estas reglas son similares a las condiciones de los nodos de los árboles de decisión, una regla para clasificar hoteles podría ser:

Regla 1: **IF** *alberca = sí* **THEN** *WiFiGratis = sí*.

Las reglas tienen dos partes:

1. El antecedente o precondición **IF** en la cual hay expresiones condicionales sobre los atributos, de manera opcional utilizando operadores lógicos, por ejemplo, *alberca = sí* OR *estrellas < 5*.
2. El consecuente **THEN** donde se expresa la predicción de la clase o categoría a la que pertenece el objeto.

Ejemplo

Recordemos el ejemplo de los hoteles visto anteriormente:

	id	hotel	estrellas	alberca	WiFi Gratis
1	1	Mariots	2	Sí	Sí
2	2	Díaz Inn	2	No	Sí
3	3	Mandarina	5	Sí	No
4	4	Le Hotel	3	Sí	No
5	5	Halton	4	No	Sí
6	6	Tromp	4	Sí	No

Un modelo basado en reglas para clasificar a los hoteles puede ser:

R1: IF estrellas = 5 OR estrellas = 3 THEN WiFiGratis = Sí R2: IF alberca = No THEN WiFiGratis = Sí R3: IF alberca = Sí THEN WiFiGratis = No

Los algoritmos clasificadores basados en reglas extraen las reglas de los datos mediante:

1. La extracción de reglas a partir de árboles de decisión. Las rutas de la raíz a las hojas son las precondiciones las hojas son los consecuentes (Han, Pei, and Kamber 2011).
2. Se generan las reglas a partir de los datos, utilizando un algoritmo de cobertura, por ejemplo RIPPER (Cohen 1995) o CN2 (Clark and Niblett 1989):

k vecinos más próximos

Este algoritmo se basa en una idea sencilla: asignar la clase que tengan los objetos del conjunto de entrenamiento que más se parezcan al objeto a clasificar. Para calcular la similaridad entre dos objetos,

se puede calcular simplemente la distancia euclídea entre los vectores de características de cada objeto. El parámetro k especifica cuantos objetos (ordenados por similaridad) se van a considerar para la asignación. En el caso más sencillo se le asigna al objeto la clase mayoritaria.

El método es muy fácil de implementar. La selección del valor de k puede afectar el desempeño del algoritmo. La figura muestra un ejemplo de clasificación para distintos valores de k . Si elegimos un valor de k muy pequeño puede ser afectado por el ruido o valores atípicos, por otro lado valores muy grandes se tenderá a considerar un número mayor de datos con otras clases. El valor del voto que asigna cada objeto puede ponderarse con respecto a la distancia.

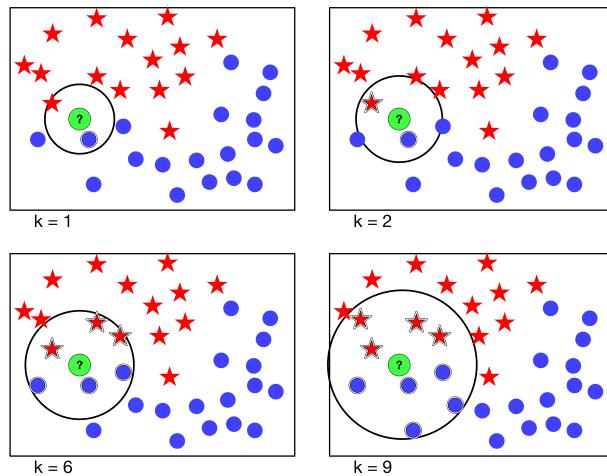


Figura 1: Efecto de la elección del número de vecinos k

Naïve Bayes

Este clasificador básico recibe el nombre de Naïve que se traduce al español como ingenuo. La razón de esto es que considera que los valores de los atributos de un objeto son variables independientes. Es fácil ver que esta consideración no siempre es real. Por ejemplo, para los atributos de un hotel: «número de estrellas» y «alberca» podemos imaginar que un hotel de más de 4 estrellas es muy probable que cuente con una o más. También es muy probable que un hotel de dos estrellas no cuente con una. Entonces, un clasificador Naïve Bayes considera que para una clase C dado un objeto con los atributos $\{A_1, A_2, \dots, A_n\}$ podemos calcular la probabilidad condicional:

$$P\{C|A_1, A_2, \dots, A_n\}$$

Esto significa tratar de estimar las probabilidades a partir del conjunto de datos de entrenamiento. Para clasificar un registro se debe de calcular la probabilidad condicional para cada clase C_j y elegir la mayor:

$$P\{C_j|A_1, A_2, \dots, A_n\} = \frac{P\{A_1, A_2, \dots, A_n|C_j\} \cdot P(C_j)}{P\{A_1, A_2, \dots, A_n\}}$$

Como solo estamos interesados en elegir la mejor opción basta con calcular:

$$P\{A_1, A_2, \dots, A_n | C_j\} \cdot P(C_j)$$

Si consideramos (ingenuamente) los atributos como variables independientes, el primer término se simplifica:

$$P\{A_1, A_2, \dots, A_n | C_j\} = P\{A_1 | C_j\} \cdot P\{A_2 | C_j\} \cdots P\{A_n | C_j\}$$

Ejemplo

Como ejemplo vamos a utilizar un fragmento del conjunto de datos de enfermedades agudas y utilizaremos un clasificador Naïve Bayes para determinar si un paciente tienen una inflamación aguda de la vejiga.

Datos:

		dolor lumbar	necesidad constante	dolor al orinar	comezón en la uretra	infección
temp	nausea					
35.5	no	yes	no	no	no	no
36.0	no	yes	no	no	no	no
36.8	no	no	yes	yes	yes	yes
37.0	no	no	yes	yes	yes	yes
37.4	no	no	yes	no	no	yes
37.1	no	no	yes	no	no	yes
37.6	no	no	yes	yes	no	yes
37.8	no	no	yes	yes	yes	yes
38.0	no	yes	yes	no	yes	no
39.0	no	yes	yes	no	yes	no
40.4	yes	yes	no	yes	no	no
40.8	no	yes	yes	no	yes	no
41.5	yes	yes	no	yes	no	no
41.5	no	yes	yes	no	yes	no

Paciente:

	dolor	necesidad	dolor al	comezón en la		
temp	nausea	lumbar	constante	orinar	uretra	infección
36.6	no	no	yes	yes	yes	yes

Como primer paso vamos a calcular $P(C_j)$:

$$P(C_j) = \frac{N}{N_c}$$

$$P('yes') = 6/14 = 0.429$$

$$P('no') = 8/14 = 0.571$$

Para calcular los atributos discretos:

$$P(A_i|C_k) = \frac{|A_i k|}{N_C k}$$

Donde,

- $|A_i k|$ es el número de registros con el atributo $A_i k$ pertenecientes a la clase C_k
- $N_C k$ es el número de registros pertenecientes a la clase C_k

En la siguiente tabla tenemos las probabilidades de los atributos discretos:

	dolor	necesidad		comezón en la	
clase	nausea	lumbar	constante	dolor al orinar	uretra
yes	0.14	0.57	0.71	0.43	0.5
no	0.86	0.43	0.29	0.57	0.5

Para el caso de datos continuos hay varias opciones @ [tan2007introduction]:

- Se discretiza el rango creando particiones y asignando un solo valor a cada una.
- Se hace una división de dos vías a partir de un valor.
- Se estima la densidad de la probabilidad.
 - Se asume que los valores siguen una distribución normal.
 - Se utilizan los datos para calcular los parámetros de la distribución.
 - Una vez que se conoce la distribución se puede calcular la probabilidad condicional.

En este caso tenemos al atributo temperatura como atributo continuo.

clase	media	stdev	distribución normal x=36.3
yes	37.28	2.38	0.16

clase	media	stdev	distribución normal x=36.3
no	39.09	0.34	0

Incluso antes de multiplicar las probabilidades vemos que en el atributo «temperatura» la clase «no» tiene cero de probabilidad, por lo que la probabilidad de la clase «yes» será mayor.

$$P(\text{Paciente}|\text{No}) = 0.86 * 0.43 * 0.29 * 0.57 * 0.5 * 0.16 = 0.0048$$

$$P(\text{Paciente}|\text{Yes}) = 0.14 * 0.57 * 0.71 * 0.43 * 0.5 * 0 = 0$$

$$P(\text{Paciente}|\text{No})P(\text{No}) = 0.0027$$

$$P(\text{Paciente}|\text{Yes})P(\text{Yes}) = 0$$

Como $P(\text{Paciente}|\text{No})P(\text{No})$ es mayor, **la clase para el registro Paciente es «yes»**

Redes Neuronales Artificiales

Support Vector Machines

Métodos Ensamble

Análisis de Asociaciones

La tarea de la minería de datos es extraer de manera eficiente patrones útiles de grandes bases de datos. La extracción de reglas de asociación surge con el fin descubrir patrones de compra interesantes minando las transacciones de compra en tiendas al menudeo (Hilderman and Hamilton 1999). Debido a su principal aplicación también se le conoce como *Market Basket Analysis* y fue propuesta inicialmente en (Agrawal, Imielinski, and Swami 1993). Las reglas que se extraen tienen que ver con los artículos que se compran en una transacción y tienen una forma de implicación, por ejemplo:

$$\{\text{Skittles}, \text{Donas}\} \implies \{\text{DietCoke}\}$$

La regla sugiere que las personas que compran *Skittles* y *Donas* también compran *Diet Coke*. Las reglas incluyen un porcentaje de transacciones que cumplen con ella. Este conocimiento puede ser utilizado por los empresarios para la toma de decisiones. Por ejemplo, surtir más un producto, hacer promociones o ver el impacto que tendría descontinuar algún artículo.

Definición formal del problema

El problema se ha definido formalmente por Agrawal, Imielinski, and Swami (1993): Sea $I = \{i_1, i_2, \dots, i_m\}$ un conjunto de m atributos binarios a los que llamaremos *ítems*. Sea $T = \{t_1, t_2, \dots, t_m\}$ un conjunto de m transacciones al que llamaremos *database*. Cada transacción t es representada por un vector binario, donde $t[k] = 1$ si en la transacción se compró el ítem i_k en caso contrario $t[k] = 0$. Sea X un conjunto de ítems en I . Decimos que la transacción t *satisface* a X , si para todos los ítems I_k en X , se cumple que $t[k] = 1$, es decir se compraron todos los ítems X . Una *regla de asociación* se define como una implicación $X \Rightarrow I_j$. Como podemos ver originalmente se consideraba en el consecuente solo un ítem, actualmente se considera a un conjunto de ítems. Los conjuntos X e I no tienen ítems en común y a ambos se les denomina *itemsets*, conjuntos de ítems.

También se definen dos restricciones que deben cumplir las reglas, el soporte y la confianza.

- **Soporte** el soporte indica la frecuencia en la que las transacciones T incluyen a cierto ítemset.

$$\text{supp}(X) = \frac{|\{t \in T; X \subseteq t\}|}{|T|}$$

- **Confianza** indica que tan frecuentemente se cumple cierta regla $X \Rightarrow I$ en T

$$\text{conf}(X \Rightarrow I) = \text{supp}(X \cup I) / \text{supp}(I)$$

Estas restricciones tienen como objetivo el filtrar aquellas reglas que no sean interesantes o que sean poco comunes. El soporte y la confianza se proporcionan al algoritmo como un umbrales mínimos que deben cumplir las reglas que resulten de la búsqueda.

El proceso de minado de reglas puede resumirse en dos pasos:

1. Encontrar todos los conjuntos de artículos que tengan la frecuencia mínima indicada por el umbral de soporte.
2. A partir de los conjuntos de artículos encontrados en el primer paso, se generan reglas que cumplan con la restricción de confianza mínima.

Las dos tareas se detallan en las siguientes secciones.

Generación de conjuntos de artículos frecuentes

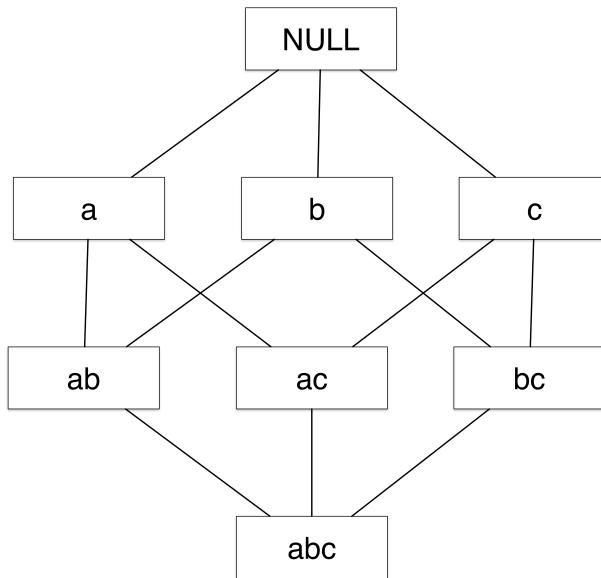


Figura 2: Rejilla de ítemset

Para enumerar todos los posibles ítemsets se emplea una rejilla. Como ejemplo en la figura se muestra la rejilla para el conjunto de ítems $I = \{a, b, c\}$. Al incrementar el número de ítems el número de ítemsets se incrementa exponencialmente. Evaluar a fuerza bruta el soporte de cada una de las transacciones en T tiene un costo computacional muy alto ya que se tiene que comparar cada ítemset candidato con todas las transacciones. La técnica *Apriori* es utilizada para reducir la complejidad computacional.

El principio Apriori

Para reducir el número de ítemsets candidatos a evaluar, se utiliza el principio Apriori :

Si un ítemset ocurre de manera frecuente, entonces todos sus subconjuntos también son frecuentes Esto se ilustra en la figura, donde suponemos que el ítemset $\{b, c\}$ es frecuente, también los serán $\{b\}$ y $\{c\}$.

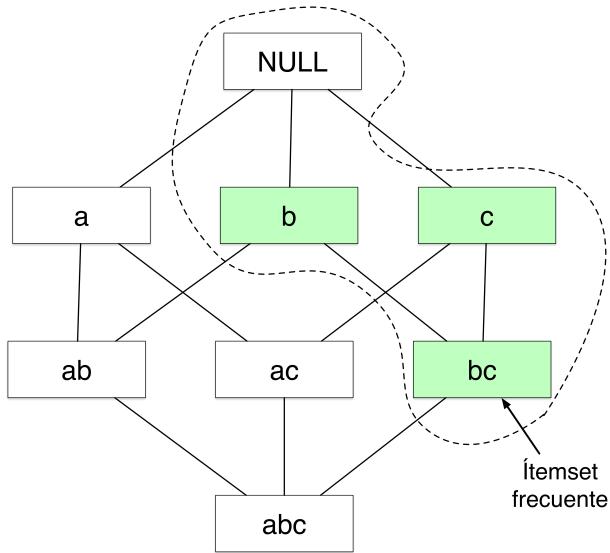


Figura 3: Rejilla de ítemset

Por otro lado, si el ítemset $\{a\}$ es infrecuente los super conjuntos también lo serán. Esto permite eliminar de la rejilla aquellos ítemsets bajo el ítemset infrecuente. Por ejemplo en la Figura, si hay evidencia de que el ítemset $\{a\}$ es infrecuente, también los serán $\{a, b\}$, $\{a, c\}$ y $\{a, b, c\}$ por lo que no hay necesidad de evaluarlos. El algoritmo de generación de ítemsets Apriori se aprovecha de este principio. Se empieza en la raíz de la rejilla evaluando el soporte de los ítemsets con un solo elemento y solo continúa evaluando los ítemsets que incluyan solo a los ítems con el soporte necesario.

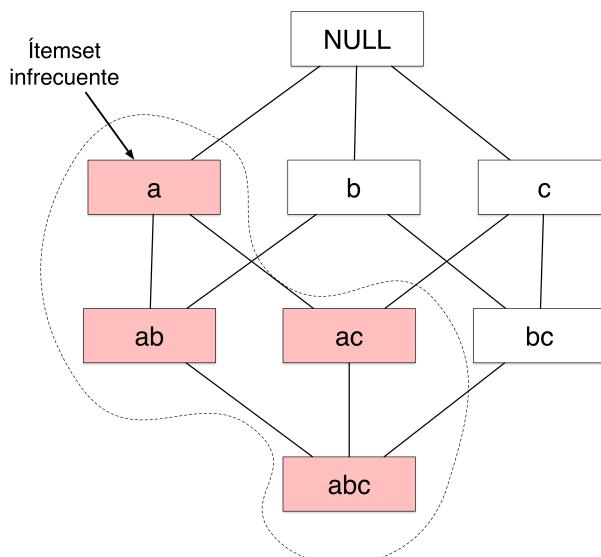


Figura 4: Rejilla de ítemset

Generación de reglas

Una vez que tenemos los ítemsets frecuentes, podemos empezar a generar reglas. Tomemos un ítemset Y con k ítems, podríamos generar $2^K - 2$ reglas de asociación sin considerar aquellas reglas con ítemsets nulos. Para generar una regla, basta dividir a Y en dos conjuntos no nulos: X y $Y - X$. Obtenemos la regla $X \implies Y - X$, ahora solo tenemos que evaluar si cumple con la restricción de confianza mínima.

Para calcular la confianza, no es necesario evaluar en todas las transacciones, ya que la función de confianza se calcula utilizando el soporte de los ítemsets y esto ya lo calculamos en el paso anterior.

Representación compacta de conjuntos de artículos frecuentes

Con el objetivo práctico de reducir el espacio de almacenamiento, se han propuesto técnicas para representar de una manera compacta a la rejilla de artículos frecuentes. Veamos una técnica basada en el concepto de artículos frecuentes máximos. Un ítemset de frecuencia máximo es aquel que en la rejilla no tiene un superconjunto inmediato superior que también sea frecuente.

En la figura se presenta una rejilla con tres ítemsets de frecuencia máximos $\{a, d\}$, $\{a, c, e\}$, $\{b, c, d, e\}$. Solo con estos ítemsets podemos derivar los ítemsets frecuentes restantes ya que están incluido (son subconjuntos) de los mismos.

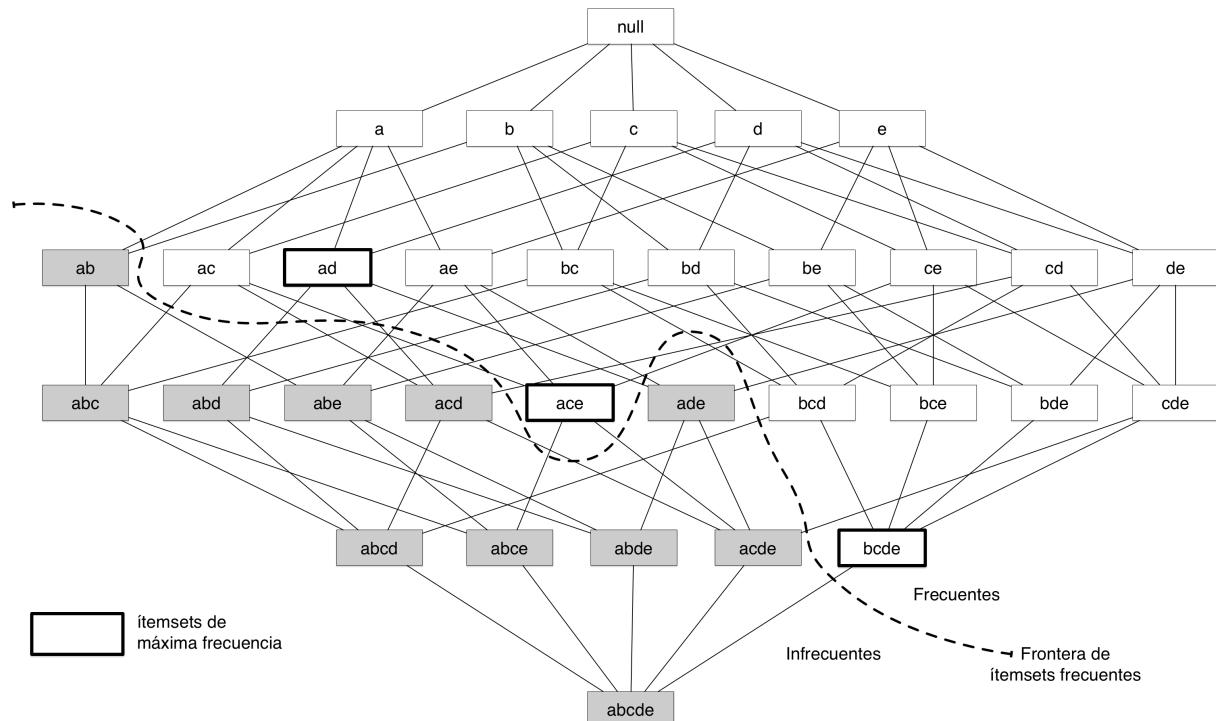


Figura 5: Rejilla de ítemset Tan, Steinbach, and Kumar (2007)

El algoritmo FP-Growth

Uno de los algoritmos más utilizados para el minado de reglas de asociación es el algoritmo FP-Growth (Han, Pei, and Yin 2000). El algoritmo aborda el problema de generación de ítemsets de una manera muy distinta a la estrategia utilizada por *Apriori*. El algoritmo propone el uso de una estructura compacta llamada *FP-tree* de donde se extraen directamente los ítemsets frecuentes:

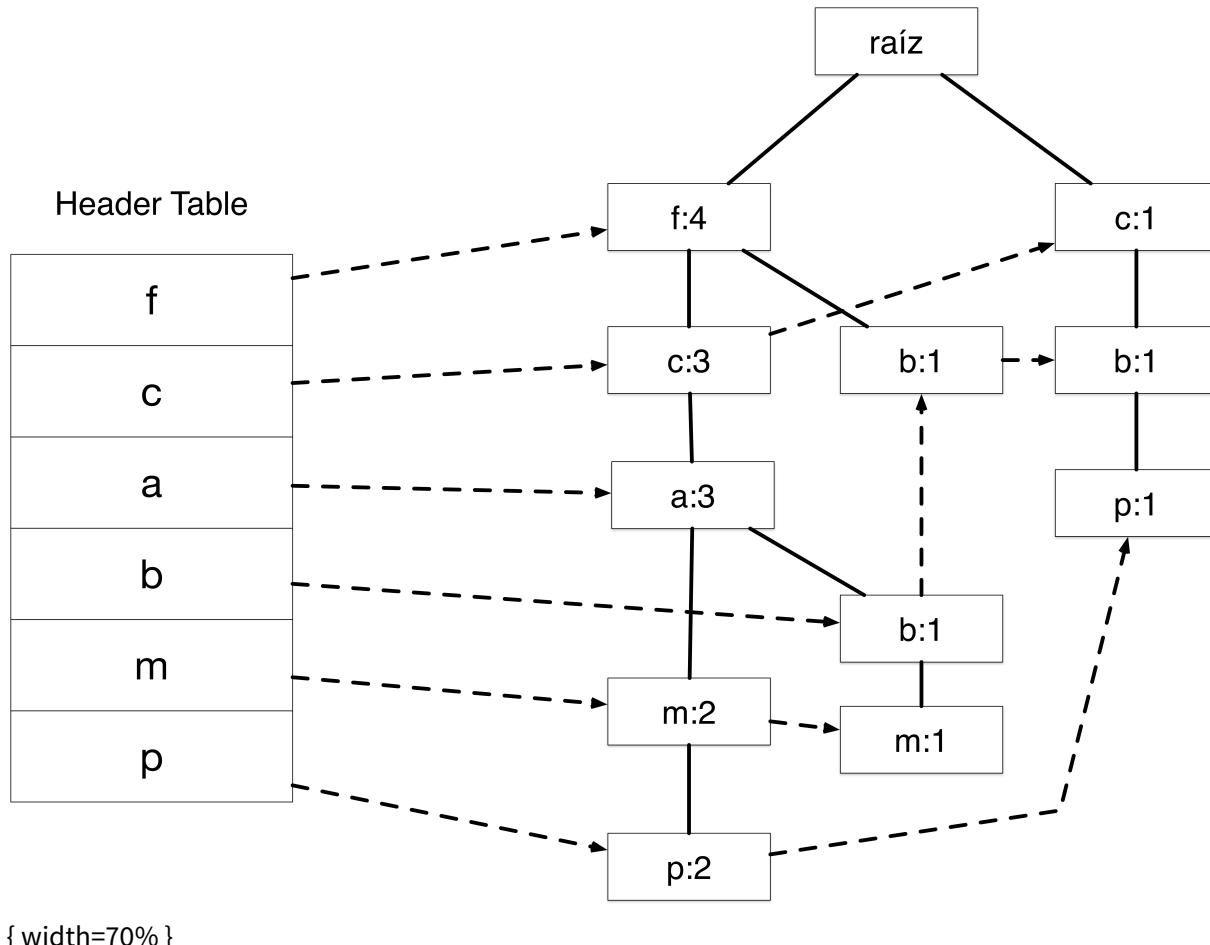
1. El algoritmo empieza haciendo un recorrido de la base de datos de transacciones *Trans*, copiando a una lista *F* solo los ítems frecuentes con su soporte. Después se ordena *F* de mayor a menor soporte y al resultado le llamaremos *L* la lista de ítems frecuentes.
2. Se crea el FP-tree, *T*, insertando el nodo raíz el cual se etiqueta como *null*.
3. Se recorre de nuevo *Trans* y para cada transacción se hace lo siguiente:
 1. Se ordenan sus ítemsets de acuerdo al orden especificado en *L*.
 2. A la lista ordenada de ítems se le considera como $[p|P]$, donde *p* es el primer elemento y *P* el resto. Hecho esto se llama al método *insert_tree*($[p|P]$, *T*).
 3. *insert_tree*($[p|P]$, *T*) hace lo siguiente. Si el árbol *T* tiene un nodo hijo *N* con el mismo nombre que *p*, se incrementa el contador de *N*; en caso contrario se crea un nuevo nodo y se la asigna 1 a su contador. El padre de este nuevo nodo será *T*. El nodo también se

liga a otros nodos con el mismo nombre por medio de una estructura adicional llamada «node-link». Si P no está vacío se llama de forma recursiva $\text{insert_tree}(P, N)$.

Ejercicio

Siguiendo el algoritmo anterior y utilizando la siguiente base de datos de transacciones, recrea en tu mente los pasos necesarios para crear el árbol de la figura. Ver los detalles en (Han, Pei, and Yin 2000):

Tid	Ítems	Ítems frecuentes (ordenados)
1	f, a, c, d, g, i, m, p	f, c, a, m, p
2	a, b, c, f, l, m; o	f, c, a, b, m
3	b, f, h, j, o,	f, b
4	b, c, k, s, p,	c, b, p
5	a, f, c, e, l, p, m, n	f, c, a, m, p



Antes de minar la estructura veamos las operaciones que nos permite la estructura:

1. Seguir el «node_link» (flechas punteadas) desde la «Header Table» cualquier ítem a_i para saber en que ítemsets participa. Por ejemplo, el ítem «p», aparece tres ítemsets, en dos ítemsets iguales {f, c, a, m, p} y en {c, b, p}.
2. Siguiendo los valores de soporte desde la raíz podemos ver cuantas veces se ha repetido el ítemset hasta el nodo deseado. Por ejemplo, {f, c, a} tiene un soporte de 3, ya que al final de la ruta en el árbol el último elemento tiene 3 (a:3).

El minado se realiza siguiendo una estrategia *bottom-up*.

Por ejemplo, empezando con el ítem p , vamos a extraer aquellos ítem sets en los que participa con un soporte de 3.

Primero vemos en que rutas participa. Siguiendo el «node_link»: vemos que en {f, c, a, m, p} y {c, b, p}. Empezamos por revisar el soporte del ítemset {p}, vemos siguiendo el «node_link» que tiene: «p:2» + «p:1» = 3. Por lo tanto tiene un soporte adecuado, decimos que es un ítemset frecuente. Siguiendo la primer ruta, ahora revisaremos {m, p}, {a, p}, {c, p} y {f, p}. Es importante actualizar el soporte de los ítems de la ruta durante el proceso. Ya que recordemos que no todos los ítemsets registrados más arriba incluyen a p . Debemos actualizar el soporte de todos a 2. Consideraremos entonces {f:2, c:2, a:2, m:2, p:2} y la otra ruta {c:1, b:1, p:1}. Al evaluar {m, e} vemos que el soporte es 2, por lo que no es frecuente. Solo {c, e} es frecuente ya que tenemos c:2 + c1, considerando la otra ruta a través del «node-link». Por lo tanto con el ítem p , solo encontramos dos ítemsets frecuentes: {p} y {c, p}. El algoritmo seguiría minando ahora con el ítem m y así sucesivamente.

El algoritmo FP-Growth, es un buen ejemplo de el uso de estructuras de datos para mejorar los algoritmos y atacar los problemas con un nuevo enfoque. Incluso en algunos casos FP-Growth mejora el desempeño de *Apriori* por varios órdenes de magnitud.

Análisis de Clusters

El análisis de clusters o grupos, es la tarea de aglutinar conjuntos de objetos similares (en cierto sentido). Los objetos que pertenecen a un cluster formado como resultado del análisis, deben ser más similares entre sí que con los objetos de otros grupos, incluso se desea que la distancia entre los grupos sea lo mayor posible. Es una de las principales tareas de la minería de datos, y el aprendizaje no supervisado.

Podemos hacer análisis de grupos sin necesidad de programar algoritmos o utilizar métodos matemáticos ya que el *agrupar objetos* es una de las tareas, que sin pensar, realizamos todos los días. ¿Cómo agruparías a los objetos de la siguiente figura?. Recuerda que se deben agrupar objetos similares. Po-

demos agrupar a los objetos, considerando su (a) ubicación, (b) tamaño, (c) color o algún otro atributo, incluso podemos considerar varias características, por ejemplo, el color y la figura. En todos estos casos hemos realizado un *análisis de grupos* sobre las figuras. Una de las preguntas que debemos hacernos al realizar este tipo de análisis es ¿Cuál de las agrupaciones es la correcta?, para ayudarnos a contestarla se han propuesto varias métricas, pero mucho depende del contexto de aplicación.

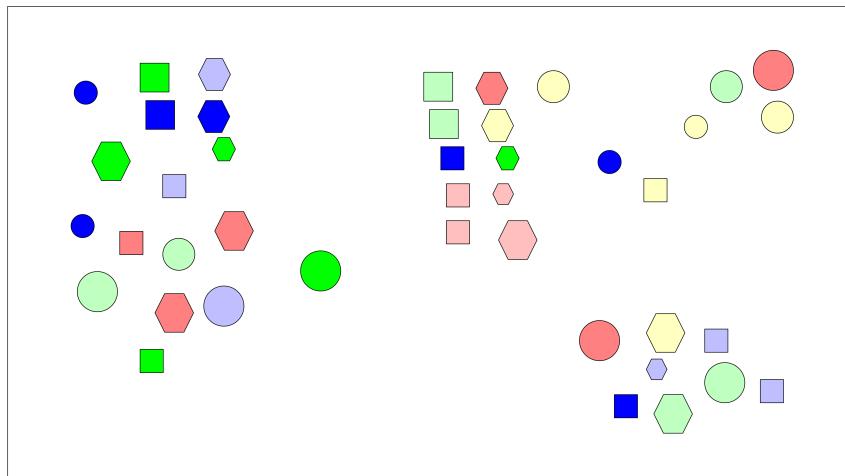


Figura 6: Conjunto de objetos con distinta posición, figura, color y tamaño.

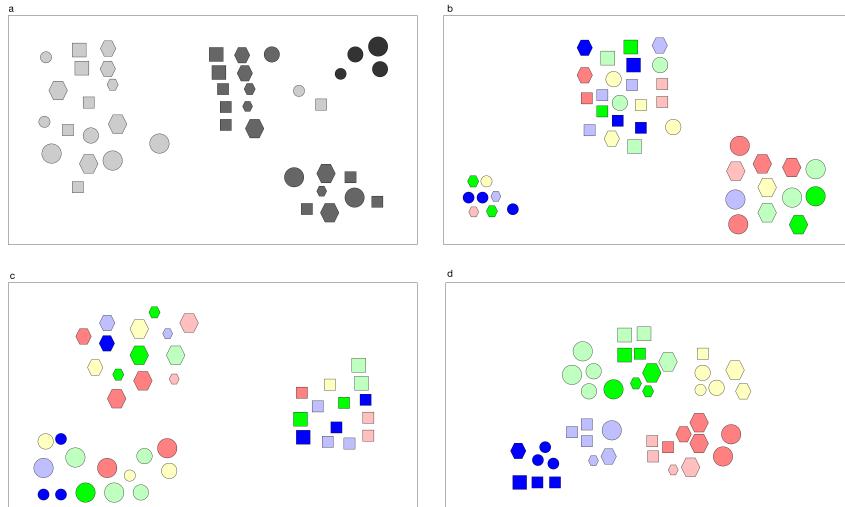


Figura 7: Distintos grupos extraídos de un análisis de grupos realizado sobre el conjunto de datos de ejemplo.

Si te fijas, esta misma actividad la realizan los científicos para clasificar o entender distintos fenómenos. Por ejemplo, el estudio de la evolución de las especies requiere de agrupar animales que tienen

características similares. En la figura siguiente, se muestra un dendograma, donde se agrupan las relaciones evolutivas entre animales del orden carnívora (imagen del Instituto de Biología Canina), donde podemos ver datos interesantes, como que el oso negro es más parecido a una morsa que a un perro.

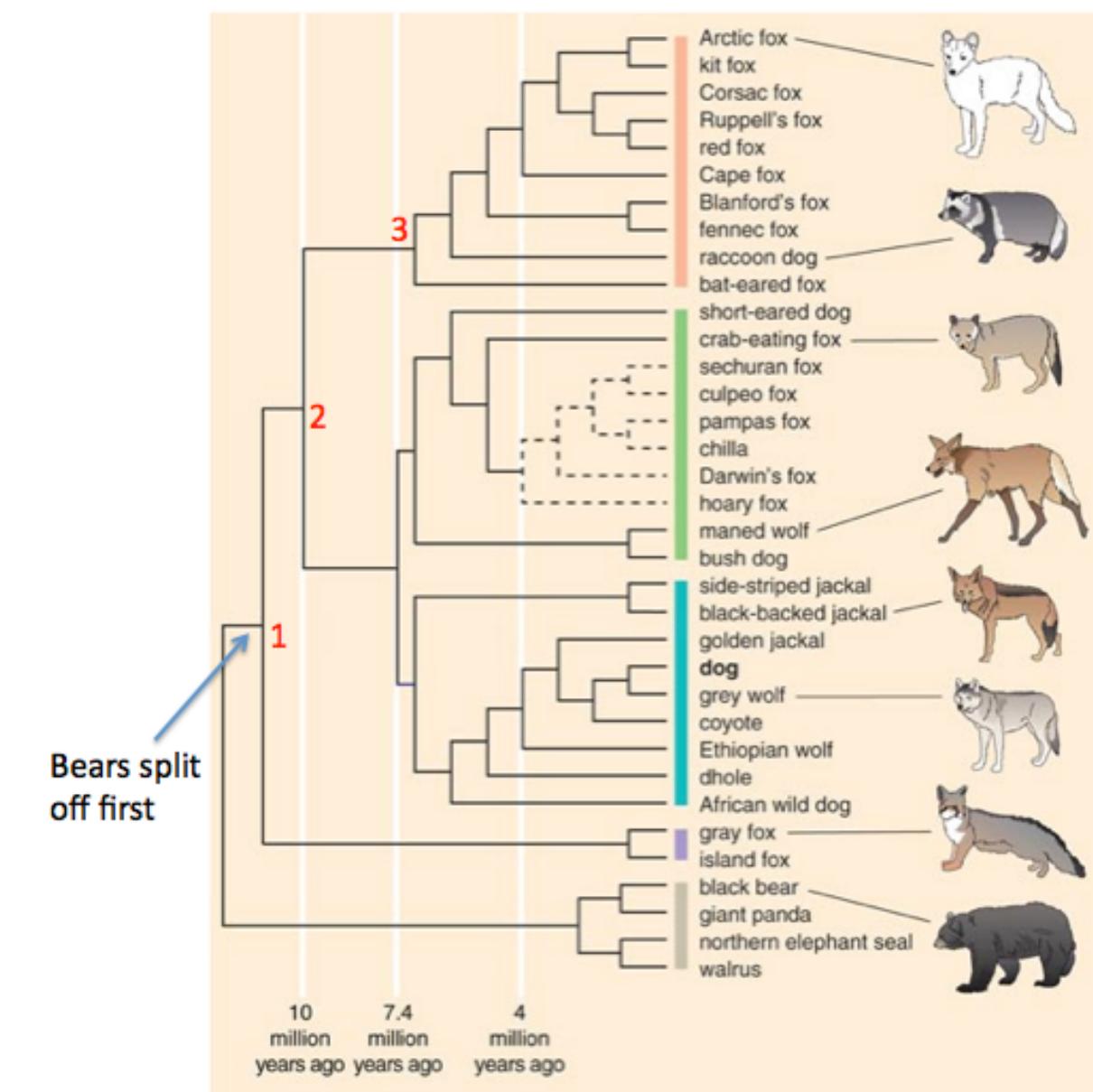


Figura 8: Imagen del Instituto de Biología Canina

Los métodos matemáticos de análisis de grupos, pueden realizar la misma tarea, pero en lugar de objetos reales como los animales o figuras, trabajan sobre datos que representan a los objetos. La representación más común es una matriz con vectores de datos continuos, categóricos u ordinarios.

Por ejemplo, las figuras analizadas anteriormente, pueden ser representadas por la siguiente matriz:

id	Figura	Posición	Color	Tamaño
1	Círculo	100, 230	0000FF	50, 50
2	Cuadrado	430, 450	CCFFCC	70, 70
3	Círculo	600, 230	FFFFCC	30, 30
4	Hexágono	300, 330	99DDFF	30, 30
5	Hexágono	700, 530	0000FF	70, 70
6	Cuadrado	200, 230	FFFFCC	70, 70

Los grupos encontrados a partir del análisis de clusters, son potencialmente *clases o categorías* de objetos: Podemos sugerir que el análisis de grupos es el descubrimiento de categorías sin necesidad de entrenamiento ya que las clases pueden extraerse de los datos sin necesidad de que sean asignados previamente. Cuidado, los algoritmos de análisis de grupos no reemplazan a los algoritmos de clasificación, ya que las categorías no solo dependen de la similaridad de ciertas características. Antes de pasar a los algoritmos veamos primero algunas de las aplicaciones de las técnicas de agrupamiento:

- **Video Juegos** Para los desarrolladores empresas es útil, identificar usuarios con comportamientos similares. Por ejemplo, a partir de capturar el movimiento y actividades de los usuarios, se han identificado tipos de jugadores como: agresivos, cazadores, exploradores, etc. Esta información se utiliza después para hacer cambios en la mecánica del video juego o para dar promociones a otros usuarios.
- **Recuperación de Información** Para mejorar el desempeño de la recuperación de la información, los documentos similares pueden almacenarse cerca unos de otros, mismo servidor, archivo, máquina, etc.
- **Datos Representativos** En lugar de trabajar con todos los datos para realizar algún análisis, se puede trabajar con objetos representativos de los distintos clusters generados previamente.
- **Cuantificación Vectorial** A partir de los clusters se pueden obtener vectores característicos. Existen técnicas de compresión como la cuantificación vectorial, que se basan en encontrar vectores característicos para un grupo de objetos similares, así en lugar de almacenar todos los datos de estos objetos, solo se almacena el identificador de su vector característico.

Clustering basado en prototipos

En este tipo de algoritmo, se tiene un objeto prototipo para cada uno de los grupos. Así la membresía de los objetos a su grupo depende de la similaridad con los prototipos. Para conjuntos de datos continuos, el prototipo es el centroide de todos los objetos que pertenecen al grupo.

Clustering basado en densidades

Los grupos se definen como una región de alta densidad rodeada de regiones de baja densidad.

Clustering basado en grafos

Los objetos dentro de la estructura de un grafo tienen conexiones intra-grupales pero no conexiones con otros grupos.

***k*-medias**

El algoritmo de *k*-medias es un algoritmo muy sencillo basado en prototipos. Solo requerimos especificar el parámetro *k* donde indicamos el número de clusters que deseamos obtener. En este algoritmo cada centroide define a un clúster. El algoritmo es el siguiente:

1. Se elige de manera aleatoria la posición de los *k* centroides.

Repetir estos pasos hasta que los centroides no se muevan:

1. Cada objeto se asigna al centroide más cercano.
2. Se Recalculan de nuevo los centroides a partir de los objetos que pertenecen a ellos.
3. Detener si ninguno cambia de posición, de otro modo continuar el ciclo.

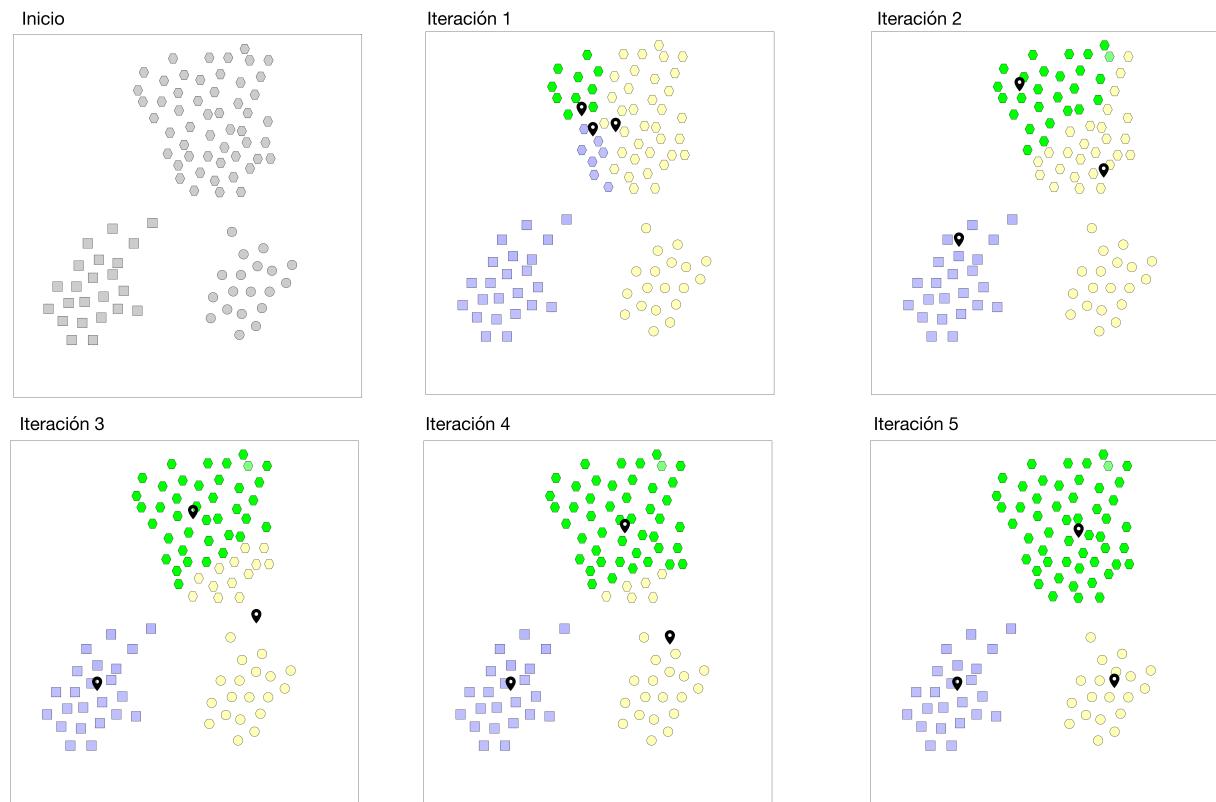


Figura 9: Ejemplo del algoritmo k -medias

Veamos los detalles del algoritmo sobre los datos de la figura x. Podemos observar que hay tres grupos **bien formados**, designados como círculos, rectángulos y hexágonos. Al inicio se posicionan los tres centroides, indicados por el ícono de lugar (). El siguiente paso es calcular la distancia o similitud entre los objetos y los centroides. En este caso utilizamos la distancia euclídea, ésta es muy común ya que es fácil de interpretar y Digamos que para cada objeto se calcula la distancia con cada centroide y se asigna al centroide más cercano, en este caso aquí lo indicamos con el color. Antes de iniciar la siguiente iteración se vuelven a calcular los centroides y como su posición cambió se inicia la iteración siguiente. En la figura podemos ver como después de cinco iteraciones y se ven pintados correctamente los grupos.

Incorrecta selección de las posiciones iniciales de los centroides

Debemos tener cuidado al ejecutar el algoritmo ya que la ubicación inicial aleatoria puede tener un efecto adverso. Como ejemplo, veamos el posicionamiento sugerido en la figura 2. Como vemos los clusters generados no son buenos ya que claramente vemos que en lugar de dos grupos en la parte superior debería de ser uno, y en la parte inferior dos en lugar de uno.

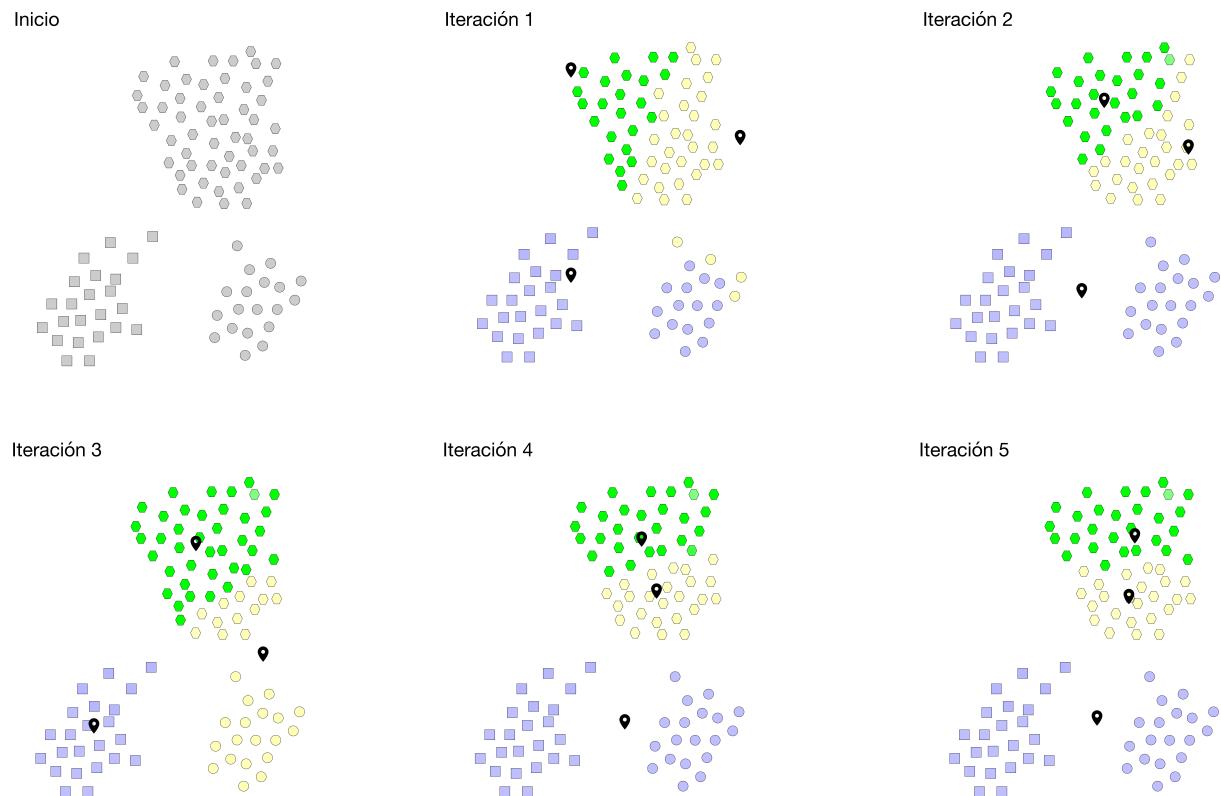


Figura 10: Incorrecta selección de posiciones iniciales de los centroides

¿Cómo evaluamos la calidad de los clusters generados?

Recordemos que lo que buscamos en minimizar la distancia interna de los miembros de un grupo y maximizar la distancia entre grupos. Como observamos en el clustering insuficiente de la figura x, en la parte superior hay dos clusters muy juntos y la parte inferior cluster con objetos muy lejanos entre sí. Una métrica muy utilizada es la de Suma del Error Cuadrático o SSE (del inglés Sum of Square Error), esta métrica considera precisamente la distancia de cada objeto a su centroide como un error, de tal manera que por lo menos eliminamos el problema de la parte inferior ya que al hacer el cálculo seguro dará un error mayor al de la agrupación correcta. La formula es esta:

$$SSE = \sum_{i=1}^k \sum_{x \in C_i} dist(m_i, x)^2$$

Donde,

- k es el número de centroides
- C_i es el conjunto de los miembros del centroide actual
- x es el objeto actual
- m_i es el centroide actual

- *dist* es la función de distancia (en este caso es el error)

Aunque no hay distancias negativas y por lo tanto no hay errores negativos, se mantiene la costumbre de elevar al cuadrado. También podemos observar que si aumentamos el número de clusters k el error es muy posible que disminuya ya que el error se distribuye. El error que suman dos miembros es probable que sea menor que el que generan 3.

¿Cómo evitamos el error provocado por la inicialización?

- Múltiples ejecuciones
- Inicialmente selecciona un número mayor de k centroides y luego de ellos.
- Utiliza otra clustering jerárquico para determinar los centroides.

Agrupamiento Jerárquico

Esta técnica busca crear una jerarquía de agrupamientos y se representa típicamente mediante un dendrograma. La representación nos permite ver como se han ido formando los grupos paso a paso, iniciando con objetos individuales en el primer nivel hasta llegar al nivel más alto que incluye a todos los objetos. A diferencia del algoritmo k -medias el cual crea un número determinado de agrupaciones en una agrupación jerárquica elegimos el nivel de agrupación deseado mediante la selección de un punto de corte en el árbol.

Primero vamos a ver el resultado final de un agrupamiento jerárquico. Distancias de <https://www.distancefromto.net/> manejando.

Ciudades	Tijuana	Ensenada	Mexicali	Diego	San Ángeles	Los Francisco	Las Vegas	San Jose	Barstow
Tijuana	83	181	28	204	763	557	557	557	557
Ensenada		240	143	287	946	669	762	339	
Mexicali			198	366	979	514	912	291	
San Diego				193	807	532	740	283	
Los Ángeles					614	433	492	146	
San Francisco						915	67	580	

Ciudades	Tijuana	Ensenada	Mexicali	Diego	San	Los	San	Las	San
					Ángeles	Francisco	Vegas	Jose	Barstow
Las Vegas								848	253
San Jose									515
Barstow									0

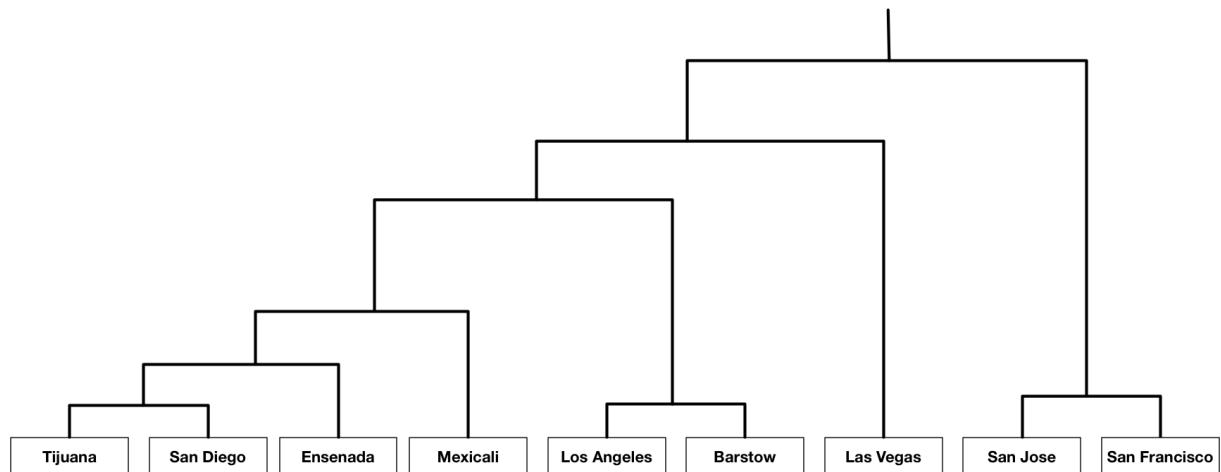


Figura 11: Ejemplo de agrupamiento jerárquico

DBSCAN

Bibliografía

Agrawal, Rakesh, Tomasz Imieliński, and Arun Swami. 1993. “Mining Association Rules Between Sets of Items in Large Databases.” In *AcM Sigmod Record*, 22:207–16. 2. ACM.

Clark, Peter, and Tim Niblett. 1989. “The Cn2 Induction Algorithm.” *Machine Learning* 3 (4). Springer: 261–83.

Cohen, William W. 1995. “Fast Effective Rule Induction.” In *Machine Learning Proceedings 1995*, 115–23. Elsevier.

Han, Jiawei, Jian Pei, and Micheline Kamber. 2011. *Data Mining: Concepts and Techniques*. Elsevier.

Han, Jiawei, Jian Pei, and Yiwen Yin. 2000. “Mining Frequent Patterns Without Candidate Generation.”

Introducción a la Minería de Datos

In *ACM Sigmod Record*, 29:1–12. 2. ACM.

Hilderman, Robert James, and Howard John Hamilton. 1999. *Knowledge Discovery and Interestingness Measures: A Survey*. Citeseer.

Tan, Pang-Ning, Michael Steinbach, and Vipin Kumar. 2007. *Introduction to Data Mining*. Pearson Education India.