CIENCIAS DE LA COMPUTACIÓN
UNIVERSIDAD DE CHILE

Curso DM
“Minería de Reglas de Asociación’’

Primavera 2023

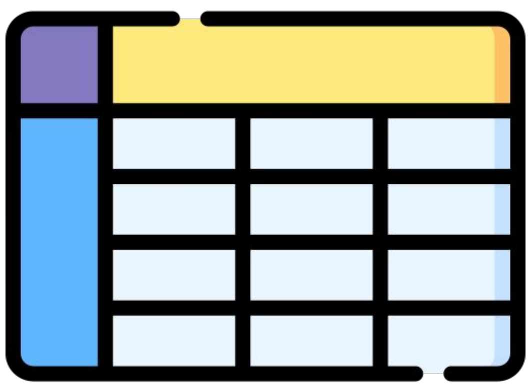
Basado en las slides de Bárbara Poblete

Minería de reglas de asociación

- Métodos para encontrar relaciones entre atributos en grandes volúmenes de datos.

Minería de reglas de asociación

Base de datos transaccional muy grande



Objetivo

Encontrar reglas de asociación entre estos atributos o conjuntos de ítems frecuentes.

Cada transacción tiene distintos items. Por ejemplo: pan, queso, mantequilla

Minería de reglas de asociación

Registro de compras
de los clientes de
una tienda

Buscamos extraer reglas como:

{pan, queso} => {mantequilla}
{cerveza, coca-cola} => {pisco}

Conjuntos de atributos frecuentes

Minería de reglas de asociación

No hay una variable objetivo que quiero modelar

- Tarea no supervisada

- Buscamos **grupos de atributos** que ocurren frecuentemente

A diferencia de clustering en donde buscamos grupos de objetos

Minería de reglas de asociación

- Tiene una respuesta exacta
- Criterio principal de evaluación de un algoritmo es su eficiencia computacional

Ej. Encontrar todas las reglas de asociación que aparecen al menos 10 veces en la base de datos.

Minería de reglas de asociación

- Primer algoritmo eficiente fue presentado en 1993 en SIGMOD (congreso importante en base de datos).

Agrawal, R., Imieliński, T., & Swami, A. (1993, June). Mining association rules between sets of items in large databases. In Proceedings of the 1993 ACM SIGMOD international conference on Management of data (pp. 207-216).

Minería de reglas de asociación

- Idealmente trabajamos sobre **datos categóricos y binarios**
- Existen adaptaciones para trabajar con secuencias, grafos y otros tipos de entradas estructuradas.

¿Para qué?

- Muchos negocios acumulan grandes cantidades de datos sobre sus operaciones diarias.
    - Ejemplo: transacciones en una tienda de retail
- Aprender de estos datos permite entender comportamiento de los clientes
- La información permite tomar decisiones (manejo de inventario, promoción de productos para venta cruzada, etc.)

Ejemplo

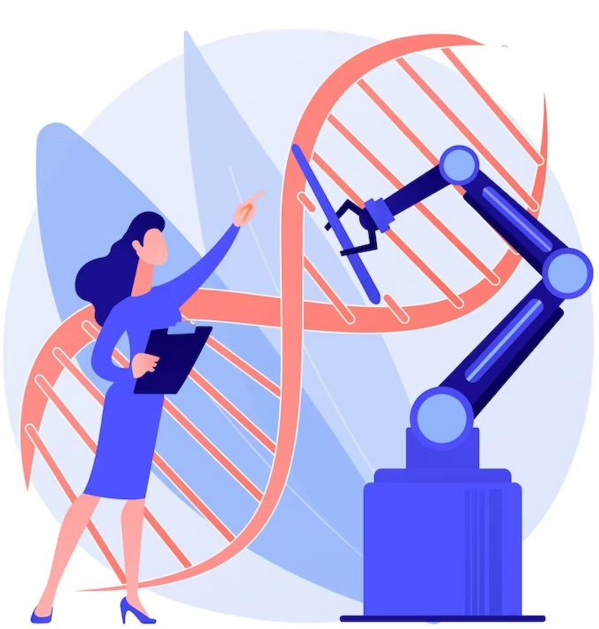
- Tenemos las siguientes transacciones en una canasta de compra:

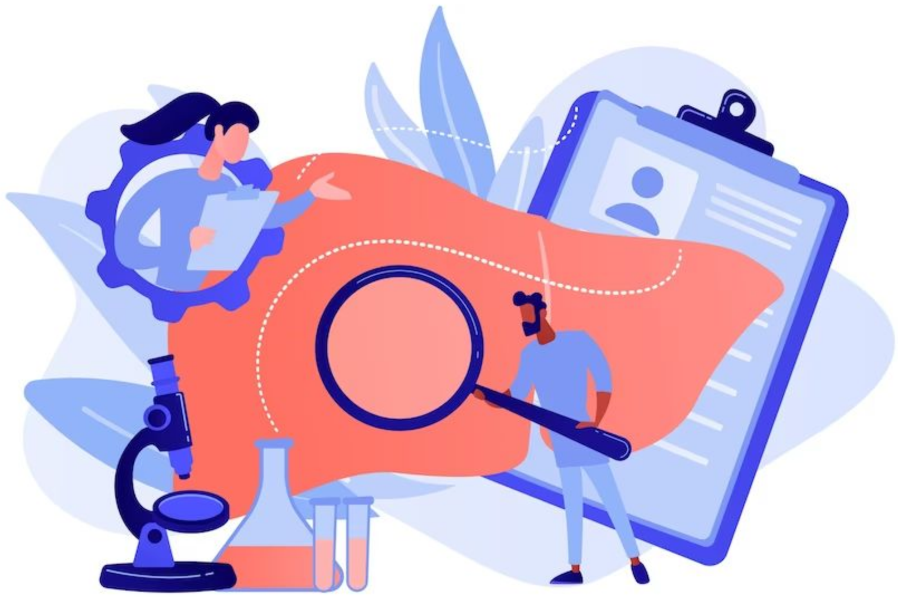
TID	Items
1	{Bread, Milk}
2	{Bread, Diapers, Beer, Eggs}
3	{Milk, Diapers, Beer, Cola}
4	{Bread, Milk, Diapers, Beer}
5	{Bread, Milk, Diapers, Cola}

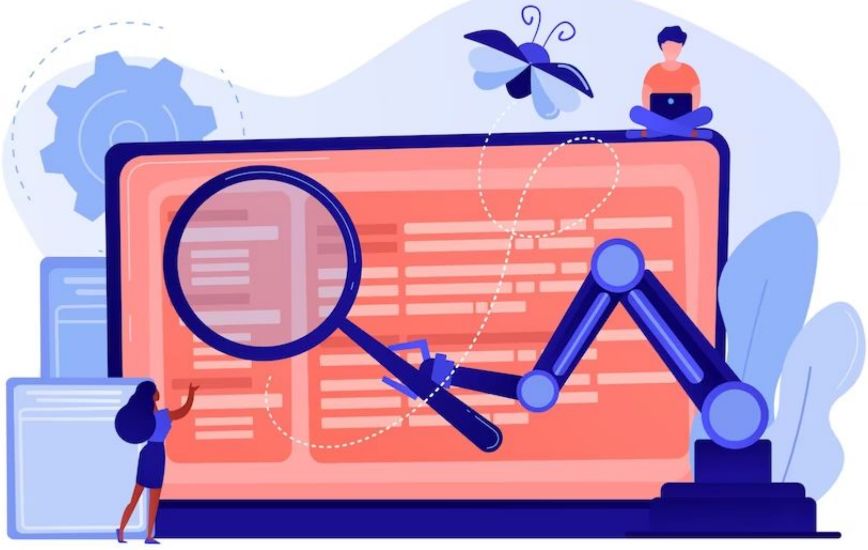
- La siguiente regla se podría extraer de estos datos:

```
{Diapers} —> {Beer}
```

Dominios de Aplicación







Conceptos

Base de datos transaccional

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Transacción

Item

Conceptos

Itemset: conjunto de uno o más ítems.

Ej. {Milk, Bread, Diaper} $\longrightarrow$ **k-itemset, k=3**

Support count (σ): Frecuencia con que ocurre un ítemset.

Ej. σ({Milk, Bread, Diaper}) = 2

Conceptos

Support: Fracción de las transacciones que contiene un ítemset

Métrica normalizada

Ej. s({Milk, Bread, Diaper}) = σ({Milk, Bread, Diaper}) / |T|

Itemset frecuente: Un itemset cuyo support es mayor o igual a un parámetro predefinido (*minsup*).

Definición de Regla de Asociación

Regla de Asociación: Expresión de implicancia de la forma:

X e Y son ítemsets

$$X \longrightarrow Y$$

antecedente   consecuente

Ej. {Milk, Bread} → {Diaper}

Métricas para evaluar reglas

Soporte o Support (s) : Fracción de las transacciones que contienen a ambos X e Y

$$\mathbf{s(X \longrightarrow Y)} = \frac{\boldsymbol{\sigma}\mathbf{(X \cup Y)}}{\mathbf{|T|}}$$

Confianza o Confidence (c) : Mide qué tan frecuentemente los ítems en Y aparecen en transacciones que contienen X

$$\mathbf{c(X \longrightarrow Y)} = \frac{\boldsymbol{\sigma}\mathbf{(X \cup Y)}}{\boldsymbol{\sigma}\mathbf{(X)}}$$

Métricas para evaluar reglas

{Milk, Diaper} → {Beer}

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

$$\mathbf{s} = \frac{\sigma(\text{Milk, Diaper, Beer})}{|T|} = \frac{2}{5} = 0{,}4$$

$$\mathbf{c} = \frac{\sigma(\text{Milk, Diaper, Beer})}{\sigma(\text{Milk, Diaper})} = \frac{2}{3} = 0{,}67$$

Minería de Reglas de Asociación

Dado un conjunto de transacciones T, el objetivo de la minería de reglas de asociación es encontrar todas las reglas que tengan:

- support >= minsup
- confidence >= minconf

La implicancia significa co-ocurrencia y no causalidad

Minería de Reglas de Asociación

- A diferencia de clasificación y clustering, aquí buscamos una solución exacta.
- Siempre pueden ocurrir asociaciones aleatorias de poco valor.
- El resultado debe ser interpretado con precaución
- Una fuerte correlación no necesariamente implica causalidad.

¿Por qué usamos support y confidence?

- Si es soporte es muy bajo
  - X e Y pueden haber co-ocurrido por azar
  - También es poco interesante desde el punto de vista del negocio
  - Sirve para eliminar reglas poco interesantes

¿Por qué usamos support y confidence?

- La confianza mide cuánto podemos confiar en la inferencia hecha por la regla.
- Mientras mayor sea la confianza, mayor será la probabilidad de observar Y en transacciones que tengan X.
- La confianza estima la probabilidad condicional: $P(Y|X)$

Algoritmo

¿Cómo podríamos encontrar todas las reglas de asociación que cumplan la condición de soporte mínimo y confianza mínima en un dataset?

Algoritmo

Aproximación por Fuerza-Bruta:

- Listar todas las reglas de asociación posibles (R)
- Calcular el *support* y *confidence* de cada regla
- Filtrar las reglas que no cumplan con las restricciones de *minsup* y *minconf*

¡Computacionalmente prohibitivo!

Para n items

$$R = 3^n - 2^{n+1} + 1$$

Muchas de las reglas no cumplen la condición

Algoritmo

Entonces queremos un algoritmo más inteligente que filtre las cosas que no son frecuentes para que no tengamos que iterar sobre todas las combinaciones posibles.

Intentaremos separar el problema en dos partes

Algoritmo

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

{Milk,Diaper} → {Beer} (s=0.4, c=0.67)

{Milk,Beer} → {Diaper} (s=0.4, c=1.0)

{Diaper,Beer} → {Milk} (s=0.4, c=0.67)

{Beer} → {Milk,Diaper} (s=0.4, c=0.67)

{Diaper} → {Milk,Beer} (s=0.4, c=0.5)

{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

- Todas las reglas listadas vienen del mismo ítemset: **{Milk, Diaper, Beer}**
- Tienen igual support
- Pueden tener diferente confidence
- El cálculo del support sólo depende de X U Y

Algoritmo

1. **Generación de ítemsets frecuentes**
   Generar todos los ítemsets que cumplan la restricción de *support >= minsup*

2. **Generación de reglas**
   Generar las reglas de alto confidence para cada ítemset, donde cada regla es una partición binaria de un ítemset frecuente

Aún así la generación de patrones frecuentes es muy costosa

Parte 1: Generación de Itemsets Frecuentes

Algoritmo

- **Estrategia fuerza bruta:** Generar todos los ítemsets y descartar todos los que no cumplan minsup

Esto requiere comparaciones del orden O(NMw) donde N es el número de transacciones, M el número de itemsets y w el largo de la transacción con más ítems.

Estrategia inteligente: Reducir el número de ítemsets candidatos a ser evaluados

El Principio Apriori

Si un itemset es frecuente, entonces todos sus subconjuntos son frecuentes.

De manera análoga, si un ítemset es infrecuente todos sus superconjuntos son infrecuentes.

Todos los itemset posibles

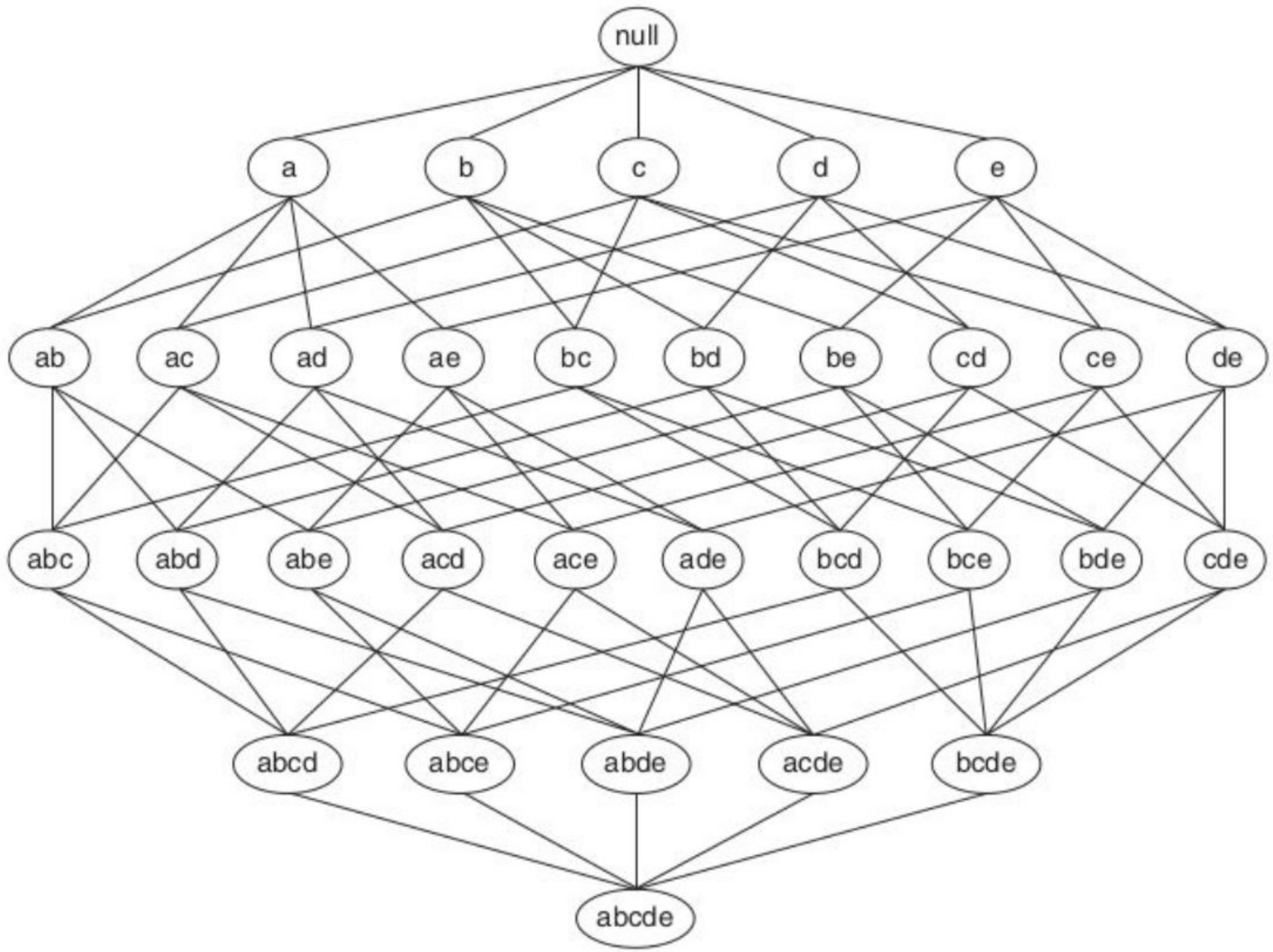
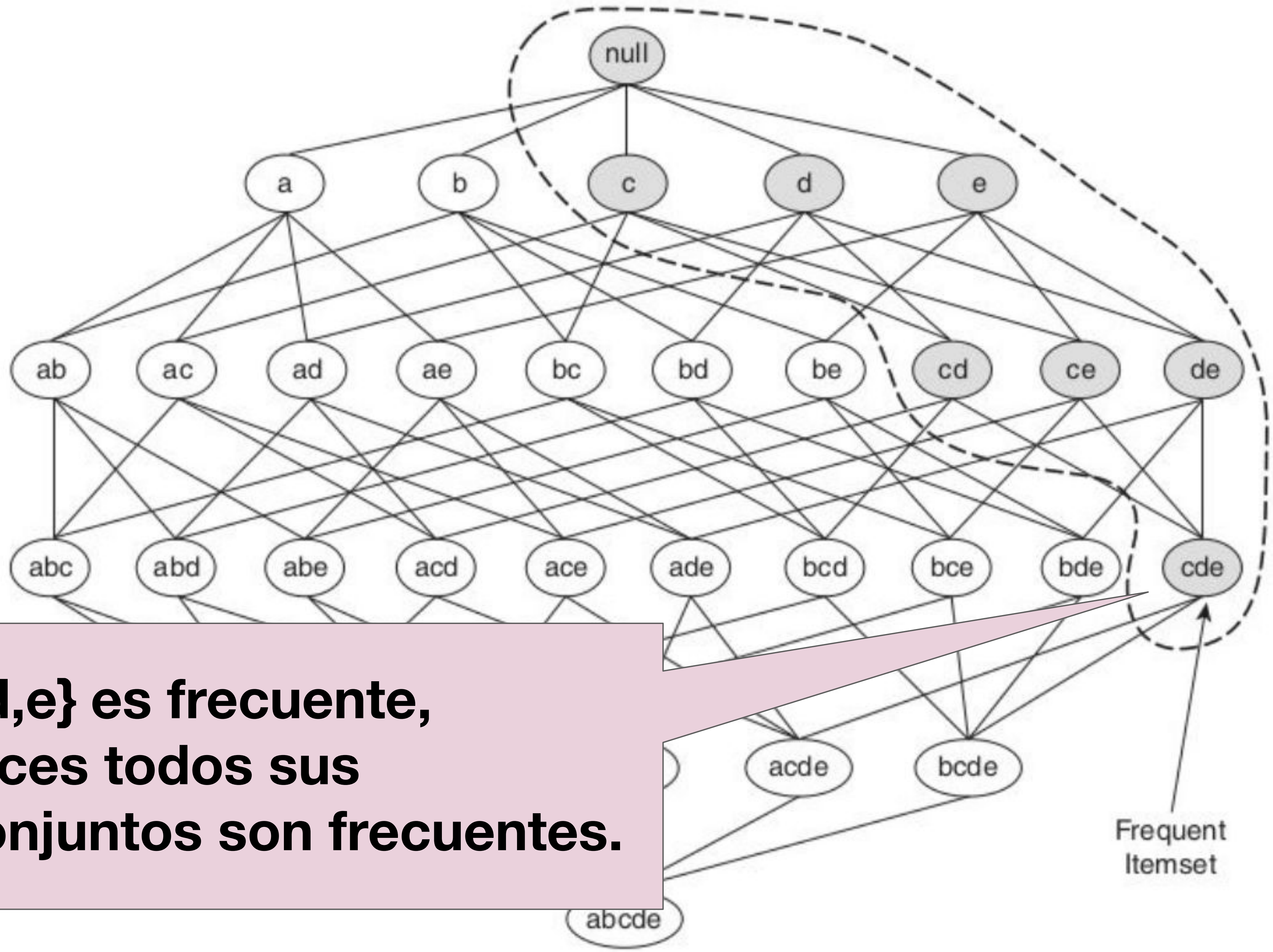


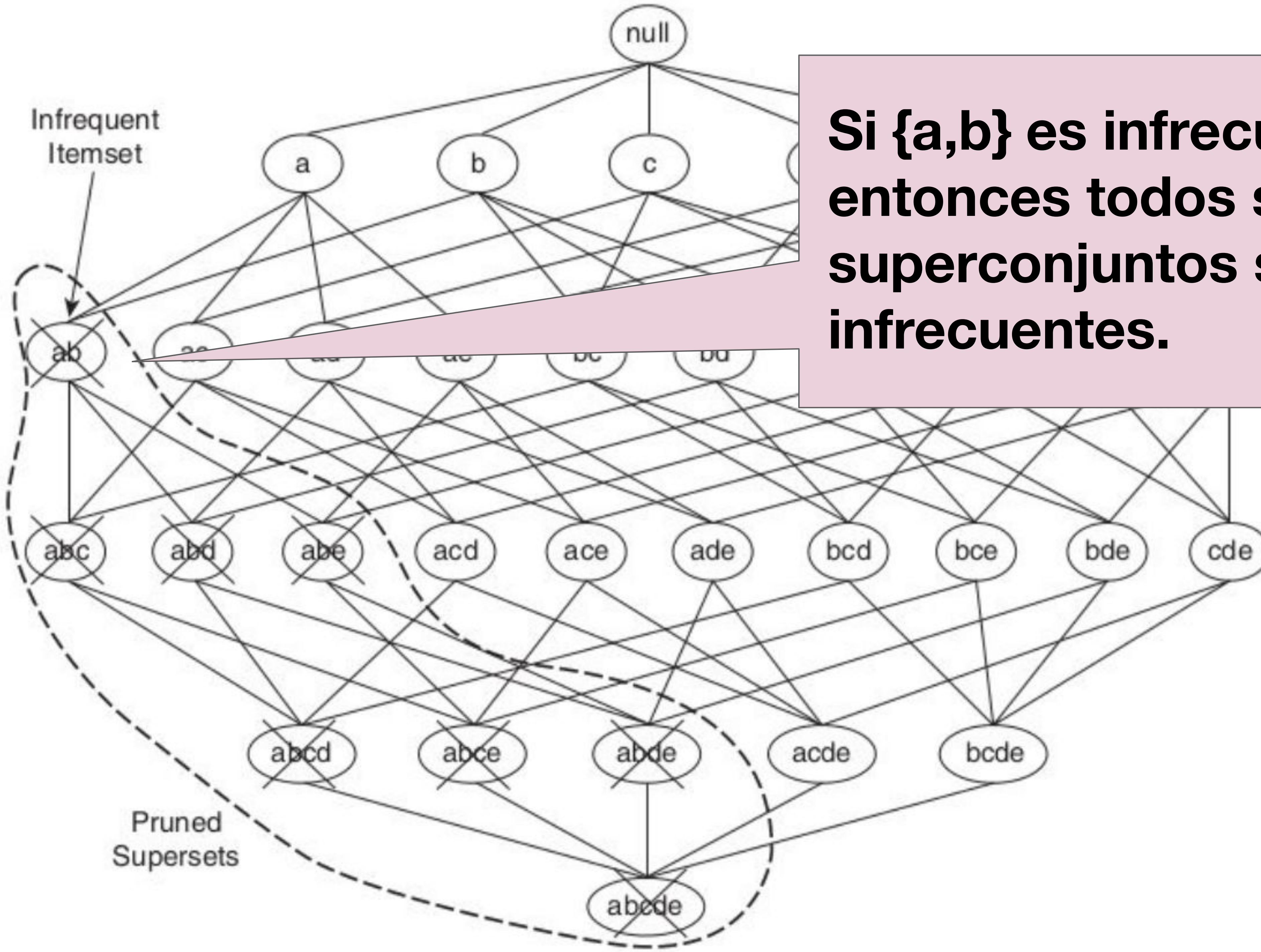
Figure 5.1. An itemset lattice.

Principio Apriori



Si {c,d,e} es frecuente, entonces todos sus subconjuntos son frecuentes.

Principio Apriori



Si {a,b} es infrecuente, entonces todos sus superconjuntos son infrecuentes.

El Principio Apriori

Apriori es el primer algoritmo para encontrar reglas de asociación que usa **poda basada en soporte** para mitigar el crecimiento exponencial de los itemset candidatos.

Objetivo: reducir la cantidad de candidatos a itemsets frecuentes aprovechando el principio apriori.

El Principio Apriori

- **Encontrar 1-itemsets es fácil**: se escanea la base de datos y se cuenta la frecuencia de cada ítem.

- **Idea**: mezclar pares de 1-itemsets frecuentes para encontrar candidatos a 2-itemsets frecuentes, luego repetir usando pares de 2-itemsets frecuentes para encontrar candidatos a 3-itemsets, y así sucesivamente

Principio Apriori

Candidate
1-Itemsets

Item	Count
Beer	3
Bread	4
Cola	2
Diapers	4
Milk	4
Eggs	1

Minimum support count = 3

Candidate
2-Itemsets

Itemset	Count
{Beer, Bread}	2
{Beer, Diapers}	3
{Beer, Milk}	2
{Bread, Diapers}	3
{Bread, Milk}	3
{Diapers, Milk}	3

Itemsets removed
because of low
support

Candidate
3-Itemsets

Itemset	Count
{Bread, Diapers, Milk}	2

El Principio Apriori

Por el principio Apriori sabemos que si X es un k-itemset frecuente, entonces todos sus (k-1)-item subsets son frecuentes también.

- Estrategia: encontrar k-itemsets mezclando (k-1)-itemsets frecuentes.

El Principio Apriori

Los ítems dentro de un itemset se ordenan lexicográficamente y así sólo mezclamos pares de itemsets que difieren en su último ítem.

– Esto nos asegura que no generamos dos veces el mismo k-itemset combinando (k-1)-itemsets.

– Ejemplo: {b,c,a} y {a,c,b} se transforman a {a,b,c}.

- Para encontrar candidatos de k-itemsets frecuentes, sólo mezclamos (k-1)-itemsets que tengan los mismos k-2 primeros ítems.

Ejemplo

Tenemos cinco 3-itemsets frecuentes

`(A B C), (A B D), (A C D), (A C E), (B C D)`

Sólo mezclamos pares de ítemsets que difieren en su último ítem:

- `(A B C)` con `(A B D)`
- `(A C D)` con `(A C E)`

Ejemplo

Tenemos cinco 3-itemsets frecuentes

`(A B C), (A B D), (A C D), (A C E), (B C D)`

Candidatos a 4-itemsets:

Chequeo que todos los sub (k-1)-itemsets del k-itemset candidato son frecuentes.

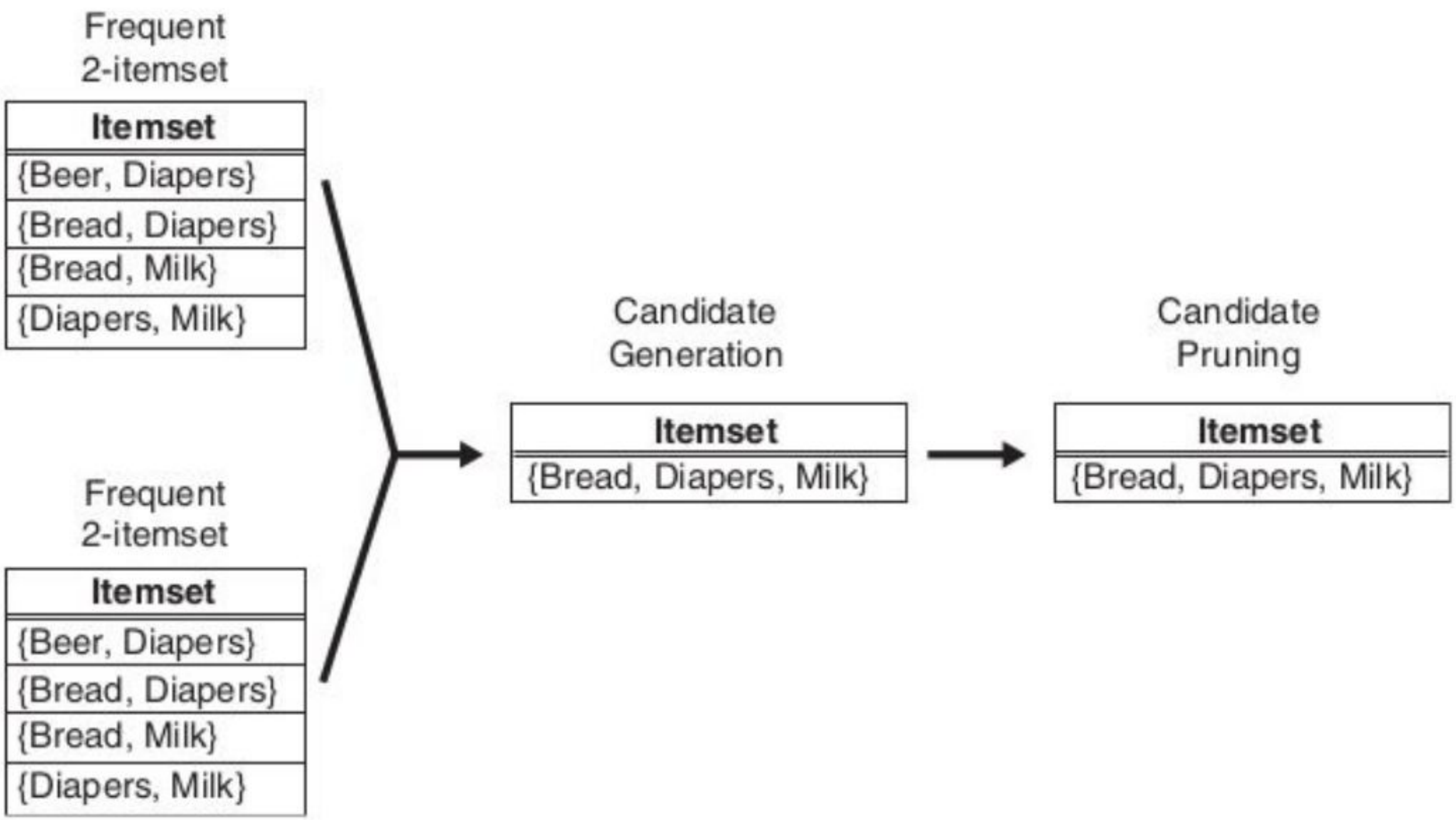
`(A B C D)` es un candidato válido porque todos sus subconjuntos son frecuentes `(A B C)(A C D)(B C D)`

`(A C D E)` No es candidato porque `(C D E)` no es frecuente

Ejemplo

Al final se deben contar todas las transacciones que contengan el k-itemset candidato. Un k-itemset puede ser infrecuente incluso si todos sus subconjuntos son frecuentes (es una condición necesaria pero no suficiente).

Principio Apriori



Algoritmo Apriori

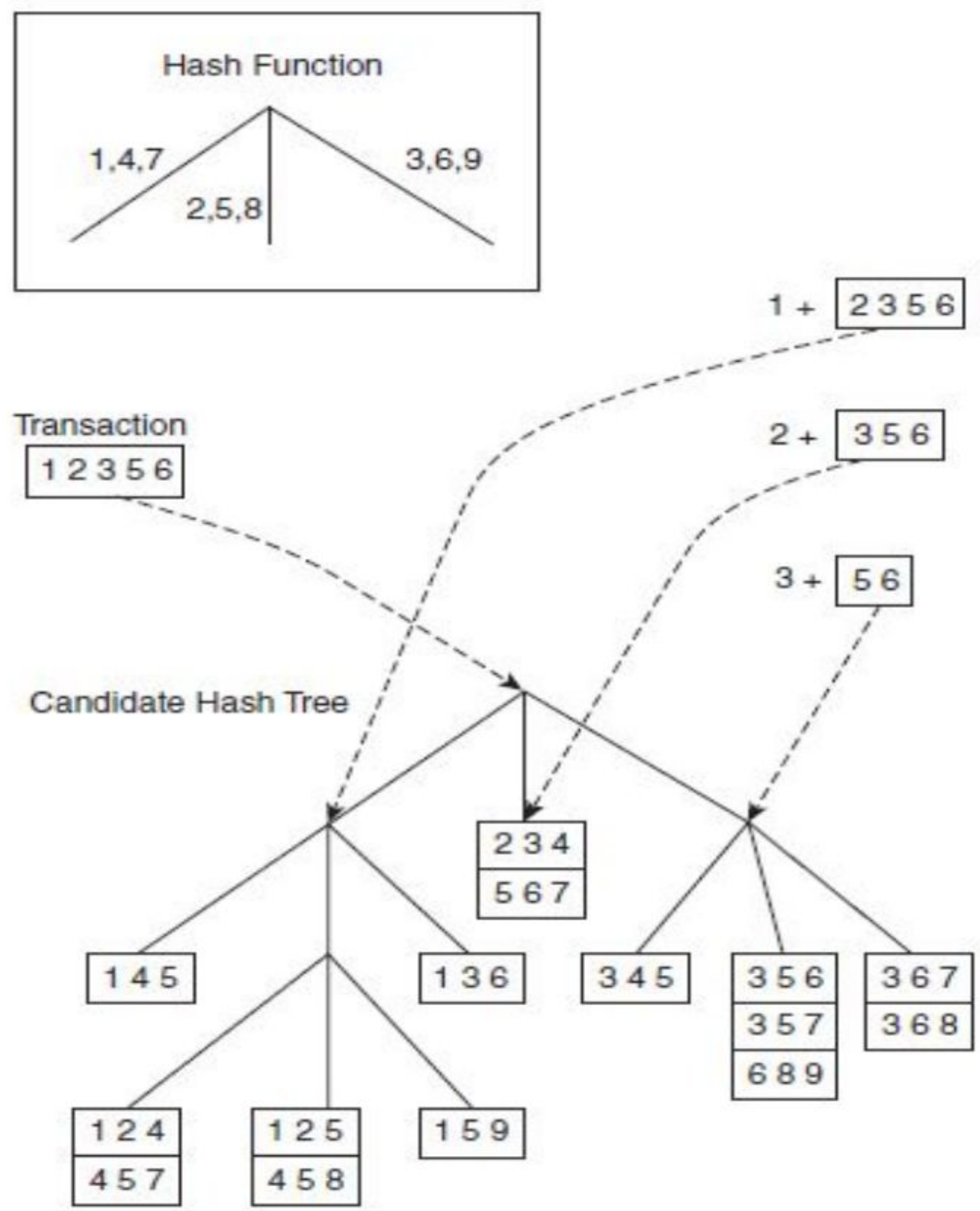
1. Encuentro los 1-itemset frecuentes escaneando la base de datos

2. **Mezcla**: Encuentro candidatos a k-itemsets frecuentes combinando pares de (k-1)-itemsets frecuentes que sólo difieran en su último elemento. (Los itemsets deben estar ordenados lexicográficamente)

3. **Poda**: chequeo que los sub-itemsets del candidato sean frecuentes. Si encuentro algún sub-itemset no frecuente descarto el candidato por principio Apriori.

4. **Conteo de soporte:** cuento el soporte del itemset candidato y chequeo si cumple el criterio minsup. Uso un árbol hash (hash tree) para hacer el conteo de manera eficiente.

Usa un Hash Tree para mantener los conteos de soporte



Algoritmo Apriori para generación de itemset frecuentes

Algorithm 5.1 Frequent itemset generation of the *Apriori* algorithm.

$k = 1$.
$F_k = \{\ i \mid i \in I \wedge \sigma(\{i\}) \geq N \times minsup\}$. {Find all frequent 1-itemsets}
repeat
    $k = k + 1$.
    C_k = candidate-gen(F_{k-1}). {Generate candidate itemsets.}
    C_k = candidate-prune$(C_k,\ F_{k-1})$. {Prune candidate itemsets.}
    for each transaction $t \in T$ **do**
        C_t = subset(C_k, t). {Identify all candidates that belong to t.}
        for each candidate itemset $c \in C_t$ **do**
            $\sigma(c) = \sigma(c) + 1$. {Increment support count.}
        end for
    end for
    $F_k = \{\ c \mid c \in C_k \wedge \sigma(c) \geq N \times minsup\}$. {Extract the frequent k-itemsets.}
until $F_k = \emptyset$
Result $= \bigcup F_k$.

Parte 2: Generación de Reglas de Asociación

Generación de Reglas a partir de un Itemset

Una vez encontrados todos los itemsets que satisfacen la restricción de *minsup*, podemos usarlos para generar reglas.

- Podemos particionar el itemset Y en dos subconjuntos no vacios X e Y – X para formar la regla X → Y – X

- Donde X → Y – X, tiene que satisfacer la restricción de *minconf*.

Generación de Reglas a partir de un Itemset

Ejemplo:

Y = {a,b,c}

X = {a,b}

Y – X = {c}

Produce la regla: **{a,b} → {c}**

Generación de Reglas a partir de un Itemset

Para el itemset Y = {a, b, c}

Se pueden generar 6 reglas 2^k-2, ignorando las que tienen antecedente o consecuente vacío.

{a, b} → {c}
{a, c}→{b}
{b, c} → {a}
{a} → {b, c}
{b}→ {a, c}
{c} → {a, b}

Como el soporte de cada regla es igual al de Y, todas estas reglas satisfacen minsup.

Generación de Reglas a partir de un Itemset

Queremos encontrar todas las reglas que satisfacen *minconf*

$$c(X \longrightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

Los valores de soporte ya fueron calculados en la fase previa y se encuentran guardados en el hash tree.

No tendremos que escanear la base de datos nuevamente! :D

Generación de Reglas a partir de un Itemset

Ejemplo:

Considere la regla **{1,2}** → **{3}**

Generada a partir del itemset **Y = {1, 2, 3}**

$$\mathbf{c} = \frac{\boldsymbol{\sigma}(\{1, 2, 3\})}{\boldsymbol{\sigma}(\{1, 2\})}$$

Como **{1, 2, 3}** es frecuente, el principio apriori nos asegura que **{1, 2}** es frecuente también y por ende el soporte del itemset **{1, 2}** se encuentra guardado en el hash tree.

¿Cómo generamos las reglas?

Estrategia fuerza bruta: Generar todas las reglas posibles a partir de todos los itemsets frecuentes y ver si cumplen con *minconf*.

Eso equivale a evaluar 2k-2 reglas (con k el tamaño del itemset).

Esto es muy costoso computacionalmente

Poda basada en confianza

Estrategia eficiente: poda basada en confianza.

Teorema:

Sea Y un itemset y X un subconjunto de Y .

- Si una regla $X \rightarrow Y - X$ no satisface minconf, entonces cualquier regla $\tilde{X} \rightarrow Y - \tilde{X}$, con $\tilde{X}$ subconjunto de X, tampoco va a satisfacer la regla de *minconf*.
- Si muevo itemsets del antecedente al consecuente no puedo subir el nivel confianza.

Ejemplo

Sea $\mathbf{Y = \{a,b,c\}}$, $\mathbf{X=\{a,b\}}$, $\mathbf{\tilde{X} = \{a\}}$, con $\mathbf{\tilde{X} \subset X}$

$\mathbf{X \to Y - X = \{a,b\} \to \{c\}}$

$\mathbf{\tilde{X} \to Y - \tilde{X} = \{a\} \to \{b,c\}}$

moví un item del antecedente al consecuente

$$c(X \to Y - X) = \frac{\sigma(Y)}{\sigma(X)} = \frac{\sigma(\{a, b, c\})}{\sigma(\{a, b\})}$$

$$c(\tilde{X} \to Y - \tilde{X}) = \frac{\sigma(Y)}{\sigma(\tilde{X})} = \frac{\sigma(\{a, b, c\})}{\sigma(\{a\})}$$

Ejemplo

$X \rightarrow Y - X = \{a,b\} \rightarrow \{c\}$

$$c(X \rightarrow Y - X) = \frac{\sigma(\{a, b, c\})}{\sigma(\{a, b\})}$$

$\tilde{X} \rightarrow Y - \tilde{X} = \{a\} \rightarrow \{b,c\}$

$$c(\tilde{X} \rightarrow Y - \tilde{X}) = \frac{\sigma(\{a, b, c\})}{\sigma(\{a\})}$$

Por principio apiori **$\sigma(\{a\}) >= \sigma(\{a, b\})$, $\sigma(\tilde{X}) >= \sigma(X)$**

$c(\{a\} \rightarrow \{b, c\})$ requiere dividir por un número más grande que en **$c(\{a, b\} \rightarrow \{c\})$**

Entonces **$c(\{a, b\} \rightarrow \{c\}) >= c(\{a\} \rightarrow \{b, c\})$**

La confianza no puede crecer si muevo itemsets del antecedente al consecuente

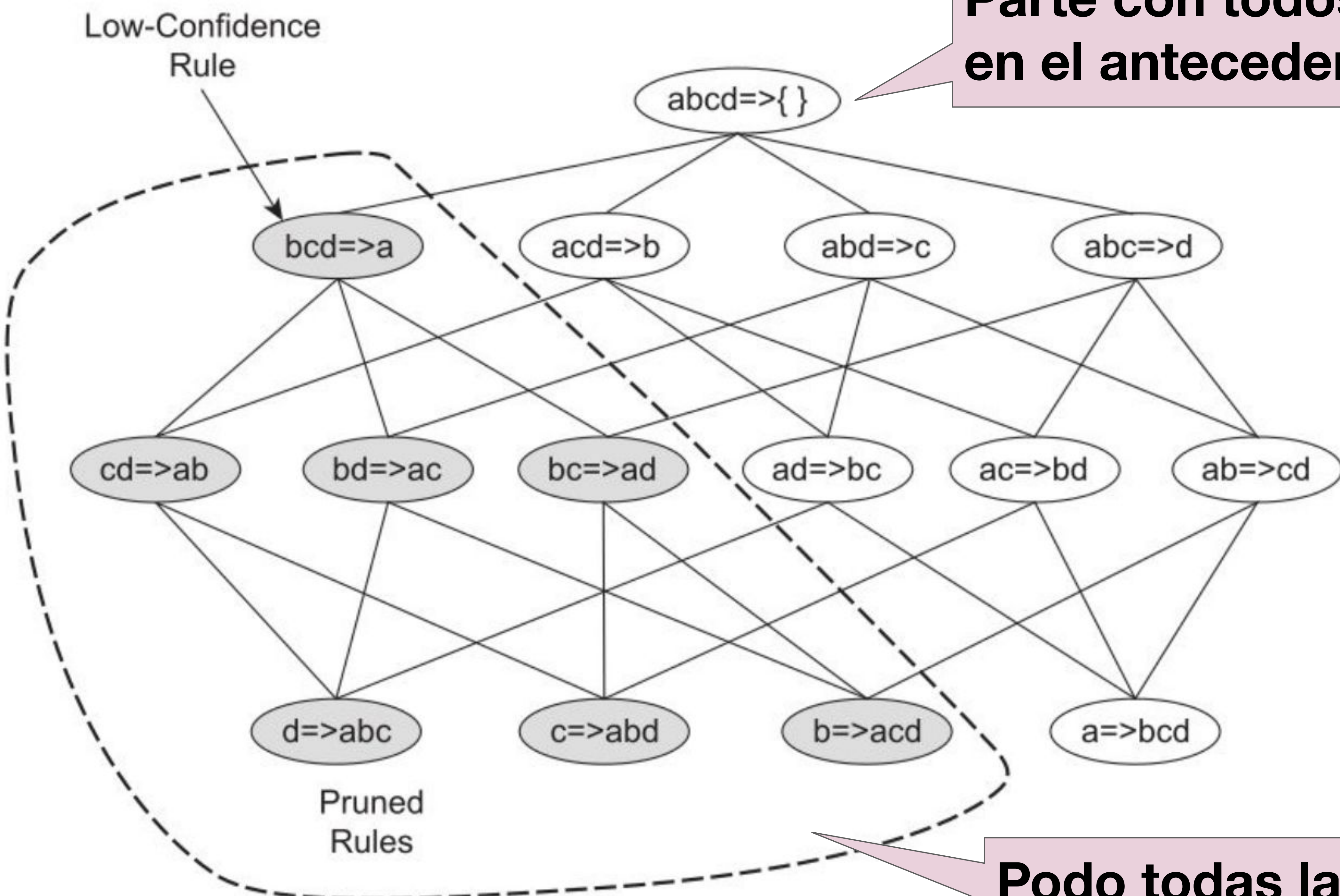
Poda basada en confianza

• La confianza no puede crecer si muevo itemsets del antecedente al consecuente.

Demostración:

- Sean dos reglas $\tilde{X} \rightarrow Y - \tilde{X}$, $X \rightarrow Y - X$, donde $\tilde{X} \subset X$.
- La confianza de estas reglas es $\sigma(Y)/\sigma(\tilde{X})$ y $\sigma(Y)/\sigma(X)$ respectivamente.
- Como $\tilde{X}$ es un subconjunto de X, $\sigma(\tilde{X}) \geq \sigma(X)$.
- Por consecuencia, la primera regla no puede tener una confianza mayor que la segunda.

Poda basada en confianza



Parte con todos los items en el antecedente

Podo todas las reglas que no cumplen con confmin

Generación de reglas eficientes en Apriori

Algorithm 5.2 Rule generation of the *Apriori* algorithm.

for each frequent k-itemset f_k, $k \geq 2$ **do**
$H_1 = \{i \mid i \in f_k\}$ {1-item consequents of the rule.}
call `ap-genrules`$(f_k, H_1.)$
end for

Algorithm 5.3 Procedure `ap-genrules`$(f_k,\ H_m)$.

$k = |f_k|$ {size of frequent itemset.}
$m = |H_m|$ {size of rule consequent.}
if $k > m + 1$ **then**
$H_{m+1} =$ candidate-gen(H_m).
$H_{m+1} =$ candidate-prune(H_{m+1}, H_m).
for each $h_{m+1} \in H_{m+1}$ **do**
$conf = \sigma(f_k)/\sigma(f_k - h_{m+1})$.
if $conf \geq minconf$ **then**
output the rule $(f_k - h_{m+1}) \longrightarrow h_{m+1}$.
else
delete h_{m+1} from H_{m+1}.
end if
end for
call ap-genrules$(f_k, H_{m+1}.)$
end if

Ejemplo Tutorial

Evaluación de Patrones

Evaluación de Patrones

- Los algoritmos de reglas de asociación tienden a producir demasiadas reglas
  - Muchas son redundantes o poco interesantes
  - Redundante si {A,B,C} → {D} y {A,B} → {D} tienen el mismo support y confidence

- Se pueden usar medidas de interés para podar/rankear los patrones derivados

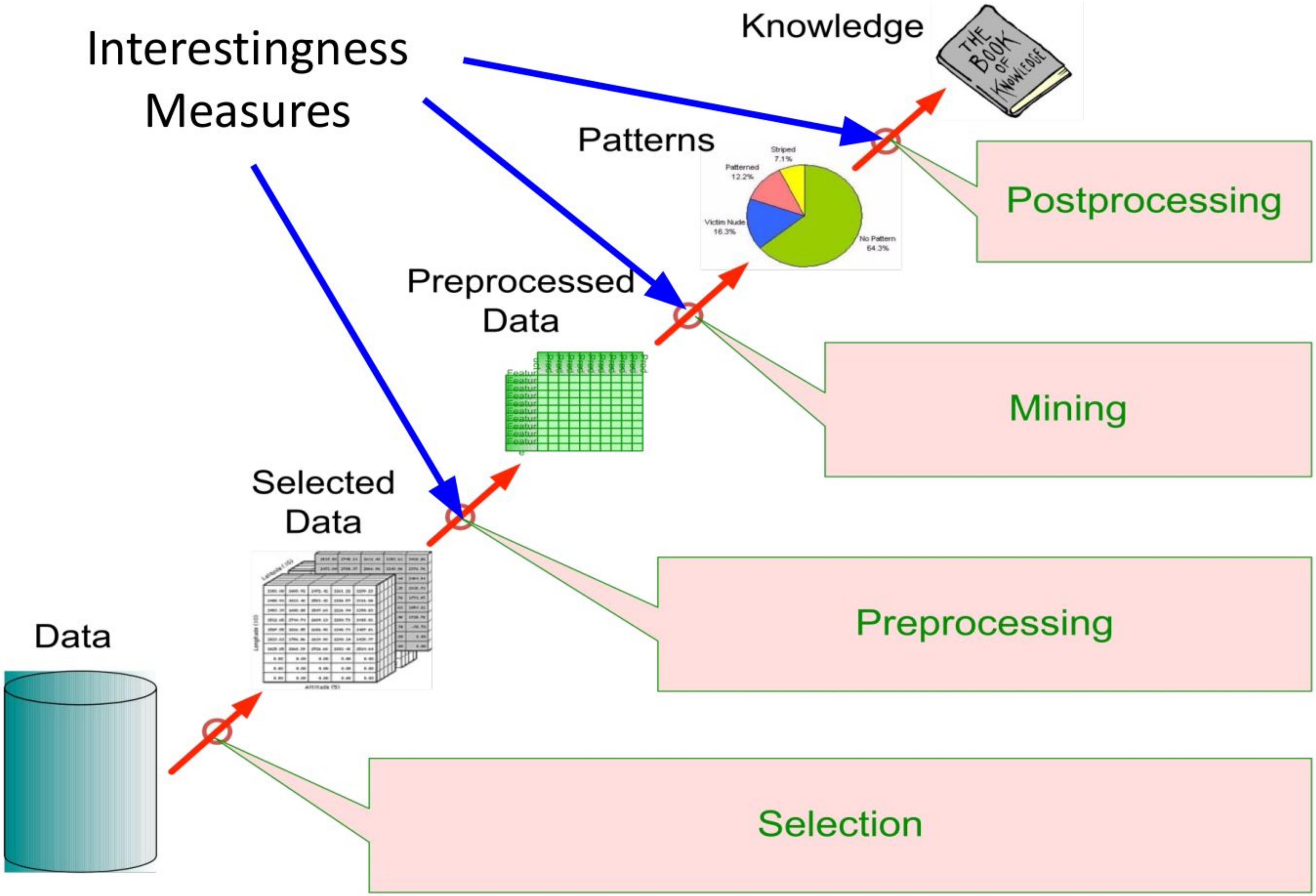
- En la formulación original de reglas de asociación, support y confidence son las únicas medidas

Medidas objetivas de interés

- Aplicamos medidas objetivas de interés.
  - Soporte y confianza son ejemplos de medidas objetivas de interés.

- Se busca descartar reglas que asocian itemsets que son independientes entre sí (estadísticamente independientes)

Uso de medidas de interés



Calculando el interés

Dada una regla X → Y, la información requerida para calcular su medida de interés se puede obtener de la tabla de contingencia

Tabla de contingencia para X → Y

	Y	$\overline{Y}$	
X	f_{11}	f_{10}	f_{1+}
$\overline{X}$	f_{01}	f_{00}	f_{o+}
	f_{+1}	f_{+0}	$\|T\|$

f_{11}: support de X e Y
f_{10}: support de X e $\overline{Y}$
f_{01}: support de $\overline{X}$ e Y
f_{00}: support de $\overline{X}$ e $\overline{Y}$

Usado para definir varias medidas

- support, confidence, lift, Gini, J-measure, etc.

Desventaja de Confidence

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

Regla de Asociación:
Tea → Coffee

Es engañoso porque aunque el confidence sea alto, saber que la persona toma te baja la probabilidad de que tome café

Support(Tea → Coffee) = 0.15
Confidence(Tea → Coffee) = P(Coffee|Tea) = 0.75
Pero P(Coffee) = 0.9

La fracción de personas que toman café independientemente si toman té es 0.9

Desventaja de Confidence

La confianza no es una métrica muy buena cuando el consecuente por si solo es infrecuente.

La confianza ignora el soporte del consecuente.

Existe otra métrica que puede ser más útil en este caso llamada Lift

Independencia Estadística

La métrica Lift se basa en el concepto de Independencia Estadística

Cuando dos cosas son independientes entre sí, la probabilidad conjunta de los eventos es igual al producto de la probabilidad marginal independiente de cada uno de los eventos.

Independencia Estadística

Población de 1000 estudiantes

- 600 estudiantes saben nadar (S)
- 700 estudiantes saben andar en bicicleta (B)
- 420 estudiantes saben nadar y andar en bicicleta (S,B)

– $P(S \wedge B) = 420/1000 = 0.42$

– $P(S) \cdot P(B) = 0.6 \cdot 0.7 = 0.42$

– $P(S \wedge B) = P(S) \cdot P(B)$ => Independencia estadística

– $P(S \wedge B) > P(S) \cdot P(B)$ => Correlación positiva

– $P(S \wedge B) < P(S) \cdot P(B)$ => Correlación negativa

Independencia Estadística

Consideran dependencia estadística
– Interest Factor o Lift

$$I(A,B) = \frac{s(A,B)}{s(A) \times s(B)} = \frac{Nf_{11}}{f_{1+}f_{+1}}.$$

Se interpreta como:

$$I(A,B) \begin{cases} = 1, & \text{if } A \text{ and } B \text{ are independent;} \\ > 1, & \text{if } A \text{ and } B \text{ are positively related;} \\ < 1, & \text{if } A \text{ and } B \text{ are negatively related.} \end{cases}$$

Ejemplo Lift/Interest

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

Regla de Asociación:
Tea → Coffee

Confidence(Tea → Coffee) = P(Coffee|Tea) = 0.75
Pero P(Coffee) = 0.9

Lift = 0.75/0.9 = 0.8333

< 1, por lo tanto está asociado negativamente

Muchas métricas

Ventajas y desventajas dependiendo de cada caso

Table 5.9. Examples of objective measures for the itemset $\{A, B\}$.

Measure (Symbol)	Definition
Correlation (ϕ)	$\frac{Nf_{11}-f_{1+}f_{+1}}{\sqrt{f_{1+}f_{+1}f_{0+}f_{+0}}}$
Odds ratio (α)	$(f_{11}f_{00})/(f_{10}f_{01})$
Kappa (κ)	$\frac{Nf_{11}+Nf_{00}-f_{1+}f_{+1}-f_{0+}f_{+0}}{N^2-f_{1+}f_{+1}-f_{0+}f_{+0}}$
Interest (I)	$(Nf_{11})/(f_{1+}f_{+1})$
Cosine (IS)	$(f_{11})/(\sqrt{f_{1+}f_{+1}})$
Piatetsky-Shapiro (PS)	$\frac{f_{11}}{N} - \frac{f_{1+}f_{+1}}{N^2}$
Collective strength (S)	$\frac{f_{11}+f_{00}}{f_{1+}f_{+1}+f_{0+}f_{+0}} \times \frac{N-f_{1+}f_{+1}-f_{0+}f_{+0}}{N-f_{11}-f_{00}}$
Jaccard (ζ)	$f_{11}/(f_{1+} + f_{+1} - f_{11})$
All-confidence (h)	$\min\left[\frac{f_{11}}{f_{1+}}, \frac{f_{11}}{f_{+1}}\right]$

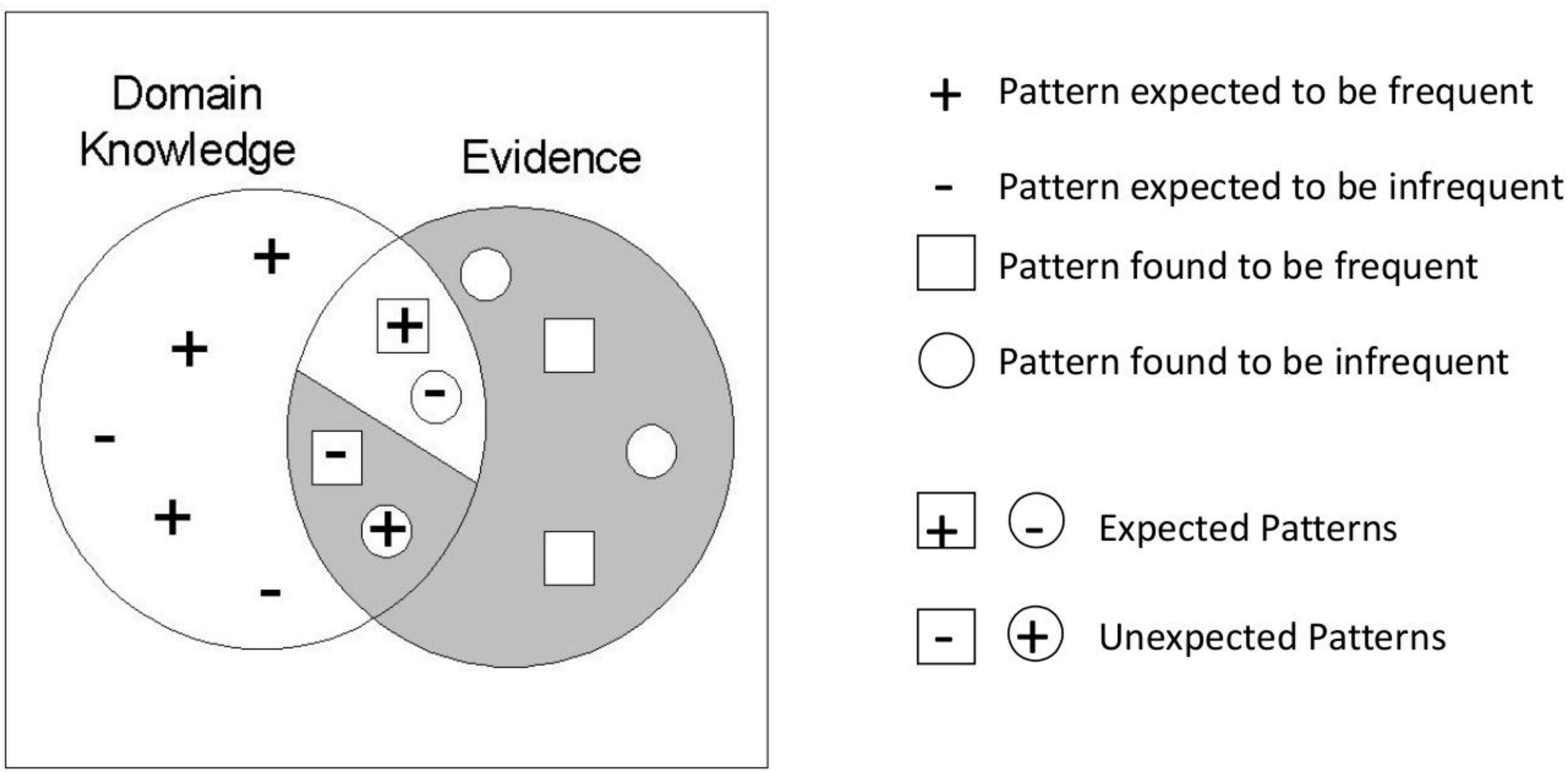
Medida de interés subjetiva

Medida objetiva:

- Rankear patrones basado en estadísticas calculadas a partir de los datos
- e.g., 21 medidas de asociación (support, confidence, Laplace, Gini, mutual information, Jaccard, etc).

Interestingness vs Unexpectedness

- Necesidad de modelar expectativas de usuario (conocimiento del dominio)
- Necesidad de combinar expectativas de los usuarios con evidencia de los datos (i.e., patrones extraídos)



dcc

CIENCIAS DE LA COMPUTACIÓN
UNIVERSIDAD DE CHILE

www.dcc.uchile.cl

/ DCCUCHILE