Reglas de asociación

Dra. Amparo López Gaona

Fac. Ciencias, UNAM
Abril 2018

Introducción

- Si fueras un vendedor y estás hablando con un cliente que compró una laptop y una cámara digital. ¿Qué le recomendarías comprar?
- El objetivo del análisis de asociación es encontrar relaciones interesantes entre artículos (productos, documentos, etc.).
- Ejemplo: relación entre compras:
  - leche, harina y huevos, frecuentemente se compran juntos. O bien,
  - Si alguien compra leche y harina a menudo también compra huevos.
- ¿Después de comprar una PC, qué más compran?

... Introducción

- Aplicaciones de encontrar relaciones:
  - Minoristas: Ubicación de productos, campañas de promoción, decisiones de surtido de mercancía, etc.
    Análisis de cesta de compra
  - Comercio electrónico: bibliotecas digitales, motores de búsqueda: Personalización
    Sistemas de recomendación, filtros colaborativos basados en artículos.
- Puede ayudar en el proceso de toma de decisiones para procesos tales como diseño de catálogos, análisis de campañas de ventas, análisis de web log (flujos de click), análisis de secuencias de ADN, clustering, clasificación, etc.

... Introducción

- Técnica propuesta por Agrawal en 1993 y ampliamente estudiada por la comunidad de BD.
- Importante porque:
  - Revela propiedades importantes e intrínsecas de los data sets.
  - Es básico para varias tareas esenciales de la minería de datos como son:
    - Análisis de asociación, correlación y causalidad.
    - Patrones secuenciales, estructurales (sub-gráficas).
    - Análisis de patrones espaciotemporal, multimedia, series de tiempo y flujos de datos.
    - Clasificación: Clasificación asociativa.
    - Análisis de clusters: clusters basados en patrones frecuentes.
    - Varias aplicaciones más.

... Introducción

- Trabaja sobre datos categóricos.
- Ejemplo típico es análisis de canasta de compra. (market basket)

- ¿Cómo puedo encontrar estas reglas de asociación?
- ¿Cuáles reglas de asociación son más interesantes?
- ¿Cómo se pueden descubrir éstas?

Formato de los datos

Para hacer un análisis de canasta de compra y obtener reglas de asociación es necesario que trabajar con bases de datos transaccionales:

TID	Artículos
1	leche, pan
2	pan, mantequilla
3	cerveza
4	leche, pan, mantequilla
5	pan, mantequilla

	leche	pan	mant.	cerveza
1	1	1	0	0
2	0	1	1	0
3	0	0	0	1
4	1	1	1	0
5	0	1	1	0

Sean

- $I = \{i_1, i_2, ..., i_n\}$ el conjunto de n atributos binarios llamados items.
- $D = \{t_1, t_2, ..., t_m\}$ el conjunto de transacciones llamado la base de datos.
- Cada transacción en D tiene un identificador único TID y un subconjunto de artículos (items) de I.

Conceptos básicos

- itemset: Conjunto de uno o más elementos.
  - $\{$`leche, pan, cereal`$\}$
- k-itemset: Conjunto de exactamente k elementos $X = \{x_1, ..., x_k\}$
- Un itemset X es frecuente si tiene un soporte mayor que un umbral dado.
- Una regla de asociación: es la representación de un patrón. Tiene la forma $X \Rightarrow Y$ con:
  - $X, Y \subseteq I$ itemsets
  - $X \cap Y = \varnothing$
  - X es el antecedente (lado izquierdo).
  - Y es el consecuente (lado derecho).
- Ejemplos:
  - `computadora` $\Rightarrow$ `sofwareDWH`
  - $\{$`Leche, pañales`$\}$ $\Rightarrow$ `cerveza`

... Conceptos básicos

Métricas de evaluación para una regla de la forma X $\Rightarrow$ Y:

- Soporte de un itemset Z es la cantidad de transacciones que lo contienen.
- Soporte de una regla $X \Rightarrow Y$

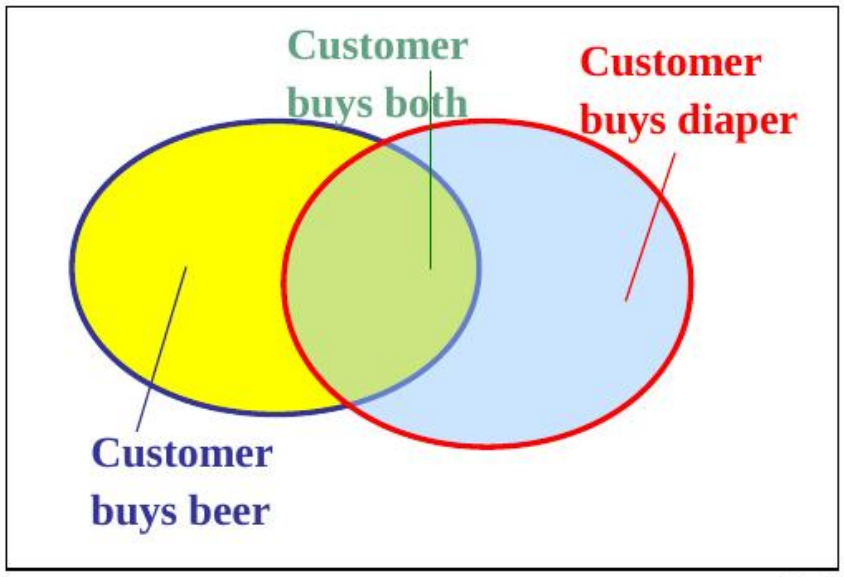
$$soporte(X \Rightarrow Y) = \frac{soporte(X \cup Y)}{N}$$

- Confianza de una regla $X \Rightarrow Y$ se define como frecuencia con que los elementos Y aparecen en transacciones que contienen a X.

$$Confianza(X \Rightarrow Y) = \frac{soporte(X \cup Y)}{soporte(X)}$$

.

... Conceptos básicos



- Reflejan la utilidad y confianza de las reglas descubiertas.
- Se considera que la regla es interesante/fuerte si satisface tanto el mínimo de soporte como de confianza.
- Los umbrales los proporciona el usuario y/o experto en el dominio. Pueden expresarse en el rango de 0 a 1.0 o bien de 0 % a 100 %

... Conceptos básicos

- Dado un conjunto de transacciones T, el problema de minado de reglas de asociación se define como la tarea de encontrar todas las reglas que tengan un soporte $\geq$ *minsop* y confianza $\geq$ *minconf* .

... Conceptos básicos (Ejemplo)

TID	DataSets Comprados
10	Cerveza, botana, pañales
20	Cerveza, café, pañales
30	Cerveza, pañales, huevos
40	Botana, huevos, leche
50	Botana, café, pañales, huevos, leche
60	Leche, huevos

- Soporte: Cerveza = 3/6=0.5, Botana = 3/6 = 0.5, Pañales = 4/6=0.66, {Cerveza, Pañales} = 3 = 50 %.
- Sean soporte mínimo = 50 % y confianza mínima = 50 %, entonces
  - Cerveza $\Rightarrow$ Pañales [soporte = 50 %, confianza = 100 %]
  - Pañales $\Rightarrow$ Cerveza (50 %, 75 %)
  - Huevos $\Rightarrow$ Leche (50 %, 75 %)
  - Cerveza $\Rightarrow$ botana (16.6 %, 30 %)
  - etc.
- Reflejan la utilidad y confianza de las reglas descubiertas.

Reglas de asociación

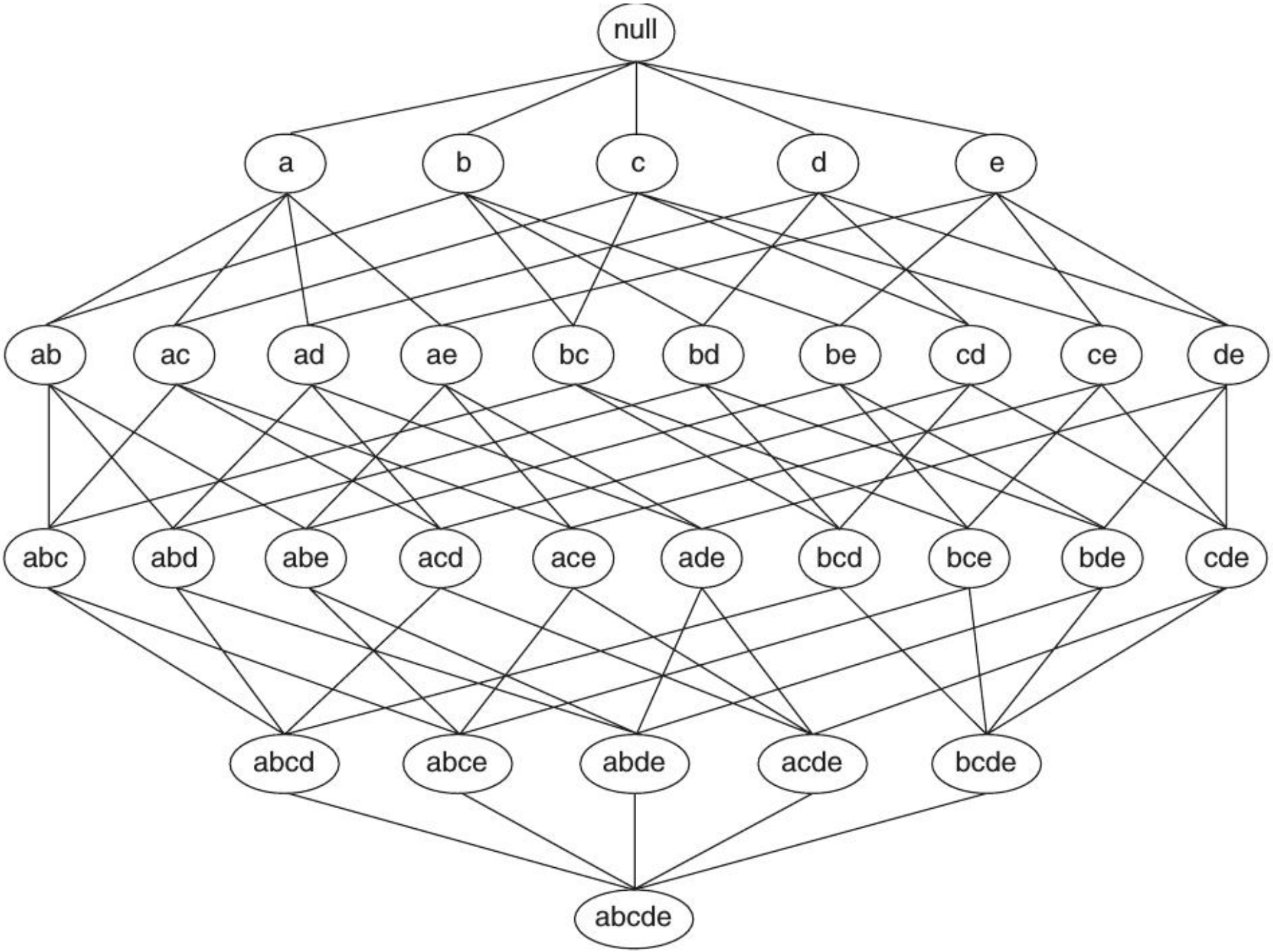
- Dado un conjunto de transacciones, encontrar reglas que ayuden a predecir la ocurrencia de un elemento basada en la ocurrencia de otros elementos en la transacción.
- En general, el proceso de minar reglas de asociación consta de dos pasos:
  - Encontrar todos los itemsets frecuentes.
  - Generar reglas de asociación fuertes de los itemsets frecuentes.
- Encontrar las reglas puede ser muy caro, computacionalmente.
- Las reglas pueden no ser interesantes.
- Métodos:
  - Fuerza bruta.
  - Apriori: enfoque de generación y prueba de un candidato.
  - FPGrowth: Un enfoque de crecimiento de patrón frecuente.
  - ECLAT: Minado de patrones frecuentes con formato de datos vertical.

Minando reglas de asociación

Fuerza bruta:

- Listar todas las posibles reglas de asociación.
- Calcular el soporte y la confianza para cada una.
- Eliminar las reglas que no satisfagan los umbrales *minsop* y *minconf*.

Computacionalmente es prohibitivo...


null
a
b
c
d
e
ab
ac
ad
ae
bc
bd
be
cd
ce
de
abc
abd
abe
acd
ace
ade
bcd
bce
bde
cde
abcd
abce
abde
acde
bcde
abcde

... Minando reglas de asociación

Reducción de conjuntos candidatos.

TID	Artículos
1	Pan, leche
2	Pan, pañales, cerveza, huevos
3	Leche, pañales, cerveza, coca
4	Pan, leche, pañales, cerveza
5	Pan, leche, pañales, coca

Ejemplos de reglas:

$$\{\text{leche, pañales}\} \Rightarrow \{\text{cerveza}\} \quad (40\,\%,\ 67\,\%)$$
$$\{\text{leche, cerveza}\} \Rightarrow \{\text{pañales}\} \quad (40\,\%,\ 100\,\%)$$
$$\{\text{pañales, cerveza}\} \Rightarrow \{\text{leche}\} \quad (40\,\%,\ 67\,\%)$$
$$\{\text{cerveza}\} \Rightarrow \{\text{leche, pañales}\} \quad (40\,\%,\ 67\,\%)$$
$$\{\text{pañales}\} \Rightarrow \{\text{leche, cerveza}\} \quad (40\,\%,\ 50\,\%)$$
$$\{\text{leche}\} \Rightarrow \{\text{pañales, cerveza}\} \quad (40\,\%,\ 50\,\%)$$

... Minando reglas de asociación

Observaciones:

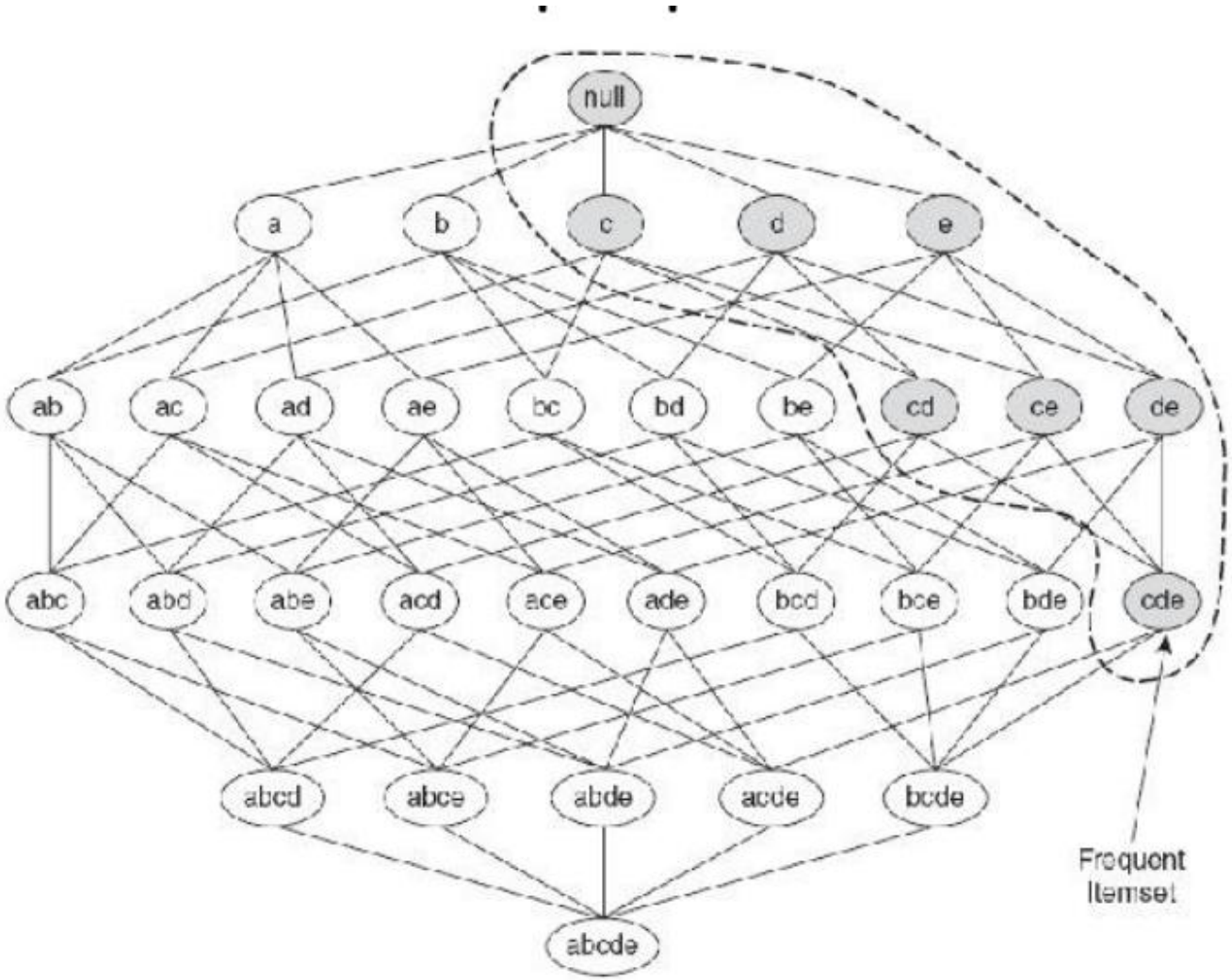
- Todas las reglas son particiones binarias del mismo itemset: $\{$`leche, pañales, cerveza`$\}$.
- Las reglas generadas de un itemset tienen igual soporte aunque pueden tener diferente confianza.
- Así que se pueden desacoplar los requerimientos de soporte y confianza.

Enfoque de dos pasos para el minado:

- Generación de los itemset frecuentes. (soporte $\geq$ minsop).
- Generación de reglas de alta confianza para cada itemset frecuente. (Cada regla es una partición binaria de un itemset fecuente). Este paso sigue siendo computacionalmente caro.

... Minando reglas de asociación

- Si un itemset es frecuente entonces todos sus subconjuntos, no vacíos, también son frecuentes. **Principio Apriori**.



... Principio Apriori

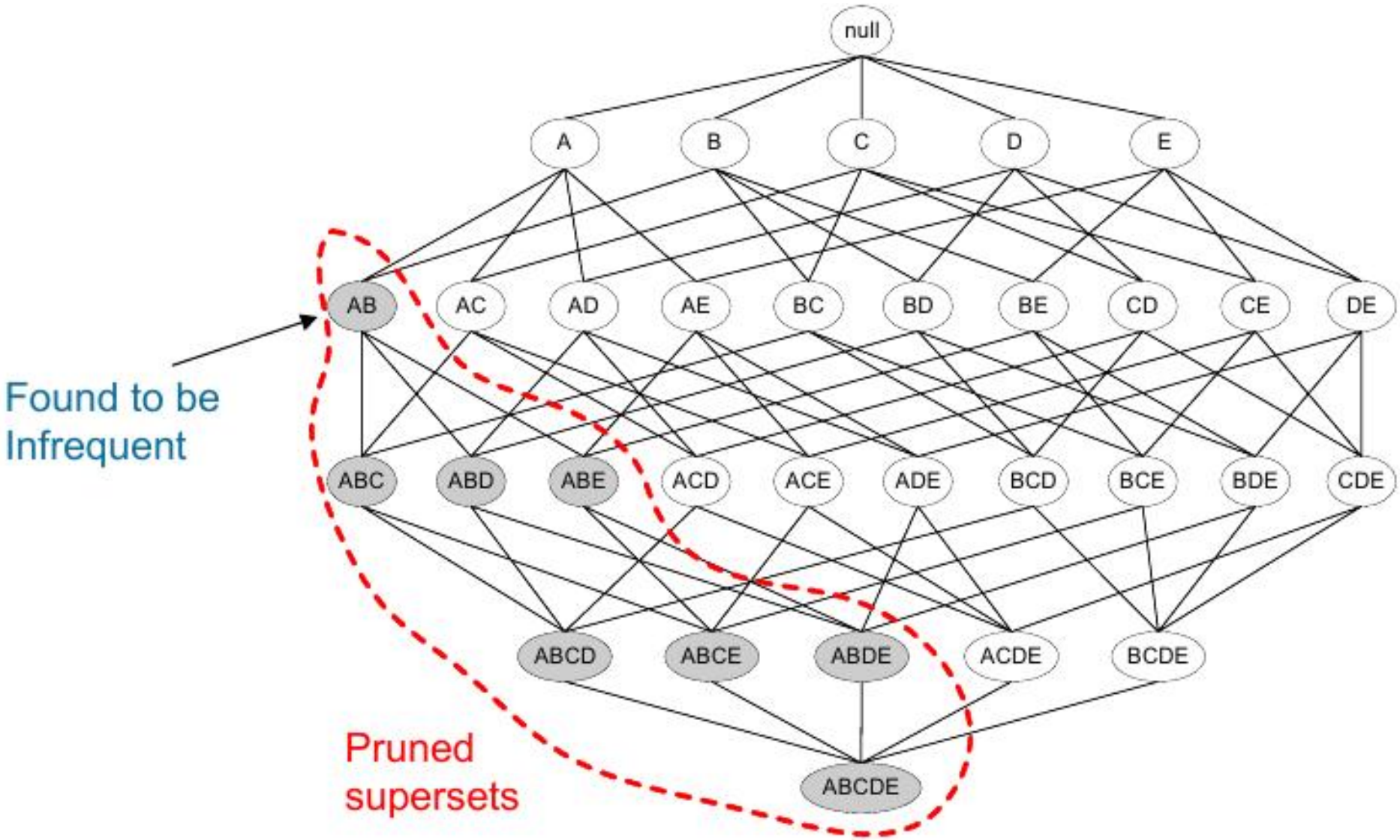
El principio Apriori lleva a la siguiente propiedad del soporte:

- El soporte de un itemset nunca excede el soporte de sus subconjuntos.

$$\forall X, Y : (X \subseteq Y) \Rightarrow soporte(X) \leq soporte(Y)$$

- Si un itemset es infrecuente, todos sus superconjuntos lo son también.
- Se puede podar el árbol de subconjuntos una vez que se encuentra un conjunto infrecuente.

... Principio Apriori



... Minando reglas de asociación usando apriori

Con soporte mínimo de 60 % = 3

TID	Artículos
1	Pan, leche
2	Pan, pañales, cerveza, huevos
3	Leche, pañales, cerveza, coca
4	Pan, leche, pañales, cerveza
5	pan, leche, pañales, coca

$\longrightarrow$

Item	Frec
Pan	4
Coca	**2**
Leche	4
Cerveza	3
Pañales	4
Huevos	**1**

$\longrightarrow$

Item	Frec
{Pan, Leche}	3
{**Pan, Cerveza** }	**2**
{Pan, Pañales }	3
{**Leche, Cerveza** }	**2**
{Leche, Pañales }	3
{Cerveza, Pañales }	3

$\longrightarrow$

Ítem	Frec
{Pan, Leche, Pañales}	3

- Con todos los conjuntos considerados = $C_1^6 + C_2^6 + C_3^6 = 41$
- Podando tomando como base el soporte = 6+6+1 = 13

Algoritmo Apriori (Pseudo código)

- Sean:
  C_k: conjunto de k-itemsets candidatos
  F_k: conjunto de k-itemsets frecuentes

- K = 1,
- Generar $F_1 = \{1\text{-itemset frecuentes}\}$
  $F_1 = \{i | i \in I \wedge \sigma(\{i\}) \geq minsup\}$
- Repetir hasta que no se identifiquen itemsets frecuentes:
  - Generar C_{k+1} , a partir de F_k.
  - Calcular el soporte para cada candidato en C_{k+1} recorriendo la BD.
  - Eliminar candidatos que son infrecuentes, dejando en F_{k+1} sólo los frecuentes.
  - k = k+1
- Regresar F_k

Generación de reglas de asociación

Con mínimo soporte = 2:

TID	id_ítem
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

$\longrightarrow$

Itemset	cont
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

$\longrightarrow$

Itemset	cont
{I1,I2}	4
{I1,I3}	4
{I1,I5}	2
{I2,I3}	4
{I2,I4}	2
{I2,I5}	2

$\longrightarrow$

Itemset	cont
{I1,I2,I3}	2
{I1,I2,I5}	2

Reglas:

$I1 \wedge I2 \Rightarrow I5$, confianza $= 2/4 = 50\,\%$

$I1 \wedge I5 \Rightarrow I2$, confianza $= 2/2 = 100\,\%$

$I2 \wedge I5 \Rightarrow I1$, confianza $= 2/2 = 100\,\%$

$I1 \Rightarrow I2 \wedge I5$, confianza $= 2/6 = 33\,\%$

$I2 \Rightarrow I1 \wedge I5$, confianza $= 2/7 = 29\,\%$

$I5 \Rightarrow I1 \wedge I2$, confianza $= 2/2 = 100\,\%$

Minado de itemsets frecuentes en formato vertical

- Datos en formato horizontal:

TID	id_ítem
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

- Datos en formato vertical:

itemset	TID-set
I1	{T100, T400, T500, T700, T800, T900}
I2	{T100, T200, T300, T400, T600, T800, T900}
I3	{T300, T500, T600, T700, T800, T900 }
I4	{T200, T400}
I5	{T100, T800}

... Minado de itemsets frecuentes en formato vertical

- Algoritmo Eclat (*Equivalence Class Transformation*).
- El minado se hace intersectando los conjuntos de TID sets de cada par de artículos frecuentes.
- En el ejemplo, si el mínimo soporte es 2 se tiene que cada artículo es frecuente y por lo tanto hay 10 intersecciones en total que llevan a ocho 2-itemsets.

itemset	TID-set
{I1,I2}	{T100, T400, T800, T900}
{I1,I3}	{T500, T700, T800, T900}
{I1,I4}	{T400 }
{I1,I5}	{T100, T800}
{I2,I3}	{T300, T600, T800, T900}
{I2,I4}	{T200, T400}
{I2,I5}	{T100,T800}
{I3,I5}	{T800}

Minado de itemsets frecuentes en formato vertical

Basados en el principio Apriori se tiene un 3-itemset es candidato sólo si cada uno de sus 2-itemsets es frecuente, de ahí que los 3-itemsets posibles sólo son:

itemset	TID-set
{I1,I2,I3}	{T800, T900}
{I1,I2,I5}	{T100, T800}

Minado de itemsets frecuentes en formato vertical

En resumen:

- Transformar los datos de formato horizontal a vertical. (en una sola pasada a la BD)
- El soporte de cada itemset es la cardinalidad del TID-set.
- Empezando con k=1, se usan los k-itemsets frecuentes para construir (k+1)-itemsets utilizando el principio Apriori.
  - Calcular la intersección de los TID-sets de los k-itemsets frecuentes.
- El proceso continua incrementando k en 1 hasta que no haya conjuntos frecuentes.

Ventajas:

- Se utiliza el principio Apriori para ir eliminando itemsets.
- No se recorre la BD varias veces.

Métodos de evaluación de los patrones

- Medidas objetivas:
  - soporte; y
  - confianza.
- Medidas subjetivas:
  Una regla (patrón) es interesante si:
  - es *inesperado* o
  - *accionable*.

... Métodos de evaluación de los patrones

¿Todas las reglas fuertes, descubiertas, son suficientemente interesantes para presentarlas al usuario?

- `pañales => cerveza (c=0.9)`
  90 % de los clientes que compran pañales, también compran cerveza.
- Parece interesante.
- La confianza es sólo una estimación de la probabilidad condicional de que suceda B dado A, A $\Rightarrow$ B.
- La confianza de una regla puede ser engañosa.
- Si se sabe que el 90 % de los clientes compra cerveza, la regla no resulta interesante.

... Métodos de evaluación de los patrones

- **lift** es una medida de correlación. Se calcula aplicando la siguiente fórmula:

$$lift_{A \Rightarrow B} = \frac{confianza(A \Rightarrow B)}{soporte(B)} = \frac{P(A \cup B)}{P(A)P(B)}$$

.

- Ejemplo: Si se sabe que la mayoría de las personas compran pan o leche. Es esperable encontrar muchas transacciones con ambos productos.
- Si `lift(leche -- > pan)` es mayor que 1, implica que los dos productos se encuentran en la misma compra más de lo esperado si fuera una casualidad.
- Un valor grande de `lift` es un fuerte indicador que una regla es importante, y refleja una conexión real entre artículos.

Minado de reglas de asociación en R

En el paquete `arules` se tiene implementación para los algoritmos.

- Apriori
  - Algoritmo de búsqueda primero a lo ancho para contar y así encontrar itemsets frecuentes y luego derivar reglas a partir de ellos.
  - Función `apriori()`.
- ECLAT
  - Encuentra itemsets frecuentes haciendo una búsqueda primero en profundidad e intersectando conjuntos en lugar de contar.
  - Función `eclat()` en el mismo paquete.

En el paquete `arulesViz` se tienen funciones para visualizaciones de las reglas e itemsets frecuentes.

La función apriori

`apriori(datos, parameter = NULL, appearance = NULL, control = NULL)`

- `datos`. Objeto de la clase transactions por ejemplo una matriz binaria o un `data.frame`.
- `parameter`. Objeto de la clase `APparameter` o una lista con nombre. El comportamiento por omisión es minar reglas con soporte mínimo de 0.1 y confianza de 0.8, máximo 10 elementos de longitud y un tiempo máximo de 5 segs. para verificar subconjuntos.
- `appearance`. Con este argumento pueden aplicarse restricciones sobre los items que aparecerán en las reglas encontradas. Por omisión no se restringe.
- `control`. Controla el rendimiento del algoritmo de minado (si los items se ordenan, reporte del progreso, etc.)

Ejemplos

Ejemplos prácticos.

Titanic (Conocimiento y preprocesamiento de datos)

Suponer que se tiene información de los pasajeros que viajaban en el Titanic, como sigue:

```
> install.packages("arules")
> library(arules)
> str(Titanic)
 table [1:4, 1:2, 1:2, 1:2] 0 0 35 0 0 0 17 0 118 154 ...
 - attr(*, "dimnames")=List of 4
  ..$ Class   : chr [1:4] "1st" "2nd" "3rd" "Crew"
  ..$ Sex     : chr [1:2] "Male" "Female"
  ..$ Age     : chr [1:2] "Child" "Adult"
  ..$ Survived: chr [1:2] "No" "Yes"
```

... Titanic (Conocimiento y preprocesamiento de datos)

```
> df <-as.data.frame(Titanic)
>
> head(df)
  Class    Sex   Age Survived Freq
1   1st   Male Child       No    0
2   2nd   Male Child       No    0
3   3rd   Male Child       No   35
4  Crew   Male Child       No    0
5   1st Female Child       No    0
6   2nd Female Child       No    0

> titanic <- NULL
> for(i in 1:4) {
   titanic <- cbind(titanic, rep(as.character(df[,i]), df$Freq))
}
> titanic <- as.data.frame(titanic)
> names(titanic) <- names(df)[1:4]
```

... Titanic (Conocimiento y preprocesamiento de datos)

```
> df <-as.data.frame(Titanic)
>
> head(df)
  Class    Sex   Age Survived Freq
1   1st   Male Child       No    0
2   2nd   Male Child       No    0
3   3rd   Male Child       No   35
4  Crew   Male Child       No    0
5   1st Female Child       No    0
6   2nd Female Child       No    0

> titanic <- NULL
> for(i in 1:4) {
   titanic <- cbind(titanic, rep(as.character(df[,i]), df$Freq))
}
> titanic <- as.data.frame(titanic)
> names(titanic) <- names(df)[1:4]
```

... Titanic (Conocimiento y preprocesamiento de datos)

```
> df <-as.data.frame(Titanic)
>
> head(df)
  Class    Sex   Age Survived Freq
1   1st   Male Child       No    0
2   2nd   Male Child       No    0
3   3rd   Male Child       No   35
4  Crew   Male Child       No    0
5   1st Female Child       No    0
6   2nd Female Child       No    0

> titanic <- NULL
> for(i in 1:4) {
   titanic <- cbind(titanic, rep(as.character(df[,i]), df$Freq))
}
> titanic <- as.data.frame(titanic)
> names(titanic) <- names(df)[1:4]
```

... Titanic (Conocimiento y preprocesamiento de datos)

```
> dim(titanic)
[1] 2201    4

> str(titanic)
'data.frame': 2201 obs. of  4 variables:
 $ Class   : Factor w/ 4 levels "1st","2nd","3rd",..: 3 3 3 3 3 3 3
 $ Sex     : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 2
 $ Age     : Factor w/ 2 levels "Adult","Child": 2 2 2 2 2 2 2 2 2 2
 $ Survived: Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...

> head(titanic)
  Class  Sex   Age Survived
1   3rd Male Child       No
2   3rd Male Child       No
3   3rd Male Child       No
4   3rd Male Child       No
5   3rd Male Child       No
6   3rd Male Child       No
```

... Titanic (Conocimiento y preprocesamiento de datos)

```
> summary(titanic)
  Class          Sex          Age        Survived
 1st :325   Female: 470   Adult:2092   No :1490
 2nd :285   Male  :1731   Child: 109   Yes: 711
 3rd :706
 Crew:885
```

... Titanic: Generación de reglas (valores por omisión)

```
> rules.all <- apriori(titanic)
parameter specification:
  confidence minval smax  arem aval  originalSupport support minlen
  0.8        0.1    1     none FALSE TRUE            0.1     1
  maxlen target  ext
  10     rules   FALSE

....

apriori - find association rules with the apriori algorithm
version 4.21 (2004.05.09)        (c) 1996-2004 Christian Borgelt
set item appearances ... [0 item(s)] done [0.00s].
set transactions ... [10 item(s), 2201 transaction(s)] done [0.00s]
sorting and recoding items ... [9 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [27 rule(s)] done [0.00s].
creating S4 object ... done [0.00s].
```

... Titanic: Mostrar reglas

```
> rules.all
set of 27 rules
> inspect(rules.all)
   lhs                     rhs                  support   confidence lift
1  {}                   => {Age=Adult}          0.9504771 0.9504771  1.0000000
2  {Class=2nd}          => {Age=Adult}          0.1185825 0.9157895  0.9635051
3  {Class=1st}          => {Age=Adult}          0.1449341 0.9815385  1.0326798
4  {Sex=Female}         => {Age=Adult}          0.1930940 0.9042553  0.9513700
5  {Class=3rd}          => {Age=Adult}          0.2848705 0.8881020  0.9343750
6  {Survived=Yes}    => {Age=Adult}             0.2971377 0.9198312  0.9677574
7  {Class=Crew}         => {Sex=Male}           0.3916402 0.9740113  1.2384742
8  {Class=Crew}         => {Age=Adult}          0.4020900 1.0000000  1.0521033
9  {Survived=No}        => {Sex=Male}           0.6197183 0.9154362  1.1639949
10 {Survived=No}        => {Age=Adult}          0.6533394 0.9651007  1.0153856
11 {Sex=Male}           => {Age=Adult}          0.7573830 0.9630272  1.0132040
12 {Sex=Female,
     Survived=Yes}   => {Age=Adult}             0.1435711 0.9186047  0.9664669
```

```
13 {Class=3rd,
    Sex=Male}      => {Survived=No} 0.1917310 0.8274510  1.2222950
14 {Class=3rd,
    Survived=No}   => {Age=Adult}   0.2162653 0.9015152 0.9484870
15 {Class=3rd,
    Sex=Male}      => {Age=Adult}   0.2099046 0.9058824 0.9530818
16 {Sex=Male,
    Survived=Yes} => {Age=Adult}   0.1535666 0.9209809 0.9689670
17 {Class=Crew,
    Survived=No}   => {Sex=Male}    0.3044071 0.9955423 1.2658514
18 {Class=Crew,
    Survived=No}   => {Age=Adult}   0.3057701 1.0000000 1.0521033
19 {Class=Crew,
    Sex=Male}      => {Age=Adult}   0.3916402 1.0000000 1.0521033
20 {Class=Crew,
    Age=Adult}     => {Sex=Male}    0.3916402 0.9740113 1.2384742
21 {Sex=Male,
    Survived=No}   => {Age=Adult}   0.6038164 0.9743402 1.0251065
22 {Age=Adult,
    Survived=No}   => {Sex=Male}    0.6038164 0.9242003 1.1751385
23 {Class=3rd,
```

... Titanic: Reglas acerca de sobrevivientes

```
> rules <- apriori(titanic, control = list(verbose=F),
        parameter = list(minlen=2, supp=0.005, conf=0.8),
        appearance = list(rhs=c("Survived=No", "Survived=Yes"),
            default="lhs"))
> quality(rules) <- round(quality(rules), digits=3)
> rules.sorted <- sort(rules, by="lift")
> inspect(rules.sorted)
   lhs                 rhs                  support   confidence  lift
1  {Class=2nd,
    Age=Child}      => {Survived=Yes}       0.011     1.000       3.096
2  {Class=2nd,
    Sex=Female,
    Age=Child}      => {Survived=Yes}       0.006     1.000       3.096
3  {Class=1st,
    Sex=Female}     => {Survived=Yes}       0.064     0.972       3.010
4  {Class=1st,
    Sex=Female,
    Age=Adult}      => {Survived=Yes}       0.064     0.972       3.010
```

```
5  {Class=2nd,
    Sex=Female} => {Survived=Yes} 0.042 0.877 2.716
6  {Class=Crew,
    Sex=Female} => {Survived=Yes} 0.009 0.870 2.692
7  {Class=Crew,
    Sex=Female,
    Age=Adult}  => {Survived=Yes} 0.009 0.870 2.692
8  {Class=2nd,
    Sex=Female,
    Age=Adult}  => {Survived=Yes} 0.036 0.860 2.663
9  {Class=2nd,
    Sex=Male,
    Age=Adult}  => {Survived=No}  0.070 0.917 1.354
10 {Class=2nd,
    Sex=Male}   => {Survived=No}  0.070 0.860 1.271
11 {Class=3rd,
    Sex=Male,
    Age=Adult}  => {Survived=No}  0.176 0.838 1.237
12 {Class=3rd,
    Sex=Male}   => {Survived=No}  0.192 0.827 1.222
```

Titanic: Eliminación de redundancia

```
> rules <- sort(rules, by="confidence")
> inspect(rules)
   lhs                                      rhs                 suppor confi
1  {Class=2nd,Age=Child}                    => {Survived=Yes}   0.011  1.000
2  {Class=2nd,Sex=Female,Age=Child}         => {Survived=Yes}   0.006  1.000
3  {Class=1st,Sex=Female}                   => {Survived=Yes}   0.064  0.972
4  {Class=1st,Sex=Female,Age=Adult}         => {Survived=Yes}   0.064  0.972
5  {Class=2nd,Sex=Male,Age=Adult}           => {Survived=No}    0.070  0.917
6  {Class=2nd,Sex=Female}                   => {Survived=Yes}   0.042  0.877
7  {Class=Crew,Sex=Female}                  => {Survived=Yes}   0.009  0.870
8  {Class=Crew,Sex=Female,Age=Adult}        => {Survived=Yes}   0.009  0.870
9  {Class=2nd,Sex=Male}                     => {Survived=No}    0.070  0.860
10 {Class=2nd,Sex=Female,Age=Adult}         => {Survived=Yes}   0.036  0.860
11 {Class=3rd,Sex=Male,Age=Adult}           => {Survived=No}    0.176  0.838
12 {Class=3rd,Sex=Male}                     => {Survived=No}    0.192  0.827
```

$X \to Y$ es **redundante** si existe $X' \subset X$ tal que $\text{conf}(X' \to Y) \geq \text{conf}(X \to Y)$

... Titanic: Eliminación de redundancia

```
> is.redundant(rules)
[1] FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE  TRUE FALS

> quality(rules) <- round(quality(rules), digits=3)
> inspect(rules[is.redundant(rules)])
  lhs                                      rhs              supp. conf. li
1 {Class=2nd,Sex=Female,Age=Child}  => {Survived=Yes} 0.006 1.000 3
2 {Class=1st,Sex=Female,Age=Adult}  => {Survived=Yes} 0.064 0.972 3
3 {Class=Crew,Sex=Female,Age=Adult} => {Survived=Yes} 0.009 0.870 2
4 {Class=2nd,Sex=Female,Age=Adult}  => {Survived=Yes} 0.036 0.860 2
```

... Titanic: Eliminación de redundancia

```
> quality(rules) <- round(quality(rules), digits=3)
> inspect(rules[!is.redundant(rules)])
  lhs                                rhs                supp.  conf.   li
1 {Class=2nd,Age=Child}           => {Survived=Yes} 0.011  1.000  3.
2 {Class=1st,Sex=Female}          => {Survived=Yes} 0.064  0.972  3.
3 {Class=2nd,Sex=Male,Age=Adult}  => {Survived=No}  0.070  0.917  1.
4 {Class=2nd,Sex=Female}          => {Survived=Yes} 0.042  0.877  2.
5 {Class=Crew,Sex=Female}         => {Survived=Yes} 0.009  0.870  2.
6 {Class=2nd,Sex=Male}            => {Survived=No}  0.070  0.860  1.
7 {Class=3rd,Sex=Male,Age=Adult}  => {Survived=No}  0.176  0.838  1.
8 {Class=3rd,Sex=Male}            => {Survived=No}  0.192  0.827  1.
```

... Titanic: Eliminación de redundancia

```
> # elimina las reglas redundantes
> rules <- rules[!is.redundant(rules)]

> inspect(rules)
   lhs                                  rhs              supp.  conf.
1  {Class=2nd,Age=Child}             => {Survived=Yes}   0.011  1.000
2  {Class=1st,Sex=Female}            => {Survived=Yes}   0.064  0.972
3  {Class=2nd,Sex=Female}            => {Survived=Yes}   0.042  0.877
4  {Class=Crew,Sex=Female}           => {Survived=Yes}   0.009  0.870
5  {Class=2nd,Sex=Male,Age=Adult}    => {Survived=No}    0.070  0.917
6  {Class=2nd,Sex=Male}              => {Survived=No}    0.070  0.860
7  {Class=3rd,Sex=Male,Age=Adult}    => {Survived=No}    0.176  0.838
8  {Class=3rd,Sex=Male}              => {Survived=No}    0.192  0.827
```

Titanic: Interpretación de las reglas

¿Qué podemos concluir de la regla:

```
 1  {Class=2nd,
     Age=Child}   => {Survived=Yes}  0.011      1.000      3.096
```

```
>rules.ninios <- apriori(titanic,
              parameter = list(minlen=2, supp=0.002, conf=0.2),
              appearance = list(rhs=c("Survived=Yes"),
                                lhs=c("Age=Child"),
                                default="none"),
              control = list(verbose=T))
```

```
> inspect(rules.ninios)
  lhs             rhs              support    confidence lift     coun
1 {Age=Child} => {Survived=Yes} 0.02589732 0.5229358  1.618821 57
```

... Interpretación de las reglas

```
> rules.ninios <- apriori(titanic,
                parameter = list(minlen=2, supp=0.002, conf=0.2),
                appearance = list(rhs=c("Survived=Yes"),
                                  lhs=c("Age=Child", "Class=1st",
                                        "Class=2nd","Class=3rd"),
                                  default="none"))

> quality(rules.ninios) <- round(quality(rules.ninios), digits=3)
> inspect(rules.ninios)
  lhs                       rhs                 supp.  conf.  lift   coun
1 {Age=Child}            => {Survived=Yes} 0.026  0.523  1.619   57
2 {Class=2nd}            => {Survived=Yes} 0.054  0.414  1.282  118
3 {Class=1st}            => {Survived=Yes} 0.092  0.625  1.934  203
4 {Class=3rd}            => {Survived=Yes} 0.081  0.252  0.780  178
5 {Class=2nd,Age=Child}  => {Survived=Yes} 0.011  1.000  3.096   24
6 {Class=1st,Age=Child}  => {Survived=Yes} 0.003  1.000  3.096    6
7 {Class=3rd,Age=Child}  => {Survived=Yes} 0.012  0.342  1.058   27
```

... Interpretación de las reglas

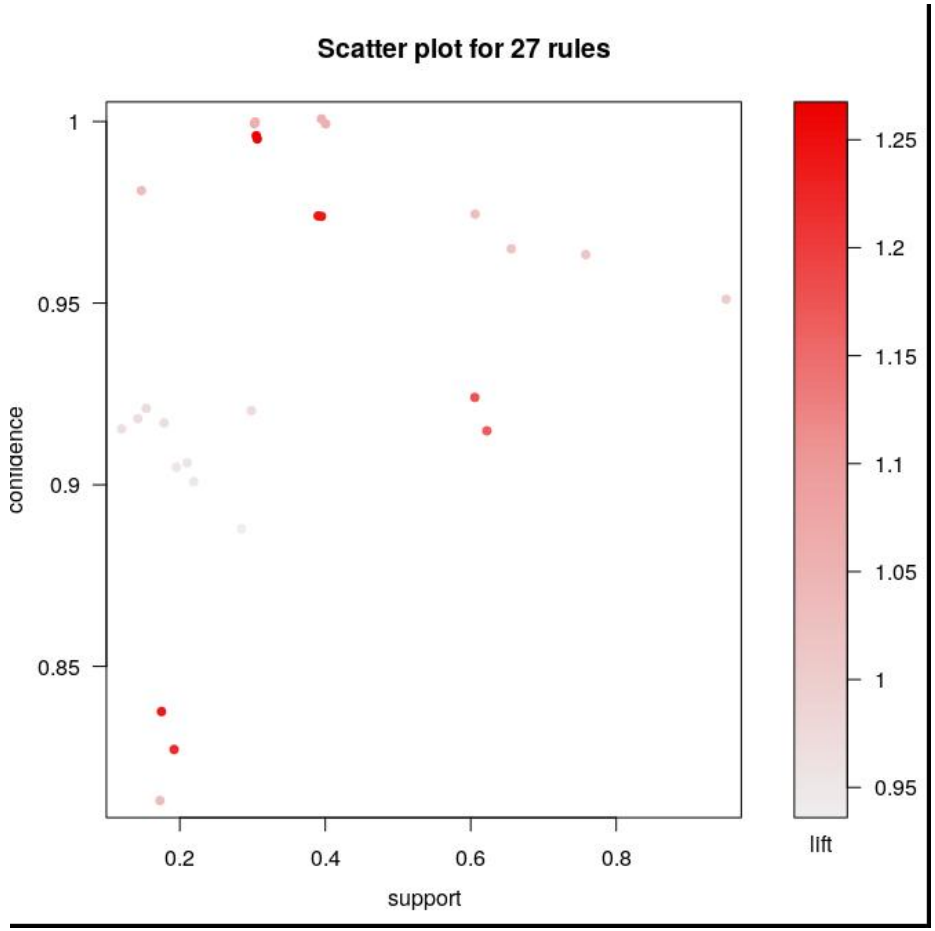
```
> rules <- apriori(titanic,
+              parameter = list(minlen=3, supp=0.002, conf=0.2),
+              appearance = list(rhs=c("Survived=Yes"),
+                                lhs=c("Class=1st", "Class=2nd",
                                       "Class=3rd",
+                                     "Age=Child", "Age=Adult"),
+                                default="none"),
+              control = list(verbose=F))
> rules.sorted <- sort(rules, by="confidence")
> quality(rules.sorted) <- round(quality(rules.sorted), digits=3)
```

... Interpretación de las reglas

```
> inspect(rules.sorted)
    lhs                              rhs                  supp.  conf.  lift  cou
[1] {Class=2nd,Age=Child}         => {Survived=Yes}       0.011  1.000  3.096  24
[2] {Class=1st,Age=Child}         => {Survived=Yes}       0.003  1.000  3.096   6
[3] {Class=1st,Age=Adult}         => {Survived=Yes}       0.090  0.618  1.912 197
[4] {Class=2nd,Age=Adult}         => {Survived=Yes}       0.043  0.360  1.115  94
[5] {Class=3rd,Age=Child}         => {Survived=Yes}       0.012  0.342  1.058  27
[6] {Class=3rd,Age=Adult}         => {Survived=Yes}       0.069  0.241  0.746 151
```

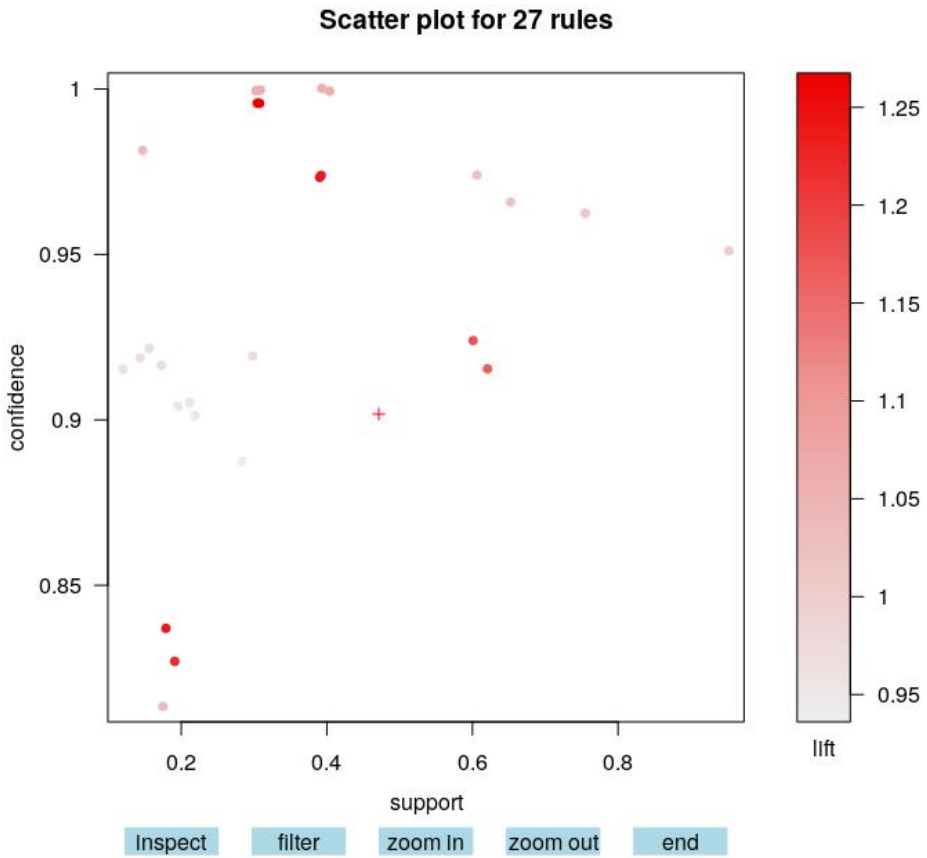
Visualización de reglas

```
install.packages("arulesViz")
library(arulesViz)
plot(rules.all)
```

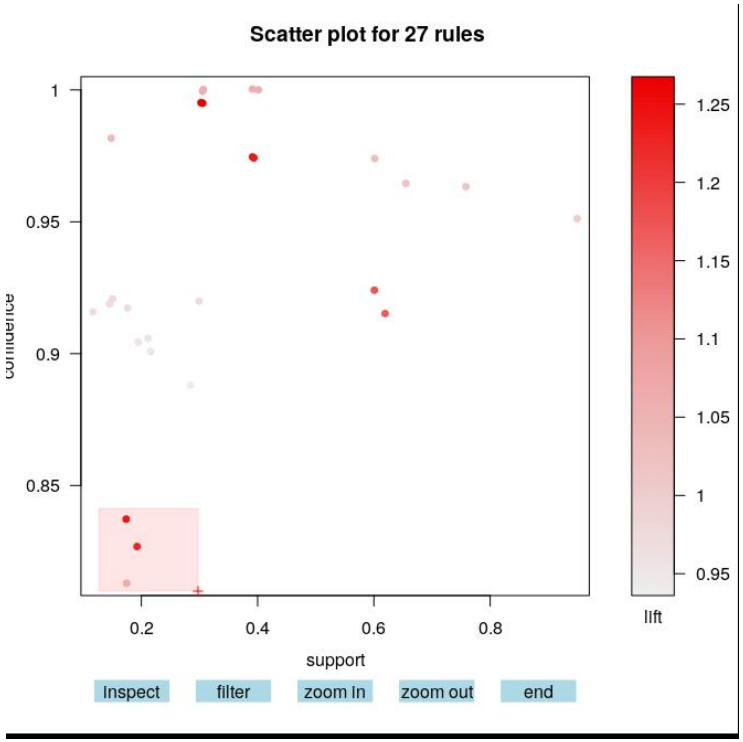


Visualización de reglas

```
  > sel <- plot(rules.all, engine="interactive")
To reduce overplotting, jitter is added! Use jitter = 0 to prevent j
Interactive mode.
Select a region with two clicks!
```



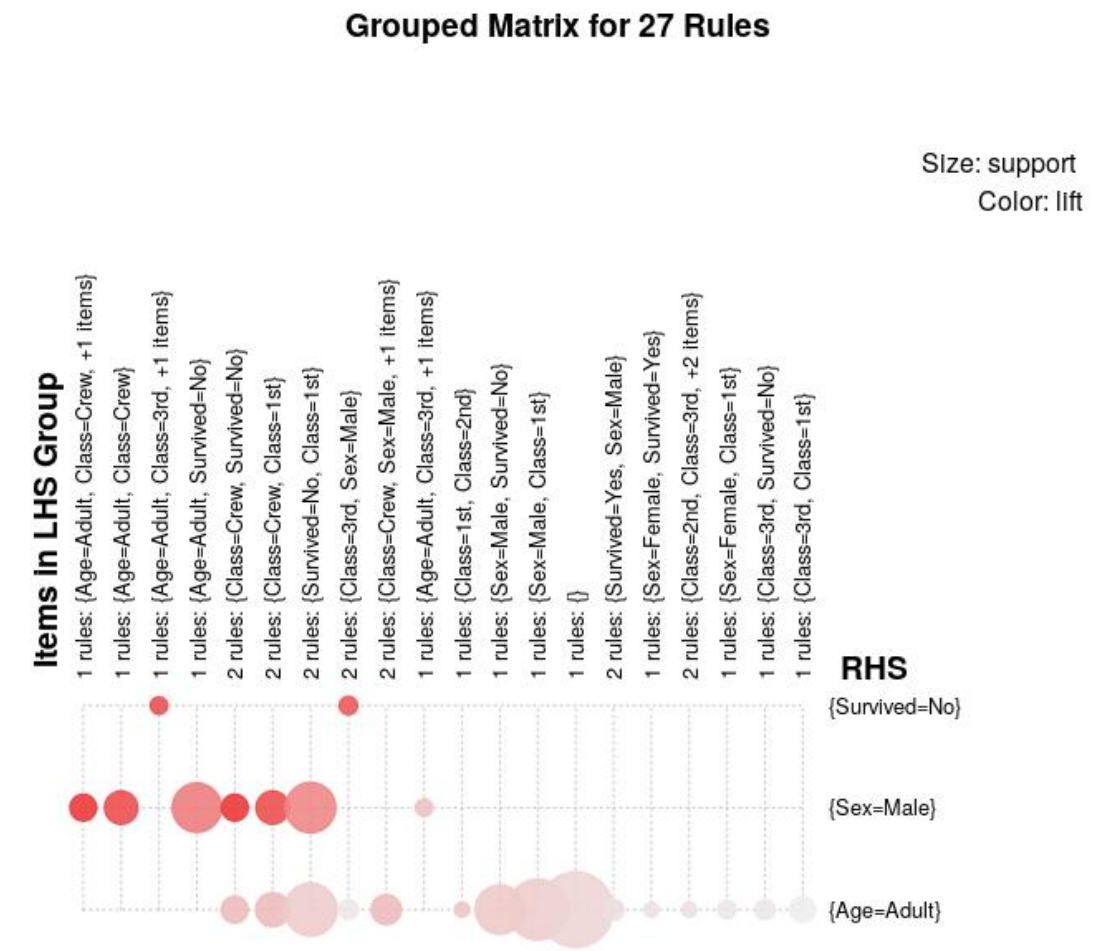
Visualización de reglas




```
Number of rules selected: 3
    lhs                                    rhs              support    co
[1] {Class=3rd,Sex=Male,Age=Adult}     => {Survived=No} 0.1758292 0.8
[2] {Class=3rd,Sex=Male}               => {Survived=No} 0.1917310 0.8
[3] {Class=3rd,Age=Adult,Survived=No} => {Sex=Male}    0.1758292 0.8
```

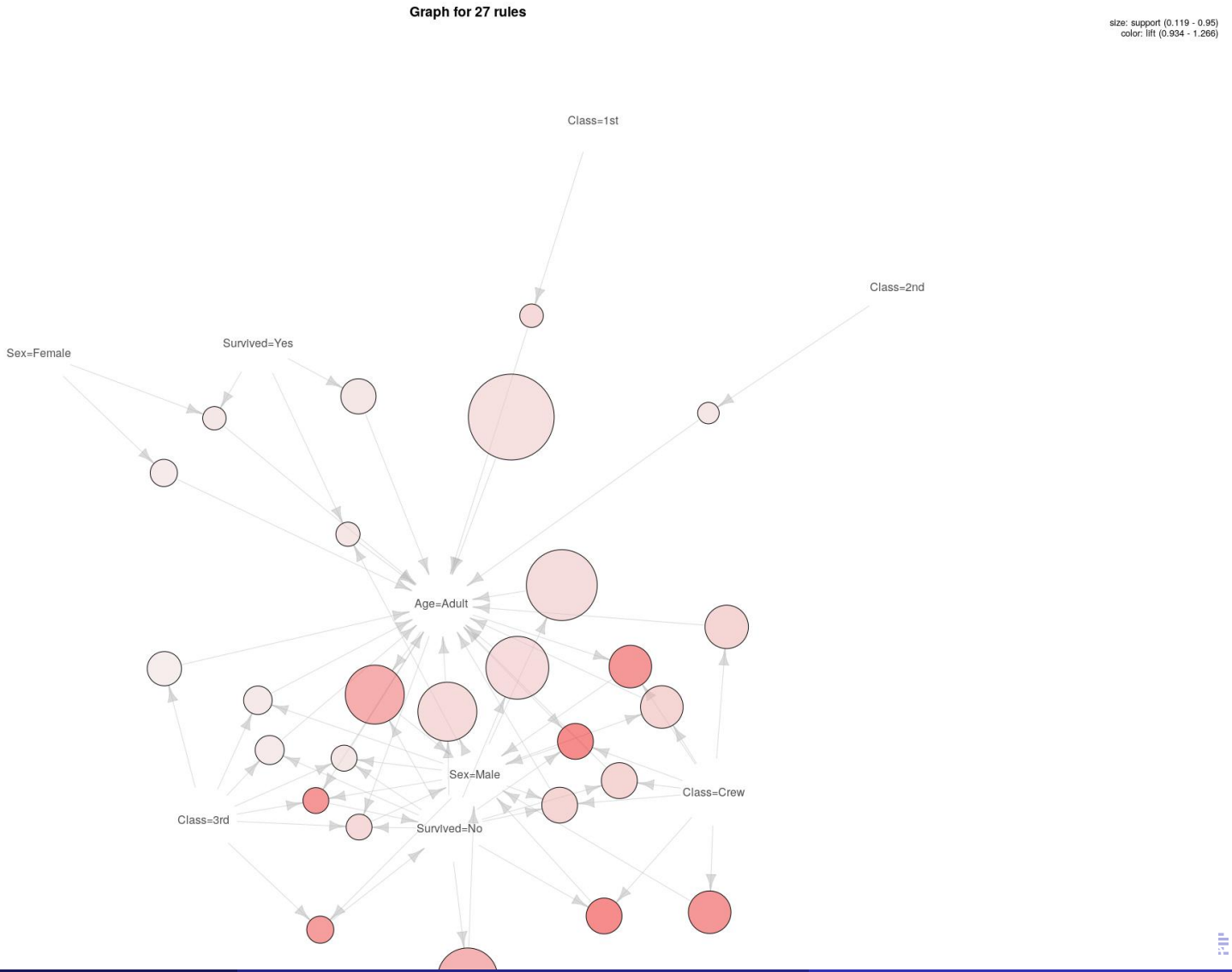
Visualización de reglas

```
plot(rules.all, method="grouped")
```



Visualización de reglas

```
plot(rules.all, method="graph")
```



Visualización de reglas

```
plot(rules, method="graph")
```