Reglas de Asociación

INAOE

Reglas de Asociación

- Objetivo: encontrar asociaciones o correlaciones entre los elementos u objetos de bases de datos transaccionales, relacionales o *datawarehouses*
- Las reglas de asociación tienen diversas aplicaciones como:
  - Soporte para la toma de decisiones
  - Diagnóstico y predicción de alarmas en telecomunicaciones
  - Análisis de información de ventas
  - Distribución de mercancías en tiendas
  - Segmentación de clientes con base en patrones de compra

Reglas de Asociación

- Son parecidas a las reglas de clasificación
- Se encuentran también usando un procedimiento de *covering*, sin embargo, en el lado derecho de las reglas, puede aparecer cualquier par o pares atributo-valor
- Para encontrar este tipo de reglas se debe de considerar cada posible combinación de pares atributo-valor del lado derecho.
- Para posteriormente podarlas usando:
  - Covertura: número de instancias predichas correctamente
  - Precisión: proporción de número de instancias a las cuales aplica la regla

Ejemplo

Transacción	Elementos Comprados
1	A,B,C
2	A,C
3	A,D
4	B,E,F

Ejemplo

- Encontrar las reglas de asociación $X \Rightarrow Z$ de la tabla anterior con:
  - Covertura mínima de 50 %
  - Precisión mínima de 50 %
- Las reglas que cumplen con estas restricciones son:
  - $A \Rightarrow C$ (50 %, 66.6 %)
  - $C \Rightarrow A$ (50 %, 100 %)

Reglas de Asociación

- Una regla de asociación es una expresión de la forma $X \Rightarrow Z$ donde X y Z son conjuntos de elementos.
- El significado intuitivo:

  Las transacciones de la base de datos que contienen X tienden a contener Z

Definiciones

- $I = \{i_1, i_2, i_3, \ldots, i_m\} \Rightarrow$ un conjunto de literales, atributos
- $D \Rightarrow$ un conjunto de transacciones T, $T \subseteq I$
- $TID \Rightarrow$ un identificador asociado a cada transacción
- $X \Rightarrow$ un conjunto de elementos $X \in I$
- Una *regla de asociación* es una implicación:
  - $X \Rightarrow Z$, $X \subseteq I$, $Z \subseteq I$ y $X \cap Z = \emptyset$

Definiciones

- *Soporte* (o *cobertura*), s, es la probabilidad de que una transacción contenga $\{X, Z\}$
- *Confianza* (o *eficiencia*), c, es la probabilidad condicional de que una transacción que contenga $\{X\}$ también contenga $\{Z\}$.

Reglas de Asociación

- Evaluamos las reglas de acuerdo al soporte y la confianza de las mismas.
- En reglas de asociación, la covertura se llama soporte (*support*) y la precisión se llama confianza (*confidence*).
- Se pueden leer como:

$$\text{soporte}(X \Rightarrow Z) = P(X \cup Z)$$

$$\text{confianza}(X \Rightarrow Z) = P(Z|X) = \frac{\text{soporte}(X \cup Z)}{\text{soporte}(X)}$$

Reglas de Asociación

- Buscamos reglas con un mínimo soporte (soporte $\geq$ *sop_min*) y confianza (confianza $\geq$ *conf_min*)
- Inicialmente buscamos (independientemente de que lado aparezcan), pares atributo-valor que cubran una gran cantidad de instancias.
- A los conjuntos de pares atributo-valor, se les llama *item-sets* y a cada par atributo-valor *item*.

Ejemplo: Análsis de Canasta de Mercado

Ejemplo: Análsis de Canasta de Mercado

- Un ejemplo típico de reglas de asociación es el análisis de la canasta de mercado.
- Osea, encontrar asociaciones entre los productos de los clientes, que pueden impactar a las estrategias mercadotécnicas
- Ya que tenemos todos los conjuntos de *itemsets*, los transformamos en reglas con la confianza mínima requerida.
- Algunos *itemsets* producen más de una regla y otros no producen ninguna.

Reglas de Asociación

- Por ejemplo, si seguimos con los datos de la tabla de "Jugar Golf", el *itemset*:

  humedad=normal, viento=no, clase=P

- Puede producir las siguientes posibles reglas:

If humedad=normal and viento=no Then clase=P 4/4
If humedad=normal and clase=P Then viento=no 4/6
If viento=no and clase=P Then humedad=normal 4/6
If humedad=normal Then viento=no and clase=P 4/7
If viento=no Then clase=P and humedad=normal 4/8
If clase=P Then viento=no and humedad=normal 4/9
If true Then humedad=normal and viento=no and clase=P 4/12

Reglas de Asociación

- Si pensamos en 100 % de éxito, entonces sólo la primera regla cumple.
- De hecho existen 58 reglas considerando la tabla completa que cubren al menos dos ejemplos con un 100 % de exactitud (*accuracy*)

Apriori (Agrawal et al. '94)

El proceso es más o menos el siguiente y sigue dos pasos:

1. Genera los *itemsets*
   - Genera todos los *itemsets* con un elemento
   - Usa estos para generar los de dos elementos, y así sucesivamente
   - Toma todos los que cumplen con el mínimo soporte (esto permite eliminar posibles combinaciones)
2. Genera las reglas revisando que cumplan con el criterio mínimo de confianza.

Algoritmo (1)

Apriori()
L_1 = find-frequent-1-itemsets(D)
for ($k = 2$; $L_{k-1} \neq$ NULL; k++)
    % generate-&-prune candidate k-itemsets
    C_k = **AprioriGen**(L_{k-1})
    forall transactions $t \in D$
        C_t = **subset**(C_k, t)
        forall candidates $c \in C_t$
            $c.count++$
    $L_k = \{c \in C_k \mid c.count \geq minsup\}$
Return $\cup_k L_k$

Algoritmo (2): AprioriGen

AprioriGen(L) % Assume transactions in lexicographic order
insert into C_k all $p.item_1, p.item_2, \ldots, p.item_{k-1}, q.item_{k-1}$
    from $p, q \in L$
where $p.item_1 = q.item_1, p.item_2 = q.item_2, \ldots,$
    $p.item_{k-1} < q.item_{k-1}$
% Prune itemsets s.t. some (k-1)-subset of c is $\notin L$
% A (k-1) itemset that is not frequent cannot be a subset of
% a frequent k-itemset, then it is removed
forall itemsets $c \in C_k$
    forall (k-1)-subsets s of c do
        if($s \notin L_{k-1}$) then
            delete c from C_k

Algoritmo (3): AssocRules y GenRules

AssocRules()
forall large itemsets $l_k, k \geq 2$
    GenRules(l_k, l_k)

GenRules(l_k, a_m) % Generate all valid rules $a \rightarrow (l_k - a)$,
    for all $a \subset a_m$
$A = \{(m-1) - \text{itemsets } a_{m-1} | a_{m-1} \subset a_m\}$
forall $a_{m-1} \in A$
    conf = **support**(l_k) / **support**(a_{m-1})
    if(conf $\geq$ *min_conf*) then
        output rule $a_{m-1} \rightarrow (l_k - a_{m-1})$ with
            confidence *conf*, *support* = support(l_k)
        if($m - 1 > 1$) then
            GenRules(l_k, a_{m-1}) % Generate rules with
                subsets of a_{m-1} as antecedents

Propiedades

- Si una conjunción de consecuentes de una regla cumple con los nivels mínimos de soporte y confianza, sus subconjuntos (consecuentes) también los cumplen.
- Por el contrario, si algún *item* no los cumple, no tiene caso considerar sus superconjuntos.
- Esto da una forma de ir construyendo reglas, con un solo consecuente, y a partir de ellas construir de dos consecuentes y así sucesivamente.

Reglas de Asociación

- Este método hace una pasada por la base de datos cada para cada conjunto de *items* de diferente tamaño.
- El esfuerzo computacional depende principalmente de la covertura mínima requerida, y se lleva prácticamente todo en el primer paso.
- El proceso de iteración del primer paso se llama *level-wise* y va considerando los superconjuntos nivel por nivel.

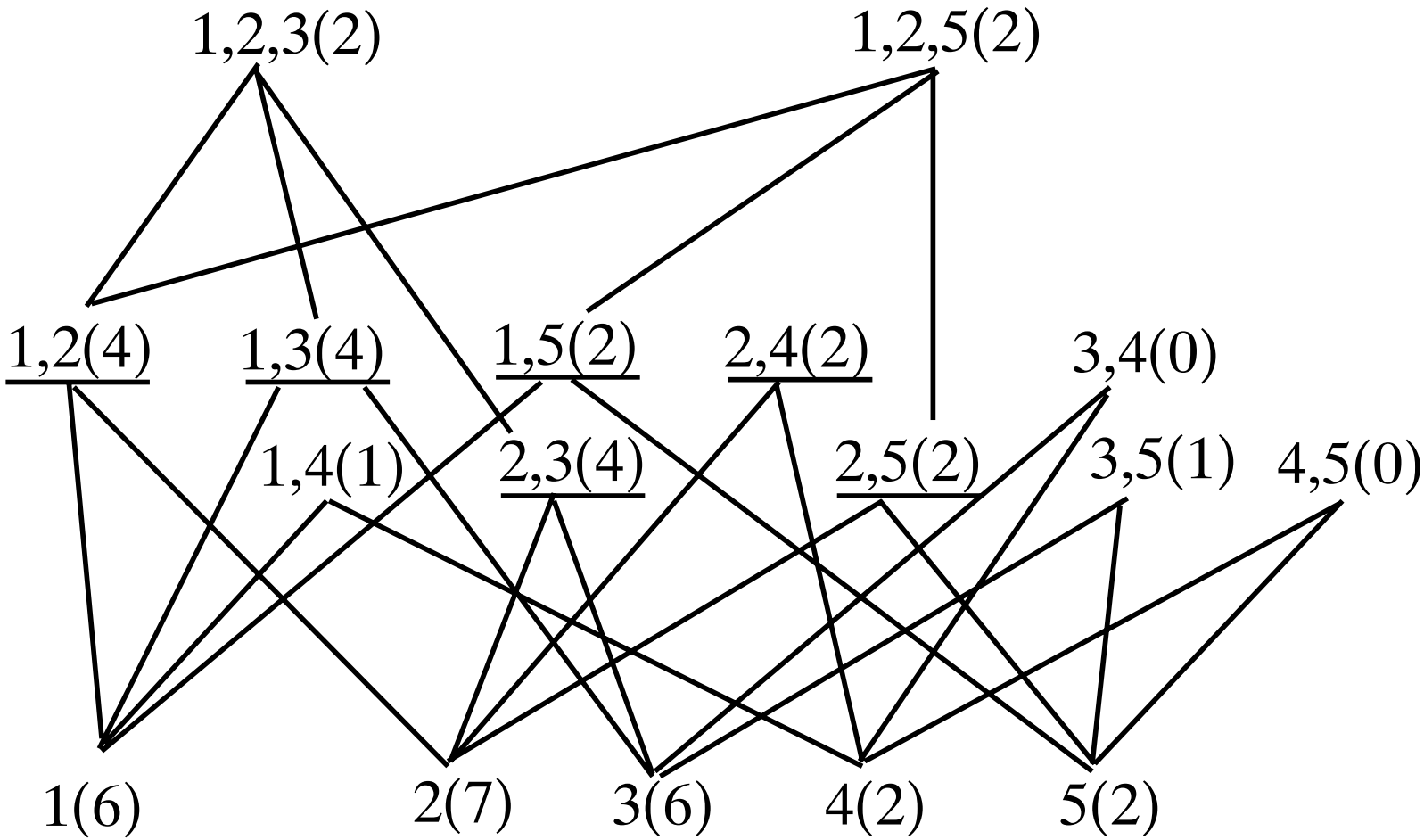
Reglas de Asociación

- Lo que se tiene es una propiedad anti-monótona: si un conjunto no pasa una prueba, ninguno de sus superconjuntos la pasan.
- Si un conjunto de *items* no pasa la prueba de soporte, ninguno de sus superconjuntos la pasan. Esto se aprovecha en la construcción de candidatos para no considerar todos.

Ejemplo

id1	p1,p2,p5
id2	p2,p4
id3	p2,p3
id4	p1,p2,p4
id5	p1,p3
id6	p2,p3
id7	p1,p3
id8	p1,p2,p3,p5
id9	p1,p2,p3

Ejemplo



Reglas de Asociación

- Una vez que tenemos los conjuntos de *items*, generar las reglas es relativamente sencillo.
  - Para cada conjunto l de *items*, genera todos sus subconjuntos.
  - Para cada subconjunto $s \subset l$, genera una regla: $s \Rightarrow (l - s)$ si:

$$\frac{\text{soporte}(l)}{\text{soporte}(s)} \geq \text{nivel\_confianza}$$

- Todas las reglas satisfacen los niveles mínimos de soporte.

Extensiones

Se han hecho algunas mejoras al algoritmo básico de reglas de asociación (Apriori) para hacerlo más eficiente:

- Usar tablas hash para reducir el tamaño de los candidatos de los *itemsets*
- Eliminar transacciones (elementos en la base de datos) que no contribuyan en superconjuntos a considerar
- Dividir las transacciones en particiones disjuntas, evaluar *itemsets* locales y luego, en base a sus resultados, estimar los globales.
- Hacer aproximaciones con muestreos en la lista de productos, para no tener que leer todos los datos

Extensiones: FP-Growth

- Posiblemente la extensión más exitosa es FP-Growth
- Transforma la base de datos en una estructura compacta FP-trees (*Frequent Pattern tree*)
- Tiene un algoritmo eficiente para encontrar patrones frecuentes
- Usa una método que descompone la tareas en tareas más pequeñas
- No genera candidatos y evita leer la base de datos repetidamente
- Es un orden de magnitud más rápido que Apriori

Extensiones: Diferentes abstracciones

- Encontrar reglas de asociación a diferentes niveles de abstracción.
- Normalmente se empieza con las clases superiores, y los resultados pueden servir para filtrar clases inferiores.
- Por ejemplo, considerar reglas de asociación sobre computadoras e impresoras, y luego sobre laptops y estaciones de trabajo, por un lado, y sobre impresoras laser y de punto por otro, etc.

Extensiones: Diferentes abstracciones

Al proceder a las subclases se puede considerar:

- un criterio de soporte uniforme
- reduciendo el criterio para las subclases
- considerar todas las subclases independientemente del criterio de soporte

Extensiones: Diferenes abstracciones

- tomando en cuenta el criterio de soporte de una de las superclases de un *item* o *k* superclases de *k items*
- considerar *items* aunque el nivel de soporte de sus padres no cumplan con el criterio de soporte, pero que sea mayor que un cierto umbral.

Al encontrar reglas de asociación a diferentes niveles de abstracción es común generar reglas redundantes o reglas que no nos dicen nada nuevo (e.g., la regla más general, ya decia lo mismo), por lo que es necesario incorporar mecanismos de filtrado.

Extensiones: Combinaciones de tablas

- Encontrar reglas de asociación combinando información de múltiples tablas o reglas de asociación multidimensionales.
- Los DataCubes pueden servir para encontrar reglas de asociación multidimensionales.

Atributos contínuos

- Las reglas de asociación, al igual que los árboles de decisión y las reglas de clasificación que hemos visto, funcionan, en su forma original, con atributos discretos.
- Al igual que en las otras técnicas se han propuesto mecanismos para manjejar atributos continuos.

Atributos contínuos

Los enfoques más comunes son:

- Discretizar antes de minar en rangos usando posiblemente jerarquías predefinidas.
- Discretizar dinámicamente durante el proceso tratando de maximizar algún criterio de confianza o reducción de longitud de reglas.
- Por ejemplo, ACRS (Association Rule Clustering System), mapea atributos cuantitativos a una rejilla y luego utiliza *clustering*.

Atributos contínuos

- Primero asigna datos a “contenedores” delimitados por rangos (que después pueden cambiar).
- Los esquemas más comunes son: contendores del mismo tamaño, contenedores con el mismo número de elementos, y contenedores con elementos uniformemente distribuidos.
- Después se encuentran reglas de asociación utilizando los contenedores.
- Una vez que se tienen las reglas, éstas se agrupan si forman rectángulos más grandes dentro de la rejilla.

Atributos contínuos

- Discretizar utilizando información semántica, i.e., formar grupos con elementos cercanos (posiblemente haciendo *clustering* sobre los atributos).
- Una vez establecidos los clusters, encontrar las reglas de asociación con esos clusters basados en distancias o similaridades.

Reglas de Asociación

- El que se encuentre una regla de asociación no necesariamente quiere decir que sea útil.
- Por ejemplo, si se analizan 10,000 compras, de las cuales 6,000 compraron videojuegos, 7,500 videos y 4,000 las dos, posiblemente se genere una regla: compra videojuegos $=>$ compra videos [soporte=4,000/10,000 = 40 % y confianza=4,000/6,000 = 66 %].
- Sin embargo, el 75 % de los clientes compran videos por lo que el comprar videojuegos reduce las posibilidades de comprar videos.

Reglas de Asociación

- La ocurrencia de un *itemset A* es independiente de otro B si $P(A \cup B) = P(A)P(B)$, en caso contrario, existe cierta dependencia o correlación.
- La correlación entre dos eventos se define como:

$$\text{corr}_{A,B} = \frac{P(A \cup B)}{P(A)P(B)}$$

- Si es menor que 1, entonces la ocurrencia de uno decrece la ocurrencia del otro
- Si es 1 son independientes
- Si es mayor que 1 la ocurrencia de uno favorece la ocurrencia de otro

Reglas de Asociación

- Con esto, se pueden encontrar reglas de asociación correlacionadas.
- Se puede estimar si la correlación es estadísticamente significativa usando una χ^2.
- Si un conjunto de elementos está correlacionado, cualquier superconjunto de este también lo está.
- Esto puede ayudar a buscar los conjuntos mínimos correlacionados y construir a partir de ahí sus superconjuntos.

Meta-Reglas

- Las meta-reglas permiten especificar la forma de las reglas.
- Podemos buscar por reglas de asociación que tengan formas específicas:
  $P_1(X, Y) \wedge P_2(X, W) \Rightarrow compra(X, \text{libros de KDD})$
  donde P_i es un predicado variable que se instancia con algún atributo de la base de datos, y las X, Y y W son posibles valores de los atributos.

Uso de Restricciones

- Se pueden usar restricciones sobre los tipos de datos, jerarquías, o formas posibles de las reglas a encontrar para reducir el espacio de búsqueda.
- Las restricciones pueden ser:
  - (i) antimonótonas (si un conjunto no satisface una condición, entonces tampoco la satisfacen sus superconjuntos),
  - (ii) monótonas (si un conjunto satisface una restricción, entonces también la satisfacen todos sus superconjuntos),
  - (iii) suscintas (*succint*) (podemos enumerar todos los conjuntos que satisfacen una restricción), (iv) convertibles (podemos converir una restricción a alguna de las clases anteriores), y (v) no convertibles.

Reglas de Asociación, de Clasificación y Árboles de Decisión

Exploración de dependencias	vs.	Predicción enfocada
Diferentes combinaciones de atributos dependientes e independientes	vs.	Predice un atributo (clase) a partir de otros
Búsqueda completa (todas las reglas encontradas)	vs.	búsqueda heurística (se encuentra un subconjunto de reglas)

Reglas de Asociación

- Los árboles usan heurística de evaluación sobre un atributo, estan basados en *splitting*, y normalmente realizan sobreajuste seguido de podado.
- Las reglas de clasificación utilizan una heurística de evaluación de condición (par atributo-valor), están basados en *covering*, y utilizan sobre todo criterios de paro (y a veces sobreajuste y podado).
- Las reglas de asociación se basan en medidas de confianza y soporte, consideran cualquier conjunto de atributos con cualquier otro conjunto de atributos.

Otros Temas Relacionados

- El mecanismo de construcción de reglas de asociación también se ha utilizado para construir reglas de clasificación
  - Lo “único” que se tiene que asegurar es que se tenga a la clase como el único consecuente
- Otro tema relacionado a clasificadores, reglas de clasificación y de asociación es: *Subgroup Discovery*
  - Busca patrones (reglas) entre objetos con respecto a una variable de interés