REGLAS DE ASOCIACIÓN

Jesús González y Eduardo Morales

Minería de Datos por Reglas de Asociación

- Encontrar asociaciones o correlaciones entre los elementos u objetos de bases de datos transacccionales, relacionales o datawarehouses.

Aplicaciones

- Soporte para toma de decisiones
- Diagnóstico y predicción de alarmas de telecomunicaciones
- Análisis de información de ventas
  - Diseño de catálogos
  - Distribución de mercancías en tiendas
  - Segmentación de clientes en base a patrones de compra

Reglas de Asociación

- Parecidas a las reglas de clasificación
- Se encuentran también con un procedimiento de ***covering***
- Principal diferencia
  - En el lado derecho de las reglas puede aparecer cualquier par o pares atributo-valor
- Para encontrar este tipo de reglas
  - Considerar cada posible combinación par atributo-valor del lado derecho

Reglas de Asociación

- Posteriormente se poda
  - Usando cobertura
    - Número de instancias predichas correctamente
  - Usando precisión
    - Proporción de núm. de instancias a las cuales aplica la regla

Ejemplo

- Encontrar las reglas de asociación X & Y → Z con las restricciones de cumplir con un mínimo de confianza y de soporte

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

Reglas con:
- Soporte mínimo de 50%
- Confianza mínima de 50%

$A \Rightarrow C$ (50%, 66.6%)
$C \Rightarrow A$ (50%, 100%)

Reglas de Asociación

- Una regla de asociación es una expresión de la forma X → Z donde X y Z son conjuntos de elementos
- Significado intuitivo
  - Las transacciones de la base de datos que contienen X tienden a contener Z
  - Se requiere un nivel de soporte y confianza mínimos

Definiciones

- 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 $X \subset I$
  - Una regla de asociación es una implicación:
    - $X \rightarrow Z$, $X \subset I$, $Z \subset I$ y $X \cap Z = \varnothing$

Definiciones

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

Evaluación de las Reglas

- Evaluamos las reglas de acuerdo al soporte y la confianza de las mismas
  - En reglas de asociación la cobertura se llama
    - Soporte (Support)
  - La precisión se llama
    - Confianza (Confidence)

Evaluación de Reglas

- Se pueden leer como

$$soporte(X \Rightarrow Z) = P(X \cup Z) = \frac{\#Trans\_con\_elementos\_en\_X\_y\_Z}{\#Total\_de\_trans}$$

$$confianza(X \Rightarrow Z) = P(Z \mid X)$$

$$confianza(X \Rightarrow Z) = \frac{soporte(X \cup Z)}{soporte(X)} = \frac{\#Trans\_que\_contienen\_X\_y\_Z}{\#trans\_que\_contienen\_X}$$

Evaluación de Reglas

- Queremos reglas con un mínimo soporte y confianza:
  - soporte $\geq$ sop_min
  - confianza $\geq$ conf_min
  - Buscamos (independientemente de en qué lado aparezcan)
    - Pares atributo-valor que cubran gran cantidad de instancias
- A los conjuntos de pares atributo valor se les llama
  - item-sets
- A cada par atributo-valor se le llama
  - item

Ejemplo

http://zulfiqar.typepad.com/photos/uncategorized/shoppingcart.jpg

Ejemplo

- Un ejemplo típico de reglas de asociación
  - *Análisis de la canasta de mercado*
  - Encontrar asociaciones entre los productos de los clientes
  - Pueden impactar las estrategias mercadotécnicas
- Después de generar todos los conjuntos de itemsets
  - Los transformamos a reglas
    - Con confianza mínima requerida
    - Algunos items producen más de una regla y otros ninguna

Ejemplo

- Siguiendo con el ejemplo del clima
  - El itemset: humedad = normal, viento = no, clase = P
- Produce las siguientes posibles reglas:
  - If humedad = normal & viento = no Then Clase = P 4/4
  - If humedad = normal & clase = P Then viento = no 4/6
  - If viento = no & clase = P Then humedad = normal 4/6
  - If humedad = normal Then viento = no & clase = P 4/7
  - If viento = no Then clase = P & humedad = normal 4/8
  - If clase = P Then viento = no & humedad = normal 4/9
  - If true Then humedad = normal & viento = no & clase = P 4/12

Ejemplo

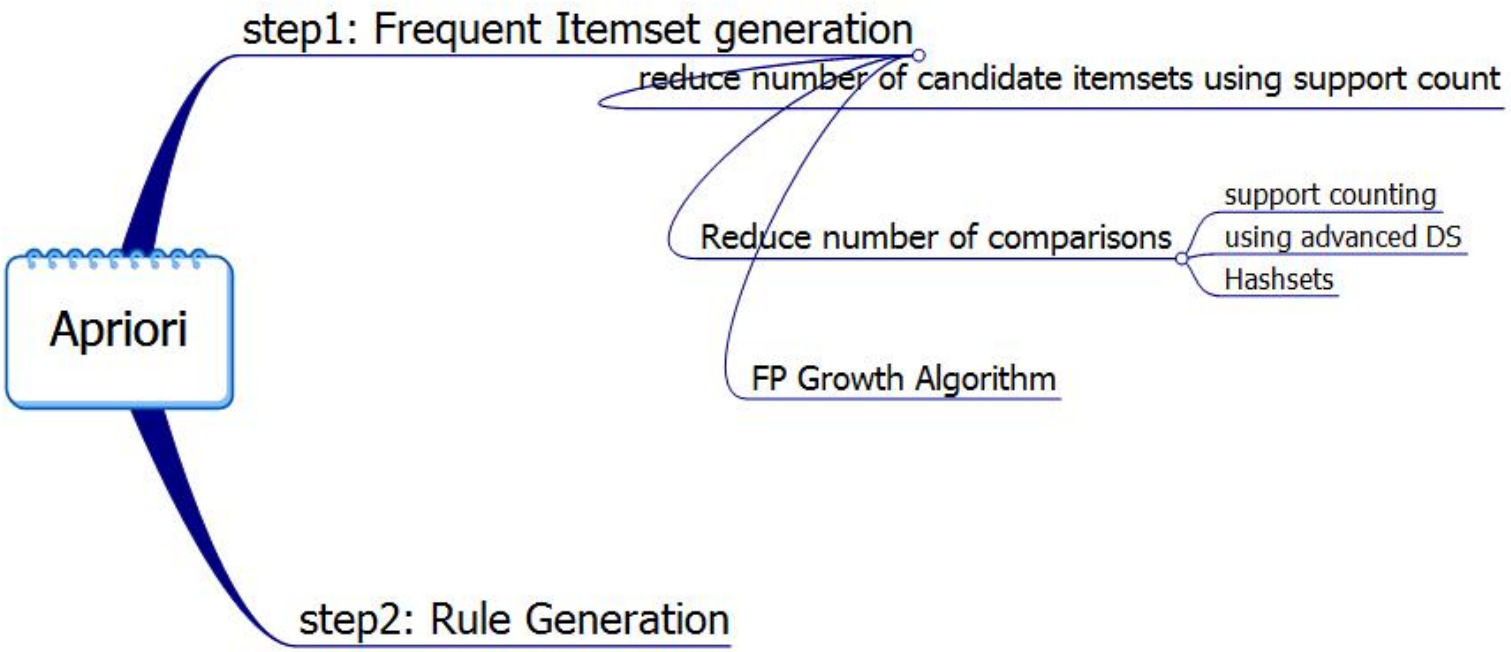
- Si pensamos en 100% de éxito, entonces sólo la primera regla cumple
- Existen 58 reglas considerando la tabla completa que cubren 2 ejemplos con 100% de exactitud

Algoritmo

- A grandes rasgos el algoritmo sigue 2 pasos
  - Apriori (Agrawal et al. 1994)

1. Genera todos los itemsets con un elemento
   - Usa estos para generar los de 2 elementos y así sucesivamente
   - Se toman todos los posibles pares con mínimo soporte
   - Permite eliminar posibles combinaciones (podas)
2. Genera las reglas checando si cumplen la confianza mínima

Algoritmo



http://prdeepakbabu.files.wordpress.com/2010/02/apriori.jpg

Pseudocódigo: Apriori

- Apriori()
- L_1= find_frequent_1-itemsets(D)
- for (k = 2; L_{k-1} != NULL; k++)
- 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 | c.count $\geq$ minsup}
- Return $\cup_k$ L_k

- AprioriGen genera los candidatos C_k (de tamaño *k*) a partir de los itemsets frecuentes de tamaño *k*-1
- Subset determina cuáles de los itemsets candidatos son realmente frecuentes en cada pasada (re-escanea y compara)

Pseudocódigo: AprioriGen

- AprioriGen(L) /*Assume transactions in lexicographic order*/
- insert into C_k all $p.item_1, p.item_2,...,p.item_{k-1},q.item_{k-1}$ from $p,q \in L$
- where $p.item_1=q.item_1, p.item_2=q.item_2,...,p.item_{k-1}<q.item_{k-1}$
- /*Prune itemsets s.t. Some (k-1)-subset of c is $\notin L$ */
- for all itemsets $c \in C_k$
- forall (k-1)-subsets s of c do
- if ($s \notin L_{k-1}$) then /* s is non-frequent in L_{k-1}
- delete c from C_k

Pseudocódigo: GenRules

- GenRules(l_k,a_m) /* Generate all valid rules $a \Rightarrow (l_k - a)$, for all $a \subset a_m$ */
- A = {(m-1)-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 ≥ min_conf) then
- output rule $a_{m-1} \Rightarrow (l_k - a_{m-1})$ with confidence conf, support = support(lk)
- if(m-1 > 1) then
- GenRules(l_k,a_{m-1}) /*Generate rules with subsets of am-1 as antecedents*/

Pseudocódigo: AssocRules

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

Observaciones

- Si una conjunción de consecuentes de una regla cumple con los niveles mínimos de soporte y confianza, sus subconjuntos consecuentes también los cumplen
- Si algún item no los cumple, no tiene caso considerar sus superconjuntos
- Da forma de construir reglas con 1solo consecuente, a partir de ellas de 2 y así sucesivamente.

Observaciones

- Este método hace una pasada por la base de datos para cada conjunto de items de diferente tamaño
- El esfuerzo computacional depende principalmente de la cobertura mínima requerida
  - Se lleva prácticamente todo el primer paso
- El proceso de iteración del primer paso se llama
  - *Level-wise*
  - Considera los superconjuntos nivel por nivel
- Tenemos una propiedad anti-monótona (downward closure)
  - Si un conjunto no pasa la prueba, ninguno de sus superconjuntos la pasan

Observaciones

- Si *n* conjuntos de items no pasan la prueba de soporte
  - Ninguno de sus superconjuntos la pasan
  - Se aprovecha en la construcción de candidatos
    - Para no considerar todos (podar)

Ejemplo

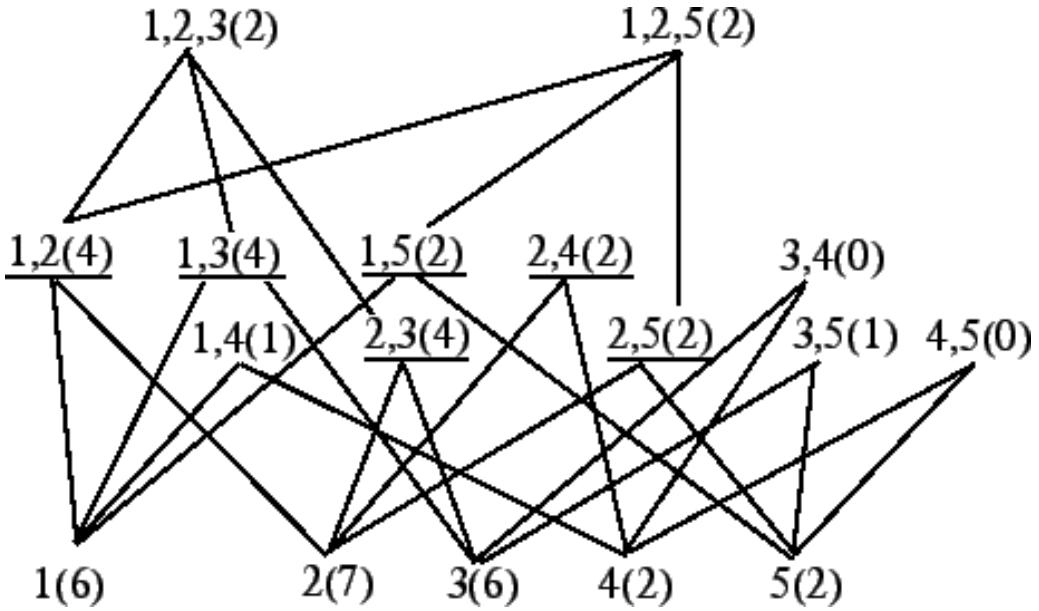
- Considerando una tabla con listas de compras

ID	Productos
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

- Proceso con los datos de la tabla anterior
  - Generación de candidatos por niveles

ID	Productos
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



Observaciones

- Teniendo los conjuntos de itemsets, 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

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

Algunas Mejoras

- Se han hecho algunas mejoras al algoritmo básico de reglas de asociación (Apriori) para hacerlo más eficiente
  - Usar tablas hash para reducir # de 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 los itemsets locales y luego, en base a sus resultados, estimar los globales

Algunas Mejoras

- Hacer aproximaciones con muestreos en la lista de productos, para no tener que leer todos los datos
- Evitar generar candidatos usando estructuras de datos alternativas, como por ejemplo, los FP-trees (Frequent Pattern tree).

Algunas Extensiones

- Encontrar reglas de asociación a diferentes niveles de abstracción
  - Normalmente inicia con las clases superiores
  - Los resultados pueden servir para filtrar clases inferiores
  - p.e. reglas de asoc. sobre computadoras e impresoras
    - Luego sobre laptops y estaciones de trabajo por un lado y sobre impresoras laser y de punto por otro, etc.

Algunas Extensiones

- Encontrar reglas de asociación a diferentes niveles de abstracción
  - Al proceder a las subclases se puede considerar:
    - Un criterio de soporte uniforme
    - Reduciendo el criterio para las subclases
    - Considerar todas las subclases indep. del criterio de soporte
    - 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

Algunas Extensiones

- Encontrar reglas de asociación a diferentes niveles de abstracción
  - Al encontrar reglas de asoc. a dif. niveles de abstracción
    - Es común generar reglas redundantes
    - Reglas que no nos dicen nada nuevo (e.g., la regla más general ya decía lo mismo)
    - Es necesario incorporar mecanismos de filtrado

Algunas Extensiones

- Encontrar reglas de asoc. combinando inf. de múltiples tablas o reglas de asoc. multidimensionales
  - Los DataCubes pueden servir para encontrar reglas de asociación multi-dimensionales

Algunas Extensiones

- Las reglas de asociación, originalmente, funcionan con atributos discretos
  - Se han propuesto mecanismos para manejar atributos continuos
    - Discretizar antes de minar en rangos usando posiblemente jerarquías predefinidas

Algunas Extensiones

- Las reglas de asociación, originalmente, funcionan con atributos discretos
  - Discretizar dinámicamente durante el proceso tratando de maximizar algún criterio de confianza o reducción de longitud de reglas
    - ie. ACRS (Association Rule Clustering System)
      - Mapea atributos cuantitativos a una rejilla y luego utiliza clustering
      - 1° asigna datos a contenedores delimitados por rangos (que después pueden cambiar)
      - Esquemas más comunes: contenedores del mismo tamaño, con el mismo # de elementos, con elementos uniformemente distribuidos
      - Después encuentra reglas de asociación usando los contenedores
      - Cuando se tienen las reglas, éstas se agrupan si forman rectángulos más grandes dentro de la rejilla

Algunas Extensiones

- Las reglas de asociación, originalmente, funcionan con atributos discretos
  - Discretizar utilizando inf. semántica
    - i.e. formar grupos con elementos cercanos (ie. con clustering sobre los atributos)
    - Ya con los clusters, se encuentran reglas de asociación con esos clusters
      - Basados en distancias o similaridades

Asociación vs. Correlación

- El que se encuentre una regla de asociación no necesariamente quiere decir que sea útil
  - ie. 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 video juegos → compra videos [sop=4,000/10,000 = 40% y confianza 4,000/6,000 = 66%]
    - Pero 75% compran videos, entonces comprar videojuegos reduce las posibilidades de comprar videos

Asociación vs. Correlación

- 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
- Correlación entre dos eventos se define como

$$corr_{A,B} = \frac{P(A \cup B)}{P(A)P(B)}$$

  - Si es menor que 1, 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 del otro

Asociación vs. Correlación

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

Meta-Reglas

- Permiten especificar la forma de las reglas
- Podemos buscar por reglas de asociación que tengan formas específicas
  - P1(X,Y) & P2(X,W) $\Rightarrow$ compra (X, "libros_de_KDD")
  - Donde Pi es un predicado variable que se instancia con algún atributo de la base de datos
  - X, Y y W son posibles valores de 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:
  - Antimonótonas
    - Si un conjunto no satisface una condición, entonces tampoco la satisfacen sus superconjuntos
  - Monótonas
    - Si un conjunto satisface una restricción, entonces también la satisfacen todos sus superconjuntos
  - Suscintas (succint)
    - Podemos enumerar todos los conjuntos que satisfacen una restricción
  - Convertibles
    - Podemos convertir una restricción a alguna de las clases anteriores
  - No convertibles

Reglas de Asociación, de Clasificación y Árboles de Decisión

- Comparación entre reglas de asociación y de clasificación
  - Exploración de dependencias vs. Predicción enfocada
  - Dif. 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, de Clasificación y Árboles de Decisión

- Los árboles usan heurística de evaluación sobre un atributo
  - Basados en splitting
  - Normalmente realizan sobreajuste seguido de podado
- Las reglas de clasificación utilizan una heurística de evaluación de condición (par atributo-valor)
  - Basados en covering
  - Utilizan sobre todo criterios de paro (y a veces sobreajuste y podado)
- Reglas de asociación se basan en medidas de confianza y soporte
  - Consideran cualquier conjunto de atributos con cualquier otro conjunto de atributos