Mining Association Rules

Mining Association Rules

- **What is Association rule mining**
- Apriori Algorithm
- Additional Measures of rule interestingness
- Advanced Techniques

What Is Association Rule Mining?

- Association rule mining
  - Finding frequent patterns, associations, correlations, or causal structures among sets of items in transaction databases
  - Understand customer buying habits by finding associations and correlations between the different items that customers place in their "shopping basket"
- Applications
  - Basket data analysis, cross-marketing, catalog design, loss-leader analysis, web log analysis, fraud detection (supervisor->examiner)

What Is Association Rule Mining?

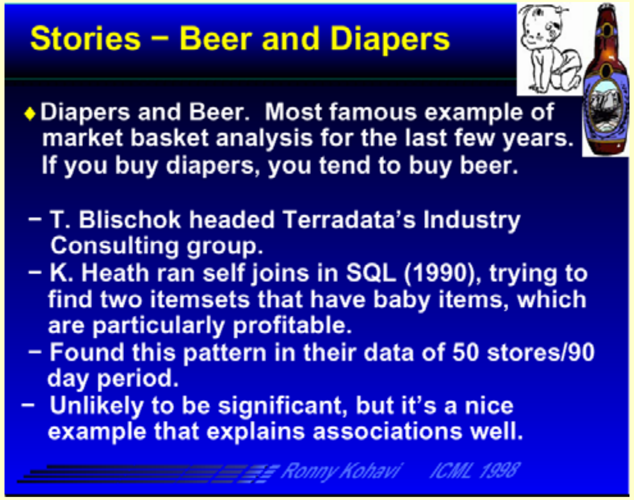
- Rule form

  Antecedent → Consequent [support, confidence]

  (support and confidence are user defined measures of interestingness)

- Examples
  - buys(x, "computer") → buys(x, "financial management software") [0.5%, 60%]
  - age(x, "30..39") ^ income(x, "42..48K") → buys(x, "car") [1%,75%]

How can Association Rules be used?



Probably mom was calling dad at work to buy diapers on way home and he decided to buy a six-pack as well.

The retailer could move diapers and beers to separate places and position high-profit items of interest to young fathers along the path.

How can Association Rules be used?

- Let the rule discovered be

  {Bagels,...} → {Potato Chips}

- Potato chips as consequent => Can be used to determine what should be done to boost its sales
- Bagels in the antecedent => Can be used to see which products would be affected if the store discontinues selling bagels
- Bagels in antecedent and Potato chips in the consequent => Can be used to see what products should be sold with Bagels to promote sale of Potato Chips

Association Rule: Basic Concepts

- Given:
  - (1) database of transactions,
  - (2) each transaction is a list of items purchased by a customer in a visit
- Find:
  - all rules that correlate the presence of one set of items (*itemset*) with that of another set of items
  - E.g., 98% of people who purchase tires and auto accessories also get automotive services done

Rule basic Measures: Support and Confidence

$$A \Rightarrow B\ [\ s, c\ ]$$

Support: denotes the frequency of the rule within transactions. A high value means that the rule involve a great part of database.

$$\text{support}(A \Rightarrow B\ [\ s, c\ ]) = p(A \cup B)$$

Confidence: denotes the percentage of transactions containing A which contain also B. It is an estimation of conditioned probability .

$$\text{confidence}(A \Rightarrow B\ [\ s, c\ ]) = p(B|A) = \text{sup}(A,B)/\text{sup}(A).$$

Example

Trans. Id	Purchased Items
1	A,D
2	A,C
3	A,B,C
4	B,E,F

Itemset:

A,B or B,E,F

Support of an itemset:

Sup(A,B)=1 Sup(A,C)=2

Frequent pattern:

Given min. sup=2, {A,C} is a frequent pattern

For minimum support = 50% and minimum confidence = 50%, we have the following rules

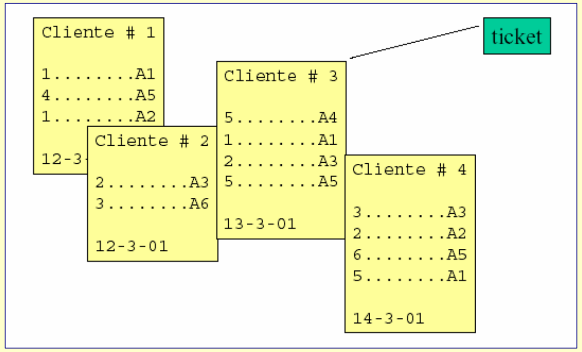
A => C with 50% support and 66% confidence

C => A with 50% support and 100% confidence

Mining Association Rules

- What is Association rule mining
- **Apriori Algorithm**
- Additional Measures of rule interestingness
- Advanced Techniques

Boolean association rules



Each transaction is represented by a Boolean vector

Cliente	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13
1	1	1	0	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	1	0	0	0	0	0	0	0
3	1	0	1	1	1	0	0	0	0	0	0	0	0
4	1	1	1	0	1	0	0	0	0	0	0	0	0
5	0	0	1	0	0	1	0	1	1	1	0	0	0
6	0	1	0	0	0	0	0	1	0	1	0	0	0
7	1	0	0	0	0	0	1	1	0	1	0	1	1
8	0	1	0	0	0	0	0	1	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	1	0	1	0

Mining Association Rules - An Example

Transaction ID	Items Bought
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

Min. support 50%
Min. confidence 50%

Frequent Itemset	Support
{A}	75%
{B}	50%
{C}	50%
{A,C}	50%

For rule $A \Rightarrow C$:

support = support({A, C}) = 50%

confidence = support({A, C}) / support({A}) = 66.6%

The Apriori principle

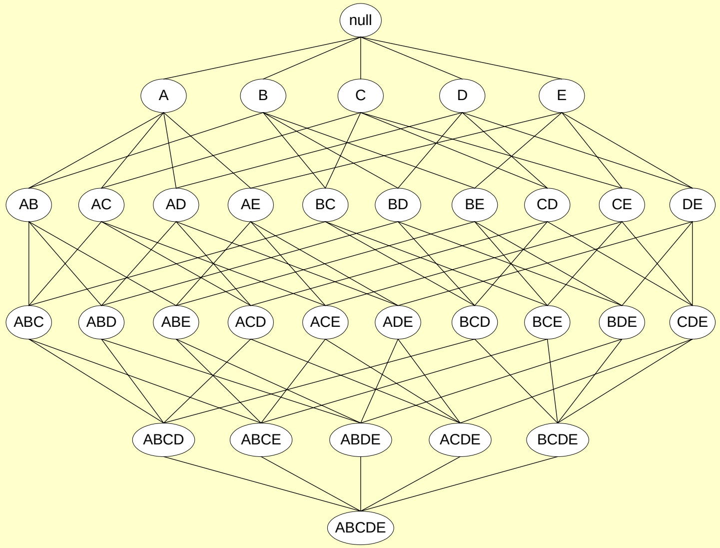
Any subset of a frequent itemset must be frequent

- A transaction containing {beer, diaper, nuts} also contains {beer, diaper}
- {beer, diaper, nuts} is frequent → {beer, diaper} must also be frequent

Apriori principle

- No superset of any infrequent itemset should be generated or tested
  - Many item combinations can be pruned

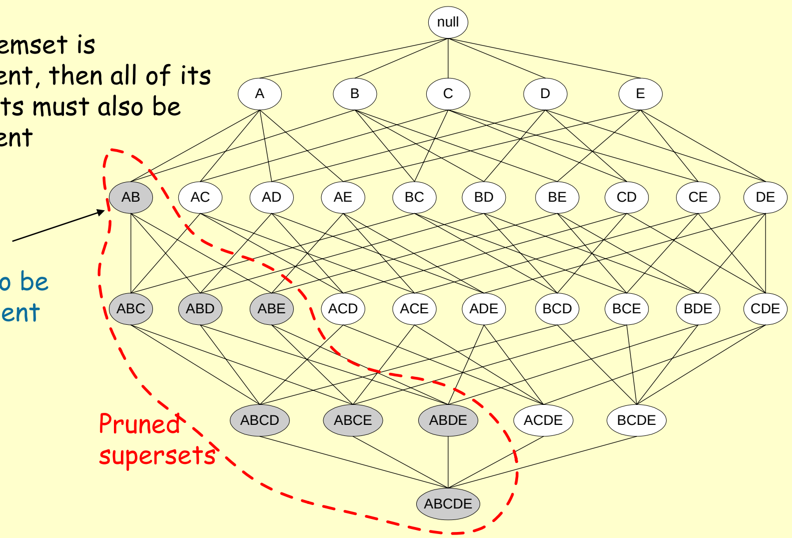
Itemset Lattice



Apriori principle for pruning candidates

If an itemset is infrequent, then all of its supersets must also be infrequent

Found to be Infrequent



Mining Frequent Itemsets (the Key Step)

- Find the *frequent itemsets:* the sets of items that have minimum support
  - A subset of a frequent itemset must also be a frequent itemset
    - Generate length (k+1) candidate itemsets from length k frequent itemsets, and
    - Test the candidates against DB to determine which are in fact frequent
- Use the frequent itemsets to generate association rules.
  - Generation is straightforward

The Apriori Algorithm — Example

Min. support: 2 transactions

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

Scan D

C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

Scan D

C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_3

itemset
{2 3 5}

Scan D

L_3

itemset	sup
{2 3 5}	2

How to Generate Candidates?

- The items in L_{k-1} are listed in an order
- Step 1: self-joining L_{k-1}

insert into C_k

select $p.item_1, p.item_2, \ldots, p.item_{k-1}, q.item_{k-1}$

from $L_{k-1}\ p, L_{k-1}\ q$

where $p.item_1=q.item_1, \ldots, p.item_{k-2}=q.item_{k-2}, p.item_{k-1} < q.item_{k-1}$

A	D	E
=	=	<
A	D	F

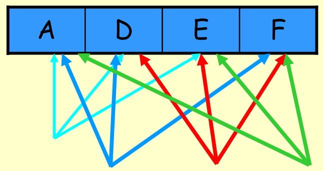
How to Generate Candidates?

- Step 2: pruning

for all *itemsets c in C_k* do

for all *(k-1)-subsets s of c* do

if *(s is not in L_{k-1})* then delete *c* from C_k



Example of Generating Candidates

- L_3={abc, abd, acd, ace, bcd}
- **Self-joining**: $L_3 * L_3$
  - abcd from abc and abd
  - acde from acd and ace
- **Pruning** *(before counting its support)*:
  - acde is removed because ade is not in L_3
- C_4={abcd}

The Apriori Algorithm

- C_k: Candidate itemset of size k L_k : frequent itemset of size k
- Join Step: C_k is generated by joining L_{k-1} with itself
- Prune Step: Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset
- Algorithm:

L_1 = {frequent items};
for (k = 1; L_k !=∅; k++) **do begin**
  C_{k+1} = candidates generated from L_k;
  for each transaction t in database do
    increment the count of all candidates in C_{k+1} that are contained in t
    L_{k+1} = candidates in C_{k+1} with min_support
  end
return $L = \cup_k L_k$;

How to Count Supports of Candidates?

- Why counting supports of candidates a problem?
  - The total number of candidates can be very huge
  - One transaction may contain many candidates
- Method:
  - Candidate itemsets are stored in a hash-tree
  - Leaf node of hash-tree contains a list of itemsets and counts
  - Interior node contains a hash table
  - Subset function: finds all the candidates contained in a transaction

Generating AR from frequent intemsets

- Confidence (A⇒B) = P(B|A) = $\frac{\text{support\_count}(\{A,B\})}{\text{support\_count}(\{A\})}$
- For every frequent itemset x, generate all non-empty subsets of x
- For every non-empty subset s of x, output the rule

  " s ⇒ (x-s) " if

$$\frac{\text{support\_count}(\{x\})}{\text{support\_count}(\{s\})} \geq \text{min\_conf}$$

From Frequent Itemsets to Association Rules

- *Q: Given frequent set {A,B,E}, what are possible association rules?*
  - A => B, E
  - A, B => E
  - A, E => B
  - B => A, E
  - B, E => A
  - E => A, B
  - __ => A,B,E (empty rule), or true => A,B,E

Generating Rules: example

Trans-ID	Items
1	ACD
2	BCE
3	ABCE
4	BE
5	ABCE

Min_support: 60%
Min_confidence: 75%

Frequent Itemset	Support
{**BCE**},{AC}	60%
{BC},{CE},{A}	60%
{BE},{B},{C},{E}	80%

Rule	Conf.
{BC} =>{E}	100%
{BE} =>{C}	75%
{CE} =>{B}	100%
{B} =>{CE}	75%
{C} =>{BE}	75%
{E} =>{BC}	75%

Exercice

TID	Items
1	Bread, Milk, Chips, Mustard
2	Beer, Diaper, Bread, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk, Chips
5	Coke, Bread, Diaper, Milk
6	Beer, Bread, Diaper, Milk, Mustard
7	Coke, Bread, Diaper, Milk

Converta os dados para o formato booleano e para um suporte de 40%, aplique o algoritmo apriori.

Bread	Milk	Chips	Mustard	Beer	Diaper	Eggs	Coke
1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	0
0	1	0	0	1	1	0	1
1	1	1	0	1	1	0	0
1	1	0	0	0	1	0	1
1	1	0	1	1	1	0	0
1	1	0	0	0	1	0	1

0.4*7= 2.8

C1	
Bread	6
Milk	6
Chips	2
Mustard	2
Beer	4
Diaper	6
Eggs	1
Coke	3

L1	
Bread	6
Milk	6
Beer	4
Diaper	6
Coke	3

C2	
Bread,Milk	5
Bread,Beer	3
Bread,Diaper	5
Bread,Coke	2
Milk,Beer	3
Milk,Diaper	5
Milk,Coke	3
Beer,Diaper	4
Beer,Coke	1
Diaper,Coke	3

L2	
Bread,Milk	5
Bread,Beer	3
Bread,Diaper	5
Milk,Beer	3
Milk,Diaper	5
Milk,Coke	3
Beer,Diaper	4
Diaper,Coke	3

C3	
Bread,Milk,Beer	2
Bread,Milk,Diaper	4
Bread,Beer,Diaper	3
Milk,Beer,Diaper	3
Milk,Beer,Coke	
Milk,Diaper,Coke	3

L3	
Bread,Milk,Diaper	4
Bread,Beer,Diaper	3
Milk,Beer,Diaper	3
Milk,Diaper,Coke	3

$$8 + C_2^8 + C_3^8 = 92 \quad >> \quad 24$$

Challenges of Frequent Pattern Mining

- Challenges
  - Multiple scans of transaction database
  - Huge number of candidates
  - Tedious workload of support counting for candidates
- Improving Apriori: general ideas
  - Reduce number of transaction database scans
  - Shrink number of candidates
  - Facilitate support counting of candidates

Improving Apriori's Efficiency

- Problem with Apriori: every pass goes over whole data.
- AprioriTID: Generates candidates as apriori but DB is used for counting support only on the first pass.
  - Needs much more memory than Apriori
  - Builds a storage set $C\hat{}_k$ that stores in memory the frequent sets per transaction
- AprioriHybrid: Use Apriori in initial passes; Estimate the size of $C\hat{}_k$; Switch to AprioriTid when $C\hat{}_k$ is expected to fit in memory
  - The switch takes time, but it is still better in most cases

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

$C\hat{}_1$

TID	Set-of-itemsets
100	{ {1},{3},{4} }
200	{ {2},{3},{5} }
300	{ {1},{2},{3},{5} }
400	{ {2},{5} }

L_1

Itemset	Support
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

$C\hat{}_2$

TID	Set-of-itemsets
100	{ {1 3} }
200	{ {2 3},{2 5} {3 5} }
300	{ {1 2},{1 3},{1 5}, {2 3}, {2 5}, {3 5} }
400	{ {2 5} }

L_2

Itemset	Support
{1 3}	2
{2 3}	3
{2 5}	3
{3 5}	2

C_3

itemset
{2 3 5}

$C\hat{}_3$

TID	Set-of-itemsets
200	{ {2 3 5} }
300	{ {2 3 5} }

L_3

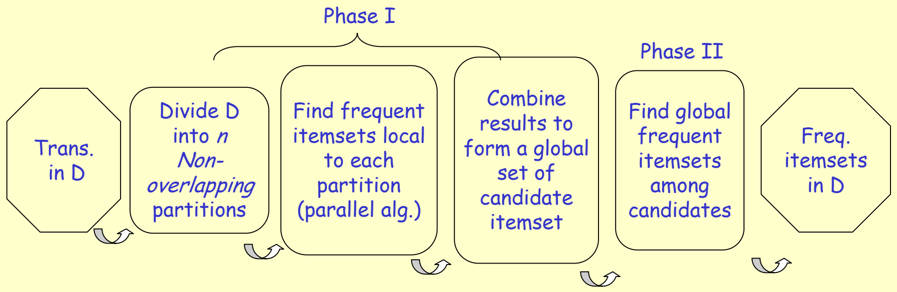
Itemset	Support
{2 3 5}	2

Improving Apriori's Efficiency

- Transaction reduction: A transaction that does not contain any frequent k-itemset is useless in subsequent scans
- Sampling: mining on a subset of given data.
  - The sample should fit in memory
  - Use lower support threshold to reduce the probability of missing some itemsets.
  - The rest of the DB is used to determine the actual itemset count.

Improving Apriori's Efficiency

- Partitioning: Any itemset that is potentially frequent in DB must be frequent in at least one of the partitions of DB (2 DB scans)
  - (support in a partition is lowered to be proportional to the number of elements in the partition)



Improving Apriori's Efficiency

- Dynamic itemset counting: partitions the DB into several blocks each marked by a start point.
  - At each start point, DIC estimates the support of all itemsets that are currently counted and adds new itemsets to the set of candidate itemsets if all its subsets are estimated to be frequent.
  - If DIC adds all frequent itemsets to the set of candidate itemsets during the first scan, it will have counted each itemset's exact support at some point during the second scan;
  - thus DIC can complete in two scans.

Comment

- Traditional methods such as database queries:
  - support hypothesis verification about a relationship such as the co-occurrence of diapers & beer.
- Data Mining methods automatically discover significant associations rules from data.
  - Find whatever patterns exist in the database, without the user having to specify in advance what to look for (data driven).
  - Therefore allow finding unexpected correlations

Mining Association Rules

- What is Association rule mining
- Apriori Algorithm
- **Additional Measures of rule interestingness**
- Advanced Techniques

Interestingness Measurements

- Are all of the strong association rules discovered interesting enough to present to the user?
- How can we measure the interestingness of a rule?

- Subjective measures
  - A rule (pattern) is interesting if
  - it is *unexpected* (surprising to the user); and/or
  - *actionable* (the user can do something with it)
  - (only the user can judge the interestingness of a rule)

Objective measures of rule interest

- Support
- Confidence or strength
- Lift or Interest or Correlation
- Conviction
- Leverage or Piatetsky-Shapiro
- Coverage

Criticism to Support and Confidence

- Example 1: (Aggarwal & Yu, PODS98)
  - Among 5000 students
    - 3000 play basketball
    - 3750 eat cereal
    - 2000 both play basket ball and eat cereal

	basketball	not basketball	sum(row)	
cereal	2000	1750	3750	75%
not cereal	1000	250	1250	25%
sum(col.)	3000	2000	5000	
	60%	40%		

play basketball $\Rightarrow$ *eat cereal* [40%, 66.7%]

misleading because the overall percentage of students eating cereal is 75% which is higher than 66.7%.

play basketball $\Rightarrow$ *not eat cereal* [20%, 33.3%]

is more accurate, although with lower support and confidence

Lift of a Rule

- **Lift (Correlation, Interest)**

$$\text{Lift}(A \rightarrow B) = \frac{\sup(A, B)}{\sup(A) \cdot \sup(B)} = \frac{P(B \mid A)}{P(B)}$$

  - A and B negatively correlated, if the value is less than 1; otherwise A and B positively correlated

X	1	1	1	1	0	0	0	0
Y	1	1	0	0	0	0	0	0
Z	0	1	1	1	1	1	1	1

rule	Support	Lift
X $\Rightarrow$Y	25%	2.00
X$\Rightarrow$Z	37.50%	0.86
Y$\Rightarrow$Z	12.50%	0.57

Lift of a Rule

- Example 1 (cont)
  - *play basketball* ⇒ *eat cereal* [40%, 66.7%] $LIFT = \frac{\frac{2000}{5000}}{\frac{3000}{5000} \times \frac{3750}{5000}} = 0.89$
  - *play basketball* ⇒ *not eat cereal* [20%, 33.3%] $LIFT = \frac{\frac{1000}{5000}}{\frac{3000}{5000} \times \frac{1250}{5000}} = 1.33$

	basketball	not basketball	sum(row)
cereal	2000	1750	3750
not cereal	1000	250	1250
sum(col.)	3000	2000	5000

Problems With Lift

- Rules that hold 100% of the time may not have the highest possible lift. For example, if 5% of people are Vietnam veterans and 90% of the people are more than 5 years old, we get a lift of 0.05/(0.05*0.9)=1.11 which is only slightly above 1 for the rule
- Vietnam veterans -> more than 5 years old.
- And, lift is symmetric:
- *not eat cereal* ⇒ *play basketball* [20%, 80%]

$$LIFT = \frac{\frac{1000}{5000}}{\frac{1250}{5000} \times \frac{3000}{5000}} = 1.33$$

Conviction of a Rule

Note that A -> B can be rewritten as ¬(A,¬B)

$$Conv(A \to B) = \frac{\sup(A) \cdot \sup(\bar{B})}{\sup(A,\bar{B})} = \frac{P(A) \cdot P(\bar{B})}{P(A,\bar{B})} = \frac{P(A)(1-P(B))}{P(A)-P(A,B)}$$

- Conviction is a measure of the implication and has value 1 if items are unrelated.
- *play basketball* ⇒ *eat cereal* [40%, 66.7%] $Conv = \frac{\frac{3000}{5000}\left(1-\frac{3750}{5000}\right)}{\frac{3000}{5000}-\frac{2000}{5000}} = 0.75$
- *eat cereal* ⇒ *play basketball* conv:0.85
- *play basketball* ⇒ *not eat cereal* [20%, 33.3%] $Conv = \frac{\frac{3000}{5000}\left(1-\frac{1250}{5000}\right)}{\frac{3000}{5000}-\frac{1000}{5000}} = 1.125$
- *not eat cereal* ⇒ *play basketball* conv:1.43

Leverage of a Rule

- **Leverage or Piatetsky-Shapiro**

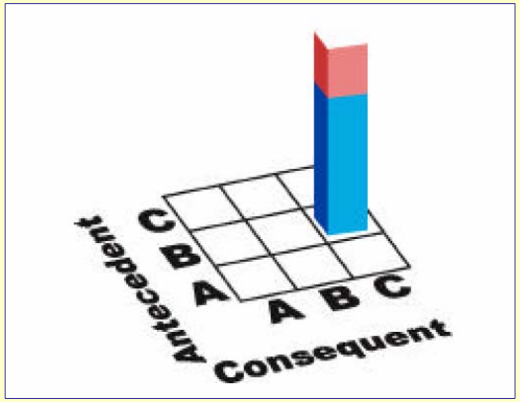
$$PS(A \to B) = \sup(A,B) - \sup(A) \cdot \sup(B)$$

- PS (or Leverage):
- is the proportion of additional elements covered by both the premise and consequence above the expected if independent.

Coverage of a Rule

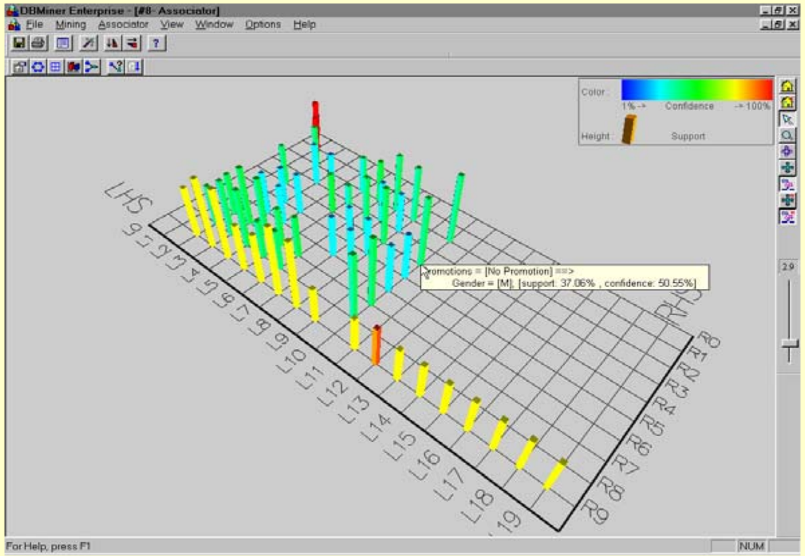
$$coverage(A \rightarrow B) = sup(A)$$

Association Rules Visualization



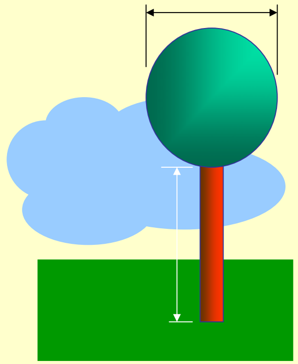
The coloured column indicates the association rule B→C.
Different icon colours are used to show different metadata values of the association rule.

Association Rules Visualization



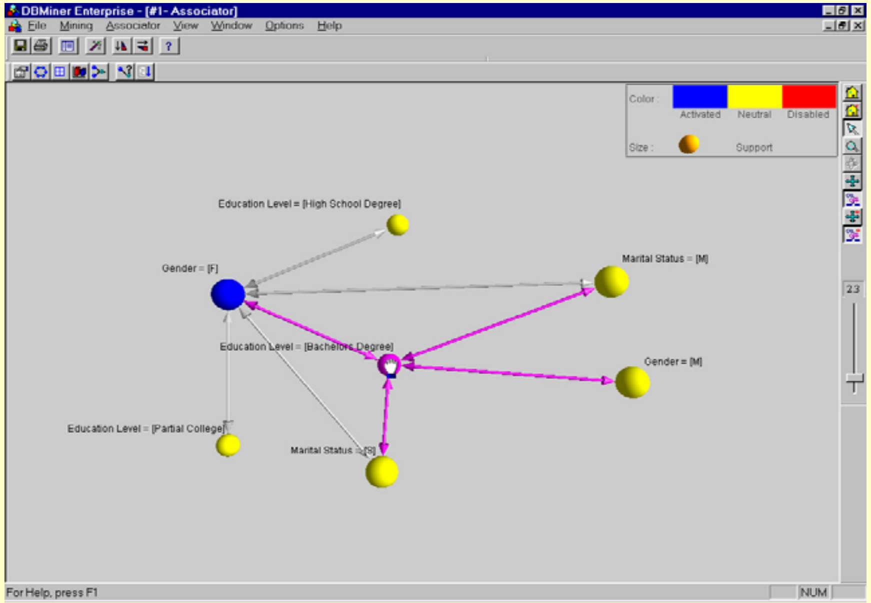
Association Rules Visualization

Size of ball equates to total support



Height equates to confidence

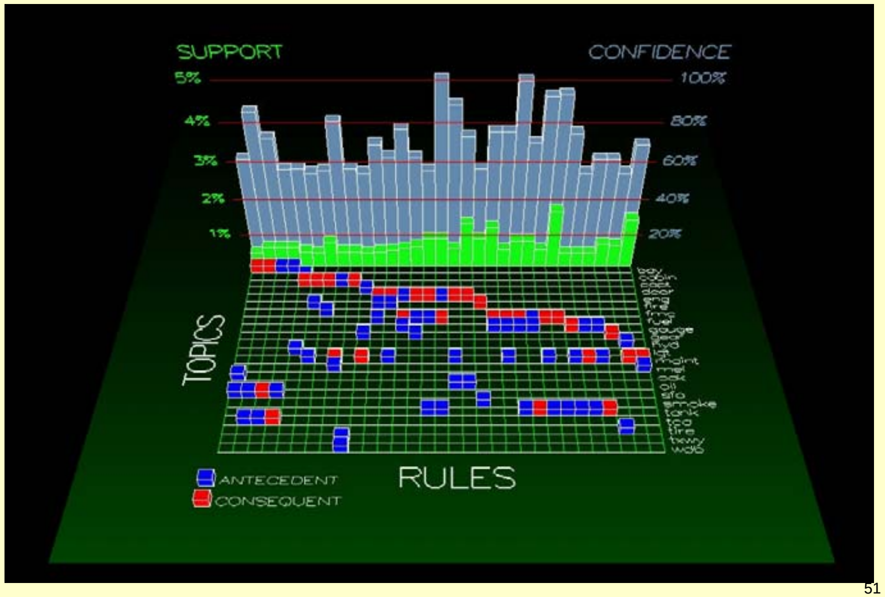
Association Rules Visualization - Ball graph



The Ball graph Explained

- A **ball graph** consists of a set of nodes and arrows. All the nodes are yellow, green or blue. The blue nodes are active nodes representing the items in the rule in which the user is interested. The yellow nodes are passive representing items related to the active nodes in some way. The green nodes merely assist in visualizing two or more items in either the head or the body of the rule.
- A **circular node** represents a *frequent* (*large*) data item. The volume of the ball represents the support of the item. Only those items which occur sufficiently frequently are shown
- An **arrow** between two nodes represents the *rule implication* between the two items. An arrow will be drawn only when the support of a rule is no less than the ***minimum support***

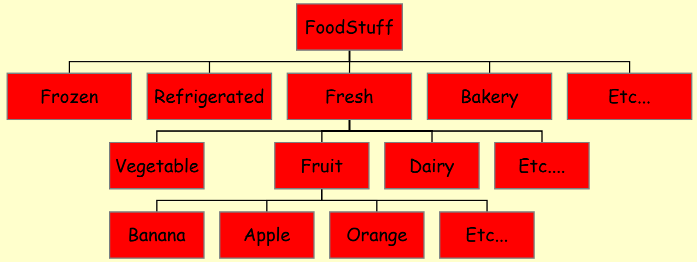
Association Rules Visualization



Mining Association Rules

- What is Association rule mining
- Apriori Algorithm
- FP-tree Algorithm
- Additional Measures of rule interestingness
- Advanced Techniques

Multiple-Level Association Rules



- **Fresh $\Rightarrow$ Bakery [20%, 60%]**
- **Dairy $\Rightarrow$ Bread [6%, 50%]**
- **Fruit $\Rightarrow$ Bread [1%, 50%] is not valid**

Items often form hierarchy.
Flexible support settings: Items at the lower level are expected to have lower support.
Transaction database can be encoded based on dimensions and levels explore shared multi-level mining

Multi-Dimensional Association Rules

- Single-dimensional rules:
  - buys(X, "milk") $\Rightarrow$ buys(X, "bread")
- Multi-dimensional rules: $\geq$ 2 dimensions or predicates
  - Inter-dimension association rules (no repeated predicates)
    - age(X,"19-25") $\wedge$ occupation(X,"student") $\Rightarrow$ buys(X,"coke")
  - hybrid-dimension association rules (repeated predicates)
    - age(X,"19-25") $\wedge$ buys(X, "popcorn") $\Rightarrow$ buys(X, "coke"

Quantitative Association Rules

age(X,"30-34") $\wedge$ income(X,"24K - 48K") $\Rightarrow$ buys(X,"high resolution TV")

Mining Sequential Patterns

10% of customers bought

"Foundation" and "Ringworld" in one transaction,

followed by

"Ringworld Engineers" in another transaction.

Sequential Pattern Mining

- Given
  - A database of customer transactions ordered by increasing transaction time
  - Each transaction is a set of items
  - A sequence is an ordered list of itemsets
- Example:
  - 10% of customers bought "Foundation" and "Ringworld" in one transaction, followed by "Ringworld Engineers" in another transaction.
  - 10% is called the support of the pattern

  (a transaction may contain more books than those in the pattern)
- Problem
  - Find all sequential patterns supported by more than a user-specified percentage of data sequences

Application Difficulties

- Wal-Mart knows that customers who buy Barbie dolls (it sells one every 20 seconds) have a 60% likelihood of buying one of three types of candy bars. What does Wal-Mart do with information like that?
- 'I don't have a clue,' says Wal-Mart's chief of merchandising, Lee Scott.
- See - KDnuggets 98:01 for many ideas www.kdnuggets.com/news/98/n01.html

Some Suggestions

- By increasing the price of Barbie doll and giving the type of candy bar free, wal-mart can reinforce the buying habits of that particular types of buyer
- Highest margin candy to be placed near dolls.
- Special promotions for Barbie dolls with candy at a slightly higher margin.
- Take a poorly selling product X and incorporate an offer on this which is based on buying Barbie and Candy. If the customer is likely to buy these two products anyway then why not try to increase sales on X?
- Probably they can not only bundle candy of type A with Barbie dolls, but can also introduce new candy of Type N in this bundle while offering discount on whole bundle. As bundle is going to sell because of Barbie dolls & candy of type A, candy of type N can get free ride to customers houses. And with the fact that you like something, if you see it often, Candy of type N can become popular.

References

- Jiawei Han and Micheline Kamber, "Data Mining: Concepts and Techniques", 2000
- Vipin Kumar and Mahesh Joshi, "Tutorial on High Performance Data Mining ", 1999
- Rakesh Agrawal, Ramakrishnan Srikan, "Fast Algorithms for Mining Association Rules", Proc VLDB, 1994 (http://www.cs.tau.ac.il/~fiat/dmsem03/Fast%20Algorithms%20for%20Mining%20Association%20Rules.ppt)
- Alípio Jorge, "selecção de regras: medidas de interesse e meta queries", (http://www.liacc.up.pt/~amjorge/Aulas/madsad/ecd2/ecd2_Aulas_AR_3_2003.pdf)

Thank you !!!