Data Mining: Concepts and Techniques

— Slides for Textbook —
— Chapter 6 & 7 —

©Jiawei Han and Micheline Kamber
Intelligent Database Systems Research Lab
School of Computing Science
Simon Fraser University, Canada
http://www.cs.sfu.ca

Chapter 6: Mining Association Rules in Large Databases

- Association rule mining
- Mining single-dimensional Boolean association rules from transactional databases
- Summary

Chapter 6: Mining Association Rules in Large Databases

- Association rule mining
  - Basic Concepts
  - Frequent Patterns
  - Association Rules
  - Support and Confidence
  - Road map

Chapter 6: Mining Association Rules in Large Databases

- Association rule mining
- Mining single-dimensional Boolean association rules from transactional databases
- Summary

Property of Frequent Patterns

- The downward closure (also called "Apriori") property of frequent patterns
  - If {beer, diaper, nuts} is frequent, so is {beer, diaper}
  - Every transaction containing {beer, diaper, nuts} also contains {beer, diaper}
  - Apriori property: Any **subset** of a frequent itemset **must be frequent**
- Efficient mining methodology
  - If any subset of an itemset S is infrequent, then there is no chance for S to be frequent—why do we even have to consider S? (Pruning)

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($\{A \cup C\}$) = 50%
- confidence = support($\{A \cup C\}$)/support($\{A\}$) = 66.6%

The Apriori principle:

- Any subset of a frequent itemset must be frequent

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
    - i.e., if {*AB*} is a frequent itemset, both {*A*} and {*B*} should be a frequent itemset
  - Iteratively find frequent itemsets with cardinality from 1 to *k* (*k*-itemset)
- Use the frequent itemsets to generate association rules
  - Note: If there is any itemset which is infrequent, its superset should not even be generated

The Apriori Algorithm

- 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
- Pseudo-code:

C_k: Candidate itemset of size k
L_k: frequent itemset of size k

L_1 = {frequent items};
for (k = 1; L_k != $\emptyset$; 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 $\cup_k L_k$;

The Apriori Algorithm — Example

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}

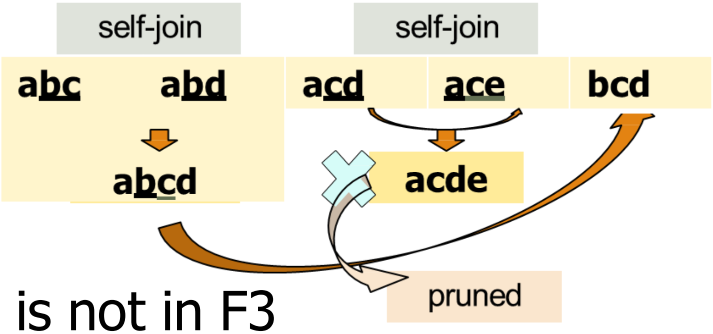
Scan D

L_3

itemset	sup
{2 3 5}	2

Apriori: Implementation Tricks

- How to generate candidates?
  - Step 1: self-joining Fk
  - Step 2: pruning
- Example of candidate-generation
  - F3 = {abc, abd, acd, ace, bcd}
  - Self-joining: F3*F3
    - abcd from abc and abd
    - acde from acd and ace
  - Pruning:
    - acde is removed because ade is not in F3
    - C4 = {abcd}



Mining Frequent Patterns Without Candidate Generation: FP-tree

- Compress a large database into a compact, Frequent-Pattern tree (FP- tree) structure
  - highly condensed, but complete for frequent pattern mining
  - avoid costly database scans
- Develop an efficient, FP-tree-based frequent pattern mining method
  - A divide-and-conquer methodology: decompose mining tasks into smaller ones
  - Avoid candidate generation: sub-database test only

FP-Growth Method: Construction of FP-Tree

1. Create the root of the tree, labeled with “null”.
2. Scan the database D a second time. (First time scanned it to create 1-itemset and then list L).
3. The items in each transaction are processed in L order (i.e. sorted order).
4. A branch is created for each transaction with items having their support count separated by colon.
5. Whenever the same node is encountered in another transaction, just increment the support count of the common node or Prefix.
6. To facilitate tree traversal, an item header table is built so that each item points to its occurrences in the tree via a chain of node-links.
7. The problem of mining frequent patterns in database is transformed to that of mining the FP-Tree.

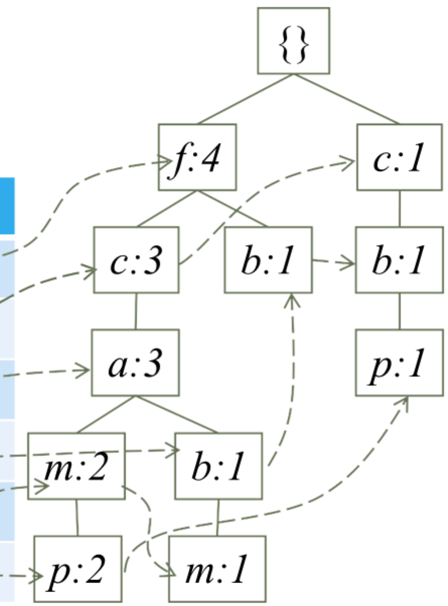
Example: Construct FP-tree from a Transactional DB

TID	Items in the Transaction	Ordered, frequent items
100	$\{f, a, c, d, g, i, m, p\}$	$\{f, c, a, m, p\}$
200	$\{a, b, c, f, l, m, o\}$	$\{f, c, a, b, m\}$
300	$\{b, f, h, j, o, w\}$	$\{f, b\}$
400	$\{b, c, k, s, p\}$	$\{c, b, p\}$
500	$\{a, f, c, e, l, p, m, n\}$	$\{f, c, a, m, p\}$

1. Scan DB once, find single item frequent pattern: Let min_sup = 3
   f:4, a:3, c:4, b:3, m:3, p:3
2. Sort frequent items in frequency descending order, f-list
   F-list = f-c-a-b-m-p
3. Scan DB again, construct FP-tree

Header Table

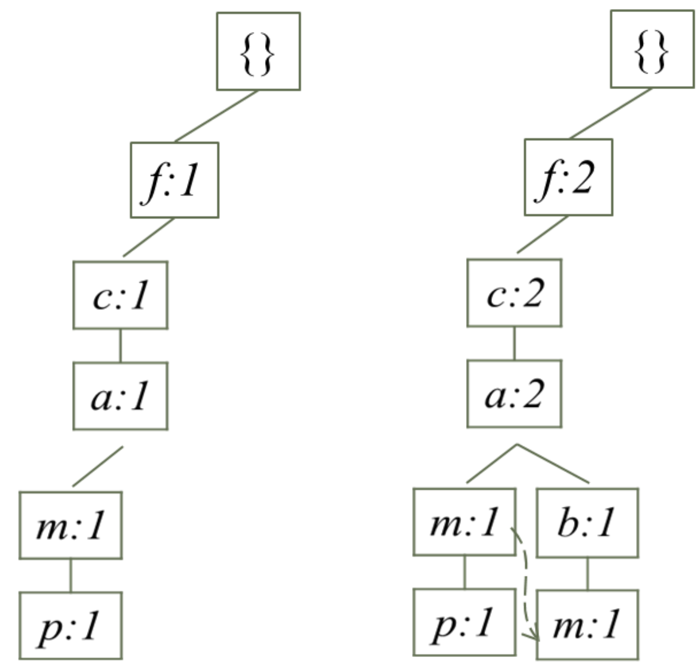
Item	Frequency	Header
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



Constructing the FP-Tree

TID	Items in the Transaction	Ordered, frequent items
100	$\{f, a, c, d, g, i, m, p\}$	$\{f, c, a, m, p\}$
200	$\{a, b, c, f, l, m, o\}$	$\{f, c, a, b, m\}$
300	$\{b, f, h, j, o, w\}$	$\{f, b\}$
400	$\{b, c, k, s, p\}$	$\{c, b, p\}$
500	$\{a, f, c, e, l, p, m, n\}$	$\{f, c, a, m, p\}$

Item	Frequency	Header
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	

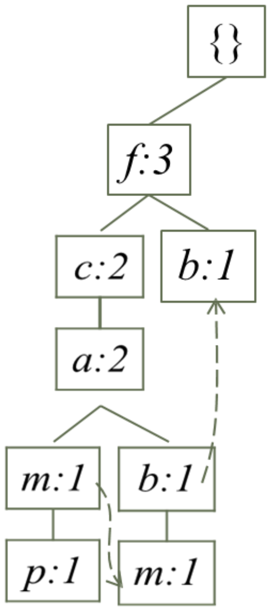


{f, c, a, m, p} **{f, c, a, b, m}**

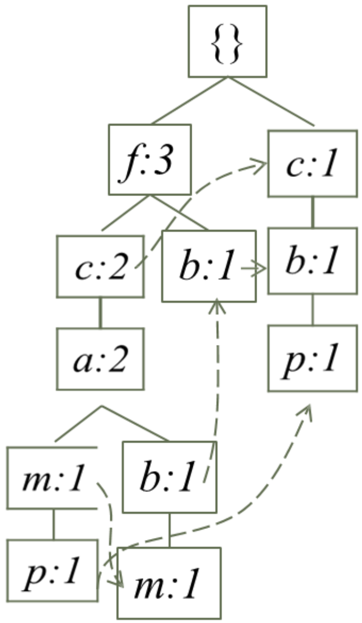
Constructing the FP-Tree

TID	Items in the Transaction	Ordered, frequent items
100	$\{f, a, c, d, g, i, m, p\}$	$\{f, c, a, m, p\}$
200	$\{a, b, c, f, l, m, o\}$	$\{f, c, a, b, m\}$
300	$\{b, f, h, j, o, w\}$	$\{f, b\}$
400	$\{b, c, k, s, p\}$	$\{c, b, p\}$
500	$\{a, f, c, e, l, p, m, n\}$	$\{f, c, a, m, p\}$

Item	Frequency	Header
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



{f, b}

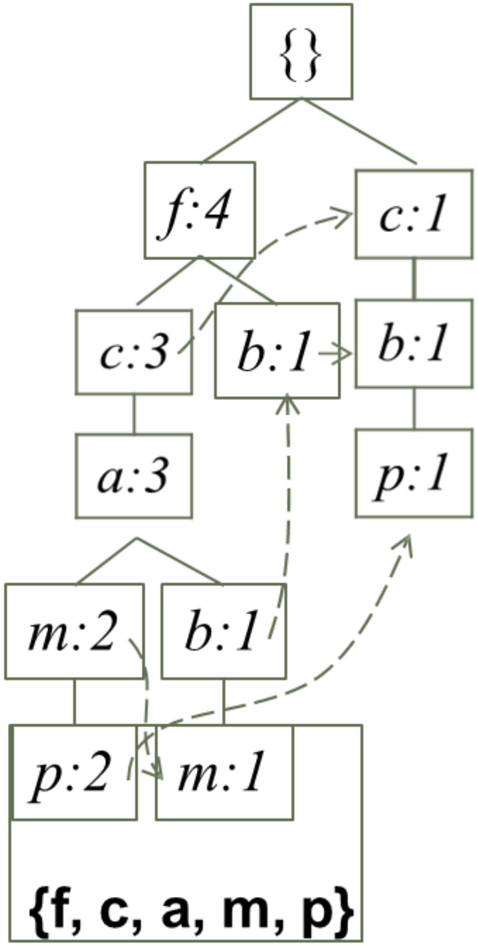


{c, b, p}

Constructing the FP-Tree

TID	Items in the Transaction	Ordered, frequent items
100	$\{f, a, c, d, g, i, m, p\}$	$\{f, c, a, m, p\}$
200	$\{a, b, c, f, l, m, o\}$	$\{f, c, a, b, m\}$
300	$\{b, f, h, j, o, w\}$	$\{f, b\}$
400	$\{b, c, k, s, p\}$	$\{c, b, p\}$
500	$\{a, f, c, e, l, p, m, n\}$	$\{f, c, a, m, p\}$

Item	Frequency	Header
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	

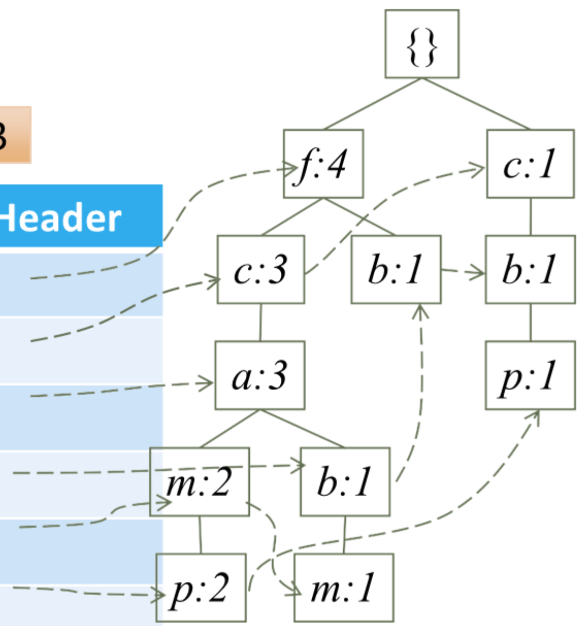


Generate Conditional Pattern Bases

- Pattern mining can be partitioned according to current patterns
  - Patterns containing p: p's conditional database: fcam:2, cb:1
  - Patterns having m but no p : m's conditional database: fca:2, fcab:1
  - …… ……

min_support = 3

Item	Frequency	Header
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	

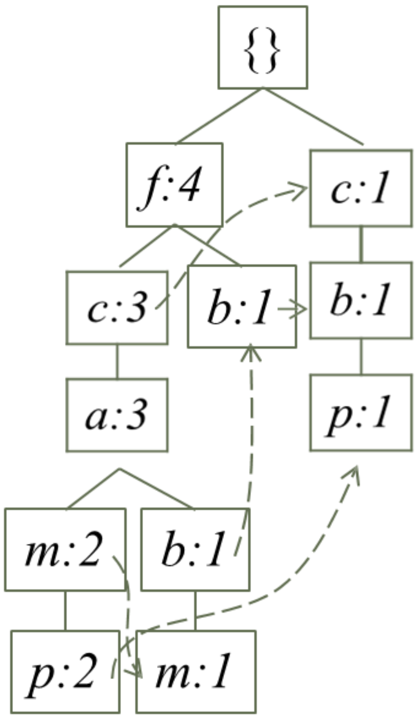


Conditional pattern bases

Item	Conditional pattern base
c	$f:3$
a	$fc:3$
b	$fca:1, f:1, c:1$
m	$fca:2, fcab:1$
p	$fcam:2, cb:1$

Generate Conditional FP-Tree

- Calculate recurring items in conditional pattern base following each item

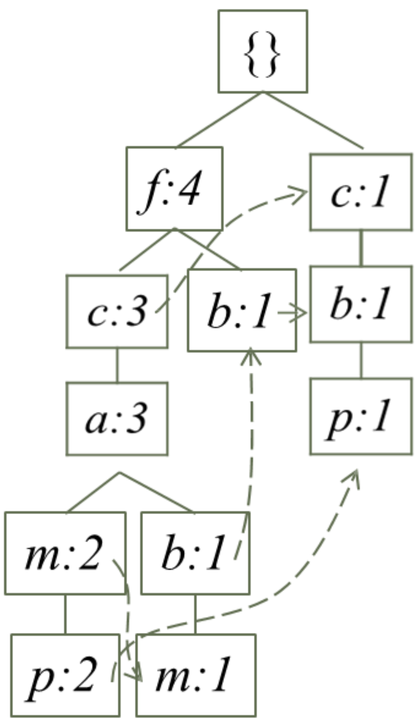


min_support = 3

Item	Conditional pattern base	Conditional FP-Tree
c	f:3	f:3
a	fc:3	f:3, c:3
b	fca:1, f:1, c:1	f:2, c:2, a:1
m	fca:2, $fcab$:1	f:3, c:3, a:3, b:1
p	$fcam$:2, cb:1	f:2, c:3, a:2, m:2, b:1

Generate Frequent Patterns

- Get itemsets for each conditional FP-tree

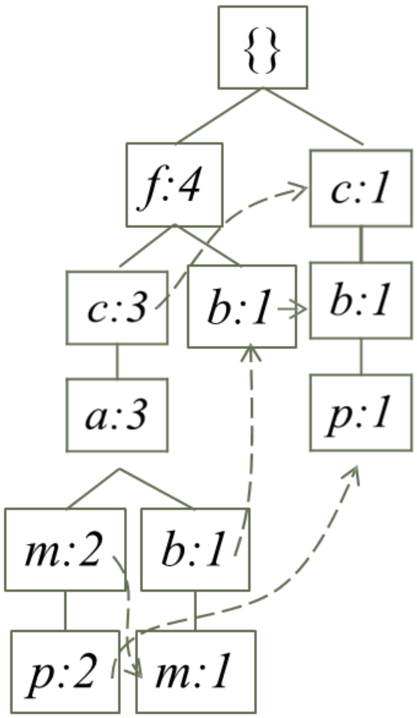


min_support = 3

Itm	Conditional pattern base	Conditional FP-Tree	Frequent Patterns
c	f:3	f:3	{f,c:3},{c:3}
a	fc:3	f:3, c:3	{f,c,a:3},{f,a:3},{c,a:3},{a:3}
b	fca:1, f:1, c:1	~~f:2, c:2, a:1~~	{b:3}
m	fca:2,fcab:1	f:3, c:3, a:3, ~~b:1~~	{f,c,a,m:3},{f,c,m:3}, {f,a,m:3},{c,a,m:3},{f,m:3}, {c,m:3},{a,m:3},{m:3}
p	fcam:2,cb:1	~~f:2~~, c:3, ~~a:2, m:2, b:1~~	{c,p:3},{p:3}

Generate Association Rules

- Create rules for each frequent itemset



Frequent Patterns
{f,c:3},{c:3}
{f,c,a:3},{f,a:3},{c,a:3},{a:3}
{b:3}
{f,c,a,m:3},{f,c,m:3}, {f,a,m:3},{c,a,m:3},{f,m:3}, {c,m:3},{a,m:3},{m:3}
{c,p:3},{p:3}

Consider this itemset:{f,c,a:3}
Generate all subsets and rules:

f -> c^a
c -> f^a
a -> f^c
f^c -> a
f^a -> c
c^a -> f
~~*f^c^a*~~

Calculate Confidence

- Calculate confidence for each rule and check minimum

Item	Frequency
f	4
c	4
a	3
b	3
m	3
p	3

Frequent Patterns
$\{f,c{:}3\}$
$\{f,c,a{:}3\}$,$\{f,a{:}3\}$,$\{c,a{:}3\}$,{a:3}

X -> Y

confidence, *c,* conditional probability that a transaction having X also contains *Y*

C = (count of X & Y) / (Count of X)

Consider this itemset:$\{f,c,a{:}3\}$

Generate all subsets and rules:

f -> c^a *conf. = 3/4*

c -> f^a *conf. = 3/4*

a -> f^c *conf. = 3/3*

f^c -> a *conf. = 3/3*

f^a -> c *conf. = 3/3*

c^a -> f *conf. = 3/3*

~~*f^c^a*~~

How to Judge if a Rule/Pattern Is Interesting?

- Pattern-mining will generate a large set of patterns/rules
  - Not all the generated patterns/rules are interesting
- Interestingness measures: Objective vs. subjective
  - Objective interestingness measures: Based on threshold values controlled by the user.
    - Support, confidence, correlation, …
  - Subjective interestingness measures: Often based on earlier user experiences and beliefs
    - Query-based: Relevant to a user's particular request
    - Against one's knowledge-base: unexpected, freshness, timeliness
    - Visualization tools: Multi-dimensional, interactive examination

Support and confidence

- If confidence gets a value of 100 % the rule is an exact rule
- Even if confidence reaches high values the rule is not useful unless the support value is high as well
- Rules that have both high confidence and support are called strong rules
- But strong rules are not necessarily interesting.

Limitation of the Support-Confidence Framework

- Are s and c interesting in association rules: "A⟹B" [s,c]?
- Example: Suppose one school may have the following statistics on # of students who may play basketball and/or eat cereal:

	play-basketball	not play-basketball	sum (row)
eat-cereal	400	350	750
not eat-cereal	200	50	250
sum (col.)	600	400	1000

2-way contingency table

- Association rule mining may generate the following:
- play-basketball ⟹ eat-cereal [40%, 66.7%] (higher s & c)
- Looks good. But if you generate another rule
- ¬ play-basketball ⟹ eat-cereal [35%, 87.5%] (high s & c)
- These two rules confuse the cereal company.

Interestingness Measure: Lift

- Measure of dependent/correlated events: lift
  $lift(B,C) = \frac{c(B \rightarrow C)}{s(C)} = \frac{s(B \cup C)}{s(B) \times s(C)}$
- lift(B,C) may tell how B and C are correlated
  - lift(B,C)=1: B and C are independent
  - > 1: positively correlated
  - < 1: negatively correlated

Lift is more telling than *s* & *c*

	B	$\neg B$	Σ_{row}
C	400	350	750
$\neg C$	200	50	250
Σ_{col}	600	400	1000

- For our example,
  $lift(B,C) = \frac{400/1000}{600/1000 \times 750/1000} = 0.89$
  $lift(B,\neg C) = \frac{200/1000}{600/1000 \times 250/1000} = 1.33$
- Thus, B and C are negatively correlated since lift(B, C) < 1;
  - B and ¬C are positively correlated since lift(B, ¬C) > 1

Is Lift Always A Good Measure?

- Null transactions: Transactions that contain neither B nor C
- Let's examine the dataset D
- BC (100) is much rarer than B¬C (1000) and ¬BC (1000), but there are many ¬B¬C (100000)
- Unlikely B & C will happen together!
- But, Lift(B, C) = 8.44 >> 1 (Lift shows B and C are strongly positively correlated!)

	B	$\neg B$	Σ_{row}
C	100	1000	1100
$\neg C$	1000	100000	101000
Σ_{col}	1100	101000	102100

null transactions

Interestingness Measures & Null-Invariance

- Null invariance: Value does not change with the # of null-transactions
- A few interestingness measures: Some are null invariant

Measure	Definition	Range	Null-Invariant
$\chi^2(A,B)$	$\sum_{i,j=0,1} \frac{(e(a_i b_j)-o(a_i b_j))^2}{e(a_i b_j)}$	$[0,\infty]$	No
$Lift(A,B)$	$\frac{s(A\cup B)}{s(A)\times s(B)}$	$[0,\infty]$	No
$AllConf(A,B)$	$\frac{s(A\cup B)}{max\{s(A),s(B)\}}$	$[0,1]$	Yes
$Jaccard(A,B)$	$\frac{s(A\cup B)}{s(A)+s(B)-s(A\cup B)}$	$[0,1]$	Yes
$Cosine(A,B)$	$\frac{s(A\cup B)}{\sqrt{s(A)\times s(B)}}$	$[0,1]$	Yes
$Kulczynski(A,B)$	$\frac{1}{2}(\frac{s(A\cup B)}{s(A)}+\frac{s(A\cup B)}{s(B)})$	$[0,1]$	Yes
$MaxConf(\text{A, B})$	$max\{\frac{s(A)}{s(A\cup B)},\frac{s(B)}{s(A\cup B)}\}$	$[0,1]$	Yes

χ^2 and *lift* are not *null-invariant*

Jaccard, cosine, AllConf, MaxConf, and *Kulczynski* are *null-invariant measures*

ExKulc: 0- negatively correlated, 0.5- neutral, 1- positively correlated

Null Invariance: An Important Property

milk vs. coffee contingency table	milk	¬milk	Σ_{row}
coffee	mc	$\neg mc$	c
¬coffee	$m\neg c$	$\neg m \neg c$	$\neg c$
Σ_{col}	m	$\neg m$	Σ

Dataset	mc	$\neg mc$	$m\neg c$	$\neg m \neg c$
D_1	10,000	1,000	1,000	100,000
D_2	10,000	1,000	1,000	100
D_3	100	1,000	1,000	100,000
D_4	1,000	1,000	1,000	100,000
D_5	1,000	100	10,000	100,000
D_6	1,000	10	100,000	100,000

- Let's look at another ex. Check the first 4 data sets.
- m and c are positively associated in D1 and D2 , because mc(10,000) is considerably greater than $m\bar{c}$(1000) and $\bar{m}c$(1000)
- Negatively associated in D3, because mc(100) is considerably lesser than $m\bar{c}$(1000) and $\bar{m}c$(1000)
- Neutral in D4, because mc(1000) is equal to $m\bar{c}$(1000) and $\bar{m}c$(1000)

Null Invariance: An Important Property

- Why is null invariance crucial for the analysis of massive transaction data?
  - Many transactions may contain neither milk nor coffee

milk vs. coffee contingency table

	milk	¬milk	Σ_{row}
coffee	mc	$\neg mc$	c
¬coffee	$m\neg c$	$\neg m \neg c$	$\neg c$
Σ_{col}	m	$\neg m$	Σ

- Lift is not null-invariant: not good to evaluate data that contain too many (D1) or too few (D2) null transactions
- Many measures are not null-invariant

Dataset	mc	$\neg mc$	$m\neg c$	$\neg m \neg c$	χ^2	$Lift$
D_1	10,000	1,000	1,000	100,000	90557	9.26
D_2	10,000	1,000	1,000	100	0	1
D_3	100	1,000	1,000	100,000	670	8.44
D_4	1,000	1,000	1,000	100,000	24740	25.75
D_5	1,000	100	10,000	100,000	8173	9.18
D_6	1,000	10	100,000	100,000	965	1.97

Null-transactions w.r.t. m and c

Comparison of Null-Invariant Measures

- Not all null-invariant measures are created equal
- D4-D6 differentiate the null-invariant measures
- Imbalance Ratio (IR) can measure which is better

2-variable contingency table

	milk	¬milk	Σ_{row}
coffee	mc	$\neg mc$	c
¬coffee	$m\neg c$	$\neg m\neg c$	$\neg c$
Σ_{col}	m	$\neg m$	Σ

All 5 are null-invariant

Dataset	mc	$\neg mc$	$m\neg c$	$\neg m\neg c$	$AllConf$	$Jaccard$	$Cosine$	$Kulc$	$MaxConf$
D_1	10,000	1,000	1,000	100,000	0.91	0.83	0.91	0.91	0.91
D_2	10,000	1,000	1,000	100	0.91	0.83	0.91	0.91	0.91
D_3	100	1,000	1,000	100,000	0.09	0.05	0.09	0.09	0.09
D_4	1,000	1,000	1,000	100,000	0.5	0.33	0.5	0.5	0.5
D_5	1,000	100	10,000	100,000	0.09	0.09	0.29	0.5	0.91
D_6	1,000	10	100,000	100,000	0.01	0.01	0.10	0.5	0.99

Subtle: They disagree on most cases

What Measures to Choose for Effective Pattern Evaluation?

- Null value cases are predominant in many large datasets
- Neither milk nor coffee is in most of the baskets; neither Mike nor Jim is an author in most of the papers; ……
- Null-invariance is an important property
- Lift, χ^2 and cosine are good measures if null transactions are not predominant
- Otherwise, choose others to judge the interestingness of a pattern (e.g. Kulczynski + Imbalance Ratio)

Summary

- Basic Concepts:
  - Frequent Patterns, Association Rules, Closed Patterns and Max-Patterns
- Frequent Itemset Mining Methods
  - The Downward Closure Property and The Apriori Algorithm
  - FPGrowth: A Frequent Pattern-Growth Approach
- Which Patterns Are Interesting?—Pattern Evaluation Methods
  - Interestingness Measures: Lift and χ^2
  - Null-Invariant Measures
  - Comparison of Interestingness Measures

References

- R. Agarwal, C. Aggarwal, and V. V. V. Prasad. A tree projection algorithm for generation of frequent itemsets. In Journal of Parallel and Distributed Computing (Special Issue on High Performance Data Mining), 2000.
- R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. SIGMOD'93, 207-216, Washington, D.C.
- R. Agrawal and R. Srikant. Fast algorithms for mining association rules. VLDB'94 487-499, Santiago, Chile.
- R. Agrawal and R. Srikant. Mining sequential patterns. ICDE'95, 3-14, Taipei, Taiwan.
- R. J. Bayardo. Efficiently mining long patterns from databases. SIGMOD'98, 85-93, Seattle, Washington.
- S. Brin, R. Motwani, and C. Silverstein. Beyond market basket: Generalizing association rules to correlations. SIGMOD'97, 265-276, Tucson, Arizona.
- S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket analysis. SIGMOD'97, 255-264, Tucson, Arizona, May 1997.
- K. Beyer and R. Ramakrishnan. Bottom-up computation of sparse and iceberg cubes. SIGMOD'99, 359-370, Philadelphia, PA, June 1999.
- D.W. Cheung, J. Han, V. Ng, and C.Y. Wong. Maintenance of discovered association rules in large databases: An incremental updating technique. ICDE'96, 106-114, New Orleans, LA.
- M. Fang, N. Shivakumar, H. Garcia-Molina, R. Motwani, and J. D. Ullman. Computing iceberg queries efficiently. VLDB'98, 299-310, New York, NY, Aug. 1998.

References (2)

- G. Grahne, L. Lakshmanan, and X. Wang. Efficient mining of constrained correlated sets. ICDE'00, 512-521, San Diego, CA, Feb. 2000.
- Y. Fu and J. Han. Meta-rule-guided mining of association rules in relational databases. KDOOD'95, 39-46, Singapore, Dec. 1995.
- T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. SIGMOD'96, 13-23, Montreal, Canada.
- E.-H. Han, G. Karypis, and V. Kumar. Scalable parallel data mining for association rules. SIGMOD'97, 277-288, Tucson, Arizona.
- J. Han, G. Dong, and Y. Yin. Efficient mining of partial periodic patterns in time series database. ICDE'99, Sydney, Australia.
- J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. VLDB'95, 420-431, Zurich, Switzerland.
- J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. SIGMOD'00, 1-12, Dallas, TX, May 2000.
- T. Imielinski and H. Mannila. A database perspective on knowledge discovery. Communications of ACM, 39:58-64, 1996.
- M. Kamber, J. Han, and J. Y. Chiang. Metarule-guided mining of multi-dimensional association rules using data cubes. KDD'97, 207-210, Newport Beach, California.
- M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A.I. Verkamo. Finding interesting rules from large sets of discovered association rules. CIKM'94, 401-408, Gaithersburg, Maryland.

References (3)

- F. Korn, A. Labrinidis, Y. Kotidis, and C. Faloutsos. Ratio rules: A new paradigm for fast, quantifiable data mining. VLDB'98, 582-593, New York, NY.
- B. Lent, A. Swami, and J. Widom. Clustering association rules. ICDE'97, 220-231, Birmingham, England.
- H. Lu, J. Han, and L. Feng. Stock movement and n-dimensional inter-transaction association rules. SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD'98), 12:1-12:7, Seattle, Washington.
- H. Mannila, H. Toivonen, and A. I. Verkamo. Efficient algorithms for discovering association rules. KDD'94, 181-192, Seattle, WA, July 1994.
- H. Mannila, H Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. Data Mining and Knowledge Discovery, 1:259-289, 1997.
- R. Meo, G. Psaila, and S. Ceri. A new SQL-like operator for mining association rules. VLDB'96, 122-133, Bombay, India.
- R.J. Miller and Y. Yang. Association rules over interval data. SIGMOD'97, 452-461, Tucson, Arizona.
- R. Ng, L. V. S. Lakshmanan, J. Han, and A. Pang. Exploratory mining and pruning optimizations of constrained associations rules. SIGMOD'98, 13-24, Seattle, Washington.
- N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. ICDT'99, 398-416, Jerusalem, Israel, Jan. 1999.

References (4)

- J.S. Park, M.S. Chen, and P.S. Yu. An effective hash-based algorithm for mining association rules. SIGMOD'95, 175-186, San Jose, CA, May 1995.
- J. Pei, J. Han, and R. Mao. CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets. DMKD'00, Dallas, TX, 11-20, May 2000.
- J. Pei and J. Han. Can We Push More Constraints into Frequent Pattern Mining? KDD'00. Boston, MA. Aug. 2000.
- G. Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. In G. Piatetsky-Shapiro and W. J. Frawley, editors, Knowledge Discovery in Databases, 229-238. AAAI/MIT Press, 1991.
- B. Ozden, S. Ramaswamy, and A. Silberschatz. Cyclic association rules. ICDE'98, 412-421, Orlando, FL.
- J.S. Park, M.S. Chen, and P.S. Yu. An effective hash-based algorithm for mining association rules. SIGMOD'95, 175-186, San Jose, CA.
- S. Ramaswamy, S. Mahajan, and A. Silberschatz. On the discovery of interesting patterns in association rules. VLDB'98, 368-379, New York, NY..
- S. Sarawagi, S. Thomas, and R. Agrawal. Integrating association rule mining with relational database systems: Alternatives and implications. SIGMOD'98, 343-354, Seattle, WA.
- A. Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases. VLDB'95, 432-443, Zurich, Switzerland.
- A. Savasere, E. Omiecinski, and S. Navathe. Mining for strong negative associations in a large database of customer transactions. ICDE'98, 494-502, Orlando, FL, Feb. 1998.

References (5)

- C. Silverstein, S. Brin, R. Motwani, and J. Ullman. Scalable techniques for mining causal structures. VLDB'98, 594-605, New York, NY.
- R. Srikant and R. Agrawal. Mining generalized association rules. VLDB'95, 407-419, Zurich, Switzerland, Sept. 1995.
- R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. SIGMOD'96, 1-12, Montreal, Canada.
- R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. KDD'97, 67-73, Newport Beach, California.
- H. Toivonen. Sampling large databases for association rules. VLDB'96, 134-145, Bombay, India, Sept. 1996.
- D. Tsur, J. D. Ullman, S. Abitboul, C. Clifton, R. Motwani, and S. Nestorov. Query flocks: A generalization of association-rule mining. SIGMOD'98, 1-12, Seattle, Washington.
- K. Yoda, T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Computing optimized rectilinear regions for association rules. KDD'97, 96-103, Newport Beach, CA, Aug. 1997.
- M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. Parallel algorithm for discovery of association rules. Data Mining and Knowledge Discovery, 1:343-374, 1997.
- M. Zaki. Generating Non-Redundant Association Rules. KDD'00. Boston, MA. Aug. 2000.
- O. R. Zaiane, J. Han, and H. Zhu. Mining Recurrent Items in Multimedia with Progressive Resolution Refinement. ICDE'00, 461-470, San Diego, CA, Feb. 2000.

http://www.cs.sfu.ca/~han/dmbook

Thank you !!!