COMP 345: Data Mining
Mining Frequent Patterns, Associations and Correlations

Slides Adapted From : Jiawei Han, Micheline Kamber & Jian Pei
Data Mining: Concepts and Techniques, 3rd ed.

1

Reminders

- Assignment 2 has been assigned
  - details on Course Website
  - Due Wed. Sept. 12th/Thurs. Sept. 13th at beginning of class
- Install WEKA by Tuesday, Sept. 11th.
  - Will need to bring laptops to class on Wed. Sept. 12th/Thurs. Sept. 13th – will be using WEKA

2

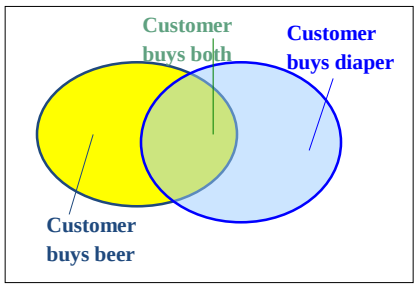
What Is Frequent Pattern Analysis?

- Frequent pattern: a pattern (a set of items, subsequences, substructures, etc.) that occurs frequently in a data set
- First proposed by Agrawal, Imielinski, and Swami [AIS93] in the context of frequent itemsets and association rule mining
- Motivation: Finding inherent regularities in data
  - What products were often purchased together?— Beer and diapers?!
  - What are the subsequent purchases after buying a PC?
  - What kinds of DNA are sensitive to this new drug?
  - Can we automatically classify web documents?
- Applications
  - Basket data analysis, cross-marketing, catalog design, sale campaign analysis, Web log (click stream) analysis, and DNA sequence analysis.

3

Basic Concepts: Frequent Patterns

Tid	Items bought
10	Beer, Nuts, Diaper
20	Beer, Coffee, Diaper
30	Beer, Diaper, Eggs
40	Nuts, Eggs, Milk
50	Nuts, Coffee, Diaper, Eggs, Milk

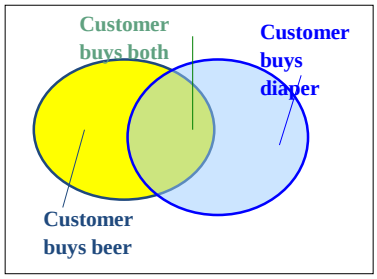


- itemset: A set of one or more items
- k-itemset X = $\{x_1, \ldots, x_k\}$
- *(absolute) support*, or, *support count* of X: Frequency or occurrence of an itemset X
- *(relative) support*, *s*, is the fraction of transactions that contains X (i.e., the probability that a transaction contains X)
- An itemset X is *frequent* if X's support is no less than a *minsup* threshold

4

Basic Concepts: Association Rules

Tid	Items bought
10	Beer, Nuts, Diaper
20	Beer, Coffee, Diaper
30	Beer, Diaper, Eggs
40	Nuts, Eggs, Milk
50	Nuts, Coffee, Diaper, Eggs, Milk



Find all the rules $X \rightarrow Y$ with minimum support and confidence

- support, *s*, probability that a transaction contains $X \cup Y$
- confidence, *c*, conditional probability that a transaction having X also contains *Y*

Let minsup = 50%, minconf = 50%

Freq. Pat.: Beer:3, Nuts:3, Diaper:4, Eggs:3, {Beer, Diaper}:3

Association rules: (many more!)

- *Beer* → *Diaper* (60%, 100%)
- *Diaper* → *Beer* (60%, 75%)

5

Closed Patterns and Max-Patterns

- A long pattern contains a combinatorial number of sub-patterns, e.g., $\{a_1, \ldots, a_{100}\}$ contains $\binom{100}{1} + \binom{100}{2} + \ldots + \binom{1}{1}\binom{0}{0}\binom{0}{0} = 2^{100} - 1 = 1.27*10^{30}$ sub-patterns!
- Solution: *Mine closed patterns and max-patterns instead*
- An itemset X is closed if X is *frequent* and there exists *no super-pattern* $Y \supset X$, *with the same support* as X (proposed by Pasquier, et al. @ ICDT'99)
- An itemset X is a max-pattern if X is frequent and there exists no frequent super-pattern $Y \supset X$ (proposed by Bayardo @ SIGMOD'98)
- Closed pattern is a lossless compression of freq. patterns
  - Reducing the # of patterns and rules

6

Closed Patterns and Max-Patterns

- Exercise: Suppose a DB contains only two transactions
  - $\langle a_1, \ldots, a_{100} \rangle$, $\langle a_1, \ldots, a_{50} \rangle$
  - Let min_sup = 1
- What is the set of closed itemset?
  - $\{a_1, \ldots, a_{100}\}$: 1
  - $\{a_1, \ldots, a_{50}\}$: 2
- What is the set of max-pattern?
  - $\{a_1, \ldots, a_{100}\}$: 1
- What is the set of all patterns?
  - $\{a_1\}$: 2, …, $\{a_1, a_2\}$: 2, …, $\{a_1, a_{51}\}$: 1, …, $\{a_1, a_2, \ldots, a_{100}\}$: 1
  - A big number: $2^{100} - 1$

7

The Downward Closure Property and Scalable Mining Methods

- The downward closure property of frequent patterns
  - Any subset of a frequent itemset must be frequent
  - If **{beer, diaper, nuts}** is frequent, so is **{beer, diaper}**
  - i.e., every transaction having {beer, diaper, nuts} also contains {beer, diaper}
- Scalable mining methods: Three major approaches
  - Apriori (Agrawal & Srikant@VLDB'94)
  - Freq. pattern growth (FPgrowth—Han, Pei & Yin @SIGMOD'00)
  - Vertical data format approach (Charm—Zaki & Hsiao @SDM'02)

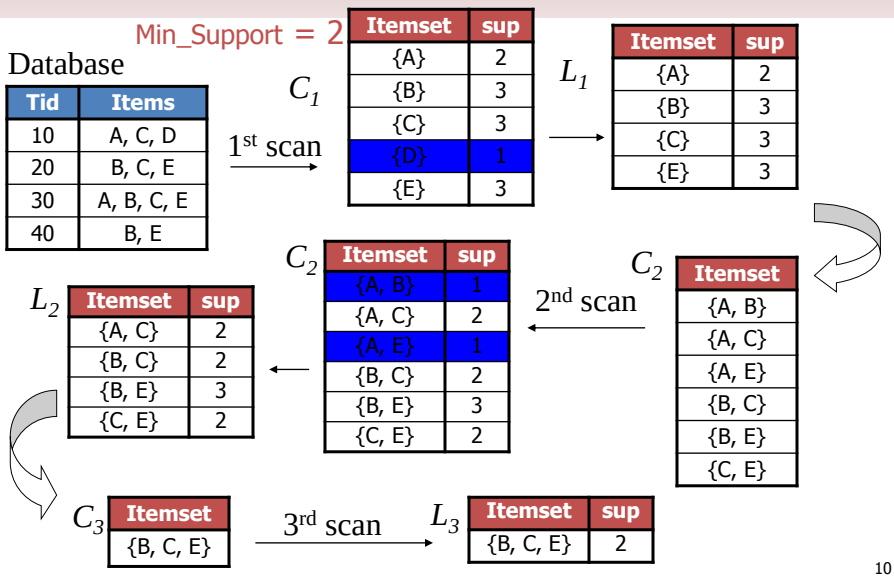
8

Apriori: A Candidate Generation & Test Approach

- Apriori pruning principle: If there is any itemset which is infrequent, its superset should not be generated/tested! (Agrawal & Srikant @VLDB'94, Mannila, et al. @ KDD' 94)
- Method:
  - Initially, scan DB once to get frequent 1-itemset
  - Generate length (k+1) candidate itemsets from length k frequent itemsets
  - Test the candidates against DB
  - Terminate when no frequent or candidate set can be generated

9

The Apriori Algorithm—An Example



10

The Apriori Algorithm (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 !=∅; 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 $\cup_k L_k$;

11

Implementation of Apriori

- How to generate candidates?
  - Step 1: self-joining L_k
  - Step 2: pruning
- Example of Candidate-generation
  - L_3={*abc, abd, acd, ace, bcd*}
  - Self-joining: L_3*L_3
    - *abcd* from *abc* and *abd*
    - *acde* from *acd* and *ace*
  - Pruning:
    - *acde* is removed because *ade* is not in L_3
  - C_4 = {*abcd*}

12

Further Improvement of the Apriori Method

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

13

Apriori Example

There's a haunted corn maze every Fall that sells lots of fun things. Items that can be purchased are:

1. Hot Cider, 2. Pumpkin, 3. Gourd, 4. Hayride, 5. Maze Tour

You are given the transaction data for a morning of sales. (Items referred to by #).

Sales ID	List of Item IDs	Sales ID	List of item IDs
Order 1	1, 2, 5	Order 6	2, 3
Order 2	2, 4	Order 7	1, 3
Order 3	2, 3	Order 8	1, 2, 3, 5
Order 4	1, 2, 4	Order 9	1, 2, 3
Order 5	1, 3		

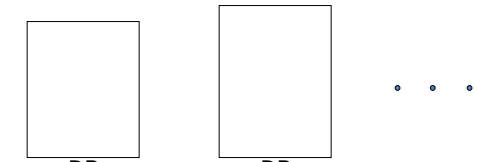
Assuming that minimum support = 2/9 (.222) and minimum confidence is 7/9 (.777)

1. Apply the Apriori algorithm to the dataset and identify ***all*** frequent k-itemsets
2. Find all ***strong*** association rules of the form X ^ Y -> Z

14

Partition: Scan Database Only Twice

- Any itemset that is potentially frequent in DB must be frequent in at least one of the partitions of DB
  - Scan 1: partition database and find local frequent patterns
  - Scan 2: consolidate global frequent patterns
- A. Savasere, E. Omiecinski and S. Navathe, *VLDB'95*



$sup_1(i) < \sigma DB_1$ $sup_2(i) < \sigma DB_2$ $sup_k(i) < \sigma DB_k$ $sup(i) < \sigma DB$

15

DHP: Reduce the Number of Candidates

- A *k*-itemset whose corresponding hashing bucket count is below the threshold cannot be frequent
  - Candidates: a, b, c, d, e
  - Hash entries
    - {ab, ad, ae}
    - {bd, be, de}
    - …
  - Frequent 1-itemset: a, b, d, e
  - ab is not a candidate 2-itemset if the sum of count of {ab, ad, ae} is below support threshold

count	itemsets
35	{ab, ad, ae}
88	{bd, be, de}
. . .	. . .
102	{yz, qs, wt}

Hash Table

- J. Park, M. Chen, and P. Yu. An effective hash-based algorithm for mining association rules. *SIGMOD'95*
  - *DHP – Direct Hashing and Pruning*

16

Sampling for Frequent Patterns

- Select a sample of original database, mine frequent patterns within sample using Apriori
- Scan database once to verify frequent itemsets found in sample, only *borders* of closure of frequent patterns are checked
  - Example: check *abcd* instead of *ab, ac, …, etc.*
- Scan database again to find missed frequent patterns
- H. Toivonen. Sampling large databases for association rules. In *VLDB'96*

17