Unit –III

Mining Frequent Patterns

Unit - III

- Basic Concepts
- Apriori Algorithm: Finding Frequent Itemsets by Confined Candidate Generation
- Generating Association Rules from Frequent Itemsets
- Mining Multilevel Associations
- Constraint-Based Frequent Pattern Mining

What Is Frequent Pattern Analysis?

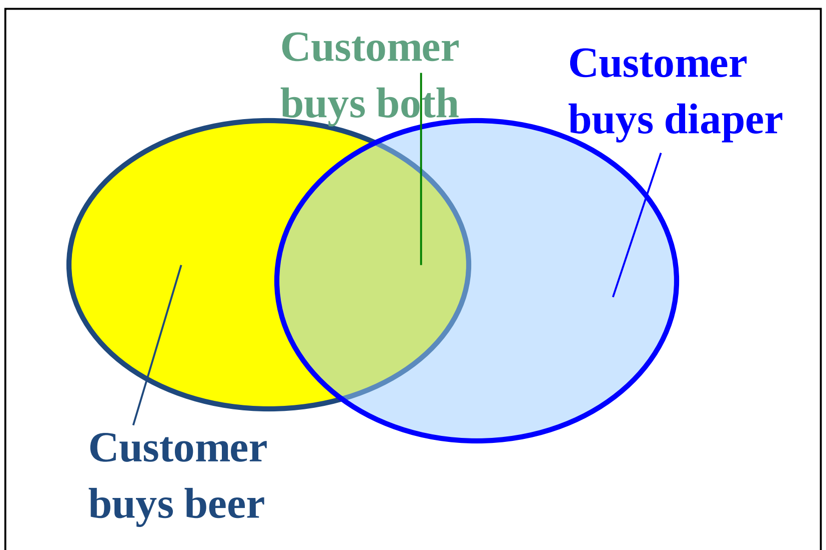
- Frequent pattern: a pattern (a set of items, subsequences, substructures, etc.) that occurs frequently in a data set
- First proposed by Agrawal, Imielinski, and Swami [AIS93] in the context of frequent itemsets and association rule mining
- Motivation: Finding inherent regularities in data
  - What 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.

Why Is Freq. Pattern Mining Important?

- Freq. pattern: An intrinsic and important property of datasets
- Foundation for many essential data mining tasks
  - Association, correlation, and causality analysis
  - Sequential, structural (e.g., sub-graph) patterns
  - Pattern analysis in spatiotemporal, multimedia, time-series, and stream data
  - Classification: discriminative, frequent pattern analysis
  - Cluster analysis: frequent pattern-based clustering
  - Data warehousing: iceberg cube and cube-gradient
  - Semantic data compression: fascicles
  - Broad applications

Basic Concepts: Frequent Patterns

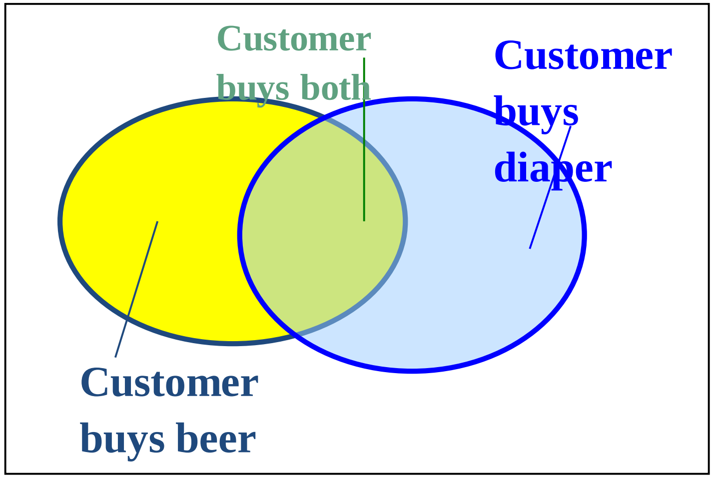
Tid	Items bought
10	Beer, Nuts, Diaper
20	Beer, Coffee, Diaper
30	Beer, Diaper, Eggs
40	Nuts, Eggs, Milk
50	Nuts, Coffee, Diaper, Eggs, Milk



- 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

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* → *Y* with minimum support and confidence
  - support, *s*, probability that a transaction contains X ∪ 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%)

Closed Patterns and Max-Patterns

- A long pattern contains a combinatorial number of sub-patterns, e.g., $\{a_1, \ldots, a_{100}\}$ contains $\binom{100}{1} + \binom{100}{2} + \ldots + \binom{100}{100} = 2^{100} - 1 = 1.27 \times 10^{30}$ 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 ⊃ 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 ⊃ X (proposed by Bayardo @ SIGMOD'98)
- Closed pattern is a lossless compression of freq. patterns
  - Reducing the # of patterns and rules

Closed Patterns and Max-Patterns

- Exercise. DB = {<a_1, …, a_{100}>, < a_1, …, a_{50}>}
  - Min_sup = 1.
- What is the set of closed itemset?
  - <a_1, …, a_{100}>: 1
  - < a_1, …, a_{50}>: 2
- What is the set of max-pattern?
  - <a_1, …, a_{100}>: 1
- What is the set of all patterns?
  - !!

Frequent Pattern Mining

- Frequent Patterns:

Frequent Patterns are patterns that occur frequently in data.

- Three types of frequent patterns
  - Frequent itemset
  - Frequent sequential pattern
  - Frequent structured pattern

Frequent Pattern Mining (cntd…)

- Frequent itemset. :A set of items, such as milk and bread, that appear frequently together in a transaction data set is a frequent itemset.

- Frequent sequential pattern : If a subsequence occurs frequently in a shopping history database, it is a frequent sequential pattern.

- Frequent structured pattern :If a substructure occurs frequently, it is called a frequent structured pattern

Frequent Pattern Mining (cntd…)

- Searches for recurring relationships in a given data set.
- Plays an essential role in associations mining.
- Helps in data classification, clustering and other data mining tasks

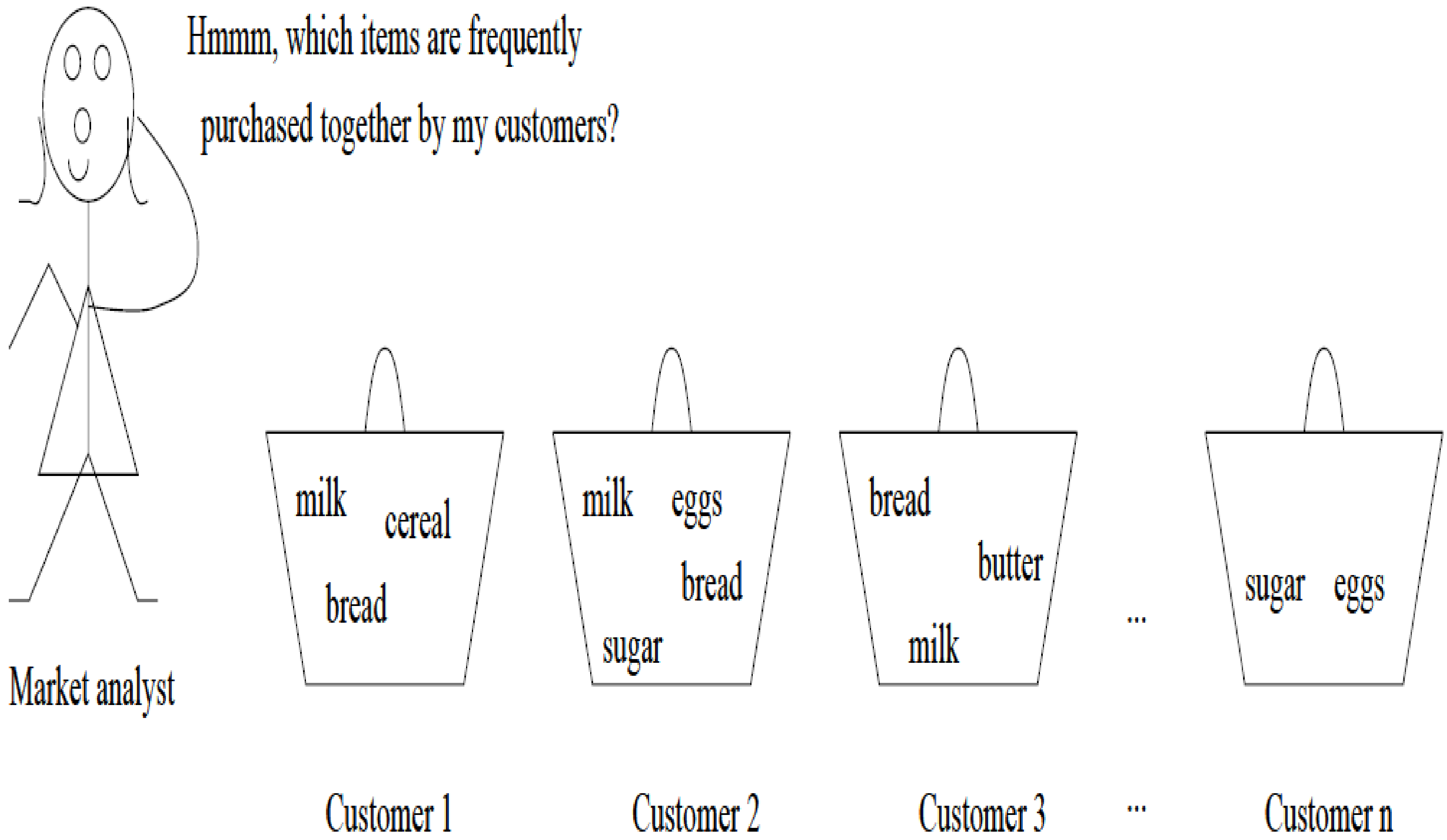
Market Basket Analysis

•The earliest form of frequent pattern mining is Market Basket Analysis.

•Consider shopping cart filled with several items.

•From marketing perspective, determining which items are frequently purchased together within the same transaction

Market Basket Analysis (cntd…)



Market Basket Analysis (cntd…)

- To categorize customer purchase behavior
- To identify actionable information
  - purchase profiles
  - profitability of each purchase profile
  - use for marketing
    - Store layouts
    - Design catalogs
    - select products for promotion
    - space allocation, product placement
- To plan marketing or advertising strategies.
- To plan which items to put on sale at reduced prices.

Transactions database Example 1

TID	Products
1	A, B, E
2	B, D
3	B, C
4	A, B, D
5	A, C
6	B, C
7	A, C
8	A, B, C, E
9	A, B, C

Attributes converted to binary flags

TID	A	B	C	D	E
1	1	1	0	0	1
2	0	1	0	1	0
3	0	1	1	0	0
4	1	1	0	1	0
5	1	0	1	0	0
6	0	1	1	0	0
7	1	0	1	0	0
8	1	1	1	0	1
9	1	1	1	0	0

Support and Confidence

$$support(A \Rightarrow B) = P(A \cup B)$$

$$confidence(A \Rightarrow B) = P(B|A) = \frac{support\_count(A \cup B)}{support\_count(A)}.$$

Transactions database Example 1

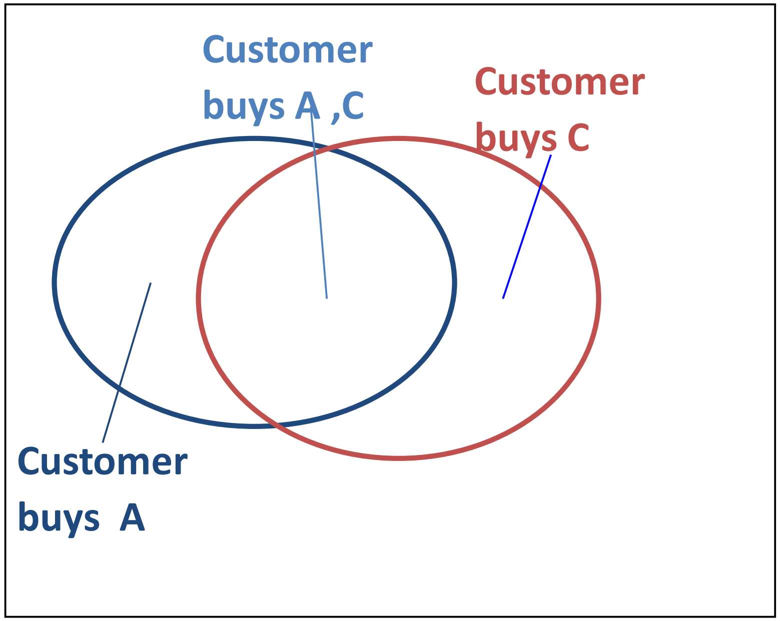
TID	Products
1	A, B, E
2	B, D
3	B, C
4	A, B, D
5	A, C
6	B, C
7	A, C
8	A, B, C, E
9	A, B, C

Examples:

$A \Rightarrow C$

Support: 4/9 = 44%

- Confidence: 4/6 = 66%



Market Basket Analysis (cntd…)

- LIMITATIONS
  - takes over 18 months to implement
  - market basket analysis only identifies hypotheses, which need to be tested
    - neural network, regression, decision tree analyses
  - measurement of impact needed
  - difficult to identify product groupings
  - complexity grows exponentially

Market Basket Analysis (cntd…)

- BENEFITS:
- simple computations
  - can be undirected (don’t have to have hypotheses before analysis)
  - different data forms can be analyzed

Apriori: A Candidate Generation & Test Approach

- Apriori Property: Any subset of a frequent itemset must be frequent
- 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

Apriori Algorithm: Finding Frequent Itemsets by Confined Candidate Generation—An Example

$Sup_{min} = 2$

Database TDB

Tid	Items
10	A, C, D
20	B, C, E
30	A, B, C, E
40	B, E

1st scan

C_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

L_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{E}	3

C_2

Itemset
{A, B}
{A, C}
{A, E}
{B, C}
{B, E}
{C, E}

2nd scan

C_2

Itemset	sup
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

L_2

Itemset	sup
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

C_3

Itemset
{B, C, E}

3rd scan

L_3

Itemset	sup
{B, C, E}	2

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 !=$\emptyset$; 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 $\cup_k L_k$;

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*}

Mining Association Rules—an Example

Transaction-id	Items bought
10	A, B, C
20	A, C
30	A, D
40	B, E, F

Min. support 50%
Min. confidence 50%

Frequent pattern	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%

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

Example

Transactional Data for an *AllElectronics* Branch

TID	*List of item_IDs*
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

Scan D for count of each candidate

C_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Compare candidate support count with minimum support count

L_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Generate C_2 candidates from L_1

C_2

Itemset
{I1, I2}
{I1, I3}
{I1, I4}
{I1, I5}
{I2, I3}
{I2, I4}
{I2, I5}
{I3, I4}
{I3, I5}
{I4, I5}

Scan D for count of each candidate

C_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I4}	1
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2
{I3, I4}	0
{I3, I5}	1
{I4, I5}	0

Compare candidate support count with minimum support count

L_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2

Generate C_3 candidates from L_2

C_3

Itemset
{I1, I2, I3}
{I1, I2, I5}

Scan D for count of each candidate

C_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

Compare candidate support count with minimum support count

L_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

Figure 6.2 Generation of the candidate itemsets and frequent itemsets, where the minimum support count is 2.

Generating association rules from frequent itemset

$$
\begin{aligned}
\{I1, I2\} &\Rightarrow I5, & \textit{confidence} &= 2/4 = 50\% \\
\{I1, I5\} &\Rightarrow I2, & \textit{confidence} &= 2/2 = 100\% \\
\{I2, I5\} &\Rightarrow I1, & \textit{confidence} &= 2/2 = 100\% \\
I1 &\Rightarrow \{I2, I5\}, & \textit{confidence} &= 2/6 = 33\% \\
I2 &\Rightarrow \{I1, I5\}, & \textit{confidence} &= 2/7 = 29\% \\
I5 &\Rightarrow \{I1, I2\}, & \textit{confidence} &= 2/2 = 100\%
\end{aligned}
$$

If the minimum strong. confidence threshold is, say, **70%,** then only the second, third, and last rules are output, because these are the only ones generated that are

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

- Association rules from frequent itemset

-multilevel Association rules

-multidimensional Association rules

Multilevel association rules

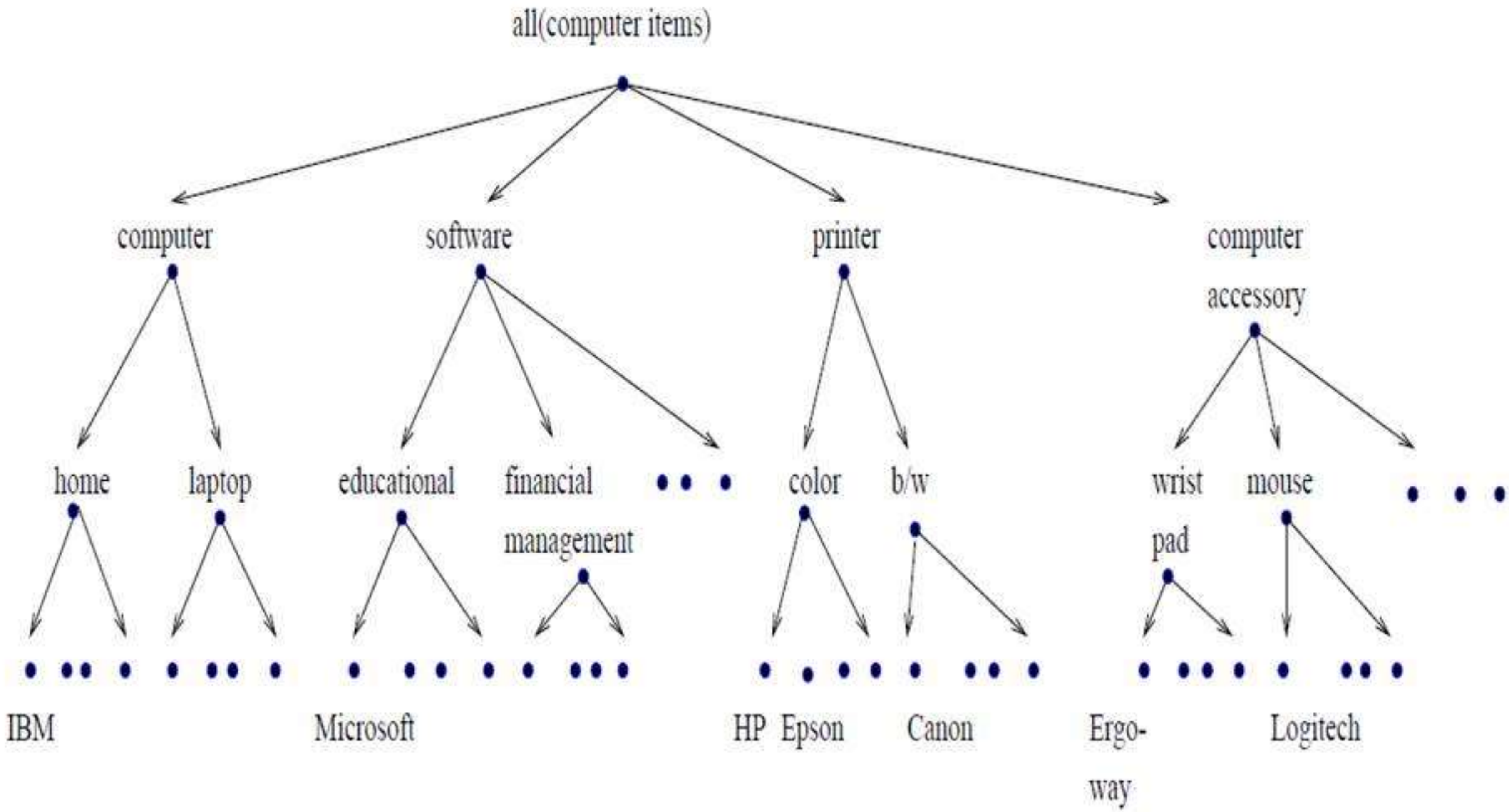


Figure 6.8: A concept hierarchy for *AllElectronics* computer items.

Approaches to mining multilevel association rules

- Using uniform minimum support for all levels (uniform support):

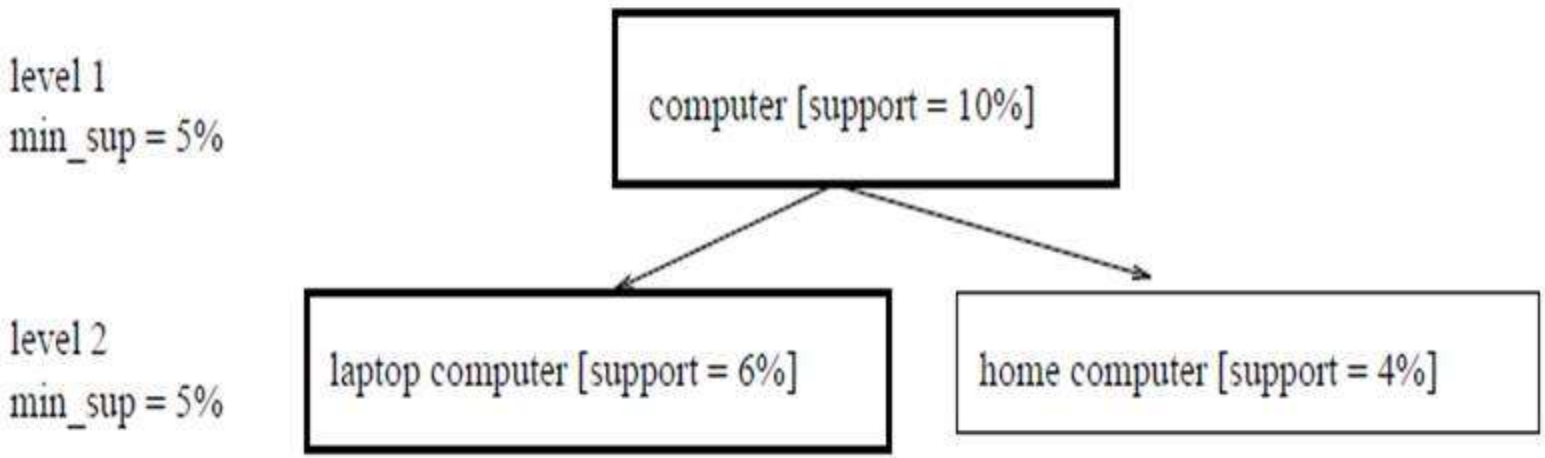


Figure 6.9: Multilevel mining with uniform support.

Using reduced minimum support at lower levels (reduced support)

- level-by-level independent

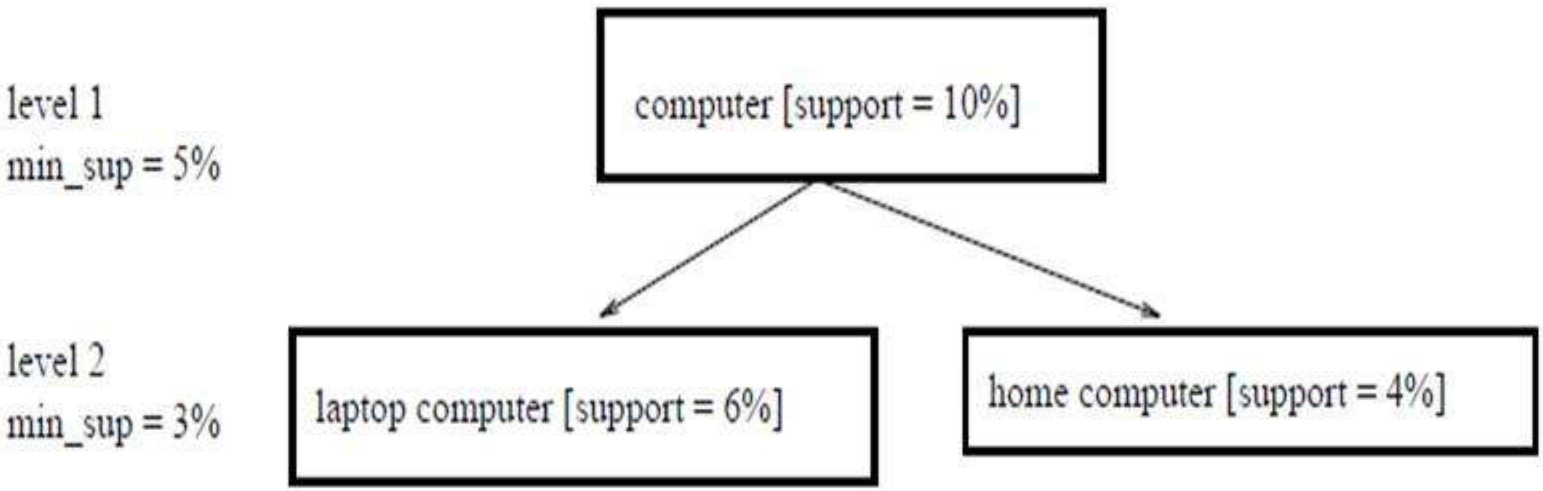


Figure 6.10: Multilevel mining with reduced support.

Using reduced minimum support at lower levels (reduced support):

level-cross filtering by single item

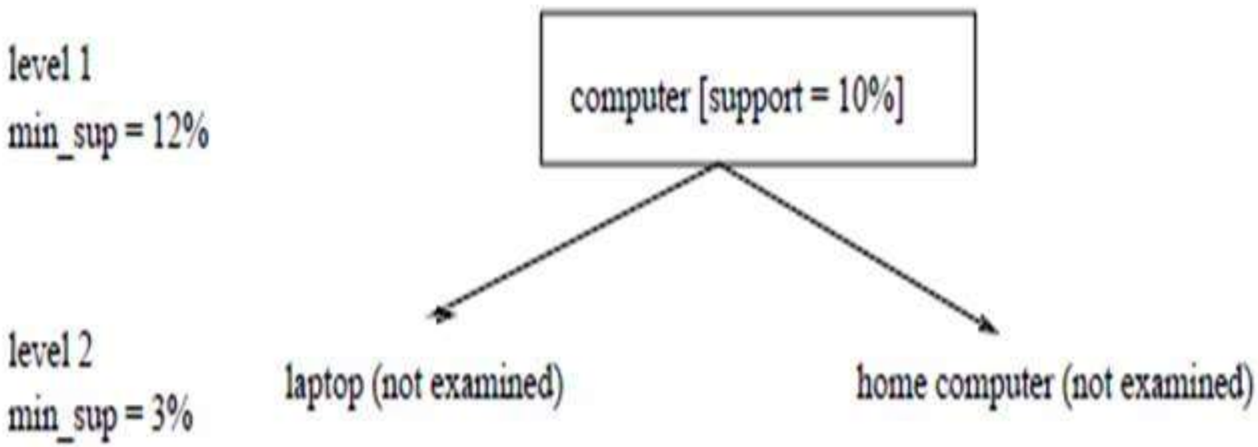


Figure 6.11: Multilevel mining with reduced support, using level-cross filtering by a single item.

Using reduced minimum support at lower levels (reduced support)

level-cross filtering by k-itemset

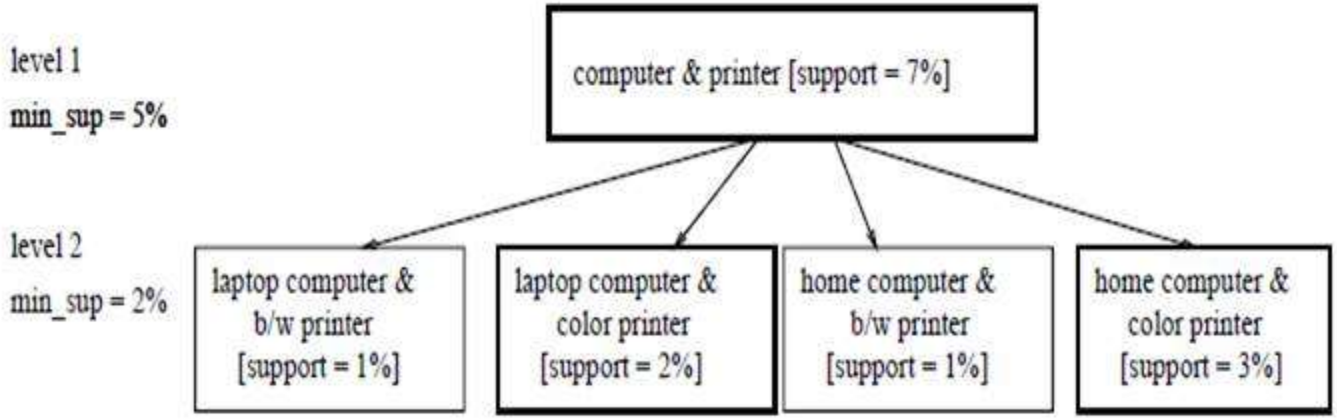


Figure 6.12: Multilevel mining with reduced support, using level-cross filtering by a k-itemset. Here, $k = 2$.

Multi-dimensional Association

- Single-dimensional rules:

  buys(X, “milk”) $\Rightarrow$ buys(X, “bread”)

- Multi-dimensional rules: $\geq$ 2 dimensions or predicates
  - Inter-dimension assoc. rules (*no repeated predicates*)

    age(X,”19-25”) $\wedge$ occupation(X,“student”) $\Rightarrow$ buys(X,“coke”)
  - hybrid-dimension assoc. rules (*repeated predicates*)

    age(X,”19-25”) $\wedge$ buys(X, “popcorn”) $\Rightarrow$ buys(X, “coke”)
- Categorical Attributes
  - finite number of possible values, no ordering among values
- Quantitative Attributes
  - numeric, implicit ordering among values

Constraint-based Data Mining

- Finding all the patterns in a database autonomously? — unrealistic!
  - The patterns could be too many but not focused!
- Data mining should be an interactive process
  - User directs what to be mined using a data mining query language (or a graphical user interface)
- Constraint-based mining
  - User flexibility: provides constraints on what to be mined
  - System optimization: explores such constraints for efficient mining—constraint-based mining

Constraints in Data Mining

- Knowledge type constraint:
  – classification, association, etc.
- Data constraint — using SQL-like queries
  – find product pairs sold together in stores in Vancouver in Dec.'00
- Dimension/level constraint
  – in relevance to region, price, brand, customer category
- Rule (or pattern) constraint
  – small sales (price < \$10) triggers big sales (sum > \$200)
- Interestingness constraint
  – strong rules: min_support $\geq$ 3%, min_confidence $\geq$ 60%