Data Mining

Chapter 5
Association Analysis: Basic Concepts

Introduction to Data Mining, 2nd Edition
by
Tan, Steinbach, Karpatne, Kumar

Association Rule Mining

- Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction

Market-Basket transactions

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Association Rules

{Diaper} → {Beer},
{Milk, Bread} → {Eggs,Coke},
{Beer, Bread} → {Milk},

Implication means co-occurrence, not causality!

Definition: Frequent Itemset

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
- **Support count (σ)**
  - Frequency of occurrence of an itemset
  - E.g. σ({Milk, Bread,Diaper}) = 2
- **Support**
  - Fraction of transactions that contain an itemset
  - E.g. s({Milk, Bread, Diaper}) = 2/5
- **Frequent Itemset**
  - An itemset whose support is greater than or equal to a *minsup* threshold

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Definition: Association Rule

- **Association Rule**
  - An implication expression of the form X → Y, where X and Y are itemsets
  - Example:
    {Milk, Diaper} → {Beer}
- **Rule Evaluation Metrics**
  - Support (s)
    - Fraction of transactions that contain both X and Y
  - Confidence (c)
    - Measures how often items in Y appear in transactions that contain X

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example:

$$\{\text{Milk}, \text{Diaper}\} \Rightarrow \{\text{Beer}\}$$

$$s = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{|\text{T}|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{\sigma(\text{Milk}, \text{Diaper})} = \frac{2}{3} = 0.67$$

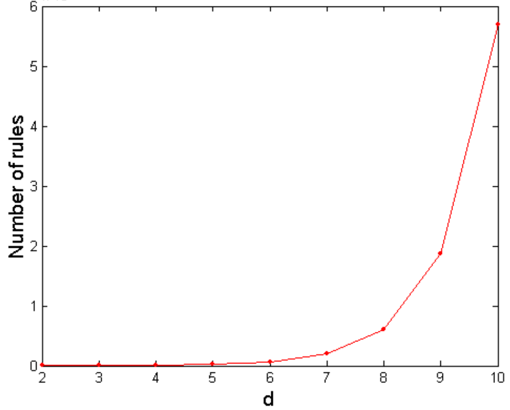
Association Rule Mining Task

- Given a set of transactions T, the goal of association rule mining is to find all rules having
  - support ≥ *minsup* threshold
  - confidence ≥ *minconf* threshold

- Brute-force approach:
  - List all possible association rules
  - Compute the support and confidence for each rule
  - Prune rules that fail the *minsup* and *minconf* thresholds
  - ⇒ Computationally prohibitive!

Computational Complexity

- Given d unique items:
  - Total number of itemsets = 2^d
  - Total number of possible association rules:



$$R = \sum_{k=1}^{d-1}\left[\binom{d}{k}\times\sum_{j=1}^{d-k}\binom{d-k}{j}\right]$$
$$= 3^d - 2^{d+1} + 1$$

If d=6, R = 602 rules

Mining Association Rules

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Rules:

{Milk,Diaper} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

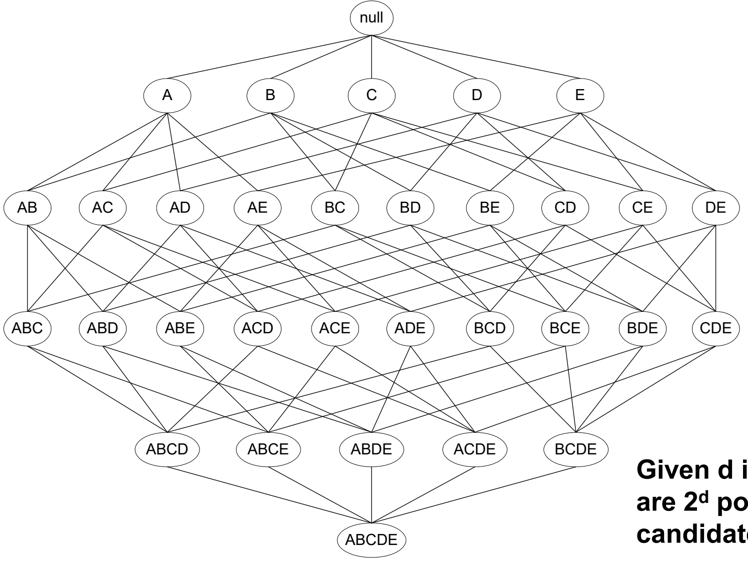
Observations:

- All the above rules are binary partitions of the same itemset:
  {Milk, Diaper, Beer}
- Rules originating from the same itemset have identical support but can have different confidence
- Thus, we may decouple the support and confidence requirements

Mining Association Rules

- Two-step approach:
  1. Frequent Itemset Generation
     - Generate all itemsets whose support ≥ minsup
  2. Rule Generation
     - Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset
- Frequent itemset generation is still computationally expensive

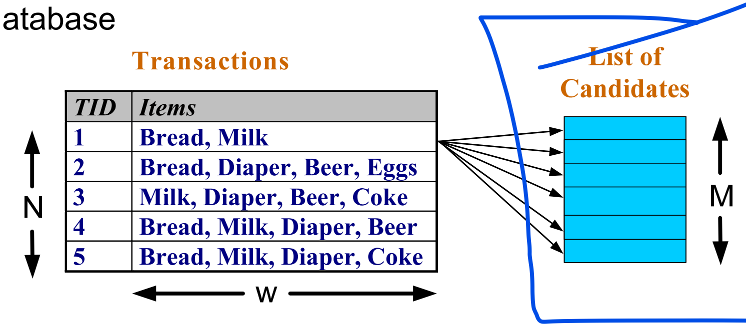
Frequent Itemset Generation



Given d items, there are 2^d possible candidate itemsets

Frequent Itemset Generation

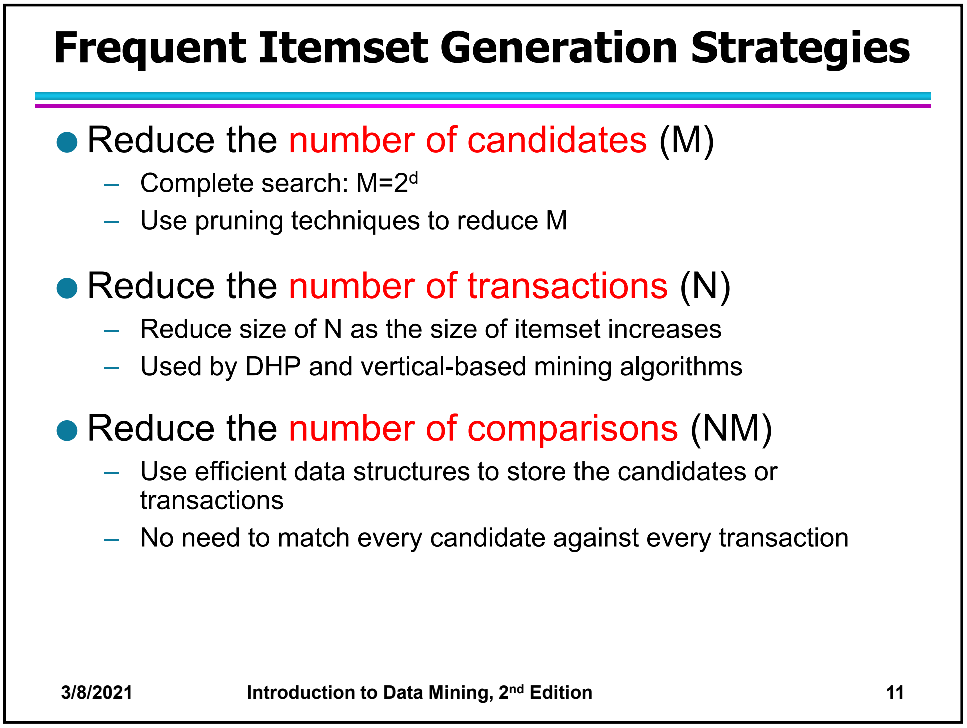
- Brute-force approach:
  - Each itemset in the lattice is a candidate frequent itemset
  - Count the support of each candidate by scanning the database



Transactions

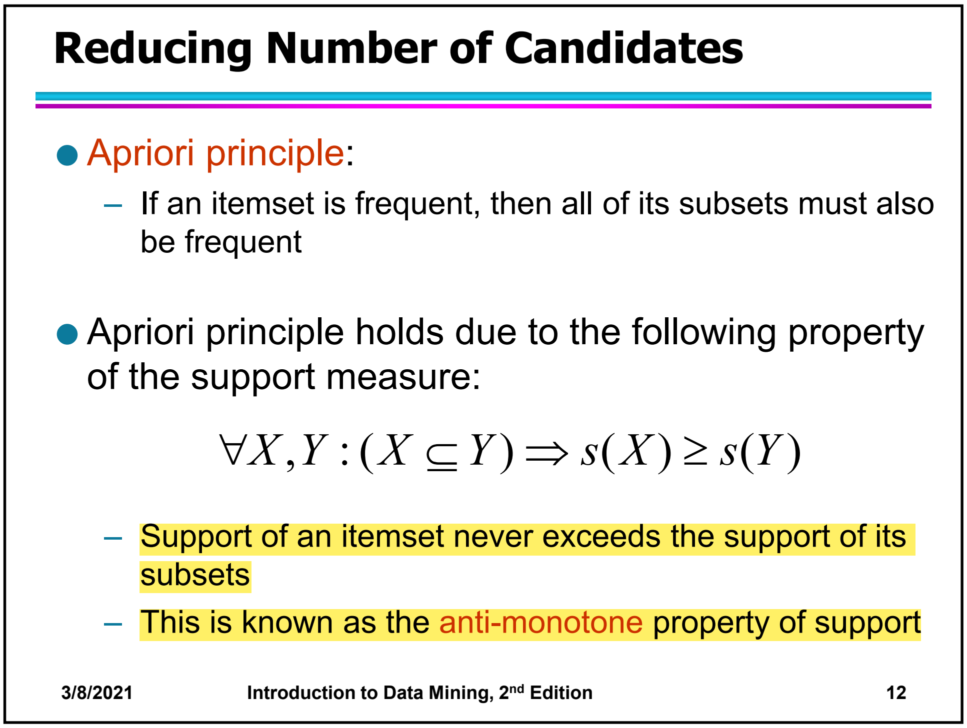
TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

  - Match each transaction against every candidate
  - Complexity ~ O(NMw) => Expensive since $M = 2^d$!!!



Frequent Itemset Generation Strategies

- Reduce the number of candidates (M)
  - Complete search: $M=2^d$
  - Use pruning techniques to reduce M
- Reduce the number of transactions (N)
  - Reduce size of N as the size of itemset increases
  - Used by DHP and vertical-based mining algorithms
- Reduce the number of comparisons (NM)
  - Use efficient data structures to store the candidates or transactions
  - No need to match every candidate against every transaction



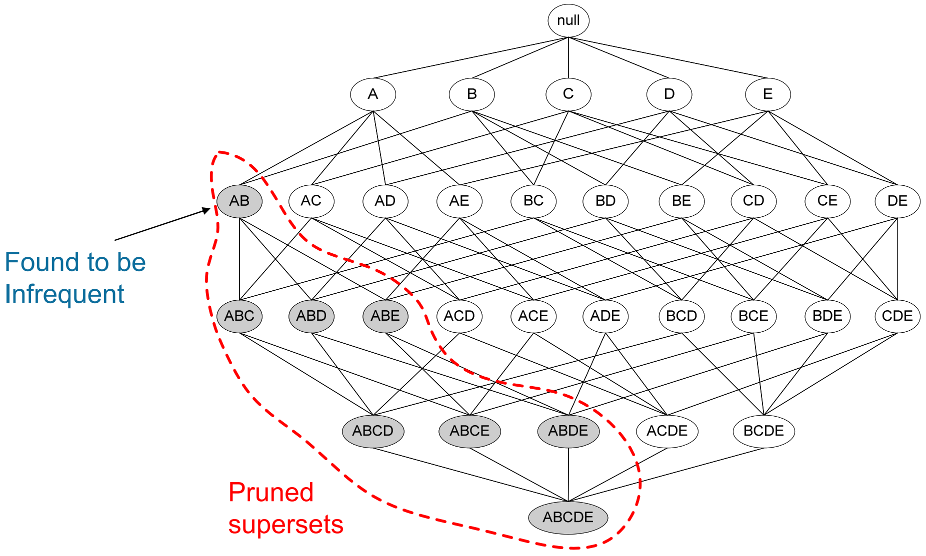
Reducing Number of Candidates

- Apriori principle:
  - If an itemset is frequent, then all of its subsets must also be frequent
- Apriori principle holds due to the following property of the support measure:

$$\forall X,Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

  - Support of an itemset never exceeds the support of its subsets
  - This is known as the anti-monotone property of support

Illustrating Apriori Principle



Found to be Infrequent

Pruned supersets

Illustrating Apriori Principle

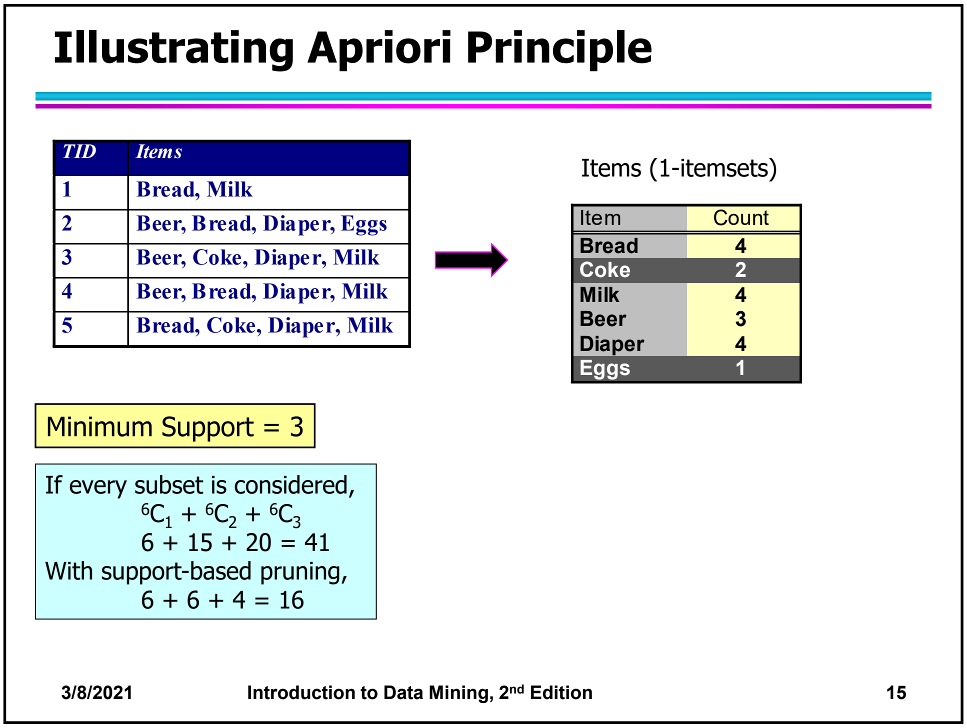
TID	*Items*
1	**Bread, Milk**
2	**Beer, Bread, Diaper, Eggs**
3	**Beer, Coke, Diaper, Milk**
4	**Beer, Bread, Diaper, Milk**
5	**Bread, Coke, Diaper, Milk**

Items (1-itemsets)

Item	Count
Bread	**4**
Coke	**2**
Milk	**4**
Beer	**3**
Diaper	**4**
Eggs	**1**

Minimum Support = 3

If every subset is considered,
${}^6C_1 + {}^6C_2 + {}^6C_3$
6 + 15 + 20 = 41
With support-based pruning,
6 + 6 + 4 = 16



Illustrating Apriori Principle

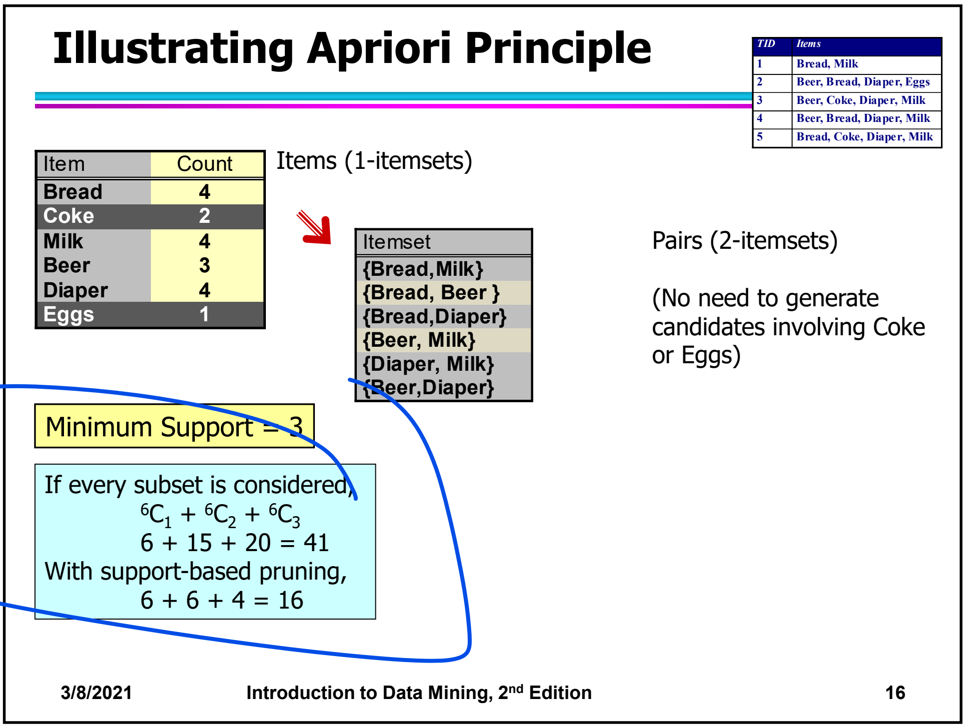
TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Items (1-itemsets)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Minimum Support = 3

If every subset is considered,
$^6C_1 + {}^6C_2 + {}^6C_3$
6 + 15 + 20 = 41
With support-based pruning,
6 + 6 + 4 = 16



Illustrating Apriori Principle

TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

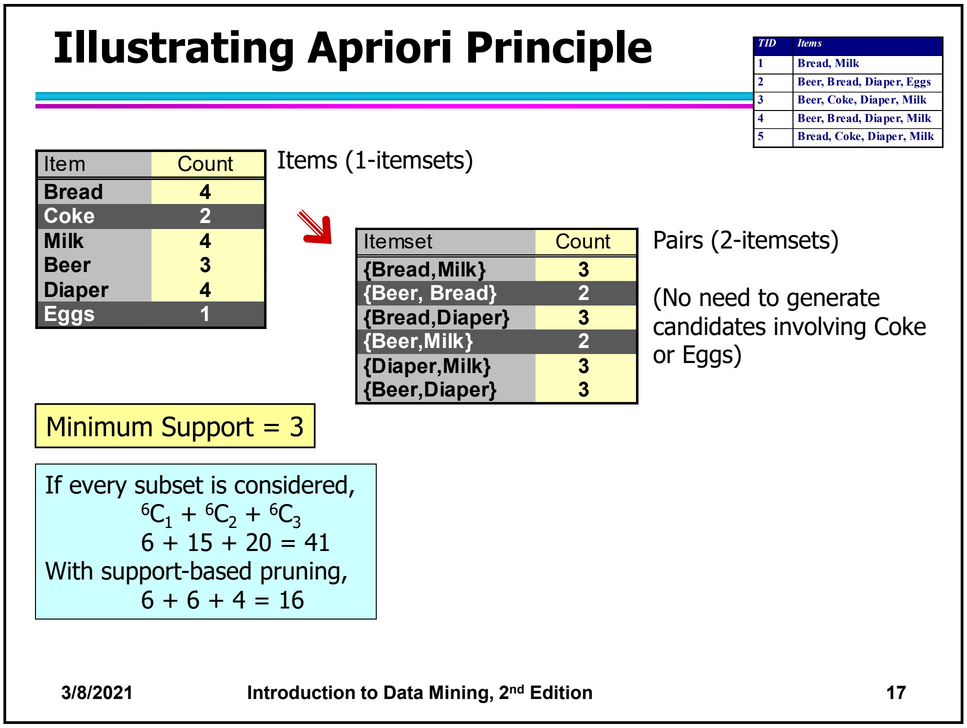
Itemset
{Bread,Milk}
{Bread, Beer }
{Bread,Diaper}
{Beer, Milk}
{Diaper, Milk}
{Beer,Diaper}

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Minimum Support = 3

If every subset is considered,
$^6C_1 + {}^6C_2 + {}^6C_3$
6 + 15 + 20 = 41
With support-based pruning,
6 + 6 + 4 = 16



Illustrating Apriori Principle

TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Items (1-itemsets)

Item	Count
Bread	**4**
Coke	**2**
Milk	**4**
Beer	**3**
Diaper	**4**
Eggs	**1**

Pairs (2-itemsets)

Itemset	Count
{Bread,Milk}	**3**
{Beer, Bread}	**2**
{Bread,Diaper}	**3**
{Beer,Milk}	**2**
{Diaper,Milk}	**3**
{Beer,Diaper}	**3**

(No need to generate candidates involving Coke or Eggs)

Minimum Support = 3

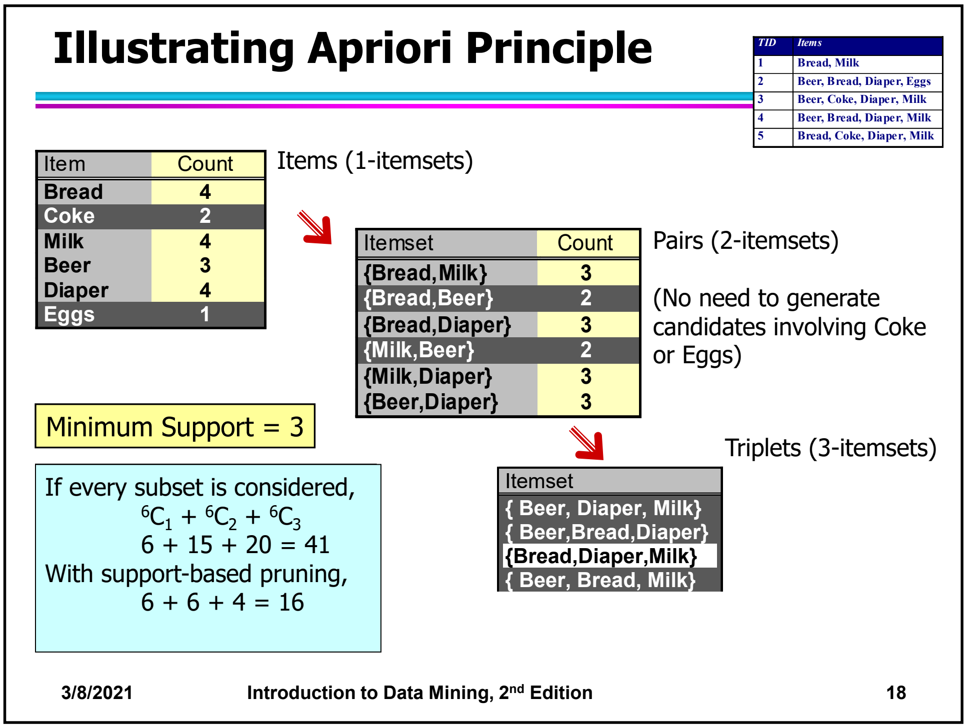
If every subset is considered,

$^6C_1 + {}^6C_2 + {}^6C_3$

6 + 15 + 20 = 41

With support-based pruning,

6 + 6 + 4 = 16



Illustrating Apriori Principle

TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Items (1-itemsets)

Item	Count
Bread	**4**
Coke	**2**
Milk	**4**
Beer	**3**
Diaper	**4**
Eggs	**1**

Pairs (2-itemsets)

Itemset	Count
{Bread,Milk}	**3**
{Bread,Beer}	**2**
{Bread,Diaper}	**3**
{Milk,Beer}	**2**
{Milk,Diaper}	**3**
{Beer,Diaper}	**3**

(No need to generate candidates involving Coke or Eggs)

Triplets (3-itemsets)

Itemset
{ Beer, Diaper, Milk}
{ Beer,Bread,Diaper}
{Bread,Diaper,Milk}
{ Beer, Bread, Milk}

Minimum Support = 3

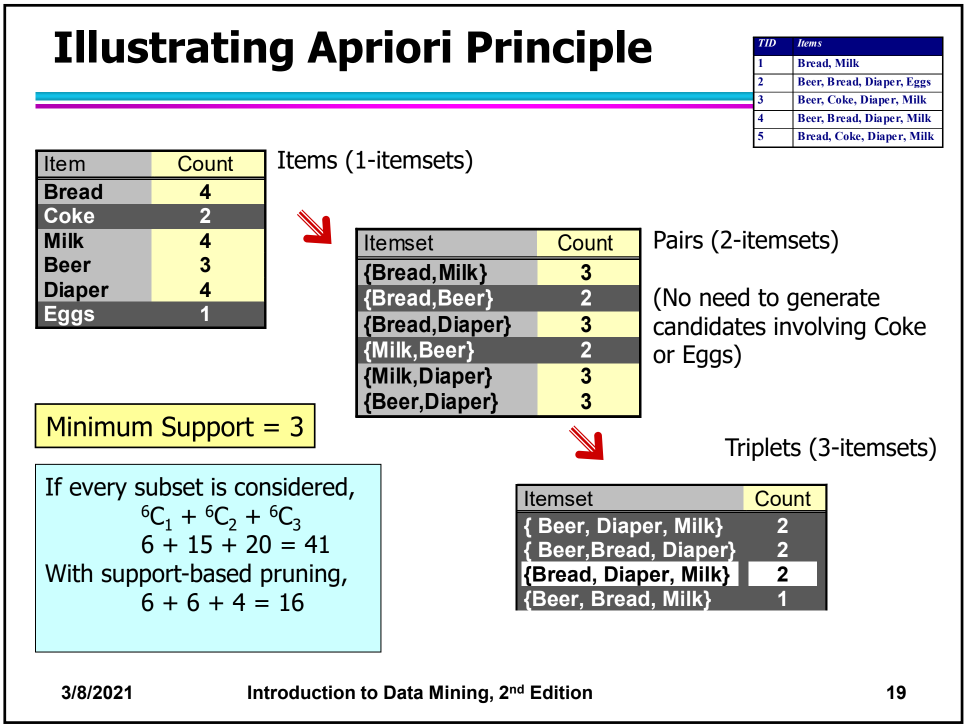
If every subset is considered,

$^6C_1 + {}^6C_2 + {}^6C_3$

6 + 15 + 20 = 41

With support-based pruning,

6 + 6 + 4 = 16



Illustrating Apriori Principle

TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Items (1-itemsets)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

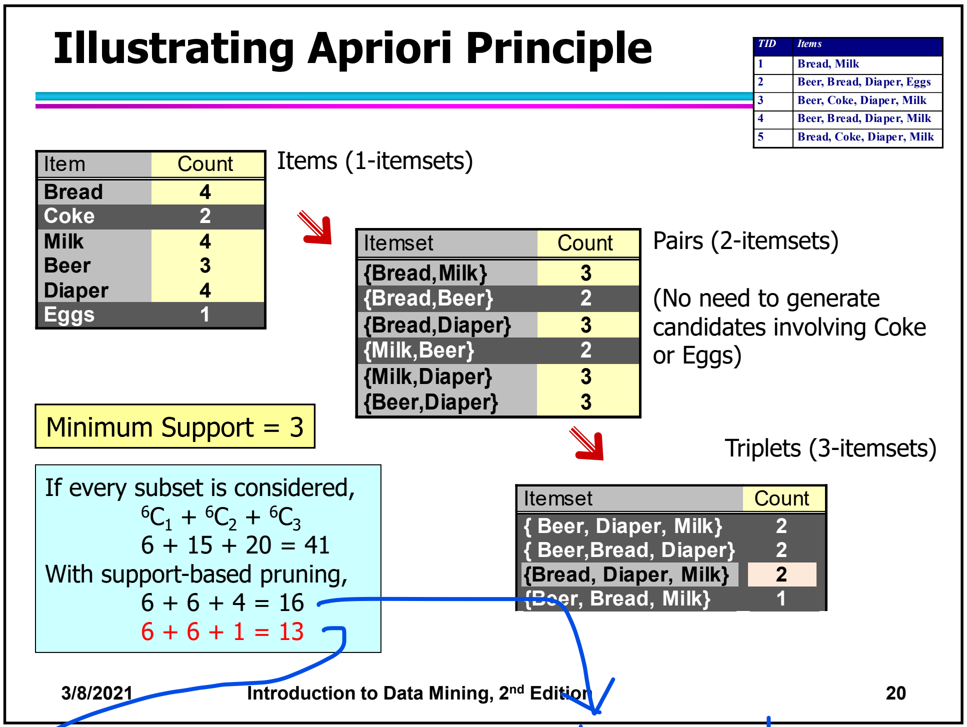
Itemset	Count
{Bread,Milk}	3
{Bread,Beer}	2
{Bread,Diaper}	3
{Milk,Beer}	2
{Milk,Diaper}	3
{Beer,Diaper}	3

Minimum Support = 3

Triplets (3-itemsets)

Itemset	Count
{ Beer, Diaper, Milk}	2
{ Beer,Bread, Diaper}	2
{Bread, Diaper, Milk}	2
{Beer, Bread, Milk}	1

If every subset is considered,
$^6C_1 + {}^6C_2 + {}^6C_3$
6 + 15 + 20 = 41
With support-based pruning,
6 + 6 + 4 = 16



Illustrating Apriori Principle

TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Items (1-itemsets)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Itemset	Count
{Bread,Milk}	3
{Bread,Beer}	2
{Bread,Diaper}	3
{Milk,Beer}	2
{Milk,Diaper}	3
{Beer,Diaper}	3

Minimum Support = 3

Triplets (3-itemsets)

Itemset	Count
{ Beer, Diaper, Milk}	2
{ Beer,Bread, Diaper}	2
{Bread, Diaper, Milk}	2
{Beer, Bread, Milk}	1

If every subset is considered,
$^6C_1 + {}^6C_2 + {}^6C_3$
6 + 15 + 20 = 41
With support-based pruning,
6 + 6 + 4 = 16
6 + 6 + 1 = 13

Prune ke pehle w/o duplicates

Itno ka support nikalna pada
Jo candidates prev stage me

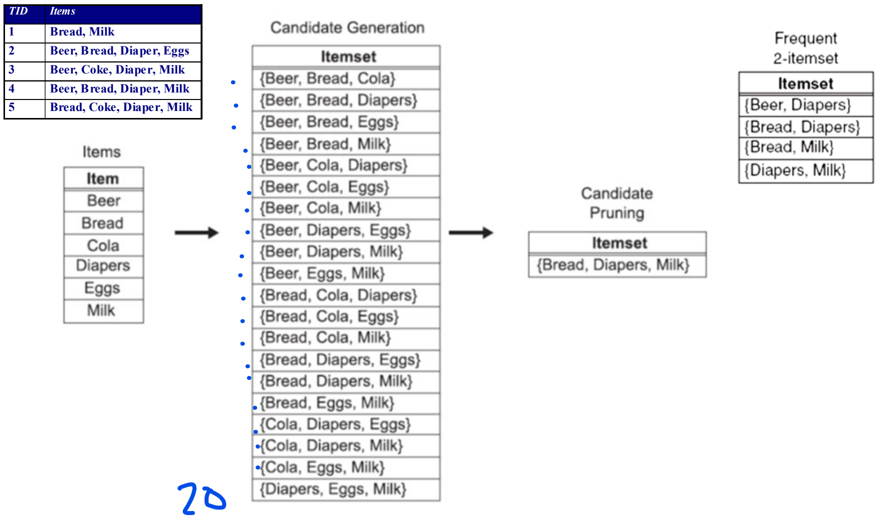
prune hue unko pehle hi prune karo

Apriori Algorithm

- F_k: frequent k-itemsets
- L_k: candidate k-itemsets

- Algorithm
  - Let k=1
  - Generate F_1 = {frequent 1-itemsets}
  - Repeat until F_k is empty
    - **Candidate Generation**: Generate L_{k+1} from F_k
    - **Candidate Pruning**: Prune candidate itemsets in L_{k+1} containing subsets of length k that are infrequent
    - **Support Counting**: Count the support of each candidate in L_{k+1} by scanning the DB
    - **Candidate Elimination**: Eliminate candidates in L_{k+1} that are infrequent, leaving only those that are frequent => F_{k+1}

Candidate Generation: Brute-force method



TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Items

Item
Beer
Bread
Cola
Diapers
Eggs
Milk

Candidate Generation

Itemset
{Beer, Bread, Cola}
{Beer, Bread, Diapers}
{Beer, Bread, Eggs}
{Beer, Bread, Milk}
{Beer, Cola, Diapers}
{Beer, Cola, Eggs}
{Beer, Cola, Milk}
{Beer, Diapers, Eggs}
{Beer, Diapers, Milk}
{Beer, Eggs, Milk}
{Bread, Cola, Diapers}
{Bread, Cola, Eggs}
{Bread, Cola, Milk}
{Bread, Diapers, Eggs}
{Bread, Diapers, Milk}
{Bread, Eggs, Milk}
{Cola, Diapers, Eggs}
{Cola, Diapers, Milk}
{Cola, Eggs, Milk}
{Diapers, Eggs, Milk}

20

Candidate Pruning

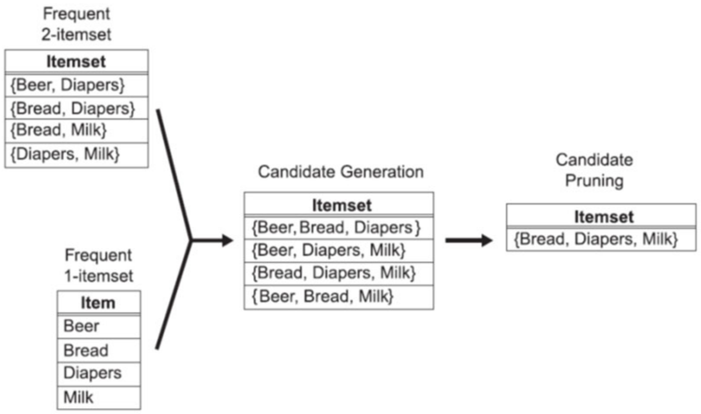
Itemset
{Bread, Diapers, Milk}

Frequent 2-itemset

Itemset
{Beer, Diapers}
{Bread, Diapers}
{Bread, Milk}
{Diapers, Milk}

Figure 5.6. A brute-force method for generating candidate 3-itemsets.

Candidate Generation: Merge Fk-1 and F1 itemsets



Frequent 2-itemset

Itemset
{Beer, Diapers}
{Bread, Diapers}
{Bread, Milk}
{Diapers, Milk}

Frequent 1-itemset

Item
Beer
Bread
Diapers
Milk

Candidate Generation

Itemset
{Beer, Bread, Diapers}
{Beer, Diapers, Milk}
{Bread, Diapers, Milk}
{Beer, Bread, Milk}

Candidate Pruning

Itemset
{Bread, Diapers, Milk}

Figure 5.7. Generating and pruning candidate k-itemsets by merging a frequent $(k-1)$-itemset with a frequent item. Note that some of the candidates are unnecessary because their subsets are infrequent.

Candidate Generation: F_{k-1} x F_{k-1} Method

- Merge two frequent (k-1)-itemsets if their first (k-2) items are identical

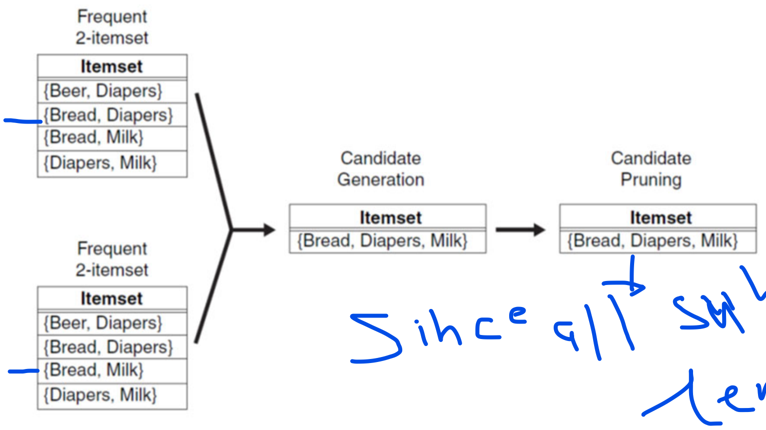
- F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE}
  - Merge(**AB**C, **AB**D) = **AB**CD
  - Merge(**AB**C, **AB**E) = **AB**CE
  - Merge(**AB**D, **AB**E) = **AB**DE

  - Do not merge(**A**BD,**A**CD) because they share only prefix of length 1 instead of length 2

Candidate Pruning

- Let F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE} be the set of frequent 3-itemsets
- L_4 = {ABCD,ABCE,ABDE} is the set of candidate 4-itemsets generated (from previous slide)
- Candidate pruning
  - Prune ABCE because ACE and BCE are infrequent
  - Prune ABDE because ADE is infrequent
- After candidate pruning: L_4 = {ABCD}

Candidate Generation: Fk-1 x Fk-1 Method



Frequent 2-itemset

Itemset
{Beer, Diapers}
{Bread, Diapers}
{Bread, Milk}
{Diapers, Milk}

Frequent 2-itemset

Itemset
{Beer, Diapers}
{Bread, Diapers}
{Bread, Milk}
{Diapers, Milk}

Candidate Generation

Itemset
{Bread, Diapers, Milk}

Candidate Pruning

Itemset
{Bread, Diapers, Milk}

Since all subsets of
len=2 (k-1)
are
freq.

Figure 5.8. Generating and pruning candidate k-itemsets by merging pairs of frequent $(k-1)$-itemsets.

using $F_{k-1} \times F_{k-1}$

Illustrating Apriori Principle

Items (1-itemsets)

Item	Count
Bread	**4**
Coke	**2**
Milk	**4**
Beer	**3**
Diaper	**4**
Eggs	**1**

Minimum Support = 3

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Itemset	Count
{Bread,Milk}	**3**
{Bread,Beer}	**2**
{Bread,Diaper}	**3**
{Milk,Beer}	**2**
{Milk,Diaper}	**3**
{Beer,Diaper}	**3**

Triplets (3-itemsets)

Itemset	Count
{Bread, Diaper, Milk}	**2**

If every subset is considered,
$^6C_1 + {}^6C_2 + {}^6C_3$
6 + 15 + 20 = 41
With support-based pruning,
6 + 6 + 1 = 13

Use of F_{k-1}xF_{k-1} method for candidate generation results in only one 3-itemset. This is eliminated after the support counting step.

Alternate F_{k-1} x F_{k-1} Method

- Merge two frequent (k-1)-itemsets if the last (k-2) items of the first one is identical to the first (k-2) items of the second.

- F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE}
  - Merge(A**BC**, **BC**D) = A**BC**D
  - Merge(A**BD**, **BD**E) = A**BD**E
  - Merge(A**CD**, **CD**E) = A**CD**E
  - Merge(B**CD**, **CD**E) = B**CD**E

Candidate Pruning for Alternate $F_{k-1} \times F_{k-1}$ Method

- Let F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE} be the set of frequent 3-itemsets

D

- L_4 = {ABCD,ABDE,ACDE,BCDE} is the set of candidate 4-itemsets generated (from previous slide)
- Candidate pruning
  - Prune ABDE because ADE is infrequent
  - Prune ACDE because ACE and ADE are infrequent
  - Prune BCDE because BCE
- After candidate pruning: L_4 = {ABCD}

Support Counting of Candidate Itemsets

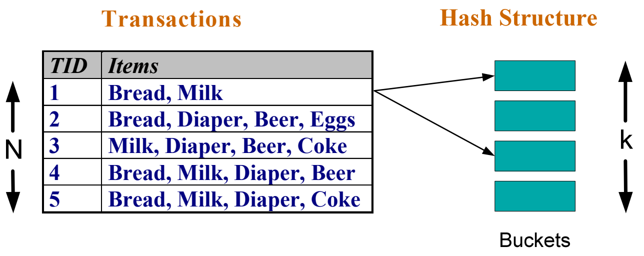
- Scan the database of transactions to determine the support of each candidate itemset
  - Must match every candidate itemset against every transaction, which is an expensive operation

TID	*Items*
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Itemset
{ Beer, Diaper, Milk}
{ Beer,Bread,Diaper}
{Bread, Diaper, Milk}
{ Beer, Bread, Milk}

Support Counting of Candidate Itemsets

- To reduce number of comparisons, store the candidate itemsets in a hash structure
  - Instead of matching each transaction against every candidate, match it against candidates contained in the hashed buckets



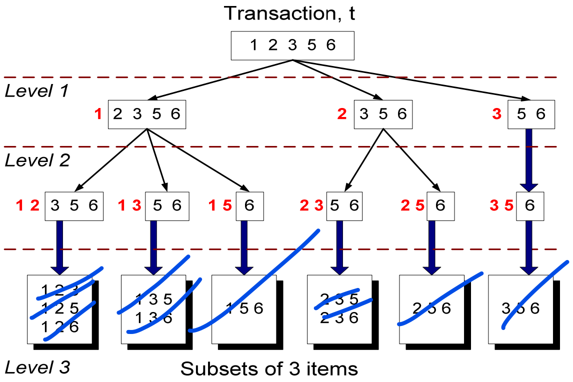
TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Support Counting: An Example

Suppose you have 15 candidate itemsets of length 3:

{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

How many of these itemsets are supported by transaction (1,2,3,5,6)?



DL

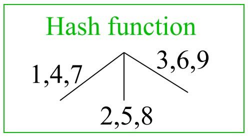
Support Counting Using a Hash Tree

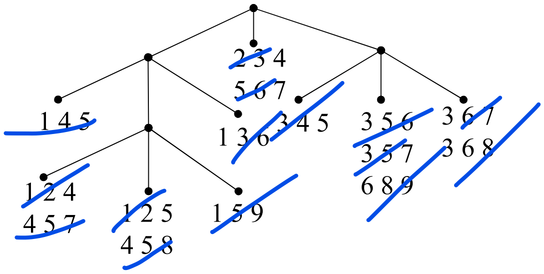
Suppose you have 15 candidate itemsets of length 3:

{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

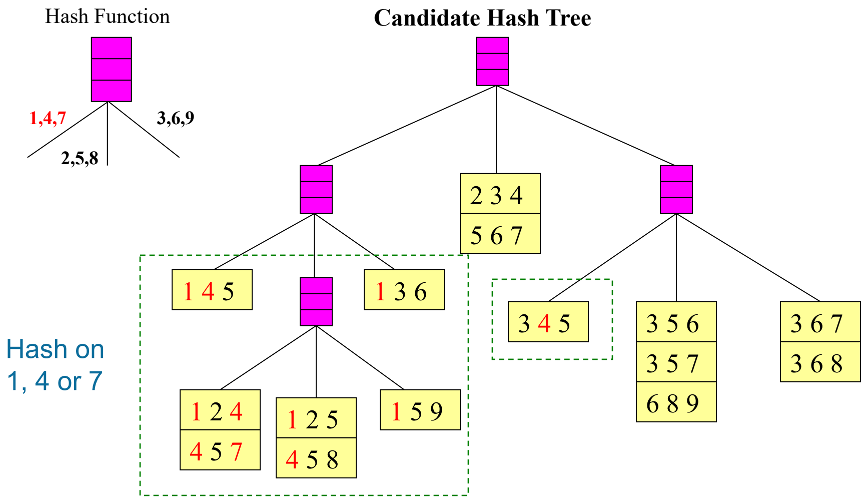
You need:

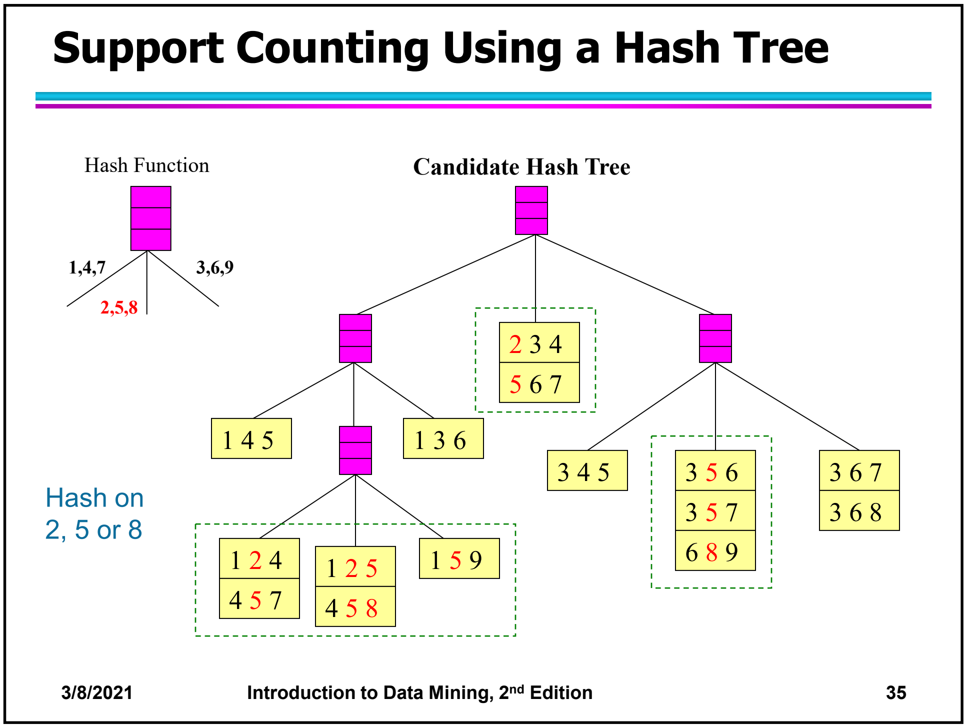
- **Hash function**
- **Max leaf size: max number of itemsets stored in a leaf node (if number of candidate itemsets exceeds max leaf size, split the node)**





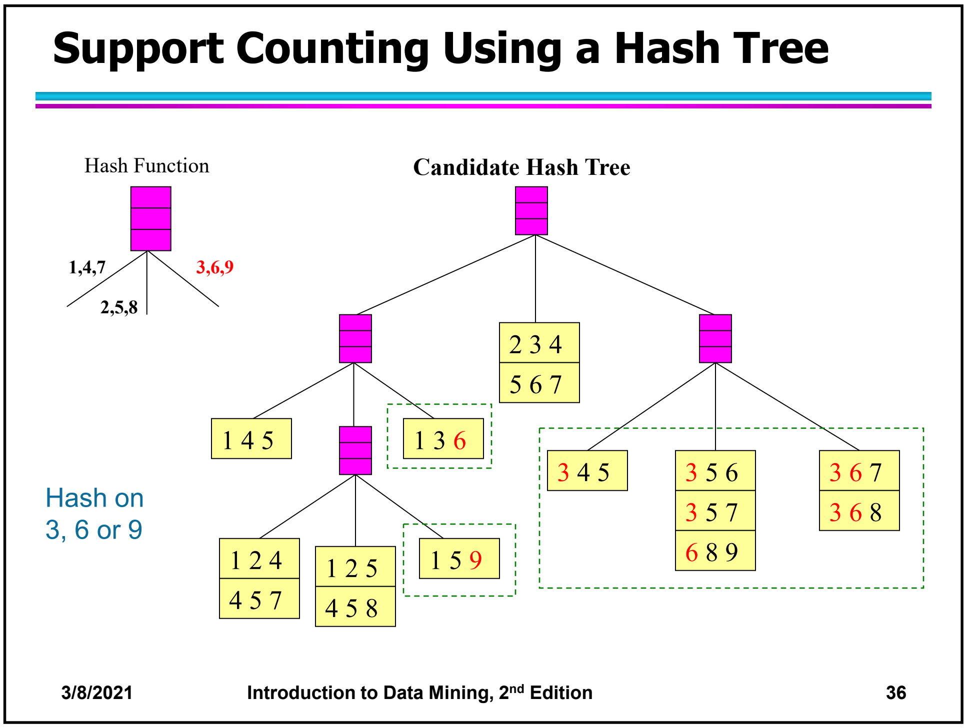
Support Counting Using a Hash Tree





Support Counting Using a Hash Tree

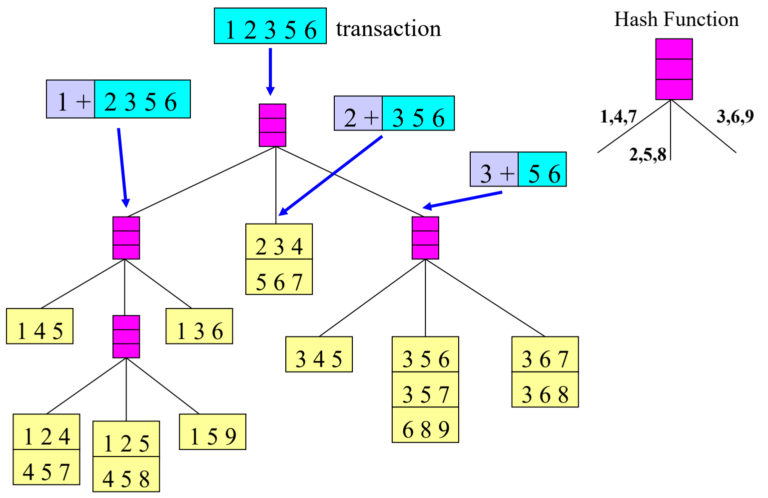
Hash on
2, 5 or 8



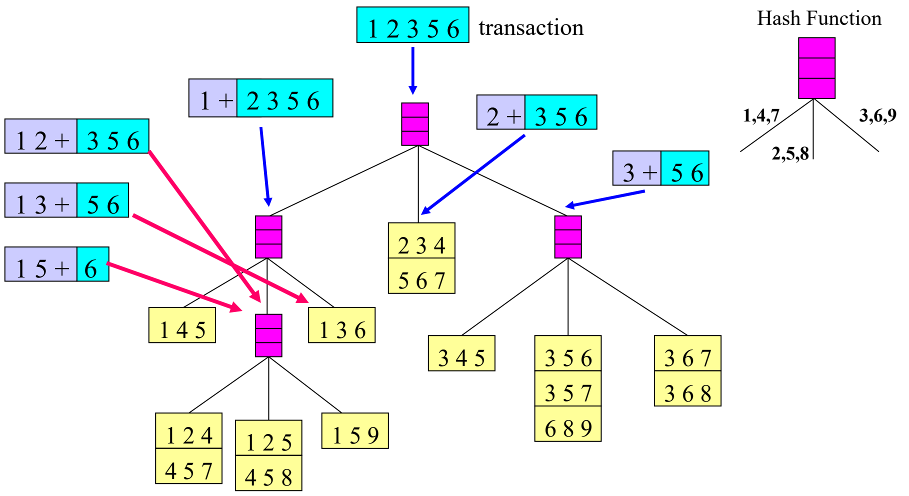
Support Counting Using a Hash Tree

Hash on
3, 6 or 9

Support Counting Using a Hash Tree

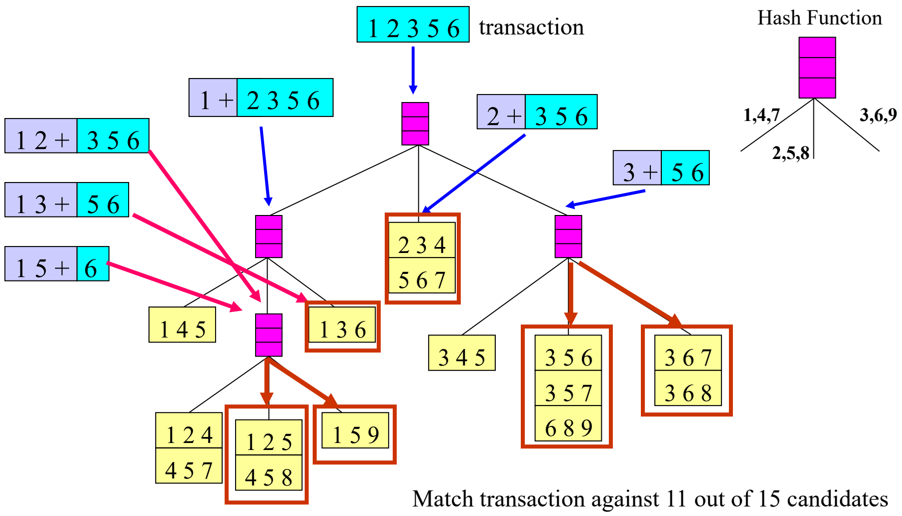


Support Counting Using a Hash Tree



DL

Support Counting Using a Hash Tree



Match transaction against 11 out of 15 candidates

Rule Generation

- Given a frequent itemset L, find all non-empty subsets f ⊂ L such that f → L – f satisfies the minimum confidence requirement
  - If {A,B,C,D} is a frequent itemset, candidate rules:

ABC →D,	ABD →C,	ACD →B,	BCD →A,
A →BCD,	B →ACD,	C →ABD,	D →ABC
AB →CD,	AC → BD,	AD → BC,	BC →AD,
BD →AC,	CD →AB,		

- If |L| = k, then there are $2^k – 2$ candidate association rules (ignoring L → ∅ and ∅ → L)

Rule Generation

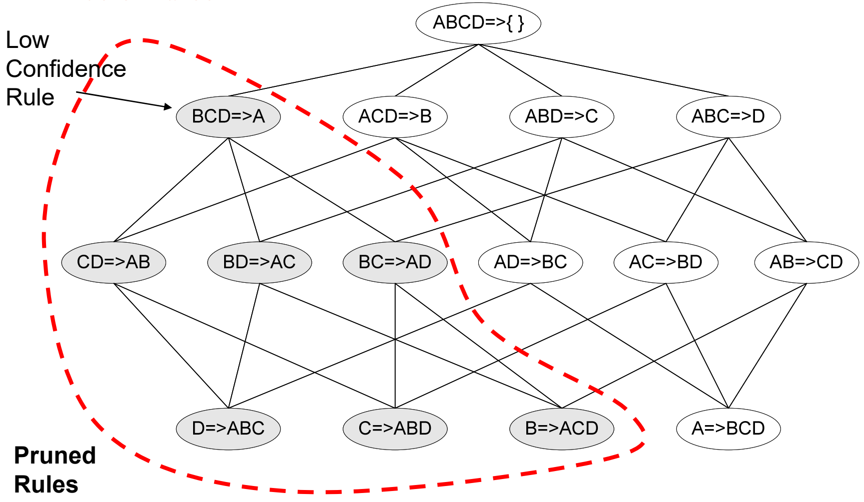
- In general, confidence does not have an anti-monotone property
  - $c(ABC \rightarrow D)$ can be larger or smaller than $c(AB \rightarrow D)$
- But confidence of rules generated from the same itemset has an anti-monotone property
  - E.g., Suppose {A,B,C,D} is a frequent 4-itemset:

$$c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$$

  - Confidence is anti-monotone w.r.t. number of items on the RHS of the rule

Rule Generation for Apriori Algorithm

Lattice of rules



Association Analysis: Basic Concepts and Algorithms

Algorithms and Complexity

Factors Affecting Complexity of Apriori

- Choice of minimum support threshold
- Dimensionality (number of items) of the data set
- Size of database
- Average transaction width

Factors Affecting Complexity of Apriori

- Choice of minimum support threshold
  - lowering support threshold results in more frequent itemsets
  - this may increase number of candidates and max length of frequent itemsets
- Dimensionality (number of items) of the data set
  - –
- Size of database
  - –
- Average transaction width
  - –

TID	*Items*
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Impact of Support Based Pruning

TID	*Items*
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Items (1-itemsets)

Item	Count
Bread	**4**
Coke	**2**
Milk	**4**
Beer	**3**
Diaper	**4**
Eggs	**1**

Minimum Support = 3

If every subset is considered,
$^6C_1 + {}^6C_2 + {}^6C_3$
6 + 15 + 20 = 41
With support-based pruning,
6 + 6 + 4 = 16

Minimum Support = 2

If every subset is considered,
$^6C_1 + {}^6C_2 + {}^6C_3 + {}^6C_4$
6 + 15 + 20 +15 = 56

Factors Affecting Complexity of Apriori

- Choice of minimum support threshold
  - lowering support threshold results in more frequent itemsets
  - this may increase number of candidates and max length of frequent itemsets
- Dimensionality (number of items) of the data set
  - More space is needed to store support count of itemsets
  - if number of frequent itemsets also increases, both computation and I/O costs may also increase
- Size of database
- Average transaction width
  -

TID	*Items*
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Factors Affecting Complexity of Apriori

- Choice of minimum support threshold
  - lowering support threshold results in more frequent itemsets
  - this may increase number of candidates and max length of frequent itemsets
- Dimensionality (number of items) of the data set
  - More space is needed to store support count of itemsets
  - if number of frequent itemsets also increases, both computation and I/O costs may also increase
- Size of database
  - run time of algorithm increases with number of transactions
- Average transaction width

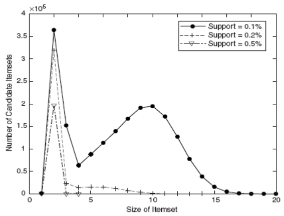
TID	*Items*
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Factors Affecting Complexity of Apriori

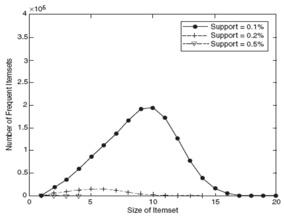
- Choice of minimum support threshold
  - lowering support threshold results in more frequent itemsets
  - this may increase number of candidates and max length of frequent itemsets
- Dimensionality (number of items) of the data set
  - More space is needed to store support count of itemsets
  - if number of frequent itemsets also increases, both computation and I/O costs may also increase
- Size of database
  - run time of algorithm increases with number of transactions
- Average transaction width
  - transaction width increases the max length of frequent itemsets
  - number of subsets in a transaction increases with its width, increasing computation time for support counting

---

Factors Affecting Complexity of Apriori

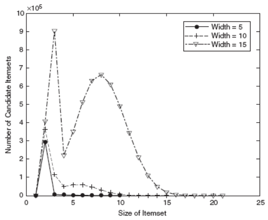


(a) Number of candidate itemsets.

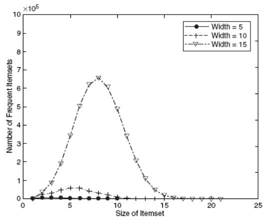


(b) Number of frequent itemsets.

Figure 6.13. Effect of support threshold on the number of candidate and frequent itemsets.



(a) Number of candidate itemsets.



(b) Number of Frequent Itemsets.

Figure 6.14. Effect of average transaction width on the number of candidate and frequent itemsets.

6

Compact Representation of Frequent Itemsets

- Some frequent itemsets are redundant because their supersets are also frequent

Consider the following data set. Assume support threshold =5

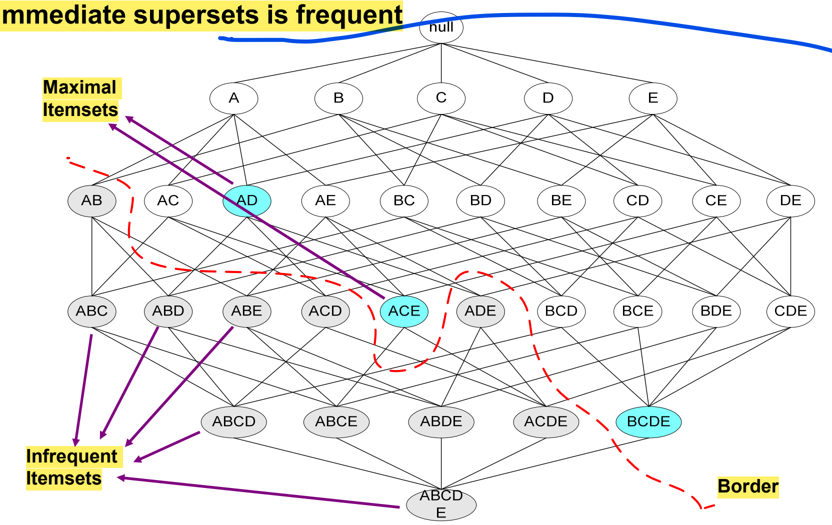
TID	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

Number of frequent itemsets $= 3 \times \sum_{k=1}^{10} \binom{10}{k}$

- Need a compact representation

Maximal Frequent Itemset

An itemset is maximal frequent if it is frequent and none of its immediate supersets is frequent



intern

D

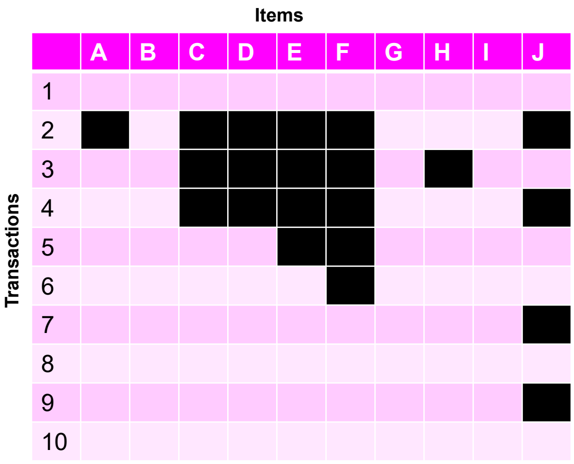
What are the Maximal Frequent Itemsets in this Data?

TID	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

Minimum support threshold = 5

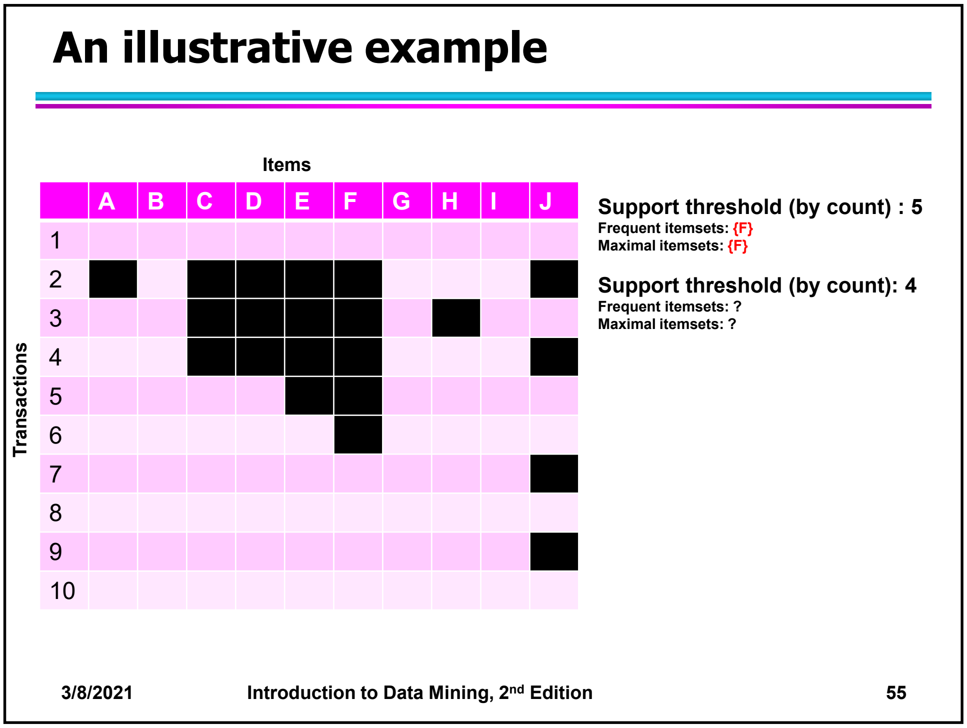
(A1-A10)
(B1-B10)
(C1-C10)

An illustrative example



	A	B	C	D	E	F	G	H	I	J
1										
2	■		■	■	■	■				■
3			■	■	■	■		■		
4			■	■	■	■				■
5					■	■				
6						■				
7										■
8										
9										■
10										

Support threshold (by count) : 5
Frequent itemsets: ?
Maximal itemsets: ?



An illustrative example

Items

Transactions

	A	B	C	D	E	F	G	H	I	J
1										
2	■		■	■	■	■				■
3			■	■	■	■		■		
4			■	■	■	■				■
5					■	■				
6						■				
7										■
8										
9										■
10										

Support threshold (by count) : 5

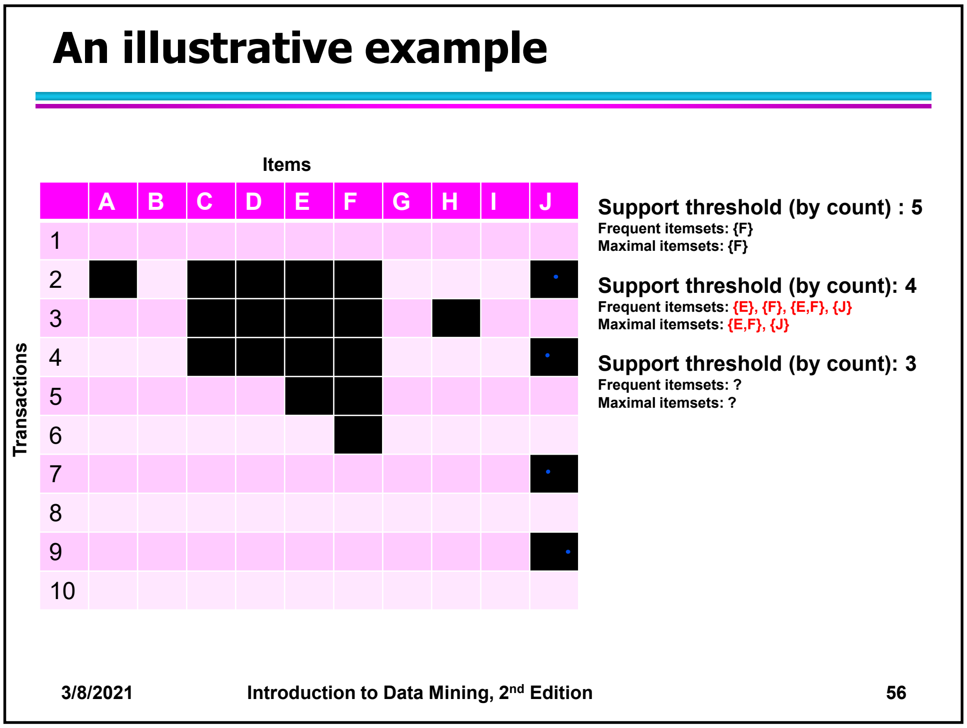
Frequent itemsets: {F}

Maximal itemsets: {F}

Support threshold (by count): 4

Frequent itemsets: ?

Maximal itemsets: ?



An illustrative example

Items

Transactions

	A	B	C	D	E	F	G	H	I	J
1										
2	■		■	■	■	■				■
3			■	■	■	■		■		
4			■	■	■	■				■
5					■	■				
6						■				
7										■
8										
9										■
10										

Support threshold (by count) : 5

Frequent itemsets: {F}

Maximal itemsets: {F}

Support threshold (by count): 4

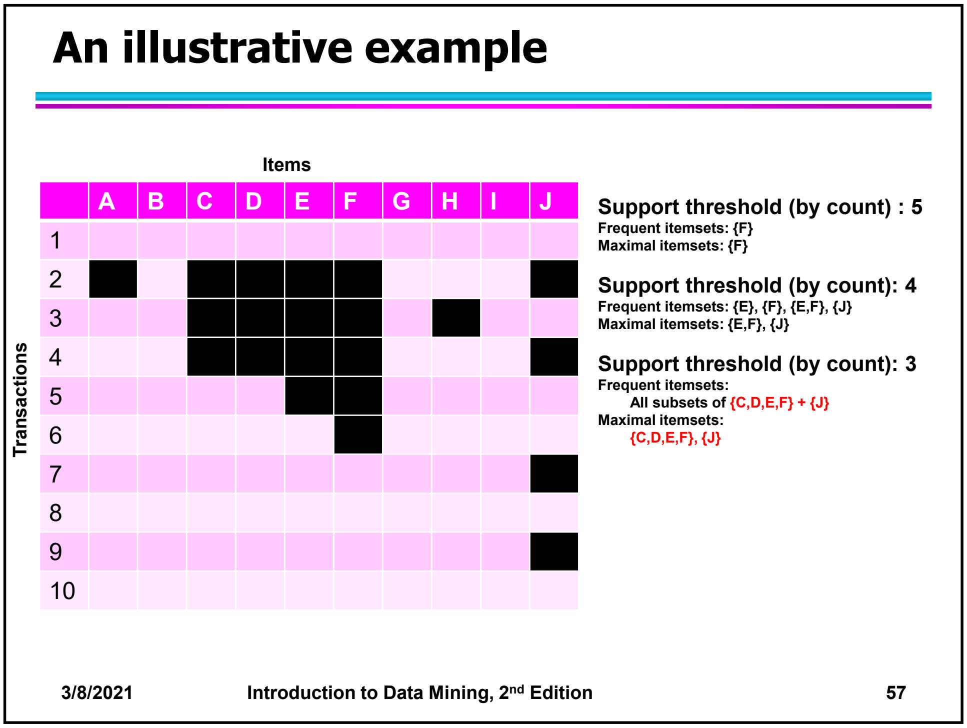
Frequent itemsets: {E}, {F}, {E,F}, {J}

Maximal itemsets: {E,F}, {J}

Support threshold (by count): 3

Frequent itemsets: ?

Maximal itemsets: ?



An illustrative example

Items

Transactions

	A	B	C	D	E	F	G	H	I	J
1										
2	■		■	■	■	■				■
3			■	■	■	■		■		
4			■	■	■	■				■
5					■	■				
6						■				
7										■
8										
9										■
10										

Support threshold (by count) : 5

Frequent itemsets: {F}

Maximal itemsets: {F}

Support threshold (by count): 4

Frequent itemsets: {E}, {F}, {E,F}, {J}

Maximal itemsets: {E,F}, {J}

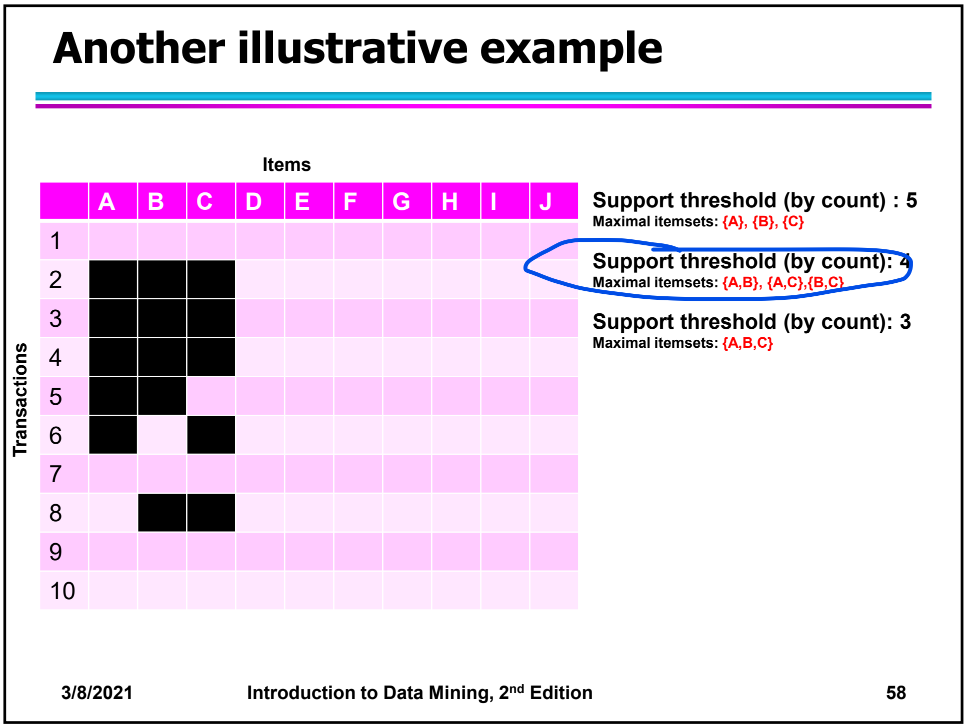
Support threshold (by count): 3

Frequent itemsets:

All subsets of {C,D,E,F} + {J}

Maximal itemsets:

{C,D,E,F}, {J}



Another illustrative example

Items

Transactions

	A	B	C	D	E	F	G	H	I	J
1										
2	■	■	■							
3	■	■	■							
4	■	■	■							
5	■	■								
6	■		■							
7										
8		■	■							
9										
10										

Support threshold (by count) : 5

Maximal itemsets: {A}, {B}, {C}

Support threshold (by count): 4

Maximal itemsets: {A,B}, {A,C},{B,C}

Support threshold (by count): 3

Maximal itemsets: {A,B,C}

Closed Itemset

- An itemset X is closed if none of its immediate supersets has the same support as the itemset X.
- X is not closed if at least one of its immediate supersets has support count as X.

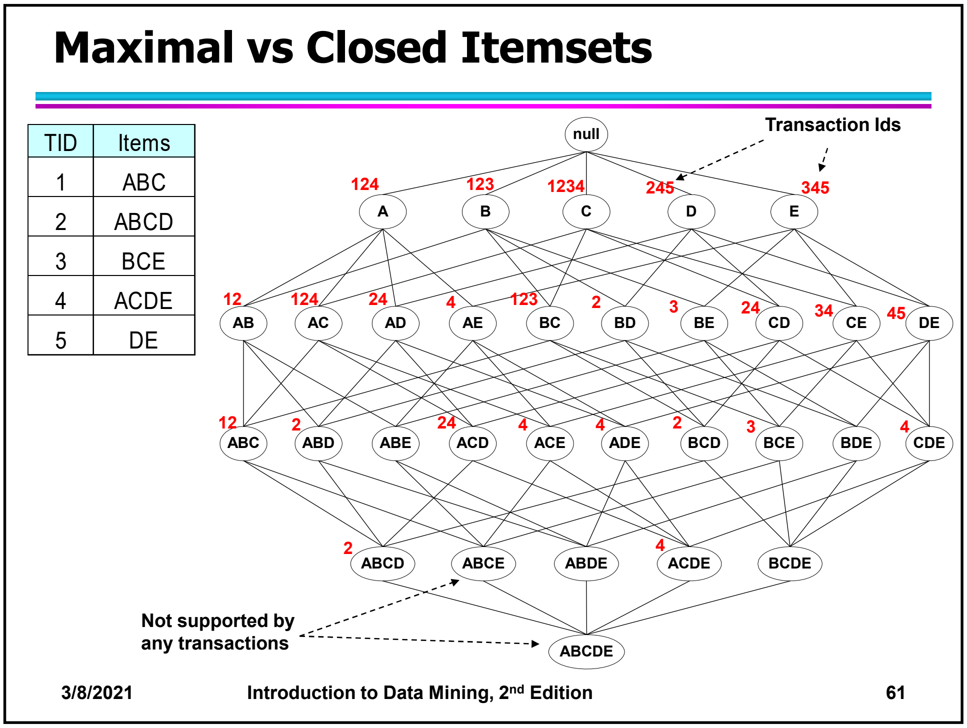
Closed Itemset

- An itemset X is closed if none of its immediate supersets has the same support as the itemset X.
- X is not closed if at least one of its immediate supersets has support count as X.

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,B,C,D}
4	{A,B,D}
5	{A,B,C,D}

Itemset	Support
{A}	4
{B}	5
{C}	3
{D}	4
{A,B}	4
{A,C}	2
{A,D}	3
{B,C}	3
{B,D}	4
{C,D}	3

Itemset	Support
{A,B,C}	2
{A,B,D}	3
{A,C,D}	2
{B,C,D}	2
{A,B,C,D}	2

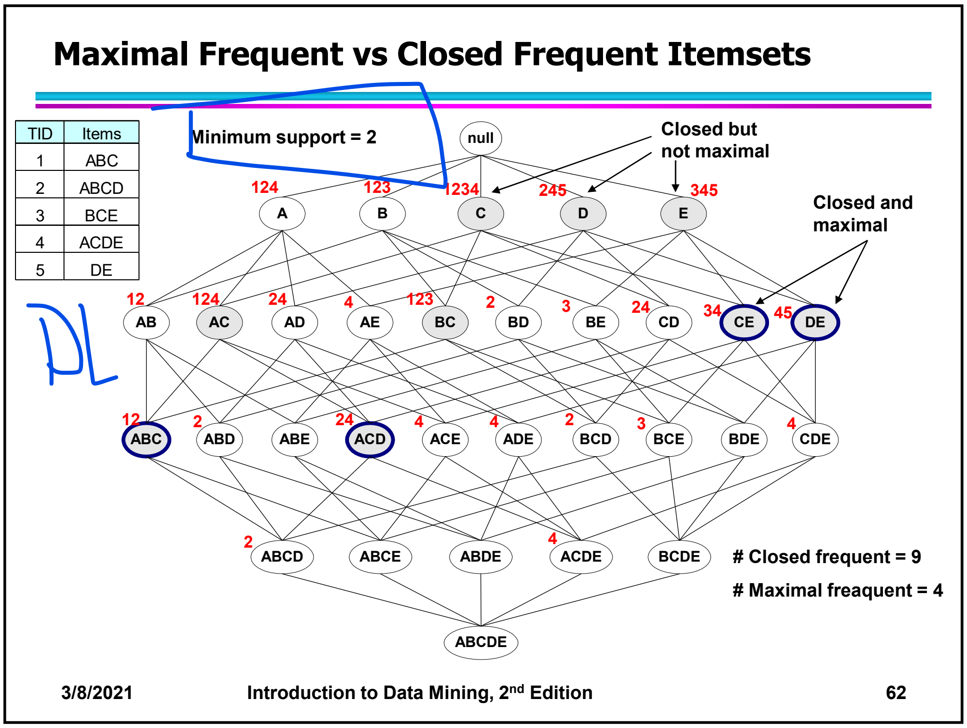


Maximal vs Closed Itemsets

TID	Items
1	ABC
2	ABCD
3	BCE
4	ACDE
5	DE

Transaction Ids

Not supported by any transactions



Maximal Frequent vs Closed Frequent Itemsets

TID	Items
1	ABC
2	ABCD
3	BCE
4	ACDE
5	DE

Minimum support = 2

DL

Closed but not maximal

Closed and maximal

Closed frequent = 9

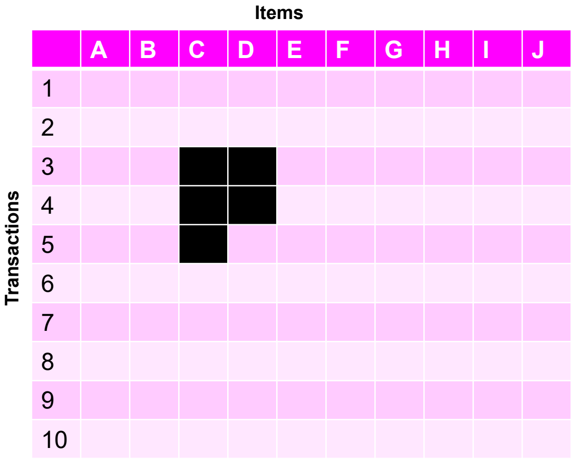
Maximal freaquent = 4

What are the Closed Itemsets in this Data?

TID	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

(A1-A10)
(B1-B10)
(C1-C10)

Example 1

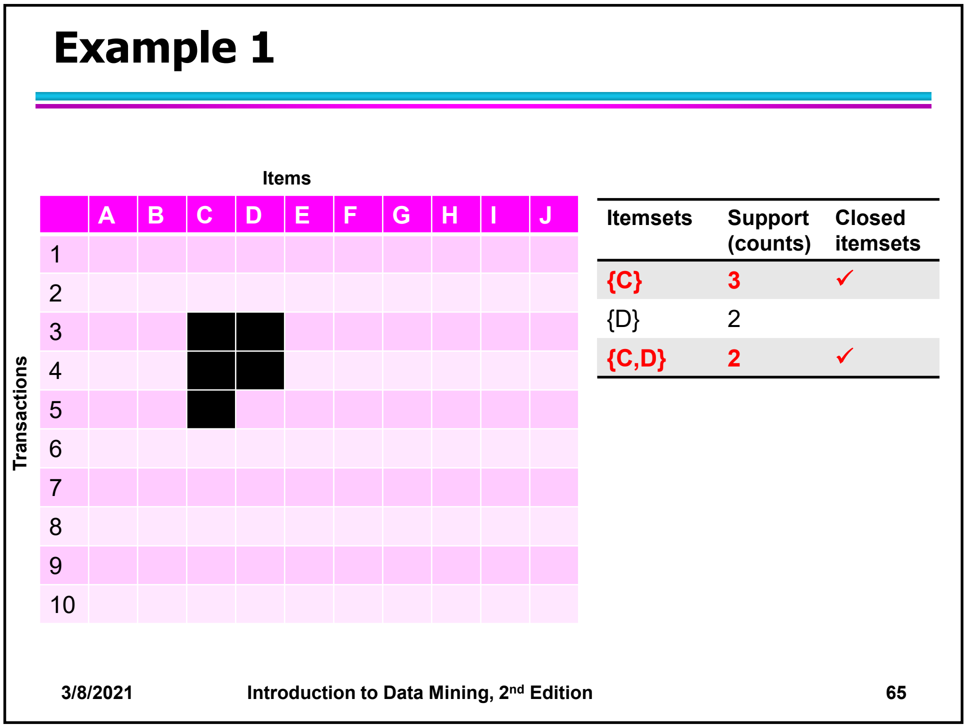


Items

Transactions

	A	B	C	D	E	F	G	H	I	J
1										
2										
3			■	■						
4			■	■						
5			■							
6										
7										
8										
9										
10										

Itemsets	Support (counts)	Closed itemsets
{C}	3	
{D}	2	
{C,D}	2	



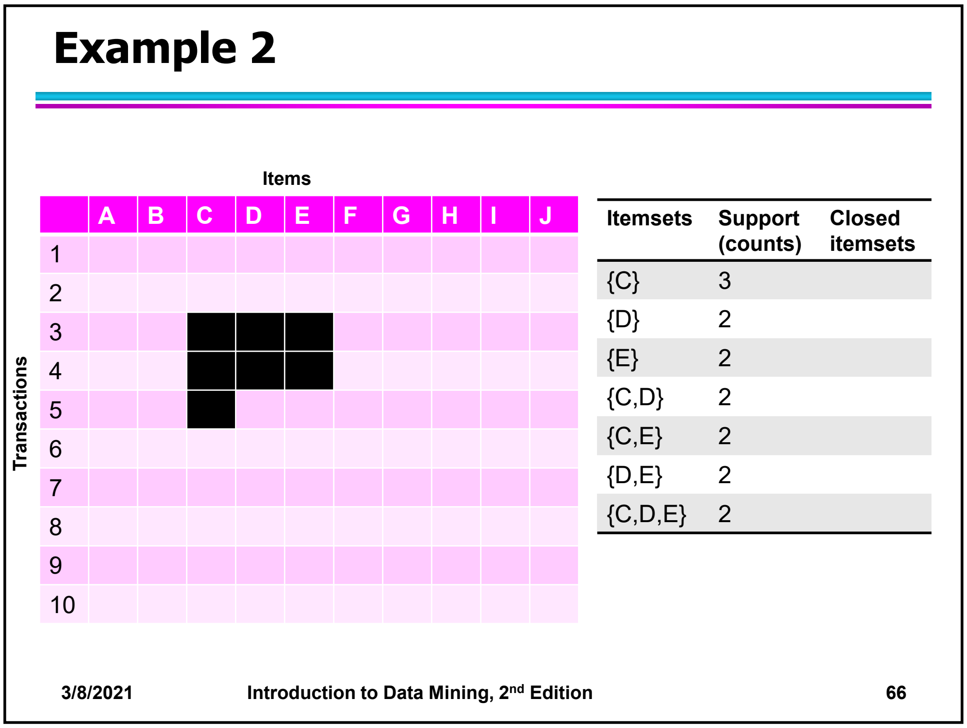
Example 1

Items

Transactions

	A	B	C	D	E	F	G	H	I	J
1										
2										
3			■	■						
4			■	■						
5			■							
6										
7										
8										
9										
10										

Itemsets	Support (counts)	Closed itemsets
{C}	**3**	✓
{D}	2	
{C,D}	**2**	✓



Example 2

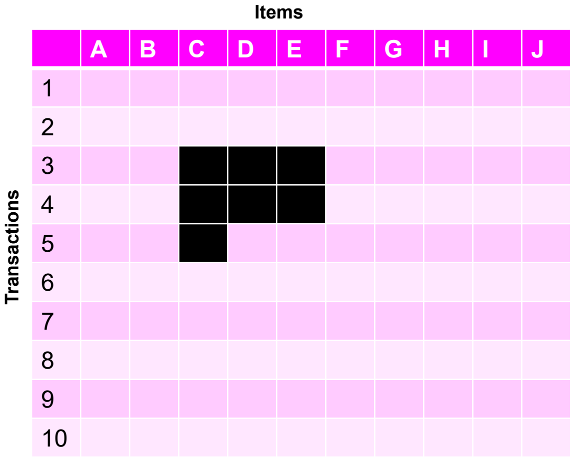
Items

Transactions

	A	B	C	D	E	F	G	H	I	J
1										
2										
3			■	■	■					
4			■	■	■					
5			■							
6										
7										
8										
9										
10										

Itemsets	Support (counts)	Closed itemsets
{C}	3	
{D}	2	
{E}	2	
{C,D}	2	
{C,E}	2	
{D,E}	2	
{C,D,E}	2	

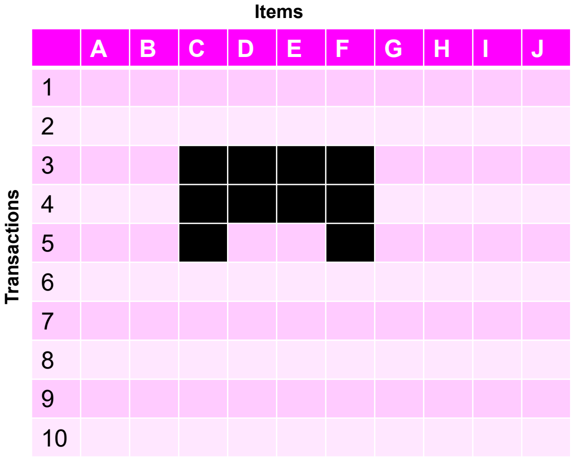
Example 2



	A	B	C	D	E	F	G	H	I	J
1										
2										
3			■	■	■					
4			■	■	■					
5			■							
6										
7										
8										
9										
10										

Itemsets	Support (counts)	Closed itemsets
{C}	**3**	✓
{D}	2	
{E}	2	
{C,D}	2	
{C,E}	2	
{D,E}	2	
{C,D,E}	**2**	✓

Example 3



	A	B	C	D	E	F	G	H	I	J
1										
2										
3			■	■	■	■				
4			■	■	■	■				
5			■			■				
6										
7										
8										
9										
10										

Closed itemsets: {C,D,E,F}, {C,F}

Example 4

Items

Transactions

	A	B	C	D	E	F	G	H	I	J
1										
2										
3			■	■	■	■				
4			■	■	■	■				
5			■							
6						■				
7										
8										
9										
10										

Closed itemsets: {C,D,E,F}, {C}, {F}

Maximal vs Closed Itemsets

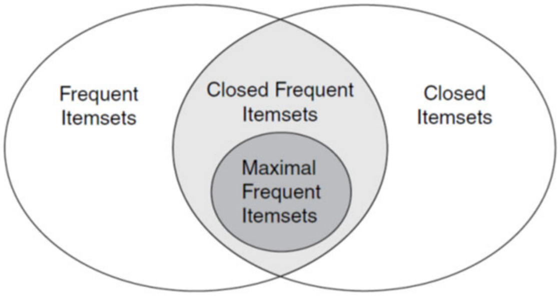
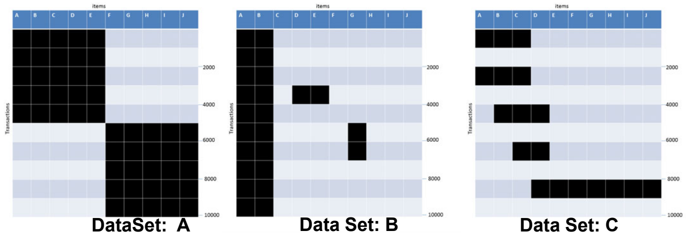


Figure 5.18. Relationships among frequent, closed, closed frequent, and maximal frequent itemsets.

Example question

- Given the following transaction data sets (dark cells indicate presence of an item in a transaction) and a support threshold of 20%, answer the following questions



DataSet: A **Data Set: B** **Data Set: C**

a. **What is the number of frequent itemsets for each dataset? Which dataset will produce the most number of frequent itemsets?**
b. **Which dataset will produce the longest frequent itemset?**
c. **Which dataset will produce frequent itemsets with highest maximum support?**
d. **Which dataset will produce frequent itemsets containing items with widely varying support levels (i.e., itemsets containing items with mixed support, ranging from 20% to more than 70%)?**
e. **What is the number of maximal frequent itemsets for each dataset? Which dataset will produce the most number of maximal frequent itemsets?**
f. **What is the number of closed frequent itemsets for each dataset? Which dataset will produce the most number of closed frequent itemsets?**

Pattern Evaluation

- Association rule algorithms can produce large number of rules

- Interestingness measures can be used to prune/rank the patterns
  - In the original formulation, support & confidence are the only measures used

Computing Interestingness Measure

- Given X → Y or {X,Y}, information needed to compute interestingness can be obtained from a contingency table

Contingency table

	Y	$\overline{Y}$	
X	f_{11}	f_{10}	f_{1+}
$\overline{X}$	f_{01}	f_{00}	f_{o+}
	f_{+1}	f_{+0}	N

f_{11}: support of X and Y
f_{10}: support of X and $\overline{Y}$
f_{01}: support of $\overline{X}$ and Y
f_{00}: support of $\overline{X}$ and $\overline{Y}$

Used to define various measures

- support, confidence, Gini, entropy, etc.

Drawback of Confidence

Customers	Tea	Coffee	…
C1	0	1	…
C2	1	0	…
C3	1	1	…
C4	1	0	…
…			

	$Coffee$	$\overline{Coffee}$	
Tea	150	50	200
$\overline{Tea}$	650	150	800
	800	200	1000

Association Rule: Tea → Coffee

Confidence ≅ P(Coffee|Tea) = 150/200 = 0.75

Confidence > 50%, meaning people who drink tea are more likely to drink coffee than not drink coffee

So rule seems reasonable

Drawback of Confidence

	Coffee	$\overline{\text{Coffee}}$	
Tea	150	50	200
$\overline{\text{Tea}}$	650	150	800
	800	200	1000

Association Rule: Tea → Coffee

Confidence= P(Coffee|Tea) = 150/200 = 0.75

but P(Coffee) = 0.8, which means knowing that a person drinks tea reduces the probability that the person drinks coffee!

⇒ Note that P(Coffee|$\overline{\text{Tea}}$) = 650/800 = 0.8125

Drawback of Confidence

Customers	Tea	Honey	…
C1	0	1	…
C2	1	0	…
C3	1	1	…
C4	1	0	…
…			

	$Honey$	$\overline{Honey}$	
Tea	100	100	200
$\overline{Tea}$	20	780	800
	120	880	1000

Association Rule: Tea → Honey

Confidence ≅ P(Honey|Tea) = 100/200 = 0.50

Confidence = 50%, which may mean that drinking tea has little influence whether honey is used or not

So rule seems uninteresting

But P(Honey) = 120/1000 = .12 (hence tea drinkers are far more likely to have honey

Measure for Association Rules

- So, what kind of rules do we really want?
  - Confidence(X → Y) should be sufficiently high
    - ◆ To ensure that people who buy X will more likely buy Y than not buy Y
  - Confidence(X → Y) > support(Y)
    - ◆ Otherwise, rule will be misleading because having item X actually reduces the chance of having item Y in the same transaction
    - ◆ Is there any measure that capture this constraint?
      - Answer: Yes. There are many of them.

Statistical Relationship between X and Y

- The criterion

  confidence(X → Y) = support(Y)

  is equivalent to:
  - P(Y|X) = P(Y)
  - P(X,Y) = P(X) × P(Y) (X and Y are independent)

  If P(X,Y) > P(X) × P(Y) : X & Y are positively correlated

  If P(X,Y) < P(X) × P(Y) : X & Y are negatively correlated

Measures that take into account statistical dependence

$$Lift = \frac{P(Y \mid X)}{P(Y)}$$

$$Interest = \frac{P(X,Y)}{P(X)P(Y)}$$

lift is used for rules while interest is used for itemsets

$$PS = P(X,Y) - P(X)P(Y)$$

$$\phi - coefficient = \frac{P(X,Y) - P(X)P(Y)}{\sqrt{P(X)[1-P(X)]P(Y)[1-P(Y)]}}$$

Example: Lift/Interest

	Coffee	$\overline{\text{Coffee}}$	
Tea	150	50	200
$\overline{\text{Tea}}$	650	150	800
	800	200	1000

Association Rule: Tea → Coffee

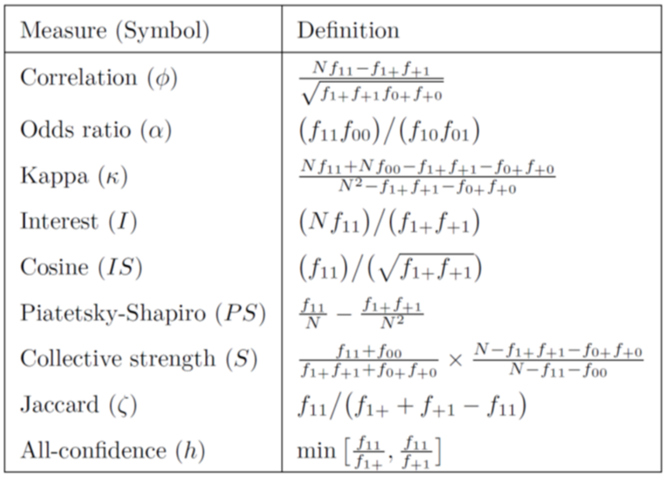
Confidence= P(Coffee|Tea) = 0.75

but P(Coffee) = 0.8

⇒ Interest = 0.15 / (0.2×0.8) = 0.9375 (< 1, therefore is negatively associated)

So, is it enough to use confidence/Interest for pruning?

There are lots of measures proposed in the literature



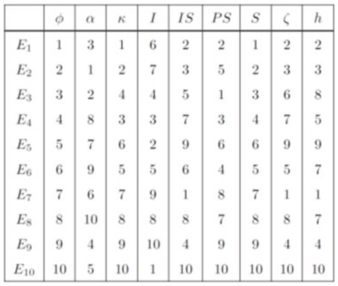
Measure (Symbol)	Definition
Correlation (ϕ)	$\frac{Nf_{11}-f_{1+}f_{+1}}{\sqrt{f_{1+}f_{+1}f_{0+}f_{+0}}}$
Odds ratio (α)	$(f_{11}f_{00})/(f_{10}f_{01})$
Kappa (κ)	$\frac{Nf_{11}+Nf_{00}-f_{1+}f_{+1}-f_{0+}f_{+0}}{N^2-f_{1+}f_{+1}-f_{0+}f_{+0}}$
Interest (I)	$(Nf_{11})/(f_{1+}f_{+1})$
Cosine (IS)	$(f_{11})/(\sqrt{f_{1+}f_{+1}})$
Piatetsky-Shapiro (PS)	$\frac{f_{11}}{N} - \frac{f_{1+}f_{+1}}{N^2}$
Collective strength (S)	$\frac{f_{11}+f_{00}}{f_{1+}f_{+1}+f_{0+}f_{+0}} \times \frac{N-f_{1+}f_{+1}-f_{0+}f_{+0}}{N-f_{11}-f_{00}}$
Jaccard (ζ)	$f_{11}/(f_{1+} + f_{+1} - f_{11})$
All-confidence (h)	$\min\left[\frac{f_{11}}{f_{1+}}, \frac{f_{11}}{f_{+1}}\right]$

Comparing Different Measures

10 examples of contingency tables:

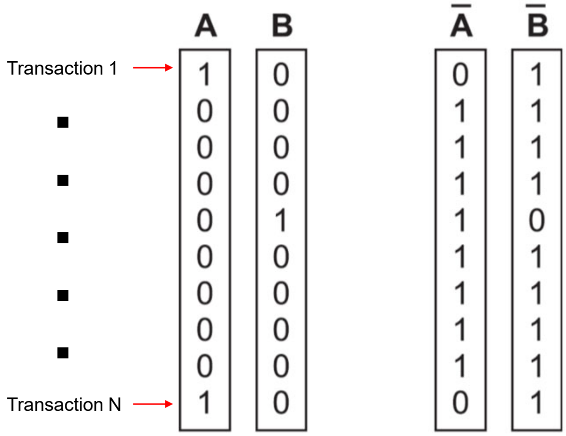
Example	f_{11}	f_{10}	f_{01}	f_{00}
E1	8123	83	424	1370
E2	8330	2	622	1046
E3	9481	94	127	298
E4	3954	3080	5	2961
E5	2886	1363	1320	4431
E6	1500	2000	500	6000
E7	4000	2000	1000	3000
E8	4000	2000	2000	2000
E9	1720	7121	5	1154
E10	61	2483	4	7452

Rankings of contingency tables using various measures:



	ϕ	α	κ	I	IS	PS	S	ζ	h
E_1	1	3	1	6	2	2	1	2	2
E_2	2	1	2	7	3	5	2	3	3
E_3	3	2	4	4	5	1	3	6	8
E_4	4	8	3	3	7	3	4	7	5
E_5	5	7	6	2	9	6	6	9	9
E_6	6	9	5	5	6	4	5	5	7
E_7	7	6	7	9	1	8	7	1	1
E_8	8	10	8	8	8	7	8	8	7
E_9	9	4	9	10	4	9	9	4	4
E_{10}	10	5	10	1	10	10	10	10	10

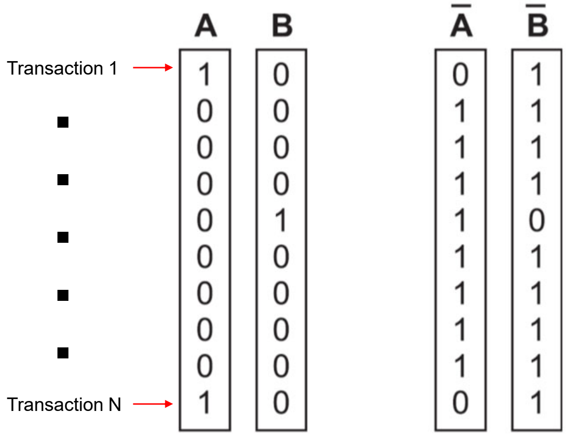
Property under Inversion Operation



	A	B	$\bar{A}$	$\bar{B}$
Transaction 1	1	0	0	1
	0	0	1	1
	0	0	1	1
	0	0	1	1
	0	1	1	0
	0	0	1	1
	0	0	1	1
	0	0	1	1
	0	0	1	1
Transaction N	1	0	0	1

(a) (b)

Property under Inversion Operation



	A	B	$\bar{A}$	$\bar{B}$
Transaction 1	1	0	0	1
	0	0	1	1
	0	0	1	1
	0	0	1	1
	0	1	1	0
	0	0	1	1
	0	0	1	1
	0	0	1	1
	0	0	1	1
Transaction N	1	0	0	1

(a) (b)

Correlation:	**-0.1667**	**-0.1667**
IS/cosine	**0.0**	**0.825**

Property under Null Addition

	B	$\overline{B}$	
A	700	100	800
$\overline{A}$	100	100	200
	800	200	1000

⇒

	B	$\overline{B}$	
A	700	100	800
$\overline{A}$	10	1100	1200
	800	1200	2000

Invariant measures:

- cosine, Jaccard, All-confidence, confidence

Non-invariant measures:

- correlation, Interest/Lift, odds ratio, etc

Property under Row/Column Scaling

Grade-Gender Example (Mosteller, 1968):

	Male	Female	
High	30	20	50
Low	40	10	50
	70	30	100

	Male	Female	
High	60	60	120
Low	80	30	110
	140	90	230

↓ 2x ↓ 3x

Mosteller:

Underlying association should be independent of the relative number of male and female students in the samples

Odds-Ratio $((f_{11+}f_{00})/(f_{10+}f_{10}))$ has this property

Property under Row/Column Scaling

Relationship between Mask use and susceptibility to Covid:

	Covid-Positive	Covid-Free	
Mask	20	30	50
No-Mask	40	10	50
	60	40	100

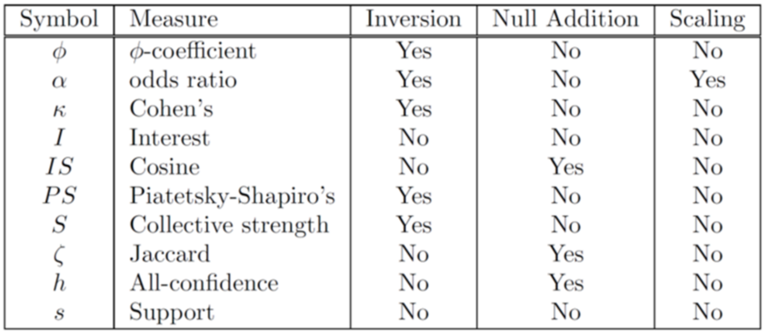
	Covid-Positive	Covid-Free	
Mask	40	300	340
No-Mask	80	100	180
	120 ↓	400 ↓	520
	2x	10x	

Mosteller:

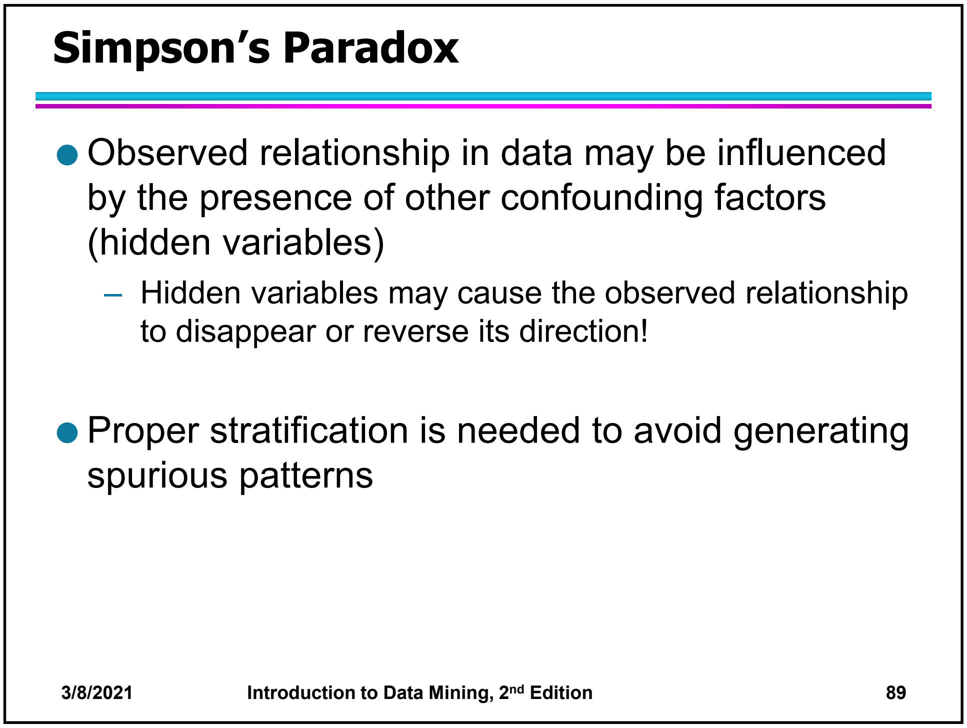
Underlying association should be independent of
the relative number of Covid-positive and Covid-free subjects

Odds-Ratio (($f_{11+}f_{00}$)/($f_{10+}f_{10}$)) has this property

Different Measures have Different Properties

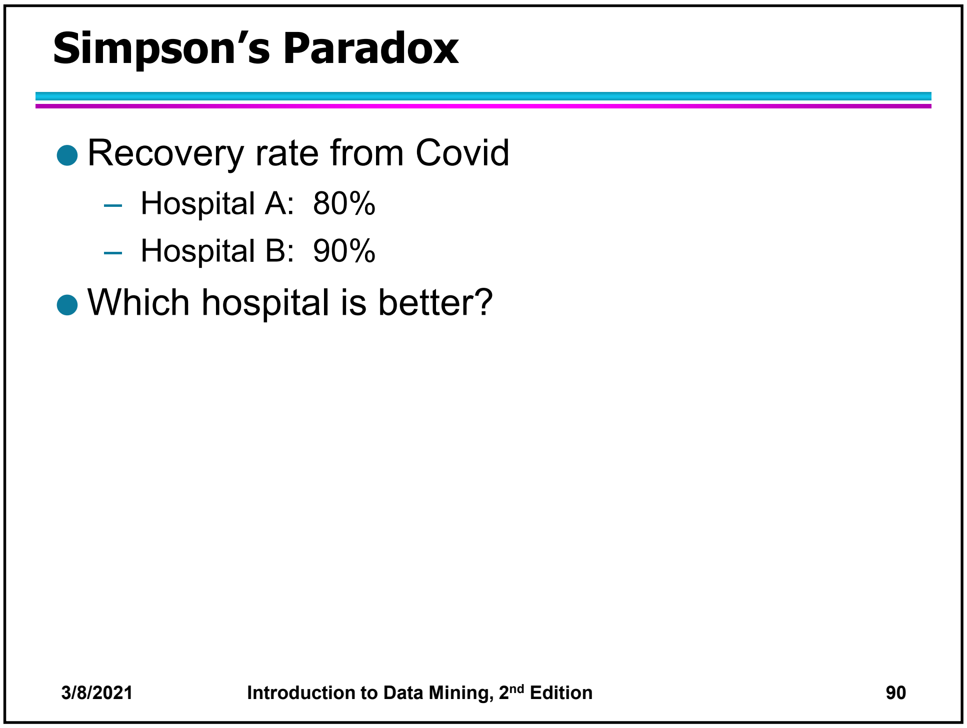


Symbol	Measure	Inversion	Null Addition	Scaling
ϕ	ϕ-coefficient	Yes	No	No
α	odds ratio	Yes	No	Yes
κ	Cohen's	Yes	No	No
I	Interest	No	No	No
IS	Cosine	No	Yes	No
PS	Piatetsky-Shapiro's	Yes	No	No
S	Collective strength	Yes	No	No
ζ	Jaccard	No	Yes	No
h	All-confidence	No	Yes	No
s	Support	No	No	No



Simpson’s Paradox

- Observed relationship in data may be influenced by the presence of other confounding factors (hidden variables)
  - Hidden variables may cause the observed relationship to disappear or reverse its direction!
- Proper stratification is needed to avoid generating spurious patterns



Simpson’s Paradox

- Recovery rate from Covid
  - Hospital A: 80%
  - Hospital B: 90%
- Which hospital is better?

Simpson's Paradox

- Recovery rate from Covid
  - Hospital A: 80%
  - Hospital B: 90%
- Which hospital is better?
- Covid recovery rate on older population
  - Hospital A: 50%
  - Hospital B: 30%
- Covid recovery rate on younger population
  - Hospital A: 99%
  - Hospital B: 98%

Simpson's Paradox

- Covid-19 death: (per 100,000 of population)
  - County A: 15
  - County B: 10
- Which state is managing the pandemic better?

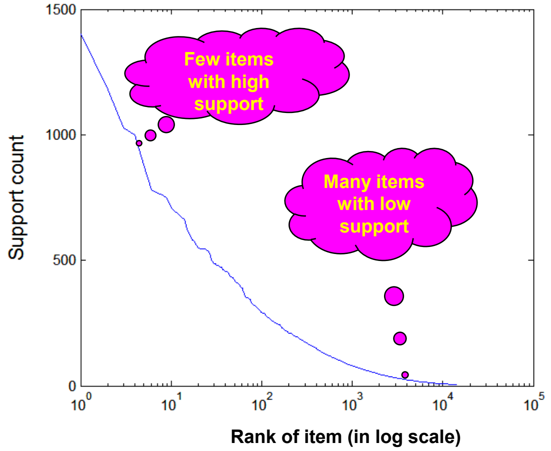
Simpson’s Paradox

- Covid-19 death: (per 100,000 of population)
  - County A: 15
  - County B: 10
- Which state is managing the pandemic better?
- Covid death rate on older population
  - County A: 20
  - County B: 40
- Covid death rate on younger population
  - County A: 2
  - County B: 5

Effect of Support Distribution on Association Mining

- Many real data sets have skewed support distribution

Support distribution of a retail data set

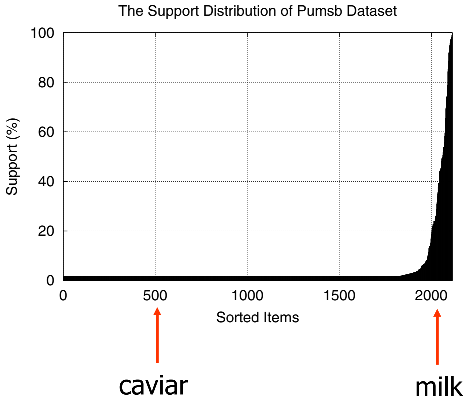


Effect of Support Distribution

- Difficult to set the appropriate *minsup* threshold
  - If *minsup* is too high, we could miss itemsets involving interesting rare items (e.g., {caviar, vodka})
  - If *minsup* is too low, it is computationally expensive and the number of itemsets is very large

---

Cross-Support Patterns



A cross-support pattern involves items with varying degree of support

- Example: {caviar,milk}

How to avoid such patterns?

A Measure of Cross Support

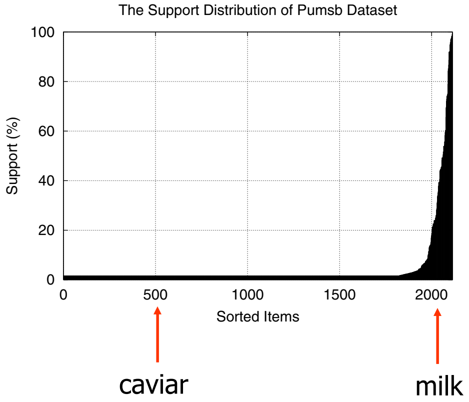
- Given an itemset, $X = \{x_1, x_2, \ldots, x_d\}$, with d items, we can define a measure of cross support, r, for the itemset

$$r(X) = \frac{\mathbf{min}\{s(x_1), s(x_2), \ldots, s(x_d)\}}{\mathbf{max}\{s(x_1), s(x_2), \ldots, s(x_d)\}}$$

where $s(x_i)$ is the support of item x_i

  - Can use $r(X)$ to prune cross support patterns

Confidence and Cross-Support Patterns



Observation:

conf(caviar→milk) is very high

but

conf(milk→caviar) is very low

Therefore,

min(conf(caviar→milk),
conf(milk→caviar))

is also very low

H-Confidence

- To avoid patterns whose items have very different support, define a new evaluation measure for itemsets
  - Known as h-confidence or all-confidence
- Specifically, given an itemset $X = \{x_1, x_2, \ldots, x_d\}$
  - h-confidence is the minimum confidence of any association rule formed from itemset X
  - hconf(X) = min(conf($X_1 \rightarrow X_2$)),

    where $X_1, X_2 \subset X, X_1 \cap X_2 = \emptyset, X_1 \cup X_2 = X$

    For example: $X_1 = \{x_1, x_2\}, X_2 = \{x_3, \ldots, x_d\}$

H-Confidence …

- But, given an itemset $X = \{x_1, x_2, \ldots, x_d\}$
  - What is the lowest confidence rule you can obtain from X?
  - Recall conf($X_1 \rightarrow X_2$) = s($X_1 \cup X_2$) / support(X_1)
    - The numerator is fixed: $s(X_1 \cup X_2) = s(X)$
    - Thus, to find the lowest confidence rule, we need to find the X_1 with highest support
    - Consider only rules where X_1 is a single item, i.e., $\{x_1\} \rightarrow X - \{x_1\}$, $\{x_2\} \rightarrow X - \{x_2\}$, …, or $\{x_d\} \rightarrow X - \{x_d\}$

$$\mathrm{hconf}(X) = \min\left\{\frac{s(X)}{s(x_1)}, \frac{s(X)}{s(x_2)}, \ldots, \frac{s(X)}{s(x_d)}\right\}$$

$$= \frac{s(X)}{\max\{s(x_1), s(x_2), \ldots, s(x_d)\}}$$

Cross Support and H-confidence

- By the anti-montone property of support

$$s(X) \leq \min\{s(x_1), s(x_2), \dots, s(x_d)\}$$

- Therefore, we can derive a relationship between the h-confidence and cross support of an itemset

$$\text{hconf}(X) = \frac{s(X)}{\max\{s(x_1),\ s(x_2),\ \dots,\ s(x_d)\}}$$

$$\leq \frac{\min\{s(x_1), s(x_2), \dots, s(x_d)\}}{\max\{s(x_1), s(x_2), \dots, s(x_d)\}}$$

$$= r(X)$$

Thus, $\text{hconf}(X) \leq r(X)$

Cross Support and H-confidence ...

- Since, $\text{hconf}(X) \leq r(X)$, we can eliminate cross support patterns by finding patterns with h-confidence < h_c, a user set threshold
- Notice that

$$0 \leq \text{hconf}(X) \leq r(X) \leq 1$$

- Any itemset satisfying a given h-confidence threshold, h_c, is called a hyperclique
- H-confidence can be used instead of or in conjunction with support

Properties of Hypercliques

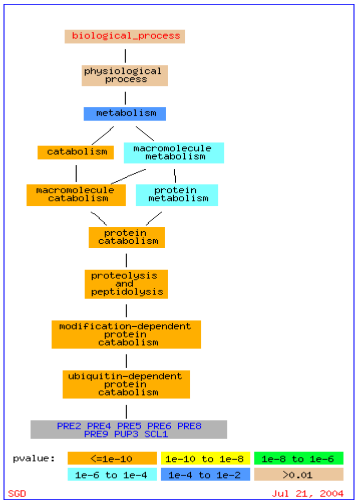
- Hypercliques are itemsets, but not necessarily frequent itemsets
  - Good for finding low support patterns
- H-confidence is anti-monotone
- Can define closed and maximal hypercliques in terms of h-confidence
  - A hyperclique X is closed if none of its immediate supersets has the same h-confidence as X
  - A hyperclique X is maximal if $\mathrm{hconf}(X) \leq \mathrm{h_c}$ and none of its immediate supersets, Y, have $\mathrm{hconf}(Y) \leq \mathrm{h_c}$

Properties of Hypercliques …

- Hypercliques have the high-affinity property
  - Think of the individual items as sparse binary vectors
  - h-confidence gives us information about their pairwise Jaccard and cosine similarity
    - Assume $x_1$and x_2 are any two items in an itemset X
    - $\mathrm{Jaccard}(x_1, x_2) \ \geq \mathrm{hconf}(X)/2$
    - $\cos(x_1, x_2) \ \geq \mathrm{hconf}(X)$
  - Hypercliques that have a high h-confidence consist of very similar items as measured by Jaccard and cosine
- The items in a hyperclique cannot have widely different support
  - Allows for more efficient pruning

Example Applications of Hypercliques

- Hypercliques are used to find strongly coherent groups of items
  - Words that occur together in documents
  - Proteins in a protein interaction network



In the figure at the right, a gene ontology hierarchy for biological process shows that the identified proteins in the hyperclique (PRE2, …, SCL1) perform the same function and are involved in the same biological process