Association Rules

Apriori Algorithm

Done By Mohammed Marwan Shahin

Training Agendas

- What is Association Rules?
- Frequent itemset
- Naïve Approach
- Apriori Algorithm workflow
- Demo1: Apply Apriori using SPMF tool
- Demo2: Apply Apriori using Python

Machine Learning types

Supervised Learning			Unsupervised Learning	
Regression	**Classification**	**Time Series**	**Clustering**	**Association Rules**
Linear Regression Polynomial Reg Decision tree Reg Random Forest Reg	Logistic Regression KNN Naïve Bayes Decision tree Random forest SVM	ARIMA SARIMA ARIMAX	K-mean H clustering Optics Chameleon	Apriori Eclat FP-growth

Market Basket Analysis

Market Basket Analysis

Market Basket Analysis is one of the key techniques used by large retailers to uncover associations between items.

Bread and Jam

Laptop and Bag

Market Basket Analysis

Market Basket Analysis is one of the key techniques used by large retailers to uncover associations between items.

A ⇒ B

IF THEN

Bread and Butter

What is Association rules ?

Frequent pattern: a pattern (a set of items, subsequences) that occurs frequently in a data set.

- Example: milk and bread, that appear frequently together in a transaction data set is a frequent itemset (frequent itemset).
- Buying first a PC, then a digital camera, and then a memory card (subsequences).

Applications:

- Basket data analysis,
- Cross-marketing,
- Catalog design,
- Sale campaign analysis,

Association Rules Metrics

$$A \Rightarrow B$$

$$Support = \frac{freq(A,B)}{N}$$

$$Confidence = \frac{freq(A,B)}{freq(A)}$$

$$Lift = \frac{Support}{Supp(A)\ x\ Supp(B)}$$

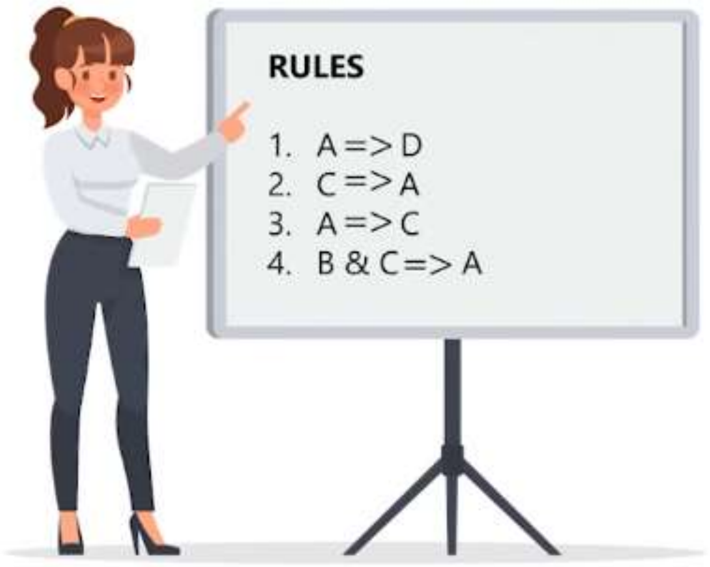
Association Rule Mining

Transaction at a Local Market

T1	A	B	C
T2	A	C	D
T3	B	C	D
T4	A	D	E
T5	B	C	E

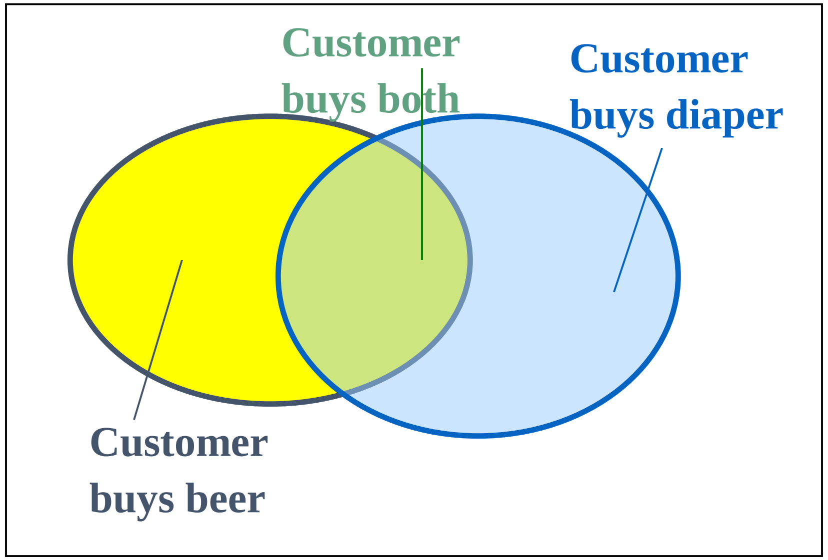
Association Rule Mining

Rule	Support	Confidence	Lift
A=>D	2/5	2/3	10/9
C=>A	2/5	2/4	5/6
A=>C	2/5	2/3	5/6
B, C=>A	1/5	1/3	5/9



Basic Concepts: Frequent Patterns

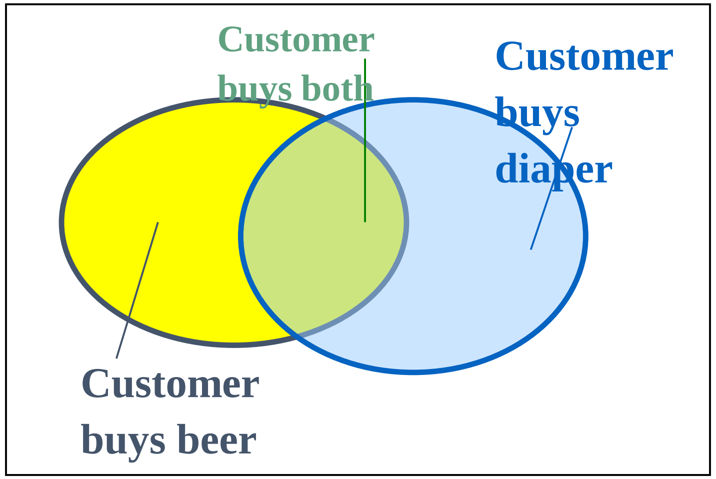
Tid	Items bought
10	Beer, Nuts, Diaper
20	Beer, Coffee, Diaper
30	Beer, Diaper, Eggs
40	Nuts, Eggs, Milk
50	Nuts, Coffee, Diaper, Eggs, Milk



- 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



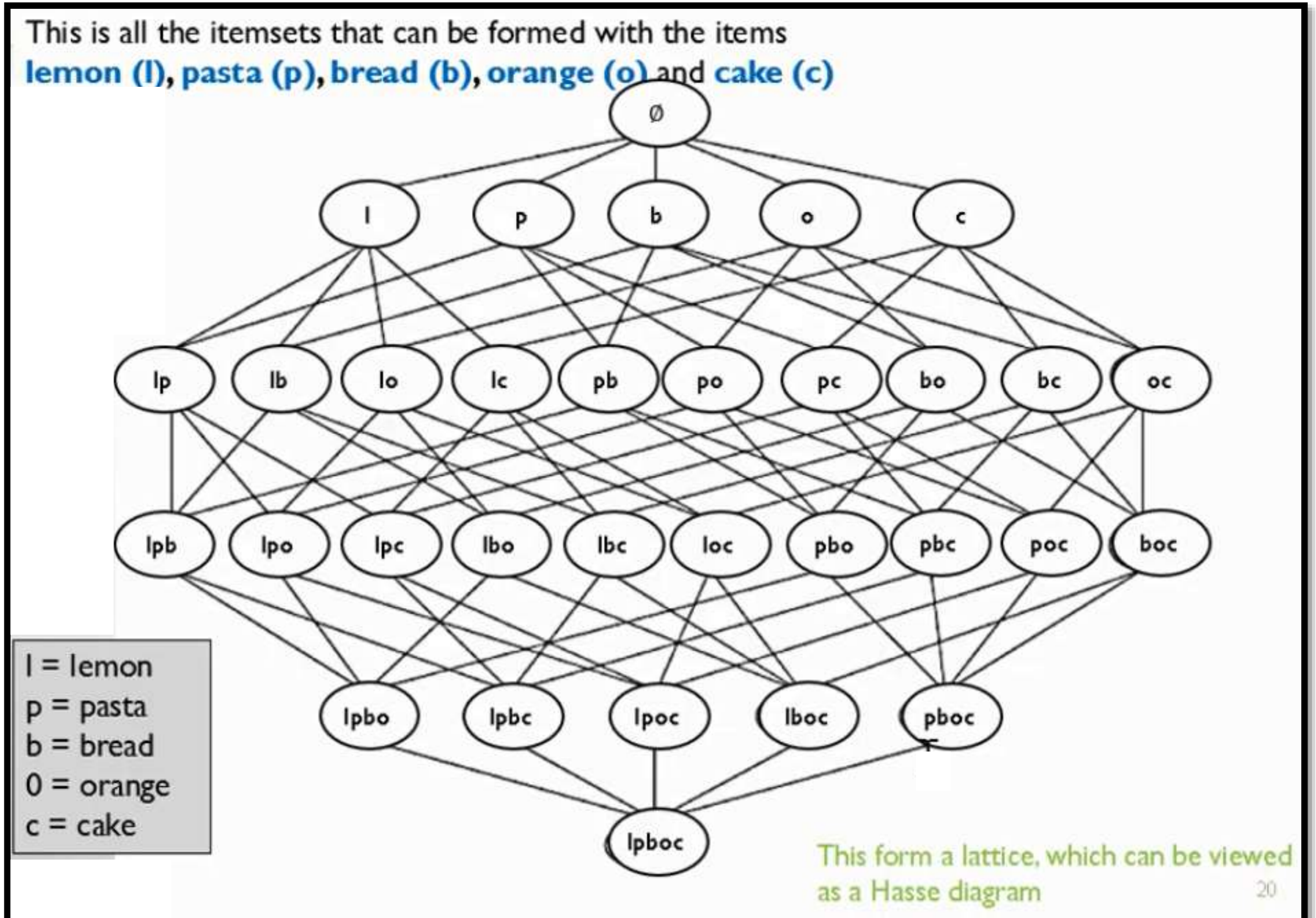
- Find all the rules $X \rightarrow Y$ with minimum support and confidence
  - support, s, probability that a transaction contains $X \cup Y$
  - $$S(x \rightarrow y) = \frac{\sigma(x \cup y)}{N}$$
  - confidence, c, conditional probability that a transaction having X also contains Y
  - $$S(x \rightarrow y) = \frac{\sigma(x \cup y)}{\sigma(x)}$$

Let minsup = 50%, minconf = 50%

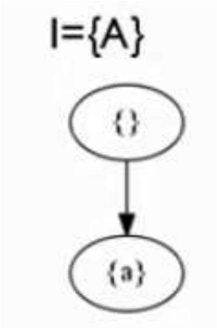
Freq. Pat.: Beer:3, Nuts:3, Diaper:4, Eggs:3, {Beer, Diaper}:3

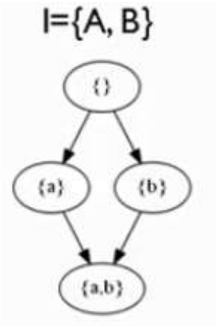
- Association rules: (many more!)
  - *Beer* → *Diaper* (60%, 100%)
  - *Diaper* → *Beer* (60%, 75%)

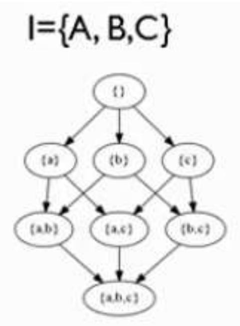
Search Space (Naïve Approach)

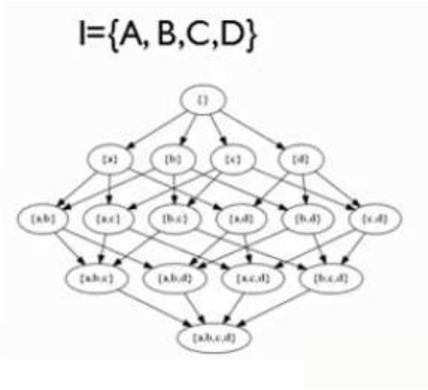


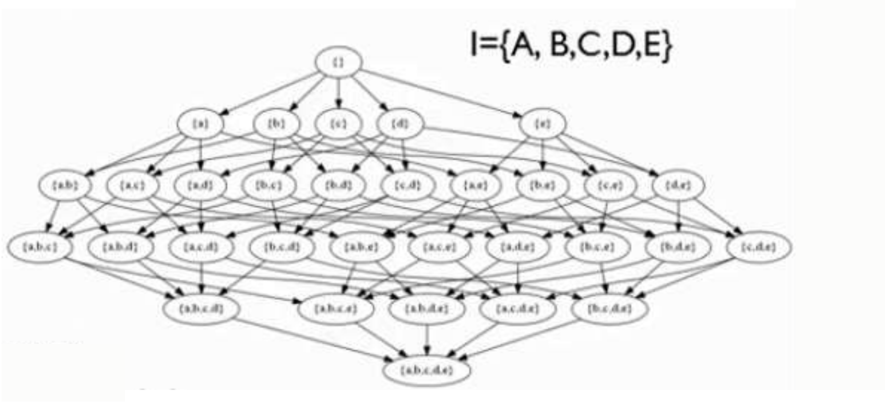
Search Space

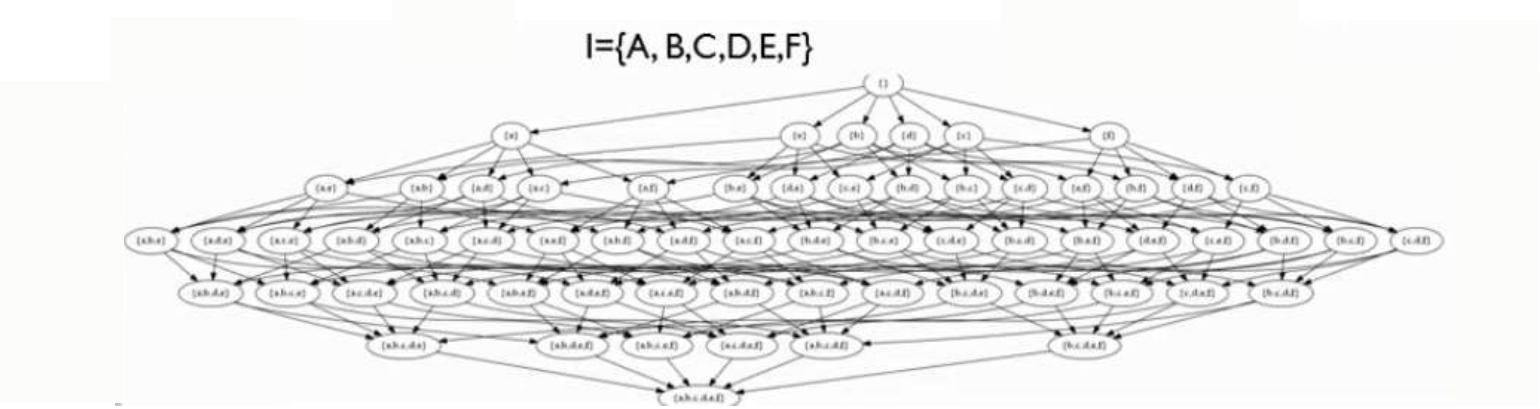












Apriori Algorithm

Uses a generate-and-test approach – generates candidate itemsets and tests if they are frequent

- Generation of candidate itemsets is expensive (in both space and time)
- Support counting is expensive
  - Subset checking (computationally expensive)
  - Multiple Database scans (I/O)

Frequent Itemset is an itemset whose support value is greater than a threshold value.

Apriori Property

Let these be two itemsets X and Y.

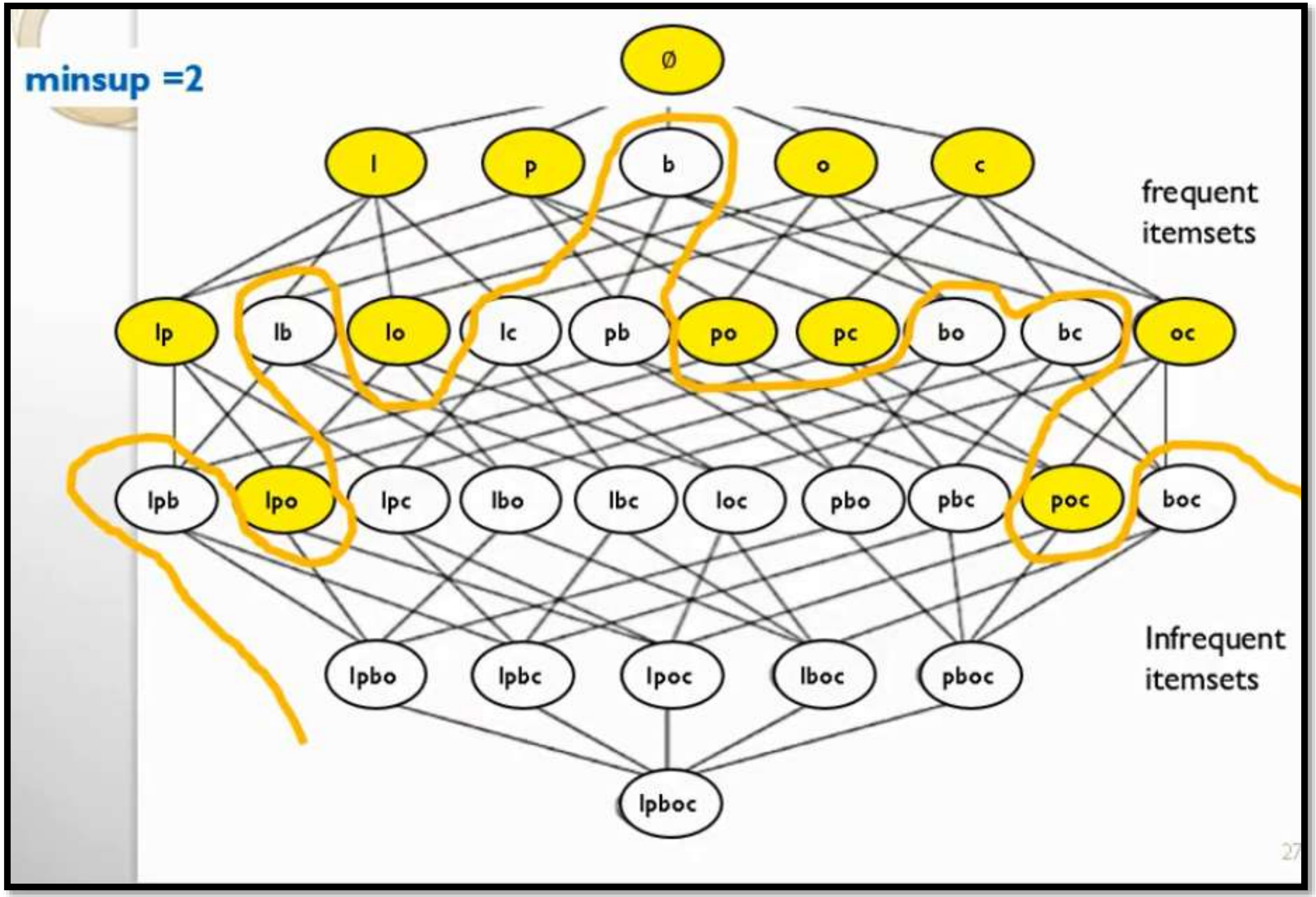
If X ⊂ Y, the support of Y is less than or equal to the support of X.

Example:

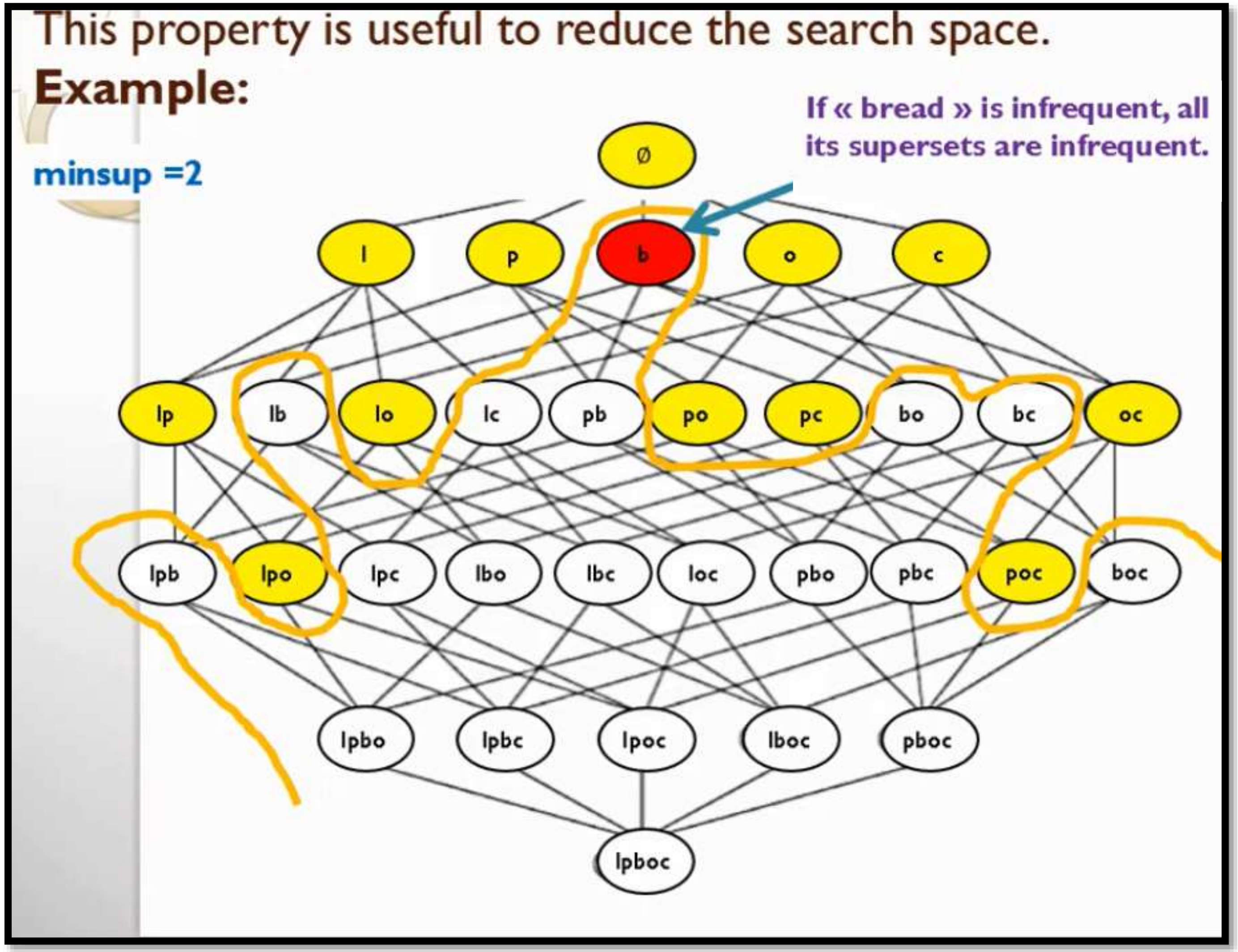
- The support of {pasta} is 4
- The support of {pasta, lemon} is 3
- The support of {pasta, lemon, orange} is 2

Transaction	Items appearing in the transaction
T1	{pasta, lemon, bread, orange}
T2	{pasta, lemon}
T3	{pasta, orange, cake}
T4	{pasta, lemon, orange, cake}

Apriori Property



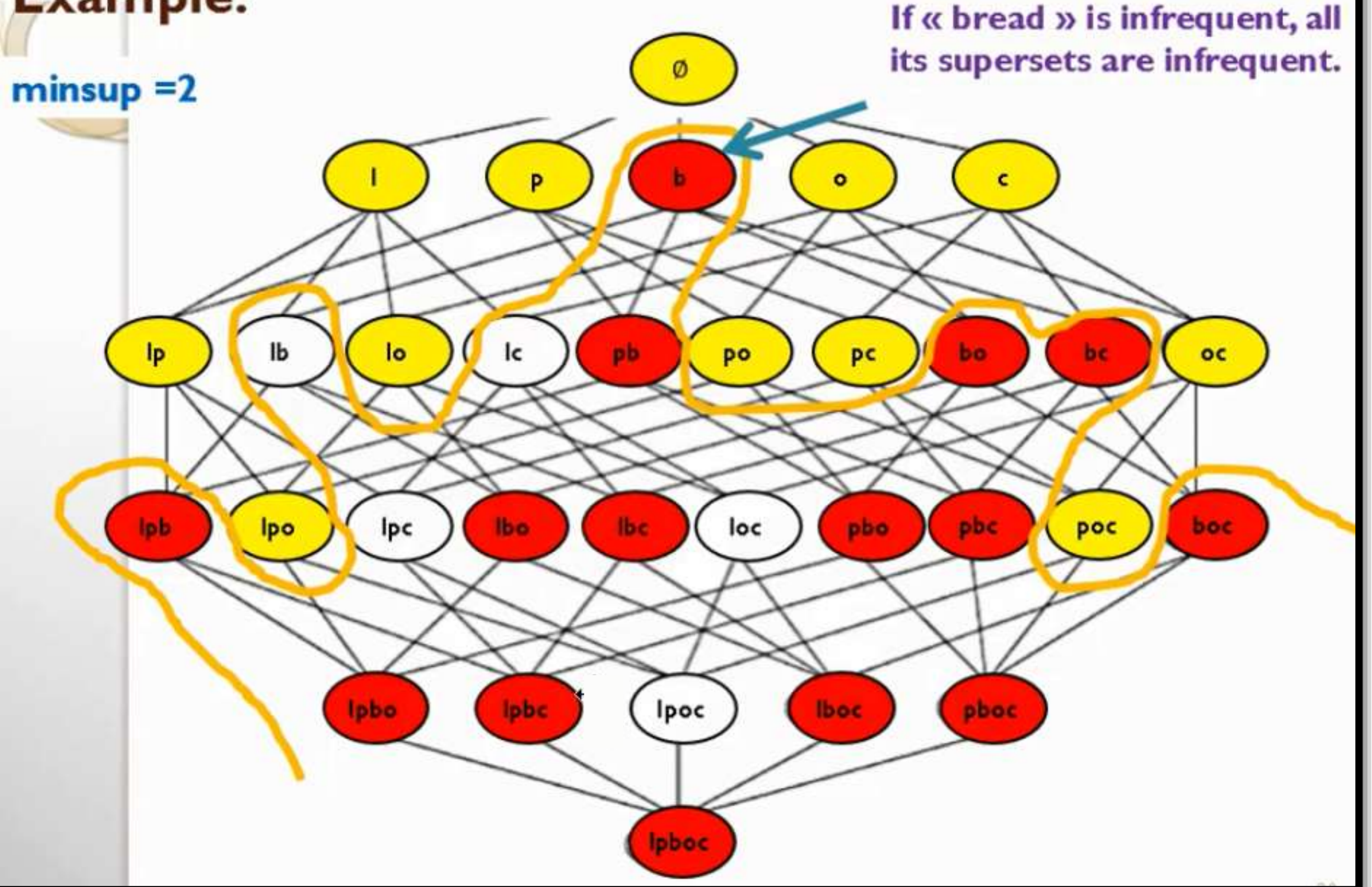
Apriori Property



Apriori Property

This property is useful to reduce the search space.

Example:



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
  - $C_4 = \{abcd\}$

Apriori Algorithm

TID	Items
T1	1 3 4
T2	2 3 5
T3	1 2 3 5
T4	2 5
T5	1 3 5

Min. Support count = 2

Apriori Algorithm – 1st Iteration

TID	Items
T1	1 3 4
T2	2 3 5
T3	1 2 3 5
T4	2 5
T5	1 3 5

C1

Itemset	Support
{1}	3
{2}	3
{3}	4
{4}	1
{5}	4

Apriori Algorithm – 1^{st} Iteration

C1

Itemset	Support
{1}	3
{2}	3
{3}	4
{4}	1
{5}	4

F1

Itemset	Support
{1}	3
{2}	3
{3}	4
{5}	4

Item sets with support value less than min. support value (i.e. 2) are eliminated

Apriori Algorithm – 2nd Iteration

TID	Items
T1	1 3 4
T2	2 3 5
T3	1 2 3 5
T4	2 5
T5	1 3 5

→

Only Items present in F1

C2

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

→

F2

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

Item sets with support value less than min. support value (i.e. 2) are eliminated

Apriori Algorithm – Pruning

TID	Items
T1	1 3 4
T2	2 3 5
T3	1 2 3 5
T4	2 5
T5	1 3 5

C3

Itemset	In F2?
{1,2,3}, {1,2}, {1,3}, {2,3}	NO
{1,2,5}, {1,2}, {1,5}, {2,5}	NO
{1,3,5},{1,5}, {1,3}, {3,5}	YES
{2,3,5}, {2,3}, {2,5}, {3,5}	YES

Apriori Algorithm – Pruning

TID	Items
T1	1 3 4
T2	2 3 5
T3	1 2 3 5
T4	2 5
T5	1 3 5

F3

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

If any of the subsets of these item sets are not there in F2 then we remove that itemset

Apriori Algorithm – 4^{th} Iteration

TID	Items
T1	1 3 4
T2	2 3 5
T3	1 2 3 5
T4	2 5
T5	1 3 5

F3

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

C3

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

Apriori Algorithm – Subset Creation

F3

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

For I = {1,3,5}, subsets are {1,3}, {1,5}, {3,5}, {1}, {3}, {5}

For I = {2,3,5}, subsets are {2,3}, {2,5}, {3,5}, {2}, {3}, {5}

- For every subsets S of I, output the rule:

S → **(I-S)** (S recommends I-S)

if ***support(I)/support(S) >= min_conf value***

Apriori Algorithm – Applying Rules

Applying Rules to Item set F3

1. **{1,3,5}**

   ✓ Rule 1: **{1,3} → ({1,3,5} - {1,3})** means 1 & 3 → 5
   Confidence = support(1,3,5)/support(1,3) = 2/3 = 66.66% > 60%
   Rule 1 is selected

   ✓ Rule 2: **{1,5} → ({1,3,5} - {1,5})** means 1 & 5 → 3
   Confidence = support(1,3,5)/support(1,5) = 2/2 = 100% > 60%
   Rule 2 is selected

   ✓ Rule 3: **{3,5} → ({1,3,5} - {3,5})** means 3 & 5 → 1
   Confidence = support(1,3,5)/support(3,5) = 2/3 = 66.66% > 60%
   Rule 3 is selected

Apriori Algorithm – Applying Rules

Applying Rules to Item set F3

1. **{1,3,5}**

✓ Rule 4: **{1}** → **({1,3,5} - {1})** means 1 → 3 & 5
Confidence = support(1,3,5)/support(1) = 2/3 = 66.66% > 60%
Rule 4 is selected

✓ Rule 5: **{3}** → **({1,3,5} - {3})** means 3 → 1 & 5
Confidence = support(1,3,5)/support(3) = 2/4 = 50% <60%
Rule 5 is rejected

✓ Rule 6: **{5}** → **({1,3,5} - {5})** means 5 → 1 & 3
Confidence = support(1,3,5)/support(3) = 2/4 = 50% < 60%
Rule 6 is rejected

Apriori Workflow (Example 2)

$Sup_{min} = 2$

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

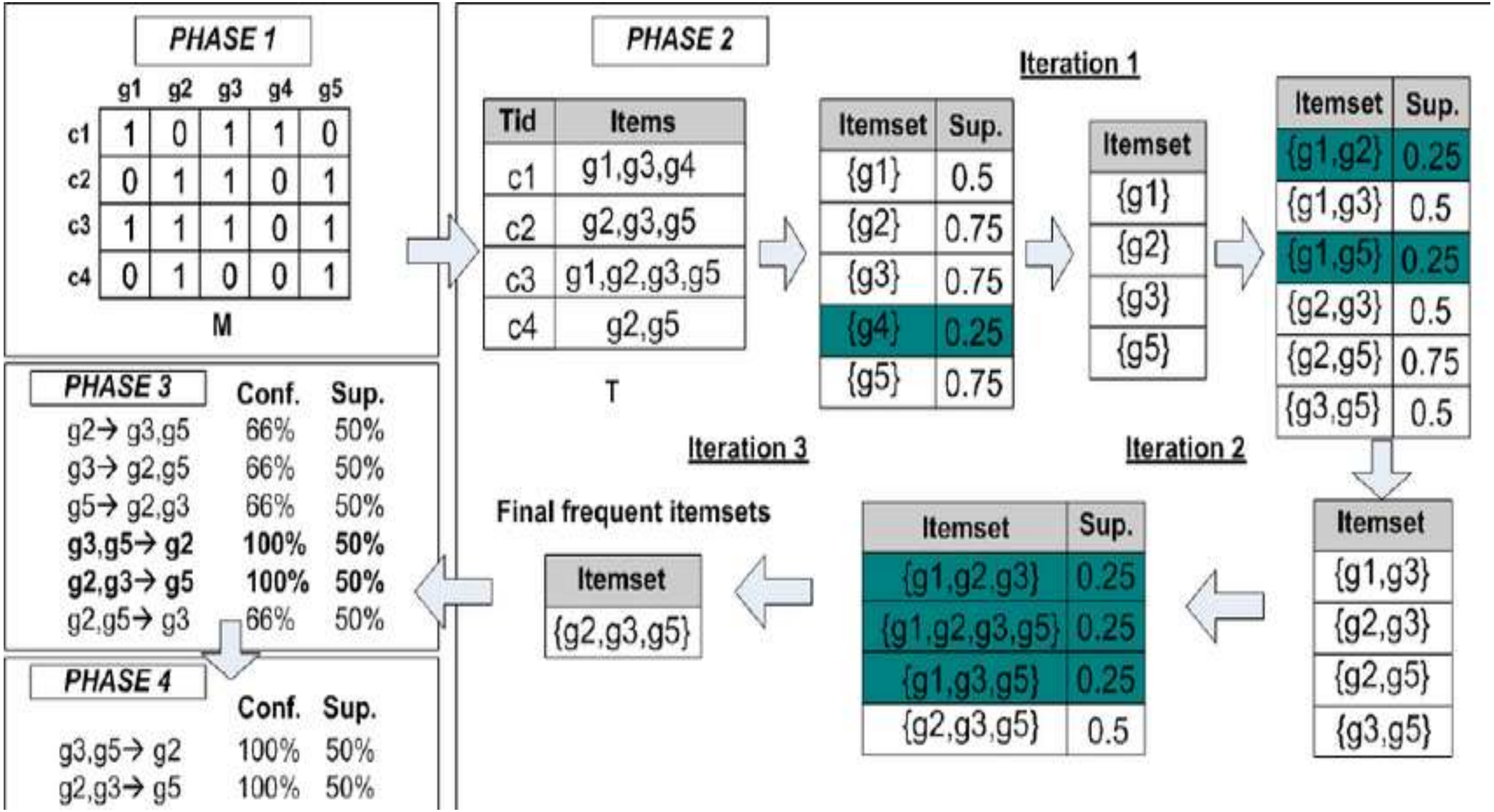
Itemset
{B, C, E}

3rd scan

L_3

Itemset	sup
{B, C, E}	2

Apriori Workflow (Example 3)



Evaluation

- Execution time
- Memory used
- Scalability

Apriori performance

The performance of Apriori depends on several factors:

- The minsup parameter: the more it is set low, the larger the search space and the number of itemsets will be.
- The number of items
- The number of transactions
- The average transaction length.

Apriori Problems

- Can generate numerous candidates.
- Require to scan the database numerous times.
- Candidates may not exist in the database.

Demo 1.

Apply Apriori Algorithm using SPMF tool.

Demo 2.

Apply Apriori Algorithm using Python.

Thank You!!