APRIORI ALGORITHM

Association Rule Mining

Association Rule Mining

- Given a set of transactions, find combinations of items (itemsets) that occur frequently

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

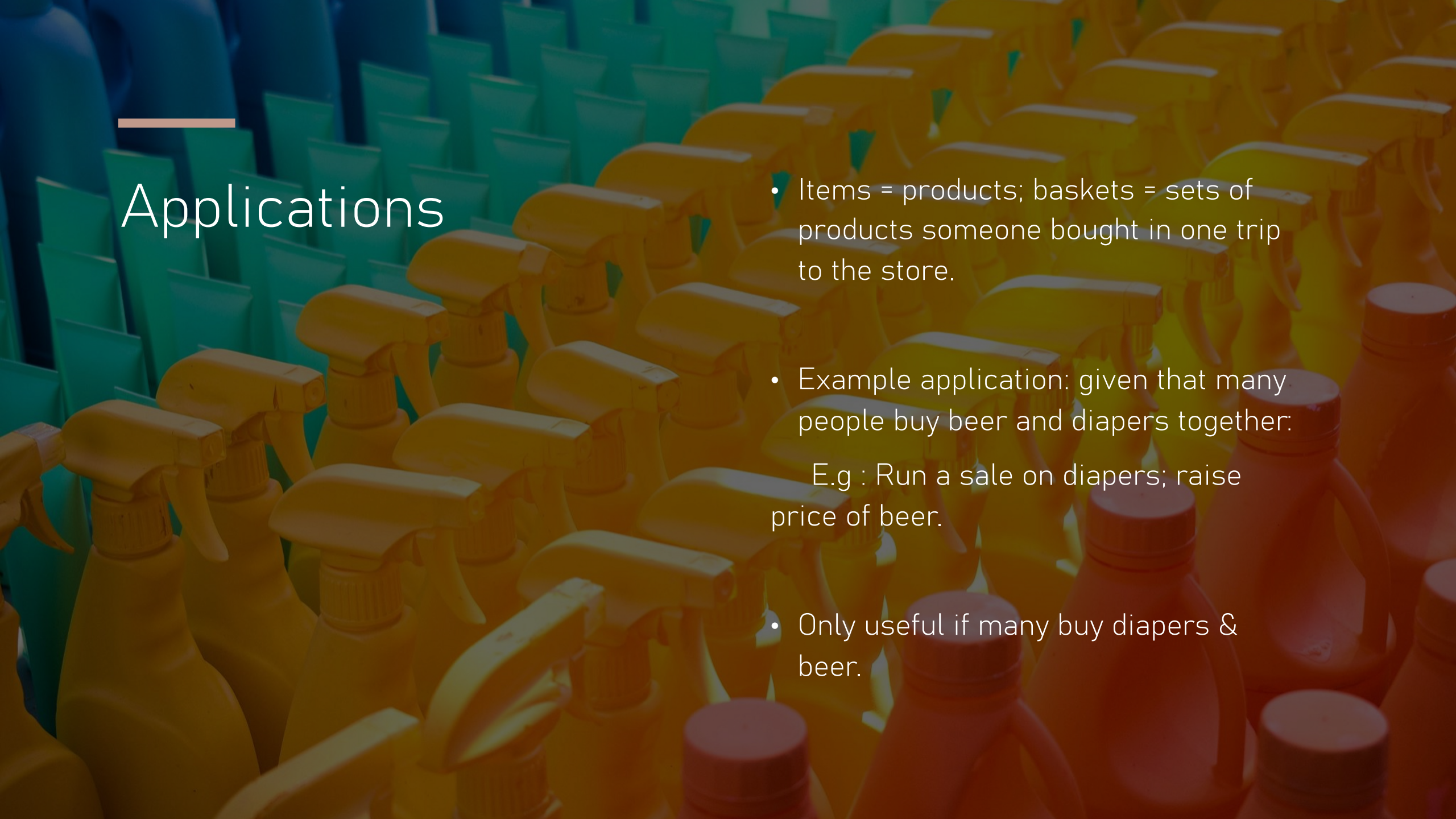
Example of Association Rules

{Diaper} → {Beer},

{Milk, Bread} → {Eggs, Coke},

{Beer, Bread} → {Milk}

Implication means co-occurrence, not causality!



Applications

- Items = products; baskets = sets of products someone bought in one trip to the store.

- Example application: given that many people buy beer and diapers together:

  E.g : Run a sale on diapers; raise price of beer.

- Only useful if many buy diapers & beer.

Applications

- Baskets = sentences; Items = documents containing those sentences
- Items that appear together too often could represent plagiarism

Applications

Baskets = patients; Items = drugs/medicine & side-effects

Has been used to detect combinations of drugs that result in particular side-effects

But requires extension: Absence of an item needs to be observed as well as presence

Definition: Frequent Itemset

Itemset

- A collection of one or more items
  Example: {Milk, Bread, Diaper}
- k-itemset
  An itemset that contains k items

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 count (σ)

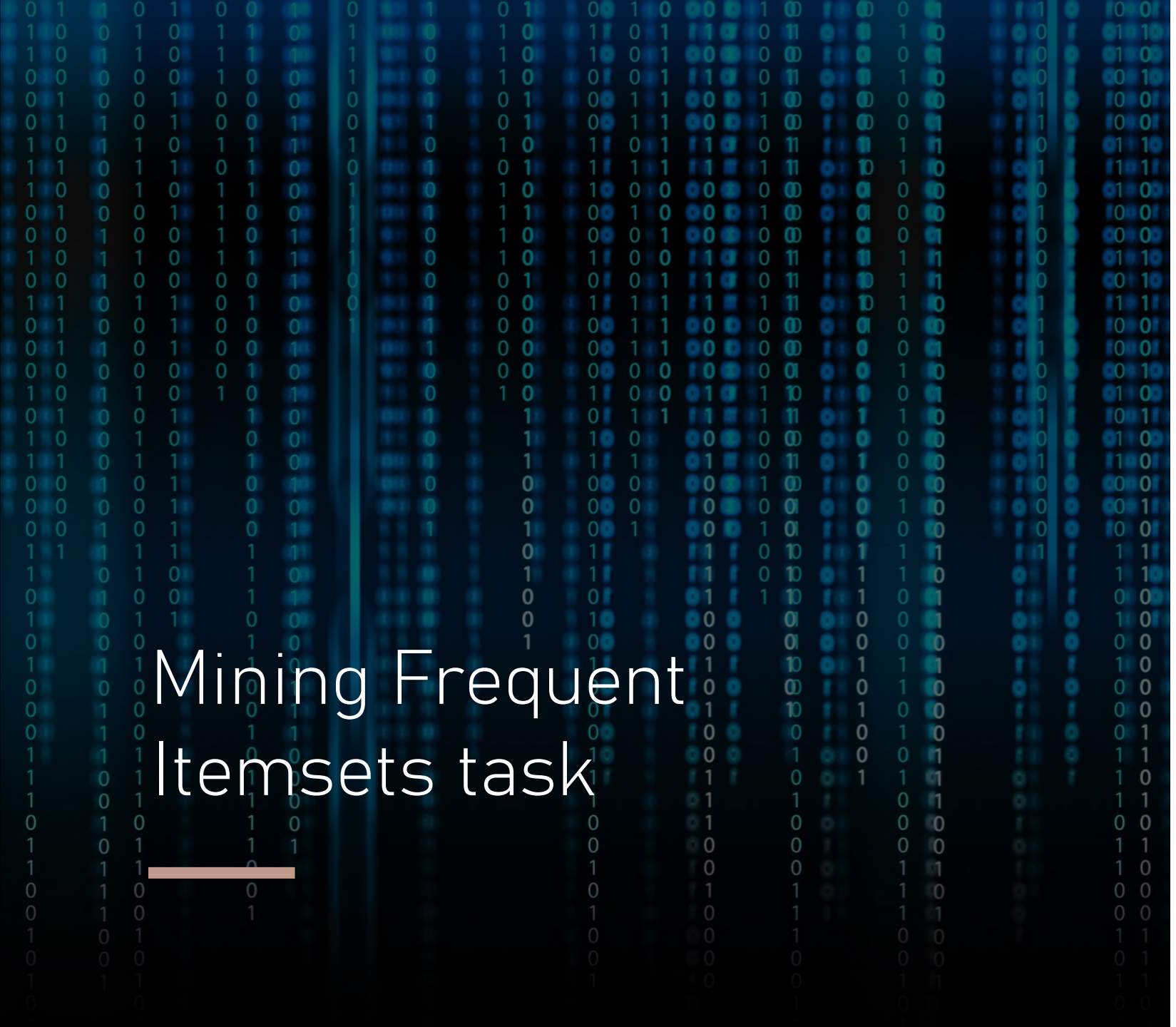
- Frequency of occurrence of an itemset
- E.g., σ({Milk, Bread, Diaper}) = 2

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



Mining Frequent Itemsets task

Input: A set of transactions T, over a set of items I

Output: All itemsets with items in I having support ≥ minsup threshold

Problem parameters:

- N = |T|: number of transactions
- d = |I|: number of (distinct) items
- w: max width of a transaction
- Number of possible itemsets?

Scale of the problem:

- WalMart sells 100,000 items and can store billions of baskets.
- The Web has billions of words and many billions of pages.

Definition: Association Rule

- Let D be database of transactions
- e.g.:

Transaction ID	Items Bought
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

- Let I be the set of items that appear in the database, e.g., I={A,B,C,D,E,F}
- A rule is defined by $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$
- e.g.: {B,C} ⇒ {E} is a rule

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

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

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

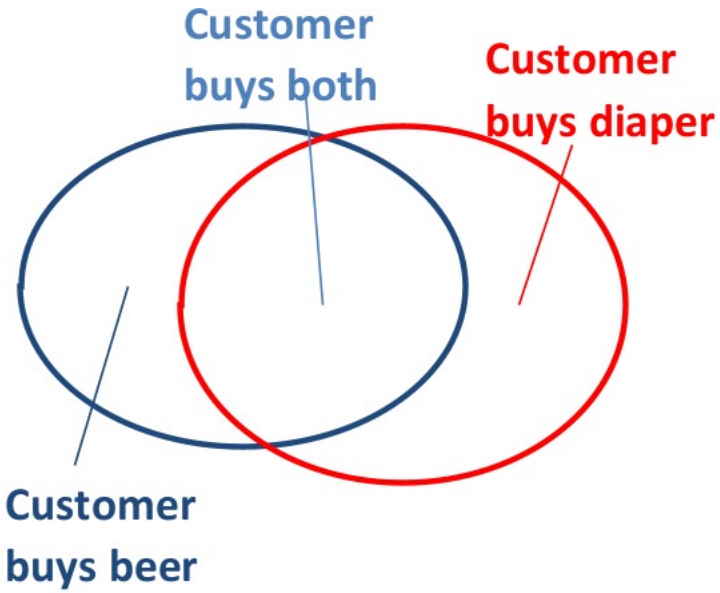
$$c = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{\sigma(\text{Milk}, \text{Diaper})} = \frac{2}{3} = 0.67$$

Rule Measures: Support and Confidence

Find all the rules X ⇒ Y with minimum confidence and support

- support, s, probability that a transaction contains {X ∪ Y}

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



Let minimum support 50%, and minimum confidence 50%, we have

A ⇒ C (50%, 66.6%)

C ⇒ A (50%, 100%)

Transaction ID	Items Bought
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

Example

TID	date	items_bought
100	10/10/99	{F,A,D,B}
200	15/10/99	{D,A,C,E,B}
300	19/10/99	{C,A,B,E}
400	20/10/99	{B,A,D}

Remember:

$$\text{conf}(X \Rightarrow Y) = \frac{\sup(X \cup Y)}{\sup(X)}$$

What is the support and confidence of the rule: {B,D} ⇒ {A}

Support:

- Percentage of tuples that contain {A,B,D} = 75%

Confidence:

$$\frac{\text{number of tuples that contain } \{A, B, D\}}{\text{number of tuples that contain } \{B, D\}} = 100\%$$

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!

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

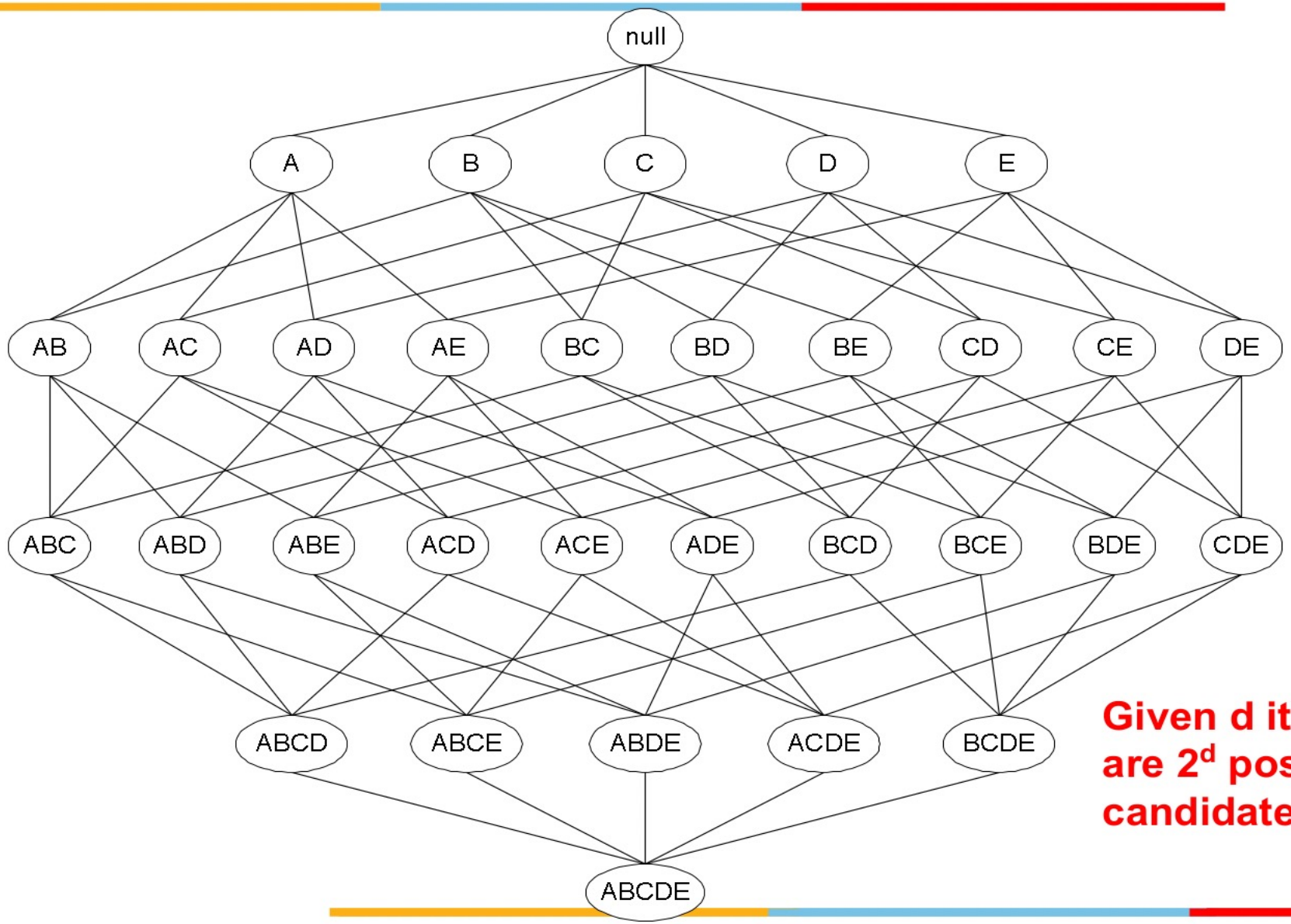
- Two-step approach:

➢*Frequent Itemset Generation:* Generate all itemsets whose support ≥ minsup

➢*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

When is the task sensible and feasible?

- If minsup = 0, then all subsets of I will be frequent and thus the size of the collection will be very large

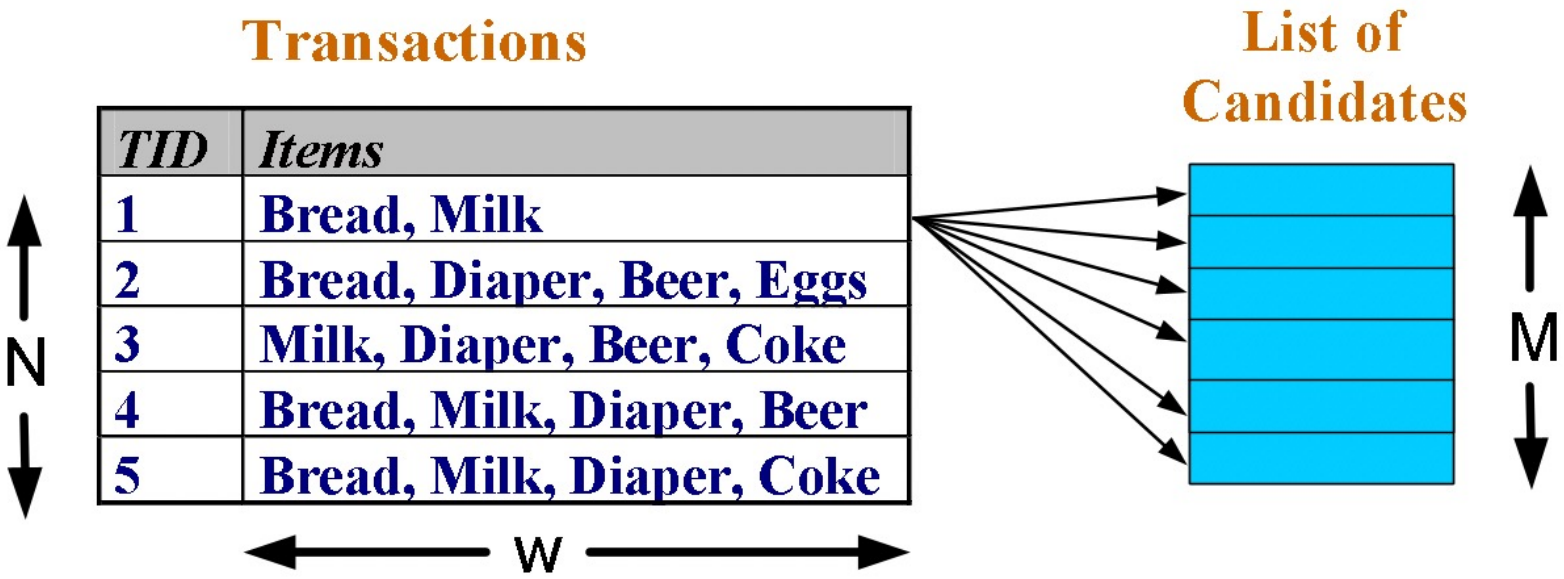
- This summary is very large (maybe larger than the original input) and thus not interesting

- The task of finding all frequent sets is interesting typically only for relatively large values of minsup

Brute-force algorithm for finding all frequent itemsets?

- Generate all possible itemsets (lattice of itemsets)
  - Start with 1-itemsets, 2-itemsets,…,d-itemsets
- Compute the frequency of each itemset from the data
  - Count in how many transactions each itemset occurs
- If the support of an itemset is above minsup report it as a frequent itemset.

Brute-force approach for finding all frequent itemsets



Complexity?

- Match every candidate against each transaction
- For M candidates and N transactions, the complexity is~ O(MNw) => Expensive since M = 2d !!!

Apriori principle

Apriori principle (Main observation):

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

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

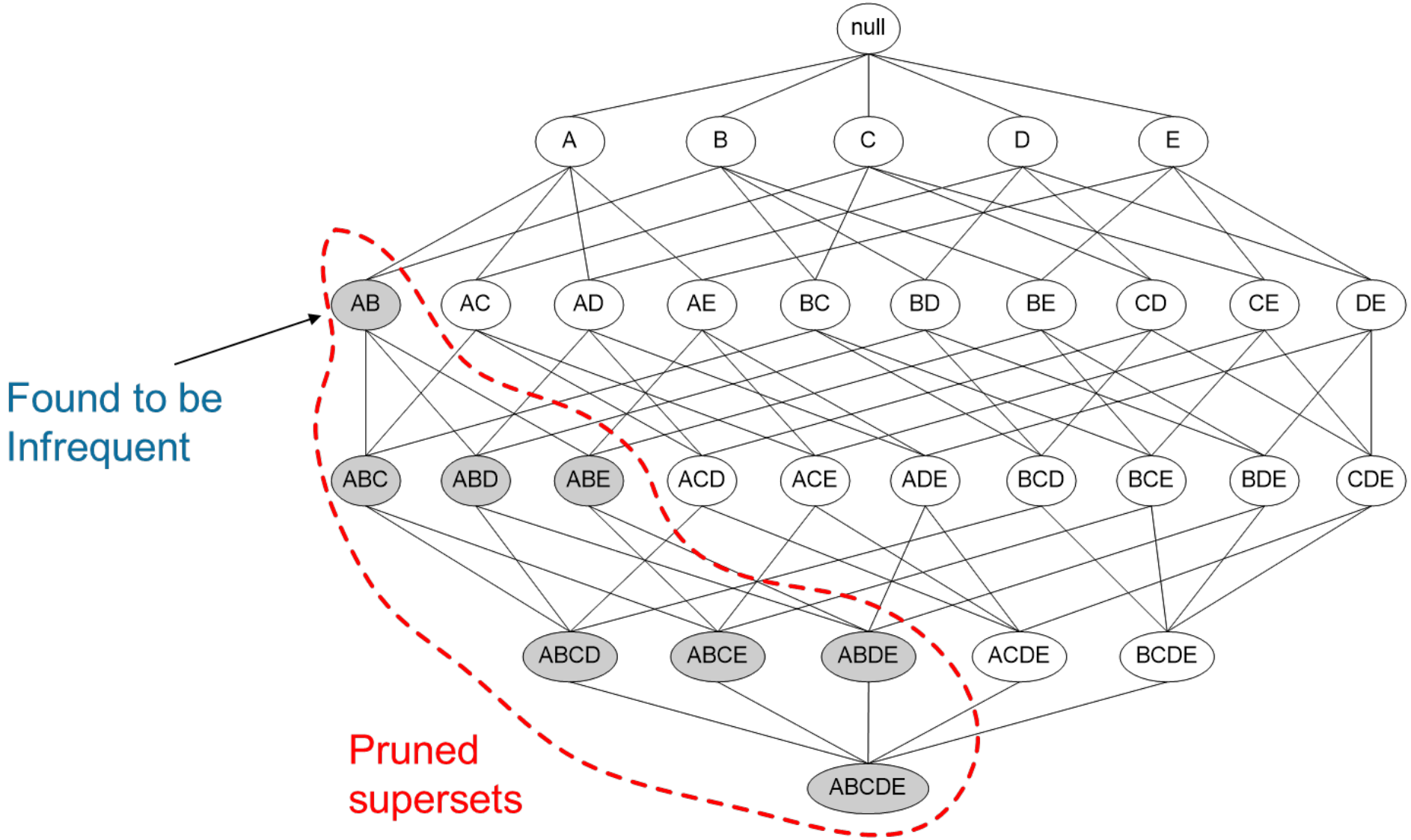
- The support of an itemset never exceeds the support of its subsets
- This is known as the anti-monotone property of support acting on the subsets of the itemsets.

Example

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

s(Bread) > s(Bread, Beer)
s(Milk) > s(Bread, Milk)
s(Diaper, Beer) > s(Diaper, Beer, Coke)

Illustrating the Apriori principle



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
    - If {AB} is a frequent itemset, both {A} and {B} should be frequent itemsets
  - Iteratively find frequent itemsets with cardinality from 1 to m (m-itemset): Use frequent k-itemsets to explore (k+1)-itemsets.

- Use the frequent itemsets to generate association rules.

Illustrating the Apriori principle

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

Items (1-itemsets)

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

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

minsup = 3/5

Triplets (3-itemsets)

Itemset	Count
{Bread,Milk,Diaper}	3

If every subset is considered,

$${}^6C_1 + {}^6C_2 + {}^6C_3 = 41$$

With support-based pruning,

$$6 + 6 + 1 = 13$$

Exploiting the Apriori principle

1. Find frequent 1-items and put them to L_k (k=1)

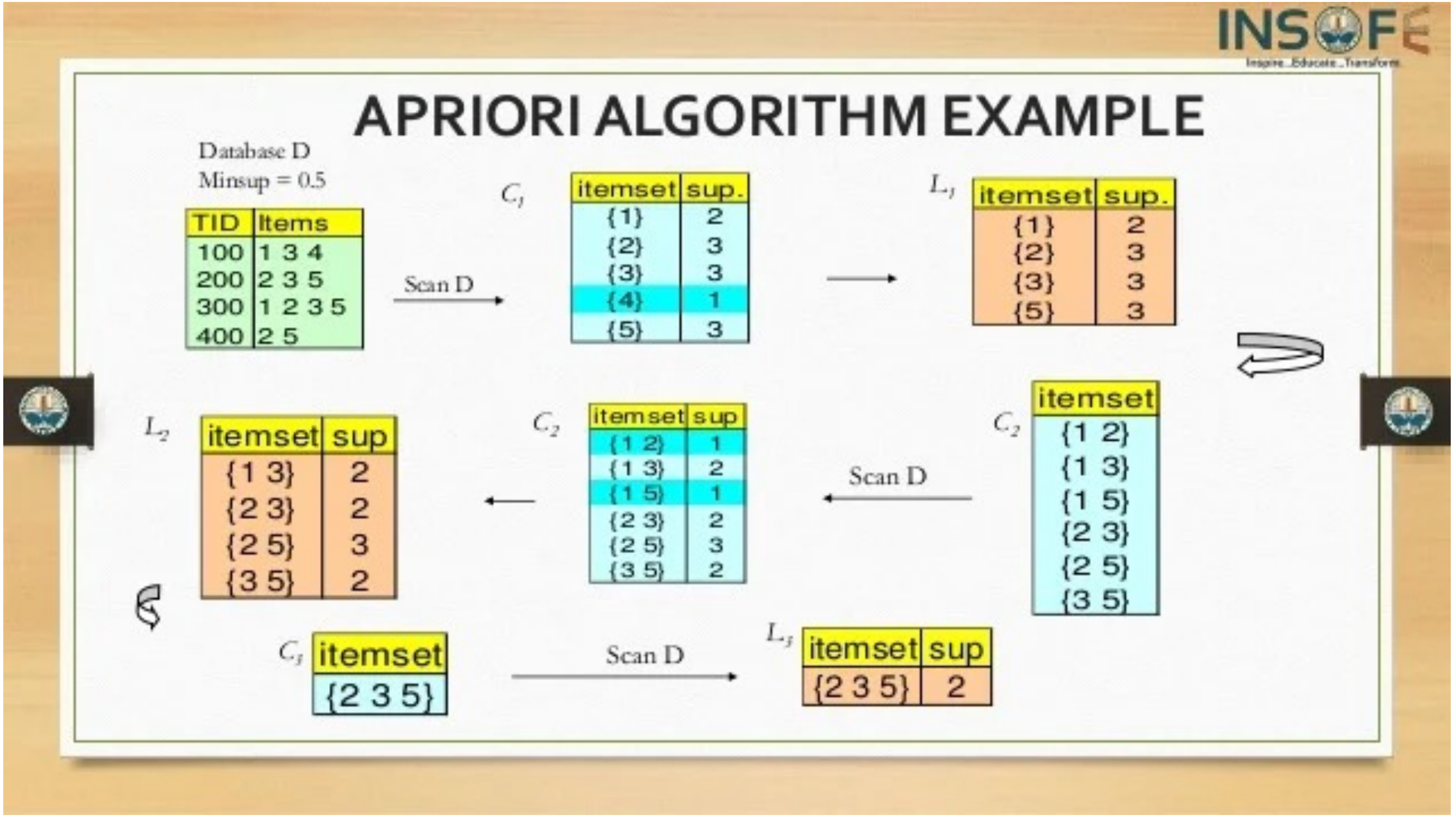
2. Use L_k to generate a collection of candidate itemsets C_{k+1} with size (k+1)

3. Scan the database to find which itemsets in C_{k+1} are frequent and put them into L_{k+1}

4. If L_{k+1} is not empty

- k=k+1
- GOTO 2

The Apriori Algorithm — Example



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;
    // join and prune steps
  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 (frequent)
  end
return $\cup_k L_k$;

Important steps in candidate generation:

- Join Step: C_{k+1} is generated by joining L_k with itself
- Prune Step: Any k-itemset that is not frequent cannot be a subset of a frequent (k+1)-itemset

Rule Generation

- Given a frequent K-itemset, Y how many association rule can it produce?
  - 2^k-2 ignoring empty rules in antecedents and consequents

- An association rule can be extracted by partitioning the itemset Y into two non-empty subsets, X and Y-X such that X -> Y-X

satisfies the confidence threshold.

- If X is 3-frequent itemset{1,2,3} what are the rules it can generate?

- Computing confidence does require you to compute support.

Rule Generation

- Given a frequent itemset L, find all non-empty subsets $f \subset L$ such that $f \rightarrow L - f$ 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 $L \rightarrow \emptyset$ and $\emptyset \rightarrow L$)

Rule Generation

- How to efficiently generate rules from frequent itemsets?

- 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., $L = \{A,B,C,D\}$: $c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$

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

Rule generation using Apriori

- Apriori uses a level wise approach for rule generation, where each level corresponds to the number of items that belong to the rule consequent.

- Initially all the high-confidence rules that have only one item in the rule consequent are extracted which are used to generate subsequent rules.

Rule Generation for Apriori Algorithm