BITS Pilani

BITS Pilani
Hyderabad Campus

Prof.Aruna Malapati
Department of CSIS

innovate achieve lead

BITS Pilani
Hyderabad Campus

Association Rule Mining

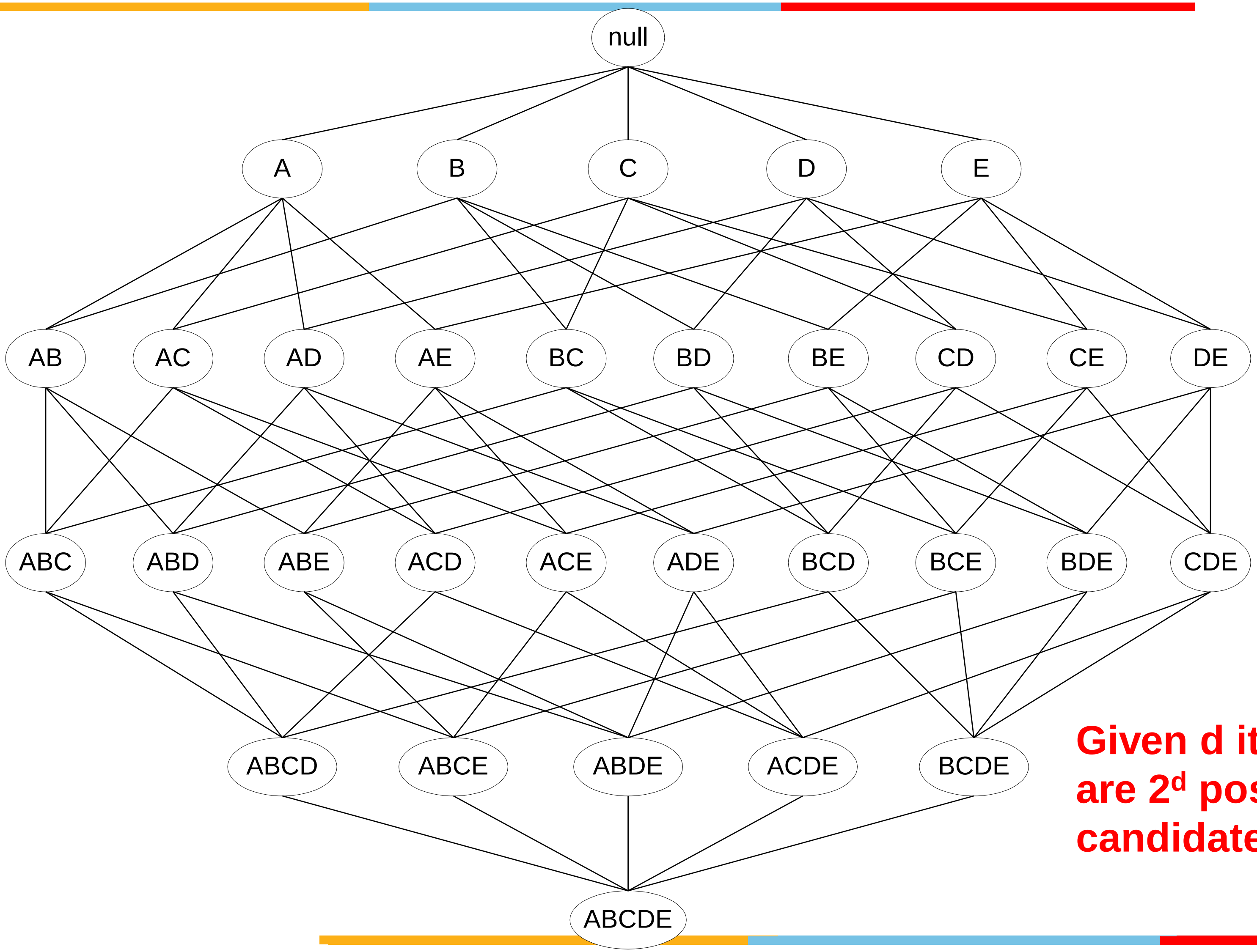
Today's Learning objective

- **Brute force algorithm for generating frequent item sets and its runtime complexity**
- **Use of support as a anti-monotonic property for reducing the runtime complexity**
- **Apriori algorithm for generating frequent item sets using support**

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

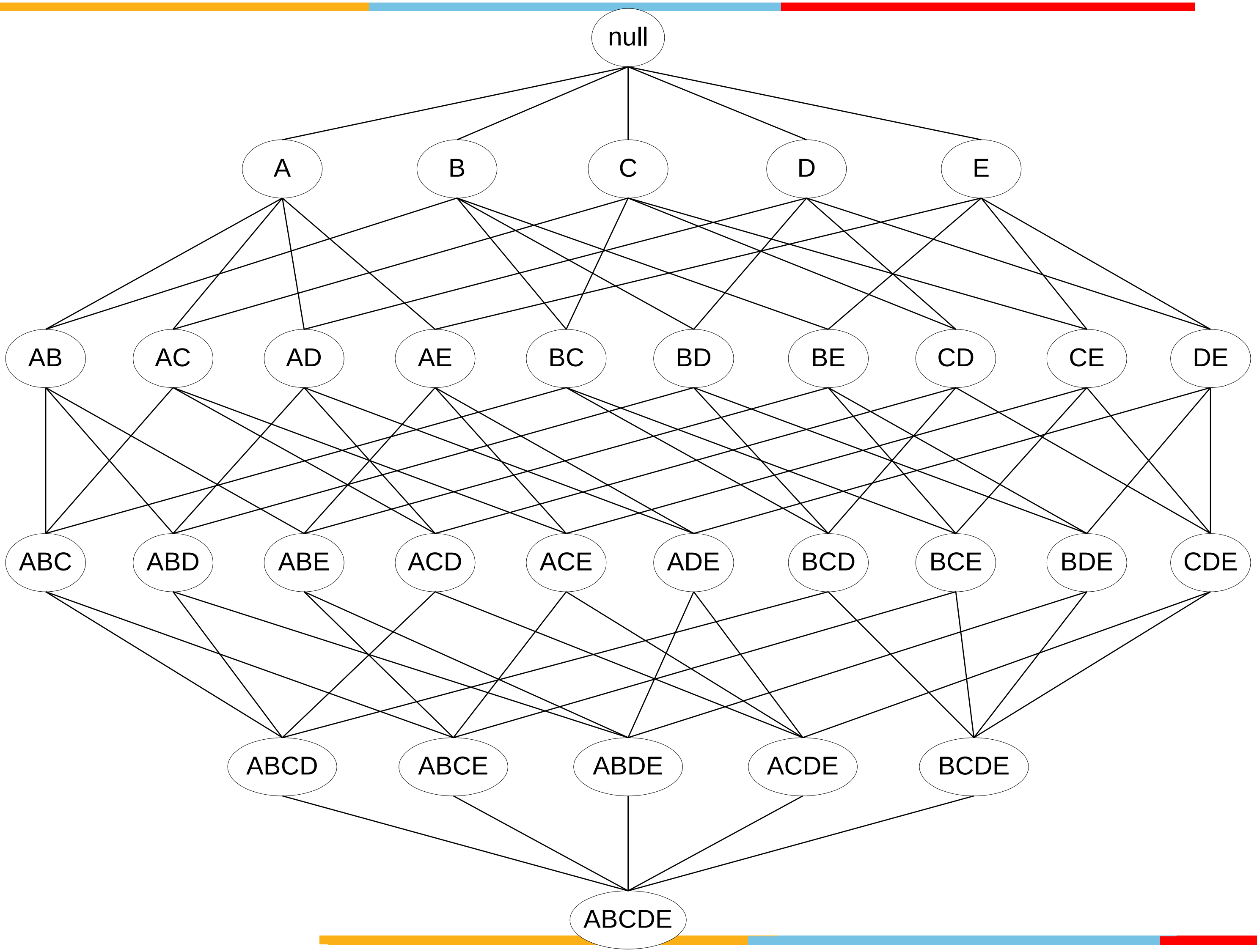


Given d items, there are 2^d possible candidate itemsets

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**

A simple algorithm for finding all frequent itemsets ??



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

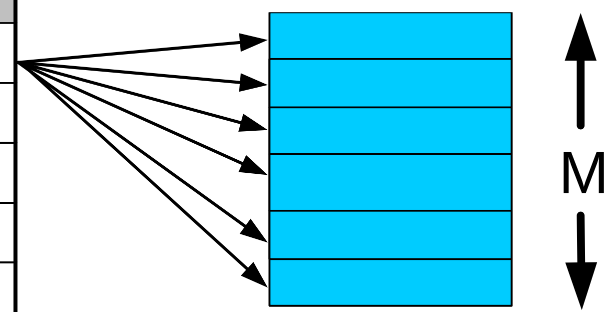
Transactions

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

N

w

List of Candidates



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

Speeding-up the brute-force algorithm

- 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
  - Use vertical-partitioning of the data to apply the 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

Reduce the number of candidates

- Apriori principle (Main observation):
  - 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:

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

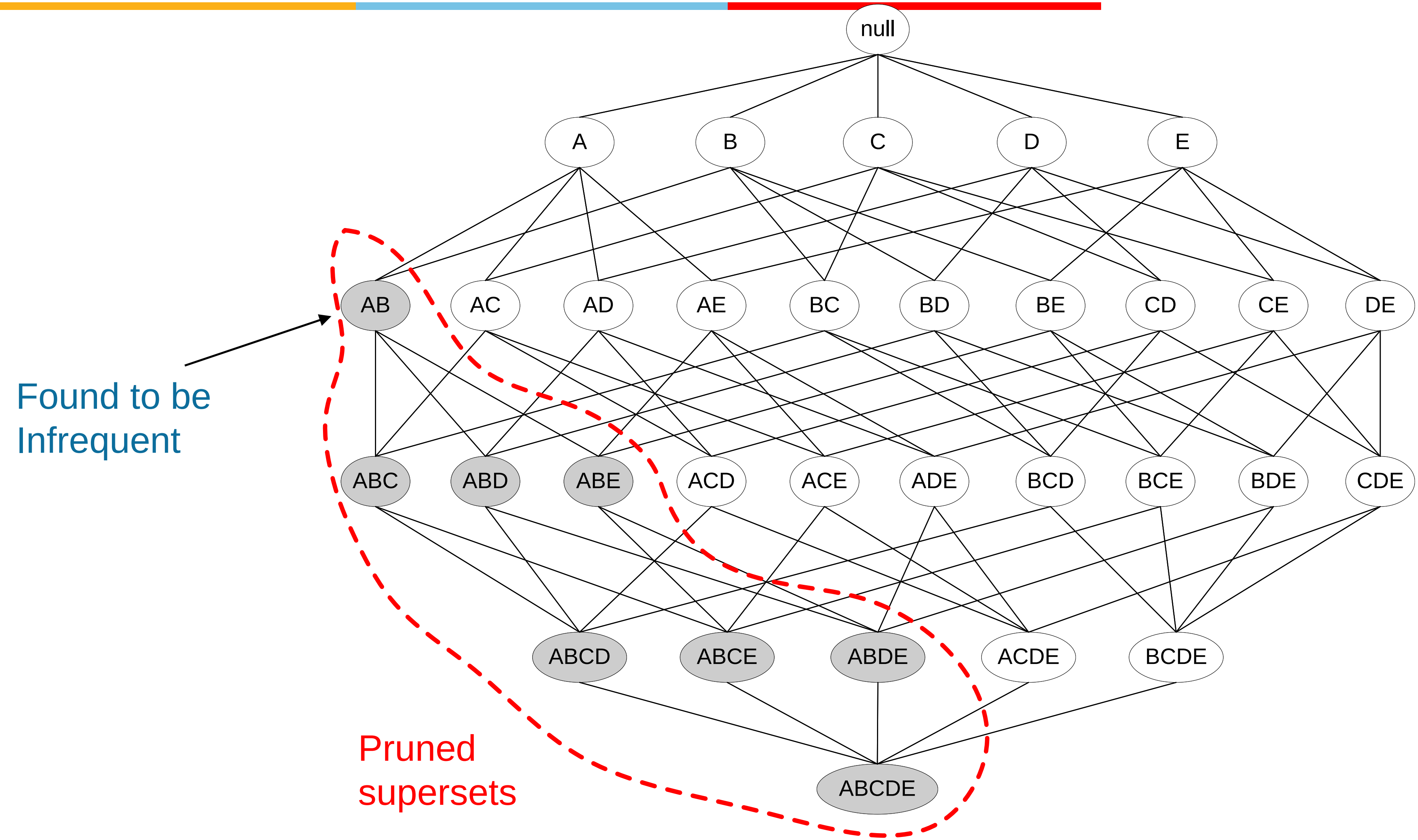
  - The support of an itemset ***never exceeds*** the support of its subsets
  - This is known as the ***anti-monotone*** property of support acting on the subsets of the itemsets.

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 using Hasse diagram



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
    - i.e., 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

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**

Triplets (3-itemsets)

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

minsup = 3/5

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

R. Agrawal, R. Srikant: "Fast Algorithms for Mining Association Rules", *Proc. of the 20th Int'l Conference on Very Large Databases*, 1994.

The Apriori Algorithm — Example

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

min_sup=2=50%

Scan D → C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

→ L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

Scan D → C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

→ L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_3

itemset
{2 3 5}

Scan D → L_3

itemset	sup
{2 3 5}	2

Join and prune steps

1. Scan D for a count of each candidate
C1={1:2,2:3,3:3,4:1,5:3}

Find 1-itemsets that are frequent L1={1:2,2:3,3:3,5:3}

2. Generate C2 candidates from L1*L1 and scan D for count of each candidate
C2={{1,2}:1,{1,3}:2,{1,5}:1,{2,3}:2,{2,5}:3,{3,5}:2}

Find 2-itemsets that are frequent
L2={{1,3}:2,{2,3}:2,{2,5}:3,{3,5}:2}

Join and prune steps (Contd..)

3. Generate C3 candidates from L2 using the join and prune steps: C3={{1 2 3} ,{1 3 5}, {2 3 5}}

prune: {{1 2 3} ,{1 3 5}}

C3={2 3 5}:2

L3: {2 3 5}

Steps for strong Association Rule Generation

- Generate all nonempty subsets for each frequent itemset
- For every nonempty subset S of Itemset I , output of the rule:
  - S --> (I - S)
  - **If** support_count (I) / support_count (S) > = minimum confidence threshold **then** rule is a **strong Association Rule.**

Generate the strong Association rule from frequent itemsets

We obtained the set of all frequent itemsets

L = { {1} {2} {3} {5} {1 3} {2 3} {2 5} {3 5} {2 3 5} }

Suppose we take I = {2 3 5}

Its all nonempty subsets are S = {2} {3} {5} {2 3} {2 5} {3 5}

Rule 1: {2} -> {3 5}

support=2/4, confidence=support {2 3 5}/support{2} = 2/3>=50%

Since the minsup and minconf >=50% this is an interesting rule

Other rules that could be generated for this 3-frequent Itemset are {3} -> {2 5} , {5}->{2 3}, {2 3} ->{5},{2 5} -> {3},{3,5}->{2}

Exercise

TID	List of Items
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

min_sup=2=50%
Min_conf=50%

1-Itemset

{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Generating 2-Itemset L1 = {I1,I2,I3,I4,I5}

Since L2 = L1 join L1 then {I1,I2,I3,I4,I5} join {I1,I2,I3,I4,I5}.

C2= [{I1,I2} {I1,I3}, {I1,I4}, {I1,I5}, {I2,I3}, {I2,I4}, {I2,I5}, {I3,I4} {I3,I5}, {I4,I4}].

Now we need to check the frequent itemsets with min support count.

Then we get -> (C2*C2) L2= [{I1,I2} {I1,I3}, {I1,I5}, {I2,I3}, {I2,I4}, {I2,I5}].

Exercise (Contd..)

3-Itemset Generation

L2= [{I1,I2} {I1,I3}, {I1,I5}, {I2,I3}, {I2,I4}, {I2,I5}]

L3 = L2 JOIN L2 i.e.

- For example, lets take {I1, I2, I3}. The 2-item subsets of it are {I1, I2}, {I1, I3} & {I2, I3}. Since all 2-item subsets of {I1, I2, I3} are members of L2, We will keep {I1, I2, I3} in C3.
- Let's take another example of {I2, I3, I5}, which shows how the pruning is performed. The 2-item subsets are {I2, I3}, {I2, I5} & {I3,I5}.
- BUT, {I3, I5} is not a member of L2 and hence it is not frequent violating Apriori Property. Thus We will have to remove {I2, I3, I5} from C3.
- C3 = {{I1, I2, I3}, {I1, I2, I5}}

Association Rule Generation

Suppose I = {I1, I2, I5}

The nonempty subsets of I are S: {I1}, {I2} {I5} {I1, I2}, {I1, I5}, {I2, I5},.

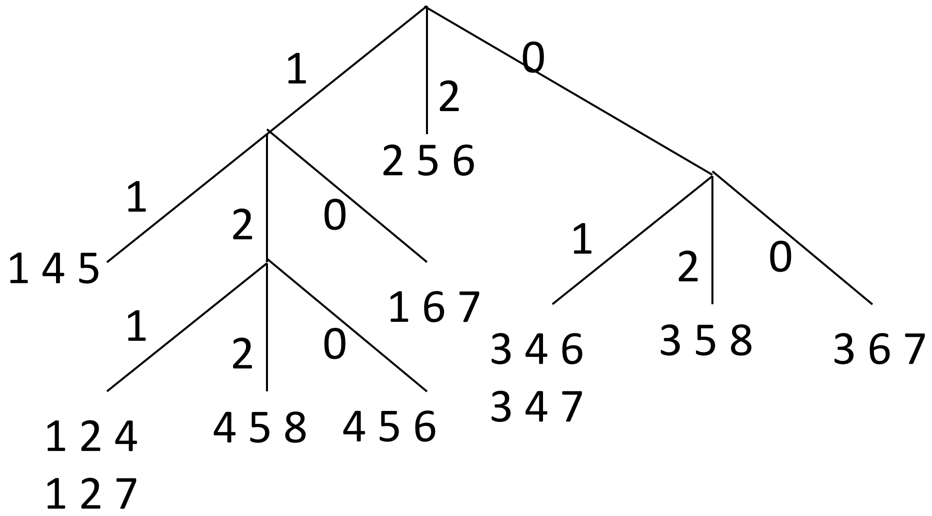
I1 -> {I2 I5}	Conf=2/6=33% Uninteresting rule sinceConf<=50%
I2 -> {I1 I5}	Conf=2/7=29% Uninteresting rule sinceConf<=50%
I5 -> {I1 I2}	Conf=2/2=100% Interesting rule since Conf>=50%
{I1 I2} -> I5	Conf=2/4=50% Interesting rule since Conf>=50%
{I1 I5} -> I2	Conf=2/2=100% Interesting rule since Conf>=50%
{I2 I5} -> I1	Conf=2/2=100% Interesting rule since Conf>=50%

The Apriori Hash Tree

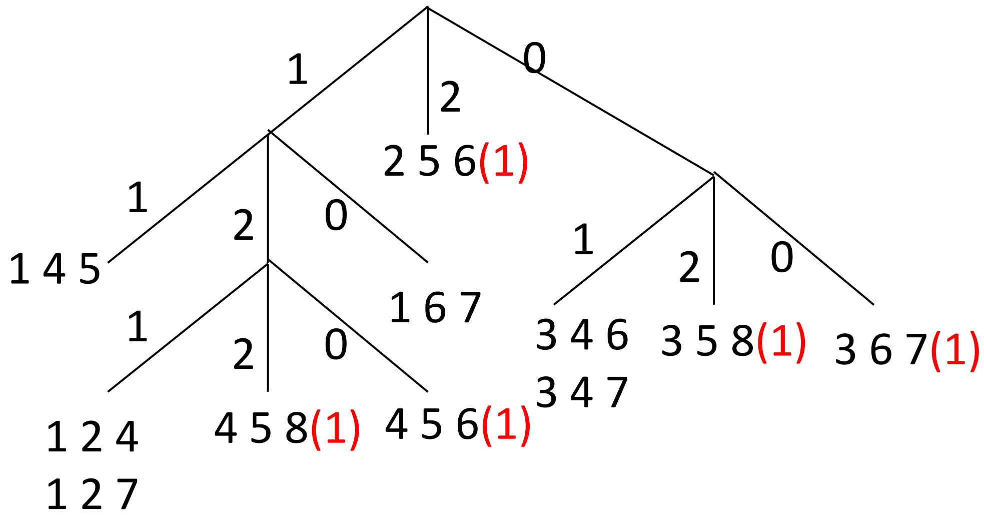
Suppose for any dataset

C3={{1 2 4},{1 2 7},{1 4 5},{1 6 7},{2 5 6},{3 4 6},{3 4 7}, {3 5 8}, {3 6 7},{4 5 6},{4 5 8},{4 5 9}}

Hash function used is X_i mod 3, Threshold =3



Support counting for Hash Tree



Suppose your transactional database is

T1={1 3 5 8}

T2={3 6 7}

T3={3 7}

T4={2 4 5 6 8}

Take home message

- Association rule mining is traditionally called Market Basket analysis.
- Support and confidence are used to find interesting rules.
- Generating Association Rules is a combinatorial problem and hence need heretics.