

8/1/19

UNIT - 5

Association Analysis

Basic concepts:-

* Many Business enterprises collect large quantities of data from their day to day operations.

* Market - Basket Transactions.

Tid	Items
1.	Bread, Milk
2.	Bread, Diaper, Beer, Eggs
3.	Milk, Diapers, Beer, coffee
4.	Bread, Milk, Diaper, Beer
5.	Bread, Milk, Diaper, coffee

The above table is an example of Market-Basket transaction, each row in this table corresponds to a transaction with TID and a set of items purchased by a given customer.

Association analysis which is useful for discovering interesting relationships hidden in large dataset.

The uncovered relationships can be represented in the form of association rules (or) a set of a frequent item sets.

For example the following rules can be extracted from the given dataset.

{Diaper} $\rightarrow$ {Beer}

{Milk, Bread} $\rightarrow$ {Eggs, coffee}

{Beer, Bread} $\rightarrow$ {Milk}

problem Definition:-

* This Describes basic terminology used in association analysis and presents a description of the task.

1) Binary Representation:-

* Market - Basket data can be represented in a binary format where each row corresponds to a transaction and each column corresponds to an item.

* An item can be treated as a binary variable, whose value is 1 if the item is present in a transaction otherwise "0".

TID	Bread	Milk	Diaper	Beer	Eggs	Coke.
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	1

Item set:-

* Let $I = \{i_1, i_2, \dots, i_d\}$ be the set of items in a market basket data and $T = \{t_1, t_2, \dots, t_n\}$ be the set of all transaction

* Each transaction T_i contains a subset of items chosen from I .

* A collection of one or more items is called an Itemset.

* An Itemset that contains k items. is called k -itemset.

eg:- $\{Milk, Bread, Diaper\}$

Support Count:-

* This is property of an item set, which refers to the no. of transactions that contain a particular itemset.

* Mathematically, the support count for $\sigma(x)$ can be stated as follows.

$$\sigma(x) = | \{ t_i \mid x \subseteq t_i, t_i \in T \} |$$

eg:- support count for $\sigma(\{Milk, Bread, Diaper\}) = 2$.

$$\sigma(\{Bread, Diaper\}) = 3$$

$$\sigma(\{Milk, Beer, cone\}) = 1$$

Association Rule:-

* An association rule is an implication expression of the form

$X \rightarrow Y$, where X & Y are disjoint itemsets.

i.e. $x \cap y = \phi$

* The strength of this rule can be measured in terms of its support and confidence.

Support:- Fraction of transactions that contain an itemset.

Confidence:- Determines, how frequently items in Y appear in a transaction that contain X .

$$\text{Support} = s(x, y) = \frac{\sigma(x \cup y)}{N}$$

$$\text{Confidence} = c(x, y) = \frac{\sigma(x \cup y)}{\sigma(x)}$$

eg:- consider the rule $\{Milk, Diaper\} \Rightarrow \{Beer\}$

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

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

01/11/19

Association Rule Discovery

* Given a set of transactions T , the goal of association rule mining is to find all rules having

— support $\geq$ minsup threshold

— confidence $\geq$ minconf threshold

→ A Brute force approach for mining association rules is to

* compute the support and confidence for every possible rule.

* The total no. of possible rules that can be extracted from a given dataset that contains 'd' items is

$$R = 3^d - 2^{d+1} + 1$$

* For the dataset shown in above table, this approach requires us to compute the support and confidence for 602 rules.

$$3^6 - 2^7 + 1 = 602$$

* More than 80% of rules are discarded after applying

minsup = 20%

minconf = 50%

* A common method adopted by many association rule mining algorithms is to decompose the problem into two major sub

tasks: 1. Frequent itemset Generation

- Generate all itemsets whose support $\geq \text{minsup}$

2. Rule Generation

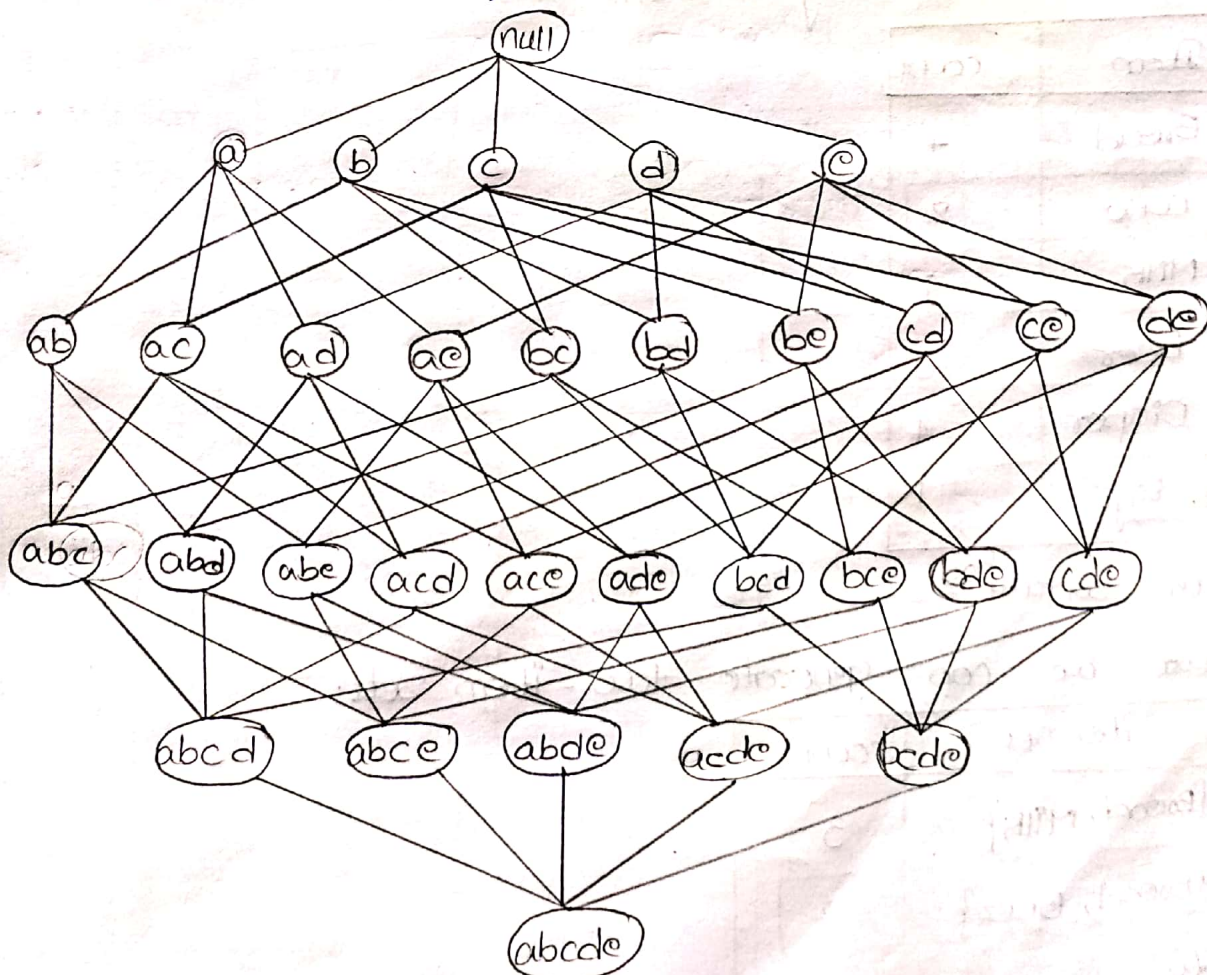
- Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset.

Frequent Itemset Generation

* A lattice structure can be used to enumerate all possible itemsets, the following figure shows an item set lattice for

$$I = \{a, b, c, d, e\}$$

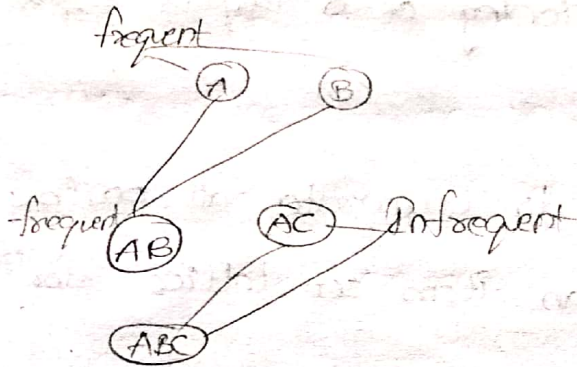
* In General a dataset that contains k -items can potentially generate upto $2^k - 1$ frequent item sets, ~~excluding~~ excluding the "null set".



A priori principle:-

If an itemset is frequent, then all of its subsets must also be frequent, conversely if an itemset is infrequent then all of its supersets must be infrequent.

From the above lattice structure if 'AB' is infrequent then subsets A, B are frequent.



Frequent itemset Generation using Apriori Algorithm:-

In this step Generating 1-item sets.

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

Minimum Support = 3

Then next we can generate two-item sets.

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

*{Milk, Beer}	2
{Milk, Diaper}	3
{Beer, Diaper}	3

Next Three item sets are generated

Itemset	count
{Bread, Milk, Diaper}	3

stand(*) indicates no support because those are removed.

If every no subset is considered,

$$6C_1 + 6C_2 + 6C_3 = 41$$

With support-based pruning

$$6 + 6 + 1 = 13.$$

eg:2

TID	Items
100	ACD
200	BCE
300	ABC E
400	BE

A database has five Transactions. Let the min support = 50% and min confidence = 80%.

1-itemsets	support
{A}	2
{B}	3
{C}	3
{E}	3

item	support
{A B}	1
{AC}	2
{AE}	1
{BC}	2
{BE}	3
{CE}	2

2-itemsets	support
{AC}	2
{BC}	2
{BE}	3
{CE}	2

Itemset	support
{BCE}	2

Solution:-

Step 1 Find all frequent itemsets

Frequent itemset:

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

24/1/19

FP - Growth Algorithm:-

* It is an alternative algorithm called FP - Growth is used to discover frequent itemsets.

* It encodes the dataset using a compact data structure called FP-tree and extracts frequent item sets directly from this structure.

FP-tree Representation:-

- * An FP-tree is a compressed representation of input data
- * It is constructed by reading the transaction one at a time and mapping each transaction on to a path in FP-Tree.
- * As different transactions can have several items in common, their paths may overlapped

Tid	Items
1	A, B
2	B, C, D
3	A, C, D, E
4	A, D, E
5	A, B, C
6	A, B, C, D
7	A
8	A, B, C
9	A, B, D
10	B, C, E

A: 8

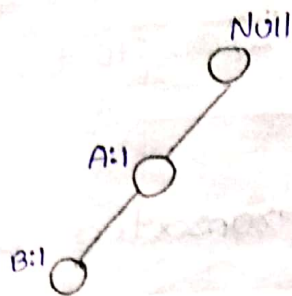
B: 7

C: 6

D: 5

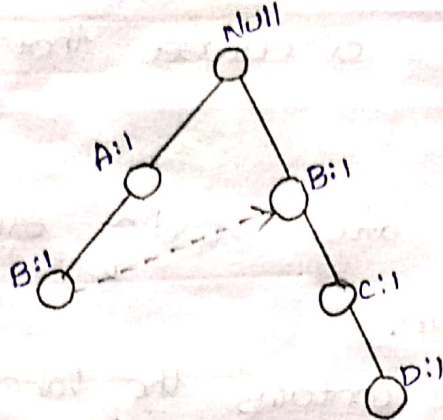
E: 3

7



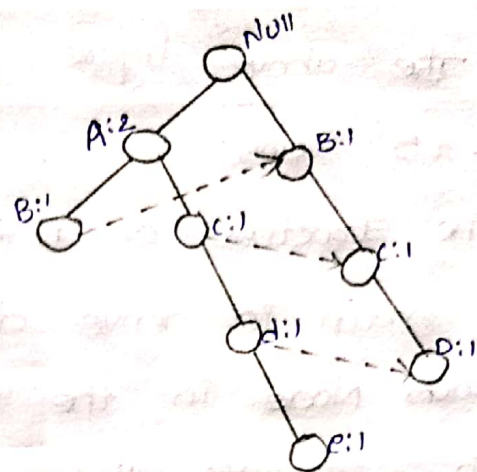
i) after Reading

TID = 1



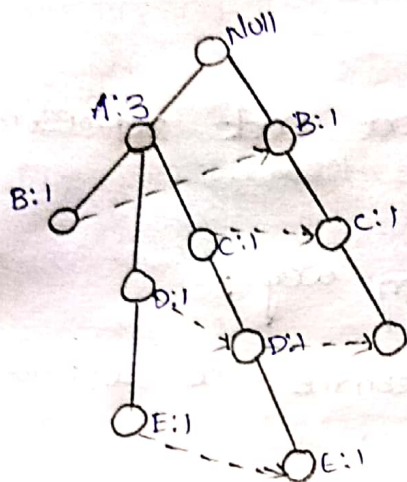
ii) after Reading

TID = 2



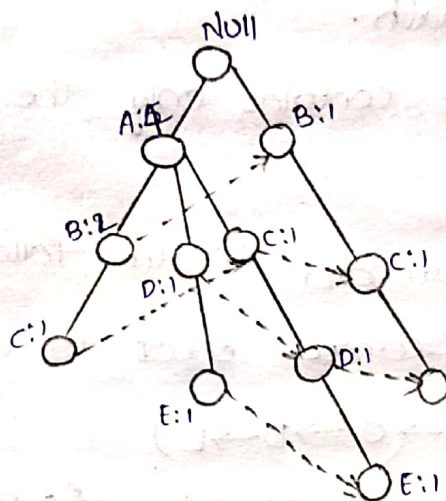
iii) After Reading

TID = 3



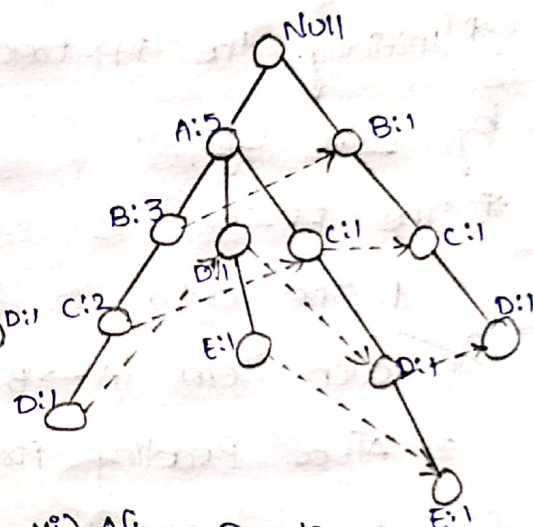
iv) After Reading

TID = 4



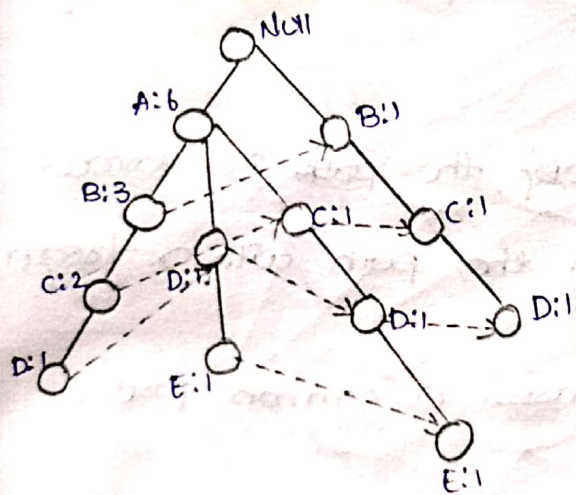
v) After Reading

TID = 5



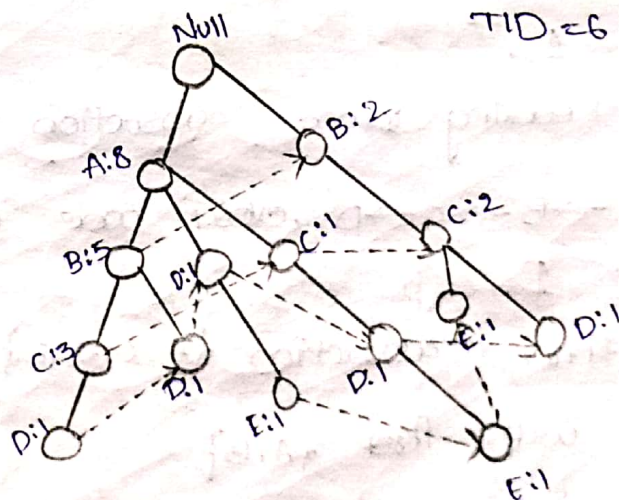
vi) After Reading

TID = 6



vii) After Reading

TID = 7



A: 8
B: 7
C: 6
D: 5
E: 3

* The above figure shows a dataset that contains 10 transactions and 5 items

* The structures of FP-Tree after reading every Transaction is also shown in above diagram.

* Each Node in the tree contains the label of an item along with a counter that shows the number of transactions mapped on to the given path.

* Initially the FP-tree contains only the root node represented by NULL symbol.

* The FP-Tree is extended in the following way :

1. The data is scanned once to determine the support count for each item ($A \rightarrow B \rightarrow C \rightarrow D \rightarrow E$)

2. After Reading First Transaction $\{A, B\}$, the path is formed from 'null $\rightarrow A \rightarrow B$ ', every node along the path has a frequency count of 1.

3. After Reading second transaction $\{B, C, D\}$, the path is formed from 'null $\rightarrow B \rightarrow C \rightarrow D$ ', every node along the path with a frequency count of 1.

4. The third transaction $\{A, C, D, E\}$ shares a common prefix item 'A' with first $\{A, B\}$

As a result the path 'Null $\rightarrow A \rightarrow C \rightarrow D \rightarrow E$ ' overlaps with 'Null $\rightarrow A \rightarrow B$ ', because of their overlapping path frequency of

$$A = 2$$

5. this process continues until every transaction has been mapped on to one of the paths in given FP-Tree.

Frequent Itemset Generation in FP-Growth Algorithm:-

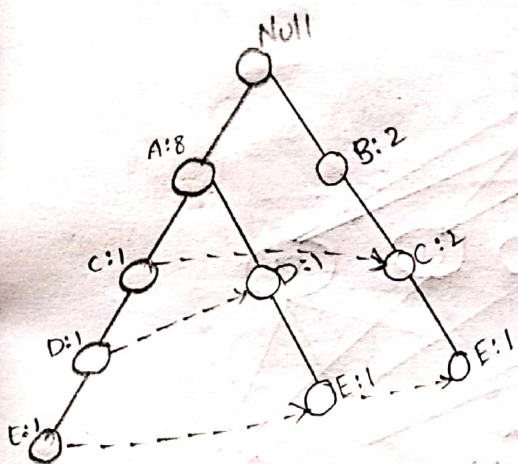
* FP-Growth is an algorithm that generates frequent itemsets from an FP-Tree by exploring the tree in a bottom-up fashion.

* This bottom-up strategy for finding frequent itemsets from ending with a particular item is equal to suffix based approach.

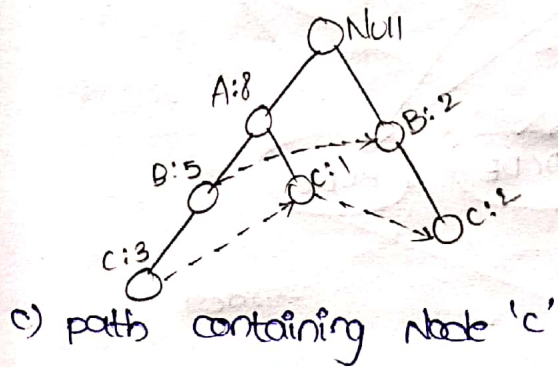
* Every transaction is mapped on to a path in FP-Tree, we can derive the frequent itemsets ending with a particular

Item Say 'e'.

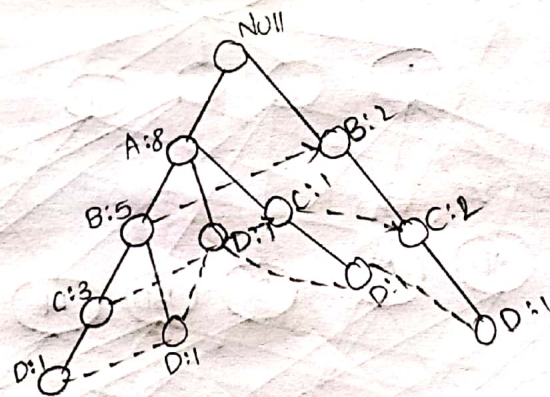
* Similarly the paths are extracted as shown below for d, b, c, a.



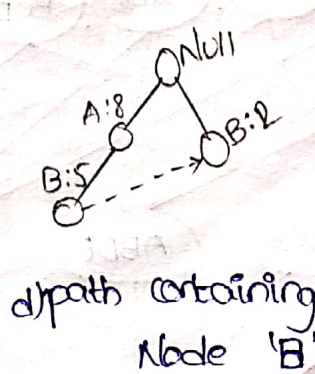
a) path containing Node 'e'



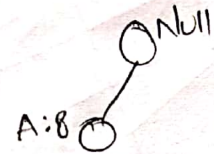
c) path containing Node 'c'



b) path containing Node 'd'



d) path containing Node 'b'



e) path containing Node 'a'

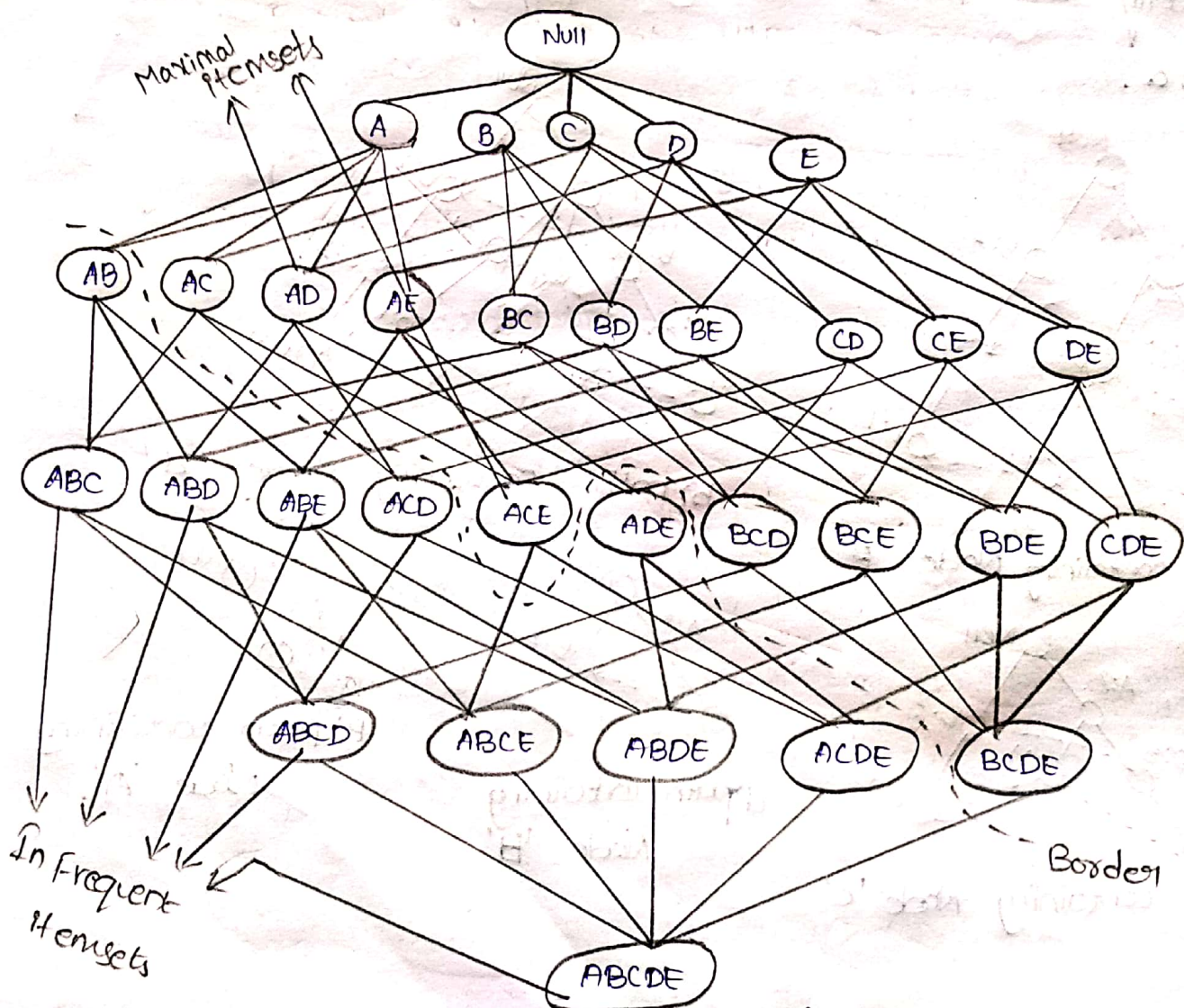
Suffix	Frequent Itemsets
E	$\{E\}, \{DE\}, \{CD, E\}, \{A, B, D, E\}, \{A, D, E\}, \{C, E\}, \{B, C, E\}$
D	$\{D\}, \{C, D\}, \{B, C, D\}, \{A, B, C, D\}, \{B, D\}, \{A, B, D\}, \{A, C, D\}, \{A, D\}$
C	$\{C\}, \{B, C\}, \{A, C\}, \{A, B, C\}$
B	$\{B\}, \{A, B\}$
A	$\{A\}$

7/2/19 ** Compact Representation of Frequent Itemsets:-

- * It has two types they are 1) Maximal frequent itemset
2) closed itemset

Maximal frequent Itemset:-

An itemset is maximal frequent if none of its immediate supersets is frequent.



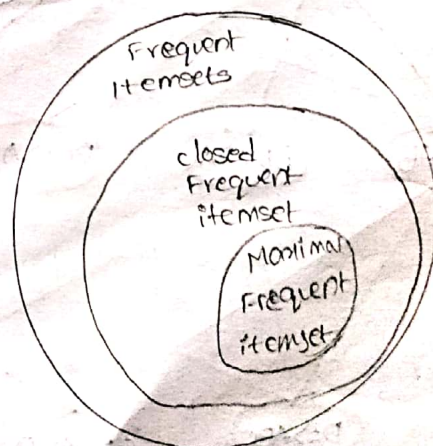
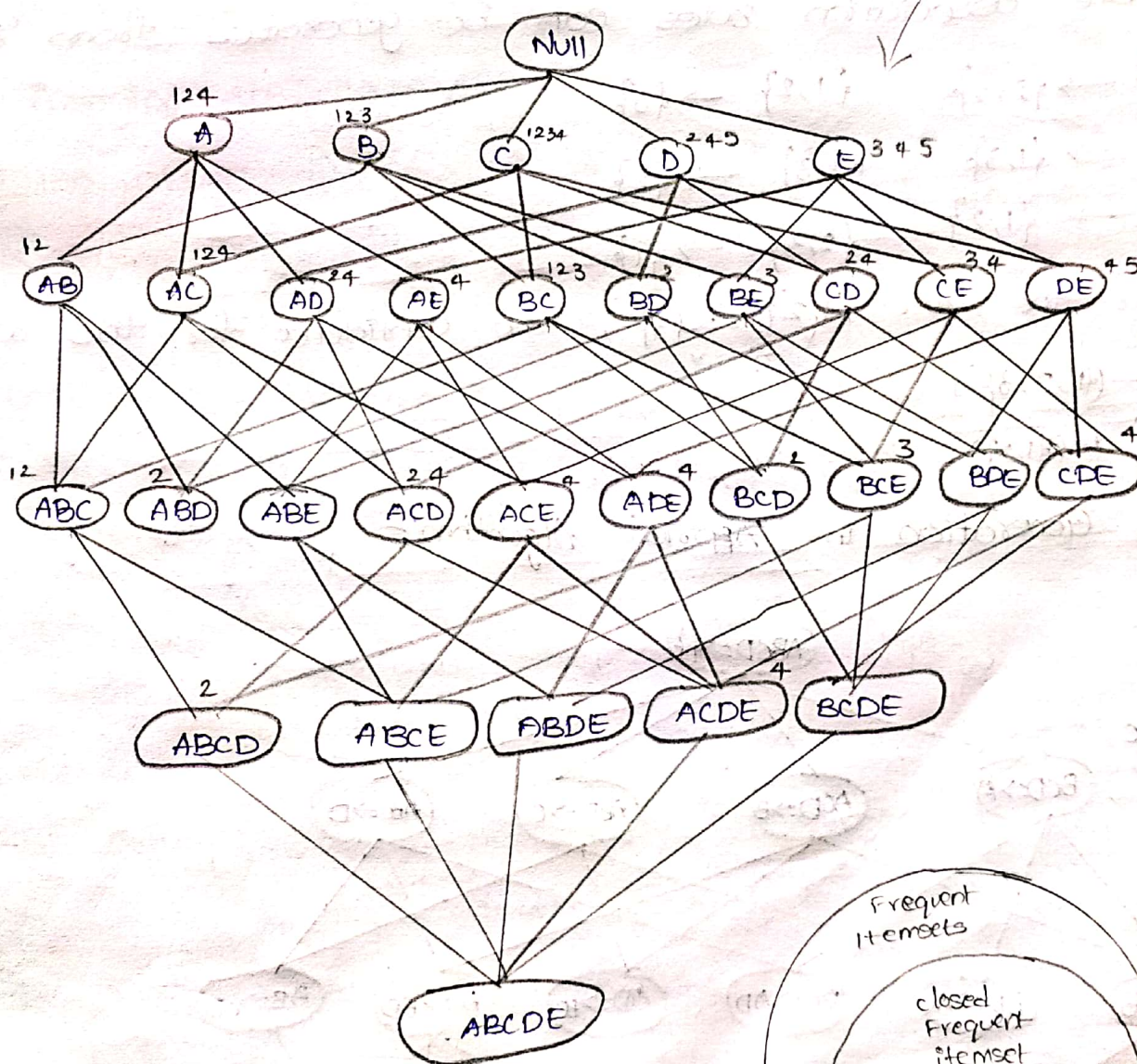
Below the Border are infrequent
Above the Border are frequent

closed itemset

* An itemset is closed if none of its immediate supersets has the same support as the itemset.

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

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



Rule Generation:-

* It describes how to abstract association rules efficiently from a given frequent itemset.

* Each frequent k -itemset, X can produce upto $2^k - 2$ association rules, ignoring rules that have an empty antecedents, (or) consequents.

$$(\emptyset \rightarrow Y \text{ (or) } Y \rightarrow \emptyset)$$

eg:- let $X = \{1, 2, 3\}$ be a frequent itemset, there are six candidate association rules can be generated from 'X'.

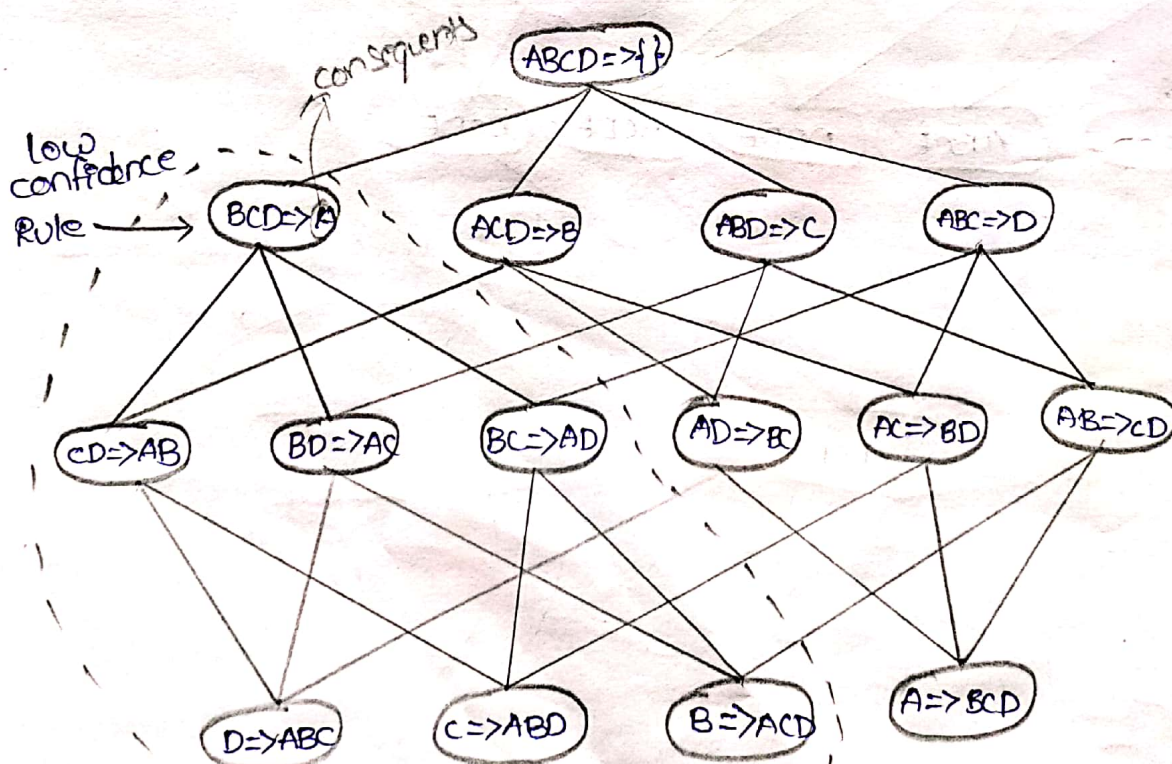
$$\begin{array}{ll} \{1\} \rightarrow \{2, 3\} & \{1, 2\} \rightarrow \{3\} \\ \{2\} \rightarrow \{1, 3\} & \{1, 3\} \rightarrow \{2\} \\ \{3\} \rightarrow \{1, 2\} & \{2, 3\} \rightarrow \{1\} \end{array}$$

consider the rule $\{1, 2\} \rightarrow \{3\}$, the confidence for this rule

$$\text{is : } \frac{\sigma(\{1, 2, 3\})}{\sigma(\{1, 2\})}$$

$$\text{confidence} = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

Rule Generation in Apriori Algorithm:- pruning - removing



pruned rules

lattice of rules.

* The apriori algorithm uses level-wise approach for generating association rules, where each level corresponds to the no. of items that belongs to the rule consequent.

* Initially all high confidence rules that have only one item in rule consequent are extracted.

* These rules are used to generate new candidate rules.

* For example if $\{ACD\} \rightarrow \{B\}$ and $\{ABD\} \rightarrow \{C\}$ are high confidence rules then a candidate rule $\{AD\} \rightarrow \{BC\}$ is generated by merging the consequents of both rules.

* The above figure shows a lattice structure to generate association rules from the frequent itemset $\{ABCD\}$.

* If any rule having low confidence rule then entire subsequent can be pruned immediately.

* Suppose the confidence for $\{BCD\} \rightarrow \{A\}$ is low then item A in its consequents can be discarded.

Support counting:-

Support counting is the process of determining the frequency of occurrence for every candidate itemset.

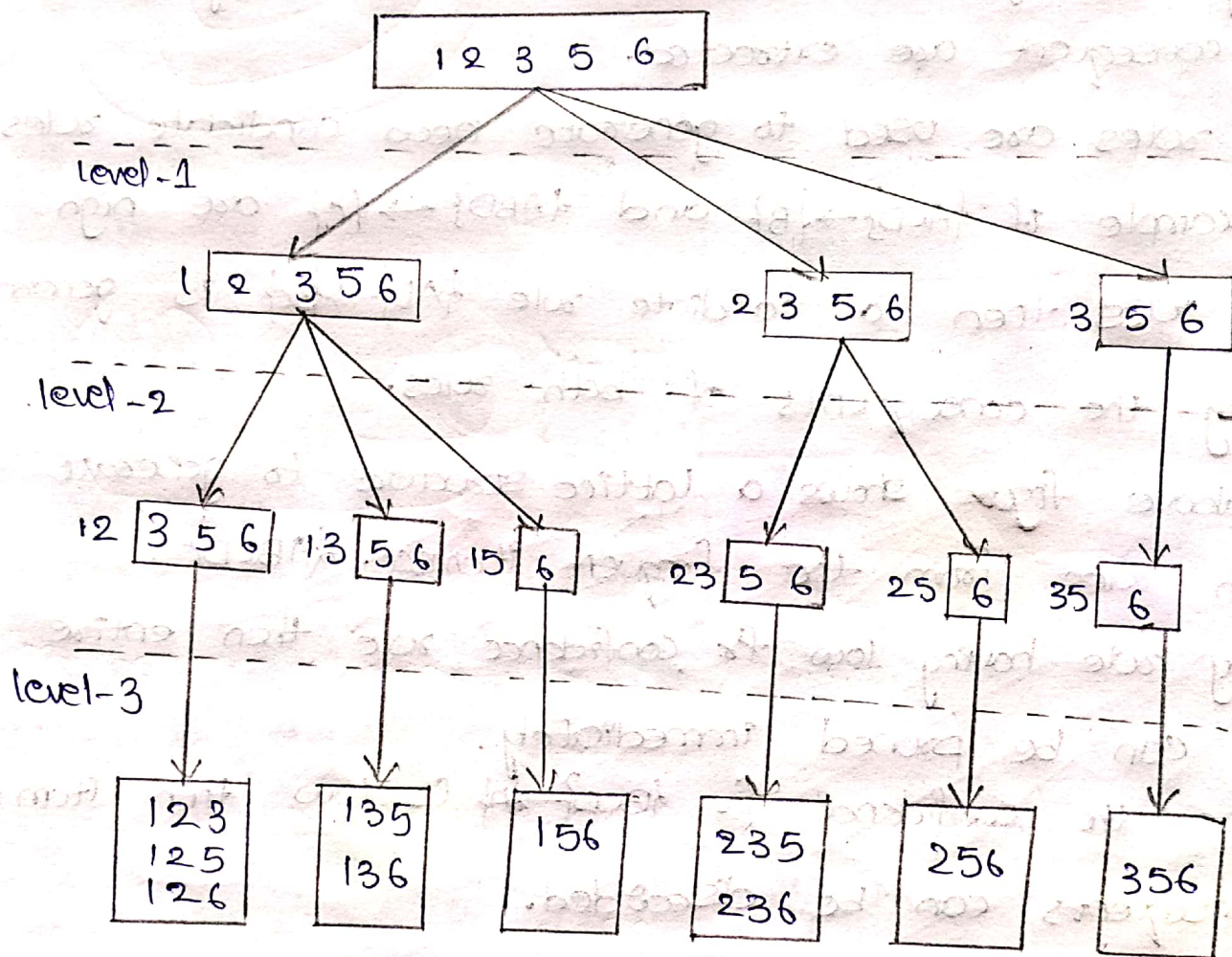
The support for doing this is to compare each transaction against every candidate item set and to update the support count.

For example consider a transaction t , that contains 5 items $\{1, 2, 3, 5, 6\}$, there are $\binom{5}{3} = 10$ itemsets of size 3. contained in this transaction.

Assume that each itemset keeps its items in increasing lexicographic order.

All 3 itemsets contain in t , must begin with 1, 2 (or) 3

It is not possible to construct a 3-itemset that begins with 5 (or) 6 because, there are only 2-items in it whose labels are greater than (or) equal to 5.



Data warehouse is subject oriented, Integrated, Time variant, Non volatile (No change in data) (loading, accessing) preprocessing, operational database (current data)

OLTP - online Transaction Process

2. Data warehouse data (historical data) (operations, Roll up, drill down, slice & dice, pivot) past 5-10 years

OLAP - online Analytical Process

ETL - Extract Transform and Loading

fact tables - measures

0 - Dimensional (apex) cuboid
n - Dimensional (base) cuboid

Measures

→ Distribute (sum, min, max)
→ Algebraic (avg, ...)
→ Statistical (mean, median, variance)

Star Schema - 1 fact table n dimensional tables

Snow flake - 1 fact table n dimensional tables