Content

- Basics and Algorithm
- Frequent Itemset Pattern & *Apriori* Principle
- FP-Growth, FP-Tree
- Handling Categorical Attributes

- 7 LH

4.1 Basics and Algorithm

4.1 Basics and Algorithm

- Many business enterprises accumulate large quantities of data from their day-to-day operations.
- For example, huge amounts of customer purchase data are collected daily at the checkout counters of grocery stores.
- Table 1 illustrates an example of such data, commonly known as Market basket transactions.
- Each row in this table corresponds to a transaction, which contains a unique identifier labeled TID and a set of items bought by a given customer.
- Retailers are interested in analyzing the data to learn about the purchasing behavior of their customers. Such valuable information can be used to support a variety of business-related applications such as marketing promotions, inventory management and customer relationship management.

TID	Items
1	{Bread, Milk}
2	{Bread, Diapers, Beer, Eggs}
3	{Milk, Diapers, Beer, Cola}
4	{Bread, Milk, Diapers, Beer}
5	{Bread, Milk, Diapers, Cola}

Table 1: Market basket transactions of five customers

- Association analysis is a methodology which is useful for discovering interesting relationships hidden in large datasets.
- The uncovered relationships can be represented in the form of association rules or sets of frequent items.
- For example, the following rule can be extracted from the data set shown in Table 1:

$$\{\text{Diapers}\} \rightarrow \{\text{Beer}\}$$

- The rule suggests that is a strong relationship exists between the sale of diapers and beer because many customers who buy diapers also buy beer.
- Retailers can use this type od rules to help them identify new opportunities for cross-selling their products to the customers.

Transaction ID	milk	bread	butter	beer
1	1	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	1	1	0
5	0	1	0	0

Support count:
Support count for an itemset is the number of transaction or market basket containing all items in the itemset l.

Support:
The support s for a particular association rule A → B is the proportion of transactions in D that contain both A and B. That is,

$$support(A \rightarrow B) = \frac{number\ of\ transactions\ containing\ both\ A\ and\ B}{total\ number\ of\ transactios}$$

Support of a rule determines how often a rule is applicable to a given data set.

- The confidence c of the association rule A → B is a measure of the accuracy of the rule, as determined by the percentage of transactions in D containing A that also contains B. In other words.

$$confidence(A \rightarrow B) = \frac{number\ of\ transactions\ containing\ both\ A\ and\ B}{number\ of\ transactios\ containing\ only\ A}$$

- Confidence determines how frequently items in B appear in the transactions that contain A.

- For example, a particular supermarket may find that of the 1000 customers shopping on a Thursday night, 200 bought diapers, and of the 200 who bought diapers, 50 bought beer. Thus, the association rule would be: “If buy diapers, then buy beer,” with a support of 50/1000 = 5% and a confidence of 50/200 = 25%.

Association Rule Discovery

- Given a set of transactions T, find all the rules having support ≥ minsup and confidence ≥ minconf, where minsup and minconf are the corresponding support and confidence thresholds. Some of approaches for association rules mining are:
  - Brute- Force Approach
  - Mining association rules
  - Itemset Lattice

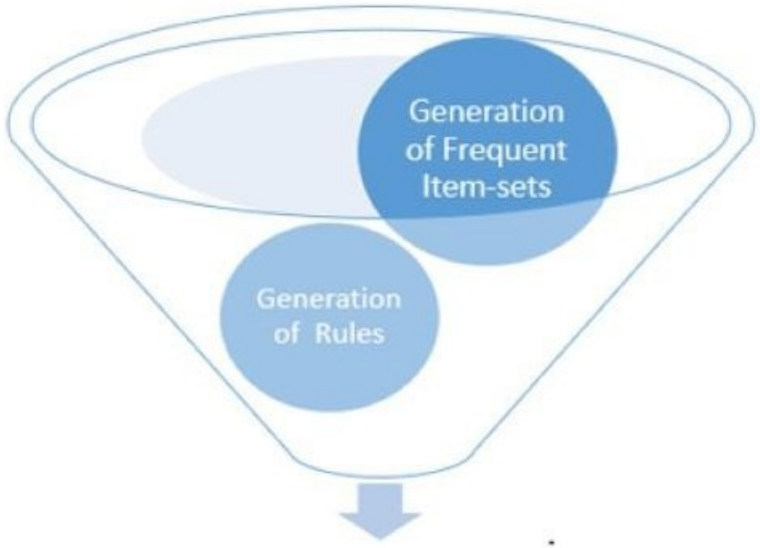
Association Rule

- Association rule learning is a type of unsupervised learning technique that checks for the dependency of one data item on another data item and maps accordingly so that it can be more profitable.
- It tries to find some interesting relations or associations among the variables of dataset.
- It is based on different rules to discover the interesting relations between variables in the database.
- The association rule learning is one of the very important concepts of machine learning, and it is employed in **Market Basket analysis, Web usage mining, continuous production, etc.**
- Here market basket analysis is a technique used by the various big retailer to discover the associations between items.
- We can understand it by taking an example of a supermarket, as in a supermarket, all products that are purchased together are put together.
- For example, if a customer buys bread, he most likely can also buy butter, eggs, or milk, so these products are stored within a shelf or mostly nearby. Consider the below diagram:

Association Rules Mining: Process

1. Frequent itemset generation

- Itemsets whose support is greater than the minimum_support are called frequent itemset.
- The main objective is to find all the itemsets that satisfy the minimum support threshold. These itemsets are called *frequent itemsets*.
- All other item-sets are filtered out.



2. Rule generation

- Association Rules are generated from the frequent itemset.
- Confidence for each of these association rules are calculated first.
- Association Rules that meets the minimum confidence thresholds are our required Association rules.

i) Brute-Force Approach

- A Brute-force approach for mining association rules is to compute the support and confidence for every possible rule.
- This approach is prohibitively expensive because there are exponentially many rules that can be extracted from a data set.
- More specifically, the total number of possible rules extracted from a dataset contains d items is : $R=3^d - 2^{(d+1)} +1$
- Even for a small data set like us this approach requires to compute many rules:
  - Data set containing 5 items: $3^5-2^6 +1 = 180$ rules
  - Data set containing 6 items: 602 rules and so on
- More than 80% of the rules are discarded after applying minsup=20% and minconf=50% thresholds, thus making most of the computation become wasted.
- To avoid performing needles computations, it would be useful to prune the rules early without having to compute their support and confidence values.

ii) Apriori Principle

- If an itemset is frequent, then all of its subsets must also be frequent.
- Suppose {c, d, e} is a frequent itemset, then all the subsets of {c, d, e} as shaded in the figure , must be frequent.
- This algorithm uses frequent datasets to generate association rules.
- It is designed to work on the databases that contain transactions.
- It is mainly used for market basket analysis and helps to understand the products that can be bought together. It can also be used in the healthcare field to find drug reactions for patients.

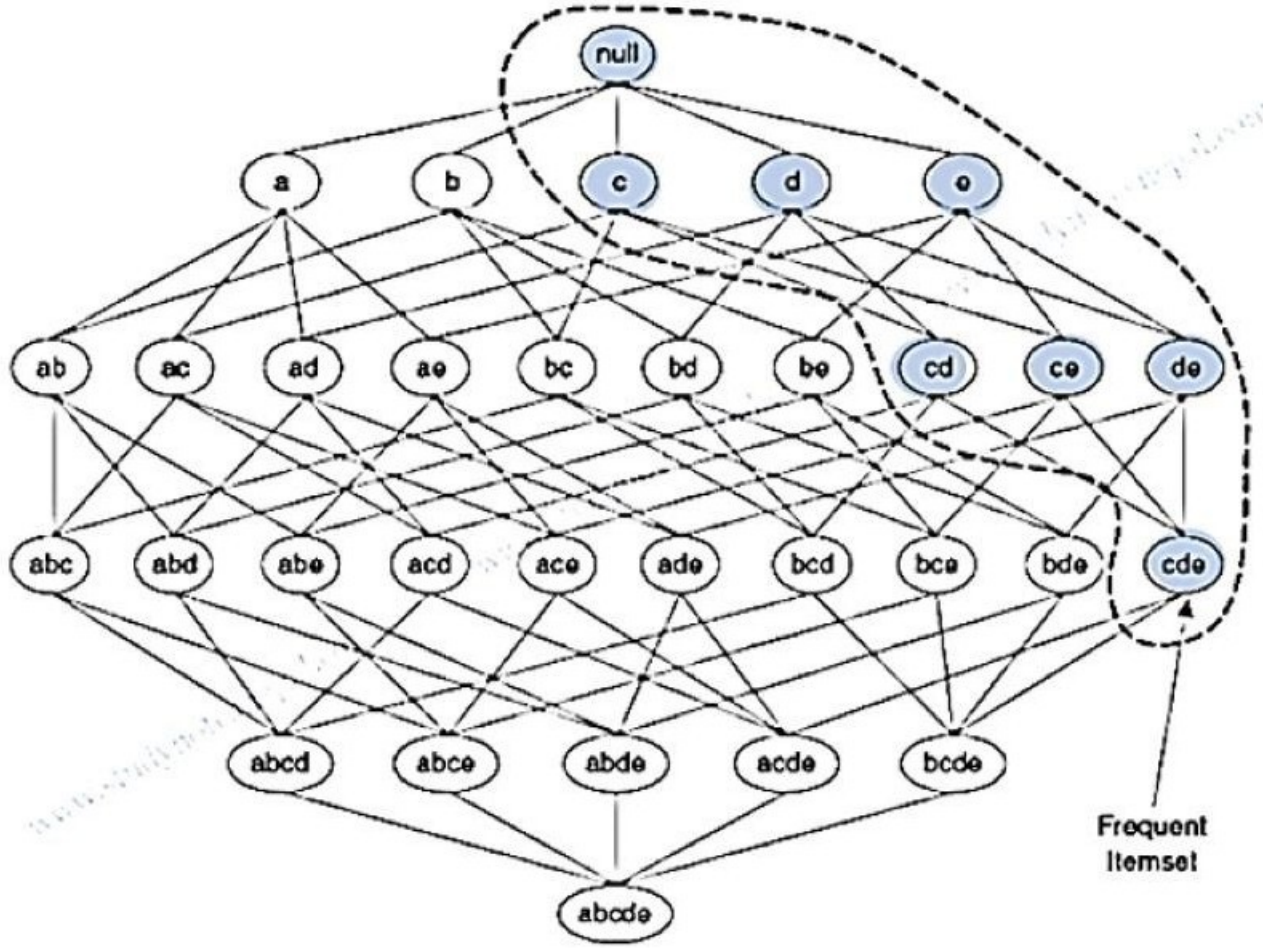


Figure 1: An illustration of the Apriori principle. If {c, d, e} is frequent then all subset of this itemset are frequent

- Conversely,

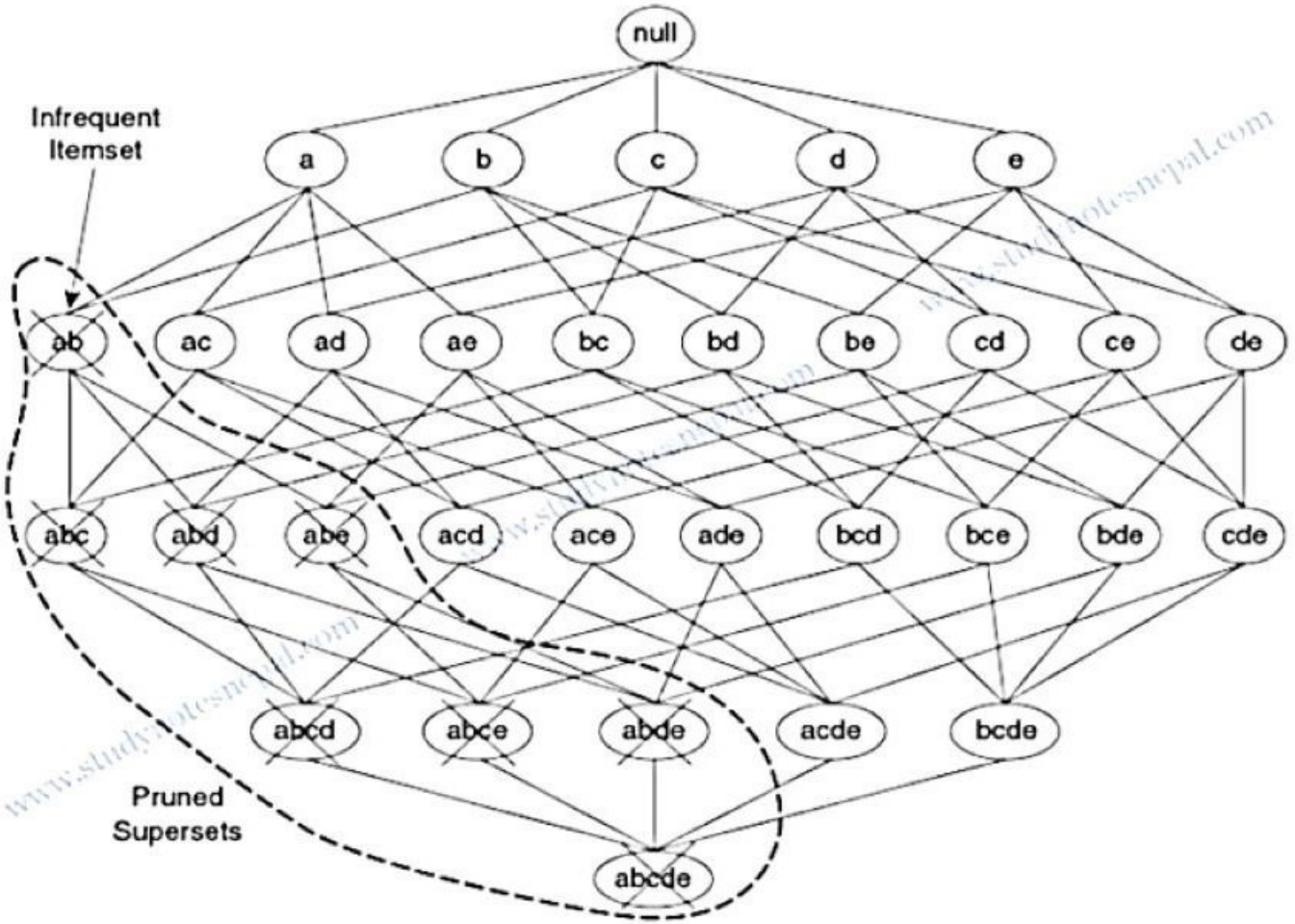


Figure 2: An illustration of support-based pruning. If {a, b} is infrequent then all superset of {a, b} are also infrequent

The following are the key concepts used in context to apriori algorithm

- Frequent Itemsets: The sets of an item that has minimum support
- Apriori Property: Any subset of frequent item-set must be frequent.
-

The Apriori algorithm was the first algorithm for frequent itemset mining to be proposed. R Agarwal and R Srikant improved it later, and it became known as **Apriori**.

To reduce the search space, this algorithm uses two steps: **"join" and "prune."** It is an iterative method for identifying the most frequent item sets.

The probability that item I is not frequent is if:

- P(I) < minimum support threshold, then I is not frequent. Here I belongs to itemset
- If an itemset set has a value less than minimum support then all of its supersets will also fall below min support, and thus can be ignored. This property is called the **Antimonotone property**.

The Apriori Algorithm for data mining includes the following steps:

- **Join Step:** By joining each item with itself.

- **Prune Step:** This step counts all of the items in the database. If a candidate item fails to fulfill the minimum support requirements, it is classified as infrequent and hence withdrawn. This step aims to reduce the size of the candidate itemsets.

Steps In Apriori Algorithm

The apriori algorithm is a series of steps that must be followed to find the most frequent itemset in a database. This data mining technique repeats the join and prune steps until the most frequently occurring itemset is found. The issue specifies a minimum assistance threshold, or the user assumes it.

- Each object is treated as a 1-itemsets candidate in the first iteration of the algorithm. Each item's occurrences will be counted by the algorithm.
- Set a minimum level of support, min sup. The set of 1-itemsets whose occurrence meets the minimum sup requirement is determined. Only those candidates with a score greater than or equal to min sup are advanced to the next iteration, while the rest are pruned.
- Next, min sup is used to find 2-itemset frequent itemsets. The 2-itemset is formed in the join phase by forming a group of 2 by combining items with itself.

- The min-sup threshold value is used to prune the 2-itemset candidates. The table will now have two itemsets, one with min-sup and the other with just min-sup.

- Using the join and prune step, the next iteration will create three –itemsets. This iteration will use the antimonotone property, which means that the subsets of 3-itemsets, i.e. the two itemset subsets of each category, will fall into min sup. The superset will be frequent if all 2-itemset subsets are frequent, otherwise, it will be pruned.

- Making 4-itemset by joining a 3-itemset with itself and pruning if its subset does not meet the min sup criteria would be the next move. When the most frequent itemset is reached, the algorithm is terminated.

T ID	Items Bought
1	{Bread, Butter, Milk}
2	{Bread, Butter}
3	{Beer, Cookies, Diapers}
4	{Milk, Diapers, Bread, Butter}
5	{Beer, Diapers}

1-Itemset	Support_count
Bread	3
Butter	3
Milk	2
Beer	2
Cookies	1
Diapers	3

Bread, Butter, Milk, Diapers, Beer

min_support = 40%,

min_support_count

= min_support x itemset_count

= 40% x 5

= 2

1-Frequent Itemset	Support_coun
Bread	3
Butter	3
Diapers	3
Milk	2
Beer	2

T ID	Items Bought
1	{Bread, Butter, Milk}
2	{Bread, Butter}
3	{Beer, Cookies, Diapers}
4	{Milk, Diapers, Bread, Butter}
5	{Beer, Diapers}

2-Itemset	Support_count
Bread, Butter	3
Bread, Diapers	1
Bread, Milk	2
Bread, Beer	0
Butter, Diapers	1
Butter, Milk	2
Butter, Beer	0
Diapers, Milk	1
Diapers, Beer	2
Milk, Beer	0

Bread, Butter, Diapers, Milk, Beer

2-Frequent Itemset	Support_count
Bread, Butter	3
Bread, Milk	2
Butter, Milk	2
Diapers, Beer	2

T ID	Items Bought
1	{Bread, Butter, Milk}
2	{Bread, Butter}
3	{Beer, Cookies, Diapers}
4	{Milk, Diapers, Bread, Butter}
5	{Beer, Diapers}

Bread, Butter, Milk, Diapers, Beer

3-Itemset	Support_count
Bread, Butter, Milk	2
Bread, Butter, Diapers	1
Bread, Butter, Beer	0
Bread, Milk, Diapers	1
Bread, Milk, Beer	0
Bread, Diapers, Beer	0
Butter, Milk, Diapers	1
Butter, Milk, Beer	0
Butter, Diapers, Beer	0
Milk, Diapers, Beer	0

1-Frequent Itemset	Support_count
Bread	3
Butter	3
Diapers	3
Milk	2
Beer	2

2-Frequent Itemset	Support_count
Bread, Butter	3
Bread, Milk	2
Butter, Milk	2
Diapers, Beer	2

3-Frequent Itemset	Support_count
Bread, Butter, Milk	2

- **min_confidence = 70%** ✓
- Confidence (X→Y) = P(Y | X) = P(X ∪ Y) / P(X)
- We have 5 frequent itemsets:
- {Bread, Butter}, {Bread, Milk}, {Butter, Milk}, {Diapers, Beer} and {Bread, Butter, Milk}.
- Therefore, candidate rules are:
- For {Bread, Butter},
  - bread>butter = 3/3 = 100% (Strong) ✓
  - butter>bread = 3/3 = 100% (Strong) ✓
- For {Bread, Milk}
  - bread>milk = 2/3 = 67% ✗
  - milk>bread = 2/2 =100% (Strong)

1-Frequent Itemset	Support_count
Bread	3
Butter	3
Diapers	3
Milk	2
Beer	2

2-Frequent Itemset	Support_count
Bread, Butter	3
Bread, Milk	2
Butter, Milk	2
Diapers, Beer	2

3-Frequent Itemset	Support_count
Bread, Butter, Milk	2

- **min_confidence = 70%**
- Confidence (X→Y) = P(Y | X) = P(X ∪ Y) / P(X)
- We have 5 frequent itemsets:
- {Bread, Butter}, {Bread, Milk}, {Butter, Milk}, {Diapers, Beer} and {Bread, Butter, Milk}.
- Therefore, candidate rules are:
- For {Butter, Milk}
  - butter>milk = 2/3 = 67% ✗
  - milk>butter = 2/2 = 100% (Strong) ✓
- For {Diapers, Beer}
  - diapers>beer = 2/3 = 67% ✗
  - beer>diapers = 2/2 = 100% (Strong)

1-Frequent Itemset	Support_count
Bread	3
Butter	3
Diapers	3
Milk	2
Beer	2

2-Frequent Itemset	Support_count
Bread, Butter	3
Bread, Milk	2
Butter, Milk	2
Diapers, Beer	2

3-Frequent Itemset	Support_count
Bread, Butter, Milk	2

- **min_confidence = 70%**
- Confidence (X→Y) = P(Y | X) = P(X ∪ Y) / P(X)
- We have 5 frequent itemsets:
- {Bread, Butter}, {Bread, Milk}, {Butter, Milk}, {Diapers, Beer} and {Bread, Butter, Milk}.
- Therefore, candidate rules are:
- For {Bread, Butter, Milk}
  - bread,butter>milk = 2/3 = 67% ✗
  - bread,milk>butter = 2/2 = 100% (Strong) ✓
  - milk,butter>bread = 2/2 = 100% (Strong) ✓
  - bread>butter,milk = 2/3 = 67% ✗
  - butter>bread,milk = 2/3 = 67% ✗
  - milk>bread,butter = 2/2=100% (Strong) ✓

Transaction	List of items
T1	I1,I2,I3
T2	I2,I3, l4
T3	I4,l5
T4	I1,I2,I3
T5	I1,I2,I3,I5
T6	I1,I2,I3,I4

Example of Apriori: Support threshold=50%, Confidence= 60%

Solution:

Support threshold=50% => 0.5*6= 3 =>min_sup=3

1. Frequency of each item.

Table 3: Items With Frequency Count

Item	Count
I1	4
I2	5
I3	4
I4	4
I5	2

2. Prune Step: Table 3 shows that the I5 item does not meet min_sup=3, thus it is deleted, only I1, I2, I3, I4 meet min_sup count.
3. Step 2: Form a 2-itemset. Find the occurrences of 2-itemset in Table 2.

Table 4: Two Itemset Data

Item	Count
I1,I2	4
I1,I3	3
I1,I4	2
I2,I3	4
I2,I4	3
I3,I4	2

4. Prune Step: Table 4reveals that item sets I1, I4, and I3, I4 do not follow min sup, so they are deleted.

Table 5: Frequent Two Itemset data

Item	Count
I1,I2	4
I1,I3	3
I2,I3	4
I2,I4	3

5. Join and Prune Step: join and prune Form a three-itemset. Find the occurrences of the 3-itemset in Table 2. Find the 2-itemset subsets that endorse min sup from Table 5.

We can see that for itemset{ I1, I2, I3} subsets, TABLE-5 contains {I1, I2}, {I3, I2},{I1, I3} indicating that {I1, I2, I3 } is frequent.

Table 6: Three Itemset Data

Item
I1,I2,I3
I1,I2,I4
I1,I3,I4

Only {I1, I2, I3} is frequent.

6. **Generate Association Rules:** From the frequent itemset discovered above the association could be:

{I1, I2} => {I3}

Confidence = support {I1, I2, I3} / support {I1, I2} = (3/ 4)* 100 = 75%

{I1, I3} => {I2}

Confidence = support {I1, I2, I3} / support {I1, I3} = (3/ 3)* 100 = 100%

{I2, I3} => {I1}

Confidence = support {I1, I2, I3} / support {I2, I3} = (3/ 4)* 100 = 75%

{I1} => {I2, I3}

Confidence = support {I1, I2, I3} / support {I1} = (3/ 4)* 100 = 75%

{I2} => {I1, I3}

Confidence = support {I1, I2, I3} / support {I2 = (3/ 5)* 100 = 60%

{I3} => {I1, I2}

Confidence = support {I1, I2, I3} / support {I3} = (3/ 4)* 100 = 75%

This shows that all the above association rules are strong if minimum confidence threshold is 60%.

Advantages

- The algorithm is simple to comprehend.
- On large itemets in large databases, the join and prune steps are simple to implement.

Disadvantages

- If the itemsets are wide and the minimum support is held low, it necessitates a lot of computation.
- The database as a whole must be scanned.

iii) F-P Growth Algorithm

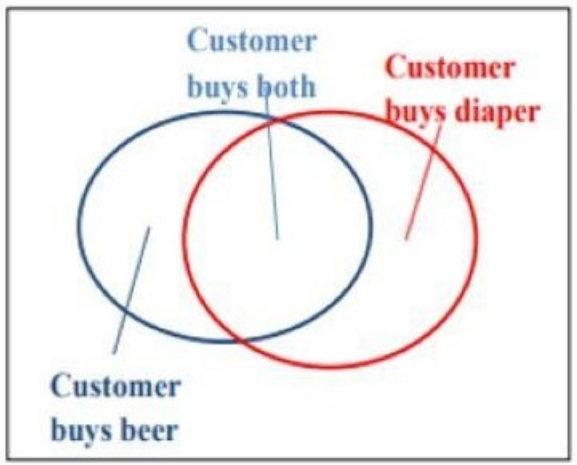
- The F-P growth algorithm stands for **Frequent Pattern**, and it is the improved version of the Apriori Algorithm.
- It represents the database in the form of a tree structure that is known as a frequent pattern tree.
- The purpose of this frequent tree is to extract the most frequent patterns.

4.2 Frequent Itemset Pattern & *Apriori* Principle

Basic Concepts

- **Itemset**
  - A set of one or more items
    - E.g.: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
- **Support count (σ)**
  - Frequency of occurrence of an itemset (number of transactions it appears)
  - E.g. **σ({Milk, Bread,Diaper}) = 2**
- **Support(s)**
  - Fraction of the transactions in which an itemset appears
  - 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



Association Rule

- X → Y, where X and Y are non-overlapping itemsets
- **{Milk, Diaper} → {Beer}**

Rule Evaluation Metrics

- **Support (s)**
  - Fraction of transactions that contain both X and Y
  - **i.e., support of the itemset $X \cup Y$**
- **Confidence (c)**
  - Measures how often items in Y appear in transactions that contain X

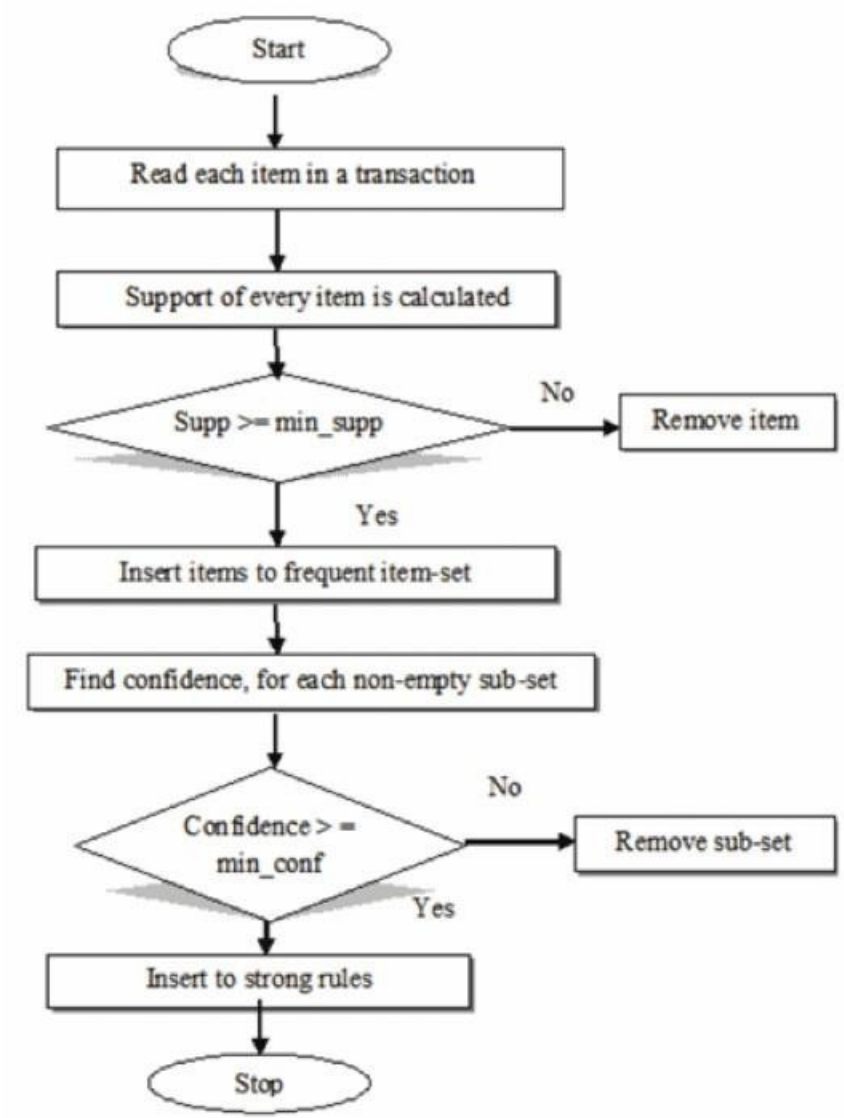
Example:

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

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

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

Apriori Algorithm



C_k : Candidate itemset of size k

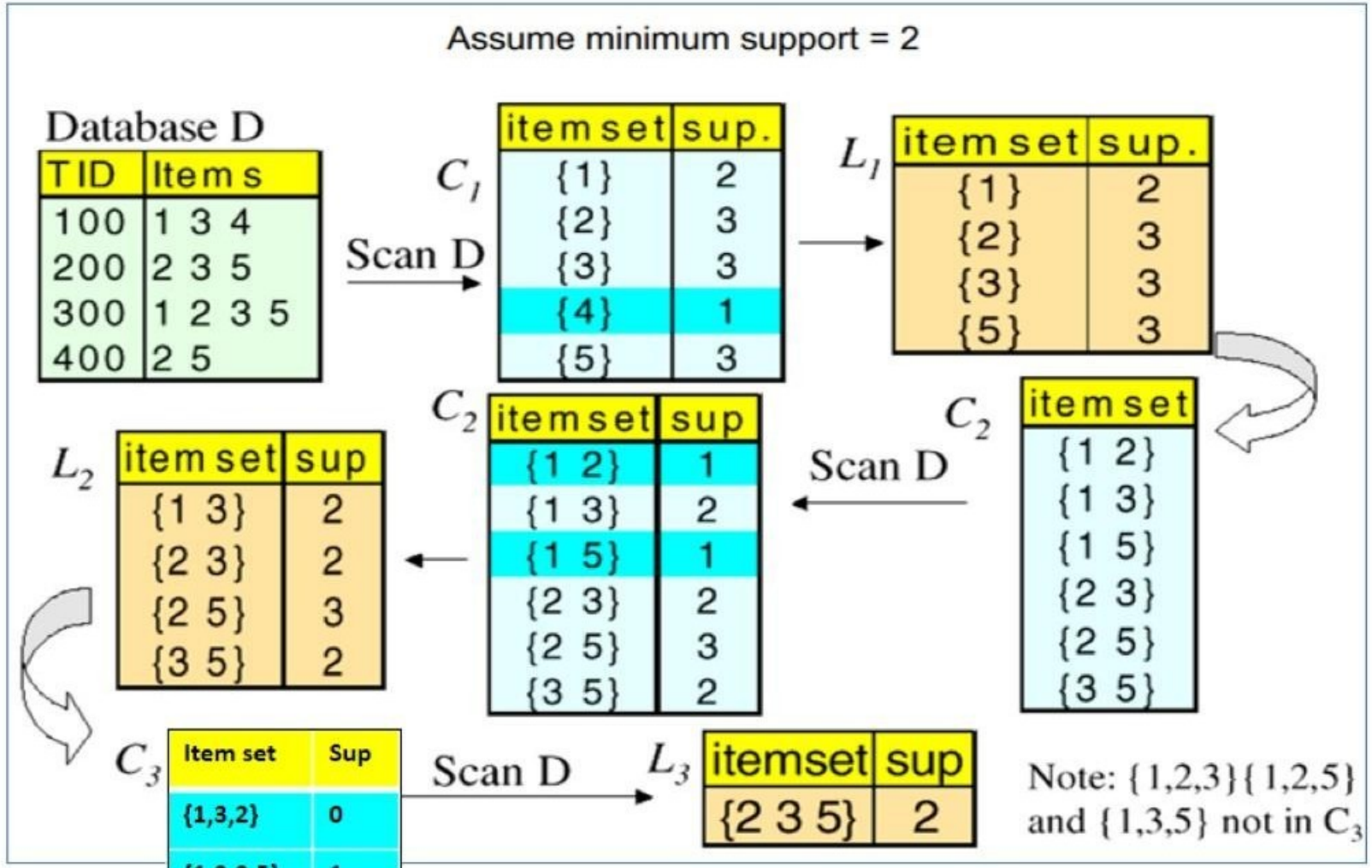
L_k : Frequent itemset of size k

$L_1 = \{\text{frequent items}\};$
for $(k = 1; L_k \mathrel{!}= \emptyset; k{+}{+})$ **do begin**
  C_{k+1} = candidates generated from L_k;
  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
end
return $\cup_k L_k$;

Solved Example 1: Apriori Algorithm

- Given the following transaction set, find the frequent itemset using Apriori Algorithm. Assume Minimum support=2, confidence = 75%. Also, evaluate the Association rules.

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



L_2

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

L_3

itemset	sup
{2 3 5}	2

- The final "frequent" item sets are those remaining in L2 and L3.
- However, {2,3}, {2,5}, and {3,5} are all contained in the larger item set {2, 3, 5}.
- Thus, the final group of item sets reported by Apriori are **{1,3}** and **{2,3,5}.**
- These are the only item sets from which we will generate association rules.

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

We have:

$$\text{Confidence } (A \rightarrow B) = \frac{\text{number of transactions containing both A and B}}{\text{Total numbers of transactions only A}}$$

- Confidence (1→3)
  $= \frac{\text{number of transactions containing both 1 and 3}}{\text{Total numbers of transactions only 1}}$
  $= \frac{2}{2}$
  $= 1.0$
- Confidence (3→1)
  $= \frac{\text{number of transactions containing both 3 and 1}}{\text{Total numbers of transactions only 3}}$
  $= \frac{2}{3}$
  $= 0.67$
- Confidence ({2,3}→5)
  $= \frac{\text{number of transactions containing both \{2,3\} and 5}}{\text{Total numbers of transactions only \{2,3\}}}$
  $= \frac{2}{2}$
  $= 1.0$

Candidate rules for {1,3}		Candidate rules for {2,3,5}			
Rule	Conf.	Rule	Conf.	Rule	Conf.
{1}→{3}	2/2 = 1.0	{2,3}→{5}	2/2 = 1.00	{2}→{5}	3/3 = 1.00
{3}→{1}	2/3 = 0.67	{2,5}→{3}	2/3 = 0.67	{2}→{3}	2/3 = 0.67
		{3,5}→{2}	2/2 = 1.00	{3}→{2}	2/3 = 0.67
		{2}→{3,5}	2/3 = 0.67	{3}→{5}	2/3 = 0.67
		{3}→{2,5}	2/3 = 0.67	{5}→{2}	3/3 = 1.00
		{5}→{2,3}	2/3 = 0.67	{5}→{3}	2/3 = 0.67

Assuming a min. confidence of 75%, the final set of rules reported by Apriori are: {1}→{3}, {2,3}→{5}, {3,5}→{2}, {5}→{2} and {2}→{5}

Assignment:

1. **Given the following transaction set, find the frequent itemset using Apriori Algorithm. Assume Minimum support=33%, confidence = 60%. Also, evaluate the Association rules.**

1	Milk	Egg	Bread	Butter
2	Milk	Butter	Egg	Ketchup
3	Bread	Butter	Ketchup	
4	Milk	Bread	Butter	
5	Bread	Butter	Cookies	
6	Milk	Bread	Butter	Cookies
7	Milk	Cookies		
8	Milk	Bread	Butter	
9	Bread	Butter	Egg	Cookies
10	Milk	Butter	Bread	
11	Milk	Bread	Butter	
12	Milk	Bread	Cookies	Ketchup

Reference: https://www.youtube.com/watch?v=43CMKRHdH30

Assignment:

2. Use the Apriori algorithm using candidate generation for finding frequent itemset and then evaluate the valid association rules: [BIM 2017, Group C]

TID	List of items
T100	A,C,D
T200	B,C,E
T300	A,B,C,E
T400	B,E

3. Given the following transaction set, find the frequent itemset using Apriori algorithm. (Minimum support =2). [BIM 2022, Group B]

Transaction	Items
T1	{pasta, lemon, bread, orange}
T2	{pasta, lemon}
T3	{pasta, orange, cake}
T4	{pasta, lemon, orange, cake}

4.3 FP-Growth and FP Tree

4.3 FP-Growth

- **Frequent Pattern Growth Algorithm**
- This algorithm is an improvement to the Apriori method.
- A frequent pattern is generated without the need for candidate generation.
- FP growth algorithm represents the database in the form of a tree called a frequent pattern tree or FP tree.
- This tree structure will maintain the association between the itemsets.
- The database is fragmented using one frequent item. This fragmented part is called “pattern fragment”.
- The itemsets of these fragmented patterns are analyzed.
- Thus with this method, the search for frequent itemsets is reduced comparatively.

FP-Tree

- Frequent Pattern Tree is a tree-like structure that is made with the initial itemsets of the database.
- The purpose of the FP tree is to mine the most frequent pattern.
- Each node of the FP tree represents an item of the itemset.
- The root node represents null while the lower nodes represent the itemsets.
- The association of the nodes with the lower nodes that is the itemsets with the other itemsets are maintained while forming the tree.

Frequent Pattern Algorithm Steps

1. Compute the *frequencies* of the itemsets in the database.
2. Arrange the itemsets in *descending order* after applying Minimum_support.
3. Generate *ordered_itemset.*
4. Construct *FP-tree*.
5. Compute *Conditional pattern base*.
6. Generate *Conditional Frequent Patterns Tree*
7. Generate *Frequent Pattern* from the Conditional FP Tree, considering the minimum_support.

Refernce: https://www.softwaretestinghelp.com/fp-growth-algorithm-data-mining/#:~:text=FP%20growth%20algorithm%20represents%20the,is%20called%20%E2%80%9Cpattern%20fragment%E2%80%9D

Solved Example 1:

- Construct FP-tree for below hypothetical dataset of transactions with each letter representing an item. Let the minimum support be 3.

Transaction ID	Items
T1	{E,K,M,N,O,Y}
T2	{D,E,K,N,O,Y}
T3	{A,E,K,M}
T4	{C,K,M,U,Y}
T5	{C,E,I,K,O,O}

Reference: https://www.geeksforgeeks.org/ml-frequent-pattern-growth-algorithm/

Reference Video: https://www.youtube.com/watch?v=7oGz4PCp9jI

Solution:

Given dataset is:

Transaction ID	Items
T1	{E,K,M,N,O,Y}
T2	{D,E,K,N,O,Y}
T3	{A,E,K,M}
T4	{C,K,M,U,Y}
T5	{C,E,I,K,O,O}

Step 1: Compute the frequencies of the itemsets in the database

Item	Frequency	
A	1	
C	2	
D	1	
E	4	✓
I	1	
K	5	✓
M	3	✓
N	2	
O	3	✓
U	1	
Y	3	✓

Step 2: Build FP set (L) in descending order after applying Minimum_support.

$$L = \{K : 5, E : 4, M : 3, O : 3, Y : 3\}$$

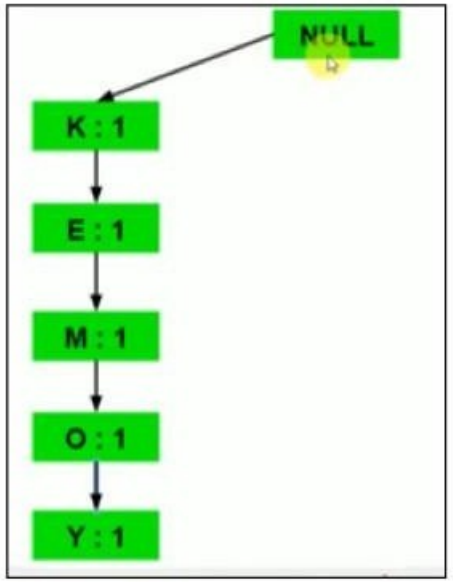
Step 3: Generate ordered_itemset.

Transaction ID	Items	Ordered-Item Set
T1	{E,K,M,N,O,Y}	{K,E,M,O,Y}
T2	{D,E,K,N,O,Y}	{K,E,O,Y}
T3	{A,E,K,M}	{K,E,M}
T4	{C,K,M,U,Y}	{K,M,Y}
T5	{C,E,I,K,O,O}	{K,E,O}

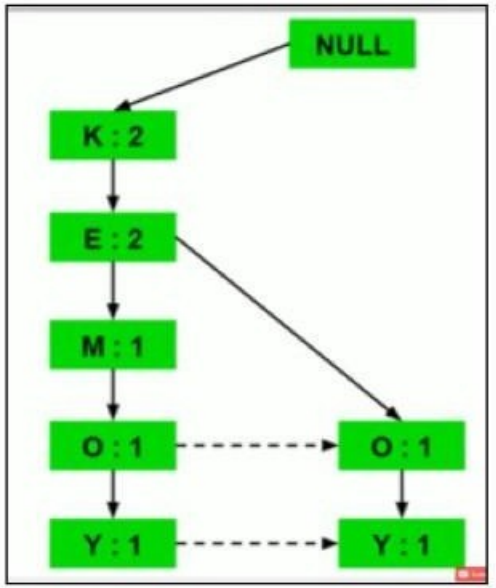
Step 4: Construct FP-tree.

Transaction ID	Items	Ordered-Item Set
T1	{E,K,M,N,O,Y}	{K,E,M,O,Y}
T2	{D,E,K,N,O,Y}	{K,E,O,Y}
T3	{A,E,K,M}	{K,E,M}
T4	{C,K,M,U,Y}	{K,M,Y}
T5	{C,E,I,K,O,O}	{K,E,O}

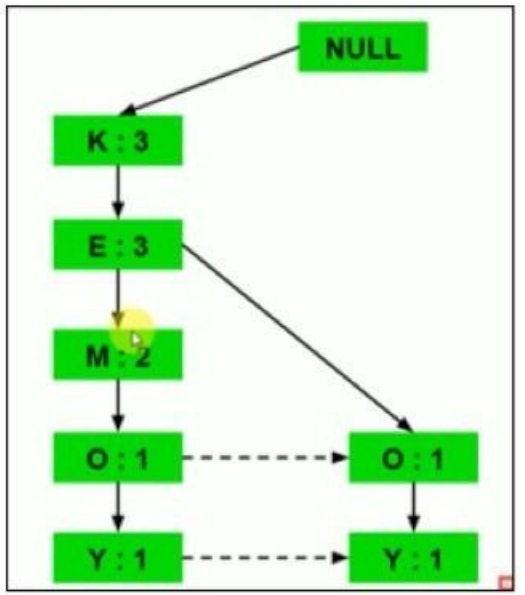
a) Inserting the set {K, E, M, O, Y}:



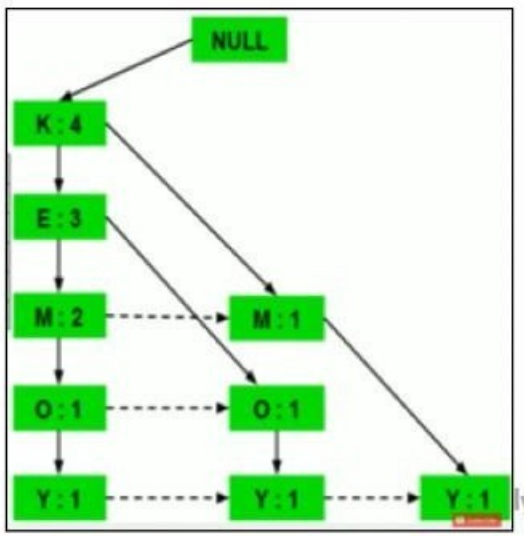
b) Inserting the set {K, E, O, Y}:



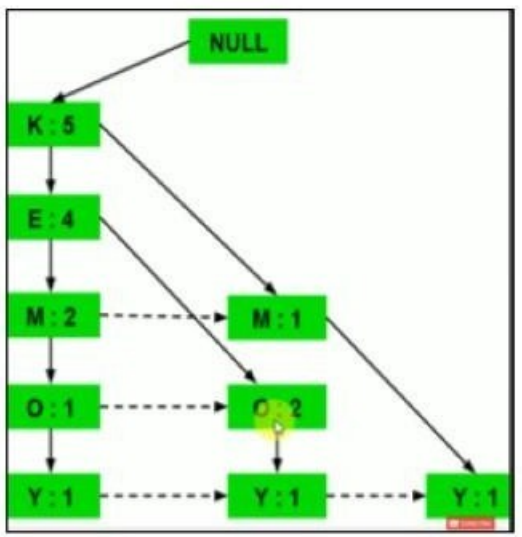
c) Inserting the set {K, E, M}:

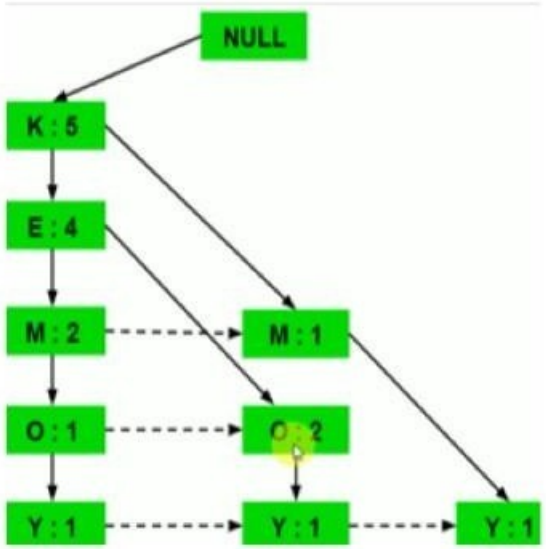


d) Inserting the set {K, M, Y}:



e) Inserting the set {K, E, O}:





Step 5: Compute *Conditional pattern base* from the FP tree.

Now, for each item, the **Conditional Pattern Base** is computed which is path labels of all the paths which lead to any node of the given item in the frequent-pattern tree.

Items	Conditional Pattern Base
Y	{{K,E,M,O : 1}, {K,E,O : 1}, {K,M : 1}}
O	{{K,E,M : 1}, {K,E : 2}}
M	{{K,E : 2}, {K : 1}}
E	{K : 4}
K	

Step 6 :Generate Frequent Patterns Tree

Items	Conditional Pattern Base	Conditional Frequent Pattern Tree
Y	{{K,E,M,O : 1}, {K,E,O : 1}, {K,M : 1}}	{K : 3}
O	{{K,E,M : 1}, {K,E : 2}}	{K,E : 3}
M	{{K,E : 2}, {K : 1}}	{K : 3}
E	{K : 4}	{K : 4}
K		

Step 7 :Generate Frequent Patterns Rules

- From the Conditional Frequent Pattern tree, the **Frequent Pattern rules** are generated by pairing the items of the Conditional Frequent Pattern Tree set to the corresponding to the item as given in the below table.

Items	Frequent Pattern Generated
Y	{<K,Y : 3>}
O	{<K,O : 3>, <E,O : 3>, <E,K,O : 3>}
M	{<K,M : 3>}
E	{<K, E: 4}
K	

Step 8 : Find the confidence for each rule and apply the confidence threshold.

Items	Frequent Pattern Generated
Y	{<K,Y : 3>}
O	{<K,O : 3>, <E,O : 3>, <E,K,O : 3>}
M	{<K,M : 3>}
E	{<K, E: 4}
K	

Step 8 : Find the confidence for each rule and apply the confidence threshold (Assume confidence=75%).

For Confidence (A→B) = number of transactions containing both A and B / Total number of transactions only A

Candidate rules for {K,Y}	Confidence.
{K}→{Y}	3/5 = 0.6
{Y}→{K}	3/3 = 1

Candidate rule for {K,O}, {E,O}, {E,K,O}

~~{K}→{O}~~	Confidence.	~~Confidence~~	Confidence
{K}→{O}	3/5 = 0.6	{O}→{E,K}	3/3 = 1
{O}→{K}	3/3 = 1	{E}→{K,O}	3/4 = 0.75
{E}→{O}	3/4 = 0.75	{K}→{E,O}	3/5 = 0.6
{O}→{E}	3/3 = 1		
{E,K}→{O}	3/4 = 0.75		
{K,O}→{E}	3/3 = 1		
{E,O}→{K}	3/3 = 1		

~~Can~~

Candidate rule for (K,M)	Confidence
{K}→{M}	3/5 = 0.6
{M}→{K}	3/3 = 1

Candidate rule for {K,E}	Confidence
{K}→{E}	4/5 = 0.8
{E}→{K}	4/3 = 1.33

Assuming a min confidence of 75% of the final set of rules reported by {Y}→{K}, {O}→{K}, {E}→{O}, ~~{O}~~ {O}→{K}, {E,K}→{O}, {K,O}→{E}, {E,O}→{K}, {O}→{E,K}, {E}→{K,O}, {M}→{K}, and {E}→{K}.

Assignment 1:

Consider the following transaction data sets. And construct the FP tree and find the FP rules.

Support threshold=50%, Confidence= 60%

Transaction	List of items
T1	I1,I2,I3
T2	I2,I3,I4
T3	I4,I5
T4	I1,I2,I4
T5	I1,I2,I3,I5
T6	I1,I2,I3,I4

Solution:

Given dataset is:

Transaction	List of items
T1	I1,I2,I3
T2	I2,I3,I4
T3	I4,I5
T4	I1,I2,I4
T5	I1,I2,I3,I5
T6	I1,I2,I3,I4

Support threshold=50% => 0.5*6= 3 => min_sup=3

Step 1: Compute the frequencies of the itemsets in the database

Item	Count
I1	4
I2	5
I3	4
I4	4
I5	2

Step 2: Build FP set (L) in descending order after applying Minimum_support.

Item	Count
I2	5
I1	4
I3	4
I4	4

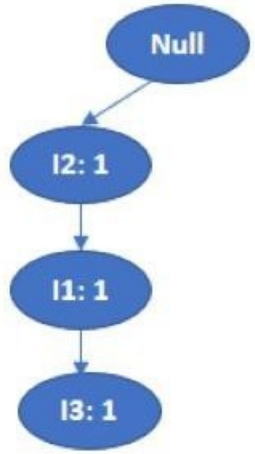
Step 3: Generate ordered_itemset.

Transaction	Items	Ordered-Item set
T1	I1, I2,I3	I2, I1,I3
T2	I2, I3, I4	I2, I3, I4
T3	I4, I5	I4
T4	I1, I2, I4	I2, I1, I4
T5	I1, I2, I3, I5	I2, I1, I3
T6	I1, I2, I3, I4	I2, I1, I3, I4

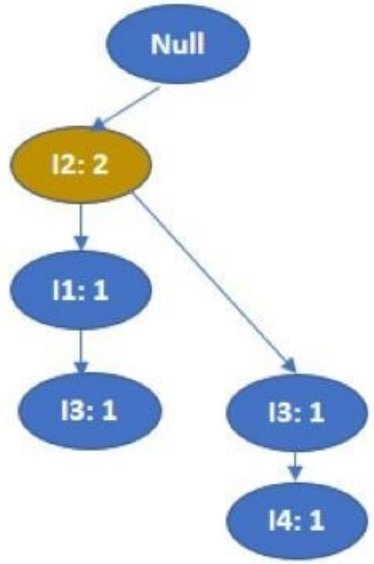
Step 4: Construct FP-tree.

Transaction	Items	Ordered-Item set
T1	I1, I2,I3	I2, I1, I3
T2	I2, I3, I4	I2, I3, I4
T3	I4, I5	I4
T4	I1, I2, I4	I2, I1, I4
T5	I1, I2, I3, I5	I2, I1, I3
T6	I1, I2, I3, I4	I2, I1, I3, I4

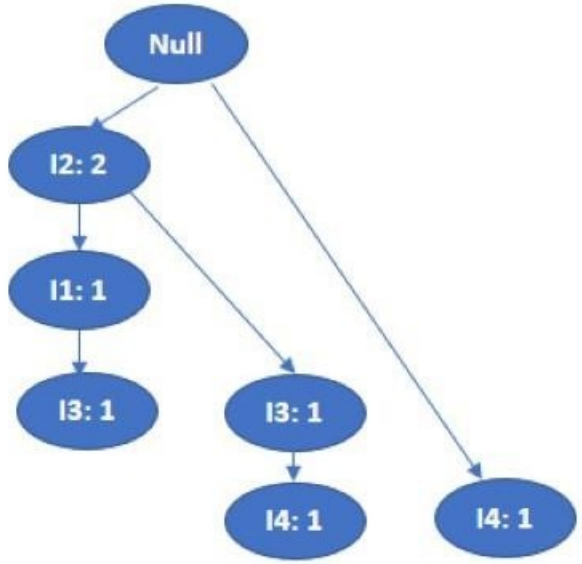
a) Inserting {I2, I1, I3}



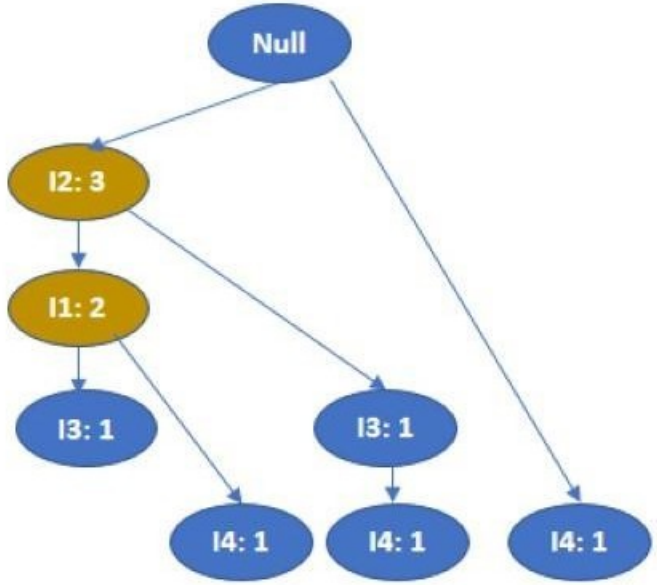
b) Inserting {I2, I3, I4}



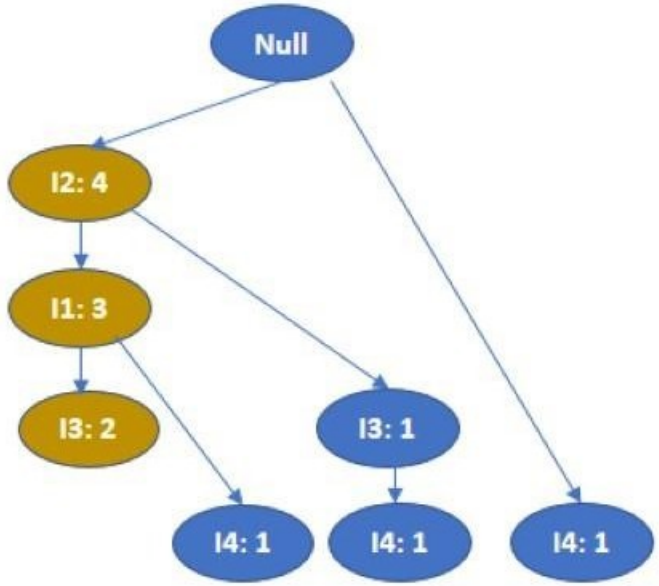
c) Inserting {I4}



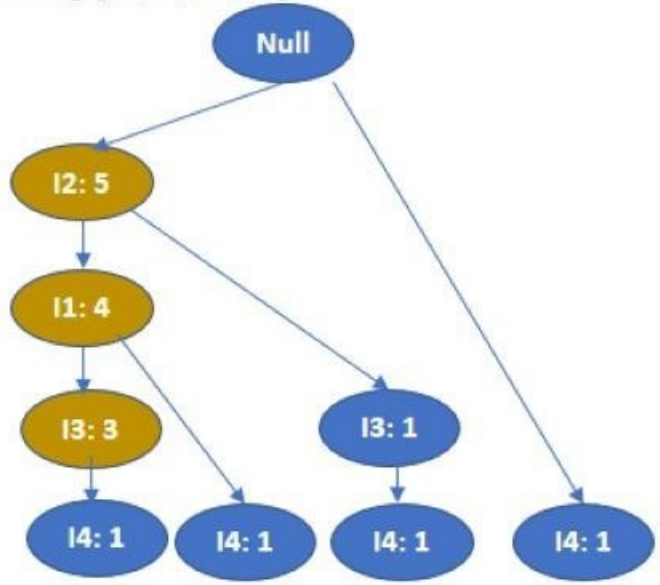
d) Inserting {I2, I1, I4}



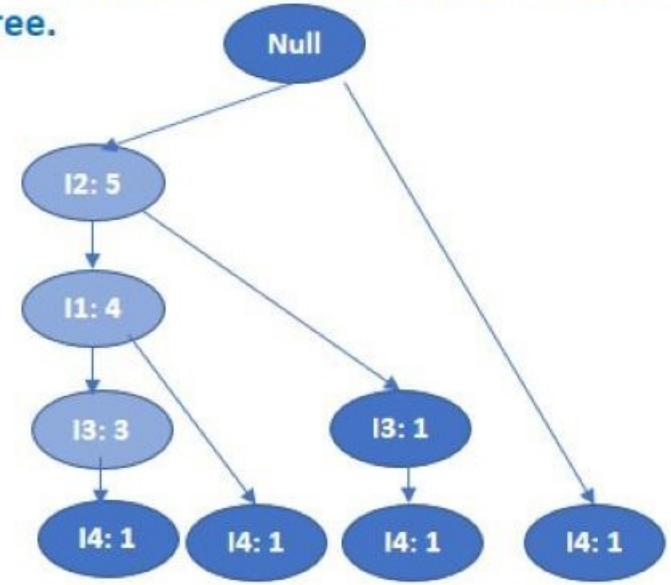
e) Inserting {I2, I1,I3}



f) Inserting {I2, I1, I3, I4}



Step 5: Compute *Conditional pattern base* from the FP tree.



Items	Conditional Patten Base
I4	{I2,I1,I3:1},(I2,I1:1}, {I2,I3:1}
I3	{I2,I1:3}, {I2 : 1}
I1	{I2: 4}
I2	

Step 6 :Generate Frequent Patterns Tree

Items	Conditional Patten Base	Conditional FP Tree set
I4	{I2,I1,I3:1}, {I2,I1:1}, {I2,I3:1}	{I2: 3}
I3	{I2,I1:3}, {I2 : 1}	{I2: 4}, {I1 :3}
I1	{I2: 4}	{I2: 4}
I2		

Step 7 :Generate Frequent Patterns Rules

- From the Conditional Frequent Pattern tree, the **Frequent Pattern rules** are generated by pairing the items of the Conditional Frequent Pattern Tree set to the corresponding to the item as given in the below table.

Items	Conditional FP Tree set	Frequent pattern
I4	{I2: 3}	{ I2, I4 : 3}
I3	{I2: 4}, {I1 :3}	{ I2, I3 : 4 }, {I1, I3 : 3}, {I2, I1, I3 : 3}
I1	{I2: 4}	[I2, I1 : 4 }
I2		

Step 8 : Find the confidence for each rule and apply the confidence threshold 60%.

Assignment 2:

Construct the FP tree for the following transactions. [BIM 2018, Group B]

TID	Items
100	{a,b,c}
200	{b,c,d}
300	{a,c,e}
400	{b,d,f}
500	{a,b,e}
600	{b,c,f}

Assignment 3:

Consider the following transaction data sets. And construct the FP tree. [BIM 2021, Group B]

TID	Items
1	E, A, D, B
2	D, A, C, E, B
3	C, A, B, E
4	B, A, D
5	D
6	D, B
7	A, D, E
8	B, C

Advantages and Disadvantages of FP algorithm

- **Advantages :**
  1. This algorithm needs to scan the database only twice when compared to Apriori which scans the transactions for each iteration.
  2. Faster since the pairing of items is not done in this algorithm.
  3. The database is stored in a compact version in memory.
  4. It is efficient and scalable for mining both long and short frequent patterns.

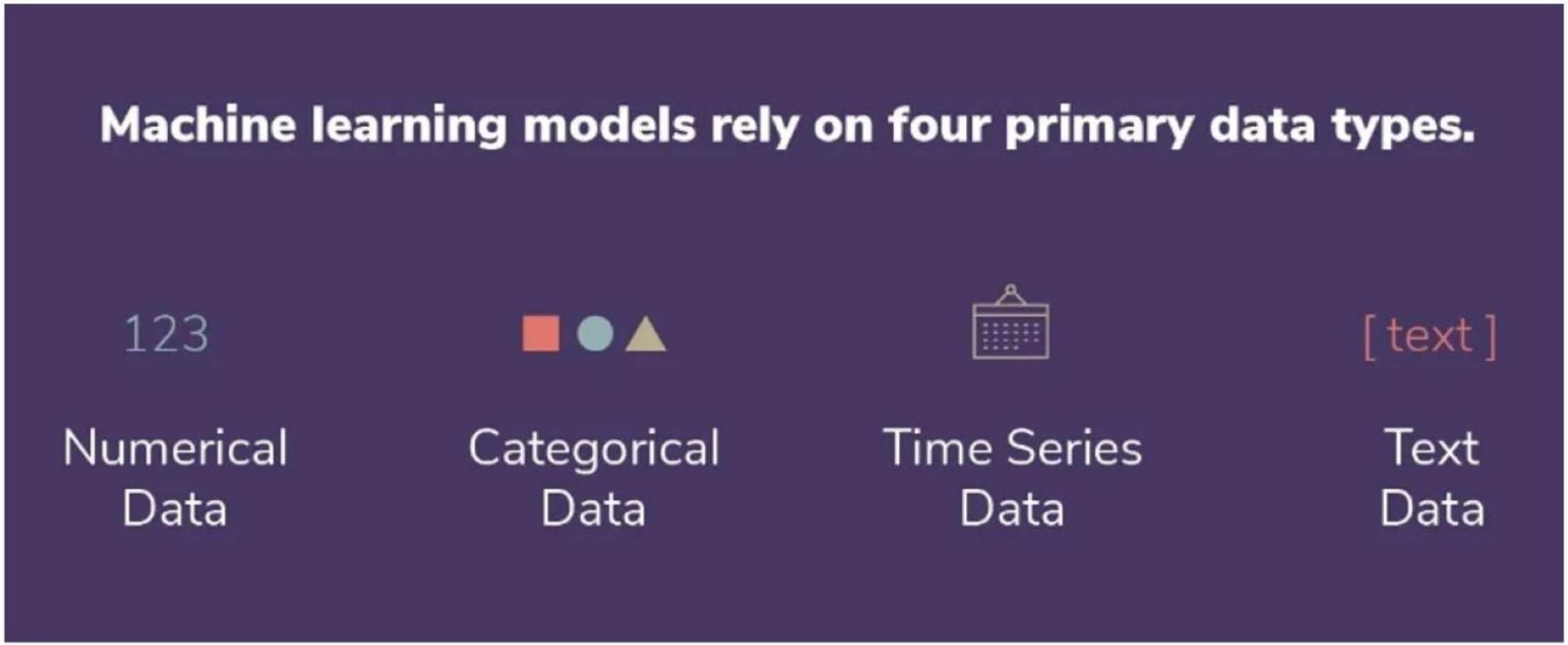
- **Disadvantages:**
  1. FP Tree is more cumbersome and difficult to build than Apriori.
  2. It may be expensive.
  3. When the database is large, the algorithm may not fit in the shared memory.

Difference between Apriori and FP Growth Algorithm

Apriori	FP Growth
1. Apriori generates frequent patterns by making the itemsets using pairings such as single item set, double itemset, and triple itemset.	1. FP Growth generates an FP-Tree for making frequent patterns.
2. Apriori uses candidate generation where frequent subsets are extended one item at a time.	2. FP-growth generates a conditional FP-Tree for every item in the data.
3. Since apriori scans the database in each step, it becomes time-consuming for data where the number of items is larger.	3. FP-tree requires only one database scan in its beginning steps, so it consumes less time.
4. A converted version of the database is saved in the memory	4. A set of conditional FP-tree for every item is saved in the memory
5. It uses a breadth-first search	5. It uses a depth-first search.

Handling Categorical Attributes

Primary data types



Categorical Attributes

- Categorical data is a type of data that is used to group information with similar characteristics, while numerical data is a type of data that expresses information in the form of numbers.
- Categorical variables are usually represented as 'strings' or 'categories' and are finite in number.
- Here are a few examples:
  - City: Delhi, Mumbai, Ahmedabad, Bangalore, etc.
  - Department: Finance, Human resources, IT, Production.
  - Degree: High school, Diploma, Bachelors, Masters, PhD.
  - Grades: A+, A, B+, B, B- etc.
- There are two kinds of categorical data-
  - **Ordinal Data:** The categories have an inherent order
  - **Nominal Data:** The categories do not have an inherent order

Handling Categorical Attributes

- Most machine learning algorithms cannot handle categorical variables unless we convert them to numerical values
- Since we are dealing with a mathematical model in machine learning, it is significant that we can convert this category into numeric numbers prior to utilizing it for training our model.

Different Approaches to Handle Categorical Data

1. Label Encoding
2. Ordinal Number Encoding
3. Binary Encoding
4. Count or Frequency Encoding
5. **One Hot Encoding**
6. One Hot Encoding with multiple categories
7. Target guided Ordinal Encoding
8. Mean Ordinal Encoding
9. Probability Ratio Encoding

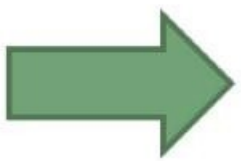
i) Label Encoding

- One of the simplest and most common solutions advertised to transform categorical variables is **Label Encoding**.
- It consists of substituting each group with a corresponding number and keeping such numbering consistent throughout the feature.

Categorical Feature	Label Encoding
United States	1
United States	1
France	2
Germany	3
United Kingdom	4
France	2

State	Confirmed	Recovered	Deaths
Punjab	2000	1500	200
Haryana	2321	1222	345
Kerala	3455	2365	400

State
Punjab
Haryana
Kerala



State
0
1
2

ii) Ordinal Encoding

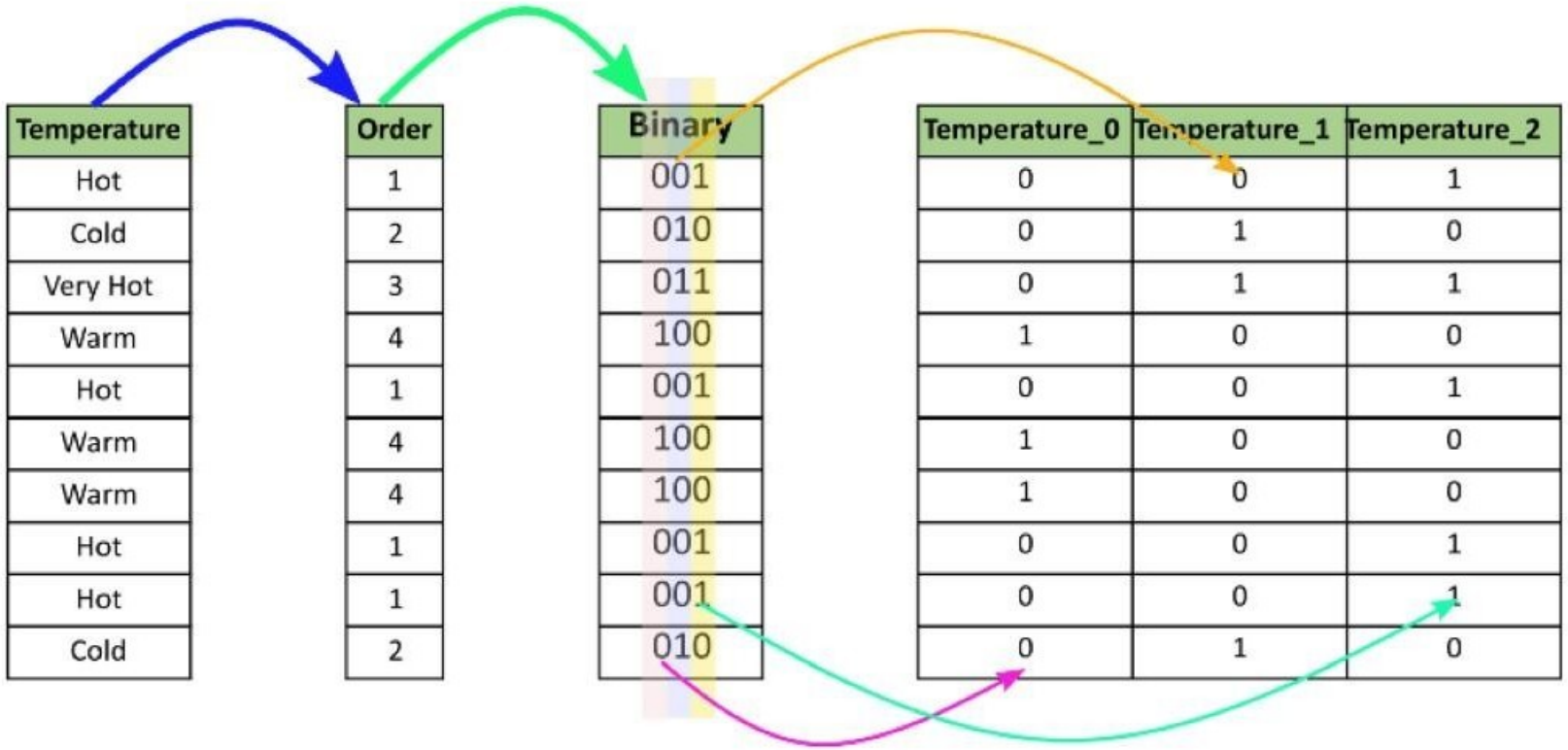
- Ordinal encoding's encoded variables retain the ordinal (ordered) nature of the variable.
- It looks similar to label encoding, the only difference being that label coding doesn't consider whether a variable is ordinal or not; it will then assign a sequence of integers.

	temperature
0	very cold
1	cold
2	warm
3	hot
4	very hot

	temperature	temp_ordinal
0	very cold	1
1	cold	2
2	warm	3
3	hot	4
4	very hot	5

iii) Binary encoding

- Binary encoding converts a category into binary digits. Each binary digit creates one feature column.



Temperature	Order	Binary
Hot	1	001
Cold	2	010
Very Hot	3	011
Warm	4	100
Hot	1	001
Warm	4	100
Warm	4	100
Hot	1	001
Hot	1	001
Cold	2	010

Temperature_0	Temperature_1	Temperature_2
0	0	1
0	1	0
0	1	1
1	0	0
0	0	1
1	0	0
1	0	0
0	0	1
0	0	1
0	1	0

iv) Frequency encoding

- The category is assigned as per the frequency of values in its total lot.

	class	data_fe
0	A	0.27
1	B	0.13
2	C	0.27
3	D	0.13
4	A	0.27
5	B	0.13
6	E	0.20
7	E	0.20
8	D	0.13
9	C	0.27
10	C	0.27
11	C	0.27
12	E	0.20
13	A	0.27
14	A	0.27

v) One Hot Encoding

- **One-Hot Encoding** is the most common, correct way to deal with non-ordinal categorical data.
- This technique is applied for nominal categorical features.
- In one Hot Encoding method, each category value is converted into a new column and assigned a value as 1 or 0 to the column.

Index	Animal
0	Dog
1	Cat
2	Sheep
3	Horse
4	Lion

One-Hot code

Index	Dog	Cat	Sheep	Lion	Horse
0	1	0	0	0	0
1	0	1	0	0	0
2	0	0	1	0	0
3	0	0	0	0	1
4	0	0	0	1	0

Exercise:

1. What do you mean by Association rules? Interpret {Diapers} → {Beer}
2. Mention the phases of Association rules mining.
3. What is frequent itemset? Give an example.
4. What is minimum support? Give its significance in association mining.
5. What is confidence? Write the formula to calculate confidence.
6. What is the importance of confidence threshold in association rules mining?
7. Mention the approaches for association rules mining.
8. What is the main disadvantage of Brute-Force method in association rules mining?
9. Write the algorithm for Apriori approach for association rules mining.
10. Write the steps for construction of FP tree.
11. Mention the purpose of FP tree.
12. Why FP tree is more advantageous than Apriori approach?
13. Why we need to handle categorical data in Data mining process?
14. Write the methods for handling categorical data in Data mining process.
15. What is One-hot encoding? Explain with a suitable example.
16. Why label encoding is not appropriate for handling categorical data?
17. How encoding is performed with Frequency encoding?

Exercise:

18. Use the Apriori algorithm using candidate generation for finding frequent itemset and then evaluate the valid association rules:

TID	Items
100	{a,b,c}
200	{b,c,d}
300	{a,c,e}
400	{b,d,f}
500	{a,b,e}
600	{b,c,f}

19. Construct the FP tree for the following transactions.

TID	List of items
T100	A,C,D
T200	B,C,E
T300	A,B,C,E
T400	B,E

Exam Questions:

1. Mention the purpose of FP tree. [BIM 2022, Group A]
2. What is the role of minimum support? [BIM 2021, Group A]
3. Write formula for confidence used in association rule mining. [BIM 2018, Group A]
4. Given the following transaction set, find the frequent itemset using Apriori algorithm. (Minimum support =2). [BIM 2022, Group B]

Transaction	Items
T1	{pasta, lemon, bread, orange}
T2	{pasta, lemon}
T3	{pasta, orange, cake}
T4	{pasta, lemon, orange, cake}

5. Consider the following transaction data sets. And construct the FP tree. [BIM 2021, Group B]

TID	Items
1	E, A, D, B
2	D, A, C, E, B
3	C, A, B, E
4	B, A, D
5	D
6	D, B
7	A, D, E
8	B, C

Exam Questions:

6. Construct the FP tree for the following transactions. [BIM 2018, Group B]

TID	Items
100	{a,b,c}
200	{b,c,d}
300	{a,c,e}
400	{b,d,f}
500	{a,b,e}
600	{b,c,f}

7. Use the Apriori algorithm using candidate generation for finding frequent itemset and then evaluate the valid association rules: [BIM 2017, Group C]

TID	List of items
T100	A,C,D
T200	B,C,E
T300	A,B,C,E
T400	B,E

End of chapter