UNIT - III

Association Rules

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 association among the variable of dataset. It is based on different rules to discover the interesting relations between variables in the database.

- 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 with in a shelf or mostly near by

Customer 1	Customer 2	Customer 3	Customer 4
milk	milk	milk	Sugar
bread	bread	bread	eggs
butter	butter	butter	
	Cerals		

For example: market Based Data

Transaction ID	Items
1	{milk, Bread, Rice, Book}
2	{Bread, Jam, Book, Pen}
3	{Jam, milk, bread, Rice, Eggs}
4	{Rice, Eggs, Pen, Book}
5	{Eggs, Pen, Milk, Bread, Jam}
6	{Eggs, Rice, Bread, Jam}

Let us consider one transaction like

{Milk, Bread, Rice, Book} {Milk} → {Bread}

{Bread, Jam, Book, Pen} {Book} → {Pen} {Bread} → {Jam}

Some similar associations

{Dishwash Liquid} → {scrubber}

{Laptop} → {Mouse}

{Floor stick} → {floor cleaner}

Itemset: {Milk, Bread, Jam, Rice, Eggs, Book, Pen}

TID	Items						
	Milk	Bread	Jam	Rice	Eggs	Books	Pen
1	1	1		1		1	
2		1	1			1	1
3	1	1	1	1	1		
4				1	1	1	1
5	1	1	1		1		1
6		1	1	1	1		

Frequent itemsets:-

Two itemsets: {Milk, Bread} {Bread, Jam} {Rice, Eggs} {Books, Pen}

Three itemsets: {Milk, Bread, Jam}, {Rice, Eggs, Bread} {Book, Pen, Eggs}

Four itemsets: {Milk, Bread, Rice, Eggs}

support:- It is a measure of how frequently a set of items occur in total no. of transactions.

{Milk, Bread} → {x, y} (x: Milk) (y: Bread)

Therefore the frequency of occurrence of x and y together in total no. of transactions is support.

{Milk, Bread, Jam} → {x, y} {x: Milk} (y: Bread, Jam)

Here the frequency of occurance of {Bread, Jam} with {Milk} in whole transaction is support

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

Confidence: It is a measure of how often items in y appears in transactions that contain x.

{milk, Bread, Jam} → {x, y} (x: milk) {y: Bread, Jam}

Therefore the frequency of occurance of x and y in all the transactions where x exists.

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

Association Rule Mining: Given a set of transactions T, the goal of association rule mining is to find all rules having

support ≥ minsup Threshold
Confidence ≥ minconf Threshold

Eg: Suppose minsup = 0.3
minconf = 0.6

Consider, {Rice, Eggs} → {x, y}

Then $\text{support}(s) = \frac{\sigma(x,y)}{N}$

$$\text{support}(s) = \frac{1+1+1}{6} = \frac{3}{6} = 0.5$$

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

$$= \frac{3}{4} = 0.75$$

TID	Items
1	{Milk, Bread, Rice, Book}
2	{Bread, Jam, Book, Pen}
3	{Jam, Milk, Bread, Rice, Eggs}
4	{Rice, Eggs, Pen, Book}
5	{Eggs, Pen, Milk, Bread, Jam}
6	{Eggs, Rice, Bread, Jam}

Association Rule Mining.

here support : 0.5 ≥ minsup (0.3)
Confidence: 0.75 ≥ minconf (0.6)

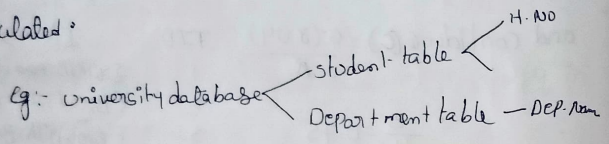
Therefore we can mine {Rice, Eggs} as a Rule.

Frequent Item set Generation:-

Association mining searches for frequent items in the data-set.

In frequent mining usually the interesting associations and correlations between itemsets in transactional and relational databases are found

Dataset:- A collection of related sets of information that is collected of separate elements but can be manipulated.



Frequent mining shows which items appear together in a transaction or relation.

Need of Association Mining:-

→ Frequent mining is generation of association rules from a transactional Dataset.

→ If there are 2 items x and y purchased frequently then its good to put them together in stores or provide some discount offer on one item on purchase of another item.

→ This can really increase the sales.

For example it is likely to find that if a customer buys Milk and bread he also buys butter.

Support :- $\frac{\sigma(A \cup B)}{\sigma N}$

Confidence: $\frac{\sigma(A \cup B)}{\sigma(A)}$

Support-count(A) : Number of transactions in which A appears If A is X or Y then it is the no. of transactions in which X and Y are present

Maximal Itemset:- An itemset is maximal frequent if none of its superset are frequent

Closed itemset:- An item set is closed if none of its immediate superset have same support count same as Itemset.

K-Itemset:- Item set which contains k items is a k-itemset

So it can be said that an itemset is frequent if the corresponding support count is ~~generated~~ greater than minimum support count.

Eg: Finding frequent items.

TransactionID	items
1	(A,C,D)
2	(B,C,D)
3	(A,B,C,D)
4	(B,D)
5	(A,B,C,D)

The APRIORI Principle :-

Apriori algorithm is a sequence of steps to be followed to find the most frequent itemset in the given database. Apriori algorithm is given by R.agarwal and R.srikant in 1994 for finding frequent itemsets in a dataset for boolean association rule. Name of the algorithm is Apriori because it uses prior knowledge of frequent itemset properties.

We apply an iterative approach or level-wise. Search where k-frequent itemsets are used to find k+1 itemsets.

To improve the efficiency of level wise Generation of frequent itemsets, an important property which helps by is called Apriori property which helps by reducing the search space.

Eg:

Transaction ID	List of items
1	I_1, I_2, I_5
2	I_2, I_4
3	I_2, I_3
4	I_1, I_2, I_4
5	I_1, I_3
6	I_2, I_3
7	I_1, I_3, I_5
8	I_1, I_2, I_3, I_5
9	I_1, I_2, I_3

minimum support count is 2.
minimum confidence is 60%.

Step 1 : K = 1

(i) Create a table containing support count of each item present in dataset called C_1 (Candidate set)

Itemset	Support count
I_1	6
I_2	7
I_3	6
I_4	2
I_5	2

(ii) Compare Candidate set items support count with minimum support count (here min-support = 2) if support-count of Candidate set items is less than min-support then remove those items. This gives us itemset L_1.

itemset	sup-count
I_1	6
I_2	7
I_3	6
I_4	2
I_5	2

L_1

Step 2 : K = 2

Generate Candidate set C_2 using L_1

itemset	sup-count
I_1, I_2	4
I_1, I_3	4
I_1, I_4	1
I_1, I_5	2
I_2, I_3	4
I_2, I_4	2
I_2, I_5	2
I_3, I_4	0
I_3, I_5	1
I_4, I_5	0

C_2

(ii) Compare Candidate (C_2) support count with minimum support count ($^{sup\ count}_{= 2}$) if support count of Candidate set item is less than min-support then remove those items) This gives us itemset L_2

itemset	sup-count
I_1, I_2	4
I_1, I_3	4
I_1, I_5	2
I_2, I_3	4
I_2, I_4	2
I_2, I_5	2

L_2

Step-3 :- Generate Candidate set C_3 using L_2

itemset	sup-count
I_1, I_2, I_3	2
I_1, I_2, I_5	2
I_1, I_3, I_5	1
I_1, I_2, I_4	1

Compare Candidate set items with min-sup-count: 2, if support count is less than min-support count remove those items. This gives L_3

$\{I_1, I_2, I_3\} = \{I_1, I_2\}\{I_1, I_3\}\{I_2, I_3\}$

In L_2 these sets are present

$\{I_1, I_2, I_5\} = \{I_1, I_2\}\{I_1, I_5\}\{I_2, I_5\}$

In L_2 these sets are present

Confidence:-

A confidence of 60% means that 60% of the customers, who purchased m

Confidence:-

A confidence of 60% means that 60% of the customers, who Purchased milk and bread also bought butter.

confidence $(A \rightarrow B)$ = support_count $(A \cup B)$ / support_count(A)

So here by taking an example of any frequent itemset, we will show the rule generation

Itemset $\{I_1, I_2, I_3\}$ from L_3

so rules can be

$[I_1 \wedge I_2] \Rightarrow I_3$ confidence =

$\sup[I_1 \wedge I_2 \wedge I_3] / \sup[I_1 \wedge I_2] = 2/4 * 100 = 0.50 = 50\%$

$[I_1 \wedge I_3] \Rightarrow [I_2]$

$\sup[I_1 \wedge I_2 \wedge I_3] / \sup[I_1 \wedge I_3] = \frac{2}{4} * 100 = 50\%$

$[I_2 \wedge I_3] \Rightarrow [I_1]$

$\sup[I_1 \wedge I_2 \wedge I_3] / \sup[I_2 \wedge I_3] = 2/4 * 100 = 50\%$

$[I_1] \Rightarrow [I_2 \wedge I_3]$

$\sup(I_1 \wedge I_2 \wedge I_3) / \sup[I_1] = 2/6 * 100 = 33\%$

$[I_2] \Rightarrow [I_1 \wedge I_3]$

$\sup[I_1 \wedge I_2 \wedge I_3] / \sup(I_2) = 2/7 * 100 = 28\%$

$[I_3] \Rightarrow [I_1 \wedge I_2]$

$\sup(I_1 \wedge I_2 \wedge I_3] / \sup(I_3) = 2/6 * 100 = 33\%$

so if minimum confidence is 50%. Then first 3 rules can be considered as strong association rules.

Another Example: minimum support count = 2

T_{ID}	List of items
T_{10}	I_1, I_3, I_4
T_{20}	I_2, I_3, I_5
T_{30}	I_1, I_2, I_3, I_5
T_{40}	I_2, I_5

The Partition Algorithms :-

This clustering method classifies the information into multiple groups based on the characteristics and similarity of the data. It is the data analysts to specify the number of clusters that has take generated for the clustering methods

In the Partition method when Database (D) that contains multiple (N) objects then the Partitioning methods construct user specified (K) Partitions of the data in which each Partition represents a cluster and a particular region. There are many algorithms that come under Partitioning methods some of the popular ones are K-Mean, PAM (K-Mediods), CLARA algorithm (Cluster large applications) etc.

K-Mean Clustering algorithm :-

The K-mean algorithm takes the input Parameter K from the user and Partitions the data set containing N objects into K clusters so that the resulting similarity among the data objects inside the group (intracluster) is high but the similarity of data objects with the data objects from the outside the cluster is low (inter cluster)

The similarity of the cluster is determined with respect to the mean value of the cluster

In K-mean clustering algorithm we should follow some steps

step 1: Take the Mean value.

step 2: Find the nearest no.of mean and Put in Cluster.

step 3 Repeat ① & ② until we get the same Mean

Example :-

Data Pth, K = {2, 4, 6, 9, 18, 16, 20, 24, 26}

NO.of clusters = 2.

Take two Random numbers as clusters (centroids) = {4, 12}

$K_1 = \{2, 4, 6\}$ $K_2 = \{9, 12, 16, 20, 24, 26\}$

Meanvalue for $K_1 = \frac{2+4+6}{3} = 4$ Meanvalue for $K_2 = \frac{9+12+16+20+24+26}{6}$

$K_2 = 18\ (17.8)$

$k_1 = \{2, 4, 6, 9\}$ $K_2 = \{12, 16, 20, 24, 26\}$

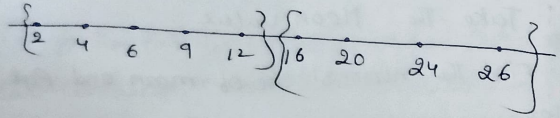
Mean $k_1 = \frac{21}{4} = 5.25 = 5$ Mean $K_2 = 19.6 = 20$

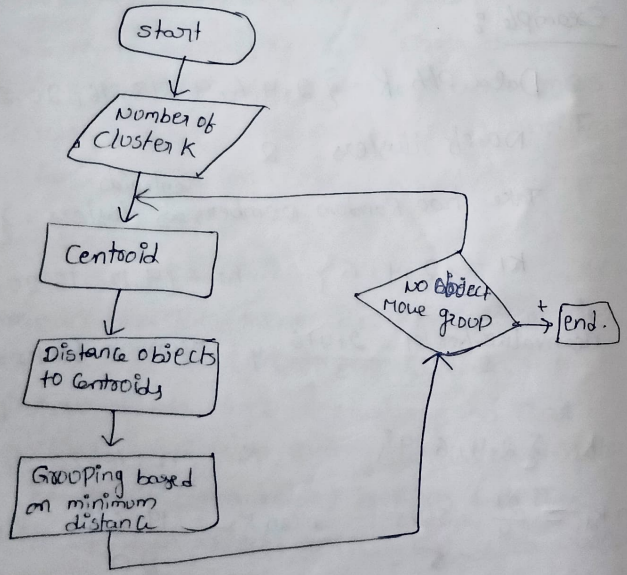
$K_1 = \{2, 4, 6, 9, 12\}$ $K_2 = \{16, 20, 24, 26\}$

Meanvalue $K_1 = 6.6$ $K_2 = 21.5 = 22$

$K_1 = \{2, 4, 6, 9, 12\}$ $K_2 = \{16, 20, 24, 26\}$

Mean Value K_1 = 6.6 : 7 Mean Value K_2 = 21.5 : 22





Divide the given sample data in two (2) clusters using k-means algorithm [Euclidean distance]

$$\sqrt{(x_H - H_1)^2 + (x_W - W_1)^2}$$

x_H → observed value, H_1 → centroid value, x_W → observed value, W_1 → centroid value

	Height	weight
1	185	72
2	170	56
3	168	60
4	179	68
5	182	72
6	188	77
7	180	71
8	180	70
9	183	84
10	180	88
11	180	67
12	177	76

FP-Growth algorithms:-

This Algorithm is an improvement to the Apriori Algorithm. A Frequent pattern (FP) 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 itemset 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 itemsets are maintained while forming the tree.

Frequent Pattern Algorithm steps:-

The Frequent pattern growth method lets find the frequent pattern without candidate generation.

1. The First step is to scan the database to find the occurrences of the itemsets in the database. This step is the same as the First step of Apriori. The count of one itemsets in the database is called support count or frequency of 1-itemset.
2. The second step is to construct the FP tree For this, Create the root of the tree. The root is represented by NULL
3. The next step is to scan the database again and examine the transactions. Examine the First transaction and find out the itemset in it. The itemset with the max count is taken at the top. The next itemset with lower count and so on. It means that the branch of the tree is constructed with transaction itemsets in descending order of count.

4) The next transaction in the database is examined. The itemsets are ordered in descending order of count. If any itemset of this transaction is already present in another branch then this transaction branch would share a common prefix to the root

This means that the common itemset is linked to the new node of another itemset in this transaction.

5) Also, the count of the itemset incremented as it occurs in the transactions. Both the common node and new node count is increased by 1 as they are created and linked according to transactions.

6) The next step is to mine the created FP tree. For this, the lowest node is examined first along with the links of the lowest nodes. The lowest node represents the frequency pattern length 1. From this traverse the path in the FP Tree. This path or paths called a Conditional Pattern base. Conditional Pattern base is a sub-database consisting of prefix paths in the FP tree occuring with the lowest node.

7. Construct a Conditional FP tree, which is formed by a count of itemsets in the path. The itemsets meeting the threshold support are considered in the Conditional FP tree.

8. Frequent Patterns are generated from the Conditional FP tree.

example:

FP-Tree for the following transaction Data set [minimum support = 30%.]

minimum no. of transactions = 0.3*8 = 2.4 ≈ 3

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

items	Frequency	priority
A	5	3
B	6	1
C	3	5
D	6	2
E	4	4

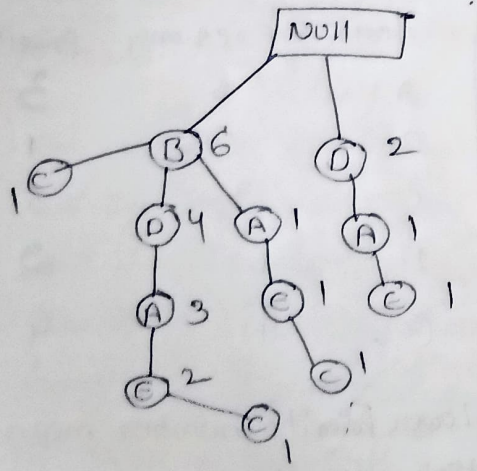
Lower priority number means higher priority.

order the items according to priority

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

FP-tree :-

In the FP tree the Root is always null



Frequency

B: 1, 2, 3, 4, 5, 6
D: 1, 2, 3, 4
A: 1, 2, 3
E: 1, 2
C: 1

A: 1
E: 1
C: 1 → 3

D: 1, 2
A: 1
E: 1
C: 1

Another Example :-

Transaction ID	List of items
T_1	I_1, I_2, I_3
T_2	I_2, I_3, I_4
T_3	I_4, I_5
T_4	I_1, I_2, I_4
T_5	I_1, I_2, I_3, I_5
T_6	I_1, I_2, I_3, I_4

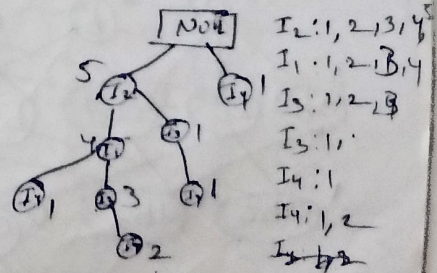
Support Threshold = 50%

minimum support = $0.5 \times 6 = 3.0$

items	Frequency	Priority
I_1	4	2
I_2	5	1
I_3	4	3
I_4	4	4
I_5	2	5

→ Remove because Count < Min-support

TID	items	ordered items
T_1	I_1, I_2, I_3	I_2, I_1, I_3
T_2	I_2, I_3, I_4	I_2, I_3, I_4
T_3	I_4, I_5	I_4
T_4	I_1, I_2, I_4	I_2, I_1, I_4
T_5	I_1, I_2, I_3, I_5	I_2, I_1, I_3
T_6	I_1, I_2, I_3, I_4	I_2, I_1, I_3, I_4



Compact Representation of Frequent item set:-

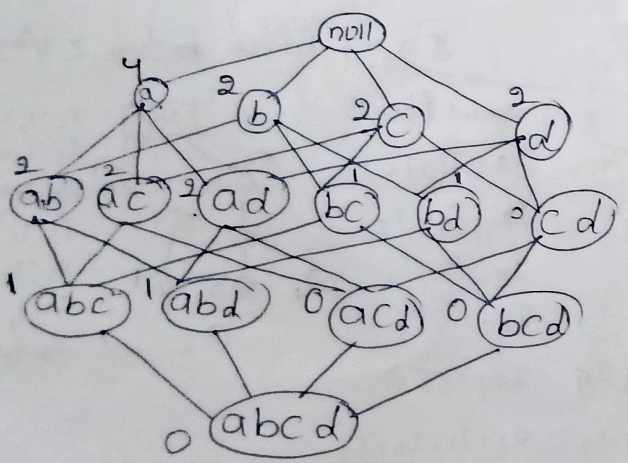
The no of frequent itemsets produced from a transaction dataset can be very large. It is useful to identify a small representative set of itemsets from which all other frequent itemsets can be derived. In this we have two types.

(i) maximal frequent itemset

A maximal frequent itemset is defined as a frequent itemset for which none of its immediate supersets are frequent.

eg:-

TID	Items
1	a, b, d
2	a, c
3	a, b, c
4	a, d



The support counts are shown on the top left of each node. Assume support count threshold = 50%. That is each itemset must occur in 2 or more transactions. Based on that threshold. The frequent itemsets are a, b, c, d, ab, ac, and ad.

out of the 7 frequent itemsets 3 are identified as maximal frequent itemsets.

ab: immediate supersets abc and abd are infrequent

ac: immediate superset abc and acd are infrequent.

ad: immediate supersets abd and acd are infrequent.

The remaining 4 frequent nodes (a, b, c and d) cannot be maximal frequent because they all have at least 1 immediate superset that is frequent.

Advantages:-

Maximal frequent itemsets provide a compact representation of all the frequent itemsets for a particular dataset.

Disadvantages:-

The support count of maximal frequent itemsets does not provide any information about the support count of their subsets. This means that an additional traversal of data is needed to determine support count for non-maximal frequent itemsets.

Frequent itemset:-

A frequent itemset is an itemset whose support is greater than some user-specified minimum support

Closed Frequent itemset:-

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

maximal frequent itemset:-

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

Need of closed and maximal itemsets:-

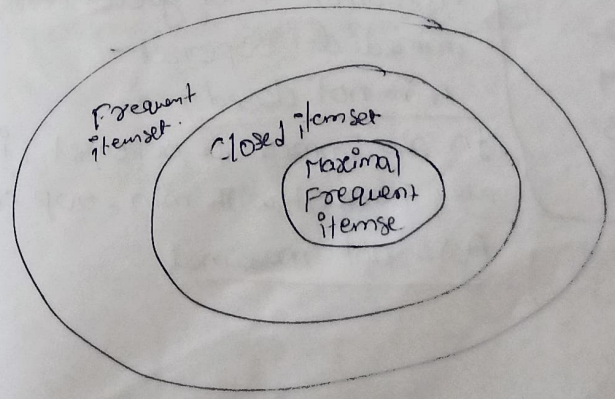
→ These closed frequent itemset and maximal frequent itemsets are useful when huge amount of data is used in association rule mining

→ If the length of frequent itemset is 'k' then by downward closure property, all of it 2^k subsets are also frequent

→ when the computation is very expensive there is no interest to find additional subsets. This can be avoided by frequent itemset with maximum length.

→ one disadvantage with maximal frequent itemsets is that even all its subsets are frequent, we do not know their supports. For mining rules, support information is very important.

→ Therefore closed frequent itemset is prefered.



eg:

TID	List of items
T_1	A, B, C, D
T_2	A, B, C, D
T_3	A, B, C
T_4	B, C, D
T_5	C, D

Min. support count = 3

find frequent, closed and maximal itemset.

item	count
A	3
B	4
C	5
D	4
.	

All items that is A, B, C and D are frequent because they support count is greater than or equal to minimum sup. count.

item	count
A, B	3
A, C	3
A, D	2
B, C	4
B, D	3
C, D	4

All 2 itemsets are frequent, except A, D.

A(3) → AB(3), AC(3), AD(2)

A(count) is not greater than its immediate superset

A is not closed.

In A's immediate superset, itemsets are present with min. sup count i.e 3

A is not maximal.

B(4) → AB(3), BC(4), BD(3)

B(count) is not greater than its immediate superset

B is not closed.

In B's immediate superset, itemset are present with min-sup-count 3.

B is not maximal.

C(5) → AC(3), BC(4), CD(2)

C(count) is greter than its immediate supersets.

C is closed.

In C's immediate superset, itemsets are present with minimum support count 3.

C is not maximal.

D(4) → AD(2), BD(3), CD(4)

D(count) is not greter than its immediate superset

D is not closed.

In D's immediate superset, itemset are present with min. sup. count 3.

D is not maximal.

Item	count
A,B,C	3
A,B,D	2
A,C,D	2
B,C,D	3

AB(3) → ABC(3), ABD(2)
AB(count) is not greter its imedate supset
AB is not closed
AB s immediate superset, itemset are present with min. support count 3
AB is not maximal.

AC(3) → ABC(3), ACD(2)
AC is not closed
AC is not maximal.

AD(2) → ABD(2), ADC(2)
AD not frequent

BC(4) → ABC(3), BCD(3)
BC is closed
BC is not maximal

BD(3) → ABD(2), BCD(3)
BD is not closed
BD is not maximal.

CD(4) → ACD(2), BCD(3)
CD is closed
CD is not maximal

item	count
A,B,C,D	2

ABC(3) → ABCD(2)
ABC is closed
ABC is maximal.
ABD(2) → ABCD(2) → not frequent.
ACD not frequent
so not maximal.

BCD(3) → ABCD(2)
BCD is closed
BCD is maximal.

A,B,C,D → 2 it is not frequent

1-item set

item	Freq	closed	Maximal
A(3)	✓	x	x
B(4)	✓	x	x
C(5)	✓	✓	x
D(4)	✓	x	x

2-item set

item	Freq	closed	Max
AB(3)	✓	x	x
AC(3)	✓	x	x
AD(2)	x	x	x
BC(4)	✓	✓	x
BD(3)	✓	x	x
CD(4)	✓	✓	x

3-item set

item	Freq	closed	Maximal
ABC(3)	✓	✓	✓
ABD(2)	x	x	x
ACD(2)	x	x	x
BCD(3)	✓	✓	✓

4 item set

item	Freq	closed	Maxim
ABCD(2)	x	x	x

TID	List of items
T_1	B, J, P
T_2	B, P
T_3	B, M, P
T_4	E, B
T_5	E, M

Min-sup count = 40%

find frequent, closed, maximal itemsets.