DATA MINING REVIEW

Frequent Itemset:

A set of items is called frequent if it satisfies a minimum threshold value for support and confidence.
Support shows transactions with items purchased together in a single transaction.
Confidence shows transactions where the items are purchased one after the other.

Problem1: Given, Support threshold =50% and 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

a) Find Min_Sup

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

b) Find count of each item:

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

c) Check which items do not meet Min_Sup and eliminate them:

Item	Count
I1	4
I2	5
I3	4
I4	4

d) Find count of 2-itemsets:

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

e) Check which items do not meet Min_Sup and eliminate them:

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

f) Find the 3-itemsets:

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

g) Subsets of the above items:

{I1, I2, I3} => {I1, I2}, {I1, I3}, {I2, I3}
{I1, I2, I4} => {I1, I2}, {I1, I4}, {I2, I4}
{I1, I3, I4} => {I1, I3}, {I1, I4}, {I3, I4}
{I2, I3, I4} => {I2, I3}, {I2, I4}, {I3, I4}

h) Check for occurrence of 2-itemsets:

{I1, I2, I3} => {I1, I2}, {I1, I3}, {I2, I3} => All items meet Min_Sup => Frequent

{I1, I2, I4} => {I1, I2}, {I1, I4}, {I2, I4} => {I1, I4} does not meet Min_Sup => Not Frequent

{I1, I3, I4} => {I1, I3}, {I1, I4}, {I3, I4} => {I1, I4} does not meet Min_Sup => Not Frequent

{I2, I3, I4} => {I2, I3}, {I2, I4}, {I3, I4} => {I3, I4} does not meet Min_Sup => Not Frequent

g) Therefore, the only frequent itemset is {I1, I2, I3}

If you are asked to generate rules for the frequent itemset:

{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%

Since, given minimum confidence threshold is 60%, all the above association rules are strong.

Shortcomings of Apriori:

Using Apriori needs a generation of candidate item sets. These item sets may be large in number if the itemset in the database is huge.

Apriori needs multiple scans of the database to check the support of each itemset generated and this leads to high costs.

These shortcomings can be overcome using the **FP growth algorithm**.

Problem2: Given, Support threshold =50% and 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

a) Find Min_Sup

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

b) Find count of each item:

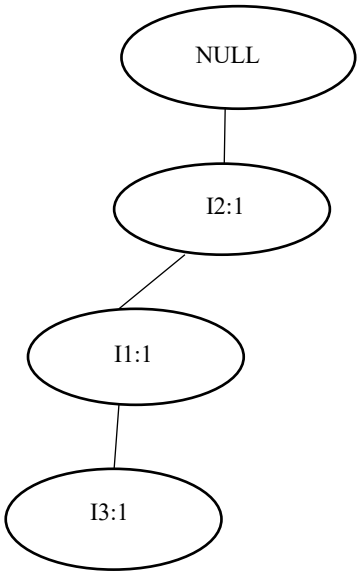
Item	Count
I1	4
I2	5
I3	4
I4	4
I5	2

c) Check which items do not meet Min_Sup and eliminate them:

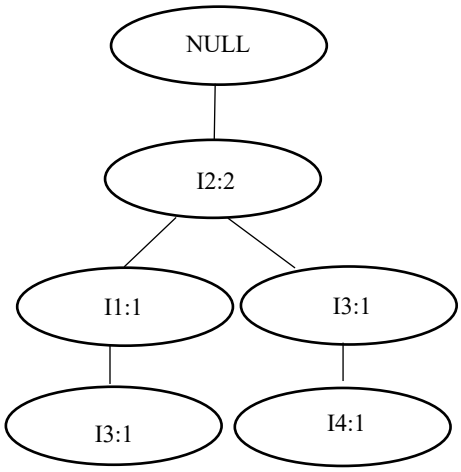
Item	Count
I1	4
I2	5
I3	4
I4	4

d) We start drawing the FP-tree as below:

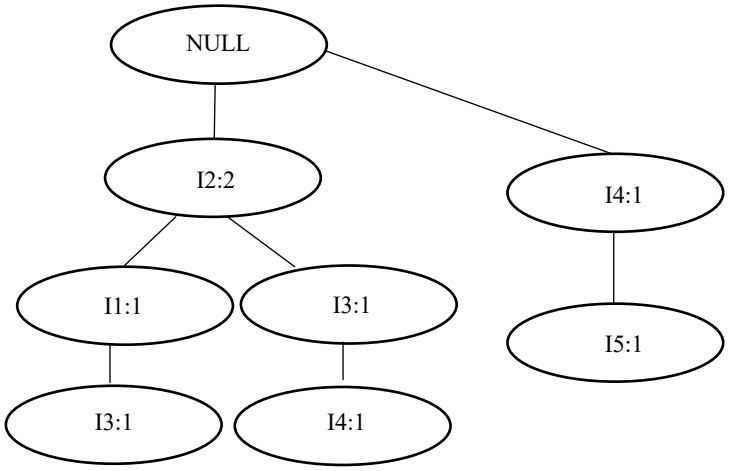
Consider the root node null and the first scan of Transaction
T1: {I1, I2, I3} contains three items {I1:1}, {I2:1}, {I3:1}, where I2 is linked as a child to root, I1 is linked to I2 and I3 is linked to I1.



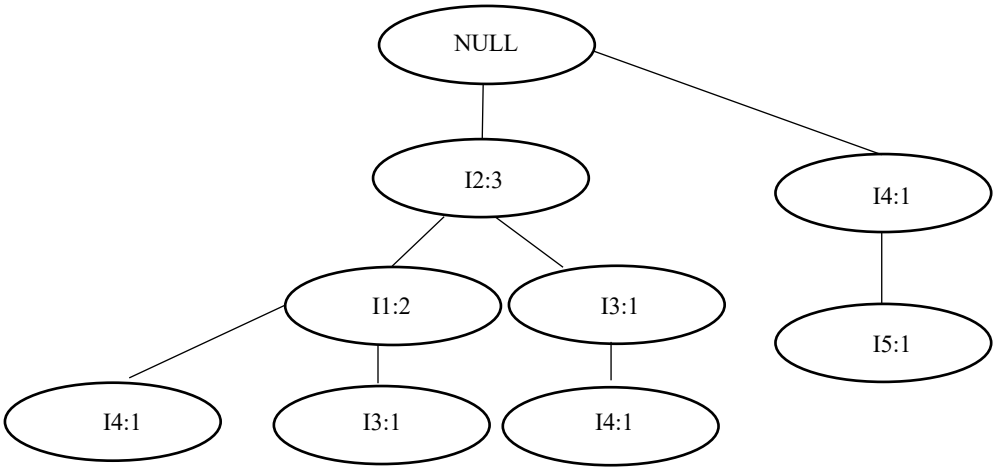
T2: {I2, I3, I4} contains I2, I3, and I4, where I2 is linked to root, I3 is linked to I2 and I4 is linked to I3. But this branch would share I2 node as common as it is already used in T1.



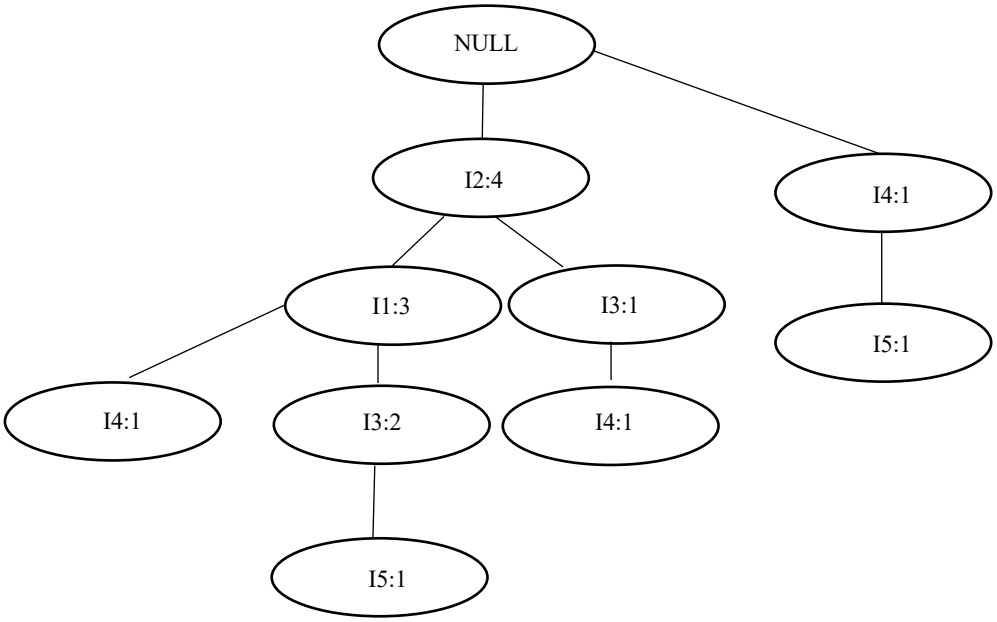
T3: {I4, I5}, a new branch with I5 is linked to I4 as a child is created.



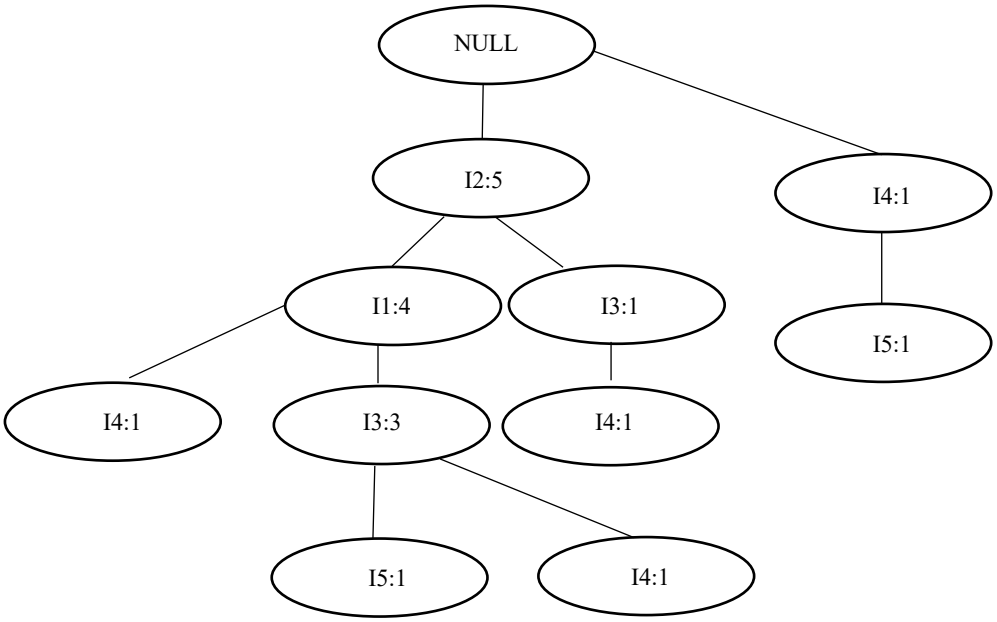
T4: {I1, I2, I4}. The sequence will be I2, I1, and I4. I2 is already linked to the root node, hence it will be incremented by 1. Similarly, I1 will be incremented by 1 as it is already linked with I2 in T1, thus {I2:3}, {I1:2}, {I4:1}.



T5: {I1, I2, I3, I5}. The sequence will be I2, I1, I3, and I5. Thus {I2:4}, {I1:3}, {I3:2}, {I5:1}.



T6: I1, I2, I3, I4. The sequence will be I2, I1, I3, and I4. Thus {I2:5}, {I1:4}, {I3:3}, {I4 1}.



Conditional Pattern Base:

I3: {I2, I1:3}, {I2:1}

I1: {I2:4}

Conditional FP tree:

I3: {I2:4}, {I1:3}

I1: {I2:4}

Frequent Patterns Generated:

I3: {I2, I3:4}, {I1, I3:3}, {I2, I1, I3:3}

I1: {I2, I1:4}

CLUSTERING:

Problem 3:

Using the following dataset:

Object	**Weight(X)**	**pH(Y)**
Medicine A	1	1
Medicine B	2	1
Medicine C	4	3
Medicine D	5	4

Cluster these objects into k=2 groups using Medicine A and Medicine B as initial centroids, Euclidean as the distance and K-Means as clustering algorithm.

Sample questions:

Which objects are re-assigned after 1st iteration?
How many iterations to find k clusters?
Which objects are in the final cluster?

Solution:

For First Iteration:

Consider given centroids:
C1 => Medicine A – (1,1)
C2 => Medicine B – (2,1)

Calculate Euclidean distance from each datapoint to each centroid:

X	**Y**
(A, C1) = 0	(A, C2) = 1
(B, C1) = 1	(B, C2) = 0
(C, C1) = 3.6	(C, C2) = 2.8
(D, C1) = 5	(D, C2) = 4.2

If X>Y push to one cluster and if Y>X put in another cluster:

So, in this case A will be in one cluster and B, C, D will be in another cluster.
Say Cluster1(A), Cluster2(B, C, D)

For Second Iteration:

C1 => Medicine A – (1,1)
C2 => Medicine B, Medicine C, Medicine D – [(2,1), (4,3), (5,4)] = [2+4+5/3, 1+3+4/3] = [11/3,8/3]

Calculate Euclidean distance from each datapoint to each centroid:

X	Y
(A, C1) = 0	(A, C2) = 3.14
(B, C1) = 1	(B, C2) = 2.36
(C, C1) = 3.6	(C, C2) = 0.47
(D, C1) = 5	(D, C2) = 1.89

So, in this case A, B will be in one cluster and C, D will be in another cluster.
Say Cluster1(A, B), Cluster2(C, D)

For Third Iteration:

C1 => Medicine A, Medicine B – [(1,1), (2,1)] = [1+2/2, 1+1/2] = [3/2,1]
C2 => Medicine C, Medicine D – [(4,3), (5,4)] = [4+5/2, 3+4/2] = [9/2,7/2]

Calculate Euclidean distance from each datapoint to each centroid:

X	Y
(A, C1) = 0.5	(A, C2) = 4.3
(B, C1) = 0.5	(B, C2) = 3.5
(C, C1) = 3.2	(C, C2) = 0.7
(D, C1) = 4.6	(D, C2) = 0.7

So, in this case A, B will be in one cluster and C, D will be in another cluster.
Say Cluster1(A, B), Cluster2(C, D)

At this point, there is no possibility of further clustering. So, we stop the process and consider final clusters as:
Cluster1(A, B), Cluster2(C, D)

Problem 4:

Generate all possible rules:

Transaction ID	Onion	Potato	Burger	Milk	Beer
T1	1	1	1	0	0
T2	0	1	1	1	0
T3	0	0	0	1	0
T4	1	1	0	1	0
T5	1	1	1	0	1
T6	1	1	1	1	1

Checking the frequency of each item:

Item	Frequency
Onion	4
Potato	5
Burger	4
Milk	4
Beer	2

Only those elements whose support is greater than or equal to support threshold are significant.
Considering support threshold to be 50%, Beer can be eliminated.

Item	Frequency
Onion(O)	4
Potato(P)	5
Burger (B)	4
Milk(M)	4

Count the frequency of pair combinations:

Itemset	Frequency
OP	4
OB	3
OM	2
PB	4
PM	3
BM	2

Again BM, OM can be eliminated as it is less than support threshold. OP, OB, PB and PM are significant.

Count the frequency of 3 combinations:

Itemset	Frequency
OPB	4
PBM	3

The set of three items purchased most frequently is OPB.

And the rules can be generated from most frequent itemset as below:

OP -> B
OB -> P
PB - > O
B -> OP
P - > OB
O -> PB

Problem 5:

Draw a boxplot for the below data (Calculate Q1, Q2, Q3, IQR, Extreme Values and Outliers if any):

90, 94, 53, 68, 79, 84, 87, 72, 70, 69, 65, 89, 85, 83, 72

First, Arrange the data in ascending order:

53, 65, 68, 69, 70, 72, 72, 79, 83, 84, 85, 87, 89, 90, 94

Min=53
Q2 = 79 (Median)
Q1 = 69
Q3 = 87
Max=94
IQR = Q3-Q1 = 87-69= 18

Outlier range =[(Q1-1.5*IQR), (Q3+1.5*IQR)] = [42, 114]
There is no data below 42 and above 114. So, no outliers