Data warehouse and Data Mining (TDT 4300)

Assignment 2

Submitted by

Name : Md Anwarul Hasan
Student ID : 583233

Task 1 Apriori Algorithm

Given the data in Table 1 (market basket transaction), your task is to describe the purchasing behavior of customers in the form of association rules. First, you need to generate the frequent itemsets and second, you need to generate the association rules. Apply the Apriori algorithm for following tasks and describe thoroughly the process and the outcome of each step.

(a) Generate frequent 2-, 3- and 4-itemsets using the $F_{k-1} \times F_{k-1}$ method. Consider the support threshold minsup = 0.5 and use the data presented in Table 1.

TID	**Items**
110	A, C, F, G, H
111	B, C, D, E, G
112	B, C, E, F, H
113	A, B, C, G
114	C, D, E, H
115	A, B, C, G, H
116	A, B, C, D, G, H
117	B, C, E, G
118	A, B, C, F, G, H
119	A, B, C, D, E, G, H

Table 1: Market basket transactions

Answer:

First we generate single item dataset with support,

1.

C_1	Support
A	6
B	8
C	10
D	4
E	5
F	3
G	8
H	7

L_1	Support
A	6
B	8
C	10
E	5
G	8
H	7

2.

C_2	Support
AB	5
AC	6
AE	1
AG	5
AH	5
BC	8
BE	4
BG	7
BH	5
CE	5
CG	8
CH	7
EG	3
EH	3
GH	5

C_2	Support
AB	5
AC	6
AG	6
AH	5
BC	8
BG	7
BH	5
CE	5
CG	8
CH	7
GH	5

3. using the $F_{k-1} \times F_{k-1}$ method for 3 item candidate generation,

C_2	Support
ABC	5
ABG	5
ABH	4
ACG	6
ACH	5
AGH	5
BCG	7
BCH	5
BGH	4
CEG	3
CEH	3
CGH	5

C_2	Support
ABC	5
ABG	5
ACG	6
ACH	5
AGH	5
BCG	7
BCH	5
CGH	5

4. Again using the $F_{k-1} \times F_{k-1}$ method for 4 item candidate generation,

C_2	Support
ABCG	5
ACGH	5
BCGH	4

C_2	Support
ABCG	5
ACGH	5

So generation of association rules for itemset ABCG:

i) frequent itemset: ABCG → {ABC, ABG, ACG, BCG}

Association rule	Support
ABC → G	1.0
ABG → C	1.0
ACG → B	0.83
BCG → A	0.71

Association rule	Support	Type
ABC → G	1.0	Strong
ABG → C	1.0	Strong
ACG → B	0.83	Strong
BCG → A	0.71	Weak

ii) frequent itemset: ABC → {AB, AC, BC}

Association rule	Support
AB → C	1.0
AC → B	0.83
BC → A	0.62

Association rule	Support	Type
AB → C	1.0	Strong
AC → B	0.83	Strong
BC → A	0.62	Weak

iii) frequent itemset: ABG → {AB, AG, BG}

Association rule	Support
AB → G	1.0
AG → B	1.0
BG → A	0.71

Association rule	Support	Type
AB → G	1.0	Strong
AG → B	1.0	Strong
BG → A	0.71	Weak

iv) frequent itemset: ACG → {AC, AG, CG}

Association rule	Support
AC → G	1.0
AG → C	1.0
CG → A	0.75

Association rule	Support	Type
AC → G	1.0	Strong
AG → C	1.0	Strong
CG → A	0.75	Weak

v) frequent itemset: AB → {A, B}

Association rule	Support
A → B	0.83
B → A	0.62

Association rule	Support	Type
A → B	0.83	Strong
B → A	0.62	Weak

vi) frequent itemset: AC → {A, C}

Association rule	Support
A → C	1.0
C → A	0.6

Association rule	Support	Type
A → C	1.0	Strong
C → A	0.6	Weak

vii) frequent itemset: AG → {A, G}

Association rule	Support
A → G	1.0
G → A	0.75

Association rule	Support	Type
A → G	1.0	Strong
G → A	0.75	Weak

So generation of association rules for itemset ACGH:

i) frequent itemset: ACGH → {ACG, ACH, AGH, CGH}

Association rule	Support
ACG → H	0.83
ACH → G	1.0
AGH → C	1.0
CGH → A	1.0

Association rule	Support	Type
ACG → H	0.83	Strong
ACH → G	1.0	Strong
AGH → C	1.0	Strong
CGH → A	1.0	Strong

ii) frequent itemset: ACH → {AC, AH, CH}

Association rule	Support
AC → H	0.83
AH → C	1.0
CH → A	0.71

Association rule	Support	Type
AC → H	0.83	Strong
AH → C	1.0	Strong
CH → A	0.71	Weak

iii) frequent itemset: AGH → {AG, AH, GH}

Association rule	Support
AG → H	0.83
AH → G	1.0
GH → A	1.0

Association rule	Support	Type
AG → H	0.83	Strong
AH → G	1.0	Strong
GH → A	1.0	Strong

iv) frequent itemset: CGH → {CG, CH, GH}

Association rule	Support
CG → H	0.62
CH → G	0.71
GH → C	1.0

Association rule	Support	Type
CG → H	0.62	Weak
CH → G	0.71	Weak
GH → C	1.0	Strong

v) frequent itemset: AH → {A, H}

Association rule	Support
A → H	0.83
H → A	0.71

Association rule	Support	Type
A → H	0.83	Strong
H → A	0.71	Weak

vi) frequent itemset: GH → {G, H}

Association rule	Support
G → H	0.62
H → G	0.71

Association rule	Support	Type
G → H	0.62	Weak
H → G	0.71	Weak

Task 2 FP-Growth Algorithm

Use the Frequent Pattern Growth algorithm to discover the frequent itemsets in the given transaction dataset (Table 1). Consider the support threshold minsup = 0.5. Construct an FP-tree and mine the frequent itemsets by creating conditional (sub-)pattern bases. Use the table notation with columns: item, conditional pattern base, conditional FP-tree, frequent patterns generated. The recursive steps of the FP-Growth algorithm must be clearly captured using the aforementioned table notation. Sort items alphabetically in case of ties in the item support. Describe thoroughly the process and the outcome of each step.

Answer:

Vertical database representation for easy counting.

	110	111	112	113	114	115	116	117	118	119
A	1	0	0	1	0	1	1	0	1	1
B	0	1	1	1	0	1	1	1	1	1
C	1	1	1	1	1	1	1	1	1	1
D	0	1	0	0	1	0	1	0	0	1
E	0	1	1	0	1	0	0	1	0	1
F	1	0	1	0	0	0	0	0	1	0
G	1	1	0	1	0	1	1	1	1	1
H	1	0	1	0	1	1	1	0	1	1

1-itemset support values.

1-Itemset	Support
A	6
B	8
C	10
D	4
E	5
F	3
G	8
H	7

1-itemset reordered based on support values.

1-Itemset	Support
C	10
B	8
G	8
H	7
A	6
E	5
D	4
F	3

Reordered transaction database

TID	**Items**
110	C, G, H, A, F
111	C, B, G, E, D
112	C, B, H, E, F
113	C, B, G, A
114	C, H, E, D
115	C, B, G, H, A
116	C, B, G, H, A, D
117	C, B, G, E
118	C, B, G, H, A, F
119	C, B, G, H, A, E, D

FP-Tree for the entire transaction database

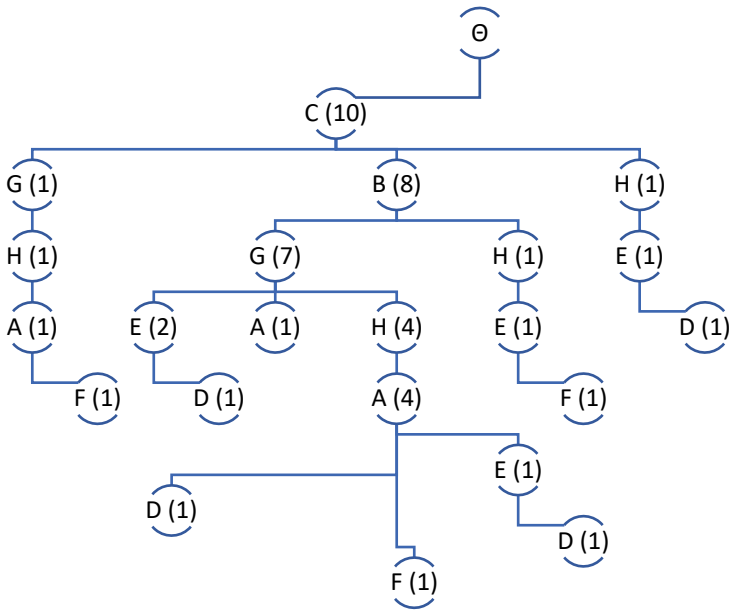


Fig: FP Tree

Focus on F:

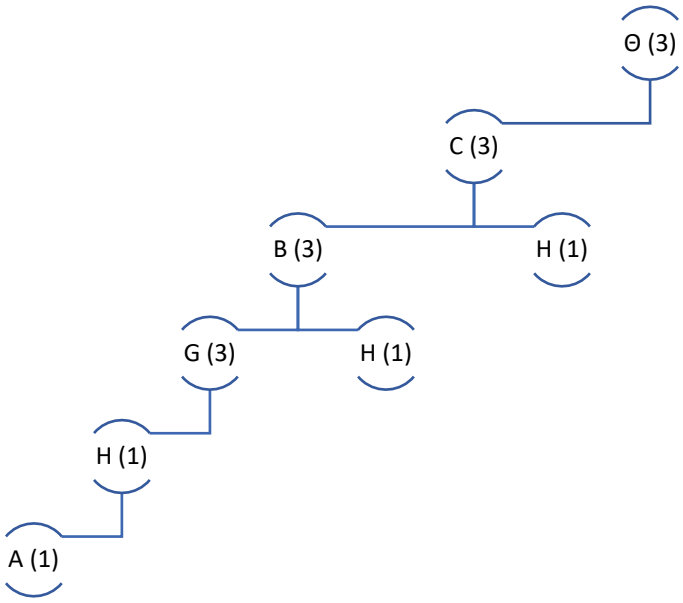
We can remove F from the FP-Tree as its support (3) is less than min. support (5)

Focus on D:

We can remove D from the FP-Tree as its support (4) is less than min. support (5)

Focus on E:

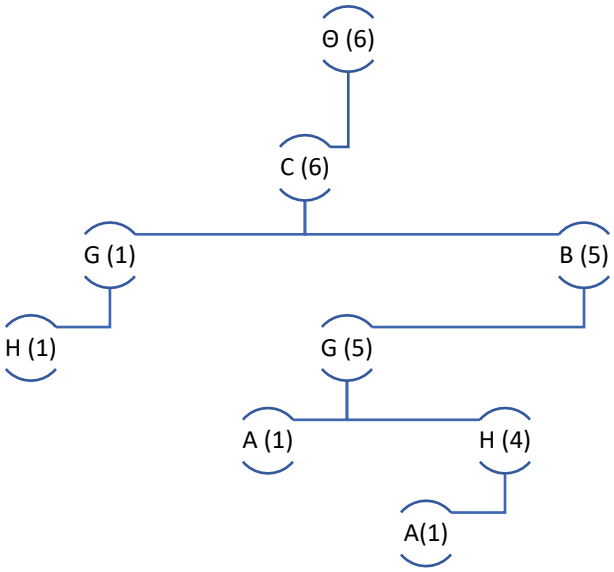
Path	Count
CBGE	2
CBGHAE	1
CBHE	1
CHE	1



We need to recursively call this as we are not in the basis condition. But none of the leafs fulfil the minsup condition. We can remove this tree.

Focus on A:

Path	Count
CGHA	1
CBGA	1
CBGHA	4

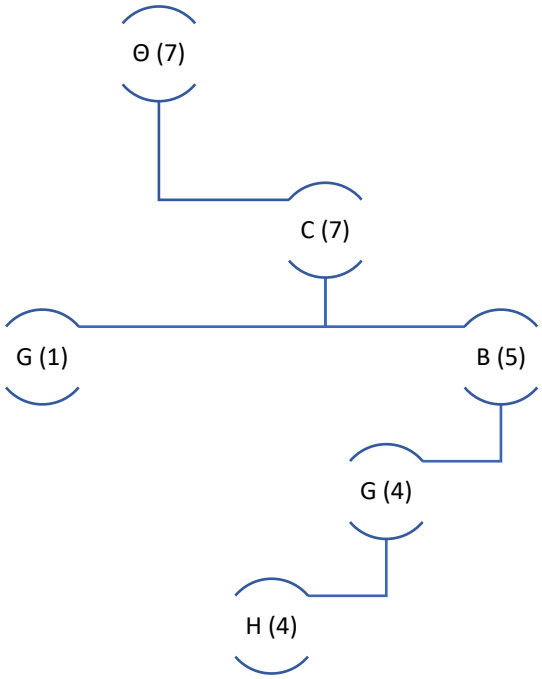


We need to recursively call this as we are not in the basis condition. But leaf A (1), H (1), H (4), G (1) will not stand as minsup is 5.

{AC(6), AB(5), AG(5), ACB(5), ACG(5), ABG(5), ACBG(5)}

Focus on H:

Path	Count
CGH	1
CBGH	4
CBH	1
CH	1



We need to recursively call this as we are not in the basis condition. But leaf G (1), H (4), G will not stand as minsup is 5.

{ HC(7), HB(5), HCB (5) }

Focus on G:

Path	Count
CG	1
CBG	7

{ GC(8), GB(7), GCB (7) }

Focus on B:

Path	Count
CB	8

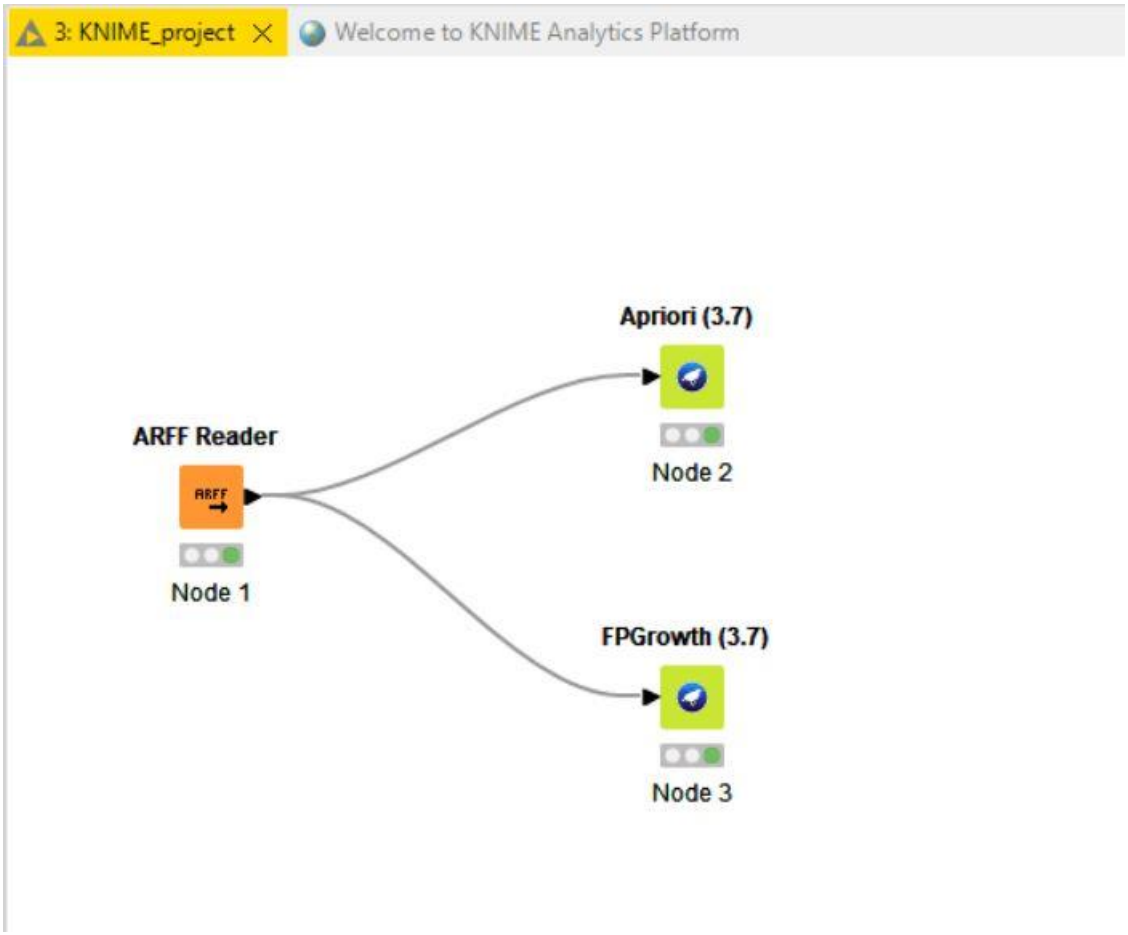
{ BC(8)}

Task 3 KNIME

For this task you will need to install and use the KNIME1 data analytics platform. You are given a file market_basket_transactions.arff which contains the very same transaction as in Table 1. Your task is to implement two simple workflows for mining association rules, one implementing Apriori algorithm and second implementing FP-Growth algorithm. Use the WEKA nodes both for Apriori and FP-Growth. Use the same parameters as in the previous tasks, e.i. minsup = 0.5 and minconf = 0.8. Present pictures of your workflows, and the outputs from both Apriori and FP-Growth nodes. Deliver also the exported KNIME workflows.

Ans:

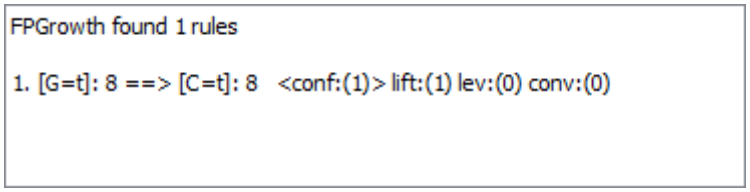
The knime workflow is,



Output from Apriori is,

```
Apriori
=======

Minimum support: 0.5 (5 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 10

Generated sets of large itemsets:

Size of set of large itemsets L(1): 6

Size of set of large itemsets L(2): 10

Size of set of large itemsets L(3): 7

Size of set of large itemsets L(4): 2

Best rules found:

 1. G=t 8 ==> C=t 8    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 2. B=t 7 ==> C=t 7    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 3. B=t 7 ==> G=t 7    <conf:(1)> lift:(1.25) lev:(0.14) [1] conv:(1.4)
 4. H=t 7 ==> C=t 7    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 5. B=t G=t 7 ==> C=t 7    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 6. B=t C=t 7 ==> G=t 7    <conf:(1)> lift:(1.25) lev:(0.14) [1] conv:(1.4)
 7. B=t 7 ==> C=t G=t 7    <conf:(1)> lift:(1.25) lev:(0.14) [1] conv:(1.4)
 8. A=t 6 ==> C=t 6    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 9. A=t 6 ==> G=t 6    <conf:(1)> lift:(1.25) lev:(0.12) [1] conv:(1.2)
10. E=t 6 ==> C=t 6    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
11. A=t G=t 6 ==> C=t 6    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
12. A=t C=t 6 ==> G=t 6    <conf:(1)> lift:(1.25) lev:(0.12) [1] conv:(1.2)
13. A=t 6 ==> C=t G=t 6    <conf:(1)> lift:(1.25) lev:(0.12) [1] conv:(1.2)
14. A=t B=t 5 ==> C=t 5    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
15. A=t B=t 5 ==> G=t 5    <conf:(1)> lift:(1.25) lev:(0.1) [0] conv:(1)
16. A=t H=t 5 ==> C=t 5    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
17. G=t H=t 5 ==> A=t 5    <conf:(1)> lift:(1.67) lev:(0.2) [2] conv:(2)
18. A=t H=t 5 ==> G=t 5    <conf:(1)> lift:(1.25) lev:(0.1) [0] conv:(1)
19. G=t H=t 5 ==> C=t 5    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
20. A=t B=t G=t 5 ==> C=t 5    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
21. A=t B=t C=t 5 ==> G=t 5    <conf:(1)> lift:(1.25) lev:(0.1) [0] conv:(1)
22. A=t B=t 5 ==> C=t G=t 5    <conf:(1)> lift:(1.25) lev:(0.1) [0] conv:(1)
23. C=t G=t H=t 5 ==> A=t 5    <conf:(1)> lift:(1.67) lev:(0.2) [2] conv:(2)
24. A=t G=t H=t 5 ==> C=t 5    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
25. A=t C=t H=t 5 ==> G=t 5    <conf:(1)> lift:(1.25) lev:(0.1) [0] conv:(1)
26. G=t H=t 5 ==> A=t C=t 5    <conf:(1)> lift:(1.67) lev:(0.2) [2] conv:(2)
27. A=t H=t 5 ==> C=t G=t 5    <conf:(1)> lift:(1.25) lev:(0.1) [0] conv:(1)
```

And the output from FP growth is,



Exported Knime workflow is attached separately as zip beside this PDF.