ASSIGN 4

Records

I Neha Moolchandani declare that I have completed this assignment completely and entirely on my own, without any consultation with others. I understand that any breach of the UAB Academic Honor Code may result in severe penalties.

Neha Moolchandani
Course: Data Mining | Professor: Dr. Chengcui Zhang

Assignment #4:

1. This problem uses the data table of Table 8.1 of the text (p. 338). Consider the first 5 records as the entirety of the table. (The main purpose of part (a) of this problem is to explain the relation between the file of “items” that Apriori and FP-growth work on, and data files such as this one.)
Interpreting the records of a file as sets of items: the set of all items is the union of the sets of values appearing in the attributes.

Table 8.1 Class-Labeled Training Tuples from the *AllElectronics* Customer Database

RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Note 1: RecordID is not to be considered an attribute.
Note 2: Because columns 3 and 5 have values in common, recode values in column 3 as “no3” and “yes3” and values in column 5 as “no5” and “yes5”.

Thus the set of items is {youth, mid-aged, senior; high, … ; no5, yes5}.

(a) The first record, as a set of items, is {youth, high, no3, fair, no5}. Write the remaining 4 records of the (truncated) file as sets of items. (16 pts)

1. {youth, high, no3, fair, no5}
2. {youth, high, no3, excellent, no5}
3. {middle_aged, high, no3, fair, yes5}
4. {senior, medium, no3, fair, yes5}
5. {senior, low, yes3, fair, yes5}

(b) Letting the minimum support be 3 records, find F1, C2, F2, C3, and F3 (or, using the notation of pages 249-253 of the text, L1;C2;L2;C3;L3). (20 pts)

Algorithm: Apriori. Find frequent itemsets using an iterative level-wise approach based on candidate generation.

Input:

- D, a database of transactions;
- *min_sup*, the minimum support count threshold.

Output: L, frequent itemsets in D.

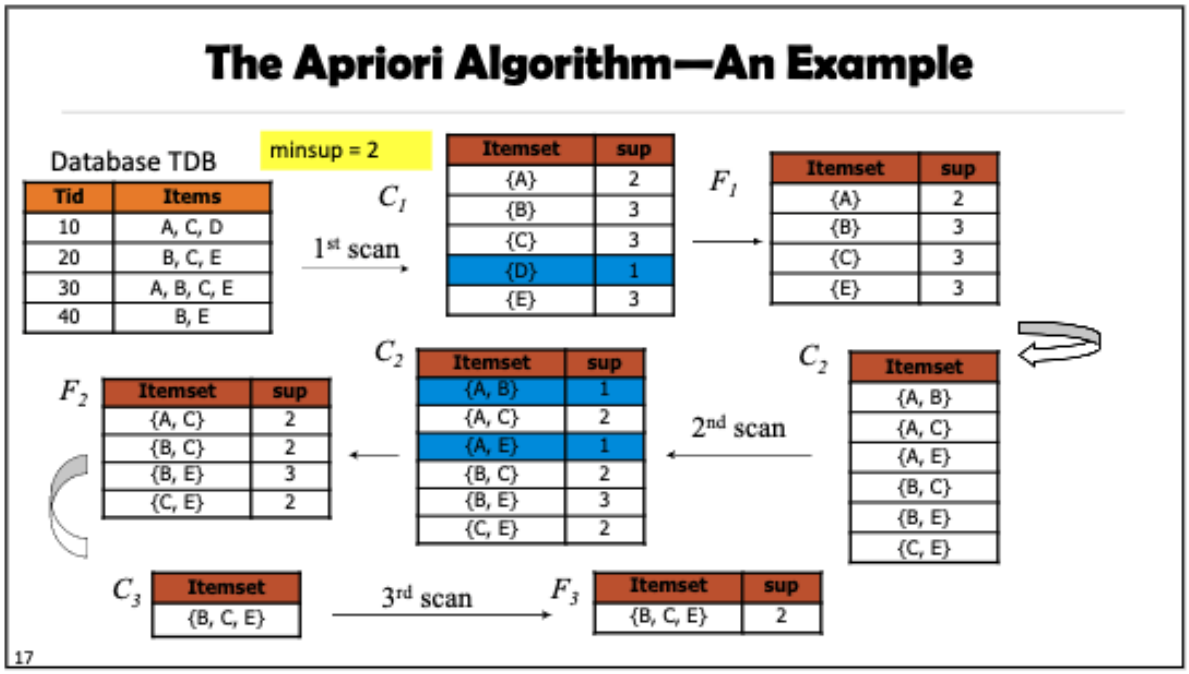
Method:

```
(1)   L1 = find_frequent_1-itemsets(D);
(2)   for (k = 2; Lk−1 ≠ φ; k++) {
(3)       Ck = apriori_gen(Lk−1);
(4)       for each transaction t ∈ D { // scan D for counts
(5)           Ct = subset(Ck, t); // get the subsets of t that are candidates
(6)           for each candidate c ∈ Ct
(7)               c.count++;
(8)       }
(9)       Lk = {c ∈ Ck | c.count ≥ min_sup}
(10)  }
(11)  return L = ∪k Lk;

procedure apriori_gen(Lk−1:frequent (k − 1)-itemsets)
(1)   for each itemset l1 ∈ Lk−1
(2)       for each itemset l2 ∈ Lk−1
(3)           if (l1[1] = l2[1]) ∧ (l1[2] = l2[2])
                  ∧... ∧ (l1[k − 2] = l2[k − 2]) ∧ (l1[k − 1] < l2[k − 1]) then {
(4)               c = l1 ⋈ l2; // join step: generate candidates
(5)               if has_infrequent_subset(c, Lk−1) then
(6)                   delete c; // prune step: remove unfruitful candidate
(7)               else add c to Ck;
(8)           }
(9)   return Ck;

procedure has_infrequent_subset(c: candidate k-itemset;
          Lk−1: frequent (k − 1)-itemsets); // use prior knowledge
(1)   for each (k − 1)-subset s of c
(2)       if s ∉ Lk−1 then
(3)           return TRUE;
(4)   return FALSE;
```

e 6.4 Apriori algorithm for discovering frequent itemsets for mining Boolean association rules.



1. {youth, high, no3, fair, no5}
2. {youth, high, no3, excellent, no5}
3. {middle_aged, high, no3, fair, yes5}
4. {senior, medium, no3, fair, yes5}
5. {senior, low, yes3, fair, yes5}

Support = Freq(a.b) / N

MinSup = 3 C1 -> 1st Scan:

ItemSet	**Support**
{youth}	2
{middle_aged}	1
{senior}	2
{High}	3
{Medium}	1
{Low}	1
{Fair}	4
{No3}	4
{yes3}	1
{No5}	2
{Yes5}	3

F1 -> Delete the 1 and 2 since MinSUpport is 3 so delete Youth, MiddleAged, Senior, Medium, Low, Yes3 and No5
Left With:

ItemSet	Support
{High}	3
{Fair}	4
{No3}	4
{Yes5}	3

C2:

ItemSet
{High,Fair}
{High,No3}
{High,Yes5}
{Fair,No3}
{Fair, Yes5}
{No3, Yes5}

C2: Second Scan

ItemSet	Support
{High,Fair}	2
{High,No3}	3
{High,Yes5}	1
{Fair,No3}	3
{Fair, Yes5}	3
{No3, Yes5}	2

F2 -> Delete the 1 and 2 since MinSUpport is 3 so delete Row1, Row3, and Row6
Left With:

ItemSet	Support
{High,No3}	3
{Fair,No3}	3
{Fair, Yes5}	3

C3:

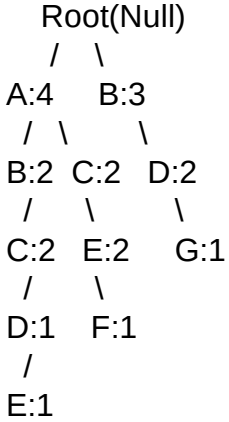
ItemSet
{Fair,No3}

F3: 3rd Scan:

{Fair,No3}	3

2. (a) construct the FP-tree for the set of records below, using minimum support threshold 1. This tree is denoted as *T*. (Items are already in order by decreasing support.) (12 pts)

{}
a: 5
b:5
c:5
d:4
e:4
f:1
g: 1

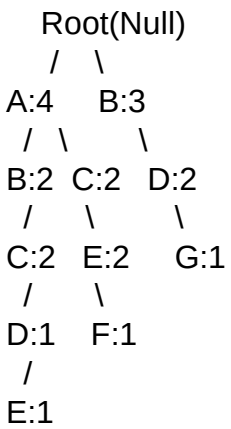


(b) Construct the conditional FP-tree for item *f*, which will be denoted as T_f. (12 pts)

Item	Conditional DataBase
A	empty
B	empty
C	ab:2, a:2
D	abc:1, ac:1, b:2
E	abc:1, ac:2
F	ace:1
G	bd:1

(c) Execute the procedure FP-growth(T_f, □), where □={*f*}. Show the step by step details and the generated patterns together with their support counts. (p. 260) (12 pts)

Selected and sorted in order of L: a: 5, b:5, c:5, d:4, e:4, f:1, g:1



(d) Construct the conditional FP-tree for item *d*, which will be denoted as T_d. (10 pts)

Item	Conditional DataBase
A	empty
B	empty
C	a:4
D	empty
E	a:3, c:3
F	a:1, c:1, 3:1
G	b:1, d:1

(e) Execute FP-growth (T_d, {*d*}). Show the step by step details and the generated patterns together with their support counts. (18 pts)

Removing F and G as MinSUpport is 1

Constructing the Tree based off

The set of items is I = {a, b, c, d, e, f, g}

a: 5, b:5, c:5, d:4, e:4, f:1, g:1

F-list: a,b,c,d,e,g

The set of records is below.

LIST
a, b, c, d, e
a, c, e, f
b, d, g
a, b, c, e
a, c, e
b, d
b
a, c, d

ORDERED FREQ ITEMLIST
A,b,c,d,e
A,c,e,f
B,d,g
a,b,c,e
a,c,e
b,d
b
a,c,d