COMP9318 Tutorial 4: Association Rule Mining

Wei Wang @ UNSW

Q1 I

Show that if $A \rightarrow B$ does not meet the minconf constraint, $A \rightarrow BC$ does not either.

Solution to Q1 I

$$
\begin{aligned}
conf(A \rightarrow BC) &= \frac{supp(ABC)}{supp(A)} \\
&\leq \frac{supp(AB)}{supp(A)} = conf(A \rightarrow B)
\end{aligned}
$$

Like Apriori, we can utilize this rule when generating association rules.

Q2 I

Given the following transactional database

1	C, B, H
2	B, F, S
3	A, F, G
4	C, B, H
5	B, F, G
6	B, E, O

1. We want to mine all the frequent itemsets in the data using the Apriori algorithm. Assume the minimum support is 30%. (You need to give the set of frequent itemsets in L_1, L_2, ..., candidate itemsets in C_2, C_3, ...).
2. Find all the association rules that involves only B, C, H (in either left or right hand side of the rule). The minimum confidence is 70%.

Solution to Q2 I

1. Apriori
   1.1 minsup $= 30\% \times 6 = 1.8$. In other words, the support of a frequent itemset must be no less than 2.
   1.2 $C_1 = \{ A, B, C, E, F, G, H, O, S \}$, scanning the DB and collect the supports as

A	B	C	E	F	G	H	O	S
1	5	2	1	3	2	2	1	1

   Therefore, $L_1 = \{ B, C, F, G, H \}$.
   1.3 C_2 is generated from L_1 by enumerating all pairs as $\{ BC, BF, BG, BH, CF, CG, CH, FG, FH, GH \}$. Scan the DB and collect the supports as (you may want to sort items in each transaction and remove non-frequent items from the DB)

BC	BF	BG	BH	CF	CG	CH	FG	FH	GH
2	2	1	2	0	0	2	2	0	0

   Therefore, $L_2 = \{ BC, BF, BH, CH, FG \}$.
   1.4 C_3 is generated from L_2 by a special enumeration-and-pruning procedure. The result is $\{ BCH \}$. Scan the DB and collect the support as

BCH
2

   Therefore, $L_3 = \{ BCH \}$.
   1.5 C_4 will be the empty set, therefore we stop here.

2. We list the frequent itemsets related to B, C, and H below:

Solution to Q2 II

B	C	H	BC	BH	CH	BCH
5	2	2	2	2	2	2

   2.1 For BC, we need to consider candidate rules: $B \to C$, and $C \to B$. The former has confidence $\frac{supp(BC)}{supp(B)} = 40\%$ and does not meet the minconf requirement. The latter rule has confidence $\frac{supp(BC)}{supp(C)} = 100\%$ and it is qualified.
   2.2 It is easy to see that any rule in the form of $B \to \dots$ will not meet the minconf requirement for the dataset. Therefore, we can repeat the above procedure and find the following rules:
   - $H \to B$ (100%)
   - $C \to H$ (100%)
   - $H \to C$ (100%)
   - $BC \to H$ (100%)
   - $BH \to C$ (100%)
   - $CH \to B$ (100%)
   - $C \to BH$ (100%)
   - $H \to BC$ (100%)

Q3 I

Compute the frequent itemset of for the data in Q2 using the FP-growth algorithm.

Solution to Q3 I

1. Similar to the first step in Apriori, count the support of all items and *normalize* the original transaction db as follows: (by removing non-frequent items and sort items in the decreasing order of their support)

Order

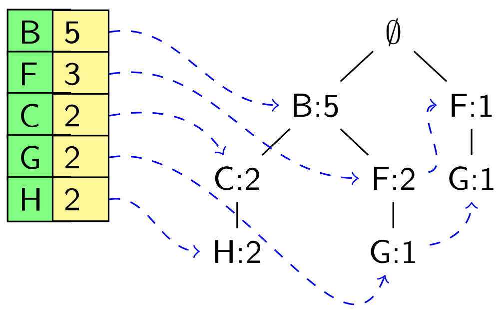
B	F	C	G	H
5	3	2	2	2

DB

1	B, C, H
2	B, F
3	F, G
4	B, C, H
5	B, F, G
6	B

We can output all frequent item: B, C, F, G, H.

2. Construct the FP-tree as:

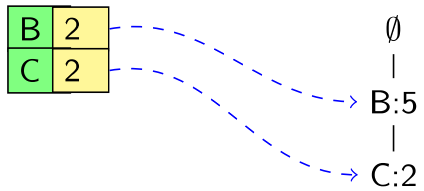


Solution to Q3 II

3. H's conditional pattern base is:

```
B C : 2
```

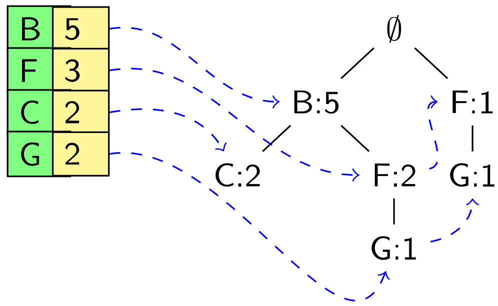
All of the items are frequent, and thus we can output: BH, CH. Construct the H-conditional FP-tree as



Since it is a single-path tree, we directly output all its combinations: BCH.

4. We track back and can now safely remove all H nodes from the initial FP-tree, as shown below.

Solution to Q3 III



We now find G's conditional pattern base as:

```
B F : 1
  F : 1
```

Only F is frequent. We output FG. It is clear that we can stop.

5. We track back and can now safely remove all G nodes from the FP-tree, and then process C's conditional pattern base:

```
B : 2
```

B is frequent, output BC, and we can stop here.

6. We track back and can now safely remove all C nodes from the FP-tree, and then process F's conditional pattern base:

Solution to Q3 IV

```
B : 2
```

B is frequent, output BF, and we can stop here.

7. Since we are left with one item (B) only, we can output stop the whole mining process.