Solutions Frequent Pattern Mining

Exercise 1: Frequent Item Set Mining

(a) Level 1:

candidate	support	frequent?
A	2	✓
B	5	✓
C	4	✓
D	2	✓
E	2	✓

Level 2:

candidate	support	frequent?
AB	2	✓
AC	2	✓
AD	0	✗
AE	0	✗
BC	4	✓
BD	2	✓
BE	1	✗
CD	1	✗
CE	1	✗
DE	1	✗

Level 3:

candidate	support	frequent?
ABC	2	✓

The "pre-candidate" BCD is generated from level 2 frequent item sets BC and BD, but it is pruned because level 2 subset CD is not frequent. Don't list BCD in the table with its count, because that suggests it was counted on the data base!

(b) First we determine the generators, and then we obtain the closed frequent item sets by taking the closure of the generators. To determine the generators, we apply Apriori with an additional pruning step: an item set is pruned when it has a subset with the same support. The normal frequency pruning is indicated by ✗, the additional A-close pruning by ✂.

Level 1:

candidate	support	generator?
A	2	✓
B	5	✓
C	4	✓
D	2	✓
E	2	✓

Level 2:

candidate	support	generator?
AB	2	✂
AC	2	✂
AD	0	✗
AE	0	✗
BC	4	✂
BD	2	✂
BE	1	✗
CD	1	✗
CE	1	✗
DE	1	✗

For example, AB is pruned because its subset A has the same support.
There are no level 3 candidates.

Next we determine the closures of the generators. These closures together form the set of closed frequent item sets. A closure has the same support as its generator, and can be obtained by taking the intersection of all transactions in which the generator occurs.

generator	closure	support
A	ABC	2
B	B	5
C	BC	4
D	BD	2
E	E	2

(c)

$$\text{confidence}(B \to C) = \frac{s(BC)}{s(B)} = \frac{4}{5}$$

$$\text{lift}(B \to C) = \frac{s(BC) \times |db|}{s(B) \times s(C)} = \frac{4 \times 6}{5 \times 4} = \frac{6}{5},$$

where $|db|$ denotes the number of transactions in the data base.

Exercise 2: Constraints

(a) Yes. Apriori starts with 1-item sets, then 2-item sets, etc. If itemset I does not satisfy the total price constraint, then neither will any of its supersets. Hence, k-item set I can only satisfy the constraint if all its level $k-1$ subsets satisfy it. Hence, we can use the same pruning principle as the support pruning of Apriori.

(b) Here the reasoning used under (a) does not apply, because the average price will go down when we add a "cheap" item to an item set.

(c) Here the phrase "Apriori style level wise search" caused some confusion with some of you. It was not intended to mean that the pruning is identical to the pruning that is performed in Apriori, but merely that we can apply some form of pruning in a level wise search. With that interpretation the answer is "Yes". We can order the items from cheap to expensive. This answer would have been sufficient!

Consider the following example. We have the constraint $\text{avg}(I.\text{price}) \leq 5$. In the table below the ordering of items on price coincides with their alphabetical order, so we can easily determine the proper order. We assume that the minimum support constraint is satisfied by all item sets so we can only perform pruning on average price.

candidate	price	pass?
A	2	✓
B	2	✓
C	6	✗
D	8	✗
E	10	✗

Items C, D and E do not satisfy the constraint, but this does not mean they can't have supersets that do. All we know is that if we add a more expensive item to them, the constraint will still be violated. Hence, the next level candidates are:

candidate	price	pass?
AB	2	✓
AC	4	✓
AD	5	✓
AE	6	✗
BC	4	✓
BD	5	✓
BE	6	✗

Note that CD, CE and DE are not candidates. For example: CD is not a candidate because C does not satisfy the constraint, and D is more expensive than C, so CD will certainly violate the constraint as well.
The level-3 candidates are (no pruning possible here):

candidate	price	pass?
ABC	$3\frac{1}{3}$	✓
ABD	4	✓
ABE	$4\frac{2}{3}$	✓
ACD	$5\frac{1}{3}$	✗
ACE	6	✗
ADE	$6\frac{2}{3}$	✗
BCD	$5\frac{1}{3}$	✗
BCE	6	✗
BDE	$6\frac{2}{3}$	✗

The level-4 candidates are:

candidate	price	pass?
$ABCD$	$4\frac{1}{2}$	✓
$ABCE$	5	✓
$ABDE$	$5\frac{1}{2}$	✗

Here we pruned $ACDE$ and $BCDE$.
Finally:

candidate	price	pass?
$ABCDE$	$5\frac{3}{5}$	✗

All in all we were able to prune 5 candidates.

Exercise 3: Frequent Tree Mining

(a) Let's denote the nodes of d_4 by v and the nodes of $T = aa \uparrow c$ by w. The nodes are numbered according to the pre-order (depth-first) traversal of the tree. T is an embedded subtree of d_4, and the corresponding matching function is:

$$\phi(w_1) = v_1 \quad \phi(w_2) = v_2 \quad \phi(w_3) = v_5$$

Verify that this matching function satisfies all the requirements:

1. The labeling is preserved: $L(w_1) = L(v_1) = a$, $L(w_2) = L(v_2) = a$, $L(w_3) = L(v_5) = c$.
2. The ancestor-descendant relation is preserved:
   - $w_1 \in \pi^*(w_2)$ and $v_1 \in \pi^*(v_2)$.
   - $w_1 \in \pi^*(w_3)$ and $v_1 \in \pi^*(v_5)$.
   - $w_2 \notin \pi^*(w_3)$ and $v_2 \notin \pi^*(v_5)$.
   - $w_3 \notin \pi^*(w_2)$ and $v_5 \notin \pi^*(v_2)$.
3. The ordering is preserved:
   - $w_1 < w_2$ and $v_1 < v_2$.
   - $w_1 < w_3$ and $v_1 < v_5$.
   - $w_2 < w_3$ and $v_2 < v_5$.

T is not an induced subtree of d_4. The matching function that worked for the embedded subtree relation does not work here, because it does not preserve the parent-child relationship: $w_1 = \pi(w_3)$ but $v_1 \neq \pi(v_5)$.

(b) No, it is not. For example, the matching function

$$\phi(w_1) = v_1 \quad \phi(w_2) = v_3 \quad \phi(w_3) = v_2$$

does not preserve the order relation, because $w_2 < w_3$, but $v_3 \not< v_2$.

(c) It occurs 8 times as an embedded subtree. The distinct matching functions are: $\phi(w_1) = v_1$, and then

1. $\phi(w_2) = v_2$ and $\phi(w_3) = v_3$
2. $\phi(w_2) = v_2$ and $\phi(w_3) = v_5$
3. $\phi(w_2) = v_3$ and $\phi(w_3) = v_5$
4. $\phi(w_2) = v_2$ and $\phi(w_3) = v_4$
5. $\phi(w_2) = v_2$ and $\phi(w_3) = v_6$
6. $\phi(w_2) = v_3$ and $\phi(w_3) = v_6$
7. $\phi(w_2) = v_4$ and $\phi(w_3) = v_5$
8. $\phi(w_2) = v_4$ and $\phi(w_3) = v_6$

The first three matching functions are also induced subtrees.

Exercise 4: Trees with Trees as Features

1. Induced: the data tree ends up in leaf node t_3, so it gets assigned the label $+$.
2. Embedded: the data tree ends up in leaf node t_2, so it gets assigned the label $-$.

Exercise 5: An Interestingness Measure

(a) Yes, it is sufficient.

$s(x \wedge y)$ is known, since $X \cup Y$ is frequent. For the same reason, $s(x)$ and $s(y)$ are known. Since

$$s(x \wedge \neg y) = s(x) - s(x \wedge y)$$

this quantity can also be computed. Likewise, we have

$$s(\neg x \wedge y) = s(y) - s(x \wedge y)$$

Finally, we have

$$s(\neg x \wedge \neg y) = 1 - \{s(x \wedge \neg y) + s(\neg x \wedge y) + s(x \wedge y)\}.$$

(b) Zero. From independence it follows that $s(x \wedge y) = s(x)s(y)$, etc., so novelty reduces to

$$s(x \wedge y)s(\neg x \wedge \neg y) - s(x \wedge \neg y)s(\neg x \wedge y) = s(x)s(y)s(\neg x)s(\neg y) - s(x)s(\neg y)s(\neg x)s(y) = 0$$

Exercise 6: Frequent Item Sets: Two Proofs

Prove the following:

(a) The set of maximal frequent item sets is a subset of the set of closed frequent item sets.

An informal proof by contradiction: suppose X is maximal frequent. This means X is frequent, and there is no superset of X that is also frequent. Now suppose X is not closed. This means there is a superset of X with the same support as X. But then this superset is also frequent, which contradicts the assumption that X is maximal frequent.

Since assuming X is not closed leads to a contradiction, we conclude that X must be closed. Overall, we may draw the conclusion that if X is a maximal frequent item set, then X is also a closed frequent item set.

(b) For any association rule $X \to Y$, if we move an item from X to Y, then the confidence can never go up.

The confidence of $X \to Y$ is given by:

$$\text{confidence}(X \to Y) = \frac{s(X \cup Y)}{s(X)}$$

If we move an item from X to Y, the numerator won't change, but the denominator will increase or stay the same. Hence, the confidence will either decrease or stay the same.

Could this property be used as a basis for pruning the search space when generating all association rules with confidence $\geq t_2$?

In generating association rules from a frequent item set Z, we could start with rules $Z \setminus \{A\} \to A$ having a single item in the right-hand side. If such a rule does not satisfy the minimum confidence constraint, there's no use in checking rules with a superset of $\{A\}$ on the right-hand-side.