Assignment 1 EECS 6412

Daniel Marchena Parreira

2018/09/20

Question 1

a) Clustering analysis would be the best task since unlike classification and regression, which analyze class-labeled (training) data sets, clustering analyzes data objects without consulting class labels. The objects are clustered or grouped based on the principle of maximizing the intraclass similarity and minimizing the inter class similarity.

b) Classification for Predictive Analysis would be the most suitable task. Furthermore, predictive analysis is the process of finding a model (or function) that describes and distinguishes data classes or concepts based on the analysis of a set of training data. We could find all clients who bought a Fujifilm Instax mini camera and based on that data predict which other clients will buy it.

c) Mining Frequent Patterns (frequent itemsets) would do the job. This technique as the name suggests, looks for patterns that occur frequently in data. A frequent itemset typically refers to a set of items that often appear together in a transactional data set for example, milk and bread, which are frequently bought together in grocery stores by many customers.

d) The retailer could come up with new promotions to boost up his sales. On that note, they could use association analysis to create association rules and discover which other product are most likely to sell together with the camera. An example of an association rule in this case would be: "buys(X,"Fujifilm Instax mini camera") →buys(X,"Chocolate bar") [support = 1%,confidence = 50%]" where support means that 1% of all the transactions under analysis show that Fujifilm Instax mini camera and Chocolate bars are purchased together, and confidence means that if a customer buys a Fujifilm Instax mini camera, there is a 50% chance that she will buy a Chocolate bar as well.

e) Outlier analysis is the technique to be used in this case, because whenever a data set may contain objects that do not comply with the general behavior or model of the data. These data objects are outliers. Outliers may be detected using statistical tests that assume a distribution or probability model for the data, or using distance measures where objects that are remote from any other cluster are considered outliers.

Question 2

To answer this questions we first need to explain how the Apriori algorithm uses prior knowledge of frequent itemset properties. Having said that, it employs an iterative approach known as a level-wise search, where k-itemsets are used to explore (k + 1)-itemsets.

The algorithm executes the following steps (supposing a min-support of 2):

1. It does a full scan of the database (D):

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

2. Apriori generates a candidate set (C1) by calculating the count of each item in D, and collecting those items that satisfy minimum support

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

3. The result set is denoted L1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

4. The process starts over, but now we will develop C2 by generating all possible combinations of L1 with itself (join step)

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

5. Another Database scan is needed in order to get the count for those new itemsets, and also for removing entries that don't meet the min-support

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

6. L2 is then generated

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

7. Once again the process starts over, but this time after making all possible combinations out of L2 (join step), we will also run a (prune step). This prune step will make sure that no infrequent itemsets previously discarded in C2 will show up as part of L3.

- **a** Apriori property: All nonempty subsets of a frequent itemset must also be frequent.
- **b** {1, 2} and {1, 5} were infrequent in C2
- **c** {1, 2, 3}, {1, 2, 5} and {1, 3, 5} are sets that will be pruned, since all three contain at least one of the 2 ({1, 2} and/or {1, 5})

itemset
{2 3 5}

8. Finally, a new db scan will be done to make sure that C3 ({2, 3, 5}) satisfies the min-support, and L3 will be outputted

itemset	sup
{2 3 5}	2

This step by step proves that the generated Ck+1 is not only a join of Lk-1 with it-self. In addition to that, the algorithm also filters the candidate set by min-support and prunes any previous infrequent subsets. Consequently, it is safe to assume that Ck+1 will never miss any frequent itemset in Lk+1.

We could also prove by contradiction by supposing that Lk+1 misses an itemset {a1, a2 ak} which is frequent. As a rule of thumb we know that each subset of this frequent itemset is frequent. Thus {a1, a2.... ak-1, ak} and {a1, a2.... ak-1, ak+1} should be present in Lk . This means that these two itemsets could have been joined in Ck+1 if present. So hypothesizing that Lk+1 misses the aforementioned itemset is wrong and if it was a frequent itemset, it was already in Lk+1.

Question 3

The first step of the FP-growth algorithm is to create a header table which contains all frequent items:

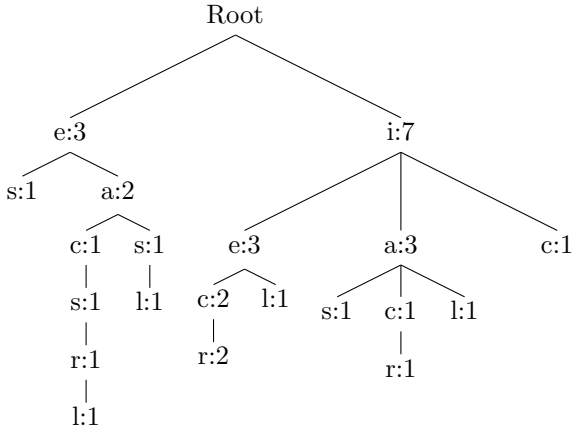
Header Table	
Item	Count
i	7
e	6
a	5
s	4
r	4
c	4
l	4

Minimum support count = 10(items) * 0.3(min-support) →3

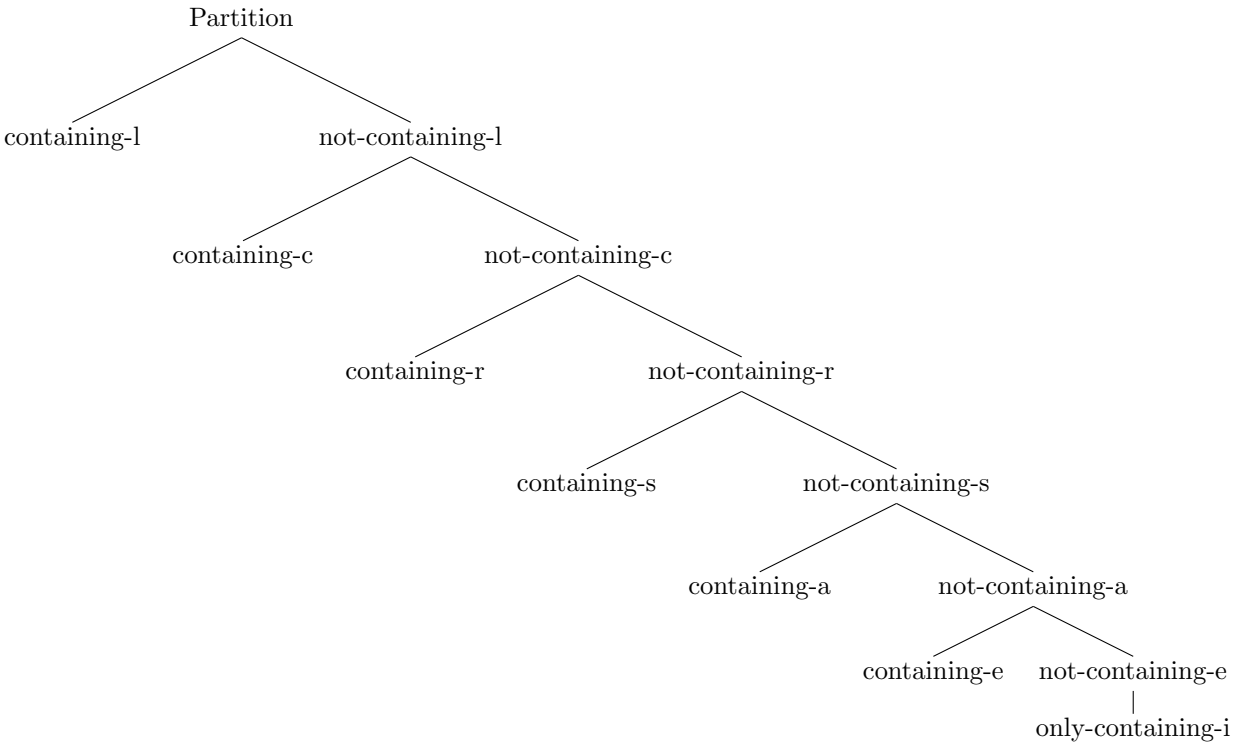
With the minimum support count in our hands, now we construct a transaction table with the infrequent items removed and ordered by frequent items in descending order:

TID	Ordered frequent items
1	{e, s}
2	{i, e, c, r}
3	{e, a, s, l}
4	{i, a, s}
5	{i, e, c, r}
6	{i, c}
7	{i, a, c, r}
8	{i, e, l}
9	{e, a, c, s, r, l}
10	{i, a, l}

Next we create our FP-Tree:



Now we partition our dataset:

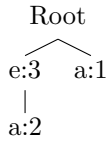


Generate Conditional pattern bases:

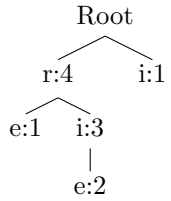
Conditional Pattern bases			
item	Cond. pattern base	Conditional FP-Tree	Remove infrequent
e	i:3	{(i:3)}|e	N/A
a	e:2, i:3	{(i:3)}|a	N/A
s	e:1, ae:2, ai:1	{(e:3, a:3)}|s	e:1, ea:2, a:1
r	sae:1, ei:2, ai:1	{(e:3, i:3)}|r	e:1, ei:2, i:1
c	rsae:1, rei:2, rai:1, i:1	{(r:4, i:4, e:3)}|c	re:1, rie:2, ri:1, i:1
l	easrc:1, eas:1, ei:1, ai:1	{(e:3, a:3)}|l	ea:2, e:1, a:1

Frequent patterns created: ie:3, ia:3, es:3, as:3, er:3, ir:3, rc:4, ic:4, ec:4, el:3, al:3

l-conditional FP-tree: no new patters

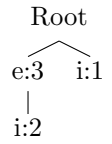


c-conditional FP-tree:

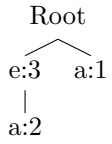


c-e-conditional pattern base: ir:2, r: 1, so frequent pattern is r:3.
c-i-conditional pattern base: r:3
Frequent patterns: cer:3, cir:3

r-conditional FP-tree: no new patterns



s-conditional FP-tree: no new patterns



Final frequent patterns: cer:3, cir:3, ie:3, ia:3, es:3, as:3, er:3, ir:3, rc:4, ic:4, ec:4, el:3, al:3, i:7, e:6, a:5, s:4, r:4, c:4, l:4

a) Containing i: cir:3, ie:3, ia:3, ir:3, ic:4, i:7
b) Freq itemsets = {c, r}, {e}, {a}, {r}, {c}

{c,r}→{i} confidence = support({ c, r, i }) ÷ support({ c, r }) = 3 ÷ 4 = 75%
{e}→{i} confidence = support({ e, i }) ÷ support({ e }) = 3 ÷ 6 = 50% (does not match minimum confidence)
{a}→{i} confidence = support({ a, i }) ÷ support({ a }) = 3 ÷ 5 = 60%
{r}→{i} confidence = support({ r, i }) ÷ support({ r }) = 3 ÷ 4 = 75%
{c}→{i} confidence = support({ c, i }) ÷ support({ c }) = 4 ÷ 4 = 100%

c) $lift(A \rightarrow B) = \frac{P(AB)}{P(A)(B)}$
$lift(\{c,r\} \rightarrow \{i\}) = \frac{P(\{c,r,i\})}{P(\{c,r\})(\{i\})} = \frac{0.3}{0.4 \cdot 0.7} > 1$
$lift(\{a\} \rightarrow \{i\}) = \frac{P(\{ai\})}{P(\{a\})(\{i\})} = \frac{0.3}{0.5 \cdot 0.7} < 1$
$lift(\{r\} \rightarrow \{i\}) = \frac{P(\{r,i\})}{P(\{r\})(\{i\})} = \frac{0.4}{0.4 \cdot 0.7} > 1$
$lift(\{c\} \rightarrow \{i\}) = \frac{P(\{c,i\})}{P(\{c\})(\{i\})} = \frac{0.4}{0.5 \cdot 0.7} > 1$

After running lift for every single frequent itemset, the only misleading rule is ({a} →{i}) since the lift result is less than one. On that note, the variables are not very correlated i.e. independent.

Question 4

In order to implement this new constraint efficiently we must begin by ordering the list of frequent single items in ascending order before creating the FP-tree. The constraint is not monotonic or anti-monotonic so we must take an approach of reordering the items.

Taking the previous question for example with the frequent single items:

Header Table		
Item	Count	Price
i	7	500
e	6	40
a	5	70
c	5	600
r	4	200
s	4	100
l	4	10

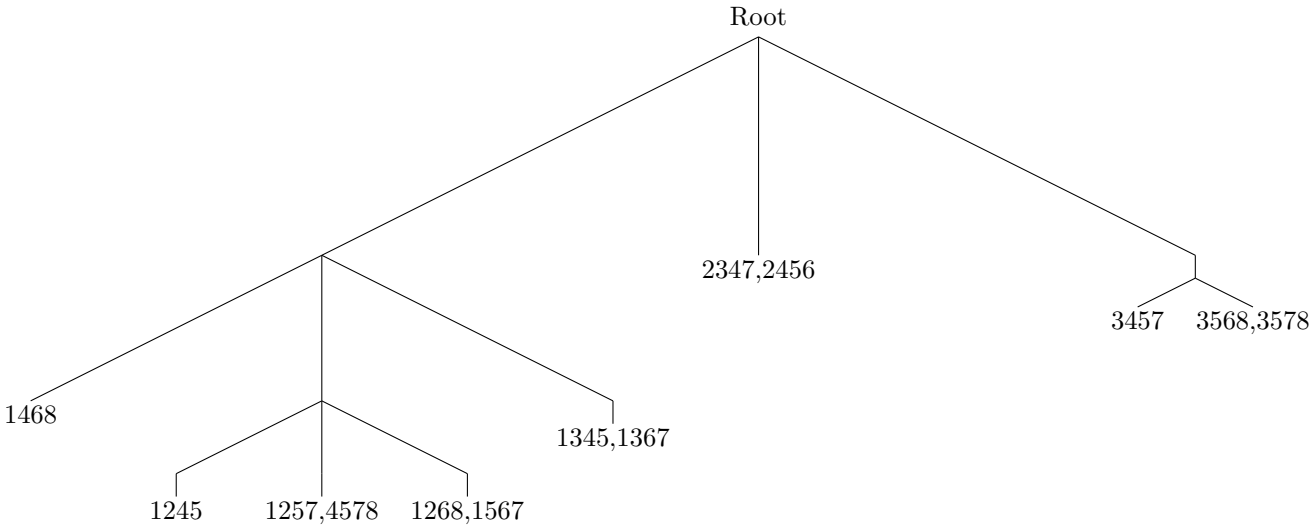
We have removed the infrequent items. But now to create our FP-tree we must reorder:

Header Table		
Item	Count	Price
l	4	10
e	6	40
a	5	70
s	4	100
r	4	200
i	7	500
c	5	600

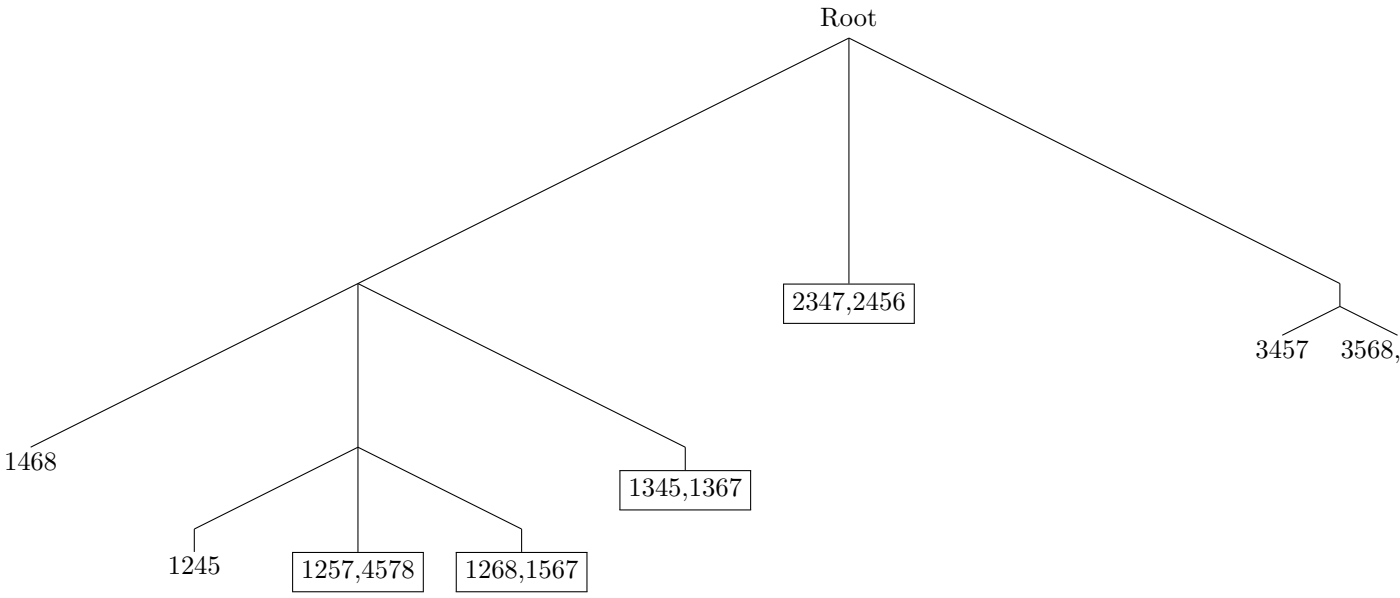
Whenever we create our new tree, it will have a property such that for each node in the tree, the average of any prefix of any node will have an lower equal average of the sequence corresponding to that node. Consequently, when we create our conditional itemsets and during the creation of our conditional pattern base, we remove the suffixes in the patterns that make the average become more than $300.

Question 5

a)



b)



Question 6

In order to efficiently mine the (global) association rules we first need to find the frequent itemsets of ΔDB. Moreover, we need to find the association rules of ΔDB, and finally after finding these frequent items we add those to the list of frequent items.

To approach this problem, we can think of ΔDB and as partitions that we want to merge. Having said that, we know that for an itemset to be frequent in the global database it must be frequent in at least one of the partitions.

Since after mining ΔDB we know it's frequent patterns ΔFI, adding them to the partition works. We are now able to mine the global database for association rules with these new frequent itemsets. However, since the item frequency count may go up, then you must mine the new database for global association rules with this new database of frequent items.