Information Systems - Lab assignment 1

Siemen Boelkens - s2788845
Hichem Bouakaz - s2525763

December 7, 2018

1 Histogram

Show the histogram of all items in the 9835 transactions

To plot the histogram of all the transactions we wrote the following function as seen in *shoppingListHistogram.m*:

```
function ShoppingListHistogram
    shoppingList = readDataFile;
    cat = categorical([shoppingList{:}]);
    histogram(cat);
    set(gca,'TickLength',[0 0])
    set(gca, 'FontSize', 8)
    xlabel(Items')
    ylabel('Frequency')
    title('Histogram showing the frequency of the shoppingList items')
end
```

See Figure 1.

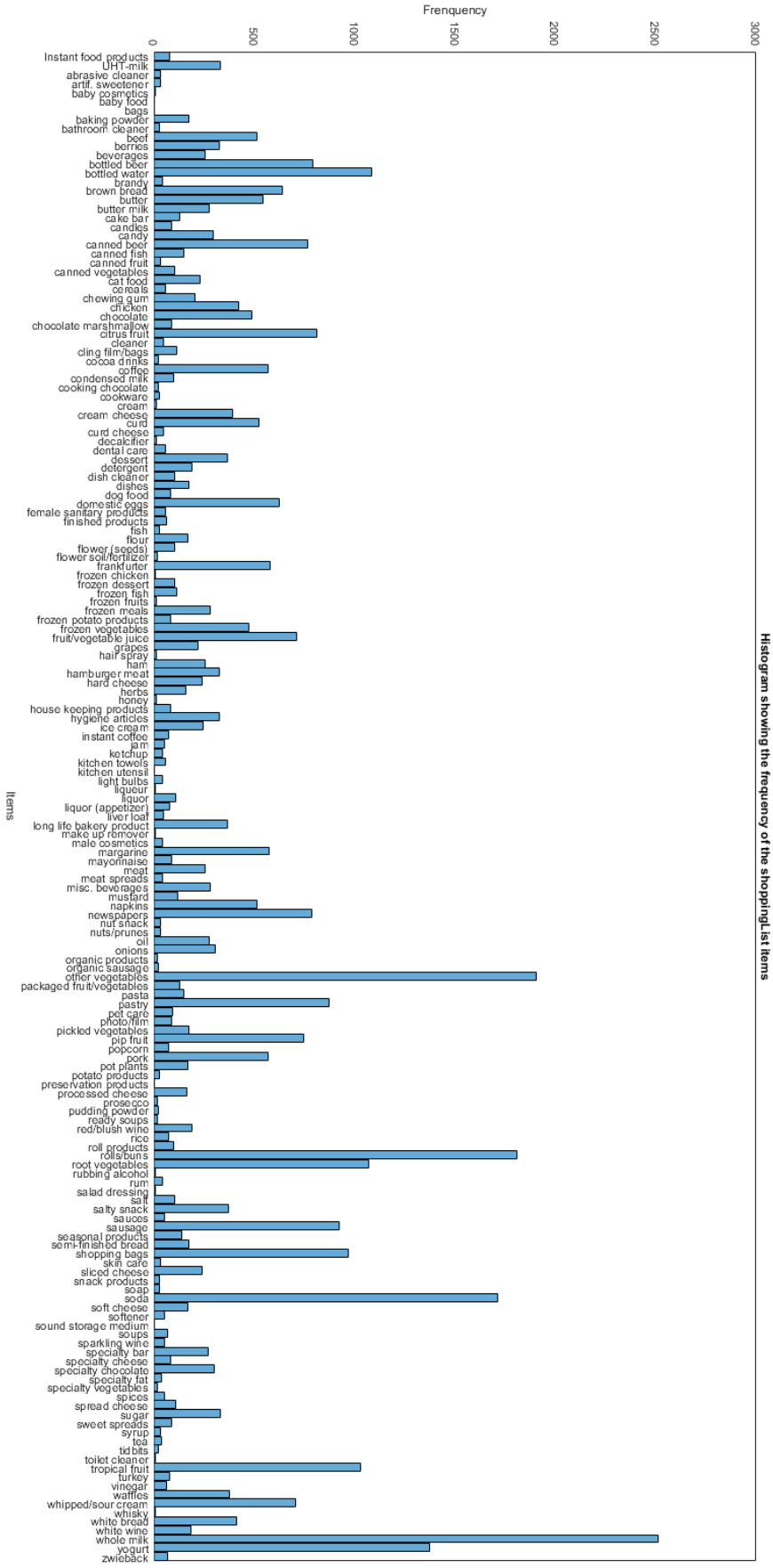

Figure 1: Histogram showing the frequency of shopping list items

2 Improving association rules

Use the given Matlab implementation (associationRules.m) and improve the selection of association rules by making it more efficient. Use slide 43 of Lecture 1 as a reference.

We've improved the implementation in two parts. Part one is that we've inverted the for loop for creating the antecedentlists, by starting with the most antecendents and slowly moving on to smaller ones. For example, if you have a frequent itemset of 4 items, it will first try to find the confidence for all combinations with 3 antecedents and 1 consequent, before moving on to two antecedents an so forth. This can be seen in Listing 1 on line 7.

The second part is that we've added a lowConfList, to store item sets of which the confidence has been calculated and of which the confidence is not equal or higher to the minimum confidence. Each item set is then checked if it is a subset of any itemsets in this list. If this is the case, the program will not calculate the confidence of this itemset and ignore it. This is done, because confidence is non-increasing as number of items in rule consequent increases. This can be seen in Listing 1 on line 4, lines 12 through 22 and lines 36 through 41.

As for the code itself, we made a copy of the *associationRules.m* into *associationRulesOptimized.m* and made all our optimizations in this file.

```
1      % We consider the itemsets that have at least two items
2  for k = 2:numel(frequentItems)
3      for j = 1:size(frequentItems{k},1)
4          lowConfList = {};
5          % Improvement: Turn around foreach loop to start with the most
6          % antecedents, then go lower instead of other way around
7          for i = numel(frequentItems{k}(j,:))-1:-1:1
8              antecendentlist = nchoosek(frequentItems{k}(j,:),i);
9              for l = 1:size(antecendentlist,1)
10                 consequence = find(~ismember(frequentItems{k}(j,:),
                       antecendentlist(l,:)));
11                 isSubsetOf = false;
12                 % Improvement: check if our current antecedentlist
                       member
13                 % is a subset of what is already in the lowConfList
14                 for x = 1:size(lowConfList)
15                     isSubsetOf = all(ismember(antecendentlist(l,:),
                           lowConfList{x}));
16                     if isSubsetOf
17                         break;
18                     end
19                 end
20                 if isSubsetOf
21                     continue;
22                 end
23                 confidence = support{k}(j)*ntrans/sum(all(dataset(:,
                       antecendentlist(l,:)),2));
24                 if confidence >= minconf
25                     sup(end+1) = support{k}(j);
26                     conf(end+1) = confidence;
27                     antecedentstr = '';
28                     for m = 1:size(antecendentlist,2)
29                         antecedentstr = [antecedentstr,trlbl{
                               antecendentlist(l,m)},','];
```

```
30                          end
31                          consequencestr = '';
32                          for m = 1:numel(consequence)
33                              consequencestr = [consequencestr,trlbl{
                                    frequentItems{k}(j,consequence(m))},','];
34                          end
35                          AR{end+1} = [antecedentstr(1:end-1),' -> ',
                                consequencestr(1:end-1),sprintf(' [%2.2f,%2.2f]
                                ',sup(end),conf(end))];
36                      else
37                          % Improvement: if confidence is not high enough,
                                store
38                          % it in the lowConfList to keep track of
                                antecendents
39                          % that do not have high enough confidence
40                          lowConfList{end+1} = antecendentlist(l,:);
41                      end
42                  end
43              end
44          end
45      end
```

Listing 1: Optimized associationRules algorithm

3 Compare execution times

Compare the time needed to generate all association rules based on the given implementation to that of the improved implementation (as per point 2 above).

After we created our optimized version of the code, we split the code up into multiple parts. The first part is the creation of the frequent itemsets, places in *getFrequentItems.m*. The second part is the original association rule calculation, put into *getConfidence.m*. Then there is our optimized version of this calculation, put into *getConfidenceOptimized.m* and lastly we have our *runAssociationRules.m*, which runs all these parts and times then. We did this, not to have to run the getFrequentItems part multiple times. Note that we did not change anything about the code for these parts, if you want, you can run *associationRules.m* or *associationRulesOptimized.m* and you'll get the same results as in *runAssociationRules.m*

We ran this *runAssociationRules.m* multiple times with different minimum support and confidence, all while using the shopping list from the data file. The results are as follows:

```
>> runAssociationRules
Support = 0.0100 and Confidence 0.2000

Calculating frequent itemsets
Elapsed time is 4.212847 seconds.

Calculating confidence using original code
Elapsed time is 0.068562 seconds.
Original code: number of rules that satisfy is 234

Calculating confidence using optimized code
Elapsed time is 0.063073 seconds.
Optimized code: number of rules that satisfy is 234
```

```
>> runAssociationRules
Support = 0.0050 and Confidence 0.4000

Calculating frequent itemsets
Elapsed time is 5.919330 seconds.

Calculating confidence using original code
Elapsed time is 0.210321 seconds.
Original code: number of rules that satisfy is 270

Calculating confidence using optimized code
Elapsed time is 0.200370 seconds.
Optimized code: number of rules that satisfy is 270


>> runAssociationRules
Support = 0.0010 and Confidence 0.8000

Calculating frequent itemsets
Elapsed time is 90.255112 seconds.

Calculating confidence using original code
Elapsed time is 6.371908 seconds.
Original code: number of rules that satisfy is 413

Calculating confidence using optimized code
Elapsed time is 5.292086 seconds.
Optimized code: number of rules that satisfy is 413


>> runAssociationRules
Support = 0.0005 and Confidence 0.6000

Calculating frequent itemsets
Elapsed time is 751.712321 seconds.

Calculating confidence using original code
Elapsed time is 42.422338 seconds.
Original code: number of rules that satisfy is 31142

Calculating confidence using optimized code
Elapsed time is 33.755210 seconds.
Optimized code: number of rules that satisfy is 31142
```

From these results we can conclude that the optimized code indeed performs faster, if only by a small proportion. But the longer the code needs to run, the larger the difference becomes.

4 Rules

How many rules satisfy the following input parameter values? support = 0.001, confidence = 0.8

The number of rules that satisfy support = 0.001 and confidence 0.8 is 413

5 Sorting the rules

Sort the association rules by confidence in descending order and list the top 30 rules.

After sorting the rules by confidence, the top 30 rules are as follows:

1. flour,root vegetables,whipped/sour cream − > whole milk [0.002,1.00]
2. oil,other vegetables,root vegetables,yogurt − > whole milk [0.001,1.00]
3. rice,sugar − > whole milk [0.001,1.00]
4. butter,domestic eggs,other vegetables,whipped/sour cream − > whole milk [0.001,1.00]
5. citrus fruit,root vegetables,tropical fruit,whipped/sour cream − > other vegetables [0.001,1.00]
6. canned fish,hygiene articles − > whole milk [0.001,1.00]
7. brown bread,pip fruit,whipped/sour cream − > other vegetables [0.001,1.00]
8. cream cheese ,domestic eggs,napkins − > whole milk [0.001,1.00]
9. cream cheese ,domestic eggs,sugar − > whole milk [0.001,1.00]
10. bottled water,other vegetables,pip fruit,root vegetables − > whole milk [0.001,1.00]
11. ham,pip fruit,tropical fruit,whole milk − > other vegetables [0.001,1.00]
12. oil,root vegetables,tropical fruit,yogurt − > whole milk [0.001,1.00]
13. butter,domestic eggs,soft cheese − > whole milk [0.001,1.00]
14. butter,hygiene articles,pip fruit − > whole milk [0.001,1.00]
15. butter,rice,root vegetables − > whole milk [0.001,1.00]
16. citrus fruit,root vegetables,soft cheese − > other vegetables [0.001,1.00]
17. curd,domestic eggs,sugar − > whole milk [0.001,1.00]
18. hygiene articles,pip fruit,root vegetables − > whole milk [0.001,1.00]
19. hygiene articles,root vegetables,whipped/sour cream − > whole milk [0.001,1.00]
20. butter,fruit/vegetable juice,tropical fruit,whipped/sour cream − > other vegetables [0.001,1.00]
21. butter,other vegetables,pork,whipped/sour cream − > whole milk [0.001,1.00]
22. butter,other vegetables,root vegetables,white bread − > whole milk [0.001,1.00]
23. citrus fruit,pastry,rolls/buns,whipped/sour cream − > whole milk [0.001,1.00]
24. grapes,tropical fruit,whole milk,yogurt − > other vegetables [0.001,1.00]
25. ham,pip fruit,tropical fruit,yogurt − > other vegetables [0.001,1.00]
26. newspapers,rolls/buns,soda,whole milk − > other vegetables [0.001,1.00]
27. rolls/buns,root vegetables,sausage,tropical fruit − > whole milk [0.001,1.00]
28. oil,other vegetables,root vegetables,tropical fruit,yogurt − > whole milk [0.001,1.00]
29. cream cheese ,other vegetables,sugar − > whole milk [0.002,0.94]
30. root vegetables,sausage,tropical fruit,yogurt − > whole milk [0.002,0.94]