Data Mining 2: Frequent Pattern Mining Summer 2018
Christian Borgelt from 03.05.2018

Exercise Sheet 2

Exercise 5 Searching for Frequent Patterns

a) What is the size of the search space of frequent item set mining?
How does it grow with the number of items?

b) Why is an exhaustive search through all possible item sets not advisable?
What is the (asymptotic) time complexity of an exhaustive search?
Is the pruned/reduced search fundamentally more efficient?
If yes, in what way? If no, why may it be useful nevertheless?

Exercise 6 Searching for Frequent Patterns

a) How many subsets of size $k-1$ does an item set with k items have?
Why is this a problem for the candidate generation?

b) Show why any frequent item set of size $k+1$ can be formed in $j = (k+1)k/2$ possible ways with the original version of the Apriori algorithm!

c) In how many ways can an *in*frequent item set of size $k+1$ be generated?
Does this number differ from the answer for frequent item sets? Why?

Exercise 7 Searching for Frequent Patterns

a) What structure of the search space are we exploiting for the search?
Why is this structure advantageous given the properties of support?

b) What is a Hasse diagram? What does it represent?
For what are we using Hasse diagrams?

c) How is the Hasse diagram transformed to improve the search?
What is the purpose of this transformation?

d) What do we achieve by assigning unique parent item sets?
Could we also assign unique child item sets?

e) Are there any item sets that always have a unique parent item set,
even in the unmodified Hasse diagram?

Exercise 8 Searching for Frequent Patterns

a) How can we actually define unique parent item sets (concrete method)?
Are there alternatives? How do unique parents help in the search?

b) After assigning unique parent item sets: In how many ways can we generate an item set of size $k+1$ from item sets of size k?

c) What is the general scheme of searching with unique parents?

Exercise 9 Canonical Form of Item Sets

a) What is a canonical form of an item set? How can it be defined?
Is this concept actually needed for frequent item set mining?
Why did we look at canonical forms for item sets anyway?

b) What is the prefix property? How is it stated (two versions)?
What advantage results from the prefix property?

c) Describe how the recursive search for frequent patterns can be simplified if the chosen canonical form has the prefix property!

d) The prefix property is a necessary condition to ensure that the search by appending items to canonical code words is exhaustive. What is needed in addition to make it sufficient?

e) What is a canonical extension rule?
Is there a canonical extension rule for frequent item set mining?