ID2222 Lab 2 report

Fynn van Westen and Axel Myrberger

November 2021

1 The solution and instructions

We have created two different scripts to solve this task. First we have run.py and algo_lib.py. Run.py is used to run the program, with input parameters and triggering of the algorithms in algo_lib.py contains the Apriori algorithm and the class to generate association rules. The Apriori class includes a check_support function, ck_generate and run. Part 2 of the task is solved with the Gen_aRules class that generates association rules. This includes a function to check against the confidence values check_conf and a function run. The outputs of both classes are saved in class attributes.

To run the program simply run the script run.py and change the location of the data set if needed. The location is available as a variable in run.py file. The input variables are dataset-file, suport and confidence. For these variables there is also default values, for the support 1% and for confidence 50%. In the run file the running time of the algorithms are also captured and printed.

Modules used:

- time
- argparse
- os
- numpy
- numpy
- itertools
- collections

2 Results

The suggested dataset from the assignment page was used for the following results.

With support level = 1% we found the following:

1. Length = 375
2. length = 9
3. length = 1

The last set of numbers is 3: (39, 704, 825): 1035 where 1035 is the number of appearances, and the tuple (39, 704, 825) contains the item ID's of the frequent set.

The total time for this algorithm was 10.8 seconds on a laptop CPU.

For part two we found the following results:

- (704,) gives 39
- (227,) gives 390
- (704,) gives 825
- (39,704) gives 825
- (39,825) gives 704
- (704,825) gives 39

This solution took only 0.03 seconds.