# Homework 2: Finding frequent item sets and associations rules

Mauro Pungo, Viktoria Sartor

November 2021

## 1 Build instructions

In order to execute the program, place the file ”T10I4D100K.dat” on the same directory as the file homework2.py and execute *python homework2.py*

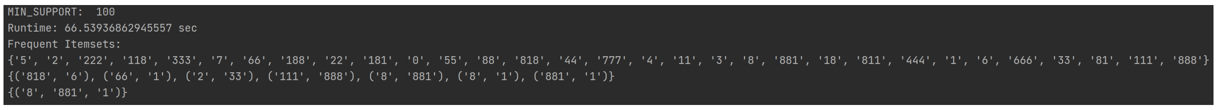
## 2 Task 1: A-priori algorithm

### 2.1 Explanation

The A-priori algorithm finds sets of items that occur frequently in a database. The metric used to define how frequent a set appears is called the support. An item set is said to be frequent on the database if its support is greater than a predefined minimum support threshold. After each iteration we delete itemset that have itemsets in them that were not frequent in the last iteration. This way only combinations if previously frequent itemsets will be used to create new itemsets. This did improve the runtime drasticly (from approx. 3h to less than 2 minutes).

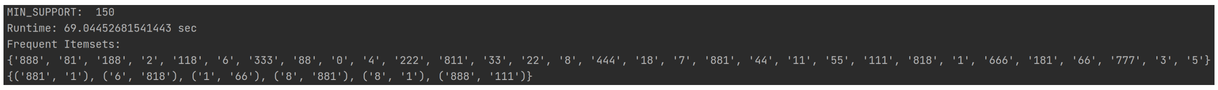
### 2.2 Results

In order to test the performance of the algorithm we used the provided transactions database. It contains 100k transactions of variable length. The chosen default value for the minimum support $s = [100, 150, 300, 1000]$ For those values, the algorithm runs in approximately $[66.54, 69.04, 57.94, 49.68]$ seconds. By increasing the value of the minimum support value, the runtime of the algorithm improves, at the cost of probably missing smaller size frequent item sets. There is no improvement between $s = 100$ and $s = 150$, because at this point we started to run the code in parallel due to time issues on our side.



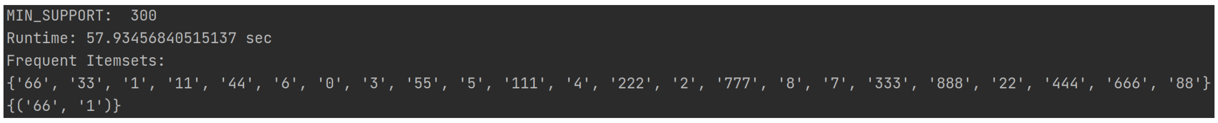
```
MIN_SUPPORT:  100
Runtime: 66.53936862945557 sec
Frequent Itemsets:
{'5', '2', '222', '118', '333', '7', '66', '188', '22', '181', '0', '55', '88', '818', '44', '777', '4', '11', '3', '8', '881', '18', '811', '444', '1', '6', '666', '33', '81', '111', '888'}
{('818', '6'), ('66', '1'), ('2', '33'), ('111', '888'), ('8', '881'), ('8', '1'), ('881', '1')}
{('8', '881', '1')}
```

Figure 1: min support = 100



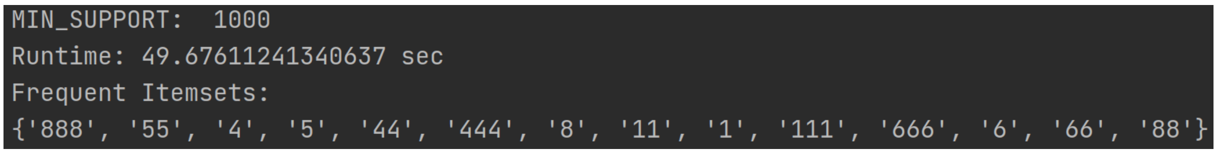
```
MIN_SUPPORT:  150
Runtime: 69.04452681541443 sec
Frequent Itemsets:
{'888', '81', '188', '2', '118', '6', '333', '88', '0', '4', '222', '811', '33', '22', '8', '444', '18', '7', '881', '44', '11', '55', '111', '818', '1', '666', '181', '66', '777', '3', '5'}
{('881', '1'), ('6', '818'), ('1', '66'), ('8', '881'), ('8', '1'), ('888', '111')}
```

Figure 2: min support = 150



```
MIN_SUPPORT:  300
Runtime: 57.93456840515137 sec
Frequent Itemsets:
{'66', '33', '1', '11', '44', '6', '0', '3', '55', '5', '111', '4', '222', '2', '777', '8', '7', '333', '888', '22', '444', '666', '88'}
{('66', '1')}
```

Figure 3: min support = 300



```
MIN_SUPPORT:  1000
Runtime: 49.67611241340637 sec
Frequent Itemsets:
{'888', '55', '4', '5', '44', '444', '8', '11', '1', '111', '666', '6', '66', '88'}
```

Figure 4: min support = 1000