METU Computer Engineering Department

# CENG 514 Data Mining

Spring 2020-2021

## Assignment I

In this assignment we will work on Association Rule Mining and elaborate on two related sub tasks.

**Q1.** Analyzing the pruning through candidate generation.

The pseudocode for the candidate generation step of Apriori algorithm is as follows:

```
Given a set of k-item frequent itemsets I as input
(Note that  k>1 and items within itemsets and also itemsets are
already sorted in increasing order).

C is the set of k-item Candidate Itemsets, initially empty

For each itemset i in I

       For each itemset j in I

             If the initial k-1 elements of itemset i and j are the
same,
                   create a k+1-item candidate itemset c
                   If c does not include any k-item non-frequent
itemset
                   C = C U c
Output C
```

***Example:***
Input {a,b}, {a,c}, {a,d}, {b,c}, {c,d}, {c,e}, {d,e}
Output should be {a,b,c}, {a,c,d},{c,d,e}

Note that {a,b} and {a,d} can be joined, as their initial k elements are the same, to form the candidate {a,b,d}. However it includes the {b,d} as a subset, which is a non-frequent 2-item itemset.

Implement the below algorithms in Python:

i. The candidate generation algorithm above
ii. Candidate generation algorithm by extending each k-item frequent itemset by an item in the data set

We expect to obtain pruning in candidates in the first algorithm. In order to analyze this:

i. Run both of the algorithms on 50 randomly created k-item frequent itemsets I. (Use the same input for both of the algorithms.)
ii. Report the reduction in the number of candidates generated per run and the average reduction.

Note that the number of reduction in candidate generation will be calculated as given in Equation 1.

$$Reduction = \frac{Number\ of\ candidates\ generated\ in\ Algorithm\ 1}{Number\ of\ candidates\ generated\ i\ Algorithm\ 2} \qquad (1)$$

Submission: Codes and the report presenting the analysis result.

**Q2**. Analyzing the Partitioning (SON) Algorithm.

As discussed in the class, SON algorithm partitions the transaction data set into chunks fitting in the memory and returns the complete result in two passes on the data. In this part of the assignment, you will simulate the first pass of the algorithm and compare the result with that of the original Apriori.

Follow the given steps:

i. For the analysis, use groceries.csv data set
ii. Run apriori function under mlxtend library in Python on groceries.csv data set under 0.02 support ratio.
iii. For i in 5 to 10
   a. Partition groceries.csv data set into i partitions
   b. Find the total number of frequent itemsets generated for all the partitions
   c. Compare the result with that of Step (ii) and report as a ratio given in Equation 2.

$$Over\ generation\ ration = \frac{Number\ of\ frequent\ itemsets\ generated\ in\ the\ first\ pass\ of\ SON}{Number\ of\ frequent\ items\ generated\ in\ the\ original\ Apriori} \qquad (2)$$

Note that apriori in mlxtend gets transaction data input in the form of one hot encoding. So you will need to convert each row in the original data set to a list, and then to one hot encoding format. You can use TransactionEncoder under mlxtend.preprocessing.

Submission: Codes and the report presenting the analysis result.