Data Mining - ID2222

Homework 2

Discovery of Frequent Itemsets and Association Rules

André Silva - Jérémy Navarro

November 16, 2020

1 Overview

We chose *Python* as the programming language and an OOP approach for the development of the project.

Information about the structure of the project, as well as instructions on how to run it follow:

```
$ unzip homework2.zip
$ cd homework2/
$ python3 main.py
```

The project is composed with:

- `classes.py`: Contains the classes implementing the several stages of finding frequent itemsets and association rules.
- `main.py`: The executable script which performs experimentation on the dataset.
- `T10I4D100K.dat`: The dataset.
- `out_s_0.1`: Output for $s = 0.1$ and $c = 0.5$

2 Classes

2.1 FrequentItemsets

The class **FrequentItemsets** takes 2 arguments:

- `baskets`: List of transactions
- `s`: Support threshold

We start off by computing the list of frequent 1-itemsets (i.e. the itemsets of size 1 which have a support greater or equal than the threshold), which is used in each of the following steps. We'll call this L_1.

Once we have computed L_1, we can generate C_2. C_2 is the list of candidate 2-itemsets, generated by the combination of elements of L_1 with L_1, given that all subsets of each candidate are frequent. We then filter according to the threshold to output L_2.

This process is repeated for all $k \leq \max(|basket|)$, so that we can find all frequent itemsets for a given transaction dataset.

2.2 (*Bonus*) AssociationRules

The class **LSH** takes 3 arguments:

- `buckets`: List of transactions.
- `c`: Confidence threshold.
- `frequentItems`: List of frequent itemsets.

We start off, given the list of frequent itemsets, by generating all possible candidate association rules whose right part has a size of 1. We then compute, for each candidate, the confidence value and save it if it is higher than the threshold.

After the first iteration, we can now generate all possible candidate association rules whose right part has a size of 2. This is done by moving a single element from the left to the right part.

The same process is repeated until we reach a k for which all $k-1$ candidate association rules have left parts of, at most, size 1.