CSOE18 Assignment #1: Frequent Pattern Mining

Due: **11:59 pm, 20 October 2023**. Focus must be on learning and discovering new stuff. Enjoy!

Objective: The objective of this project is to learn the concepts of Frequent Pattern Mining and to implement it for a real time dataset.

General Instructions:
Total Points: **20**
This assignment needs to be done individually and implemented in Python.
As always, please feel free to approach me in person in office hours or through Teams, if you have any questions. I'd be happy to help out.

Marking Criteria: Your submission will be marked using the following criteria.

- Clarity of the code with proper indentation and comments.
- Showing good efforts through completed tasks.
- Optimization of the code and working with other datasets not mentioned in the assignment.
- Showing attention to details through a good quality project report.

Your code will be run on Anti-plagiarism software. Copying others contents will be seriously viewed, which will lead to heavy penalties for both the donor and the recipient. I encourage you, not to rush at the last moment.

Dataset Description

The dataset ("categories.txt") in this folder consists of the category lists of 77,185 places in the United States. Each line corresponds to the category list of one place, where the list consists of a number of category instances (e.g., hotels, restaurants, etc.) that are separated by semicolons. An example line is provided below:

Local Services; IT Services & Computer Repair

In the example above, the corresponding place has two category instances: "Local Services" and "IT Services & Computer Repair".

Tasks

1. You need to implement the Apriori algorithm (Frequent Pattern Mining) and use it to mine category sets that are frequent in the input data. After implementing the Apriori algorithm, please set the relative minimum support to 0.01 and run it on the 77,185 category lists.
In other words, you need to extract all the category sets that have an absolute support no smaller than 771.

a. Please output all the length-1 frequent categories (item sets) with their absolute supports in the descending order of their support count, into a text file named *patterns_1.txt*. Also report the total count of the frequent item sets. Every line corresponds to exactly one frequent category and should be in the following format: (7 points including implementing Apriori)
#Total count
Category : Support

For example, suppose a category (Fast Food) has an absolute support 3000, then the line corresponding to this frequent category set in *patterns.txt* should be:
Fast Food : 3000

b. Write all the frequent category sets along with their absolute supports in the descending order of their support count, into a text file named *patterns_all.txt*. Also report the total count of the frequent item sets. Every line corresponds to exactly one frequent category set and should be in the following format: (4 points)

#Total count
Category_1,category_2,category_3,... : Support

For example, suppose a category set (Fast Food; Restaurants) has an absolute support 2851, then the line corresponding to this frequent category set in *patterns_all.txt* should be:
#Total count
Fast Food; Restaurants : 2851

2. Mine the set of all Closed Frequent Itemsets (CFI's) for the relative minimum support of 0.01 that you used to mine the frequent item sets and write the output in the descending order of support count, into a file named *patterns_close.txt* in the following format. Also report the total count of the frequent item sets. (4 points)
Category : Support
#Total_count

What is expected?

- Use either Merge Sort/Quick Sort/Heap sort for your sorting approach.
- Your code should have proper indentation and comments.

Write a brief report in PDF format containing the following (5 points):

a. A brief summary of your results on the given dataset and other datasets, if you have tried.
b. Discuss, if any, the algorithmic optimizations you have used in your implementation.
c. Discuss the experiences and lessons you learnt from the implementation.
d. References you have used.

Other Logistics:

This is to be completed and submitted to Moodle. By the due date, submit one zip file containing:

- Source file, Executable file, a readme file explaining how to compile/run your program.
- Three '.txt' files containing your outputs.
- Name of the folder should be your roll number and assignment number, For example – "**123456789_Assignment_1.zip**".
- Include any other reference materials you have used in this assignment.

END OF ASSIGNMENT