CSCI 5702/7702 – Data Mining and Analytics
Fall 2015 - Assignment 3

Due: 10/30/2015 8:59pm

Project Objectives

- To implement the basic APRIORI algorithm and test it under different configurations;
- To analyze the basic APRIORI algorithm performance.

Important notes before you start

- This is an extensive assignment and takes more time to do than previous assignments, so please start early!
- This is an individual assignment, which means it is OK to discuss with your classmates, but it is not OK to work together or share code. All codes will be checked manually and by using the MOSS script.
- Similar libraries or programs of frequent pattern mining can be found on-line, but you are prohibited to use these resources, which means you cannot include public libraries, or modify existing programs, since the purpose of this programming assignment is to help you understand and implement frequent pattern mining processing step by step. You need to develop your code from the scratch.
- Use Java as your programming language.
- If you would like to use C++ or Python as your programming language, you need to email the TA to get instructions; otherwise, your submission will not be graded.

Project Description (100 Points)

Roadmap for implementing this assignment

1. Know Your Data.
2. Download and install a JAVA IDE, such as **Eclipse** or **NetBeans**.
3. Complete the provided JAVA application. You need to implement the basic APRIORI algorithm to find frequent patterns from the provided datasets.
4. Test your application under the provided configurations in Section 4 and analyze the results.

1. Know Your Data

There are 3 datasets in the assignment package under the "*Datasets*" folder which are named as "*dataset1.txt*", "*dataset2.txt*", and "*dataset3.txt*". All of them are subsets of the "*retail*" dataset which contains retail market basket data from an anonymous Belgian retail store. The "*retail*" dataset was originally used in the following paper:

- Brijs T., Swinnen G., Vanhoof K., and Wets G. (1999), The use of association rules for product assortment decisions: a case study, in: Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining, San Diego (USA), August 15-18, pp. 254-260. ISBN: 1-58113-143-7.

Each line in the .txt file shows the items of one transaction (transaction IDs are not provided) and items are separated by a space " ". For example the first line in *dataset1* is: 30 31 32 which means the first transaction contains items 30, 31, and 32.
More information about each dataset is provided in **table 2** where:

- |T| shows the number of transactions in the dataset.

Dataset	|T|
dataset1	100
dataset2	400
dataset3	800

Table 2 – Datasets Properties

2. Download and Install a JAVA IDE

You could use any standard JAVA IDE of your preference. You are not limited to the Eclipse IDE.

- For Eclipse:
    1. Download JAVA JDK from here.in
    2. Install JAVA JDK on your machine. You could find installation tutorials for different operating systems below:
        - Windows
        - Linux (Ubuntu)
        - Mac OS
    3. You can download the Eclipse IDE (JAVA developer edition) from here.

3. Implement the Basic APRIORI Algorithm

Your application must contain 3 classes:

- *CandidateGen.JAVA: which generates potential frequent candidates (C_K) for each level/round of the iterative APRIORI algorithm*
- *SupportCounter.JAVA: which computes the support count for generated candidates and generates L_K.*
- *Main.JAVA: which contains the main method (your application execution will start from this class) and calls in other components.*

3.1. Input

Assume the following inputs for your application:

- *minSupport: you need to define it as a constant in your main class. It is in the form of **frequency count** and must be an integer. For example if minSupport=3, the itemset "1 3 5" will be frequent if at least 3 transactions in the dataset contain it.*
- *Dataset location: your application needs to obtain the address where the dataset file is located as a string. You need to define the address variable as a constant.*
- *Output location: your application must generate and save all output files in this directory inside of your application named "**Output**".*

3.2. Output:

Your application needs to generate the following files:

- *Ci.txt: which contains,* ***the number of generated candidates, computation time for*** C_i ***(in Nano seconds)*** *and* ***the list of all generated candidates*** *in the* ***i****th iteration. Your application must generate* ***k separate files*** *where* ***k*** *is the length of the longest frequent itemset. For example if k=3, 3 output files must be generated: C1.txt, C2.txt, and C3.txt*
- *Li.txt: which contains,* ***the number of frequent candidates in the iteration****,* ***computation time for*** L_i ***(in Nano seconds)*** *and* ***the list of all frequent candidates*** *in the* ***i****th iteration. Your application must generate* ***k separate files*** *where* ***k*** *is the length of the longest frequent itemset. For example if k=3, 3 output files must be generated: L1.txt, L2.txt, and L3.txt*

- *Summary.txt: which contains the following items:*
    - *The minSupport value.*
    - *Total computation time for the CandidateGenerator* ($\sum_{i=1}^{k} T(C)_i$).
    - *Total computation time for the SupportCoutner* ($\sum_{i=1}^{k} T(L)_i$).
    - *Total computation time* = $\sum_{i=1}^{k} T(C)_i + \sum_{i=1}^{k} T(L)_i$.
    - *Total number of frequent itemsets.*
    - *Total number of infrequent itemsets.*

- *Frequent.txt: which contains the list of all* ***frequent itemsets*** *and their* ***support counts****.*
- *Infrequent.txt: which contains the list of all itemsets generated by the CandidateGenerator which turned out to be* ***infrequent,*** *along with their* ***support counts****.*

❖ Your output should follows the same **format** as the example output files located under the ***"Output"*** folder of the assignment 3 package; otherwise you will not receive any points for this assignment.

4. Analyze the APRIORI Algorithm

In the following, test your application using the specified configurations, analyze the results according to the provided instructions **i** to **iii**, and then answer the corresponding question according to your analysis.

A. Test your application with ***dataset1***, ***dataset2*** and ***dataset3*** (***minSupport =6***). Analyze the result according to **i** to **iii** below, and explain how the ***dataset size*** affects the ***computation time***.

- **i.** Draw a chart in which the Y axis shows the computation time in Nano seconds for the ***candidateGen*** component while the X axis shows the dataset size.
- **ii.** Draw a chart in which the Y axis shows the computation time in Nano seconds for the ***SupportCounter*** component while the X axis shows the dataset size.
- **iii.** Draw a chart in which the Y axis shows the ***total computation time*** in seconds (candidate generator + frequent pattern miner) while the X axis shows the dataset size.

B. Test your application with ***minSupport = 3, 4, and 5*** (using ***dataset2***). Analyze the result according to **i** to **iii** below, explain how the ***minSupport*** value affects the ***computation time.***

- **i.** Draw a chart in which the Y axis show the computation time in Nano seconds for the ***candidateGen*** component while the X axis shows the minSupport value.
- **ii.** Draw a chart in which the Y axis show the computation time in seconds for the ***SupportCounter*** component while the X axis shows the minSupport value.
- **iii.** Draw a chart in which the Y axis show the total computation time in Nano seconds while the X axis shows the minSupport value.

Submission Guideline

Please submit your assignment through **Canvas** before the deadline; late submissions are not accepted! You are allowed to submit your assignment multiple times, but only the last submission (**before the deadline**) will be recorded and graded.

Your submission should be a single **zip** file named ***<Your CU Denver Portal ID>-A3.zip***. For example, if your CU Denver Portal ID is "*john*", the file name would be: ***john-A3.zip***.

Your submission must contain the following materials:

- ***Code directory****: which contains your implemented JAVA code.*
- ***Output directory****: which contains **2** directories as follows:*
  - ***A directory****: which contains 3 folders named "dataset1", "dataset2" and "dataset3". Each directory contains all **.txt** output files described in Section 3.2 for the **Part A** of Section 4.*
  - ***B directory****: which contains 3 folders named "minSup3", "minSup4" and "minSup5". Each directory contains all **.txt** output files described in Section 3.2 for the **Part B** of Section 4.*
- ***Analysis.pdf****: which contains your answers and charts for parts A and B of Section 4.*

Notes:

- If your output is not correct, you will not receive any points for your code.

- Please download your assignment after submission and make sure it is not corrupt. We won't be responsible for the corrupted submissions and will not be able to take a resubmission after the deadline.

  You are highly encouraged to ask your question on Piazza under the "a3" folder. Please DO NOT include your solutions in the comments you share on Piazza. Feel free to help other students with general questions.