CS8803 BDS / CS4365

Homework Assignment 3

(Programming Category)

Student Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student Session: cs8803 or CS4365 (circle one)

In this third programming assignment, you are given one type of programming problem to gain hand-on experience with Frequent Pattern Mining (FRM). There are three problems and you only need to choose one.

1. The first problem can be solved by hand, though you are encouraged to try to write a program to solve it.
2. The second problem requires you to run two open source implementation of the FRM algorithms and compare their performance.
3. The third problem asks you to implement an algorithm for FRM with an option to include some optimization technique(s).

Feel free to choose any of your favorite programming language: Java, C, Perl, Python. This assignment is designed to help you understand and use Data Mining and Machine Learning packages in Mahout/R/DEKA and also to encourage you to design some optimizations for the baseline algorithms.

**Post Date**: Monday of Week 8 (Oct. 5)

**Due Date**: midnight on Friday of Week 10 (Oct. 16 with no penalty grace period until midnight on Saturday of Oct. 17)

**Problem 1. Learning Association Rule Mining by Example**

Consider the following transaction database:

|  |  |
| --- | --- |
| **Transaction ID** | **Items** |
| T1 | A, B, C, D |
| T2 | A, B, C, E |
| T3 | A, B, E, F, H |
| T4 | A, C, H |

Suppose that minimum support is set to 50% and minimum confidence to 60%.

You are asked to answer the following four questions. You can answer all five subquestions by hand. You are encouraged to write a baseline program to answer the first four questions though it is optional. You can also find some source code at <http://fimi.ua.ac.be/src/> and WEKA/R/Mahout website.

a) List all frequent itemsets together with their support.

b) Which of the itemsets from a) are closed? Which of the itemsets from a) are maximal?

c) For all frequent itemsets of maximal length, list all corresponding association rules satisfying the requirements on (minimum support and) minimum confidence together with their confidence.

d) Using the lift of an association rule defined as follows to compute the lift for the association rules from c).

lift = confidence / support(head)

e) Why are only those association rules interesting that have a lift (significantly) larger than 1.0? [Hint: the following equations are true]

lift = confidence / support(head)

= confidence / ( support(body) \* support(head) / support(body) )

= confidence / expected\_confidence

**Problem 2. Hand-on Experience with** **Frequent Pattern Mining Algorithms**

Your task for this assignment is to compare and evaluate two different implementations of the Apriori algorithm for frequent itemsets mining. There are a few online repositories for frequent pattern mining implementations, most notably [FIMI repository](http://fimi.cs.helsinki.fi/) (http://fimi.ua.ac.be/src/). You can study them.

You are asked to choose two algorithms from open source packages, such as [FIMI repository](http://fimi.cs.helsinki.fi/) (<http://fimi.ua.ac.be/src/>) or WEKA or Mahout, and run the two algorithms on one or two datasets of different sizes and compare their performance and illustrate the comparison results. You can find the datasets from [frequent itemset mining datasets](http://fimi.cs.helsinki.fi/data/) at <http://fimi.ua.ac.be/data/>.

**Deliverable:**

Write a brief report in PDF or word to present your results on the test dataset and other datasets if you have experimented with.  Explain and discuss the results.

Your submission should be a zip or tar file that contains the PDF or word report as well as the program deliverables, including your source files, the executable, a readme file explaining how to compile/run your program, the output file for the test dataset, and the setting of your minimum support for the results.

**Problem 3. Implementing a** **Frequent Pattern Mining Algorithm**

Your task for this problem is to implement and evaluate the Apriori-based algorithm for frequent itemsets mining, originally proposed by Agrawal et al. for frequent itemsets mining (see <http://rakesh.agrawal-family.com/papers/vldb94apriori.pdf> ). You can find the pseudocode and its related procedures from course note (Lecture 9). You can use any programming language that you are familiar with.

The program should be executable with 3 parameters: the name of the input dataset file, the threshold of minimum support count, and the name of the output file.  The minimum support count should be an integer.  An itemset is frequent if its support count is larger or equal to this threshold. The program should output a file that contains all the frequent itemsets together with their support.  The output file should have the following format: each line contains a single frequent itemset as a list of items separated by whitespace.  At the end of the line, its support is printed between a pair of parenthesis.  For example: A B C (5) represents an itemset containing items A, B and C with a support count of 5.

You are encouraged to use existing or your own optimization techniques for the Apriori algorithm. If you do, explain and discuss the techniques you have used and/or provide the appropriate references in the report.

Test your implementation on a dataset. You can choose one of the [frequent itemset mining datasets](http://fimi.cs.helsinki.fi/data/) at <http://fimi.ua.ac.be/data/>.

There are a few online repositories for frequent pattern mining implementations, most notably [FIMI repository](http://fimi.cs.helsinki.fi/) (http://fimi.ua.ac.be/src/). You can study them.

**Deliverable.**

Write a brief report in PDF or word to present your results on the test dataset and other datasets if you have experimented with.  Explain and discuss the results. If the algorithmic optimizations are used in your implementation or the algorithms you used for evaluation and comparison, discuss the experiences and lessons you have learned from the implementation.

Your submission should be a zip or tar file that contains the PDF or word report as well as the program deliverables, including your source files, the executable, a readme file explaining how to compile/run your program, the output file for the test dataset, and the setting of your minimum support for the results.