CSC440 Data Mining Project 1
Siyu Nan

Readme

The program is written under python 3.5.2. Editor is Spyder. Command line tool is Terminal.

For apriori.py,
If you are using command line tool, then type python apriori.py. If editor, simply run it. There is a timer in the main function. If you delete three '#' I put in the main function located close to the end of the script, running time will be shown. You can also set min_sup in the main function located close to the end of the script to any number from 0 to 1.
Output file: apriori_output.csv
With different min_sup setting, the program may takes up to 4 minutes to run.

For improved_apriori.py
If you are using command line tool, then type python improved_apriori.py. If editor, simply run it. There is a timer in the main function. If you delete three '#' I put in the main function located close to the end of the script, running time will be shown. You can also set min_sup in the main function located close to the end of the script to any number from 0 to 1.
Output file: improved_apriori_output.csv
With different min_sup setting, the program may takes up to 4 minutes to run.

For fpgrowth.py
If you are using command line tool, then type python fpgrowth.py. If editor, simply run it. There is a timer in the main function. If you delete three '#' I put in the main function located close to the end of the script, running time will be shown. You can also set min_sup in the main function located close to the end of the script to any number from 0 to 1.
Output file: fpgrowth_output.csv
With different min_sup setting, the program may takes up to 2 seconds to run.

Report

The time that takes the Apriori, FPGrowth, and the improved algorithm to process the UCI adult_data.csv comes in the following order: FPGrowth < improved algorithm < Apriori. In my programs, set min_sup being 0.5 (you can set it to any number in (0,1)). Apriori takes around 2'50'', improved_apriori takes around 2'10'', fp-growth takes less than 2 second.

As indicated in book, Apriori takes longer since it creats all possible candidate sets and need multiple scans through translist to see if candidates meet min_sup whereas fp growth only needs two scans. It constructs tree structure and mine it recursively.
The improved edition decrease number of scans required. Starting from L2, we only scan through transactions having the item with the min support count in the n-itemset. A simple example is (x,y) is one of the 2-itemset of C2 and sup_count(x)<sup_count(y). We only scan transaction whose ID is related to x. Since we need both x and y in transaction to make it L2, there is no point of doing extra scanning on those with y but not x.
Although the number of scans has greatly decreased comparing to the Apriori algorithm, it still requires more than two scans, thus, required much more time than the fp-growth algorithm.