CS6530 Analytic Databases

Assignment: Frequent Pattern Mining

Marks: 60

This assignment must be done individually.

Start Date: 08-03-2018

Due Date: 02-04-2018

Overview

This project is aimed at developing your own frequent pattern mining algorithm, and comparing the performance/result of multiple frequent pattern mining algorithms. You would use SPMF Library for this assignment. We recommend you to do the assignment using Java so that your implementation can have comparable performance as that of SPMF library. You can also do it in R/Python.

We provide you with two retail datasets: retail1 and retail2. Dataset retail1 contains 541,909 transactions and 2603 items. The meaning of each item is given in the file retail1Attributes.xlsx. You can use the item names in retail1Attributes.xlsx to interpret your results. Dataset retail2 is a bigger dataset with more number of items per transaction. For this dataset the item names are not known. Both these datasets are taken from here.

Exercises

Exercise 1 [15 points]: Compare the performance of the following three frequent itemset mining algorithms: (1) Apriori, (2) FP-Growth, and (3) ECLat (mining using vertical data format). Compare the performance of each algorithm for the two retail data sets. Analyze the computation time of the three algorithm by varying the minimum support threshold. You should plot two line-graphs, one for each dataset. In each graph, on the x-axis you should vary the minimum support and on the y-axis plot the computation time taken by the three algorithms. You should learn the situations where one algorithm performs better than the others. (All the algorithms are available in SPMF library)

Exercise 2 [15 points]: Compare the performance of the following three frequent itemset mining algorithm: (1) FP-Growth, (2) FPClose, and (3) FPMax, which mine the frequent itemset, frequent closed itemsets and frequent maximal itemsets respectively (All the algorithms are available in SPMF library). Compare the performance of each algorithm for the two retail data sets. Analyze the computation time of the three algorithm by varying the minimum support threshold. For each dataset, present your result in the form of table as shown below:

Note: You can change the value of min-sup if you feel the above values are too low or too high. This will depend on the dataset.

	min-sup = 0.5%	min-sup = 1%	min-sup = 2%	min-sup = 5%
FP-Growth	Time	Time	Time	Time
	#itemsets	#itemsets	#itemsets	#itemsets
FPClose	Time	Time	Time	Time
	#itemsets	#itemsets	#itemsets	#itemsets
FPMax	Time	Time	Time	Time
	#itemsets	#itemsets	#itemsets	#itemsets

Exercise 3: Implement the MS-Apriori algorithm discussed in the class. (See Section 2.4 in the book Web Data Mining). You can exclude the rule generation step. You need to consider the following:

- Multiple minimum support
- Support difference constraint, and
- Item constraint

To set the minimum support for each item you should do the following: Assign a MIS value to each item according to its actual support/frequency in the data set T. For example, if the actual support of item i in T is $\sup(i)$, then the MIS value for i may be computed with $\delta \times \sup(i)$, where δ is a parameter $(0 \leq \delta \leq 1)$ and is the same for all items in T.

You should consider following two types of item constraints:

- *Cannot–be-together*: sets of items cannot be in the same itemsets (in pairwise manner). For example, if the two itemsets {1, 2, 3} and {6, 7, 9 10} have cannot-be-together constraint, then itemset like {1, 2, 3, 6, 7, 9 10, 11, 20} is not allowed but {1, 2, 6, 7, 9 10, 11, 20} is fine.
- *Must-have*: Every itemset must have these items. For example must-have (1 or 2) would mean that every itemset must have the items with id 1, 2 or both 1 and 2.

Exercise 4: Use the MSApriori algorithm available in the SPMF library. Unlike Exercise 3, here you only need to consider the multiple minimum support and support difference constraint. Generate the multiple minimum support using the approach described in Exercise 3.

Note: Support difference constraint is not part of SPMF. Either the same can be implemented by changing candidate gen in SPMF or take the result from SPMF and apply the constraint on the result.

Exercise 5 [30 points]: For Exercise 3 and 4 assume support difference constraint $\varphi = 5\%$ and the minimum support parameter $\delta = 0.5$. Feel free to change these values if you feel they are too high or too low. Report your used value in the report. (For this exercise use only the small retail1 dataset)

a. Take the top five itemsets (from the FP growth algorithm) with maximum support from Exercise 1. Call these the *cannot-be-together* sets. From each of these five sets take the item with least item id and take their union. Call this your *must-have* set. What is your *cannot-be-together* and *must-have* set? (Ignore the itemsets of size 1. Consider itemsets with at least 2 or 3 items.)
b. Use the above *cannot-be-together* and *must-have* set to obtain the frequent itemsets using Exercise 3. Report the number of obtained frequent itemsets?
c. Report the number of frequent itemsets you obtain from Exercise 4? How does this number compare with the number of itemsets that you obtain from your implementation of MS-Apriori in Exercise 3 (just ignore the item constraints)?
d. You can use SPMF library to implement Exercise 3 by applying the item constraint as an additional filtering on the result that you obtain from Exercise 4. Report the computation time that your code takes in Exercise 3 for both with and without item constraint. Compare it with the computation time taken by the implementation in SPMF library.

Notes

- Getting out of heap memory error while using SPMF: Allocate more memory to JVM
- Deliverables - a) Source Code b) Report based on the assignment