CS412 Report

Zhuang, Yiwei

1. Briefly explain what algorithms you use in Step4~Step6.

For step4, I use the apriori algorithm, which will generate the frequent pattern by previous level and prune the combination that does not meet the support until no new pattern could be generated.

For step5, basically I first read the file, which is generated by apriori algorithm, pattern-i.txt, then do some data processing. From the definition of closed pattern, we will not have a pattern that is the sub-pattern in the file with same support. Based on this condition, I iterate over the file and do some prune. For max pattern, we will not have a pattern that is a sub-pattern of the other. So I do the same thing but not checking the support as closed-pattern. There is a function called contain sublist, which will check two lists whether one is the sublist of the other.

For step6, I first use the equation on the website calculate the purity for each pattern, basically the value ranges from -1.1 to 1.9 while most value is in the range of 0.8 to 1.2. As a result, I combine this value with support to get the new sequence and consider purity as a factor. I replace old support by support \* purity^3 then I sort this value to see new pattern. The new pattern makes some distinct pattern in original file with high support to become the top patterns.

1. Answer all the questions in *Question to ponder*.

***Question to ponder A****: How you choose min\_sup for this task? Note that we prefer min\_sup to be the consistent percentage (e.g. 0.05 / 5%) w.r.t. number of lines in topic files to cope with various-length topic files.*

*Explain how you choose the min\_sup in your report. Any reasonable choice will be fine.*

Here, I choose my min support to be about 1%. For each topic file, I print their length and find they are around 8000 words in each. So I think 1%, which makes the support bigger than about 80 will be a good implication for a frequent pattern

***Question to ponder B:*** *Can you figure out which topic corresponds to which domain based on patterns you mine? Write your observations in the report.*

Pattern0: Machine Learning

Pattern1: Data Mining

Pattern2: Theory

Pattern3: Information Retrieval   
pattern4: Database

From Pattern-I files, we can see the top frequent words and patterns. For each pattern, I find out some words is frequent in this file while not frequent in other files. For example, in pattern-0, we can see word ‘machine’, ‘learning’ are frequent while not frequent in other files, so I label it as Machine Learning area. Furthermore, I can see it in my purity directory. So from my observation, I figure out their corresponding topic.

***Question to ponder C:*** *Compare the result of frequent patterns, maximal patterns and closed patterns, is the result satisfying? Write down your analysis.*

Yes. I think the result is satisfying. For generating those files, I directly use the files from pattern-i.txt, and do the prune. As a result, if my frequent pattern is implemented correctly, so does my closed pattern and max pattern.

1. List your source file names and their corresponding Steps.

Preprocessing.py

Reorganize.py

Apriori.py

Max\_close.py

Re\_rank.py

Main.py

For configuration, we need pandas in python.

For running steps, first running Preprocessing.py will generate the “vocab.txt” and dictionary “title.txt” in step2, then running reorganize.py will generate the topic-i.txt for 5 files. Then running Main.py will call function in corresponding file and then generate frequent pattern, closed pattern, max pattern, purity pattern files. You can change the min support in Main.py at top. In Main.py , it will first generate the term index of frequent pattern then transfer them to a words.