CS412 Data Mining

MP3 Report

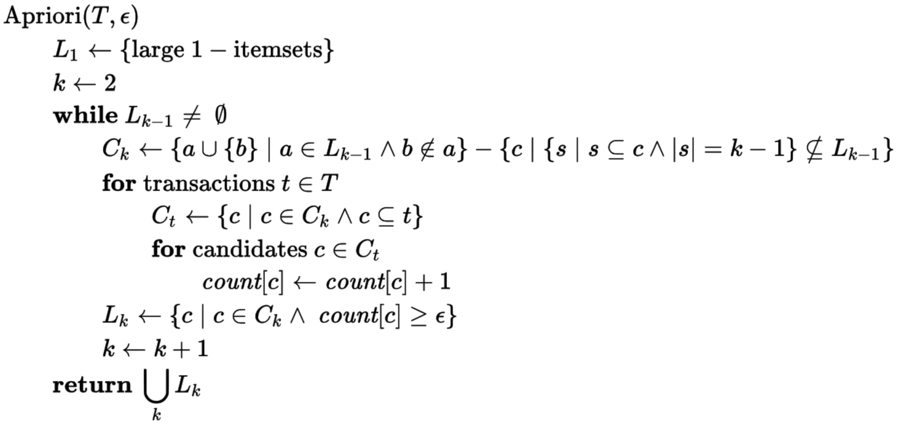
Bangqi Wang

Bwang34

Nov.3, 2016

Algorithm

Step4 uses the apriori algorithm to find the frequent pattern. The algorithm first searched for the 1-item sets and pruned the pattern below the min support frequency. The algorithm stored the remaining pattern in remaining list and the pruned pattern in pruning list. Then the algorithm constructed the longer pattern by combining the pattern with less length. For pattern in level k, the algorithm constructed the pattern from pattern sets in level k-1 and pruned the patterns that contained the sub-pattern already pruned by level k-1. The code below is the pseudocode from Wiki page.

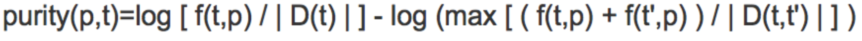


```
Apriori($T, \epsilon$)
    $L_1 \gets \{\text{large } 1 - \text{itemsets}\}$
    $k \gets 2$
    while $L_{k-1} \neq \emptyset$
        $C_k \gets \{a \cup \{b\} \mid a \in L_{k-1} \wedge b \notin a\} - \{c \mid \{s \mid s \subseteq c \wedge |s| = k-1\} \nsubseteq L_{k-1}\}$
        for transactions $t \in T$
            $C_t \gets \{c \mid c \in C_k \wedge c \subseteq t\}$
            for candidates $c \in C_t$
                $count[c] \gets count[c] + 1$
        $L_k \gets \{c \mid c \in C_k \wedge \; count[c] \geq \epsilon\}$
        $k \gets k + 1$
    return $\bigcup_k L_k$
```

(https://en.wikipedia.org/wiki/Apriori_algorithm)

step5 uses the pattern sets from step4. For max pattern, the algorithm traversed the pattern sets and check if there is any other pattern is the superset of the pattern iterated. If the pattern iterated is the subset of any other pattern, the algorithm will prune the pattern and continue iterating. For closed pattern, the algorithm is similar to max pattern checking. The algorithm will traverse the pattern and check if there is any pattern is the superset of the pattern iterated and has the same frequency. The algorithm will only prune the pattern that is the subset of other pattern and has same frequency.

Step6 uses the pattern sets from step4. The algorithm calculated the purity for each item with equation from website.



$purity(p,t)=\log [f(t,p) / | D(t) |] - \log (\max [(f(t,p) + f(t',p)) / | D(t,t') |])$

The algorithm calculated the support of f(t,p) and f(t’,p) with pattern sets from step4 and store the pattern sets as set data structure to calculate the |D(t,t’)| by union the two sets. The final result will be stored in tuple with three elements, T(purity, support, pattern). The algorithm will sort the tuple according to the purity in descending order. If multiple patterns have same purity, the algorithm will sort the pattern according to their support in descending order.

Q&A

A: I choose the min support as 0.005 for this task, because each topic has around 10k documents and the estimated min support will be around 50. The min support 50 is a reasonable number because there are only around 100 1-item patterns in this threshold. The length of the patterns set will be around 150 and the length of the pattern will be in range 1 to 4.

B: The ***topic-0*** is in ***Database (DB)*** domain because the most frequent patterns are ‘query’, ‘database’, ‘object’, and so on. The ***topic-1*** is in ***Information Retrieval (IR)*** domain because the most frequent patterns are ‘information’, ‘web’, ‘retrieval’, ‘system’, and so on. The ***topic-2*** is in the ***Theory (TH)*** domain because there are a lot of ‘learning’, ‘algorithm’, ‘network’, and so on. The ***topic-3*** is in the ***Machine Learning (ML)*** domain because the most frequent patterns are ‘clustering’, ‘classification’, ‘detection’, and so on. The ***topic-4*** is in the ***Data Mining (DM)*** domain because there are many patterns, such as ‘data’, ‘mining’, ‘pattern’, and so on.

C: The number of original frequent patterns is the largest and the number of the max patterns should be the least because max pattern has a loose pruning condition. The number of closed patterns should locate between the number of frequent patterns and the number of max patterns. In my output files, the numbers of frequent patterns are around 160 and the numbers of max patterns are around 130. However, the numbers of the closed patterns are almost the same to the the numbers of the frequent patterns because there is little pattern that has superset with same frequency.

Source File:

The source file is mp3.py. Running the source file with command python mp3.py. The source file will calculate all patterns step by step.