Assignment 2: Association Analysis

Due: October 20 2023 11:59PM

Two datasets (Market, Gene) are provided. For each dataset, we provide the transaction-item representation discussed in class—Each row denotes a transaction, and each transaction consists of a set of items.

In this assignment, you are asked to implement Apriori algorithm that discovers a collection of frequent itemsets from a transaction database.

Template is for Python 3. You are asked to fill in two functions: *apriori_gen* and *get_freq*. *apriori_gen* generates new candidate itemsets based on the frequent itemsets found in the previous iteration, and prunes the itemsets containing any infrequent itemset of size *k*-1. *get_freq* returns the candidate itemsets that meet a minimum support threshold.
Do not change the input and output of the two functions in the template.

Please take the following steps:

1. Implement Apriori algorithm. The pseudo code can be found below.

```
Apriori (T, minSupport){
    // T is the database and minSupport is the minimum support
  $C_1$ = {the set of unique single items in T}
  $L_1$ = {the set of single item whose support is not less than minSupport}
  for (k = 2; $L_{k-1} \neq \emptyset$; k++ ){
      //generate and prune candidate set $C_k$
      $C_k$ is a list of itemsets in which each itemset is formed by merging two itemsets in $L_{k-1}$ if their first k-2 items are identical
      Remove an itemset from $C_k$ if any (k-1)-subset of this candidate itemset is not in the frequent itemset list $L_{k-1}$
      //Count the support of each candidate itemset
      for each transaction t in database do{
          for each candidate $c$ in $C_k$
              // increment the support count of all candidate itemsets that are contained in transaction t
              if c is a subset of t then $\text{count}[c] \leftarrow \text{count}[c] + 1$
      }
      for each candidate $c$ in $C_k$
              // Judge if a candidate itemset is frequent or not
              if the support of c is not less than minSupport
                  then include c in $L_k$
  }
  return $\{L_1, L_2, \ldots, L_k\}$
}
```

If you are not sure about whether it is OK to use a certain function, please post your question on Piazza.

2. Apply your implemented Apriori on the Market dataset with minimum support threshold=50%. You should get the following candidate itemsets (*C_k*) and frequent itemsets (*L_k*):

Candidate itemsets:
C_2: {Eggs,Key-chain}, {Eggs,Mango}, {Eggs,Onion}, {Eggs,Yo-yo}, {Key-chain, Mango}, {Key-chain, Onion}, {Key-chain,Yo-yo}, {Mango,Onion}, {Mango,Yo-yo}, {Onion,Yo-yo}
C_3: {Eggs,Key-chain,Onion}, {Key-chain,Mango,Onion}

Frequent itemsets:
L_1: {Eggs}, {Key-chain}, {Mango}, {Onion}, {Yo-yo}
L_2: {Eggs,Key-chain}, {Eggs,Onion}, {Key-chain,Mango}, {Key-chain,Onion}, {Key-chain,Yo-yo}, {Mango, Onion}
L_3: {Eggs, Key-chain,Onion}

3. If you get the same collection of itemsets at Step 2, you can proceed to apply your implemented Apriori algorithm on the Gene dataset with minimum support threshold=50%. You should be able to get 51 length-1 frequent itemsets (*L_1*), 1275 length-2 candidate itemsets (*C_2*), 29 length-2 frequent itemsets (*L_2*), 20 length-3 candidate itemsets (*C_3*) and 2 length-3 frequent itemsets (*L_3*).

4. Prepare your submission. Your final submission should be a zip file named as Assignment2.zip. In the zip file, you should include:

- The Python code.
- Report: A word or pdf file named as Assignment2 (.doc, .docx or .pdf). The report should consist of the following parts: 1) The frequent itemsets you obtain on Gene dataset (L1, L2, L3). 2) The length-3 candidate itemsets generated during Apriori (C3) on Gene dataset. 3) The codes of the two functions: *apriori_gen* and *get_freq*.

5. Submit the zip file under Assignment 2 on Brightspace.

Please refer to Course Syllabus for late submission policy and academic integrity policy. This assignment must be done independently. Running your submitted code should be able to reproduce the results in the report.