Assignment 1, DWDM ‘17

Deadline: 11:55 PM, 31/08/2017

PART 1 : Association Rule Mining using APRIORI

INSTRUCTIONS:

1. You may do the assignment in Python, C/C++ or Java.
2. It is an individual assignment, so no teams.
3. Plagiarism, in any form, will be severely punished.
4. Evaluation will be fully automated. Strictly adhere to the submission format. No queries regarding the same will be entertained later.
5. DO NOT USE ANY EXTERNAL LIBRARY FOR THE APRIORI ALGORITHM.

PROBLEM STATEMENT:

You have to code the Apriori algorithm to mine frequent itemsets and association rules from a given dataset.

Apriori algorithm:

http://nikhilvithlani.blogspot.in/2012/03/apriorialgorithmfordataminingmade.html

SPECIFICATIONS:

Specifications will be provided in a “ config.csv ” file, located in the same folder as the source code. The config file will contain the absolute path of the input data file, the absolute path of the file where you have to write the algorithm outputs, support and confidence values, and a “flag” variable indicating whether you have to find the frequent itemsets and/or association rules corresponding to the given support and confidence values.

Refer to the sample files provided on the link below before reading further .
https://drive.google.com/open?id=0B85ZKiq7OmiWRU14bDVfUEt0eEU

config.csv:
Text before the commas will remain the same in the actual config files, however, order of the lines may change. The values of the support and confidence parameters will lie in the range [0,1].

The “flag” parameter in the config file can take 2 values: 0/1, as follows:
if flag==0:
You have to mine only the frequent itemsets for the given support.
if flag==1:
You have to mine both the frequent itemsets as well as the association rules for the given support and confidence values.

input.csv:
Input data file will be a comma separated (.csv) file, containing one transaction per line. The location of the input file will be against the key “input” in the config file.

output.csv:
You have to run the apriori algorithm on the given input data for the support and confidence values provided in the config file, and write the output to the file provided in the config.csv file against the key “output”. The output file will always be a comma separated (.csv) file.

There should be only one frequent itemset per line, where the associated frequent items are comma separated. In case of the association rules, items on the left hand side should be comma separated, followed by “=>”, and continued with the items on the right hand side of the rule. There should be only one association rule per line.
Refer to the sample output file provided above for further clarifications.
The first line of the output file should contain an integer N, specifying the total number of distinct frequent itemsets mined by the algorithm. Next N lines should contain one frequent itemset per line in the format specified above.
If the flag is 1, the N+2nd line will contain an integer M, specifying the total number of distinct association rules mined by the algorithm. Next M lines should contain one association rule per line in the format specified above.
Note that two frequent itemsets are distinct if no permutation of the first itemset is exactly the same as the other itemset.
Similarly, two association rules are distinct if no combination of the permutations of the left hand side and the right hand side items of the first rule is the same as that of the other rule.

**** NOTE ****
The order of the items in a frequent itemset and the left hand/right hand of an association rule won’t matter. Similarly, you can output the various frequent itemsets and the association rules in any order. However, all the frequent itemsets should come before the association rules, as specified in the output format.

** NOTES **

● We will copy the config.csv file to the rollNumber_Assignment_1/src folder and your code will be expected to read it from there itself.
● You can have other folders/files in the rollNumber_Assigment_1 folder.

GRADING:

● This assignment carries a weightage of 10% in the final grading of the course.
● Any form of plagiarism will result in a straight 0.
● The assignment will be evaluated on a scale of 10, as follows:
○ 5 marks for a working solution.
○ 4 marks for the time complexity and scalability of the code.
○ 1 mark for the viva, which would be conducted at the time of the evaluation. You will be asked some basic questions related to the apriori algorithm and your code.

PART 2 : Decision Trees

S.No	Age	Income	Student	Credit Rating	Buys_Computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle aged	medium	no	excellent	yes
13	middle aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Construct a decision tree based on the above table using the highest information gain principle. Please show all the steps taken to construct the decision tree.

UPLOAD FORMAT:

Upload a zip file with the following directory structure:

- rollNumber_Assignment_1
  - Q1
    - src (folder containing the source code of part 1).
    - Readme.txt (a file which explains how to run the code).
  - Q2
    - Image(s) containing the hand-written solution to part 2.