MSC-BDT5002, Spring 2020

Knowledge Discovery and Data Mining

Assignment 1

Deadline: March 15th, 2020 11:59pm

Submission Guidelines

1. Assignments should be submitted to mscbd5002spring20@gmail.com as attachments.
2. **Attachments should be named in the format of**: Ax_itsc_stuid.zip.

   E.g. for a student with ITSC account: zxuav, student id: 20201234, the 1st assignment should be named as: A1_zxuav_20201234.zip.
3. You need to zip the following three files together:
   1) A1_itsc_stuid_answer**.pdf**: please put all your answers in this document including the readme pages and output answers for Q1 & Q2.
   2) A1_itsc_stuid_Q1_code: this is a **folder** that should contain all your source code for Q1.
   3) A1_itsc_stuid_Q2_code: same as above
4. For programming language, in principle, **python** is preferred.
5. TA will check your source code carefully, so your code MUST be **runnable**, your result MUST be **reproducible**.
6. Keep your code clean and comment it clearly. Missing the **necessary** comments will be deducted a certain score.
7. Your grade will be based on the correctness, efficiency and clarity.
8. Please check carefully before submitting to avoid multiple submissions.
9. Submissions after the deadline or not following the rules above are **NOT** accepted.
10. **Plagiarism will lead to zero points.**

(Please read the guidelines carefully)

Q1. Hash Tree (40 marks)

Suppose we have 36 candidate item sets of length 3:

{1 2 3}, {1 3 9}, {1 4 5}, {1 4 6}, {1 5 7}, {1 5 9}, {1 6 8}, {1 6 9}, {1 8 9}

{2 3 9} {2 5 6}, {2 5 7}, {2 5 9}, {2 6 7}, {2 6 8}, {2 6 9}, {2 7 8}, {2 7 9}, {2 8 9}

{3 4 6}, {3 4 8}, {3 7 8}

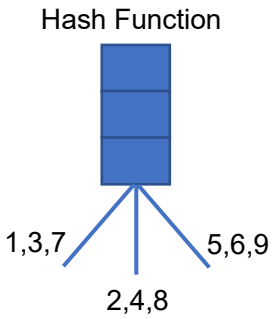
{4 5 6}, {4 5 8}, {4 5 9}, {4 7 8}, {4 7 9}, {4 8 9}

{5 6 7}, {5 6 8}, {5 7 8}, {5 7 9}, {5 8 9}

{6 7 9}, {6 8 9}

{7 8 9}

The hash function is shown in the figure below.



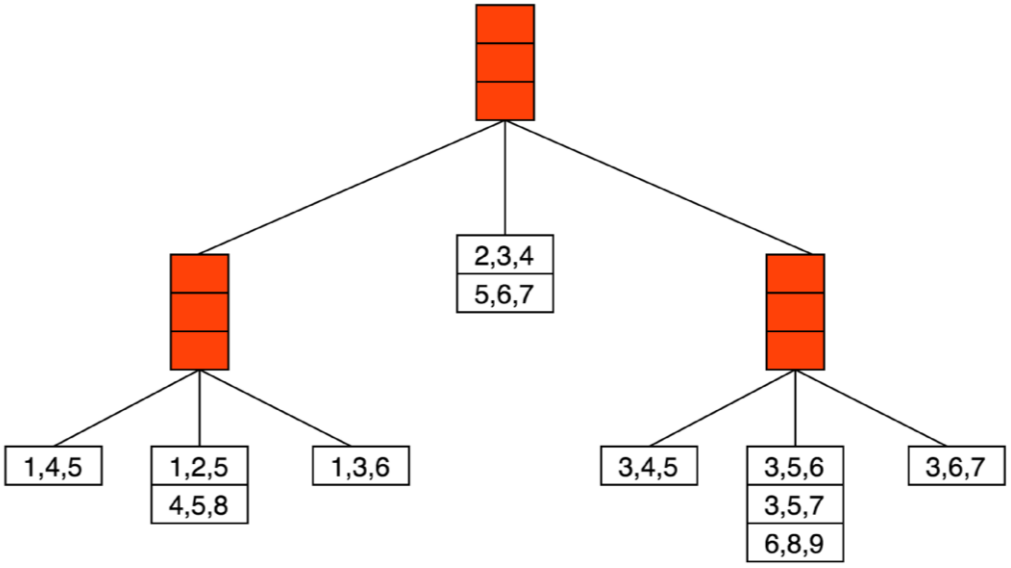
(a) Please write a program to generate a hash tree with **max leaf size 3**, output the nested list (or nested dict) of the hash tree **hierarchically** and draw the structure of the hash tree (you can write program to draw this hash tree or just manually draw it according to the nested list you output). **Please write the nested list/dict and the hash tree together in the A1_itsc_stuid_answer.pdf.** (35 marks)

Give an example:

If the nested list is (underline is just to make the structure clearer; you don't need to draw it in your assignment):

[[[1,4,5], [[1,2,5],[4,5,8]], [1,3,6]], [[2,3,4], [5,6,7]], [[3,4,5], [[3,5,6], [3,5,7], [6,8,9]], [3,6,7]]]

Then the corresponding hash tree is:



(b) Given a transaction that contains items {1, 3, 5, 6, 7, 9}, how many comparisons are needed using the hash tree which you generate above? Please circle these candidates in the hash tree. **No programming required.** (5 marks)

- **Notes:**

1. You **MUST** code by yourself to complete the algorithm.

2. The hash tree must be constructed by your algorithm. In other words, if the dataset changes, your algorithm should also output the correct answer.

Q2. FP-Tree (60 marks)

Suppose you are given some transactions and a vocabulary that map terms to indexes. Please use FP-Tree algorithm to discover the frequent itemsets.

- **Data Description:**

topi-i.txt:

Input file of frequent pattern mining algorithms. Each line represents a transaction with indices of terms.

format: term1_index term2_index term3_index ...

Columns are separated by a space.

vocab.txt

Dictionary that maps term index to term.

format: term_index term

Columns are separated by a space.

pattern-i.txt:

The file you need to submit, which contains your result for this frequent pattern mining task. Each line represents a transaction with frequent itemsets sorted in descending order of support count.

format: support_count term1 term2 ...

support_count and term1 are separated by a tab, while terms are separated by a space.

Here we give an example:

233 rule association
230 random
227 finding
203 mining pattern

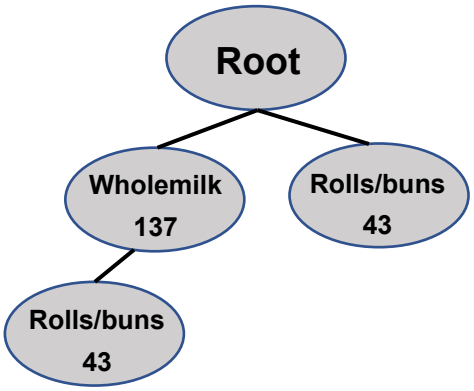
- **Questions:**

(a) Please write a program to implement FP-growth algorithm and find all frequent itemsets with **support >= 400** in the given dataset. (42 marks)

(b) Based on the result of (a), please print out those FP-conditional trees whose height is larger than 1. (18 marks)

Give an example of problem (b)'s output:

For the tree as follows:



We expect you print the result like:

["Null Set 1", [["whole milk 137", "rolls/buns 43"], "rolls/buns 43"]]

- **Notes:**

You **MUST** code by yourself to complete the algorithm.