MSC-BDT5002, Fall 2018
Knowledge Discovery and Data Mining

Assignment 1

Deadline: Sep. 28th, 11:59pm, 2018

Submission Guidelines

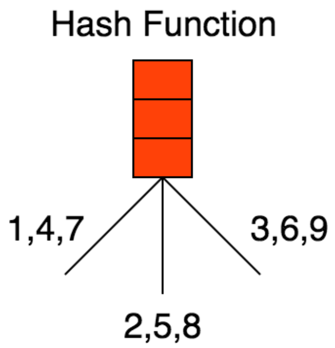
- Assignments should be submitted to mscbdt5002fall18@gmail.com as attachments.
- **Attachments should be named in the format of**: Ax_itsc_stuid.zip.
  e.g. for a student with **ITSC account**: zxuav, student id: 20181234, the 1st assignment should be named as: A1_zxuav_20181234.zip.
- You need to zip 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
- For programming language, in principle, **python** is preferred.
- TA will check your source code carefully, so your code MUST be **runnable**, your result MUST be **reproducible**.
- Keep your code clean and comment it clearly. Missing the **necessary comments** will be deducted a certain score.
- Your grade will be based on the correctness, efficiency and clarity.
- Please check carefully before submitting to avoid multiple submissions.
- Submissions after the deadline or not following the rules above are **NOT** accepted.
- **Plagiarism will lead to zero points.**

(Please read the guidelines carefully)

Q1. Hash Tree (40 marks)·

Suppose we have 35 candidate item sets of length 3:
{1 2 4}, {1 2 9}, {1 3 5}, {1 3 9}, {1 4 7}, {1 5 8}, {1 6 7}, {1 7 9}, {1 8 9}
{2 3 5}, {2 4 7}, {2 5 6}, {2 5 7}, {2 5 8}, {2 6 7}, {2 6 8}, {2 6 9}, {2 7 8}
{3 4 5}, {3 4 7}, {3 5 7}, {3 5 8}, {3 6 8}, {3 7 9}, {3 8 9},
{4 5 7}, {4 5 8}, {4 6 7}, {4 6 9}, {4 7 8},
{5 6 7}, {5 7 9}, {5 8 9}, {6 7 8}, {6 7 9}

The hash function is shown in the figure below.



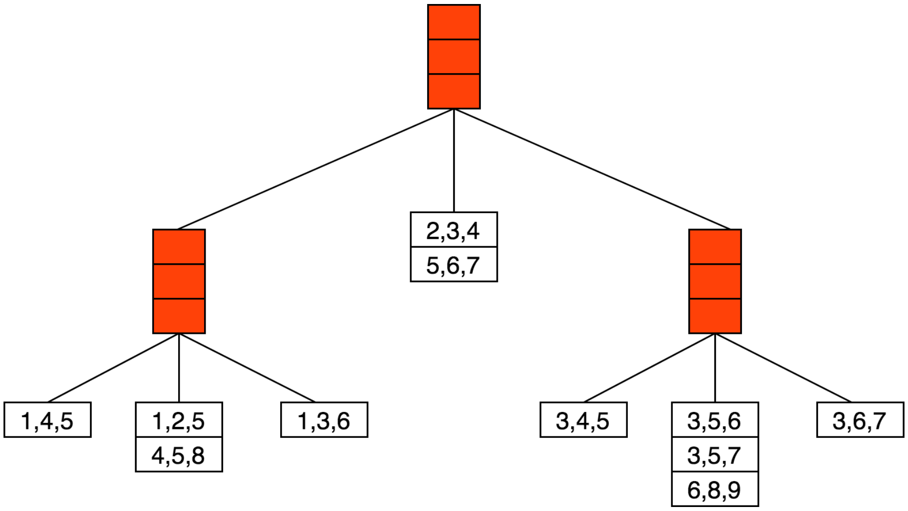
(a) Please write a program to generate a hash tree with **max leaf size** [redacted] 3, output the nested list (or nested dict) of the hash tree **hierarchically** and draw the structure of the hash tree (you can write program to draw this hash tree or just manually draw it according to the nested list 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:

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]]]

The corresponding hash tree is:



(b) Given a transaction that contains items {1, 2, 4, 6, 8, 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)

Frequent Pattern Mining is very important for the retail industry to increase profits. Suppose you are the owner of a grocery, there is a sale records of your store.

Data Description:

groceries.csv: This is a .csv file that contains totally 9835 records and each record records every single transaction in the grocery store. The following table is an example of it.

1	beef
2	butter, sugar, fruit/vegetable juice, newspapers
3	frankfurter, rolls/buns, soda
4	packaged fruit/vegetables
...	...

sample_submission.csv: This is a sample submission format for Question (a), you should follow this template to save your result.

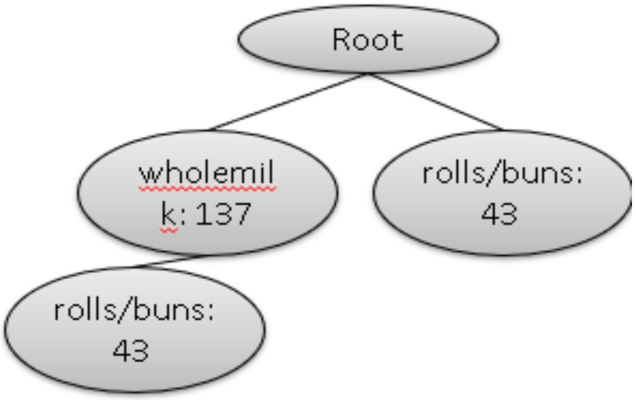
Questions:

(a) Please write a program to implement FP-growth algorithm and find all frequent itemsets with **support >= 300** 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:

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