CS6350 Spring 2022

Homework #3

Submission Deadline: 11:59 pm, March 27th

In this homework, you will use spark (pyspark or scala) to solve the following problems.

Q1:

Write a spark script to find total number of common friends for any possible friend pairs. The key idea is that if two people are friend then they have a lot of mutual/common friends.

For example,

Alice's friends are Bob, Sam, Sara, Nancy Bob's friends are Alice, Sam, Clara, Nancy Sara's friends are Alice, Sam, Clara, Nancy

As Alice and Bob are friend and so, their mutual friend list is [Sam, Nancy]
As Sara and Bob are not friend and so, their mutual friend list is empty. (**In this case you may exclude them from your output**).

Input
files:
1. mutual.txt

The input contains the adjacency list and has multiple lines in the following format:
<User><TAB><Friends>

Here, <User> is a unique integer ID corresponding to a unique user and <Friends> is a comma-separated list of unique IDs (<User> ID) corresponding to the friends of the user. Note that the friendships are mutual (i.e., edges are undirected): if A is friend with B then B is also friend with A. The data provided is consistent with that rule as there is an explicit entry for each side of each edge. So when you make the pair, always consider (A, B) or (B, A) for user A and B but not both.

Output: The output should contain one line per user in the following format:
<User_A>, <User_B><TAB><Mutual/Common Friend Number>

where <User_A> & <User_B> are unique IDs corresponding to a user A and B (A and B are friend). < Mutual/Common Friend Number > is total number of common friends between user A and user B.

Q2.
Please answer this question by using dataset from Q1.
Find friend pair(s) whose number of common friends is less than the average number among all the pairs.

Output Format:
<User_A>, <User_B><TAB><Mutual/Common Friend Number>

Please use the following dataset.
1. mutual.txt

Please use Apache Spark to derive some statistics from **Yelp Dataset.**
Data set info:

The dataset files are as follows and columns are separate using '**::**'
business.csv. **review.csv.** **user.csv.**

Data set Description.
The data set comprises of **three** csv files, namely user.csv, business.csv and review.csv.

business.csv file contain basic information about local businesses.
business.csv file contains the following columns:
"business_id"::"full_address"::"categories"
'business_id': (a unique identifier for the business)
'full_address': (localized address),
'categories': [(localized category names)]

review.csv file contains the star rating given by a user to a business. Use user_id to associate this review with others by the same user. Use business_id to associate this review with others of the same business.
review.csv file contains the following columns:
"review_id"::"user_id"::"business_id"::"stars"
'review_id': (a unique identifier for the review)
'user_id': (the identifier of the reviewed business),
'business_id': (the identifier of the authoring user),
'stars': (star rating, integer 1-5), the rating given by the user to a business

user.csv file contains aggregate information about a single user across all of Yelp
user.csv file contains the following columns:
"user_id"::"name"::"url"
'user_id': (unique user identifier),
'name': (first name, last initial, like 'Matt J.'), this column has been made anonymous to preserve privacy
'url': url of the user on yelp

Note: **::** is Column separator in the files.

Q3:

Please list the 'name' and 'rating' of users that reviewed businesses which are in "Stanford"

This will require you to use **all three files.**

Sample output:

Username	**Rating**
John Snow	4.0

Q4:

List the business_ID, full address, and categories of the Top 10 businesses having the greatest number of ratings.

This will require you to use **review.csv** and **business.csv files.**

Sample output:

business id	**full address**	**categories**	**no. of ratings**
xdf12344444444	CA 91711	List ['Local Services', 'Carpet Cleaning']	50

Q5:

Find names of top 10 users who have the least contribution in the reviews. The contribution is measured by the percentage of the total reviews that a particular user gave.

For example:

```
+----------------+----------------+
|name            |contribution|
+----------------+----------------+
| John Snow   |22%           |
```

Q6.

Given two input matrices, matrix1.txt and matrix2.txt, use spark's MLlib to carry out distributed matrix multiplication. If your output is in a specific format describe it in a separate Readme file. For reference:

- https://medium.com/balabit-unsupervised/scalable-sparse-matrix-multiplication-in-apache-spark-c79e9ffc0703
- https://towardsdatascience.com/preserve-row-indices-with-spark-matrix-multiplication-8007e21ea28f

What to submit

(i) Submit the source code via the eLearning website.
(ii) Submit the output file for each question.
(iii) Any Readme file if required.