CS550: Massive Data Mining and Learning
Homework 1

Due 11:59pm Sunday, February 25, 2018
Only one late period is allowed for this homework (11:59pm Monday 2/26)

Submission Instructions

Assignment Submission Include a signed agreement to the Honor Code with this assignment. Assignments are due at 11:59pm. All students must submit their homework via Sakai. Students can typeset or scan their homework. Students also need to include their code in the final submission zip file. Put all the code for a single question into a single file.

Late Day Policy Each student will have a total of *two* free late days, and for each homework only one late day can be used. If a late day is used, the due date is 11:59pm on the next day.

Honor Code Students may discuss and work on homework problems in groups. This is encouraged. However, each student must write down their solutions independently to show they understand the solution well enough in order to reconstruct it by themselves. Students should clearly mention the names of all the other students who were part of their discussion group. Using code or solutions obtained from the web is considered an honor code violation. We check all the submissions for plagiarism. We take the honor code seriously and expect students to do the same.

Discussion Group (People with whom you discussed ideas used in your answers):
Georgios Chantzialexiou

On-line or hardcopy documents used as part of your answers:

I acknowledge and accept the Honor Code.

(Signed) CM

If you are not printing this document out, please type your initials above.

Answer to Question 1

The source code is attached named FriendRecommendation.java.
The algorithm consists of two phases, the Map Phase and the Reduce Phase. I will go on to describe what happens in each phase.

Map Phase

In this phase we write < key, r, m > . The key value will be the user id for the user that will get the recommendation, the second value (r) will be the user id of the user that will be recommended to the key user and the third value will be the id of the mutual friend. Since we do not want to recommend users that are already friends with the key user, we will set m=-1 for all users that are already friends with key user. More specifically we emit the following:

```
for (user : Users)
        for (friend : user.friendsList)
                write <user, friend, -1 >

for (user : Users)
        for(friend1 : user.friendsList)
                for(friend2 : user.friendList)
                        write <friend1, friend2, user>
```

Note: These are all the possible combinations of friends in the user friends list.

Reduce Phase

The basic idea is that we just sum up how many mutual friends they have been between the key and r users. If any of them has mutual friend -1 (m=-1) they are already friends and we don't make that recommendation.
More specifically:

1. For each key we create a HashMap. This HashMap has the recommended users as keys and their mutual friends in a list. If we encounter a mutual friend with value -1 then they are already friends and we set the list of mutual friends to null.
2. We then sort the HashMap by transforming it into a TreeMap. We need to implement a Comparator for the TreeMap. This Comparator sorts the items of the HashMap in descending order, based on the size of their mutual Friends List.
3. Finally, we write the 10 friends with the most mutual friends (or less if there are not that many.

Recommendations for Specified Users

924 → 439,2409,6995,11860,15416,43748,45881
8941 → 8943,8944,8940
8942 → 8939,8940,8943,8944
9019 → 9022,317,9023
9020 → 9021,9016,9017,9022,317,9023
9021 → 9020,9016,9017,9022,317,9023
9022 → 9019,9020,9021,317,9016,9017,9023
9090 → 16380,961,1347,1357,1371,1379,1380,1385,1390,1392
9092 → 9095,546,1357,2196,2694,2773,2812,3937,5231,5957
9093 → 142,5040,6157,14284,21298,42704,48442,125,338,2196

Answer to Question 2(a)

Answer to Question 2(b)

Answer to Question 2(c)

Answer to Question 2(d)

Answer to Question 2(e)

Answer to Question 3(a)

Answer to Question 3(b)

Answer to Question 3(c)