The solution for CS550 spring 2019 assignment 1

Name: Wenhao Luo, RUID:191004111, Email: w.luo.bosonfields@rutgers.com

---

Question_1:

Issue(i): The source code as a single source code file named as the question number (e.g.,question_1.java).

Sol:

- The relevant code file is question_1.scala.

Issue(ii): Include in your writeup a short paragraph describing your algorithm to tackle this problem.

Sol:

- Step 1: Firstly, I map the user with the user's each friends, which is called ***current_friends_pairs***, generating a map with pairs (user, friend). In this step the blanks or users who don't have friends are eliminated temporarily;
- Step 2: Join the ***current_friends_pairs*** with itself, deleting the tuples with same elements, which is called ***del_repeat***, generating a map with tuples like: (user, (friend1, friend2))
- Step 3: Get the tuples of (friend1, friend2) and subtract the tuples in ***current_friends_pairs*** to delete the current friends, then map the tuple (friend1, friend2) => ((friend1, friend2), 1). The ((friend1, friend2), 1) can also be writen as ((user1, user2), 1). I later reduced them by the value, which is called ***pair_value***. The tuple just like ((user1, user2),

value), in which the value is the number of mutual friends of friend1 and friend2

- Step 4: Group the ***pair_value*** by Keys and extract the top 10 friends of user1 by the value in descending order. That is pick the (user2, value) with the top 10 values in the bucket of user1.

Issue(iii): Include in your writeup the recommendations for the users with following user IDs: 924, 8941, 8942, 9019, 9020, 9021, 9022, 9990, 9992, 9993.

Sol:

User	top10 friends recommended
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
9990	13134,13478,13877,34299,34485,34642,37941
9992	9987,9989,35667,9991
9993	9991,13134,13478,13877,34299,34485,34642,37941

Question_2:

Include your properly named code file (e.g., question_2.java or question_2.py), and include the answers to the following questions in your writeup:

Sol:

- The relevant code file is question_2.scala

Issue(i): Explanation for 2(a): A drawback of using confidence is that it ignores Pr(B). Why is this a drawback? Explain why lift and conviction do not suffer from this drawback?

Sol:

- The drawback of ignoring Pr(B) is that when B is quite popular, even though B and A are independent, the ***conf*** result can imply they are highly dependent. E.g. if B is prevalent, but it has no corralation with A that for all their intersections: $Pr(A \bigcap B) = Pr(A) * Pr(B)$ which means $conf = (Pr(B|A) = Pr(B)$. Since Pr(B) is large, we may get the conclusion that conf$A \longrightarrow B$ is big and $A \longrightarrow B$ is a valid rule.
- For the other two evalutions: ***lift*** and ***conv***, we can see the two formulas decline the impact of Pr(B) or S(B) in different ways. In ***lift***, the formula divide the S(B) and in ***conv***, the formula subtract the S(B). So they do not have such drawback.

Issue(ii): Proofs and/or counterexamples for 2(b): A measure is symmetrical if measure(A → B) = measure(B → A). Which of the measures presented here are symmetrical? For each measure, please provide either a proof that the measure is symmetrical, or a counterexample that shows the measure is not symmetrical.

Sol:

- The evaluate measure ***lift*** is symmetrical since:

$$\begin{aligned} lift(A \longrightarrow B) &= Pr(B|A)/Pr(B) = Pr(A \bigcap B)/(Pr(A) * Pr(B)) \\ &= Pr(A|B)/Pr(A) = lift(B \longrightarrow A) \end{aligned}$$

- The evaluation measures ***conf*** and ***conv*** is directional since $Pr(A|B$ can be different with $Pr(B|A)$, so the $conf(A \longrightarrow B)$ can be vary with $conf(B \longrightarrow A)$. It is similar for the ***conv*** that the difference in the two

conf cannot be eliminated upon the formula. So they are not symmetrical.

Issue(iii): Explanation for 2(c): A measure is desirable if its value is maximal for rules that hold 100% of the time (such rules are called perfect implications). This makes it easy to identify the best rules. Which of the above measures have this property? Explain why.

Sol:

- The ***conf*** and ***conv*** evaluation is desirable based on the value of maximal for rules, but ***lift*** does not follow that: e.g. if A and B always appear at the same time, which means that $Pr(A|B) = 1$. In this case, the value of evaluations are listed below:

$$
\begin{aligned}
&conf(A \longrightarrow B) = 1 \\
&conv(A \longrightarrow B) = +\infty \\
&lift(A \longrightarrow B) = 1/Pr(A)
\end{aligned}
$$

- We can see that the value of ***lift*** can vary as the $Pr(A)$ changes in the situation, but the value of ***conf*** and ***conv*** can remain the maximum value at this maximal situation.

Issue(iv): Top 5 rules with confidence scores for 2(d).

Sol:

Top 5 rules ordered by their confidence value:

$$
\begin{aligned}
DAI93865 &=> FRO40251 \quad Conf: 1.0 \\
GRO85051 &=> FRO40251 \quad Conf: 0.99917626 \\
GRO38636 &=> FRO40251 \quad Conf: 0.99065423 \\
ELE12951 &=> FRO40251 \quad Conf: 0.990566 \\
DAI88079 &=> FRO40251 \quad Conf: 0.9867257
\end{aligned}
$$

Issue(v): Top 5 rules with confidence scores for 2(e).

Sol:

$$
\begin{aligned}
(GRO85051, DAI31081) &=> FRO40251 \quad conf: 1.0 \\
(ELE92920, DAI23334) &=> DAI62779 \quad conf: 1.0 \\
(GRO85051, SNA80324) &=> FRO40251 \quad conf: 1.0 \\
(DAI88079, DAI62779) &=> FRO40251 \quad conf: 1.0 \\
(GRO85051, DAI75645) &=> FRO40251 \quad conf: 1.0
\end{aligned}
$$

Question_3:

Issue(i): Suppose a column has m 1's and therefore (n-m) 0's. Prove that the probability we get "don't know" as the min-hash value for this column is at most $(\frac{n-k}{n})^m$***.***

Sol:

- For a list with m 1's and (n-m) 0's, the number for all the possible combinations is:

$$N_{random} = \binom{n}{m}$$

- For a list with m 1's and (n-m) 0's, when ***k*** rows are specificly defined and 1's cannot put in these k rows. The number for all the possible combinations based on the above condition is:

$$N_{1 \ avoid \ k \ rows} = \binom{n-k}{m}$$

- The probability we get "don't know" is $N_{1 \ avoid \ k \ rows} / N_{random}$. Expanding the factorial elements we will have:

$$
\begin{aligned}
N_{1\ avoid\ k\ rows}/N_{random} &= \frac{(n-k)(n-k-1)\cdots(n-k-m+1)*m!}{n(n-1)\cdots(n-m+1)*m!} \\
&= \frac{(n-k)(n-k-1)\cdots(n-k-m+1)}{n(n-1)\cdots(n-m+1)} \\
&= \frac{\prod_{i=0}^{m-1}(n-k-i)}{\prod_{i=0}^{m-1}(n-i)}
\end{aligned}
$$

- Set $f(x)=\frac{x-k}{x}, x>k>0$, then we will easily get $f^{'}(x)=\frac{2k}{x}$, which means that $f(x)$ is monotonically increasing function.
- So for all the value of i from 0 to m-1, we will have $\frac{n-k-i}{n-i} \leq \frac{n-k}{n}$ all the time. Upon this, we will get:

$$
N_{1\ avoid\ k\ rows}/N_{random} = \frac{\prod_{i=0}^{m-1}(n-k-i)}{\prod_{i=0}^{m-1}(n-i)} \leq (\frac{n-k}{n})^m
$$

Issue(ii): Suppose we want the probability of "don't know" to be at most e^{-10}. Assuming n and m are both very large (but n is much larger than m or k), give a simple approximation to the smallest value of k that will assure this probability is at most e^{-10}. Hints: (1) You can use $(\frac{n-k}{n})^m$ as the exact value of the probability of "don't know." (2) Remember that for large x, $(1-\frac{1}{x})^x \approx 1/e$.

Sol:

- To meet the requirment, we will have:

$$
\begin{aligned}
&(\frac{n-k}{n})^m \leq e^{-10} \\
&(1-\frac{1}{n/k})^{m(n/k)(k/n)} \leq e^{-10} \\
&((1-\frac{1}{n/k})^{n/k})^{mk/n} \approx (e^{-1})^{mk/n} \leq e^{-10} \\
&mk/n \geq 10 \\
&k \geq 10n/m
\end{aligned}
$$

Issue(ii): Give an example of two columns such that the probability (over cyclic permutations only) that their min-hash values agree is not the same as their Jaccard similarity. In your answer, please provide (a) an example of a matrix with two columns (let the two columns correspond to sets denoted by S1 and S2) (b) the Jaccard similarity of S1 and S2, and (c) the probability that a random cyclic permutation yields the same min-hash value for both S1 and S2.

Sol:

- If our two sets have several consecutive zeros, the cyclic permutations maybe not be sufficient.
- Assuming two sets with length of 5: $S1 = [0, 0, 0, 1, 0]^T$ and $S2 = [0, 0, 0, 1, 1]^T$, the Jaccard similarity of S1 and S2 is $Jac(S1, S2) = 1/2 = 0.5$.
- However, by using cyclic min-hashing, we will find that the value of min-hashing will remain the same from row 1 to row 4, so the min-hashing similarity is $Sim(h(S1), h(S2)) = 4/5 = 0.8$
- Similarily, for sets $S3 = [0, 0, 0, 0, 1]^T$ and $S4 = [0, 0, 0, 1, 1]^T$, their Jaccard similarity remain the same: $Jac(S1, S2) = 1/2 = 0.5$, but their similarity of min-hashing function is (Sim(h(S1), h(S2)) = 1/5 = 0.2\), based on the cyclic perputation.