CS550: Massive Data Mining and Learning
Homework 1

Christos Mitropoulos - cm1012
email c.mitro@rutgers.edu

Github repository:
https://github.com/CMitropoulos/MassiveDataMining/tree/master/Homework1

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:

FriendRecommendation.java, FriendRecommendationResults/, Apriori.java, FrequentItemSetsResults/

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 and is 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 do 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)

Confidence ignores Pr(B) since it is defined as the probability of occurrence of B in the basket if the basket already contains A. This is a drawback, since items with high support that are contained in many baskets will naturally produce large confidence values with many items. A real life scenario would be the following:

A lot of people buy milk. Since milk is contained in most of the baskets, it will yield high confidence values with items that are not necessarily associated with milk and therefore we will come up with many association rules that are not necessarily useful. If we give the store managers a million association rules that meet our thresholds for confidence, they cannot even read them, let alone act on them.

Lift does not suffer from this drawback because it takes into account the Support(B) int the denominator. The more common item B is, the lower lift value it will yield for the same confidence value. In the previous example with the milk, since milk is really common, its $S(B) = \frac{Support(B)}{N}$ will be higher and therefore the $lift(A \to B)$ will be lower.

Conviction also depends from the Support(B). Similar to lift, the higher the Support(B) is the smaller the conviction value will be.

In conclusion, conviction and Lift do not suffer from this drawback since they take into account the Support(B) and common items like milk that appear in almost any basket will not provide us with useless association rules.

Answer to Question 2(b)

I will use the Proof by Contradiction method for Confidence and Lift to prove if the measures are symmetrical. Unless we end up in something that is always true, the initial statement is not true. I will prove that Conviction is not symmetrical with a counterexample.

Confidence

Assume for the sake of contradiction that Confidence is a symmetrical measure. Also assume that A, B are different items. Then:

$$\begin{gathered}
conf(A \rightarrow B) = conf(B \rightarrow A) \Leftrightarrow \\
P(B|A) = P(A|B) \Leftrightarrow \\
\frac{P(A \cap B)}{P(A)} = \frac{P(B \cap A)}{P(B)} \Leftrightarrow \\
\frac{1}{P(A)} = \frac{1}{P(B)} \Leftrightarrow \\
P(A) = P(B)
\end{gathered}$$

which is not always true for all items A, B. Therefore **Confidence is not symmetrical.**

Lift

Assume for the sake of contradiction that Lift is a symmetrical measure. Also assume that A, B are different items. Then:

$$\begin{gathered}
lift(A \rightarrow B) = lift(B \rightarrow A) \Leftrightarrow \\
\frac{conf(A \rightarrow B)}{S(B)} = \frac{conf(B \rightarrow A)}{S(A)} \Leftrightarrow \\
\frac{P(B|A) \times N}{Support(B)} = \frac{P(A|B) \times N}{Support(A)} \Leftrightarrow \\
\frac{P(A \cap B)}{P(A) \times Support(B)} = \frac{P(B \cap A)}{P(B) \times Support(A)} \Leftrightarrow \\
P(A) \times Support(B) = P(B) \times Support(A) \Leftrightarrow \\
\frac{P(A)}{P(B)} = \frac{Support(A)}{Support(B)}
\end{gathered}$$

which is true for all items A, B. Therefore **Lift is symmetrical.**

Conviction

I will use a counterexample for Conviction. Assuming we have the following baskets:
Basket1: A, B
Basket2: A
Basket3: B
Basket4: B
Then $conv(A \rightarrow B) = \frac{1-S(B)}{1-conf(A \rightarrow B)} = \frac{1-0.75}{1-\frac{0.25}{0.75}} = 0.37$
$conv(B \rightarrow A) = \frac{1-S(A)}{1-conf(B \rightarrow A)} = \frac{1-0.75}{1-\frac{0.25}{0.5}} = 1$
Therefore **Conviction is not symmetrical.**

Answer to Question 2(c)

Confidence

We know that $conf(A \rightarrow B) = P(B|A) = \frac{P(A \cap B)}{P(A)}$.
So $conf(A \rightarrow B)$ will be max when $P(A \cap B) = P(A)$. This is true when $A \subset B$.
However, when $A \subset B$ we have a perfect implication since whenever A is selected we are certain that B will be selected too. Therefore, **Confidence is a desirable measure.**

Lift

Lift depends on the value of $P(B)$ and we may find B in baskets which do not contain A. **Lift is not a desirable measure.**

Conviction

Conviction maximum value is infinity, when confidence gets its maximum value, which is 1. **Conviction is a desirable measure.**

Answer to Question 2(d)

$conf(DAI93865 \rightarrow FRO40251) = 1$
$conf(GRO85051 \rightarrow FRO40251) = 0.99917626$
$conf(GRO38636 \rightarrow FRO40251) = 0.99065423$
$conf(ELE12951 \rightarrow FRO40251) = 0.990566$
$conf(DAI88079 \rightarrow FRO40251) = 0.9867257$

Answer to Question 2(e)

$conf(DAI23334, ELE92920 \rightarrow DAI62779) = 1$
$conf(DAI31081, GRO85051 \rightarrow FRO40251) = 1$
$conf(DAI55911, GRO85051 \rightarrow FRO40251) = 1$
$conf(DAI62779, DAI88079 \rightarrow FRO40251) = 1$
$conf(DAI75645, GRO85051 \rightarrow FRO40251) = 1$

Answer to Question 3(a)

The number of different combinations with m 1's out of n is $\binom{n}{m}$. The number of these columns that do not have 1 in any of the k selected rows is $\binom{n-k}{m}$. So we can easily calculate the the probability of getting "don't know" by dividing $\frac{\binom{n-k}{m}}{\binom{n}{m}}$.
By simplifying this function we get $\frac{n-k}{n} \times \frac{n-k-1}{n-1} \times ... \times \frac{n-k-m+1}{n-m+1}$, where each of these factors is at most $\frac{n-k}{n}$, thus the product is at most $(\frac{n-k}{n})^m$.

Answer to Question 3(b)

We want
$(\frac{n-k}{n})^m \leq e^{-10} \Rightarrow$
$(1-\frac{k}{n})^m \leq e^{-10} \Rightarrow$
$((1-\frac{k}{n})^{\frac{n}{k}})^{\frac{mk}{n}} \leq e^{-10}$
Since $k << n$ we can approximate $(1-\frac{k}{n})^{\frac{n}{k}}$ by $\frac{1}{e}$
Therefore, we want $e^{\frac{-mk}{n}} \leq e^{-10} \Rightarrow \frac{-mk}{n} \leq -10 \Rightarrow \frac{mk}{n} \geq 10 \Rightarrow k \geq \frac{10n}{m}$
So the lower bound on k is $\frac{10n}{m}$

Answer to Question 3(c)

The two columns (sets) are $[0,0,1]^T and [1,0,1]^T$. $Jaccard similarity = \frac{(001) AND (101)}{(001) OR (101)} = 0.5$
All the possible cyclic permutations are: (1,2,3) , (3,1,2) , (2,3,1) and the corresponding minhash values are (3,1) , (1,1) , (1,1). Thus, the probability of same minhash values are $\frac{2}{3}$