uOttawa

Professional Master's in Artificial Intelligence
Fundamentals for Applied Data Science (DTI 5126)

Subject: Assignment 3 (AR &CF)

By

Mohamed Sayed Abdelwahab Hussein

(300273145)

Mhuss073@uottawa.ca

Under Supervision
Dr. Olubisi Runsewe

Part A (Association Rules):

I. Given a simple transactional database X: Using the threshold values support = 25% and confidence = 60%.

a. Find all frequent item sets in database X

	A	B	C	D	E	F	G
A	5	3	3	4	1	2	2
B	3	4	2	2	0	1	2
C	3	2	5	4	1	2	3
D	4	2	4	6	1	4	3
E	1	0	1	1	1	0	1
F	2	1	2	4	0	4	2
G	2	2	3	3	2	2	5

Item set 1

Item set 2

Item set 2

Itemset	Support
{A}	5
{B}	4
{C}	5
{D}	6
{E}	1
{F}	4
{G}	5

Item set 3

Itemset	Support
{A, B, C}	1
{A, B, D}	2
{A, B, F}	1
{A, B, G}	1
{A, C, D}	3
{A, C, F}	1
{A, C, G}	1
{A, D, F}	2
{A, D, G}	1
{A, F, G}	0
{B, C, D}	1
{B, C, G}	1
{B, D, G}	0
{C, D, F}	2
{C, D, G}	2
{C, F, G}	1
{D, F, G}	2

Itemset	Support
{A, B}	3
{A, C}	3
{A, D}	4
{A, E}	1
{A, F}	2
{A, G}	2
{B, C}	2
{B, D}	2
{B, E}	0
{B, F}	1
{B, G}	2
{C, D}	4
{C, E}	1
{C, F}	2
{C, G}	3
{D, E}	1
{D, F}	4
{D, G}	3
{E, F}	0
{E, G}	1
{F, G}	2

Itemset	Support
{A, B}	3
{A, C}	3
{A, D}	4
{A, F}	2
{A, G}	2
{B, C}	2
{B, D}	2
{B, G}	2
{C, D}	4
{C, F}	2
{C, G}	3
{D, F}	4
{D, G}	3
{F, G}	2

Rule	Support	Confidence
A B	0.375	0.6
B A	0.375	0.75
A C	0.375	0.6
C A	0.375	0.6
A D	0.5	0.8
D A	0.5	0.6667
A F	0.25	0.4
F A	0.25	0.5
A G	0.25	0.4
G A	0.25	0.4
B C	0.25	0.5
C B	0.25	0.4
B D	0.25	0.5
D B	0.25	0.333
B G	0.25	0.5
G B	0.25	0.4
C D	0.5	0.8
D C	0.5	0.667
C F	0.25	0.4
F C	0.25	0.5
C G	0.375	0.6
G C	0.375	0.6
D F	0.5	0.6667
F D	0.5	1
D G	0.375	0.5
G D	0.375	0.6
F G	0.25	0.5
G F	0.25	0.4
AB D	0.25	0.667
AD B	0.25	0.5
BD A	0.25	1
AC D	0.375	1
AD C	0.375	0.75
CD A	0.375	0.75
CD F	0.25	0.5
CF D	0.25	1
DF C	0.25	0.9
CD G	0.25	0.9
CG D	0.25	0.667
DG C	0.25	0.667
DF G	0.25	0.5
DG F	0.25	0.667
FG D	0.25	1
AD F	0.25	0.5
AF D	0.25	1
DF A	0.25	0.5

➔

Rule	Support	Confidence	Lift
A B	0.375	0.6	1.2
B A	0.375	0.75	1.2
A C	0.375	0.6	0.96
C A	0.375	0.6	0.96
A D	0.5	0.8	1.0667
D A	0.5	0.6667	1.0667
C D	0.5	0.8	1.0667
D C	0.5	0.667	1.0667
C G	0.375	0.6	0.96
G C	0.375	0.6	0.96
G D	0.375	0.6	0.8
F D	0.5	1	1.333
D F	0.5	0.667	1.333
AB D	0.25	0.667	0.8889
BD A	0.25	1	1.6
AC D	0.375	1	1.33
AD C	0.375	0.75	1.2
CD A	0.375	0.75	1.2
AF D	0.25	1	1.333
CF D	0.25	1	1.333
CG D	0.25	0.667	0.889
DG C	0.25	0.667	1.0667
DG F	0.25	0.667	1.333
FG D	0.25	1	1.333

c. Analyse misleading associations for the rule set obtained in (b).

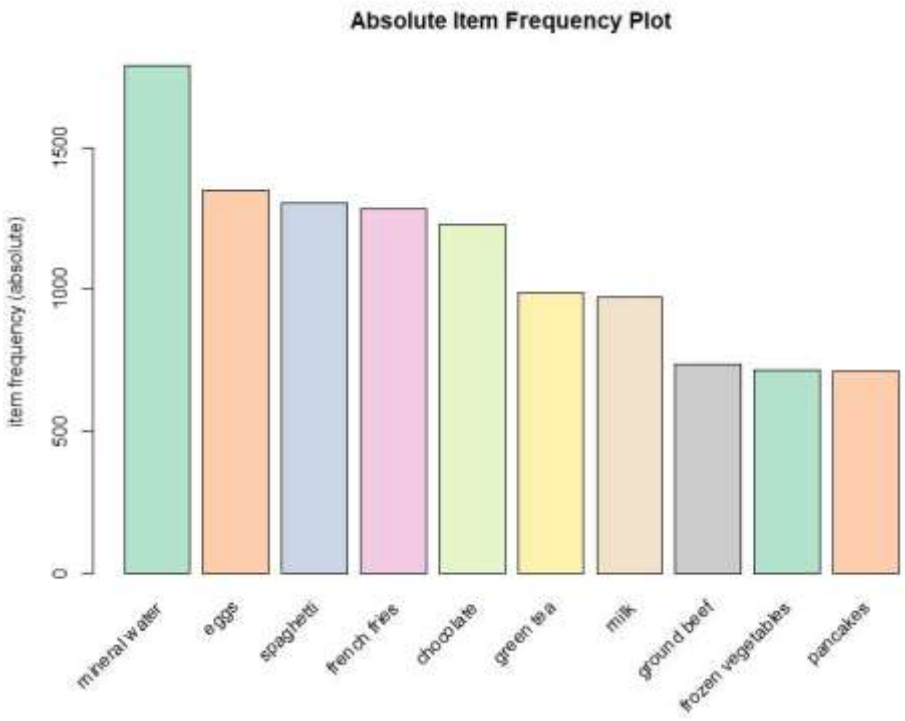
the misleading assassination rules having lift value less than 1 which means the correlation between the items is negative. Such as (AC, CA, AB D, CG D).

II. Store is interested in determining the associations between items purchased from its departments.

a. Generate a plot of the top 10 transactions.

```
> summary(df)
transactions as itemMatrix in sparse format with
 7501 rows (elements/itemsets/transactions) and
 5729 columns (items) and a density of 0.0005421748
```

```
# plot the frequency of items
itemFrequencyPlot(df,topN=10,
                  type="absolute",
                  col=brewer.pal(8,'Pastel2'),
                  main="Absolute Item Frequency Plot")
```



b. Generate association rules using minimum support of 0.002, minimum confidence of 0.20, and maximum length of 3. Display the rules, sorted by descending lift value.

```
# Min Support as 0.002, confidence as 0.20, maximum length as 3
association_rules <- apriori(df, parameter = list(supp=0.002, conf=0.20,maxlen=3))
summary(association_rules)
```

We got set of 2023 rules, we will sort them descending by lift value

```
> inspect(sort(association_rules, by = "lift"))
      lhs                                rhs                     support     confidence coverage    lift
[1]   {escalope,mushroom cream sauce}    => {pasta}               0.002532996 0.4418605  0.005732569 28.088096
[2]   {escalope,pasta}                   => {mushroom cream sauce} 0.002532996 0.4318182  0.005865885 22.650826
[3]   {mushroom cream sauce,pasta}       => {escalope}            0.002532996 0.9500000  0.002666311 11.976387
[4]   {parmesan cheese,tomatoes}         => {frozen vegetables}   0.002133049 0.6666667  0.003199573  6.993939
[5]   {mineral water,whole wheat pasta}  => {olive oil}           0.003866151 0.4027778  0.009598720  6.115863
[6]   {frozen vegetables,parmesan cheese} => {tomatoes}           0.002133049 0.3902439  0.005465938  5.706081
[7]   {burgers,herb & pepper}           => {ground beef}         0.002266364 0.5483871  0.004132782  5.581345
[8]   {light cream,mineral water}        => {chicken}             0.002399680 0.3272727  0.007332356  5.455273
[9]   {ground beef,shrimp}               => {herb & pepper}       0.002932942 0.2558140  0.011465138  5.172131
[10]  {fromage blanc}                    => {honey}               0.003332889 0.2450980  0.013598187  5.164271
[11]  {ground beef,low fat yogurt}       => {herb & pepper}       0.002399680 0.2500000  0.009598720  5.054582
[12]  {spaghetti,tomato sauce}           => {ground beef}         0.003066258 0.4893617  0.006265831  4.980600
[13]  {meatballs,spaghetti}              => {tomatoes}            0.002133049 0.3333333  0.006399147  4.873944
[14]  {light cream}                      => {chicken}             0.004532729 0.2905983  0.015597920  4.843951
[15]  {frozen vegetables,herb & pepper}  => {ground beef}         0.002799627 0.4666667  0.005999200  4.749616
[16]  {mineral water,tomato sauce}       => {ground beef}         0.002666311 0.4651163  0.005732569  4.733836
[17]  {pasta}                            => {escalope}            0.005865885 0.3728814  0.015731236  4.700812
[18]  {french fries,herb & pepper}       => {ground beef}         0.003199573 0.4615385  0.006932409  4.697422
[19]  {cereals,spaghetti}                => {ground beef}         0.003066258 0.4600000  0.006665778  4.681764
[20]  {french fries,ground beef}         => {herb & pepper}       0.003199573 0.2307692  0.013864818  4.665768
```

c. Select the rule from QII-b with the greatest lift. Compare this rule with the highest lift rule for maximum length of 2.

highest lift rule for maximum length of 3

```
> inspect(sort(association_rules, by = "lift")[1])
    lhs                                rhs     support     confidence coverage    lift    count
[1] {escalope,mushroom cream sauce} => {pasta} 0.002532996 0.4418605  0.005732569 28.0881 19
```

highest lift rule for maximum length of 2

```
> inspect(sort(association_rules2, by = "lift")[1])
    lhs                 rhs      support     confidence coverage   lift     count
[1] {fromage blanc} => {honey} 0.003332889 0.245098   0.01359819 5.164271 25
```

i. Which rule has the better lift?
Rule with max_length of 3 has better lift (28.0881).

ii. Which rule has the greater support?
Rule with max_length of 2 has greater support (0.002532996).

iii. If you were a marketing manager, and could fund only one of these rules, which would it be, and why?
I think rule with max_length of 3 is better to fund because it has high lift and high confidence.

Part B (Course Recommender System using Collaborative Filtering):

1. Computing correlations between all student pairs. For which students is it possible to compute correlations with E.N.

$$\mathrm{Corr}(U_1, U_2) = \frac{\sum(r_{1,i} - \bar{r}_1)(r_{2,i} - \bar{r}_2)}{\sqrt{\sum(r_{1,i} - \bar{r}_1)^2}\sqrt{\sum(r_{2,i} - \bar{r}_2)^2}},$$

$$Corr(EN, LN) = \frac{(4-3.75)(4-3) + (4-3.75)(4-3) + (2-3)(3-3.75)}{\sqrt{(4-3.75)^2 + (4-3.75)^2 + (3-3.75)^2} * \sqrt{(4-3)^2 + (4-3)^2 + (2-3)^2}}$$

$$\frac{1.25}{0.829 * 1.73} = 0.871$$

The same calculation for the rest of correlations: -

Corr(EN,MH) = $\frac{-0.1667}{0.1667}$ = -1

Corr(EN,FH) = 0

Corr(EN,FL) = 0

Corr(EN,AH) = 0

Corr(EN,RW) = 0

Corr(EN,MG) = 0

Corr(EN,DS) = 0

Corr(EN,DU) = 0

Corr(EN,GL) = 0

Corr(EN,SA) = 0

Corr(EN,BA) = 0

Corr(EN,AF) = 0

Corr(EN,KG) = 0

2. which single course should we recommend to E.N and why?

The highest correlation with EN is LN, so we will recommend "Python" because it`s not common and has the highest rate.

3. Use R to compute the cosine similarity between users.

I will create csv file for online statistics courses given in table 14.16

```
courses <- read_csv("course.csv")
View(courses)

#We can now remove user ids
ratingmat = as.matrix(courses[,-1])
cosine(ratingmat)
```

```
          LN        MH        JH        EN        DU        FL        GL        AH        SA        RW        BA
LN 1.0000000 0.5354529 0.4040610 0.7190319 0.4040610 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
MH 0.5354529 1.0000000 0.7730207 0.2482286 0.7730207 0.6246950 0.6246950 0.6246950 0.0000000 0.0000000 0.0000000
JH 0.4040610 0.7730207 1.0000000 0.3746343 1.0000000 0.7071068 0.7071068 0.7071068 0.0000000 0.0000000 0.0000000
EN 0.7190319 0.2482286 0.3746343 1.0000000 0.3746343 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
DU 0.4040610 0.7730207 1.0000000 0.3746343 1.0000000 0.7071068 0.7071068 0.7071068 0.0000000 0.0000000 0.0000000
FL 0.0000000 0.6246950 0.7071068 0.0000000 0.7071068 1.0000000 1.0000000 1.0000000 0.0000000 0.0000000 0.0000000
GL 0.0000000 0.6246950 0.7071068 0.0000000 0.7071068 1.0000000 1.0000000 1.0000000 0.0000000 0.0000000 0.0000000
AH 0.0000000 0.6246950 0.7071068 0.0000000 0.7071068 1.0000000 1.0000000 1.0000000 0.0000000 0.0000000 0.0000000
SA 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 1.0000000 0.4472136 1.0000000
RW 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.4472136 1.0000000 0.4472136
BA 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 1.0000000 0.4472136 1.0000000
MG 0.2020305 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.7071068 0.3162278 0.7071068
AF 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 1.0000000 0.4472136 1.0000000
KG 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 1.0000000 0.4472136 1.0000000
DS 0.7619048 0.3123475 0.4714045 0.8830216 0.4714045 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
          MG        AF        KG        DS
LN 0.2020305 0.0000000 0.0000000 0.7619048
MH 0.0000000 0.0000000 0.0000000 0.3123475
JH 0.0000000 0.0000000 0.0000000 0.4714045
EN 0.0000000 0.0000000 0.0000000 0.8830216
DU 0.0000000 0.0000000 0.0000000 0.4714045
FL 0.0000000 0.0000000 0.0000000 0.0000000
GL 0.0000000 0.0000000 0.0000000 0.0000000
AH 0.0000000 0.0000000 0.0000000 0.0000000
SA 0.7071068 1.0000000 1.0000000 0.0000000
RW 0.3162278 0.4472136 0.4472136 0.0000000
BA 0.7071068 1.0000000 1.0000000 0.0000000
MG 1.0000000 0.7071068 0.7071068 0.0000000
AF 0.7071068 1.0000000 1.0000000 0.0000000
KG 0.7071068 1.0000000 1.0000000 0.0000000
DS 0.0000000 0.0000000 0.0000000 1.0000000
```

Calculated the cosine similarity using lsa library in R

4. Based on the cosine similarities of the nearest students to E.N., which course should be recommended to E.N.?

As we can see from the above matrix the maximum similarity to **EN** is **DS**, but all rating courses by **DS** also common with **EN**, so we can't recommend any course of **DS** to **EN.**
So, we will see the next one is **LN** with value of (**0.7190319**), then we found that the maximum rating with **LN** is (SQL, R Prog and Python), so we will recommend **Python** course because it`s not rated by **EN**.

5. Apply item-based collaborative filtering to this dataset (using R) and based on the results, recommend a course to E.N.

```
#create dataframe
user <- c('LN','MH','JH','EN','DU','FL','GL', 'AH' ,'SA','RW','BA','MG','AF','KG','DS')
SQL <- c(4,3,2,4,4 ,NA,NA,NA,NA,NA,NA,NA,NA,NA,4)
Spatial <- c(NA ,4,2 ,NA ,4,4,4,3,NA,NA,NA,NA,NA,NA,NA)
PA1 <- c(NA,NA,NA,NA,NA,NA,NA,NA, 4,2,4,4,4,3,NA)
DM_IN_R <- c(NA,NA,NA ,4,NA,NA,NA,NA,NA,NA,NA,NA,NA,NA,2)
PYTHON <- c(3,4,NA,NA,NA,NA,NA,NA,NA,NA,NA,NA,NA,NA,NA)
FORECAST <- c(2 ,NA,NA,NA,NA,NA,NA,NA,NA,NA,NA,4,NA,NA,NA)
R_PROG <- c(4 ,NA,NA ,4 ,NA,NA,NA,NA,NA,NA,NA,NA,NA,NA,4)
HADOOP <- c(NA,NA,NA,NA,NA,NA,NA,NA,NA,4, NA,NA ,NA,NA,NA)
REGRESSION <- c(2 ,NA,NA,3,NA,NA,NA,NA,NA,NA,NA,NA,NA,NA ,NA)
df <- data.frame(user,SQL,Spatial,PA1,DM_IN_R,PYTHON,FORECAST,R_PROG,HADOOP,REGRESSION)

#convert to matrix and remove first column
ratingmat2 = as.matrix(df[,-1])
#Convert ratings matrix to real rating matrx which makes it dense
ratingmat2 = as(ratingmat2, "realRatingMatrix")
#Create Recommender Model. The parameters are IBCF and Cosine similarity.
rec_mod = Recommender(ratingmat2, method = "IBCF", param=list(method="Cosine", k=9))
#Obtain top 3 recommendations for 4th user entry in dataset
Top_5_pred = predict(rec_mod, ratingmat2[4], n=3)
#Convert the recommendations to a list
Top_5_List = as(Top_5_pred, "list")
Top_5_List
```

```
> Top_3_List
[[1]]
[1] "FORECAST" "Spatial"  "PYTHON"
```

Based on above results we will recommend **Forecast**.