Kunal Mehta
50352960
kunalkam

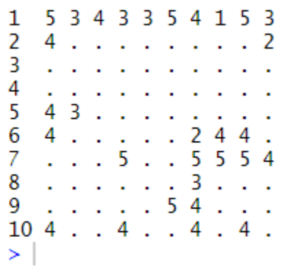
Homework 5

Problem 3

1. The first step was to load the recommenderlab package and access the MovieLense dataset. This was straightforward, and a cursory examination of MovieLense shows that it is of the form realRatingsMatrix.

2. The next step was to specify k, the number of most similar users to user i that must be taken into consideration. I arbitrarily chose it to be 30.

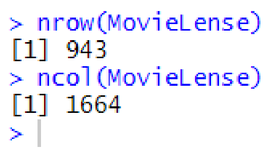
3. To get a feel of what the matrix was like, I printed out the first 10 rows and columns of it using getRatingMatrix()



```
1  5 3 4 3 3 5 4 1 5 3
2  4 . . . . . . . . 2
3  . . . . . . . . . .
4  . . . . . . . . . .
5  4 3 . . . . . . . .
6  4 . . . . . 2 4 4 .
7  . . . 5 . . 5 5 5 4
8  . . . . . . 3 . . .
9  . . . . . 5 4 . . .
10 4 . . 4 . . 4 . 4 .
>
```

   The dots represent missing ratings for the corresponding user, movie pair

4. Just to make sure that the number of users and movies specified in the problem statement was accurate, I displayed the number of rows and columns of MovieLense using nrow and ncol.



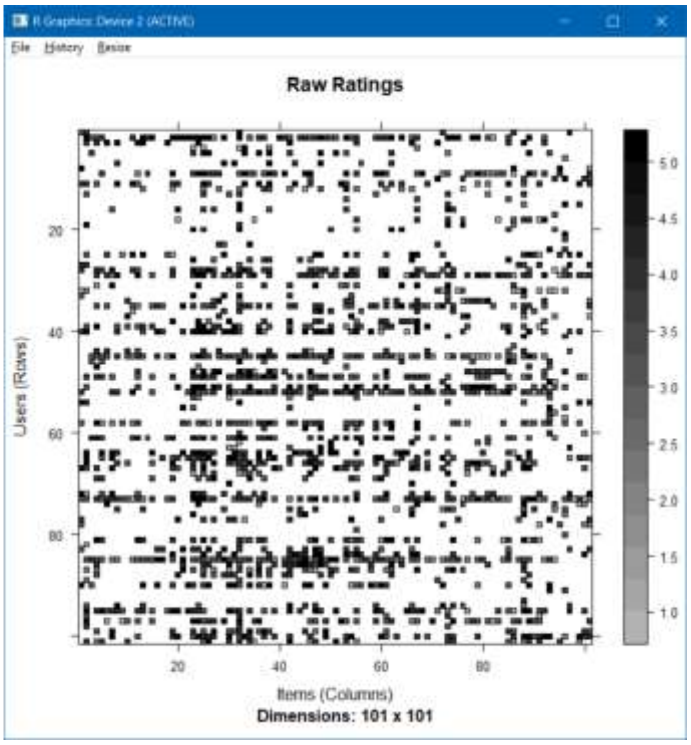
```
> nrow(MovieLense)
[1] 943
> ncol(MovieLense)
[1] 1664
>
```

   As expected, the values were accurate.

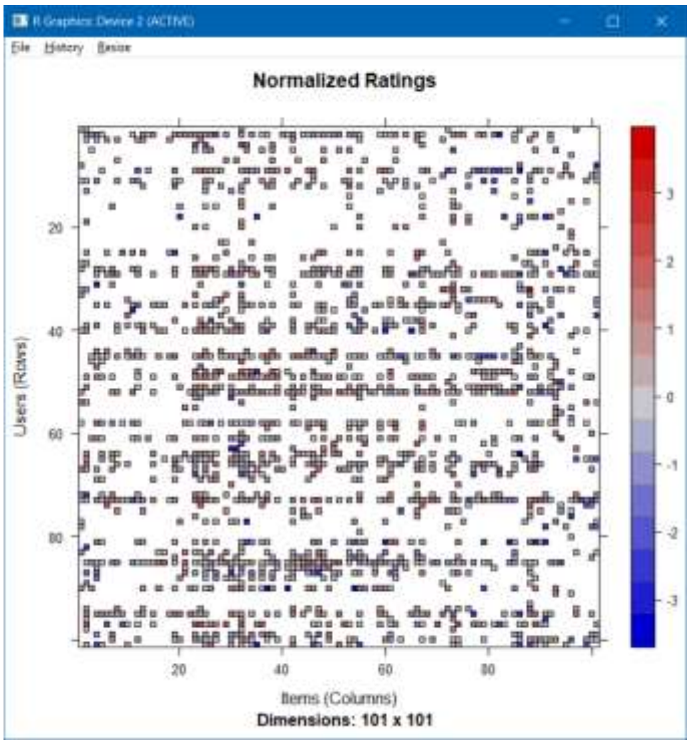
5. The next step I did was to normalize the ratings and store it in dats. To see what it looked like, I visualized both the raw and normalized ratings for a small subset of the data, the 100 x 100 rows and columns in the range 150:250.

Kunal Mehta
50352960
kunalkam

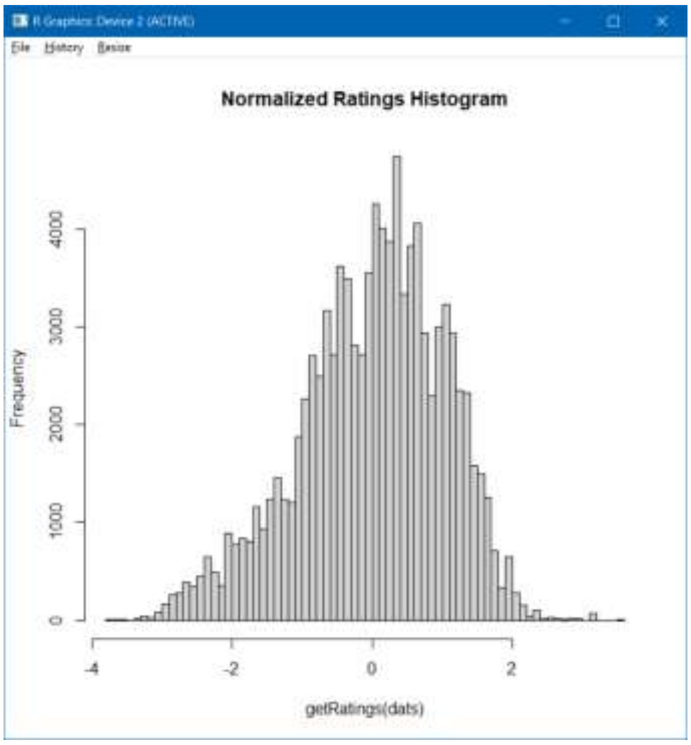
Raw ratings:



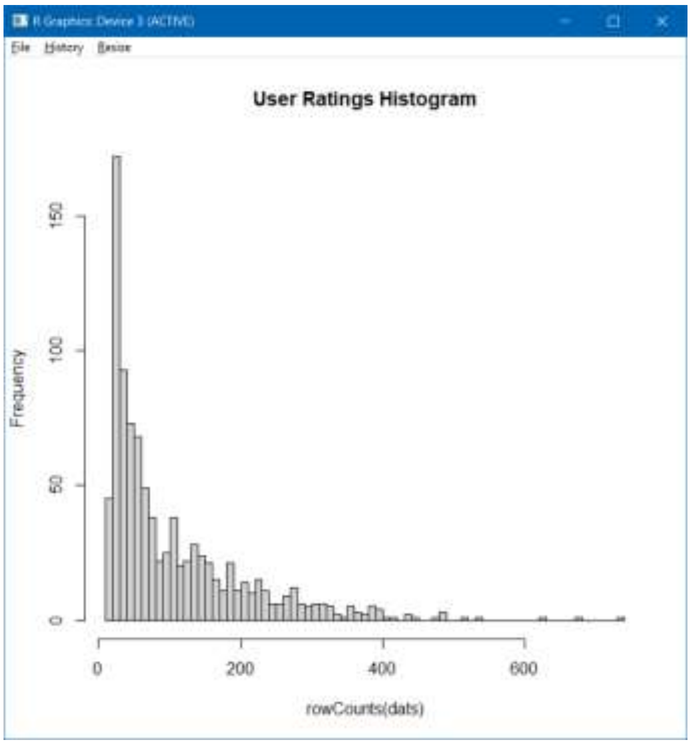
Normalized ratings:



6. Next, to gain some more insight into the users and their rating patterns, I plotted histograms of the data.
   Normalized Ratings Histogram:

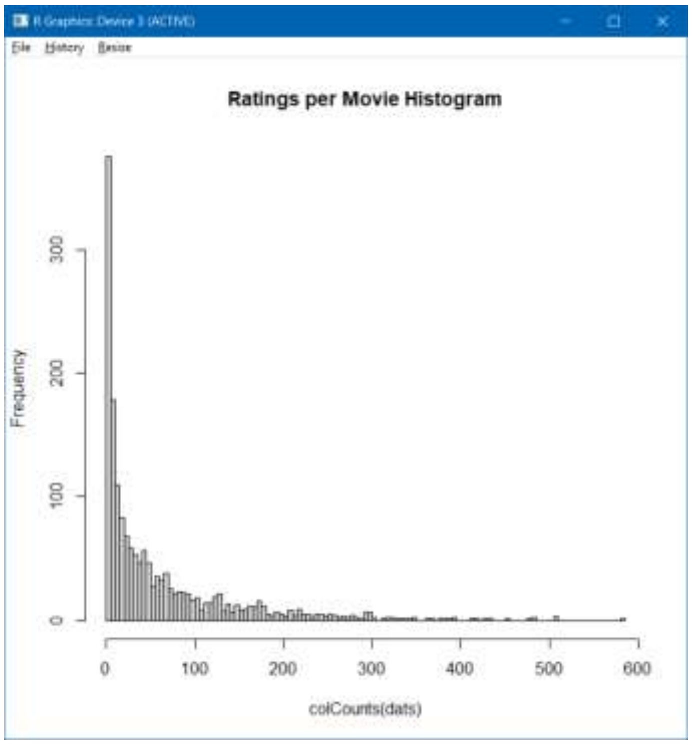


   A mostly normal distribution, with greater tendency for users to rate negatively, which is a common phenomenon.
   User Ratings Histogram:



Kunal Mehta
50352960
kunalkam

Ratings per Movie Histogram:



7. To find the k most similar users to i, I needed to make a similarity matrix. Now, the similarity matrix will be the same when calculated, so there is no point calculating it for every user i. So I calculated the similarity matrix using similarity, with cosine method as an argument, and converted it into a matrix. I also initialized a matrix with zeros that will be my final matrix.

8. The next step was to iterate through all users. I set a loop to run from 1 to 943, an iteration for each user. In each iteration I do the following:
    a. Run a loop from 1 to 1664, checking the rating of each movie for that user to find the first movie that is not rated. This movie number is stored in j
    b. Create an empty list users, which will store all the users that have watched j
    c. Run a loop from 1 to 943 that check's each user's rating for movie j and if the rating is not NA, then append it to users
    d. Take k to be length of users if no of users that have watched j is less than k
    e. List the similarity of chosen users and order it in descending order, followed by k subset
    f. Design the recommender system for user i using just the ratings of the chosen k users, with method as User based Collaborative Filtering
    g. Predict the ratings for user i using the model created in the previous step, making sure to keep type as ratingMatrix
    h. The last step is to add all these predictions to the final matrix

9. This entire code block gives a final matrix that can simply be queried to give the rating, predicted or original of any user x and movie y as final[x,y].
   Some sample outputs for this is:

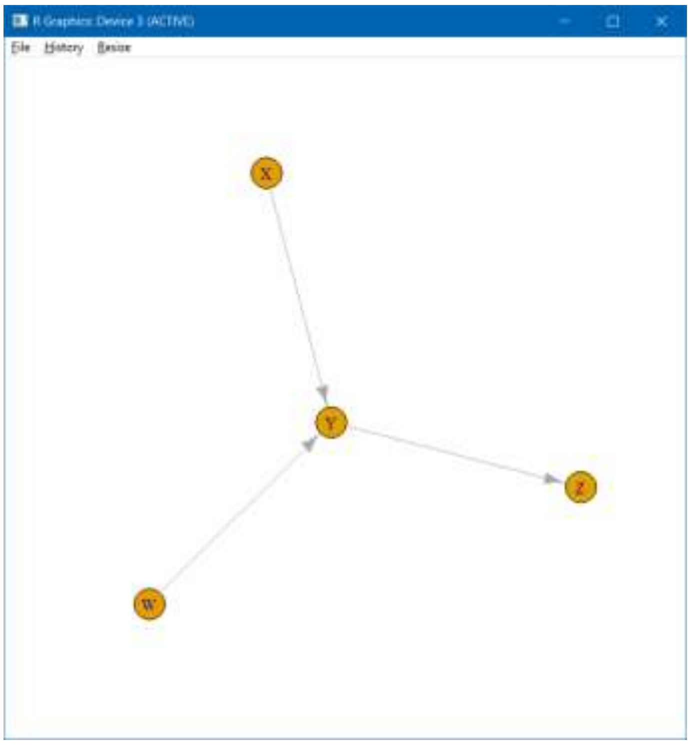
Kunal Mehta
50352960
kunalkam

```
> kk[2,1]
[1] 4
> final[2,1]
[1] 4
> kk[3,1000]
[1] NA
> final[3,1000]
[1] 2.764706
> kk[123,1345]
[1] NA
> final[123,1345]
[1] 3.851852
```

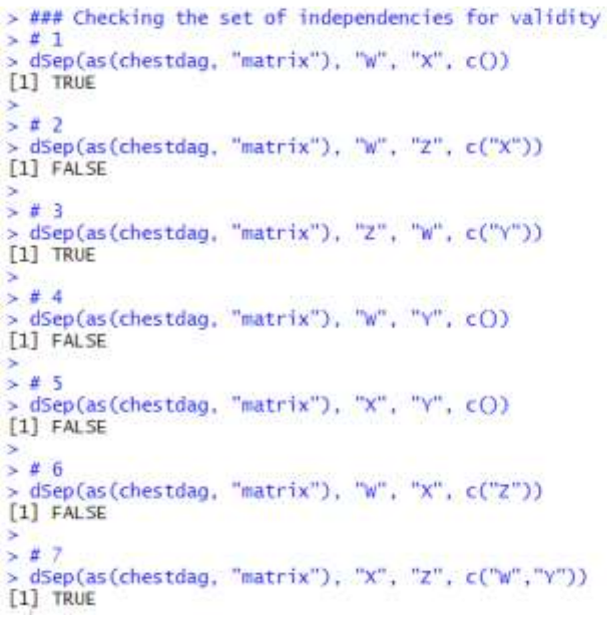
Any query for this matrix will give the rating for the corresponding user and movie. Kk is the MovieLense data in matrix form, as you see, kk[2,1] = final[2,1], since this is an original rating. Bit for 3,1000, the original data had no rating, and the predicted rating is 2.764706. Similarly for 123,1345, the new predicted rating is 3.851852.

Problem 2

1. The Bayesian Net that obeys the independencies is shown below:



I also tested the graph to see if the independencies corresponded. The output for the same is shown below:



```
> ### Checking the set of independencies for validity
> # 1
> dSep(as(chestdag, "matrix"), "W", "X", c())
[1] TRUE
>
> # 2
> dSep(as(chestdag, "matrix"), "W", "Z", c("X"))
[1] FALSE
>
> # 3
> dSep(as(chestdag, "matrix"), "Z", "W", c("Y"))
[1] TRUE
>
> # 4
> dSep(as(chestdag, "matrix"), "W", "Y", c())
[1] FALSE
>
> # 5
> dSep(as(chestdag, "matrix"), "X", "Y", c())
[1] FALSE
>
> # 6
> dSep(as(chestdag, "matrix"), "W", "X", c("Z"))
[1] FALSE
>
> # 7
> dSep(as(chestdag, "matrix"), "X", "Z", c("W","Y"))
[1] TRUE
```