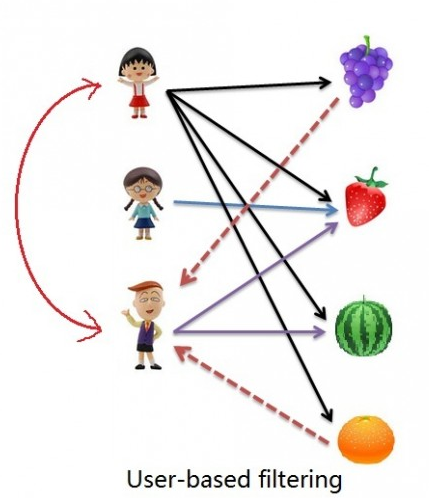
**Movie Ratings Recommender Using R**

By Anamika Jatkar

**Introduction:**

Analyzing the movielens dataset and building a recommendation system based on the rating given to the movies by the users in the dataset to predict user ratings for new movies.

The base approach used to develop this recommendation system was User based collaborative filtering. Basic idea behind the approach is predicting ratings based on the ratings already given by users to a set of movies. The system matches one user’s ratings against other users to find the similarity in their tastes. Based on this similarity the ratings for other movies given by this user are predicted. In other words, using this similarity index, the system recommends movies rated highly by similar users but not yet rated by this user. The following illustration represents this approach most accurately.



**Data used:**

The dataset used for this exercise is the movielens dataset, which has around 6000 movies and they are rated by users belonging to different age groups and occupations.

A very basic exploratory analysis of the data was done using ggplots prior to developing the recommender system.

**Approach and code snippet:**

The recommender lab package of R was used to develop the system. The recommenderlab package uses a data-structure ratingMatrix to provide a common interface for rating data. This data structure implements many of the methods of matrix structure in R such as: dim(), dimnames(), colCounts(), rowCounts(), colMeans(), rowMeans() , colSums() and rowSums().

The method used for accomplishing the actual prediction part is the UBCF (user based collaborative filtering) available in R recommenderlab package. The UBCF has two parameter methods that define how the similarity distance is measured: cosine and jaccard. After trying out both methods on the same dataset, the jaccard method was found to be more suitable for the movielens 6000 movie dataset.

The Jaccard coefficient originally coined *coefficient de communauté* by [Paul Jaccard](https://en.wikipedia.org/wiki/Paul_Jaccard) measures similarity between finite sample sets.

Following is the code snippet that explains the approach in details :

# Installing the required packages in R for this project

library**(**recommenderlab**)**

library**(**reshape2**)**

library**(**ggplot2**)**

# Read training file along with header

tr **<-** read.csv**(**"C:/Users/Anamikja/Desktop/Anamika/Springboard/train\_v2.csv", header **=** **TRUE)**

head**(**tr**)**

# Removing 'id' column since it is not needed

tr**<-**tr**[**,**-**c**(**1**)]**

tr**[**tr**$**user**==**1,**]**

# re working the data to resemble a matrix like structure

g**<-**acast**(**tr, user **~** movie**)**

class**(**g**)**

# Converting g into an actual matrix

R**<-**as.matrix**(**g**)**

# Converting R into realRatingMatrix data structure

# realRatingMatrix is a recommenderlab sparse-matrix like data-structure

r **<-** as**(**R, "realRatingMatrix"**)**

r

# normalizing the rating matrix

r\_m **<-** normalize**(**r**)**

r\_m

# viewing it as a list

as**(**r\_m, "list"**)**

# plotting an image plot of raw-ratings & normalized ratings

# A column represents one specific movie and ratings by users

# are shaded.

# Note that some items are always rated 'black' by most users

# while some items are not rated by many users

# On the other hand a few users always give high ratings

# as in some cases a series of black dots cut across items

image**(**r, main **=** "Raw Ratings"**)**

image**(**r\_m, main **=** "Normalized Ratings"**)**

# Creating a recommender object (model)

# UBCF: User-based collaborative filtering

# Parameter 'method' decides similarity measure

# Jaccard method of distance measuring is implemented

rec**=**Recommender**(**r**[**1**:**nrow**(**r**)]**,method**=**"UBCF", param**=**list**(**normalize **=** "Z-score",method**=**"Jaccard",nn**=**5, minRating**=**1**))**

# Depending upon your selection, examine what you got

print**(**rec**)**

names**(**getModel**(**rec**))**

getModel**(**rec**)$**nn

############Create predictions#############################

# This prediction does not predict movie ratings for test.

# But it fills up the user 'X' item matrix so that

# for any userid and movieid, I can find predicted rating

# 'type' parameter decides whether you want ratings or top-n items

# To get top-10 recommendations for a user :

# predict(rec, r[1:nrow(r)], type="topNList", n=10)

recom **<-** predict**(**rec, r**[**1**:**nrow**(**r**)]**, type**=**"ratings"**)**

recom

########## Examination of model & experimentation #############

# Converting prediction into list, user-wise

as**(**recom, "list"**)**

# Study and Compare the following:

as**(**r, "matrix"**)** # Has lots of NAs. 'r' is the original matrix

as**(**recom, "matrix"**)** # Is full of ratings. NAs disappear

as**(**recom, "matrix"**)[**,1**:**10**]** # Show ratings for all users for items 1 to 10

as**(**recom, "matrix"**)[**5,3**]** # Rating for user 5 for item at index 3

as.integer**(**as**(**recom, "matrix"**)[**5,3**])** # Just to get the integer value

as.integer**(**round**(**as**(**recom, "matrix"**)[**6039,8**]))** # Just to get the correct integer value

as.integer**(**round**(**as**(**recom, "matrix"**)[**368,3717**]))**

# Convert all your recommendations to list structure

rec\_list**<-**as**(**recom,"list"**)**

head**(**summary**(**rec\_list**))**

# Access this list. User 2, item at index 2

rec\_list**[[**2**]][**2**]**

# Convert to data frame all recommendations for user 1

u1**<-**as.data.frame**(**rec\_list**[[**1**]])**

attributes**(**u1**)**

class**(**u1**)**

# Create a column by name of id in data frame u1 and populate it with row names

u1**$**id**<-**row.names**(**u1**)**

# Check movie ratings are in column 1 of u1

u1

# Now access movie ratings in column 1 for u1

u1**[**u1**$**id**==**3952,1**]**

########## Create final File from model #######################

# Read test file

test**<-**read.csv**(**"C:/Users/Anamikja/Desktop/Anamika/Springboard/test\_v2.csv",header**=TRUE)**

head**(**test**)**

# Get ratings list

rec\_list**<-**as**(**recom,"list"**)**

head**(**summary**(**rec\_list**))**

ratings**<-NULL**

# For all lines in test file, one by one

**for** **(** u **in** 1**:**length**(**test**[**,2**]))**

**{**

# Read userid and movieid from columns 2 and 3 of test data

userid **<-** test**[**u,2**]**

movieid**<-**test**[**u,3**]**

# Get as list & then convert to data frame all recommendations for user: userid

u1**<-**as.data.frame**(**rec\_list**[[**userid**]])**

# Creating a (second column) column-id in the data-frame u1 and populate it with row-names

# We use row.names() function

u1**$**id**<-**row.names**(**u1**)**

# Now access movie ratings in column 1 of u1

x**=** u1**[**u1**$**id**==**movieid,1**]**

# print(u)

# print(length(x))

# If no ratings were found, assign 0.

**if** **(**length**(**x**)==**0**)**

**{**

ratings**[**u**]** **<-** 0

**}**

**else**

**{**

ratings**[**u**]** **<-**x

**}**

**}**

length**(**ratings**)**

tx**<-**cbind**(**test**[**,1**]**,round**(**ratings**))**

# Write to a csv file

write.table**(**tx,file**=**"Finalfile.csv",row.names**=FALSE**,col.names**=FALSE**,sep**=**','**)**

**Future research:**

The approach used here was purely user based, since there was little metadata around the movies. For richer movie datasets that have more information around the movies themselves, a hybrid approach of item based collaborative filtering and user based collaborative filtering is worth researching into. It will require massive etls on various datasets and a fine tuning of the model based on multiple training sets, but that approach will be more intuitive than just the user based or items based collaborative filtering.

Another area of research that is possible post this and can leverage the hybrid recommendation approach is analyzing movie reviews and nano-genres stemming from those reviews. Calculating a similarity index based off that will generate more accurate recommendations for users.

**Intended clients:**

The customers who can leverage this system in making their business decisions are movie studios and production houses. Based on their movie genres and ratings given by the users to similar movies in the past the movie studios can make the following key decisions:

1. When to release a particular movie?
2. What regions and demographics should the movie be promoted to more or less?
3. Who is the actual target audience and how to reach them in minimum costs?
4. Which TV channels or digital or broadcast platforms the movie rights be sold to and at what costs based on the platforms demographic and average audience?

Until now these decisions are taken based on basic information. With the recommender systems in place these companies can make informed decisions that would cut losses and make more profits.

**Data References:**

The dataset acquired for this exercise is available free for academic purposes by the GroupLens Research team.

*GroupLens Research has collected and made available rating data sets from the MovieLens web site (*[*http://movielens.org*](http://movielens.org/)*). The data sets were collected over various periods of time, depending on the size of the set.*