Project 2

Igor Balagula

February 17, 2017

## Objective

Take some recommendation data and implement at least two different recommendation algorithms on the data. For example, content-based, user-user CF, and/or item-item CF. Evaluate different approaches, using different algorithms, normalization techniques, similarity methods, neighborhood sizes, etc.

## Data

The 100K MovieLens dataset of movies will be used. The dataset contains records of 672 users and their ratings of 9066 movies.

Load data

setwd("E:\\Igor\\CUNY\\DATA 643 - Recommender Systems\\Projects\\Project\_02\\Data")  
ratings<-read.csv("ratings.csv")  
movies<-read.csv("movies.csv",stringsAsFactors=FALSE)

Review loaded data

head(ratings)

## userId movieId rating timestamp  
## 1 1 31 2.5 1260759144  
## 2 1 1029 3.0 1260759179  
## 3 1 1061 3.0 1260759182  
## 4 1 1129 2.0 1260759185  
## 5 1 1172 4.0 1260759205  
## 6 1 1263 2.0 1260759151

head(movies)

## movieId title  
## 1 1 Toy Story (1995)  
## 2 2 Jumanji (1995)  
## 3 3 Grumpier Old Men (1995)  
## 4 4 Waiting to Exhale (1995)  
## 5 5 Father of the Bride Part II (1995)  
## 6 6 Heat (1995)  
## genres  
## 1 Adventure|Animation|Children|Comedy|Fantasy  
## 2 Adventure|Children|Fantasy  
## 3 Comedy|Romance  
## 4 Comedy|Drama|Romance  
## 5 Comedy  
## 6 Action|Crime|Thriller

Remove timestap from ratings dataframe

ratings <- ratings[,c(1,2,3)]

Create ratings matrix. Rows = userId, Columns = movieId

ratingmat <- dcast(ratings, userId~movieId, value.var = "rating", na.rm=FALSE)

Remove userIds in the first column

ratingmat <- as.matrix(ratingmat[,-1])

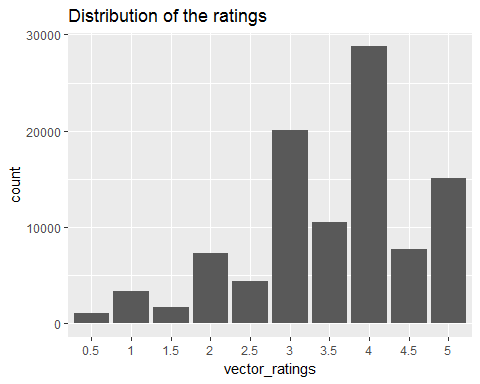
Convert rating matrix into a recommenderlab sparse matrix

ratingmat <- as(ratingmat, "realRatingMatrix")  
ratingmat

## 672 x 9069 rating matrix of class 'realRatingMatrix' with 100024 ratings.

Let's see how ratings are distributed

vector\_ratings <- as.vector(ratingmat@data)  
vector\_ratings <- vector\_ratings[vector\_ratings != 0]  
vector\_ratings <- factor(vector\_ratings)  
qplot(vector\_ratings) + ggtitle("Distribution of the ratings")

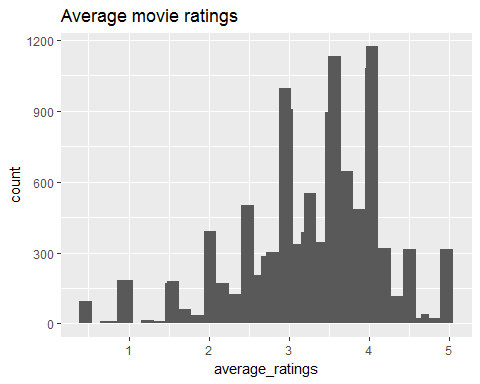


The most common rating is 4.

Now let's see how the average ratings for each individual movie are distributed

average\_ratings <- colMeans(ratingmat)  
qplot(average\_ratings) + stat\_bin(binwidth = 0.1) +  
ggtitle("Average movie ratings")

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



To remove some extreme cases (with very small number of ratings) we will reduce the matrix - select only those movies that were rated 50 or more times and only users who rated 50 or more movies

ratingmat <- ratingmat[rowCounts(ratingmat)>= 50,colCounts(ratingmat)>=50 ]  
ratingmat

## 427 x 453 rating matrix of class 'realRatingMatrix' with 38538 ratings.

Separate data into a training and test data sets (as described in the textbook p.78)

set.seed(1)  
percentage\_training <-0.8  
items\_to\_keep <- 5  
rating\_threshold <- 3  
n\_eval <- 1  
eval\_sets <- evaluationScheme(data=ratingmat, method="split", train=percentage\_training, given=items\_to\_keep,  
goodRating=rating\_threshold,k=n\_eval)  
eval\_sets

## Evaluation scheme with 5 items given  
## Method: 'split' with 1 run(s).  
## Training set proportion: 0.800  
## Good ratings: >=3.000000  
## Data set: 427 x 453 rating matrix of class 'realRatingMatrix' with 38538 ratings.

getData(eval\_sets, "train")

## 341 x 453 rating matrix of class 'realRatingMatrix' with 30234 ratings.

Now we will evaluate prediction models with different parameters.

1. User-based collaborative filtering; distance method = cosine; normalize= "Z-Score"

recommender\_model <- Recommender(getData(eval\_sets, "train"), method = "UBCF", param=list(normalize= "Z-Score",method="Cosine",nn=30))  
recom <- predict(recommender\_model, getData(eval\_sets, "known"), type="ratings")  
predict\_acc\_ubcf\_cosine\_z <- calcPredictionAccuracy(recom, getData(eval\_sets, "unknown"))[1]   
predict\_acc\_ubcf\_cosine\_z

## RMSE   
## 0.9575594

1. User-based collaborative filtering; distance method = jaccard; normalize= "Z-Score"

recommender\_model <- Recommender(getData(eval\_sets, "train"), method = "UBCF", param=list(normalize= "Z-Score",method="jaccard",nn=30))  
recom <- predict(recommender\_model, getData(eval\_sets, "known"), type="ratings")  
predict\_acc\_ubcf\_jaccard\_z <- calcPredictionAccuracy(recom, getData(eval\_sets, "unknown"))[1]   
predict\_acc\_ubcf\_jaccard\_z

## RMSE   
## 0.9688468

1. User-based collaborative filtering; distance method = pearson; normalize= "Z-Score"

recommender\_model <- Recommender(getData(eval\_sets, "train"), method = "UBCF", param=list(normalize= "Z-Score",method="pearson",nn=30))  
recom <- predict(recommender\_model, getData(eval\_sets, "known"), type="ratings")  
predict\_acc\_ubcf\_pearson\_z <- calcPredictionAccuracy(recom, getData(eval\_sets, "unknown"))[1]   
predict\_acc\_ubcf\_pearson\_z

## RMSE   
## 0.9402985

1. Item-based collaborative filtering; distance method = cosine; normalize= "Z-Score"

recommender\_model <- Recommender(getData(eval\_sets, "train"), method = "IBCF", param=list(normalize= "Z-Score",method="Cosine"))  
recom <- predict(recommender\_model, getData(eval\_sets, "known"), type="ratings")  
predict\_acc\_ibcf\_cosine\_z <- calcPredictionAccuracy(recom, getData(eval\_sets, "unknown"))[1]   
predict\_acc\_ibcf\_cosine\_z

## RMSE   
## 1.189704

1. Item-based collaborative filtering; distance method = jaccard; normalize= "Z-Score"

recommender\_model <- Recommender(getData(eval\_sets, "train"), method = "IBCF", param=list(normalize= "Z-Score",method="jaccard"))  
recom <- predict(recommender\_model, getData(eval\_sets, "known"), type="ratings")  
predict\_acc\_ibcf\_jaccard\_z <- calcPredictionAccuracy(recom, getData(eval\_sets, "unknown"))[1]   
predict\_acc\_ibcf\_jaccard\_z

## RMSE   
## 1.07831

1. Item-based collaborative filtering; distance method = pearson; normalize= "Z-Score"

recommender\_model <- Recommender(getData(eval\_sets, "train"), method = "IBCF", param=list(normalize= "Z-Score",method="pearson"))  
recom <- predict(recommender\_model, getData(eval\_sets, "known"), type="ratings")  
predict\_acc\_ibcf\_pearson\_z <- calcPredictionAccuracy(recom, getData(eval\_sets, "unknown"))[1]   
predict\_acc\_ibcf\_pearson\_z

## RMSE   
## 1.151008

1. User-based collaborative filtering; distance method = cosine; normalize= "center"

recommender\_model <- Recommender(getData(eval\_sets, "train"), method = "UBCF", param=list(normalize= "center",method="Cosine",nn=30))  
recom <- predict(recommender\_model, getData(eval\_sets, "known"), type="ratings")  
predict\_acc\_ubcf\_cosine\_c <- calcPredictionAccuracy(recom, getData(eval\_sets, "unknown"))[1]   
predict\_acc\_ubcf\_cosine\_c

## RMSE   
## 0.9574755

1. User-based collaborative filtering; distance method = jaccard; normalize= "center"

recommender\_model <- Recommender(getData(eval\_sets, "train"), method = "UBCF", param=list(normalize= "center",method="jaccard",nn=30))  
recom <- predict(recommender\_model, getData(eval\_sets, "known"), type="ratings")  
predict\_acc\_ubcf\_jaccard\_c <- calcPredictionAccuracy(recom, getData(eval\_sets, "unknown"))[1]   
predict\_acc\_ubcf\_jaccard\_c

## RMSE   
## 0.9684662

1. User-based collaborative filtering; distance method = pearson; normalize= "center"

recommender\_model <- Recommender(getData(eval\_sets, "train"), method = "UBCF", param=list(normalize= "center",method="pearson",nn=30))  
recom <- predict(recommender\_model, getData(eval\_sets, "known"), type="ratings")  
predict\_acc\_ubcf\_pearson\_c <- calcPredictionAccuracy(recom, getData(eval\_sets, "unknown"))[1]   
predict\_acc\_ubcf\_pearson\_c

## RMSE   
## 0.9409478

1. Item-based collaborative filtering; distance method = cosine; normalize= "center"

recommender\_model <- Recommender(getData(eval\_sets, "train"), method = "IBCF", param=list(normalize= "center",method="Cosine"))  
recom <- predict(recommender\_model, getData(eval\_sets, "known"), type="ratings")  
predict\_acc\_ibcf\_cosine\_c <- calcPredictionAccuracy(recom, getData(eval\_sets, "unknown"))[1]   
predict\_acc\_ibcf\_cosine\_c

## RMSE   
## 1.183001

1. Item-based collaborative filtering; distance method = jaccard; normalize= "center"

recommender\_model <- Recommender(getData(eval\_sets, "train"), method = "IBCF", param=list(normalize= "center",method="jaccard"))  
recom <- predict(recommender\_model, getData(eval\_sets, "known"), type="ratings")  
predict\_acc\_ibcf\_jaccard\_c <- calcPredictionAccuracy(recom, getData(eval\_sets, "unknown"))[1]   
predict\_acc\_ibcf\_jaccard\_c

## RMSE   
## 1.07831

1. Item-based collaborative filtering; distance method = pearson; normalize= "center"

recommender\_model <- Recommender(getData(eval\_sets, "train"), method = "IBCF", param=list(normalize= "center",method="pearson"))  
recom <- predict(recommender\_model, getData(eval\_sets, "known"), type="ratings")  
predict\_acc\_ibcf\_pearson\_c <- calcPredictionAccuracy(recom, getData(eval\_sets, "unknown"))[1]   
predict\_acc\_ibcf\_pearson\_c

## RMSE   
## 1.153347

## Conclusion

Judging by RMSE values User-Based Collaborative Filtering provides more accurate predictions as compared to Item-Based Collaborative Filtering. Among UBCF all of the methods provide similar accuracy of predictions. However Pearson distance coupled with "center" normalization method provides the best prediction accuracy within UBCF.