hw3_643_DieudonneO

Dieudonne

July03, 2016

RECOMMENDER SYSTEM ON MOVIE LENS DATA

INTRODUCTION

This is the third mini project I wrote for my course Data 643 at CUNY #THE GOAL HERE IS TO USE MATRIX SINGULAR VALUE DECOMPOSITION TO DO RECOMMENDATIONS

I use mainly recommenderlab,write few functions and predict recommendations to users using various filtering methods and i compare the methods

There are 2 sets of data u.data which is ratings data and u.item data which is movie data

The data are located here http://grouplens.org/datasets/movielens/

```
library(recommenderlab)
library(reshape2)
```

FUNCTION TO GRAB THE DATA

```
##laod ratings data
get.Data <- function(){
  ratings <- read.delim("~/Downloads/u.data.txt", header=F)
  colnames(ratings) <- c("userID","movieID","rating", "timestamp")

  ## load movies data
  movies <- read.delim("~/Downloads/u.item.txt", sep="|", header=F, stringsAsFactors = FALSE)
  colnames(movies)[colnames(movies)=="V1"] <- "movieID"
  colnames(movies)[colnames(movies)=="V2"] <- "name"

  return(list(ratings=ratings, movies=movies))

}
```

FUNCTION FOR DATA PREPARATION AND PROCESSING

```
Pre.Process = function(ratings, movies)
{
  ratings[,2] <- dataList$movies$name[as.numeric(ratings[,2])]

  # remove duplicate entries for any user-movie combination
  ratings <- ratings[!duplicated(ratings[,1:2]),]
}
```

Function to Create movie ratingMatrix from rating Data and movie data

```
Create.Rating.Matrix <- function(ratings)
{
  # converting the ratingData data frame into rating matrix
  Ratings.Mat <- dcast( ratings, userID ~ movieID, value.var = "rating" , index="userID")
  ratings <- Ratings.Mat[,2:ncol(Ratings.Mat)]

  Ratings.Mat.Fin <- as(ratings, "matrix")  ## cast data frame as matrix
  movie.Rating.Mat <- as(Ratings.Mat.Fin, "realRatingMatrix")   ## create the realRatingMatrix
  ### setting up the dimnames ###
  dimnames(movie.Rating.Mat)[[1]] <- row.names(ratings)
  return (movie.Rating.Mat)
}
```

MODELS

```
evaluateModels <- function(movie.Rating.Mat)

{
  ## Find out and analyse available  recommendation algorithm options for realRatingMatrix data
  recommenderRegistry$get_entries(dataType = "realRatingMatrix")

  scheme <- evaluationScheme(movie.Rating.Mat, method = "split", train = .9,
                             k = 1, given = 10, goodRating = 4)

  algorithms <- list(
    RANDOM = list(name="RANDOM", param=NULL),
    POPULAR = list(name="POPULAR", param=NULL),
    UBCF = list(name="UBCF", param=NULL),
    IBCF= list(name="IBCF",param=NULL),
    SVD=list(name="SVD",param=NULL)
  )

  # run algorithms, predict next n movies
  res <- evaluate(scheme, algorithms, n=c(1, 3, 5, 10, 15, 20))

  ## select the first results

  return (res)
}
```

VISUALIZATION

```
graphs <- function(res)
{
```

```r
  # Draw ROC curve
  plot(res, annotate = 1:5, legend="topright")

  # See precision / recall
  plot(res, "prec/rec", annotate=5, legend="topright", xlim=c(0,.22))
}
```

CREATE FUNCTION FOR PREDICTION MODEL

```r
create.Model <-function (movie.Rating.Mat,method){

  model <- Recommender(movie.Rating.Mat, method = method)
  names(getModel(model))
  getModel(model)$method

  getModel(model)$nn

  return (model)
}
```

RATINGS PREDICTIONS USING USER BASED C FILTERING RECOMMENDATIONS

```r
rec <- function(movie.Rating.Mat, model, userID, n)
{

  ### PREDICT THE TOP N recommendations for given user
  Top.N.List <-predict(model,movie.Rating.Mat[userID],n=n)
  as(Top.N.List,"list")
}
```

LOAD MOVIE LENS DATA

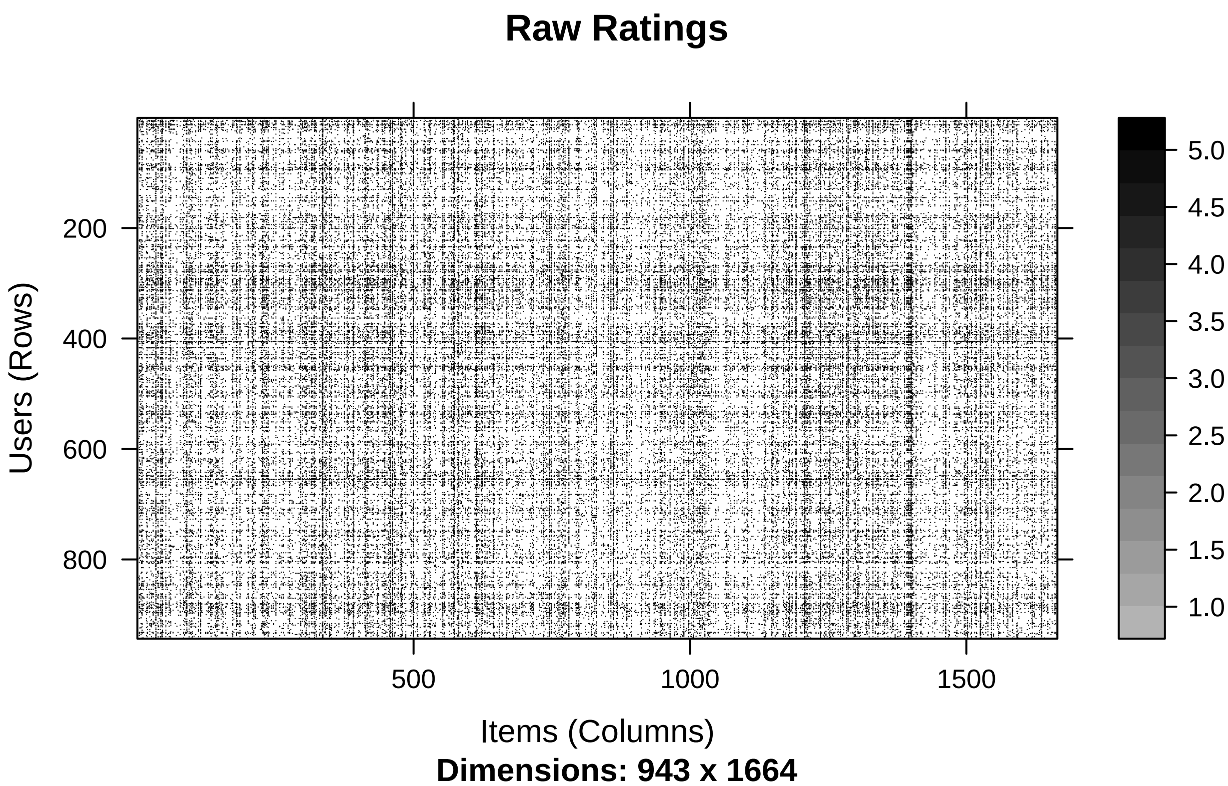
```r
dataList<- get.Data()
```

DATA PREPARATION AND PROCESSING

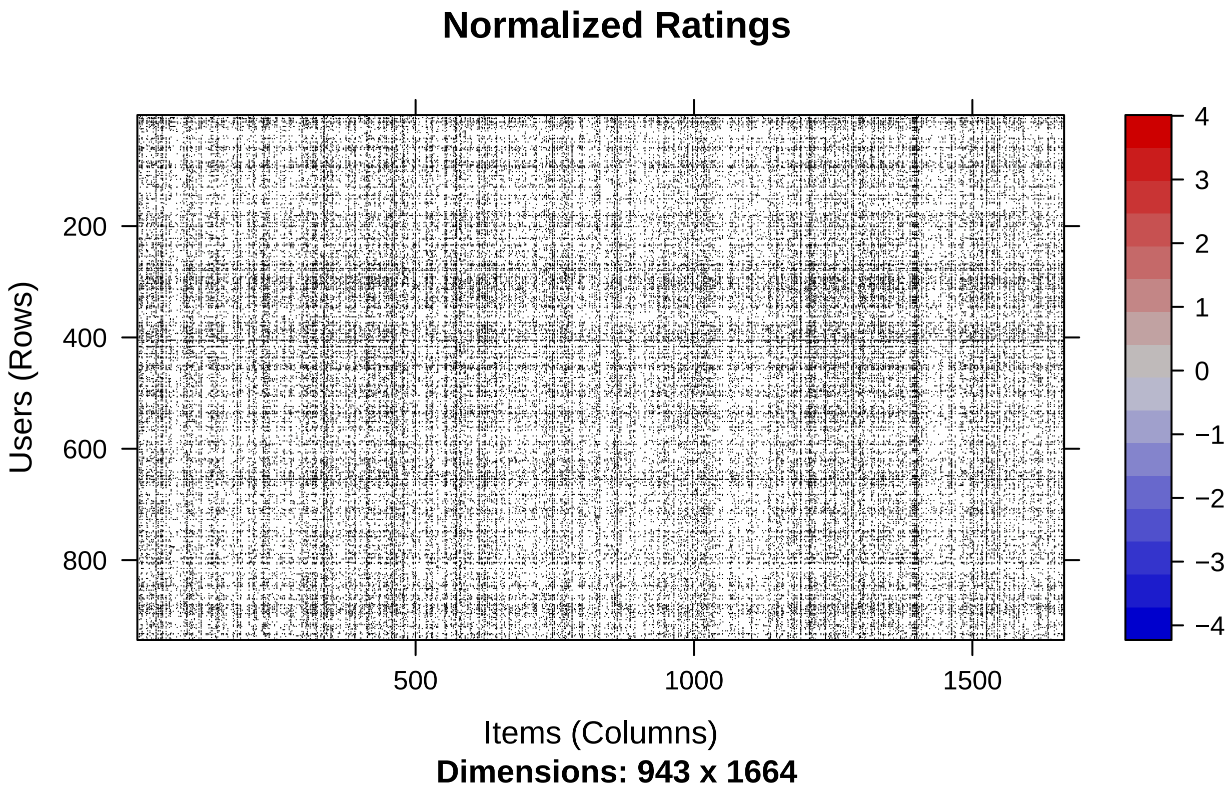
```r
ratings<- Pre.Process(dataList$ratings, dataList$movies)
```

NORMALIZATION,BINARIZATION, REAL RATING MATRIX

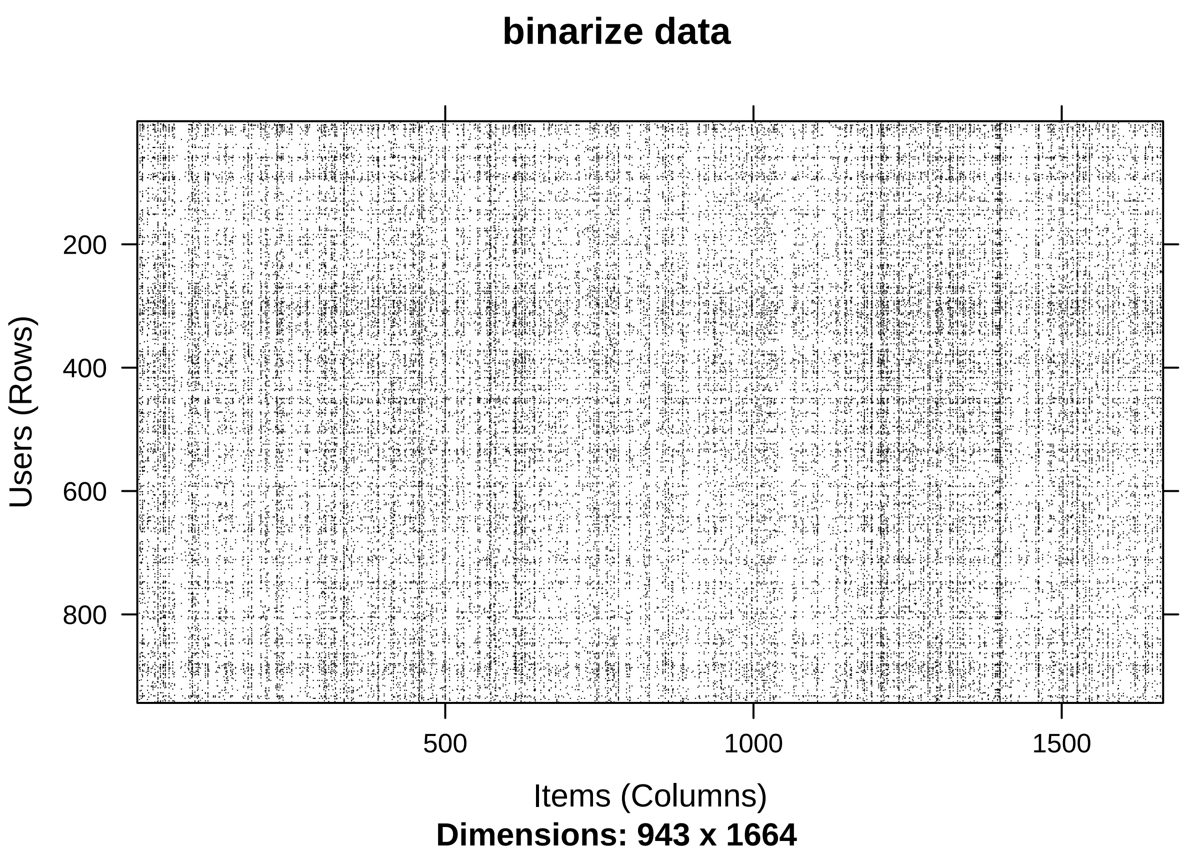
```r
library(ggplot2)
library(Hmisc)
movie.Rating.Mat<- Create.Rating.Matrix(ratings)
l=as(movie.Rating.Mat,"list")
m<-as(movie.Rating.Mat,"matrix")
rm<-normalize(movie.Rating.Mat)
image(movie.Rating.Mat,main="Raw Ratings")
```




```r
image(rm,main="Normalized Ratings")
```




```
bm<-binarize(movie.Rating.Mat,minRating=4)
image(bm,main="binarize data")
```



MODELS EVALUATION

```
evalList <- evaluateModels(movie.Rating.Mat)
```

```
## RANDOM run fold/sample [model time/prediction time]
##   1  [0.004sec/0.409sec]
## POPULAR run fold/sample [model time/prediction time]
##   1  [0.015sec/0.086sec]
## UBCF run fold/sample [model time/prediction time]
##   1  [0.009sec/1.435sec]
## IBCF run fold/sample [model time/prediction time]
##   1  [57.351sec/0.424sec]
## SVD run fold/sample [model time/prediction time]
##   1  [0.01sec/14.211sec]
```

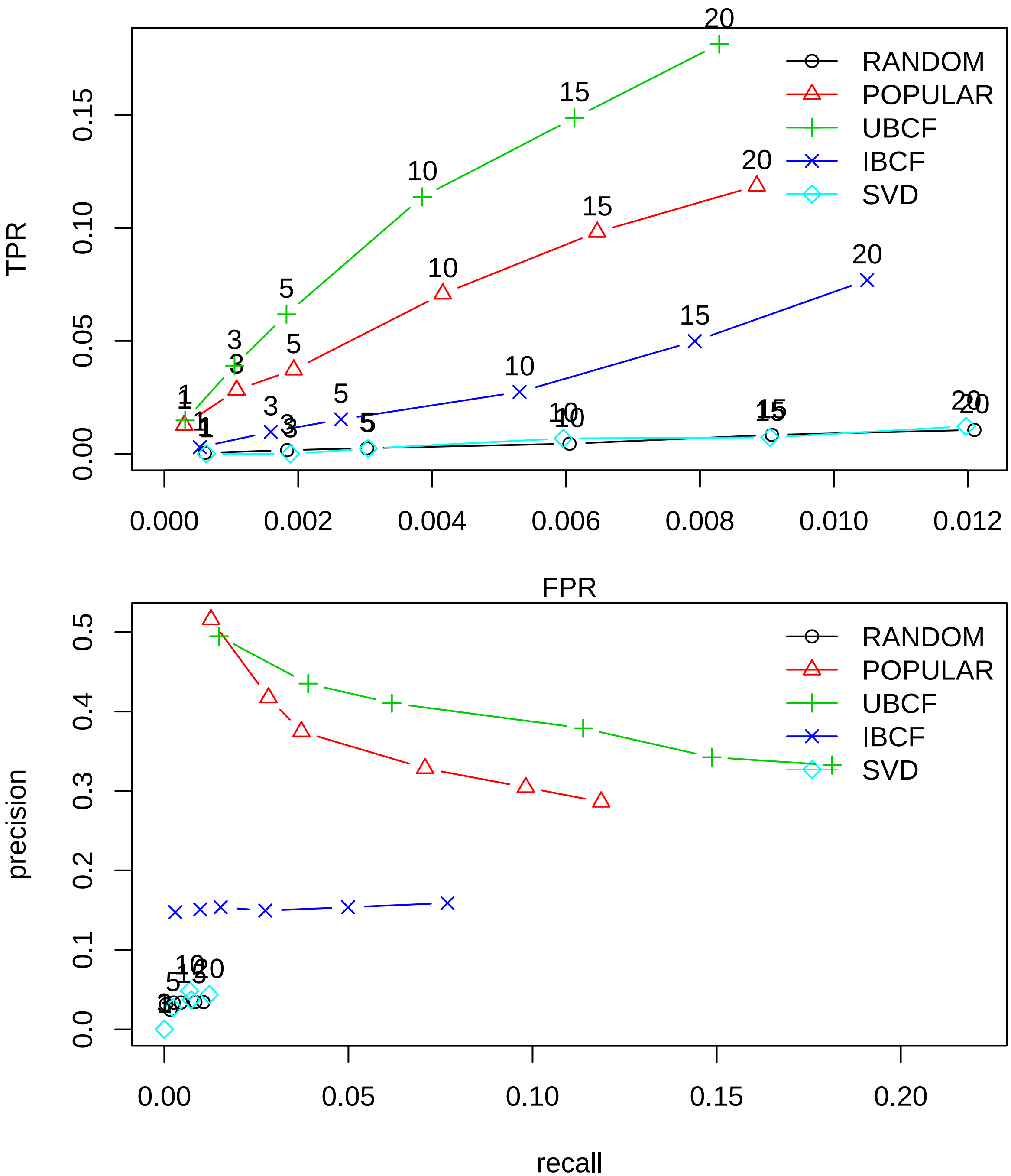
```
evalList
```

```
## List of evaluation results for 5 recommenders:
## Evaluation results for 1 folds/samples using method 'RANDOM'.
## Evaluation results for 1 folds/samples using method 'POPULAR'.
## Evaluation results for 1 folds/samples using method 'UBCF'.
## Evaluation results for 1 folds/samples using method 'IBCF'.
## Evaluation results for 1 folds/samples using method 'SVD'.
```

The plot for comparing “Random”, “Popular”, “UBCF”,IBCF based recommender algorithm is shown:

plot evaluation result

```r
graphs(evalList)
```



CLEARLY UBCF got the better metrics compare to the other methods

CONFUSION MATRIX FOR ALL METHODS

```
getConfusionMatrix(evalList[["UBCF"]])[[1]][,1:4]
```

```
##           TP         FP       FN       TN
## 1  0.4947368  0.5052632 57.29474 1595.705
## 3  1.3052632  1.6947368 56.48421 1594.516
## 5  2.0526316  2.9473684 55.73684 1593.263
## 10 3.7894737  6.2105263 54.00000 1590.000
## 15 5.1368421  9.8631579 52.65263 1586.347
## 20 6.6526316 13.3473684 51.13684 1582.863
```

```
getConfusionMatrix(evalList[["IBCF"]])[[1]][,1:4]
```

```
##           TP         FP       FN       TN
## 1  0.1473684  0.8526316 57.64211 1595.358
## 3  0.4526316  2.5473684 57.33684 1593.663
## 5  0.7684211  4.2315789 57.02105 1591.979
## 10 1.4947368  8.5052632 56.29474 1587.705
## 15 2.3052632 12.6947368 55.48421 1583.516
## 20 3.1789474 16.8210526 54.61053 1579.389
```

```
getConfusionMatrix(evalList[["POPULAR"]])[[1]][,1:4]
```

```
##           TP         FP       FN       TN
## 1  0.5157895  0.4842105 57.27368 1595.726
## 3  1.2526316  1.7473684 56.53684 1594.463
## 5  1.8736842  3.1263158 55.91579 1593.084
## 10 3.2842105  6.7157895 54.50526 1589.495
## 15 4.5684211 10.4315789 53.22105 1585.779
## 20 5.7263158 14.2736842 52.06316 1581.937
```

```
getConfusionMatrix(evalList[["RANDOM"]])[[1]][,1:4]
```

```
##            TP         FP       FN       TN
## 1  0.03157895  0.9684211 57.75789 1595.242
## 3  0.07368421  2.9263158 57.71579 1593.284
## 5  0.16842105  4.8315789 57.62105 1591.379
## 10 0.33684211  9.6631579 57.45263 1586.547
## 15 0.51578947 14.4842105 57.27368 1581.726
## 20 0.68421053 19.3157895 57.10526 1576.895
```

LET DO THE RECOMMENDATION BASED ON "UBCF"

```r
rec_model <- create.Model(movie.Rating.Mat, "UBCF")
userID <- 1
topN <- 5
rec(movie.Rating.Mat, rec_model, userID, topN)
```

```
## [[1]]
## [1] "Glory (1989)"               "Schindler's List (1993)"
## [3] "Close Shave, A (1995)"     "Casablanca (1942)"
## [5] "Leaving Las Vegas (1995)"
```

```r
userID<-2
topN<-10
rec(movie.Rating.Mat, rec_model, userID, topN)
```

```
## [[1]]
##  [1] "Lone Star (1996)"                "Boot, Das (1981)"
##  [3] "Dead Man Walking (1995)"         "Celluloid Closet, The (1995)"
##  [5] "Return of the Jedi (1983)"       "Casablanca (1942)"
##  [7] "Angels and Insects (1995)"       "Breaking the Waves (1996)"
##  [9] "Seven Years in Tibet (1997)"     "Welcome to the Dollhouse (1995)"
```

Let recommend the top 10 movies for users with ID between 5 and 15

```r
#for (userID in 5:15){
 # print("We recommend you those movies")
#  print(rec(movie.Rating.Mat,rec_model,userID,topN))
#}
rec_model2 <- create.Model(movie.Rating.Mat, "IBCF")
userID <- 1
topN <- 5
rec(movie.Rating.Mat, rec_model2, userID, topN)
```

```
## [[1]]
## [1] "2 Days in the Valley (1996)"  "American in Paris, An (1951)"
## [3] "Basquiat (1996)"              "Boys, Les (1997)"
## [5] "Brassed Off (1996)"
```

```r
userID<-2
topN<-10
rec(movie.Rating.Mat, rec_model2, userID, topN)
```

```
## [[1]]
##  [1] "12 Angry Men (1957)"          "2001: A Space Odyssey (1968)"
##  [3] "African Queen, The (1951)"    "Alien (1979)"
##  [5] "Aliens (1986)"                "Amadeus (1984)"
##  [7] "Apocalypse Now (1979)"        "Babe (1995)"
##  [9] "Back to the Future (1985)"    "Beautiful Thing (1996)"
```

```r
rec_model3 <- create.Model(movie.Rating.Mat, "POPULAR")
userID <- 1
topN <- 5
rec(movie.Rating.Mat, rec_model3, userID, topN)
```

```
## [[1]]
## [1] "Schindler's List (1993)"
## [2] "Titanic (1997)"
## [3] "L.A. Confidential (1997)"
## [4] "Casablanca (1942)"
## [5] "One Flew Over the Cuckoo's Nest (1975)"
```

```r
userID<-2
topN<-10
rec(movie.Rating.Mat, rec_model3, userID, topN)
```

```
## [[1]]
##  [1] "Raiders of the Lost Ark (1981)"   "Silence of the Lambs, The (1991)"
##  [3] "Schindler's List (1993)"          "Shawshank Redemption, The (1994)"
##  [5] "Empire Strikes Back, The (1980)"  "Return of the Jedi (1983)"
##  [7] "Usual Suspects, The (1995)"       "Casablanca (1942)"
##  [9] "Pulp Fiction (1994)"              "Princess Bride, The (1987)"
```

```r
rec_model4 <- create.Model(movie.Rating.Mat, "RANDOM")
userID <- 1
topN <- 5
rec(movie.Rating.Mat, rec_model4, userID, topN)
```

```
## [[1]]
## [1] "Cats Don't Dance (1997)"    "Nightwatch (1997)"
## [3] "Promesse, La (1996)"        "Tomorrow Never Dies (1997)"
## [5] "Father of the Bride (1950)"
```

```r
userID<-2
topN<-10
rec(movie.Rating.Mat, rec_model4, userID, topN)
```

```
## [[1]]
##  [1] "Rent-a-Kid (1995)"
##  [2] "Last Man Standing (1996)"
##  [3] "Gang Related (1997)"
##  [4] "Adventures of Priscilla, Queen of the Desert, The (1994)"
##  [5] "Palmetto (1998)"
##  [6] "Bio-Dome (1996)"
##  [7] "Wend Kuuni (God's Gift) (1982)"
##  [8] "Beautiful Thing (1996)"
##  [9] "Dark City (1998)"
## [10] "Prophecy, The (1995)"
```

RECOMMENDATION USING MANUAL SV DECOMPOSITION

```
library(reshape2)
library(dplyr)
library(recommenderlab)
library(NMF)
# Reload the part of the data
movies<-dataList$movies
rn<- normalize(movie.Rating.Mat)
rn <- as(rn, "matrix")
rn[is.na(rn)] <- 0
rn.svd <- svd(rn)

s<- cumsum(rn.svd$d) / sum(rn.svd$d)
k <- min(which(s >= 0.6))
```

WE CAN HAVE SMALLER OR GREATER DIMENSIONS REDUCTION DEPENDING ON THE VALUE of s above for s>=0.6 ,I found the optimal value to be 242,the 943*943 can be reduced to 242

```
k
```

```
## [1] 242
```

```
d <- diag(sqrt(rn.svd$d[1:k]))
u <- rn.svd$u[,1:k]
v <- rn.svd$v[1:k,]

w<- data.frame(u%*%d%*%v)
colnames(w) <- 1:943

df<- data.frame(matrix(NA, nrow = 943, ncol = 943))
for(i in 1:ncol(w)) {
  df[i,] <- colnames(w[i,order(w[i,],decreasing = TRUE)])
}

subdf<- df[5,1]
movieUser<- subset(ratings, userID == subdf)
df2 <- head(movieUser[order(-movieUser$rating),],5)
```

HERE WE RECOMMEND TO THE USER with userID ,the movie with rating 5 based on users who are most similar to userID ,their userIDs are to the left.

```
df2
```

```
##       userID                  movieID rating timestamp
## 51924    703       Men in Black (1997)      5 875242990
## 55297    703          Star Wars (1977)      5 875242813
## 58954    703      Jerry Maguire (1996)      5 875242787
## 65785    703            Twister (1996)      5 875242852
## 69668    703 Return of the Jedi (1983)      5 875242762
```

Here we recommend movies to the userID 5 with rating greater than 2 with the most similar users to userID 5

```
movieUser <- subset(ratings, userID == 5 & rating > 2)
df3<-head(movieUser)
df3
```

```
##      userID                         movieID rating timestamp
## 173       5                GoldenEye (1995)      3 875636053
## 440       5       From Dusk Till Dawn (1996)      4 875636198
## 1334      5                Toy Story (1995)      4 875635748
## 1396      5             Sudden Death (1995)      3 875635225
## 1483      5 Silence of the Lambs, The (1991)      3 875720691
## 1743      5           Aristocats, The (1970)      3 875721196
```