Valiveti Manikanta Bhuvanesh

19BCD7088

Experiment 9

AIM: Write a program to perform Movie recommendation system

Code:

```
library(ggplot2)
library(data.table)
library(reshape2)
library(recommenderlab)
movie_data <- read.csv("movies.csv",stringsAsFactors=FALSE)
rating_data <- read.csv("ratings.csv")
head(movie_data)
head(rating_data)
str(movie_data)
summary(movie_data)
str(rating_data)
summary(rating_data)
movie_genre <- as.data.frame(movie_data$genres, stringsAsFactors=FALSE)
head(movie_genre)
library(data.table)
movie_genre2 <- as.data.frame(tstrsplit(movie_genre[,1], '[|]',
                                        type.convert=TRUE),
                              stringsAsFactors=FALSE)
colnames(movie_genre2) <- c(1:10)
list_genre <- c("Action", "Adventure", "Animation", "Children",
                "Comedy", "Crime","Documentary", "Drama", "Fantasy",
                "Film-Noir", "Horror", "Musical", "Mystery","Romance",
                "Sci-Fi", "Thriller", "War", "Western")
genre_mat1 <- matrix(0,10330,18)
```

```
genre_mat1[1,] <- list_genre
colnames(genre_mat1) <- list_genre
for (index in 1:nrow(movie_genre2)) {
  for (col in 1:ncol(movie_genre2)) {
    gen_col = which(genre_mat1[1,] == movie_genre2[index,col])
    genre_mat1[index+1,gen_col] <- 1
  }
}

genre_mat2 <- as.data.frame(genre_mat1[-1,], stringsAsFactors=FALSE)
for (col in 1:ncol(genre_mat2)) {
  genre_mat2[,col] <- as.integer(genre_mat2[,col])
}
str(genre_mat2)
head(genre_mat2)
SearchMatrix <- cbind(movie_data[,1:2], genre_mat2[])
head(SearchMatrix)
ratingMatrix <- dcast(rating_data, userId~movieId, value.var = "rating",
na.rm=FALSE)
ratingMatrix <- as.matrix(ratingMatrix[,-1])
ratingMatrix <- as(ratingMatrix, "realRatingMatrix")
head(ratingMatrix)
head(rating_data)
recommendation_model <- recommenderRegistry$get_entries(dataType =
"realRatingMatrix")
names(recommendation_model)
lapply(recommendation_model, "[[", "description")
recommendation_model$IBCF_realRatingMatrix$parameters
```

```
similarity_mat <- similarity(ratingMatrix[1:4, ],method = "cosine",which = "users")

as.matrix(similarity_mat)

image(as.matrix(similarity_mat), main = "User's Similarities")

movie_similarity <- similarity(ratingMatrix[, 1:4], method ="cosine", which = "items")

as.matrix(movie_similarity)

image(as.matrix(movie_similarity), main = "Movies similarity")

rating_values <- as.vector(ratingMatrix@data)

unique(rating_values)

Table_of_Ratings <- table(rating_values)

Table_of_Ratings

library(ggplot2)

movie_views <- colCounts(ratingMatrix)

table_views <- data.frame(movie = names(movie_views),views = movie_views)

table_views <- table_views[order(table_views$views,decreasing = TRUE), ]

table_views$title <- NA

for (index in 1:10325){

  table_views[index,3] <- as.character(subset(movie_data,movie_data$movieId == table_views[index,1])$title)

}

table_views[1:6,]

ggplot(table_views[1:6, ], aes(x = title, y = views)) +

  geom_bar(stat="identity", fill = 'steelblue') +

  geom_text(aes(label=views), vjust=-0.3, size=3.5) +

  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +

  ggtitle("Total Views of the Top Films")

image(ratingMatrix[1:20, 1:25], axes = FALSE, main = "Heatmap of the first 25 rows and 25 columns")
```

```
movie_ratings <- ratingMatrix[rowCounts(ratingMatrix) >
50,colCounts(ratingMatrix) > 50]

movie_ratings

minimum_movies<- quantile(rowCounts(movie_ratings), 0.98)

minimum_users <- quantile(colCounts(movie_ratings), 0.98)

image(movie_ratings[rowCounts(movie_ratings) >
minimum_movies,colCounts(movie_ratings) > minimum_users],main =
"Heatmap of the top users and movies")

average_ratings <- rowMeans(movie_ratings)

qplot(average_ratings, fill=I("steelblue"), col=I("red")) +ggtitle("Distribution of
the average rating per user")

normalized_ratings <- normalize(movie_ratings)

sum(rowMeans(normalized_ratings) > 0.00001)

image(normalized_ratings[rowCounts(normalized_ratings) >
minimum_movies,colCounts(normalized_ratings) > minimum_users],main =
"Normalized Ratings of the Top Users")

binary_minimum_movies <- quantile(rowCounts(movie_ratings), 0.95)

binary_minimum_users <- quantile(colCounts(movie_ratings), 0.95)

good_rated_films <- binarize(movie_ratings, minRating = 3)

image(good_rated_films[rowCounts(movie_ratings) >
binary_minimum_movies,colCounts(movie_ratings) >
binary_minimum_users],main = "Heatmap of the top users and movies")

sampled_data<- sample(x = c(TRUE, FALSE),size =
nrow(movie_ratings),replace = TRUE,prob = c(0.8, 0.2))

training_data <- movie_ratings[sampled_data, ]

testing_data <- movie_ratings[!sampled_data, ]




library(recommenderlab)
```

```
recommendation_system <- recommenderRegistry$get_entries(dataType
="realRatingMatrix")

recommendation_system$IBCF_realRatingMatrix$parameters

recommen_model <- Recommender(data = training_data,method =
"IBCF",parameter = list(k = 30))

recommen_model

class(recommen_model)

model_info <- getModel(recommen_model)

class(model_info$sim)

dim(model_info$sim)

top_items <- 20

image(model_info$sim[1:top_items, 1:top_items],main = "Heatmap of the first
rows and columns")

sum_rows <- rowSums(model_info$sim > 0)

table(sum_rows)

sum_cols <- colSums(model_info$sim > 0)

qplot(sum_cols, fill=I("steelblue"), col=I("red"))+ ggtitle("Distribution of the
column count")

top_recommendations <- 10

predicted_recommendations <- predict(object = recommen_model,

                                     newdata = testing_data,

                                     n = top_recommendations)

predicted_recommendations

user1 <- predicted_recommendations@items[[1]]

movies_user1 <- predicted_recommendations@itemLabels[user1]

movies_user2 <- movies_user1

for (index in 1:10){

  movies_user2[index] <- as.character(subset(movie_data,
```

movie_data$movieId ==
movies_user1[index])$title)

}

movies_user2

recommendation_matrix <- sapply(predicted_recommendations@items,

function(x){ as.integer(colnames(movie_ratings)[x]) }) #
matrix with the recommendations for each user

recommendation_matrix[,1:5]

OUTPUT:

```
> movie_data <- read.csv("movies.csv",stringsAsFactors=FALSE)
> rating_data <- read.csv("ratings.csv")
> head(movie_data)
  movieId                               title                                        genres
1       1                    Toy Story (1995) Adventure|Animation|Children|Comedy|Fantasy
2       2                      Jumanji (1995)                  Adventure|Children|Fantasy
3       3             Grumpier Old Men (1995)                              Comedy|Romance
4       4            Waiting to Exhale (1995)                        Comedy|Drama|Romance
5       5 Father of the Bride Part II (1995)                                      Comedy
6       6                         Heat (1995)                       Action|Crime|Thriller
> head(rating_data)
  userId movieId rating  timestamp
1      1      16    4.0 1217897793
2      1      24    1.5 1217895807
3      1      32    4.0 1217896246
4      1      47    4.0 1217896556
5      1      50    4.0 1217896523
6      1     110    4.0 1217896150
> str(movie_data)
'data.frame':   10329 obs. of  3 variables:
 $ movieId: int  1 2 3 4 5 6 7 8 9 10 ...
 $ title  : chr  "Toy Story (1995)" "Jumanji (1995)" "Grumpier Old Men (1995)" "Waiting to Exhale
995)" ...
 $ genres : chr  "Adventure|Animation|Children|Comedy|Fantasy" "Adventure|Children|Fantasy" "Come
Romance" "Comedy|Drama|Romance" ...
> summary(movie_data)
    movieId          title              genres
 Min.   :     1   Length:10329       Length:10329
 1st Qu.:  3240   Class :character   Class :character
 Median :  7088   Mode  :character   Mode  :character
 Mean   : 31924
 3rd Qu.: 59900
 Max.   :149532
> str(rating_data)
'data.frame':   105339 obs. of  4 variables:
 $ userId   : int  1 1 1 1 1 1 1 1 1 1 ...
 $ movieId  : int  16 24 32 47 50 110 150 161 165 204 ...
 $ rating   : num  4 1.5 4 4 4 4 3 4 3 0.5 ...
 $ timestamp: int  1217897793 1217895807 1217896246 1217896556 1217896523 1217896150 1217895940 1
897864 1217897135 1217895786 ...
> summary(rating_data)
     userId         movieId          rating       timestamp
 Min.   :  1.0   Min.   :     1   Min.   :0.500   Min.   :8.286e+08
 1st Qu.:192.0   1st Qu.:  1073   1st Qu.:3.000   1st Qu.:9.711e+08
 Median :383.0   Median :  2497   Median :3.500   Median :1.115e+09
 Mean   :364.9   Mean   : 13381   Mean   :3.517   Mean   :1.130e+09
 3rd Qu.:557.0   3rd Qu.:  5991   3rd Qu.:4.000   3rd Qu.:1.275e+09
 Max.   :668.0   Max.   :149532   Max.   :5.000   Max.   :1.452e+09
```

```
> movie_genre <- as.data.frame(movie_data$genres, stringsAsFactors=FALSE)
> head(movie_genre)
                            movie_data$genres
1 Adventure|Animation|Children|Comedy|Fantasy
2                  Adventure|Children|Fantasy
3                              Comedy|Romance
4                        Comedy|Drama|Romance
5                                      Comedy
6                       Action|Crime|Thriller
> library(data.table)
> movie_genre2 <- as.data.frame(tstrsplit(movie_genre[,1], '[|]',
+                                         type.convert=TRUE),
+                               stringsAsFactors=FALSE)
> colnames(movie_genre2) <- c(1:10)
> list_genre <- c("Action", "Adventure", "Animation", "Children",
+                 "Comedy", "Crime","Documentary", "Drama", "Fantasy",
+                 "Film-Noir", "Horror", "Musical", "Mystery","Romance",
+                 "Sci-Fi", "Thriller", "War", "Western")
> genre_mat1 <- matrix(0,10330,18)
> genre_mat1[1,] <- list_genre
> colnames(genre_mat1) <- list_genre
> for (index in 1:nrow(movie_genre2)) {
+   for (col in 1:ncol(movie_genre2)) {
+     gen_col = which(genre_mat1[1,] == movie_genre2[index,col])
+     genre_mat1[index+1,gen_col] <- 1
+   }
+ }
>
> genre_mat2 <- as.data.frame(genre_mat1[-1,], stringsAsFactors=FALSE)
> for (col in 1:ncol(genre_mat2)) {
+   genre_mat2[,col] <- as.integer(genre_mat2[,col])
+ }
> str(genre_mat2)
'data.frame':   10329 obs. of  18 variables:
 $ Action     : int  0 0 0 0 0 1 0 0 1 1 ...
 $ Adventure  : int  1 1 0 0 0 0 0 1 0 1 ...
 $ Animation  : int  1 0 0 0 0 0 0 0 0 0 ...
 $ Children   : int  1 1 0 0 0 0 0 1 0 0 ...
 $ Comedy     : int  1 0 1 1 1 0 1 0 0 0 ...
 $ Crime      : int  0 0 0 0 0 1 0 0 0 0 ...
 $ Documentary: int  0 0 0 0 0 0 0 0 0 0 ...
 $ Drama      : int  0 0 0 1 0 0 0 0 0 0 ...
 $ Fantasy    : int  1 1 0 0 0 0 0 0 0 0 ...
 $ Film-Noir  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Horror     : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Musical    : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Mystery    : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Romance    : int  0 0 1 1 0 0 1 0 0 0 ...
```

```
> head(genre_mat2)
  Action Adventure Animation Children Comedy Crime Documentary Drama Fantasy Film-Noir Horror
1      0         1         1        1      1     0           0     0       1         0      0
2      0         1         0        1      0     0           0     0       1         0      0
3      0         0         0        0      1     0           0     0       0         0      0
4      0         0         0        0      1     0           0     1       0         0      0
5      0         0         0        0      1     0           0     0       0         0      0
6      1         0         0        0      0     1           0     0       0         0      0
  Musical Mystery Romance Sci-Fi Thriller War Western
1       0       0       0      0        0   0       0
2       0       0       0      0        0   0       0
3       0       0       1      0        0   0       0
4       0       0       1      0        0   0       0
5       0       0       0      0        0   0       0
6       0       0       0      0        1   0       0
> SearchMatrix <- cbind(movie_data[,1:2], genre_mat2[])
> head(SearchMatrix)
  movieId                              title Action Adventure Animation Children Comedy Crime
1       1                   Toy Story (1995)      0         1         1        1      1     0
2       2                     Jumanji (1995)      0         1         0        1      0     0
3       3            Grumpier Old Men (1995)      0         0         0        0      1     0
4       4           Waiting to Exhale (1995)      0         0         0        0      1     0
5       5 Father of the Bride Part II (1995)      0         0         0        0      1     0
6       6                          Heat (1995)      1         0         0        0      0     1
  Documentary Drama Fantasy Film-Noir Horror Musical Mystery Romance Sci-Fi Thriller War Western
1           0     0       1         0      0       0       0       0      0        0   0       0
2           0     0       1         0      0       0       0       0      0        0   0       0
3           0     0       0         0      0       0       0       1      0        0   0       0
4           0     1       0         0      0       0       0       1      0        0   0       0
5           0     0       0         0      0       0       0       0      0        0   0       0
6           0     0       0         0      0       0       0       0      0        1   0       0
> ratingMatrix <- dcast(rating_data, userId~movieId, value.var = "rating", na.rm=FALSE)
> ratingMatrix <- as.matrix(ratingMatrix[,-1])
> ratingMatrix <- as(ratingMatrix, "realRatingMatrix")
> head(ratingMatrix)
6 x 10325 rating matrix of class ‘realRatingMatrix’ with 468 ratings.
> head(rating_data)
  userId movieId rating  timestamp
1      1      16    4.0 1217897793
2      1      24    1.5 1217895807
3      1      32    4.0 1217896246
4      1      47    4.0 1217896556
5      1      50    4.0 1217896523
6      1     110    4.0 1217896150
> recommendation_model <- recommenderRegistry$get_entries(dataType = "realRatingMatrix")
> names(recommendation_model)
 [1] "HYBRID_realRatingMatrix"       "ALS_realRatingMatrix"
 [3] "ALS_implicit_realRatingMatrix" "IBCF_realRatingMatrix"
```

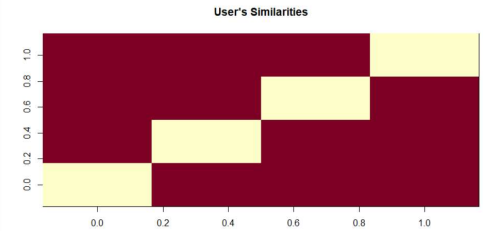
```
> recommendation_model <- recommenderRegistry$get_entries(dataType = "realRatingMatrix")
> names(recommendation_model)
 [1] "HYBRID_realRatingMatrix"       "ALS_realRatingMatrix"
 [3] "ALS_implicit_realRatingMatrix" "IBCF_realRatingMatrix"
 [5] "LIBMF_realRatingMatrix"        "POPULAR_realRatingMatrix"
 [7] "RANDOM_realRatingMatrix"       "RERECOMMEND_realRatingMatrix"
 [9] "SVD_realRatingMatrix"          "SVDF_realRatingMatrix"
[11] "UBCF_realRatingMatrix"
> lapply(recommendation_model, "[[", "description")
$HYBRID_realRatingMatrix
[1] "Hybrid recommender that aggegates several recommendation strategies using weighted averages."

$ALS_realRatingMatrix
[1] "Recommender for explicit ratings based on latent factors, calculated by alternating least squar
es algorithm."

$ALS_implicit_realRatingMatrix
[1] "Recommender for implicit data based on latent factors, calculated by alternating least squares
 algorithm."

$IBCF_realRatingMatrix
[1] "Recommender based on item-based collaborative filtering."

$LIBMF_realRatingMatrix
[1] "Matrix factorization with LIBMF via package recosystem (https://cran.r-project.org/web/package
s/recosystem/vignettes/introduction.html)."

$POPULAR_realRatingMatrix
[1] "Recommender based on item popularity."

$RANDOM_realRatingMatrix
[1] "Produce random recommendations (real ratings)."

$RERECOMMEND_realRatingMatrix
[1] "Re-recommends highly rated items (real ratings)."

$SVD_realRatingMatrix
[1] "Recommender based on SVD approximation with column-mean imputation."

$SVDF_realRatingMatrix
[1] "Recommender based on Funk SVD with gradient descend (https://sifter.org/~simon/journal/2006121
1.html)."

$UBCF_realRatingMatrix
```

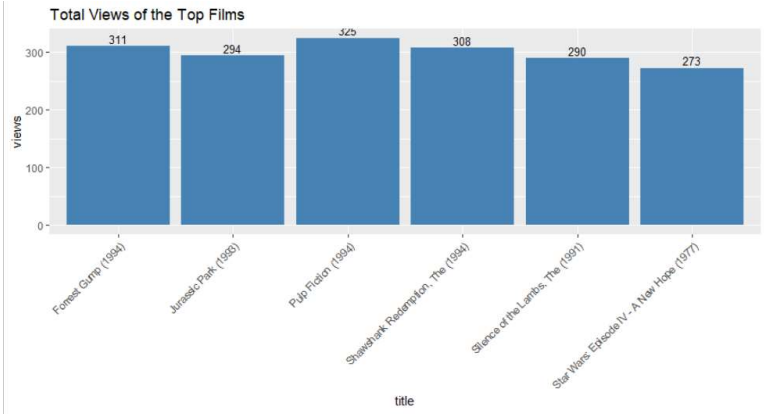
```
> recommendation_model$IBCF_realRatingMatrix$parameters
$k
[1] 30

$method
[1] "Cosine"

$normalize
[1] "center"

$normalize_sim_matrix
[1] FALSE

$alpha
[1] 0.5

$na_as_zero
[1] FALSE

> similarity_mat <- similarity(ratingMatrix[1:4, ],method = "cosine",which = "users")
> as.matrix(similarity_mat)
          1         2         3         4
1 0.0000000 0.9760860 0.9641723 0.9914398
2 0.9760860 0.0000000 0.9925732 0.9374253
3 0.9641723 0.9925732 0.0000000 0.9888968
4 0.9914398 0.9374253 0.9888968 0.0000000
> image(as.matrix(similarity_mat), main = "User's Similarities")
> movie_similarity <- similarity(ratingMatrix[, 1:4], method ="cosine", which = "items")
> as.matrix(movie_similarity)
          1         2         3         4
1 0.0000000 0.9669732 0.9559341 0.9101276
2 0.9669732 0.0000000 0.9658757 0.9412416
3 0.9559341 0.9658757 0.0000000 0.9864877
4 0.9101276 0.9412416 0.9864877 0.0000000
> image(as.matrix(movie_similarity), main = "Movies similarity")
> rating_values <- as.vector(ratingMatrix@data)
> unique(rating_values)
 [1] 0.0 5.0 4.0 3.0 4.5 1.5 2.0 3.5 1.0 2.5 0.5
> Table_of_Ratings <- table(rating_values)
> Table_of_Ratings
rating_values
      0     0.5       1     1.5       2     2.5       3     3.5       4     4.5       5
6791761    1198    3258    1567    7943    5484   21729   12237   28880    8187   14856
> library(ggplot2)
> movie_views <- colCounts(ratingMatrix)
> table_views <- data.frame(movie = names(movie_views),views = movie_views)
> table_views <- table_views[order(table_views$views,decreasing = TRUE), ]
> table_views$title <- NA
```

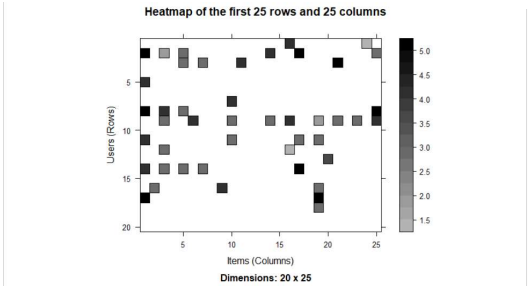
```
> ggplot(table_views[1:6, ], aes(x = title, y = views)) +
+   geom_bar(stat="identity", fill = 'steelblue') +
+   geom_text(aes(label=views), vjust=-0.3, size=3.5) +
+   theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
+   ggtitle("Total Views of the Top Films")
> image(ratingMatrix[1:20, 1:25], axes = FALSE, main = "Heatmap of the first 25 rows and 25 column
s")
> movie_ratings <- ratingMatrix[rowCounts(ratingMatrix) > 50,colCounts(ratingMatrix) > 50]
> movie_ratings
420 x 447 rating matrix of class 'realRatingMatrix' with 38341 ratings.
> minimum_movies<- quantile(rowCounts(movie_ratings), 0.98)
> minimum_users <- quantile(colCounts(movie_ratings), 0.98)
> image(movie_ratings[rowCounts(movie_ratings) > minimum_movies,colCounts(movie_ratings) > minimum_u
sers],main = "Heatmap of the top users and movies")
> average_ratings <- rowMeans(movie_ratings)
> qplot(average_ratings, fill=I("steelblue"), col=I("red")) +ggtitle("Distribution of the average ra
ting per user")
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
> normalized_ratings <- normalize(movie_ratings)
> sum(rowMeans(normalized_ratings) > 0.00001)
[1] 0
> image(normalized_ratings[rowCounts(normalized_ratings) > minimum_movies,colCounts(normalized_ratin
gs) > minimum_users],main = "Normalized Ratings of the Top Users")
> binary_minimum_movies <- quantile(rowCounts(movie_ratings), 0.95)
> binary_minimum_users <- quantile(colCounts(movie_ratings), 0.95)
> good_rated_films <- binarize(movie_ratings, minRating = 3)
> image(good_rated_films[rowCounts(movie_ratings) > binary_minimum_movies,colCounts(movie_ratings) >
 binary_minimum_users],main = "Heatmap of the top users and movies")
> sampled_data<- sample(x = c(TRUE, FALSE),size = nrow(movie_ratings),replace = TRUE,prob = c(0.8,
 0.2))
> training_data <- movie_ratings[sampled_data, ]
> testing_data <- movie_ratings[!sampled_data, ]
>
>
>
>
> library(recommenderlab)
> recommendation_system <- recommenderRegistry$get_entries(dataType ="realRatingMatrix")
> recommendation_system$IBCF_realRatingMatrix$parameters
$k
[1] 30

$method
[1] "Cosine"

$normalize
[1] "center"
```

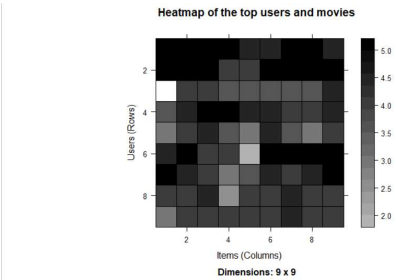
```
> recommen_model <- Recommender(data = training_data,method = "IBCF",parameter = list(k = 30))
> recommen_model
Recommender of type 'IBCF' for 'realRatingMatrix'
learned using 352 users.
> class(recommen_model)
[1] "Recommender"
attr(,"package")
[1] "recommenderlab"
> model_info <- getModel(recommen_model)
> class(model_info$sim)
[1] "dgCMatrix"
attr(,"package")
[1] "Matrix"
> dim(model_info$sim)
[1] 447 447
> top_items <- 20
> image(model_info$sim[1:top_items, 1:top_items],main = "Heatmap of the first rows and columns")
> sum_rows <- rowSums(model_info$sim > 0)
> table(sum_rows)
sum_rows
 30
447
> sum_cols <- colSums(model_info$sim > 0)
> qplot(sum_cols, fill=I("steelblue"), col=I("red"))+ ggtitle("Distribution of the column count")
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
> top_recommendations <- 10
> predicted_recommendations <- predict(object = recommen_model,
+                                      newdata = testing_data,
+                                      n = top_recommendations)
> predicted_recommendations
Recommendations as 'topNList' with n = 10 for 68 users.
> user1 <- predicted_recommendations@items[[1]]
> movies_user1 <- predicted_recommendations@itemLabels[user1]
> movies_user2 <- movies_user1
> for (index in 1:10){
+   movies_user2[index] <- as.character(subset(movie_data,
+                                              movie_data$movieId == movies_user1[index])$title)
+ }
> movies_user2
 [1] "Heat (1995)"
 [2] "Leaving Las Vegas (1995)"
 [3] "In the Line of Fire (1993)"
 [4] "Schindler's List (1993)"
 [5] "Searching for Bobby Fischer (1993)"
 [6] "Blade Runner (1982)"
 [7] "Silence of the Lambs, The (1991)"
```

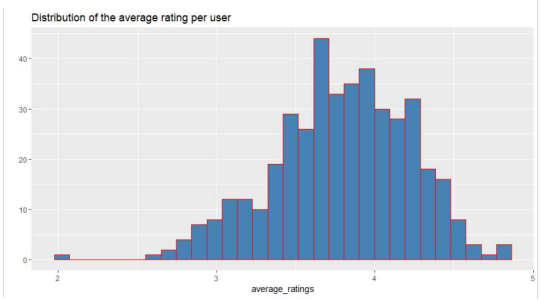
```
> recommendation_matrix <- sapply(predicted_recommendations@items,
+                                 function(x){ as.integer(colnames(movie_ratings)[x]) })
th the recommendations for each user
>
> recommendation_matrix[,1:5]
      [,1] [,2]  [,3]  [,4]  [,5]
 [1,]    6   17  1201   253    47
 [2,]   25   21 58559  1183   318
 [3,]  474   39   913  5445  1204
 [4,]  527  265   903  4995  2571
 [5,]  529  288  1193 48394  3578
 [6,]  541  368   318   596  1704
 [7,]  593  508  3578  1997 60069
 [8,]  750  590  1997 68954   161
 [9,]  908  661   551  2278  2028
[10,]  913  708  2329  1704  1250
>
```

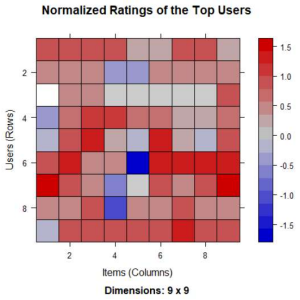


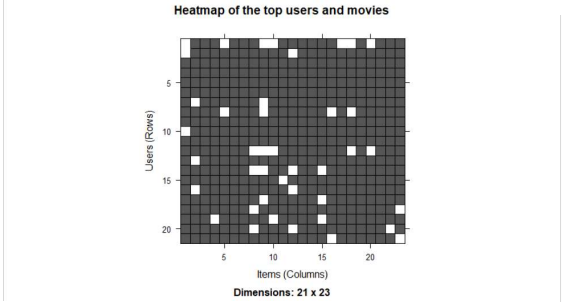

Total Views of the Top Films
311
294
325
308
290
273
Views
0
100
200
300
Forrest Gump (1994)
Jurassic Park (1993)
Pulp Fiction (1994)
Shawshank Redemption, The (1994)
Silence of the Lambs, The (1991)
Star Wars: Episode IV - A New Hope (1977)
title

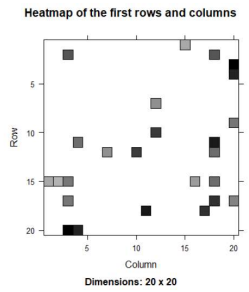

Heatmap of the first 25 rows and 25 columns
Users (Rows)
Items (Columns)
Dimensions: 20 x 25

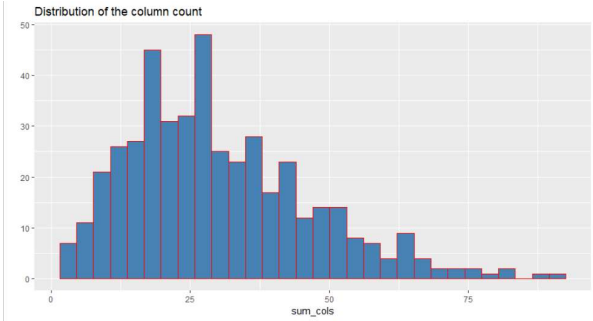

Heatmap of the top users and movies
Users (Rows)
Items (Columns)
Dimensions: 9 x 9


Distribution of the average rating per user
average_ratings


Normalized Ratings of the Top Users
Users (Rows)
Items (Columns)
Dimensions: 9 x 9


Heatmap of the top users and movies
Users (Rows)
Items (Columns)
Dimensions: 21 x 23


Heatmap of the first rows and columns
Row
Column
Dimensions: 20 x 20


Distribution of the column count
sum_cols