ANA 515 Assignment 4 Data Analytics Project

Shreyesh Shetty

2022-10-14

```r
movie_genre <- as.data.frame(movie_data$genres, stringsAsFactors=FALSE)
library(data.table)
movie_genre2 <- as.data.frame(tstrsplit(movie_genre[,1], '[|]',
                                   type.convert=TRUE),
                         stringsAsFactors=FALSE) #DataFlair
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]) #Author DataFlair
    genre_mat1[index+1,gen_col] <- 1
}
}
genre_mat2 <- as.data.frame(genre_mat1[-1,], stringsAsFactors=FALSE) #remove first row, which was the g
for (col in 1:ncol(genre_mat2)) {
  genre_mat2[,col] <- as.integer(genre_mat2[,col]) #convert from characters to integers
}
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 … $ Sci-Fi : int 0 0 0 0 0 0 0 0 0 0 … $ Thriller : int 0 0 0 0 0 1 0 0 0 1 … $ War : int 0 0 0 0 0 0 0 0 0 0 … $ Western : int 0 0 0 0 0 0 0 0 0 0 …

```r
SearchMatrix <- cbind(movie_data[,1:2], genre_mat2[])
head(SearchMatrix)
```

movieId title Action Adventure Animation 1 1 Toy Story (1995) 0 1 1 2 2 Jumanji (1995) 0 1 0 3 3 Grumpier Old Men (1995) 0 0 0 4 4 Waiting to Exhale (1995) 0 0 0 5 5 Father of the Bride Part II (1995) 0 0 0 6 6 Heat (1995) 1 0 0 Children Comedy Crime Documentary Drama Fantasy Film-Noir Horror Musical 1 1 1 0 0 0 1 0 0 0 2 1 0 0 0 0 1 0 0 0 3 0 1 0 0 0 0 0 0 0 4 0 1 0 0 1 0 0 0 0 5 0 1 0 0 0 0 0 0 0 6 0 0 1 0 0 0 0 0 0 Mystery Romance Sci-Fi Thriller War Western 1 0 0 0 0 0 0 2 0 0 0 0 0 0 3 0 1 0 0 0 0 4 0 1 0 0 0 0 5 0 0 0 0 0 0 6 0 0 0 1 0 0

```r
ratingMatrix <- dcast(rating_data, userId~movieId, value.var = "rating", na.rm=FALSE)
ratingMatrix <- as.matrix(ratingMatrix[,-1]) #remove userIds
#Convert rating matrix into a recommenderlab sparse matrix
ratingMatrix <- as(ratingMatrix, "realRatingMatrix")
ratingMatrix
```

668 x 10325 rating matrix of class ‘realRatingMatrix’ with 105339 ratings.

```r
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”

```r
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 squares 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/packages/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/20061211.html).”

$UBCF_realRatingMatrix [1] “Recommender based on user-based collaborative filtering.”

```r
recommendation_model$IBCF_realRatingMatrix$parameters
```

$k [1] 30

$method [1] “cosine”

$normalize [1] “center”

$normalize_sim_matrix [1] FALSE

$alpha [1] 0.5

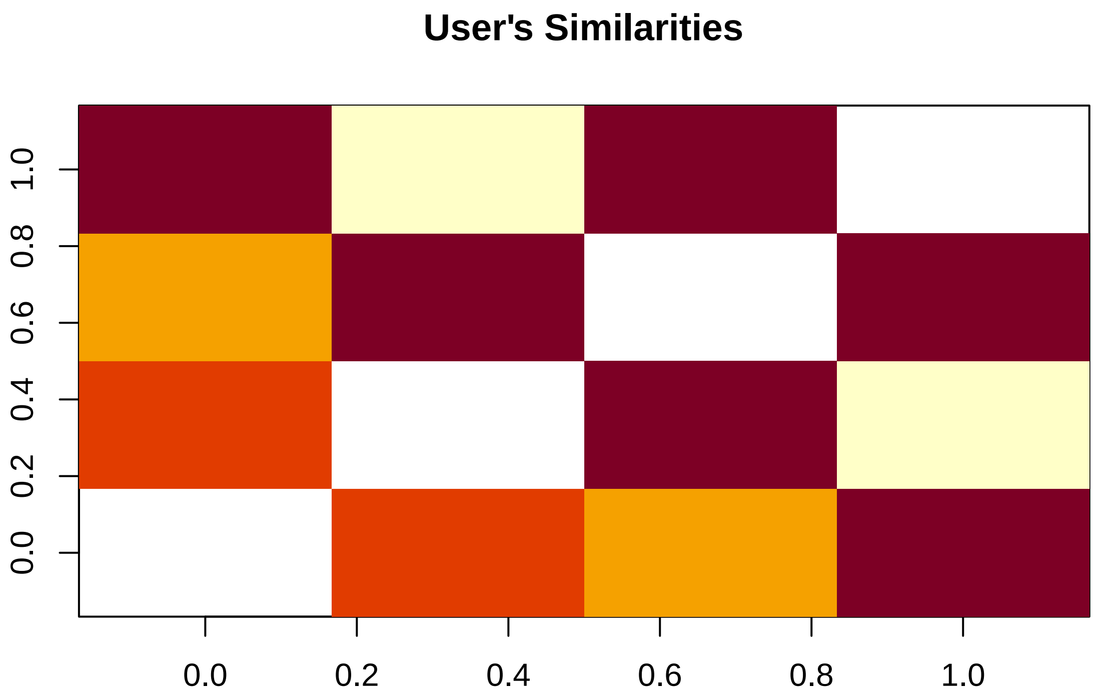
$na_as_zero [1] FALSE

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

```
      1          2          3          4
```

1 NA 0.9880430 0.9820862 0.9957199 2 0.9880430 NA 0.9962866 0.9687126 3 0.9820862 0.9962866 NA 0.9944484 4 0.9957199 0.9687126 0.9944484 NA

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

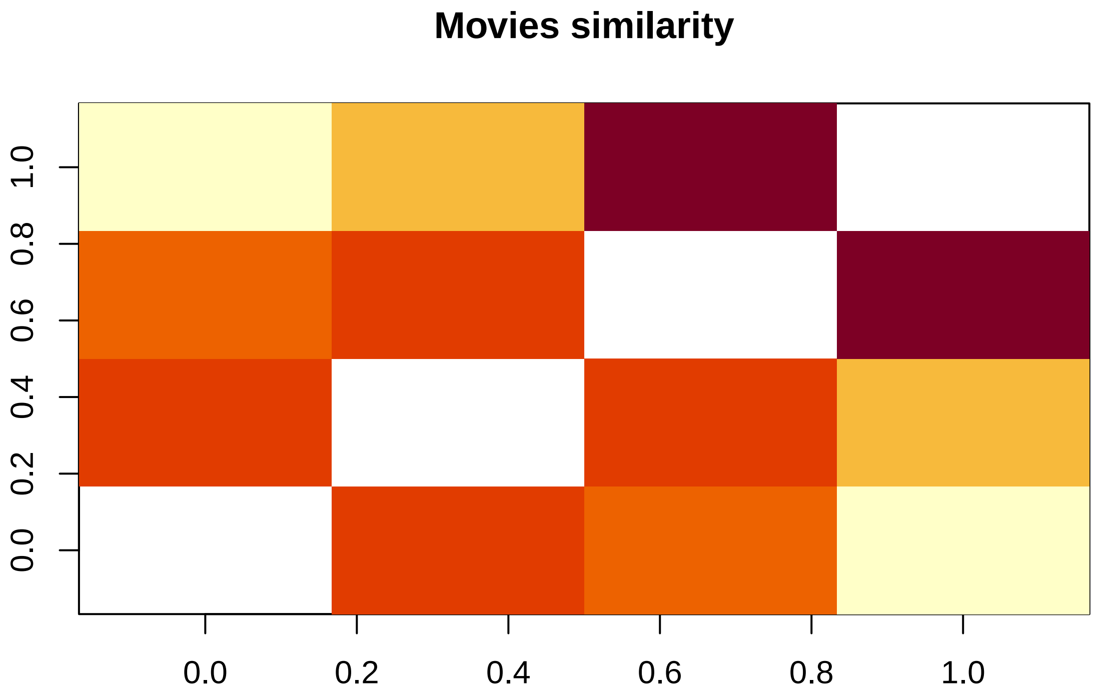


```
movie_similarity <- similarity(ratingMatrix[, 1:4], method =
                                 "cosine", which = "items")
as.matrix(movie_similarity)
```

```
      1          2          3          4
```

1 NA 0.9834866 0.9779671 0.9550638 2 0.9834866 NA 0.9829378 0.9706208 3 0.9779671 0.9829378 NA 0.9932438 4 0.9550638 0.9706208 0.9932438 NA

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



```
rating_values <- as.vector(ratingMatrix@data)
unique(rating_values) # extracting unique ratings
```

[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) # creating a count of movie ratings
Table_of_Ratings
```

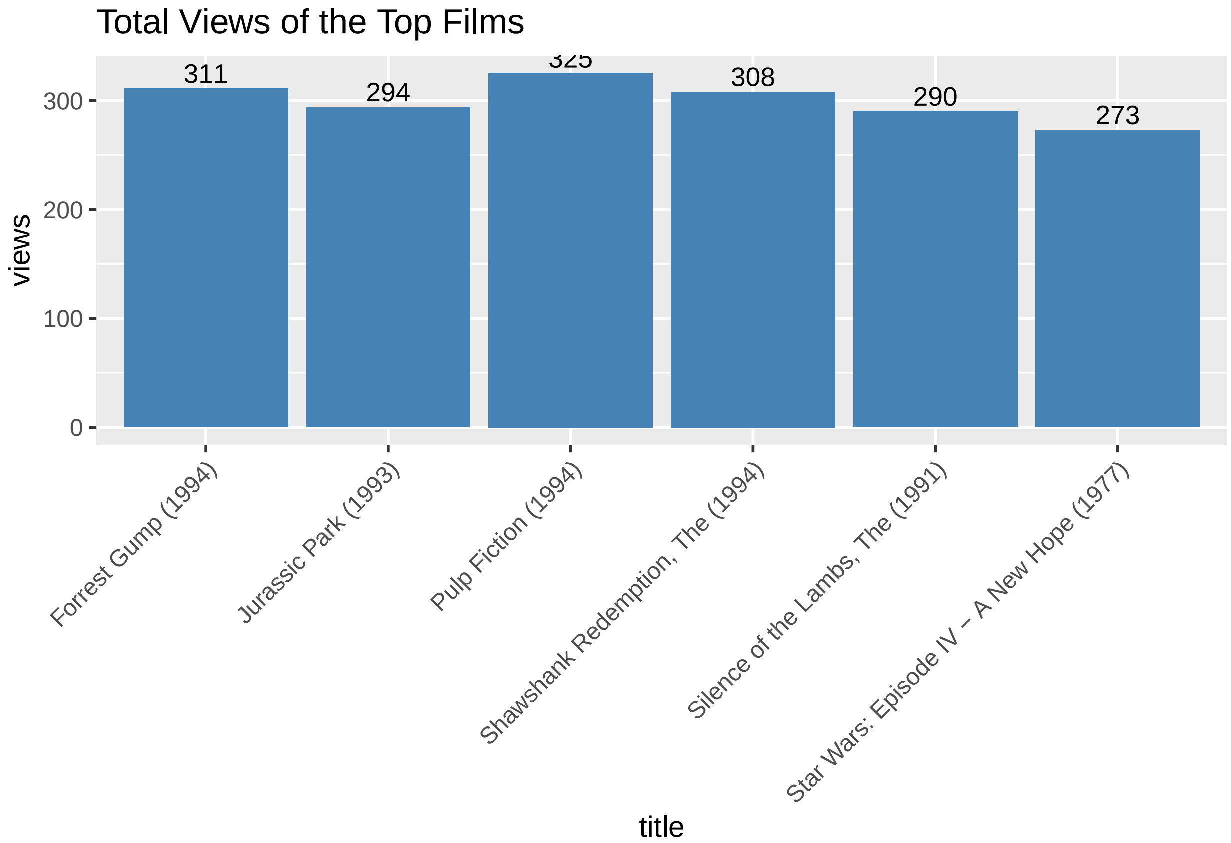
rating_values 0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 6791761 1198 3258 1567 7943 5484 21729 12237 28880 8187 5 14856

```
library(ggplot2)
movie_views <- colCounts(ratingMatrix) # count views for each movie
table_views <- data.frame(movie = names(movie_views),
                          views = movie_views) # create dataframe of views
table_views <- table_views[order(table_views$views,
                                 decreasing = TRUE), ] # sort by number of views
table_views$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,]
```

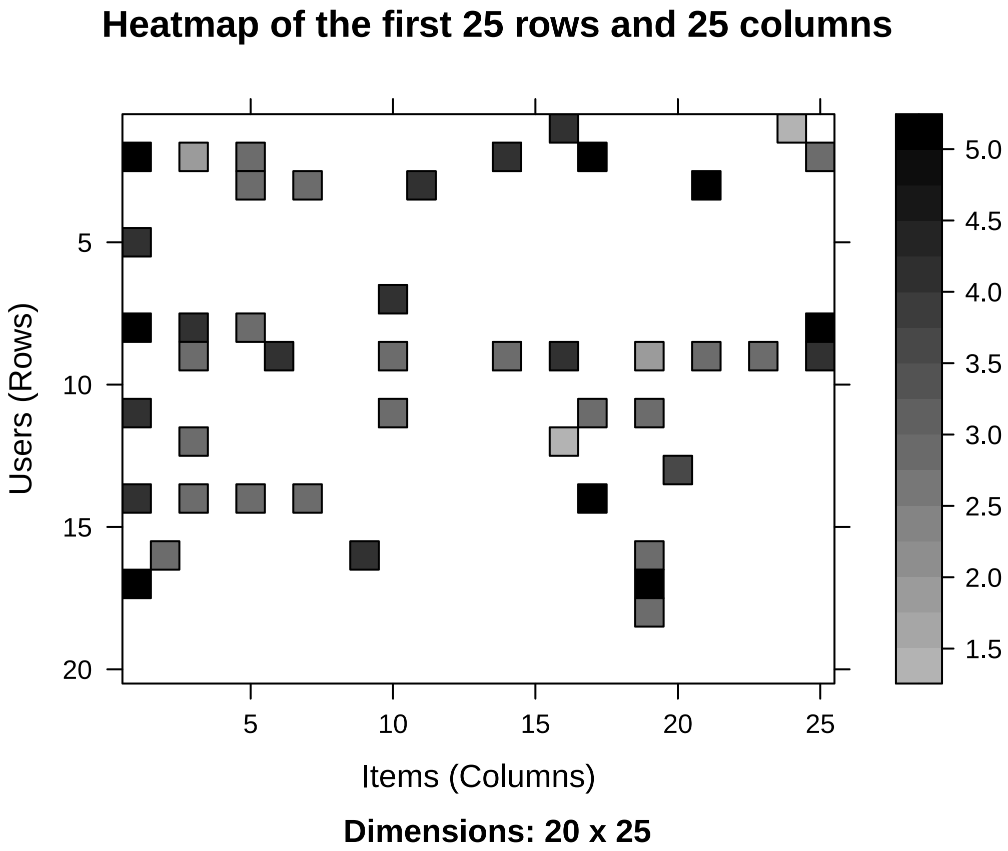
```
movie views                                              title
```

296 296 325 Pulp Fiction (1994) 356 356 311 Forrest Gump (1994) 318 318 308 Shawshank Redemption, The (1994) 480 480 294 Jurassic Park (1993) 593 593 290 Silence of the Lambs, The (1991) 260 260 273 Star Wars: Episode IV - A New Hope (1977)

```r
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")
```



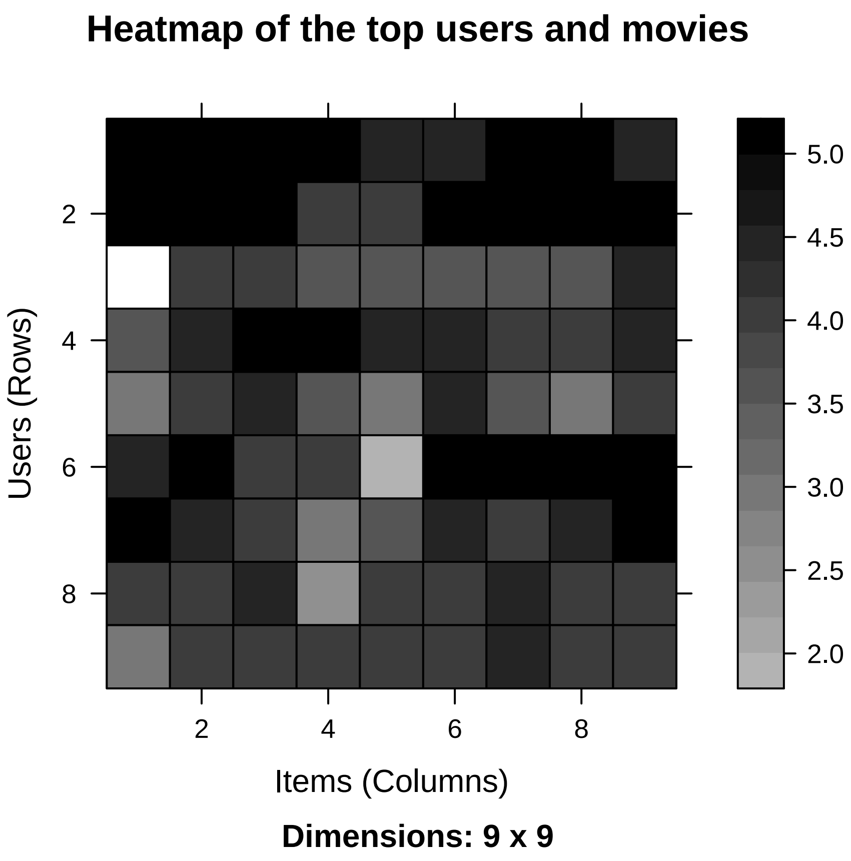
```r
image(ratingMatrix[1:20, 1:25], axes = FALSE, main = "Heatmap of the first 25 rows and 25 columns")
```



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

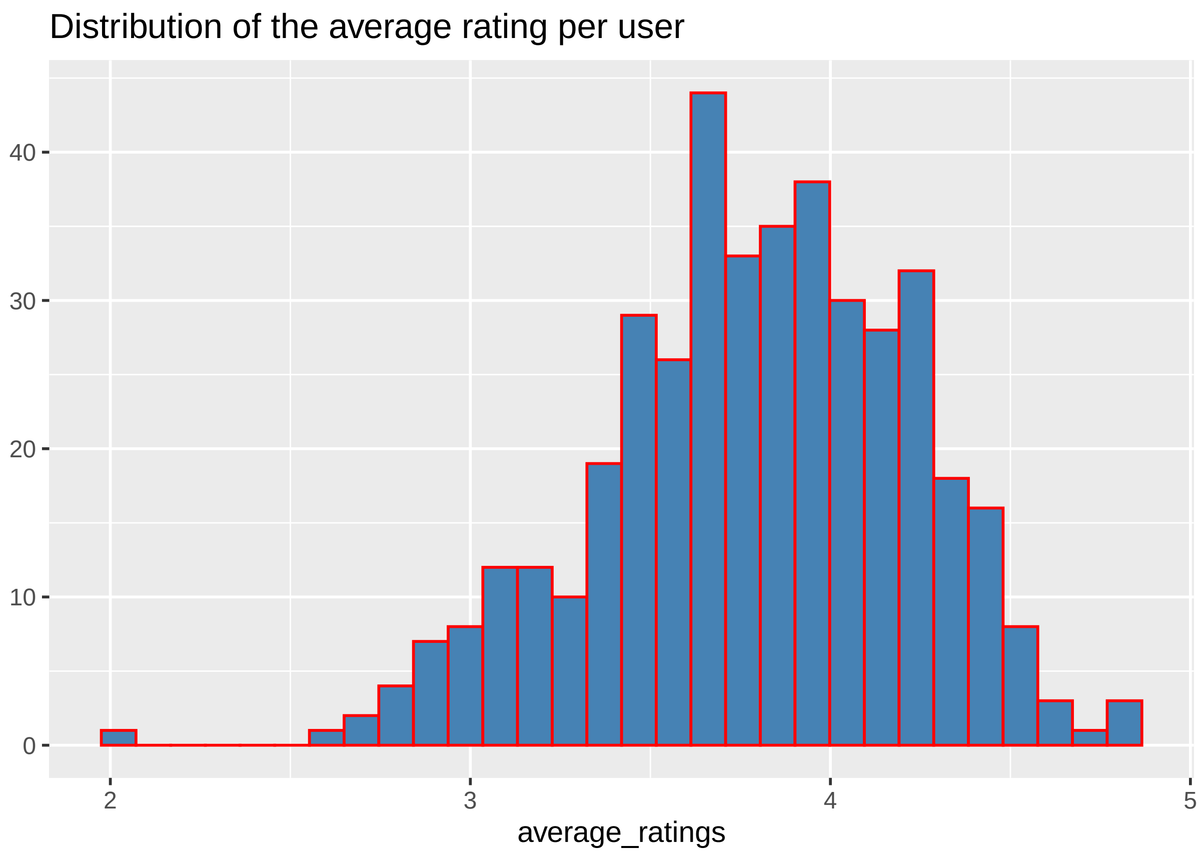
```
420 x 447 rating matrix of class 'realRatingMatrix' with 38341 ratings.
```

```
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_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")
```

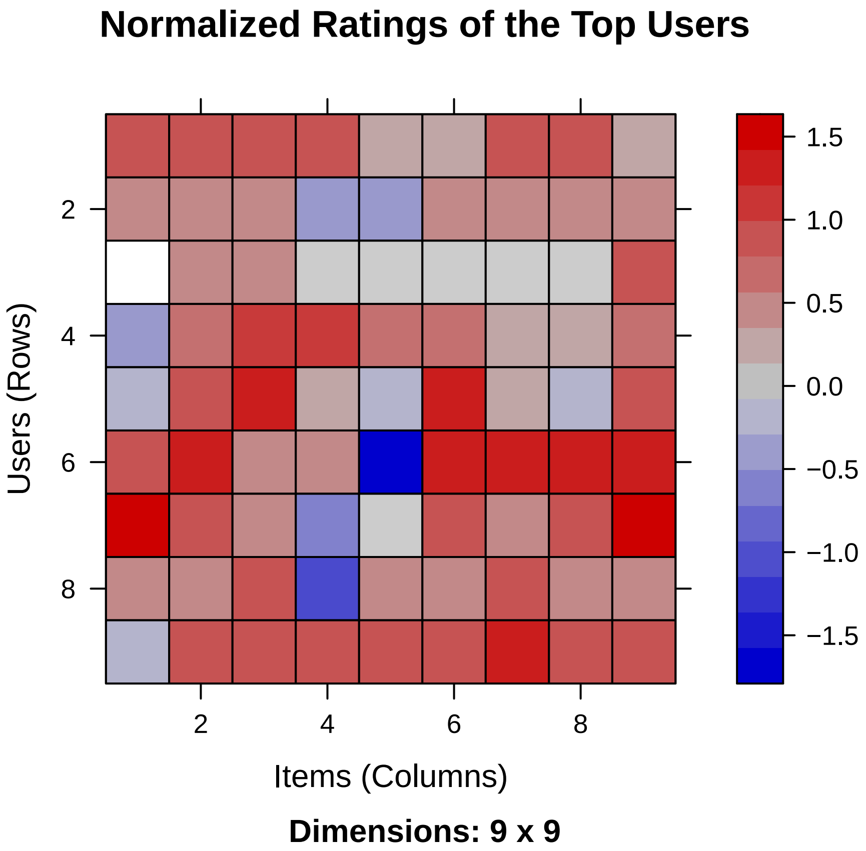
```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



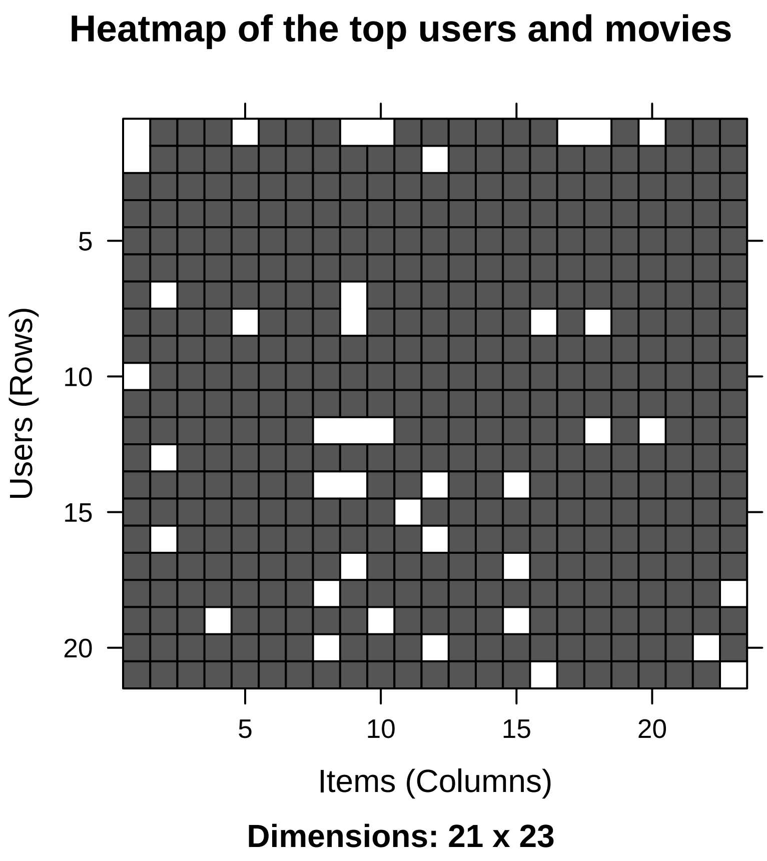
```
normalized_ratings <- normalize(movie_ratings)
sum(rowMeans(normalized_ratings) > 0.00001)
```

```
[1] 0
```

```
image(normalized_ratings[rowCounts(normalized_ratings) > minimum_movies,
                           colCounts(normalized_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)
#movies_watched <- binarize(movie_ratings, minRating = 1)
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")
```



```r
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, ]

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

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

Recommender of type ‘IBCF’ for ‘realRatingMatrix’ learned using 340 users.

```
class(recommen_model)
```

[1] "Recommender" attr(,"package") [1] "recommenderlab"

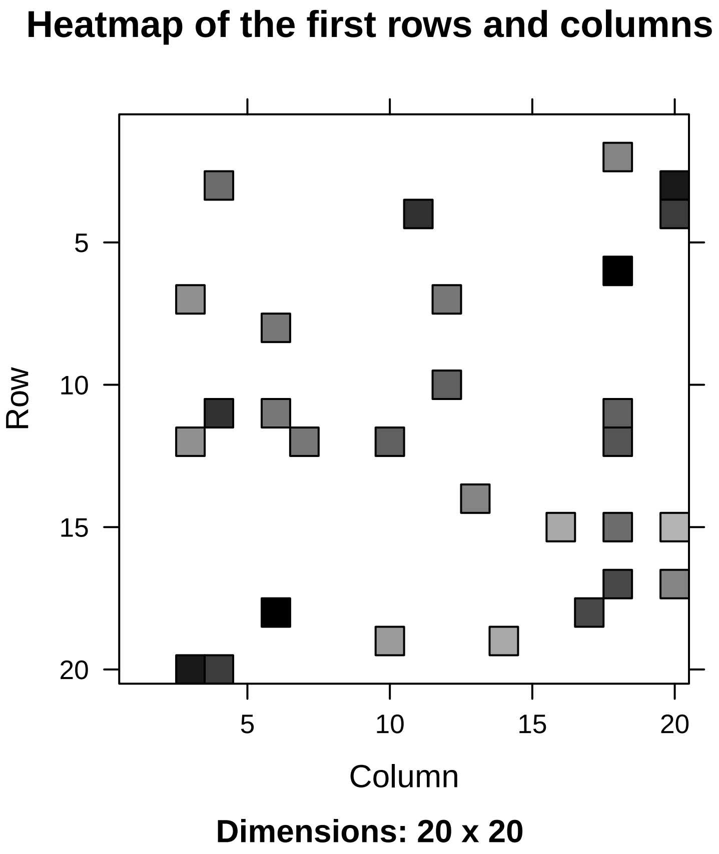
```
model_info <- getModel(recommen_model)
class(model_info$sim)
```

[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")
```

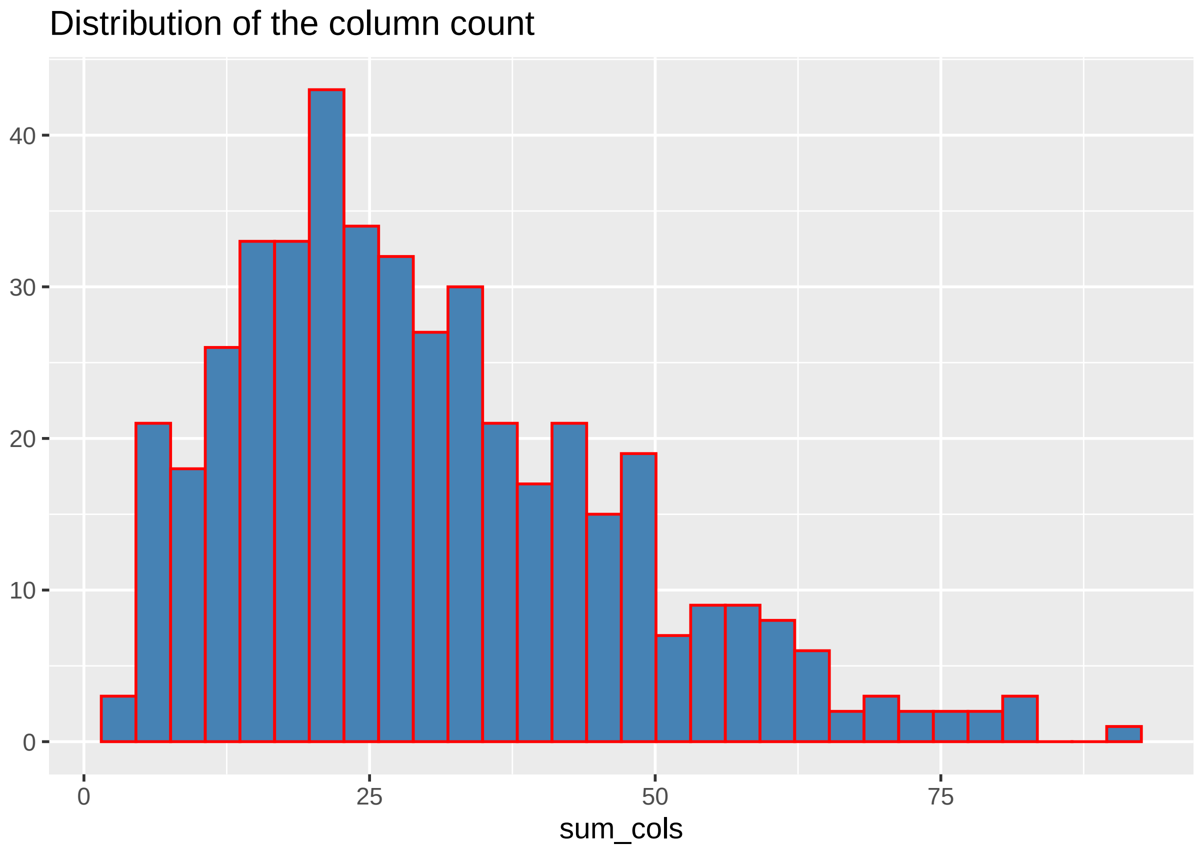


```
sum_rows <- rowSums(model_info$sim > 0)
table(sum_rows)
```

sum_rows 30 447

```r
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`.
```



```r
top_recommendations <- 10 # the number of items to recommend to each user
predicted_recommendations <- predict(object = recommen_model,
                          newdata = testing_data,
                          n = top_recommendations)
predicted_recommendations
```

Recommendations as ‘topNList’ with n = 10 for 80 users.

```r
user1 <- predicted_recommendations@items[[1]] # recommendation for the first user
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] “Toy Story (1995)”
[2] “Casino (1995)”

[3] “Twelve Monkeys (a.k.a. 12 Monkeys) (1995)” [4] “Seven (a.k.a. Se7en) (1995)”
[5] “Taxi Driver (1976)”
[6] “Blade Runner (1982)”
[7] “Trainspotting (1996)”
[8] “Vertigo (1958)”
[9] “Casablanca (1942)”
[10] “2001: A Space Odyssey (1968)”

```
recommendation_matrix <- sapply(predicted_recommendations@items,
                        function(x){ as.integer(colnames(movie_ratings)[x]) }) # matrix with the recommen
#dim(recc_matrix)
recommendation_matrix[,1:4]
```

```
    0    1     2      3
```

[1,] 1 7 7 551 [2,] 16 10 293 1343 [3,] 32 11 508 3081 [4,] 47 36 1047 3578 [5,] 111 161 1302 3897 [6,] 541 235 2329 49530 [7,] 778 329 2542 1704 [8,] 903 364 3253 4306 [9,] 912 440 6365 3175 [10,] 924 474 7147 1748