MOVIE RECOMMENDATION SYSTEM

By
Shashank M(AP18110010528)
Krishna Murari B(AP18110010533)
Hemanth M(AP18110010535)
Mouna B(AP18110010549)
Akhil K(AP18110010588)

NOVEMBER 15, 2021
BIG DATA ANALYTICS COURSE PROJECT
SRM UNIVERSITY AP

INDEX

Abstract

Recommender systems have outlined our selections on-line whether shopping for a book on Amazon or selecting a motion-picture show on Netflix. However, they're still in an exceedingly aborning stage of development and much from being excellent. During this paper, we tend to discuss motion-picture show recommender systems above all, numerous recommendation techniques, and the challenges they face. we tend to conjointly try and critically examine some work done on motion-picture show recommendation systems and discuss some analysis papers that have helped solve many challenges two-faced by these recommendation systems. However, despite of these advances, recommender systems still have to be compelled to be to be improved to an additional significant extent to be simpler in creating correct recommendations on a good array of applications. Nowadays, the advanced system has created finding the items simple that we want. motion-picture show recommendation systems aim at serving to motion-picture show enthusiasts by suggesting what motion-picture show to observe while not having to travel through the long method of selecting from an outsized set of flicks that go up to thousands and millions that's long and confusing. during this article, we have a tendency to aim to scale back human effort by suggesting movies supported the user's interests. To handle such issues, we have a tendency to introduced a model combining each content-based and cooperative approaches. it'll offer more and more express outcomes compared to completely different systems that area unit supported a content-based approach. Content-based recommendation systems area unit strained to folks, these systems don't inflict things out of the box, so limiting your option to explore additional. Hence, we've targeted on a system that resolves these problems.

Recommender procedures comprehend a category of manners and algorithms which will advocate "relevant" things to users. the long run forecast behavior is predicated on former knowledge through a large number of techniques together with matrix resolving.

This report discusses recommendations of the flicks. A motion-picture show recommendation is very important in our social life because of its strength in providing increased diversion. Such a system will counsel a collection of flicks to users supported their interest or the recognition of the flicks. A recommendation system is employed to counsel things to buy or to visualize. They direct users towards those things which might meet their wants by lowering an outsized information of data. A recommender system, or a recommendation system (sometimes exchange 'system' with a word like a platform or associate degree engine), may be a taxonomic group of data filtering system that seeks to predict the "rating" or "preference" a user would offer to associate degree item.

Introduction

Recommendation systems have wedged or maybe redefined our lives in many ways. One example of this impact is however our on-line looking expertise is being redefined. As we have a tendency to flick through product, the advice system provides recommendations of product we would have an interest in. despite the angle - business or shopper, Recommendation systems are vastly useful. and massive information is that the actuation behind Recommendation systems. A typical Recommendation system cannot do its job while not enough information and massive information provides many users information like past purchases, browsing history, and feedback for the advice systems to produce relevant and effective recommendations. during a shell, even the foremost advanced Recommenders

As business necessities increase, there's a high dependence on extracting relevant info from an outsized quantity of information to drive business solutions. what is more, because the quantity of accessible info grows, new issues emerge as people struggle to select the things they need to look at or utilize. this can be wherever the recommender system enters the image. They assist North American nation in creating judgments by learning our preferences or the preferences of like users.

Today, recommender systems square measure one in all the foremost used algorithms in information science.

They have intensive expertise with a range of fields, starting from diversion to e-commerce. With their implementation, Recommender Systems have shown to be useful in increasing company revenues and shopper happiness.

Almost each huge corporation employs them in some capability. Netflix uses it to propose movies to customers, YouTube uses it to see that video to play next on auto play, and Facebook uses it to suggest sites to love and other people to follow.

In this project, a cooperative filtering recommender (CFR) system for recommending movies is developed.

Literature Survey

As we have a tendency to all understand there are plenty of streaming services nowadays like Netflix, Amazon Prime, Hot star, to call some and these streaming services have humungous content libraries to decide on from however none of them have a correct system to pick and to determine so as to observe a particular picture as a result of none of them have all the films, a number of the foremost common drawback that happens is once a replacement user signs up for the service they need taken and don't have any plan what to observe additionally, the streaming service also cannot offer any recommendations as a result of they need terribly less data on the user like age and site, since there are multiple language this is often additionally a haul in suggesting for these services as a result of the user can’t choose that language of pictures of films to observe and additionally it's been ascertained that these services choose to recommend movies that are recently discharged and avoid previous movies and films with lower rating also are avoided thinking that the user won't like that movie. Since these services are continuously learning from the user, the formula might need to cope with monumental quantity of knowledge from all of its users and it'd suggest movies antecedently watched by the user or movies that the user failed to like once watched. Another issue is that a service is being shared by multiple folks and therefore the suggestions from them is targeted for multiple users that the system doesn’t understand and it would recommend a picture supported alternative users interest to somebody else.

Dataset

To make our recommendation system, we've got used the Movie Dataset. you'll be able to notice the Movie.csv and ratings.csv file that we've got employed in our Recommendation System Project (Click here). This dataset consists of 105339 ratings applied over 10329 movies.

A summary of *movies* is given below, together with several first rows of a dataframe:

```
##     movieId                                        title     
##  Min.   :     1   Confessions of a Dangerous Mind (2002):   2  
##  1st Qu.:  3248   Emma (1996)                           :   2  
##  Median :  7300   Eros (2004)                           :   2  
##  Mean   : 42200   Saturn 3 (1980)                       :   2  
##  3rd Qu.: 76232   War of the Worlds (2005)              :   2  
##  Max.   :193609   '71 (2014)                            :   1  
##                   (Other)                               :9731  
##            genres    
##  Drama         :1053  
##  Comedy        : 946  
##  Comedy|Drama  : 435  
##  Comedy|Romance: 363  
##  Drama|Romance : 349  
##  Documentary   : 339  
##  (Other)       :6257  

##   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
```

And here is a summary and a head of *ratings*:

```
##      userId         movieId          rating        timestamp    
##  Min.   :  1.0   Min.   :     1   Min.   :0.500   Min.   :8.281e+08  
##  1st Qu.:177.0   1st Qu.:  1199   1st Qu.:3.000   1st Qu.:1.019e+09  
##  Median :325.0   Median :  2991   Median :3.500   Median :1.186e+09  
##  Mean   :326.1   Mean   : 19435   Mean   :3.502   Mean   :1.206e+09  
##  3rd Qu.:477.0   3rd Qu.:  8122   3rd Qu.:4.000   3rd Qu.:1.436e+09  
##  Max.   :610.0   Max.   :193609   Max.   :5.000   Max.   :1.538e+09  
##   userId movieId rating timestamp
## 1      1       1      4 964982703
## 2      1       3      4 964981247
## 3      1       6      4 964982224
## 4      1      47      5 964983815
## 5      1      50      5 964982931
## 6      1      70      3 964982400
```

Both *usersId* and *movieId* are presented as integers and should be changed to factors. Genres of the movies are not easily usable because of their format, so we need to do the changes to get in the correct usable format.

Word cloud using tags.csv file.

Pre-processing

From the higher than table, we tend to observe that the user Id column, additionally because the movie Id column, carries with its integers. what is more, we'd like to convert the genres gift within the movie data frame into a lot of usable formats by the users. to try to thus, we are going to initial produce a one-hot cryptography to form a matrix that includes of corresponding genres for every of the films.

List of genres

Here a matrix of corresponding genres for each movie is created.

```
##   Action Adventure Animation Children Comedy Crime Documentary Drama
## 1      0         1         1        1      1     0           0     0
## 2      0         1         0        1      0     0           0     0
## 3      0         0         0        0      1     0           0     0
## 4      0         0         0        0      1     0           0     1
## 5      0         0         0        0      1     0           0     0
## 6      1         0         0        0      0     1           0     0
##   Fantasy Film-Noir Horror Musical Mystery Romance Sci-Fi Thriller War
## 1       1         0      0       0       0       0      0        0   0
## 2       1         0      0       0       0       0      0        0   0
## 3       0         0      0       0       0       1      0        0   0
## 4       0         0      0       0       0       1      0        0   0
## 5       0         0      0       0       0       0      0        0   0
## 6       0         0      0       0       0       0      0        1   0
##   Western
## 1       0
## 2       0
## 3       0
## 4       0
## 5       0
## 6       0
```

Word cloud of all the movie Genre

Search Matrix

A *search matrix* is created for an easy search of a movie by its genre.

```
##   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
```

It is clearly seen that each movie can correspond to either one or more than one genre.

Rate Matrix

The rating matrix is turned into a sparse matrix of type realRatingMatrix before it was used to develop a recommender system with recommenderlab.

```
## 610 x 9724 rating matrix of class 'realRatingMatrix' with 100836 ratings.
```

Proposed Method

ITEM-based Collaborative Filtering Model

Item-based CF may be a model-based approach that produces recommendations supported the link between things inferred from the rating matrix. the idea behind this approach is that users can like things that are the same as different things they like.

Collaborative filtering may be a branch of advice that takes account of the data regarding totally different users. The word “collaborative” refers to the very fact that users collaborate with one another to advocate things. In fact, the algorithms understand of user ratings and preferences.

The start line may be a rating matrix within which rows correspond to users and columns correspond to things. The core rule is predicated on these steps:

1. for every 2 things, live however similar they're in terms of getting received similar ratings by similar users

2. for every item, establish the k most similar things

3. for every user, establish the things that are most the same as the user’s purchases

Implementation of Code

```
#set working directory

setwd("D:/R files/movie recommendation/movie_rec_in_R")

#import libraries

library(recommenderlab)

library(data.table)

library(reshape2)

library(ggplot2)

#import data set

movies_df <- read.csv("C:/Users/murar/Downloads/movie-recommendation-main/movie-
recommendation-main/movies.csv")

ratings_df <- read.csv("C:/Users/murar/Downloads/movie-recommendation-main/movie-
recommendation-main/ratings.csv")

str(movies_df)

#get summary

summary(movies_df)

summary(ratings_df)

#clean the data

movie_genre<-as.data.frame(movies_df$genres , stringsAsFactors = FALSE)

movie_genre2<-as.data.frame(tstrsplit(movie_genre[,1],

‘[|]',

                              type.convert = TRUE),

                         stringsAsFactors = FALSE)

colnames(movie_genre2)<-c(1:10)
```

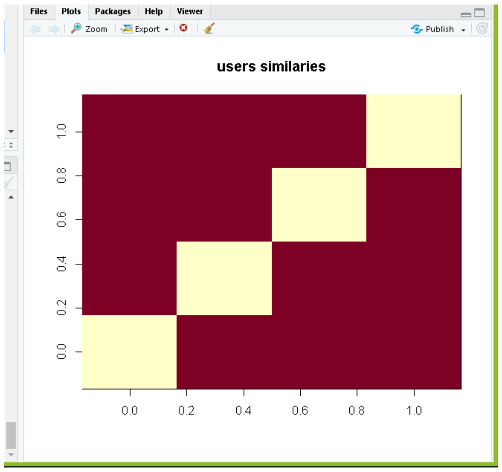
```
list_genre <- c("Action", "Adventure", "Animation", "Children",
                "Comedy", "Crime","Documentary", "Drama", "Fantasy",
                "Film-Noir", "Horror", "Musical", "Mystery","Romance",
                "Sci-Fi", "Thriller", "War", "Western")
genre_matrix1<-matrix(0,10330,18)
genre_matrix1[1,] <-list_genre
colnames(genre_matrix1) <- list_genre
for (index in 1:nrow(movie_genre2)) {
  for (col in 1:nrow(movie_genre2)) {
    gen_col <- which(genre_matrix1[1,] == movie_genre2[index , col])
    genre_matrix1[index + 1 , gen_col] <- 1
  }
}
genre_matrix2 <-as.data.frame(genre_matrix1[-1,] , stringsAsFactors = FALSE)
for (col in 1:ncol(genre_matrix2)) {
  genre_matrix2[,col] <-as.integer(genre_matrix2[,col])
}
str(genre_matrix2)
#create the search matrix
search_matrix <-cbind(movies_df[,1:2] , genre_matrix2)
#use recommendation matrix
rating_matrix <-dcast(ratings_df , ratings_df$userId~ratings_df$movieId  , value.var = "rating" ,
na.rm = FALSE)
rating_matrix <- as.matrix(rating_matrix[,-1])
#convert rating matrix into a recommendation matrix
rating_matrix <- as(rating_matrix , "realRatingMatrix")
rating_matrix
recommendation_model <- recommenderRegistry$get_entries(dataType = "realRatingMatrix")
names(recommendation_model)
```

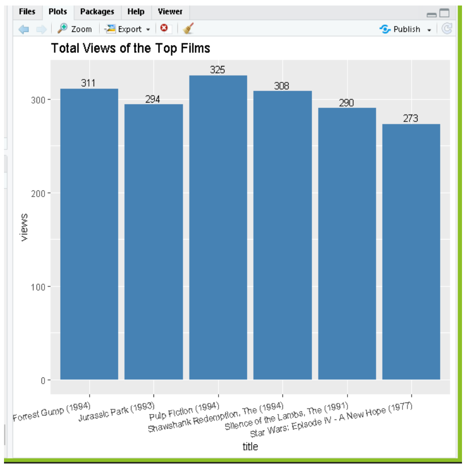
```
lapply(recommendation_model , "[["  ,"description")

recommendation_model$IBCF_realRatingMatrix$parameters

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

as.matrix(similarity_mat)

image(as.matrix(similarity_mat) , main = "users similaries")

rating_values <- as.vector(rating_matrix@data)

unique(rating_values) # extracting unique ratings

rating_table <-table(rating_values)

rating_table

movies_view<-colCounts(rating_matrix)

table_view<-data.frame(movie = names(movies_view),views = movies_view)

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

table_view$title <-NA

for (index in 1:10325) {

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

}

table_view[1:6 , ]


ggplot(table_view[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 = 10, hjust = 1)) +

  ggtitle("Total Views of the Top Films")

image(rating_matrix[1:20 , 1:20], axes=FALSE , main= "heat map")

movie_ratings<- rating_matrix[rowCounts(rating_matrix)>50,

                    colCounts(rating_matrix)>50]

movie_ratings

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

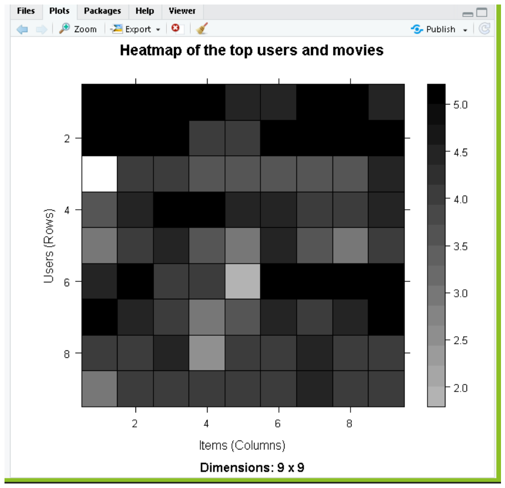
```
minimum_users<-quantile(colCounts(movie_ratings) , 0.98)
image(movie_ratings[rowCounts(movie_ratings) > minimum_movies,
                    colCounts(movie_ratings) > minimum_users],
      main = "Heatmap of the top users and movies")
average_ratings<-rowMeans(movie_ratings)
qplot(average_ratings , fill=I("steelblue") , col=I("red"))+
   ggtitle("distribution of the average rating per")
normlizied_rating<-normalize(movie_ratings)
sum(rowMeans(normlizied_rating)>0.00001)
image(normlizied_rating[rowCounts(normlizied_rating) > minimum_movies,
                        colCounts(normlizied_rating) > minimum_users],
      main = "normalized rating of the top users")
binary_minimum_movies <- quantile(rowCounts(movie_ratings) ,0.95)
binary_minimum_users <- quantile(colCounts(movie_ratings),0.95)
good_rated_flims<-binarize(movie_ratings , minRating = 3)
image(good_rated_flims[rowCounts(movie_ratings) > binary_minimum_movies ,
                       colCounts(movie_ratings) > binary_minimum_users],
      main = "heatmap of top usera 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,]
recommendation_syatem<-recommenderRegistry$get_entries(dataType="realRatingMatrix")
recommendation_syatem$IBCF_realRatingMatrix$parameters
recomm_model <- Recommender(data = training_data,
                            method="IBCF",
                        parameter=list(k=30))
recomm_model
class(recomm_model)
model_info<-getModel(recomm_model)
class(model_info$sim)
dim(model_info$sim)
top_items<-20
image(model_info$sim[1:top_items , 1:top_items],
     main="heatmap of the first row and column")
sum_rows<-rowSums(model_info$sim >0)
table(sum_rows)
sum_cols<-colSums(model_info$sim >0)
qplot(sum_cols , fill=I("blue") , col=I("red")) + ggtitle("distribution of the column count")
top_recommendation<-10  # the number of items to recommend to each user
predicted_recommendations<-predict(object = recomm_model,
                              newdata  = testing_data,
                              n=top_recommendation)
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(movies_df,
                                     movies_df$movieId==movies_user1[index])$title)
}
movies_user2
recommendation_matrix <- sapply(predicted_recommendations@items,
                            function(x){ as.integer(colnames(movie_ratings)[x]) }) # matrix with the
recommendations for each user
```

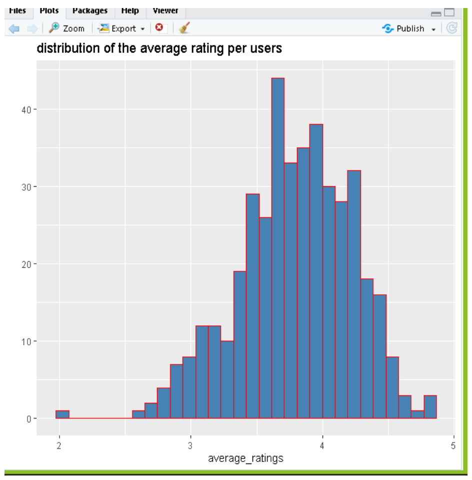
```
#dim(recc_matrix)
recommendation_matrix[,1:4]
number_of_items <- factor(table(recommendation_matrix))
chart_title <- "Distribution of the Number of Items for IBCF"
qplot(number_of_items, fill=I("steelblue"), col=I("red")) + ggtitle(chart_title)
number_of_items_sorted <- sort(number_of_items, decreasing = TRUE)
number_of_items_top <- head(number_of_items_sorted, n = 4)
table_top <- data.frame(as.integer(names(number_of_items_top)),
                number_of_items_top)
for(i in 1:4) {
  table_top[i,1] <- as.character(subset(movies_df,
                        movies_df$movieId == table_top[i,1])$title)
}
colnames(table_top) <- c("Movie Title", "No. of Items")
head(table_top)
```

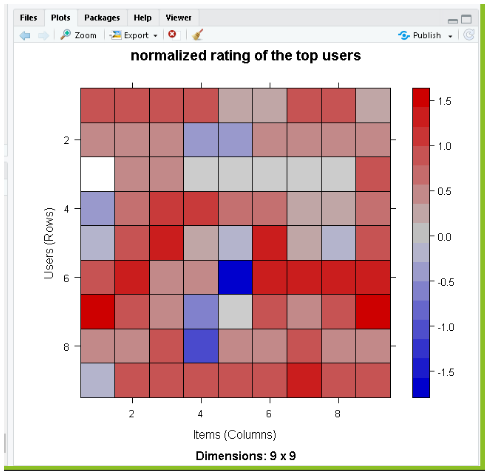
Graphs

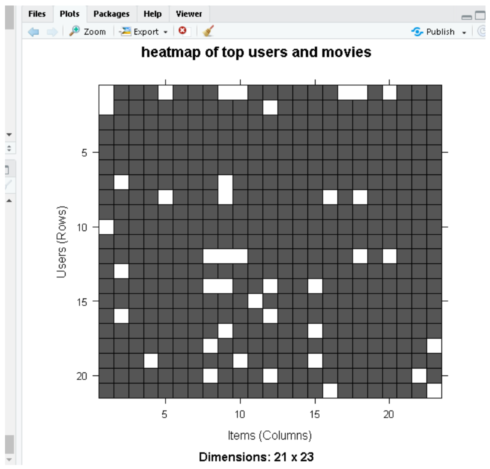


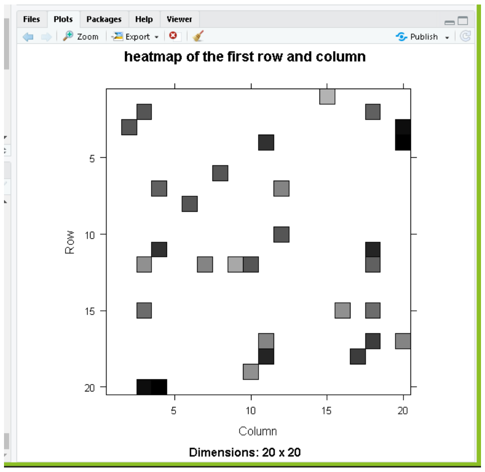


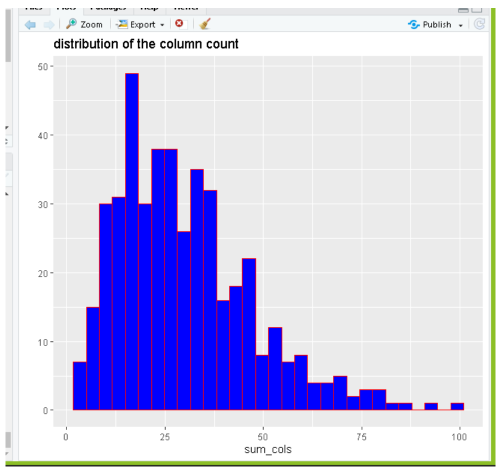

Files Plots Packages Help Viewer
Zoom Export Publish
Heatmap of the top users and movies
Users (Rows)
Items (Columns)
Dimensions: 9 x 9

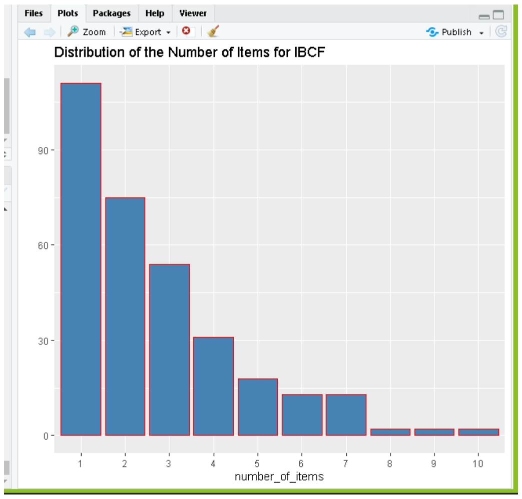

Files Plots Packages Help Viewer
Zoom Export Publish
distribution of the average rating per users
average_ratings


Files Plots Packages Help Viewer
Zoom Export Publish
normalized rating of the top users
Users (Rows)
Items (Columns)
Dimensions: 9 x 9


Files Plots Packages Help Viewer
Zoom Export Publish
heatmap of top users and movies
Users (Rows)
Items (Columns)
Dimensions: 21 x 23


Files Plots Packages Help Viewer
Zoom Export Publish
heatmap of the first row and column
Row
Column
Dimensions: 20 x 20


Zoom Export Publish
distribution of the column count
sum_cols



Conclusion

The basis of this project is the R extension package recommender lab, which is designed primarily for building and testing recommender algorithms. The package enables the creation of evaluation schemes depending on acknowledged methodologies, that can subsequently be used to assess and compare recommender algorithms.

We just built a movie recommender system based on the collaborative filtering method using the recommender lab library. We were successful in recommending ten movies that the user is likely to enjoy. Aside from the Movie Lens dataset, the recommender lab library might be used to generate recommendations using other datasets.

With enough data, collaborative filtering may be a valuable tool for data scientists to offer new goods or commodities to people. If you have rich metadata about your items, you might potentially apply a content-based approach to suggestions.

Recommendation Engine is your companion and counsellor, assisting you in making the best decisions by presenting customized options and creating a personalization for you. Without a question, recommendation engines are becoming increasingly critical and widespread in the digital era.

References

1. "recommenderlab: A Framework for Developing and Testing Recommendation Algorithms", Michael Hahsler (2019), SMU - Cran-r-vignettes
2. https://blog.datasciencedojo.com/movie-recommender-systems/
3. https://medium.com/@james_aka_yale/the-4-recommendation-engines-that-can-predict-your-movie-tastes-bbec857b8223
4. https://www.sas.com/content/dam/SAS/en_us/doc/conclusionpaper1/recommendation-systems-107451.pdf