MINOR PROJECT

ON

MOVIE RECOMMENDATION SYSTEM

REPORT FILE

BACHELOR OF TECHNOLOGY

(COMPUTER SCIENCE & ENGINEERING)

SUBMITTED BY:

Nitesh Kesharwani

Nisha Kumari

Vinay Pratap

UNDER THE GUIDANCE OF:

Mr Sachin Singh

IN

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ROORKEE INSTITUTE OF TECHNOLOGY

ROORKEE, UTTRAKHAND, INDIA

(2021-2022)

CERTIFICATE

I hereby certify that the work which is being presented in these entitled "**Movie Recommendation System**" in partial fulfilment of the requirement for the award of degree of Bachelor of Technology and submitted in Department of Computer Science of Roorkee Institute of Technology, Roorkee, is an authentic record of my own work carried out under the supervision of **Mr Sachin Singh.**

The matter presented in this report has not been submitted by me anywhere for the award of any other Degree of this or any other institute.

NITESH KESHARWANI

NISHA KUMARI

VINAY PRATAP

This is to clarify that the above statement made by the candidate is correct to the best of our knowledge.

Date: 04 MAY 2022

HOD | Project IN charge

(DR. DEEPAK ARYA) | **Mr Sachin Singh**

<u>STUDENT INFORMATION</u>

- NAME: ***NITESH KESHARWANI***
    - ***NISHA KUMARI***
    - ***VINAY PRATAP***
- COURSE: ***BTECH***
- BRANCH: ***COMPUTER SCIENCE AND ENGINEERING***
- UNIVERSITY ROLL NUMBER: ***190240101067***
    - ***190240101066***
    - ***190240101122***
- YEAR: ***3rd YEAR (2019-23)***
- COLLEGE NAME: ***ROORKEE INSTITUTE OF TECHNOLOGY***
- PROJECT NAME: ***Movie Recommendation System***
- SUBMITTED TO: ***Mr Sachin Singh***

ABOUT PROJECT

Have you ever been on an online streaming platform like Amazon Prime, Voot, Netflix, and so on? After watching a movie on these platforms, that platform starts recommending us different movies and TV shows related to the previously watched content. Just wonder, how the movie streaming platform can suggest users the content that can appeal to them. This can be achieved by a system known as Movie Recommendation System. This system is capable of learning user's watching patterns and providing them with relevant suggestions for more such movies. Having witnessed the fourth industrial revolution where Artificial Intelligence and other technologies are dominating the market, it is sure that everyone must have come across a recommendation system in their everyday life. So, our team of three people had tried to develop a similar recommendation system model through machine learning, and R language. In this project of making an recommendation system, I will be feeding the genre types a particular movie is watched by a user and accordingly similar recommendations are provided to that user.

SOURCE CODE

```
#RECOMMENDATION SYSTEM CODE ####

#Install the below mentioned 4 packages if you don't have
them in your systems to run the code.

install.packages("recommenderlab")
install.packages("ggplot2")
install.packages("data.table")
install.packages("reshape2")
#Finish installation of these packages to use some of their
libraries.

#Importing libraries required in this project

library(recommenderlab)
library(ggplot2)
library(data.table)
library(reshape2)


#Reading data from movies.csv in movie_data
#Please ensure your directory to avoid any error in reading
the data.
#The directory can be checked by getwd() function
#The directory can be set by setwd() function
```

```
movie_data <-
read.csv("movies.csv",stringsAsFactors=FALSE)

rating_data <- read.csv("ratings.csv")

#Using View() function to view the data in movies.csv and
ratings.csv files.
View(movie_data)
View(rating_data)

#Using str() function for compactly displaying the internal
structure of a R object.
str(movie_data)
str(rating_data)

#Using summary() function to get the minimum value,
maximum value, 1st to 4rth quartile in the dataset.
summary(movie_data)
summary(rating_data)


#Using head() function to display the top 6 data entries in
the data set.
head(movie_data)
head(rating_data)
```

#HEADING TOWARDS DATA PRE_PROCESSING ####

#we need to convert the genres present in the movie_data data frame into a more usable format by the users.
#In order to do so, we will first create a one-hot encoding to create a matrix that comprises of corresponding genres for each of the films.
#Taking the genre columnn of movies.csv files whose stringASFactors value is ZERO as a data frame.

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

#Viewing the dataset
View(movie_genre)


movie_genre2 <- as.data.frame(tstrsplit(movie_genre[,1],
'[|]',
                              type.convert=TRUE),
                    stringsAsFactors=FALSE)
View(movie_genre2)


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")


#Making matrices here:
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) #remove first row, which was the
genre list
for (col in 1:ncol(genre_mat2)) {
  genre_mat2[,col] <- as.integer(genre_mat2[,col]) #convert
from characters to integers
}

str(genre_mat2)
```

#Making ‘search matrix’ that will allow to perform an easy search of the films by specifying the genre present in the list.
SearchMatrix <- cbind(movie_data[,1:2], genre_mat2[])
head(SearchMatrix)

#There are movies that have several genres like Toy Story has genre: Animated film, Comedy, Fantasy, and Children.
#This applies to the majority of the films.
#For the movie recommendation system to make sense of the ratings through recommenderlabs,
#we have to convert the matrix into a sparse matrix one.
#This new matrix is of the class ‘realRatingMatrix’. This is performed as follows:

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
#Sparse matrix is a matrix having most of the elements zero
ratingMatrix <- as(ratingMatrix, "realRatingMatrix")
ratingMatrix

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

```
names(recommendation_model)

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

recommendation_model$IBCF_realRatingMatrix$paramete
rs



#Now we will explore similar data by collaborative filtering
#Collaborative Filtering involves suggesting movies to the
users that are based on collecting preferences from many
other users.
#With the help of recommenderlab, we compute
similarities using various operators like cosine, pearson as
well as jaccard.

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

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

#In the above matrix, each row and column represents a
user. We have taken four users and each cell in this matrix
#represents the similarity that is shared between the two
users.

movie_similarity <- similarity(ratingMatrix[, 1:4], method =
```

```
                        "cosine", which = "items")
as.matrix(movie_similarity)

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

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


#creating a table of ratings that will display the most
unique ratings.
Table_of_Ratings <- table(rating_values) # creating a count
of movie ratings
Table_of_Ratings

#Most viewed movies visualization

#We will explore the most viewed movies in our data set
now.
#We will first count the number of views in a film and then
organize them in a
#table that would group them in descending order.

movie_views <- colCounts(ratingMatrix) # count views for
each movie
table_views <- data.frame(movie = names(movie_views),
                    views = movie_views) # create data frame of
views
table_views <- table_views[order(table_views$views,
```

```
                        decreasing = TRUE), ] # sorting 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,]
```

#visualizing a bar plot for the total number of views of the top films.
#We will carry this out using ggplot2.

```
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")
```

#From the above bar-plot, we observe that Pulp Fiction is the most-watched film followed by Forrest Gump.

#Visualizing a heat map of the movie ratings.
#This heat map will contain first 25 rows and 25 columns as follows:

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


#Performing Data Preparation
#We will conduct data preparation in the following three
steps –
#1.Selecting useful data.
#2.Normalizing data.
#3.Binarizing the data.



#For finding useful data in our data set, we have set the
threshold for the minimum number
#of users who have rated a film as 50.
#This is also same for minimum number of views that are
per film.
#This way, we have filtered a list of watched films from
least-watched ones.


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

#From the above output of ‘movie_ratings’, we observe
that there are 420 users and 447 films as
#opposed to the previous 668 users and 10325 films.
```

```
#We can now delineate our matrix of relevant users as
follows–

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")


#visualizing the distribution of the average ratings per user.

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


#HEADING TOWARDS DATA NORMALIZATION


#In the case of some users, there can be high ratings or low
ratings provided to all of the watched films.
#This will act as a bias while implementing our model.
#In order to remove this, we normalize our data.
#Normalization is a data preparation procedure to
standardize the numerical values in a column to a common
scale value.
```

#This is done in such a way that there is no distortion in the range of values.
#Normalization transforms the average value of our ratings column to 0.
#We then plot a heat map that delineates our normalized ratings.

```
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")


#PERFORMING DATA BINARIZATION- Final Step

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,
```

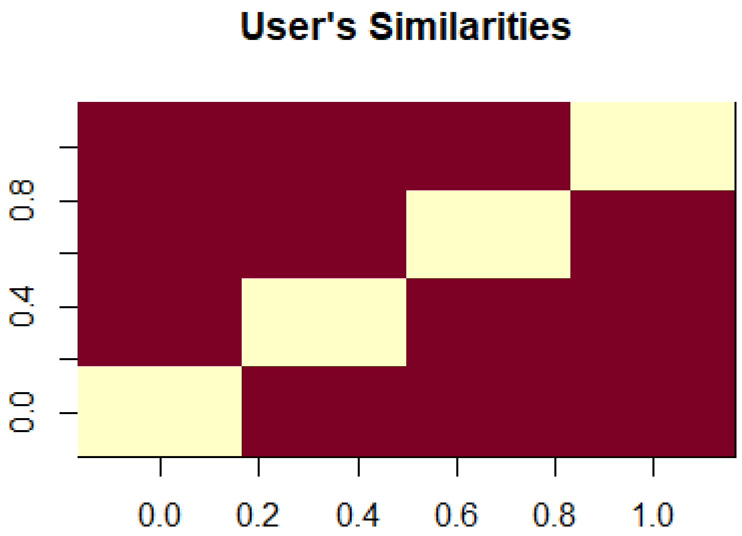
```
                colCounts(movie_ratings) >
binary_minimum_users],
    main = "Heatmap of the top users and movies")


#Collaborative Filtering System
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, ]


#Building the Recommendation System
recommendation_system <-
recommenderRegistry$get_entries(dataType
="realRatingMatrix")
recommendation_system$IBCF_realRatingMatrix$paramet
ers

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


#Let us now explore our data science recommendation
system model as follows –
```

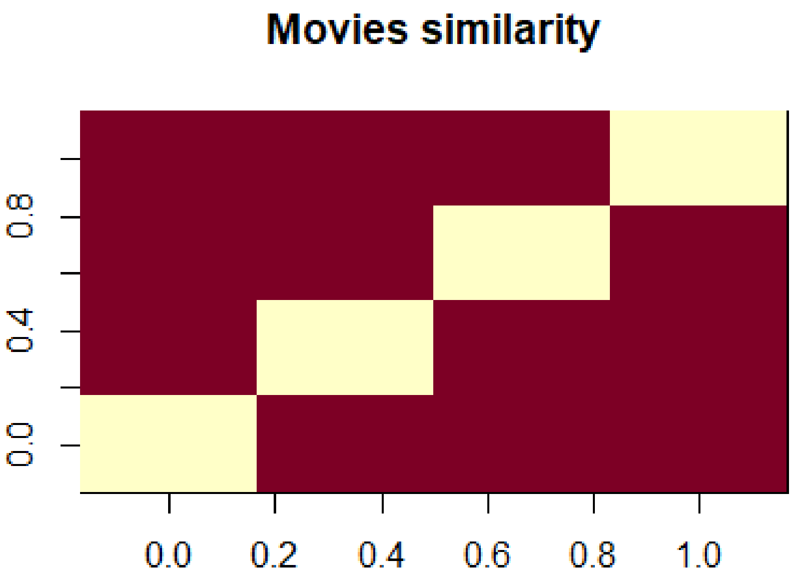
```
#Using the getModel() function, we will retrieve the recommen_model.
#We will then find the class and dimensions of our similarity matrix that is contained within model_info.
#Finally, we will generate a heatmap, that will contain the top 20 items and visualize the similarity shared between them.


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")


#We will carry out the sum of rows and columns with the similarity of the objects above 0.
#We will visualize the sum of columns through a distribution as follows –

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 # the number of items to recommend to each user
```

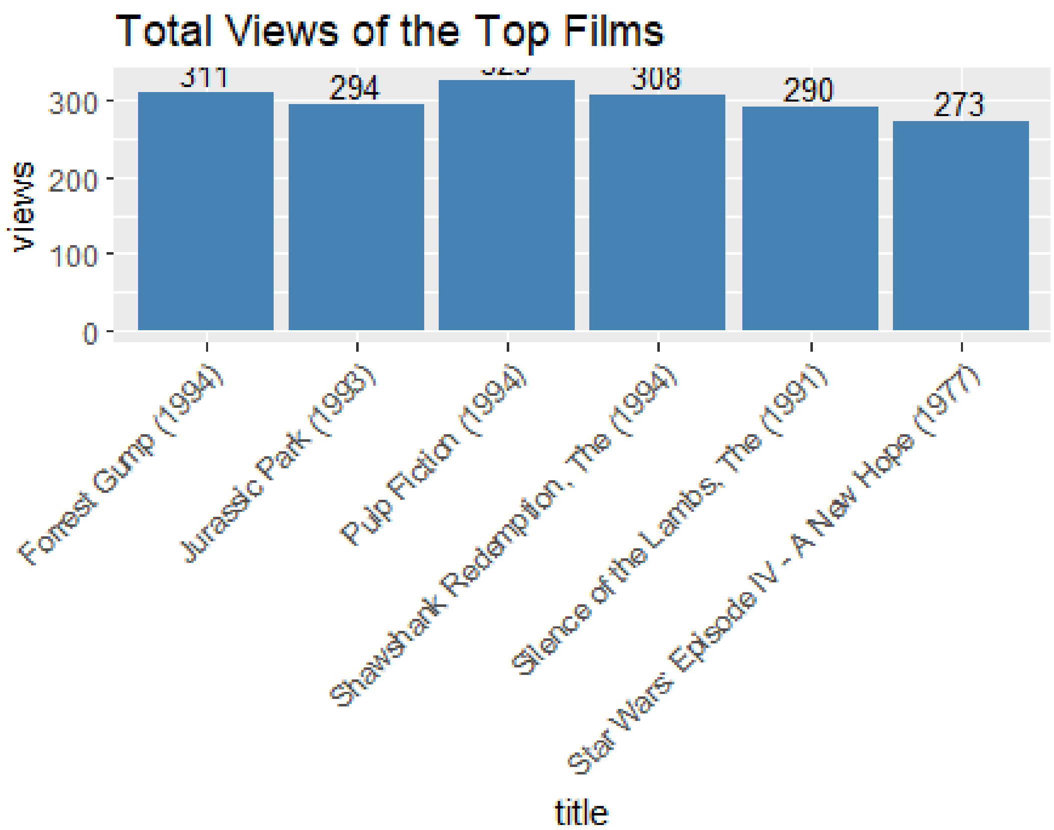
```
predicted_recommendations <- predict(object =
recommen_model,
                                      newdata = testing_data,
                                      n = top_recommendations)
predicted_recommendations


#build Recommender System on data set
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


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]
```

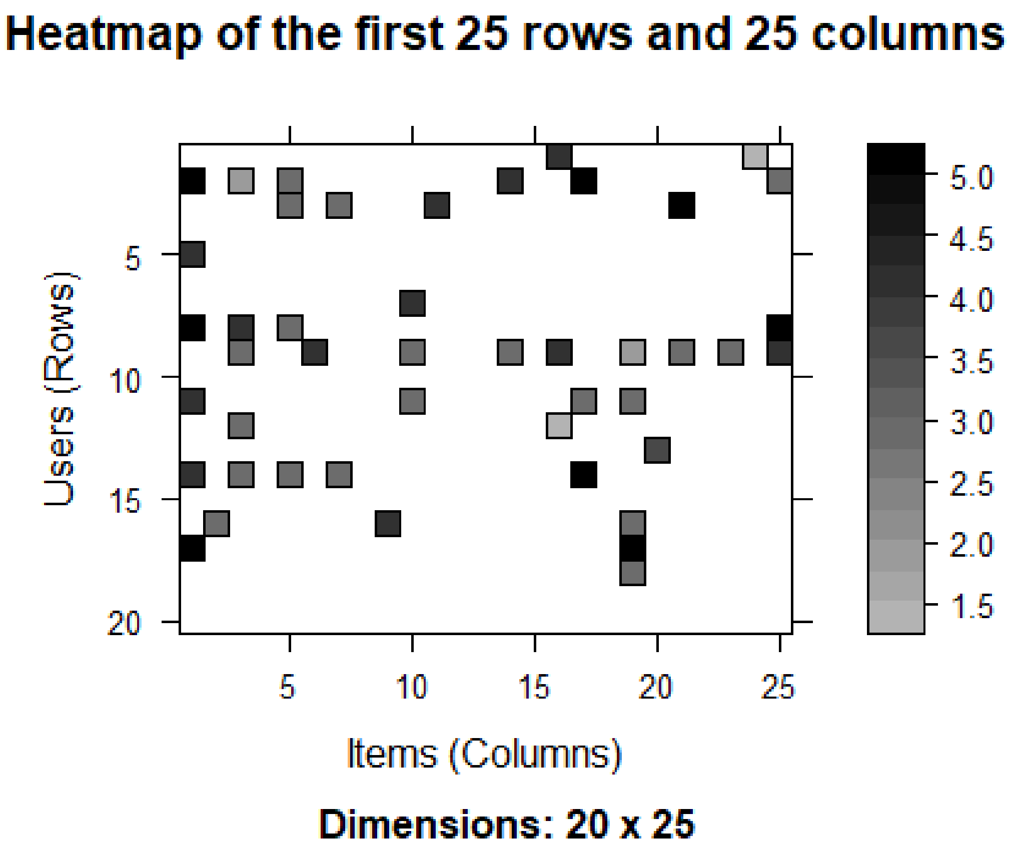
OUTPUT

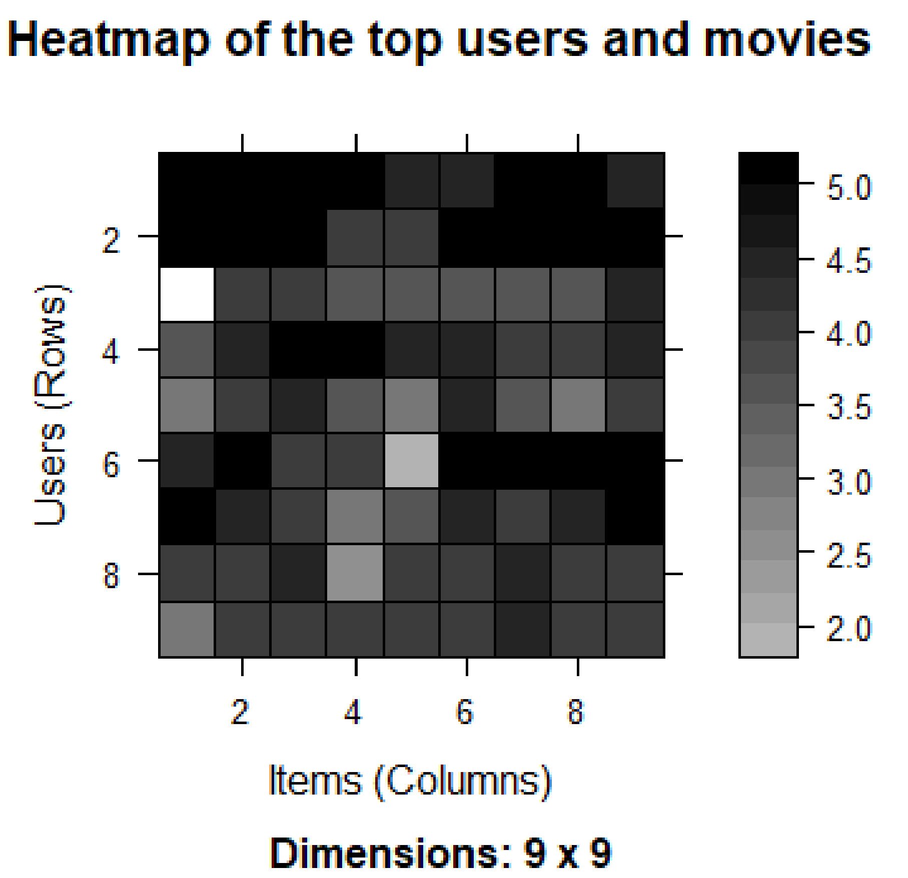
Here are some images of the data analysed and used in this project:

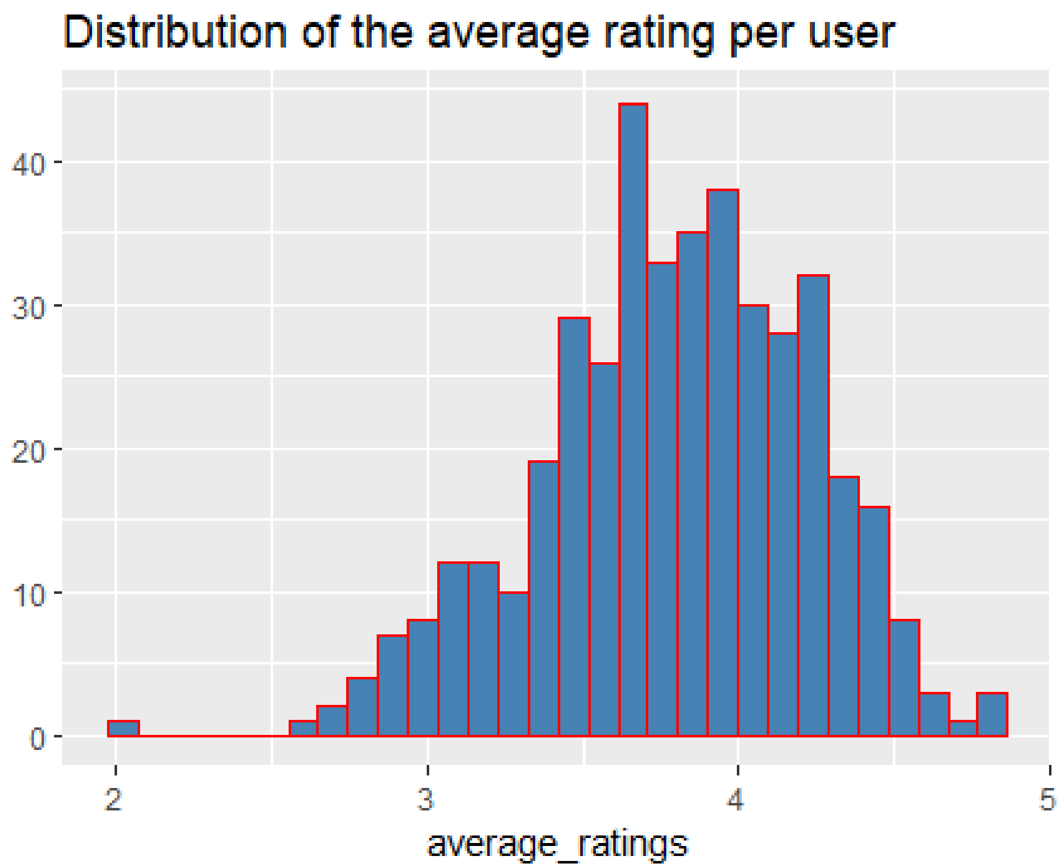


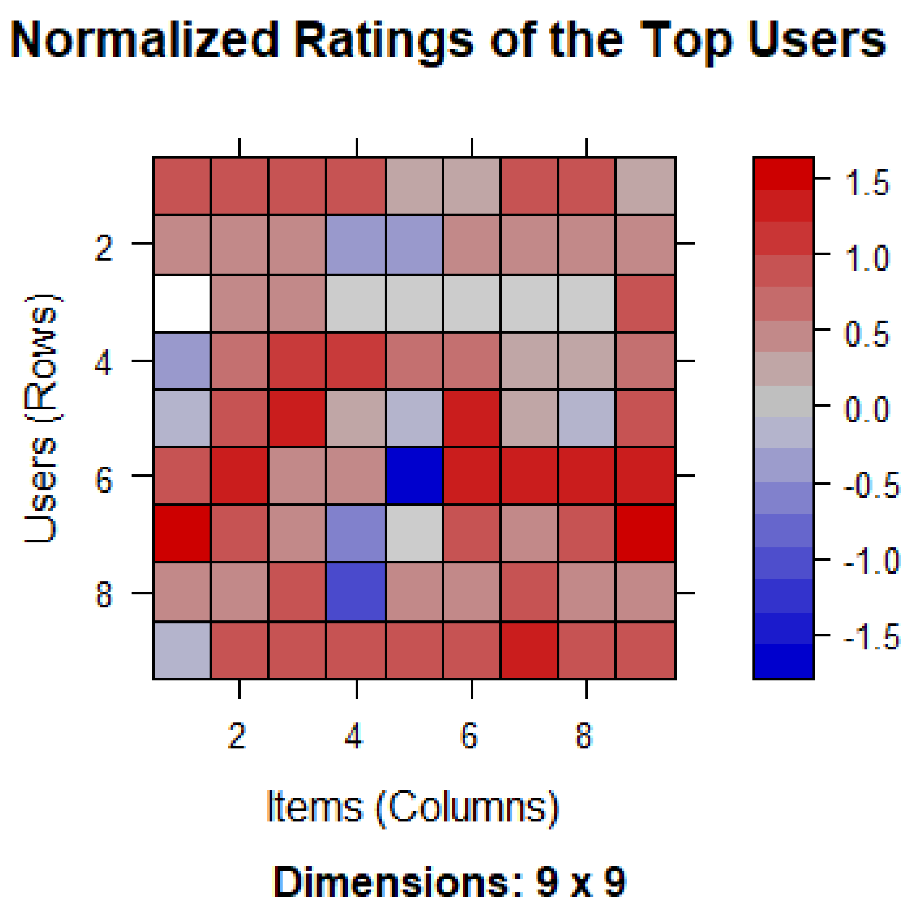


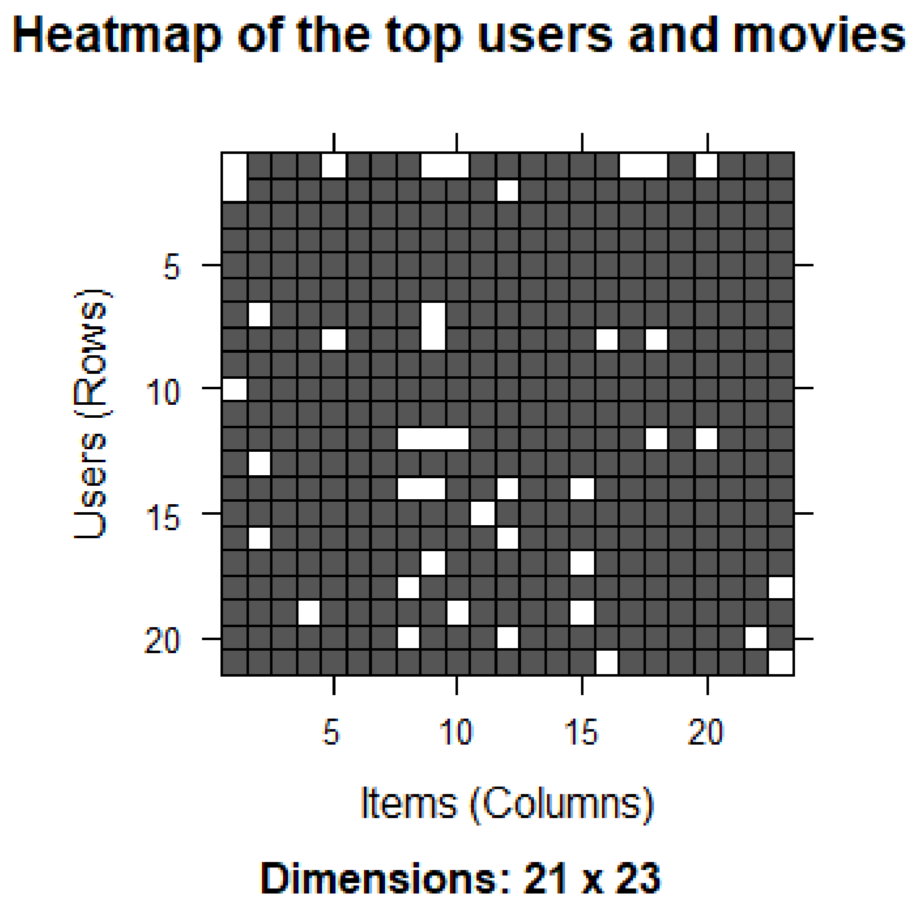

Total Views of the Top Films
311
294
308
290
273
views
300
200
100
0
Forest Gump (1994)
Jurassic Park (1993)
Pulp Fiction (1994)
Shawshank Redemption, The (1994)
Silence of the Lambs, The (1991)
Star Wars: Episode IV - A New Hope (1977)
title


Heatmap of the first 25 rows and 25 columns
Users (Rows)
5
10
15
20
Items (Columns)
5
10
15
20
25
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
Dimensions: 20 x 25

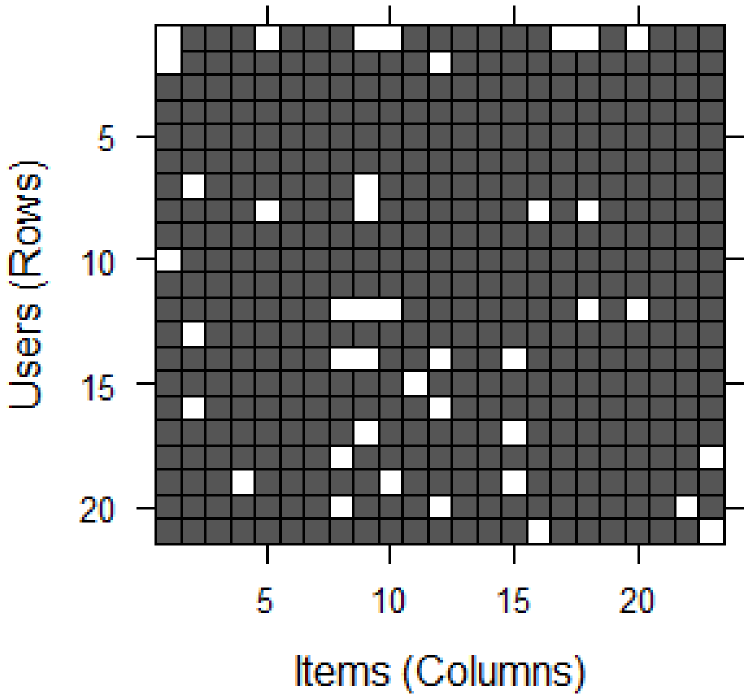

Heatmap of the top users and movies
Users (Rows)
Items (Columns)
Dimensions: 9 x 9
2
4
6
8
5.0
4.5
4.0
3.5
3.0
2.5
2.0


Distribution of the average rating per user
0
10
20
30
40
2
3
4
5
average_ratings


Normalized Ratings of the Top Users
Users (Rows)
2
4
6
8
Items (Columns)
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
Dimensions: 9 x 9

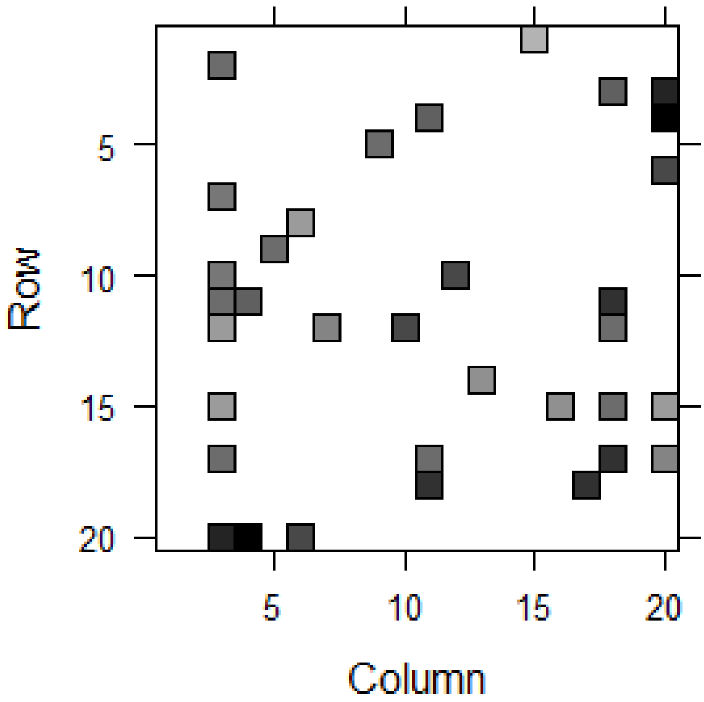

Heatmap of the top users and movies
Users (Rows)
5
10
15
20
Items (Columns)
Dimensions: 21 x 23

Heatmap of the top users and movies



Dimensions: 21 x 23

Heatmap of the first rows and columns



Dimensions: 20 x 20

Mechanism involved in developing the Movie Recommendation System–

- Machine learning which covers almost full concept of this project inculcating concepts as feeding of data, prediction, making conclusive outputs.

- R language for doing the programming part.

- Libraries of R such as recommenderlab. Data.table, ggplot2, reshape2.

- Movies have different genres as per their story; we will be categorizing user interest in a particular movie as per their genre and suggesting them similar movies of the same genre.

FUTURE SCOPE OF THE PROJECT

1. Recommendation systems help E-commerce sites to increase their sales. A very famous movie recommendation system named MOVREC, based on collaborative filtering approach makes use of the

information provided by users, analyses them and then recommends the movie that is best suited to the user at that time using k-means clustering algorithm.

2. Recommendation system helps to personalize a platform and help the user find something they like. It really enhance the user experience through personalized recommendations, we need dedicated recommender system. In today's scenario where everyone is opting for online platforms to watch movies, these recommendation systems give users more and more suggestions to watch movies on their developed platforms.
3. The Movie Recommendation System develops interest in users to watch more movies as users' brain stimulated interest in them to watch more and more movies.

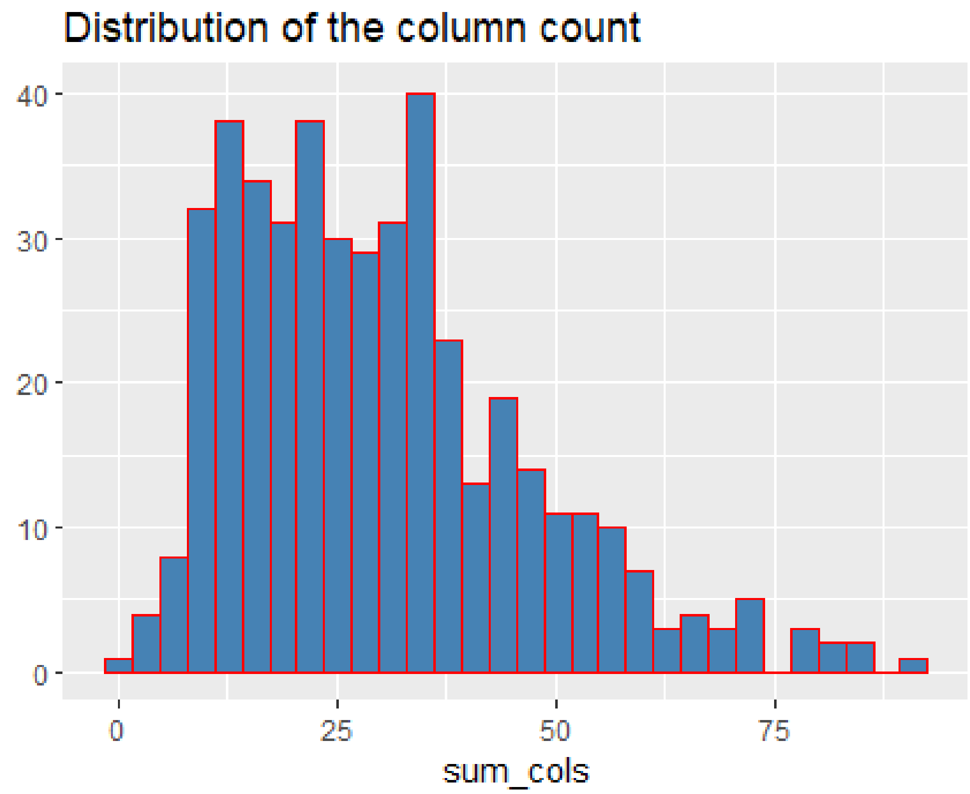
LANGUAGES USED IN THE PROJECT

In this project, Recommendation System, 'R' programming language is used. In this language various libraries have been imported so that the code can run efficiently and give the desirable output.

Few of the libraries that are used here are:

1) recommenderlab
2) ggplot2
3) data.table
4) reshape2

- An image from the data stat of the project



SUMMARY

In today’s scenario, many people opt online movie watching systems rather than going out to cinema halls gradually and gradually. Secondly, many a times people due to their busy schedule are not able to watch movies in theatres, then to go on to online platforms to watch movies, while watching such movies, people come across other movies of similar genres as per their watch. This gives users wider scope of watching movies of their interest and also these platforms earn more money when people watch more and more movies. Recommendation Systems are the most popular type of machine learning applications that are used in all sectors. They are an improvement over the traditional classification algorithms as they can take many classes of input and provide similarity ranking based algorithms to provide the user with accurate results. These recommendation systems have evolved over time and have incorporated many advanced machine learning techniques to provide the users with the content that they want.