Proyecto Movielens

JOSE IGNACIO SORDO SIERPE

04/01/2022

1. Introduction

This project is carried out as a final summary of the Data Science Harvard Edx courses, based on the programming language R.

During the project, a movie recommendation system will be created using the Movielens dataset, in its 10M version.

The objective of a recommendation system is to generate suggestions for new items or to predict the usefulness of a specific item for a particular user.

In Machine Learning, to perform data analysis and make predictions, the datasets are divided into two different samples, a training sample and a validation sample. In this way, the overfitting that would occur if the machine trains with all the data with which it also makes the predictions is eliminated.

In our case, the sample has been divided into 90% for the training set (edx) and 10% for the validation sample (validation)

The training sample consists of 9000055 observations with a total of 6 variables.

The variables are “userId”, “movieId”, “rating”, “timestamp”, “title”, “genres” UserId — Numeric asigned to each user Movie Id — Numeric asigned to each movie Rating — Categorical rating of the movie, range from 0.5 to 5 Timestamp — This variable represents the time and data in which the rating was provided. The units are seconds since January 1, 1970 Title — Name ot the movie and year Genres — Movies genres

The validation sample consists of a total of 999999 observations.

The metric that will be used to evaluate the predictions in the validation set is RMSE (residual mean squared error). And the goal is to get a value lower than 0.86490.

RMSE <- function(true_ratings, predicted_ratings){sqrt(mean((true_ratings - predicted_ratings)^2))}

To execute the project we will follow the following phases: exploration and visualization of the data, methodology used, results obtained and final conclusions.

The data is downloaded according to the instructions in the MovieLens 10M data set provided by the course.

```
# Note: this process could take a couple of minutes

if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: tidyverse
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.0     v forcats 0.5.1
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
## Warning: package 'tibble' was built under R version 4.0.5
```

```
## Warning: package 'tidyr' was built under R version 4.0.5
```

```
## Warning: package 'readr' was built under R version 4.0.5
```

```
## Warning: package 'purrr' was built under R version 4.0.5
```

```
## Warning: package 'dplyr' was built under R version 4.0.5
```

```
## Warning: package 'stringr' was built under R version 4.0.5
```

```
## Warning: package 'forcats' was built under R version 4.0.5
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: caret
```

```
## Warning: package 'caret' was built under R version 4.0.5
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: data.table
```

```
## Warning: package 'data.table' was built under R version 4.0.5
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
## The following object is masked from 'package:purrr':
##
##     transpose
```

```
library(tidyverse)
library(caret)
library(data.table)
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.0.5
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(kableExtra)
```

```
## Warning: package 'kableExtra' was built under R version 4.0.5
```

```
##
## Attaching package: 'kableExtra'
## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```
# MovieLens 10M dataset:
# https://grouplens.org/datasets/movielens/10m/
# http://files.grouplens.org/datasets/movielens/ml-10m.zip

dl <- tempfile()
download.file("http://files.grouplens.org/datasets/movielens/ml-10m.zip", dl)

ratings <- fread(text = gsub("::", "\t", readLines(unzip(dl, "ml-10M100K/ratings.dat"))),
                 col.names = c("userId", "movieId", "rating", "timestamp"))

movies <- str_split_fixed(readLines(unzip(dl, "ml-10M100K/movies.dat")), "\\::", 3)
colnames(movies) <- c("movieId", "title", "genres")

# if using R 4.0 or later:
movies <- as.data.frame(movies) %>% mutate(movieId = as.numeric(movieId),
                                           title = as.character(title),
                                           genres = as.character(genres))


movielens <- left_join(ratings, movies, by = "movieId")

# Validation set will be 10% of MovieLens data
set.seed(1, sample.kind="Rounding") # if using R 3.5 or earlier, use `set.seed(1)`
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
test_index <- createDataPartition(y = movielens$rating, times = 1, p = 0.1, list = FALSE)
edx <- movielens[-test_index,]
temp <- movielens[test_index,]

# Make sure userId and movieId in validation set are also in edx set
validation <- temp %>%
  semi_join(edx, by = "movieId") %>%
  semi_join(edx, by = "userId")

# Add rows removed from validation set back into edx set
removed <- anti_join(temp, validation)
```

```
## Joining, by = c("userId", "movieId", "rating", "timestamp", "title", "genres")
```

```
edx <- rbind(edx, removed)

rm(dl, ratings, movies, test_index, temp, movielens, removed)
```

2. Methodology

2.1. Data description

First of all we are going to see a visualization of the data set, to understand how it is structured.

```
#dataset visualization
head(edx)
```

```
##    userId movieId rating timestamp                         title
## 1:      1     122      5 838985046              Boomerang (1992)
## 2:      1     185      5 838983525               Net, The (1995)
## 3:      1     292      5 838983421               Outbreak (1995)
## 4:      1     316      5 838983392               Stargate (1994)
## 5:      1     329      5 838983392 Star Trek: Generations (1994)
## 6:      1     355      5 838984474       Flintstones, The (1994)
##                           genres
## 1:                Comedy|Romance
## 2:         Action|Crime|Thriller
## 3:  Action|Drama|Sci-Fi|Thriller
## 4:       Action|Adventure|Sci-Fi
## 5: Action|Adventure|Drama|Sci-Fi
## 6:       Children|Comedy|Fantasy
```

We analyze the number of users that the data set has, the number of movies and the number of movie genres.

```
# unique movies, users and genres

edx %>% summarise(
  uniq_movies = n_distinct(movieId),
  uniq_users = n_distinct(userId),
  uniq_genres = n_distinct(genres))
```

```
##   uniq_movies uniq_users uniq_genres
## 1       10677      69878         797
```

The average rating of the films is obtained with

```
#rating mean
media<-mean(edx$rating)
media
```

```
## [1] 3.512465
```

To see the statistical distribution of the films we took a summary. Besides, We also obtain the number of movies for each rating.

```
#movies valoration
summary(edx$rating)
```

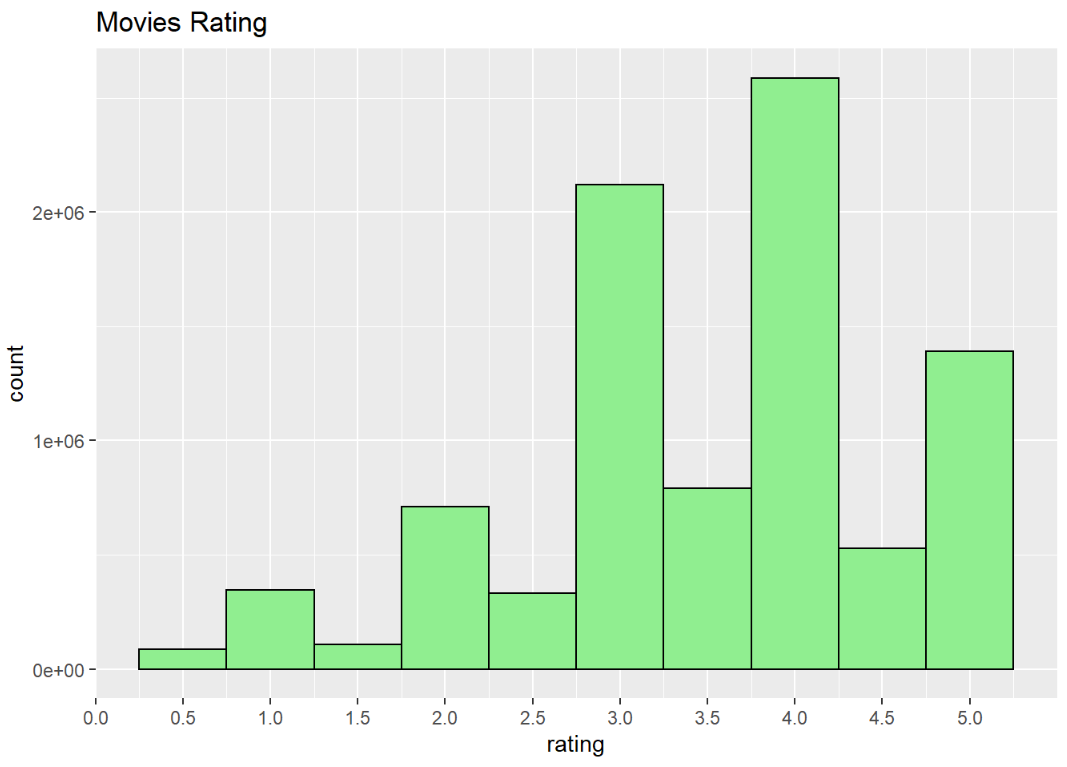
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.500   3.000   4.000   3.512   4.000   5.000
```

```
#movies number by valoration

table_ratings<-table(edx$rating)
table_ratings
```

```
##
##     0.5       1     1.5       2     2.5       3     3.5       4     4.5       5
##   85374  345679  106426  711422  333010 2121240  791624 2588430  526736 1390114
```

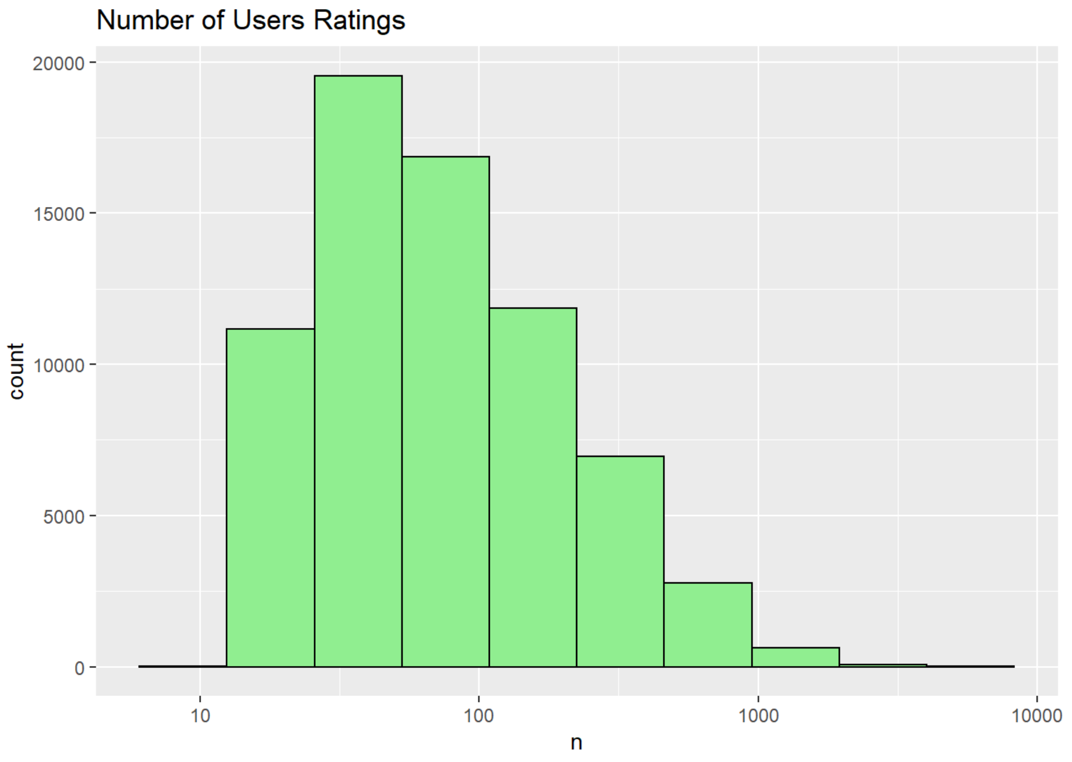
Making a bar graph we see the distribution. And since the integer values present a greater number of valuations.

```
#graphics
ggplot(edx,aes(rating))+
  geom_bar(width =0.50, fill="lightgreen", color="black")+
  scale_x_continuous(breaks = seq(0, 5, by=0.5))+
  labs(title= "Movies Rating")
```



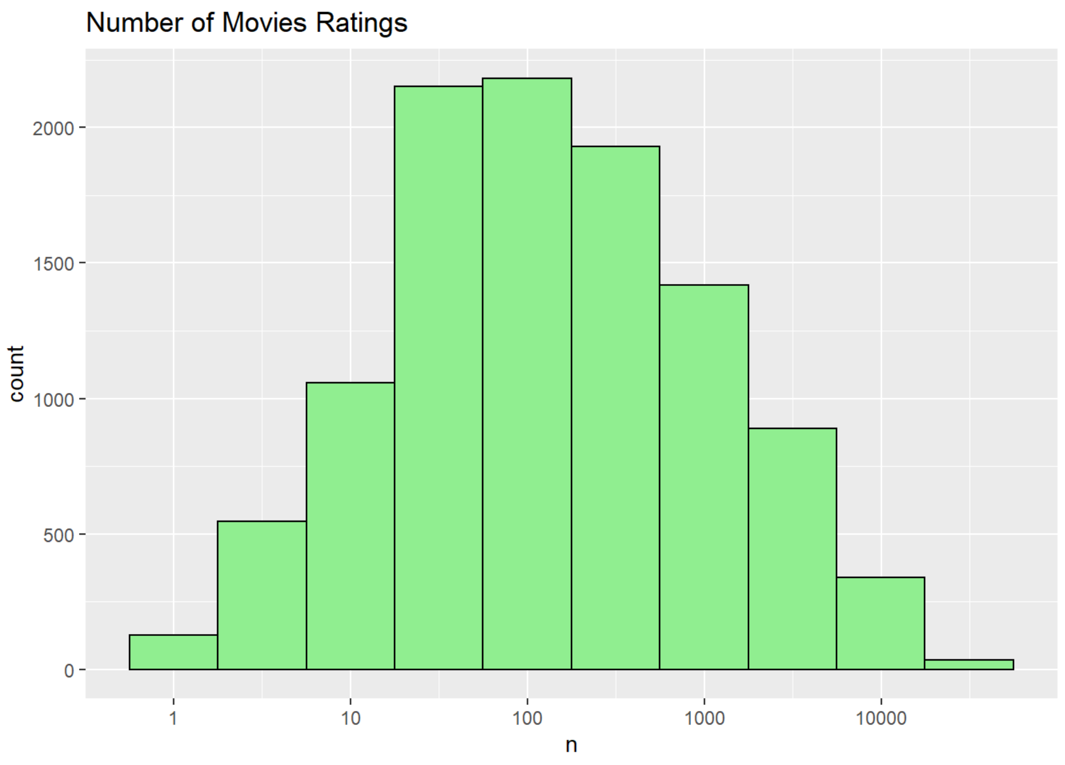
The distribution by users is:

```
#Distribution of Users
edx %>% group_by(userId) %>% summarize(n = n()) %>%
  ggplot(aes(n)) + geom_histogram(fill = "lightgreen",color="black", bins = 10) +
  scale_x_log10() +
  ggtitle("Number of Users Ratings")
```



The distribution by ratings is as follows

```
#Distribution of Movie Ratings
edx %>% group_by(movieId) %>% summarize(n = n()) %>%
  ggplot(aes(n)) + geom_histogram(fill = "lightgreen", color = "black", bins = 10) +
  scale_x_log10() +
  ggtitle("Number of Movies Ratings")
```



It is important to see which films have been the ones that have been valued the most times.

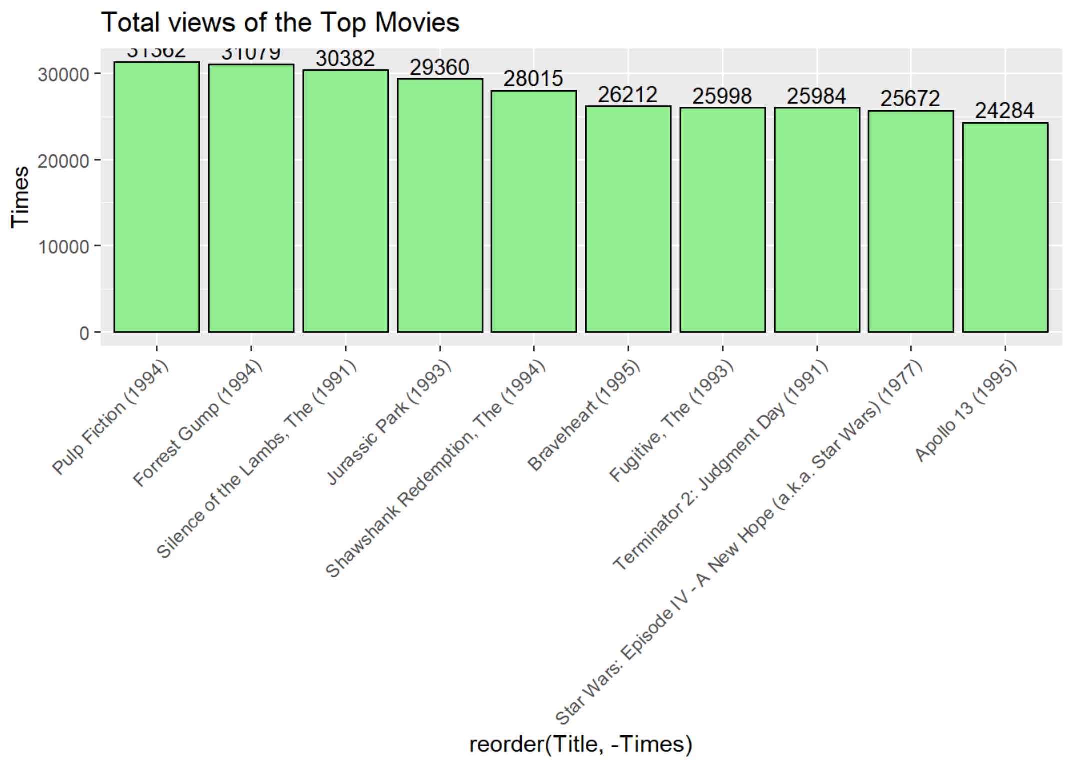
```
#most rated movies

conteo<-data.frame(table(edx$title))
names(conteo)<-c("Title","Times")

conteo<-conteo[order(conteo$Times, decreasing=TRUE),]

head (conteo,10)
```

```
##                                                          Title Times
## 7671                                       Pulp Fiction (1994) 31362
## 3513                                       Forrest Gump (1994) 31079
## 8603                          Silence of the Lambs, The (1991) 30382
## 5151                                      Jurassic Park (1993) 29360
## 8507                          Shawshank Redemption, The (1994) 28015
## 1403                                         Braveheart (1995) 26212
## 3639                                       Fugitive, The (1993) 25998
## 9422                          Terminator 2: Judgment Day (1991) 25984
## 8987 Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977) 25672
## 533                                          Apollo 13 (1995) 24284
```

The films that have been rated the most times have been Pulp Fiction followed by Forrest Gump, and The Silence of the Lambs, all of them above 30,000 ratings.

Graphically ordered from highest to lowest are observed below:

```
#most rated movies graphics

ggplot(conteo[1:10, ], aes(x= reorder(Title,-Times), y=Times))+
  geom_bar(stat="identity", fill="lightgreen",color="black")+
  geom_text(aes(label=Times),vjust=-0.3, size=3.5)+
  theme(axis.text.x = element_text(angle=45, hjust=1))+
  ggtitle("Total views of the Top Movies")
```



And the ones that have been valued the least times, with a single assessment, are the following:

```
#less rated movies

conteo<-data.frame(table(edx$title))
names(conteo)<-c("Title","Times")

conteo<-conteo[order(conteo$Times, decreasing=FALSE),]

head (conteo,10)
```

```
##                                                                  Title
## 13                                    1, 2, 3, Sun (Un, deuz, trois, soleil) (1993)
## 20                                                          100 Feet (2008)
## 98                                                                 4 (2005)
## 172                                               Accused (Anklaget) (2005)
## 177                                                    Ace of Hearts (2008)
## 178                                               Ace of Hearts, The (1921)
## 200 Adios, Sabata (Indio Black, sai che ti dico: Sei un gran figlio di...) (1971)
## 230                                                      Africa addio (1966)
## 282                                                        Aleksandra (2007)
## 728                                           Bad Blood (Mauvais sang) (1986)
##     Times
## 13      1
## 20      1
## 98      1
## 172     1
## 177     1
## 178     1
## 200     1
## 230     1
## 282     1
## 728     1
```

2.2. Data Analysis

The analysis of the recommendation system model will be built step by step the model taking into account the following sequence:

a. Simple model calculated on the mean.

```
# mean rating
media <- mean(edx$rating)
media
```

```
## [1] 3.512465
```

The result of the first model is

```
#testing in validation set

model_number0 <- RMSE(validation$rating, media)
model_number0
```

```
## [1] 1.061202
```

b. Model calculated on the difference in the average rating of each film and the overall average rating (b_i)

```
# calculate b_i using the training set
movie_avgs <- edx %>%
  group_by(movieId) %>%
  summarize(b_i = mean(rating - media))




# predicted ratings
predicted_ratings_bi <- media + validation %>%
  left_join(movie_avgs, by='movieId') %>%
  .$b_i

#testing in validation set
model_number1 <- RMSE(validation$rating,predicted_ratings_bi)
model_number1
```

```
## [1] 0.9439087
```

c. Model calculated with the average user rating (b_u)

```
#b.movie + user effect modelo

#calculate b_u using the training set

user_avgs <- edx %>%
  left_join(movie_avgs, by='movieId') %>%
  group_by(userId) %>%
  summarize(b_u = mean(rating - media - b_i))

#predicted ratings
predicted_ratings_bu <- validation %>%
  left_join(movie_avgs, by='movieId') %>%
  left_join(user_avgs, by='userId') %>%
  mutate(pred = media + b_i + b_u) %>%
  .$pred

#testing in validation set
model_number2 <- RMSE(validation$rating,predicted_ratings_bu)
model_number2
```

```
## [1] 0.8653488
```

d. Model calculated by adding the time effect (b_t)

```
#c.movie + user + time effect

# previously create a copy of validation set , valid, and create the date feature which is the timestamp converted to a datetime object  and  rounded by wee
k.

validation_copy <- validation
validation_copy <- validation_copy %>%
  mutate(date = round_date(as_datetime(timestamp), unit = "week"))

# calculate time effects ( b_t) using the training set
temp_avgs <- edx %>%
  left_join(movie_avgs, by='movieId') %>%
  left_join(user_avgs, by='userId') %>%
  mutate(date = round_date(as_datetime(timestamp), unit = "week")) %>%
  group_by(date) %>%
  summarize(b_t = mean(rating - media - b_i - b_u))

# predicted ratings
predicted_ratings_bt <- validation_copy %>%
  left_join(movie_avgs, by='movieId') %>%
  left_join(user_avgs, by='userId') %>%
  left_join(temp_avgs, by='date') %>%
  mutate(pred = media + b_i + b_u + b_t) %>%
  .$pred

#testing in validation set
model_number3 <- RMSE(validation_copy$rating,predicted_ratings_bt)
model_number3
```

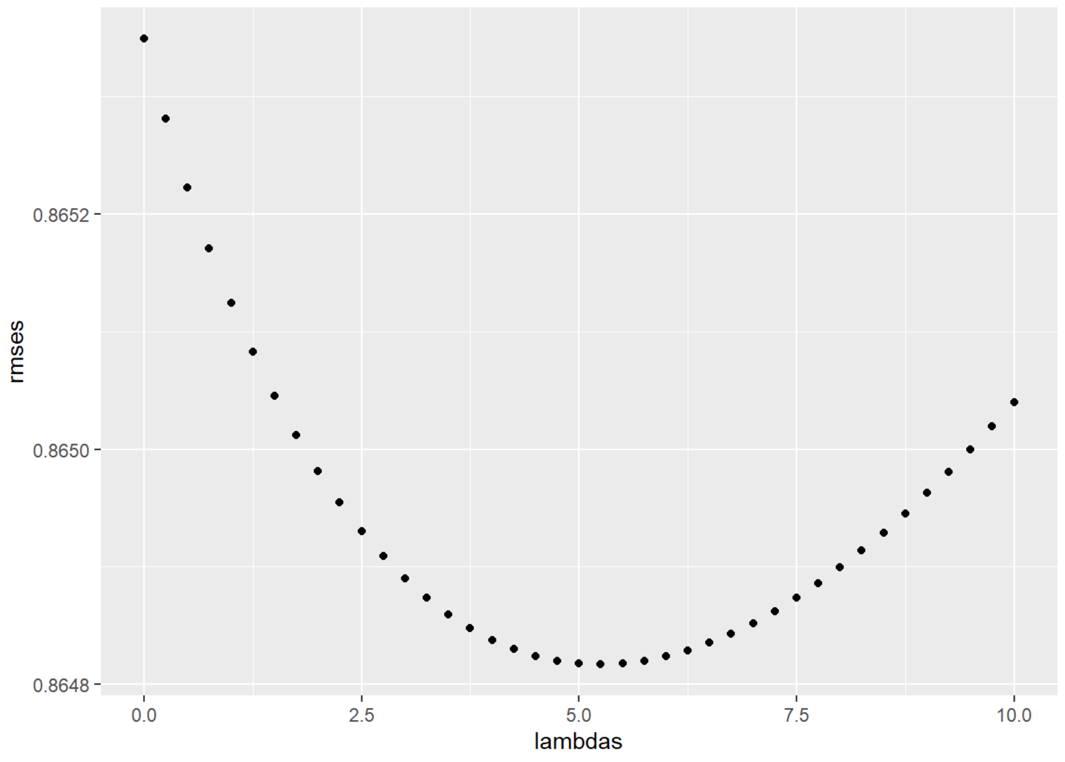
```
## [1] 0.8652511
```

e. Regularized model with user average optimizing lambda

```
#regularization movie + user effect model

lambdas <- seq(0, 10, 0.25)

rmses <- sapply(lambdas, function(l){

  mu <- mean(edx$rating)

  b_i <- edx %>%
    group_by(movieId) %>%
    summarize(b_i = sum(rating - media)/(n()+l))

  b_u <- edx %>%
    left_join(b_i, by="movieId") %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_i - media)/(n()+l))

  predicted_ratings <-
    validation %>%
    left_join(b_i, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    mutate(pred = media + b_i + b_u) %>%
    pull(pred)
  return(RMSE(predicted_ratings, validation$rating))
})
```

The graphical representation of the curve is as follows.

```
#graphic lambda
qplot(lambdas, rmses)
```



The value of lambda that minimizes RMSE is

```
lambdas[which.min(rmses)]
```

```
## [1] 5.25
```

The result of the model on the validation set is

```
#testing in validation set
model_number4 <- min(rmses)
model_number4
```

```
## [1] 0.864817
```

3. Results

The summary of the final values for each model is shown in the following table.

```
resultados_totales <- data.frame(methods=c("Simple media","Movie effect (b_i)","Movie + user effects (b_u)","Movie + user + time effects (b_t)", "Regulariz
ed Movie + User Effect Model"),
                                 rmse = c(model_number0, model_number1, model_number2, model_number3, model_number4))

kable(resultados_totales) %>%
  kable_styling(bootstrap_options = "striped" , full_width = F , position = "center") %>%
  kable_styling(bootstrap_options = "bordered", full_width = F , position ="center") %>%
  column_spec(1,bold = T ) %>%
  column_spec(2,bold =T ,color = "black" , background ="lightgreen")
```

methods	rmse
Simple media	**1.0612018**
Movie effect (b_i)	**0.9439087**
Movie + user effects (b_u)	**0.8653488**
Movie + user + time effects (b_t)	**0.8652511**
Regularized Movie + User Effect Model	**0.8648170**

The lowest value obtained is 0.8648170 with the regularization model using a lambda parameter of 5.25

This value meets the target set by being below 0.86490.

Final Conclusions

Recommendation systems are having an extraordinary boom in e-commerce. Finding the most appropriate strategy and reaching the potential customer positively influences sales and, ultimately, business success in the eCommerce sector.

The project has been developed in accordance with the provisions of Professor Irizarry's reference book.

As a summary of the report, it can be noted that as different effects have been incorporated into the simple model of the use of the average, the RMSE has been falling from the value of 1.0612018 (a very bad value because it is higher than 1) to 0.8648170, which represents an improvement of 18.51% compared to the initial model.

It is feasible to continue making different assumptions and apply different effects in order to improve the result obtained, and that line of research can be followed by future students.

References

Irizarry, R. (2020). Introducción a la ciencia de datos. Análisis de datos y algoritmos de predicción con R. Harvard University. Mendoza, G., Laureano, Y., & Pérez de Celis, M. (2019). Métricas de similaridad y evaluación para sistemas de recomendación de filtrado colaborativo. RITI Journal Vol. 7, 224-240. Vozalis, E., & Margaritis, K. (2003). Analysis or Recommender Systems. In Proceedings or the 6th Hellenic European Conference on Computeer Mathematica and its Applications (HERCMA-2003). Athens.