Title: Movielens

Author: Monica Bustamante

Date: May/27/2020

The Movie Lens is a project to develop and train algoritmo the analysis the customers preferences in an overviwe of the data, analysis, results and conclusions.

The Methods to used for analysis consist of preparing the data: Cleaning, exploration, visualization, a

1 INSTALL PACKAGES, LIBRARIES

```
#INSTALL PACKAGES AND LIBRARIES
list.of.packages <- c("lubridate","stringi",
                      "lattice", "tidyverse", "caret",
                      "tidyr","stringr","ggplot2",
                      "readr")
new.packages <- list.of.packages[!(list.of.packages %in%
                                  installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
```

```
#Install packages
install.packages("rmarkdown")
```

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

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

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

```
#Install libraries
library(ggplot2)
library(readr)
library(lubridate)
library(stringi)
library(stringi)
library(tidyverse)
library(caret)
library(tidyr)
library(stringr)
```

2 DOWNLOAD DATA SET, SPLIT AND MUTATE.

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

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

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

```
 #Mutate, rename title
 movies <- as.data.frame(movies) %>% mutate(movieId = as.numeric(levels(movieId))[movieId]
                                            title = as.character(title),
                                            genres = as.character(genres))
```

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

3 VALIDATION AND TRAIN DATA SET

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

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

```

```
#validation dataset
validation  <- validation %>% select(-rating)
```

GENERAL QUESTIONS

How many rows and columns are there in the edx dataset?

```
#To see more information about the dataset
head(edx, 5)
```

A data.frame: 5 × 6

	userId	movieId	rating	timestamp	title	genres
	<int>	<dbl>	<dbl>	<int>	<chr>	<chr>
1	1	122	5	838985046	Boomerang (1992)	Comedy|Romance
2	1	185	5	838983525	Net, The (1995)	Action|Crime|Thriller
4	1	292	5	838983421	Outbreak (1995)	Action|Drama|Sci-Fi|Thriller
5	1	316	5	838983392	Stargate (1994)	Action|Adventure|Sci-Fi
6	1	329	5	838983392	Star Trek: Generations (1994)	Action|Adventure|Drama|Sci-Fi

```
#Dimension Dataset
dim(edx)
```

```
9000055 · 6
```

```
str(edx)
```

```
'data.frame':   9000055 obs. of  6 variables:
 $ userId   : int  1 1 1 1 1 1 1 1 1 1 ...
 $ movieId  : num  122 185 292 316 329 355 356 362 364 370 ...
 $ rating   : num  5 5 5 5 5 5 5 5 5 5 ...
 $ timestamp: int  838985046 838983525 838983421 838983392 838983392 838984474 838983653
 $ title    : chr  "Boomerang (1992)" "Net, The (1995)" "Outbreak (1995)" "Stargate (199
 $ genres   : chr  "Comedy|Romance" "Action|Crime|Thriller" "Action|Drama|Sci-Fi|Thrille
```

```
#General information about dataset
summary(edx)
```

```
     userId         movieId          rating        timestamp        
 Min.   :    1   Min.   :    1   Min.   :0.500   Min.   :7.897e+08  
 1st Qu.:18124   1st Qu.:  648   1st Qu.:3.000   1st Qu.:9.468e+08  
 Median :35738   Median : 1834   Median :4.000   Median :1.035e+09  
 Mean   :35870   Mean   : 4122   Mean   :3.512   Mean   :1.033e+09  
 3rd Qu.:53607   3rd Qu.: 3626   3rd Qu.:4.000   3rd Qu.:1.127e+09  
 Max.   :71567   Max.   :65133   Max.   :5.000   Max.   :1.231e+09  
    title              genres         
 Length:9000055     Length:9000055    
 Class :character   Class :character  
 Mode  :character   Mode  :character  
```

```
#How many rows and columns are there in the edx dataset
paste('The dataset has',nrow(edx),'rows and',ncol(edx),'columns.')
```

'The dataset has 9000055 rows and 6 columns.'

```
#To see more information about dataset
edx %>% summarise(
  uniq_movies = n_distinct(movieId),
  uniq_users = n_distinct(userId),
  uniq_genres = n_distinct(genres))
```

A data.frame: 1 × 3

uniq_movies	uniq_users	uniq_genres
<int>	**<int>**	**<int>**
10677	69878	797

```
#Mean of rating dataset
rating_mean <- mean(edx$rating)
rating_mean
```

3.51246520160155

How many zeros were given as ratings in the edx dataset?

```
#How many zeros were given as ratings in the edx dataset.
paste(sum(edx$rating == 0), 'ratings and',
      sum(edx$rating == 3),'ratings with 3')
```

'0 ratings and 2121238 ratings with 3'

```
edx %>% filter(rating == 3) %>% tally()
```

```
A
data.frame:
1 × 1
n
<int>
```

How many different movies are in the edx dataset?

```
#How many different movies are in the edx dataset
n_distinct(edx$movieId)
```

```
10677
```

```
edx %>% summarize(n_movies = n_distinct(movieId))
```

```
A
data.frame:
1 × 1
n_movies
<int>
10677
```

How many different users are in the edx dataset?

```
#How many different users are in the edx dataset. n_distinct or lenght
n_distinct(edx$userId)
```

```
69878
```

```
edx %>% summarize(n_users = n_distinct(userId))
```

```
A
data.frame:
1 × 1
n_users
<int>
69878
```

How many movie ratings are in each of the following genres in the edx

```
# str_detect
genres = c("Drama", "Comedy", "Thriller", "Romance")
sapply(genres, function(g) {
    sum(str_detect(edx$genres, g))
})
```

```
})

# separate_rows, much slower!
edx %>% separate_rows(genres, sep = "\\|") %>%
    group_by(genres) %>%
    summarize(count = n()) %>%
    arrange(desc(count))
```

```
Drama:      3910127 Comedy:      3540930 Thriller:      2325899 Romance:      1712100
```

```
#Movie ratings by Drama. str_detect Detect The Presence Or Absence Of A Pattern In A Strin
drama <- edx %>% filter(str_detect(genres,"Drama"))
paste('Drama has',nrow(drama),'movies')
```

```
'Drama has 3910127 movies'
```

```
#Movie ratings by Comedy
comedy <- edx %>% filter(str_detect(genres,"Comedy"))
paste('Comedy has',nrow(comedy),'movies')
```

```
'Comedy has 3540930 movies'
```

```
##Movie ratings by Thriller
thriller <- edx %>% filter(str_detect(genres,"Thriller"))
paste('Thriller has',nrow(thriller),'movies')
```

```
'Thriller has 2325899 movies'
```

```
#Movie ratings by Romance
romance <- edx %>% filter(str_detect(genres,"Romance"))
paste('Romance has',nrow(romance),'movies')
```

```
'Romance has 1712100 movies'
```

Which movie has the greatest number of ratings?

```
#Greatest number of ratings. Arrange rows by variables
edx %>% group_by(title) %>%
summarise(number = n()) %>%
arrange(desc(number))
```

A tibble: 10676 × 2

title	number
<chr>	<int>
Pulp Fiction (1994)	31362
Forrest Gump (1994)	31079
Silence of the Lambs, The (1991)	30382
Jurassic Park (1993)	29360
Shawshank Redemption, The (1994)	28015
Braveheart (1995)	26212
Fugitive, The (1993)	25998
Terminator 2: Judgment Day (1991)	25984
Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)	25672
Apollo 13 (1995)	24284
Batman (1989)	24277
Toy Story (1995)	23790
Independence Day (a.k.a. ID4) (1996)	23449
Dances with Wolves (1990)	23367
Schindler's List (1993)	23193
True Lies (1994)	22823
Star Wars: Episode VI - Return of the Jedi (1983)	22584
12 Monkeys (Twelve Monkeys) (1995)	21891
Usual Suspects, The (1995)	21648
Fargo (1996)	21395
Speed (1994)	21361
Aladdin (1992)	21173
Matrix, The (1999)	20908
Star Wars: Episode V - The Empire Strikes Back (1980)	20729
Seven (a.k.a. Se7en) (1995)	20311
American Beauty (1999)	19950
Raiders of the Lost Ark (Indiana Jones and the Raiders of the Lost Ark) (1981)	19678
Back to the Future (1985)	19034
Mission: Impossible (1996)	18992
Ace Ventura: Pet Detective (1994)	18959
⋮	⋮
Please Vote for Me (2007)	1
Quarry, The (1998)	1
Quiet City (2007)	1
Relative Strangers (2006)	1
Ring of Darkness (2004)	1
Rockin' in the Rockies (1945)	1
Säg att du älskar mig (2006)	1
Shadows of Forgotten Ancestors (1964)	1
Small Cuts (Petites coupures) (2003)	1
Splinter (2008)	1
Stacy's Knights (1982)	1
Stone Angel, The (2007)	1
Strange Planet (1999)	1

Strange Planet (1999)	1
Sun Alley (Sonnenallee) (1999)	1
Sun Shines Bright, The (1953)	1
Symbiopsychotaxiplasm: Take One (1968)	1
Tattooed Life (Irezumi ichidai) (1965)	1
Testament of Orpheus, The (Testament d'Orphée) (1960)	1
Tokyo! (2008)	1
Train Ride to Hollywood (1978)	1
Twice Upon a Time (1983)	1
Uncle Nino (2003)	1

What are the five most given ratings in order from most to least?

When Time Ran Out... (a.k.a. The Day the World Ended) (1980)	1

```
#Sort a variable in descending order.
edx %>% group_by(rating) %>%
summarize(count = n()) %>%
top_n(5) %>%
  arrange(desc(count))
```

```
Selecting by count
```

A tibble: 5 × 2

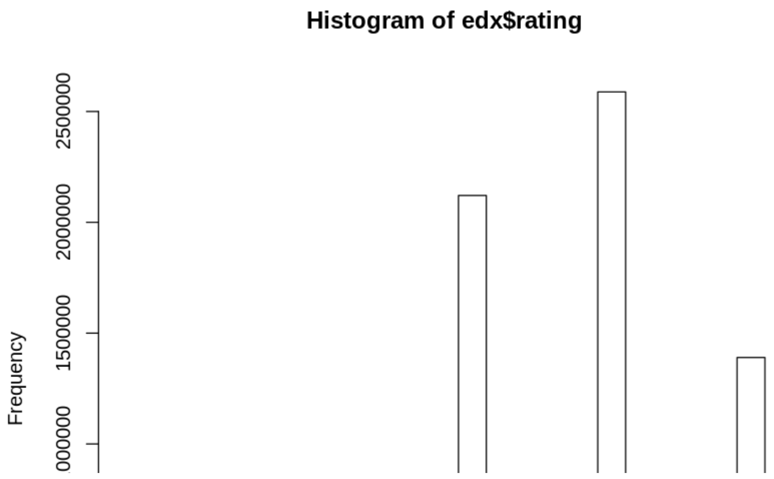
rating	count
<dbl>	<int>
4.0	2588430
3.0	2121240
5.0	1390114
3.5	791624
2.0	711422

```
head(sort(-table(edx$rating)),5)
```

```
       4        3        5      3.5        2 
-2588430 -2121240 -1390114  -791624  -711422 
```

```
hist(edx$rating)
summary(edx$rating)
```

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

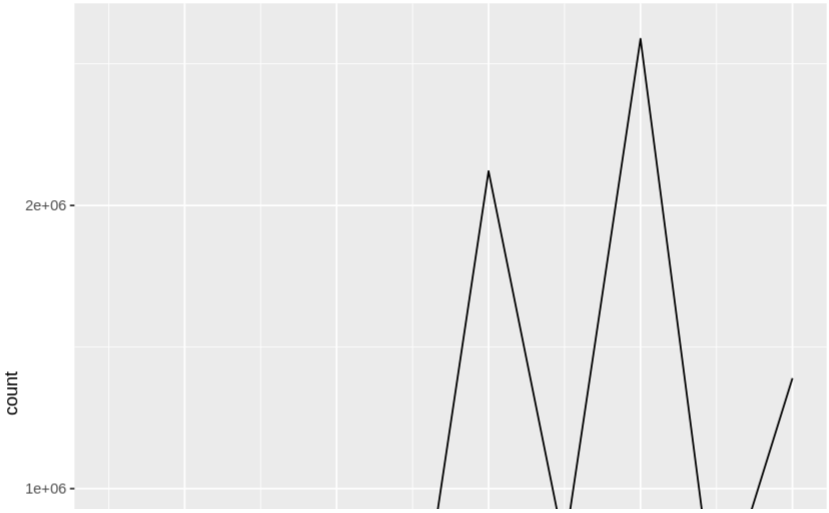


True or False: In general, half star ratings are less common than whole fewer ratings of 3.5 than there are ratings of 3 or 4, etc.).

```
#Rating movies
rating4 <- table(edx$rating)["4"]
rating35 <- table(edx$rating)["3.5"]
rating3 <- table(edx$rating)["3"]

Result <- (rating35 < rating3 && rating35 < rating4)

print(Result)

rm(rating3, rating35,  rating4, Result)
```

```
[1] TRUE
```

```
#Graphic Rating movies
edx %>%
  group_by(rating) %>%
  summarize(count = n()) %>%
  ggplot(aes(x = rating, y = count)) +
  geom_line()
```



4 MODELING

Predicted movie ratings and calculates RMSE.

Movie rating predictions will be compared to the true ratings in the validation set using RMSE

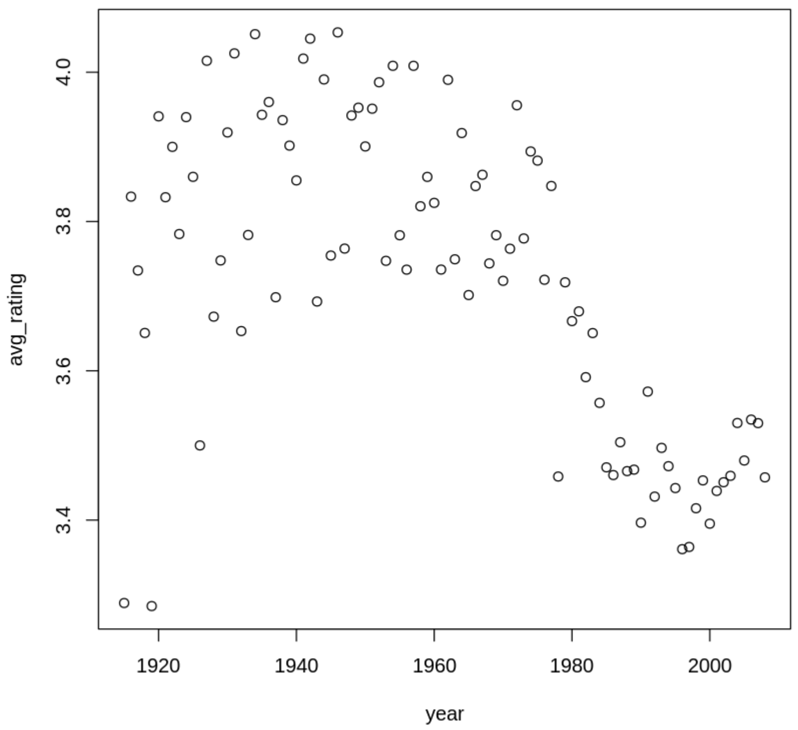
```
data <- movies %>% separate_rows(genres, sep ="\\|")
DAT.aggregate <- aggregate(formula = cbind(n = 1:nrow(dat)) ~ genres, data = data, FUN = l
```

```
#Size of dataset
movielens <- left_join(ratings,
                       movies, by = "movieId")
nrow(movielens)
```

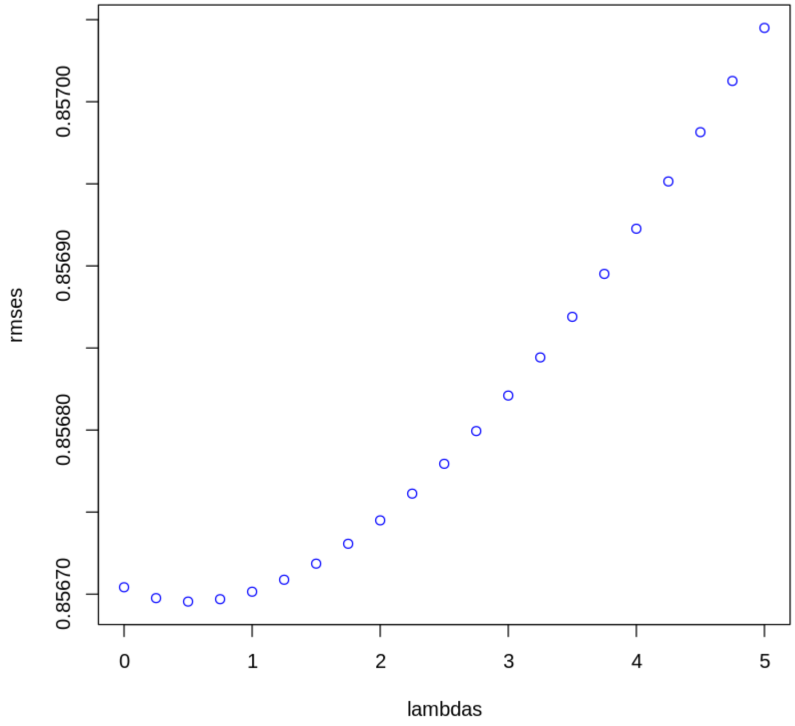
```
10000054
```

```
#Creates Year column.
edx <- edx %>%
  mutate(title = str_trim(title)) %>%
  extract(title, c("title_tmp", "year"),
          regex = "^(.*) \\(([0-9 \\-]*)\\)$",
          remove = F) %>%
  mutate(year = if_else(str_length(year) > 4,
                        as.integer(str_split(year, "-",
                                             simplify = T)[1]),
                        as.integer(year))) %>%
  mutate(title = if_else(is.na(title_tmp), title, title_tmp)) %>%
  select(-title_tmp)  %>%
  mutate(genres = if_else(genres == "(No Genres Listed)",
                          `is.na<-`(genres), genres))
validation <- temp %>%
  semi_join(edx, by = "movieId") %>%
  semi_join(edx, by = "userId")
  semi_join(edx, by = "userId")
```

```
avg_ratings <- edx %>%
group_by(year) %>%
summarise(avg_rating = mean(rating))
plot(avg_ratings)
```



```
#Root Mean Square Error
RMSE <- function(true_ratings, predicted_ratings){
        sqrt(mean((true_ratings - predicted_ratings)^2))
      }

lambdas <- seq(0, 5, 0.25)
rmses <- sapply(lambdas,function(l){
  mu <- mean(edx$rating) #The mean of ratings from training set

  Movie_effect <- edx %>%  #Adjust mean by movie effect
    group_by(movieId) %>%
    summarize(Movie_effect = sum(rating - mu)/(n()+l))

  Movie_user <- edx %>% #Ajdust mean by movie effect and user
    left_join(Movie_effect, by="movieId") %>%
    group_by(userId) %>%
    summarize(Movie_user = sum(rating - Movie_effect - mu)/(n()+l))

  predicted_ratings <-
    edx %>%
    left_join(Movie_user, by = "userId") %>%
    left_join(Movie_user, by = "userId") %>%
    left_join(Movie_effect, by = "movieId") %>%
    mutate(pred = mu + Movie_effect + Movie_user) %>%
    .$pred #Predict ratings

  return(RMSE(predicted_ratings, edx$rating))
})
plot(lambdas, rmses,
    col = "blue")
```



```
#Calculate Lambda optimal RMSE
lambda <- lambdas[which.min(rmses)]
paste('RMSE',min(rmses),'Lambda',lambda)
```

```
'RMSE 0.856695492876063 Lambda 0.5'
```

CONCLUSION:

Predict a list of rated movies.

Discovered patterns: as people prefer movies with a medium to high rating. (3 to 5).

The movies preferred by the customers was the end of the 1980 and 1990 periods.