# Movie Rating Prediction Project: Movie Lens Dataset

HarvardX - PH125.9x Data Science: Capstone Course

Mauricio Rabelo Soares

18 julho, 2022

## Introduction

A recommendation system, is a subclass of information filtering system that provide suggestions for items that are most pertinent to a particular user. Typically, the suggestions refer to various decision-making processes, such as what product to purchase, what music to listen to, or what movies to watch. [1] The goal of this project is to create a recommendation system, which the RMSE $< 0.86490$, using all the tools we have learn throughout the multi-part course in HarvardX's Data Science Professional Certificate series. As presented in the Machine Learning course [2] [3], a recommendation systems for a movie use ***ratings*** $(y_{u,i})$ that ***users*** $(u)$ have given to some items, or ***movies*** $(i)$, to make specific recommendations to specific user. In this model, movies for which a high rating is predicted for a given user are then recommended to that user. For this challenge we going to use a subset of dataset provided by the GroupLens. In our case the subset of this data is provided by MovieLens 10M movie ratings. This a stable benchmark dataset with **10 million ratings** applied to **10,000 movies** by **72,000 users**.

## Methods

The methods section explains the process and techniques used in this project, in the first part, then explains the data cleaning used to extract and clean the data, in the second part. In the third part of this section we present the data exploration and visualization of the data to gain some insights. The fourth, and last part of this section, we show the modeling approach used in this project.

### The process and techniques used

Recommendation systems are particularly useful when an individual needs to choose an item from a potentially overwhelming number of items that a service may offer. The recommendations system build in this project are made using the same process and techniques used by the winners of the Netflix challenges[4], which is presented in Chapter 34 Large datasets at Irizarry (2019) [5], and some new approaches[Qiu et al. (2021)][6]

The process begin with the code provided by the HarvardX PH125.9xData Science: Capstone, where it tidies up the data and create the train (`edx`) and validation (`validation`) sets. After the data cleaning we explore the data to gain some insight trough visualization and selected table. This insights is the basis of the modelling approach of this project, that starts building the simplest possible recommendation system:

[1] https://en.wikipedia.org/wiki/Recommender_system
[2] HarvardX - PH125.8x Data Science: Machine Learning https://www.edx.org/course/data-science-machine-learning
[3] 34.7 Recommendation systems - https://rafalab.github.io/dsbook/large-datasets.html#recommendation-systems
[4] https://en.wikipedia.org/wiki/Netflix_Prize
[5] https://rafalab.github.io/dsbook/large-datasets.html#fnref112
[6] https://github.com/Airborne737/MovieLens_Harvard

predict the same rating for all movies regardless of user, and ends with recommendation system that use parallel matrix factorization, the product of two matrices of lower dimensions, $P_{n\times k}$ and $Q_{n\times k}$ were the $P$ is user matrix, and $Q$ is the movie matrix, the `recosystem`.

## Data cleaning

For this project we going to use a subset of dataset provided by the GroupLens. In our case the subset of this data is provided by MovieLens 10M movie ratings. This a stable benchmark dataset with **10 million ratings** applied to **10,000 movies** by **72,000 users**. This version of the dataset was released in January of 2009[7]. The code provided by HarvardX load the library, then download the file from grouplens site and read the file. After read the file the code create the data frame `movies` and `movielens`. With the `movielens` data frame the code create the `edx` and `validation` datasets that will be used to train and test our final algorithm.

```
############################################################
# Create edx set, validation set (final hold-out test set)
############################################################

# Note: this process could take a couple of minutes

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

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

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

dl <- tempfile()
download.file("https://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)`
test_index <- createDataPartition(y = movielens$rating, times = 1, p = 0.1, list = FALSE)
edx <- movielens[-test_index,]
temp <- movielens[test_index,]
```

[7] https://grouplens.org/datasets/movielens/10m/

```
# 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)
```

## Data exploration and visualization

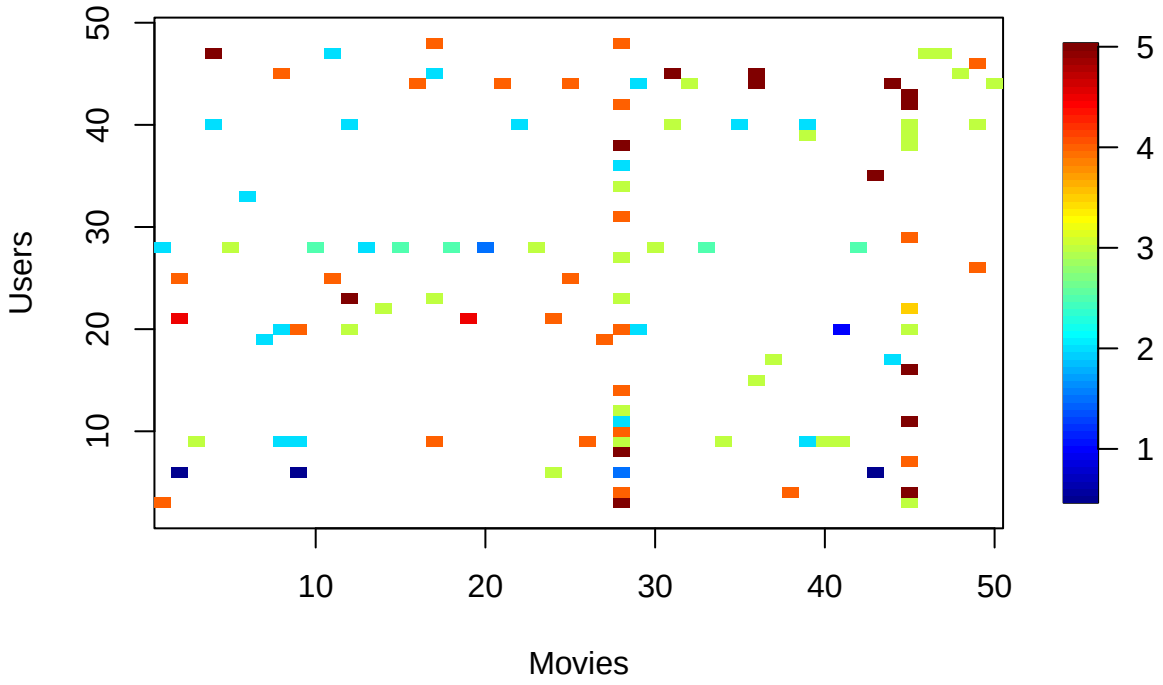
The exploration and visualization of the data provide insightful information about the users, the movies and ratings. The first 5 rows of the dataset that we use in this project is presented in table 1. The columns `movieId`, `userId`and `rating`are the variable of interest.

```
library(fields)
library(tidyverse)
library(knitr)
library(kableExtra)
library(grid)
library(ggplot2)
library(lattice)
library(gridExtra)
dl <- tempfile()
download.file("https://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")

rm(dl, ratings, movies)
```

```
knitr::kable(head(movielens %>% as_tibble(),5),
             caption = "The first 5 rows of the dataset, movielens",
             digits = 2,
             align = "cccccc",
             position = "b") %>%
  kable_styling(latex_options = "scale_down")
```

The variables, `movieId`, `userId` and `rating`, that is going to be used to build the model is presented as a matrix in the figure 1. The figure have the movies (`moveId`) in the x axis, the users in the y axis (`userId`), and the respective rating (`rating`). The matrix, extract from a sample of 50 users and 50 movies, provide some insights about the behavior of some users, the preference for some movies, and the sparse of the matrix. The goal of this project is to fill the blank spaces with a rate.

```
#matrix users x movies x rating
users <- sample(unique(movielens$userId), 50)
movielens %>% filter(userId %in% users) %>%
  select(userId, movieId, rating) %>%
  spread(movieId, rating) %>% select(sample(ncol(.), 50)) %>%
  as.matrix() %>% t(.) %>%
  image.plot(1:50, 1:50,. , xlab="Movies", ylab="Users")
```

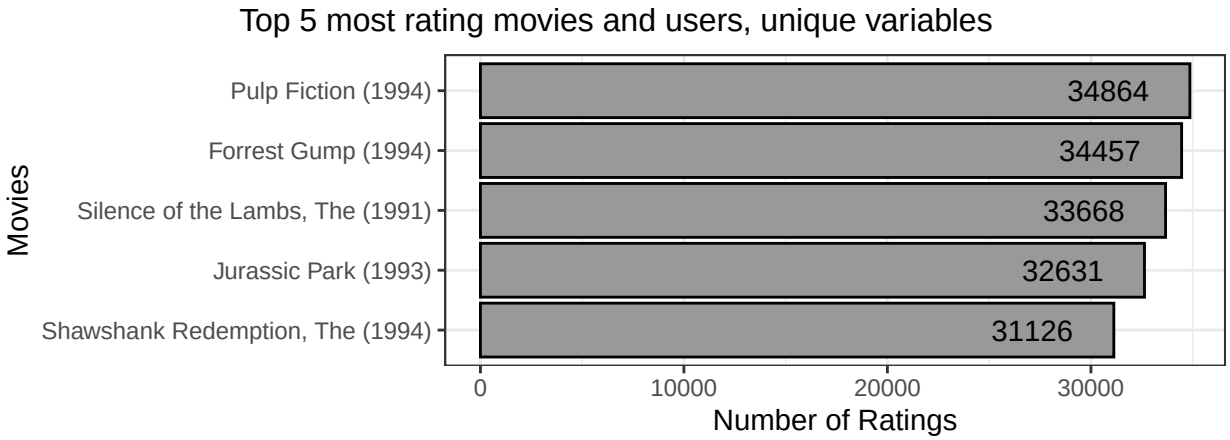



To see a potential flaw in the data we make a slice of the top 5 most rated movies and users. The unique users that provided ratings, the unique movies that were rated and the unique rating provided by a unique user to a unique movie, are presented to illustrate the possible matrix *users* $\times$ *movies* 69,878 x 10,677 = XXXX and the realized matrix 9000055. The extremes values confirm that some users are much more actives than others, rating more than 50% of the total unique movies, and some movies have been rated for more than 1/3 of the total unique users.

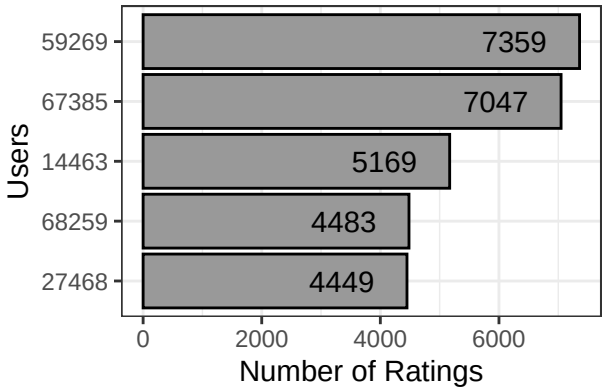
Table 1: The first 5 rows of the dataset, movielens

| userId | movieId | rating | timestamp | title | genres |
|---|---|---|---|---|---|
| 1 | 122 | 5 | 838985046 | Boomerang (1992) | Comedy\|Romance |
| 1 | 185 | 5 | 838983525 | Net, The (1995) | Action\|Crime\|Thriller |
| 1 | 231 | 5 | 838983392 | Dumb & Dumber (1994) | Comedy |
| 1 | 292 | 5 | 838983421 | Outbreak (1995) | Action\|Drama\|Sci-Fi\|Thriller |
| 1 | 316 | 5 | 838983392 | Stargate (1994) | Action\|Adventure\|Sci-Fi |

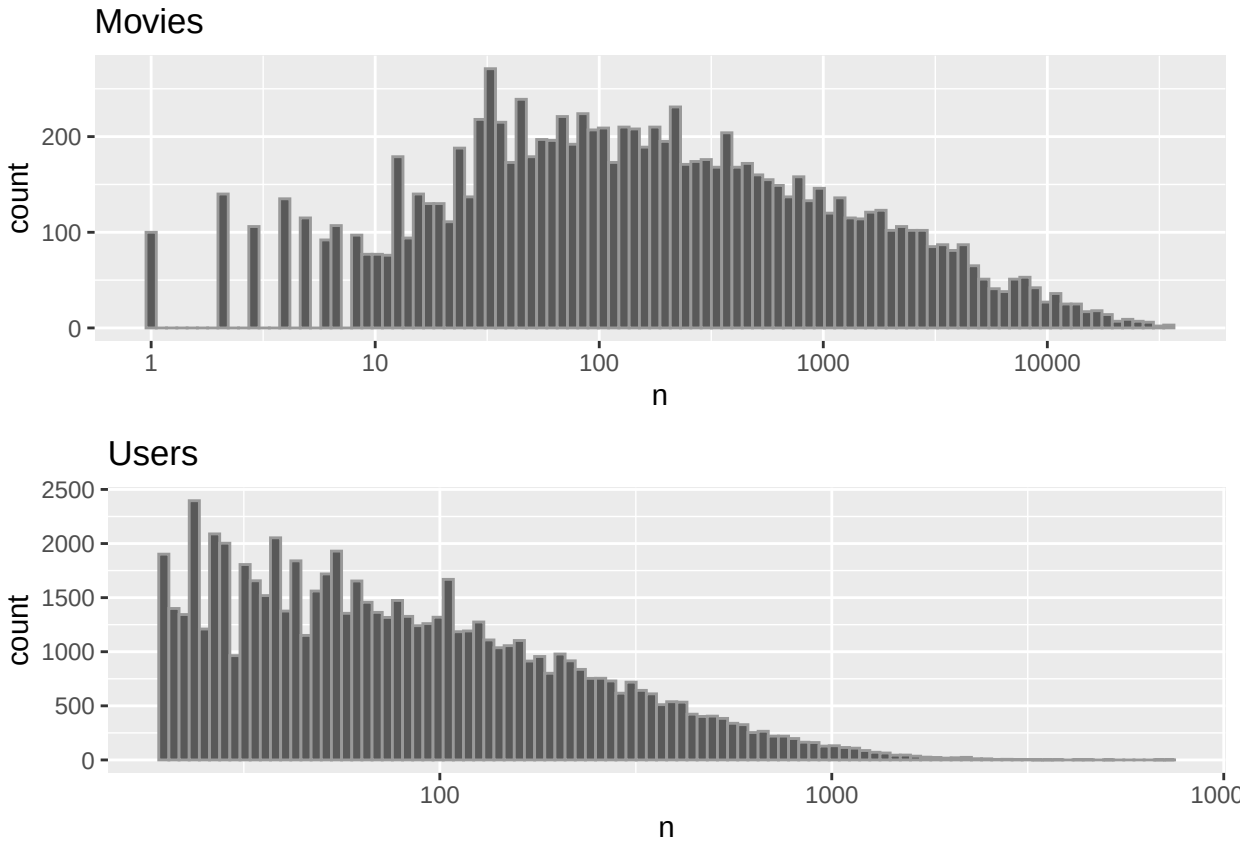
```r
# Unique users, movies, rating
p0 <- tableGrob(movielens %>% summarize(n_users = n_distinct(userId),
                                        n_movies = n_distinct(movieId),
                                        n_rating = length(rating)))

# Top 5 movies
p1 <- movielens %>%
  group_by(title) %>%
  summarize(count = n()) %>%
  arrange(-count) %>%
  top_n(5, count) %>%
  ggplot(aes(count, reorder(title, count))) +
  geom_bar(color = "black", fill = "#999999", stat = "identity") +
  geom_text(aes(label=count), position=position_dodge(width=0.9), hjust=1.5) +
  xlab("Number of Ratings") +
  ylab("Movies") +
  theme_bw()

# Top 5 users
p2 <- movielens %>%
  group_by(userId) %>%
  summarize(count = n()) %>%
  arrange(-count) %>%
  top_n(5, count) %>%
  ggplot(aes(count, reorder(userId, count))) +
  geom_bar(color = "black", fill = "#999999", stat = "identity") +
  geom_text(aes(label=count), position=position_dodge(width=0.9), hjust=1.5) +
  xlab("Number of Ratings") +
  ylab("Users") +
  theme_bw()

#Top 5 most rating movies and users, unique variables
gridExtra::grid.arrange(p1,
               arrangeGrob (p0, p2, ncol = 2),
               nrow = 2,
               top = "Top 5 most rating movies and users, unique variables")
```




| | n_users | n_movies | n_rating |
|---|---|---|---|
| *1* | 69878 | 10677 | 10000054 |




The dataset distribution presented trough histograms provide some insights about the general proprieties of the data. As showed in the slice before some movies get rated more than others, and some user are more active than others.




## Modeling approach

Loss function

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{u,i}\left(\hat{y}_{u,i} - y_{u,i}\right)^2}$$

A first model

$$Y_{u,i} = \mu + \varepsilon_{u,i}$$

Modeling movie effects

$$Y_{u,i} = \mu + b_i + \varepsilon_{u,i}$$

User effects

$$Y_{u,i} = \mu + b_i + b_u + \varepsilon_{u,i}$$

Penalized least squares

$$\sum_{u,i}\left(y_{u,i} - \mu - b_i\right)^2 + \lambda\sum_i b_i^2$$

$$\hat{b}_i(\lambda) = \frac{1}{\lambda + n_i}\sum_{u=1}^{n_i}\left(Y_{u,i} - \hat{\mu}\right)$$

Choosing the penalty terms

$$\sum_{u,i} \left(y_{u,i} - \mu - b_i - b_u\right)^2 + \lambda \left(\sum_i b_i^2 + \sum_u b_u^2\right)$$

A typical solution for P and Q is given by the following optimization problem (Chin, Zhuang, et al. 2015a, 2015b):

$$\min_{P,Q} \sum_{(u,v)\in R} \left[f(p_u, q_v; r_{u,v}) + \mu_P ||p_u||_1 + \mu_Q ||q_v||_1 + \frac{\lambda_P}{2}||p_u||_2^2 + \frac{\lambda_Q}{2}||q_v||_2^2\right]$$

where (u,v) are locations of observed entries in R, ru,v is the observed rating, f is the loss function, and P, Q, P, Q are penalty parameters to avoid overfitting.

# Results

Section that presents the modeling results and discusses the model performance

# Conclusion

Section that gives a brief summary of the report, its limitations and future work

# References


bnwicks - https://github.com/bnwicks/Capstone/blob/master/MovieLens.R

Airborne737 - https://www.rpubs.com/Airborne737/movielens

R Markdown: Gerando relatórios usando o R (Parte 1) - https://www.youtube.com/watch?v=obxa5VH4WvY

R Markdown: Gerando relatórios usando o R (Parte 2) - https://www.youtube.com/watch?v=tcNx0QbDPBo

Irizarry, Rafael A. 2019. *Introduction to Data Science*. Chapman; Hall/CRC. https://doi.org/10.1201/9780429341830.

Qiu, Yixuan, David Cortes, Chih-Jen Lin, Yu-Chin Juan, Wei-Sheng Chin, Yong Zhuang, Bo-Wen Yuan, Meng-Yuan Yang, and other contributors. See file AUTHORS for details. 2021. "Recosystem: Recommender System Using Matrix Factorization." https://CRAN.R-project.org/package=recosystem.