Reader's digest https://www.rd.com/list/classic-family-movies/

MovieLens Recommendation System

Data Science: Capstone!

Capstone Project

Final course Professional Certificate in Data Science

June. 2024

Sandra Lucía Rodríguez Ochoa

1 Table of Contents

Contents

2 Introduction

This is the MovieLens project for Capstone course of the HarvardX Professional Certificate in Data Science program (PH125.9x). A movie recommendation system will be constructed based on MovieLens Database that contains a set of real word movie ratings.

Recommendation system is a class of machine learning that uses data to help predict what people are looking for among an exponentially growing number of options. Netflix, YouTube, Tinder, and Amazon are all examples of recommender systems in use. The systems entice users with relevant suggestions based on the choices they make or analyze similarities between users and/or item interactions.

In this case, the objective is to predict the rating that different users give to a set of different movies, using all the tools we have learn throughout the courses in Data Science series. For the exercise it will be used the version of MovieLens included in the dslabs package that is just a small subset of a much larger dataset with millions of ratings.

The MovieLens is comprised of ratings, ranging from 1 to 5, from 943 users on 1682 movies, along the project development an exploratory review of data and predictors will be performed, training and tests subsets will be used to model a machine learning algorithm that predict ratings. Finally, a validation set will be used to calculate the Root Mean Square Error that will be the metric used for evaluation. The result expects a RMSE less than 0.8649.

3 Executive Summary

Technology and AI are advancing too quickly that perhaps in the today's world we get more recommendations from Artificial Intelligence models than from our friends. The objective of this Capstone project is to build a movie recommendation system, a machine learning algorithm will be modeled to predict the rating that a group of users give to a set of movies.

For this purpose, a subset of the entire MovieLens database, a popular dataset for recommender systems, will be used and partitioned in training and test sets to model the algorithm; at the beginning the data will be explored to identify the predictors and get familiar with the variables and their distributions, then the function will be feed including the

effect of various items, at the end, as validation function, the Root Mean Square error parameter will be calculated as following to evaluate how close the predictions are to the true values, trying to meet the RMSE target: less than 0.8649, finally results and conclusions will be presented including actual limitations and further work.

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

4 Method & Analysis

4.1 Get the data.

MoviLens dataset contains a set of movie ratings from the MovieLens website, a movie recommendation service. This dataset was collected and maintained by GroupLens, a research group at the University of Minnesota.(https://www.tensorflow.org/datasets/catalog/movielens). We will use the 10M version of the MovieLens dataset to make the computation easier.

The code to generate the work and validation datasets were given as a statement of the project; after install the required libraries, the dataset is downloaded from the website: https://grouplens.org/datasets/movielens/10m/. Two files: ratings file containing the rating information for each UserId applied to a MovieId and movie file containing information relative to Movies: MovieId, Tittle and Genres, both files were imported, converted to Data Frames and then joined together in a single data frame to facilitate handling.

"Final hold-out test" was a partition of 10% of MovieLens data, this subset, will be used for evaluating the RMSE of final algorithm at the end of the project. The rest of the data was contained in an additional dataset called "edx", which will later be partitioned for modeling purposes in training and test sets to design and test your algorithm.

4.2 Data exploration.

First we will explore "edx" set:

```
> class(edx)
[1] "data.frame"
> dim(edx)
[1] 9000055       6
> summary(edx)
     userId          movieId          rating       timestamp           title              genres
 Min.   :    1   Min.   :    1   Min.   :0.500   Min.   :7.897e+08   Length:9000055     Length:9000055
 1st Qu.:18124   1st Qu.:  648   1st Qu.:3.000   1st Qu.:9.468e+08   Class :character   Class :character
 Median :35738   Median : 1834   Median :4.000   Median :1.035e+09   Mode  :character   Mode  :character
 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
> range(edx$rating)
[1] 0.5 5.0
```

edx set consists in a Data Frame with 9,000,055 observations (Rows) and 6 Columns: userId, movieId, rating, timestamp, title and genres, there are 69,878 different users and 10,677 different movies and 797 different combination of genres; users rate the movies with calcifications between 0,5 and 5,0, were 5,0 is the best evaluation. The timestamp column contains an integer that represent seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970, also if we explore the column title, we can identify that for each movie the release year is between parenthesis at the end of the title. The column "genres" contains one or more genres related to the movie separated by a "|".

4.3 Data transformation

Two columns were added to edx dataset to make features independent and give them a more comprehensive format. Column "year" will be an integer containing the year the movie was released, and column "year_rating" will contain the year the user gave the rating.

In the rating_year column, we could find ratings since 1995 until 2009 year.

```
> range(edx$year_rating)
[1] 1995 2009
```

Let's see the top ten earliest and latest releases in dataset:

Earliest releases date from 1915 year

	movieId	year	title
	<int>	<int>	<chr>
1	7065	1915	Birth of a Nation, The (1915)
2	7243	1916	Intolerance (1916)
3	62383	1916	20,000 Leagues Under the Sea (1916)
4	8511	1917	Immigrant, The (1917)
5	48374	1917	Father Sergius (Otets Sergiy) (1917)
6	3309	1918	Dog's Life, A (1918)
7	64245	1918	Bell Boy, The (1918)
8	2821	1919	Male and Female (1919)
9	2823	1919	Spiders Part 1: The Golden Lake, The (Die Spinnen, 1. Teil: Der Goldene See) (1919)
10	3132	1919	Daddy Long Legs (1919)

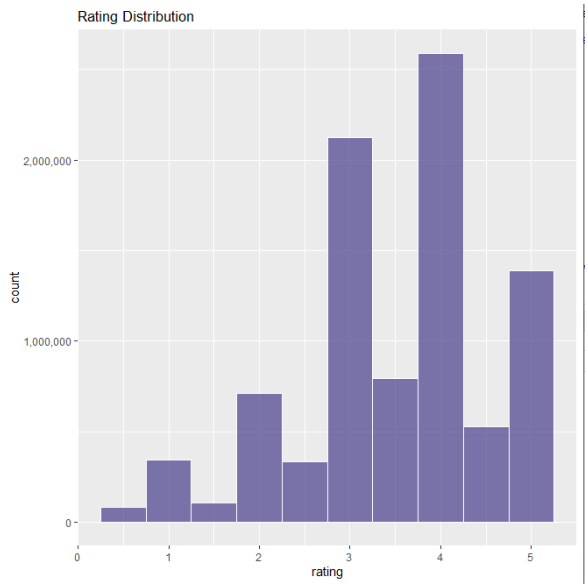
Latest releases in edx dataset date from 2008 year

	movieId	year	title
	<int>	<int>	<chr>
1	64839	2008	Wrestler, The (2008)
2	64957	2008	Curious Case of Benjamin Button, The (2008)
3	64969	2008	Yes Man (2008)
4	64983	2008	Valkyrie (2008)
5	64986	2008	Birds of America (2008)
6	64999	2008	War of the Worlds 2: The Next Wave (2008)
7	65006	2008	Impulse (2008)
8	65088	2008	Bedtime Stories (2008)
9	65126	2008	Choke (2008)
10	65130	2008	Revolutionary Road (2008)

4.4 Data Analysis

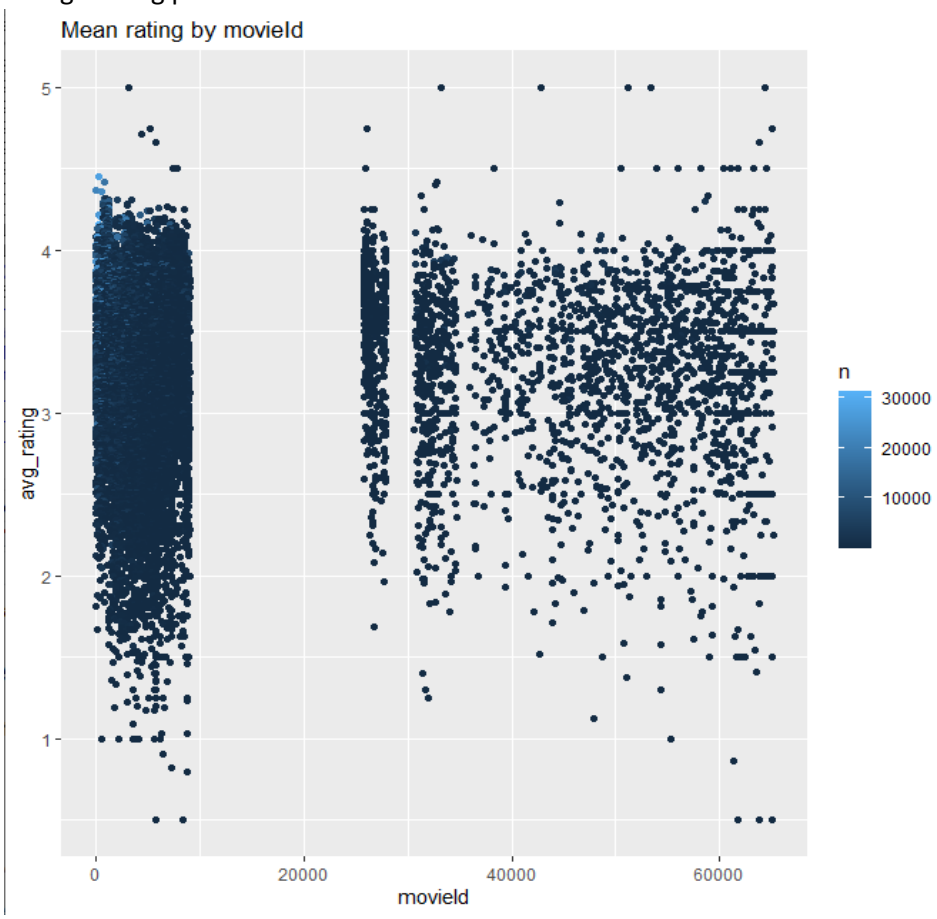
The following graphics show de characteristics of the data features and the relation between them, this kind of analysis help to identify the different bias and effects that each one aports to the rating that is the variable of interest that we want to predict.

- Rating distribution



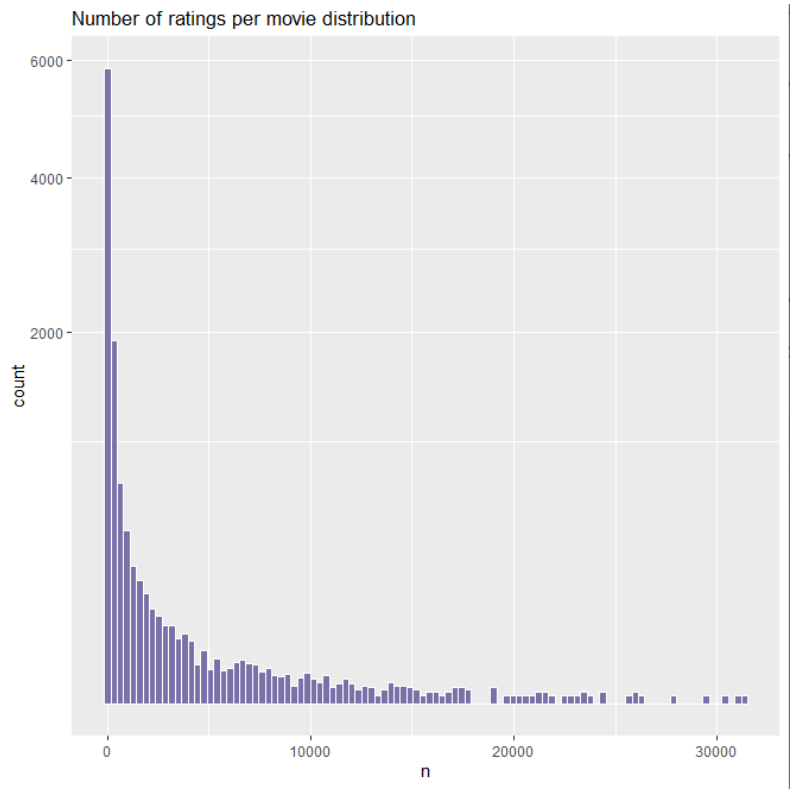
It is possible to observe that the rating used more frequently is the range between 4 and 4.5, we also observe that integer ratings like 2,3,4,5 are more common than ratings with decimal components like 3.5, 4.5 etc.

- Average rating per movie



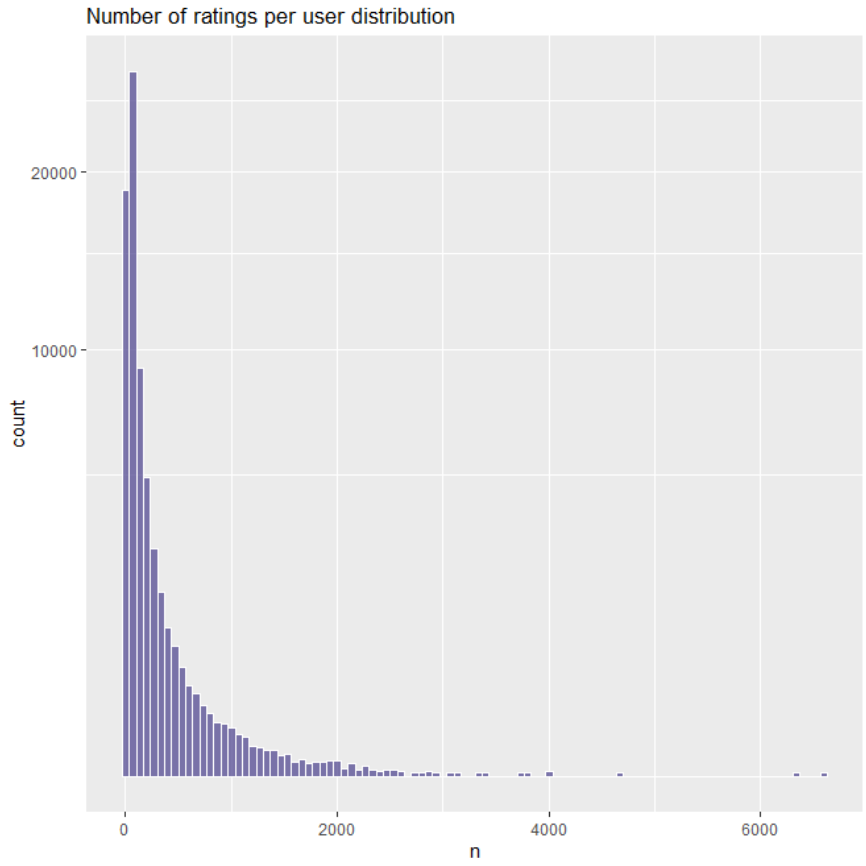
This scatter plot shows the distribution of the average rating per movieId, the color scale shows the number of ratings that each movie had, we can see that there is a lot of variation of ratings and there are more movies with less than 10,000 ratings than more than 10,000 ratings, there are a couple of movies that have close to 30,000 ratings.

- Number of ratings per movie distribution



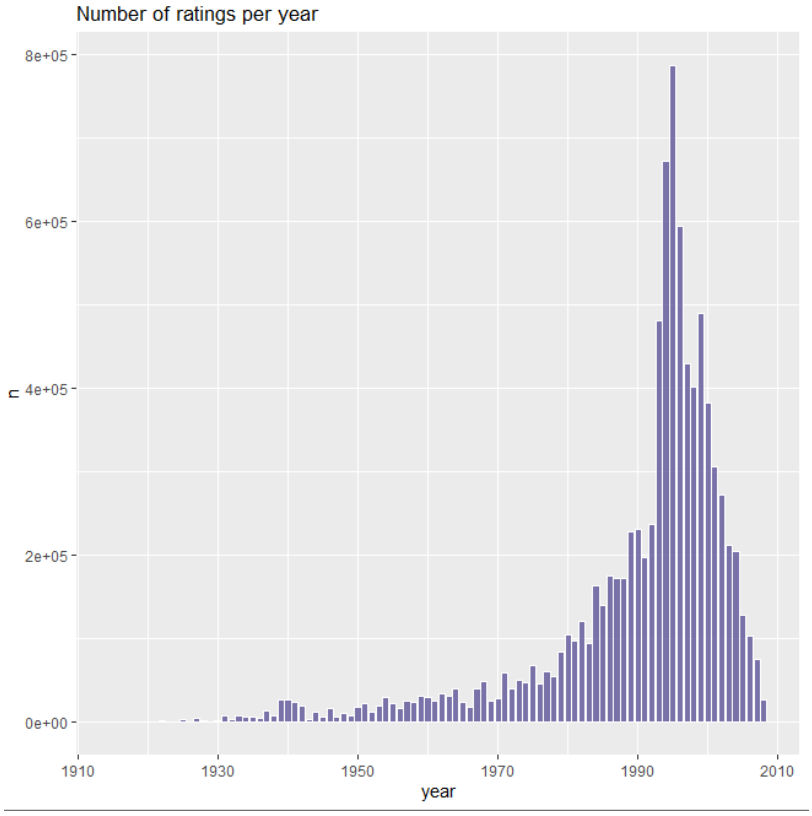
The histogram shows the number of ratings per movie distribution, there are almost 6,000 movies that have less than 500 ratings, most of the movies have few ratings, when predicting results based on a low number of samples the error increases, so regularization seems to be an appropriate technique for this case.

- Number of ratings per user distribution



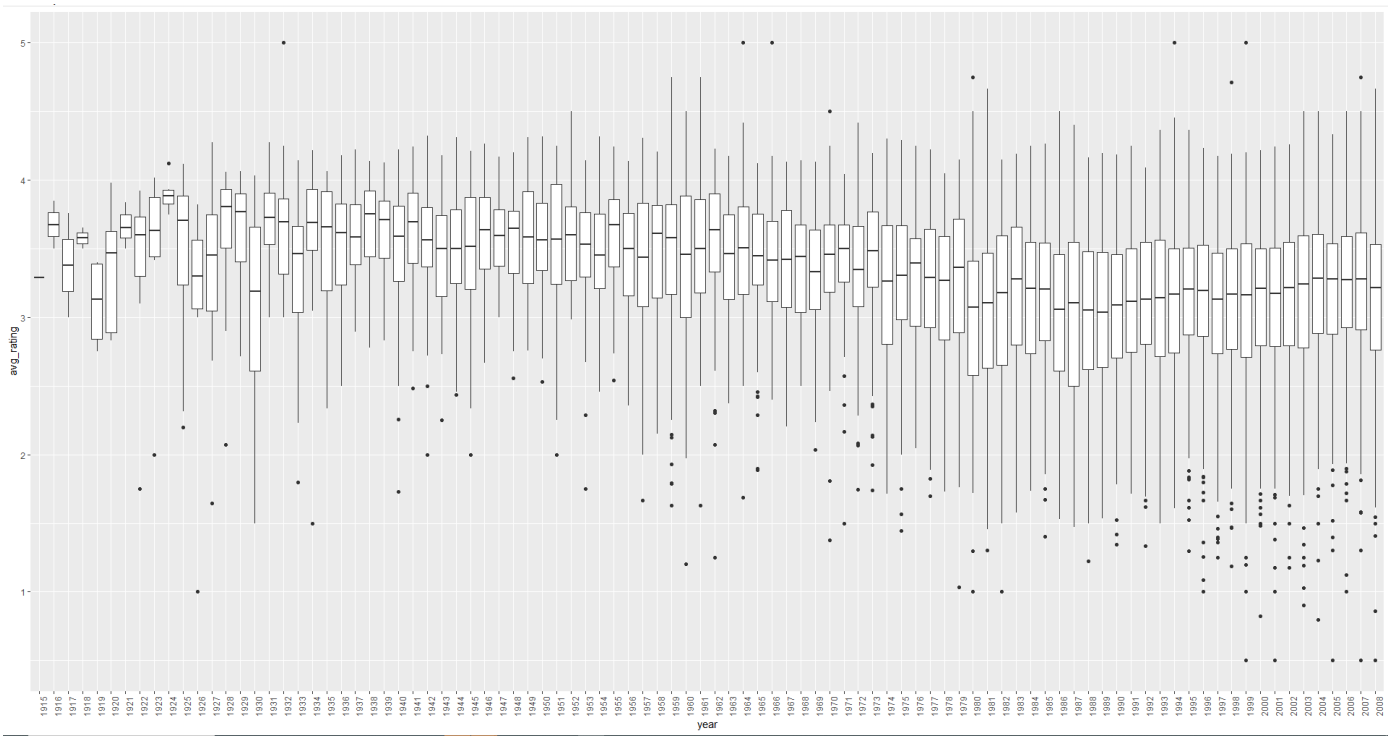
As well as movies, the majority of users have few ratings, almost all users have rated less than 2,000 movies, the userId 59269, is out of the media and has rated 6,491 movies.

- Number of ratings per year



The movies released after 1980 have the major number of ratings, this is not a surprise, it is to be expected that recently released movies are more watched than those that have been on the market for a long time and therefore have a higher number of ratings.

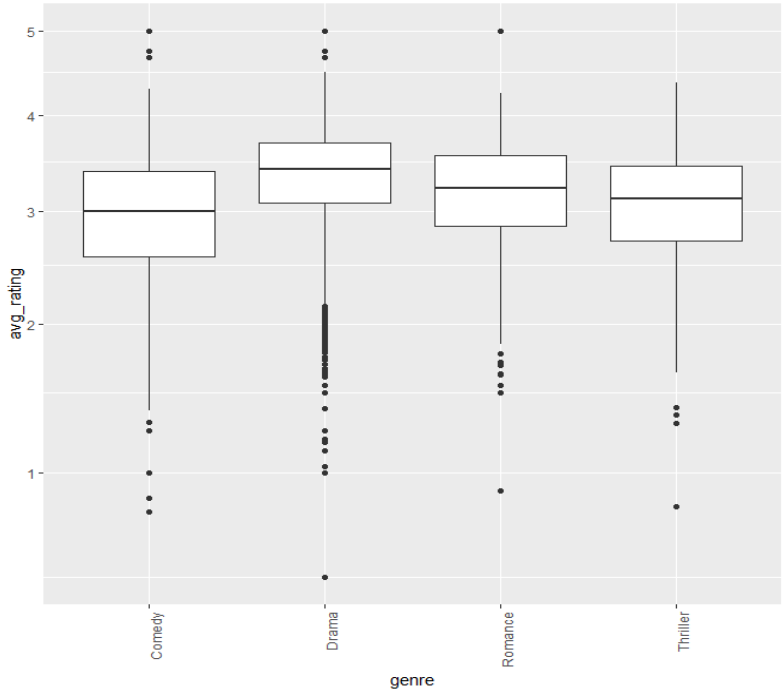
- Box plot – rating per year



Over the years the average rating does not appear to have varied significantly, with movies released after 1980 having a somewhat lower average rating than those released in earlier years. The lowest average is obtained for films released in 1989.

From the graph it can also be concluded that the ratings given to films released in years after 1980 show greater variability than those released in earlier years.

- Box plot – rating per common genres

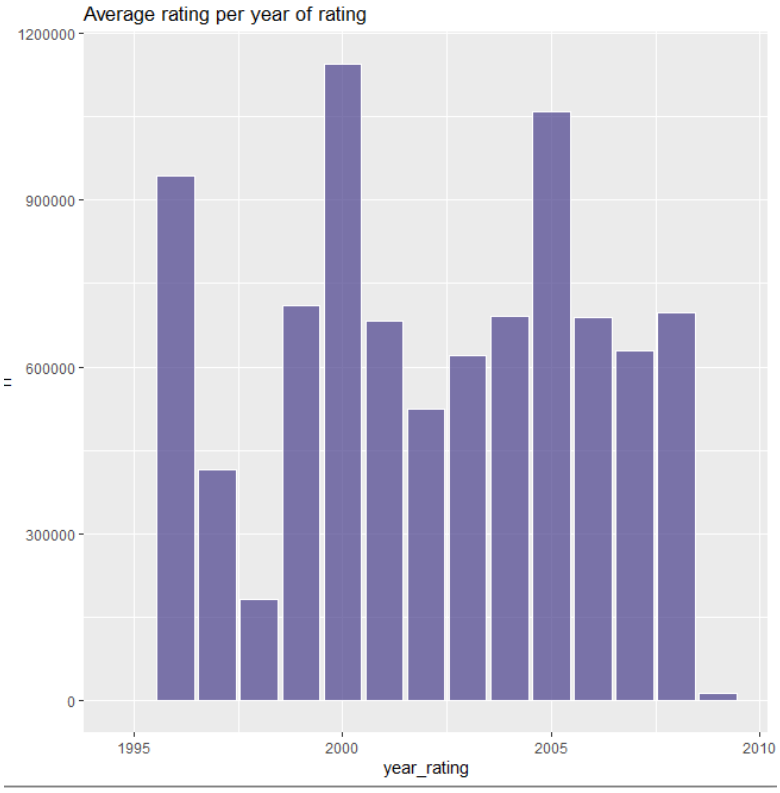


Drama is the highest rated genre; however, the average rating difference by genre differs by less than one unit. The drama genre also shows the highest number of outliers.

- Box plot – rating per common genres

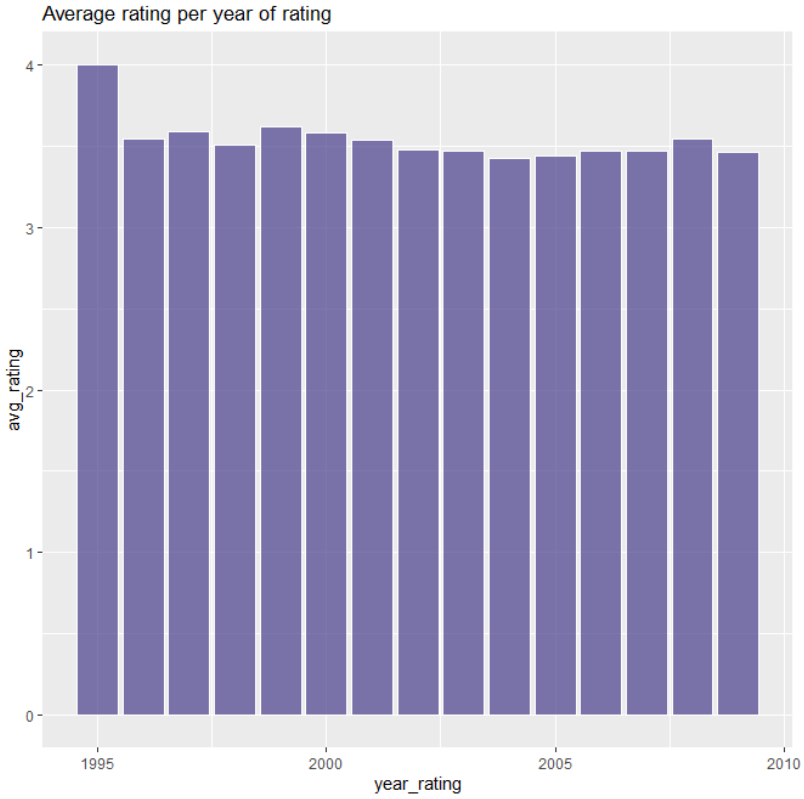
This scatter plot shows the variability of the ratings, each color groups a genre and the average ratings per movie are displayed. Once again it could be observed slightly better rated the drama genre

- Number of ratings per year of rating



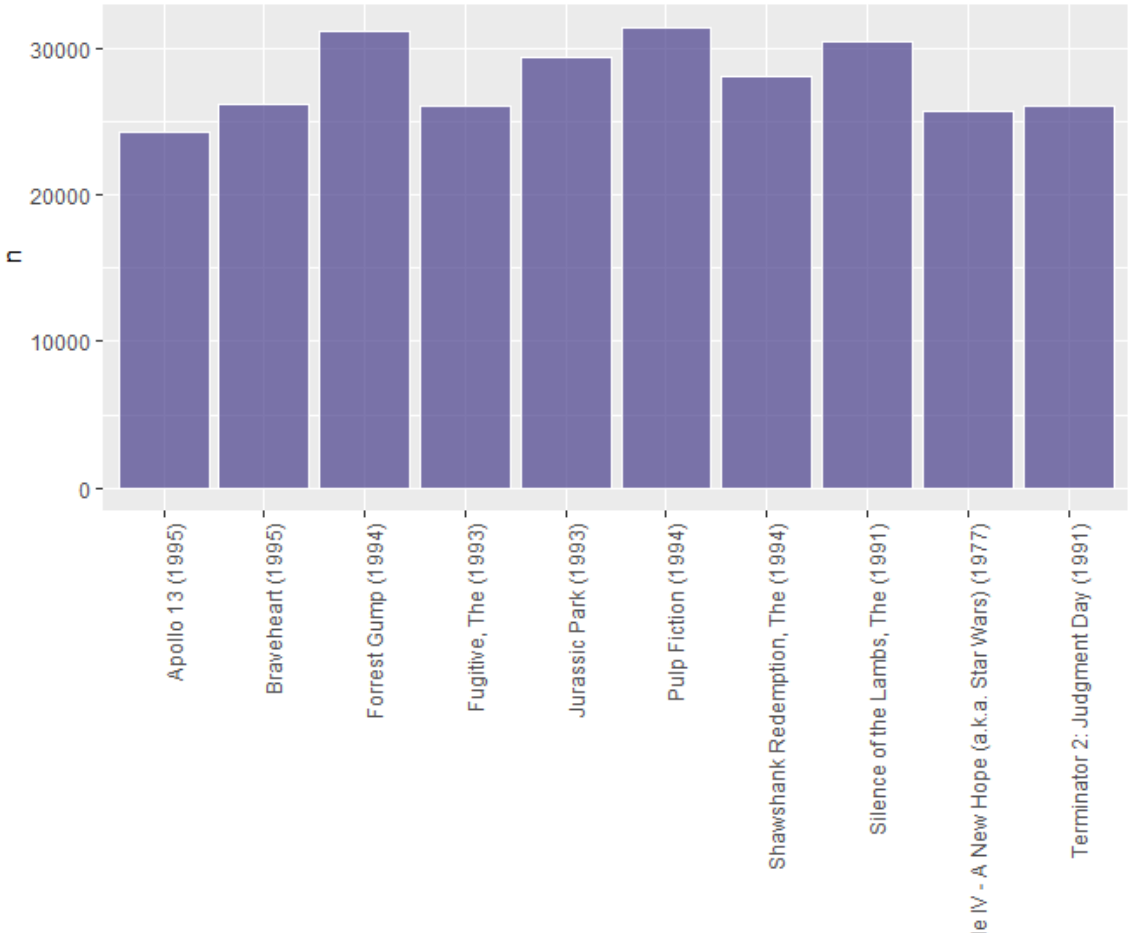
For the years 2000 and 2005 a higher number of ratings were obtained than in the other years, the lowest number of ratings were obtained in 1998, with a total significantly lower than the average over the years.

- Average rating per year



With the exception of the first year, 1995, the average movie rating did not vary significantly, only the first year was close to 4.0, somewhat higher than the average.

The following chart shows the 10 films with the highest number of ratings



4.5 The method

First of all, the edx dataset will be partitioned in train and test sets, the train set will contain the 80% of the data and will be used to model the algorithm, the other 20% will form the test set to calculate the performance indicator RMSE and validate the results.

The previous exploratory analysis shows that each of the features alters the rating value in some way, some such as the movieId, the userId or the genre exhibit greater influence than the others: year of release and rating date. In this sense, the effects of each of the features will be introduced one by one and the results will be evaluated with test sets.

The RMSE function will serve as a metric to validate the result and will be calculated as follows:

RMSE: Residual mean squared error:

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

Where N is the number of user-movie combinations, $y_{u,i}$ is the rating for movie i by user u, and $\hat{y}_{u,i}$ is the rating prediction.

In the next step, we will build models and then, we will compare them to each other.

The simplest model

The first model that we will assume is the same rating for all movies and all users, in this case, the estimate that minimizes the root mean squared error is the average rating of all movies across all users.

$$\hat{y}_{u,i} = \mu + \epsilon_{u,i}$$

where, μ represents the true rating for all movies and users and $\in$ represents independent errors sampled from the same distribution centered at zero.

The RMSE calculated from the edx test set is: 1.0607045

The movie effect.

As we mentioned before adding the variation effect of each feature we could improve prediction results, first we will observe the effect introduced by the *"movieId"* feature.

This is the effect caused just because some movies are frequently rated better than others. To improve our first equation, we will add the term representing the effect of the movie. So, the new model can be described as follows:

$$\hat{y}_{u,i} = \mu + b_i + \epsilon_{u,i}$$

Where b_i represents the average ranking of movie *i* and can be estimated in the following way:

$$\hat{y}_{u,i} - \mu$$

The RMSE calculated from the edx test set is: 0.9437144

The user effect.

As well as the MovieId, the "*userId*" feature effect also could be included in our equation, as the average rating for user *u*, thence the new algorithm will be:

$$\hat{y}_{u,i} = \mu + b_i + b_u + \epsilon_{u,i}$$

Where b_u represents the average ranking of user u and can be estimated in the following way:

$$\hat{y}_{u,i} - \mu - b_i$$

The RMSE calculated from the edx test set is: 0.8661625.

The genre effect.

Once again, we will add the term that represents the genre effect, as the average rating for each genre *g*, the new model could be expressed as follows:

$$\hat{y}_{u,i} = \mu + b_i + b_u + b_g + \epsilon_{u,i}$$

Where b_g represents the average rating for genre g and will be calculated as follows:

$$\hat{y}_{u,i} - \mu - b_i - b_u$$

The RMSE calculated from the edx test set is: 0.8658242.

The release year effect

Now we will add the release year effect, as the average rating for each release year *y*, the new model could be written as follows:

$$\hat{y}_{u,i} = \mu + b_i + b_u + b_g + b_y + \epsilon_{u,i}$$

Where b_y represents the average rating for release year *y* and will be calculated as follows:

$$\hat{y}_{u,i} - \mu - b_i - b_u - bg$$

The RMSE calculated from the edx test set is: 0.8656468.

The rating year effect

Finally, we will add the term that represents the rating year effect, as the average rating for each rating year *yr*, the new model could be written as follows:

$$\hat{y}_{u,i} = \mu + b_i + b_u + b_g + b_y + b_{yr} + \epsilon_{u,i}$$

Where b_{yr} represents the average rating for rating year *yr* and can be expressed as follows.

$$\hat{y}_{u,i} - \mu - b_i - b_u - bg - by$$

The RMSE calculated from the edx test set is: 0.8655341

4.6 Adding Regularization to final model

Regularization allows us avoid overfitting or underfitting problems, we do not want the model to memorize the training data, nor to be too simple to acquire its complexity. In this exercise, regularization is applied to compensate the effects caused by movies, users, genres, release years with very few ratings that can influence the prediction, our task is to find the value of lambda that will minimize the RMSE to optimize the recommendation system.

The general idea is to add a penalty for large values of bi, bu, bg, by to the sum that we minimize, then:

$$\widehat{b_i}(\lambda) = \frac{1}{\lambda + n_i} \sum_{u=1}^{n_i} (Y_{u,i} - \hat{\mu})$$

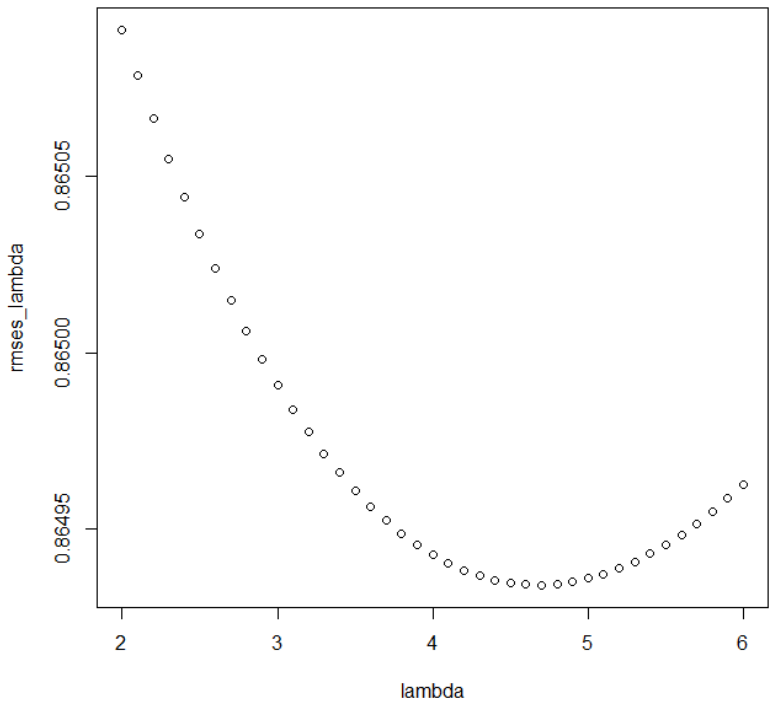
$$\widehat{b_u}(\lambda) = \frac{1}{\lambda + n_i} \sum_{u=1}^{n_i} (Y_{u,i} - \hat{\mu} - \hat{b}_i)$$

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

$$\widehat{b_y}(\lambda) = \frac{1}{\lambda + n_i} \sum_{u=1}^{n_i} (Y_{u,i} - \hat{\mu} - \hat{b}_i - \hat{b}_u - \hat{b}_g)$$

$$\widehat{b_{yr}}(\lambda) = \frac{1}{\lambda + n_i} \sum_{u=1}^{n_i} (Y_{u,i} - \hat{\mu} - \hat{b}_i - \hat{b}_u - \hat{b}_g - \hat{b}_y)$$

From a sequence of lambda, we select the one that minimize the RMSE, the following plot shows the RMSE vs lamda, so we can find the optimal lambda at: with the best RMSE: 4.8



5 Results

The following table shows the validation results starting from the simplest model: the same rating for all movies, adding one by one the effects introduced by movie, user, genre, release year, rating year and finally including regularization to the selected algorithm.

	RMSE
Mean	1.0607045
Movie Effect	0.9437144
Movie and User Effect	0.8661625
Movie, User and Genre Effect	0.8658242
Movie, User, Genre and Release year Effect	0.8656468
Movie, User, Genre, Release year and Rating year Effect	0.8655341
Regularization	0.8641624

We can realize that the improvement between the simplest model to the once that consider the movie effect is almost a 12% which is a considerable improvement, the next model that include the user effect, improves a 8%; but the effect of the following features only improves the algorithm by 0.1%. Finally with the regularization, the desired objective is achieved with 0.8641624 by improving the RMSE target by 0,00074 .

6 Conclusions

Throughout the project we have been examining the MovieLens dataset, we have explored its structure, identified their features and pointed out those that have the greatest influence on the prediction of the desired rating, we also have wrangled the data to extract as much information as possible from the dataset. We have studied different linear regression models, identifying the biases that each feature incorporates and modifying the algorithm to minimize the RMSE.

The model evaluation performance metric RMSE (Root mean squared error) showed that the Linear regression algorithm with regularized effects is an appropriate recommender system to predict ratings in the validation set, the objective of achieving an RMSE below 0.86490 was met with the final model proposed.

Future research can analyze the problem from the perspective of matrix factorization, ensemble methods, distributed Random Forests and other non-linear transformations, maybe these models generate higher levels of accuracy, but they will also be much more computationally demanding.

7 References

Irizzary,R., 2018,Introduction to Data Science, Data Analysis and Prediction Algorithms with R; https://rafalab.github.io/dsbook/

Soham,D., Building a Movie Recommendation System with Machine Learning; https://www.analyticsvidhya.com/blog/2020/11/create-your-own-movie-movie-recommendation-system/