MovieLens

Agustin Daniel Ross

sep - 2021, Argentina

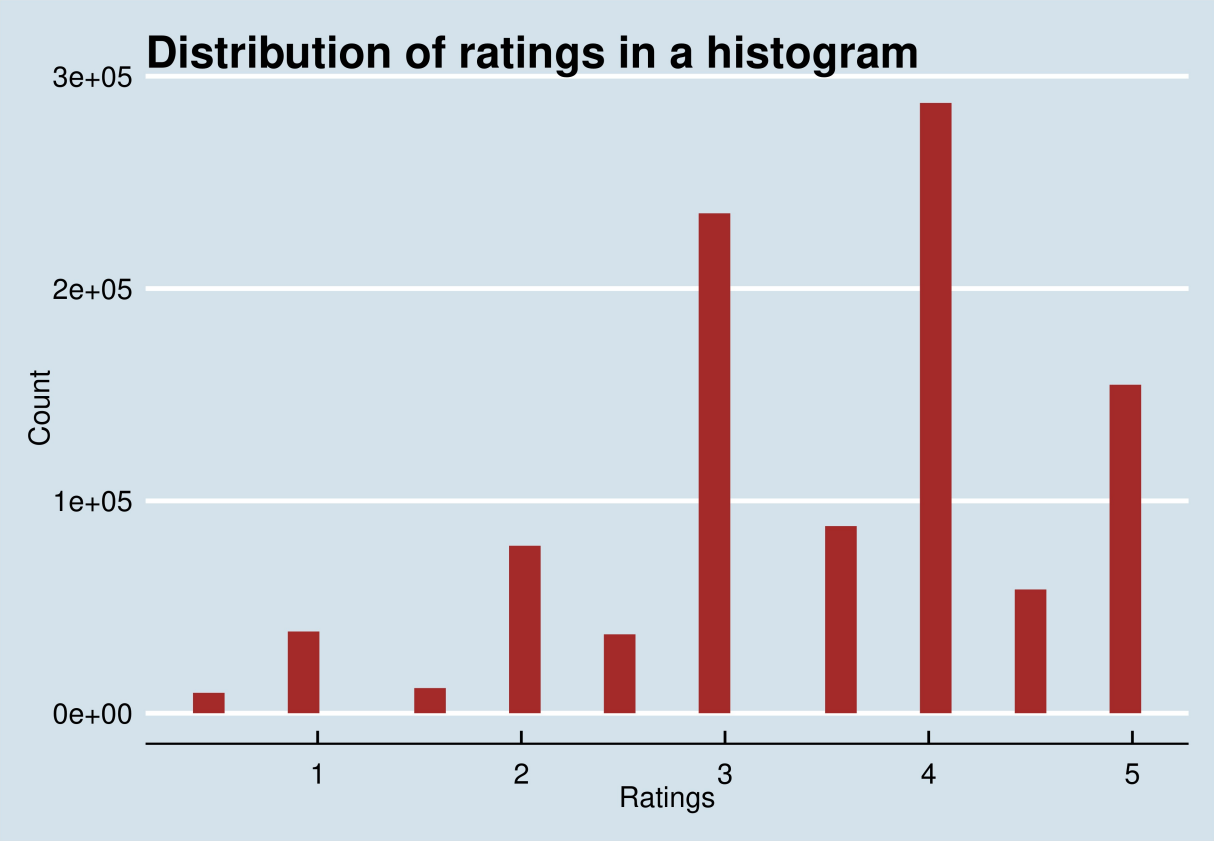
Introduction section: Exploratory analysis and project's goal.

For this project I will use the MovieLens dataset. MovieLens is a database with over 10 million ratings for over 10,000 movies by around 70,000 users. The dataset includes all sorts of information, like the identification of the user and the movie, the rating of each and every movie and genre on which the movie is categorized.

The main goal of this final project is to predict movie ratings with a given dataset with several features. To reach this goal, the dataset is divided into two groups: the train set and the validation set. The validation set is 10% of the original data and is not used in the construction of the model, it is used to validate the performance of the model and allow us to make comparisons between different models. This division of the data is, in my opinion, one of the most crucial steps in the model creation, and this is because the fundamental goal of Machine Learning is to generalize beyond the data observations that are used to train the models. We want to evaluate the model to estimate the quality of its generalization for the data that did not create the model. However, since future observations are unknown and we cannot check the accuracy of our predictions for future observations right now, we have to use some of the data for which we already know the answer as a proxy for future data. Evaluating the model with the same data that has been used for training is not useful, since models that can "remember" the training data rather than generalize, then reusing the training data to test the model would lead to overfitting.

Let's start with some exploratory analysis of the dataset. MovieLens is a dataset with exactly 10000054 observations, it has 69878 unique users provided ratings and 10677 unique movies rated. As you can see, there is a lot of information we can include in the model.

The most important element to see if you want to know how good/bad is a movie, is his rating. The rating is a number between 1 and 5, let's see how it is distributed:

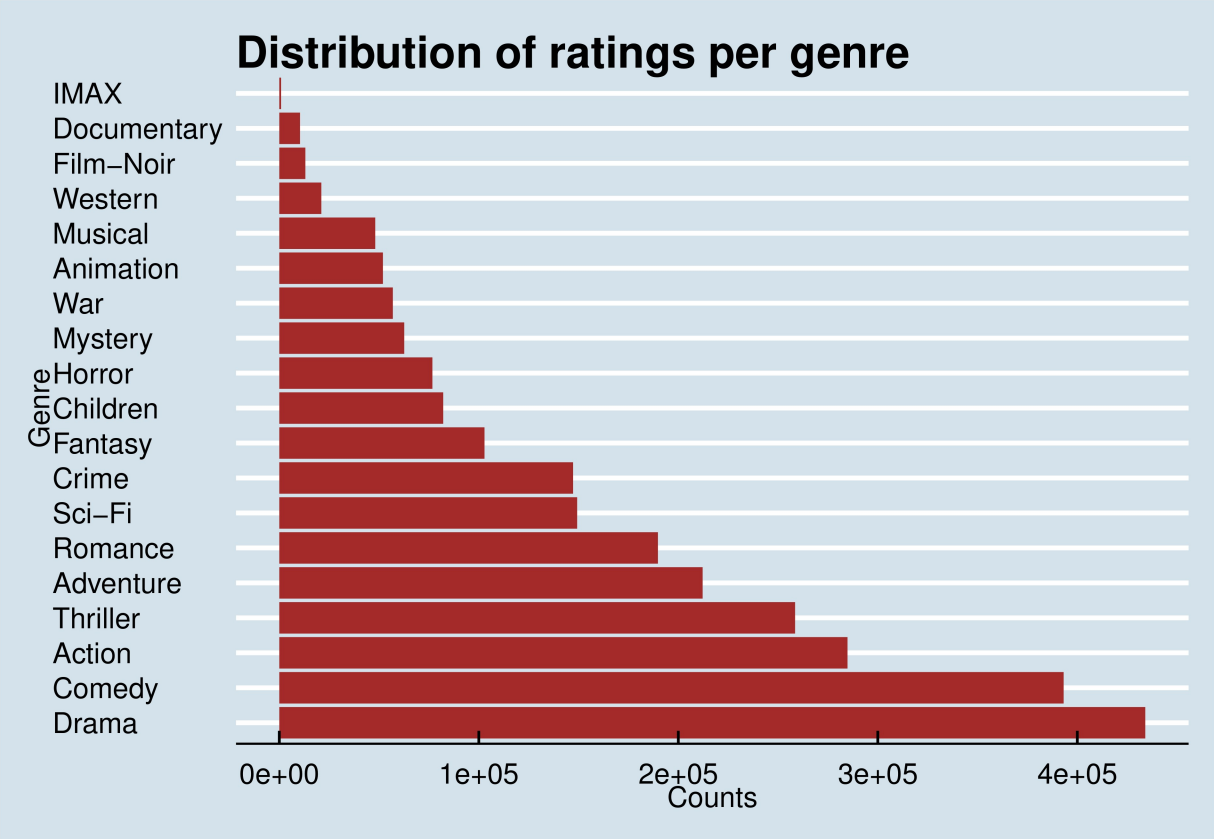


An interesting thing to see in the histogram of ratings above is how integer ratings were more frequent than half-integers.

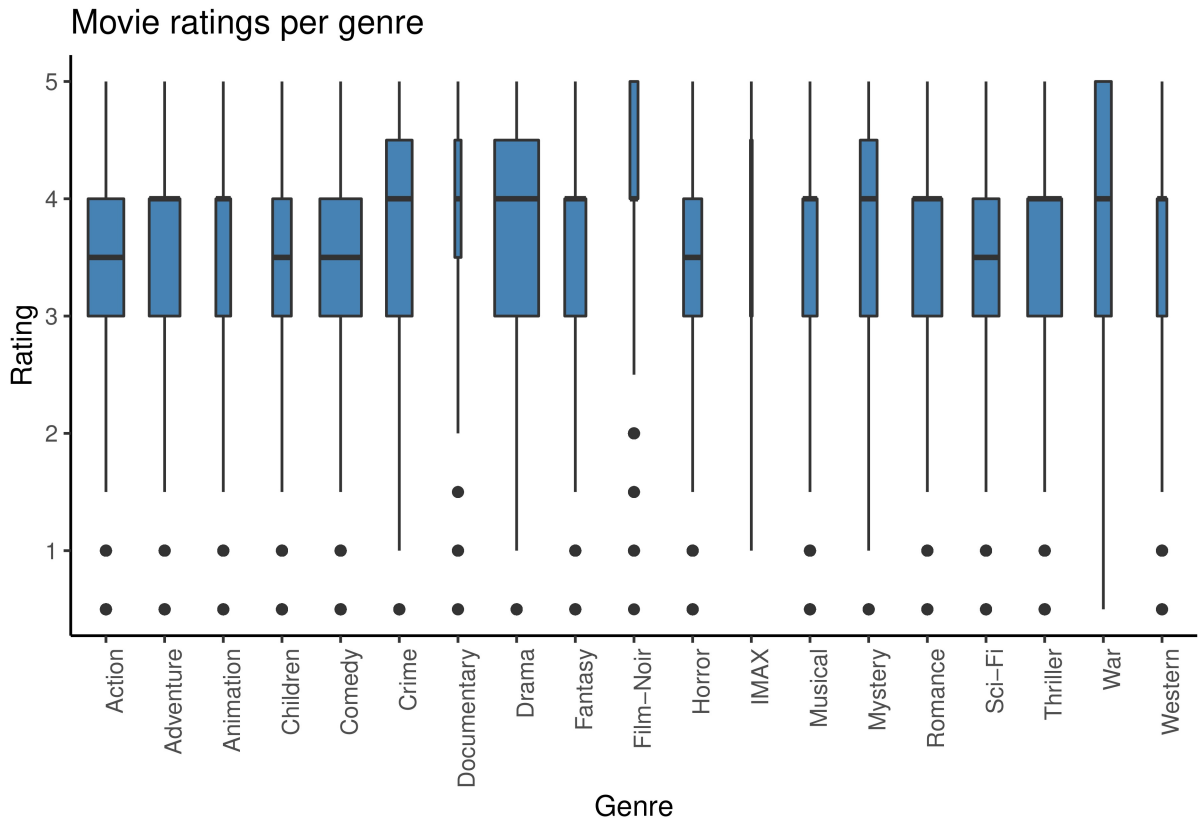
Let's now focused on the genre attribute of the dataset. You can see the list of genres here:

```
##  [1] "Comedy"             "Romance"            "Action"
##  [4] "Crime"              "Thriller"           "Drama"
##  [7] "Sci-Fi"             "Adventure"          "Children"
## [10] "Fantasy"            "War"                "Animation"
## [13] "Musical"            "Western"            "Mystery"
## [16] "Film-Noir"          "Horror"             "Documentary"
## [19] "IMAX"               "(no genres listed)"
```

It is also interesting to know if all movies are equally rated, or if there are some types of movies with more ratings than others. To see that, I created a plot that shows the number of times the movies of each genre were rated.



As you can see, the most popular rated genre types are Drama and Comedy. Let's now combine the information of rating and the information of genre. To see it graphically I made a boxplot of the distribution of the ratings per genre, you can see it below:



Analysis section. The Models creation.

Now we understood better the dataset, let's proceed to the creation of the recommendation model. From this point further, I used the "edx" dataset to create the model and the "validation" dataset to test it.

The methodology I decided to follow is: Create first very basic models (the ones given in the "Machine Learning" course) and then, in a progressive way, create more advanced ones. I used the Root Mean Square Error (RMSE) to make the comparison between models. For the RMSE, LESS is BETTER, you will see how the RMSE will decrease when the model gets more complex. The RMSE is defined by the following function:

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

First Model: Only the rating's mean

The most basic model is to predict the same rating regardless of the user, movie, or genre. To do that, I calculated the general mean of ratings.

```
## 1st Model: Only the rating's mean

ratings_mean <- mean(edx$rating)
```

```
model_1_prediction <- ratings_mean

# Let's see the RMSE:

RMSE_MODEL_1 <- RMSE(validation$rating, model_1_prediction)
RMSE_MODEL_1
```

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

As you can see, I reached a very high RMSE. Let's try to improve it.

Second Model: Movie's effect

Some movies have higher ratings than others, so the following model considers the movie effect. To do that, I estimate the movie effect as the average of the ratings by a movie.

```
## 2nd Model: Let's include to the 1st model the movie's effect:

movies_contribution<- edx %>%
  group_by(movieId) %>%
  summarize(Mg_contribution_of_movies = mean(rating - ratings_mean))

model_2_prediction <- ratings_mean + validation %>%
  left_join(movies_contribution, by='movieId') %>%
                      .$Mg_contribution_of_movies

# Let's see the RMSE:

RMSE_MODEL_2 <- RMSE(validation$rating, model_2_prediction)
RMSE_MODEL_2
```

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

As you can see, the RMSE improves between the first and the second model, but it is still very high. Let's try to improve it even more.

Third Model: Addition of User effect.

It has been showed how each movie is different and the addition of a "movie effect" is necessary, but also every user is different, so now I include the "user effect". Some users rate movies higher than others, so the next model considers both the movie and the user effect. I estimate the user effect as the average of the ratings per user.

```
## 3rd Model: Let's include to the 2nd model the user's effect:

user_contribution <- edx %>%
      left_join(movies_contribution, by='movieId') %>% group_by(userId) %>%
      summarize(Mg_contribution_of_users = mean(rating - ratings_mean -
```

```
                                                Mg_contribution_of_movies))

model_3_prediction <- validation %>%
  left_join(movies_contribution, by='movieId') %>%
  left_join(user_contribution, by='userId') %>%
  mutate(M3_pred = ratings_mean + Mg_contribution_of_movies +
           Mg_contribution_of_users) %>%
  .$M3_pred

# Let's see the RMSE:

RMSE_MODEL_3 <- RMSE(validation$rating, model_3_prediction)
RMSE_MODEL_3
```

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

As you can see, the RMSE is lower than the second and the first model. I have already included: Rating's effect (1st model), Movie's effect (2nd model) and User's effect (3rd model). Maybe the RMSE reached is "acceptable", but there is still a very powerful attribute I did not use yet: the genre. Let's include it in the final model and try to improve the RMSE even more!

Fourth Model: Addition of Genre Effect.

```
## 4th Model: I have already included: Rating's effect (1st model), Movie's effect
## (2nd model) and User's effect (3rd model). In this 4th model I will include
## the Genre, that is the only variable that I did not use already.


# Now, continuing with the method I used in the previous models, let's create the 4th model:

genres_contribution <- edx %>% left_join(movies_contribution, by = "movieId")%>%
      left_join(user_contribution, by = "userId") %>% group_by(genres) %>%
      summarize(Mg_contribution_of_genres = mean(rating - ratings_mean -
      Mg_contribution_of_movies - Mg_contribution_of_users))


model_4_prediction <- validation %>%
                      left_join(movies_contribution, by = "movieId") %>%
                      left_join(user_contribution, by = "userId") %>%
                      left_join(genres_contribution, by = c("genres")) %>%
                      mutate(M4_pred = ratings_mean + Mg_contribution_of_movies +
                      Mg_contribution_of_users + Mg_contribution_of_genres) %>%
                      .$M4_pred


# Let's see the RMSE:

RMSE_MODEL_4 <- RMSE(validation$rating, model_4_prediction)
RMSE_MODEL_4
```

```
## [1] 0.8649469
```

As you can see, the RMSE is the lowest, this means that the genre is a good feature to take into consideration.

Results section. Resume of the models results.

It has been shown four different models, the first as the simplest one, and the fourth as the most complex one.

Here I show to you the final results of the four models created:

```
ALL_RMSE
```

```
## # A tibble: 4 x 2
##   method          RMSE
##   <chr>          <dbl>
## 1 Model Number 1 1.06
## 2 Model Number 2 0.944
## 3 Model Number 3 0.865
## 4 Model Number 4 0.865
```

The best one:

```
print(ALL_RMSE[which.min(ALL_RMSE$RMSE),1])
```

```
## # A tibble: 1 x 1
##   method
##   <chr>
## 1 Model Number 4
```

```
#The RMSE is approx 0.8649

print(RMSE_MODEL_4)
```

```
## [1] 0.8649469
```

Conclusion section

The goal of this project was to predict movie ratings from a database with over 10 million evaluations and with several features. To do that, I considered first the general rating of the entire dataset, then the impact of movies, after the one of the users and finally the genres effect to the ratings. I divided the dataset into training and validation to avoid overfitting. The best-fitted model was the one that take into consideration the general rating's mean, the movie's effect, the user's effect and the genre's effect. This model achieved an RMSE of 0.864946886172894. This RMSE can be improved, but it is a very good RMSE according to the course standard.