Jaime Prades Movielens

Jaime Prades

mar 2023 - Chile

Introduction section: Exploratory analysis and project's goal.

In this project, I will utilize the MovieLens dataset which consists of over 10 million ratings for over 10,000 movies by around 70,000 users. The dataset includes various information such as the identification of the users and movies, movie ratings, and movie genre.

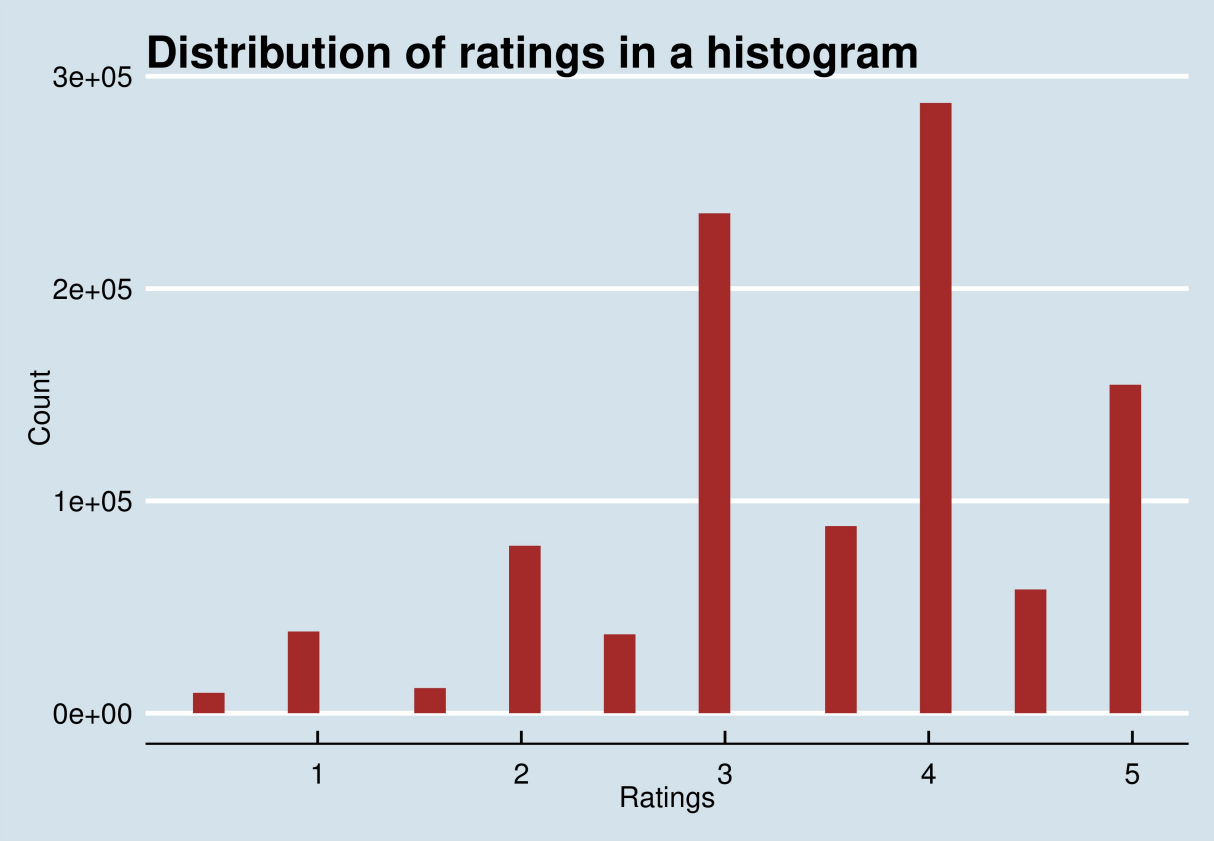
The primary objective of this project is to predict movie ratings based on the provided dataset and its various features. To achieve this goal, the dataset will be divided into two groups - the training set and the validation set ("final holdout test"). The validation set, accounting for 10% of the original data, will not be used in the model construction but rather to validate the model's performance and allow for comparisons between different models.

In my opinion, dividing the data into the training and validation sets is a crucial step in the model creation process. This is because the ultimate goal of Machine Learning is to generalize beyond the training data observations. We want to evaluate the model's ability to generalize for data that it has not seen before, as the future observations are unknown and we cannot directly verify the accuracy of our predictions for them.

Using the validation set as a proxy for future data helps us to estimate the quality of the model's generalization. Evaluating the model using the same data that was used for training is not useful as it could lead to overfitting, where the model simply "remembers" the training data rather than generalizing.

Let's begin with an exploratory analysis of the dataset. The MovieLens dataset contains exactly 10000054 observations and has 69878 unique users who have provided ratings and 10677 unique movies that have been rated. As you can see, there is a vast amount of information available for inclusion in the model.

The most crucial factor in determining the quality of a movie is its rating, which ranges from 1 to 5. Let's examine its distribution:

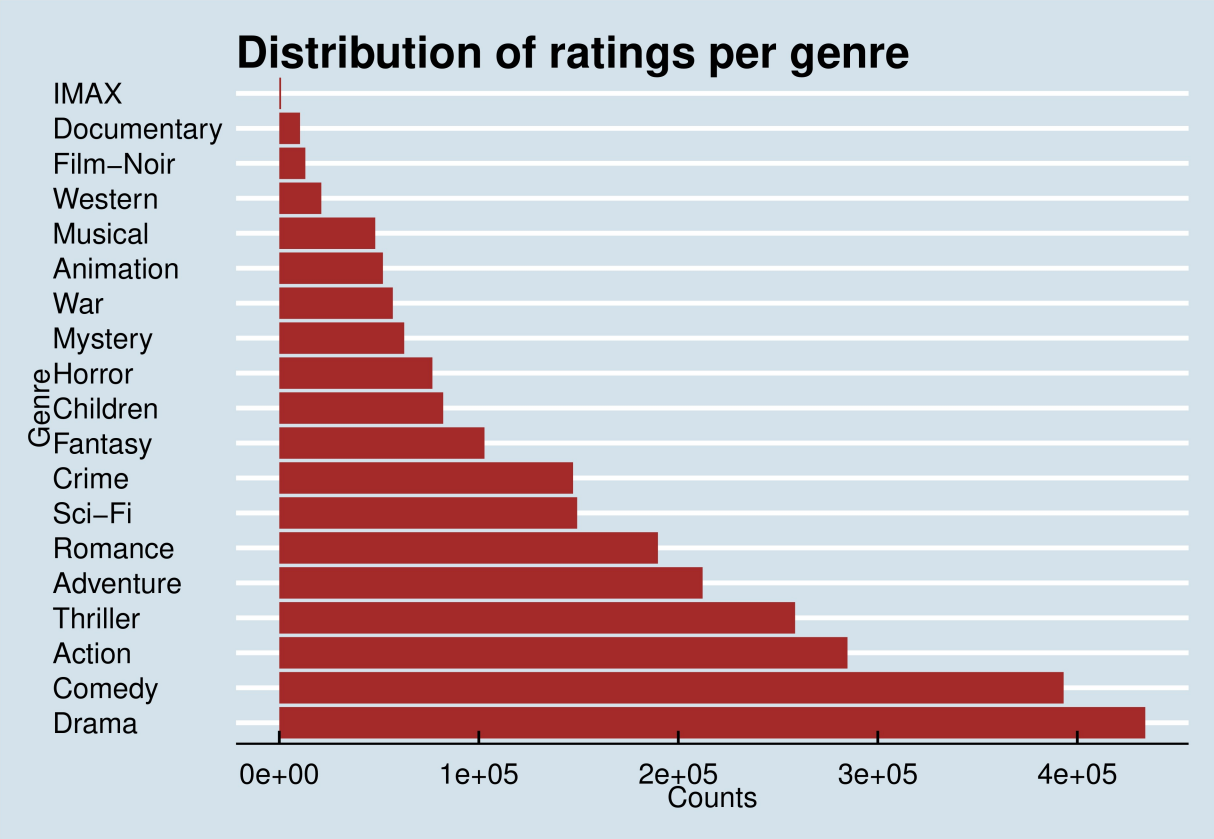


An interesting observation from the histogram of ratings is that integer ratings are more frequent than half-integer ratings.

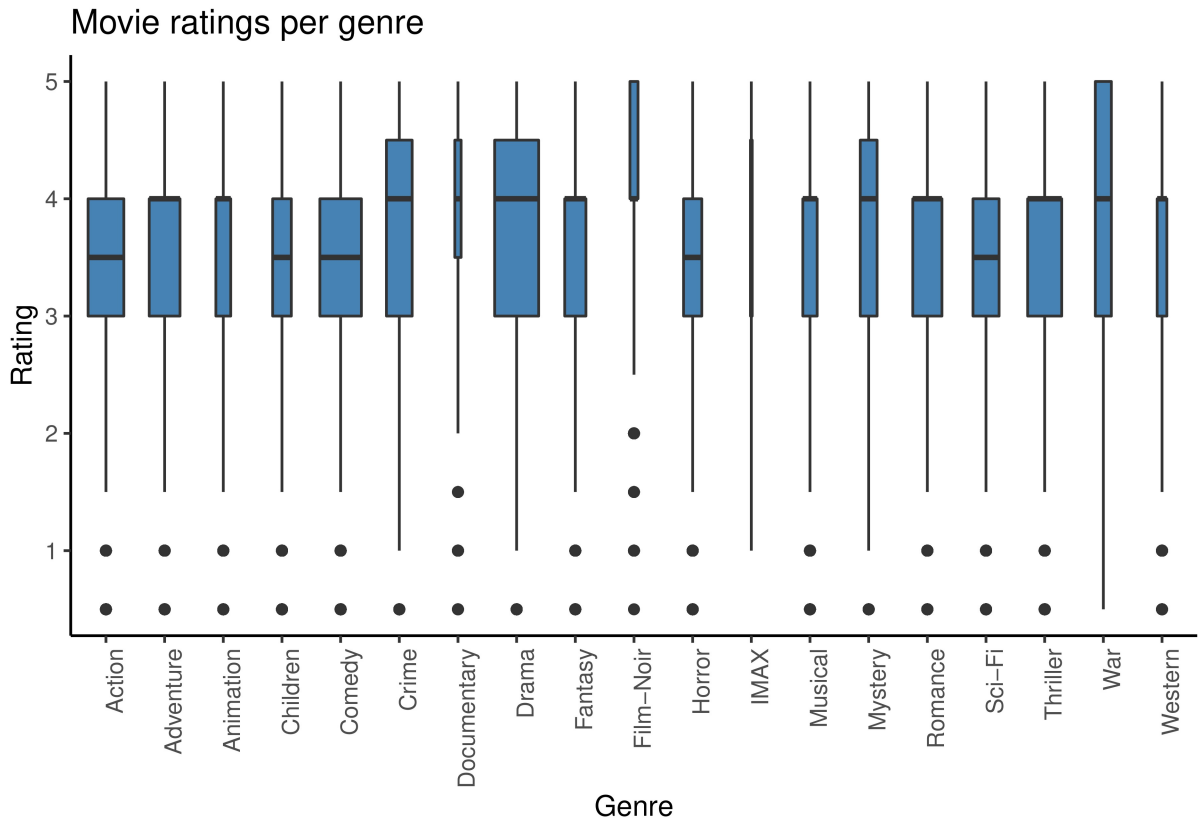
Now, let's focus on the genre attribute in the dataset. The list of genres can be seen here:

```
##  [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 worthwhile to investigate if all movies are rated equally, or if certain types of movies receive more ratings than others. To do this, I created a plot that displays the number of times movies of each genre were rated.



As observed, the most highly rated genres are Drama and Comedy. Now, let's combine the information on ratings with the genre information. To visualize this, I created a boxplot of the rating distribution per genre, which can be seen below:



Analysis section. The Models creation.

Now that we have a better understanding of the dataset, let's move on to creating the recommendation model. From this point forward, I will use the "edx" dataset to train the model and the "final_holdout_test" dataset to test it.

The methodology I have chosen is to start with basic models (the ones covered in the "Machine Learning" course) and gradually progress to more advanced ones. I will use the Root Mean Square Error (RMSE) to compare the models. For the RMSE, LESS is BETTER, and you will see the RMSE decrease as the model becomes 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 simplest model is to predict the same rating for all movies, regardless of the user or genre. To achieve this, I calculated the overall 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(final_holdout_test$rating, model_1_prediction)
RMSE_MODEL_1
```

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

As you can see, the RMSE for this model is quite high. Let's try to improve it.

Second Model: Movie's effect

In this model, I take into account that some movies have higher ratings than others. To do this, I calculate the movie's impact as the average rating for each movie.

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

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

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

# Let's see the RMSE:

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

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

As seen in the results, the RMSE improved between the first and second models, but it still has room for improvement. Now, let's attempt to further reduce the RMSE.

Third Model: Addition of User effect.

It has been demonstrated that each movie is unique and the inclusion of a "movie impact" is necessary. Similarly, every user is different, so the next model includes both the movie impact and the "user impact." I calculate the user impact as the average rating 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) %>%
      dplyr::summarize(Mg_contribution_of_users = mean(rating - ratings_mean -
                                                  Mg_contribution_of_movies))
```

```r
model_3_prediction <- final_holdout_test %>%
  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(final_holdout_test$rating, model_3_prediction)
RMSE_MODEL_3
```

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

As we can observe, the RMSE has improved compared to the previous models, which incorporated only the rating, movie, and user effects. However, we have not yet utilized the genre information. In the final model, we will include the genre attribute and aim to further reduce the RMSE.

Fourth Model: Addition of Genre Effect.

```r
## 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) %>%
      dplyr::summarize(Mg_contribution_of_genres = mean(rating - ratings_mean -
      Mg_contribution_of_movies - Mg_contribution_of_users))


model_4_prediction <- final_holdout_test %>%
                      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(final_holdout_test$rating, model_4_prediction)
RMSE_MODEL_4
```

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

With the inclusion of the genre attribute, we have significantly improved the RMSE. This demonstrates that genre is a valuable factor to consider when creating a recommendation model.

Results section. Resume of the models results.

The four models were demonstrated, starting with the simplest one and ending with the most complex one. The following table showcases the final results of all four models:

```
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 aim of this project was to predict movie ratings based on a large database containing over 10 million evaluations and various features. The approach was to first consider the overall rating of the dataset, then the influence of the movies, followed by the influence of users, and finally the impact of genres on ratings. To prevent overfitting, the dataset was divided into training and validation sets. The best model was found to be the one that took into account the mean general rating, the effect of movies, the effect of users, and the impact of genres. This model achieved an RMSE of 0.864946886172894, which is considered a good result according to course standards and can still be improved further.