MovieLens project

Federico Huxhagen

December 2020

Overview

The purpose of this project is to build a movie recommendation system with the MovieLens dataset, by implementing Machine Learning techniques. Essentially, our goal is to predict, given data, what rating a user would give to a specific movie.

The MovieLens dataset was cleaned and then partitioned into 'edx' and 'validation sets, both of which include six variables: userId, movieId, rating, timestamp, title and genres. After that, I built the model that would make it possible to predict the ratings. Finally, the model was tested on the validation set, using RMSE to determine its precision.

Methods

To start with, the original dataset was cleaned and organized to include the six variables we are going to work with, and it was partitioned into 'edx' and the 'validation' (which includes 10% of the data) sets.

It is important to analyse the structure of our 'edx' set:

```
## Classes 'data.table' and 'data.frame':   9000055 obs. of  6 variables:
##  $ userId   : int  1 1 1 1 1 ...
##  $ movieId  : num  122 185 292 316 329 ...
##  $ rating   : num  5 5 5 5 5 ...
##  $ timestamp: int  838985046 838983525 838983421 838983392 838983392 ...
##  $ title    : chr  "Boomerang (1992)" "Net, The (1995)" ...
##  $ genres   : chr  "Comedy|Romance" "Action|Crime|Thriller" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

As I have mentioned, there are six different variables: userId, movieId, rating, timestamp, title and genres. My approach was to build the recommendation system taking these into consideration.

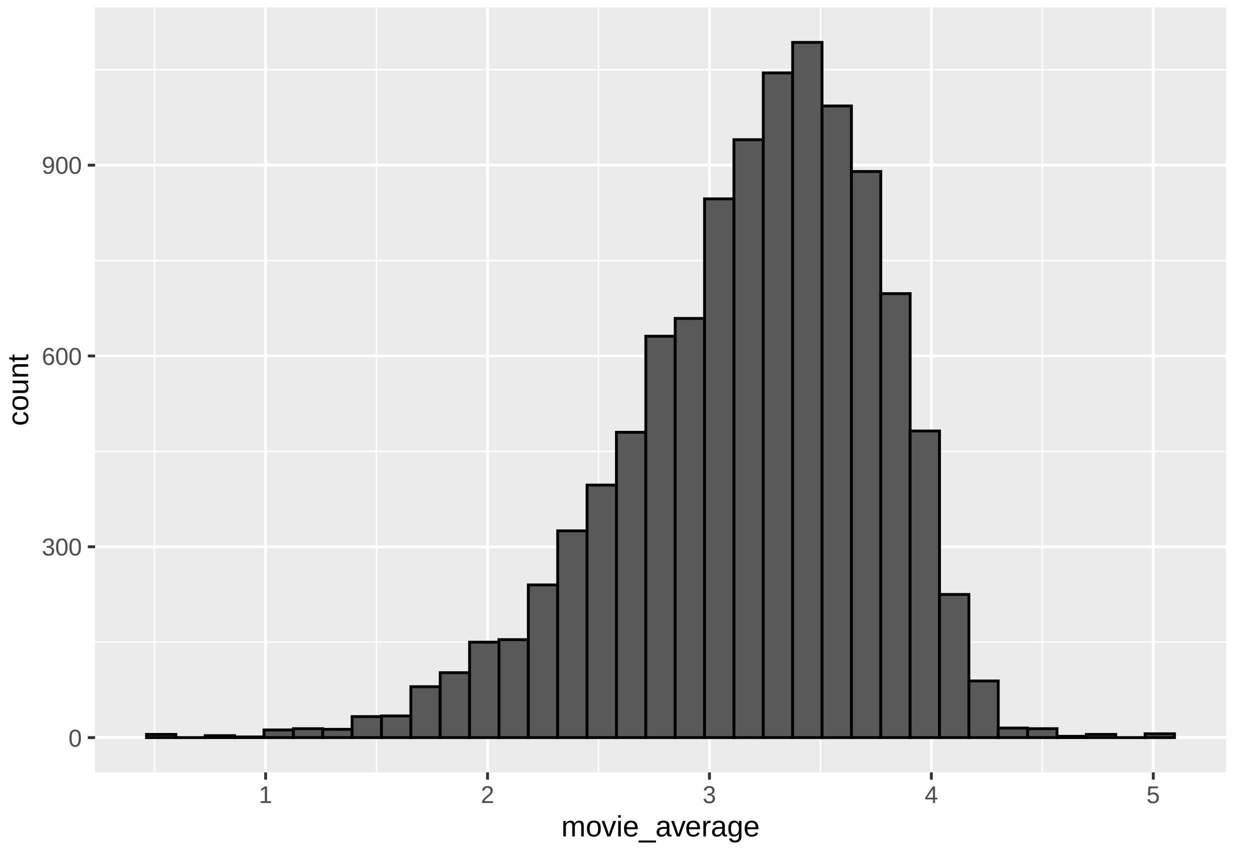
The first step I took was to find the average rating across all movies, which we will call mu.

We compute it like this:

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

Then, if a movie gets a rating different from the average, we can argue that this is because of the different variables that are present.

To continue, it would be interesting to see if some movies are generally rated higher than others: essentially, is there 'movie bias'? We can see that in this plot:



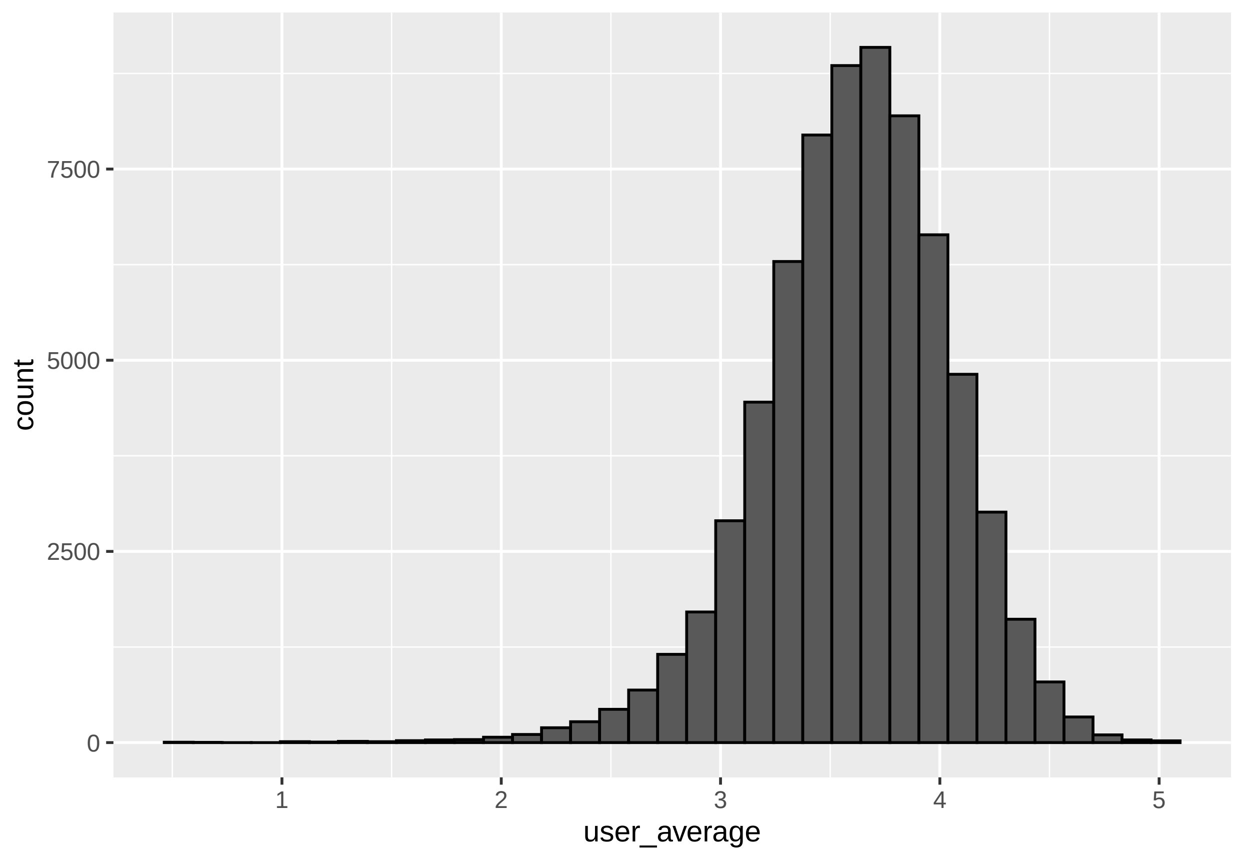
It is clear that some movies are indeed rated higher on average than others. But how can we quantify this? We can write our model like this:

$Rating = \mu + b_i + error$

b_i is the variable that accounts for movie bias. We can compute it by subtracting the average for all movies from the actual rating in each entry, and then take the average across all observations for each specific film.

```
b_i <- edx %>%
  group_by(movieId) %>%
  summarize(b_i = mean(rating - mu))
```

Now, we want to see if some users have a tendency to rate movies (in general) higher than others. We can plot the number of users against the average rating per user and we get this histogram:



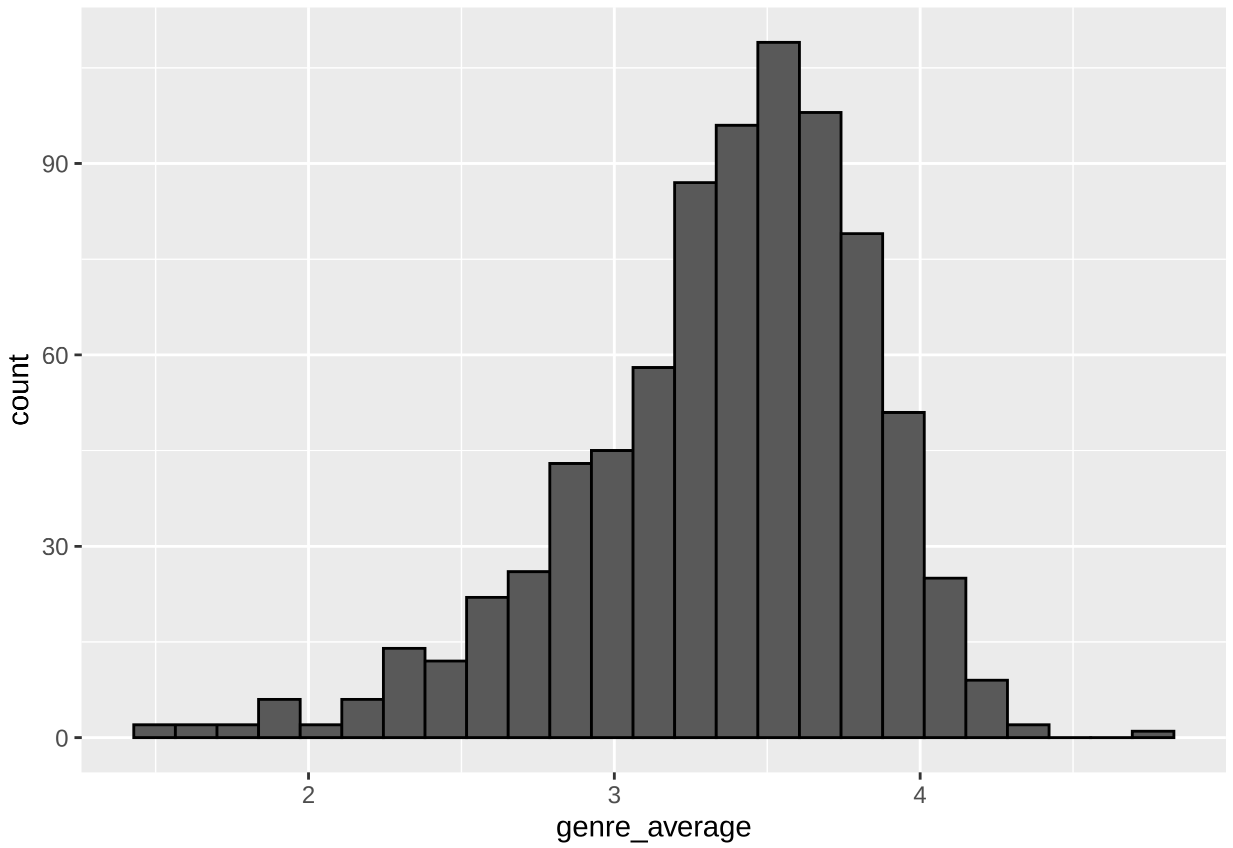
Clearly, different users tend to rate movies in different ways. We can add a new parameter to our model that accounts for user bias (b_u):

$Rating = \mu + b_i + b_u + error$

We can compute b_u for each user by subtracting μ and b_i from the actual rating, and then take the mean for each user.

```r
b_u <- edx %>%
  left_join(b_i, by = "movieId") %>%
  group_by(userId) %>%
  summarize(b_u = mean(rating - mu - b_i))
```

To keep improving our model, it would be interesting to see if different movie genres have an impact on ratings (essentially, are some genres rated higher on average than others?). We can build another histogram to see this:



Once again, it is evident that movies of different genres have different ratings on average. We add a new parameter that accounts for this genre bias (b_g), resulting in the following model:

$Rating = \mu + b_i + b_u + b_g + error$

We compute b_g by subtracting μ, b_i and b_g from the actual rating for each entry, and then take the average for each of the 797 genres in the dataset.

```r
b_g <- edx %>%
  left_join(b_i, by='movieId') %>%
  left_join(b_u, by='userId') %>%
  group_by(genres) %>%
  summarize(b_g = mean(rating - mu - b_i - b_u))
```

We are now ready to test the model on the validation test. We will predict a rating for each entry in our validation set (y_hat). For each case, we will simple compute the sum of $\mu + b_i + b_u + b_g$

```r
y_hat <- validation %>%
  left_join(b_i, by='movieId') %>%
  left_join(b_u, by='userId') %>%
  left_join(b_g, by="genres")%>%
  mutate(pred = mu + b_i + b_u + b_g) %>%
  pull(pred)
```

Now that we have a prediction, there is a detail we must take into account. The ratings in the edx dataset range from 0.5 to 5.

```
##   min_value max_value
## 1       0.5         5
```

Now, let's look at the maximum and minimum values in our predictions.

```
##    min_value max_value
## 1 -0.4781138  6.098296
```

So, we have values that are lower than 0.5 and others higher than 5. This can never be the case. Therefore, we interpret that if a user should give a movie over a 5-star rating, they would give it the highest possible rating (5). And if we predict a user would give a movie less than a 0.5 rating (which means they hated it), they would give it the lowest possible rating.

To correct this, we need to find which predicted ratings are lower than 0.5 and replace them with 0.5, and find which are higher than 5 and replace them with 5.

```
ind_lower<- which(y_hat<0.5)
y_hat[ind_lower]<- 0.5

ind_over<- which(y_hat>5)
y_hat[ind_over]<- 5
```

Now that we have a final prediction for the validation set, we can examine our model's performance.

Results

I have mentioned that we will use RMSE to analyze how well our model works. Let's create a function that takes as input the actual ratings and our predicted ratings and then computes the RMSE.

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

Now, we can compute the RMSE of our model:

```
RMSE(validation$rating, y_hat)
```

```
## [1] 0.8647516
```

I have been able to build a model with an RMSE of 0.8647516, lower than our goal of 0.86490.

Conclusions

We started out the with a clear objective: to build a recommendation system that would predict the rating for a specific movie by a user knowing some data, as precisely as possible. We have built a model that accounts for movie bias (the fact that some movies are better than others), user bias (some users are less demanding when it comes to movies than others) and genre bias (the fact that certain genres are better rated on average than others). We have tested the model and obtained an RMSE of 0.8647516.

However, is it possible to do better? It certainly is. For starters, we did not use the "timestamp" variable in our model, which could have an influence we have not taken into account.

Apart from that, regularization has not been used when determining the parameters of our model. If a movie has just a few very high ratings, we assume that it has a high b_i. However, this is not necessarily the case. It is necessary to regularize and, in a way, correct the parameters of those films/genres/users that have few ratings. If we do this, it is possible to obtain better estimates.

It is possible, by adding these changes to our model, to make it even more precise.