MovieLens Project

Bardh Kukaj

2024-01-24

Contents

Introduction

This report is written for the course named "HarvardX PH125.9x - Data Science: Capstone" which is part of the HarvardX Professional Certificate Data Science Program.

The purpose of this report is to predict the ratings (ranging from 0.5 starts to 5 stars) for a movie data set, namely the MovieLens 10 million data set. The success of the model used to predict the ratings is measured by the root mean square estimate, or RMSE. The RMSE measures, on average, the differences between the predicted values by the model, and the actual values. The guidance on the course is to seek an RMSE value lower than 0.86490. Incidentally, a zero value of RMSE indicates that it is a perfect model. In this case, a value of 0.86490 implies that the rating is off by that same figure. This report achieves an RMSE of 0.86458, which is lower than the sought figure set by the course. This is achieved by creating a regularized model accounting for the movie bias or effect (namely, the bias of the movie itself, or how famous it is), the user bias or effect (as some users might be too generous or too stingy with their ratings), the genre effect (some genres are generally more likable than the others), and the time effect (how trends affected different genres across different periods of time).

The rest of the report is structured as follows. The next section, Analysis and methodology, will further detail the rationale behind selecting the above-mentioned variables. It will also provide the reader with the thinking process behind the entire project, e.g. what do I do first, what next, or why do I choose x variable, among others. This section includes some graphs to better illustrate the thinking process for this report. The next section focuses on the results. It includes explanations regarding the model used and the RMSE result. Lastly, the "Concluding remarks" section recapitulates everything explained so far, explains the limitations of this report, and provides some recommendation for future work.

It should be noted that three files in total are submit as part of this project, namely the R script, the RMD file, and the PDF file which is generated by the latter file. Since the code can be found both in the first two files, most of it has not been added in the PDF file so that it is written more succinctly and does not overwhelm the reader with too many details.

Analysis and methodology

Exploring data

After loading the data and setting up for analysis (see the R script or the RMD files for the code) data is explored to understand what we are working with.

It should be noted that the original data set was initially divided in a ratio of 9 to 1, where the former was allocated to the with which we are working with to try and predict the ratings (the "edx" data set). The latter was allocated to the ("final hold out") data set which is used to validate the results we come up with in the first data set.

The "edx" data set then is divided into two data sets (train set, and test set) of the same ratio as the first one.

Table 1: Exploring the data

	userId	movieId	rating	timestamp	title	genres
1	1	122	5	838985046	Boomerang (1992)	Comedy|Romance
2	1	185	5	838983525	Net, The (1995)	Action|Crime|Thriller
4	1	292	5	838983421	Outbreak (1995)	Action|Drama|Sci-Fi|Thriller
5	1	316	5	838983392	Stargate (1994)	Action|Adventure|Sci-Fi
6	1	329	5	838983392	Star Trek: Generations (1994)	Action|Adventure|Drama|Sci-Fi
7	1	355	5	838984474	Flintstones, The (1994)	Children|Comedy|Fantasy

Table 2: Variables summary

	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	NA	NA
	3rd Qu.:53607	3rd Qu.: 3626	3rd Qu.:4.000	3rd Qu.:1.127e+09	NA	NA
	Max. :71567	Max. :65133	Max. :5.000	Max. :1.231e+09	NA	NA

Table 3: Checking for missing data

	x
userId	FALSE
movieId	FALSE
rating	FALSE
timestamp	FALSE
title	FALSE
genres	FALSE

We can see that there are six different variables. The first table provides a glimpse at the content of the data, while the second one provides some descriptive statistics. The third table is important as it states that no missing values ("NA") are found in all the variables.

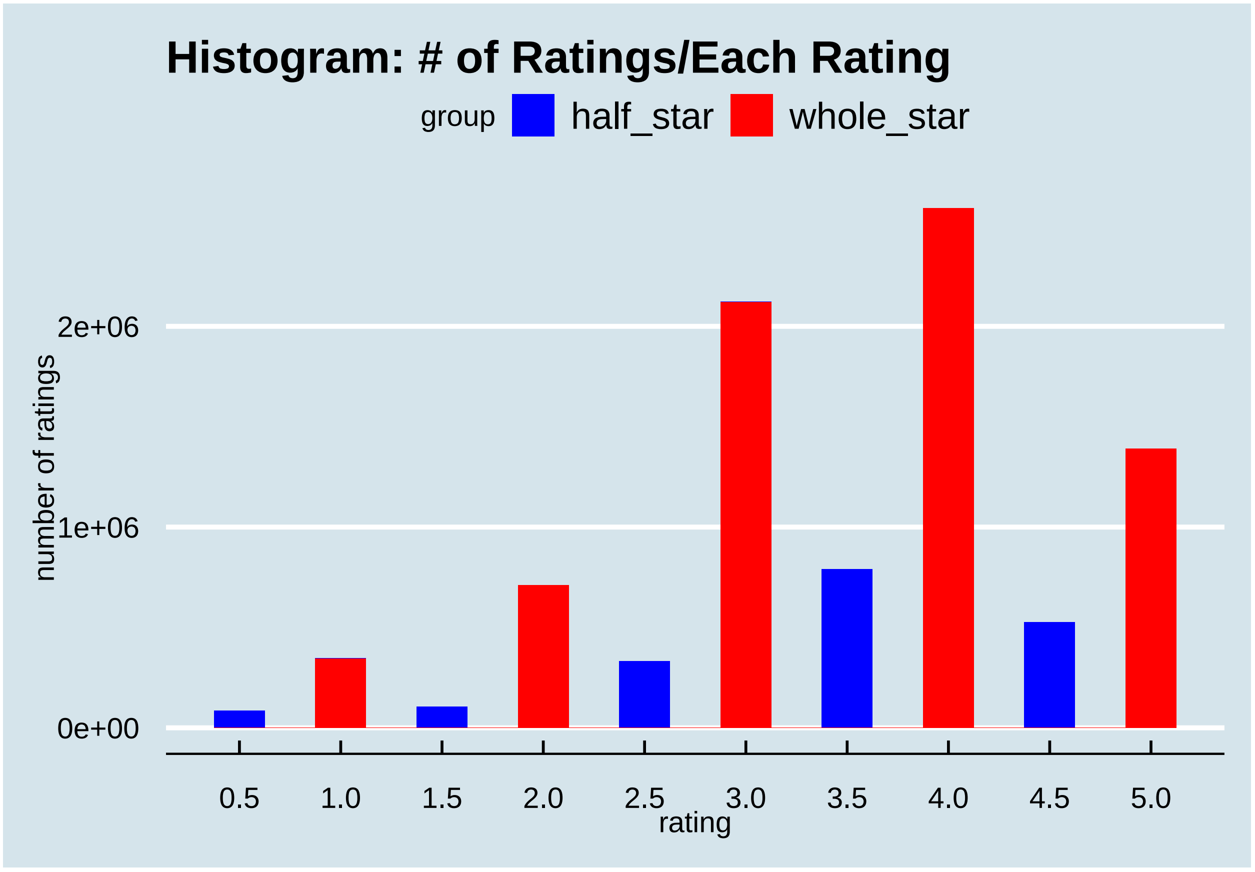
Due to biases towards movies of genres from different periods, I decide to add the "year" variable by extracting it from the "title" variable.

```r
edx <- edx %>% mutate(year=as.numeric(str_sub(title,-5,-2)))
train_set <- train_set %>% mutate(year=as.numeric(str_sub(title,-5,-2)))
test_set <- test_set %>% mutate(year=as.numeric(str_sub(title,-5,-2)))
final_holdout_test <- final_holdout_test %>% mutate(year=as.numeric(str_sub(title,-5,-2)))
```

We can already start thinking what variables would best explain the variations in the ratings. However, before we do that, it is worth analyzing further the ratings themselves.

Ratings, after all, are the variable the variation of which we are trying to explain using the other variables available in the data set. We know from the table above that the average rating in the data set is 3.5 (see the mean figure under the "rating" variable in the table above), and the median is 4.

The graph below shows that there is a tendency to give full stars, instead of half stars (Note: remember that the ratings range from 0.5 stars to 5 stars). The most common rating is the 4 star, otherwise known as the mode of the data set, which incidentally coincides with the median as well. We can notice from this graph that the users have a bias towards using full stars, rather than half stars. But, what other biases or effects can we uncover from the data, which will ultimately help us predict the ratings?



Variables for the model

As mentioned earlier, I have used four variables from the dataset to predict the ratings using different models, namely the user bias, movie bias, genre effects, and genre trends.

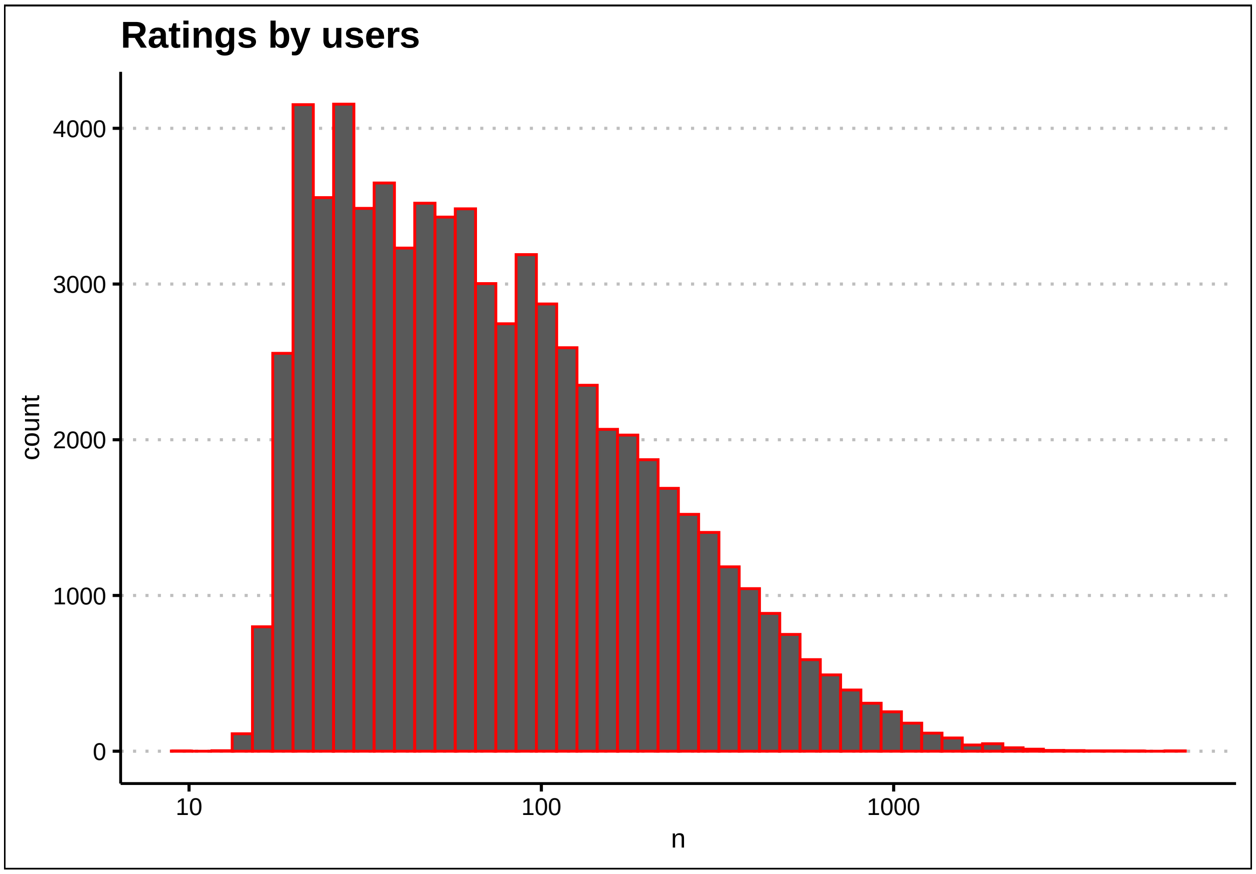
Regarding the first two, the table below shows the numbers of distinct users and distinct movies.

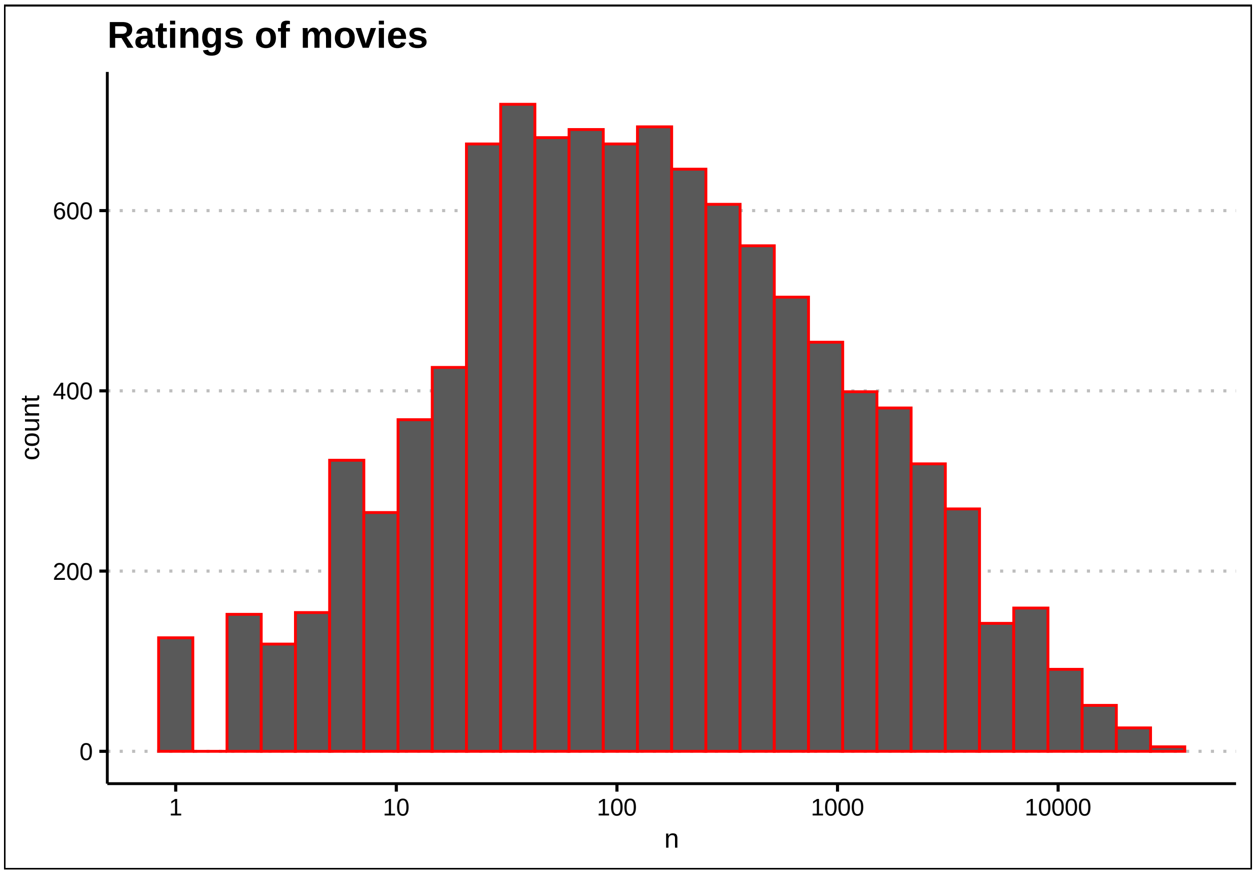
Table 4: Distinct Users and Movies

n_users	n_movies
69878	10677

Knowing that we have roughly 9 million entries (rows) or ratings, and approximately 70,000 users rating almost 11,000 movies, it is intuitive that biases exist both in the behavior of users and in the movies themselves (how famous they are).

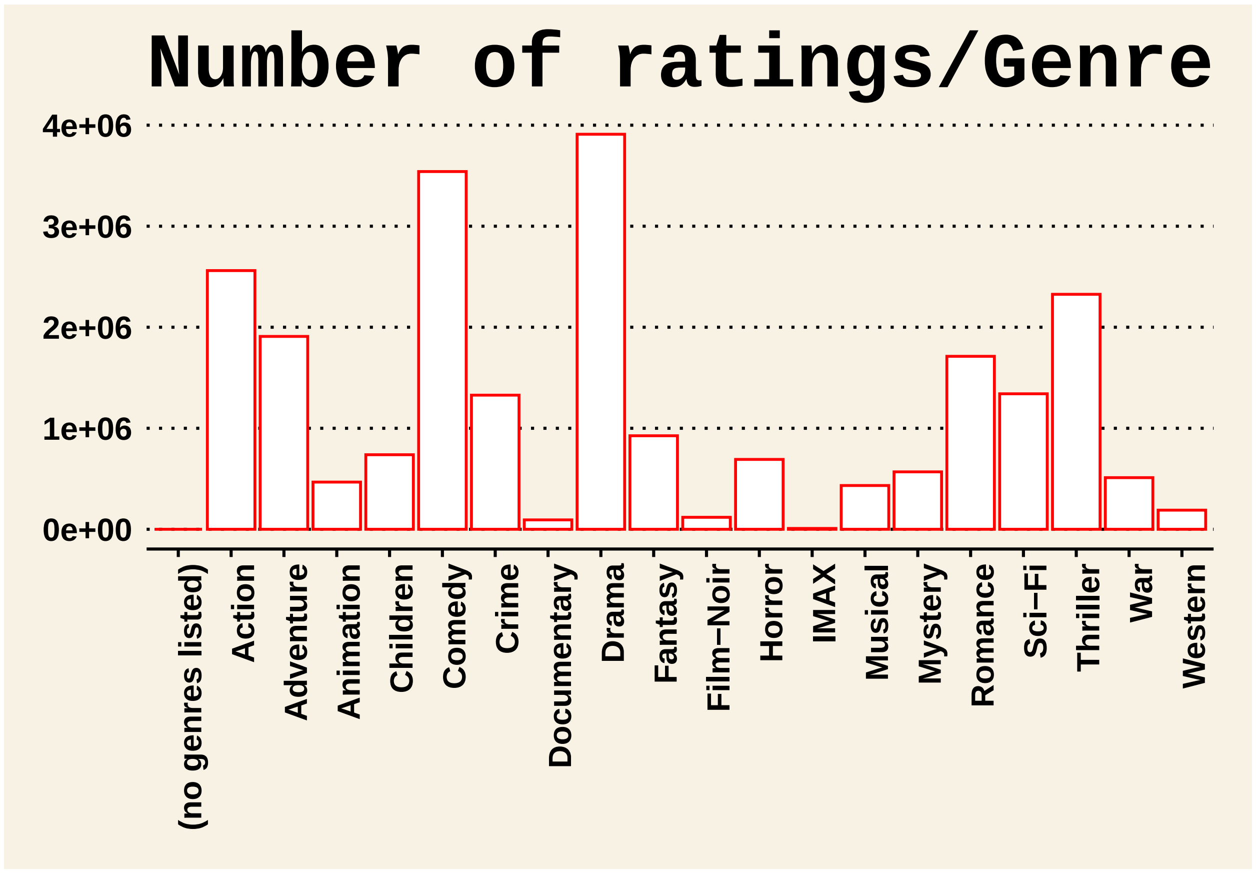
The graphs below try to uncover these biases visually.

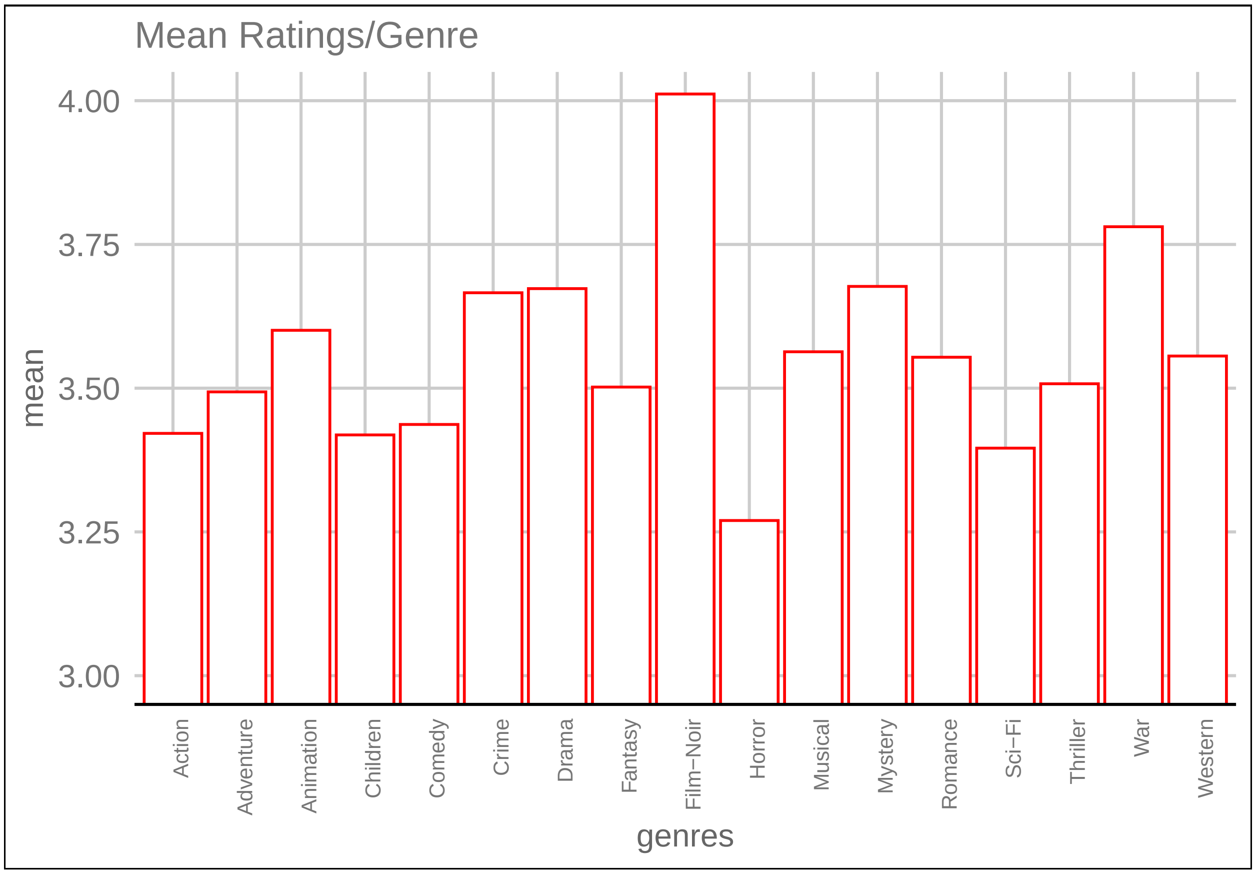




It can be noticed that some users rate too many movies compared to the others, and some movies received too many ratings compared to the others. These stark differences in both variables warrant further investigation, and for this reason they are added in the model.

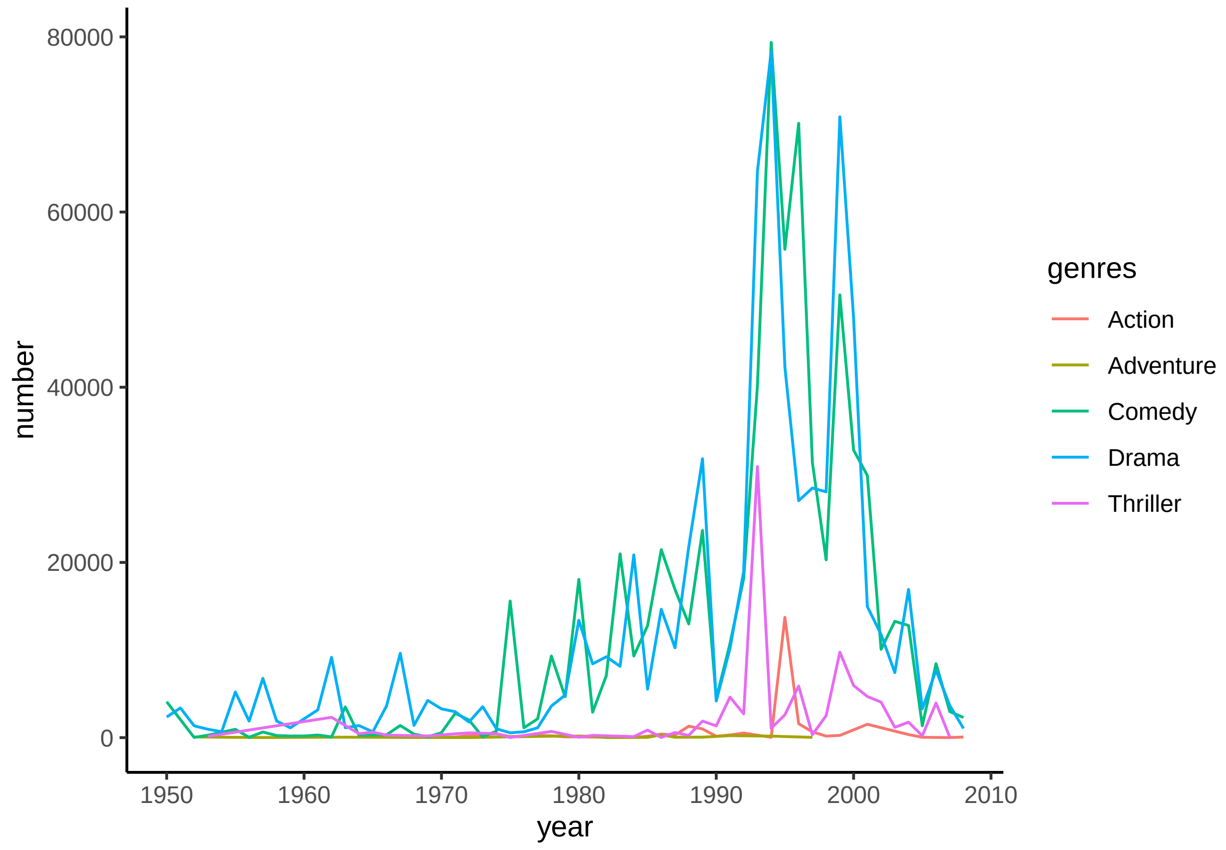
For movie aficionados it is clear that some genres are rated more (in terms of frequency) and higher. The graphs below visualize this statement.


Number of ratings/Genre
4e+06
3e+06
2e+06
1e+06
0e+00
(no genres listed)
Action
Adventure
Animation
Children
Comedy
Crime
Documentary
Drama
Fantasy
Film–Noir
Horror
IMAX
Musical
Mystery
Romance
Sci–Fi
Thriller
War
Western



It can be noticed that certain genres stand out both in terms of how much they are rated, and how high. For instance genres of Drama, Comedy and Action stand out in terms of number of ratings. Further, in terms of high ratings, genres of Film-Noir, War and Drama stand out. These insights, same as the ones for the two previous variables, provide sufficient evidence to add the genre effect as one of the explanatory variables behind the variations in the ratings.

Lastly, genres, same as movies, from different periods of time are rated differently from the users. This can be noticed also in the graph below. For instance, comedies and dramas from mid-70s to 2000s receive many ratings. Action and Thriller post-90s also receive more ratings.



80000
60000
40000
20000
0
number
1950
1960
1970
1980
1990
2000
2010
year
genres
Action
Adventure
Comedy
Drama
Thriller

Specification of the Model and Results

The cornerstone variable that is used in thinking about explain the variations in the ratings is the mean (average) of all ratings. Incidentally, the baseline model, which is explained below, and is used as a reference point for the other models, uses the ratings mean as the sole explanatory variable.

```r
mu_hat <- mean(train_set$rating)
```

The second variable is the movies bias. We have seen above the rationale behind using this variable, and we have visually shown the bias in the data. The variable is coded as "mb" and the code for creating this variable is shown below.

```r
mb <- train_set %>%
  group_by(movieId) %>%
  summarize(mb = mean(rating - mu_hat))
```

The third variable is the user bias/effects, and is defined as follows.

```r
ub <- train_set %>%
  left_join(mb, by='movieId') %>%
  group_by(userId) %>%
  summarize(ub = mean(rating - mu_hat - mb))
```

The fourth variable is the genre effects, and below is shown the specification in R.

```r
ge <- train_set %>%
  left_join(mb, by='movieId') %>%
  left_join(ub, by='userId') %>%
  group_by(genres) %>%
  summarize(ge = mean(rating - mu_hat - mb - ub))
```

The last variable used is genre preference from different periods, and it is defined as follows.

```r
te <- train_set %>%
  left_join(mb, by='movieId') %>%
  left_join(ub, by='userId') %>%
  left_join(ge, by='genres') %>%
  group_by(year) %>%
  summarize(te = mean(rating - mu_hat - mb - ub - ge))
```

Note that the specification of variables changes when we use regularization. The changes will be highlighted prior to outlining the regularization model.

Baseline model

As mentioned earlier, the first model - the baseline takes into account only the mean of ratings as the explanatory variable, and the error term is added at the end.

See below the specification of the model.

$$Rating_{u,i} = \hat{\mu} + \varepsilon_{u,i}$$

The formulae below calculate the RMSEs for model (both using the train set, and the validation set), they check whether the results are below the required threshold for the RMSE, and save the results as a tibble. The result is 1.06120, which is above the threshold.

```
baseline_model <- RMSE(train_set$rating,mu_hat)
baseline_model_validation <- RMSE(final_holdout_test$rating,mu_hat)

baseline_model < 0.86490
```

```
## [1] FALSE
```

```
baseline_model_validation < 0.86490
```

```
## [1] FALSE
```

```
rmse_summary <- tibble(method="Baseline model", RMSE = baseline_model_validation)
```

Movie, User, Genre and Genre Trends Effects Model

The second model uses all the specified variables above as explanatory variables. The model is specified as follows.

$$Rating_{u,i} = \hat{\mu} + MovieBias + UserBias + GenreEffect + GenreTrendsEffect + \varepsilon_{u,i}$$

The code below shows the predictions (for both the train set and the validation set), the RMSEs calculations, checking whether they are below the required threshold and the storing of results in the tibble created above.

```
# predictions
predicted_ratings_2 <- train_set %>%
  left_join(mb, by = "movieId")%>%
  left_join(ub, by = "userId") %>%
  left_join(ge, by = "genres") %>%
  left_join(te, by = "year") %>%
  mutate(pred = mu_hat + mb + ub + ge + te) %>%
  pull(pred)

predicted_ratings2_validation <- final_holdout_test %>%
  left_join(mb, by = "movieId")%>%
  left_join(ub, by = "userId") %>%
  left_join(ge, by = "genres") %>%
  left_join(te, by = "year") %>%
  mutate(pred = mu_hat + mb + ub + ge + te) %>%
  pull(pred)

# calculating the RMSEs

second_model <- RMSE(predicted_ratings_2, train_set$rating)
second_model_validation <- RMSE(predicted_ratings2_validation, final_holdout_test$rating)

#checking the results against the set value of the exercise

second_model < 0.86490
```

```
## [1] TRUE
```

```
second_model_validation < 0.86490
```

```
## [1] FALSE
```

```
# storing the results

rmse_summary <- bind_rows(rmse_summary,tibble(method="Movie, User, Genre and Time effects model",RMSE=se
```

When checking the results against the threshold we get the "FALSE" result, meaning that they are above the threshold.

Regularized Movie, User and Genre Effects Model

The last model that is developed for this report is the Regularized Movie, User and Genre Effects Model. The reason for using this models is that it adds a penalizing term (lambda) for large estimates. Therefore, it penalizes noisy data, and provides better estimates for our model. Firstly, we specify the lambdas and the variables before making the prediction. It should be noted that initially we create a function which loops through the set sequence for the lambdas (from 0 to 10, with increments of 0.2).

```
# Specifying the lambdas
lambda <- seq(0, 10, 0.20)

# creating the function to find the lambda which minimizes the RMSE in both train and validation sets;
# Function using the train set

RMSE_f <- sapply(lambda, function(x){
  mb <- train_set %>%
    group_by(movieId) %>%
    summarize(mb = sum(rating - mu_hat)/(n()+x))
  ub <- train_set %>%
    left_join(mb, by = "movieId") %>%
    group_by(userId) %>%
    summarize(ub = sum(rating - mu_hat - mb)/(n()+x))
  ge <- train_set %>%
    left_join(mb, by = "movieId") %>%
    left_join(ub, by = "userId") %>%
    group_by(genres) %>%
    summarize(ge = sum(rating - mu_hat - mb - ub)/(n()+x))
  predicted_ratings3 <- train_set %>%
    left_join(mb, by = "movieId") %>%
    left_join(ub, by = "userId") %>%
    left_join(ge, by = "genres") %>%
    mutate(pred = mu_hat + mb + ub + ge) %>%
    pull(pred)
  return(RMSE(predicted_ratings3, train_set$rating))
})

# Function using the validation set

RMSE_final <- sapply(lambda, function(x){
  mb <- train_set %>%
    group_by(movieId) %>%
    summarize(mb = sum(rating - mu_hat)/(n()+x))
  ub <- train_set %>%
    left_join(mb, by = "movieId") %>%
    group_by(userId) %>%
    summarize(ub = sum(rating - mu_hat - mb)/(n()+x))
  ge <- train_set %>%
    left_join(mb, by = "movieId") %>%
    left_join(ub, by = "userId") %>%
    group_by(genres) %>%
    summarize(ge = sum(rating - mu_hat - mb - ub)/(n()+x))
  predicted_ratings3 <- final_holdout_test %>%
    left_join(mb, by = "movieId") %>%
    left_join(ub, by = "userId") %>%
    left_join(ge, by = "genres") %>%
    mutate(pred = mu_hat + mb + ub + ge) %>%
    pull(pred)
  return(RMSE(final_holdout_test$rating,predicted_ratings3))
})

# Test to check if the value is below the threshold

RMSE_final[which.min(RMSE_final)] < 0.86490
```

```
## [1] TRUE
```

```
# The value is below the threshold, therefore we save it

lambda_final <- lambda[which.min(RMSE_final)]
```

We now substitute the lambda_final values, which incidentally is 5, with the x in the code above. The code below after specifying correctly the variables with the lambda value, it makes the predictions, and checks whether they are below the required threshold also on the validation set (or the "final_holdout_test" data set). This model works as it achieves a value below the required threshold. The result is 0.86485. They are stored in the tibble that summarizes the results.

```
# Subsituting X with lambda_final = 5

mb <- train_set %>%
  group_by(movieId) %>%
  summarize(mb = sum(rating - mu_hat)/(n()+lambda_final))

ub <- train_set %>%
  left_join(mb, by = "movieId") %>%
  group_by(userId) %>%
  summarize(ub = sum(rating - mu_hat - mb)/(n()+lambda_final))

ge <- train_set %>%
  left_join(mb, by = "movieId") %>%
  left_join(ub, by = "userId") %>%
  group_by(genres) %>%
  summarize(ge = sum(rating - mu_hat - mb - ub)/(n()+lambda_final))
```

```
predicted_ratings3 <- final_holdout_test %>%
  left_join(mb, by = "movieId") %>%
  left_join(ub, by = "userId") %>%
  left_join(ge, by = "genres") %>%
  mutate(pred = mu_hat + mb + ub + ge) %>%
  pull(pred)

# double-checking that the third model is below the required RMSE threshold

third_model_validation <- RMSE(predicted_ratings3, final_holdout_test$rating)

third_model_validation < 0.86490
```

```
## [1] TRUE
```

```
# saving the results and adding two figures after the decimal point to the tibble (from the default 3 t

rmse_summary <- bind_rows(rmse_summary,tibble(method="Regularized Movie, User, and Genre effects model"

rmse_summary <- rmse_summary %>% mutate(RMSE = sprintf("%.5f", RMSE))

kable(rmse_summary, caption = "Summary of RMSEs")
```

Table 5: Summary of RMSEs

method	RMSE
Baseline model	1.06120
Movie, User, Genre and Time effects model	0.86526
Regularized Movie, User, and Genre effects model	0.86485

Concluding remarks

In this project I tried to achieve an RMSE result below the required threshold of 0.86490. I tried three different models, namely the Baseline model (using only the mean of ratings) as the sole explanatory variable in the variation of ratings. The other two models are the Movie, User, Genre and Time effects model, and the Regularized Movie, User, and Genre effects model. The third model achieves an RMSE score below the required threshold. The result is 0.86485.

I could have used fewer variables in both the second and third model, but I wanted to practice more writing code and working with longer codes. And this project has taught me a lot both in terms of writing code, and thinking about modeling in general.

I tried different models as well, such as through the recommenderlab package, but my laptop could not process it due to high memory usage. Therefore for other students, I recommend trying such packages to explaining the variations of the ratings as they are different in the way that the code above is written, and they most probably can yield better results.