MovieLens Capstone Project Report

HarvardX PH125.9x - Data Science: Capstone

Seyed M. Seyedtorabi

2024-Feb-05

I. Introduction

Recommendation algorithms have become a part of our daily life through different technology and entertainment applications such as YouTube, Netflix, and Spotify. The main goal of a recommendation algorithm is to use data to enhance user experience by recommending the most relevant and engaging content for each and every specific user. These contents can be videos, movies, and songs for YouTube, Netflix, and Spotify, respectively.

The purpose of this report is to document and present the result of the analysis and modelling performed on the "MovieLens" data set in order to create a recommendation algorithm. This data set, which is provided by HarvardX as course material for the course HarvardX PH125.9x: Data Science: Capstone, contains about 10 million movie ratings for many differnet movies and by many different users. The data set also includes information such as timestamp of rating, movie title, and genres. The data set is initally patritioned into 2 parts, namely **"edx"** and **"final_holdout_test"**. Here is a quick look at the **edx** part of the data set:

```
knitr::kable(head(edx), row.names = FALSE)
```

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

In this data set, the ratings range from 0.5 to 5. Also, there are 69878 users, and 10677 movies in the data set. But since not every user rates every movie, the number of rows (9000055) is much smaller than the simple multiplication of the number of users and movies. In fact, if we turn this data set into a matrix with users as rows and movies as columns, a big majority of the entries would be missing and the sparsity of the matrix would be 0.987937. Here, the goal of a recommendation algorithm would be to estimate the missing ratings in that matrix. We will learn more about the characteristics of the data set in the Analysis section.

In this project the **edx** set is initially used to do the exploratory data analysis (EDA) to get to know the data set, then it is partitioned into training and test sets to for analysis and modelling required for building the recommendation algorithm. The metric to evaluate the performance of the model is set to be the root mean squared error **(RMSE)**:

$$RMSE = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(y_j - \hat{y}_j)^2}$$

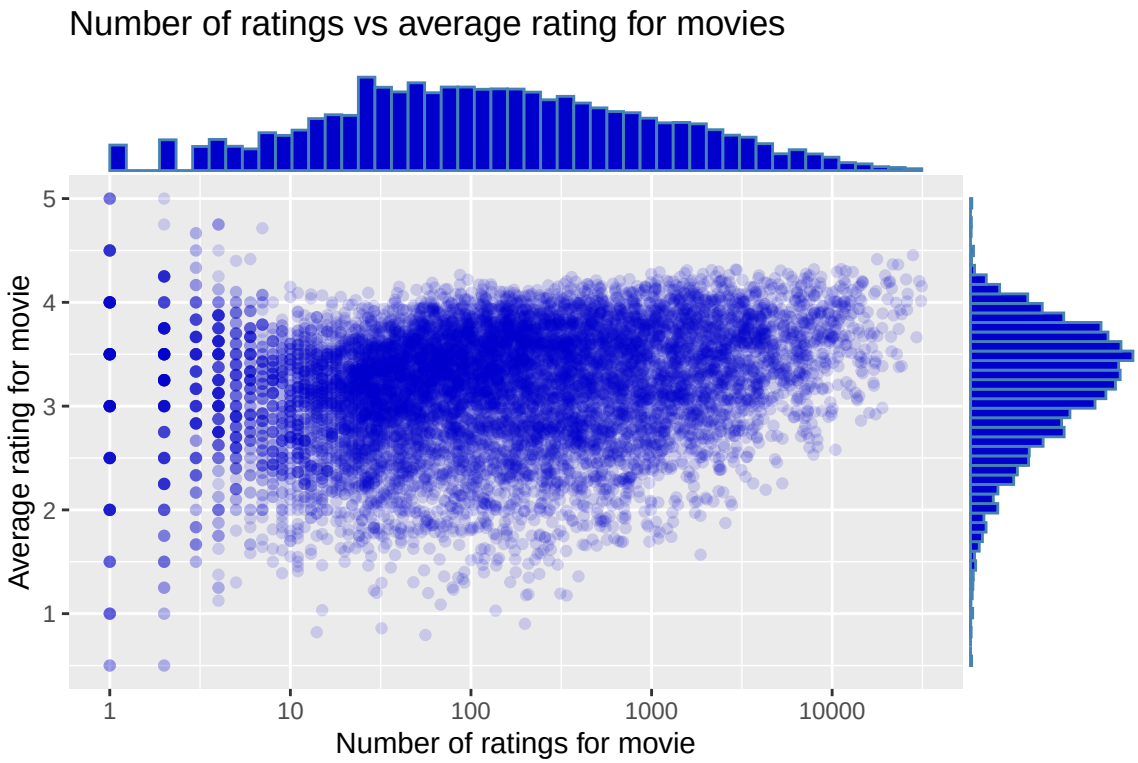
where y_j is the actual rating for the movie given by the specific user, and $\hat{y}_j$ is our estimation of that rating. After using **edx** set to build a model with sufficiently low RMSE, the model is tested on the **final_holdout_test** set to evaluate the performance of the final iteration of the model. Since the data set is very large, machine learning models from the **Caret** package were not trained on the data. Instead, a computationally more efficient approach similar to the one described in the aforementioned HarvardX course was adapted.

So far, we have outlined the problem definition, objectives, and the scope of the project. In the next sections, the results of the EDA will be documented and presented with data visualizations, the results of different iterations of the model will be shown, and the report will end with concluding remarks and potential future work for model improvement.

II. Analysis

As the first step in our journey for analysis and modelling of the data set, we need to perform some EDA and data visualizations to better understand our data set. This step is crucial, because in order to find the right features to use in our models, we need to know the different relationships that may exist between different variables in the data set.

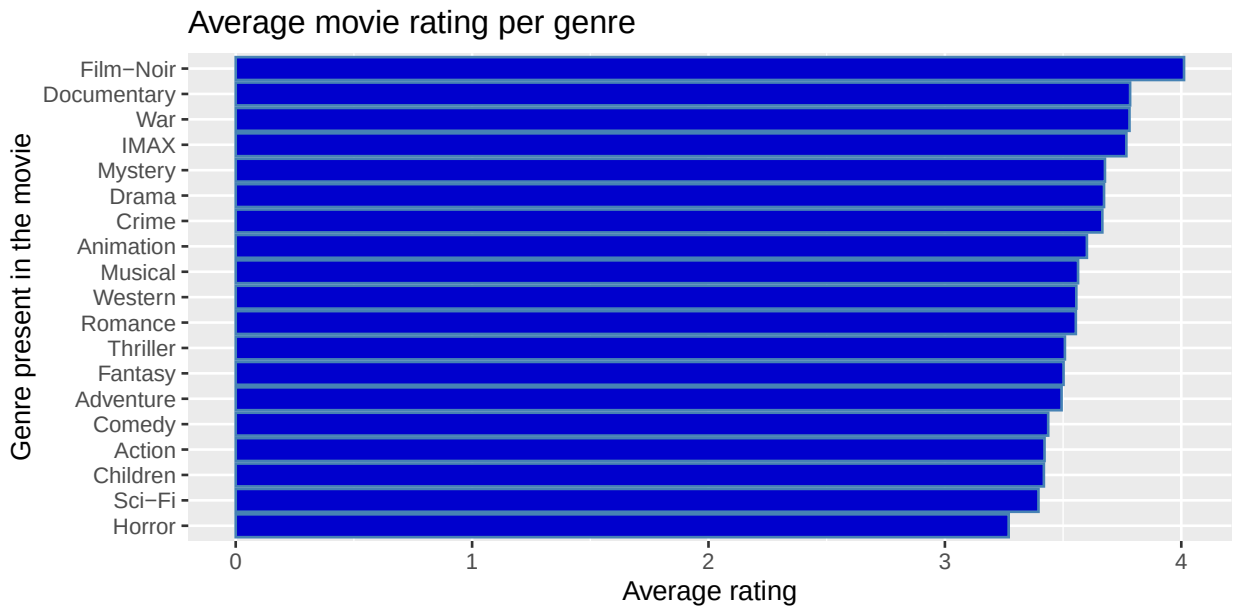
As said in the previous section, not every movie in the data set is rated by every user. This means that some movies are rated more than others. To understand the relationship between number of ratings for a movie vs how it is rated, we can have a look at the following plot:



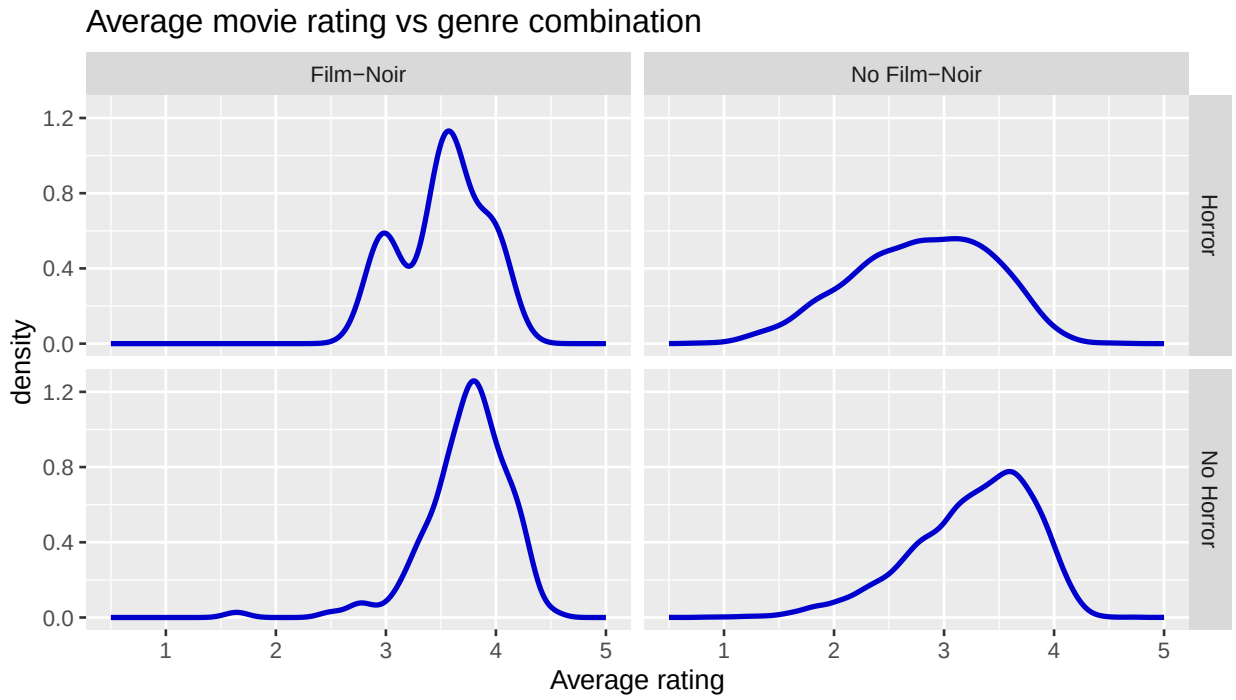
Please note that the x-axis of the plot is in logarithmic scale. This already tells us that great variation in the number of ratings for movies. Looking at our data set, we can find that 46.5% of the movies are rated less than 100 times. The plot shows that there is a relationship between the number of ratings for a movie and the average rating, especially for movies that are rated more than 100 times. We can also see that the peak of the histogram for average rating of movies is around 3.5. We can find that the sample average for the average rating of movies is 3.1917355. Since number of movie ratings are different for different movies, and there seems to be a relationship between the number of ratings and the average rating for a movie, we can conclude that the individual movies have an effect on their ratings, and this "movie effect" needs to be considered in our recommendation algorithm.

In the previous section we mentioned that the genres are also included int the data set. Each movie is assigned a combination of genres, and there are 797 distinct genre combinations in the data set. Since each genre combination is made of a number of core genres, we can try to look at the core genres instead.

The following plot shows the average rating for each **core genre** that is included in the genre combinations in our data set:

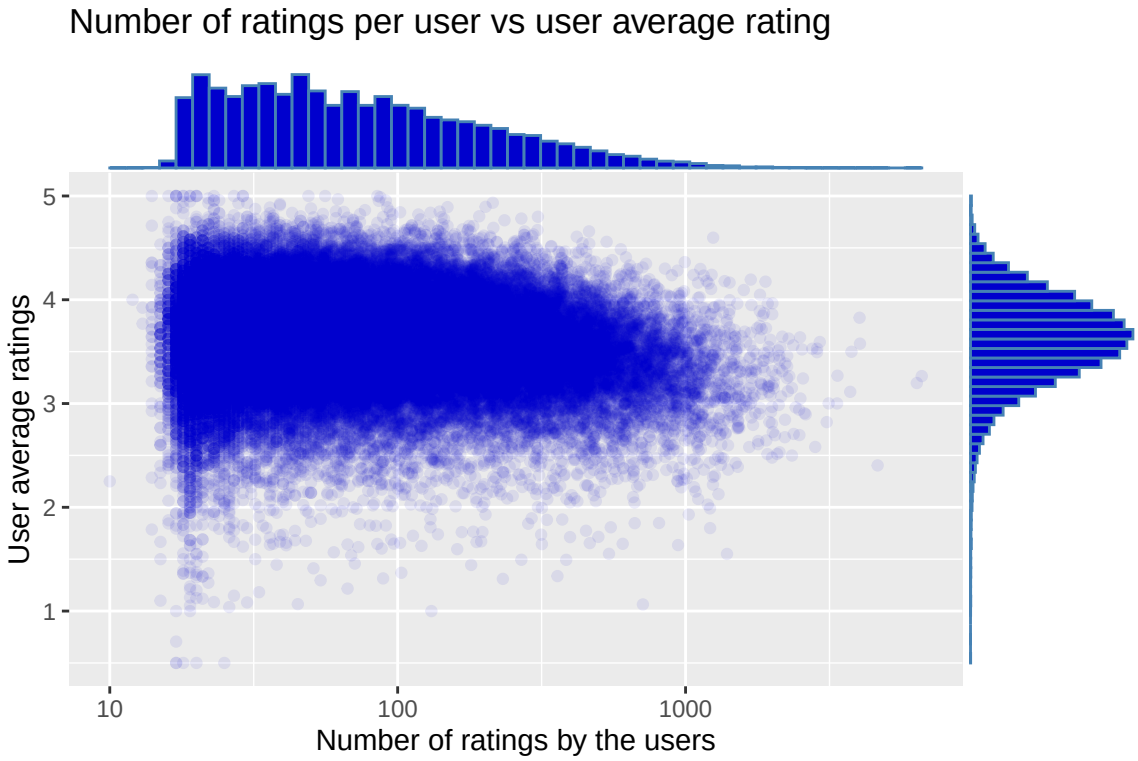


As it can be seen, mean rating for different genres can be quite different. The highest mean rating is given to "Film-Noir", while the lowest one is for movies with the genre "Horror". We can further look into the genre effect on movie rating by considering the following plot:



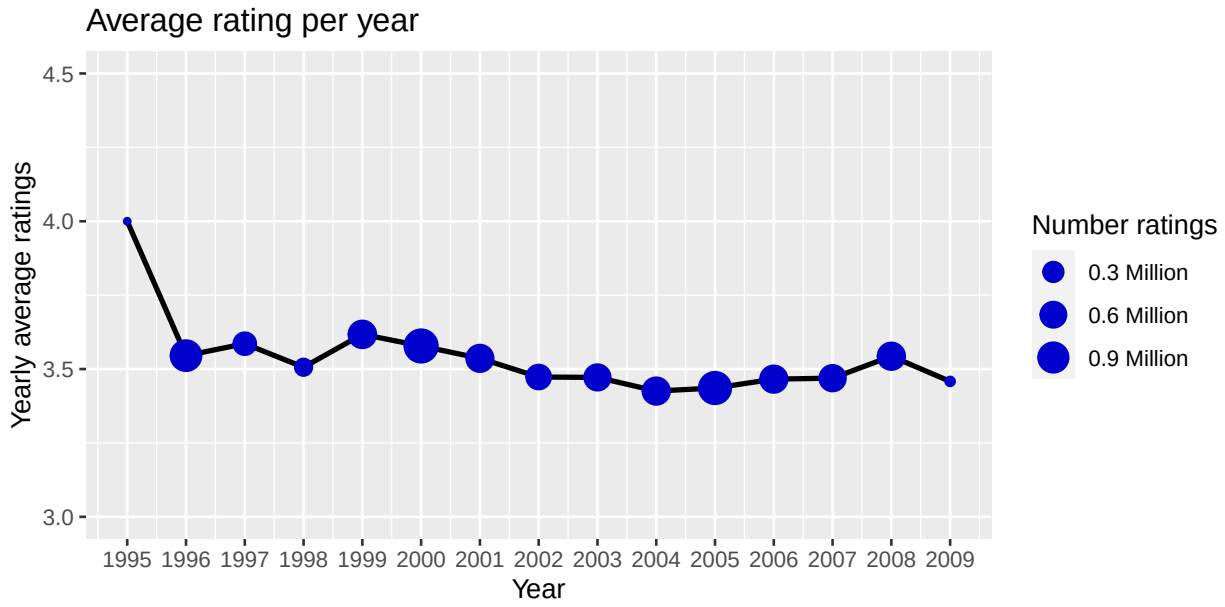
Depending on whether or not the 2 genres "Film-Noir" and "Horror" are present in movies, we can get very different ratings for movies. Looking at the two previous plots, there seems to be a genre dependency on rating. So we can use genre as a feature in our recommendation algorithm.

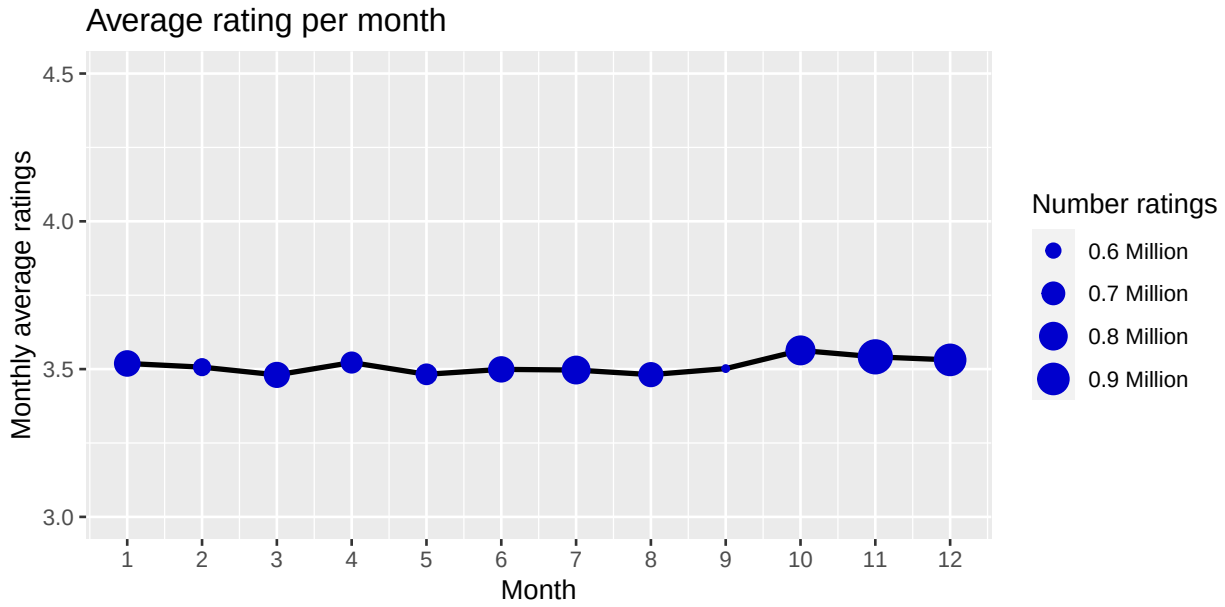
As outlined in the introduction section, a hypothetical user-movie matrix with ratings as entries would be a very sparse matrix. This implies that different users have rated different number of movies. To explore this a little bit more, we can look at the following plot:



This plot provides us a number of insights. The most important thing is that there is a great difference between different users in terms of the number of their ratings. While most users rated less than 100 movies, a few users have rated 1000 or more movies. This clearly shows the different level of engagement for users in ratings. However, the number of ratings does not seem to be correlated to the average rating given by the user. Also the distribution of average ratings by users seems to be normal. In general, this plot makes it plausible to consider a "user effect" in our recommendation algorithm, as different users behave differently.

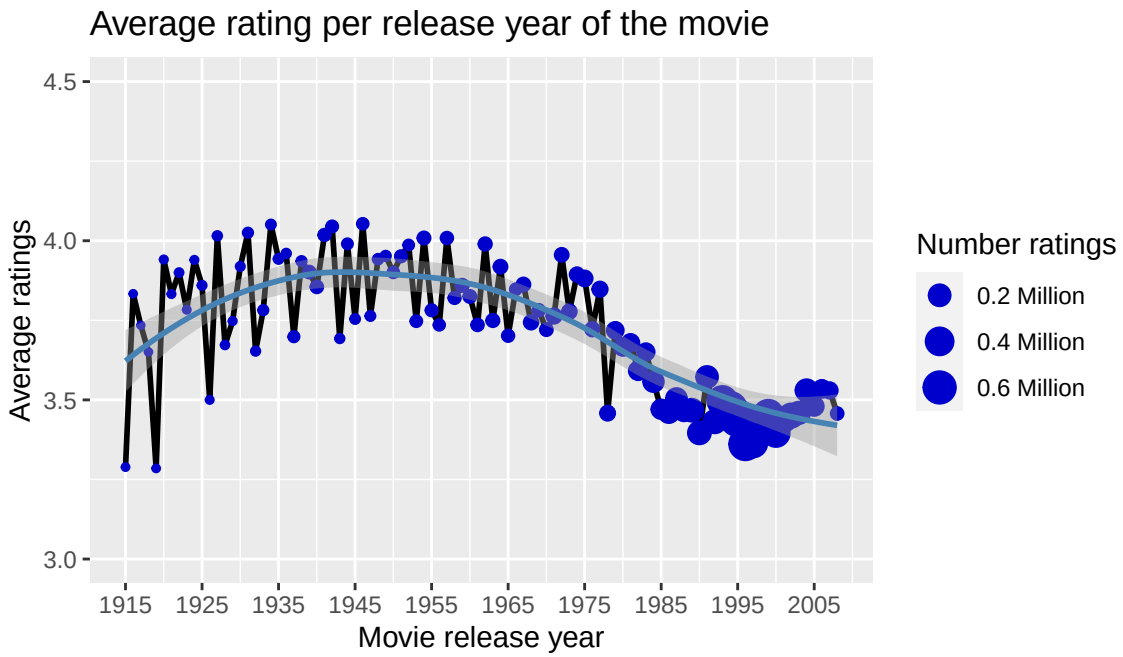
The timestamp column in the data set provides us the exact time of the rating for the ratings in the data set. One can extract and investigate the month and year of the dating to see if they affect the rating in any way, shape, or form:





The previous last plot shows that the month of the rating does not have any significant effect on the ratings. And although there might be a weak year dependency of the ratings, we are not going to consider year as an important factor in our model.

Movie title is another information presented in our data set. However, for the careful eye, the movie title contains another important information, which is the release year. The release year of the movie is provided at the end of the movie title in parenthesis, which makes it easy to extract using the string pattern detection functions from the **stingr** package. The following plot shows the release year vs average rating of movies:



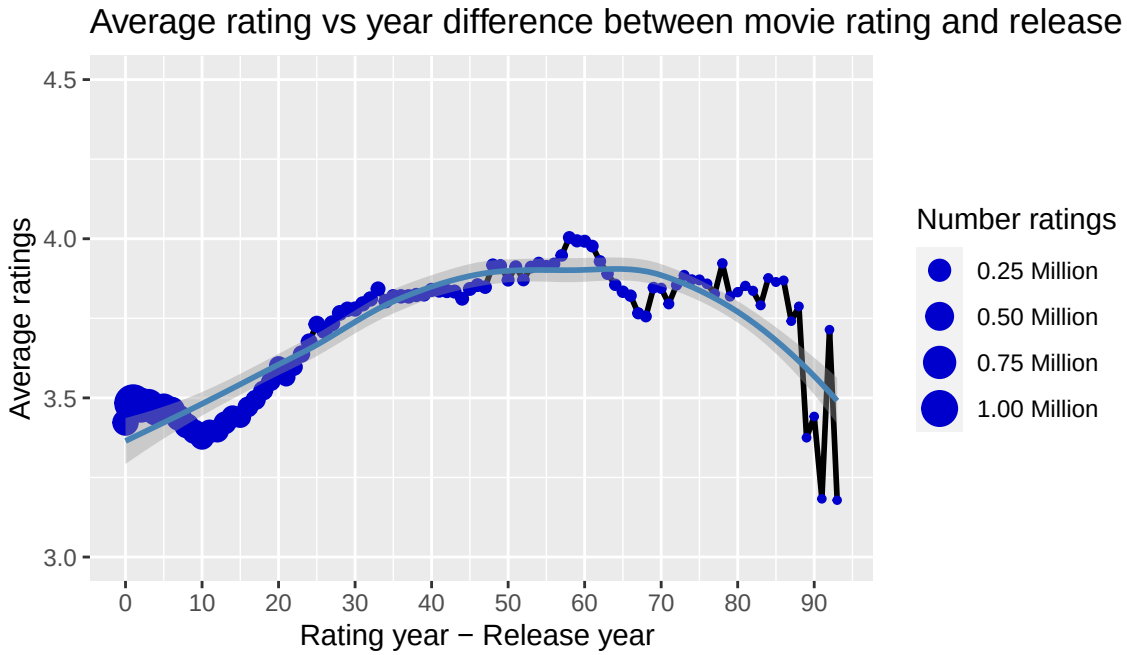
There seems to be a relationship between the release year of movies and their average ratings. Therefore, it is plausible to use the release year as a factor in our recommendation model. As an additional observation from the last plot, we can also see that the number of ratings for the recent movies is significantly more than older movies based on the size of the points in the plot.

After extracting the release year, we can also calculate the year difference between rating and release years. But for the year difference to make sense, it has to have a positive value for every rating. However, we can see that a small number of rows have negative year difference:

```
head(edx %>%
       filter(years_difference < 0) %>%
       select(title, release_year, rating_year, years_difference))
```

```
##                     title release_year rating_year years_difference
## 1 Dangerous Ground (1997)         1997        1996               -1
## 2       Relic, The (1997)         1997        1996               -1
## 3 Dangerous Ground (1997)         1997        1996               -1
## 4     Gone Fishin' (1997)         1997        1996               -1
## 5       Relic, The (1997)         1997        1996               -1
## 6 Dangerous Ground (1997)         1997        1996               -1
```

As it can be seen, the release year has been extracted successfully for these movies. So the problem is either with the rating timestamp, or with the release year provided in the movie title. We can filter out the rows with negative values int the "years_difference" columns and make the following visualization:



Similar to release year, the year difference between release and rating of a movie seems to be an important factor to take into account in our modelling task. One may argue that this year difference is unknown for ratings that have not happened yet. In other words, you cannot know the year difference between the rating and the release years if the user has not rated the movie yet. The answer to this objection would be that we can assume the user will rate the movie in the current year, and predict a rating based on that assumption. In case the user does not rate the movie this year, we can update our prediction next year, and we can do it for every year that the movie is not rated. This reveals that aging of a movie will have an effect on the predicted rating in such a model.

Based on our data exploration, we have found that it is reasonable to take into account the effect of the following factors into the recommendation model:

1) Movie
2) User
3) Genres
4) Release year
5) Difference between rating and release years (aging effect)

We will take into account these factors in different iterations of our model. However, we will start with the simple approach of predicting the average rating for all the movies in the test set. The following table shows different factors taken into account in each iteration of the model:

Iteration	Factors	Formula
0	Average	$y_{i,u} = \mu + \epsilon_{i,u}$
1	Average + Movie	$y_{i,u} = \mu + b_i + \epsilon_{i,u}$
2	Average + Movie + User	$y_{i,u} = \mu + b_i + b_u + \epsilon_{i,u}$
3	Average + Movie + User + Genre	$y_{i,u} = \mu + b_i + b_u + b_g + \epsilon_{i,u}$
4	Average + Movie + User + Genre + Year difference	$y_{i,u} = \mu + b_i + b_u + b_g + b_{yd} + \epsilon_{i,u}$
5	Average + Movie + User + Genre + Year difference + Release year	$y_{i,u} = \mu + b_i + b_u + b_g + b_{yd} + b_{rl} + \epsilon_{i,u}$

Iteration	Factors	Formula
6	Same as iteration 5 but regularized	$y_{i,u} = \mu + b_i + b_u + b_g + b_{yd} + b_{rl} + \epsilon_{i,u}$

In the formulas above, μ represents the average rating of the movies in the respective data set. The subscripts i and u represent the movie i and the user u. The terms $y_{i,u}$ and $\epsilon_{i,u}$ represent, respectively, the rating and the random variations thereof for the movie i and the user u. The terms b with different subscripts represent the effect of different respective factors taken into account. The last row is for the final iteration of the model, in which the effect of different factors are regularized. Regularization is a widely used technique in data science, and here we use it to make sure that a small number of observations related to a specific movie, user, or other mentioned factors do not have a disproportionately large effect on the rating predictions.

III. Results

In this section, we will present the results related to each of the iterations of the model. We will also show the performance of the final model in terms of RMSE for the **final_holdout_test** data set. However, we do not use this data set anywhere else, instead to develop our models, we split the **edx** data set into the training set and the test set:

```
test_indices <- createDataPartition(edx$rating, times=1, p=0.2, list=FALSE)
test_set <- edx[test_indices,]
train_set <- edx[-test_indices,]
```

We also need to make sure that there is no new user or movie in the test set compared to training set:

```
test_set <- test_set %>%
  semi_join(train_set, by = "movieId") %>%
  semi_join(train_set, by = "userId")
```

We will use the **test_set** to evaluate the model performance for each iteration, and once we are satisfied with one of the iterations, we use the model to evaluate the performance on the **final_holdout_test** set.

III.I. Iteration 0

Iteration 0 is the simple approach of predicting the average rating for all movies, as mentioned in the previous section. Here is the formula:

$$y_{i,u} = \mu + \epsilon_{i,u}$$

We use the **train_set** to calculate μ, and that would be our first rating prediction for all movies in the **test_set**. Then we calculate the RMSE. The result is shown in the following table:

Iteration	Factors	RMSE
0	Average	1.060704

Needless to say, the RMSE value for this model is not satisfactory and we need to iterate on the model to improve the RMSE value.

III.II. Iteration 1

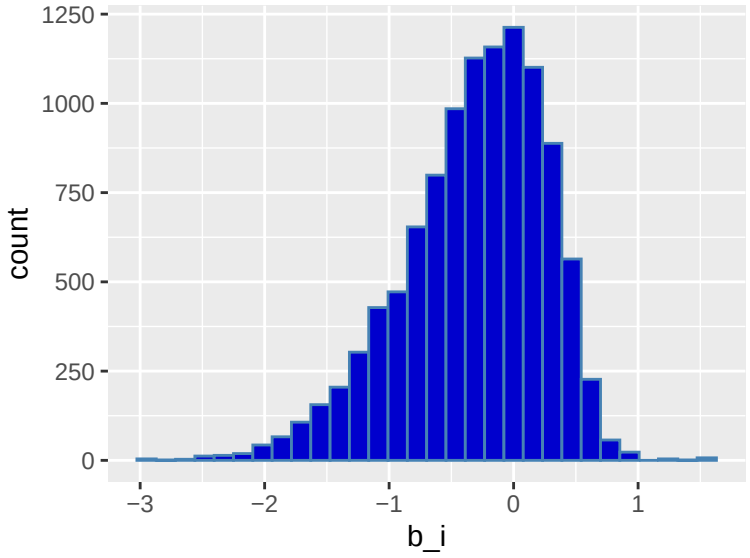
Iteration 1 takes into account the average rating and the effect of each movie. The governing equation in this model would be:

$$y_{i,u} = \mu + b_i + \epsilon_{i,u}$$

We group the **train_set** on movieId to calculate b_i for each movie, and then we use them together with μ to predict the ratings in the **test_set**. The resulting RMSE is provided in the following table:

Iteration	Factors	RMSE
0	Average	1.0607045
1	Average + Movie	0.9437144

Here is the distribution of b_i values:



The RMSE is better now, but far from satisfactory. Next, we take into account the user effect.

III.III. Iteration 2

Iteration 2 takes into account the factors in iteration 1, and additionally takes into account the user effect. The governing equation in this model would be:

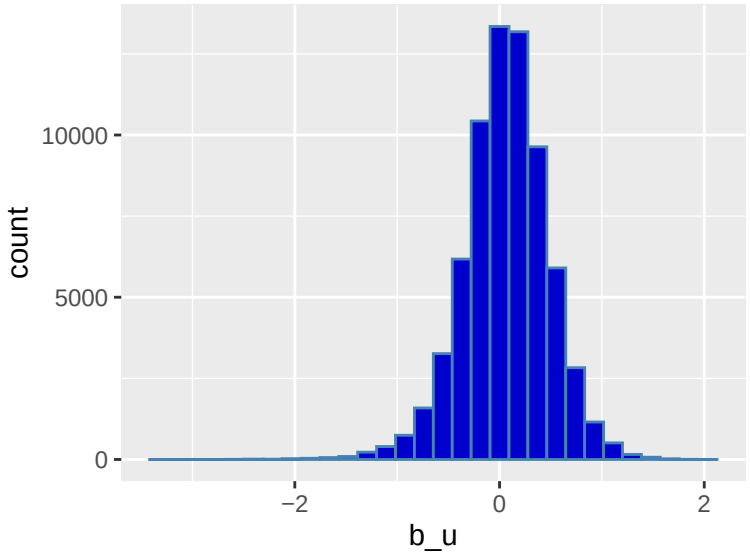
$$y_{i,u} = \mu + b_i + b_u + \epsilon_{i,u}$$

We group the **train_set** on userId to calculate b_u for each user, and then we use them together with the effect of factors from the last iteration to predict the ratings in the **test_set**. The resulting RMSE is provided in the following table:

Iteration	Factors	RMSE
0	Average	1.0607045
1	Average + Movie	0.9437144

Iteration	Factors	RMSE
2	Average + Movie + User	0.8661625

Here is the distribution of b_u values:



We still need to improve RMSE.

III.IV. Iteration 3

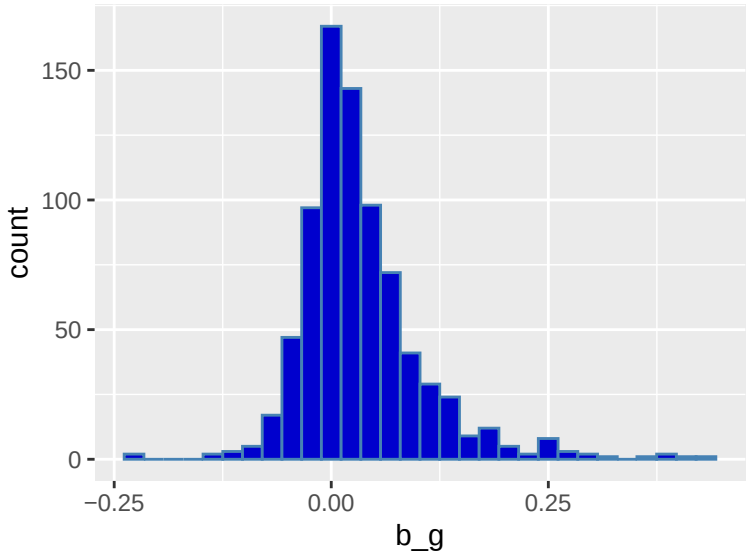
Iteration 3 takes into account the factors in iteration 2, and additionally takes into account the genre effect. The governing equation in this model would be:

$$y_{i,u} = \mu + b_i + b_u + b_g + \epsilon_{i,u}$$

We group the **train_set** on genres to calculate b_g for each user, and then we use them together with the effect of factors from the last iteration to predict the ratings in the **test_set**. The resulting RMSE is provided in the following table:

Iteration	Factors	RMSE
0	Average	1.0607045
1	Average + Movie	0.9437144
2	Average + Movie + User	0.8661625
3	Average + Movie + User + Genre	0.8658242

Here is the distribution of b_g values:



Next, we improve our RMSE. by taking into account the effect of difference in years of rating and release.

III.V. Iteration 4

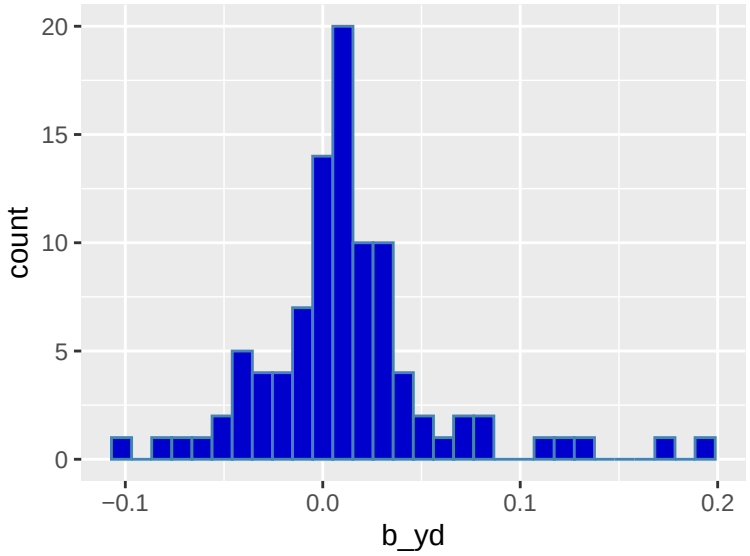
Iteration 4 takes into account the factors in iteration 3, and additionally takes into account the aging (year difference between rating and release) effect. The governing equation in this model would be:

$$y_{i,u} = \mu + b_i + b_u + b_g + b_{yd} + \epsilon_{i,u}$$

We group the **train_set** on **year difference** to calculate b_{yd} for each year difference, and then we use them together with the effect of factors from the last iteration to predict the ratings in the **test_set**. The resulting RMSE is provided in the following table:

Iteration	Factors	RMSE
0	Average	1.0607045
1	Average + Movie	0.9437144
2	Average + Movie + User	0.8661625
3	Average + Movie + User + Genre	0.8658242
4	Average + Movie + User + Genre + Year difference	0.8653950

Here is the distribution of b_{yd} values:



Next, take into account the release year of the movie to further improve our RMSE.

III.VI. Iteration 5

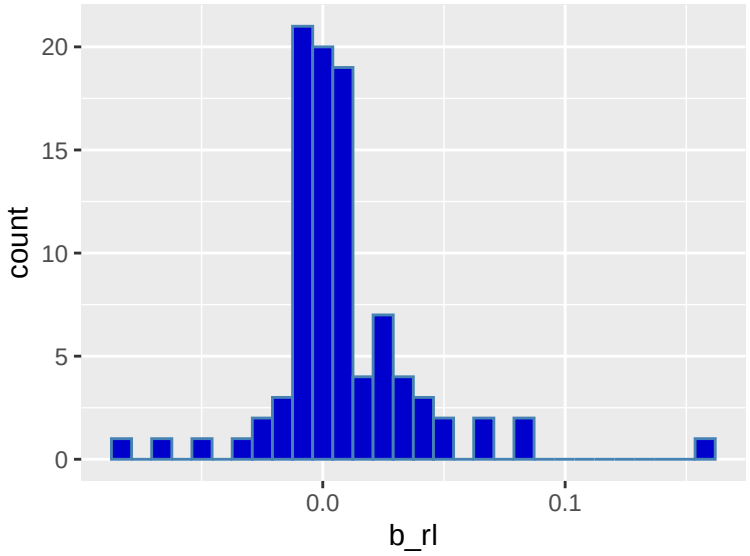
Iteration 5 takes into account the factors in iteration 4, and additionally takes into account the release year effect. The governing equation in this model would be:

$$y_{i,u} = \mu + b_i + b_u + b_g + b_{yd} + b_{rl} + \epsilon_{i,u}$$

We group the **train_set** on **release year** to calculate b_{rl} for each release year, and then we use them together with the effect of factors from the last iteration to predict the ratings in the **test_set**. The resulting RMSE is provided in the following table:

Iteration	Factors	RMSE
0	Average	1.0607045
1	Average + Movie	0.9437144
2	Average + Movie + User	0.8661625
3	Average + Movie + User + Genre	0.8658242
4	Average + Movie + User + Genre + Year difference	0.8653950
5	Average + Movie + User + Genre + Year difference + release year	0.8652384

Here is the distribution of b_{rl} values:

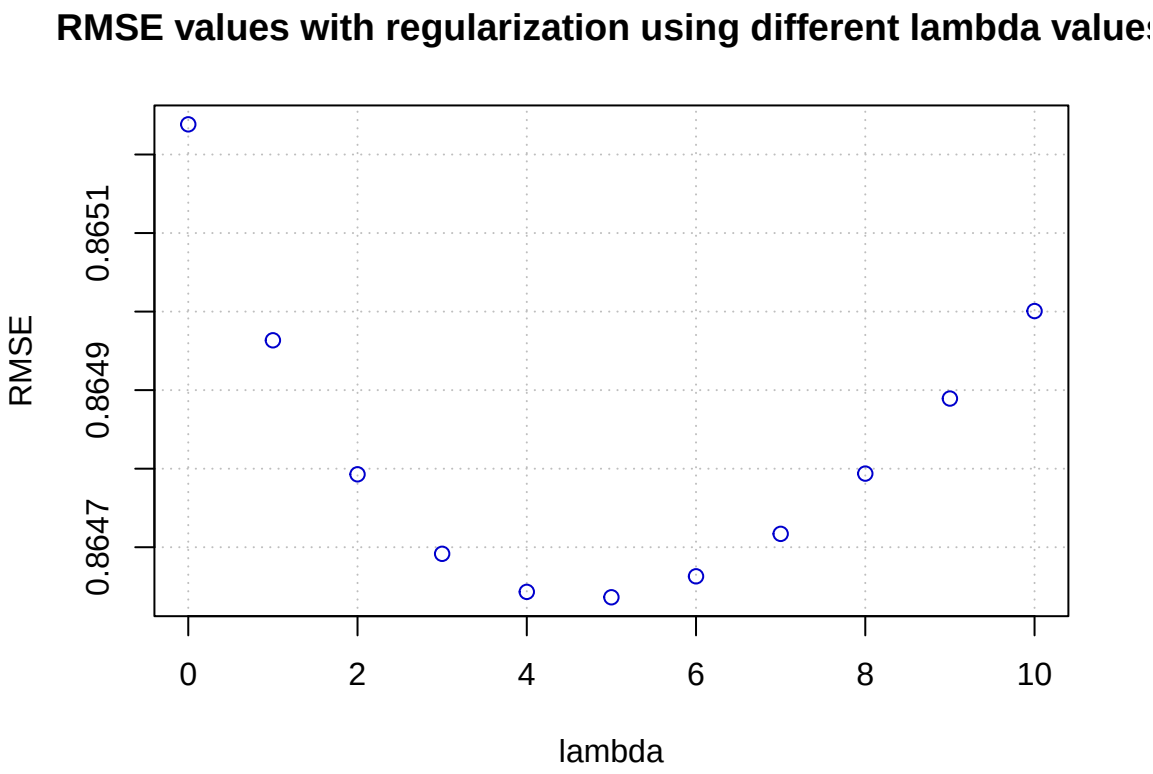


To further improve our RMSE, we need to regularize the effects used in iteration 5.

III.V. Iteration 6

Iteration 6 takes into account the effect of factors in iteration 5 and follows the same formulation. However, the effect of factors are regularized. This means that we introduce a penalty factor λ, which is used to penalize the effect of factors when they are based on very limited numbers of observations. In other words, the effect of factors can shrink closer to zero in case their corresponding observations are very limited in number. Regularization does not have any significant effect when the effects are based on sufficient numbers of observations.

For a good regularization, we need to find the optimal penalty λ, to make sure we decrease RMSE as much as possible. For this reason, we try out all integer values from 0 to 10. The following plot shows the value of RMSE under different penalty factors λ:



We are now able to predict the rating of movies in the **test_set** with an RMSE of 0.8646363 when we use the optimal λ value of 5. The following table summarizes our results so far. Note that these RMSE are calculated based on predictions for ratings in the **test_set**. We will use the latest iteration of the model to predict the ratings in the **final_holdout_test** set.

Iteration	Factors	RMSE
0	Average	1.0607045
1	Average + Movie	0.9437144
2	Average + Movie + User	0.8661625
3	Average + Movie + User + Genre	0.8658242
4	Average + Movie + User + Genre + Year difference	0.8653950
5	Average + Movie + User + Genre + Year difference + release year	0.8652384
6	Same as iteration 5 but regularized	0.8646363

III.VI. Testing the final model

Now that we have a satisfactory RMSE, we can use the entire **edx** set to train our model, and we use the **final_holdout_test** set for testing it by calculating the RMSE based on predicted and real ratings in the **final_holdout_test** set. Needless to say, we will use the optimal λ value of 5 for regularization. When we do so, we find the RMSE value of 0.8638654, which satisfies the requirement to obtain the full grade on model performance for this capstone project. The following table summarizes all of our results:

Iteration	Factors	RMSE
0	Average	1.0607045
1	Average + Movie	0.9437144
2	Average + Movie + User	0.8661625
3	Average + Movie + User + Genre	0.8658242
4	Average + Movie + User + Genre + Year difference	0.8653950
5	Average + Movie + User + Genre + Year difference + release year	0.8652384
6	Same as iteration 5 but regularized	0.8646363
7	Same as iteration 6 but tested on final_holdout_test set	0.8638654

IIII. Conclusion

In this project, we analyzed the MovieLens data set and used it to develop the basis for a movie recommendation model. To do so, firstly, we performed EDA and different visualizations on the data set to learn about relationships and correlations that exist in the data. Then we adapted an iterative approach to improve the performance of the model at each step by taking into account the effect of different factors such as movie, genre, and release year. Once we included all factors that made sense based on our EDA, we regularized the effect of each factor after finding the optimal regularization penalty value λ through a 1-D grid search. In the end, we used our best performing model and put it to test against the **final_holdout_test** set, and obtained an RMSE of 0.8638654. This RMSE value can be further improved by performing matrix factorization through PCA or SVD and including the resulting influences into our model.