Building a Movie Recommendation System using the MovieLens 10M Dataset

Harvardx PH125.9x Capstone Project

Tibor Nagy

4/12/2021

Contents

1. Introduction

The purpose of this project is to use a publicly available dataset to build a movie recommendation system as a part of the Harvardx Professional Data Scientist program. The project is based on the “MovieLens 10M Dataset” which is released by GroupLens research lab in the Department of Computer Science and Engineering at the University of Minnesota.
The dataset is a part of the much larger Movielens dataset, it contains 10 million ratings and 100,000 tag applications applied to 10,000 movies by 72,000 users. It is widely used in education, because working with this dataset provides a great opportunity to the students to exercise and develop their data science skills such as data wrangling, visualization, data analysis.
The dataset is divided to two subsets by the R code provided by Harvardx. The larger subset (called “edx”) contains the 90% of the data and can be used as a training set during the development of the machine learning algorithm. The smaller subset (called “validation”) contains the remaining data. It cannot be used for training purposes, it is only for validation of the final algorithm.
In this project we try to build a recommendation system, which predict user ratings on the validation data. The goal during the training of the algorithm is to achieve as small RMSE (Root Mean Squared Error) as possible. The final goal is to achieve RMSE < 0. 86490.

2. Exploratory Data Analysis

At first, we need to be familiarized with the two datasets (”edx” and “validation”) we are working with.

2.1 Dataset Dimensions

We will use the “edx” as the training set and save the “validation” for evaluating the RMSE of the final algorithm.

Table 1: Dataset Dimensions

Dataset	No. of Rows	No of Columns
edx	9000055	6
validation	999999	6

2.2 Missing Data

It is important to know if the dataset contains any missing values, because they can cause us difficulties during the algorithm development, so we need to address them.

Table 2: Number of missing data in each columns

	x
userId	0
movieId	0
rating	0
timestamp	0
title	0
genres	0

The above table tells us we are lucky, there are no missing values.

2.3 Dataset Structure

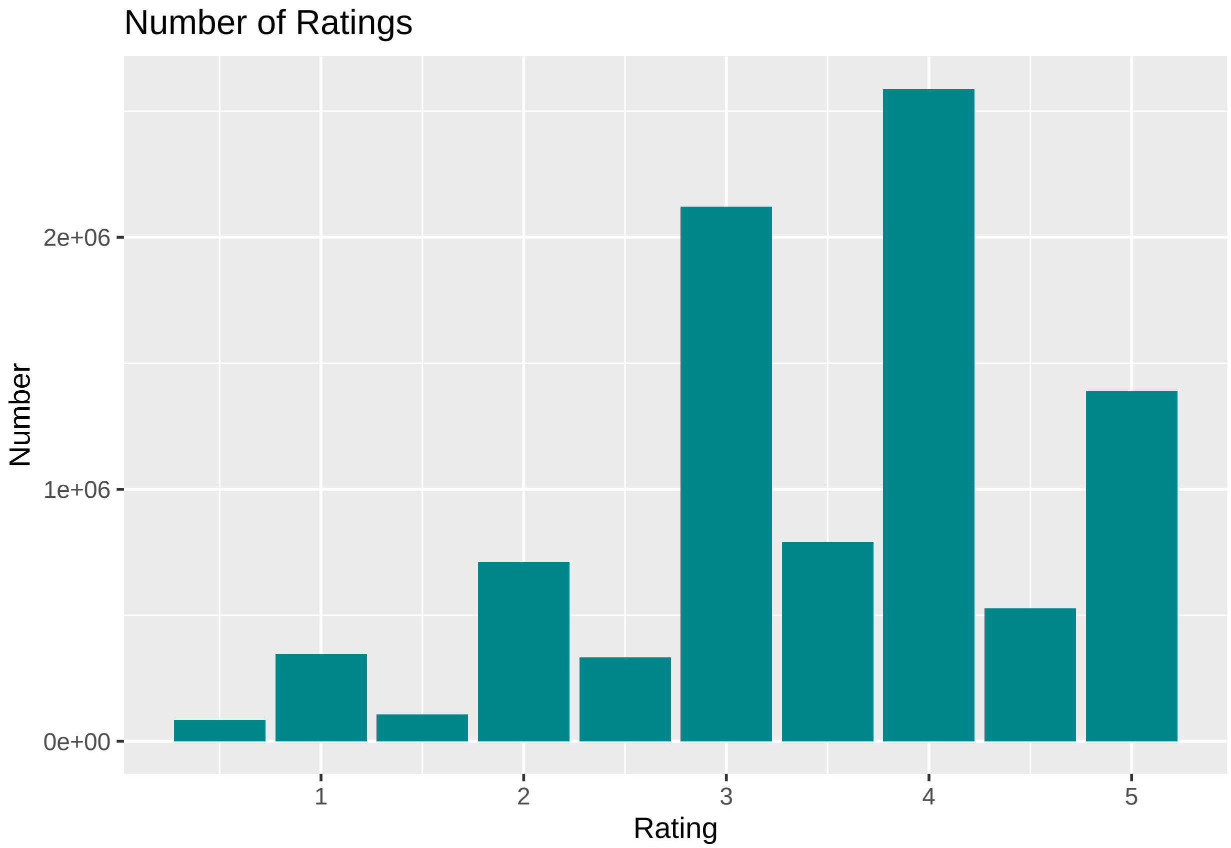
The datasets are in tidy format. They contain six columns (features) and give us the following information:

- userId <integer>: the users are anonymized. The feature contains a unique ID for each user
- movieId <numeric>: contains unique ID for each movie
- rating <numeric>: contains the rating given by the user to the movie. Ratings are scales from 0.5 to 5 with 0.5 increments
- timestamp <integer>: contains the timestamp for the time of rating
- title <character>: contains the title of the movie and the year of release
- genres <character>: a pipe-separated list, describes the genre or genres that the movie belongs to

Table 3: Preview of the dataset

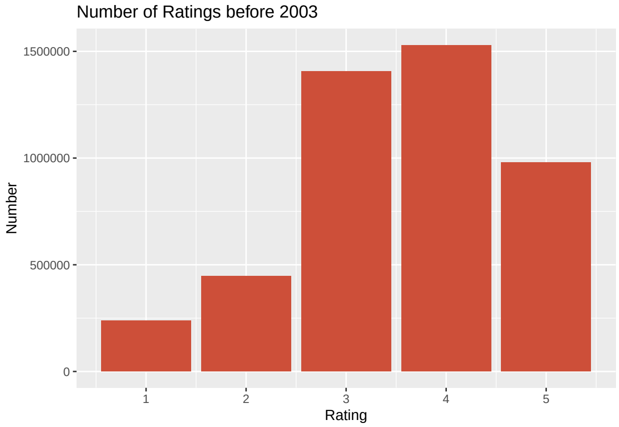
userId	movieId	rating	timestamp	title	genres
1	122	5	838985046	Boomerang (1992)	Comedy|Romance
1	185	5	838983525	Net, The (1995)	Action|Crime|Thriller
1	292	5	838983421	Outbreak (1995)	Action|Drama|Sci-Fi|Thriller
1	316	5	838983392	Stargate (1994)	Action|Adventure|Sci-Fi
1	329	5	838983392	Star Trek: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

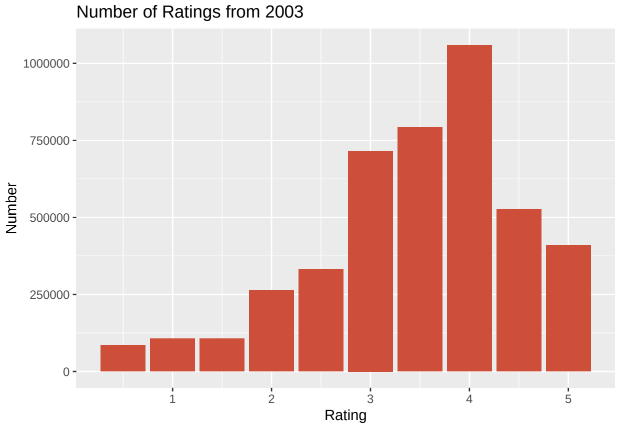
2.4 Distribution of Ratings



The above chart tells as that the half star ratings are less common than the whole star ratings. The distribution is biased towards the high ratings. 3-star rating is the most common, followed by and 4-star and 5-star ratings.

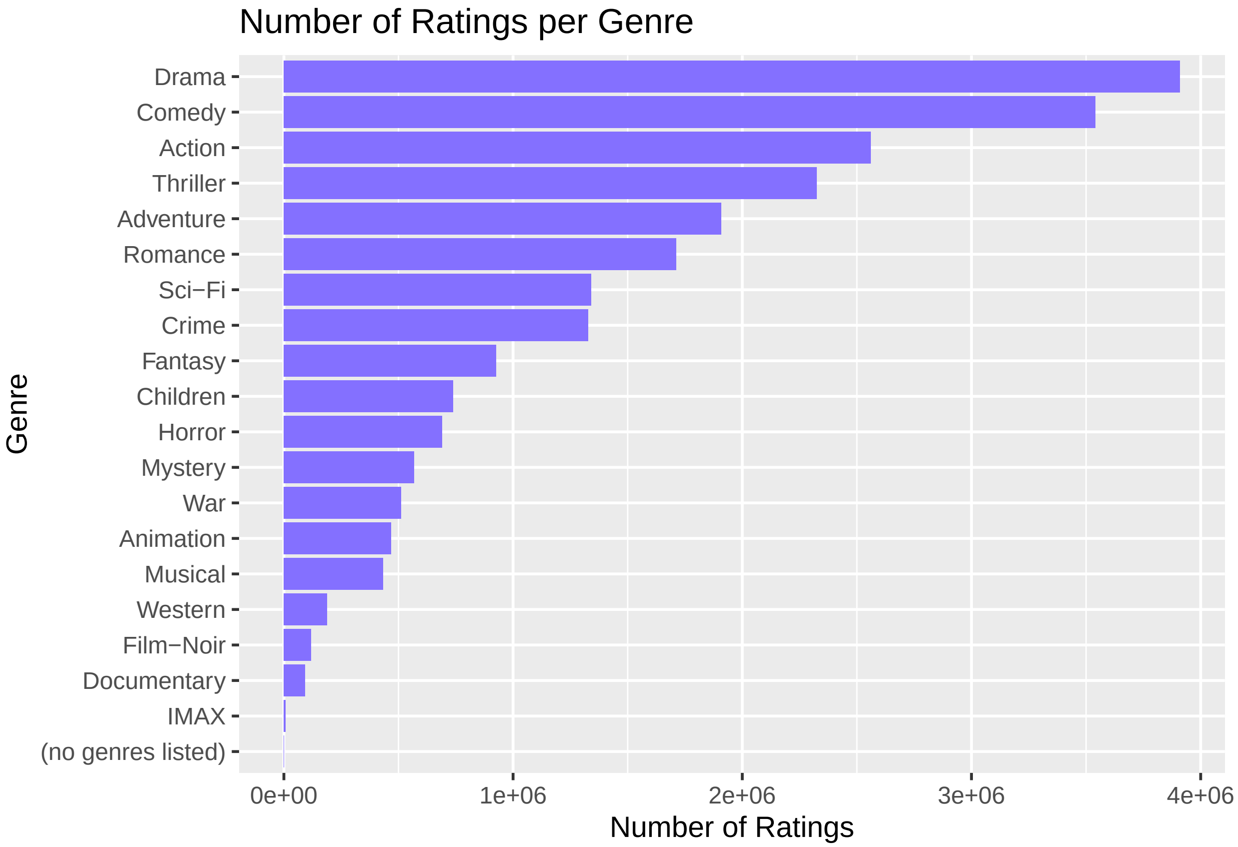
The reason of the relatively small number of half star ratings is that the dataset is combined from two smaller datasets. The ratings before 2003 are scales from 1 to 5 star with whole star increments. The half star increments were introduced in 2003.





2.5 Distribution of Genres

In the “Genres” column contains pipe-separated lists, selected from several genres. All the possible genres and their distribution can be seen in the chart below:



From the plot above we see that the distribution of genres is not even. Drama, Comedy and Action are the most common genres.

2.6 Best and worst movies

Table 4: 10 Best movies

Title	Mean Rating	Number of Ratings
Blue Light, The (Das Blaue Licht) (1932)	5.0	1
Fighting Elegy (Kenka erejii) (1966)	5.0	1
Hellhounds on My Trail (1999)	5.0	1
Satan’s Tango (SÃ¡tÃ¡ntangÃ³) (1994)	5.0	2
Shadows of Forgotten Ancestors (1964)	5.0	1
Sun Alley (Sonnenallee) (1999)	5.0	1
Constantine’s Sword (2007)	4.8	2
Human Condition II, The (Ningen no joken II) (1959)	4.8	4
Human Condition III, The (Ningen no joken III) (1961)	4.8	4
Who’s Singin’ Over There? (a.k.a. Who Sings Over There) (Ko to tamo peva) (1980)	4.8	4

Table 5: 10 Worst movies

Title	Mean Rating	Number of Ratings
Accused (Anklaget) (2005)	0.5	1
Besotted (2001)	0.5	2
Confessions of a Superhero (2007)	0.5	1
Hi-Line, The (1999)	0.5	1
War of the Worlds 2: The Next Wave (2008)	0.5	2
Hip Hop Witch, Da (2000)	0.8	14
SuperBabies: Baby Geniuses 2 (2004)	0.8	56
Disaster Movie (2008)	0.9	32
From Justin to Kelly (2003)	0.9	199
Criminals (1996)	1.0	2

The above tables list the 10 worst and the 10 best movies and their average ratings as well as the number of ratings for each movie. We see that these movies were rated by very few users, they are mostly obscure ones. During the development of our algorithm, we need to address this size effect because it can negatively affect the accuracy of our algorithm. Predictions based on only a few ratings increase the uncertainty.

3. Models

In this chapter we start from the simplest possible algorithm (predicting the same rating for all movies regardless of user), and continue with more and more complex models until we obtain the desired RMSE < 0.86490.

We will use the "edx" dataset to train the algorithms, we save the validation set for final evaluation. The edx set will be partitioned to a train_set and a test_set. They will contain the 80% and the remaining 20% of the data respectively.
The RMSE will be calculated with the following equation:

$$RMSE = \sqrt{\frac{1}{N}\sum(\hat{y}_{u,i} - y_{u,i})^2}$$

3.1. Average Rating Model

In this model we will predict the same rating for all movies regardless of user. We want to minimize the RMSE, so in this case we will predict the average of all ratings for all movies and users μ (3.51). Our model looks like this:

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

Where μ is the average of all ratings, the $\epsilon_{u,i}$ is the "noise" centered at 0 which describes the random variability.

We collect the results of our models into one summary table:

Table 6: Model accuracy

Method	RMSE
Average Rating Model	1.0599

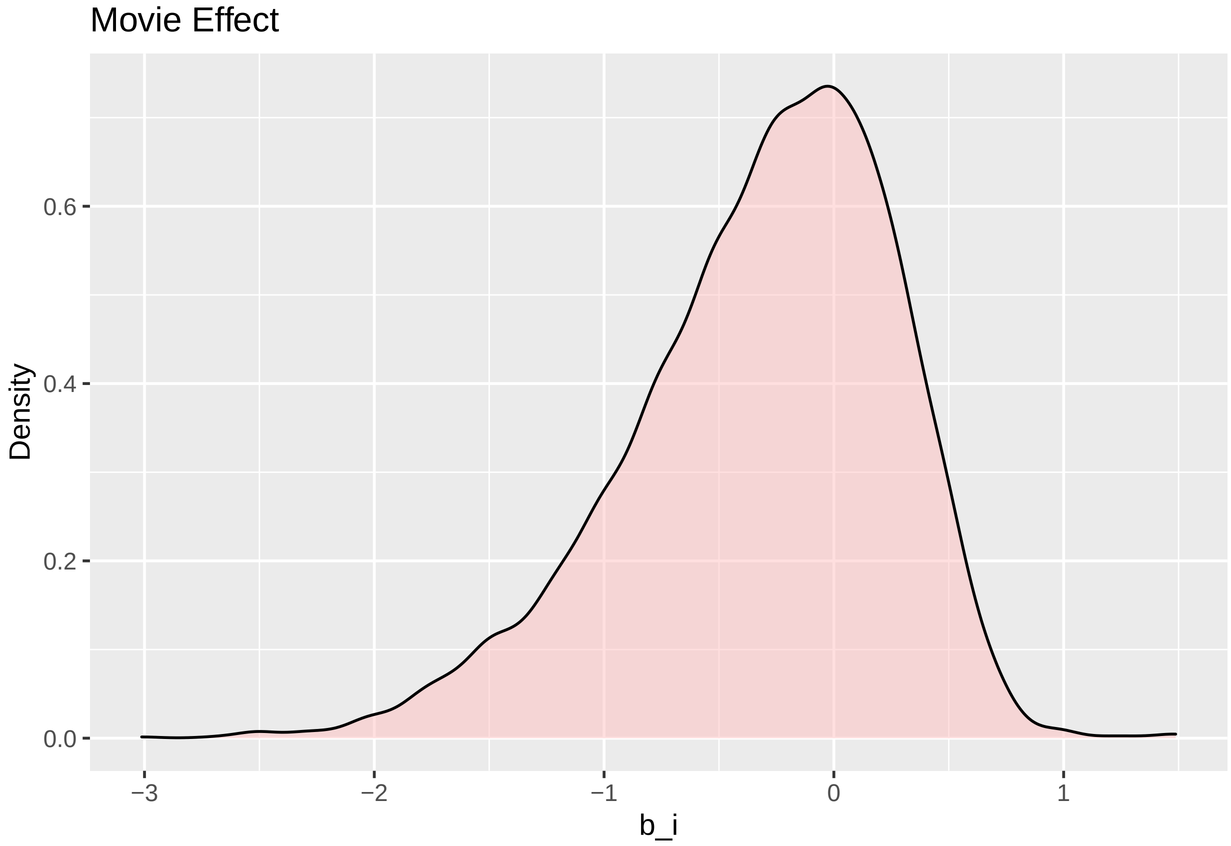
3.2. Movie Effect Model

The next step to obtain more accuracy is to augment our first model with the movie effect term. This method is based on the observation, that some movies have higher ratings in general than the other. To address this movie effect, we introduce a term b_i to represent the deviation of the average rating of movie i from the overall mean of ratings μ. Our second model will look like this:

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

Where μ is the average of all ratings, b_i is the movie effect and the $\epsilon_{u,i}$ is the "noise".

We plot the distribution of the b_i values just to verify there are variability across movies, which means adding the movie effect to our model will improve our prediction.



We can see there is significant variation across movies.

Table 7: Model accuracy

Method	RMSE
Average Rating Model	1.0599
Movie Effect Model	0.9437

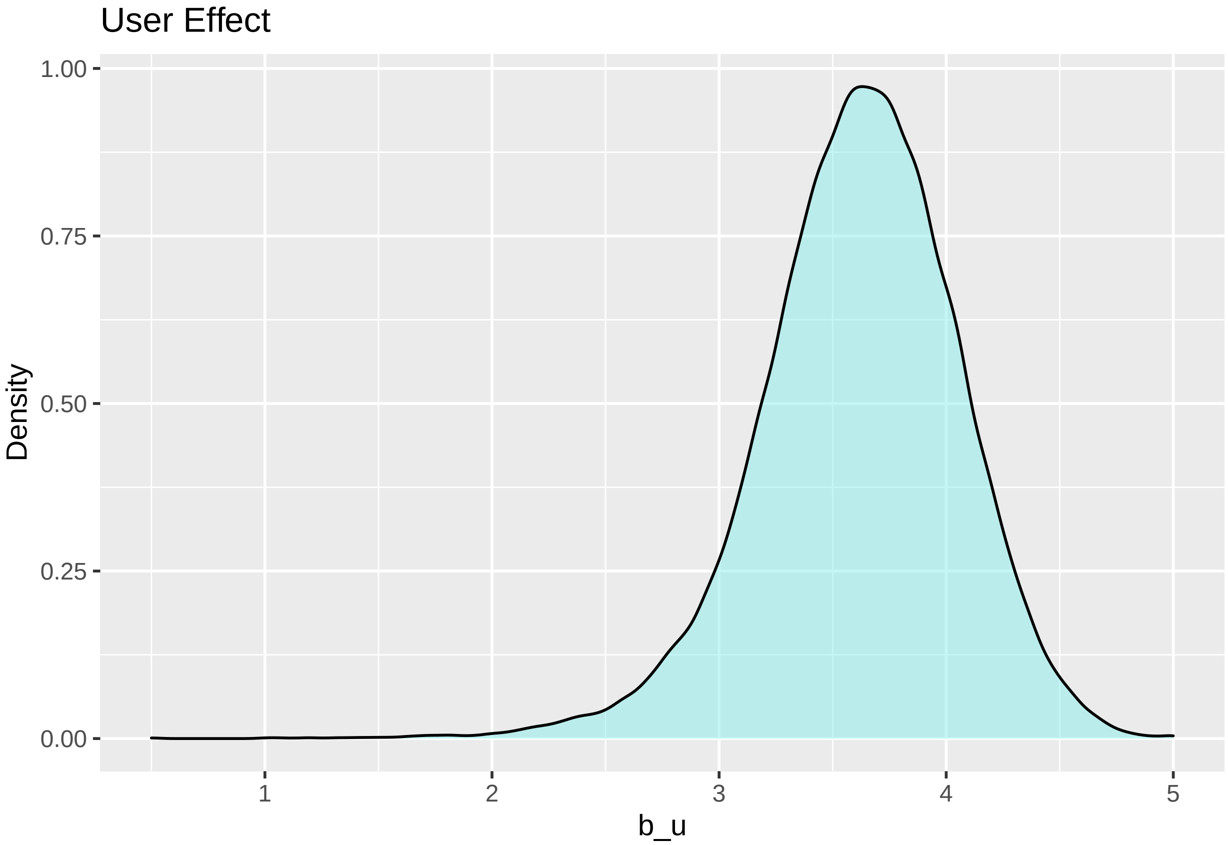
We still did not obtain our goal, but we have a better RMSE compared to the first model. This means we are on the right track, but we need a bit more detailed model.

3.3. Movie and User Effect Model

To improve the accuracy, we can also take the user's properties into account. Some users give higher ratings on average than the others. We augment our model with the user specific effect b_u. b_u represents the deviation of the average rating of user u from the overall mean of ratings μ.

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

We plot the average rating for users to see if there is variability across users.



We can see there is significant variation across users.

Table 8: Model accuracy

Method	RMSE
Average Rating Model	1.0599
Movie Effect Model	0.9437
Movie and User Effect Model	0.8659

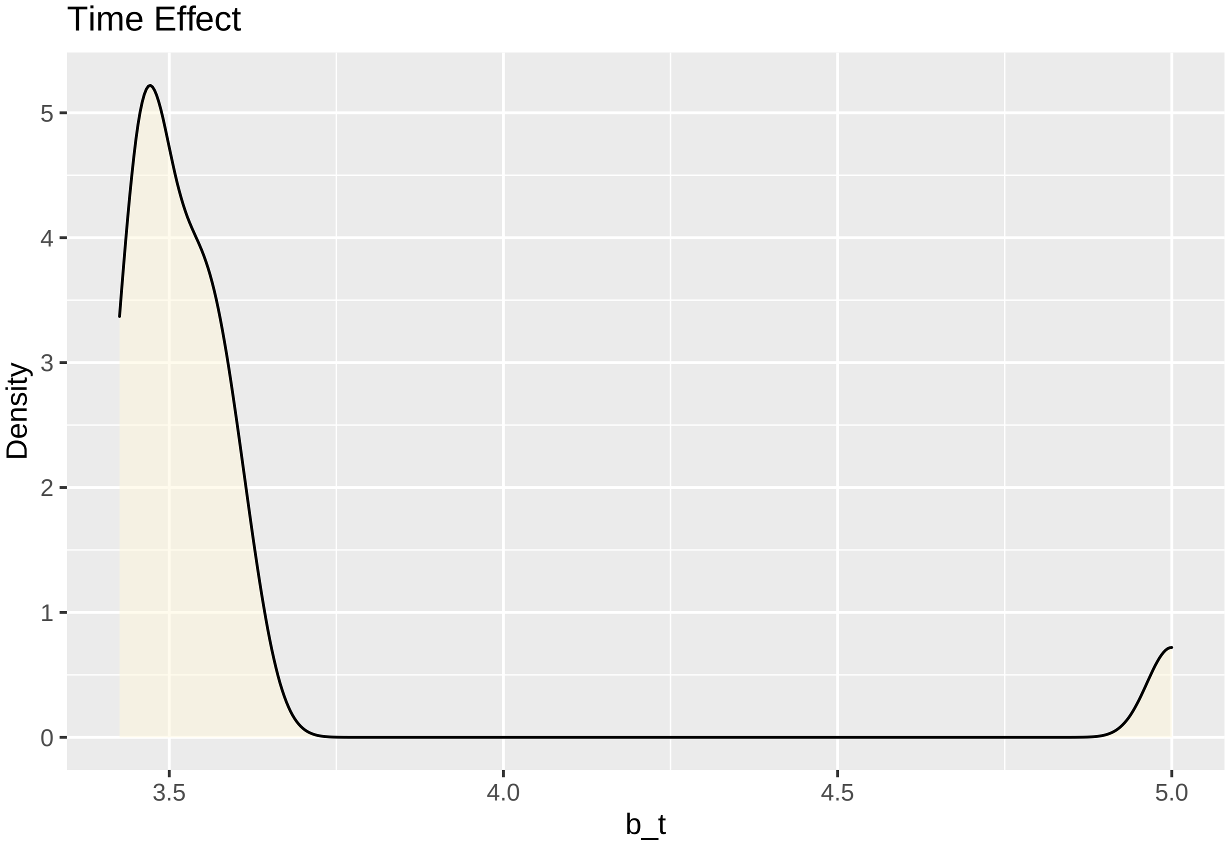
We can see this is our most accurate model yet. It provides RMSE = 0.8659. But unfortunately, we still not obtained our target RMSE.

3.4. Movie, User and Time Effect Model

The next effect we consider is the influence of the release time of the movie on the ratings. Maybe the users are more lighthearted for the old classics and stricter for the newer movies. We augment our model with the time specific effect b_t. bt represents the deviation of the average rating of movies released in the same year from the overall mean of ratings.

$$Y_{u,i,t} = \mu + b_i + b_u + b_t + \epsilon_{u,i,t}$$

Lets see the density plot of b_t:



We can see there is some variation across years.

Table 9: Model accuracy

Method	RMSE
Average Rating Model	1.0599
Movie Effect Model	0.9437
Movie and User Effect Model	0.8659
Movie, User and Time Eff. Model	0.8659

The introduction of the time effect to our model decreased the RMSE very very slightly. We do not see any change in 4 decimals. Despite this, we keep the effect in the model, maybe we will have better luck on the validation set. To achieve the goal RMSE of this project, we need to introduce a more advanced technique in our final model.

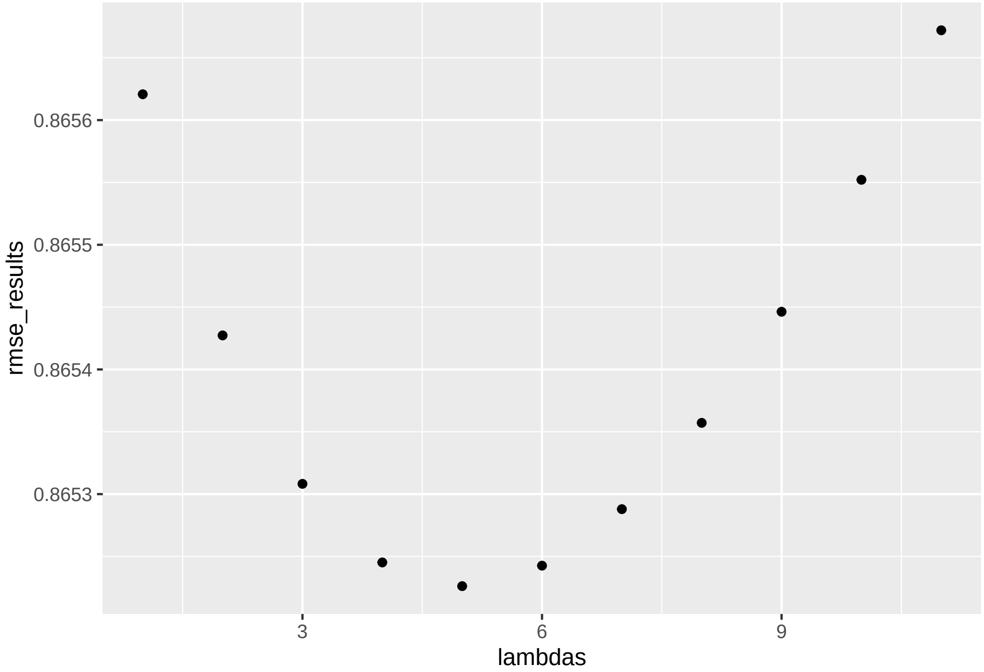
3.5. Regularized Model

As we saw in chapter 2.6 there are several movies that were rated by only a few users. Therewith there are several users who rated only a few movies. And there are years when only a few movies were released. Predictions based on only a few ratings increase the uncertainty and therefore the inaccuracy of our model. Regularization allows us to penalize these effects, so in our final model we will use the following formula:

$$Y_{u,i,t} = \mu + b_{i,reg} + b_{u,reg} + b_{t,reg} + \epsilon_{u,i,t}$$

We introduce a penalty (λ) for b_i that influenced by movies with very few ratings, for b_i which is influenced by users who only rated a small number of movies and for b_t that influenced by release years when only a few movies were released. In the above equation we noted these regularized factors with $b_{i,reg}$, $b_{i,reg}$ and $b_{i,reg}$ respectively. λ is a tuning parameter, we can choose it to minimize the RMSE. To reduce computation time, we will tune λ in two steps.
At first, we will try λs from a scale from 0 to 10, with whole number increments, then we will set a new narrower interval around the λ that minimizes the RMSE, and we will use increments 0.2.

The λ that minimizes the RMSE at the first step is: 5



So we set a new scale from 4 to 6 with increments 0.2.

Our final λ is: 5 :)

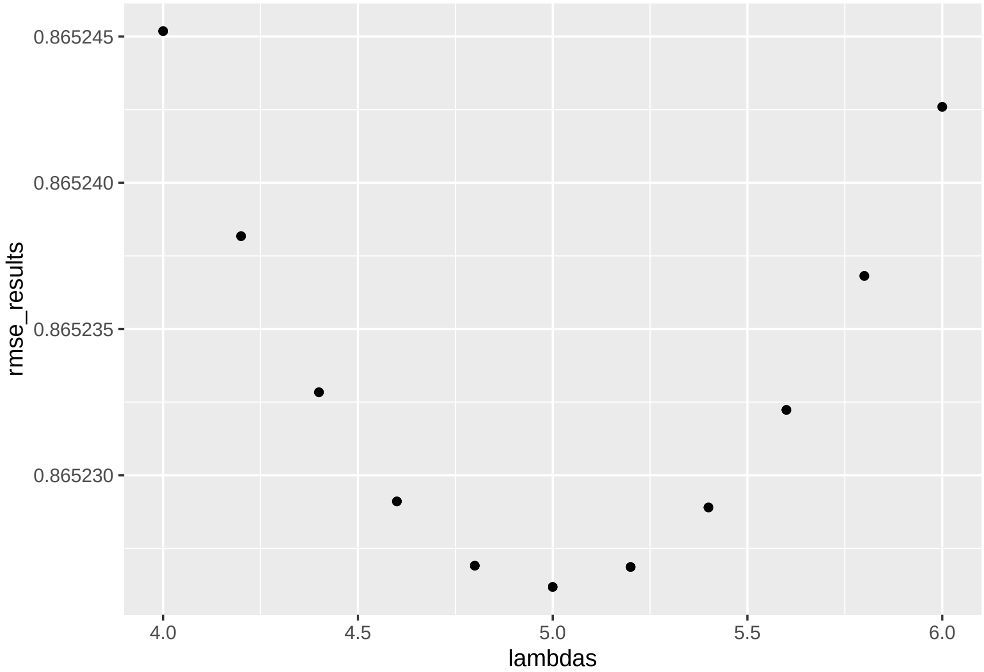


Table 10: Model accuracy

Method	RMSE
Average Rating Model	1.0599
Movie Effect Model	0.9437
Movie and User Effect Model	0.8659
Movie, User and Time Eff. Model	0.8659
Regularized Model	0.8652

3.6. Final Touch

The predicted ratings cannot be less than 0.5 or greater than 5. Let's see how many of the predicted ratings are outside of the valid range.

Table 11: Prediction

< 0.5	> 5
141	2836

We can round the outside ratings to the nearest valid values.

Table 12: Model accuracy

Method	RMSE
Average Rating Model	1.0599
Movie Effect Model	0.9437
Movie and User Effect Model	0.8659
Movie, User and Time Eff. Model	0.8659
Regularized Model	0.8652
Regularized Model (Capped)	0.8651

3.6. Final result on validation set

Finally, we calculate our prediction on the "validation" set. Appropriate method to achieve better accuracy if we use the entire "edx" dataset as the training set, and use the optimal λ parameter value we determined earlier. (Ref. link: Harvardx Discussion)

Table 13: Model accuracy

Method	RMSE
Average Rating Model	1.0599
Movie Effect Model	0.9437
Movie and User Effect Model	0.8659
Movie, User and Time Eff. Model	0.8659
Regularized Model	0.8652
Regularized Model (Capped)	0.8651
Final RMSE	0.8647

4. Summary

We have successfully described a way to build up the recommendation algorithm to predict movie ratings using MovieLens 10M Dataset in this report. We started from the simplest possible algorithm, and progressively improved it by taking more and more effects into account. In our final model we took the overall average rating, the movie, user and the time effects into account, and used the regularization technique and some final improvement to obtain an acceptable RMSE.

The final goal was to achieve an RMSE less than 0.86490. The final RMSE from our algorithm is 0.8647, so we achieved our project goal and the initial criteria of the HarvardX Data Science: Capstone project.

5. Future Cosiderations

However we achieved the final goal of this project, our average error is still higher than 0.85 stars, so we have left room for further improvements. If in the future we will need even better accuracy, we can improve our final model for example with the following techniques.

5.1 Genre Effects

Similar to the movie and user effects we could consider genre effect. We could analyze the average rating of the genres. Some genres may be rated higher on average than the others. We could also consider user+genre effects because of the fact that certain users prefer certain genres. Addressing these effect would improve the model performance.

5.2 Rating Time Effect

Rating time may have an effect on the ratings of the users. Maybe we become stricter or more lighthearted as we grow older. If we find correlation between the rating time and the mean rating of a user, we could include it in our model.
We could also consider the effect of the time difference between rating and the release of the movie on the ratings of users. As the saying goes: Time makes the memories nicer. Maybe we give higher ratings for movies that we saw a while ago.

5.3 Matrix Factorization

Matrix factorization is a highly effective, and widely used technique in building of recommendation systems. We could consider to build a model based on this technique. We could convert our dataset into a matrix. Each row of the matrix could be assigned to a user, and each column assigned to a movie. By using matrix factorization we could decompose our matrix to two smaller rectangular matrices, which could be handled much easier than the original large matrix.

5.4 Rounding

We saw in Chapter 2.4, that the ratings before 2003 are scales from 1 to 5 star with whole star increments. The half star increments were introduced in 2003. We could further improve the accuracy of the model by rounding our predictions to the nearest half-integer when predicting ratings from 2003 and later, and to nearest integer when predicting ratings before 2003.