Movie recommendation system

Rachid elkhayat

3/20/2020

The Outline

1. Introduction
The project goal
The key steps implemented

2. Method and Analysis
Data preparation
Data exploration and visualization
Modeling and Testing
- Average Value Model
- Movie- effect Model
- User and movie effects Model

3. The results

4. Conclusion

---

1. Introduction

Streaming service companies i.e. Netflix, Amazon prime, Hulu, ...etc. are growing at a very fast pace since the last decade. what is the secret of this success? why people often opt to create their own profile and adding their preferences? How machine learning has contributed in the progress of this industry?

In this project, a movie recommendation system will be created based on the algorithms used by the winner's team of the open competition initiated by Netflix in 2006 to predict the user ratings of films based on previous ratings.

For that pusrpose we will be working on a subset of 10 million observations of Movielens dataset, the original dataset includes 27M obs, 58000 movies, 280k users that were collected by the GroupLens Research. We will discover the features, insights and trends of the data set in the data exploration section.

The project goal

The goal of this project is to predict the user ratings for movies using different machine-learning models. The scale of the rating starts from 0 to 5 where 5 indicates highest rating. The main challange of this project is to identify the different biases in the ratings, quantify it and test the results to check its effect on the predicted ratings. To do so we need to train different ML models and identify the success for the model by a loss function (RMSE), which will represent the standard deviations between the predicted outcomes and the true ratings. The target is to achieve RMSE value < 0.86490 by calculating the different of the means using the below equation:

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

The key steps implemented to achieve the desired result:

- Downloading & splitting the dataset into training and validation set.
- Exploring the data by looking at different features, trends, numbers.
- Applying different machine learning models to achieve the desired goal:
  - Model- based : The rating is the same across all movies and users
  - Movie- effect : Adding the movie effect "b_i"
  - User and movie effect : Adding the movie effect "b_i" and the user effect "b_u"
- Regularization -Imposing penalty on the high prediction values that was obtained from small sample size.

2. Method and Analysis

Data preparation

To start off with this project we need to download the dataset and split it into two subsets.

A subset of 90% of the movielens dataset, that will be used to train our models(Training set) and another subset of 10% of the movielens dataset, will be used to test the model and obtain the RMSE value(validation set)

Note that the validation set will only be used for testing

```
# Note: this process could take a couple of minutes

if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
library(dplyr)
library(ggplot2)
library(tidyr)
library(knitr)
library(rmarkdown)

# MovieLens 10M dataset:
# https://grouplens.org/datasets/movielens/10m/
# http://files.grouplens.org/datasets/movielens/ml-10m.zip

dl <- tempfile()
download.file("http://files.grouplens.org/datasets/movielens/ml-10m.zip", dl)

ratings <- fread(text = gsub("::", "\t", readLines(unzip(dl, "ml-10M100K/ratings.dat"))),
                 col.names = c("userId", "movieId", "rating", "timestamp"))

movies <- str_split_fixed(readLines(unzip(dl, "ml-10M100K/movies.dat")), "\\::", 3)
colnames(movies) <- c("movieId", "title", "genres")
movies <- as.data.frame(movies) %>% mutate(movieId = as.numeric(levels(movieId))[movieId],
                                           title = as.character(title),
                                           genres = as.character(genres))

movielens <- left_join(ratings, movies, by = "movieId")

# Validation set will be 10% of MovieLens data
set.seed(1, sample.kind="Rounding")
# if using R 3.5 or earlier, use `set.seed(1)` instead
test_index <- createDataPartition(y = movielens$rating, times = 1, p = 0.1, list = FALSE)
```

```
edx <- movielens[-test_index,]
temp <- movielens[test_index,]

# Making sure userId and movieId in validation set are also in edx set
validation <- temp %>%
  semi_join(edx, by = "movieId") %>%
  semi_join(edx, by = "userId")

# Adding rows removed from validation set back into edx set
removed <- anti_join(temp, validation)
edx <- rbind(edx, removed)

rm(dl, ratings, movies, test_index, temp, movielens, removed)
```

Data exploration and visualization

To get a feel of the edx data we will have a look at its structure, number of features and dimension:

```
head <- head(edx)
knitr::kable(head)
```

	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

We notice that each rating for a movie is represented in a row. The genre for each movie are clubbed together and separated by "|", the title and the release year of the movie are clubbed together as well. Note that we will not be working with the genre and time series in this project

Checking the number of rows and columns

```
dim(edx)
```

```
## [1] 9000055       6
```

The summary of the dataset that provide an insight about the stastics and spread of the data that we are working on. it is always a good practise to use the summary before going into visualization as it provides a great descriptive analysis that help us understand our dataset centeral tendacy and the dispersion.

```
summary(edx)
```

```
##      userId         movieId          rating        timestamp
##  Min.   :    1   Min.   :    1   Min.   :0.500   Min.   :7.897e+08
##  1st Qu.:18124   1st Qu.:  648   1st Qu.:3.000   1st Qu.:9.468e+08
##  Median :35738   Median : 1834   Median :4.000   Median :1.035e+09
##  Mean   :35870   Mean   : 4122   Mean   :3.512   Mean   :1.033e+09
##  3rd Qu.:53607   3rd Qu.: 3626   3rd Qu.:4.000   3rd Qu.:1.127e+09
##  Max.   :71567   Max.   :65133   Max.   :5.000   Max.   :1.231e+09
##     title              genres
##  Length:9000055     Length:9000055
##  Class :character   Class :character
```

```
##  Mode  :character   Mode  :character
##
##
##
```

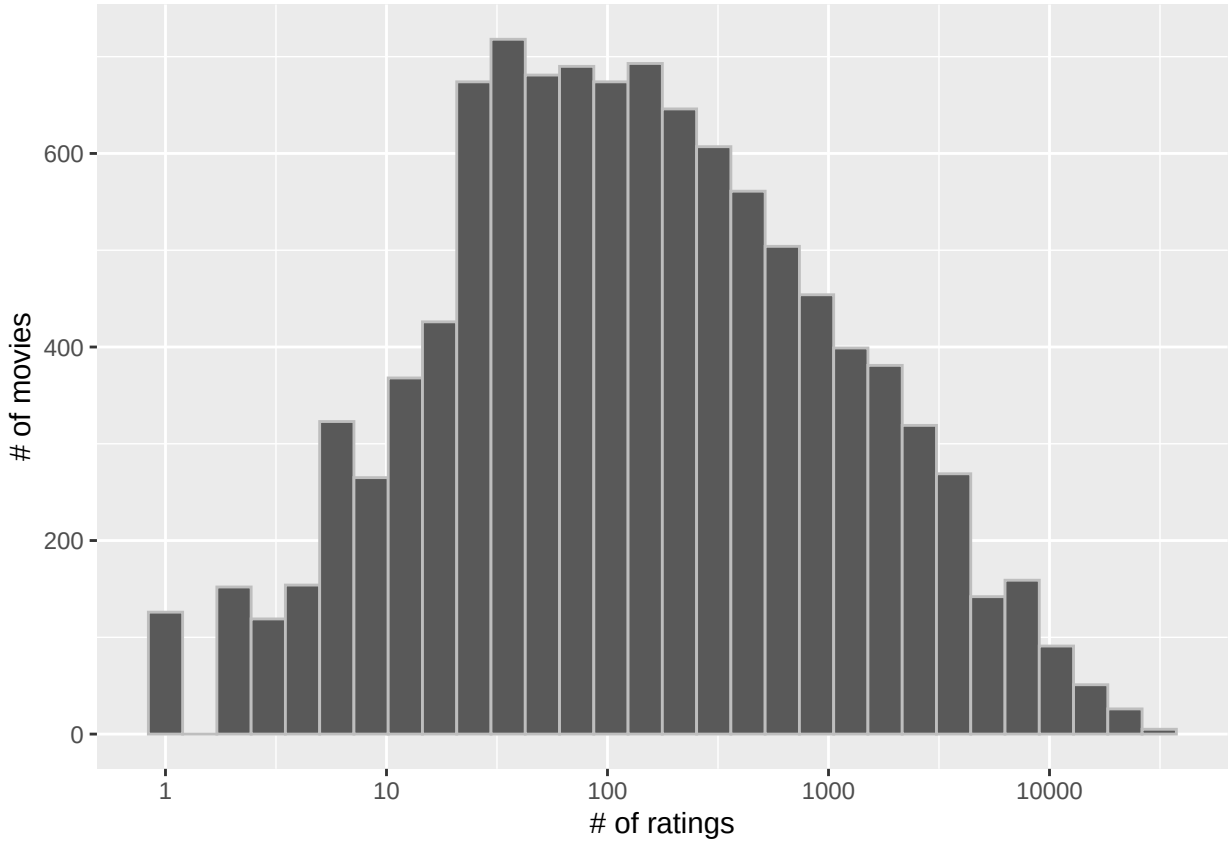
We can conclude the below key points looking at the summary table:

- The rating mean is 3.512 which implies that the overall average rating tend to be more on the positive.
- The rating input starts from '0.5' no '0' was found.
- The data doesnt have NAs & rating since the min starts at '0.5'

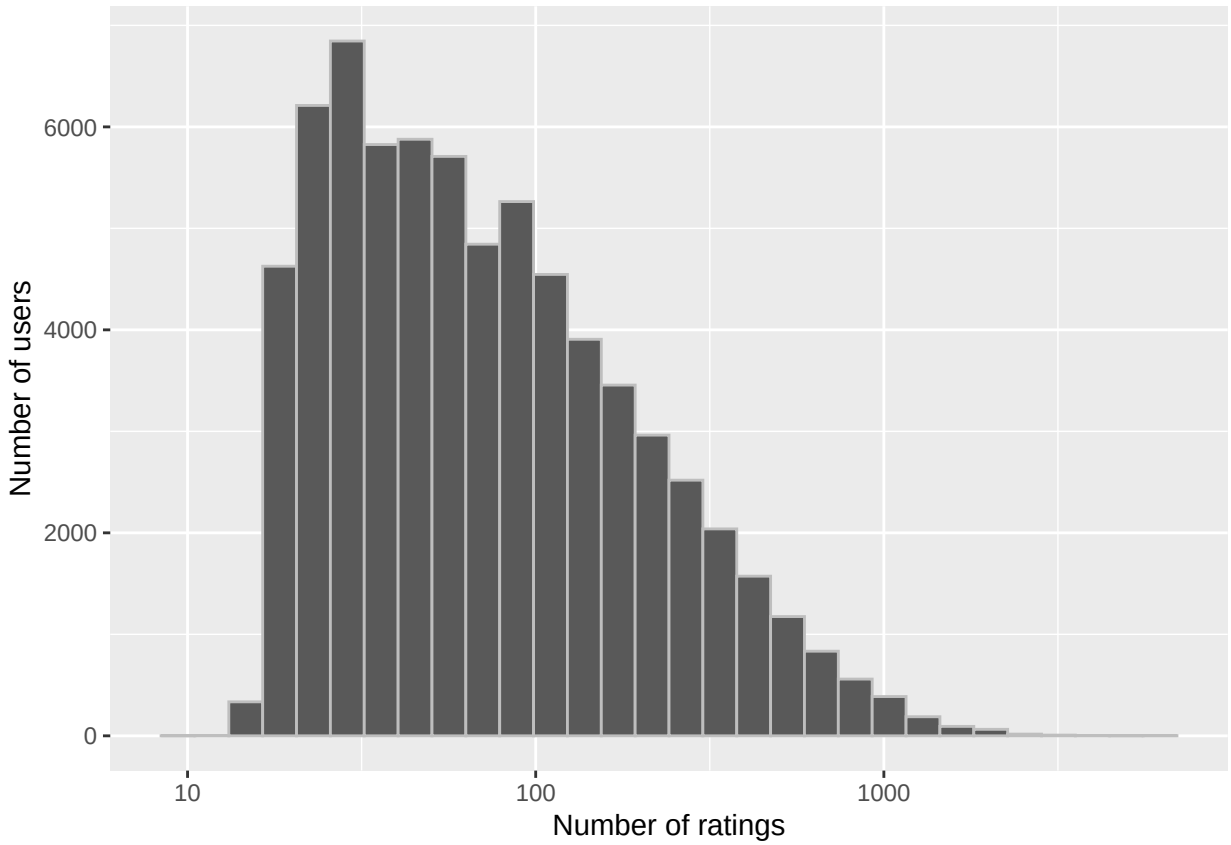
The distinct number of movies equals to 10677

The distinct number of users equals to 69878

Plotting ratings# per movie in a histogram to observe how some movies are rated differently more than others.



Plotting ratings per users to shows the difference of rating of different movies by the users



Modeling and Testing

Based on the finding from our data exploration we will approach the data by fitting the models taking into account the variables that is effecting the predictions. we will start by the basic approach and then move on by adding the other baised features related to the movies and users.

The three models to be applied:

- Average Value
- Movie-effect
- User and Movie effects

Starting by defining our loss function which will gives us an indicition about how far is our predictions from the true data

This function will compute the average total variation between the predicted values and the true values. if RMSE is bigger than the goal value, then we will need to use more complicated module.

```
RMSE <- function(y_hat, y){
  sqrt(mean((y_hat - y)^2))
}
```

Average Value Model:

Considering rating is the same across all movies and users regardless of user and movie. The estimate that minimizes the residual mean squared error is the least squares estimate of mu and epsilon represents the error generates from a random variation, we use the below equation:

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

The overall average rating (on the training set (edx)) equals to mu = 3.5124652,

Testing results and save RMSE in results table for comparison

Method	RMSE
Average Value Model	1.061202

This result is considered very big as the error value is more than 1, so our predictions might vary with more than one star.

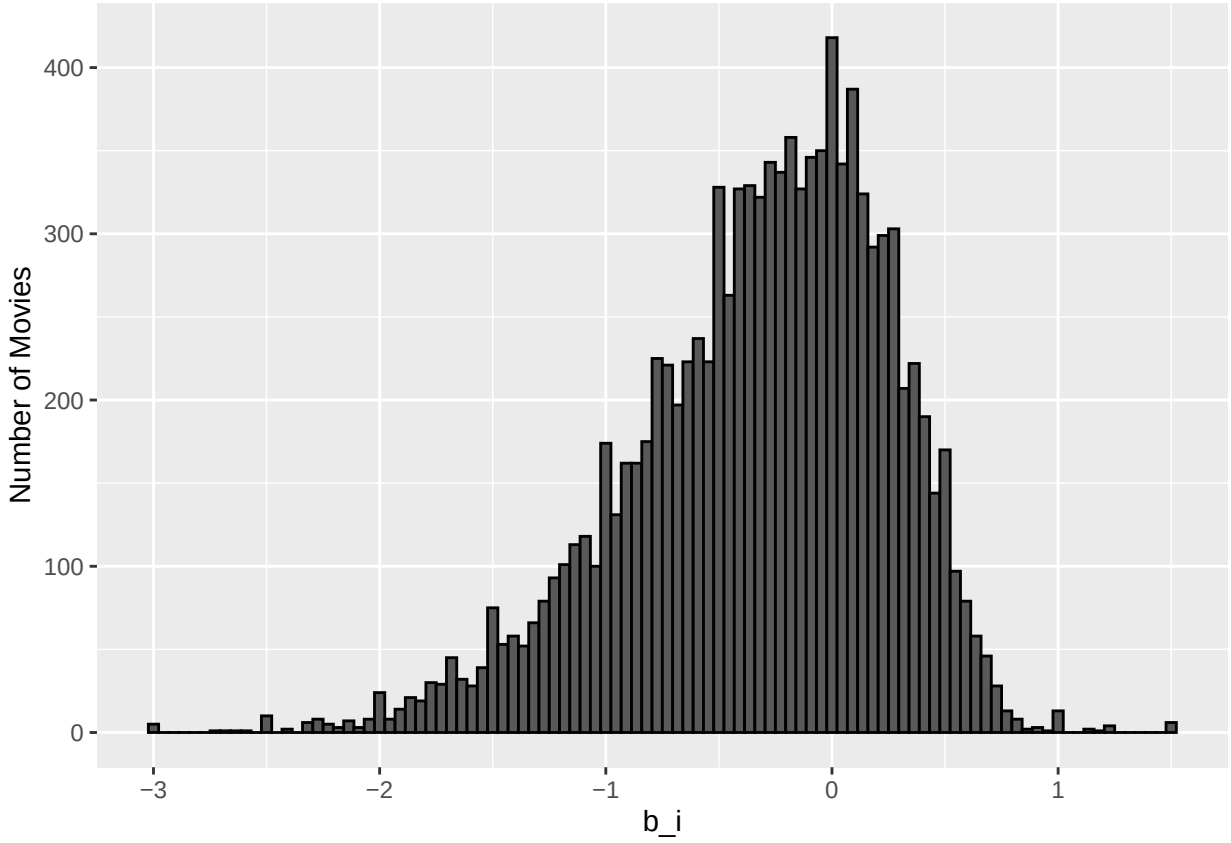
Movie effect Model:

In our analysis we saw that different movies are rated differently, some more than others, Adding the movie effect b_i to the revious model should lead to better predictions and therefore lower RMSE.where b_i is represents the average rating of movie i.

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

Due to the fact that we will not be able to compute the least square estimates for b-i as the data is very big , we will calculate b-i for each movie i using the following equation: b_i = mean(rating - mu)

Visualizing b_i by plotting a histogram, as we can see from the graph it is skewed to the left which implies that it has a negative effect on the overall rating. Note that since mu =3.5124652 then b-i of 1.4875348 would result in a perfect rating.



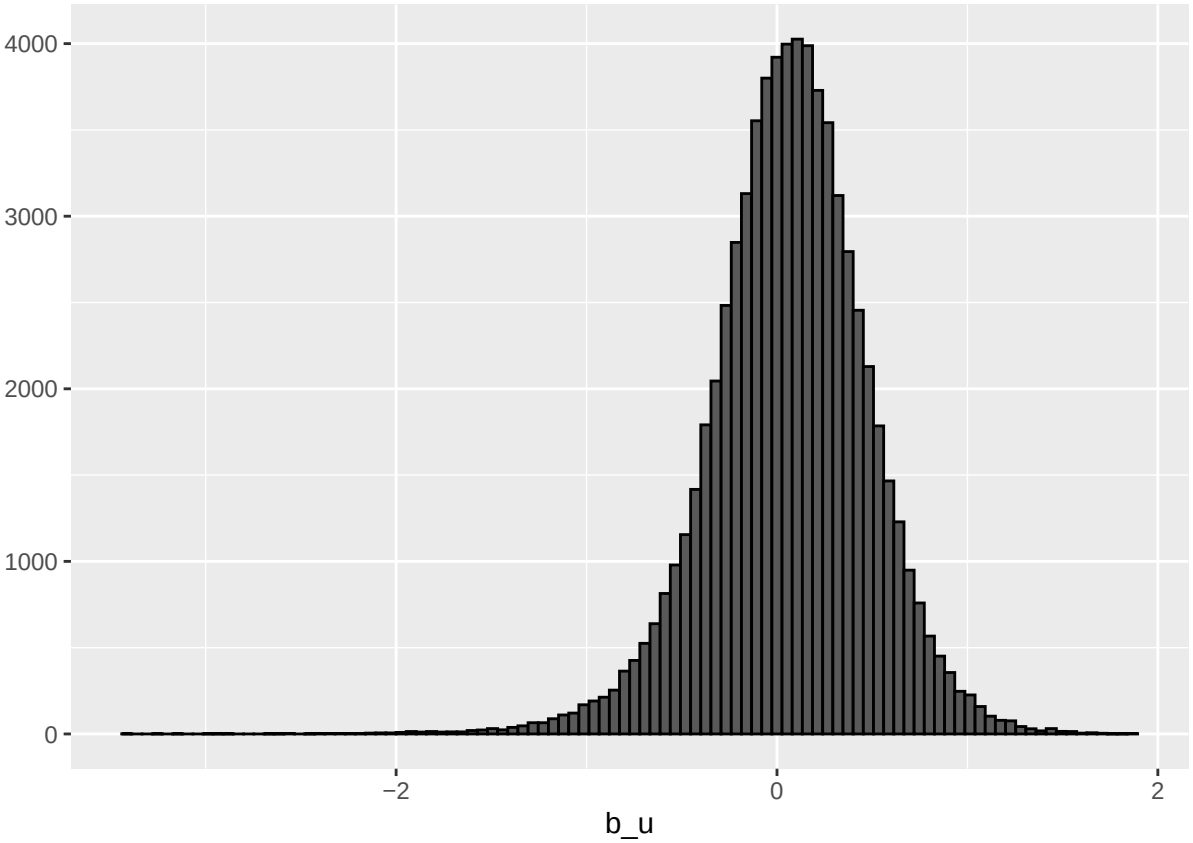
After predicting and testing the we add the RMSE value to the results table

Method	RMSE
Average Value Model	1.0612018
Movie-effect Model	0.9439087

```
Movie- effect Model
```

User and movie effects Model

Since different users rate movies differently some users tend to rate high while some others tend to give low rating as we have seen. that can be clearly seen in te below histogram:



We will add the user effect to the previous model by using the below equation.

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

After Predicting results and Testing the model on the validation set we add the new results to the table:

Method	RMSE
Average Value Model	1.0612018
Movie-effect Model	0.9439087
User and Movie Model	0.8653488

To see why there is no major decrease in the RMSE we will have a look at the list of the 10 top and worst

movies as precicted by the "Movie effect Model".

Top 10 movies

title	b_i	n
Hellhounds on My Trail (1999)	1.487535	1
Satan's Tango (Sátántangó) (1994)	1.487535	2
Shadows of Forgotten Ancestors (1964)	1.487535	1
Fighting Elegy (Kenka erejii) (1966)	1.487535	1
Sun Alley (Sonnenallee) (1999)	1.487535	1
Blue Light, The (Das Blaue Licht) (1932)	1.487535	1
Who's Singin' Over There? (a.k.a. Who Sings Over There) (Ko to tamo peva) (1980)	1.237535	4
Human Condition II, The (Ningen no joken II) (1959)	1.237535	4
Human Condition III, The (Ningen no joken III) (1961)	1.237535	4
Constantine's Sword (2007)	1.237535	2

We can see that all these movies have been rated very few times, we notice that most of the movies are obscure movies.

Worst 10 movies

title	b_i	n
Besotted (2001)	-3.012465	2
Hi-Line, The (1999)	-3.012465	1
Accused (Anklaget) (2005)	-3.012465	1
Confessions of a Superhero (2007)	-3.012465	1
War of the Worlds 2: The Next Wave (2008)	-3.012465	2
SuperBabies: Baby Geniuses 2 (2004)	-2.717822	56
Hip Hop Witch, Da (2000)	-2.691037	14
Disaster Movie (2008)	-2.653090	32
From Justin to Kelly (2003)	-2.610455	199
Criminals (1996)	-2.512465	2

Also these movie have been rated only few times, we also notice that most of the movies are obscure movies

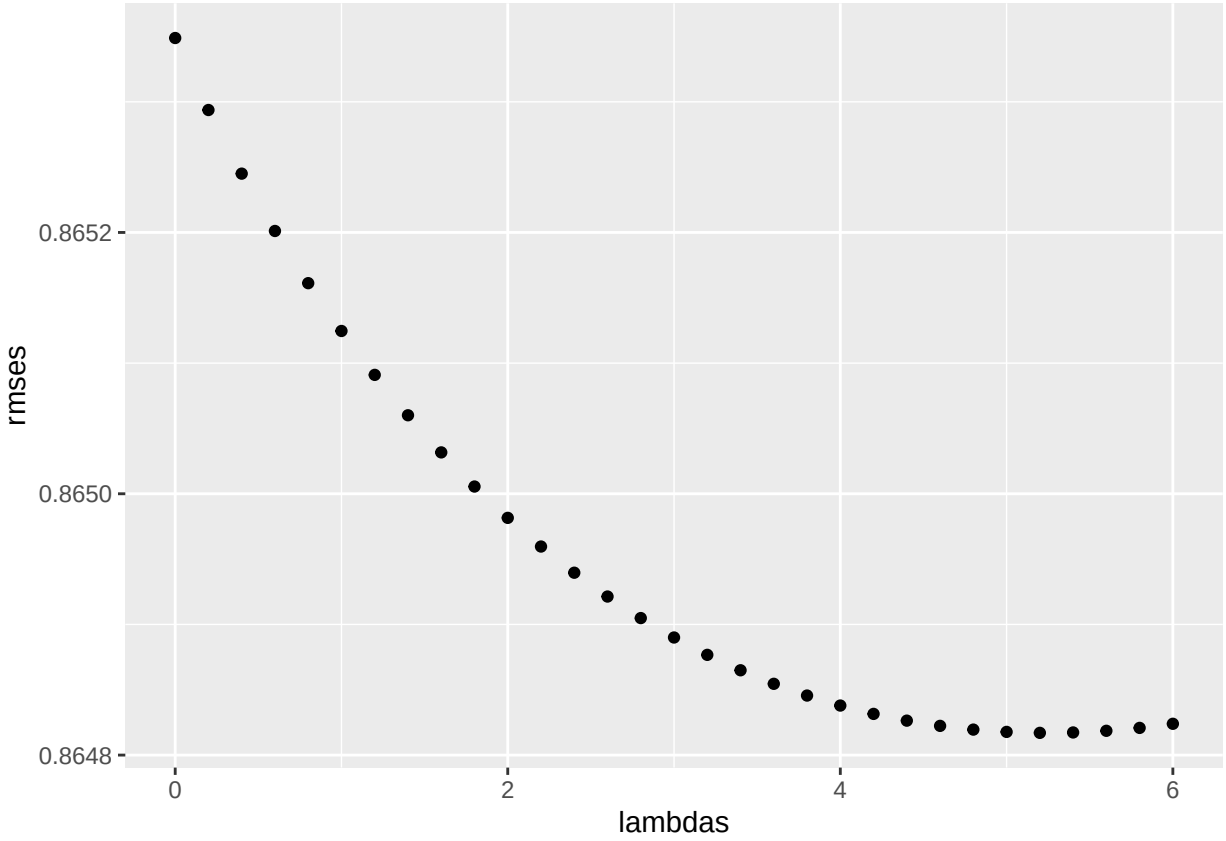
Our rating predictions further reduced the RMSE, However, the smaller the sample size the more unreliable is the prediction.

We need to take an extra measure to avoid the noisy data coming from small sample size. For that reason we will use the Regularization to reduce the effect of overfitting and reduce the RMSE.

Lambda will be used as a tuning parameter to penalize the prediction values that are very high that were the outcome of movies with very few ratings and in users that only rated a very small number of movies.

To get the value of lambda we will use cross-validation The below function will find the RMSE value by looping through different value of lambdas, the lambda that results in the lowest rmse will be chosen.

Plotting the rmses vs lambda to identify the best value of lambda



The value of lambda that resulted in least rmse value equals to 5.2

3. The results

Method	RMSE
Average Value Model	1.0612018
Movie-effect Model	0.9439087
User and Movie Model	0.8653488
Results after regularization	0.8648170

We can clearly see how applying the penalty helped in decreasing the RMSE, this prossess helped to elemenate the outliers and therefore giving a fair chance for the movies to be ranked by the user ratings.

4. Conclusion

We have seen that Machine learning can play a vital role in recommendation systems by predicting the estimate rating for the users, this proccess helps to personalise the expereience of the user by recommending the right movie to the right user.

The approach for machine learning models can differ based on the Dataset and the feature provided, in our case we looked into two main factores movies and users effects. using the loss function that help us to spot the error betwen the predicted and the true values we where able to achieve the RMSE that satisfy the goal of the project.

This result can be enhanced by using different ML approaches, which were not used in this project. Advanced

ML learning models require a greater computation power due to the size of the dataset, one of the major models used for recommendation system is matrix factorization that is based on collaborative filtering algorithms.