Capstone Project: MovieLens

Jayashri Gopalakrishnan

5/28/2020

Introduction

Recommendation systems have gained popularity over the past decade due to the popularity of firms such as Amazon and Netflix. These firms have experienced unparalleled growth in the over the years primarily due to their excellent recommendation systems that have been able to draw and retain customers for years. Consumers are appreciative of a customized experience physically (i.e. in-person) and online. A lot of businesses are taking on a similar approach to personalize their offerings to customers so that customers will return to them after their initial purchase or use of their service or product.

The objective of this project is to create a movie recommendation system using a machine learning algorithm to predict movie ratings. The MovieLens dataset being used for analysis contains approximately 10 million ratings, and is further divided into a 9 million training set and 1 million validation set. In the training dataset, there are approximately 70,000 users and 11,000 movies from many genres such as Action, Adventure, Drama, etc. After thorough analysis, several machine learning algorithms have been developed and compared to get maximum accuracy when predicting movie ratings. The chosen algorithm is based on providing a Root Mean Squared Error of less than 0.86490.

The dataset contains 6 variables: *userId*, *movieId*, *rating*, *timestamp*, *title*, *genres*.

For accomplishing the goal, the Regualrized Movie, User, Year, and Genre Model achieves an RMSE of **0.862**, which is less than the target of 0.8649

Data Loading

The following code was provided to load the data and split it into training and test sets

```
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")

# 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,]

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

# Add 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)
```

Unnecessary files have been removed at the end. Before proceeding to analyze data, a few other libraries are installed for the purpose of visualization.

Methods and Analysis

Summary Statistics

The MovieLens dataset is divided into two: *edx* for training and *validation* for testing. Upon initial analysis, the edx has 6 variables. Below is the first 6 rows of the edx dataset:

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

Here is the summary statistics of the edx dataset:

```
##      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
##
##
##
```

Data Cleaning and Exploration

Before analyzing the data further, it is necessary to clean the dataset. The previous section indicates that the year of release of the movie is merged with the title. The year will be extracted as a separate column for our analysis. Additionally, if a movie has multiple genres, the listed genres are sepearated by the "|" character. For our analysis, each genre is to be parsed and separated into a new row.

The following table indiactes the number of unique movies and users in the database. There are approximately 70000 unique users and 11000 unique movies.

```
##   no_of_users no_of_movies
## 1       69878        10677
```

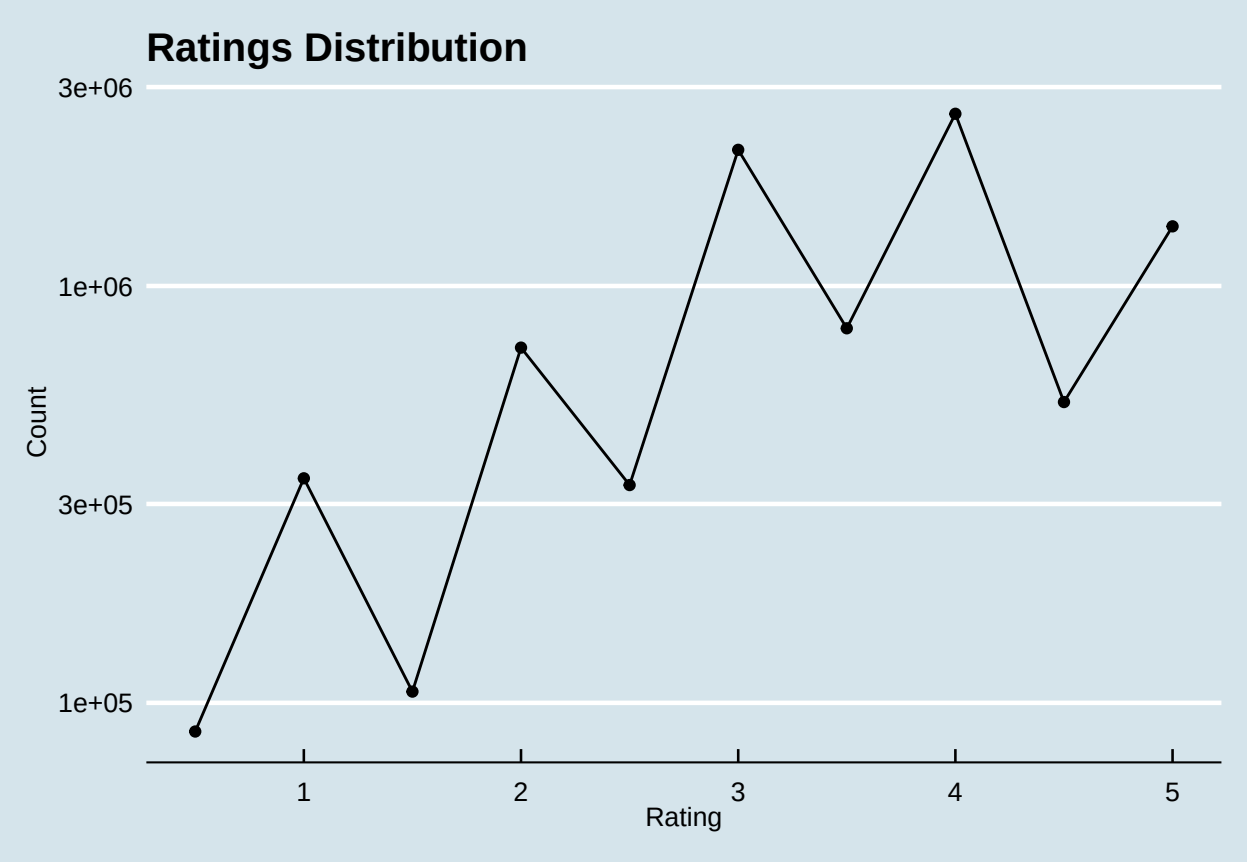
Now that the genres have been parsed, here are the most popular genres based on the number of ratings:

```
## # A tibble: 20 x 2
##    genres               count
##    <chr>                <int>
##  1 Drama              3910127
##  2 Comedy             3540930
##  3 Action             2560545
##  4 Thriller           2325899
##  5 Adventure          1908892
##  6 Romance            1712100
##  7 Sci-Fi             1341183
##  8 Crime              1327715
##  9 Fantasy             925637
## 10 Children            737994
## 11 Horror              691485
## 12 Mystery             568332
## 13 War                 511147
## 14 Animation           467168
## 15 Musical             433080
## 16 Western             189394
## 17 Film-Noir           118541
## 18 Documentary          93066
## 19 IMAX                  8181
## 20 (no genres listed)       7
```

Data Visualization

Ratings

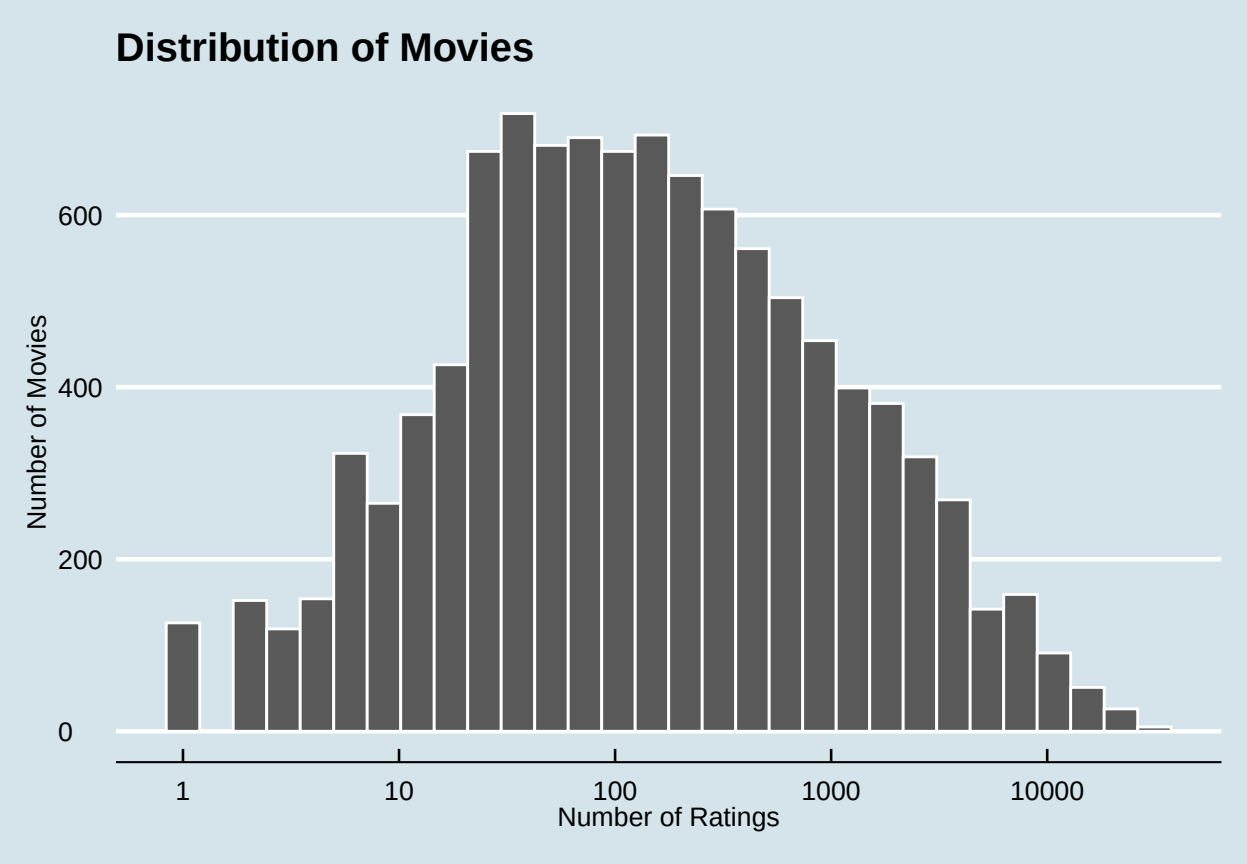
Each user rates a movie between 0.5 to 5 in 0.5 point increments. The graph below indicates that most movies are rated between 3 and 5. Half star ratings are less common than whole star ratings.



Movies

Out of the 10677 movies available in the dataset, a majority of the movies are rated more than once. Some movies are more popular and more likely to be watched than others.

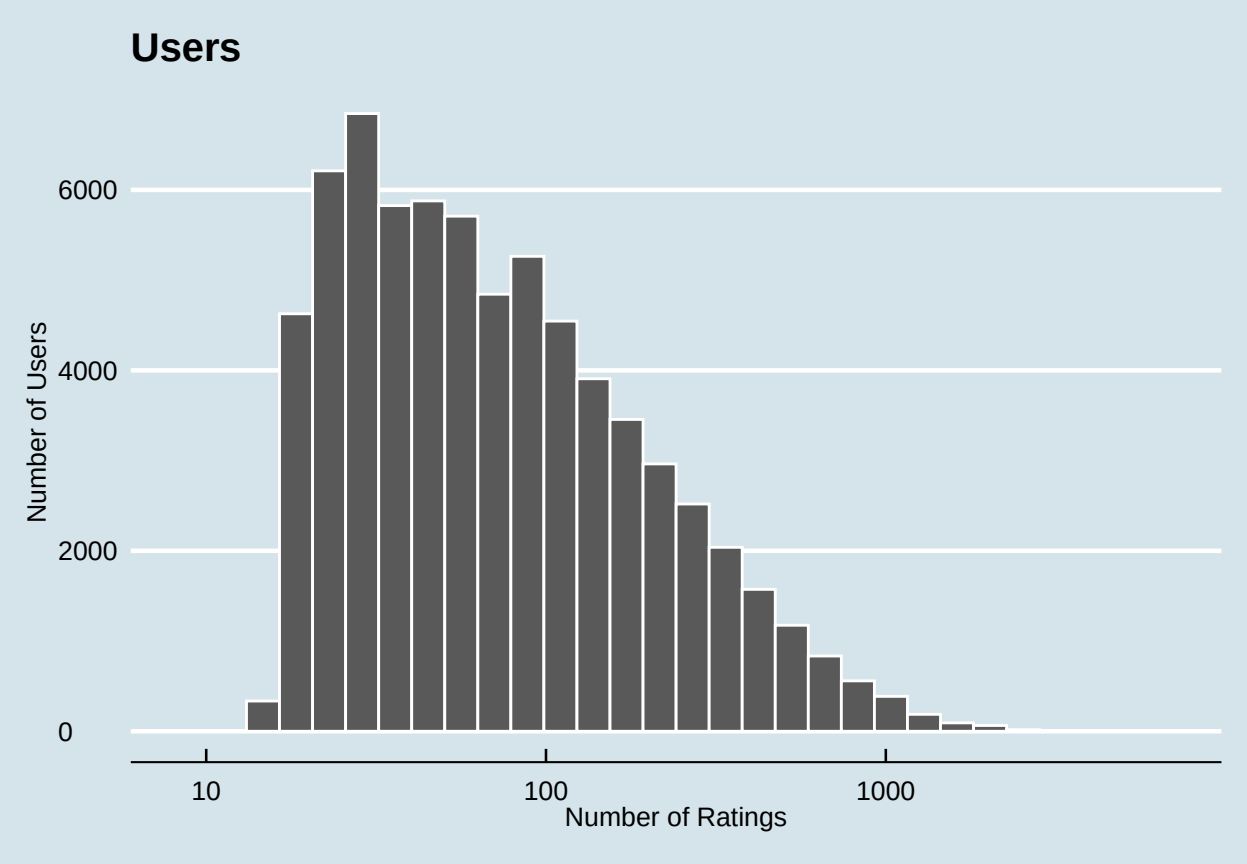
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Users

There 69878 unique users in the dataset. The user distribution is right skewed. User activity is varied. Most users rate a few movies, while some users rate more than a 1000 movies.

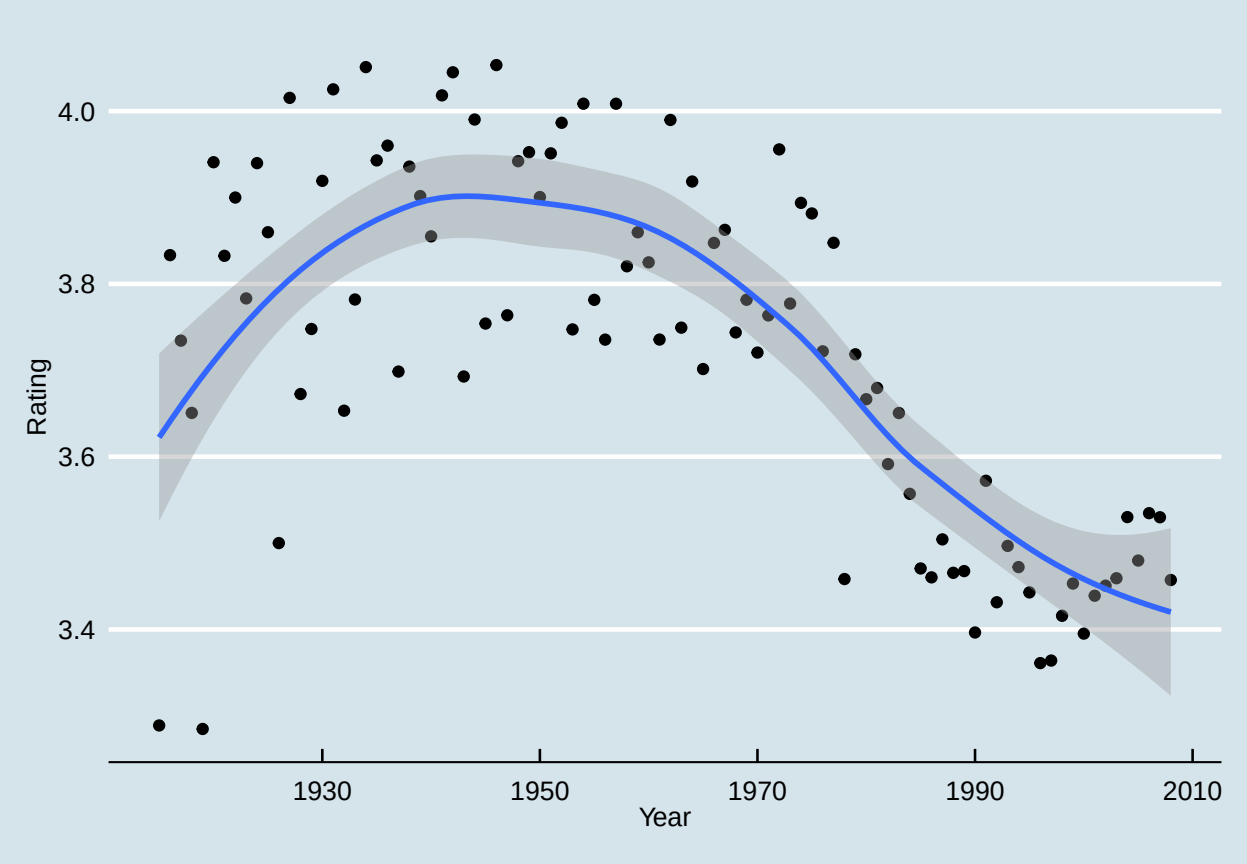
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Release year

Looking at the graph below, it is found that movies more recently released have had a lower rating than earlier movies (movies from the 1930s - 1950s).

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



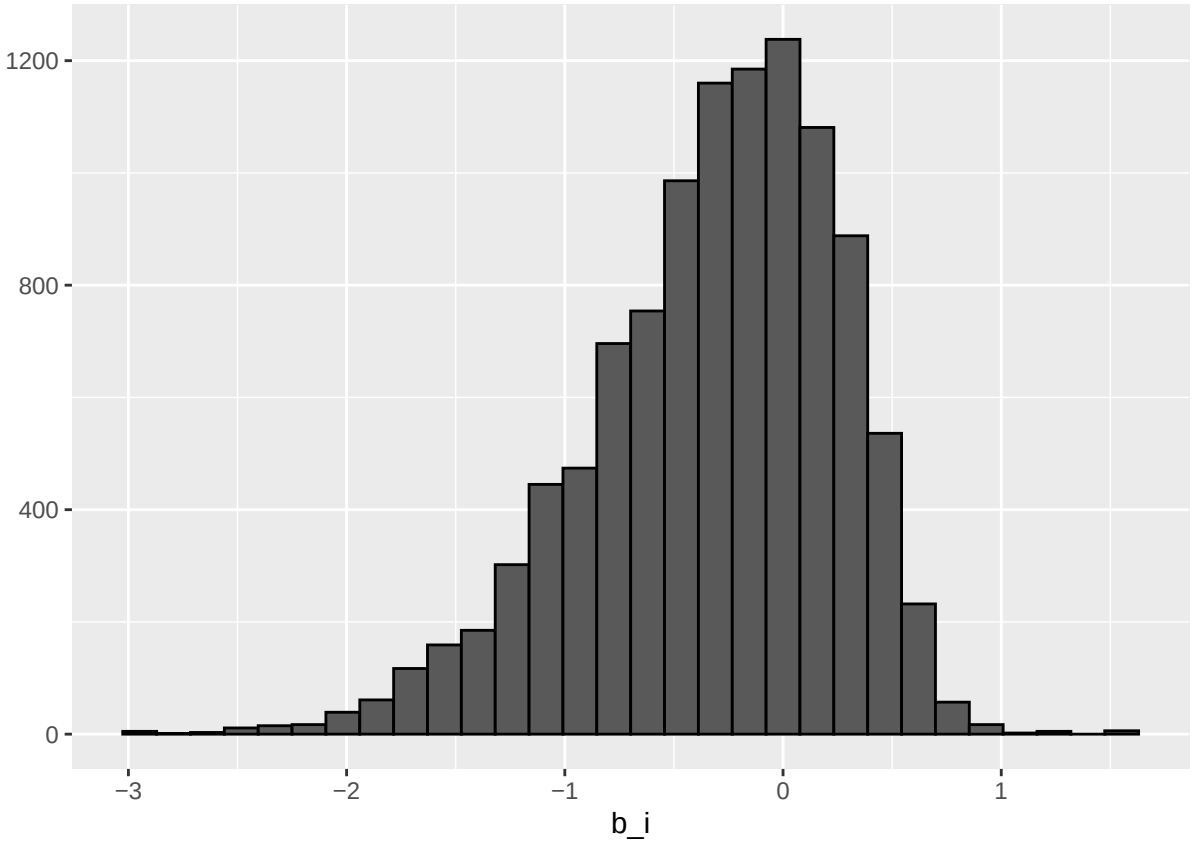
Modeling Approach

The ultimate objective of this machine learning project is to create a model which minimizes the root mean square error (RMSE). The RMSE is also called as the loss function, which calculates the square of difference between actual value and predicted value.

Penalty term - Movie effect

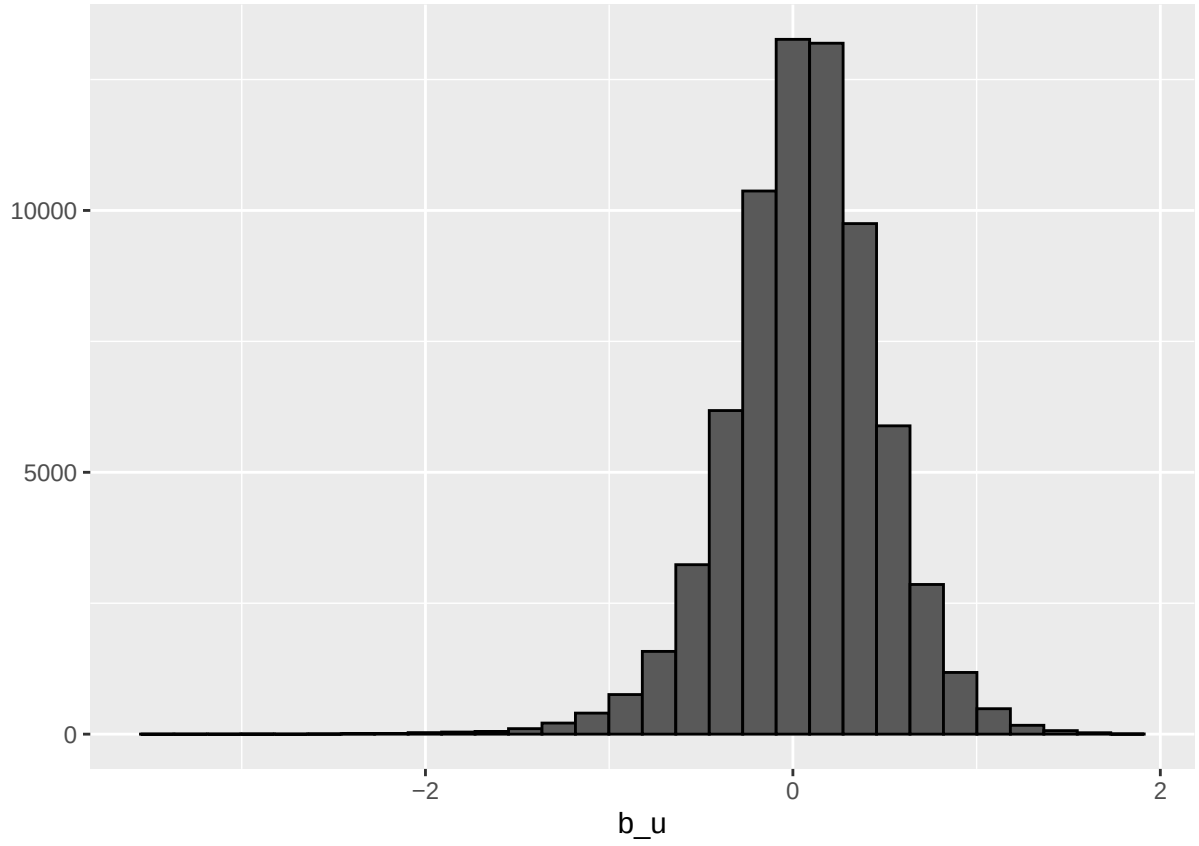
Not all movies are the same. Some are much more popular and successful than others. The movie effect is accounted for by calculating the difference from the mean rating (from the edx dataset). The below histogram is left skewed (negatively skewed) which implies that more movies have negative effects.

```
## Warning: `data_frame()` is deprecated, use `tibble()`.
## This warning is displayed once per session.
```



Penalty term - User effect

Since each user has different tastes and prefernces, they rate every movie differently. The below histogram displays the distribution for the user effect.



Model Creation

Five models will be evaluated based on their RMSEs. The model with the lowest RMSE will be chosen.

Naive Model

The Naive model creates a prediction system based solely on the sample mean, implying that every predicted rating will be the sample mean. The RMSE of this model is more than 1.

```
## [1] 1.061202
```

method	RMSE
Using mean only	1.061202

Movie Effect Model

The RMSE is reduced by adding the movie effect as indicated below:

method	RMSE
Using mean only	1.0612018
Movie Effect Model	0.9439087

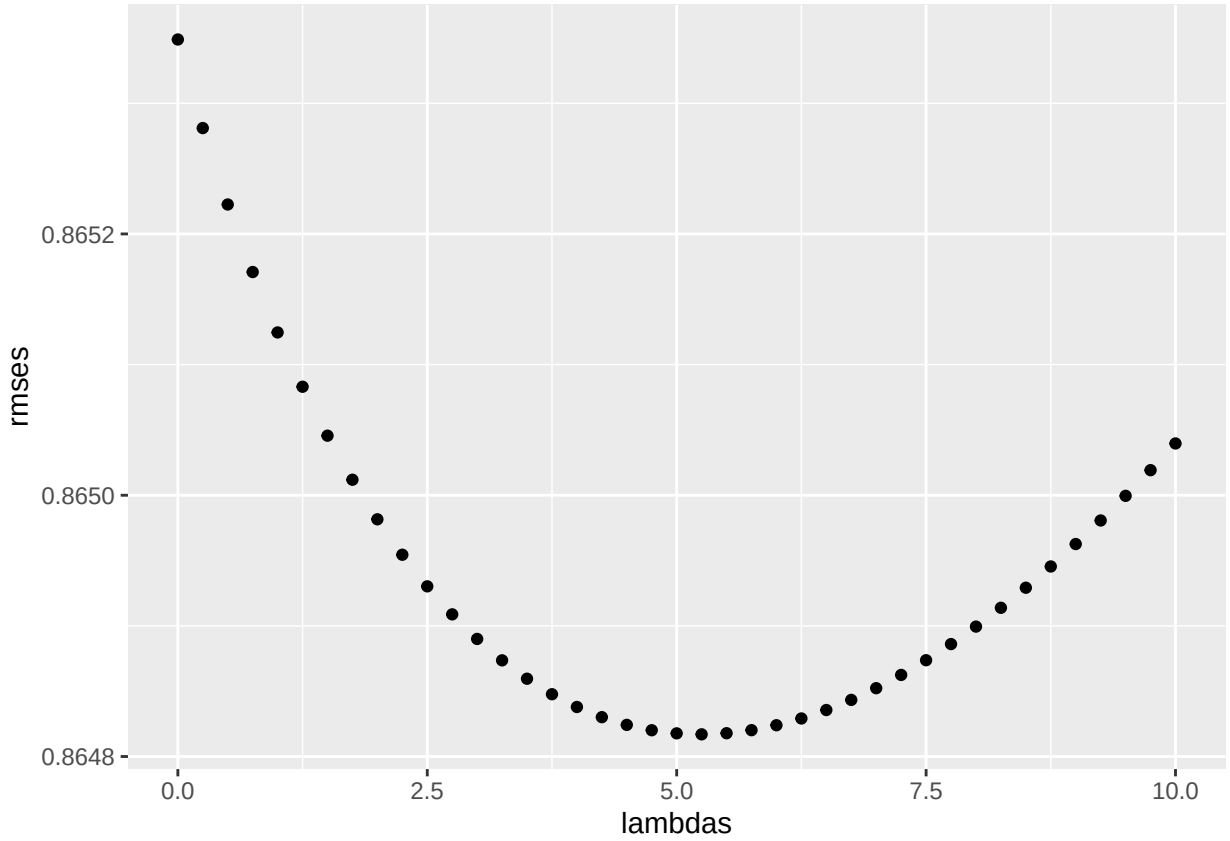
Movie and User Effect Model

The RMSE is further minimized by adding the user effect as shown below:

method	RMSE
Using mean only	1.0612018
Movie Effect Model	0.9439087
Movie and User Effect Model	0.8653488

Regularized Movie and User Effect

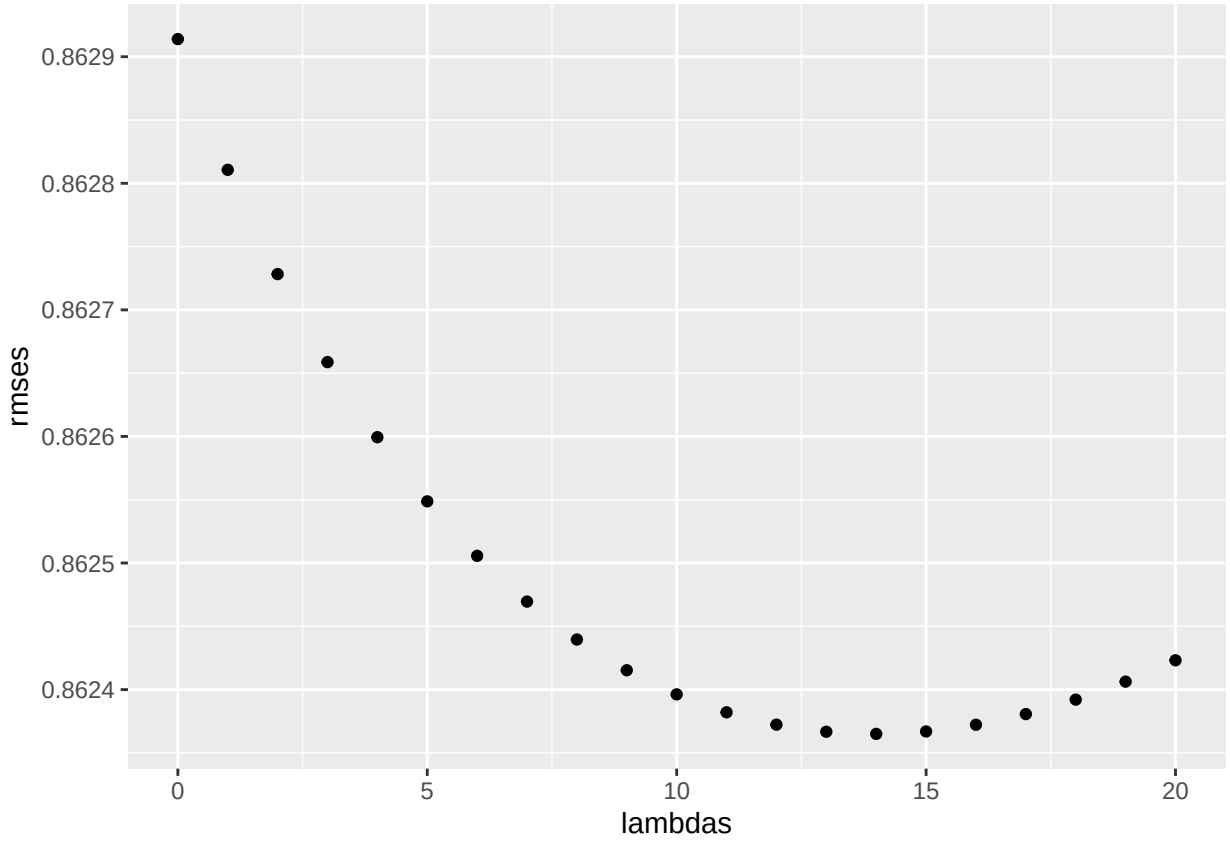
The data visualizations indicated that some users rate very few movies and some movies are rated much less than others. These inconsistencies create large errors, which can adversely impact RMSE calculations. Regularization is implemented to reduce the risk of overfitting by assigning a penalty term (lambda). We choose a lambda which minimizes RMSE. As indicated below, the RMSE is further minimized.



method	RMSE
Using mean only	1.0612018
Movie Effect Model	0.9439087
Movie and User Effect Model	0.8653488
Regularized Movie and User Effect Model	0.8648170

Regularized Movie, User, Year and Genre Model

The previous regularized model is created with the addition of the genre effect and year (release year) effect. The RMSE has been reduced substantially in comparison to the Naive Model.



method	RMSE
Using mean only	1.0612018
Movie Effect Model	0.9439087
Movie and User Effect Model	0.8653488
Regularized Movie and User Effect Model	0.8648170
Regularized Movie, User, Year, and Genre Effect Model	0.8623650

Results

RMSE values

The Naive model has the highest RMSE of approximately 1.061. The Regularized Movie, User, Year and Genre Effect Model has the lowest RMSE of approximately 0.862. Based on the table below, the regularization model showed substantial improvements in RMSE in comparison to the Naive Model as well as the Movie Effect model. Our exploratory analysis indicated that some data points (outliers) could have potentially impacted the loss function results. The regularization approach helped penalize these data point to avoid overfitting.

method	RMSE
Using mean only	1.0612018
Movie Effect Model	0.9439087
Movie and User Effect Model	0.8653488
Regularized Movie and User Effect Model	0.8648170
Regularized Movie, User, Year, and Genre Effect Model	0.8623650

Conclusion

This project involved a comprehsive overview of the various topics learnt throughout the HarvardX's: Professional Data Science Certification Program. The edx dataset was cleaned and explored, while also creating visualizations that enabled me to further analyze key findings throught the project. Five models were built for the purpose of creating a recommendation system with a minimized RMSE. We can ultimately be satisfied with the predicted ratings of the Regularized Movie, User, Year and Genre Effect Model which has an RMSE of 0.862.

Limitations and Further Work

One major limitation of this project is the duration taken to run the code. I believe that the efficiency of the model could be improved by running the code in parallel, rather than sequentially. Running the code sequentially takes a lot of time to test and improvize.

Another limitation of the model is that this model probably would not work for new users who have never rated a movie before. However, I believe that this is the reson why Netflix has newly registered users choose movies they would like to watch before giving them access to the entire library. This would help netflix recommend the first few movies they are likely to watch.

The model's accuracy could be further improved if additional variables such as the user's age, sex, location were available. This would help account for an age group effect, female/male effect, and regional preferences when predicting ratings and recommending movies.