HarvardX Movie Lens Data Science Project: Movie Recommendation System

JST

November 25, 2023

Movie Lens Data Science Project

Problem Statement

This project was created as an example of how to create a movie recommendation system analogous to today's most popular movie streaming platforms.The emphasis is on predicting ratings using the public Movie Lens dataset.

Initial Setup

Load Libraries

```r
library(recommenderlab)
library(ggplot2)
library(data.table)
library(reshape2)
library(tidyverse)
library(caret)
library(stringr)
library(magrittr)
library(tm)
library(wordcloud)
if (!require(dplyr)) {
  install.packages("dplyr")
  library(dplyr)
}

options(timeout = 120)
```

Project Code

Data files and required code to utilize for this specific project in order to train, test, and evaluate the system.

```r
#Import data files

dl <- "ml-10M100K.zip"

ratings_file <- "ml-10M100K/ratings.dat"
if(!file.exists(ratings_file))
  unzip(dl, ratings_file)

movies_file <- "ml-10M100K/movies.dat"
if(!file.exists(movies_file))
  unzip(dl, movies_file)

#Create Data Frames

ratings <- as.data.frame(str_split(readLines(ratings_file), fixed("::"), simplify = TRUE),
                         stringsAsFactors = FALSE)
colnames(ratings) <- c("userId", "movieId", "rating", "timestamp")
ratings <- ratings %>%
  mutate(userId = as.integer(userId),
         movieId = as.integer(movieId),
         rating = as.numeric(rating),
         timestamp = as.integer(timestamp))

movies <- as.data.frame(str_split(readLines(movies_file), fixed("::"), simplify = TRUE),
                        stringsAsFactors = FALSE)
colnames(movies) <- c("movieId", "title", "genres")
movies <- movies %>%
  mutate(movieId = as.integer(movieId))

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

# Final hold-out test set will be 10% of MovieLens data
set.seed(1, sample.kind="Rounding") # if using R 3.6 or later
# set.seed(1) # if using R 3.5 or earlier
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 final hold-out test set are also in edx set
final_holdout_test <- temp %>%
  semi_join(edx, by = "movieId") %>%
  semi_join(edx, by = "userId")

# Add rows removed from final hold-out test set back into edx set
removed <- anti_join(temp, final_holdout_test)
edx <- rbind(edx, removed)
```

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

#Train and Test Sets

set.seed(1, sample.kind="Rounding") # if using R 3.6 or later
# set.seed(1) # if using R 3.5 or earlier
test_index <- createDataPartition(y = edx$rating, times = 1, p = 0.1, list = FALSE)
edx_train <- edx[-test_index,]
temp <- edx[test_index,]

edx_test <- temp %>%
  semi_join(edx_train, by = "movieId") %>%
  semi_join(edx_train, by = "userId")

removed <- anti_join(temp, edx_test)
edx_train <- rbind(edx_train, removed)

rm(test_index, temp, removed)
```

Exploratory Analysis

Basic exploratory analysis to determine what the dataframe looks like along with other basic information such as the number of columns and what the titles of the columns are in the data frame.

```r
edx %>% as_tibble()
```

```
## # A tibble: 9,000,055 x 6
##    userId movieId rating timestamp title                                  genres
##     <int>   <int>  <dbl>     <int> <chr>                                  <chr>
##  1      1     122      5 838985046 Boomerang (1992)                       Comed~
##  2      1     185      5 838983525 Net, The (1995)                        Actio~
##  3      1     292      5 838983421 Outbreak (1995)                        Actio~
##  4      1     316      5 838983392 Stargate (1994)                        Actio~
##  5      1     329      5 838983392 Star Trek: Generations (1994)          Actio~
##  6      1     355      5 838984474 Flintstones, The (1994)                Child~
##  7      1     356      5 838983653 Forrest Gump (1994)                    Comed~
##  8      1     362      5 838984885 Jungle Book, The (1994)                Adven~
##  9      1     364      5 838983707 Lion King, The (1994)                  Adven~
## 10      1     370      5 838984596 Naked Gun 33 1/3: The Final Insult (1~ Actio~
## # i 9,000,045 more rows
```

```r
str(edx)
```

```
## 'data.frame':    9000055 obs. of  6 variables:
##  $ userId   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ movieId  : int  122 185 292 316 329 355 356 362 364 370 ...
##  $ rating   : num  5 5 5 5 5 5 5 5 5 5 ...
##  $ timestamp: int  838985046 838983525 838983421 838983392 838983392 838984474 838983653 838984885 8
##  $ title    : chr  "Boomerang (1992)" "Net, The (1995)" "Outbreak (1995)" "Stargate (1994)" ...
##  $ genres   : chr  "Comedy|Romance" "Action|Crime|Thriller" "Action|Drama|Sci-Fi|Thriller" "Action|Ad
```

```r
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
##
##
##
```

```r
head(edx)
```

```
##   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
```

```r
tail(edx)
```

```
##     userId movieId rating  timestamp                         title
## 3    30445    8394    0.5 1200074027           Hi-Line, The (1999)
## 42   32620   33140    3.5 1173562747         Down and Derby (2005)
## 513  40976   61913    3.0 1227767528           Africa addio (1966)
## 61   59269   63141    2.0 1226443318 Rockin’ in the Rockies (1945)
## 77   60713    4820    2.0 1119156754  Won’t Anybody Listen? (2000)
## 834  64621   39429    2.5 1201248182                Confess (2005)
##                      genres
## 3                     Drama
## 42          Children|Comedy
## 513             Documentary
## 61  Comedy|Musical|Western
## 77              Documentary
## 834          Drama|Thriller
```

```
nrow(edx)
```

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

```
ncol(edx)
```

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

```
n_distinct(edx$userId)
```

```
## [1] 69878
```

```
edx %>%
  summarize(n_users = n_distinct(userId),
            n_movies = n_distinct(movieId))
```

```
##   n_users n_movies
## 1   69878    10677
```

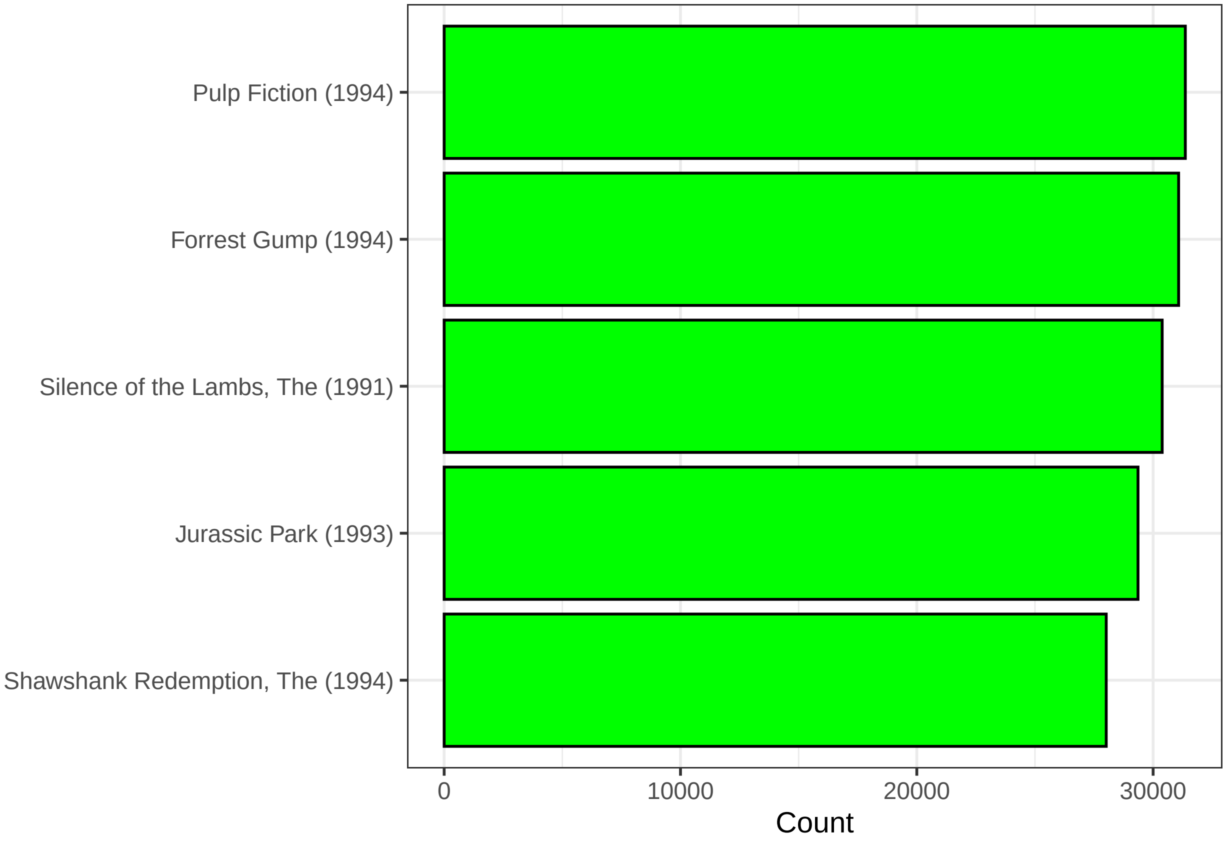
Top 5 Most Popular Movies

Movies from the 90's dominate the Top 5 most highly rated movies.

```
edx_movie_ratings <- edx %>% group_by(title) %>% summarize(ratings_count = n())

edx_sorted_movies <- edx_movie_ratings %>% arrange(desc(ratings_count))

top_5 <- head(edx_sorted_movies, 5)

print(top_5)
```

```
## # A tibble: 5 x 2
##   title                            ratings_count
##   <chr>                                    <int>
## 1 Pulp Fiction (1994)                      31362
## 2 Forrest Gump (1994)                      31079
## 3 Silence of the Lambs, The (1991)         30382
## 4 Jurassic Park (1993)                     29360
## 5 Shawshank Redemption, The (1994)         28015
```

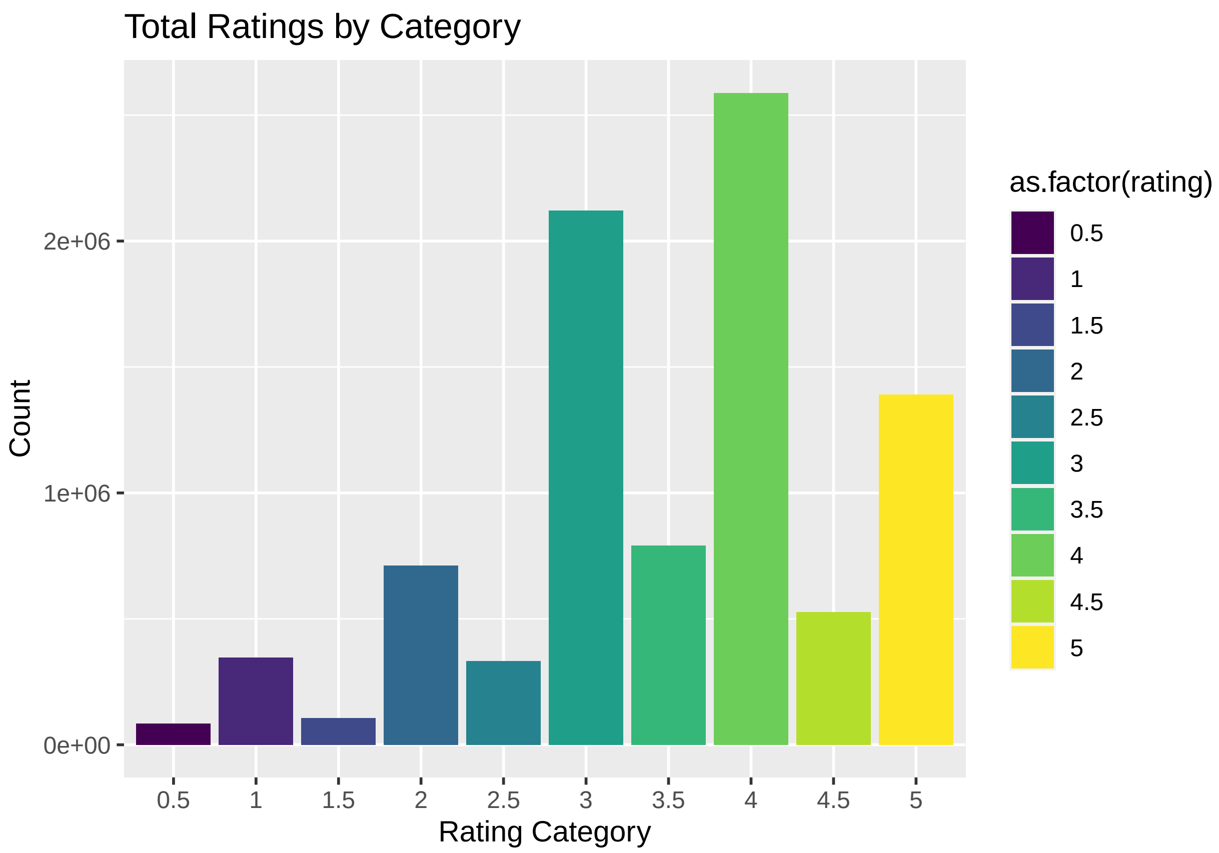
```
edx %>%
  group_by(title) %>%
  summarize(count = n()) %>%
  arrange(-count) %>%
  top_n(5, count) %>%
  ggplot(aes(count, reorder(title, count))) +
  geom_bar(color = "black", fill = "green", stat = "identity") +
  xlab("Count") +
  ylab(NULL) +
  theme_bw()
```



Ratings by Count

The Ratings by count graph provides valuable insight into which number has the most ratings. The number that is rated the most is 4 and the second highest is 3.

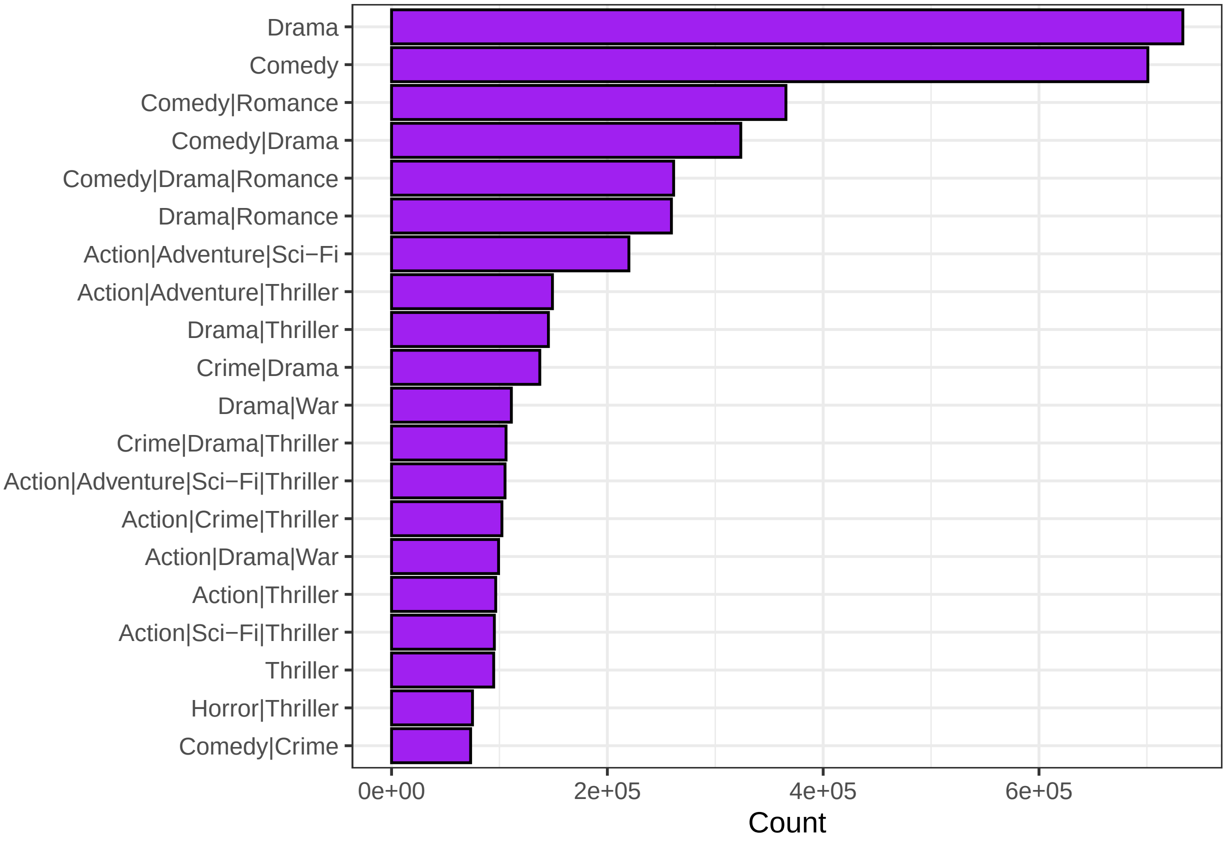
```
edx_ratings <- edx %>% group_by(rating) %>% summarize(ratings_count = n())

ggplot(edx_ratings, aes(x = as.factor(rating), y = ratings_count, fill = as.factor(rating))) +
  geom_bar(stat = "identity") +
  scale_fill_viridis_d() +
  labs(title = "Total Ratings by Category",
       x = "Rating Category",
       y = "Count")
```



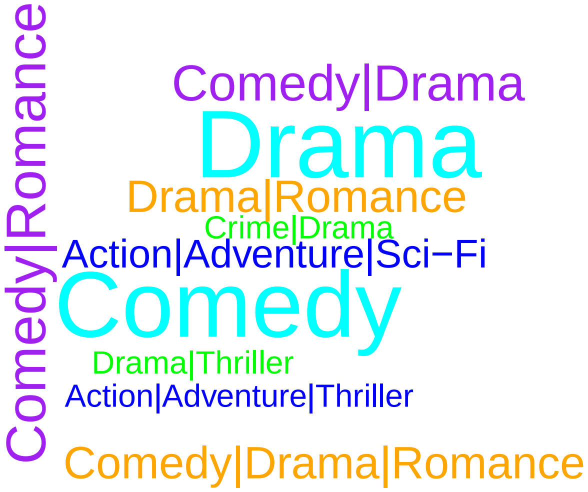
Most Popular Genres

Drama and Comedy are the most popular genres by far. The two genres and subsets of them make up the top 5 most popular genres.

```
edx %>%
  group_by(genres) %>%
  summarize(count = n()) %>%
  arrange(-count) %>%
  top_n(20, count) %>%
  ggplot(aes(count, reorder(genres, count))) +
  geom_bar(color = "black", fill = "purple", stat = "identity") +
  xlab("Count") +
  ylab(NULL) +
  theme_bw()
```



```
#V2
edx_wordcl <- edx %>%
  group_by(genres) %>%
  summarize(count = n()) %>%
  arrange(-count) %>%
  top_n(10)

edx_wordcl <- na.omit(edx_wordcl)

color_palette <- c("red", "green", "blue", "orange", "purple", "pink", "brown", "gray", "yellow", "cyan

wordcloud(words = edx_wordcl$genres, freq = edx_wordcl$count, scale=c(3,0.5), colors=color_palette, min
```



Models

Model 1 - Average

The first model will build the Recommendation System and Residual Mean Squared Error (RMSE) will be used to evaluate accuracy of the model.

```
RMSE <- function(true_ratings, predicted_ratings){
     sqrt(mean((true_ratings - predicted_ratings)^2))
}

mu_hat <- mean(edx_train$rating)
mu_hat
```

```
## [1] 3.512456
```

```
naive_rmse <- RMSE(edx_test$rating, mu_hat)
naive_rmse
```

```
## [1] 1.060054
```

```
predictions <- rep(2.5, nrow(edx_test))
RMSE(edx_test$rating, predictions)
```

```
## [1] 1.465938
```

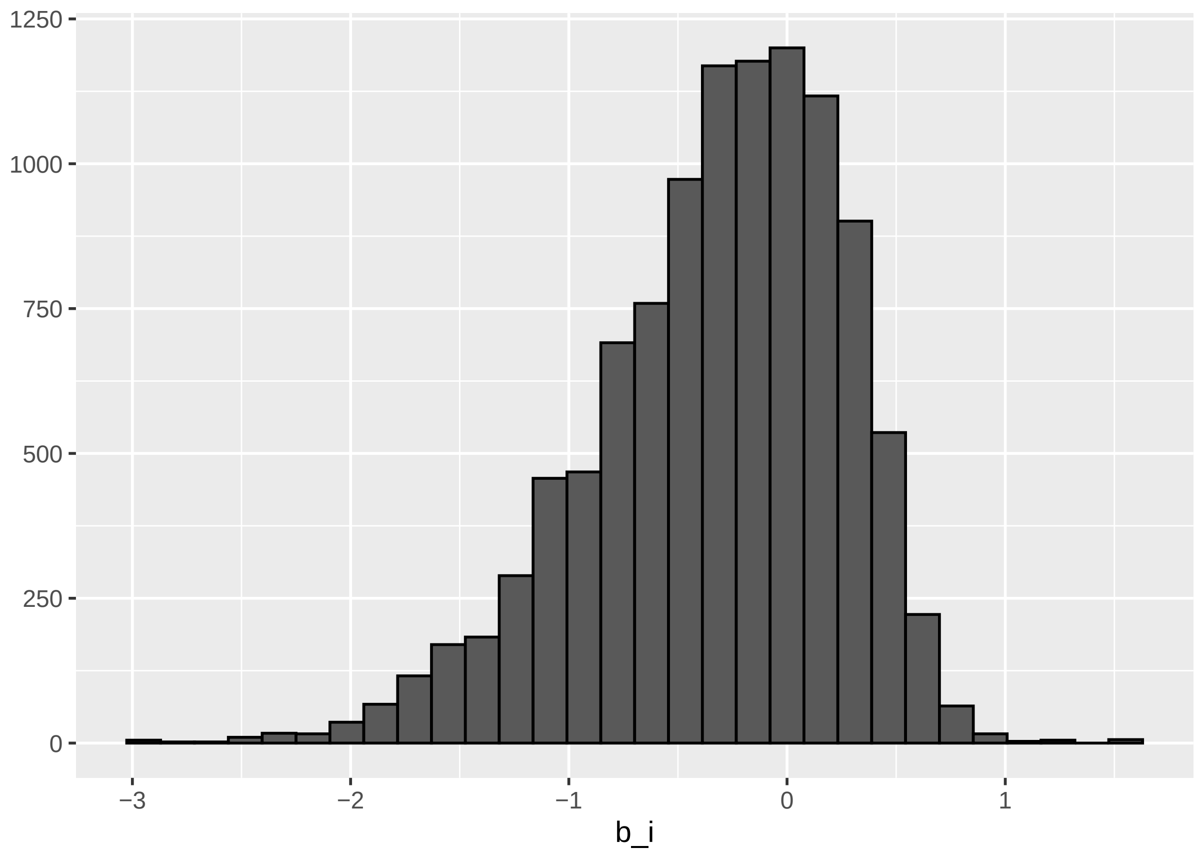
```
rmse_results <- tibble(method = "Just the average", RMSE = naive_rmse)

rmse_results
```

```
## # A tibble: 1 x 2
##   method            RMSE
##   <chr>            <dbl>
## 1 Just the average  1.06
```

Model 2

Model 2 is implementing a movie effect model. This model focuses on collaborative filtering which is a commonly used techniqe. The collaborative filtering provides personalized suggestions by leveraging group data to make recommendations for another user. Therefore, it is based on user-based collaborative filtering.

```
mu <- mean(edx_train$rating)
movie_avgs <- edx_train %>%
     group_by(movieId) %>%
     summarize(b_i = mean(rating - mu))


movie_avgs %>% qplot(b_i, geom ="histogram", bins = 30, data = ., color = I("black"))
```



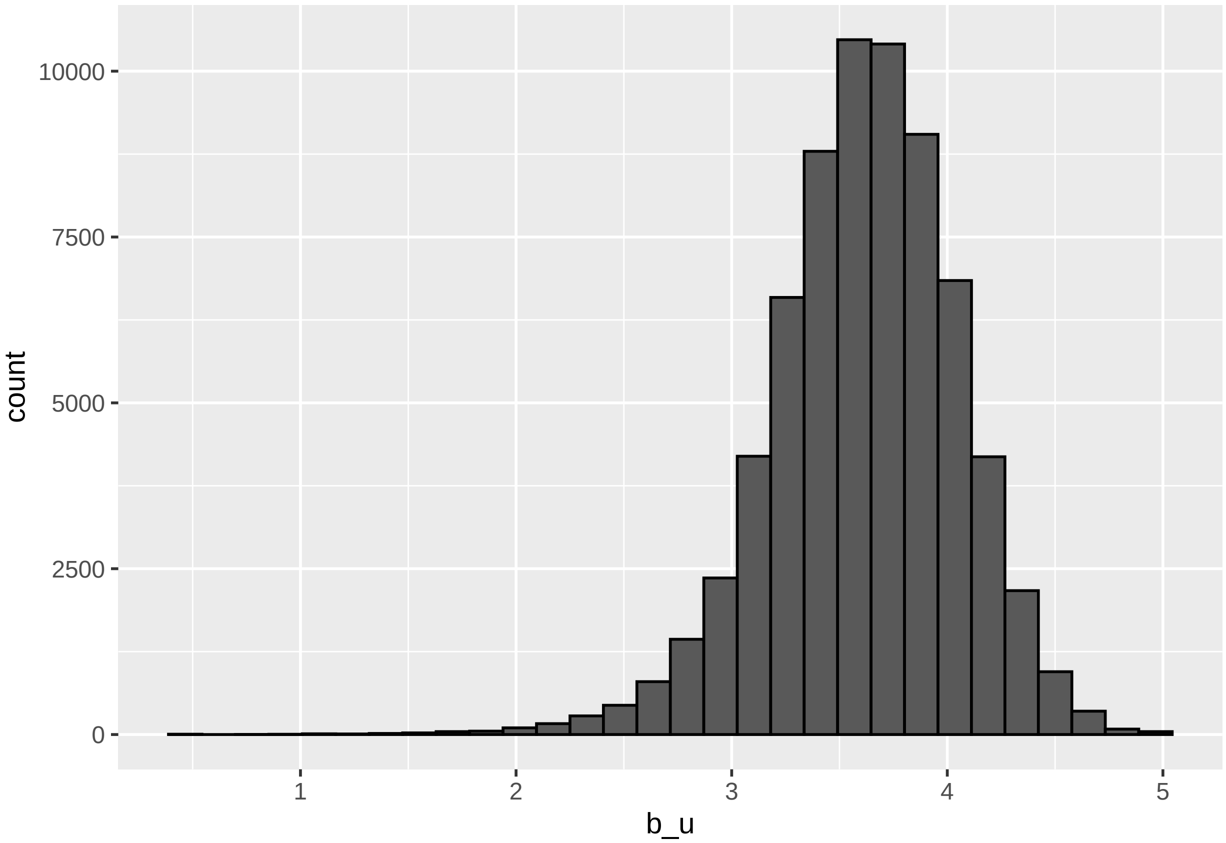
```
predicted_ratings <- mu + edx_test %>%
     left_join(movie_avgs, by='movieId') %>%
     pull(b_i)

model_1_rmse <- RMSE(predicted_ratings, edx_test$rating)
rmse_results <- bind_rows(rmse_results,
                          tibble(method="Movie Effect Model",
                                     RMSE = model_1_rmse ))
rmse_results %>% knitr::kable()
```

method	RMSE
Just the average	1.0600537
Movie Effect Model	0.9429615

Model 3

Model 3 focuses on implementing both movie effects and user effects.

```
edx_train %>%
     group_by(userId) %>%
     summarize(b_u = mean(rating)) %>%
     filter(n()>=100) %>%
     ggplot(aes(b_u)) +
     geom_histogram(bins = 30, color = "black")
```



```
# lm(rating ~ as.factor(movieId) + as.factor(userId))
user_avgs <- edx_train %>%
    left_join(movie_avgs, by='movieId') %>%
    group_by(userId) %>%
    summarize(b_u = mean(rating - mu - b_i))

predicted_ratings <- edx_test %>%
    left_join(movie_avgs, by='movieId') %>%
    left_join(user_avgs, by='userId') %>%
    mutate(pred = mu + b_i + b_u) %>%
    pull(pred)

model_2_rmse <- RMSE(predicted_ratings, edx_test$rating)

rmse_results <- bind_rows(rmse_results,
                          tibble(method="Movie + User Effects Model",
                                       RMSE = model_2_rmse ))
rmse_results %>% knitr::kable()
```

method	RMSE
Just the average	1.0600537
Movie Effect Model	0.9429615
Movie + User Effects Model	0.8646843

Model 4

Provides a regularized move effect model

```
edx_test %>%
     left_join(movie_avgs, by='movieId') %>%
     mutate(residual = rating - (mu + b_i)) %>%
     arrange(desc(abs(residual))) %>%
     dplyr::select(title,  residual) %>% slice(1:10) %>% pull(title)
```

```
##  [1] "From Justin to Kelly (2003)"      "Shawshank Redemption, The (1994)"
##  [3] "Shawshank Redemption, The (1994)" "Godfather, The (1972)"
##  [5] "Godfather, The (1972)"            "Godfather, The (1972)"
##  [7] "Godfather, The (1972)"            "Usual Suspects, The (1995)"
##  [9] "Schindler's List (1993)"          "Schindler's List (1993)"
```

```
movie_titles <- edx %>%
     dplyr::select(movieId, title) %>%
     distinct()

# 10 best
movie_avgs %>% left_join(movie_titles, by="movieId") %>%
     arrange(desc(b_i)) %>%
     dplyr::select(title, b_i) %>%
     slice(1:10) %>%
     pull(title)
```

```
##  [1] "Hellhounds on My Trail (1999)"
##  [2] "Satan's Tango (Sátántangó) (1994)"
##  [3] "Shadows of Forgotten Ancestors (1964)"
##  [4] "Fighting Elegy (Kenka erejii) (1966)"
##  [5] "Sun Alley (Sonnenallee) (1999)"
##  [6] "Blue Light, The (Das Blaue Licht) (1932)"
##  [7] "Who's Singin' Over There? (a.k.a. Who Sings Over There) (Ko to tamo peva) (1980)"
##  [8] "Life of Oharu, The (Saikaku ichidai onna) (1952)"
##  [9] "Human Condition II, The (Ningen no joken II) (1959)"
## [10] "Human Condition III, The (Ningen no joken III) (1961)"
```

```
# 10 worst
movie_avgs %>% left_join(movie_titles, by="movieId") %>%
     arrange(b_i) %>%
     dplyr::select(title, b_i) %>%
     slice(1:10) %>%
     pull(title)
```

```
##  [1] "Besotted (2001)"
##  [2] "Hi-Line, The (1999)"
##  [3] "Accused (Anklaget) (2005)"
##  [4] "Confessions of a Superhero (2007)"
##  [5] "War of the Worlds 2: The Next Wave (2008)"
##  [6] "SuperBabies: Baby Geniuses 2 (2004)"
##  [7] "Disaster Movie (2008)"
##  [8] "From Justin to Kelly (2003)"
##  [9] "Hip Hop Witch, Da (2000)"
## [10] "Criminals (1996)"
```

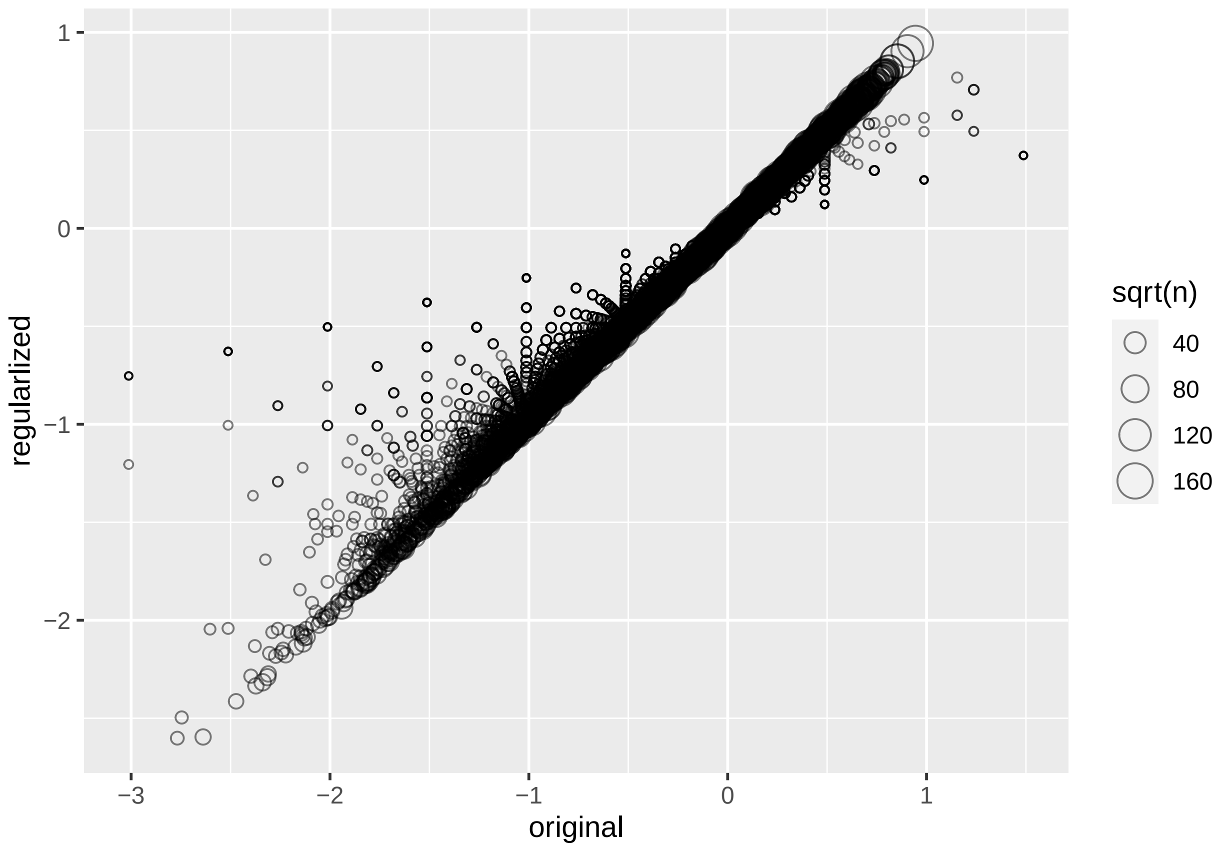
```
edx_train %>% count(movieId) %>%
     left_join(movie_avgs) %>%
     left_join(movie_titles, by="movieId") %>%
     arrange(desc(b_i)) %>%
     slice(1:10) %>%
     pull(n)
```

```
##  [1] 1 1 1 1 1 1 4 2 4 4
```

```
edx_train %>% dplyr::count(movieId) %>%
     left_join(movie_avgs) %>%
     left_join(movie_titles, by="movieId") %>%
     arrange(b_i) %>%
     slice(1:10) %>%
     pull(n)
```

```
##  [1]   1   1   1   1   2  47  30 183  11   1
```

```
lambda <- 3
mu <- mean(edx_train$rating)
movie_reg_avgs <- edx_train %>%
     group_by(movieId) %>%
     summarize(b_i = sum(rating - mu)/(n()+lambda), n_i = n())


tibble(original = movie_avgs$b_i,
           regularlized = movie_reg_avgs$b_i,
           n = movie_reg_avgs$n_i) %>%
     ggplot(aes(original, regularlized, size=sqrt(n))) +
     geom_point(shape=1, alpha=0.5)
```



```
edx_train %>%
     count(movieId) %>%
     left_join(movie_reg_avgs, by="movieId") %>%
     left_join(movie_titles, by="movieId") %>%
     arrange(desc(b_i)) %>%
     dplyr::select(title, b_i, n) %>%
     slice(1:10) %>%
     pull(title)
```

```
##  [1] "Shawshank Redemption, The (1994)"
##  [2] "Godfather, The (1972)"
##  [3] "Usual Suspects, The (1995)"
##  [4] "Schindler's List (1993)"
##  [5] "Rear Window (1954)"
##  [6] "Casablanca (1942)"
##  [7] "Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)"
##  [8] "Double Indemnity (1944)"
##  [9] "Seven Samurai (Shichinin no samurai) (1954)"
## [10] "Paths of Glory (1957)"
```

```
edx_train %>%
     dplyr::count(movieId) %>%
     left_join(movie_reg_avgs, by="movieId") %>%
     left_join(movie_titles, by="movieId") %>%
     arrange(b_i) %>%
     dplyr::select(title, b_i, n) %>%
     slice(1:10) %>%
     pull(title)
```

```
##  [1] "SuperBabies: Baby Geniuses 2 (2004)"
##  [2] "From Justin to Kelly (2003)"
##  [3] "Disaster Movie (2008)"
##  [4] "Pokémon Heroes (2003)"
##  [5] "Barney's Great Adventure (1998)"
##  [6] "Glitter (2001)"
##  [7] "Gigli (2003)"
##  [8] "Carnosaur 3: Primal Species (1996)"
##  [9] "Pokemon 4 Ever (a.k.a. Pokémon 4: The Movie) (2002)"
## [10] "Faces of Death 6 (1996)"
```

```
predicted_ratings <- edx_test %>%
     left_join(movie_reg_avgs, by='movieId') %>%
     mutate(pred = mu + b_i) %>%
     pull(pred)

model_3_rmse <- RMSE(predicted_ratings, edx_test$rating)
rmse_results <- bind_rows(rmse_results,
                          tibble(method="Regularized Movie Effect Model",
                                      RMSE = model_3_rmse ))
rmse_results %>% knitr::kable()
```

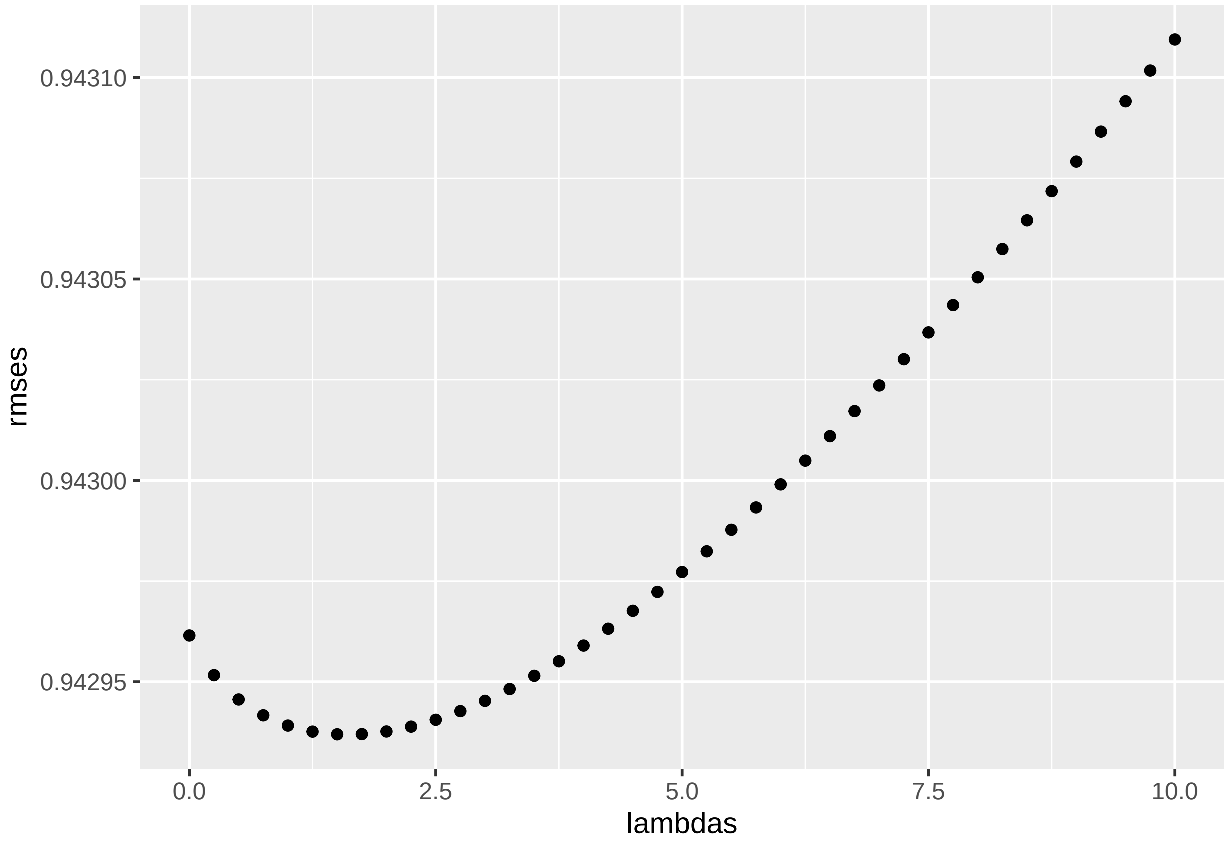
method	RMSE
Just the average	1.0600537
Movie Effect Model	0.9429615
Movie + User Effects Model	0.8646843
Regularized Movie Effect Model	0.9429453

Model 5

Model 5 focuses on model selection for regularized movie and user effect combined.

```
lambdas <- seq(0, 10, 0.25)
mu <- mean(edx_train$rating)
just_the_sum <- edx_train %>%
     group_by(movieId) %>%
     summarize(s = sum(rating - mu), n_i = n())
rmses <- sapply(lambdas, function(l){
     predicted_ratings <- edx_test %>%
          left_join(just_the_sum, by='movieId') %>%
          mutate(b_i = s/(n_i+l)) %>%
          mutate(pred = mu + b_i) %>%
          pull(pred)
```

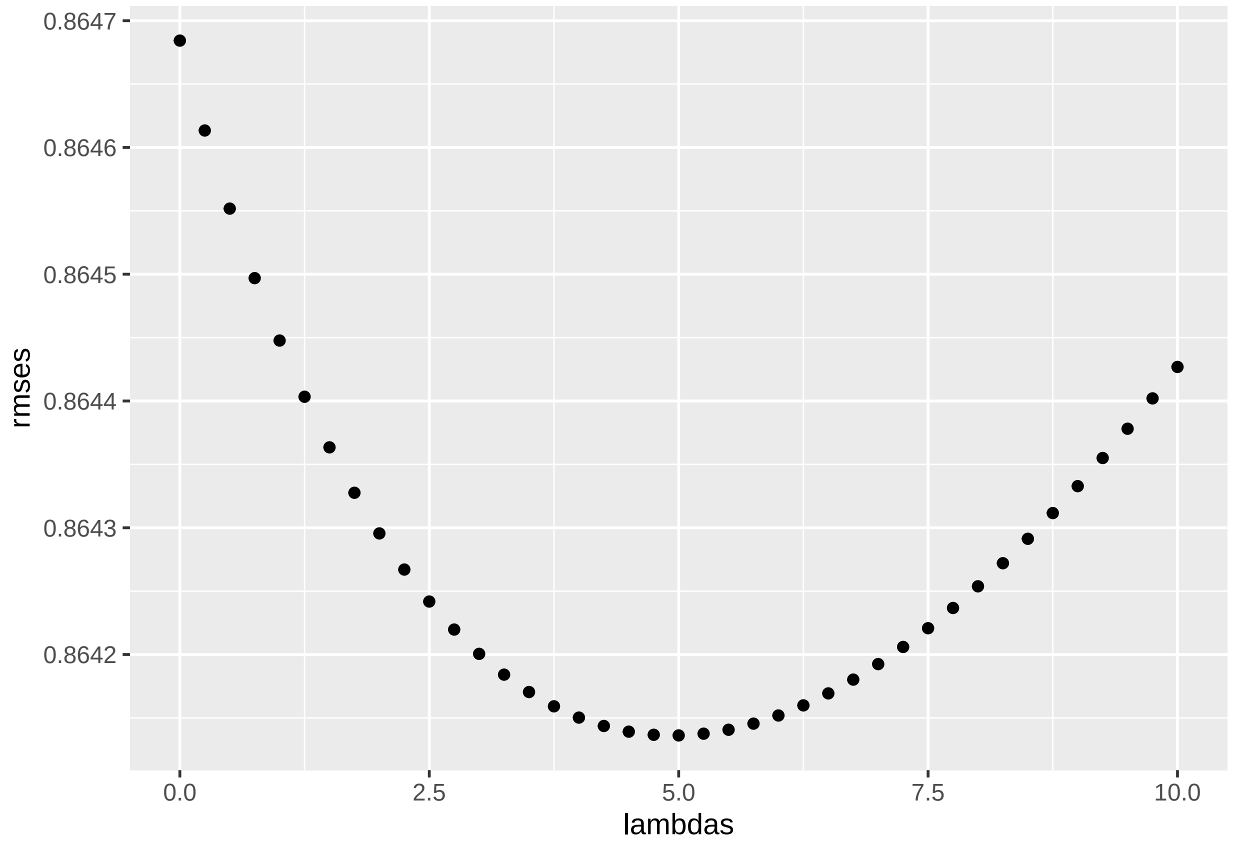
```
    return(RMSE(predicted_ratings, edx_test$rating))
})
qplot(lambdas, rmses)
```



```
lambdas[which.min(rmses)]
```

```
## [1] 1.5
```

```
lambdas <- seq(0, 10, 0.25)
rmses <- sapply(lambdas, function(l){
    mu <- mean(edx_train$rating)
    b_i <- edx_train %>%
        group_by(movieId) %>%
        summarize(b_i = sum(rating - mu)/(n()+l))
    b_u <- edx_train %>%
        left_join(b_i, by="movieId") %>%
        group_by(userId) %>%
        summarize(b_u = sum(rating - b_i - mu)/(n()+l))
    predicted_ratings <-
        edx_test %>%
        left_join(b_i, by = "movieId") %>%
        left_join(b_u, by = "userId") %>%
        mutate(pred = mu + b_i + b_u) %>%
        pull(pred)
    return(RMSE(predicted_ratings, edx_test$rating))
})

qplot(lambdas, rmses)
```



```
lambda <- lambdas[which.min(rmses)]
lambda
```

```
## [1] 5
```

```
rmse_results <- bind_rows(rmse_results,
                          tibble(method="Regularized Movie + User Effect Model",
                                     RMSE = min(rmses)))
rmse_results %>% knitr::kable()
```

method	RMSE
Just the average	1.0600537
Movie Effect Model	0.9429615
Movie + User Effects Model	0.8646843
Regularized Movie Effect Model	0.9429453
Regularized Movie + User Effect Model	0.8641362

Evaluation

The final hold out test that was created in the beginning is now used to test against our models.

```r
mu <- mean(final_holdout_test$rating)
l <- 0.15
b_i <- final_holdout_test %>%
  group_by(movieId) %>%
  summarize(b_i = sum(rating - mu)/(n() + l))

b_u <- final_holdout_test %>%
  left_join(b_i, by='movieId') %>%
  group_by(userId) %>%
  summarize(b_u = sum(rating - b_i - mu)/(n() +l))

predicted <- final_holdout_test %>%
  left_join(b_i, by = "movieId") %>%
  left_join(b_u, by = "userId") %>%
  mutate(pred = mu + b_i +  b_u) %>% .$pred

RMSE(predicted, final_holdout_test$rating)
```

```
## [1] 0.8252108
```

JST HarvardX Data Science Capstone Project