RMSE-Analysis

Mruthyum J Thatipamala

September 18, 2020

Step1: Clean up heap memory and plots - Optimizing memory of environment. This happens at several places in the code

```
# Clear environment
rm(list = ls())
# Clear console
cat("\014")
```

```
# Clear plots
if(!is.null(dev.list())) dev.off()
```

```
## null device
##           1
```

Step2: Installing packages and loading libraries

```
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")
if(!require(lubridate)) install.packages("lubridate", repos = "http://cran.us.r-project.org")
if(!require(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")
#if(!require(magrittr)) install.packages("magrittr", repos = "http://cran.us.r-project.org")
if(!require(dplyr)) install.packages("dplyr", repos = "http://cran.us.r-project.org")


library(caret)
library(tidyverse)
#library(magrittr)
library(dplyr)
library(ggplot2)
library(lubridate)
library(rmarkdown)
```

Step3: Reading the smaller set of edx and validation data from CSV files and saving them to dataframes. Merging them into one big dataframe each using 'rbind' function

```
newtrialset1 <- read.csv(file = "newtrialset1.csv", head = TRUE, sep="\t")
newtrialset2 <- read.csv(file = "newtrialset2.csv", head = TRUE, sep="\t")
newtrialset3 <- read.csv(file = "newtrialset3.csv", head = TRUE, sep="\t")
newtrialset4 <- read.csv(file = "newtrialset4.csv", head = TRUE, sep="\t")
newtrialset5 <- read.csv(file = "newtrialset5.csv", head = TRUE, sep="\t")

newvalidation1 <- read.csv(file = "newvalidation1.csv", head = TRUE, sep="\t")
newvalidation2 <- read.csv(file = "newvalidation2.csv", head = TRUE, sep="\t")
newvalidation3 <- read.csv(file = "newvalidation3.csv", head = TRUE, sep="\t")

trialset <- rbind(newtrialset1, newtrialset2, newtrialset3, newtrialset4, newtrialset5)
rm(list = c("newtrialset1", "newtrialset2", "newtrialset3", "newtrialset4", "newtrialset5"))

validation <- rbind(newvalidation1, newvalidation2, newvalidation3)
rm(list = c("newvalidation1", "newvalidation2", "newvalidation3"))
```

Step4: It is observed from the data that movies belonging some genres are watched more and ratings are also higher. On contrary, some genres are watched less and rating are also low.

```
high_boxplot_genres_rating <- trialset %>% filter(genres %in% c("Drama", "Comedy", "Comedy|Romance", "C
str(high_boxplot_genres_rating)
```

```
## 'data.frame':    794025 obs. of  2 variables:
##  $ genres: chr  "Comedy|Romance" "Drama" "Drama" "Comedy|Drama|Romance" ...
##  $ rating: num  5 5 4 4.5 3 2 3 3 3 3 ...
```

```
head(high_boxplot_genres_rating)
```

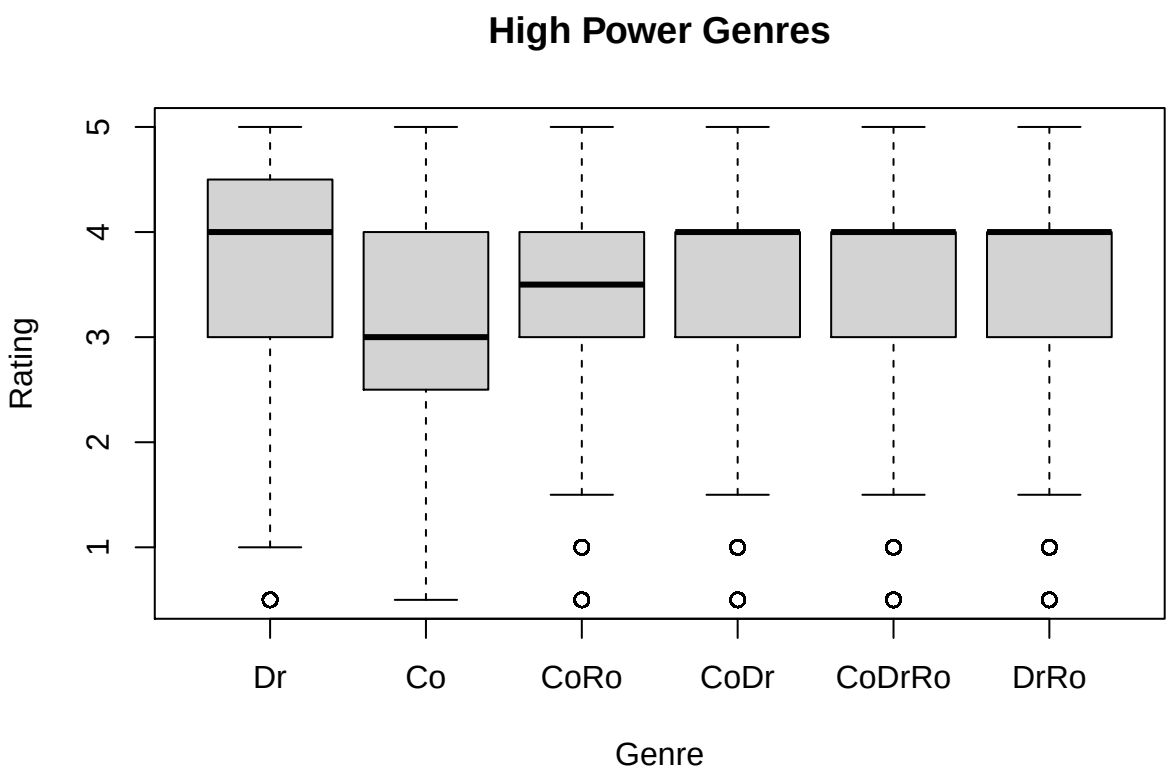
```
##                 genres rating
```

```
## 1       Comedy|Romance    5.0
## 2                Drama    5.0
## 3                Drama    4.0
## 4 Comedy|Drama|Romance    4.5
## 5       Comedy|Romance    3.0
## 6         Comedy|Drama    2.0
```

```
mean(high_boxplot_genres_rating$rating)
```

```
## [1] 3.513566
```

```
high_boxplot_genres_rating$genres <- factor(high_boxplot_genres_rating$genres, levels = c("Drama", "Com

boxplot(rating ~ genres, data = high_boxplot_genres_rating, xlab = "Genre", ylab = "Rating", main = "Hig
```




```
# Clear plot
if(!is.null(dev.list())) dev.off()
```

```
## null device
##           1
```

```
low_boxplot_genres_rating <- trialset %>% filter(genres %in% c("Action|Drama|Horror|Sci-Fi", "Action|Ro
str(low_boxplot_genres_rating)
```

```
## 'data.frame':    11 obs. of  2 variables:
##  $ genres: chr  "Adventure|Comedy|Drama|Fantasy|Mystery|Sci-Fi" "Adventure|Horror|Romance|Sci-Fi" "A
##  $ rating: num  4.5 5 3 4.5 5 3 2 4 0.5 4 ...
```

```
head(low_boxplot_genres_rating)
```

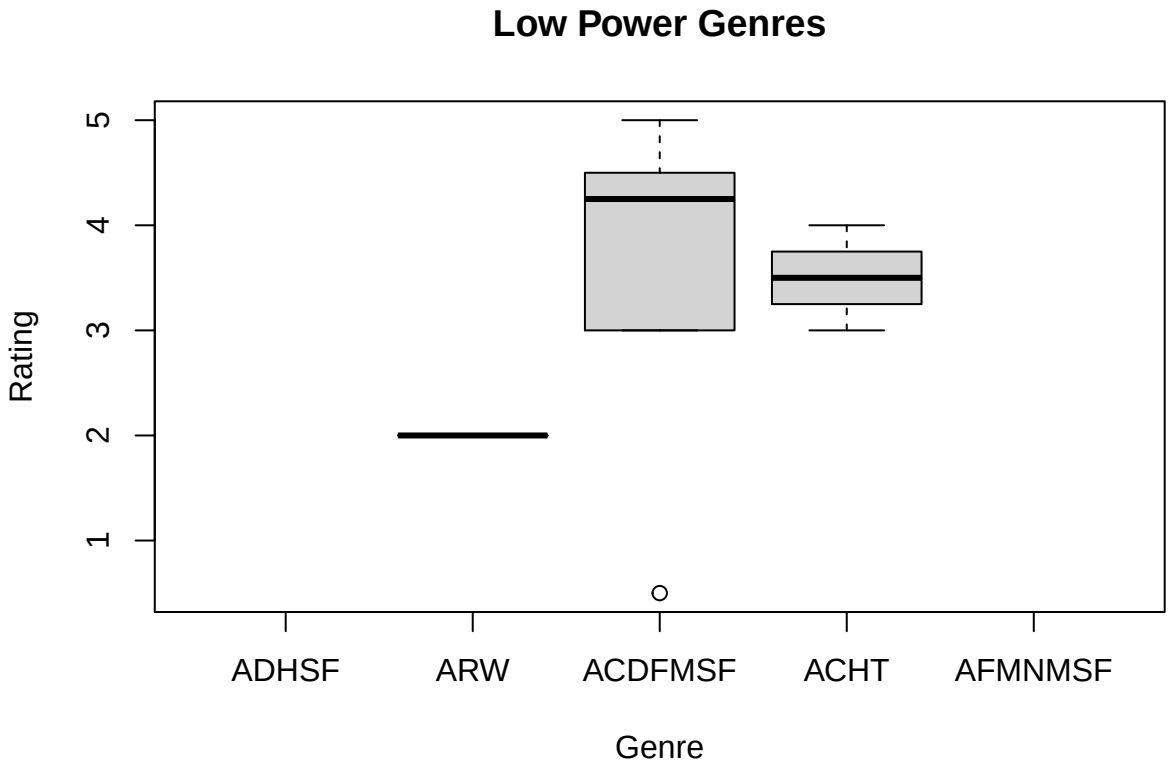
```
##                                       genres rating
## 1 Adventure|Comedy|Drama|Fantasy|Mystery|Sci-Fi    4.5
## 2               Adventure|Horror|Romance|Sci-Fi    5.0
```

```
## 3                Adventure|Crime|Horror|Thriller    3.0
## 4 Adventure|Comedy|Drama|Fantasy|Mystery|Sci-Fi    4.5
## 5 Adventure|Comedy|Drama|Fantasy|Mystery|Sci-Fi    5.0
## 6 Adventure|Comedy|Drama|Fantasy|Mystery|Sci-Fi    3.0
```

```
mean(low_boxplot_genres_rating$rating)
```

```
## [1] 3.545455
```

```
low_boxplot_genres_rating$genres <- factor(low_boxplot_genres_rating$genres, levels = c("Action|Drama|H

boxplot(rating ~ genres, data = low_boxplot_genres_rating, xlab = "Genre", ylab = "Rating", main = "Low
```




```
# Clear plot
if(!is.null(dev.list())) dev.off()
```

```
## null device
##           1
```

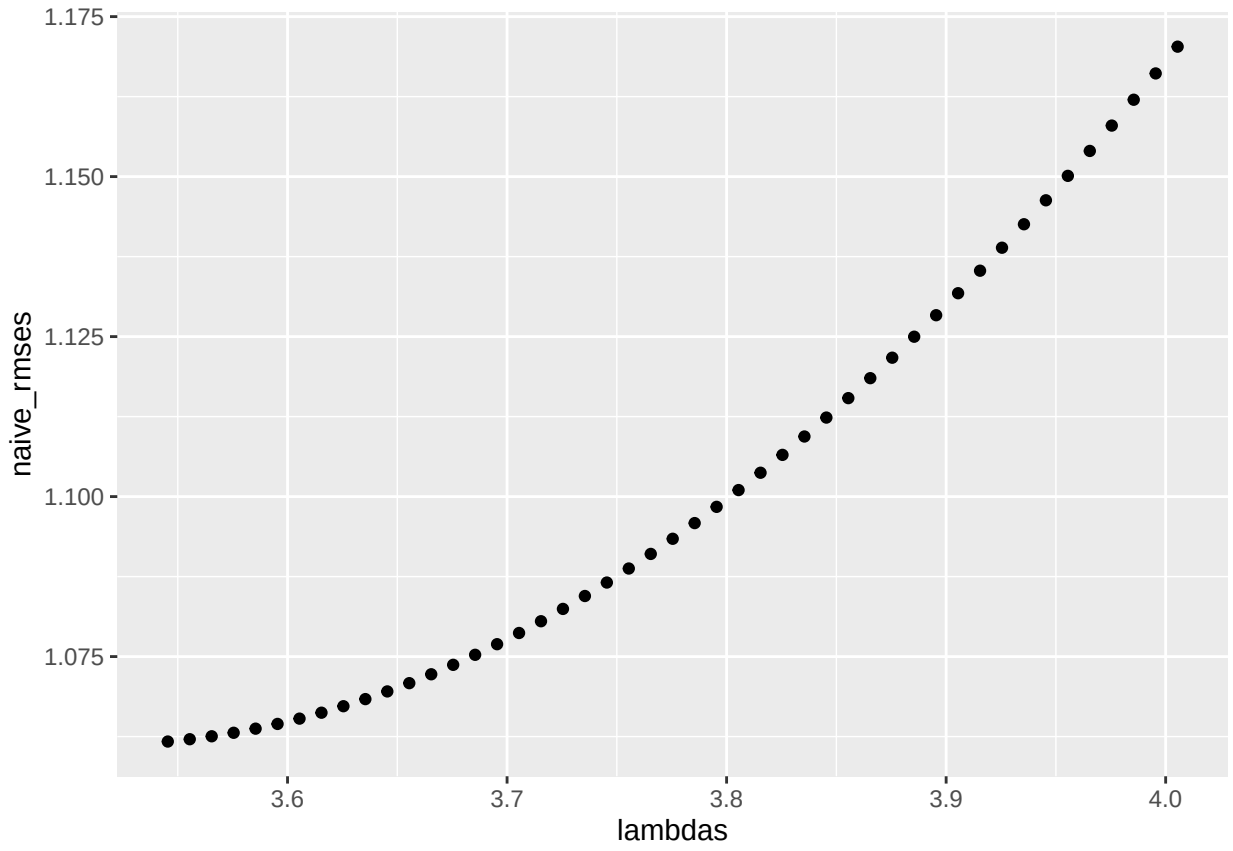
Step5:Defining RMSE function, a function that computes the RMSE for vectors of ratings and their corresponding predictors

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

Step6:Instead of taking the mean of entire dataset, we take the lower end, add increments to reach upper end to calculate the lowest naive rmse and its 'mu'. This a similar to weighted/average distribution analysis

```
lambdas<-seq(mean(low_boxplot_genres_rating$rating),mean(high_boxplot_genres_rating$rating)+0.5, 0.01)

rm(list = c("high_boxplot_genres_rating", "low_boxplot_genres_rating"))
```

```
naive_rmses <- sapply(lambdas, function(l){
  return(RMSE(validation$rating,l))
})

qplot(lambdas, naive_rmses)
```




```
# Clear plots
if(!is.null(dev.list())) dev.off()
```

```
## null device
##           1
```

```
#(Weighted) average of ratings across the train set
mu <- lambdas[which.min(naive_rmses)]
mu
```

```
## [1] 3.545455
```

```
rm(list = c("lambdas", "naive_rmses"))
```

Step6: Movie bias - We observe from the data that different movies are rated differently. Modeling movie effects by a bias, b_i

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

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
#Data set for testng the movie bias effect and calculating RMSE
movieset <- read.csv(file = "movieset.csv", head = TRUE, sep="\t")

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

predicted_ratings <- predicted_ratings %>% replace_na(mu)

RMSE(predicted_ratings, movieset$rating)
```

```
## [1] 0.940034
```

```
rm(list = c("predicted_ratings", "movieset"))
```

Step7: User bias - there is substantial variability in rating for a given movie. Different users give different rating for the same movie. Calculating user effects thru a bias

```
user_avgs <- trialset %>%
  left_join(movie_avgs, by='movieId') %>%
  group_by(userId) %>%
  summarize(b_u = mean(rating - mu - b_i))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
#Data set for testng the user bias effect and calculating RMSE
userset <- read.csv(file = "userset.csv", head = TRUE, sep="\t")

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

predicted_ratings <- predicted_ratings %>% replace_na(mu)

RMSE(predicted_ratings, userset$rating)
```

```
## [1] 0.8469732
```

```
rm(list = c("predicted_ratings", "userset"))
```

Step7: Genres bias - As calculated and observed in Step4movies belonging some genres are watched more and ratings are also higher. On contrary, some genres are watched less and rating are also low. Inclusion of genre bias

```
genres_avgs <- trialset %>%
  left_join(movie_avgs, by='movieId') %>%
  left_join(user_avgs, by='userId') %>%
  group_by(genres) %>%
  summarize(b_g = mean(rating - mu - b_i - b_u))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
#Data set for testng the genres bias effect and calculating RMSE
genresset <- read.csv(file = "genresset.csv", head = TRUE, sep="\t")

predicted_ratings <- genresset %>%
  left_join(movie_avgs, by='movieId') %>%
  left_join(user_avgs, by='userId') %>%
```

```
  left_join(genres_avgs, by='genres') %>%
  mutate(pred = mu + b_i + b_u + b_g) %>% pull(pred)

predicted_ratings <- predicted_ratings %>% replace_na(mu)

RMSE(predicted_ratings, genresset$rating)
```

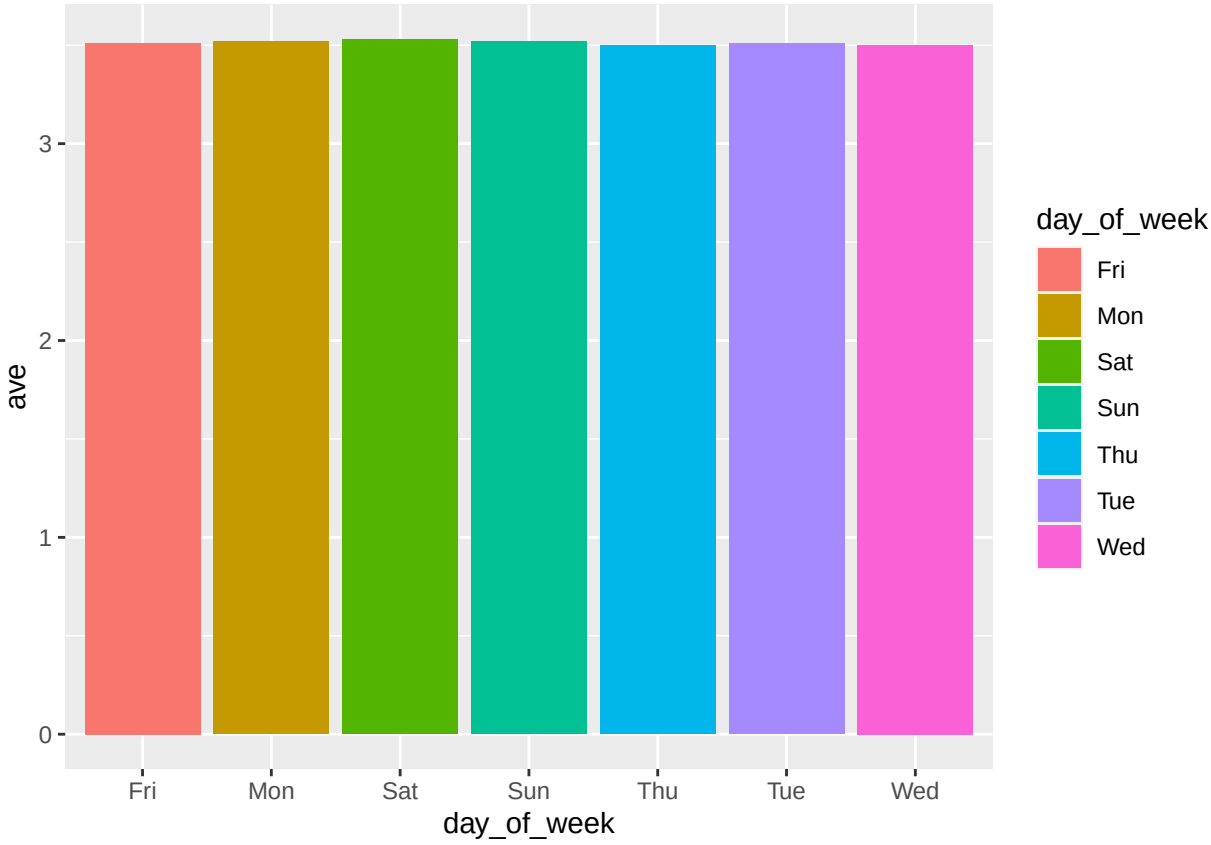
```
## [1] 0.846615
```

```
rm(list = c("predicted_ratings", "genresset"))
```

Step8: Time bias. How the ratings averages are varying on a day of the week can be observed from the bar charts plotted in the below given code. Though small change, the average of ratings given on Tues/Wed/Thursdays are less compared to other days. Accommodating time/day bias.

```
trialset %>% group_by(day_of_week) %>% summarize(total = n(), ave = mean(rating)) %>% ggplot(aes(x=day_
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```




```
#Clear plots
if(!is.null(dev.list())) dev.off()
```

```
## null device
##           1
```

```
time_avgs <- trialset %>%
  left_join(movie_avgs, by='movieId') %>%
  left_join(user_avgs, by='userId') %>%
```

```
  left_join(genres_avgs, by='genres') %>%
  group_by(day_of_week) %>%
  summarize(b_d = mean(rating - mu - b_i - b_u - b_g))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
#Data set for testng the time bias effect and calculating RMSE
timeset <- read.csv(file = "timeset.csv", head = TRUE, sep="\t")

predicted_ratings <- timeset %>%
  left_join(movie_avgs, by='movieId') %>%
  left_join(user_avgs, by='userId') %>%
  left_join(genres_avgs, by='genres') %>%
  left_join(time_avgs, by='day_of_week') %>%
  mutate(pred = mu + b_i + b_u + b_g + b_d) %>% pull(pred)

RMSE(predicted_ratings, timeset$rating)
```

```
## [1] 0.8466151
```

```
rm(list = c("predicted_ratings", "timeset"))
```

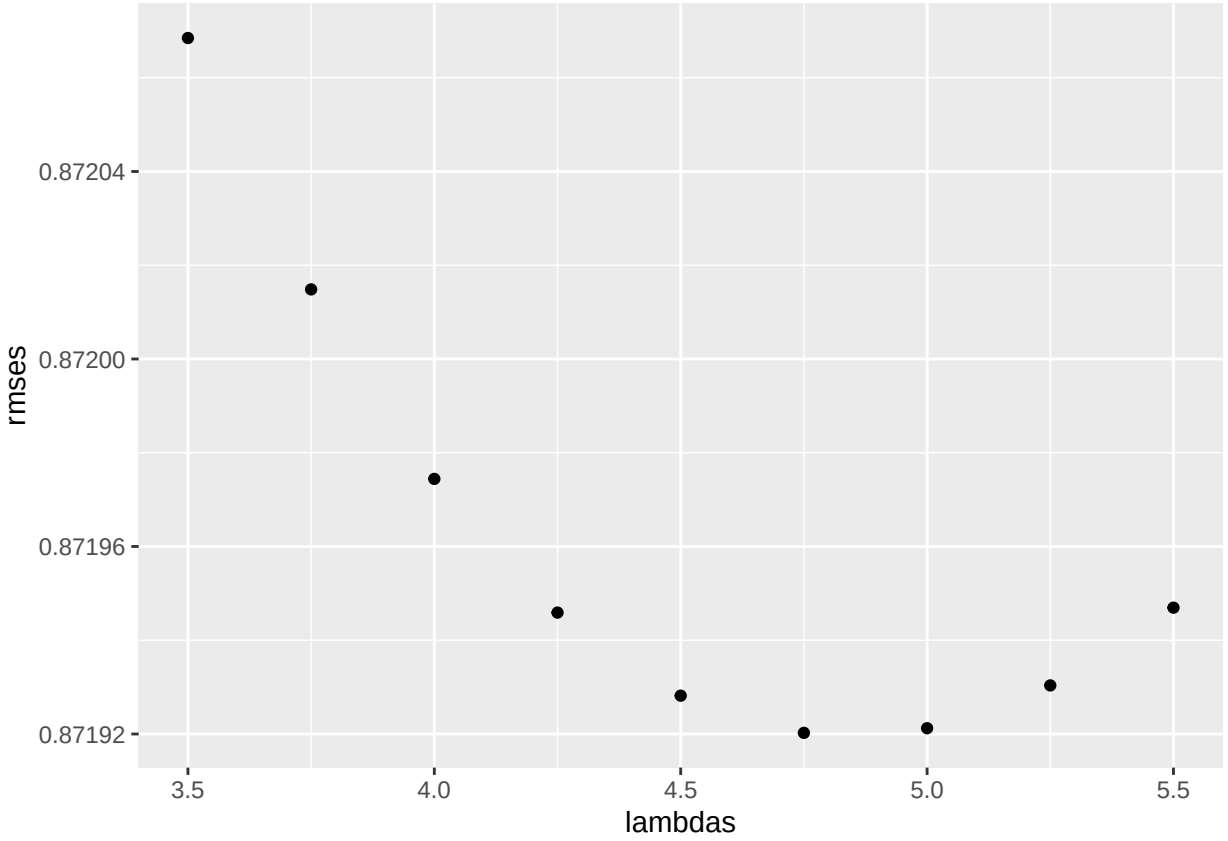
Final Step: Putting together all biases together and calculating the RMSE against validation set. This analysis also includes a tuning parameter lambda and use it for cross-validation to accommodate regularization for total variability of effect of sizes.

```
lambdas <- seq(3.5, 5.5, 0.25)
rmses <- sapply(lambdas, function(l){

b_i <- trialset %>% group_by(movieId) %>% summarize(b_i = sum(rating - mu)/(n()+l))

b_u <- trialset %>% left_join(b_i, by="movieId") %>% group_by(userId) %>% group_by(userId) %>% summariz

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

b_d <- trialset %>% left_join(movie_avgs, by='movieId') %>%
  left_join(user_avgs, by='userId') %>%
  left_join(genres_avgs, by='genres') %>%
  group_by(day_of_week) %>%
  summarize(b_d = sum(rating - mu - b_i - b_u - b_g)/(n()+l))

predicted_ratings <- validation %>%
  left_join(b_i, by='movieId') %>%
  left_join(b_u, by='userId') %>%
  left_join(b_g, by='genres') %>%
  left_join(b_d, by='day_of_week') %>%
  mutate(pred = mu + b_i + b_u + b_g + b_d) %>% pull(pred)

predicted_ratings <- predicted_ratings %>% replace_na(mu)

RMSE(predicted_ratings, validation$rating)
 return(RMSE(predicted_ratings, validation$rating))
})
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
rm(list = c("trialset", "movie_avgs", "user_avgs", "genres_avgs"))
qplot(lambdas, rmses)
```




```
# Clear plots
if(!is.null(dev.list())) dev.off()
```

```
## null device
##           1
```

```
final_rmse <- min(rmses)
final_rmse
```

```
## [1] 0.8719202
```