HarvardX Data Science Capstone: Last.fm Artist Recommender System

JST

2023-12-01

Last.fm Artist Recommender System

I. Project Topic

In this project I will create an artist recommender system using the Last.fm dataset from 2011.The dataset is comprised of 2,000 users on the Last.fm website.Data gathered includes music artist listening information, social networking, and tagging from Last.fm users. This report will emphasize recommending artists to users and a basic data exploration of the information. The User Collaborative Filtering method will be used which uses the data from other users to recommend songs from similar users.

About the Data Set

It is the first version which was released in 2011.It contains data from 1893 users, 17,632 artists, and 11946 tags.

Initial Setup

Load Libraries

```
library(ggplot2)
library(viridis)
library(tidyverse)
library(caret)
library(tm)
library(wordcloud)
library(recommenderlab)
library(reshape2)

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.In order to prepare for creating the recommender system, all the appropriate steps must be followed.

```
#Extract Data Files

zipped_file <- "hetrec2011-lastfm-2k.zip"

extracted_dir <- "C:/Users/visde/OneDrive/Desktop/LastFM"

unzip(zipped_file, exdir = extracted_dir)

cat("Files extracted to:", extracted_dir, "\n")
```

```
## Files extracted to: C:/Users/visde/OneDrive/Desktop/LastFM
```

```
# Read Files

artists_file <- "artists.dat"

USR_artists <- "user_artists.dat"

#create tibbles for data analysis

artists <- read_tsv(artists_file)

artists <- artists %>%
  rename(artistID = id)

artists <- select(artists, -url, -pictureURL)

user_artists <- read_tsv(USR_artists)

fm_total_set <- left_join(artists, user_artists, by = "artistID")

# Create 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 = fm_total_set$weight, times = 1, p = 0.1, list = FALSE)
last.fm <- fm_total_set[-test_index,]
temp <- fm_total_set[test_index,]

# Make sure uID and movieId in final hold-out test set are also in edx set
final_holdout_test <- temp %>%
  semi_join(last.fm, by = "artistID") %>%
  semi_join(last.fm, by = "userID")

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

```
last.fm <- rbind(last.fm, removed)

rm(fm_total_set, artists, user_artists, test_index, temp, 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 = last.fm$weight, times = 1, p = 0.1, list = FALSE)
last.fm_train <- last.fm[-test_index,]
temp <- last.fm[test_index,]

last.fm_test <- temp %>%
  semi_join(last.fm_train, by = "artistID") %>%
  semi_join(last.fm_train, by = "userID")

removed <- anti_join(temp, last.fm_test)
last.fm_train <- rbind(last.fm_train, removed)

rm(test_index, temp, removed)
```

Exploratory Analysis

Basic exploratory analysis to verify number of artists and users for data quality check.

```
last.fm %>% as_tibble()
```

```
## # A tibble: 84,688 x 4
##    artistID name            userID weight
##       <dbl> <chr>            <dbl>  <dbl>
##  1        1 MALICE MIZER        34    212
##  2        1 MALICE MIZER       274    483
##  3        1 MALICE MIZER       785     76
##  4        2 Diary of Dreams    135   1021
##  5        2 Diary of Dreams    257    152
##  6        2 Diary of Dreams    325   3466
##  7        2 Diary of Dreams    397     56
##  8        2 Diary of Dreams    560    134
##  9        2 Diary of Dreams    935    428
## 10        2 Diary of Dreams   1551    868
## # i 84,678 more rows
```

```
str(last.fm)
```

```
## tibble [84,688 x 4] (S3: tbl_df/tbl/data.frame)
##  $ artistID: num [1:84688] 1 1 1 2 2 2 2 2 2 2 ...
##  $ name    : chr [1:84688] "MALICE MIZER" "MALICE MIZER" "MALICE MIZER" "Diary of Dreams" ...
##  $ userID  : num [1:84688] 34 274 785 135 257 ...
##  $ weight  : num [1:84688] 212 483 76 1021 152 ...
```

```r
summary(last.fm)
```

```
##     artistID          name              userID         weight
##  Min.   :    1   Length:84688       Min.   :   2   Min.   :     1
##  1st Qu.:  441   Class :character   1st Qu.: 503   1st Qu.:   106
##  Median : 1312   Mode  :character   Median :1032   Median :   258
##  Mean   : 3431                      Mean   :1038   Mean   :   739
##  3rd Qu.: 4565                      3rd Qu.:1572   3rd Qu.:   610
##  Max.   :18745                      Max.   :2100   Max.   :352698
```

```r
head(last.fm)
```

```
## # A tibble: 6 x 4
##   artistID name            userID weight
##      <dbl> <chr>            <dbl>  <dbl>
## 1        1 MALICE MIZER        34    212
## 2        1 MALICE MIZER       274    483
## 3        1 MALICE MIZER       785     76
## 4        2 Diary of Dreams    135   1021
## 5        2 Diary of Dreams    257    152
## 6        2 Diary of Dreams    325   3466
```

```r
tail(last.fm)
```

```
## # A tibble: 6 x 4
##   artistID name               userID weight
##      <dbl> <chr>               <dbl>  <dbl>
## 1    18674 Attila İlhan         2095     95
## 2    18677 Lena Chamamyan       2095     82
## 3    18696 Hakan Yeşilyurt      2095     31
## 4    18723 Electrosoul System   2099     87
## 5    18729 Atalyja              2100    280
## 6    18737 Ciccone Youth         454    560
```

```r
nrow(last.fm)
```

```
## [1] 84688
```

```r
ncol(last.fm)
```

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

```r
n_distinct(last.fm$userID)
```

```
## [1] 1892
```

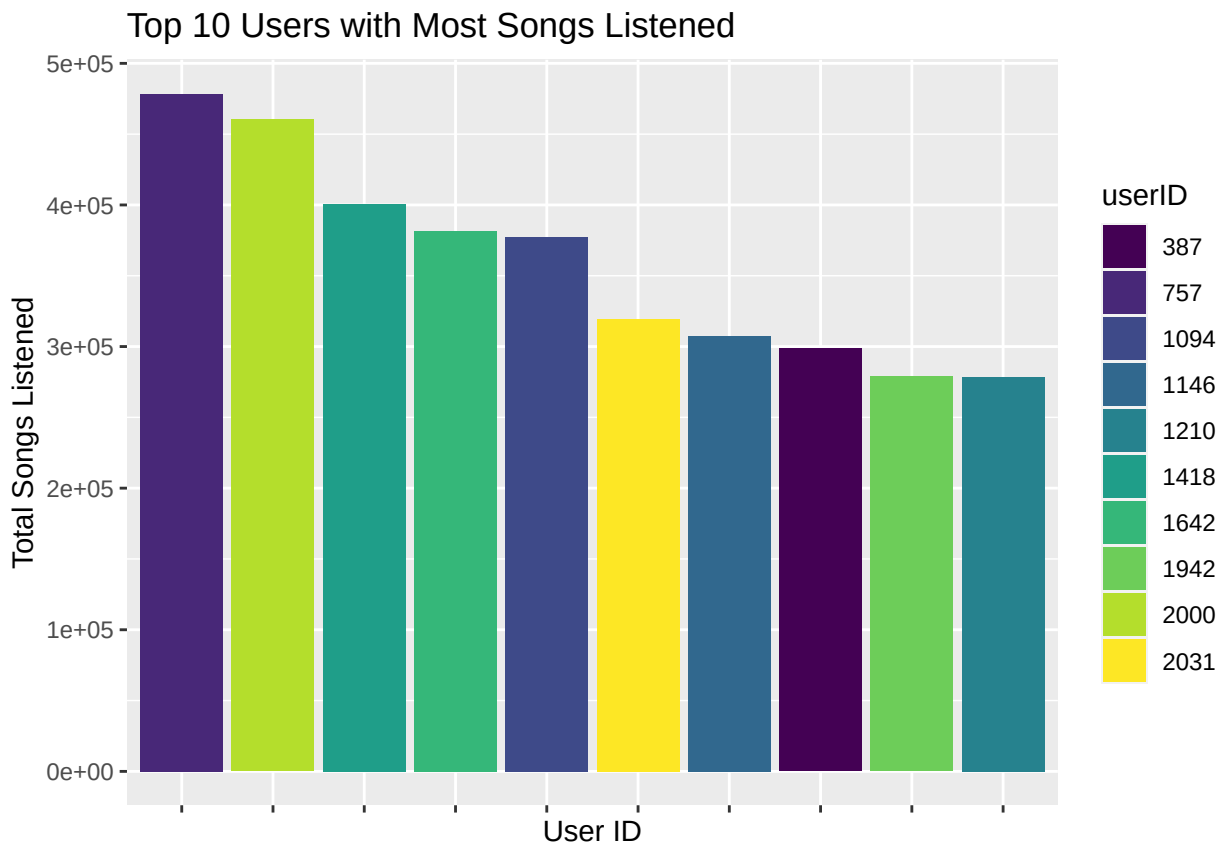
```r
last.fm %>%
  summarize(n_users = n_distinct(userID),
            n_artists = n_distinct(artistID))
```

```
## # A tibble: 1 x 2
##   n_users n_artists
##     <int>     <int>
## 1    1892     17632
```

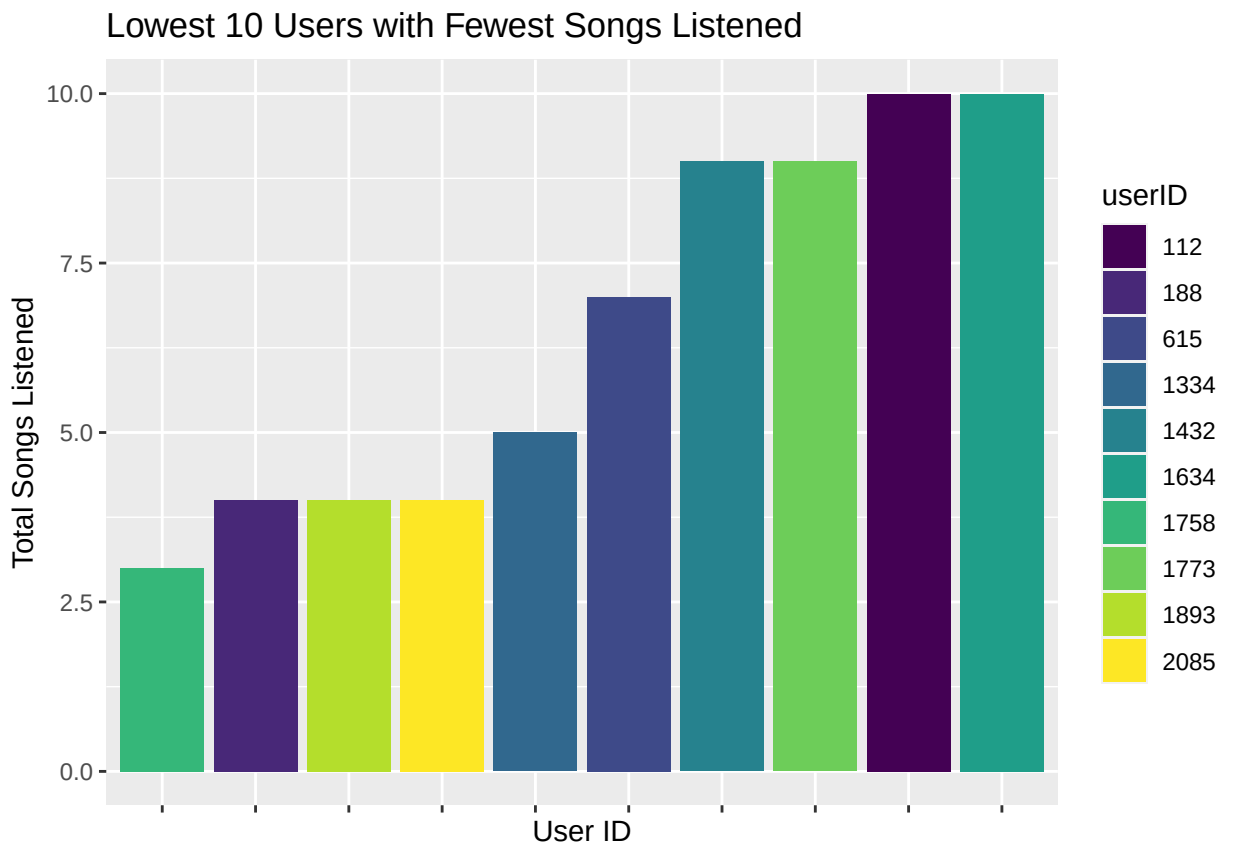
Insights

Listening Count analysis by user

```
#Average Listening Count by User
pop_artists <- last.fm %>%
  group_by(artistID) %>%
  summarize(listening_count = sum(as.numeric(weight), na.rm = TRUE)) %>%
  arrange(desc(listening_count))

#Top 10 Users

top_10_users <- last.fm %>%
  group_by(userID) %>%
  summarize(total_songs_listened = sum(as.numeric(weight), na.rm = TRUE)) %>%
  arrange(desc(total_songs_listened)) %>%
  head(10)

top_10_users$userID <- as.factor(top_10_users$userID)

ggplot(top_10_users, aes(x = reorder(userID, -total_songs_listened), y = total_songs_listened, fill = us
  geom_bar(stat = "identity") +
  scale_fill_viridis_d() +  # Use the viridis color palette
  labs(title = "Top 10 Users with Most Songs Listened",
       x = "User ID",
       y = "Total Songs Listened") +
  theme(axis.text.x = element_blank())
```

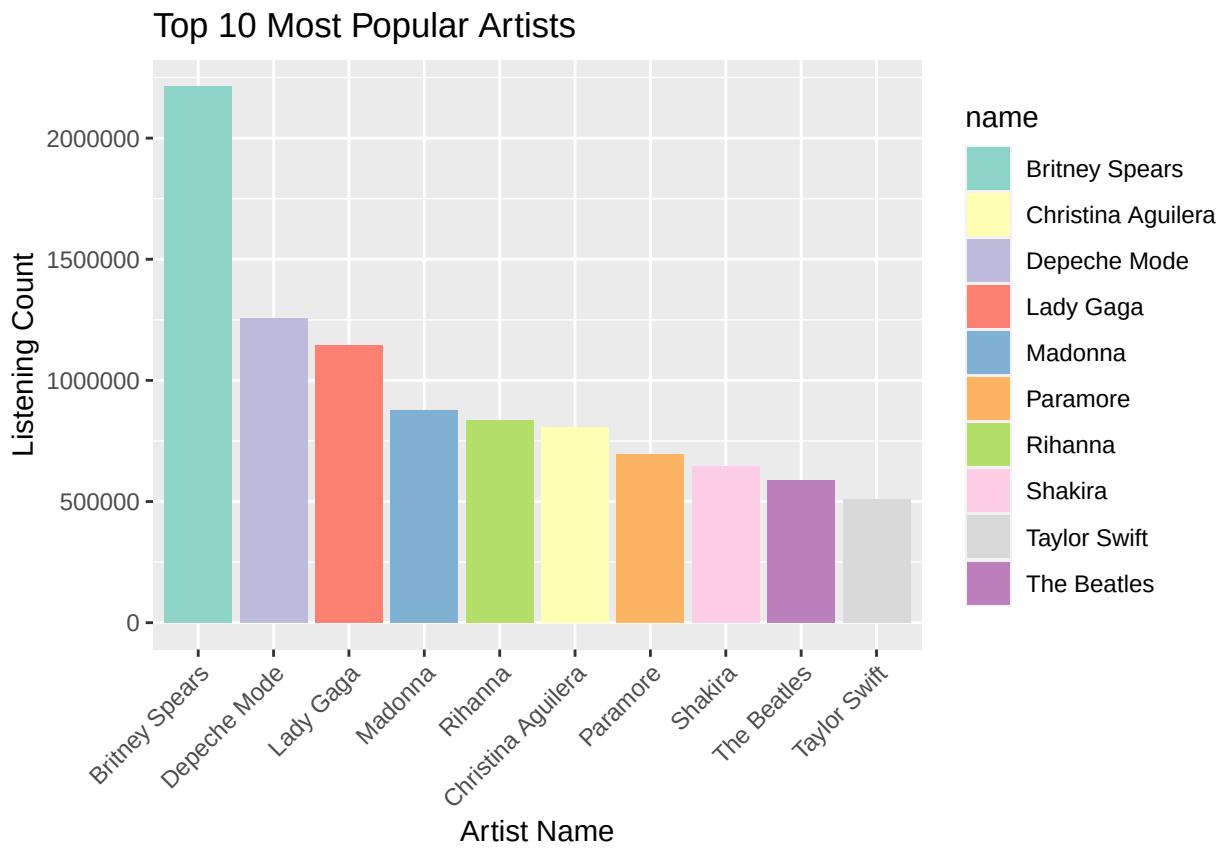



```
#Lowest 10 Users

lowest_10_users <- last.fm %>%
  group_by(userID) %>%
  summarize(total_songs_listened = sum(as.numeric(weight), na.rm = TRUE)) %>%
  arrange(total_songs_listened) %>%
  head(10)

lowest_10_users$userID <- as.factor(lowest_10_users$userID)

ggplot(lowest_10_users, aes(x = reorder(userID, total_songs_listened), y = total_songs_listened, fill =
  geom_bar(stat = "identity") +
  scale_fill_viridis_d() +  # Use the viridis color palette
  labs(title = "Lowest 10 Users with Fewest Songs Listened",
       x = "User ID",
       y = "Total Songs Listened") +
  theme(axis.text.x = element_blank())
```



Top 10 Most Popular Artists based on User Listener count

```
#V1
pop_artists <- last.fm %>%
  group_by(name) %>%
  summarize(listening_count = sum(as.numeric(weight))) %>%
  arrange(desc(listening_count))

top_10 <- head(pop_artists, 10)
top_20 <- head(pop_artists,20)

#Top 10
ggplot(top_10, aes(x = reorder(name, -listening_count), y = listening_count, fill = name)) +
  geom_bar(stat = "identity") +
  labs(title = "Top 10 Most Popular Artists",
       x = "Artist Name",
       y = "Listening Count") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_fill_brewer(palette = "Set3")
```




```
wordcloud(words = top_10$name, freq = top_10$listening_count, scale=c(3,0.5),
          colors=brewer.pal(8, "Set3"), max.words=10,
          random.order=FALSE, rot.per=0.35,
          main="Top 10 Most Popular Artists")
```

Models

Prep data for recommender system. Data will need to be condensed in order to provide more accurate recommendations. In order to reduce skews and outliers, the data will be truncated in order to ensure that useres that haven't had at least 20 listens will not be included.

```
#need to make the dataset wide
```

MOdel 1

```
# Calculate the mean of weight
RMSE <- function(true_weight, predicted_weight){
  sqrt(mean((true_weight - predicted_weight)^2))
}

mu_hat <- mean(last.fm_train$weight)
mu_hat
```

```
## [1] 736.7717
```

```
naive_rmse <- RMSE(last.fm_test$weight, mu_hat)
naive_rmse
```

```
## [1] 2788.506
```

```
predictions <- rep(2.5, nrow(last.fm_test))
RMSE(last.fm_test$weight, predictions)
```

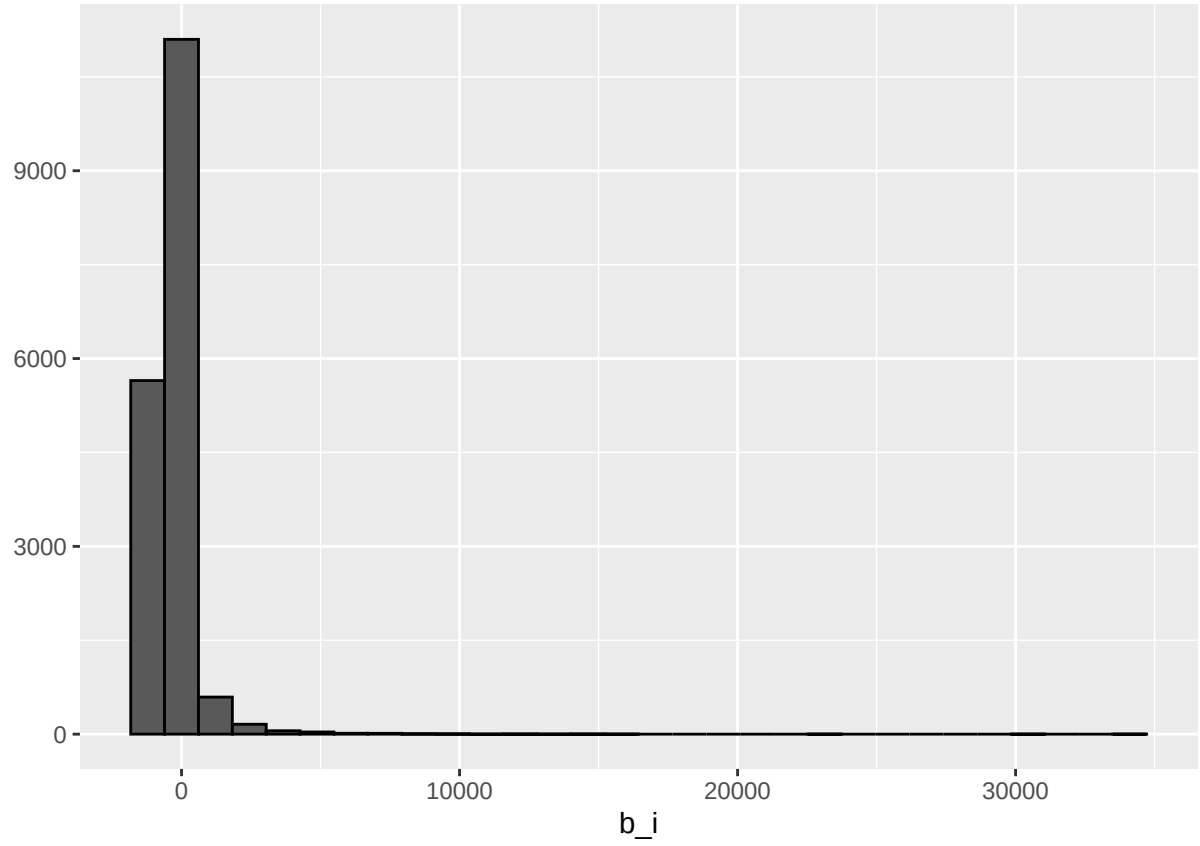
```
## [1] 2890.257
```

```
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 2789.
```

Model 2

```
mu <- mean(last.fm_train$weight)
listen_avgs <- last.fm_train %>%
  group_by(artistID) %>%
  summarize(b_i = mean(weight - mu))


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

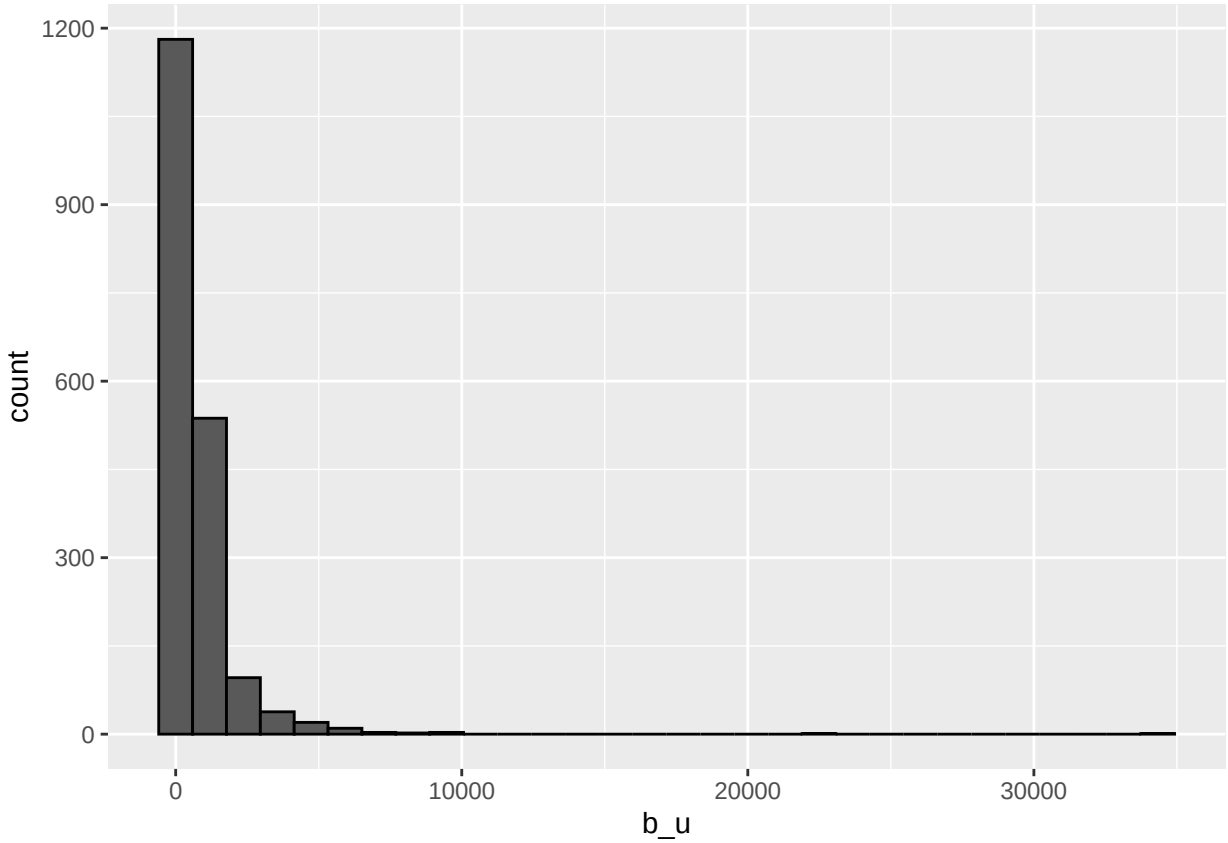



```
predicted_ratings <- mu + last.fm_test %>%
  left_join(listen_avgs, by='artistID') %>%
  pull(b_i)

model_1_rmse <- RMSE(predicted_ratings, last.fm_test$weight)
rmse_results <- bind_rows(rmse_results,
                          tibble(method="Listen Effect Model",
                                 RMSE = model_1_rmse ))
rmse_results %>% knitr::kable()
```

method	RMSE
Just the average	2788.506
Listen Effect Model	2776.217

Model 3

```
last.fm_train %>%
  group_by(userID) %>%
  summarize(b_u = mean(weight)) %>%
  filter(n()>=100) %>%
  ggplot(aes(b_u)) +
  geom_histogram(bins = 30, color = "black")
```




```
# lm(rating ~ as.factor(movieId) + as.factor(userId))
user_avgs <- last.fm_train %>%
  left_join(listen_avgs, by='artistID') %>%
  group_by(userID) %>%
  summarize(b_u = mean(weight - mu - b_i))

predicted_ratings <- last.fm_test %>%
  left_join(listen_avgs, by='artistID') %>%
  left_join(user_avgs, by='userID') %>%
  mutate(pred = mu + b_i + b_u) %>%
  pull(pred)

model_2_rmse <- RMSE(predicted_ratings, last.fm_test$weight)

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

method	RMSE
Just the average	2788.506
Listen Effect Model	2776.217
Artist + Listen Effects Model	2708.438

Model 4

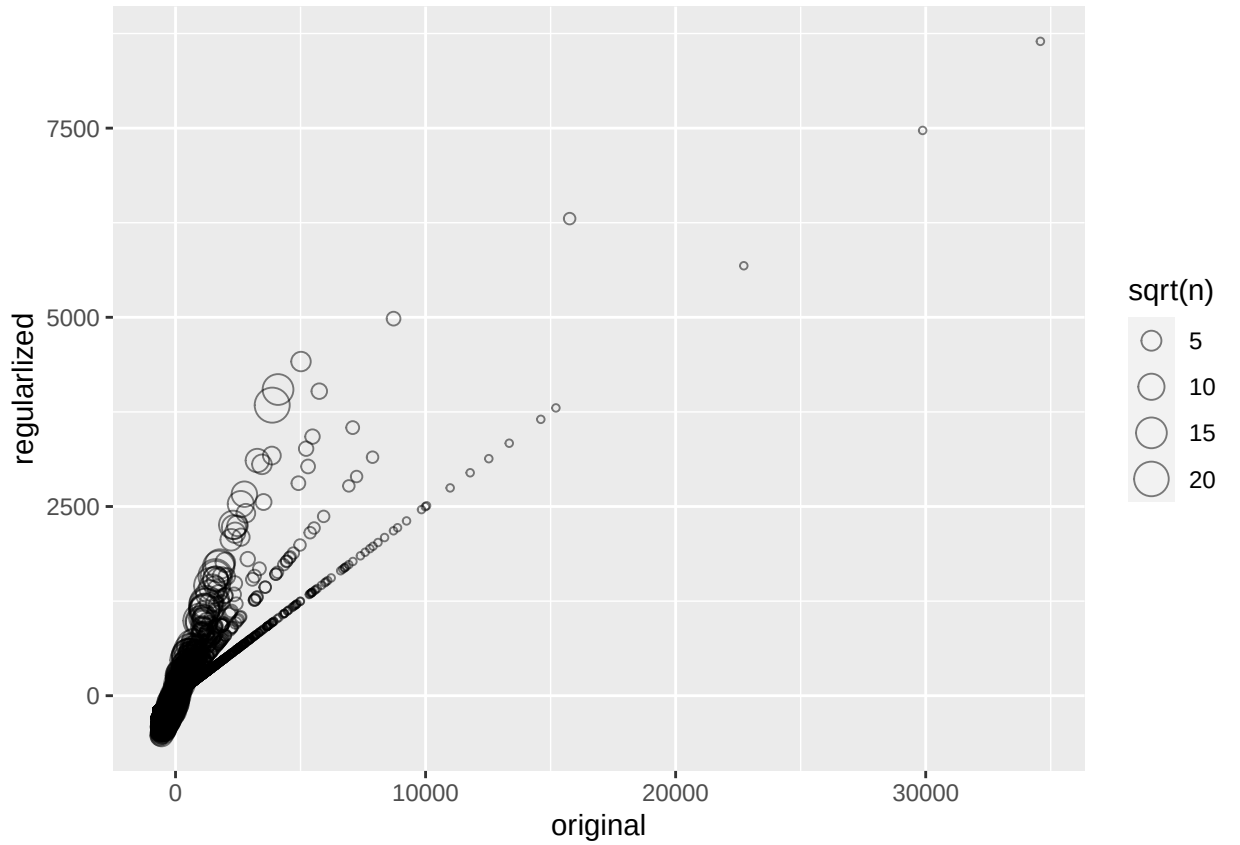
```
last.fm_test %>%
  left_join(listen_avgs, by='artistID') %>%
  mutate(residual = weight - (mu + b_i)) %>%
  arrange(desc(abs(residual))) %>%
  dplyr::select(name,  residual) %>% slice(1:10) %>% pull(name)
```

```
##  [1] "Britney Spears"     "Taylor Swift"       "Depeche Mode"
##  [4] "Christina Aguilera" "Kylie Minogue"      "Rihanna"
##  [7] "Rihanna"            "The Beatles"        "Rihanna"
## [10] "Black Eyed Peas"
```

```
artist_names <- last.fm %>%
  dplyr::select(artistID, name) %>%
  distinct()

last.fm_train %>% count(artistID) %>%
  left_join(listen_avgs) %>%
  left_join(artist_names, by="artistID") %>%
  arrange(desc(b_i)) %>%
  slice(1:10) %>%
  pull(n)
```

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

```
lambda <- 3
mu <- mean(last.fm_train$weight)
listen_reg_avgs <- last.fm_train %>%
  group_by(artistID) %>%
  summarize(b_i = sum(weight - mu)/(n()+lambda), n_i = n())

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




```
last.fm_train %>%
  count(artistID) %>%
  left_join(listen_reg_avgs, by="artistID") %>%
  left_join(artist_names, by="artistID") %>%
  arrange(desc(b_i)) %>%
  dplyr::select(name, b_i, n) %>%
  slice(1:10) %>%
  pull(name)
```

```
##  [1] "Viking Quest"              "Tyler Adam"
##  [3] "Johnny Hallyday"           "Rytmus"
##  [5] "Bushido"                   "Sarah Brightman"
##  [7] "Depeche Mode"              "Tangerine Dream"
##  [9] "Britney Spears"            "DICKY DIXON LAKE RECORDS"
```

```
last.fm_train %>%
  dplyr::count(artistID) %>%
  left_join(listen_reg_avgs, by="artistID") %>%
  left_join(artist_names, by="artistID") %>%
  arrange(b_i) %>%
  dplyr::select(name, b_i, n) %>%
  slice(1:10) %>%
  pull(name)
```

```
##  [1] "[unknown]"                 "Emily Osment"
```

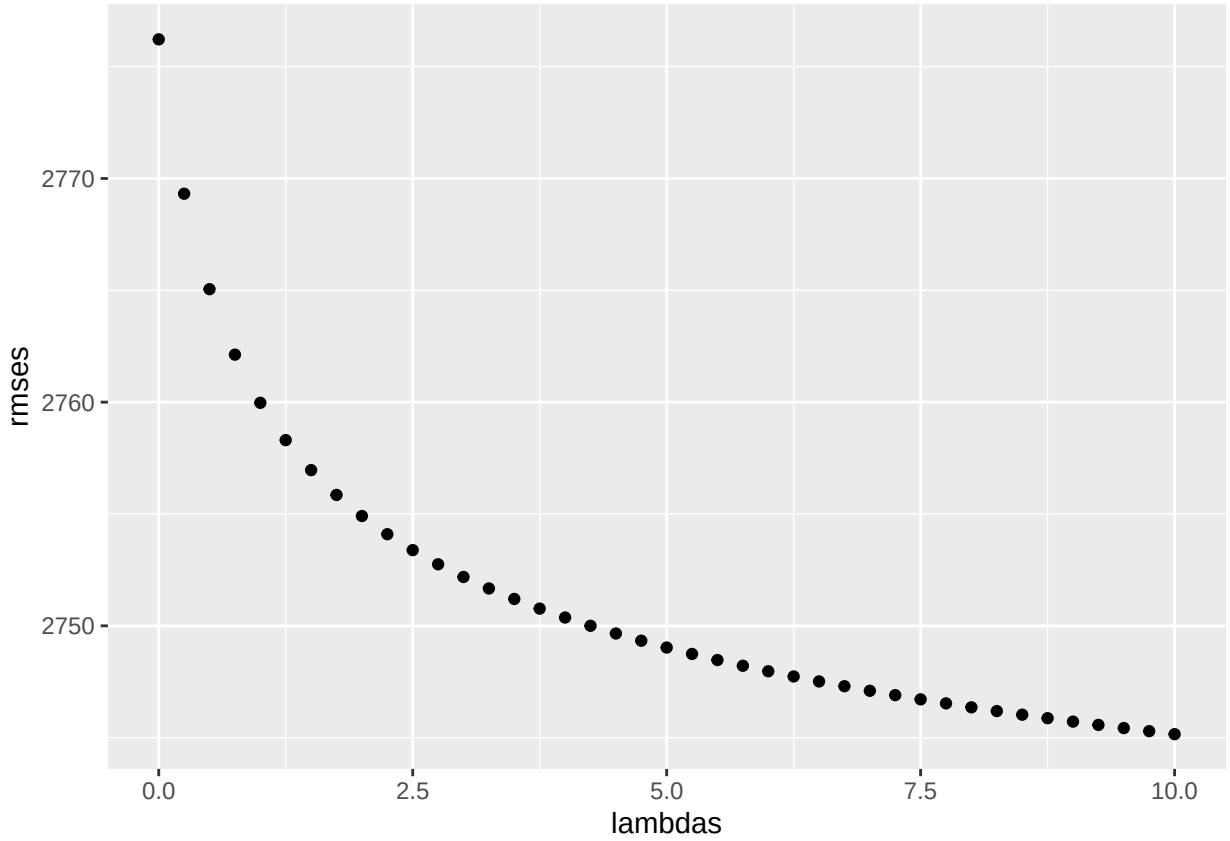
```
##  [3] "Billy Idol"                "Slash"
##  [5] "Kansas"                    "Selena Gomez & Demi Lovato"
##  [7] "Blue Öyster Cult"          "Lenny Kravitz"
##  [9] "Journey"                   "Sean Paul"
```

```
predicted_ratings <- last.fm_test %>%
  left_join(listen_reg_avgs, by='artistID') %>%
  mutate(pred = mu + b_i) %>%
  pull(pred)

model_3_rmse <- RMSE(predicted_ratings, last.fm_test$weight)
rmse_results <- bind_rows(rmse_results,
                          tibble(method="Regularized Listener Effect Model",
                                 RMSE = model_3_rmse ))
rmse_results %>% knitr::kable()
```

method	RMSE
Just the average	2788.506
Listen Effect Model	2776.217
Artist + Listen Effects Model	2708.438
Regularized Listener Effect Model	2752.185

Model 5

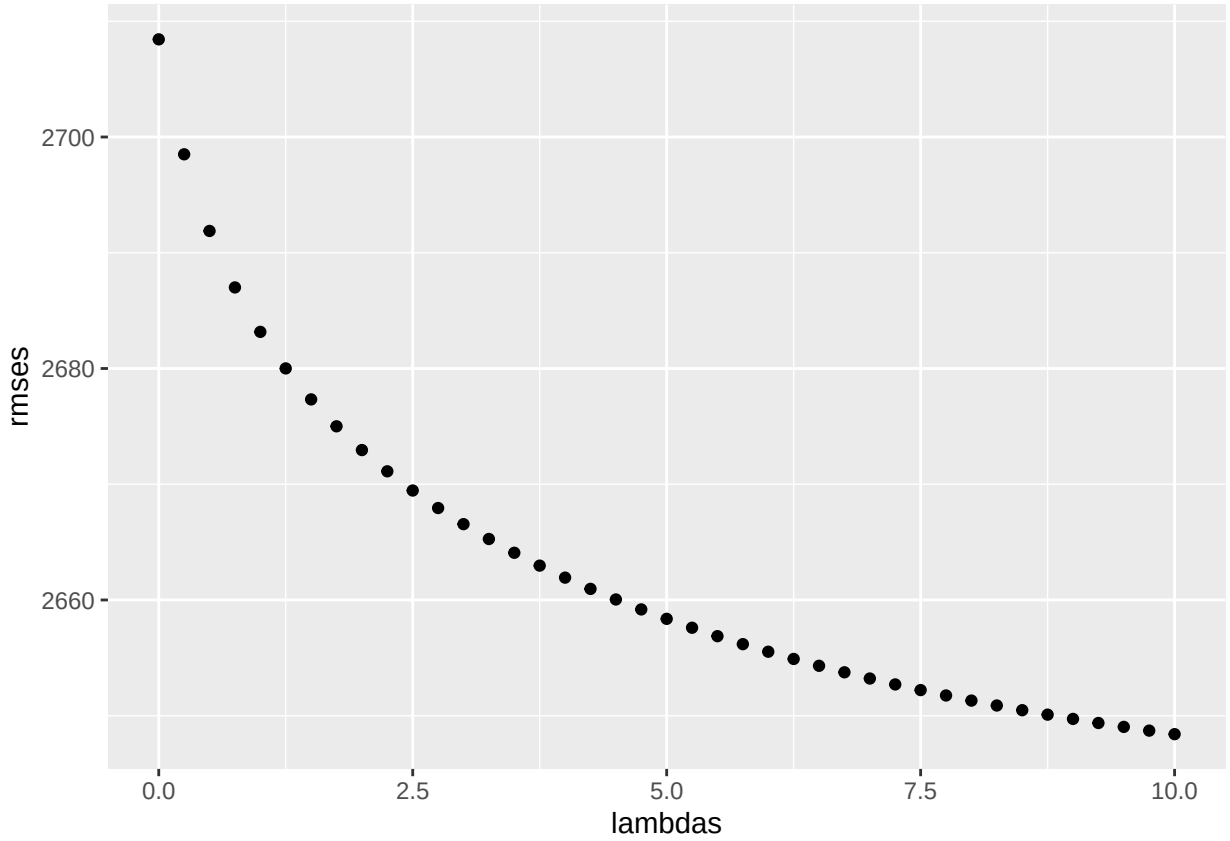
```
lambdas <- seq(0, 10, 0.25)
mu <- mean(last.fm_train$weight)
just_the_sum <- last.fm_train %>%
  group_by(artistID) %>%
  summarize(s = sum(weight - mu), n_i = n())
rmses <- sapply(lambdas, function(l){
  predicted_ratings <- last.fm_test %>%
    left_join(just_the_sum, by='artistID') %>%
    mutate(b_i = s/(n_i+l)) %>%
    mutate(pred = mu + b_i) %>%
    pull(pred)
  return(RMSE(predicted_ratings, last.fm_test$weight))
})
qplot(lambdas, rmses)
```




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

```
## [1] 10
```

```
lambdas <- seq(0, 10, 0.25)
rmses <- sapply(lambdas, function(l){
  mu <- mean(last.fm_train$weight)
  b_i <- last.fm_train %>%
    group_by(artistID) %>%
    summarize(b_i = sum(weight - mu)/(n()+l))
  b_u <- last.fm_train %>%
    left_join(b_i, by="artistID") %>%
    group_by(userID) %>%
    summarize(b_u = sum(weight - b_i - mu)/(n()+l))
  predicted_ratings <-
    last.fm_test %>%
    left_join(b_i, by = "artistID") %>%
    left_join(b_u, by = "userID") %>%
    mutate(pred = mu + b_i + b_u) %>%
    pull(pred)
  return(RMSE(predicted_ratings, last.fm_test$weight))
})

qplot(lambdas, rmses)
```




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

```
## [1] 10
```

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

method	RMSE
Just the average	2788.506
Listen Effect Model	2776.217
Artist + Listen Effects Model	2708.438
Regularized Listener Effect Model	2752.185
Regularized Listener + User Effect Model	2648.407

Evaluation

```
mu <- mean(final_holdout_test$weight)
l <- 0.15
```

```r
b_i <- final_holdout_test %>%
  group_by(artistID) %>%
  summarize(b_i = sum(weight - mu)/(n() + l))

b_u <- final_holdout_test %>%
  left_join(b_i, by='artistID') %>%
  group_by(userID) %>%
  summarize(b_u = sum(weight - b_i - mu)/(n() +l))

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

RMSE(predicted, final_holdout_test$weight)
```

```
## [1] 4459.602
```

Conclusion

Based on the rmse results it's clear that there will need to be more work done in order to have the training model that we would like to have.A deeper dive should be take into the last.fm dataset that was created and cleaned. It likely needs to be truncated as well in order to produce the recommendations we would like to have for this particular case.

JST HarvardX Data Science Capstone Project: LastFM Recommender System