MovieLens Project: Harvardx: PH125.9x Data Science

Frederico de Almeida Meirelles Palma

January 07, 2020

1 - Project Overview

The project is to create a movie recommendation system using MovieLens dataset. It's one of the Capstone exercise from HarvardX: PH125.9x Data Science course.

2 - Introduction

The objective is to train a machine learning algorithm using the inputs in one subset (`edx`) to predict movie ratings in the validation set (`validation`). The provided data is a 65.6MB zip file with around 10M from MovieLens dataset collected by the GroupLens research lab, Department of Computer Science and Engineering at the University of Minnesota and it's available in this url: http://files.grouplens.org/datasets/movielens/ml-10m.zip. Running the code below, the datasets are already generated enabling data analysis and exploration. After that, the process is to train a machine learning algorithm using the inputs in one subset to predict movie ratings in the validation set.

2.1 - Creating edx set and validation set

```
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
```

```r
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)

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

3 - Data Analysis

3.1 - The loss-function

The loss-function used is the Root-Mean-Squared-Error(RMSE), already gived, defined as:

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

In R we will use this function:

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

3.2 - Exploring edx dataset

The dataset was already provided in a usable format and it was splitted in 2 datasets, one for training (`edx`) with 90% and the other for validation (`validation`) with 10% of MovieLens data. The `edx` data set contains the following features:

```r
dim(edx)
```

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

9.000.055 rows and 6 columns.

```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
```

The 6 columns are : userID, movieId, rating, timestamp, title and genres.

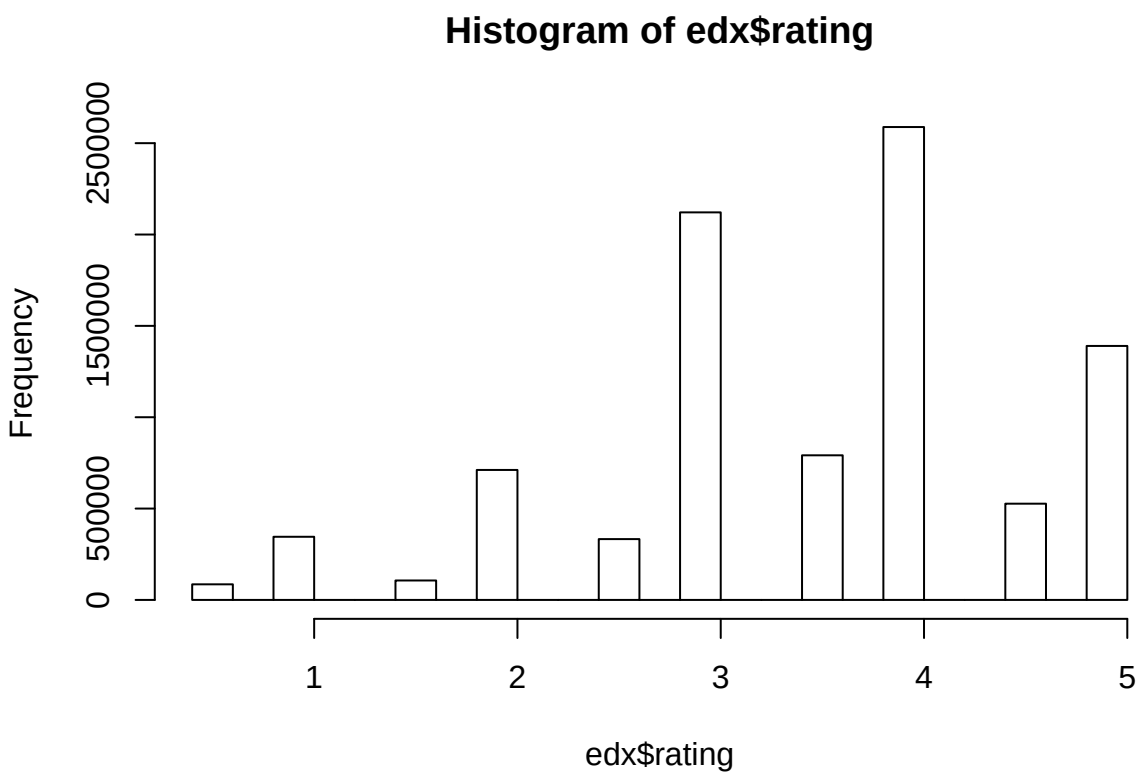
```
table(edx$rating)
```

```
##
##     0.5       1     1.5       2     2.5       3     3.5       4     4.5
##   85374  345679  106426  711422  333010 2121240  791624 2588430  526736
##       5
## 1390114
```

The rating rage is from 0.5 to 5.

The rating histogram is:

```
hist(edx$rating)
```




```
edx %>%
group_by(rating) %>%
```

```
  summarize(count = n()) %>%
  arrange(desc(count)) %>%
knitr::kable()
```

rating	count
4.0	2588430
3.0	2121240
5.0	1390114
3.5	791624
2.0	711422
4.5	526736
1.0	345679
2.5	333010
1.5	106426
0.5	85374

The top 3 ratings are: 4, 3 and 5.

Rating Summary:

```
summary(edx$rating)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.500   3.000   4.000   3.512   4.000   5.000
```

The analysis above show us that the data set doesn't have rating missing values.

Summarize distinct users and movies:

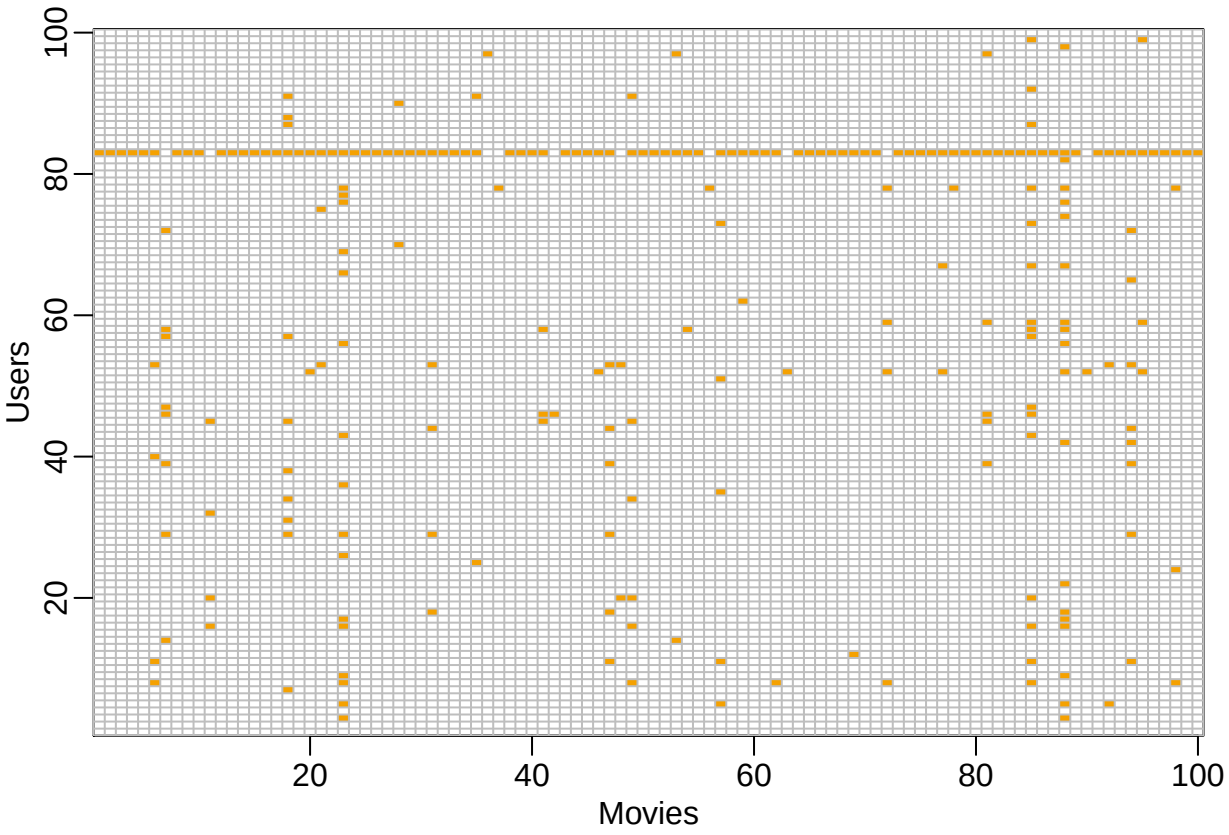
```
edx %>%
  summarize(n_users = n_distinct(userId),
            n_movies = n_distinct(movieId))
```

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

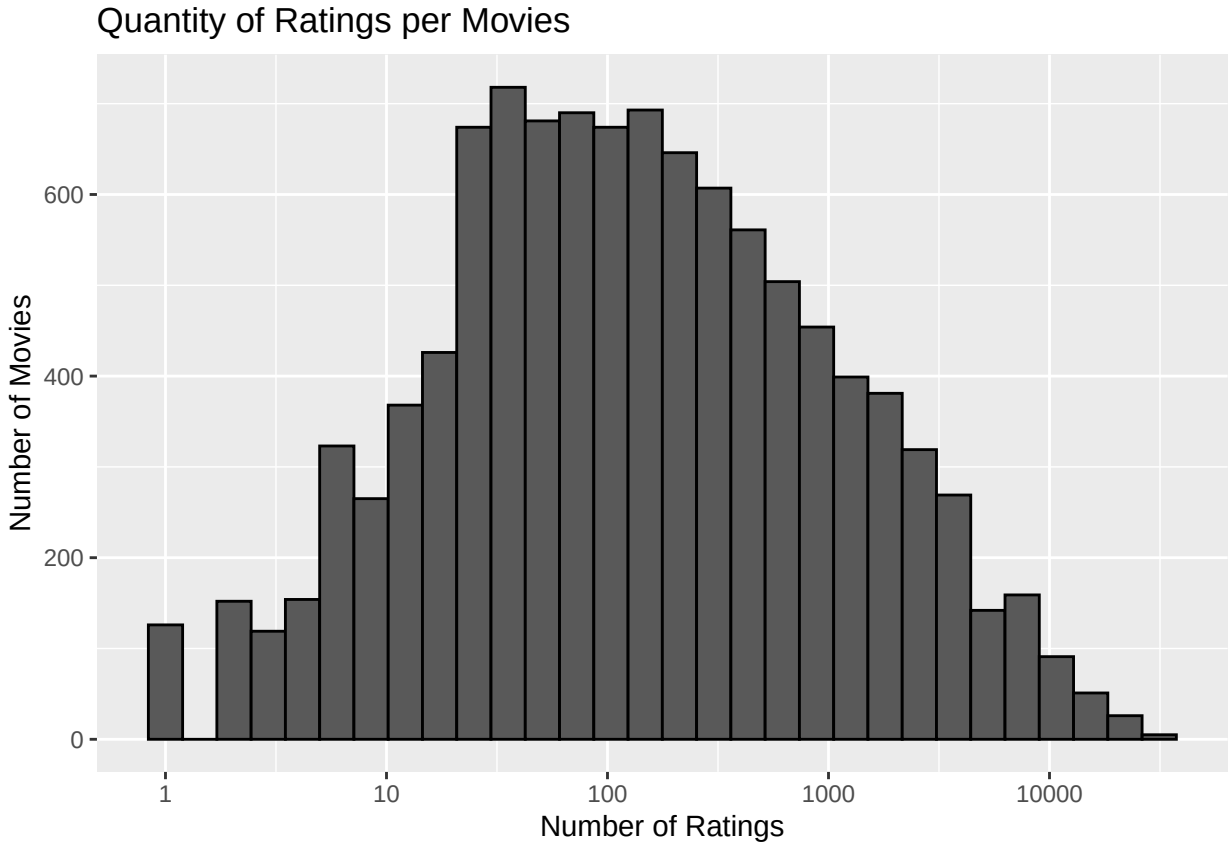
The `edx` dataset has 69.878 unique users and 10.677 unique movies.

Getting 100 users and 100 movies we can show that is a sparse matrix:

```
users <- sample(unique(edx$userId), 100)
rafalib::mypar()
edx %>% filter(userId %in% users) %>%
  select(userId, movieId, rating) %>%
  mutate(rating = 1) %>%
  spread(movieId, rating) %>% select(sample(ncol(.), 100)) %>%
  as.matrix() %>% t(.) %>%
  image(1:100, 1:100,. , xlab="Movies", ylab="Users")
abline(h=0:100+0.5, v=0:100+0.5, col = "grey")
```


Users
Movies
20
40
60
80
100

Some movies have only one rating, others few ratings and the dataset has a group that are frequently rated:



Listing movies with one rating:

```
edx %>%
group_by(title) %>%
  summarize(count = n()) %>%
  filter(count == 1) %>%
knitr::kable()
```

title	count
1, 2, 3, Sun (Un, deuz, trois, soleil) (1993)	1
100 Feet (2008)	1
4 (2005)	1
Accused (Anklaget) (2005)	1
Ace of Hearts (2008)	1
Ace of Hearts, The (1921)	1
Adios, Sabata (Indio Black, sai che ti dico: Sei un gran figlio di...) (1971)	1
Africa addio (1966)	1
Aleksandra (2007)	1
Bad Blood (Mauvais sang) (1986)	1
Battle of Russia, The (Why We Fight, 5) (1943)	1
Bellissima (1951)	1
Big Fella (1937)	1
Black Tights (1-2-3-4 ou Les Collants noirs) (1960)	1
Blind Shaft (Mang jing) (2003)	1
Blue Light, The (Das Blaue Licht) (1932)	1
Borderline (1950)	1

title	count
Brothers of the Head (2005)	1
Chapayev (1934)	1
Cold Sweat (De la part des copains) (1970)	1
Condo Painting (2000)	1
Confess (2005)	1
Confessions of a Superhero (2007)	1
Cruel Story of Youth (Seishun zankoku monogatari) (1960)	1
David Holzman's Diary (1967)	1
Dead Man's Letters (Pisma myortvogo cheloveka) (1986)	1
Deadly Companions, The (1961)	1
Death Kiss, The (1933)	1
Delgo (2008)	1
Demon Lover Diary (1980)	1
Deux mondes, Les (2007)	1
Devil's Chair, The (2006)	1
Diggers (2006)	1
Diminished Capacity (2008)	1
Dirty Dozen, The: The Fatal Mission (1988)	1
Dischord (2001)	1
Dog Day (Canicule) (1984)	1
Dog Run (1996)	1
Dogwalker, The (2002)	1
Double Dynamite (1951)	1
Down and Derby (2005)	1
Du côté de la côte (1958)	1
Emerald Cowboy (2002)	1
Face of a Fugitive (1959)	1
Fallout (1998)	1
Family Game, The (Kazoku gêmu) (1983)	1
Father Sergius (Otets Sergiy) (1917)	1
Fighting Elegy (Kenka erejii) (1966)	1
Fireproof (2008)	1
Fists in the Pocket (I Pugni in tasca) (1965)	1
Flandres (2006)	1
Flu Bird Horror (2008)	1
Fools' Parade (1971)	1
Forgotten One, The (1990)	1
Forty Shades of Blue (2005)	1
Gaucho, The (1927)	1
God's Sandbox (Tahara) (2002)	1
Gold Raiders (1951)	1
Guard Post, The (G.P. 506) (2008)	1
Hellhounds on My Trail (1999)	1
Hexed (1993)	1
Hey Hey It's Esther Blueburger (2008)	1
Hi-Line, The (1999)	1
Hundred and One Nights, A (Cent et une nuits de Simon Cinéma, Les) (1995)	1
Impulse (2008)	1
In Bed (En la cama) (2005)	1
In the Winter Dark (1998)	1
Jimmy Carter Man from Plains (2007)	1
Just an American Boy (2003)	1

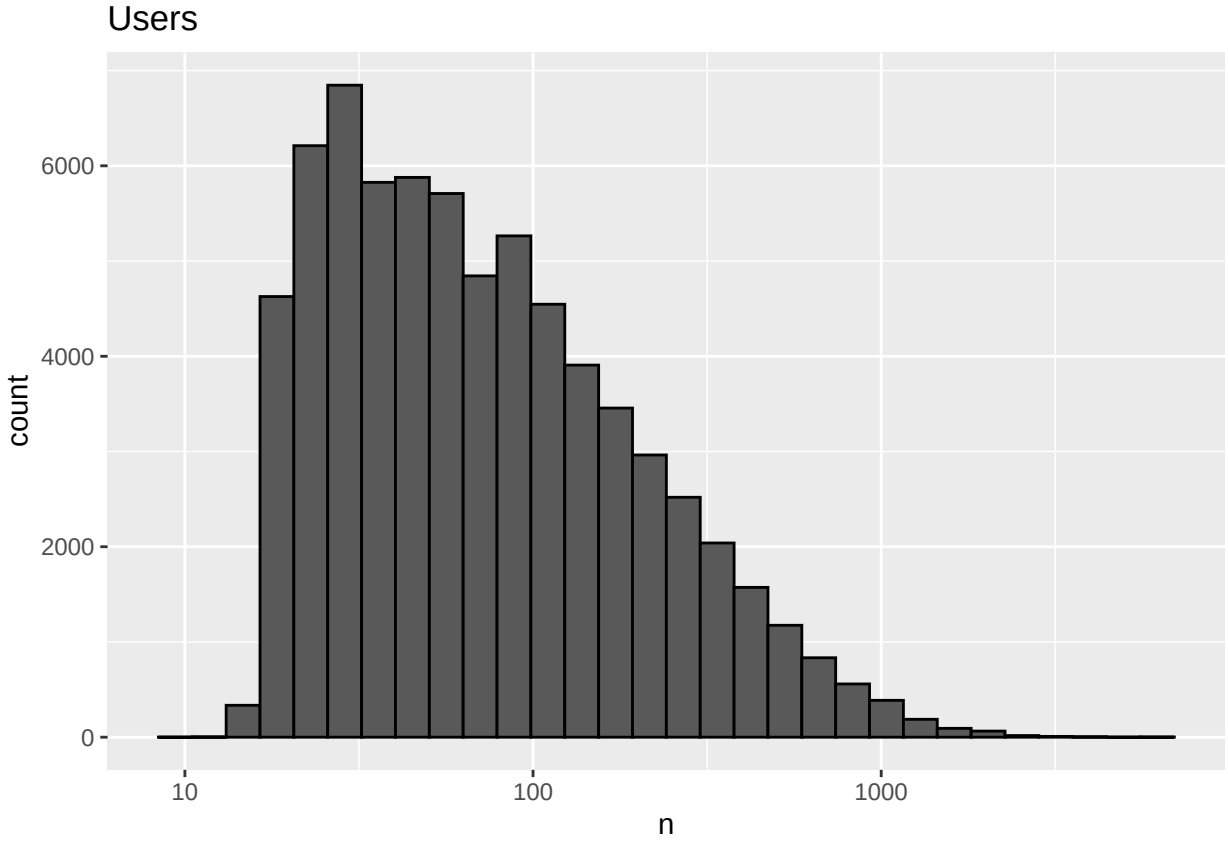
title	count
Kanak Attack (2000)	1
Kansas City Confidential (1952)	1
Krabat (2008)	1
Ladrones (2007)	1
Last Time, The (2006)	1
Lessons of Darkness (Lektionen in Finsternis) (1992)	1
Living 'til the End (2005)	1
Long Night, The (1947)	1
Love (2005)	1
Love Forbidden (Défense d'aimer) (2002)	1
Love Life (2001)	1
Mala Noche (1985)	1
Malaya (1949)	1
Man Named Pearl, A (2006)	1
Man of Straw (Untertan, Der) (1951)	1
Mesmerist, The (2002)	1
Mickey (2003)	1
Monkey's Tale, A (Les Château des singes) (1999)	1
Moonbase (1998)	1
Mr. Wu (1927)	1
Much Ado About Something (2001)	1
Music Room, The (Jalsaghar) (1958)	1
Nazis Strike, The (Why We Fight, 2) (1943)	1
Neil Young: Human Highway (1982)	1
Once in the Life (2000)	1
One Hour with You (1932)	1
Part of the Weekend Never Dies (2008)	1
Please Vote for Me (2007)	1
Quarry, The (1998)	1
Quiet City (2007)	1
Relative Strangers (2006)	1
Ring of Darkness (2004)	1
Rockin' in the Rockies (1945)	1
Säg att du älskar mig (2006)	1
Shadows of Forgotten Ancestors (1964)	1
Small Cuts (Petites coupures) (2003)	1
Splinter (2008)	1
Stacy's Knights (1982)	1
Stone Angel, The (2007)	1
Strange Planet (1999)	1
Sun Alley (Sonnenallee) (1999)	1
Sun Shines Bright, The (1953)	1
Symbiopsychotaxiplasm: Take One (1968)	1
Tattooed Life (Irezumi ichidai) (1965)	1
Testament of Orpheus, The (Testament d'Orphée) (1960)	1
Tokyo! (2008)	1
Train Ride to Hollywood (1978)	1
Twice Upon a Time (1983)	1
Uncle Nino (2003)	1
Valerie and Her Week of Wonders (Valerie a týden divu) (1970)	1
Variety Lights (Luci del varietà) (1950)	1
Vinci (2004)	1

title	count
When Time Ran Out... (a.k.a. The Day the World Ended) (1980)	1
Where A Good Man Goes (Joi gin a long) (1999)	1
Won't Anybody Listen? (2000)	1
Young Unknowns, The (2000)	1
Zona Zamfirova (2002)	1

126 movies have only 1 rating, the predictions of future ratings for them will be difficult.

Some users are more active than others at rating:

```
edx %>%
  dplyr::count(userId) %>%
  ggplot(aes(n)) +
  geom_histogram(bins = 30, color = "black") +
  scale_x_log10() +
  ggtitle("Users")
```



4 - Model Development

The RMSE will be our measure of model accuracy. RMSE is commonly used in movie rating prediction algorithms. Results larger than 1 means that the error is larger than one star, which is not a good result.

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

N is the number of user/movie combinations and the sum occurring over all these combinations. As shown the RMSE function in R is:

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

Lower results, means better predictions.

To create a recomendation system model, we will start with a simplest model called Average Rating Model that assumes the same rating for all movies and all users, as learned from the course, if μ represents the true rating for all movies and users and $\epsilon_{u,i}$ represents independent errors sampled from the same distribution centered at zero, then we have:

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

The least squares estimate of μ is the average rating of all movies across all users. After that, the model will be improved adding a term, b_i that represents the Average Rating for Movie i, the Movie Effect, as follow:

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

Then, we will improve adding the User Effect, b_u, to model:

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

After that, to improve the results, the regularization will be applied. Regularization constrains the total variability of the effect sizes by penalizing large estimates that come from small sample sizes such as an obscure film with only a few very low ratings. As showed in the course, we can select the bias values using a regularization factor λ as follows:

$$\hat{b}_i(\lambda) = \frac{1}{\lambda + n_i} \sum_{i=1}^{n_i} (Y_{i,u,g} - \hat{\mu})$$

$$\hat{b}_u(\lambda) = \frac{1}{\lambda + n_u} \sum_{u=1}^{n_u} \left(Y_{i,u,g} - \hat{b}_i - \hat{\mu}\right)$$

$$\hat{b}_g(\lambda) = \frac{1}{\lambda + n_g} \sum_{g=1}^{n_g} \left(Y_{i,u,g} - \hat{b}_i - \hat{b}_u - \hat{\mu}\right)$$

4.1 - Average Rating Model

Let's create the simplest possible recommendation system using the average of all the ratings:

```
mu <- mean(edx$rating)
mu
```

```
## [1] 3.512465
```

The first naive RMSE is:

```
naive_rmse <- RMSE(validation$rating, mu)
naive_rmse
```

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

So, it's not a good result.

```
rmse_ar <- tibble(Method = "Average Rating", RMSE = naive_rmse)
knitr::kable(rmse_ar)
```

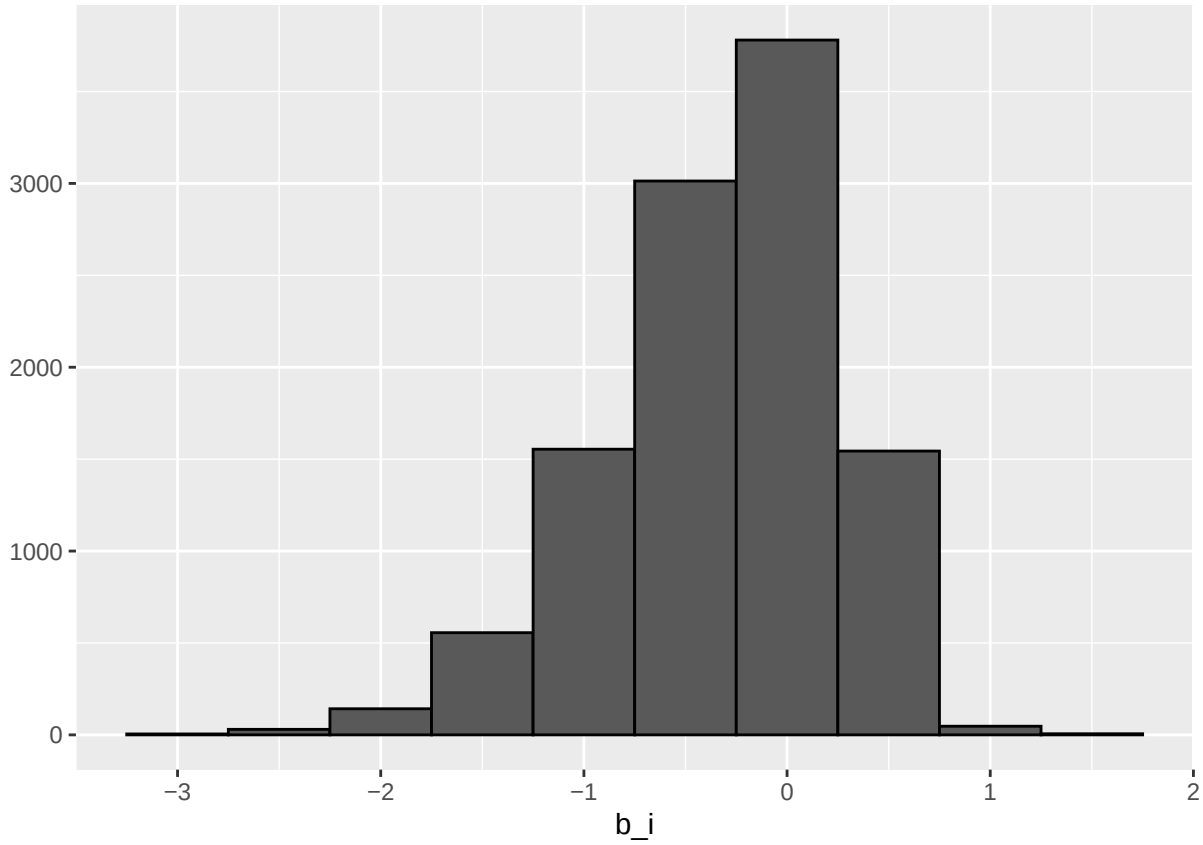
Method	RMSE
Average Rating	1.061202

Let's improve the prediction adding Movie Effect.

4.2 - Movie Effect Model

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

```
movie_avgs %>% qplot(b_i, geom ="histogram", bins = 10, data = ., color = I("black"))
```



Improving the model adding b_i.

```
predicted_ratings <- mu + validation %>%
  left_join(movie_avgs, by='movieId') %>%
  .$b_i

model_2_rmse <- RMSE(predicted_ratings, validation$rating)

rmse_me <- tibble(Method = "Movie Effect Model", RMSE = model_2_rmse)
```

```
knitr::kable(rmse_me)
```

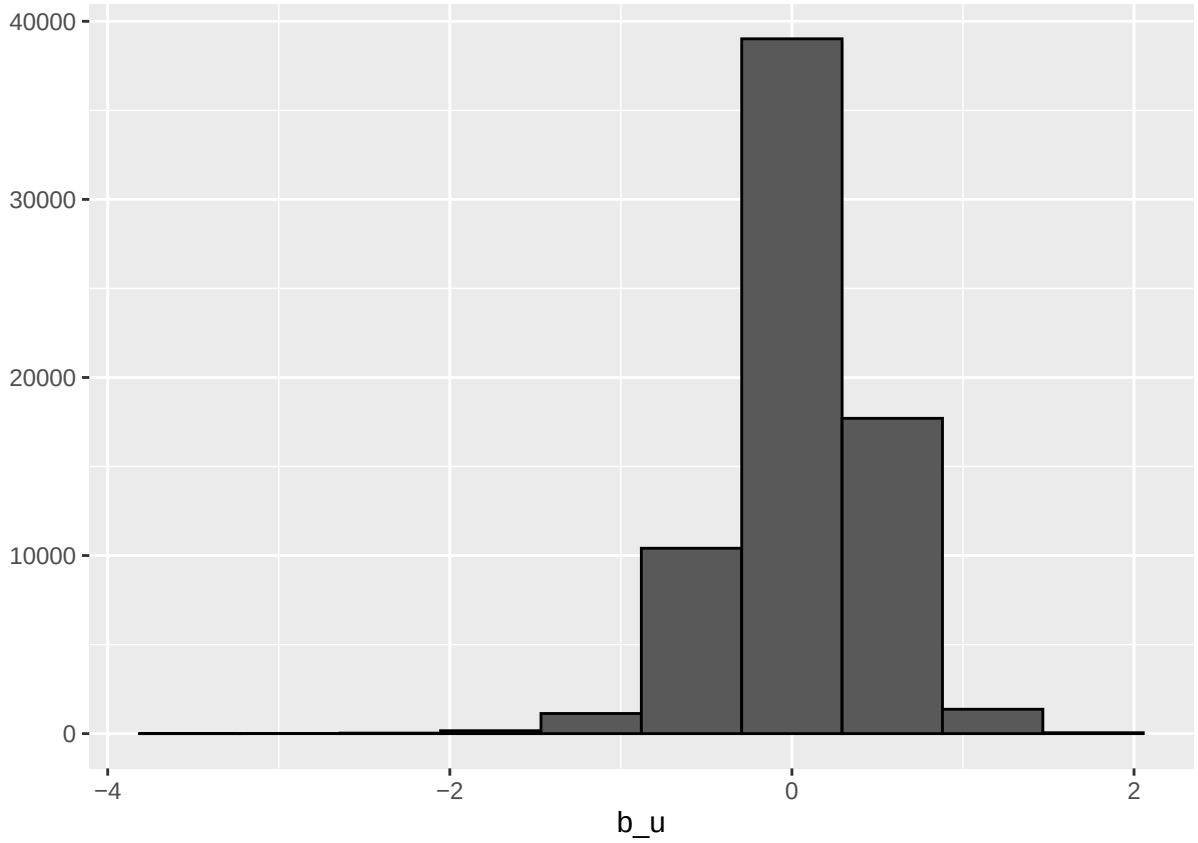
Method	RMSE
Movie Effect Model	0.9439087

Now, let's add the User Effect, b_u.

4.3 - User Effect Model

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

```
user_avgs %>% qplot(b_u, geom ="histogram", bins = 10, data = ., color = I("black"))
```



Adding User Effect, b_u:

```
predicted_ratings <- validation %>%
  left_join(movie_avgs, by='movieId') %>%
  left_join(user_avgs, by='userId') %>%
  mutate(pred = mu + b_i + b_u) %>%
  .$pred

model_3_rmse <- RMSE(predicted_ratings, validation$rating)
```

```r
rmse_ue <- tibble(Method = "Movie and User Effects Model", RMSE = model_3_rmse)

knitr::kable(rmse_ue)
```

Method	RMSE
Movie and User Effects Model	0.8653488

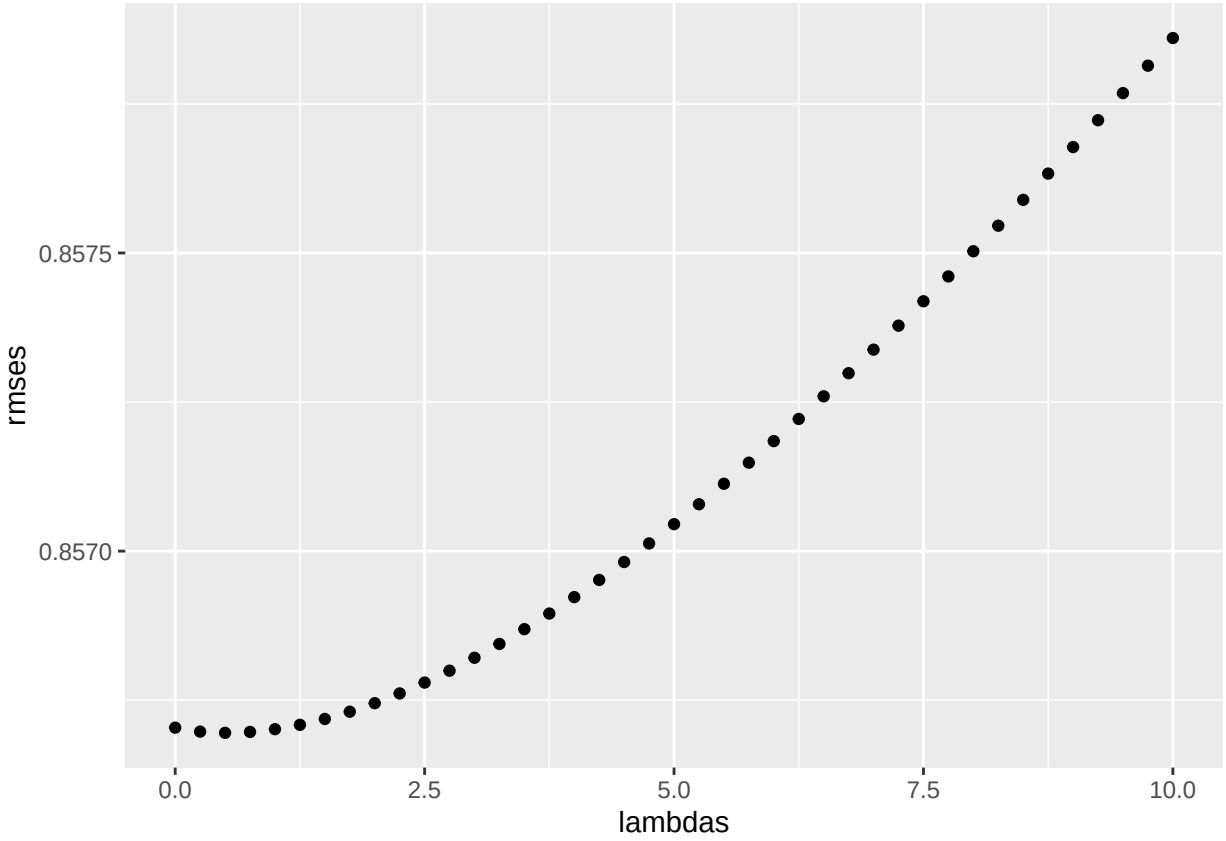
4.4 - Regularization

Now, let's apply regularization to improve the results:

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

Ploting lambdas and RMSEs:

```r
qplot(lambdas, rmses)
```



Getting the lambda that gives the less rmse:

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

```
## [1] 0.5
```

Adding regularization:

```
rmse_r <- tibble(Method = "Regularized Movie and User Effect Model", RMSE = min(rmses))
knitr::kable(rmse_r)
```

Method	RMSE
Regularized Movie and User Effect Model	0.8566952

Results

The best model for this project is the Regularized Movie and User Effect:

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

With the lowest RMSE value:

```
knitr::kable(rmse_r)
```

Method	RMSE
Regularized Movie and User Effect Model	0.8566952

5 - Conclusion

We built a machine learning algorithm to predict movie ratings with MovieLens dataset using Regularized Movie and User Effect Model. The best, lowest, RMSE value found is 0.8566952 using this model. We can affirm that RMSE value can be improved adding other effects (genre for example). Combining others machine learning algorithms could also improve the results.

6 - System Information

Follow the system information and R version:

```
## [1] "System Information:"
```

```
##                _
## platform       x86_64-pc-linux-gnu
## arch           x86_64
## os             linux-gnu
## system         x86_64, linux-gnu
## status
## major          3
## minor          6.2
## year           2019
## month          12
## day            12
## svn rev        77560
## language       R
## version.string R version 3.6.2 (2019-12-12)
## nickname       Dark and Stormy Night
```