Report on Movie Recommendation System

Jesus Barbosa

21 de enero de 2019

Movie Recommendation System

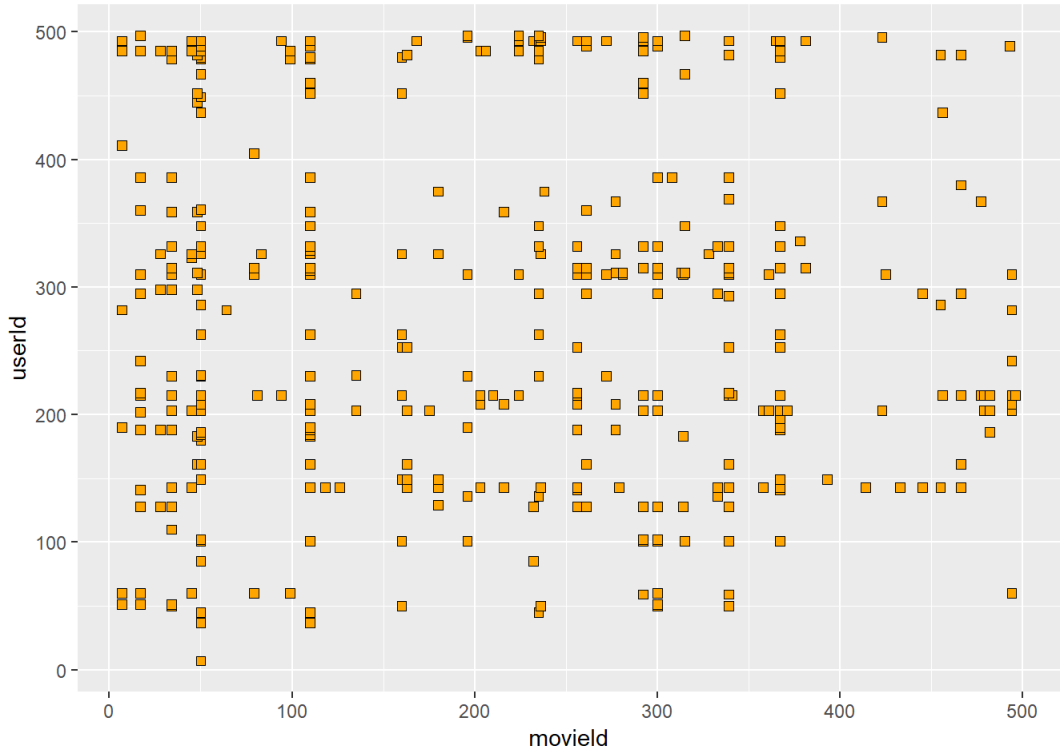
Introduction

We will try to predict the rating that a user would give to a movie, based on information from ratings given by others users

We will use the database of ratings of the laboratory of GroupLens research: https://grouplens.org/datasets/movielens/

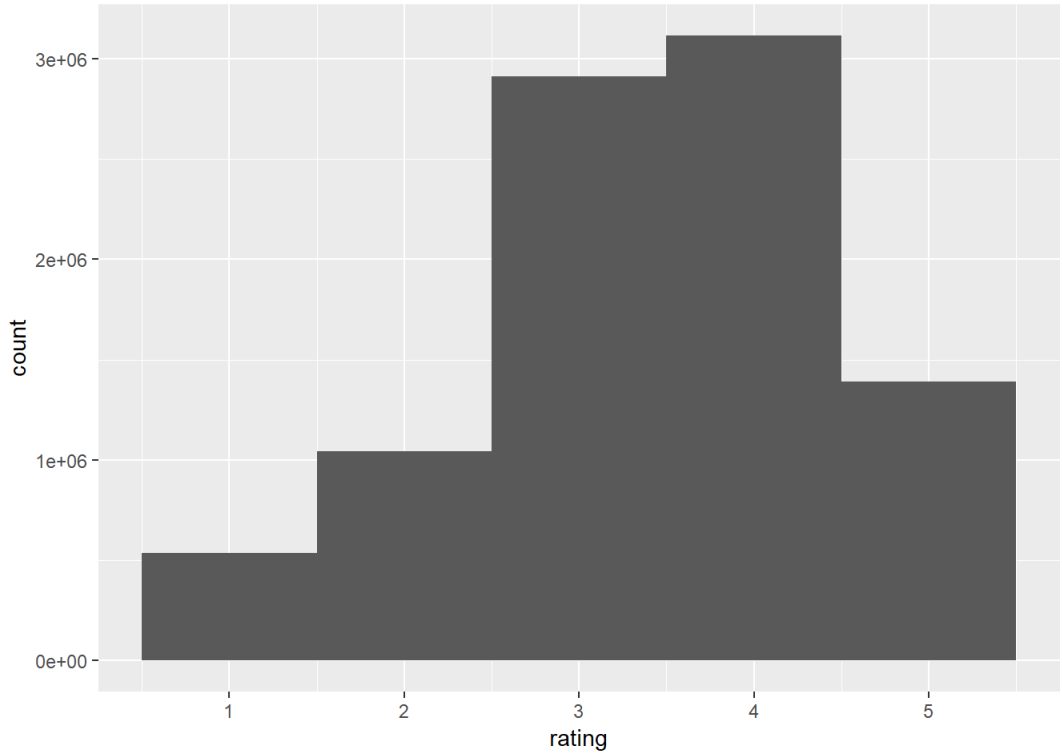
We can imagine our data as a large matrix with users in the rows and movies in the columns, with many empty squares, we can observe it here, with a sample of 100 users and 100 movies

```
set.seed(1)
u_id <- sample(500, 100)
m_id <- sample(500, 100)
edx %>% filter(userId %in% u_id & movieId %in% m_id) %>%
  ggplot(aes(movieId, userId)) +
  geom_point(shape = "square filled", fill = "orange", size = 2)
```



We can observe the distribution of ratings like this

```
edx %>% ggplot(aes(rating)) + geom_histogram(binwidth = 1)
```



We observe that the distribution of ratings approaches a normal distribution, so we can as a first model calculate the average of the ratings as our prediction

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

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

We note that the average rating is 3.51, we can validate against a set of test data, to observe the difference that we have to predict, the error RMSE

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

RMSE(mu, validation$rating)
```

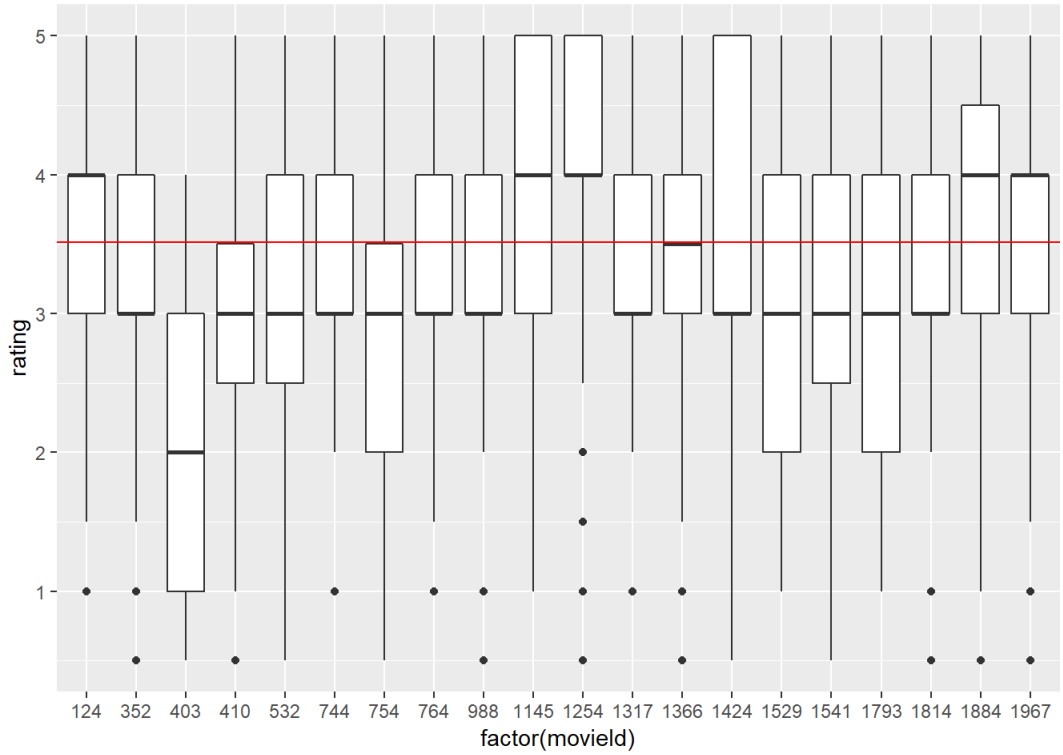
```
## [1] 1.061202
```

We note that we get an error of 1.06. Users rate from 1 to 5, so we have an error of more than 1 point

Movie effect

We know from experience that some movies are rated higher what others. We can observe it here

```
set.seed(1)
m_id <- sample(1:2000, 20)
edx %>% filter(movieId %in% m_id) %>%
  group_by(movieId) %>%
  ggplot(aes(factor(movieId), rating)) +
  geom_boxplot() +
  geom_hline(col = "red", yintercept = 3.51)
```



We observe that the films are rated more or less than the average that we calculated previously of 3.51, we call this a bias, we can calculate an average of the bias for each movie to add this effect to the model in this way

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

Now we can predict the ratings again using this new model

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

RMSE(predicted_ratings, validation$rating)
```

```
## [1] 0.9439087
```

Regularization

We see that our prediction improved, we obtained a rmse of 0.94, let's see where is where we are failing in the prediction

```
validation %>%
  left_join(movie_avgs, by = "movieId") %>%
  mutate(residual = rating - (mu + b_i)) %>%
  arrange(desc(abs(residual))) %>%
  select(title, residual) %>%
  slice(1:10) %>% knitr::kable()
```

title	residual
PokÃ©mon Heroes (2003)	3.970803
Shawshank Redemption, The (1994)	-3.955131
Shawshank Redemption, The (1994)	-3.955131
Shawshank Redemption, The (1994)	-3.955131
Godfather, The (1972)	-3.915366
Godfather, The (1972)	-3.915366
Godfather, The (1972)	-3.915366

title	residual
Usual Suspects, The (1995)	-3.865854
Usual Suspects, The (1995)	-3.865854
Usual Suspects, The (1995)	-3.865854

Let's see which are the best and worst movies according to our prediction, for this we are going to generate data of the titles of the movies

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

We observe which are the 10 best movies according to our prediction

```
movie_avgs %>% left_join(movie_titles, by = "movieId") %>%
  arrange(desc(b_i)) %>%
  select(title, b_i) %>%
  slice(1:10) %>%
  knitr::kable()
```

title	b_i
Hellhounds on My Trail (1999)	1.487535
Satan's Tango (SÃ¡tÃ¡ntangÃ³) (1994)	1.487535
Shadows of Forgotten Ancestors (1964)	1.487535
Fighting Elegy (Kenka erejii) (1966)	1.487535
Sun Alley (Sonnenallee) (1999)	1.487535
Blue Light, The (Das Blaue Licht) (1932)	1.487535
Who's Singin' Over There? (a.k.a. Who Sings Over There) (Ko to tamo peva) (1980)	1.237535
Human Condition II, The (Ningen no joken II) (1959)	1.237535
Human Condition III, The (Ningen no joken III) (1961)	1.237535
Constantine's Sword (2007)	1.237535

And we observe which are the 10 worst movies according to our prediction

```
movie_avgs %>% left_join(movie_titles, by = "movieId") %>%
  arrange(b_i) %>%
  select(title, b_i) %>%
  slice(1:10) %>%
  knitr::kable()
```

title	b_i
Besotted (2001)	-3.012465
Hi-Line, The (1999)	-3.012465
Accused (Anklaget) (2005)	-3.012465
Confessions of a Superhero (2007)	-3.012465
War of the Worlds 2: The Next Wave (2008)	-3.012465
SuperBabies: Baby Geniuses 2 (2004)	-2.717822
Hip Hop Witch, Da (2000)	-2.691037
Disaster Movie (2008)	-2.653090
From Justin to Kelly (2003)	-2.610455
Criminals (1996)	-2.512465

And we observe which are the best and worst movies according to our prediction

The best movies

```
edx %>% count(movieId) %>%
  left_join(movie_avgs) %>%
  left_join(movie_titles, by="movieId") %>%
  arrange(desc(b_i)) %>%
  select(title, b_i, n) %>%
  slice(1:10) %>%
  knitr::kable()
```

```
## Joining, by = "movieId"
```

title	b_i	n
Hellhounds on My Trail (1999)	1.487535	1
Satan’s Tango (SÃ¡tÃ¡ntangÃ³) (1994)	1.487535	2
Shadows of Forgotten Ancestors (1964)	1.487535	1
Fighting Elegy (Kenka erejii) (1966)	1.487535	1
Sun Alley (Sonnenallee) (1999)	1.487535	1
Blue Light, The (Das Blaue Licht) (1932)	1.487535	1
Who’s Singin’ Over There? (a.k.a. Who Sings Over There) (Ko to tamo peva) (1980)	1.237535	4
Human Condition II, The (Ningen no joken II) (1959)	1.237535	4
Human Condition III, The (Ningen no joken III) (1961)	1.237535	4
Constantine’s Sword (2007)	1.237535	2

The worst movies

```
edx %>% count(movieId) %>%
  left_join(movie_avgs) %>%
  left_join(movie_titles, by="movieId") %>%
  arrange(b_i) %>%
  select(title, b_i, n) %>%
  slice(1:10) %>%
  knitr::kable()
```

```
## Joining, by = "movieId"
```

title	b_i	n
Besotted (2001)	-3.012465	2
Hi-Line, The (1999)	-3.012465	1
Accused (Anklaget) (2005)	-3.012465	1
Confessions of a Superhero (2007)	-3.012465	1
War of the Worlds 2: The Next Wave (2008)	-3.012465	2
SuperBabies: Baby Geniuses 2 (2004)	-2.717822	56
Hip Hop Witch, Da (2000)	-2.691037	14
Disaster Movie (2008)	-2.653090	32
From Justin to Kelly (2003)	-2.610455	199
Criminals (1996)	-2.512465	2

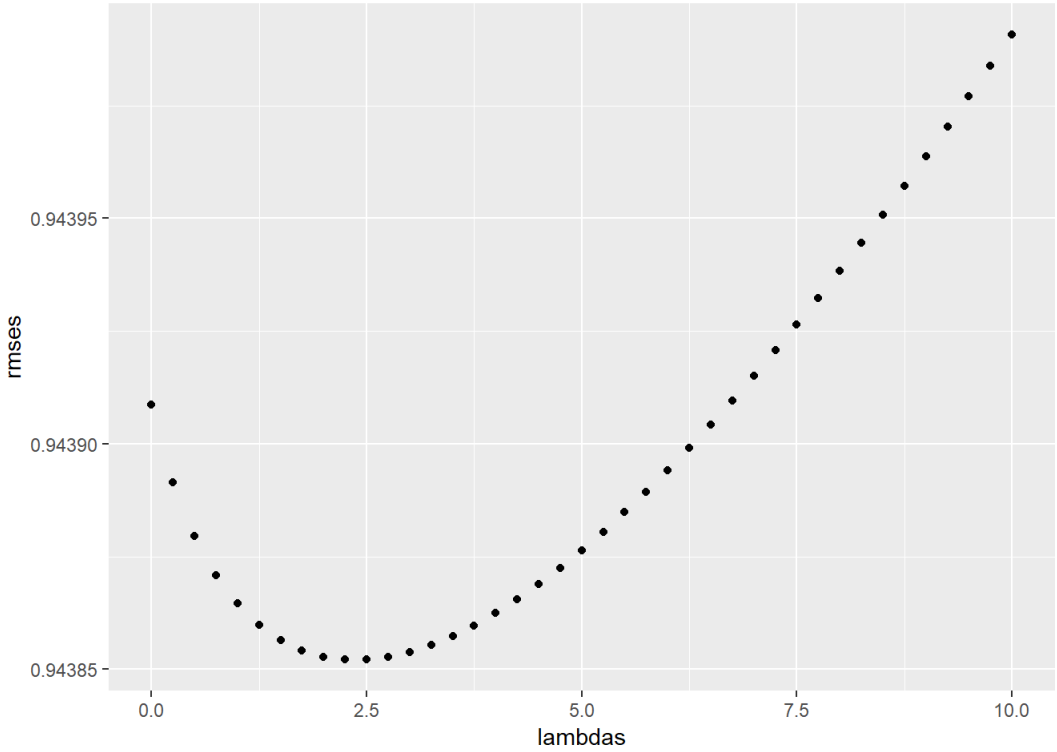
We see that they were movies rated by very few users, that makes our prediction grow or reduce a lot

We need regularization to penalize large estimates that come from small samples, we do it like that

```
lambda <- 2.5
movie_reg_avgs <- edx %>%
  group_by(movieId) %>%
  summarize(b_i = sum(rating - mu)/(n()+lambda), n_i = n())
```

We can optimize the lambda parameter, looking for the one that minimizes the rmse using cross validation

```
lambdas <- seq(0, 10, 0.25)

mu <- mean(edx$rating)
just_the_sum <- edx %>%
  group_by(movieId) %>%
  summarize(s = sum(rating - mu), n_i = n())

rmses <- sapply(lambdas, function(l){
  predicted_ratings <- validation %>%
    left_join(just_the_sum, by='movieId') %>%
    mutate(b_i = s/(n_i+l)) %>%
    mutate(pred = mu + b_i) %>%
    .$pred
  return(RMSE(predicted_ratings, validation$rating))
})
qplot(lambdas, rmses)
```




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

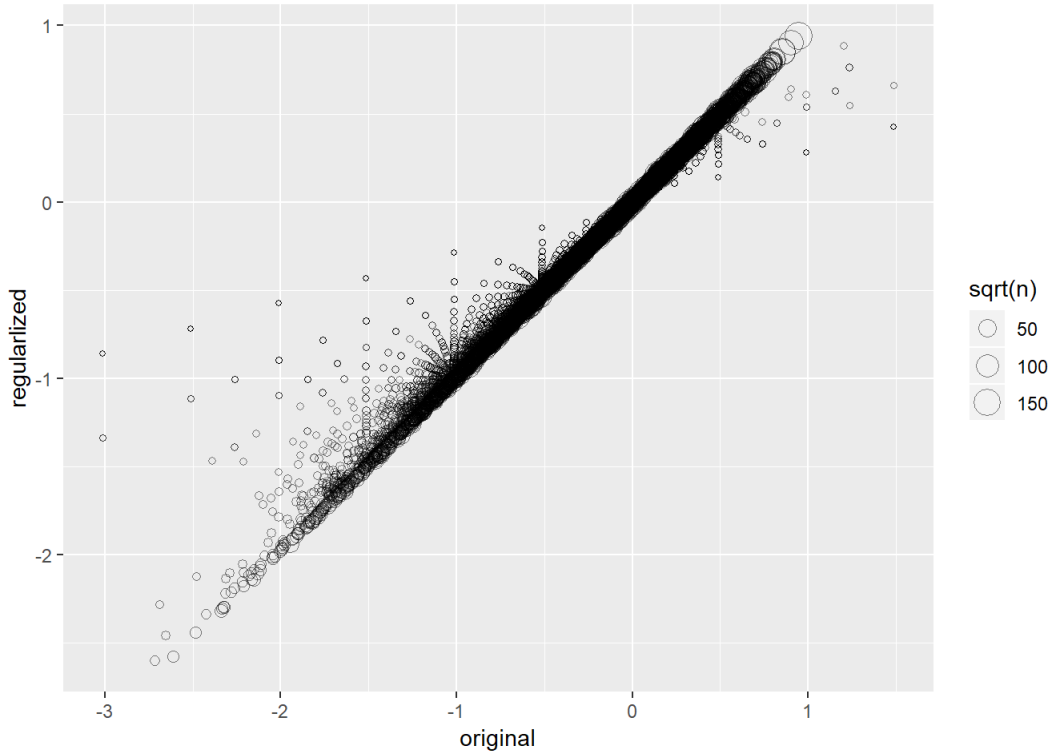
```
## [1] 2.5
```

We see that the lambda that minimice the rmse, we recalculate the averages of movies regularized with the optimized lamda

```
lambda <- lambdas[which.min(rmses)]
movie_reg_avgs <- edx %>%
  group_by(movieId) %>%
  summarize(b_i = sum(rating - mu)/(n()+lambda), n_i = n())
```

We can see how the estimates were reduced by plotting against the previous estimates

```
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)
```



Now we can see the best and worst movies already regularized

Best movies already regularized

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

```
## Joining, by = "movieId"
```

title	b_i	n
Shawshank Redemption, The (1994)	0.9425819	28015
Godfather, The (1972)	0.9027736	17747
More (1998)	0.8855520	7
Usual Suspects, The (1995)	0.8532899	21648
Schindler's List (1993)	0.8509364	23193
Casablanca (1942)	0.8077788	11232
Rear Window (1954)	0.8059324	7935
Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)	0.8027275	2922
Third Man, The (1949)	0.7982878	2967
Double Indemnity (1944)	0.7974264	2154

Worst movies already regularized

```
edx %>%
  count(movieId) %>%
  left_join(movie_reg_avgs) %>%
  left_join(movie_titles, by="movieId") %>%
  arrange(b_i) %>%
  select(title, b_i, n) %>%
  slice(1:10) %>%
  knitr::kable()
```

```
## Joining, by = "movieId"
```

title	b_i	n
SuperBabies: Baby Geniuses 2 (2004)	-2.601676	56
From Justin to Kelly (2003)	-2.578067	199
Disaster Movie (2008)	-2.460837	32
PokÃ©mon Heroes (2003)	-2.438765	137
Carnosaur 3: Primal Species (1996)	-2.338264	68
Glitter (2001)	-2.319841	339
Pokemon 4 Ever (a.k.a. PokÃ©mon 4: The Movie) (2002)	-2.305711	202
Gigli (2003)	-2.300797	313
Barney's Great Adventure (1998)	-2.297353	208
Hip Hop Witch, Da (2000)	-2.283304	14

We add regularization to the model and predict again

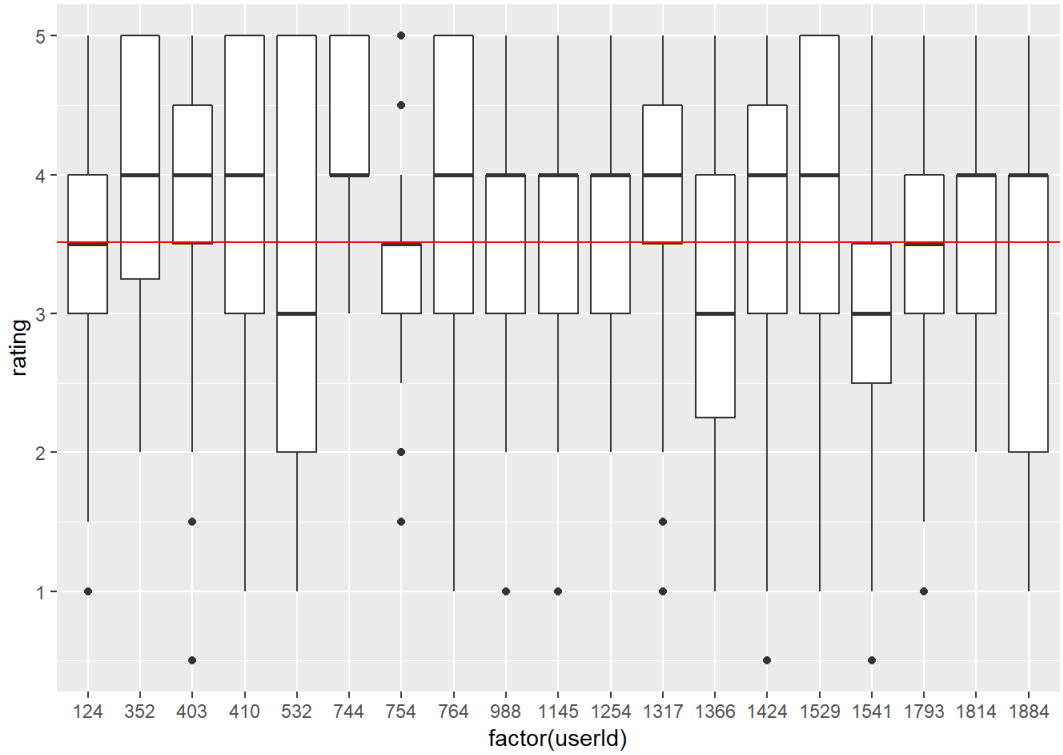
```
predicted_ratings <- validation %>%
  left_join(movie_reg_avgs, by = "movieId") %>%
  mutate(pred = mu + b_i) %>%
  .$pred

RMSE(predicted_ratings, validation$rating)
```

```
## [1] 0.9438521
```

User effect

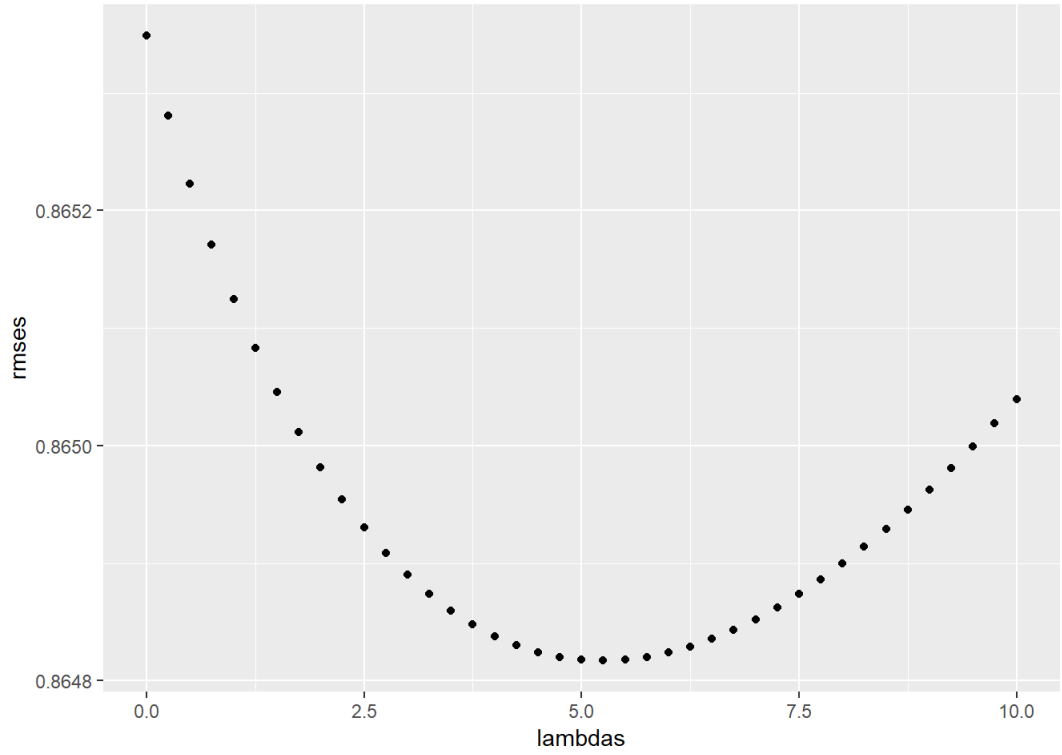
Now we see if there is also a user effect, we also know from experience that some users qualify more than others, we can see it from a sample of 20 users

```
set.seed(1)
u_id <- sample(1:2000, 20)
edx %>% filter(userId %in% u_id) %>%
  group_by(userId) %>%
  ggplot(aes(factor(userId), rating)) +
  geom_boxplot() +
  geom_hline(col = "red", yintercept = 3.51)
```



We can add this user effect and optimizing the parameter lambda in the following way

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

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

qplot(lambdas, rmses)
```




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

```
## [1] 5.25
```

Results

With this last model we obtain a rmse of 0.86!

```
min(rmses)
```

```
## [1] 0.864817
```

Conclusion

We finish with a simple model but with good results. We can improve the model by adding clustering to find patterns in the predictors and improve the prediction