Movielens - HarvardX PH125.9x Capstone Assignment

Pandiyarajan A

24 May 2019

Movielens recommendation system.

Movielens Data Overview

This data set consists of *9,000,055 ratings* from *69,878 users* on *10,677 movies*. We will now analyse the data and build a recommendation system with low RMSE value.

users	movies	ratings
69878	10677	9000055

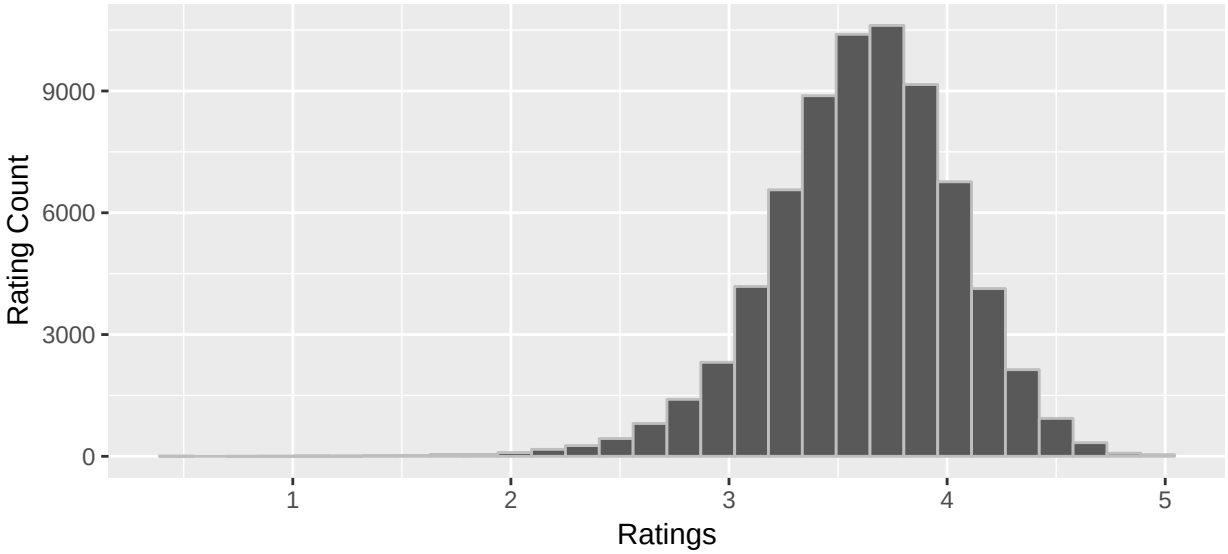
Approaches

- Data Exploration
- Data Preparation for Analysis
- Define Loss Fucntion
- Checking for Bias and Approximating
- Regularization

Data Exploration

As the data set we have is well formated and tidy, we may not require data scubbing / data cleaning. lets now dive in and start exploring the data and lets see if we can make any insigths that could help us model a better system.

Rating Distribution

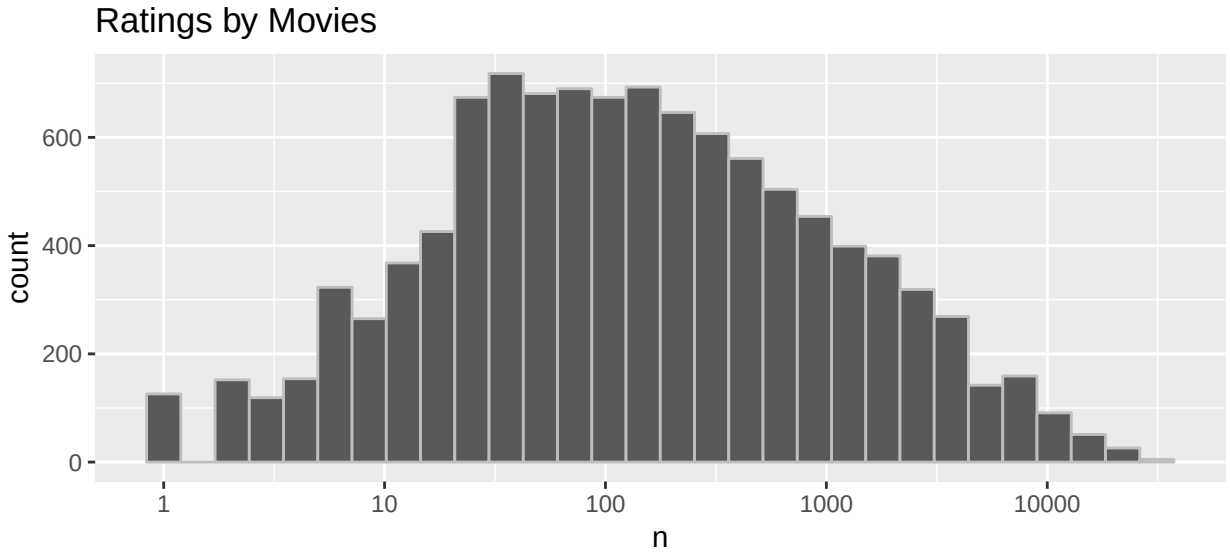


We could see the most of the movies are rated around approximatly *3.5*. However we could get decide between median, mean and mode based on the problem we are solving.

mean	median	mode
3.512465	4	4

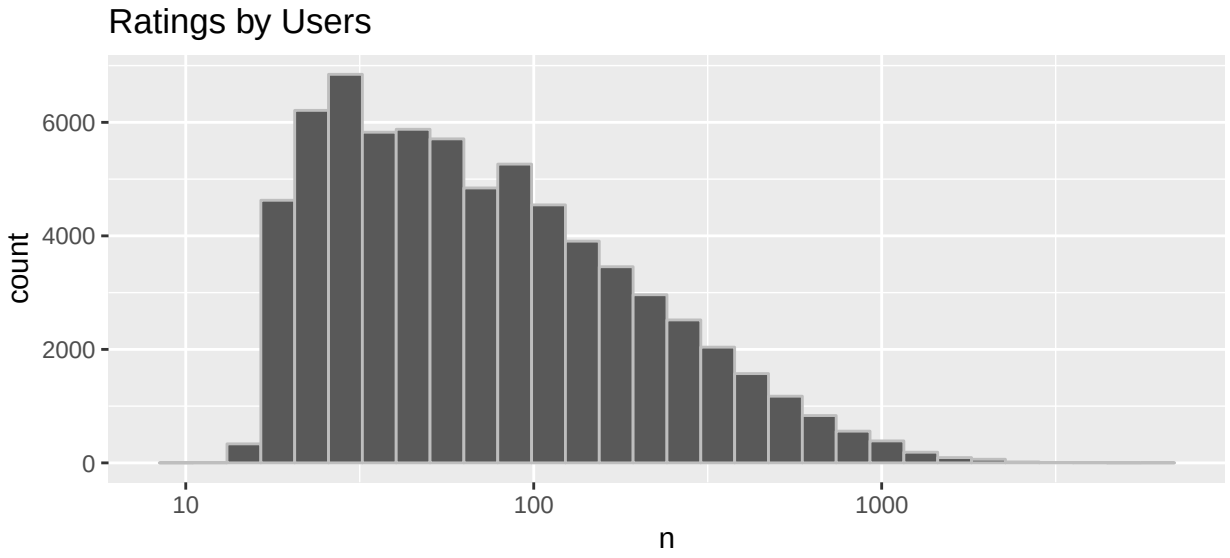
As we see with our data set the median, mode are equal and mean is a near by value. Moving further I will use the mean value for futher calculations.

Ratings by Movies



We see movies are rated more/less number of times than the others. This may be because of the popularity factor.

Ratings by Users



Here we observe the user rating behaviour. Users rate more/less number of movies than the other users.

Loss Function

We will be using Root mean squared error RMSE as our loss function as per instruction.

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

Data Preparation for Analysis

For further analysis lets split the data into **train** and **test** sets. We use ***createDataPartition*** method to do the job. We take *75% for training* and *25% for testing.*

```
set.seed(2019)

test_index <-
  createDataPartition(
    y = edx$rating,
    times = 1,
    p = 0.25,
    list = FALSE
  )

edx_test <- edx[test_index,]   # test set 25% of edx
edx_train <- edx[-test_index,] # train set 75% of edx

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

Checking for Bias and Approximating

We estimate the movie effect and user effect by taking the average of the residuals obtained after removing the overall mean and the movie effect from the ratings.

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

predicted_ratings_muem <-  edx_test %>%
  left_join(movie_avgs,'movieId') %>%
  left_join(user_avgs, 'userId') %>% mutate(pred = mu_train + b_i + b_u) %>% .$pred

predicted_ratings_muem[is.na(predicted_ratings_muem)] <- 0
```

By fixing the biasing the calculated **RMSE** is

```
## [1] 0.8661754
```

With our estimates lets find the best *10 movies.*

title	b_i	n
Hellhounds on My Trail (1999)	1.487535	1
Jack-O (1995)	1.487535	1
Satan's Tango (SÃ¡tÃ¡ntangÃ³) (1994)	1.487535	2
Emmanuel's Gift (2005)	1.487535	1
Shadows of Forgotten Ancestors (1964)	1.487535	1

title	b_i	n
Fighting Elegy (Kenka erejii) (1966)	1.487535	1
Angus, Thongs and Perfect Snogging (2008)	1.487535	1
Hospital (1970)	1.487535	1
Who's Singin' Over There? (a.k.a. Who Sings Over There) (Ko to tamo peva) (1980)	1.237535	4
Money (Argent, L') (1983)	1.237535	2

With our estimates lets find the Worst *10 movies*.

title	b_i	n
Besotted (2001)	-3.012465	2
Manslaughter (1922)	-3.012465	1
Hi-Line, The (1999)	-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)	-2.658807	41
Hip Hop Witch, Da (2000)	-2.603374	11
From Justin to Kelly (2003)	-2.590387	154
Disaster Movie (2008)	-2.554132	24
Criminals (1996)	-2.512465	1

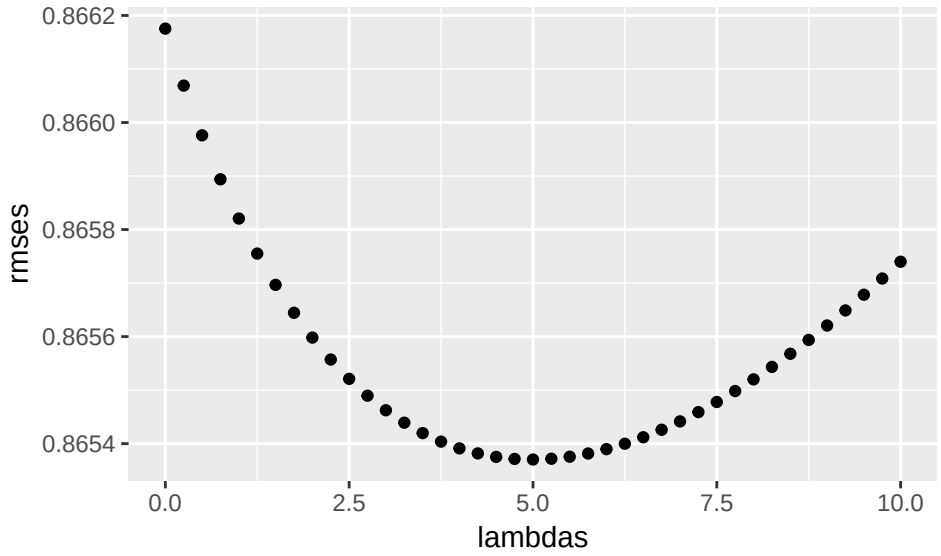
These results are not convincing as we know these are not the best and worst 10 movies from our knowledge on movies. These are because we could see that these are rated very few time and it happens that they have been rated very high and low.

To over come this issue lets regularize the data. This could help us improve reduce the RMSE value.

Regularization

Identifying the best regularization parameter.

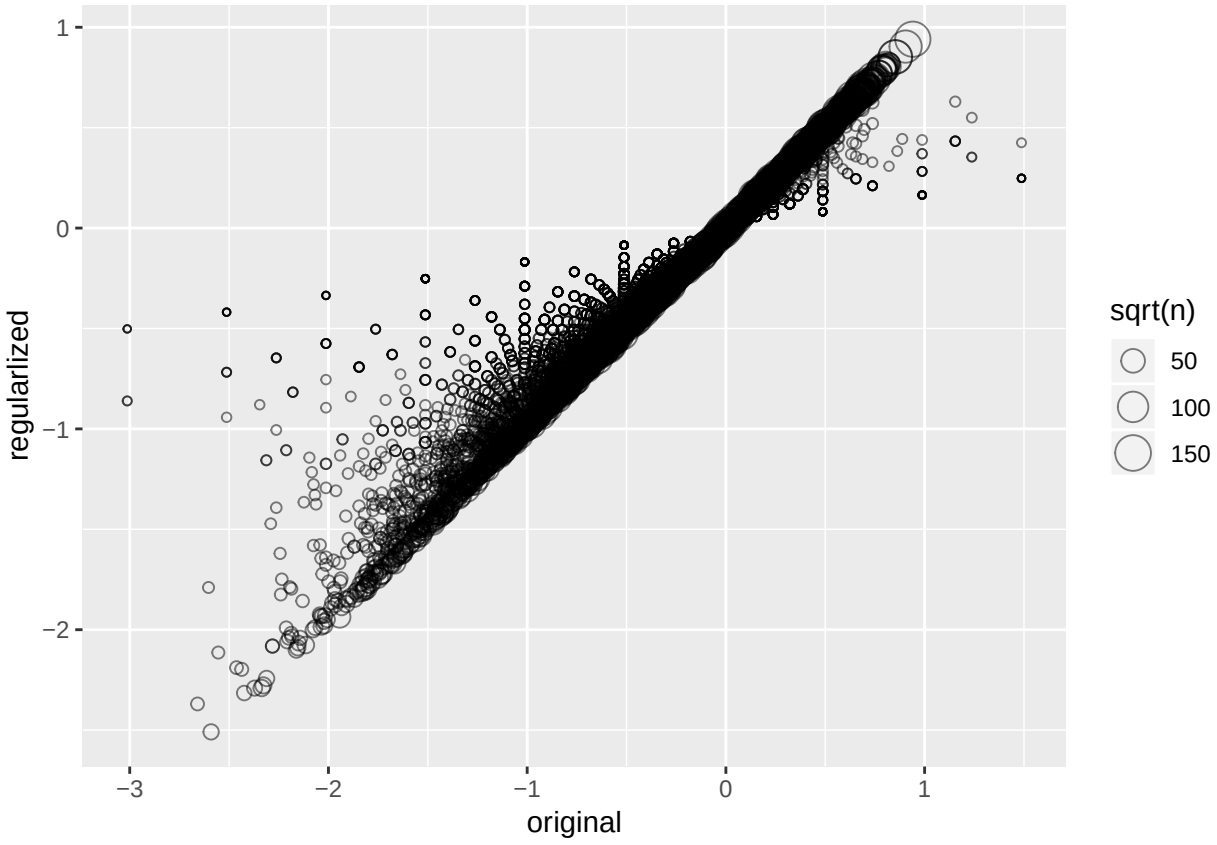
Estimating the best normalization value (lambda)



Optimal lambda

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

Effect of Regularization



Result

The best lowest error value (**RMSE**) obtained after fixing the user, movie bias and regularization is

```
## [1] 0.8653704
```

We have reduced the RMSE to a great extent.

With our updated estimates lets find the best *10 movies*.

title	b_i	n
Shawshank Redemption, The (1994)	0.9405545	21077
Godfather, The (1972)	0.9038817	13353
Schindler's List (1993)	0.8521169	17219
Usual Suspects, The (1995)	0.8519715	16260
Paths of Glory (1957)	0.8180748	1166
Third Man, The (1949)	0.8152656	2215
Double Indemnity (1944)	0.8099716	1580
Casablanca (1942)	0.8050986	8499
Lives of Others, The (Das Leben der Anderen) (2006)	0.8002627	817
Rear Window (1954)	0.7980954	5957

With our updated estimates lets find the Worst *10 movies*.

title	b_i	n
From Justin to Kelly (2003)	-2.508929	154
SuperBabies: Baby Geniuses 2 (2004)	-2.369806	41
PokÃ©mon Heroes (2003)	-2.315480	107
Pokemon 4 Ever (a.k.a. PokÃ©mon 4: The Movie) (2002)	-2.290953	142
Glitter (2001)	-2.290433	252
Gigli (2003)	-2.277283	227
Barney's Great Adventure (1998)	-2.242866	164
Carnosaur 3: Primal Species (1996)	-2.197518	46
Faces of Death: Fact or Fiction? (1999)	-2.188858	40
Disaster Movie (2008)	-2.113764	24

This is much better and convincing.

Conclusion

I hope that I have provided all the necessary details to make this report understandable and computationally reproducible. Thanks for taking the time to review my work and best wishes to you with Data Science!
