# Movie Recommendation

*Jasper Linke*

*6/17/2019*

## Introduction

This project uses a linear regression machine learning algorithm to predict movie ratings. The final algorithm bases its prediction on user and movie averages in ratings.

The data set is based on an actual Netflix coding challenge to improve the movie recommendation algorithm of the company. The present data set was prepared for download by HavardX. The data and code can be downloaded from this Git hub repository: https://github.com/jprlink/movie-recom.git

This brief report will outline the methods and steps of analysis and will conclude by discussing the findings.

## Methods and Analysis

The following packages are needed for the analysis:

```
library(tidyverse)
```

```
## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
```

```
## -- Attaching packages ------------------------------------------------------------------------ tidyve
```

```
## v ggplot2 3.1.1       v purrr   0.3.2
## v tibble  2.1.1       v dplyr   0.8.0.1
## v tidyr   0.8.3       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0
```

```
## -- Conflicts --------------------------------------------------------------------------- tidyverse_co
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

The test ("validation") and training ("edx") data is loaded. As the training set is too large to be uploaded to Git hub, please download it from edx and change the file reference according to your file system if you want to run the code.

Before further analysis, the data was checked for missing values. There are none.

The training set includes 999,999 rows and six variables: user IDs, movie IDs, time stamps, movie titles and genres. Movies are most commonly rated a 4, followed by a 3 and a 5. The data set includes 10,677 movies and 69878 Netflix users. Pulp Fiction and Forest Gump are the most frequently rated movies, while The

Shawshank Redemption and The Godfather are, on average, the most highly rated movies (taking only into account movies with at least 100 ratings).

```
## 'data.frame':    9000055 obs. of  6 variables:
##  $ userId   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ movieId  : num  122 185 292 316 329 355 356 362 364 370 ...
##  $ rating   : num  5 5 5 5 5 5 5 5 5 5 ...
##  $ timestamp: int  838985046 838983525 838983421 838983392 838983392 838984474 838983653 838984885 83
##  $ title    : chr  "Boomerang (1992)" "Net, The (1995)" "Outbreak (1995)" "Stargate (1994)" ...
##  $ genres   : chr  "Comedy|Romance" "Action|Crime|Thriller" "Action|Drama|Sci-Fi|Thriller" "Action|A
```

| | userId | movieId | rating | timestamp | title | genres |
|---|---|---|---|---|---|---|
| 1 | 1 | 122 | 5 | 838985046 | Boomerang (1992) | Comedy\|Romance |
| 2 | 1 | 185 | 5 | 838983525 | Net, The (1995) | Action\|Crime\|Thriller |
| 4 | 1 | 292 | 5 | 838983421 | Outbreak (1995) | Action\|Drama\|Sci-Fi\|Thriller |
| 5 | 1 | 316 | 5 | 838983392 | Stargate (1994) | Action\|Adventure\|Sci-Fi |
| 6 | 1 | 329 | 5 | 838983392 | Star Trek: Generations (1994) | Action\|Adventure\|Drama\|Sci-Fi |
| 7 | 1 | 355 | 5 | 838984474 | Flintstones, The (1994) | Children\|Comedy\|Fantasy |

| rating | n |
|---|---|
| 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 |

```
## [1] 10677
```

```
## [1] 69878
```

| title | n_rating |
|---|---|
| Pulp Fiction (1994) | 31362 |
| Forrest Gump (1994) | 31079 |
| Silence of the Lambs, The (1991) | 30382 |
| Jurassic Park (1993) | 29360 |
| Shawshank Redemption, The (1994) | 28015 |
| Braveheart (1995) | 26212 |

| title | avg_rating |
|---|---|
| Shawshank Redemption, The (1994) | 4.455131 |
| Godfather, The (1972) | 4.415366 |
| Usual Suspects, The (1995) | 4.365854 |
| Schindler's List (1993) | 4.363493 |
| Casablanca (1942) | 4.320424 |
| Rear Window (1954) | 4.318651 |

A first "naive" model is built based on the mean rating. It therefore assumes the same rating for all movies and users, while all differences are explained by random variation. The model predicts movie ratings with a Residual Mean Square Error (RMSE) of 1.06.

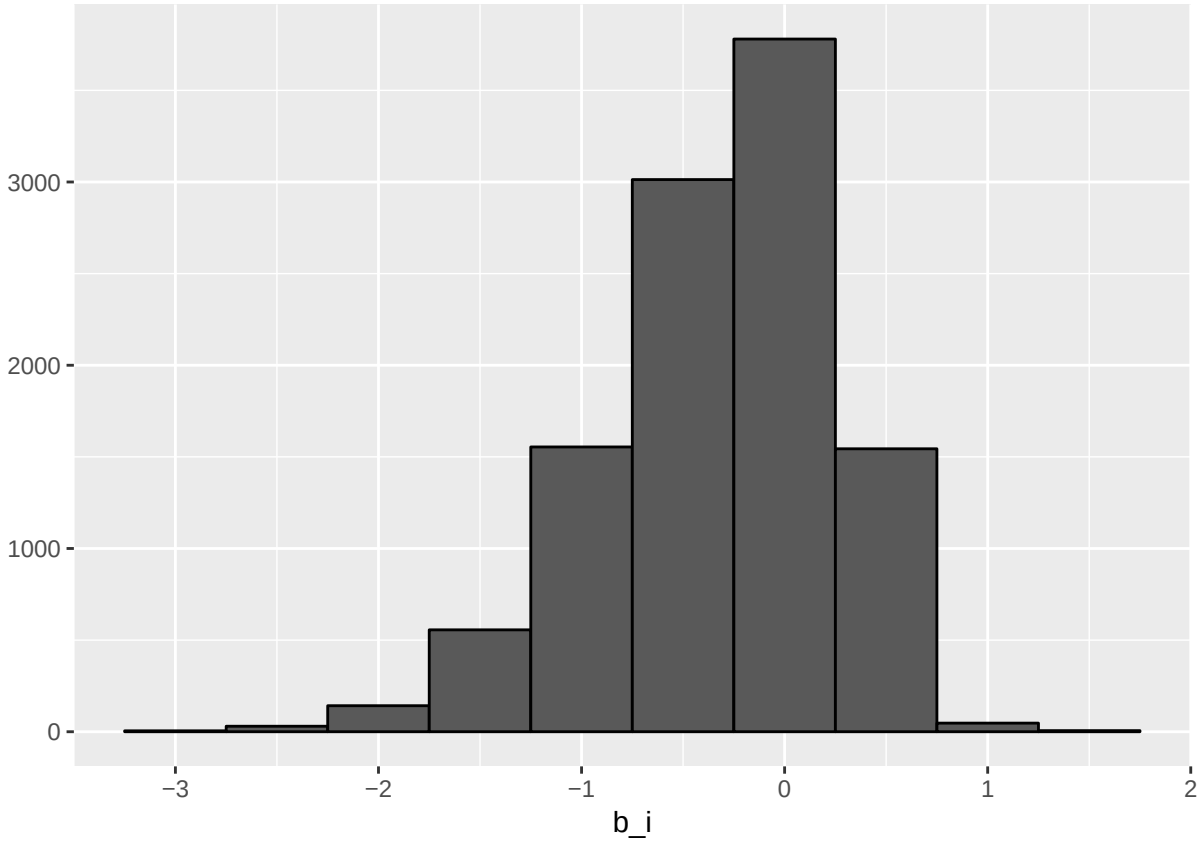
```
mu_hat <- mean(edx$rating)

naive_rmse <- RMSE(edx$rating, mu_hat)

rmse_results <- tibble(method = "Just the average", RMSE = naive_rmse)
```

A second model takes into account that some movies are just generally rated higher than others. Producing an RMSE of 0.942, the model performs better than the "naive" average-based model.

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



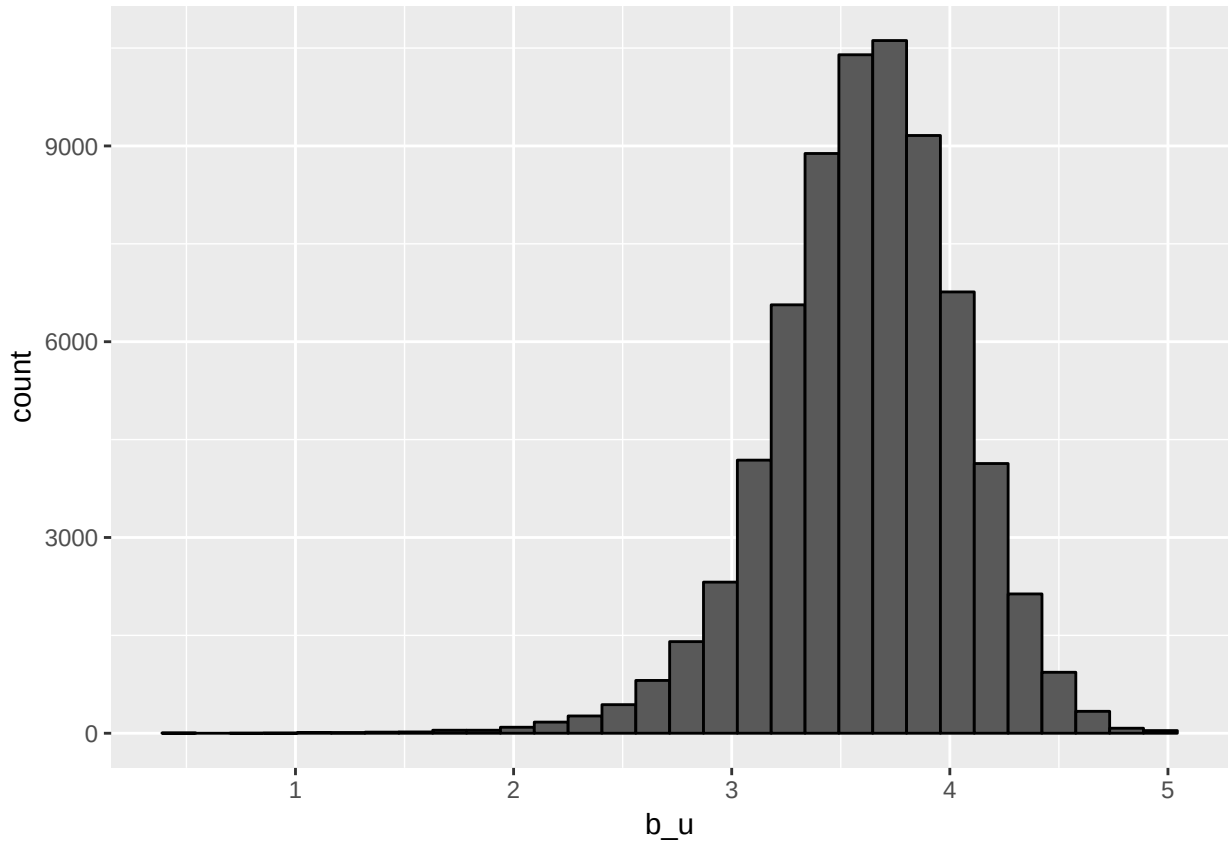

```
predicted_ratings <- mu + edx %>%
  left_join(movie_avgs, by='movieId') %>%
  pull(b_i)

model_1_rmse <- RMSE(predicted_ratings, edx$rating)
rmse_results <- bind_rows(rmse_results,
                          tibble(method="Movie Effect Model",
                                 RMSE = model_1_rmse))
rmse_results %>% knitr::kable()
```

| method | RMSE |
|---|---|
| Just the average | 1.0603313 |
| Movie Effect Model | 0.9423475 |

A third model takes into account variation in rating behavior across different users. Adding this "user effect" further reduces the RMSE to 0.857.

```
edx %>%
  group_by(userId) %>%
  summarize(b_u = mean(rating)) %>%
  filter(n()>=100) %>%
  ggplot(aes(b_u)) +
  geom_histogram(bins = 30, color = "black")
```




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

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

model_2_rmse <- RMSE(predicted_ratings, edx$rating)
```

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

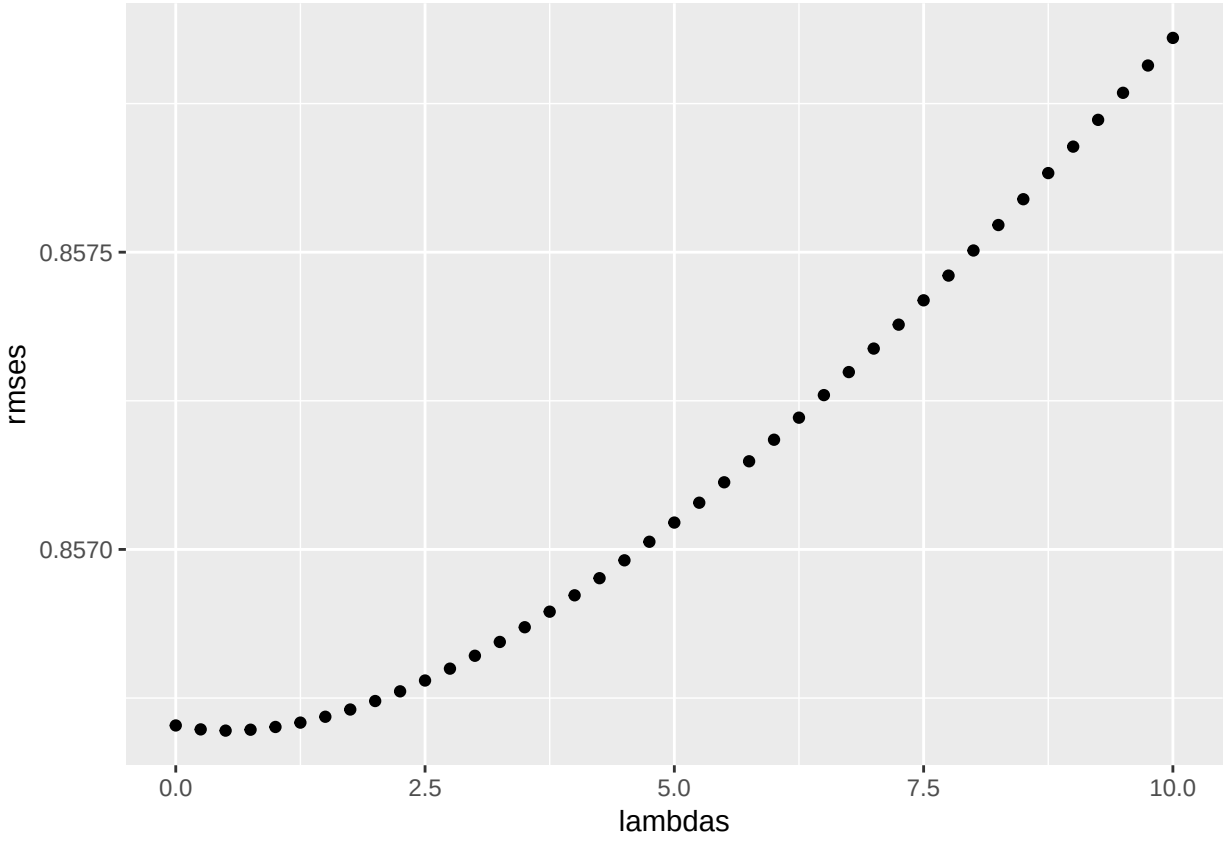
| method | RMSE |
|---|---|
| Just the average | 1.0603313 |
| Movie Effect Model | 0.9423475 |
| Movie + User Effects Model | 0.8567039 |

Another model that integrates a time effect (week-based) does not considerable improve the prediction. A model measuring a genre effect was unfortunately not possible due to the strong computing power required.

Hence, the movie and user effect model was further refined using regularization. This tuning process defines penalty parameter lambda that reduces the effect of movies and users with only a small a number of observations.

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

qplot(lambdas, rmses)
```




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

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

| method | RMSE |
|---|---|
| Just the average | 1.0603313 |
| Movie Effect Model | 0.9423475 |
| Movie + User Effects Model | 0.8567039 |
| Regularized Movie + User Effect Model | 0.8566952 |

The regularization process only slightly reduces the RMSE further to 0.856. The reason for the small effect might be that the overall sample size is already very large.

The final model is now evaluated based on the test data. The RMSE is only slightly higher than in the previous models based on the training data.

```
mu <- mean(edx$rating)
b_i <- edx %>%
    group_by(movieId) %>%
    summarize(b_i = sum(rating - mu)/(n()+lambda))
b_u <- edx %>%
    left_join(b_i, by="movieId") %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_i - mu)/(n()+lambda))
```

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

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

```
## [1] 0.8652226
```

## Discussion of results

The modelling process shows that knowing the average rating per movie and user already significantly improves the prediction accuracy compared to a simple mean-based model. The algorithm can be used to provide recommendations to Netflix user on the movies that might correspond to their preferences.

These results should, however, be treated with caution, because users who like or do not like a movie will not necessarily rate it. In this project, "rating predicts rating" - nothing more and nothing less: This does not mean that rating predicts users' actual perception of a movie.

## Conclusion

Using a linear regression machine learning algorithm, this small project could significantly improve the accuracy of predicting movie ratings, compared to a simple average-based model. Taking into account other factors such as movie genre, user's location and age might further improve accuracy.