# movielens_recommendation pdf

*Bobby Arrington*

*June 3, 2019*

### Abstract

Recommendations systems use ratings that users have given items to make specific recommendations to users. These ratings are usually in the range 1 to 5 stars, but there are some raters who give half stars. In this study, considering that the average rating of users has a certain stability, we propose a personalized fitting pattern  to predict any missing ratings based on the similarity score set, which combines both the user-based and item-based. We will not use non-rating factors such as user's (movie goer) age, gender, education, occupation, movie's release date and price. However, we will use some vector adjustment to come up with least "RMSE". We will use the experimental results on the MovieLens dataset to show that our proposed algorithms can increase the accuracy of our recommendation. That it can be used to predict what rating a given user will give a specific item. Items for which a high rating is predicted for specific users are then recommended to that user.

### Introduction

This is the "Capstone Project: All Learners", In this project, we will be creating a movie recommendation system using the MovieLens dataset. In project we will be using the large version of the "MovieLens" dataset with 10 millions ratings of the latest movies.  We will be creating a recommendation system using all the tools we have used throughout the previous 8 courses in this series. Additioally, we will be using the provide "Test and Validation" script to create the "edx" and "validation" datasets.

The output of this process will be least "RMSE" ans predicated movie ratings.

### Method/Analysis

```
## Loading required package: tidyverse

## -- Attaching packages --------------------------------------------------------------------------------

## v ggplot2 3.1.1     v purrr   0.3.2
## v tibble  2.1.2     v dplyr   0.8.1
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

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

## Loading required package: caret

## Loading required package: lattice

##
## Attaching package: 'caret'

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

## Joining, by = c("userId", "movieId", "rating", "timestamp", "title", "genres")
```

```r
library(tidyr)
library(dplyr)
library(tidyverse)
library(tinytex)
data(edx)
```

```
## Warning in data(edx): data set 'edx' not found
```

```r
data(validation)
```

```
## Warning in data(validation): data set 'validation' not found
```

```
 Building the Recommendation System
 Creating the beginning RMSE
```

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

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

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

```r
naive_rmse <- RMSE(validation$rating, mu_hat)
naive_rmse
```

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

```r
predictions <- rep(2.5, nrow(validation))
RMSE(validation$rating, predictions)
```

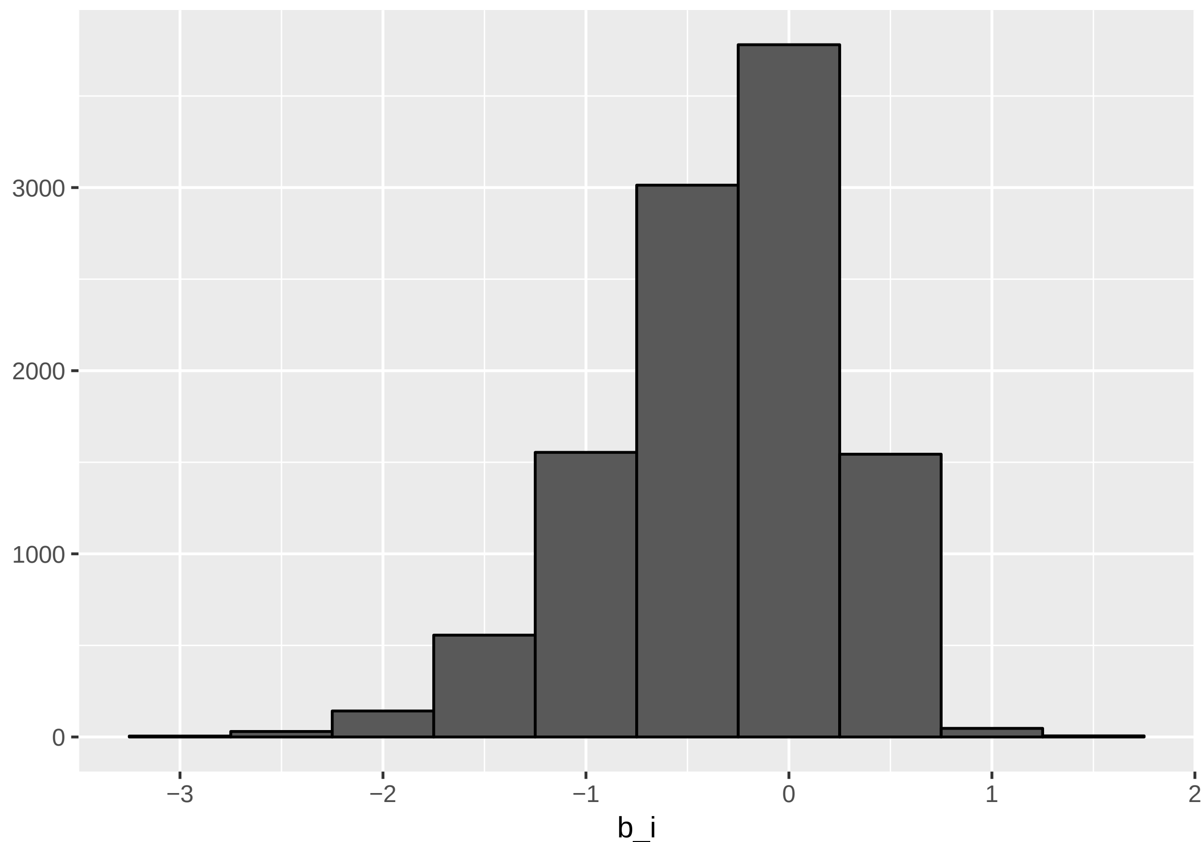
```
## [1] 1.46641
```

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

```
## Warning: `data_frame()` is deprecated, use `tibble()`.
## This warning is displayed once per session.
```

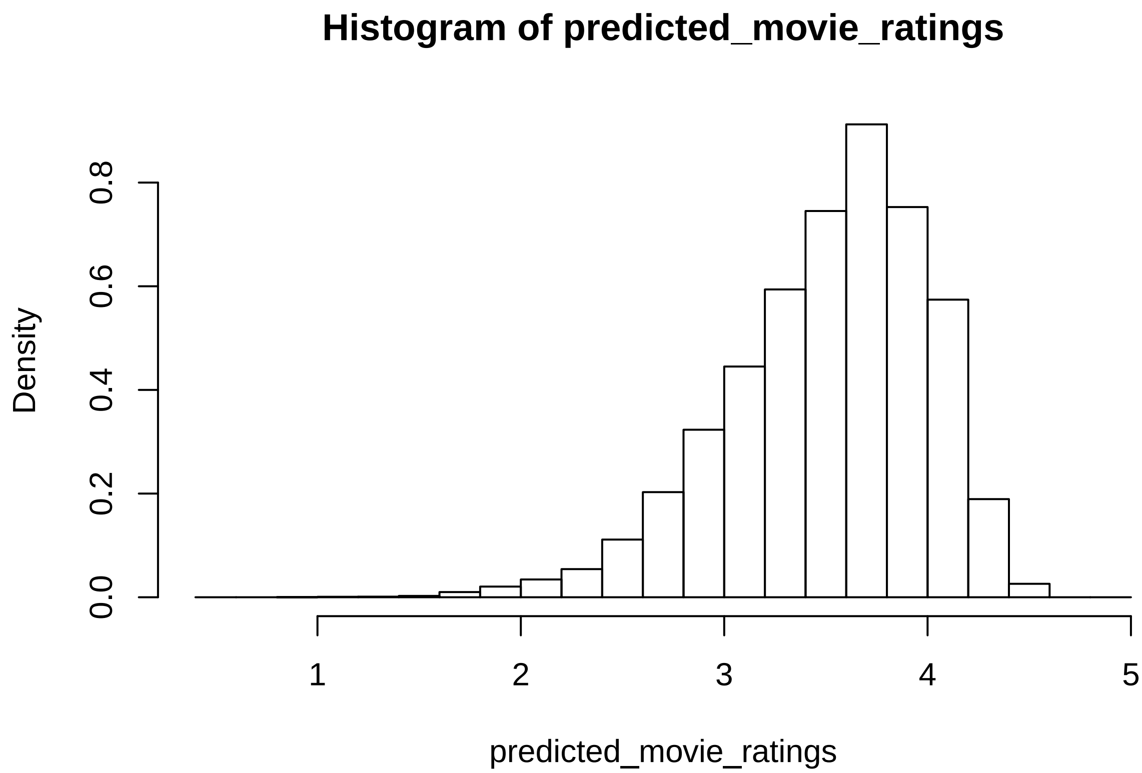
```
The code below replaces the long running
# fit <- lm(rating ~ as.factor(userId), data = movielens)
Additionally, it computes the movie average ratings
```

```
Movie Predicated Avg Ratings Histogram
```




Creating the Movie Predictions

Movie Density Prediction Histogram




Summarized Movie Predictions Table

```
##
## Call:
##  density.default(x = predicted_movie_ratings)
##
## Data: predicted_movie_ratings (999999 obs.); Bandwidth 'bw' = 0.02763
##
##        x                 y
##  Min.   :0.4171   Min.   :0.0000000
##  1st Qu.:1.5835   1st Qu.:0.0002733
##  Median :2.7500   Median :0.0381022
##  Mean   :2.7500   Mean   :0.2141053
##  3rd Qu.:3.9165   3rd Qu.:0.4200160
##  Max.   :5.0829   Max.   :1.0707028
```

Creating the RMSE Predictions for Final Table

```
model_1_rmse <- RMSE(predicted_movie_ratings, validation$rating)
```

The Results of combined Movie + User effects Model

```
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Movie Effect Model",
                                     RMSE = model_1_rmse ))
```

```
## Creating the User Avg
```

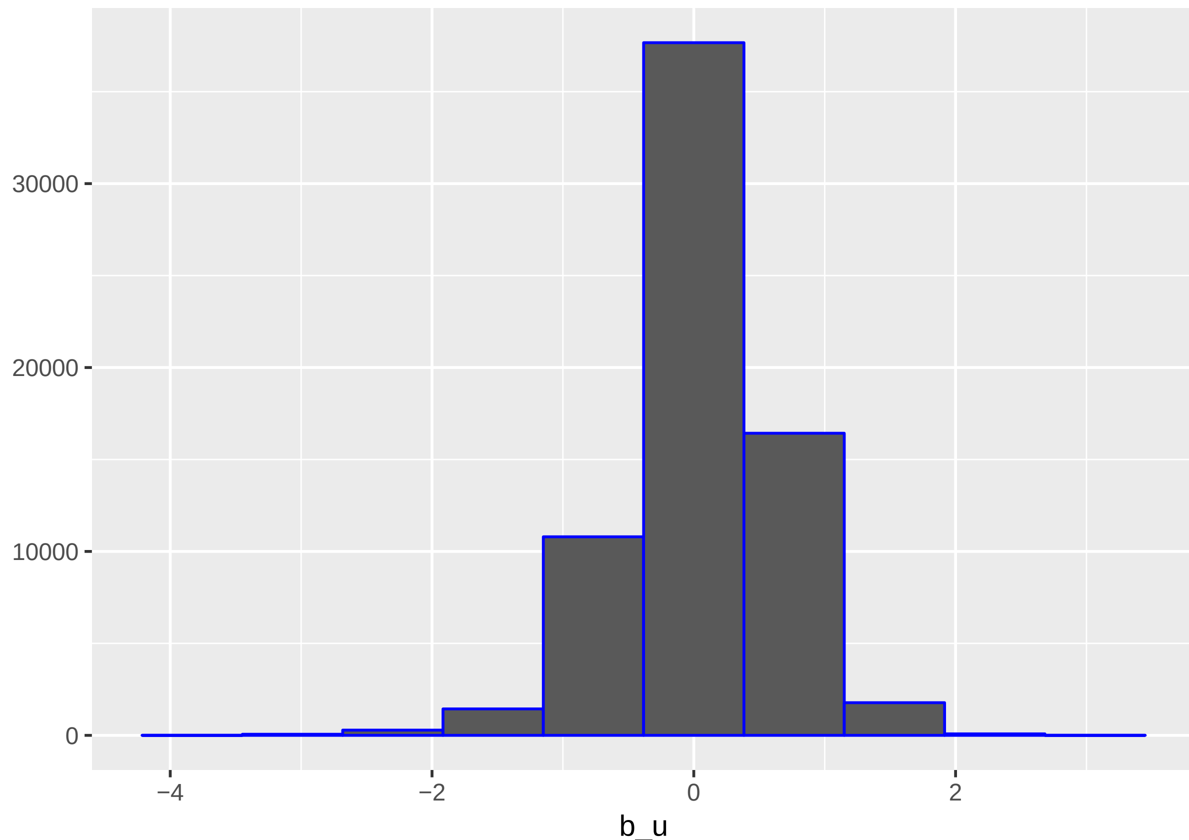
```
user_avgs <- validation %>%
        left_join(movie_avgs, by = "movieId") %>%
        group_by(userId) %>%
        summarize(b_u = mean(rating - mu - b_i))
```

```
## Creating the User Avg Predictions
```

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

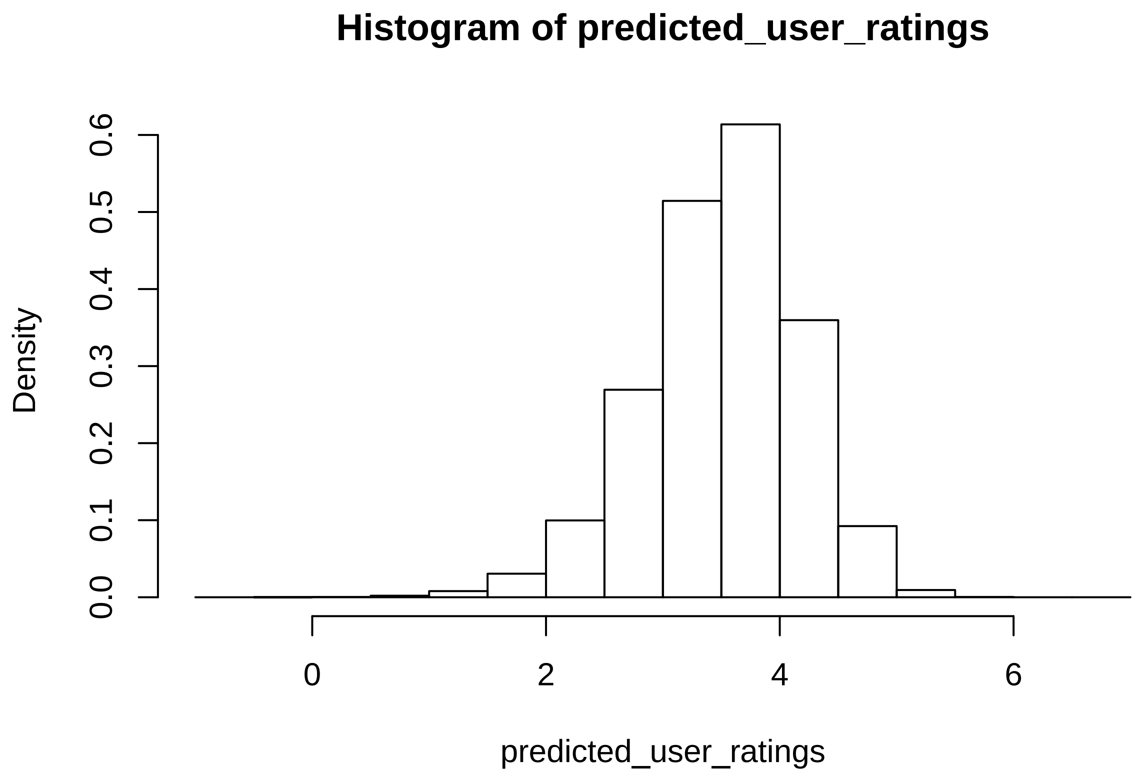
```
## Displaying the User Avg Histogram
```

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




```
## Creating the User Density Prediction Histogram
```

```
hist(predicted_user_ratings, freq = F)
```




## Displaying the Summarized User Predictions Table

```
density(predicted_user_ratings)
```

```
##
## Call:
##  density.default(x = predicted_user_ratings)
##
## Data: predicted_user_ratings (999999 obs.);  Bandwidth 'bw' = 0.03637
##
##        x                y
##  Min.   :-0.915   Min.   :0.0000000
##  1st Qu.: 1.023   1st Qu.:0.0001584
##  Median : 2.962   Median :0.0091512
##  Mean   : 2.962   Mean   :0.1288481
##  3rd Qu.: 4.900   3rd Qu.:0.1865366
##  Max.   : 6.839   Max.   :0.6389534
```

## Creating the RMSE Predictions for Final Table

```
model_2_rmse <- RMSE(predicted_user_ratings, validation$rating)
```

## The Results of combined Movie + User effects Model

```
rmse_results <- bind_rows(rmse_results,
                          data_frame(method = "Movie + User effects Model",
                                     RMSE = model_2_rmse))
```

*** Results ***

```
## The following code creates a table with the three RMSE
rmse_results %>% knitr::kable()
```

| method | RMSE |
|---|---|
| Just the average | 1.0612018 |
| Movie Effect Model | 0.9439087 |
| Movie + User effects Model | 0.8292477 |

*** Conclusions ***

```
Base on the results in the above table, the Movie + User Effects Model
yeilds the lowest RMSE of 0.8292477
```

## The End