HarvardX: PH125.9x Data Science

MovieLens Rating Prediction Project

April 10, 2019

Philip K W Ng

I. Introduction

A recommender system is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item.

Recommender systems are utilized in a variety of areas including movies, music, news, books, research articles, search queries, social tags, and products in general. There are also recommender systems for experts, collaborators, jokes, restaurants, garments, financial services, life insurance, online dating, and Twitter pages.

This project will build a movie recommendation system using the 10M MovieLens Dataset collected by GroupLens Research, which includes 10,000,000 ratings on 10,000 movies by 72,000 users.

II. Executive Summary

The objective of this project is to train a machine learning algorithm that predicts user ratings (from 0.5 to 5 stars) using the inputs of a provided subset (edx dataset) to predict movie ratings in a provided validation set.

The root-mean-square error (RMSE) is a frequently used measure of the differences between values (sample or population values) predicted by a model or an estimator and the values observed. RMSE is a measure of accuracy, to compare forecasting errors of different models for a dataset and not between datasets, as it is scale-dependent. RMSE is always non-negative, and a value of 0 (almost never achieved in practice) would indicate a perfect fit to the data. In general, a lower RMSE is better than a higher one.

The following key steps are followed in order to perform the analysis and make the conclusion:

- Prepare Data
- Summarize Dataset
- Visualize Dataset
- Evaluate Algorithm
- Evaluate Validation set

In the project, three models ("Simple Average", "Movie_Effect" and "Movie+User_Effect") are developed and their accuracy is assessed using their resulting RMSE. Finally, the best resulting model, "Movie + User_Effect Model" with RMSE of 0.8426, is ran directly on the validation set to predict the movie ratings. The RMSE result on validation dataset of 0.8294 is lower than the results on test dataset of 0.8426, suggesting that the "Moive+User_Effect" model is likely a reliable prediction model.

Details of the analysis will be explained in detail in the subsequent sections.

III. Prepare Data

edx dataset

The following edx sex is used to perform the analysis in this project. The “ggplot2” package is also added to the edx set.

```
## Loading required package: tidyverse

## -- Attaching packages ---------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------- tidyverse 1.2.1 --

## v ggplot2 3.1.0      v purrr   0.3.0
## v tibble  2.0.1      v dplyr   0.8.0.1
## v tidyr   0.8.3      v stringr 1.4.0
## v readr   1.3.1      v forcats 0.3.0

## -- Conflicts ------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------ tidyverse_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")
```

Training and Testing dataset

The test and training datasets are derived using edx set: 80% sample for training, and 20% sample for testing.

```
set.seed(1)
train_index <- createDataPartition(y = edx$rating, times = 1, p = 0.8, list = FALSE)
train_set <- edx[train_index,]
temp <- edx[-train_index,]
test_set <- temp %>%
semi_join(train_set, by = "movieId") %>%
semi_join(train_set, by = "userId")
removed <- anti_join(temp, test_set)

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

train_set <- rbind(train_set, removed)
rm(temp, removed)
```

IV. Summarize Dataset

The following ways are used to look at the raw data from different perspectives: shape, size, type, general layout. Inspecting data helps build up intuition and identify questions for the edx and validation datasets.

```
summary(edx)

##     userId       movieId        rating       timestamp
##  Min.   :    1   Min.   :    1   Min.   :0.500   Min.   :7.897e+08
##  1st Qu.:18124   1st Qu.:  648   1st Qu.:3.000   1st Qu.:9.468e+08
##  Median :35738   Median : 1834   Median :4.000   Median :1.035e+09
##  Mean   :35870   Mean   : 4122   Mean   :3.512   Mean   :1.033e+09
##  3rd Qu.:53607   3rd Qu.: 3626   3rd Qu.:4.000   3rd Qu.:1.127e+09
##  Max.   :71567   Max.   :65133   Max.   :5.000   Max.   :1.231e+09
##    title              genres
##  Length:9000055     Length:9000055
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##

str(edx)

## '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 83
8984885 838983707 838984596 ...
##  $ 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
dventure|Sci-Fi" ...

dim(edx)

## [1] 9000055       6

summary(validation)

##     userId       movieId        rating       timestamp
##  Min.   :    1   Min.   :    1   Min.   :0.500   Min.   :7.897e+08
##  1st Qu.:18096   1st Qu.:  648   1st Qu.:3.000   1st Qu.:9.467e+08
##  Median :35768   Median : 1827   Median :4.000   Median :1.035e+09
##  Mean   :35870   Mean   : 4108   Mean   :3.512   Mean   :1.033e+09
##  3rd Qu.:53621   3rd Qu.: 3624   3rd Qu.:4.000   3rd Qu.:1.127e+09
##  Max.   :71567   Max.   :65133   Max.   :5.000   Max.   :1.231e+09
##    title              genres
##  Length:999999      Length:999999
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

```
str(validation)

## 'data.frame':    999999 obs. of  6 variables:
##  $ userId   : int  1 1 1 2 2 2 3 3 4 4 ...
##  $ movieId  : num  231 480 586 151 858 ...
##  $ rating   : num  5 5 5 3 2 3 3.5 4.5 5 3 ...
##  $ timestamp: int  838983392 838983653 838984068 868246450 868245645 868245920 1136075494 1
133571200 844416936 844417070 ...
##  $ title    : chr  "Dumb & Dumber (1994)" "Jurassic Park (1993)" "Home Alone (1990)" "Rob Roy (1995)"
 ...
##  $ genres   : chr  "Comedy" "Action|Adventure|Sci-Fi|Thriller" "Children|Comedy" "Action|Drama|Ro
mance|War" ...

dim(validation)

## [1] 999999      6
```

The results below show the top 10 genres:

```
edx %>% separate_rows(genres, sep = "\\|")%>%
group_by(genres) %>%
summarize(count = n()) %>%
arrange(desc(count))

## # A tibble: 800 x 2
##    genres                                      count
##    <chr>                                       <int>
##  1 Drama                                      733296
##  2 Comedy                                     700889
##  3 Comedy|Romance                             365468
##  4 Comedy|Drama                               323637
##  5 Comedy|Drama|Romance                       261425
##  6 Drama|Romance                              259355
##  7 Action|Adventure|Sci-Fi                    219938
##  8 Action|Adventure|Thriller                  149091
##  9 Drama|Thriller                             145373
## 10 Crime|Drama                                137387
## # ... with 790 more rows
```

The results below show the top 10 movies:

```
edx %>% group_by(movieId, title)%>%
summarize(count = n()) %>%
arrange(desc(count))

## # A tibble: 10,677 x 3
## # Groups:   movieId [10,677]
##    movieId title                                count
##      <dbl> <chr>                                <int>
##  1     296 Pulp Fiction (1994)                  31362
##  2     356 Forrest Gump (1994)                  31079
##  3     593 Silence of the Lambs, The (1991)     30382
##  4     480 Jurassic Park (1993)                 29360
##  5     318 Shawshank Redemption, The (1994)     28015
```

```
## 6    110 Braveheart (1995)                                  26212
## 7    457 Fugitive, The (1993)                               25998
## 8    589 Terminator 2: Judgment Day (1991)                  25984
## 9    260 Star Wars: Episode IV - A New Hope (a.k.a. Star Wars)  (19~ 25672
## 10    150 Apollo 13 (1995)                                  24284
## # ... with 10,667 more rows
```

The results below show the top 10 movies by rating:

```
edx %>% group_by(rating, title)%>%
summarize(count = n()) %>%
arrange(desc(count))

## # A tibble: 88,248 x 3
## # Groups:   rating [10]
##    rating title                                          count
##     <dbl> <chr>                                          <int>
## 1      5 Shawshank Redemption, The (1994)                14769
## 2      5 Pulp Fiction (1994)                             13441
## 3      5 Silence of the Lambs, The (1991)                11805
## 4      5 Schindler's List (1993)                         11533
## 5      5 Star Wars: Episode IV - A New Hope (a.k.a. Star Wars)  (197~ 11276
## 6      4 Fugitive, The (1993)                            10948
## 7      5 Forrest Gump (1994)                             10466
## 8      3 Batman (1989)                                   10399
## 9      4 Silence of the Lambs, The (1991)                10289
## 10     5 Usual Suspects, The (1995)                      10088
## # ... with 88,238 more rows
```

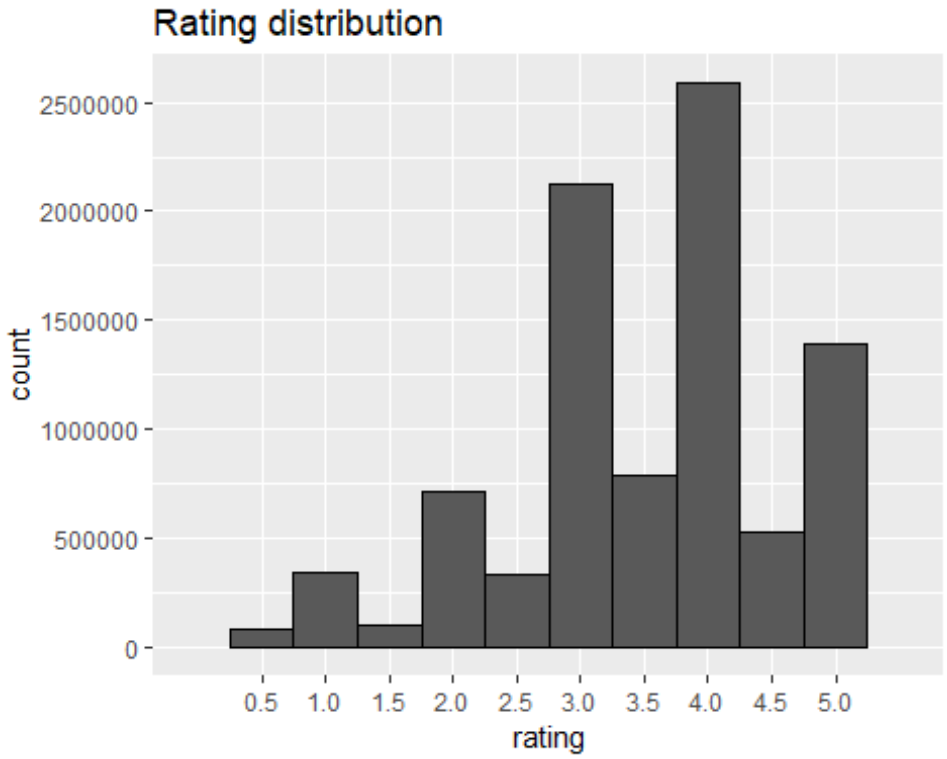
V. Visualize Dataset

Data visualization is perhaps the fastest and most useful way to summarize and learn more about the data.

Visualization means creating charts and plots from the raw data. Plots of the distribution or spread of attributes can help spot outliers, strange or invalid data.

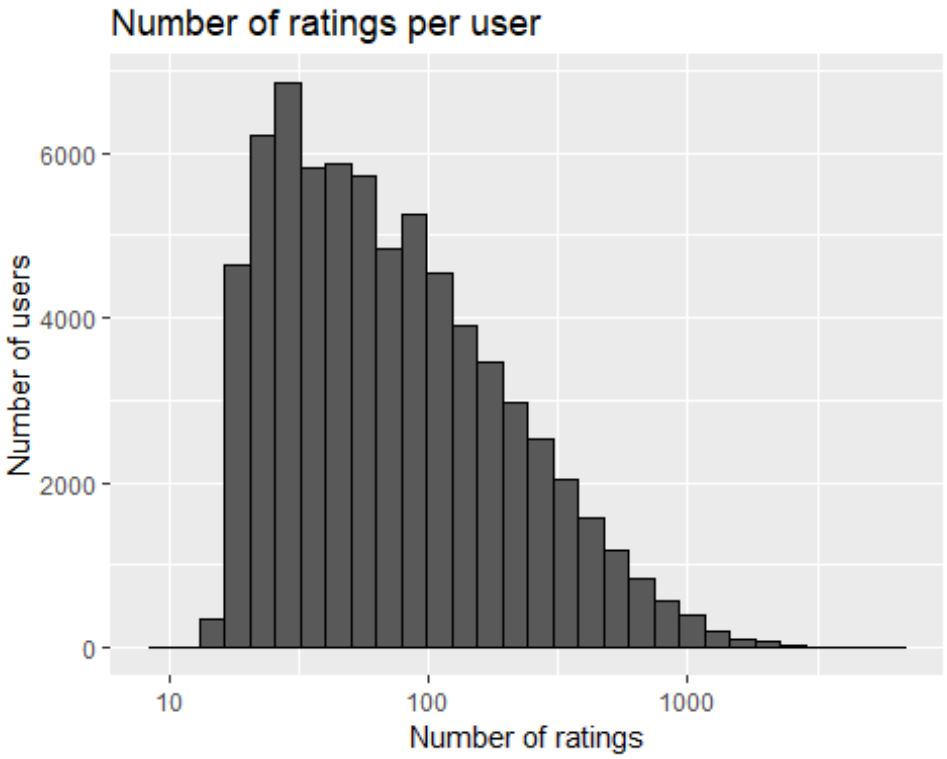
Rating Distribution: Users give full-star ratings more frequently than half-star ratings.

```
edx %>%
ggplot(aes(rating)) +
geom_histogram(binwidth = 0.5, color = "black") +
scale_x_discrete(limits = c(seq(0.5,5,0.5))) +
scale_y_continuous(breaks = c(seq(0, 3000000, 500000))) +
ggtitle("Rating distribution")
```



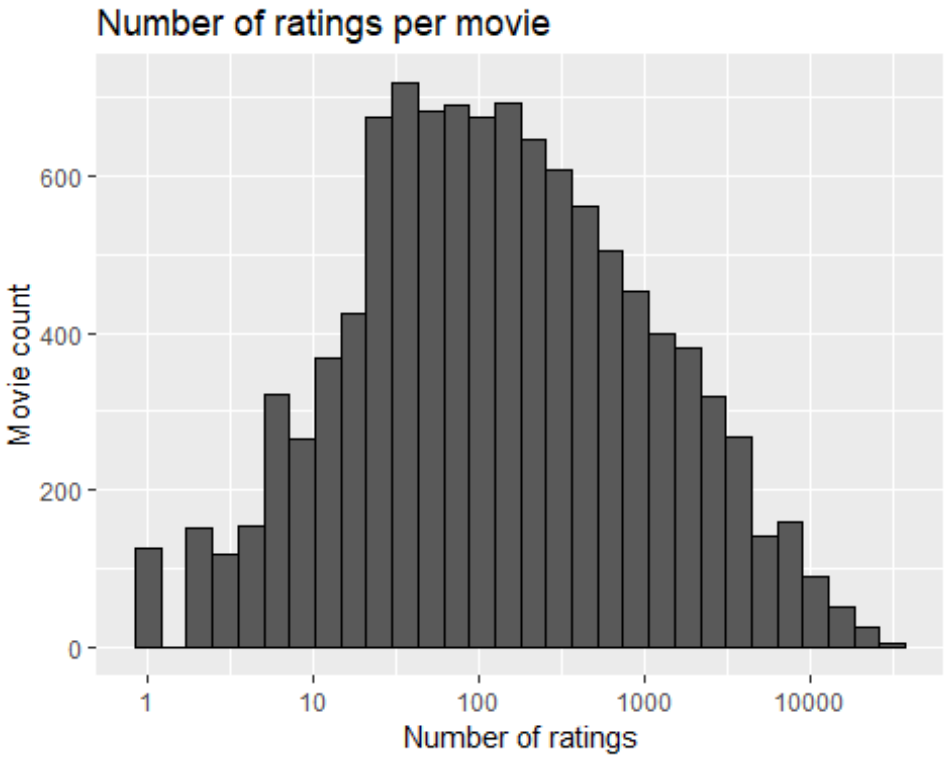
Rating No. Per User: A lot of users rate hundreds of movies.

```
edx %>% count(userId) %>%
ggplot(aes(n)) +
geom_histogram(bins = 30, color = "black") +
scale_x_log10() +
xlab("Number of ratings") +
ylab("Number of users") +
ggtitle("Number of ratings per user")
```



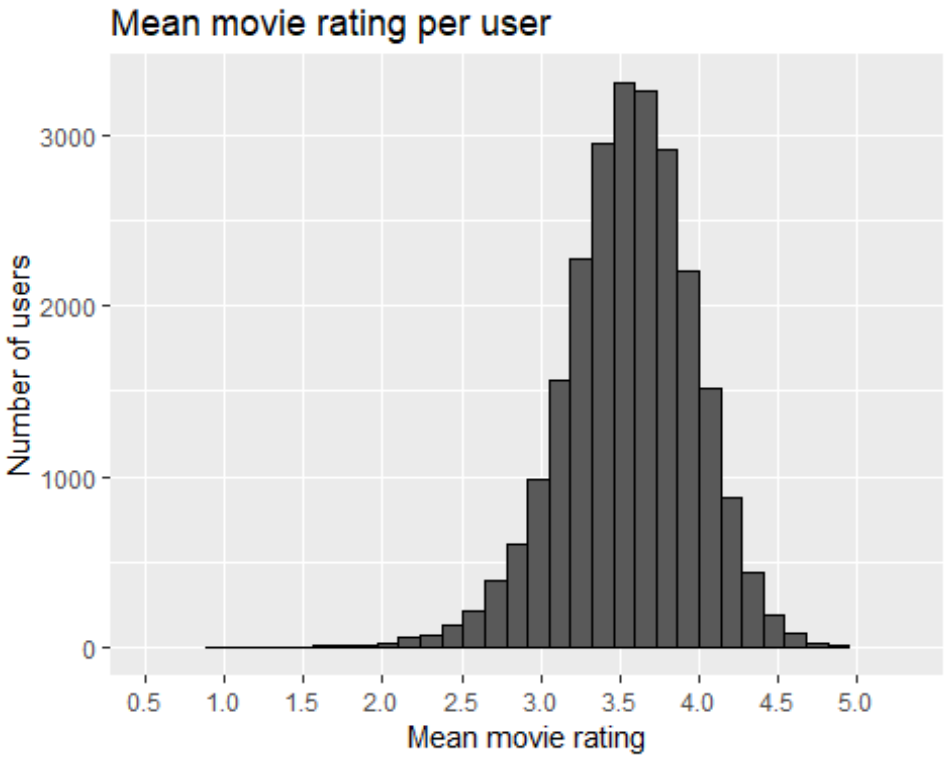
Rating No. Per Movie: Most movies were rated hundreds or even thousands of times.

```
edx %>%
count(movieId) %>%
ggplot(aes(n)) +
geom_histogram(bins = 30, color = "black") +
xlab("Number of ratings") +
ylab("Movie count") +
scale_x_log10() +
ggtitle("Number of ratings per movie")
```



Mean Movie Rating Per User: After shortlisting those users that have rated at least 100 movies, it is found that most users gave ratings of 3.0, 3.5 and 4.0.

```
edx %>%
group_by(userId) %>%
filter(n() >= 100) %>%
summarise(mean_rating = mean(rating)) %>%
ggplot(aes(mean_rating)) +
geom_histogram(bins = 30, color = "black") +
xlab("Mean movie rating") +
ylab("Number of users") +
ggtitle("Mean movie rating per user") +
scale_x_discrete(limits = c(seq(0.5,5,0.5)))
```



VI. Evaluate Algoritum

The following RMSE function is used to assess three alogritums in this section.

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

1st model: Simple Average Model

The 1st model predicts rating using the dataset's mean rating, and all differences in movie ratings are explained by random variation. Following is the equation used for the calculation:

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

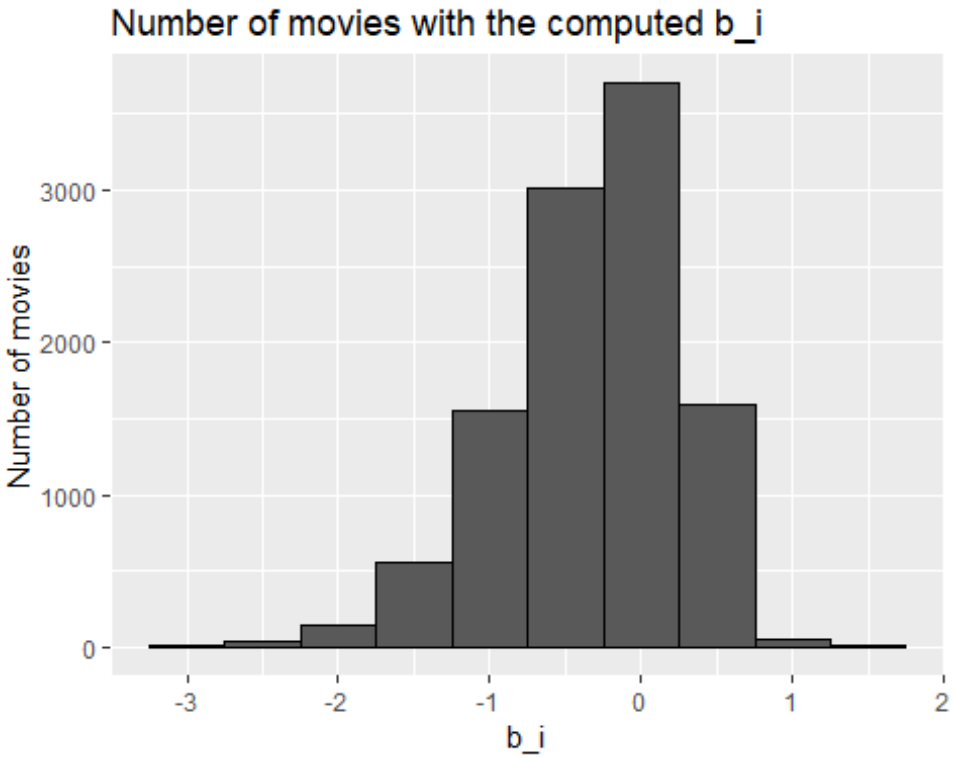
```
mu_hat <- mean(train_set$rating)
model_1_rmse <- RMSE(test_set$rating, mu_hat)
rmse_results <- data_frame(Model = "Simple Average", RMSE = model_1_rmse)

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

rmse_results%>%knitr::kable()
```

Model	RMSE
Simple Average	1.059735

2nd model: Movie_Effect Model

Using the average mean rating of all movies may not be appropriae as popular movies are likely rated more than unpopular movies. Hence, in order to improve prediction, the average mean rating of each movie is compared to the average mean rating, and the estimation deviation and the resulting variables ("b" or bias) are used to predict using the following equation: $Y_{u,i} = \mu + b_i + \epsilon_{u,i}$

```
mu <- mean(train_set$rating)
movie_avgs <- train_set %>%
group_by(movieId) %>%
summarize(b_i = mean(rating - mu))
movie_avgs %>% qplot(b_i, geom ="histogram", bins = 10, data = ., color = I("black"),
ylab = "Number of movies", main = "Number of movies with the computed b_i")
```



The above histogram shows that the rating data skew to the left, which is a result of a lower boundary in a dataset, suggesting that most ratings are higher than the mean rating of all movies.

```
predicted_ratings <- mu + test_set %>%
left_join(movie_avgs, by='movieId') %>%
.$b_i
model_2_rmse <- RMSE(predicted_ratings, test_set$rating)
rmse_results <- bind_rows(rmse_results,
data_frame(Model="Movie_Effect",
RMSE = model_2_rmse ))
rmse_results %>% knitr::kable()
```

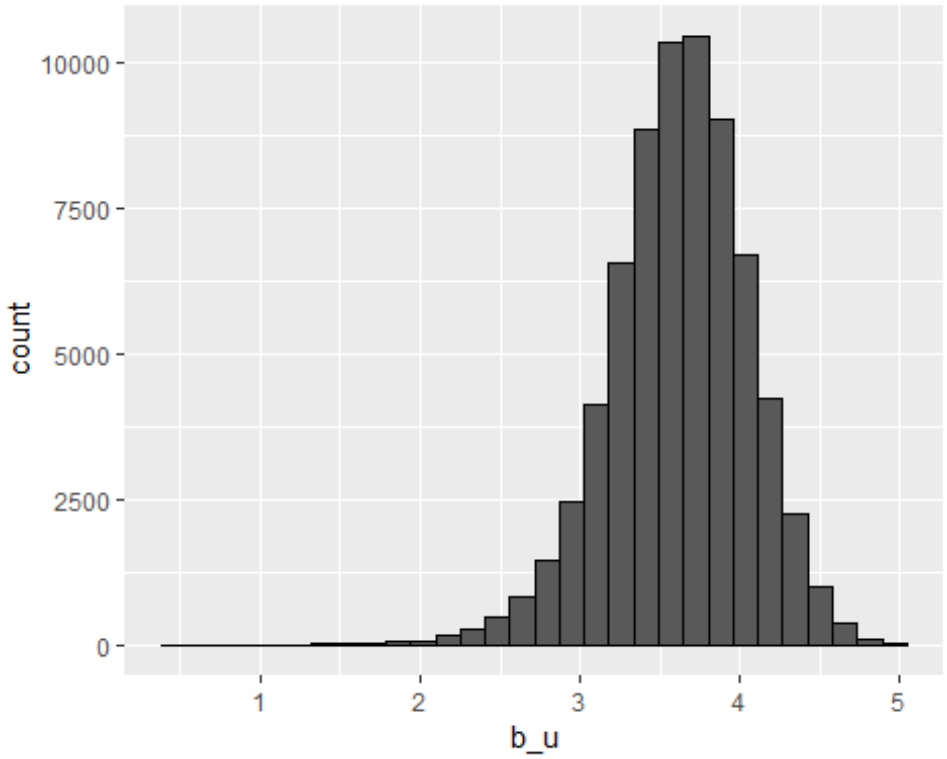
Model	RMSE
Simple Average	1.059735
Movie_Effect	0.943203

The RMSE results show that the 2nd model is an improvement of the 1st model.

3rd model: Movie+User_Effect Model

It is found that there is substantial variability across users: some users rate many movies while others are selective. Hence, the average rating for user μ is only computed for those that have rated over 100 movies, and the following equation is used for the prediction: $Y_{u,i} = \mu + b_i + b_u + \epsilon_{u,i}$

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



The above histogram shows that the rating data are more normally distributed compared to the 2nd Model, suggesting that the 3rd Model may produce more reliable results than the 2nd Model.

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

predicted_ratings <- test_set %>%
left_join(movie_avgs, by='movieId') %>%
left_join(user_avgs, by='userId') %>%
mutate(pred = mu + b_i + b_u) %>%
.$pred
model_3_rmse <- RMSE(predicted_ratings, test_set$rating)
rmse_results <- bind_rows(rmse_results,
data_frame(Model="Movie + User_Effect",
RMSE = model_3_rmse ))
rmse_results %>% knitr::kable()
```

Model	RMSE
Simple Average	1.0597347
Movie_Effect	0.9432030
Movie + User_Effect	0.8426298

It is shown that the RMSE is further reduced using the 3rd model.

VII. Evaluate validation set

Based on the results from the preceding section, the best resulting model, “Movie + User_Effect Model”, is ran directly on the validation set to predict the movie ratings. It is found that the RMSE of the validation set is 0.8294.

```
user_avgs_validation <- validation %>%
left_join(movie_avgs, by='movieId') %>%
group_by(userId) %>%
summarize(b_u = mean(rating - mu - b_i))
predicted_ratings <- validation %>%
left_join(movie_avgs, by='movieId') %>%
left_join(user_avgs_validation, by='userId') %>%
mutate(pred = mu + b_i + b_u) %>%
.$pred
model_rmse_validation <- RMSE(predicted_ratings, validation$rating)
model_rmse_validation

## [1] 0.8294231
```

VIII. Conclusion

In this project, three models (“Simple Average”, “Movie_Effect” and “Movie+User_Effect”) are developed to predict movie rating, and their accuracy is assessed using their resulting RMSE. The best resulting model, “Movie + User Effects Model” with RMSE of 0.8426, is ran directly on the validation set to predict the movie ratings. The RMSE result on validation dataset of 0.8294 is lower than the best results on test dataset of 0.8426 (3rd model), suggesting that the “Movie+User_Effect” model is likely a reliable prediction model.