MovieLens Report

Nkonzo Sithole

2/17/2021

#Introduction

#Environment preparation

```
knitr::opts_chunk$set(echo = TRUE, fig.align = 'center', cache=FALSE, cache.lazy = FALSE)
```

##Install and load libraries

#installing

```
if(!require(tidyverse)) install.packages("tidyverse")
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages --------------------------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
```

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

```
if(!require(kableExtra)) install.packages("kableExtra")
```

```
## Loading required package: kableExtra
```

```
##
## Attaching package: 'kableExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```
if(!require(tidyr)) install.packages("tidyr")
if(!require(tidyverse)) install.packages("tidyverse")
if(!require(stringr)) install.packages("stringr")
if(!require(forcats)) install.packages("forcats")
if(!require(ggplot2)) install.packages("ggplot2")
if(!require(lubridate)) install.packages("lubridate")
```

```
## Loading required package: lubridate
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
```

```
##     date, intersect, setdiff, union
```

```
if(!require(caret)) install.packages("caret")
```

```
## Loading required package: caret
```

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

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

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

##**Loading**

#Data preperation ##Create test set, validation set from movieLens that has 10M records in the dataset

```
dl <- tempfile()
set.seed(1)
download.file("http://files.grouplens.org/datasets/movielens/ml-10m.zip", dl)

#Create data frames

ratings <- read.table(text = gsub("::", "\t", readLines(unzip(dl, "ml-10M100K/ratings.dat"))),
                      col.names = c("userId", "movieId", "rating", "timestamp"))

movies <- str_split_fixed(readLines(unzip(dl, "ml-10M100K/movies.dat")), "\\::", 3)
colnames(movies) <- c("movieId", "title", "genres")
movies <- as.data.frame(movies) %>% mutate(movieId = as.numeric(movieId),
                                           title = as.character(title),
                                           genres = as.character(genres))

movielens <- left_join(ratings, movies, by = "movieId")
head(movielens)
```

```
##   userId movieId rating timestamp                         title
## 1      1     122      5 838985046              Boomerang (1992)
## 2      1     185      5 838983525               Net, The (1995)
## 3      1     231      5 838983392          Dumb & Dumber (1994)
## 4      1     292      5 838983421               Outbreak (1995)
## 5      1     316      5 838983392               Stargate (1994)
## 6      1     329      5 838983392 Star Trek: Generations (1994)
##                          genres
## 1                Comedy|Romance
## 2         Action|Crime|Thriller
## 3                        Comedy
## 4  Action|Drama|Sci-Fi|Thriller
## 5       Action|Adventure|Sci-Fi
## 6 Action|Adventure|Drama|Sci-Fi
```

```
#Total records
nrow(movielens$genres)
```

```
## NULL
```

```
#distinct genres
n_distinct(movielens$genres)
```

```
## [1] 797
```

Split movielens data, Validation set will be 10% of MovieLens data

```
set.seed(1)
test_index <- createDataPartition(y = movielens$rating, times = 1, p = 0.1, list = FALSE)
test <- movielens[-test_index,]
train <- movielens[test_index,]
# sami_join will ensure that userId and movieId are in validation dataset
validation <- train %>%
  semi_join(test, by = "movieId") %>%
  semi_join(test, by = "userId")
# Add rows removed from validation set back into test set
removed <- anti_join(train, validation)
```

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

```
test <- rbind(test, removed)
```

Data Summary and Explortory Data Analysis

```
str(test)
```

```
## 'data.frame':    9000061 obs. of  6 variables:
##  $ userId   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ movieId  : num  122 185 231 292 316 329 355 356 362 364 ...
##  $ rating   : num  5 5 5 5 5 5 5 5 5 5 ...
##  $ timestamp: int  838985046 838983525 838983392 838983421 838983392 838983392 838984474 838983653 83
##  $ title    : chr  "Boomerang (1992)" "Net, The (1995)" "Dumb & Dumber (1994)" "Outbreak (1995)" ...
##  $ genres   : chr  "Comedy|Romance" "Action|Crime|Thriller" "Comedy" "Action|Drama|Sci-Fi|Thriller"
```

```
summary(test)
```

```
##      userId         movieId          rating        timestamp
##  Min.   :    1   Min.   :    1   Min.   :0.500   Min.   :7.897e+08
##  1st Qu.:18122   1st Qu.:  648   1st Qu.:3.000   1st Qu.:9.468e+08
##  Median :35743   Median : 1834   Median :4.000   Median :1.035e+09
##  Mean   :35869   Mean   : 4120   Mean   :3.512   Mean   :1.033e+09
##  3rd Qu.:53602   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:9000061     Length:9000061
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

```
#from the summary of data we see that the rating is between 1 to 5, with mean of the rating of 3.512 an

test %>% group_by(rating) %>% summarize(count = n()) %>% top_n(5) %>%
    arrange(desc(count))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

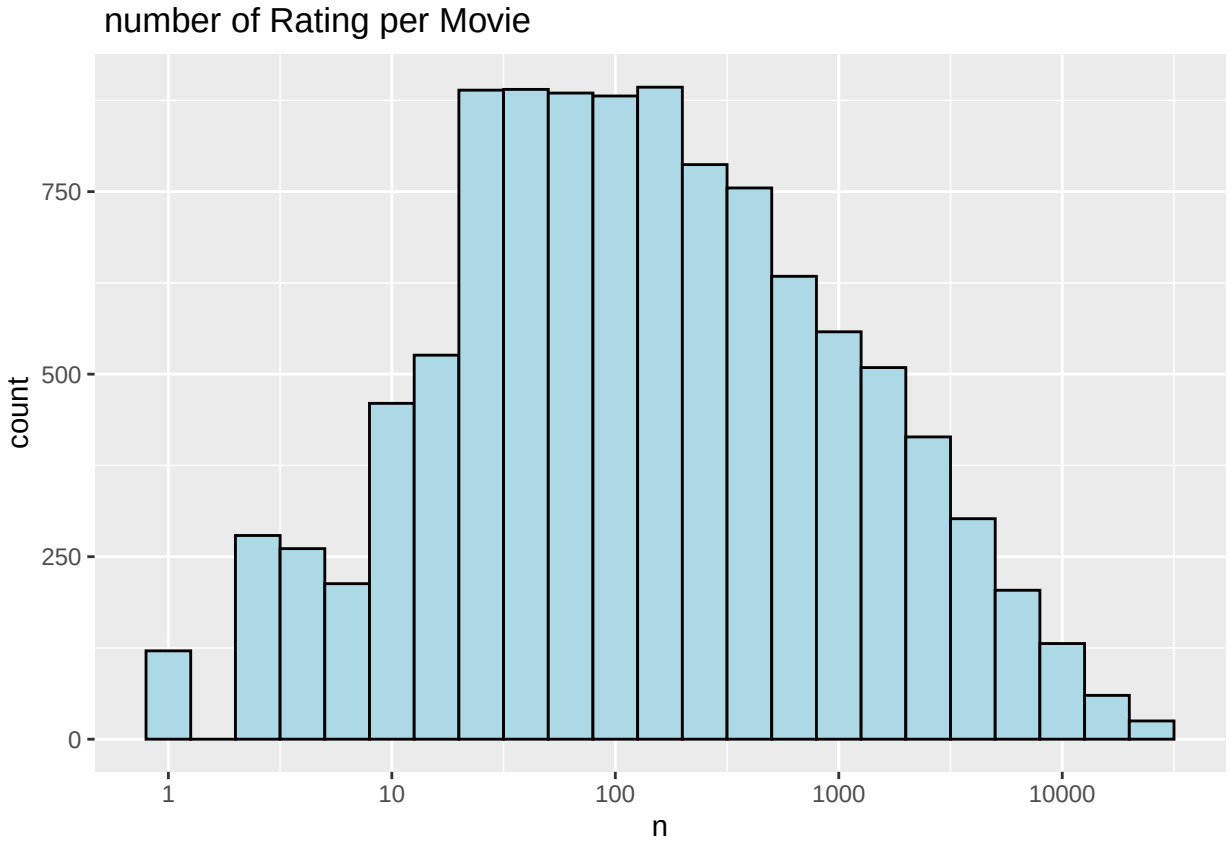
```
## Selecting by count
```

```
## # A tibble: 5 x 2
##   rating   count
##    <dbl>   <int>
## 1    4   2588021
## 2    3   2121638
## 3    5   1390541
## 4    3.5  792037
## 5    2    710998
```

```
test %>%
  summarize(n_users = n_distinct(userId),
            n_movies = n_distinct(movieId))
```

```
##   n_users n_movies
## 1   69878    10677
```

```
test %>% count(movieId) %>% ggplot(aes(n))+
  geom_histogram(color = "black" , fill= "light blue",bins = 30 , binwidth = 0.2)+
  scale_x_log10()+
  ggtitle(" number of Rating per Movie")+
  theme_gray()
```




```
# Some movies are not rated and some are rated about 1000 times.
```

#*Predictions**

```
##Create RMSE function that will be used in our models, where y_hat vector is for predicted ratings
RMSE <- function(true_ratings, y_hat){
   sqrt(mean((true_ratings - y_hat)^2))
```

```
}
```

##First Model

```
#Creating first model for the predicted ratings driven by avarage ratings only
y_hat <- mean(test$rating)

rmse_m1 <- RMSE(test$rating,y_hat)
cat("RMSE from Model 3: ", rmse_m1)
```

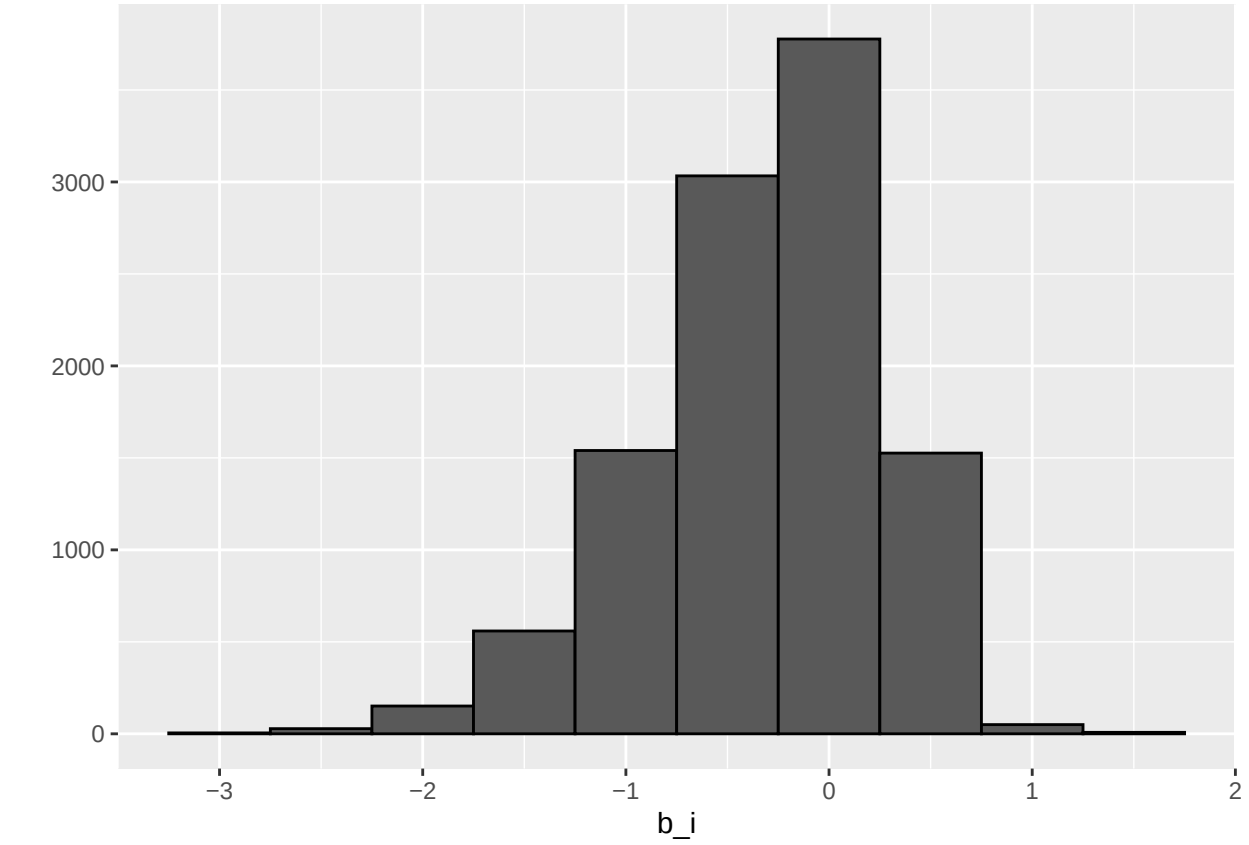
```
## RMSE from Model 3:  1.060393
```

##Second Model

```
#Second model driven by difference between ratings and avarage ratings b_i
mu <- mean(test$rating)
movie_avgs <- test %>%
   group_by(movieId) %>%
   summarize(b_i = mean(rating - mu))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
#b_i histogram graph looks normal
movie_avgs %>% qplot(b_i, geom ="histogram", bins = 10, data = ., color = I("black"))
```




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

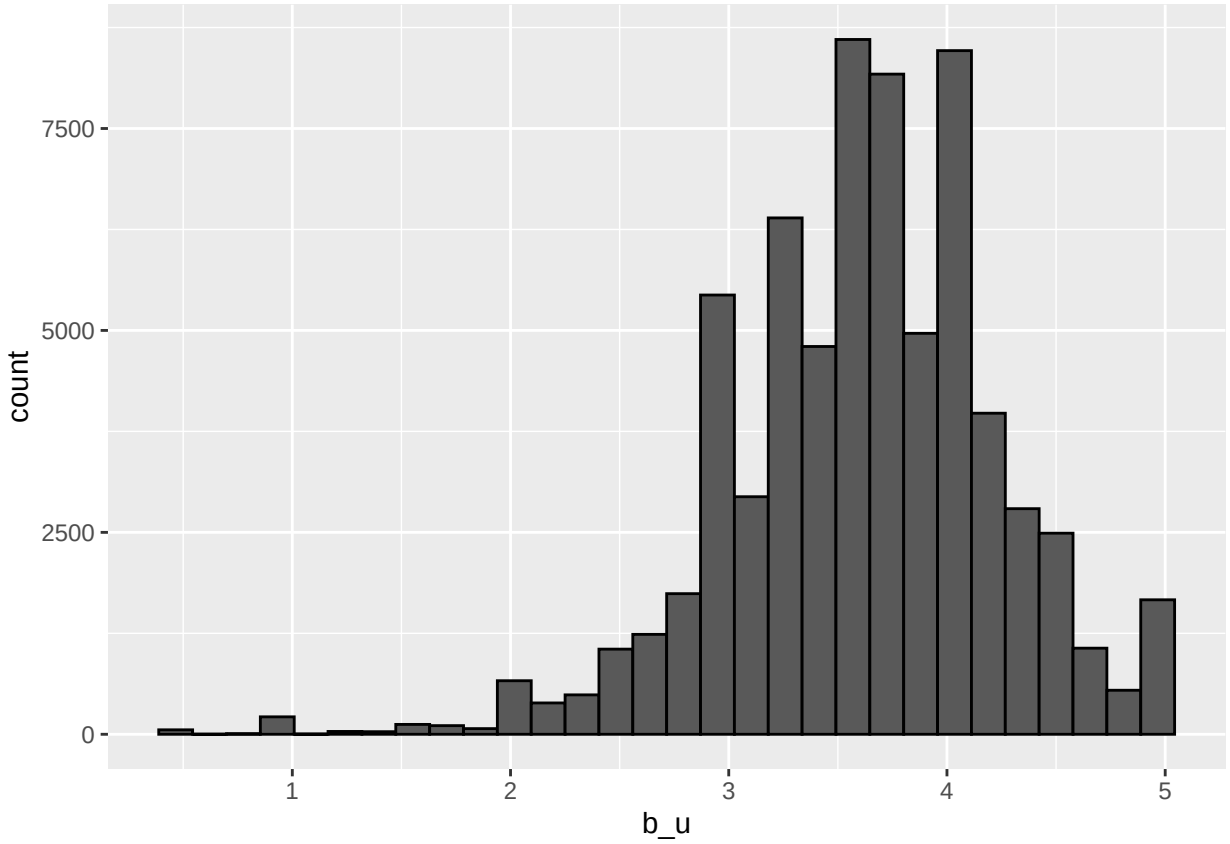
```
rmse_m2 <- RMSE(predicted_ratings, test$rating)
cat("RMSE from Model 2: ", rmse_m2)
```

```
## RMSE from Model 2:  0.942368
```

#Third Model

```
#Visualising b_i based on data grouped by userID
train %>%
  group_by(userId) %>%
  summarize(b_u = mean(rating)) %>%
  filter(n()>=100) %>%
  ggplot(aes(b_u)) +
  geom_histogram(bins = 30, color = "black")
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```




```
#finding b_u, the difference from on rating with mean and b_i
user_avgs <- train %>%
  left_join(movie_avgs, by='movieId') %>%
  group_by(userId) %>%
  summarize(b_u = mean(rating - mu - b_i))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
#finding predicted_ratings, the sum of b_u , mean and b_i
predicted_ratings <- test %>%
  left_join(movie_avgs, by='movieId') %>%
```

```
  left_join(user_avgs, by='userId') %>%
  mutate(pred = mu + b_i + b_u) %>%
  pull(pred)

rmse_m3 <- RMSE(predicted_ratings, test$rating)

## RMSE of the validation set

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

rmse_m3_final <- RMSE(validation$rating, valid_pred_rating)
cat("RMSE from Model 3: ", rmse_m3_final)
```

```
## RMSE from Model 3:  0.8293253
```

All models results

```
cat("RMSE from Model 1: ", rmse_m1)
```

```
## RMSE from Model 1:  1.060393
```

```
cat("RMSE from Model 2: ", rmse_m2)
```

```
## RMSE from Model 2:  0.942368
```

```
cat("RMSE from Model 3: ", rmse_m3_final)
```

```
## RMSE from Model 3:  0.8293253
```

Conclusion

```
#We can see that the RMSE improved when ratings per user are considered, Although we must note the poss
```