# HarvardX Data Science Capstone Project Submission: MovieLens

*SIU LUNG DAVID LAW*

*12/29/2019*

## Overview

The purpose of this project is to create a movie recommendation system using the edx file created from the 10M MovieLens datase included in the dslabs package, and then test it. The edx database contains 9000055 entries, listing the ratings given by 69878 users on 10677 movies. The following table summarises the columns and their important characteristics:

| Columns | Classes | Additional Info | Examples |
|---|---|---|---|
| userId | numeric | can be any positive integer | 1 |
| movieId | numeric | can be any positive integer | 122 |
| rating | numeric | can be 0 to 5 in steps of 0.5 | 3, 4.5, 5 |
| timestamp | integer | a sequence of integers | 1111623327 |
| title | character | movie title; and year of the movie is included in this column inside a bracket | Boomerang (1992) |
| genres | character | a movie can be multi-genre | Comedy\|Romance |

In this project, the rating is considered to be the outcome and learners can use the remaining columns as features to create a model to 'predict' the rating. The recommendation system will then be tested by a validation file created by the instructor, which has the same columns as the edx file but with fewer entries. The testing comprises of calculating the Root Mean Square Error ("RMSE"):

$$RMSE = \sqrt{\frac{1}{N}\sum(\hat{R} - R)^2} \tag{1}$$

between the actual ratings $R$ in the validation file and the predicted ratings $\hat{R}$. The resulting RMSE will be measured against the threshold 0.86499, the threshold as determined by the course instructor to score maximum points in this project.

The recommendation system proposed by me is based on the following formula:

$$R_i = \mu + M_i + U_i + G_i + \epsilon_i + Adj_i \tag{2}$$

where $i = 1, 2, 3, ...$ corresponds to the $i$th rating entry. Therefore, $R_i$ is the $i$th movie rating. $\mu$, $M_i$, $U_i$, $G_i$, $\epsilon_i$ denote the mean rating, movie bias, user bias, genres bias and independent errors sampled from the same distribution centered at 0 respectively that are related to the $i$th entry. Please note $M_i$ can be identical to $M_j$ even if $i \neq j$ as the $i$th and $j$th entries as long as these entries be related to the same movie. The same applies to $U_i$ and $G_i$. $Adj_i$ is the adjustment term keeps the sum within the range of 0 to 5, and is defined as:

$$Adj_i = \begin{cases} 5 - (\hat{\mu} + \hat{M}_i + \hat{U}_i + \hat{G}_i), & \text{if } (\hat{\mu} + \hat{M}_i + \hat{U}_i + \hat{G}_i) > 5 \\ 0, & \text{if } 0 \leq (\hat{\mu} + \hat{M}_i + \hat{U}_i + \hat{G}_i) \leq 5 \\ -(\hat{\mu} + \hat{M}_i + \hat{U}_i + \hat{G}_i), & \text{if } (\hat{\mu} + \hat{M}_i + \hat{U}_i + \hat{G}_i) < 0 \end{cases}$$

with the hat $\hat{}$, similar to the $\hat{R}$ above, deontes the estimate of such variables.

In the edx file, the genre of a movie is usually a combination of several sub-genres. For example, the genure for the movie Stargate (1994) is Action|Adventure|Sci-Fi, and the genre of the movie Star Trek: Generations (1994) is Action|Adventure|Drama|Sci-Fi. In this analysis, we consider each combination as a unique genre and so two genres are considered as different as long as their sub-genres combination are not identical. For example, we consider Action|Adventure|Sci-Fi is a completely different genre to Action|Adventure|Drama|Sci-Fi.

Therefore, the predicted $\hat{R}_i$ is provided by the following formula:

$$\hat{R}_i = \hat{\mu} + \hat{M}_i + \hat{U}_i + \hat{G}_i + Adj_i \tag{3}$$

Running this recommendation system against the valdiation dataset, the resulting RMSE is 0.8647676, which is lower than the threshold 0.86499 mentioned above. Therefore, this proejct is a certain success.

## Methods

We will explain each of the steps in the recommendtion model creation process in the following sub-section:

- Creating edx and valdiation file (codes provided by instructors)
- Preprocessing
- Calibrating Mean
- Calibrating Movie Bias
- Calibrating User Bias
- Calibrating Genres Bias
- Creating Wrapper function and Making Adjustment

### Creating edx and validation file

Codes to generate the edx and validation file are provided by instructor and we do not repeat it in the pdf report.

### Preprocessing

In the edx file, the class of userId and movieId are numeric and genres is char. However, these should be categorical features rather than numerical features. Therefore, we should change their classes to factors with the following codes. We will make all these changes on the file edx2, which is a duplicate of the file edx. The file edx is created by the instrutor, and we will keep it unchanged.

```
edx2<-edx
edx2$movieId<-as.factor(edx2$movieId)
edx2$userId<-as.factor(edx2$userId)
edx2$genres<-as.factor(edx2$genres)
```

### Calibrating Mean

In this step, we will find out the mean of all ratings (i.e. $\mu$ mentioned in Equation (2) above). We will use m to keep this value in our program.

```
m<-mean(edx2$rating)
m
```

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

### Calibrating Movie Bias

Next, we will estimate the movie bias, which is the impact of rating due to the movie and the term $M_j$ in Equation (2) above, by the following:

$$\hat{M}_j = \frac{\sum_{j \in J} R_j - \mu}{\#J}$$

where $J$ denotes the set of entries that have the same movie as entry $j$..

In R, we will use the following code to generate the data frame b_m, which will contain the movieID and the corresponding bias.

```
b_m<-edx2%>%group_by(movieId)%>% summarise(b_m=mean(rating)-m)
head(b_m)
```

```
## # A tibble: 6 x 2
##   movieId    b_m
##   <fct>    <dbl>
## 1 1        0.415
## 2 2       -0.307
## 3 3       -0.365
## 4 4       -0.648
## 5 5       -0.444
## 6 6        0.303
```

### Calibrating User Bias

Then we will estimate the user bias, which is the impact of rating due to the user and the term $U_k$ in Equation (2) above, by the following:

$$\hat{U_k} = \frac{\sum_{k \in K} R_k - \mu - M_k}{\#K}$$

where $K$ denotes the set of entries that have the same user as entry $k$. In R, we will use the following code to generate the data frame b_u, which will contain the userId and the corresponding bias.

```
b_u<-left_join(edx2, b_m)%>%group_by(userId)%>% summarise(b_u=mean(rating-m-b_m))
head(b_u)
```

```
## # A tibble: 6 x 2
##   userId     b_u
##   <fct>    <dbl>
## 1 1       1.68
## 2 2      -0.236
## 3 3       0.264
## 4 4       0.652
## 5 5       0.0853
## 6 6       0.346
```

### Calibrating Genres Bias

Now, we will estimate the genres bias, which is the impact of rating due to the user and the term $G_l$ in Equation (2) above), by the following:

$$\hat{G_l} = \frac{\sum_{l \in L} R_l - \mu - M_l - U_l}{\#L}$$

where $L$ denotes the set of entries that have the same genres as entry $l$.

In R, we will use the following code to generate the data frame b_g, which will contains the genres and the corresponding bias.

```
b_g<-left_join(left_join(edx2, b_m),b_u) %>%
  group_by(genres)%>% summarise(b_g=mean(rating-m-b_m-b_u))
head(b_g)
```

```
## # A tibble: 6 x 2
##   genres                                                b_g
##   <fct>                                               <dbl>
## 1 (no genres listed)                                 0.232
## 2 Action                                            -0.0345
## 3 Action|Adventure                                  -0.0131
## 4 Action|Adventure|Animation|Children|Comedy         0.0109
## 5 Action|Adventure|Animation|Children|Comedy|Fantasy -0.0199
## 6 Action|Adventure|Animation|Children|Comedy|IMAX     0.0139
```

### Creating Wrapper Function and Making Adjustments

Now, we have calibrated all the parameters needed to make the predictions based on Equation (2). However, we still need to make a final step to create a function so that any user can obtain the predition output by only inputing a dataframe similar to edx. The function will also incorporate the *Adj* term. The following are the codes for such a function rating_p:

```
rating_p <- function(movies) {
  movies$movieId<-as.factor(movies$movieId) #change movieID from numeric to factor
  movies$userId<-as.factor(movies$userId) #change userID from numeric to factor
  movies$genres<-as.factor(movies$genres) #change genres from numeric to factor
  left_join(left_join(left_join(movies, b_g),b_u),b_m)%>% #combine files
    #calculate/cap/floor the results
  mutate(p0=m+b_m+b_u+b_g, p1=ifelse(p0>5, 5, p0), p=ifelse(p1<0, 0, p1))%>%
    pull(p)
}
```

The function above makes use of the parameters m, the dataframe b_g, b_u and b_m which we calibrated previously using the edx dataset. These parameters will remain constant and will not be affected by the input database.

The output of this function rating_p is a vector containing the predicted movie ratings correspdoning to the entries in the input file. For example, if we want to calculate p as the predicted ratings for the entries in the training edx file, we can use the following code:

```
p<-rating_p(edx) #calculate the predicted ratings
```

For further illustration, we can calculate the RMSE of the predicted ratings against the original ratings in the training edx file:

```
RMSE(p, edx$rating)  #calcualte RMSE between the predicted and actual rating of edx```
```

```
## [1] 0.8561992
```

## Results

In this section, we will use the validation file created by the instructor to test against our model. The codes for creating the prediction are:

```
p_val<-rating_p(validation) #calculate the predicted ratings based on the validation file
```

And the codes below will calculate the RMSE for the predicted ratings against the ratings in the validation file:

```
RMSE(p_val, validation$rating) #calculate RMSE of validation predicted and actual ratings
```

```
## [1] 0.8647678
```

This figure is lower than the threshold 0.86499 mentioned above.

## Conclusion and Future Works

We have just created a great recommendation system. The recommendation system takes into account the average ratings of all entries, movie bias, user bias and genre bias. We also adjust the result by ensuirng the predicted values stay within the range of 0 and 5. We test the system using the validation file created by the instructor, and the resulting RMSE between the predicted rating and actual rating is 0.8647678, which is below the threshold 0.86499.

I want to take this opportunity to thank Professor Rafael Irizarry for organising this great course. I have learnt tremendously from this. Equally important, I would like to thank all the staff instructors, who have provided tremendous help and insight in the discussion board and make this course even greater.

Finally, I would like to provide some recommendations for future works. There are a few interesting pattern I have identified, but for the time being I do not have time or the computation power to explore them further.

### Genres

Intutitively, each rating provider should have his or her favorite genre. Therefore, the recommendation system can take into account a bias based on the combination of user and genre.

In addition, the genre used in this recommendation system does not take into account the sub-genre category. For example, we consider Action|Adventure|Sci-Fi is a completely different genre to Action|Adventure|Drama|Sci-Fi. In reality, these two genres should be closely related as their sub-genre categories are closely related to each other. A model that is based on such sub-category may enhance the performance. However, such computation will take a lot of computation power. As an exmaple, my computer cannot finish the exeution of following code:

```
edx %>% separate_rows(genres, sep = "\\|") %>%
  group_by(genres) %>%
  summarize(R= mean(rating))
```

Therefore, this research can only be completed by people with much more computing power.

### Rating Time

It appears there are some bias for movies that were rated in the year 1995 or 1996, when this movie rating system just starts. The bias can be seen from the folowing codes:

```
library(lubridate)
edx2<-edx2%>%mutate(year= year(as_datetime(timestamp)))
left_join(left_join(left_join(edx2, b_m),b_u),b_g) %>% group_by(year)%>%
  summarise(b_t=mean(rating-m-b_m-b_u-b_g))
```

```
## # A tibble: 15 x 2
##     year       b_t
##    <dbl>     <dbl>
##  1  1995  0.226
##  2  1996  0.0135
##  3  1997  0.00595
##  4  1998  0.000586
##  5  1999  0.00210
##  6  2000  0.000406
##  7  2001 -0.000337
##  8  2002 -0.00439
##  9  2003  0.00691
## 10  2004 -0.00569
## 11  2005 -0.00274
## 12  2006 -0.00514
```

```
## 13  2007 -0.00689
## 14  2008 -0.00618
## 15  2009  0.00167
```

Intiutively, it is understable there are such bias. When the movie rating system just starts, most likely the users will start to rate their favorite movies first and then rate their less favorite movies later. Naturally, people give higher ratings to their favorite movies, resulting in a positive bias for the early years.

This bias can potentially be incorporated in future work to produce a better recommendation system.