edx Capstone - Movielens

Saigopal

5/7/2021

1. Introduction

This report is prepared for the edx Data Science: Capstone project - the final course in HarvardX's Data Science Professional Certificate series. The assignment is based on the Netflix challenge of improving a recommendation system.

Recommendation system is one where User's rate their experience on a specific product, which could be a movie or food or any products, to express their satisfaction. It is usually measured in ordered scale with high value indicating satisfaction and are used by several industry [Netflix, Amazon, Youtube, Food delivery industries etc..] for targeted advertisements of their products to customers. It also improves user's satisfaction.

The goal of this assignment is to develop a model that reduce the error for predicted ratings. The script to download the data was provided in the project introduction section of the course. The script also included some preliminary wrangling of the data and splitting of the data into edx and validation datasets.

The edx dataset had nine million rows/observations. There were six column variables, including the outcome variable "rating". The other five variables were userId, movieId, title of the movie, genre(s) of the movie and timestamp in seconds. There were no missing values in the dataset.

```
dim(edx)
```

```
[1] 9000055       6
```

```
head(edx, n =4)
```

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

```
any(is.na(edx))
```

```
[1] FALSE
```

Key steps done in this analysis includes

- Creating new variables from the data and exploring their relation with outcome
- Hierarchical Clustering using correlation distance and grouping of movies
- Least square estimation of rating with movie clusters and other predictors identified along with movie and user bias

These are explained in detail section 2 and 3 respectively.

2. Methods and Analysis

The unit of observation in this analysis is a user/movie pair.

The number of unique user's and movie's in the edx dataset is shown below. The other two tables displays the number of user's and movie's in the bottom five, middle ninety and top five in terms of number of ratings, the total number of ratings between them and the proportion of observations in the data they represent.

```
Number of User's = 69878
```

```
Number of Movie's =  10677
```

```
User's Distribution in dataset
```

Group	number	total_ratings	percentage_dataset
Bottom five	3470	62484	0.0069426
Middle ninety	62934	6229173	0.6921261
Top five	3474	2708398	0.3009313

```
Movies's Distribution in dataset
```

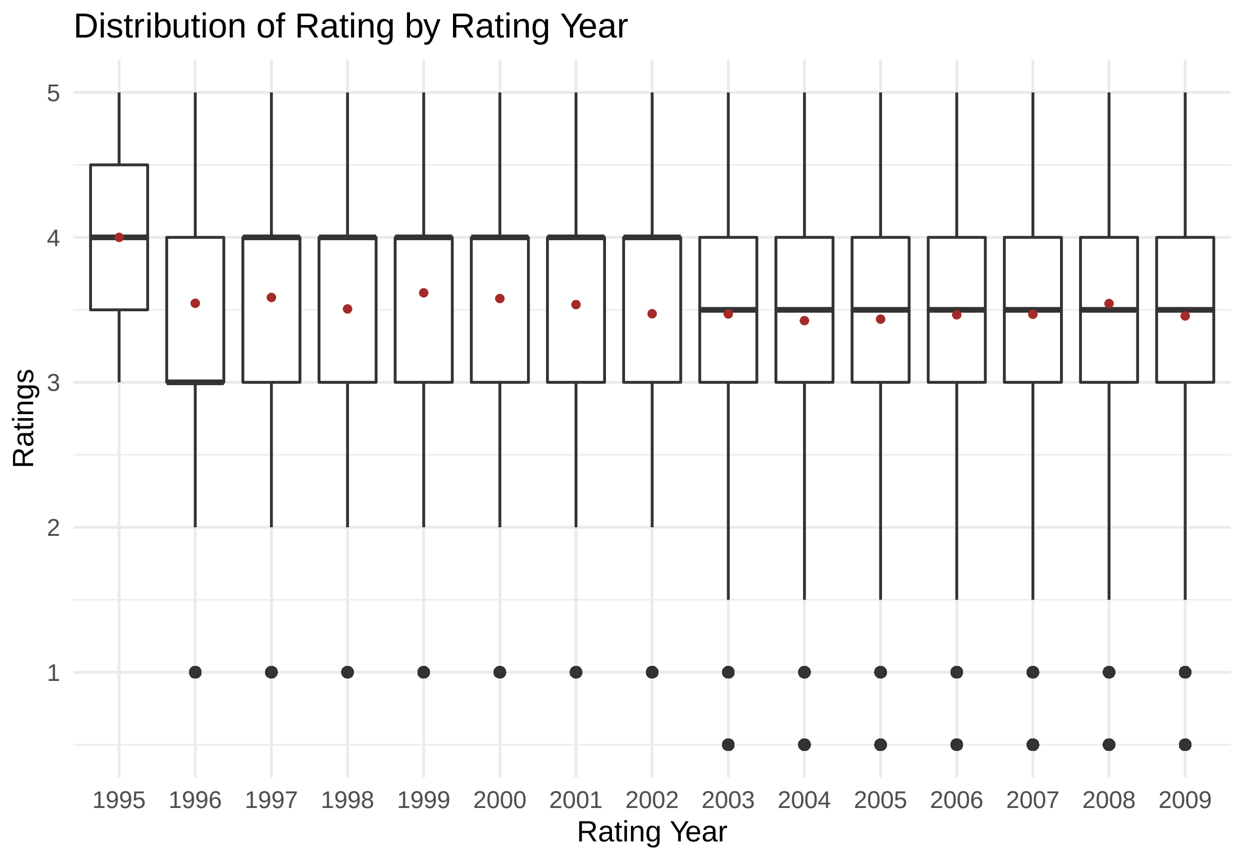
Group	number	total_ratings	percentage_dataset
Bottom five	397	787	0.0000874
Middle ninety	9746	4347274	0.4830275
Top five	534	4651994	0.5168851

The distribution of user's and movie's in the dataset is heavily skewed with less than 5% of the movies contributing more than 50% of the data and 5% of the users making up 30% of the data. This 5% of (534) movies will be included in hierarchical clustering analysis in Section 2.2.2.

2.1 Creating new variables form timestamp column

Timestamp were stored in seconds since epoch - 1 January 1970. Two variables were created using this, the year a movie was rated and the day of the week the movie was rated. This was done using the year and as_datetime function's of the lubridate package and weekday function of base R.

Distribution of rating by year is shown in the figure (dot inside the box is mean rating) and the average rating by day of the week is summarized in table below.



day_week	Number	Mean	SD
Monday	1410155	3.516998	1.059797
Tuesday	1431735	3.510664	1.061510
Wednesday	1332505	3.501416	1.063426
Thursday	1241493	3.500727	1.059157
Friday	1232250	3.512657	1.061754
Saturday	1125445	3.528958	1.058779
Sunday	1226472	3.517915	1.057107

There are only 2 observations for the year 1995. The average rating across years do not appear to change much. For average rating by day of week, there is a slight increase in Saturday and a decrease in Wednesday and Thursday. But these difference are very small about 0.01.

Both these variables are not considered for inclusion in the model.

2.2 Creating new variables from title column

Several new variables were created using this column. The year a movie was released and clusters of movies with similar rating pattern.

2.2.1 Year a Movie released

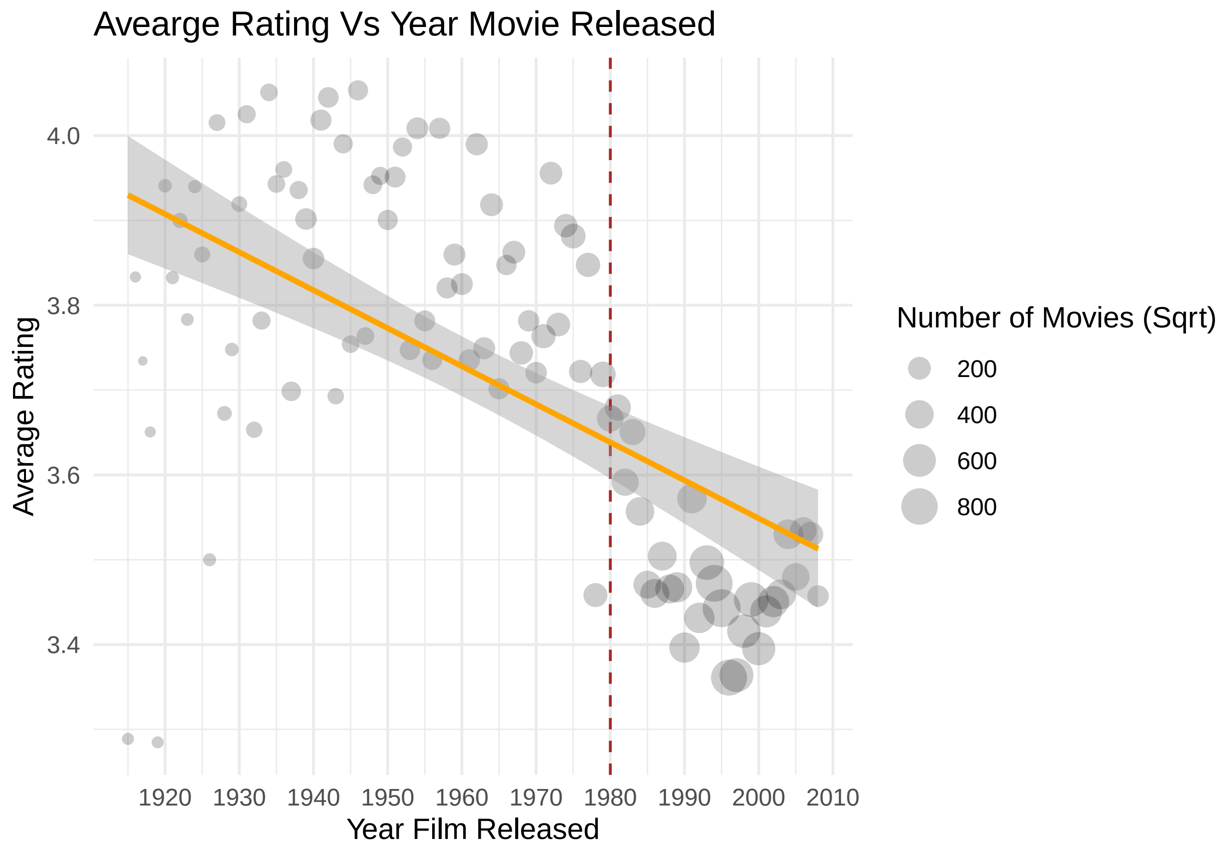
The year a movie released was captured in the title column at the end of the title inside parenthesis. To extract these, the regex pattern shown below was used.

```
film_year_pattern<- "\\(19[0-9][0-9]\\)|\\(20[0-9][0-9]\\)"
```

This regex is explained as a string that must start and end with parenthesis. The first two character inside the parenthesis can be 19 or 20 and the next two characters can be any number. This was extracted using str_extract function and the parenthesis removed using str_remove and the character converted to numeric.

The average rating of films by the year they released was explored and there was no linear relationship observed. The movies released after the year 1980 (inclusive) appeared to have a lower average than those released before (dots on right side of the dashed line - See Figure below). A new binary variable (film_1980) was created to captures this grouping (0 = movies before 1980 and 1 = after).

The mean rating in these two groups is shown below the figure. This binary variable will be included in the regression model.



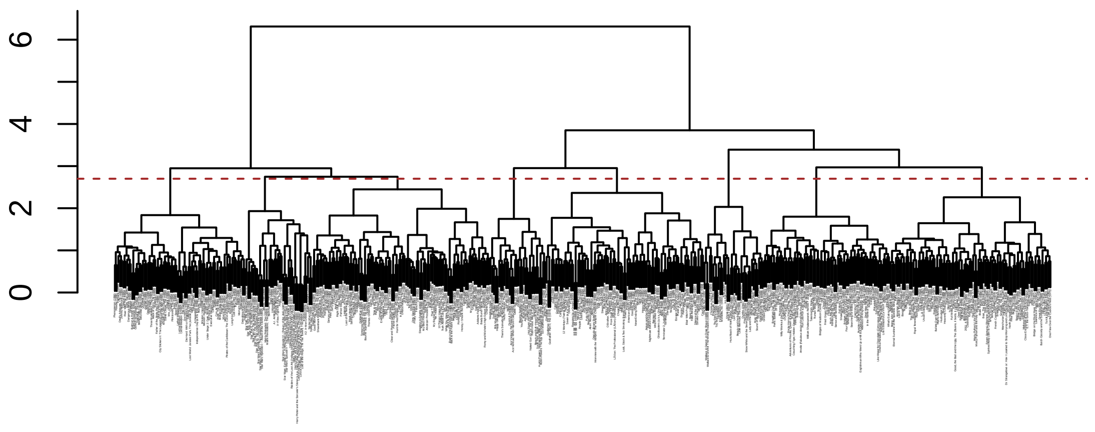
```
Mean rating: Movies before 1980 =  3.81634

Mean rating: Movies after 1980 =  3.456153
```

2.2.2 Hierarchical Clustering analysis

Clustering analysis was done to identify movie clusters to be included in the regression model. Correlation distance was calculated using the Dist function of the **amap** package and for clustering, the hclust function was used with *Ward method*.

Ward's minimum variance method minimize the within the group variation https://uc-r.github.io/hc_clustering, thereby the cluster would be movies with very similar rating pattern. The movies which were rated many times (> 95th percentile) and users who have rated more than 50 times are used for this analysis.

The tree was cut by visualization of the dendrogram at a height of 2.7 forming eight movie groups (See Figure below)



```
movies_clusters
  1   2   3   4   5   6   7   8 
101  92  90  72  38  32  75  34 
```

Patterns in the eight group can be identified by movie names. The groupings which were discussed in the course can be seen in the groups. Some examples below.

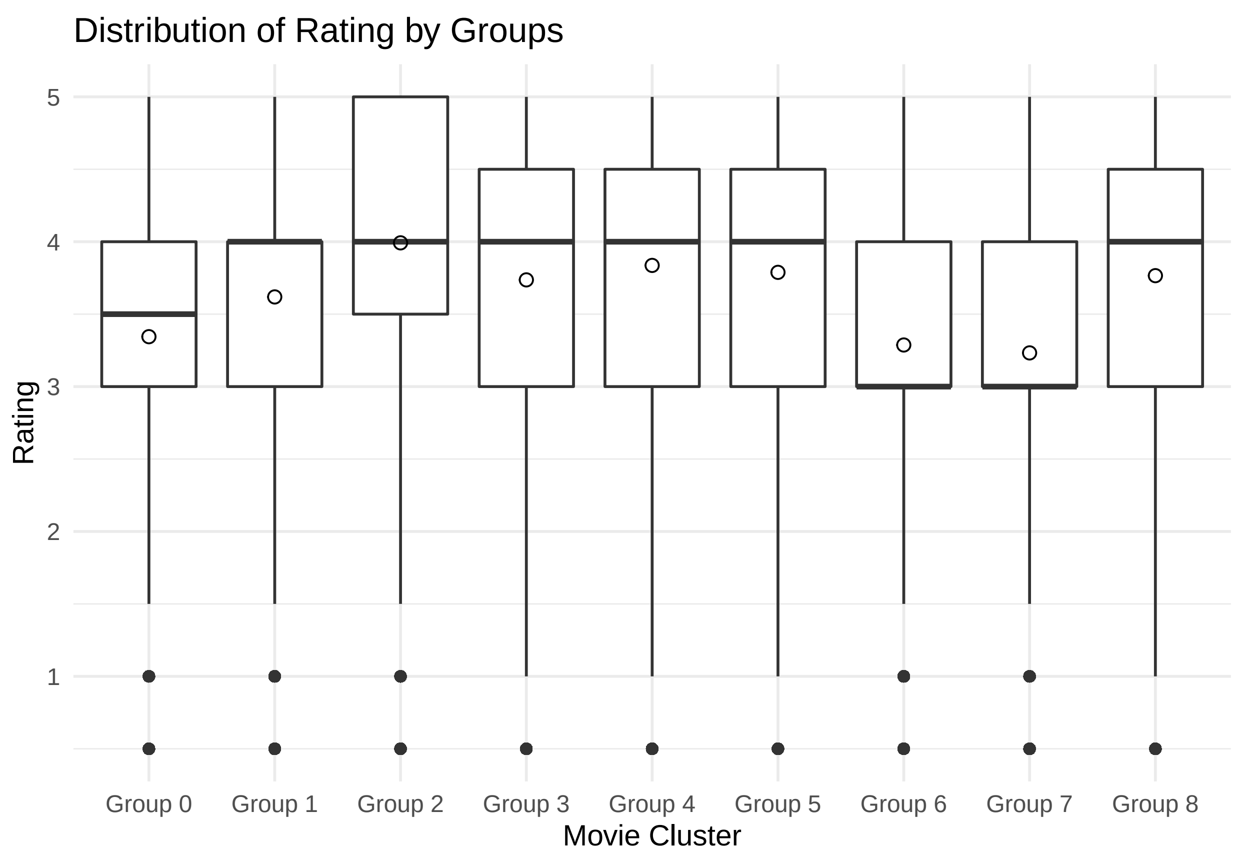
```
g3[str_detect(g3, "Donnie Br|Godfa|Goodfe|Carlito|Heat|Casino")]
```

```
[1] "Carlito's Way (1993)"            "Casino (1995)"                  
[3] "Donnie Brasco (1997)"            "Godfather, The (1972)"          
[5] "Godfather: Part II, The (1974)"  "Godfather: Part III, The (1990)"
[7] "Goodfellas (1990)"               "Heat (1995)"                    
```

```
g5[c(22,29,36,10,6,12)]
```

```
[1] "Star Trek II: The Wrath of Khan (1982)"                   
[2] "Star Wars: Episode I - The Phantom Menace (1999)"         
[3] "X-Files: Fight the Future, The (1998)"                    
[4] "Lord of the Rings: The Fellowship of the Ring, The (2001)"
[5] "Harry Potter and the Prisoner of Azkaban (2004)"          
[6] "Lord of the Rings: The Two Towers, The (2002)"            
```

The distribution of the ratings across these groups are displayed below.



These eight groups were binarized (0s = No and 1s = Yes) for model building and all eight are included in the model. The reason for 0 and 1 are explained in Section 3.

2.3 Creating new variables from genre column

All the unique genre were first identified by using the str_split function of the stringr package and unique function of base R. There were 19 unique genres and a element no genres listed.

```
unique(unlist(str_split(edx$genres, pattern = "\\|")))
```

```
 [1] "Comedy"             "Romance"            "Action"
 [4] "Crime"              "Thriller"           "Drama"
 [7] "Sci-Fi"             "Adventure"          "Children"
[10] "Fantasy"            "War"                "Animation"
[13] "Musical"            "Western"            "Mystery"
[16] "Film-Noir"          "Horror"             "Documentary"
[19] "IMAX"               "(no genres listed)"
```

Binary variables were created for each of the nineteen genres. The average rating within these genres are tabulated below.

Genre	No	Yes	Difference
Film-Noir	3.505803	4.011625	0.5058217
War	3.496307	3.780813	0.2845055

Genre	No	Yes	Difference
Drama	3.389041	3.673131	0.2840901
Documentary	3.509633	3.783487	0.2738536
Horror	3.532660	3.269815	-0.2628449
IMAX	3.512233	3.767693	0.2554604
Crime	3.485909	3.665925	0.1800167
Mystery	3.501375	3.677001	0.1756265
Sci-Fi	3.532905	3.395743	-0.1371618
Action	3.548674	3.421405	-0.1272690
Comedy	3.561473	3.436908	-0.1245654
Children	3.520839	3.418715	-0.1021238
Animation	3.507637	3.600644	0.0930062
Musical	3.509895	3.563305	0.0534096
Romance	3.502752	3.553813	0.0510619
Western	3.511531	3.555918	0.0443867
Adventure	3.517559	3.493544	-0.0240148
Fantasy	3.513671	3.501946	-0.0117249
Thriller	3.514134	3.507676	-0.0064586

Although some genres had difference of less than 0.1, all the genre are included as predictors in the model. The reason for this is most movies were categorized in multiple genre's and a additive relationship is assumed in the model to be built.

3. Results

A linear regression model is chosen for this assignment. This model *builds on the model* described in the eighth course. Key difference between the two models are listed below.

- Instead of **Naive Mu**, the Least Square Estimate [**LSE**] **Mu** is used
- The predictors for the linear regression are film_1980s (the film released before 1980 or after), movie cluster/group 1 through 8 and all of the 19 genres, meaning the *year, genre and the cluster effect* are added to the model

Matrix algebra is used to solve the least square equation, as it is much quicker than lm function. The connection between matrix algebra and LSE is shown in a small example for reviewer's not familiar with matrix algebra.

```
dat<- data.frame(x1 = c(rep("Yes", 4), rep("No", 4)), y =rnorm(8))
dat
   x1          y
1 Yes  0.5545116
2 Yes  0.3662907
3 Yes  0.3711162
4 Yes -1.3269716
5  No  1.3352418
6  No  0.8970502
7  No -0.9556451
8  No  0.5022905

lm(y ~ x1, data = dat)$coeff
```

```
(Intercept)       x1Yes
  0.4447343  -0.4534976

y<- dat$y

X<- dat %>% mutate(x1 = ifelse(x1 == "Yes", 1, 0), intercept = 1) %>%
  select(x1, intercept) %>% as.matrix()
X
     x1 intercept
[1,]  1         1
[2,]  1         1
[3,]  1         1
[4,]  1         1
[5,]  0         1
[6,]  0         1
[7,]  0         1
[8,]  0         1

betahats<- solve(crossprod(X)) %*% crossprod(X, y)

betahats
                [,1]
x1        -0.4534976
intercept  0.4447343
remove(X, y, dat, betahats)
```

The coefficients are the same. The order of the intercept do not matter. For prediction, the test set will be converted into a matrix containing the predictors to be included in the model with intercept (a column of 1's) and multiplied by the betahats. The R scripts for the LSE model are printed in the document for ease of review.

3.1 Model building and evaluation

The **edx dataset** was split into train and test set. The number of observations in the test test was 900 thousand. The train set had 8 million observations. The reason for this choice was to train the model with a sufficiently large number so that the standard error will be negligible and the model metric can be evaluated in a dataset with number similar to the final validation set.

The Residual Mean Square Error (RMSE) was used to evaluate model performance. The Naive Mu model explained in course is compared along to gauge the performance of this LSE model to be used here.

```
dim(train_set) # number of rows and number of columns
```

```
[1] 8100056      38
```

```
dim(test_set)
```

```
[1] 899999     38
```

3.2 RMSE without Movie and User Bias

The RMSE of the Naive Mu model is presented below.

```
                  Model    RMSE
1 Null model (Naive Mu) 1.06053
```

The RMSE of the LSE model is computed by the following script.

```
true_ratings<- test_set$rating

y<- train_set$rating

X<- train_set %>% mutate(intercept = 1) %>%
  select(10, 12:39) %>% as.matrix() # column 10 is film_year, 12 to 38 are
                                    # movie cluster and genres, 39 intercept

test_matrix<- test_set %>% mutate(intercept = 1) %>%
  select(10, 12:39) %>% as.matrix()

betahats<- solve(crossprod(X)) %*% crossprod(X, y)

predicted_ratings<- test_matrix %*% betahats

rmse<- RMSE(true_ratings = true_ratings, predicted_ratings = predicted_ratings)

models<- rbind(models, c("LSE Mu", rmse))

remove(X, test_matrix, rmse, predicted_ratings)

models
```

```
                  Model             RMSE
1 Null model (Naive Mu) 1.06052990054772
2                LSE Mu 1.01051870262163
```

A big difference in RMSE between models, around 0.05. The 10 largest coefficient/effects are listed below.

```
            ceoffs_name   estimate
intercept     intercept  3.5552840
group5           group5  0.6326968
documentary documentary  0.5097206
group3           group3  0.5006840
group2           group2  0.4695682
group4           group4  0.4694517
group8           group8  0.4631394
imax               imax  0.3631250
film_1980     film_1980 -0.3607968
group1           group1  0.3108931
```

These coefficients can be interpreted. Examples- for a movie with no genre, released before 1980 and is not part of any of the eight group, the intercept will be the predicted rating. For a IMAX movie in group 1 released after 1980, the predicted rating would be: intercept + 0.36312 (IMAX coeff) + 0.31089 (group1 coeff) - 0.3607968 (film_1980 coeff).

3.3 RMSE with Movie and User Bias

In this section, the movie bias (b_i) and user bias (b_u) are added to both models. The RMSE of the Naive Mu model with Movie and User bias is shown below.

```
                              Model              RMSE
1              Null model (Naive Mu)  1.06052990054772
2                             LSE Mu  1.01051870262163
3 Naive Mu + Movie bias + User bias 0.865115016060473
```

A big drop is seen after adding the movie and user bias. The predicted rating and RMSE for the LSE model was calculated next using the script below.

```
train_mu<- mean(train_set$rating)

train_bi <- train_set %>% select(userId, movieId, rating) %>%
  group_by(movieId) %>% summarise(b_i = mean(rating - train_mu))

train_bu<- train_set %>% select(userId, movieId, rating) %>%
  left_join(train_bi, by="movieId") %>%
  group_by(userId) %>% summarise(b_u = mean(rating - train_mu - b_i))


X<- train_set %>% group_by(movieId) %>%
  mutate(b_i = mean(rating - train_mu)) %>% ungroup() %>%
  group_by(userId) %>% mutate(b_u = mean(rating - train_mu - b_i)) %>%
  ungroup() %>%
  mutate(intercept = 1) %>%
  select(10, 12:41) %>%  # column 10 is film_year, 12 to 38 movie groups
                         # and genres, 39, 40 and 41 - b_i, b_u and intercept
  as.matrix()

test_matrix<- test_set %>%
  left_join(train_bi, by = "movieId") %>%
  left_join(train_bu, by = "userId") %>%
  mutate(intercept = 1) %>%
  select(10, 12:41) %>%
  as.matrix()

betahats<- solve(crossprod(X)) %*% crossprod(X, y)

predicted_ratings<- test_matrix %*% betahats

rmse<- RMSE(true_ratings = true_ratings, predicted_ratings = predicted_ratings)

models<- rbind(models, c("LSE Mu + Movie Bias + User Bias", rmse))

remove(X, test_matrix, predicted_ratings, rmse,train_bi,train_bu)

models
```

```
                              Model              RMSE
1              Null model (Naive Mu)  1.06052990054772
```

```
2                            LSE Mu  1.01051870262163
3 Naive Mu + Movie bias + User bias 0.865115016060473
4   LSE Mu + Movie Bias + User Bias 0.864736107580122
```

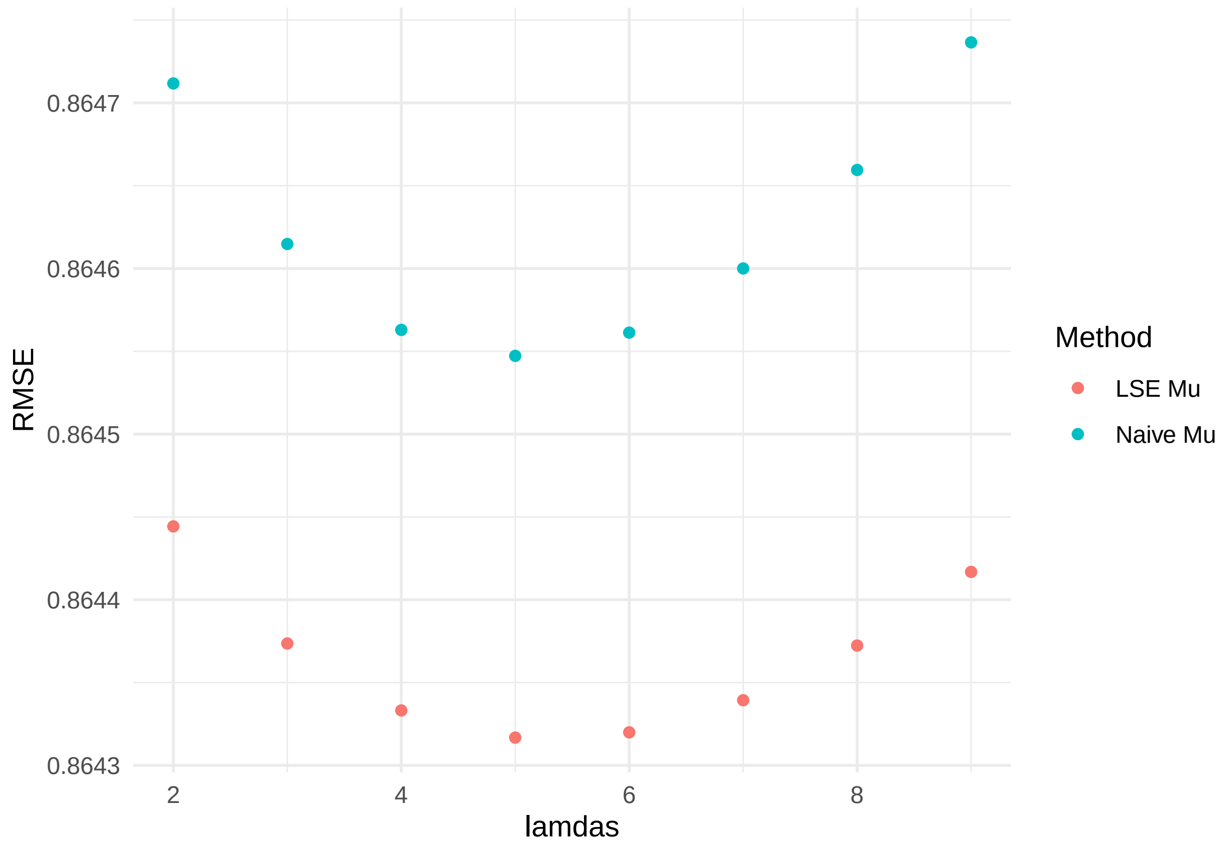
There is a reduction in LSE model also, but the difference between the naive and LSE Mu model's reduces. This is not unexpected as the user's and movie's own average rating is the biggest determinant of predicted rating. This can be appreciated by looking at the betahats (estimates) below. The user and movie bias has the biggest coefficients and that of other predictors as drastically reduced when compared to previous models without bias. Nevertheless, the LSE model performed better than the naive model and RMSE is also less than the targeted RMSE of 0.86490.

```
            ceoffs_name    estimate
intercept     intercept  3.53274327
b_i                 b_i  1.00896895
b_u                 b_u  1.00390911
group7           group7 -0.05448204
documentary documentary  0.05232650
group1           group1 -0.04446705
group6           group6 -0.03311726
imax               imax -0.03201916
children       children -0.03084404
```

The movie groups and genre will only add/reduce around 0.05 to the predicted rating and the b_i and b_u are the main determinant of predicted rating.

3.4 Regularization

Regularization was done on both the models using the penalty term (lamda's) from two to nine, increasing by one and the results are plotted.



The residual mean square error of both model reduce equally and the one using LSE Mu perform better than the Naive Mu. The best lamda was 5.

3.5 Training the final model using enire edx dataset

The linear regression model with predictors - film_1980, movie groups 1 through 8, all the genres, regularized user and movie bias with the penalty term (lamda = 5) will be final model for testing with validation set.

The betahats, movie bias (b_i) and the user's bias (b_u) are calculated on the entire edx dataset.

3.7 Data wrangling of validation set and testing of the final model

The column variables created for edx dataset will be created for the validation set and the **Final RMSE** on the validation set is computed in this section.

```
validation_matrix<- validation %>%
  left_join(edx_b_i, by = "movieId") %>%
  left_join(edx_b_u, by = "userId") %>%
  mutate(intercept = 1)  %>%
  select(10, 12:41) %>%
  as.matrix()

colnames(validation_matrix)
```

```
 [1] "film_1980"   "group1"      "group2"      "group3"      "group4"
 [6] "group5"      "group6"      "group7"      "group8"      "comedy"
[11] "romance"     "action"      "crime"       "thriller"    "drama"
[16] "sci_fi"      "adventure"   "children"    "fantasy"     "war"
[21] "animation"   "musical"     "western"     "mystery"     "film_noir"
[26] "horror"      "documentary" "imax"        "b_i"         "b_u"
[31] "intercept"
```

```r
predicted_ratings<- validation_matrix %*% edx_betahats

Final_RMSE<- RMSE(true_ratings = validation$rating, predicted_ratings = predicted_ratings)

Final_RMSE
```

```
[1] 0.8646087
```

4. Conclusion

The goal of this assignment was to build a model that reduces error for predicted rating. Recommendation system is a useful data to have for many industries and is allows for personalized user's experience.

For this assignment, several new features/variables were created from the data and their relationship with rating was studied and modeled. Linear regression with film year (movies before or after 1980), movie clusters identified by Hierarchical clustering analysis, 19 genre(s) and regularized movie and user bias was used as predictors. The final RMSE on the validation set was **0.8646087**.

Limitations

There are several limitations in this model. A few listed below

- The clustering analysis done here will not be perfect, as all user's did not rate all movies. The Dist function calculates correlation on complete pairwise observation, thereby reducing the number used in each correlation distance
- The user' s were not grouped for this model
- The model assumed a additive relationship between all the predictors and interaction, if any was not tested and accounted for
- There might be several other data structures this model did not factor in. Example - Do a user rating's decline after sometime etc..

Matrix factorization could have offset several of these limitation- it assigns a rating for movie for which a user has not given rating, thereby eliminating missing values. Matrix factorization followed by Principal Component analysis could have captured the resdiuals which represents both user and movies groupings.