# MovieLens

Kiran Joshi

June 9, 2020

## Overview

Predicting Movies and Product ratings are in high demand due to the additional revenue that can be generated by presenting a list of movies or products they could be interested in based on their historical views or purchases. In this challenge, we were expected to come up with a model that would train on a large data set (edx) and when predicted using the validation set (validation) would generate an error of no more than **0.86490** for maximum credit.

There are a lot of methods available to fit our model, many of them could not handle the large objects that were needed to analyze the raw data, and hence were rejected.The methodology chosen incorporated 4 significant effects that could produce a lower than expected error by an iterative process of introducing one potentially significant variable and studying its effect on the RMSE. The effects of the individual variables were then regularized to avoid any skewing of the ratings by outliers.

## Analysis

Lets load all the libraries required

Create the edx and validation data by getting it from the movielens site.

Initial inspection of the data set shows that there are no NA values to be cleaned up, and that there are no 0 rating that needs to be imputed as well

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

```
## [1] FALSE
```

```
any(edx$rating==0)
```

```
## [1] FALSE
```

### Data Cleaning

We can notice that the title of the movie contains two pieces of information which can be seperated out into two variables, title and the year the movie was released.

The timestamp is in the epoch format, and can easily be converted into a datetime if we decide to use that variable.

The genre variable contains all the genre that a movie belongs to seperated by "|" and we will not split it out as that complex genre themselves could be treated as unique.

Lets first seperate out the title and year from the edx and validation data frames.This introduces another variable year to the data set.

```
edx <- edx %>% mutate(year = as.numeric(substr(str_extract(title,"\\(\\d\\d\\d\\d\\)"),2,5)),
                      title = substr(title,1,regexpr("\\(\\d\\d\\d\\d\\)",title)-2))
```

```
validation <- validation %>%
              mutate(year = as.numeric(substr(str_extract(title,"\\(\\d\\d\\d\\d\\)"),2,5)),
                     title = substr(title,1,regexpr("\\(\\d\\d\\d\\d\\)",title)-2))
```

### Data Partition

Since we are not allowed to use the validation data set for testing our models, we will start off by creating a training set and test set from the edx data

```
#create test and train sets
set.seed(1,sample.kind = "Rounding")
test_index <- createDataPartition(y=edx$rating,times = 1,p = 0.2,list = FALSE) # test set = 20%
train_set <- edx %>% slice(-test_index)
temp <- edx %>% slice(test_index)
#cleanup
rm(test_index)
```

to make sure the test data has all the users and movies in the training set, we identify the ones that do not and remove it from the test set and move it back to the training set.

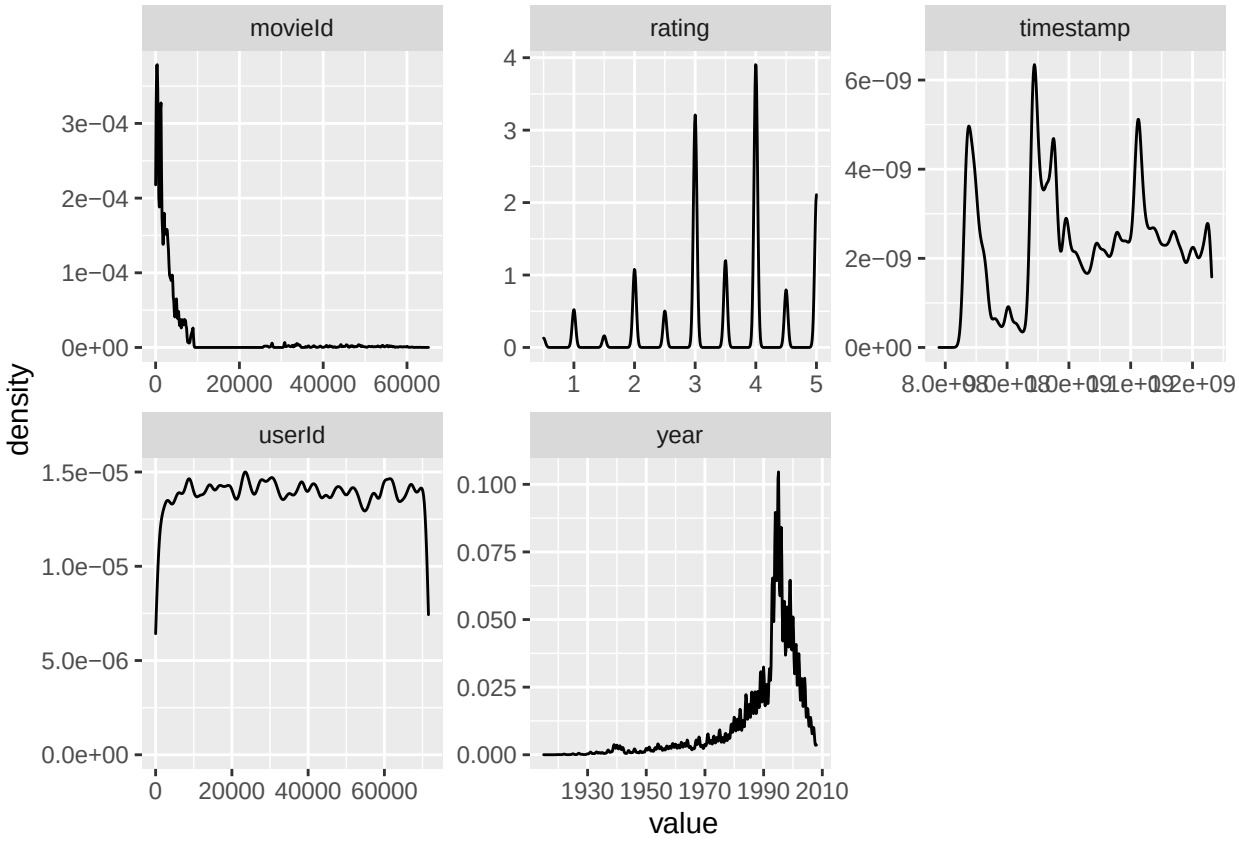
```
test_set <- temp %>% semi_join(train_set,by = "movieId","userId")
removed <- anti_join(temp, test_set)
train_set <- rbind(train_set, removed)
#cleanup
rm(temp,removed)
```

The data set is too large for my laptop and R to handle any kind of standard models to be fit using the caret train function on the whole training set. Hence we will try to introduce one variable at a time to predict the outcome using residual values to identify the bias.

### Exploration Plot

Lets plot the data for deeper understanding

```
train_set %>%
  keep(is.numeric) %>%                     # Keep only numeric columns
  gather() %>%                             # Convert to key-value pairs
  ggplot(aes(value)) +                     # Plot the values
  facet_wrap(~ key, scales = "free") +     # In separate panels
  geom_density()                           # Densities
```




**Observations:**

1. Some movies have lot of ratings, while the majority have very few ratings
2. Most of the movies have a 3 and 4 rating.
3. Certain periods of time seems to have more ratings than othgers, this could be the effect of some blockbuster movies.
4. Most of the ratings are from the period of 1970 till 2008 in the dataset peaking at around 1994/1995

### Average Rating Model

Lets start by making the assumption that the rating of a movie is just the average $\mu$ of all the movie ratings (from the rating plot) and variation of the movie ratings from one to another is explained by $\epsilon_{\mathbf{u,i}}$ , where $u$ is a specific user rating the movie $i$

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

By calculating the average $\mu$ of all the movie ratings, we get

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

```
## [1] 3.512478
```

lets calculate the RMSE for this method

```
rmse_mu <- RMSE(mu,test_set$rating)
rmse_mu
```

```
## [1] 1.059904
```

We need a table to hold the RMSE's as we go along, and we will store them in a table with the method used and the RMSE obtained that method when run against the test set to compare and contrast.

```
RMSE_table <- data.frame(method="Average Model",RMSE=rmse_mu)
RMSE_table %>% kable()
```

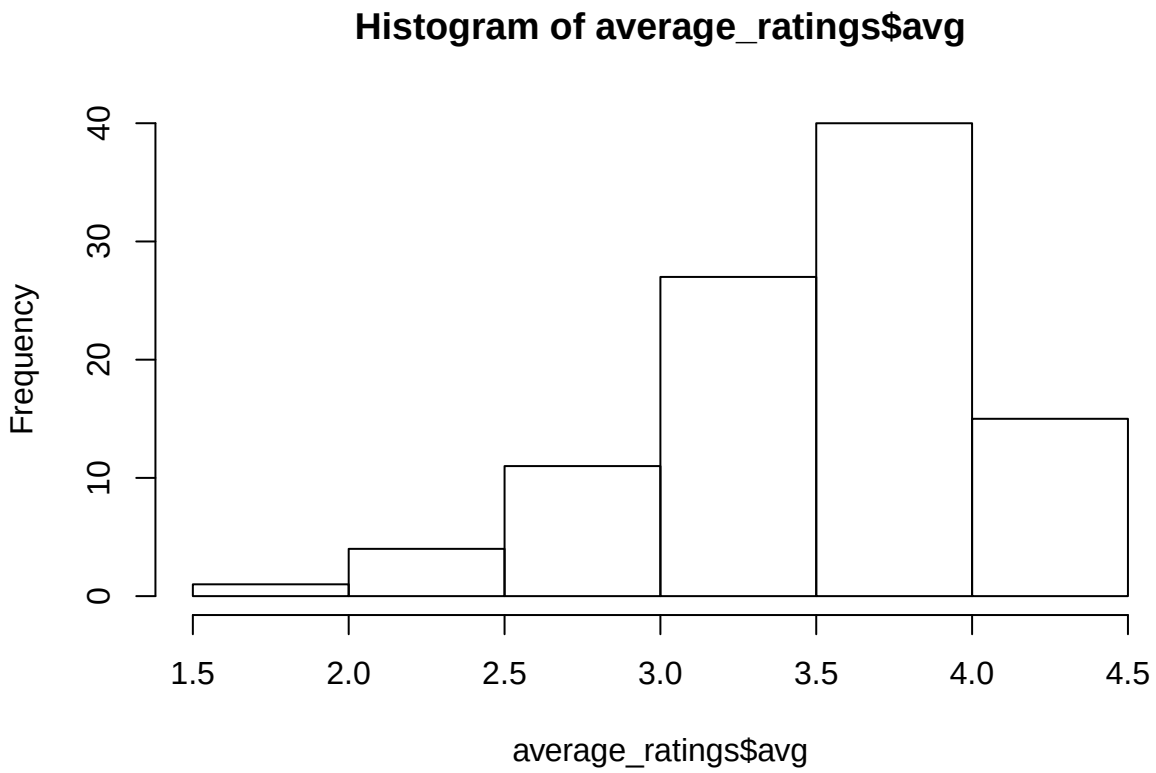
| method | RMSE |
|---|---|
| Average Model | 1.059904 |

As we can see, 1.05 is a large RMSE, and we will try to bring it down by identifying more effects and see if the error component $\epsilon_{u,i}$ can be reduced.

### Average and Movie bias Model

We will now explore the data provided further and see if there is movie bias that can be identified.In our observations, we saw that some movies $i$ got a lot of ratings than others (from the movieId exploration plot) which explains a movie effect. lets now sample a few movies and see if they are randomly distributed, so that we can estimate the bias by applying the central limit theorem.

```
set.seed(1,sample.kind = "Rounding")
movies <- data.frame(sample(train_set$movieId,100))
names(movies) <- "movieId"
average_ratings <- movies %>% left_join(train_set,by = "movieId") %>% group_by(movieId) %>%
                   summarize(avg = mean(rating))
hist(average_ratings$avg)
```



we observe that the distribution is almost normal centered around 3.5 - 4 with some movies getting lower ratings and some higher.

We will try to estimate the rating by introducing another effect $b_i$ to the residual which can reduce our RMSE like this

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

```
bi <- train_set %>% group_by(movieId) %>%  # grouping by the movies
      summarize(b_i = mean(rating - mu))    # mean of the residual
```

Lets predict our ratings with this model and calculate the RMSE with the movie bias factored in from the test set.

```
predict_mu_bi <-
  test_set %>% left_join(bi,by = "movieId") %>%
  mutate(mu_bi = mu + b_i) # get the movie bias and add it to average rating
```

Calculating the RMSE with this prediction which is the combination of the average and the movie bias

```
rmse_mu_bi <- RMSE(predict_mu_bi$mu_bi,test_set$rating)
rmse_mu_bi
```

```
## [1] 0.9437429
```

Lets add it to the table and view it.

```
RMSE_table <- bind_rows(RMSE_table,data.frame(method="Average and Movie bias Model",RMSE=rmse_mu_bi))
RMSE_table %>% kable()
```

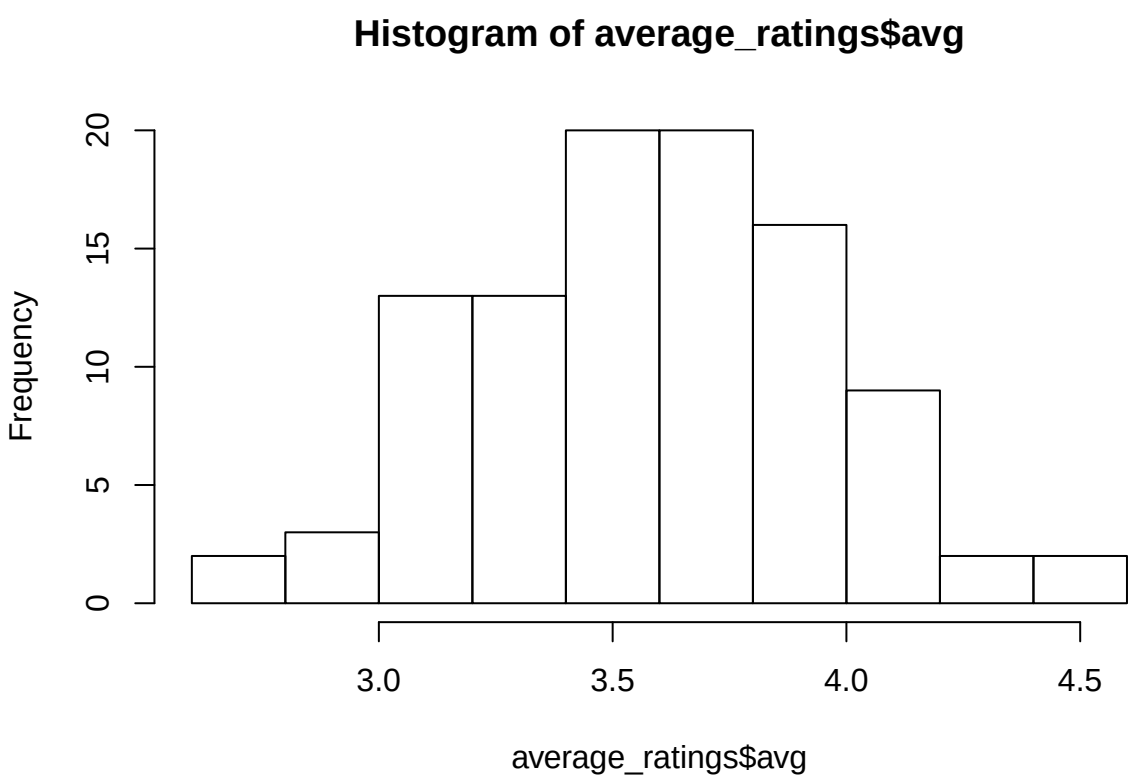
| method | RMSE |
|---|---|
| Average Model | 1.0599043 |
| Average and Movie bias Model | 0.9437429 |

That is a significant reduction in the RMSE, There could still be some other effects that could be identified. Let us consider this scenario, where a user rates specific movies higher than other users, this bias denoted by $b_u$ could be extracted from the residuals of the first two methods.

### Average, Movie bias and User bias Model

we see that the users typically have a similar density (from the exploratory plot userId) and can pick a subset of the users to see if there is a user bias, to do this lets randomly pick a 100 users and see if the distribution of their ratings follow a pattern.

```
set.seed(1,sample.kind = "Rounding")
users <- data.frame(sample(train_set$userId,100))
names(users) <- "userId"
average_ratings <- users %>% left_join(train_set,by = "userId") %>% group_by(userId) %>%
                   summarize(avg = mean(rating))
hist(average_ratings$avg)
```




As we can see, some users give movies low ratings, while some users give them a high rating. this shows the prevelance of a user bias

The prediction is now represented as below.

$\mathbf{Y_{u,i}} = \mu + \boldsymbol{b_i} + \boldsymbol{b_u} + \epsilon_{\mathbf{u,i}}$

we can use the similar modeling technique as done before to identify the user bias

```
bu <- train_set %>% left_join(bi,by = "movieId") %>% #join the movie bias table
  group_by(userId) %>%  # grouping by the users and then movies that they rated
  summarize(b_u = mean(rating - mu - b_i )) # mean of the residua
```

We can now apply the user bias to the model and predict the ratings of the movies and see if it makes a difference in the RMSE

```
predict_mu_bi_bu <-
  test_set %>% left_join(bi,by = "movieId") %>% # join test set and train set movie bias on movie id
  left_join(bu,by = "userId") %>% # join test set and train set user bias on user id
  mutate(mu_bi_bu = mu + b_u + b_i) # get the movie bias, user bias and add it to average rating
```

Calculating the RMSE with this prediction which is the combination of the average,the movie bias and user bias

```
rmse_mu_bi_bu <- RMSE(predict_mu_bi_bu$mu_bi_bu,test_set$rating)
rmse_mu_bi_bu
```

```
## [1] 0.8659319
```

```
RMSE_table <- bind_rows(RMSE_table,data.frame(method="Average, Movie bias and User bias Model",
                                              RMSE=rmse_mu_bi_bu))
RMSE_table %>% kable()
```

| method | RMSE |
|---|---|
| Average Model | 1.0599043 |
| Average and Movie bias Model | 0.9437429 |
| Average, Movie bias and User bias Model | 0.8659319 |

We have achieved significant improvement from 1.05 to about 0.8659.

We will now see if the genre has any bearing on the rating of the movies, and if it does, we will determine how we can fit a model with the residual values and its effect on the RMSE.

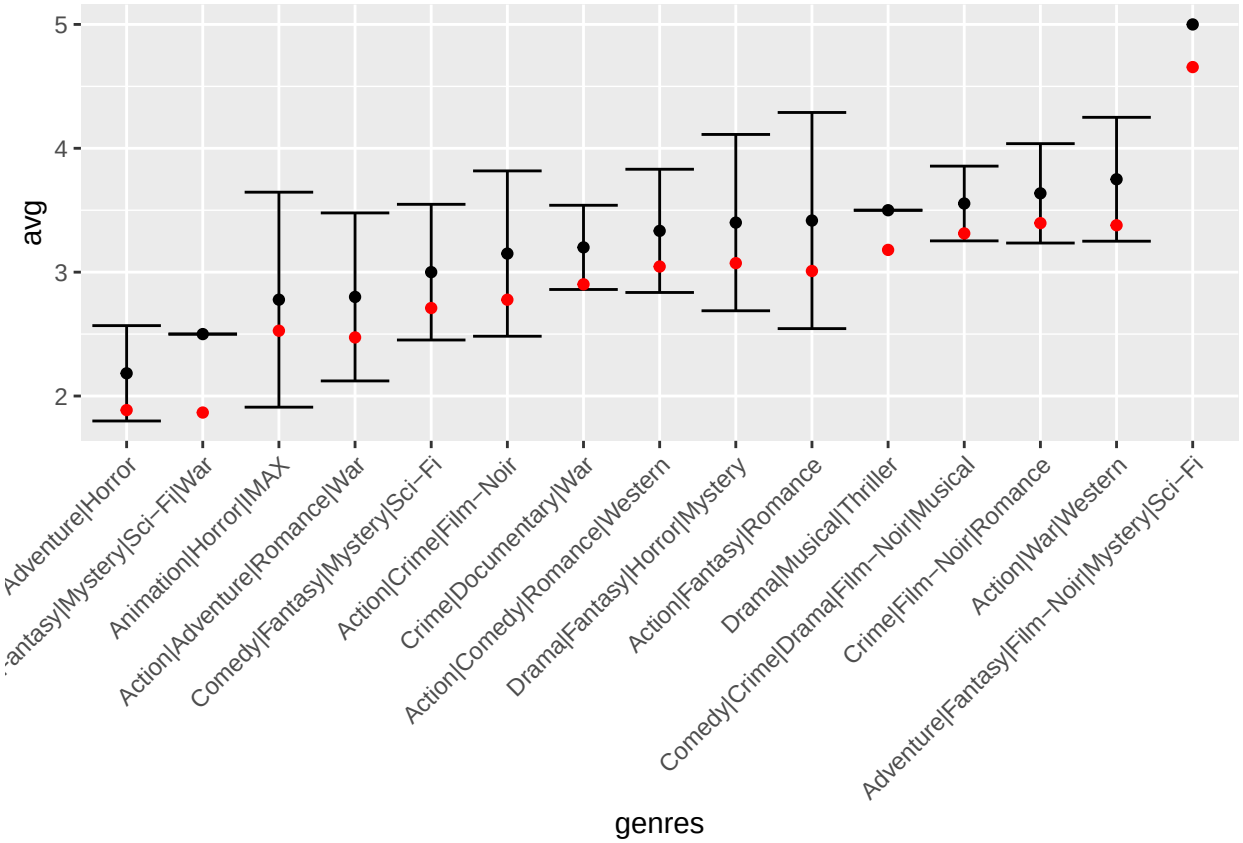
The below code tries to contrast the mean movie ratings by genre and the mean of the estimate we have so far, to observe the genre bias. This also helps us adjust our estimate based on the distance for each genre.

```
bg <-
    train_set %>%
    left_join(bi, by="movieId") %>%
    left_join(bu, by="userId") %>%
    group_by(genres) %>%    # join the tables geneated earlier and group them by genre
    summarize(n = n(),
              avg = mean(rating),
              avgerr = mean(b_u+b_i+mu),
              se = sd(rating)/sqrt(n()),
              b_g = mean(rating-b_u-b_i-mu)) %>%
    mutate(genres = reorder(genres, avg)) #arrange by genre and their averages so it
                                          #can be plotted in order
```

Lets plot the first 15 genres

```
bg  %>% top_n(15) %>%
  ggplot(aes(x = genres, y = avg, ymin = avg - 2*se, ymax = avg + 2*se )) +
  geom_point(aes(x = genres, y = avg),color = "black") +
  geom_errorbar() +
  geom_point(aes(x = genres, y = avgerr),color = "red") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
## Selecting by b_g
```




We notice that the prediction so far follows the average for the genres, but due to the presence of a genre bias, they are not at the mean. It also shows that the genres get different average ratings based on the popularity of the genres themselves.

Lets try to estimate the genre bias which will help in reducing the error and the RMSE as well by this prediction equation.

$$\mathbf{Y_{u,i}} = \mu + \boldsymbol{b_i} + \boldsymbol{b_u} + \boldsymbol{b_g} + \epsilon_{\mathbf{u,i}}$$

Lets actually do the prediction with our test set and see if we can get a reduction in RMSE

```
predict_mu_bi_bu_bg <-
    test_set %>%
    left_join(bi, by = "movieId") %>%
    left_join(bu, by = "userId") %>%
    left_join(bg, by = "genres") %>%
    mutate(mu_bi_bu_bg = mu + b_i + b_u + ifelse(is.na(b_g),0,b_g))
# not all genres would have been captured in trainset and hence some movies
# in the testset may generate an NA value for genre bias

rmse_mu_bi_bu_bg <- RMSE(predict_mu_bi_bu_bg$mu_bi_bu_bg,test_set$rating)

RMSE_table <- bind_rows(RMSE_table,data.frame(
              method="Average, Movie,User and Genre bias Model",RMSE=rmse_mu_bi_bu_bg))
RMSE_table %>% kable()
```

| method | RMSE |
|---|---|
| Average Model | 1.0599043 |
| Average and Movie bias Model | 0.9437429 |
| Average, Movie bias and User bias Model | 0.8659319 |
| Average, Movie,User and Genre bias Model | 0.8655941 |

This shows a significant reduction in RMSE, clarifying that the genre bias was a significant effect.

In all the above models considered, there is a small percentage of bad movies with high rating and good movies with low rating affecting the overall models ability to predict with greater accuracy. We will fine tune the model we have now with regularization, which relies on running the model repetitively with different weights represented by $\lambda$ known as cross-validation to get the lowest RMSE possible.

### Regularization model

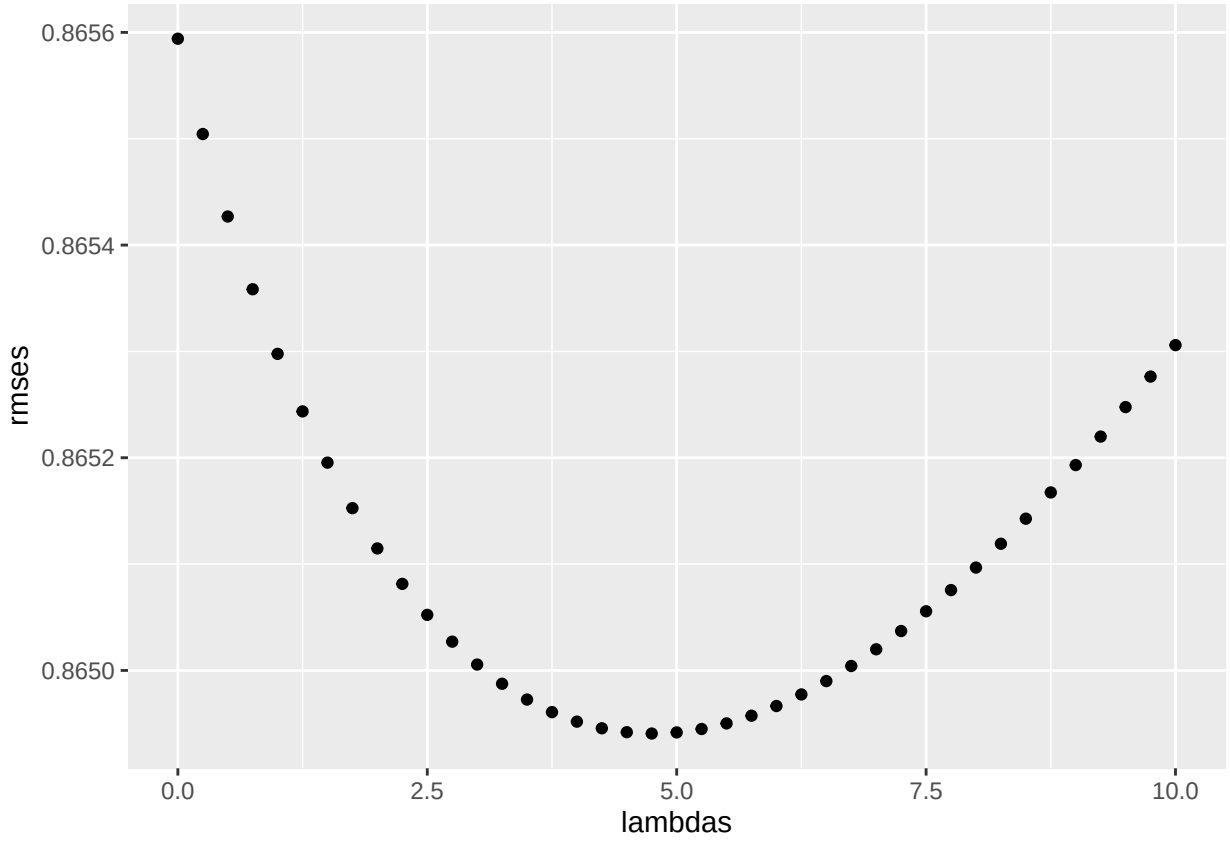
Since we will be running the model repetitively, we will create a function with the current model that accepts a variable $\lambda$ and runs it for different values of it. We then plot the RMSE's obtained against the different values of $\lambda$ used to get the lowest RMSE possible. This value of $\lambda$ will be used for the final run against the validation set.

by adding the term $\lambda$ to the denominator, we are making sure that for low values of **n** the weight of $\lambda$ seems large and reduces the overall effect, and for higher values of **n** the same $\lambda$ appears small and does not cause much alteration to the effect

```
lambdas <- seq(0, 10, 0.25) # weights for which we will plot RMSE's

#function to get the regularized RMSE for the lamda passed
rmse_reg <- function(lambda){
  #Train on trainset
    mu <- mean(train_set$rating)
    b_i <- train_set %>%
      group_by(movieId) %>%
      summarize(b_i = sum(rating - mu)/(n()+lambda))
    b_u <- train_set %>%
      left_join(b_i, by="movieId") %>%
      group_by(userId) %>%
      summarize(b_u = sum(rating - b_i - mu)/(n()+lambda))
    b_g <- train_set %>%
      left_join(b_i, by="movieId") %>%
      left_join(b_u, by="userId") %>%
      group_by(genres) %>%
      summarize(b_g = sum(rating - b_u - b_i - mu)/(n()+lambda))
    #Predict on testset
    predict_rating <- test_set %>%
      left_join(b_i, by = "movieId") %>%
      left_join(b_u, by = "userId") %>%
      left_join(b_g, by = "genres") %>%
      mutate(mu_bi_bu_bg = mu + b_i + b_u + b_g)
    #RMSE
    return(RMSE(predict_rating$mu_bi_bu_bg, test_set$rating))
  }
# get the RMSE's for all the lambda's
rmses <- sapply(lambdas,rmse_reg)
# plot the lamda's and their corresponding RMSE's
```

```
qplot(lambdas, rmses)
```




We can see that the value of lamda that corresponds to the low value of **0.86494** is about **5.0**

Setting lamda to this value in our regularized model and save the RMSE in the table.

```
rmse_regu <- rmse_reg(5.0)
RMSE_table <- bind_rows(RMSE_table,data.frame(
                method="Regularized Average, Movie, User and Genre bias Model",RMSE=rmse_regu))
RMSE_table %>% kable()
```

| method | RMSE |
|---|---|
| Average Model | 1.0599043 |
| Average and Movie bias Model | 0.9437429 |
| Average, Movie bias and User bias Model | 0.8659319 |
| Average, Movie,User and Genre bias Model | 0.8655941 |
| Regularized Average, Movie, User and Genre bias Model | 0.8649417 |

We did achieve some reduction in RMSE, but not significant, which leads me to believe that we are very close to a situation where we will be over training the model if we tried harder.

### Note

I would have tried to seperate out the individual genre and weighted out each one of them in the proprtion of their raings and created a sum of their individual weights to come up with a combined genre bias, but since this is not a recommendation model, I preferred this approach.

I opted not to go with the neighborhood models as I could not fit any model due to the size of the data. Matrix factorization would have been the next thing to consider if the RMSE would not have been reduced significantly.The data size unless significantly filtered out was not suitable for any models studied to create an ensemble model.

During my research, I did chance upon LASSO/RIDGE method, that was able to run thru the data, but since we did not have too many effects in this exercise, it was not able to pick out any effects that had more significance over the others, and the coefficients that it generated for Genre and Year yielded higher RMSEs to be included in this exercise.

## Result

Lets now train our model with the full edx data and see if it concurs with our analysis. We will have to generate a new but similar function to run against the **edx** data set to train and **validation** data set to predict our responses with a lambda $\lambda$ set to **5.0** and obtain an RMSE

```
#function to get the regularized RMSE for the lamda passed
rmse_result <- function(lambda){
  #Train on edx
    mu <- mean(edx$rating)
    b_i <- edx %>%
      group_by(movieId) %>%
      summarize(b_i = sum(rating - mu)/(n()+lambda))
    b_u <- edx %>%
      left_join(b_i, by="movieId") %>%
      group_by(userId) %>%
      summarize(b_u = sum(rating - b_i - mu)/(n()+lambda))
    b_g <- edx %>%
      left_join(b_i, by="movieId") %>%
      left_join(b_u, by="userId") %>%
      group_by(genres) %>%
      summarize(b_g = sum(rating - b_u - b_i - mu)/(n()+lambda))
    #Predict on validation
    predict_rating <- validation %>%
      left_join(b_i, by = "movieId") %>%
      left_join(b_u, by = "userId") %>%
      left_join(b_g, by = "genres") %>%
      mutate(mu_bi_bu_bg = mu + b_i + b_u + b_g)
    #RMSE
    return(RMSE(predict_rating$mu_bi_bu_bg, validation$rating))
}
# execute the function with lambda set to 5.0
rmse_final <- rmse_result(5.0)
# Result when run on validation set
RMSE_table <- bind_rows(RMSE_table,data.frame(
                  method="Result when model run on validation set",RMSE=rmse_final))
RMSE_table %>% kable()
```

| method | RMSE |
|---|---|
| Average Model | 1.0599043 |
| Average and Movie bias Model | 0.9437429 |
| Average, Movie bias and User bias Model | 0.8659319 |
| Average, Movie,User and Genre bias Model | 0.8655941 |
| Regularized Average, Movie, User and Genre bias Model | 0.8649417 |

| method | RMSE |
|---|---|
| Result when model run on validation set | 0.8644501 |

We have obtained better RMSE of **0.8644501** on the validation set when compared to the test set. This confirms no over training and an adequate model to present the results.

We can probably better the results marginally by running the cross-validation against the validation set, but since no operation was allowed on the validation set, the final results are as presented.

## Conclusion

Movies get rated by various methods and viewers to provide potential viewers a reference by which they can make their decisions. This methodology can be further expanded into movie recommendation system where a movie is picked based on the viewers prior history of movies watched.

In this report, we have identified the main source or effects that determine the rating a movie would recieve.

They have been identified as

1. The average rating of all the movies recieve which is about 3.5 out of a max 5 rating.
2. A movie bias that is a factor of how viewers rate the same movie differently.
3. A user bias that factors in how the same viewer rates movies differently.
4. A genre bias that factors in how some genre's are rated differently than others.

Some of the other methodology tried to pick the effects failed due to the nature and size of information provided for analysis which include linear regression, k - nearest neighbor and random forest.

Other models like RIDGE/LASSO gave inconsistent or higher errors, suggested very low values of $\lambda$ and coefficients for the significant effects and hence not considered to be reported.

## References

Recommendation systems: https://rafalab.github.io/dsbook/large-datasets.html#recommendation-systems

Ridge and Lasso regression: http://www.sthda.com/english/articles/37-model-selection-essentials-in-r/153-penalized-regression-essentials-ridge-lasso-elastic-net/