MOVIELENS Project

by Yaya Bamba

PROJECT OVERVIEW

For this project, I will be creating a movie recommendation system using the MovieLens dataset. I will use the 10M version of the MovieLens dataset to make the computation a little easier. The entire dataset can be found at this link : https://grouplens.org/datasets/movielens/latest/

I will train a machine learning algorithm using the inputs in edx dataset (training set) to predict movie ratings in the validation set. Not all movies were rated in the dataset by all Users. Our goal is to predict which rating a user will give to a particular movie based on other users' ratings for that movie and other features in the dataset.

On the agenda for this report I will have the below sections :

- SETTING THE DATASET
- DATA PRE-PROCESSING
- EXPLORATORY DATA ANALYSIS
- FEATURE ENGINEERING
- DATA PREPARATION
- MODELLING
- RESULTS
- PROJECT CONCLUSIONS

I will start by preprocessing the data to make it fit for our machine learning algorithms, do some data exploration to find out leading patterns in the data, and understand which features drives the ratings in the dataset best, do some modelling and finally conclude based on my results .

First and foremost, let's start by setting the data for the project

SETTING THE DATASET

I will be using the below libraries.

```
# load the required libraries -
 library(caret, warn.conflicts = FALSE, quietly=TRUE)
 library(data.table, warn.conflicts = FALSE, quietly=TRUE)
 library(tidyverse, warn.conflicts = FALSE, quietly=TRUE)
```

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

```
## v tibble  2.0.1       v purrr   0.3.0
## v tidyr   0.8.2       v dplyr   0.8.0.1
## v readr   1.3.1       v stringr 1.4.0
## v tibble  2.0.1       v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------------------------------------------------
---------------------------------------------------- tidyverse_conflicts() --
## x dplyr::between()   masks data.table::between()
## x dplyr::filter()    masks stats::filter()
## x dplyr::first()     masks data.table::first()
## x dplyr::lag()       masks stats::lag()
## x dplyr::last()      masks data.table::last()
## x purrr::lift()      masks caret::lift()
## x purrr::transpose() masks data.table::transpose()
```

```r
library(dslabs, warn.conflicts = FALSE, quietly=TRUE)
library(dplyr, warn.conflicts = FALSE, quietly=TRUE)
library(stringr, warn.conflicts = FALSE, quietly=TRUE)
library(lubridate, warn.conflicts = FALSE, quietly=TRUE)
library(e1071, warn.conflicts = FALSE, quietly=TRUE)
library(corrplot, warn.conflicts = FALSE, quietly=TRUE)
```

```
## corrplot 0.84 loaded
```

```r
library(ggplot2, warn.conflicts = FALSE, quietly=TRUE)
library(gtable, warn.conflicts = FALSE, quietly=TRUE)
library(grid, warn.conflicts = FALSE, quietly=TRUE)
library(gridExtra, warn.conflicts = FALSE, quietly=TRUE)
```

Let's have a summary of the dataset and a glimse of the table

```r
dim(edx)
```

```
## [1] 9000055       6
```

The new table have 9000055 rows and 6 columns

```r
n_distinct(edx$movieId)
```

```
## [1] 10677
```

```r
length(unique(edx[["userId"]]))
```

```
## [1] 69878
```

There are 10677 disctinct movies in the dataset and those movies were rated by 69878 different users

```r
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 movie with the greatest numbers of raters, the one most frequently rated by the users is PULP FICTION (I never watched that movie, but I have to admit that this picked my interest for this movie)

```r
head(edx)
```

```
##   userId movieId rating timestamp                          title
## 1      1     122      5 838985046               Boomerang (1992)
## 2      1     185      5 838983525                Net, The (1995)
## 4      1     292      5 838983421                Outbreak (1995)
## 5      1     316      5 838983392                Stargate (1994)
## 6      1     329      5 838983392 Star Trek: Generations (1994)
## 7      1     355      5 838984474        Flintstones, The (1994)
##                           genres
## 1                 Comedy|Romance
## 2          Action|Crime|Thriller
## 4   Action|Drama|Sci-Fi|Thriller
## 5        Action|Adventure|Sci-Fi
## 6  Action|Adventure|Drama|Sci-Fi
## 7        Children|Comedy|Fantasy
```

```
glimpse(edx)
```

```
## Observations: 9,000,055
## Variables: 6
## $ userId    <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ movieId   <dbl> 122, 185, 292, 316, 329, 355, 356, 362, 364, 370, 37...
## $ rating    <dbl> 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5...
## $ timestamp <int> 838985046, 838983525, 838983421, 838983392, 83898339...
## $ title     <chr> "Boomerang (1992)", "Net, The (1995)", "Outbreak (19...
## $ genres    <chr> "Comedy|Romance", "Action|Crime|Thriller", "Action|D...
```

A quick glimpse at the dataset give us an idea of how the table now looks. we can see that we might have to do some preprocessing :

- the movie premier year is tied to the movie title . we will need to extract it in order to compute the movie age by the time it was rated by a specific user
- Timestamps represents the date in time the movie has been rated by a particular user . we will need to convert the timestamp to year in order to explore its relation with the movie age
- Ratings are made on a 5-star scale, with half-star increments. we will need to compute the average rating per movie for exploration purposes

Lets move to preprocessing and see how we can prepare our data for machine learning steps

DATA PRE-PROCESSING

Preparing the data is a prerequisite to get the best results from machine learning algorithms we intend to use in this project. In this section, we will prepare edx dataset in order to best expose its structure to machine learning algorithms in R. we will be using the caret package. Based on the look on the data we will proceed with the below preprocessing :

- Extract the movie premier year in order to compute the movie age
- convert the rating's timestamp to year in order to explore its relation with the movie age
- Create and average rating for each movie in the dataset

we need to make sure that the data in the column make sense and is clean. let's check if the rated year is consistent. We will also check consistency in the premier date column

```
# check if there is a premier year greater than the year 2019
edx %>% filter(premier_year > 2019) %>% group_by(movieId, title, premier_year) %>% summarize(n = n())
```

```
## # A tibble: 6 x 4
## # Groups:   movieId, title [6]
##   movieId title                                          premier_year     n
##     <dbl> <chr>                                                 <dbl> <int>
## 1     671 Mystery Science Theater 3000: The Movie (1996)         3000  3280
## 2    2308 Detroit 9000 (1973)                                    9000    22
## 3    4159 3000 Miles to Graceland (2001)                         3000   714
## 4    5310 Transylvania 6-5000 (1985)                             5000   195
## 5    8864 Mr. 3000 (2004)                                        3000   146
## 6   27266 2046 (2004)                                            2046   426
```

```
# check is there is a premier date inferior to the year 1900

edx %>% filter(premier_year < 1900) %>% group_by(movieId, title, premier_year) %>% summarize(n = n())
```

```
## # A tibble: 8 x 4
## # Groups:   movieId, title [8]
##   movieId title                                          premier_year     n
##     <dbl> <chr>                                                 <dbl> <int>
## 1    1422 Murder at 1600 (1997)                                 1600  1566
## 2    4311 Bloody Angels (1732 HÃ¸tten: Marerittet Har e~         1732     9
## 3    5472 1776 (1972)                                           1776   185
## 4    6290 House of 1000 Corpses (2003)                          1000   367
## 5    6645 THX 1138 (1971)                                       1138   464
## 6    8198 1000 Eyes of Dr. Mabuse, The (Tausend Augen d~        1000    24
## 7    8905 1492: Conquest of Paradise (1992)                     1492   134
## 8   53953 1408 (2007)                                           1408   466
```

We will look into those movie titles to the correct premier year and replace the wrong premier years by the correct ones

```
#replace the incorrect premier years after looking up on the movie title

edx[edx$movieId == "671", "premier_year"] <- 1996
edx[edx$movieId == "2308", "premier_year"] <- 1973
edx[edx$movieId == "4159", "premier_year"] <- 2001
edx[edx$movieId == "5310", "premier_year"] <- 1985
edx[edx$movieId == "8864", "premier_year"] <- 2004
edx[edx$movieId == "27266", "premier_year"] <- 2004
edx[edx$movieId == "1422", "premier_year"] <- 1997
edx[edx$movieId == "4311", "premier_year"] <- 1998
edx[edx$movieId == "5472", "premier_year"] <- 1972
edx[edx$movieId == "6290", "premier_year"] <- 2003
edx[edx$movieId == "6645", "premier_year"] <- 1971
edx[edx$movieId == "8198", "premier_year"] <- 1960
edx[edx$movieId == "8905", "premier_year"] <- 1992
edx[edx$movieId == "53953", "premier_year"] <- 2007
```

FEATURES ENGINEERING

Before doing some exploratory data analysis, we need to add additional features to the data set that will help us improve the predictive accuracy of our marchine learning model. We will add the below useful feature which are derived from the original features :

- Movie age (movie_age)
- average movie rating : movie_avg_rat
- average user rating : User_avg_rat
- average rating by movie age :age_avg_rat
- Number of votes per movie : numbVotes

```
#Calculate the  movie age  by the time the movie was rated by the user and add the variable to the dataset e
dx

edx <- edx %>% mutate( movie_age = year_rated - premier_year)

#Calculate average rating by movie add the column to edx dataset

edx <- edx %>% group_by(movieId) %>% mutate(movie_avg_rat = mean(rating))

# Calculate average rating by  user  and add the column to edx dataset

edx <- edx %>% group_by(userId) %>% mutate(user_avg_rat = mean(rating))

# Calculate average rating by movie age and add the column to edx dataset

edx <- edx %>% group_by(movie_age) %>% mutate(age_avg_rat = mean(rating))

# Calculate   the number of  votes per movie and add the column to edx dataset - this this how many time a s
pecific movie was rated

edx <- edx %>% group_by(movieId) %>% mutate(numbVotes = length(unique(userId)))
```

EXPLORATORY DATA ANALYSIS

In this phase we will try to have a better understanding of the dataset. we will do take a look at the features , study their correlation between them, study their correlation with the outcome variable (rating) and look for outliers in the data.

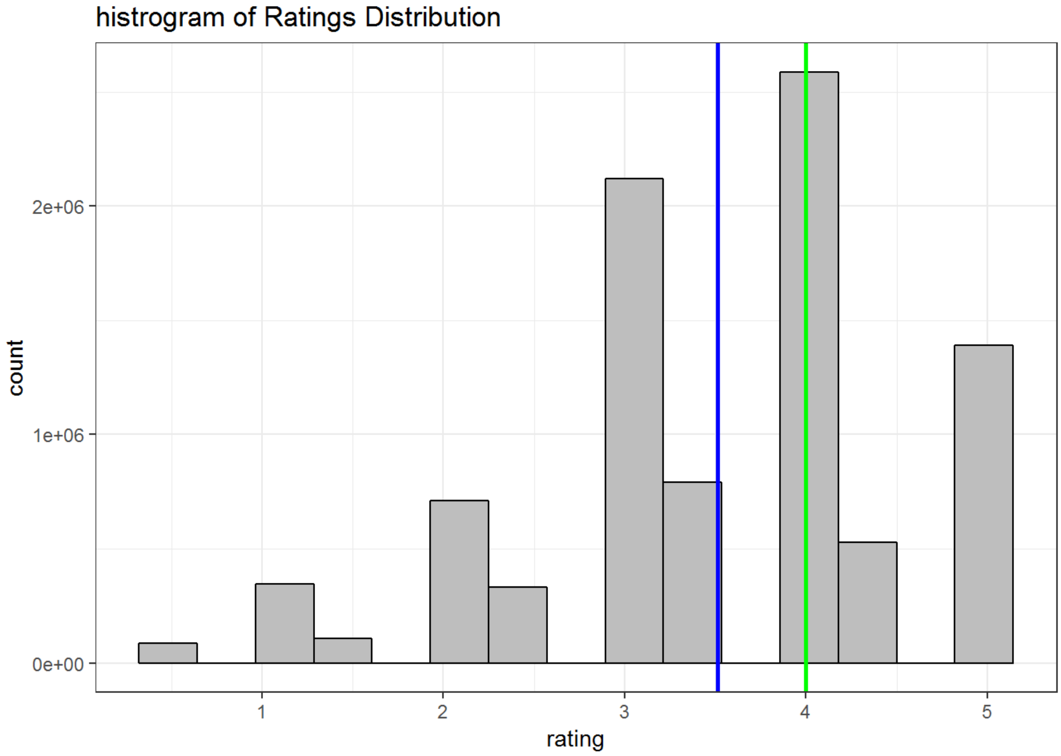
1.Study outcome variable Let's compute some useful summary statistics about our outcome variable (ratings) to have an idea of the data distribution

```
#Calculate summary statistics
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           premier_year    year_rated
##  Length:9000055     Length:9000055     Min.   :1900   Min.   :1995
##  Class :character   Class :character   1st Qu.:1987   1st Qu.:2000
##  Mode  :character   Mode  :character   Median :1994   Median :2002
##                                        Mean   :1990   Mean   :2002
##                                        3rd Qu.:1998   3rd Qu.:2005
##                                        Max.   :2010   Max.   :2009
##    movie_age       movie_avg_rat    user_avg_rat    age_avg_rat
##  Min.   :-12.00   Min.   :0.500   Min.   :0.500   Min.   :2.269
##  1st Qu.:  2.00   1st Qu.:3.218   1st Qu.:3.252   1st Qu.:3.438
##  Median :  7.00   Median :3.591   Median :3.529   Median :3.478
##  Mean   : 11.93   Mean   :3.512   Mean   :3.512   Mean   :3.512
##  3rd Qu.: 16.00   3rd Qu.:3.876   3rd Qu.:3.800   3rd Qu.:3.483
##  Max.   :108.00   Max.   :5.000   Max.   :5.000   Max.   :4.119
##    numbVotes
##  Min.   :    1
##  1st Qu.: 1634
##  Median : 4223
##  Mean   : 6787
##  3rd Qu.: 9862
##  Max.   :31362
```

The rating range is 0.5 to 5 . We Notice that the mean and median are close but the Mean (3.512) is less that the Median (4) which means that the rating distribution is slightly skewed to the Left. Let's graph that :

```
# plot the grapht using ggplot

ggplot(data= edx) +
  geom_histogram(mapping = aes(x=rating), bins = 15, boundary = 0, fill= "gray", col = "black") +
  geom_vline(xintercept = mean(edx$rating), col="blue", size= 1 ) +
  geom_vline(xintercept = median(edx$rating), col="red", size= 1 ) +
  geom_vline(xintercept = getmode(edx$rating), col="green", size= 1 ) +
   ggtitle ("histrogram of Ratings Distribution") +
  theme_bw()
```



As we can see, the distribution of the variable Rating is not perfectly normal, but we are going to proceed and assume it is normal as it is close enough to normal. since this is a regression problem, we don't have to worry about class imbalance (As oppose to if we had a classification project)

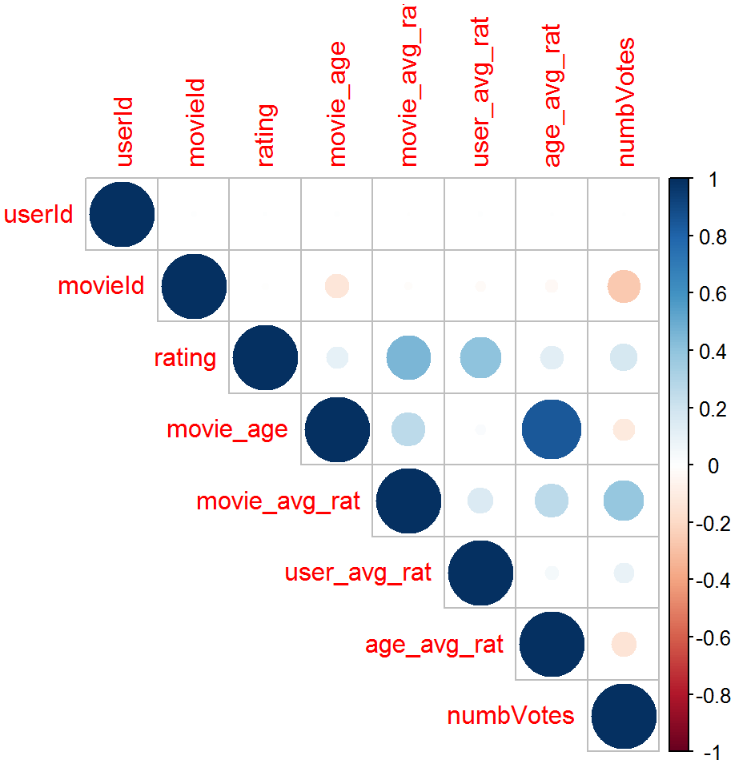
2.Study features

Lets understand the features by computing Summary statistics statistics about them. right now we anticipate 5 features based on the features in the final edx dataset

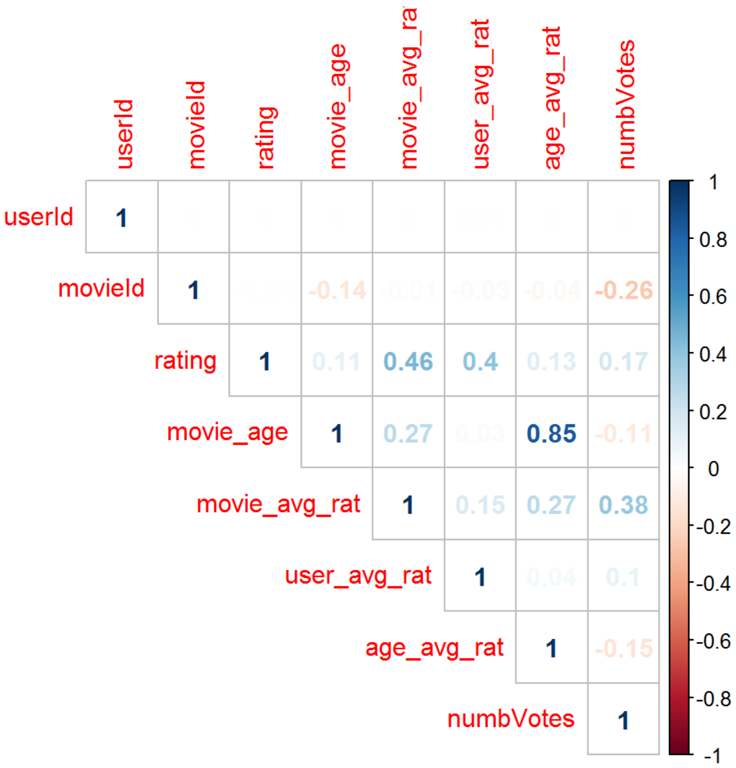
- Each movie age at the time of rating (movie_age)
- average rating by movie (movie_avg_rat)
- average rating by user (User_avg_rat)
- average rating by age (age_avg_rat)
- Number of votes/ratings per movie (numbVotes)

We will now do a correlation plot. This correlation plot wll display a chart showing the correlation between all feature variables. It will also let us anticipate if we need all our variables features in our model. We don't want our feature variables to present a high correlation between them as this will lead to Multicolinearity. If our dataset has perfectly positive or negative attributes then there is a high chance that the performance of the model will be impacted by a problem called "Multicollinearity". Multicollinearity happens when one predictor variable in a multiple regression model can be linearly predicted from the others with a high degree of accuracy (Correlation between 0.7 and 1).

Since we are using regression, lets check multicolinearity before moving further



with the correlation numbers

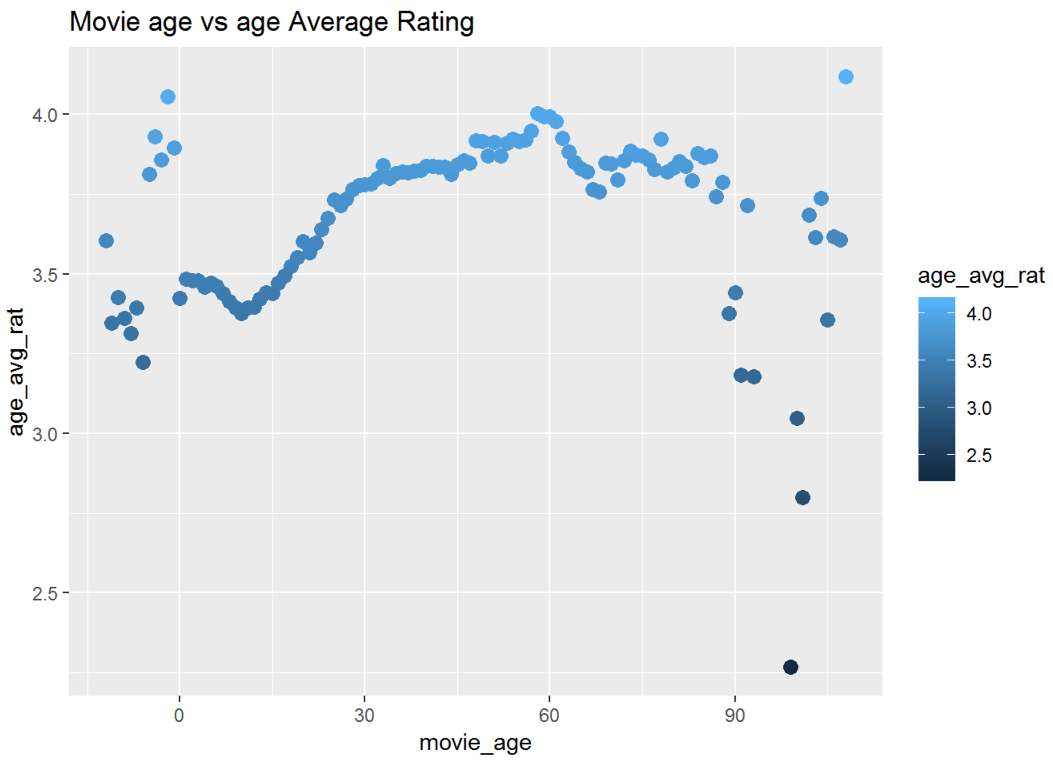


From the correlation matrix, we can have the below insights :

Insight 1 the predictive outcome is positively correlated to the movie average rating (0.46) , the user average rating (0.4) and midly to the number of votes (0.17). we will keep those 3 features as dependant variables

Insight 2 : there is a strong correlation between movie_age and age_avg_rat variables (0.85) . We will drop the age average rating from the features to avoid multicolinearity effects.

```
# age of movie vs average movie rating
edx <- as.data.frame(edx) # convert edx to data frame to allow for the ploting
edx %>%
  ggplot(aes(movie_age, age_avg_rat)) +
  geom_point(aes(col=age_avg_rat), size = 3) +
  ggtitle("Movie age vs age Average Rating")
```

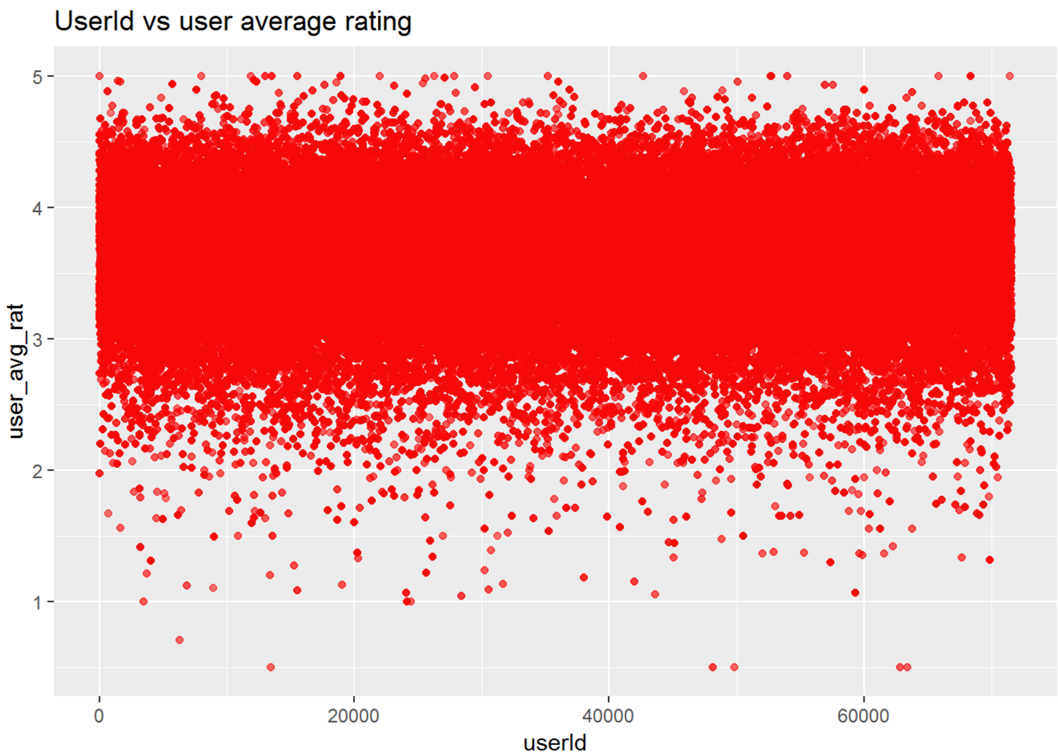


we can see this high correlation by plotting the movie age vs age Average rating scaterplot. we can clearly see that new movies tend to have higher ratings with a positive correlation . Then when the movie is old at the time of rating (Around 90 years old), the ratings tends to drop to lower levels (below 3.25). the older the movie, le lower the rating

Explore relationship between users and ratings

Let's plot the relationship between Users and Users average rating

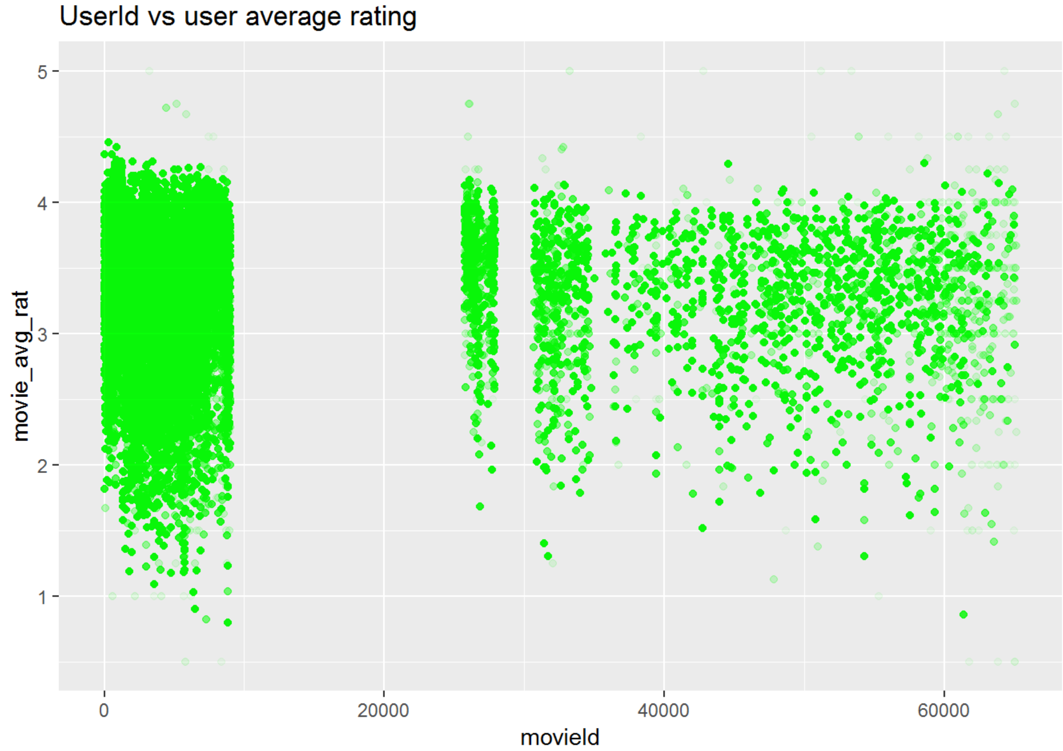
```
# userId vs average movie rating

edx  %>%
ggplot(aes(userId, user_avg_rat)) +
geom_point(alpha = 1/20, colour = "red") +
ggtitle("UserId vs user average rating")
```



Insight 3 : we can see on the above plot that on average users consistently rate movies between 2.5 and 4.5

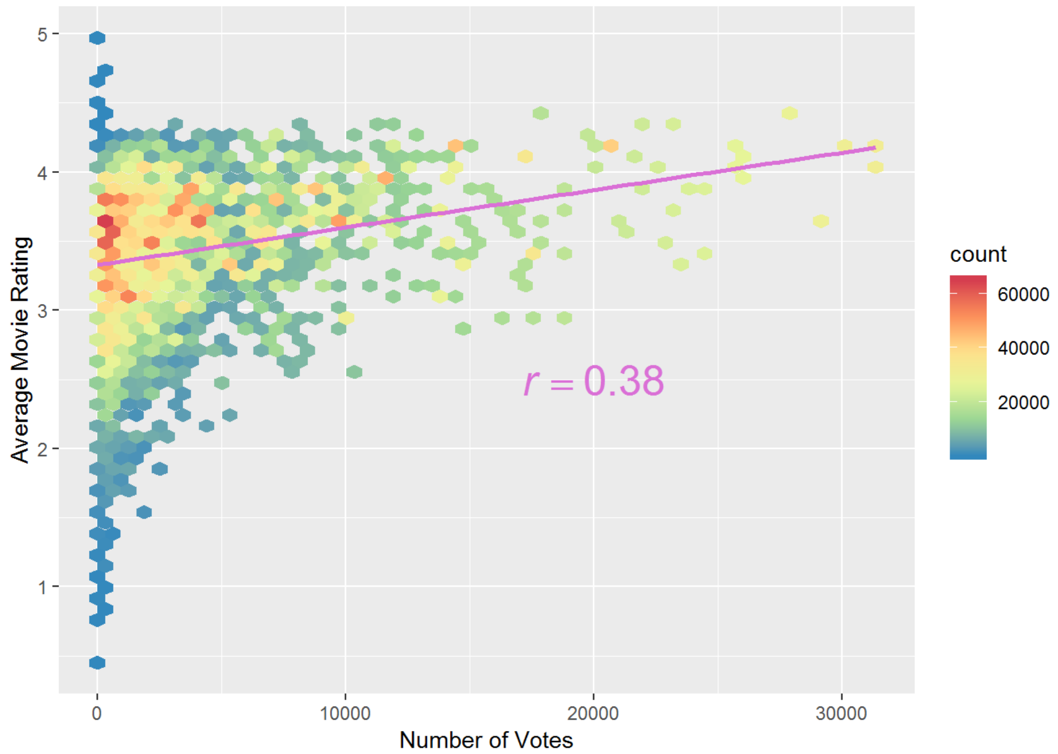
Explore relationship between Movies and movie average ratings

Let’s plot the relationship between movies and movies average rating



Insight 4 : Movies tend to be consistently rated between 1.5 and 4.5. this is telling us that movies rated below 1.5 are few

let’s check the relashinship between the number of votes a movie have and the average rating for this movie ?



Insight 5 : There is a positive correlation between number of votes and average movie rating (38%).

MOVIE GENRES ANALYSIS

We can notice that for most of the movies, several genres are piped together. we need to separate those genres to analyze movie genre impact on rating

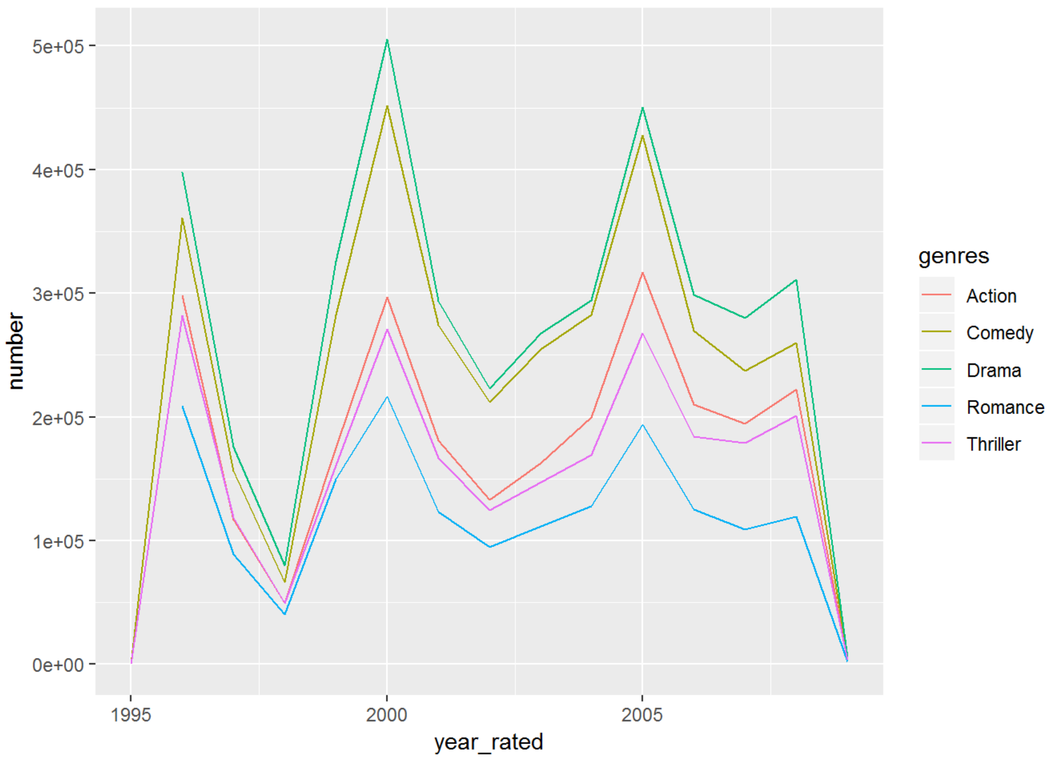
```
## # A tibble: 20 x 2
##    genres              count
##    <chr>               <int>
##  1 Drama                5336
##  2 Comedy               3703
##  3 Thriller             1705
##  4 Romance              1685
##  5 Action               1473
##  6 Crime                1117
##  7 Adventure            1025
##  8 Horror               1013
##  9 Sci-Fi                754
## 10 Fantasy               543
## 11 Children              528
## 12 War                   510
## 13 Mystery               509
## 14 Documentary           481
## 15 Musical               436
## 16 Animation             286
## 17 Western               275
## 18 Film-Noir             148
## 19 IMAX                   29
## 20 (no genres listed)      1
```

we can see that spliting the genres creates duplicate rows in the dataset edx_genres as new rows are created for movies classified in multiples genres.

When we summarize the UNIQUE movies per genre we notice that Drama, comedy and Actions are the ones with the most movies.

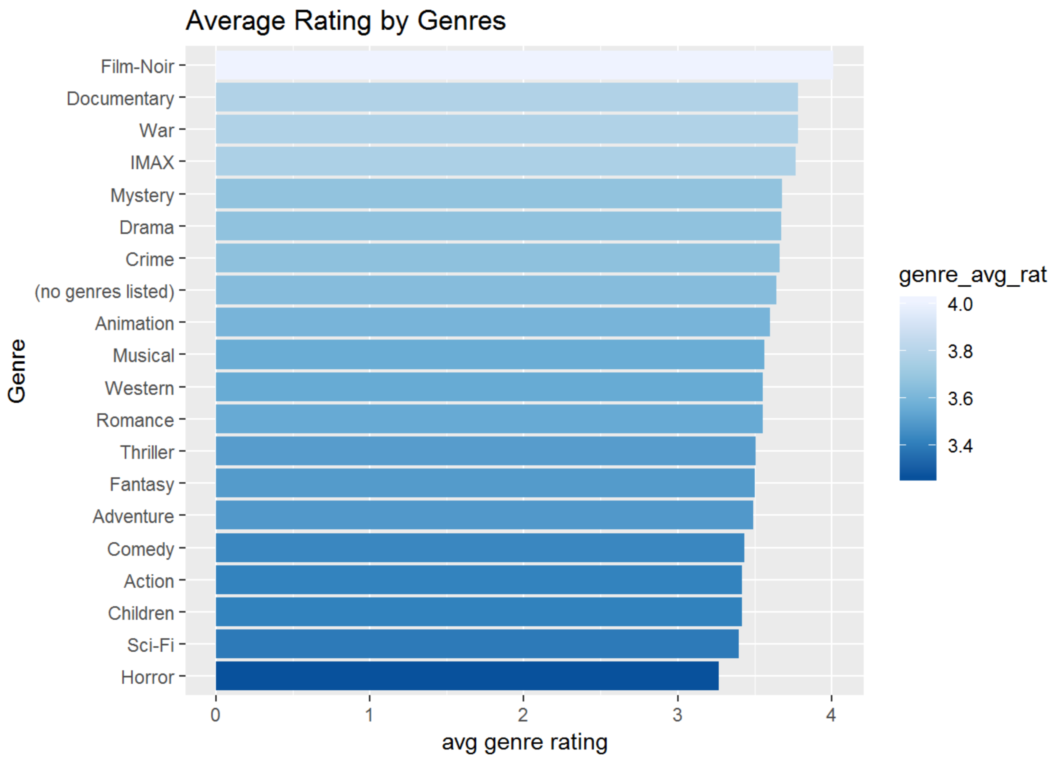
Let's take a look at how the popularity of the genre evolve over the years

plot the genre popularity over the years- we will only plot the top 5 popular genres- we will also filter from 1900



Drama and comedy remains the most popular genres over the years. I am curious to see what is the average rating for those movies and if users tend to give better ratings to the movies in thoses categories.

```
## # A tibble: 20 x 2
##    genres             genre_avg_rat
##    <chr>                      <dbl>
##  1 Film-Noir                   4.01
##  2 Documentary                 3.78
##  3 War                         3.78
##  4 IMAX                        3.77
##  5 Mystery                     3.68
##  6 Drama                       3.67
##  7 Crime                       3.67
##  8 (no genres listed)          3.64
##  9 Animation                   3.60
## 10 Musical                     3.56
## 11 Western                     3.56
## 12 Romance                     3.55
## 13 Thriller                    3.51
## 14 Fantasy                     3.50
## 15 Adventure                   3.49
## 16 Comedy                      3.44
## 17 Action                      3.42
## 18 Children                    3.42
## 19 Sci-Fi                      3.40
## 20 Horror                      3.27
```



surprisingly, the most voted movie genres are not the ones with the highest average rating. Horror is the least rated movi genre, suggesting that people don't like to be scared.

The variables or variables to remove will have low correlation with the outcome variable RATING and /or high correlation with the other predictors/features. From those criteria, Lets drop the variable age_avg_rat.

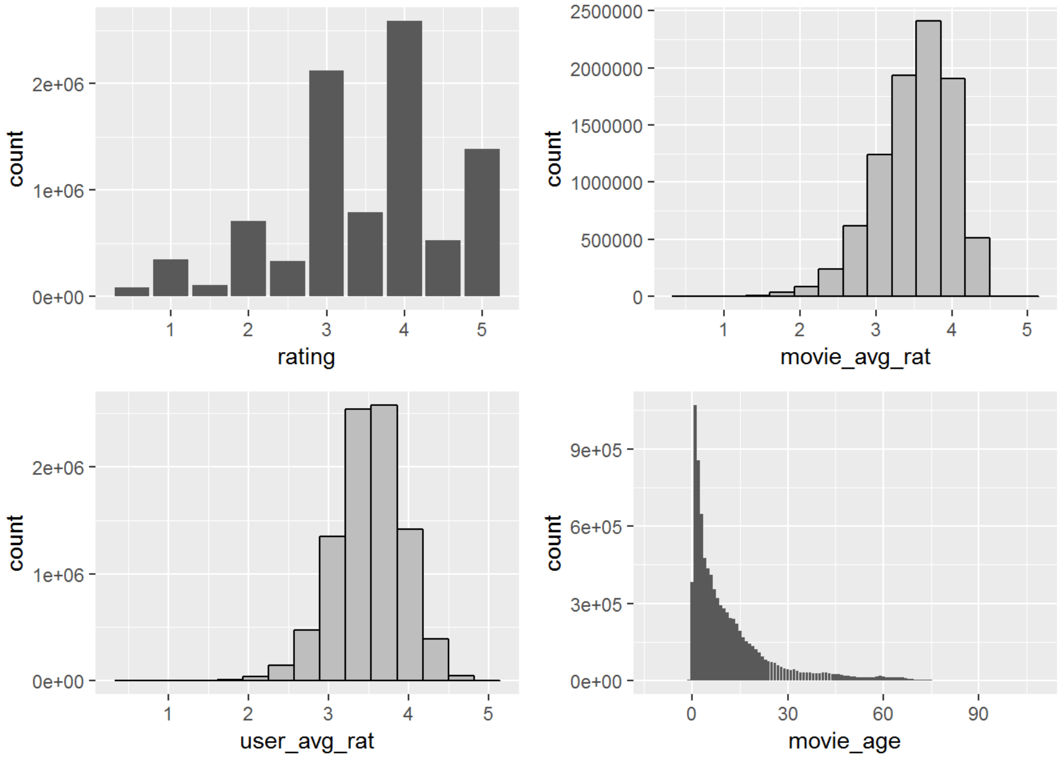
```
##   userId movieId rating timestamp                         title
## 1      1     122      5 838985046              Boomerang (1992)
## 2      1     185      5 838983525               Net, The (1995)
## 3      1     292      5 838983421               Outbreak (1995)
## 4      1     316      5 838983392               Stargate (1994)
## 5      1     329      5 838983392 Star Trek: Generations (1994)
## 6      1     355      5 838984474       Flintstones, The (1994)
##                          genres premier_year year_rated movie_age
## 1                Comedy|Romance         1992       1996         4
## 2         Action|Crime|Thriller         1995       1996         1
## 3  Action|Drama|Sci-Fi|Thriller         1995       1996         1
## 4       Action|Adventure|Sci-Fi         1994       1996         2
## 5 Action|Adventure|Drama|Sci-Fi         1994       1996         2
## 6       Children|Comedy|Fantasy         1994       1996         2
##   movie_avg_rat user_avg_rat numbVotes
## 1      2.858586            5      2178
## 2      3.129334            5     13469
## 3      3.418011            5     14447
## 4      3.349677            5     17030
## 5      3.337457            5     14550
## 6      2.487787            5      4831
```

CHECK FOR OUTLIERS

Lets check those features distribution to see if they are normal . We will also check for outliers and other inconsistent data points with boxplots

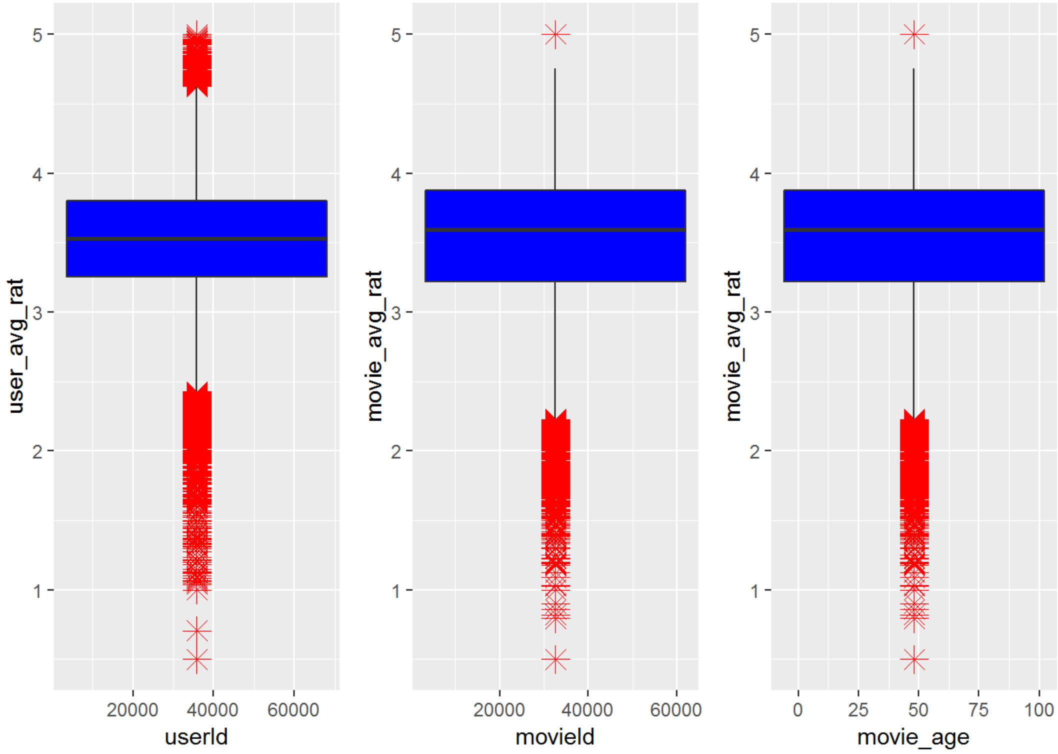
For a given continuous variable, outliers are those observations that lie outside 1.5*IQR, where IQR, the 'Inter Quartile Range' is the difference between 75th and 25th quartiles. Look at the points outside the whiskers in below box plot.

```
# Plot rating histogram
 h1 <- edx %>% ggplot(aes(rating)) +   geom_bar()
   # geom_histogram(binwidth = 1, fill = "blue", col = "black")


# Plot  movie average rating histogram

 h2 <- ggplot(data= edx) +
      geom_histogram(mapping = aes(x= movie_avg_rat), bins = 15, boundary = 0, fill= "gray", col = "black")

# Plot user average rating histogram

  h3 <- ggplot(data= edx) +
      geom_histogram(mapping = aes(x= user_avg_rat), bins = 15, boundary = 0, fill= "gray", col = "black")

# Plot  movie age  histogram

  h4 <- edx %>% ggplot(aes(movie_age)) + geom_bar()


# arrange the histograms side by side for bettre observations

grid.arrange(h1, h2, h3, h4,  nrow = 2, ncol = 2)
```



Insight 6 : Ratings plot suggest there are some of the movies have very few votes which result on either very large average for 1 vote of a small average rating. This suggest the existence of ouliers for ratings below . we will check this by checking the boxplots. We can also see that newer movies get more votes than the older ones.

Let's print the boxplots to observe any outliers



For the movie average rating, most of the outliers are below the 25 quartile. We will identify those outliers and count them , and then remove them when we do the data preparation step

We can have a quick look at the top 10 moVies that received the most votes

```
## # A tibble: 10,677 x 6
## # Groups:   movieId, title, genres, numbVotes [10,677]
##    movieId title                 genres        numbVotes movie_avg_rat count
##      <dbl> <chr>                 <chr>             <int>         <dbl> <int>
##  1     296 Pulp Fiction (1994)   Comedy|Crime~     31362          4.15 31362
##  2     356 Forrest Gump (1994)   Comedy|Drama~     31079          4.01 31079
##  3     593 Silence of the Lamb~ Crime|Horror~     30382          4.20 30382
##  4     480 Jurassic Park (1993) Action|Adven~     29360          3.66 29360
##  5     318 Shawshank Redemptio~ Drama             28015          4.46 28015
##  6     110 Braveheart (1995)     Action|Drama~     26212          4.08 26212
##  7     457 Fugitive, The (1993) Thriller          25998          4.01 25998
##  8     589 Terminator 2: Judgm~ Action|Sci-Fi     25984          3.93 25984
##  9     260 Star Wars: Episode ~ Action|Adven~     25672          4.22 25672
## 10     150 Apollo 13 (1995)      Adventure|Dr~     24284          3.89 24284
## # ... with 10,667 more rows
```

lets take a look at the most voted ones and their ratings

```
## # A tibble: 10,677 x 6
## # Groups:   movieId, title, genres, numbVotes [10,677]
##    movieId title                    genres     numbVotes movie_avg_rat count
##      <dbl> <chr>                    <chr>          <int>         <dbl> <int>
##  1    3191 Quarry, The (1998)       Drama              1           3.5     1
##  2    3226 Hellhounds on My Trail~ Documenta~         1           5       1
##  3    3234 Train Ride to Hollywoo~ Comedy             1           3       1
##  4    3356 Condo Painting (2000)    Documenta~         1           3       1
##  5    3383 Big Fella (1937)         Drama|Mus~         1           3       1
##  6    3561 Stacy's Knights (1982)   Drama              1           1       1
##  7    3583 Black Tights (1-2-3-4 ~ Drama|Mus~         1           3       1
##  8    4071 Dog Run (1996)           Drama              1           1       1
##  9    4075 Monkey's Tale, A (Les ~ Animation~         1           1       1
## 10    4820 Won't Anybody Listen? ~ Documenta~         1           2       1
## # ... with 10,667 more rows
```

We can notice based on this sample that the most voted movies tend to have higher ratings . But we can also see that movies with the less votes can also have a high rating. for example the movie Hellhounds on My Trail (1999) has a rating of 5 with only one vote.

DATA PREPARATION

Remove outliers we spotted when doing data exploration. we will call the new dataset edx_ml

I will consider an outlier everything below or above the 0.25 and 0.75 quantiles.

```
# Use the summary statistic to find the  75th and 25th percentiles
  summary(edx$rating)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.500   3.000   4.000   3.512   4.000   5.000
```

```
# Find the lower and upper fence that defines the outliers

  lowerFence <- Q1 - 1.5 * IQR
  upperFence <- Q3 + 1.5 * IQR
  lowerFence #this is the 25th percentile
```

```
## 25%
## 1.5
```

```
  upperFence #this is the 75th percentiel
```

```
## 75%
## 5.5
```

The 25th percentile is the lowerFence of the boxplot and has a value of 1.5 when the uperFence is the 75th percentile with a value of 5.5

Lets count the number of outliers rows based on those values.

```
# count the number of outliers

length(which(edx$rating %in% OutVals))
```

```
## [1] 431053
```

There are 431 053 outliers rows : those rose all have a rating < 1.5 .

Let's remove the outliers rows from the dataset and call the new dataset edx_ml tat we will consider for our modelling. we will keep the ones qith the outliers and will compare the odelling results with both dataset; this will allow us to see the effect of the ouliers on our models

Let's take a look at the new dataset edx_ml

```
# drop unwanted colums
edx_ml <- edx_ml %>% select(-timestamp, -premier_year, -year_rated)

# summary statistics

dim(edx_ml)
```

```
## [1] 8569002       9
```

```
n_distinct(edx_ml$movieId)
```

```
## [1] 10664
```

```
n_distinct(edx_ml$userId)
```

```
## [1] 69870
```

```
summary(edx_ml)
```

```
##      userId         movieId          rating         title          
##  Min.   :    1   Min.   :    1   Min.   :1.500   Length:8569002    
##  1st Qu.:18122   1st Qu.:  648   1st Qu.:3.000   Class :character  
##  Median :35750   Median : 1794   Median :4.000   Mode  :character  
##  Mean   :35871   Mean   : 4142   Mean   :3.644                     
##  3rd Qu.:53610   3rd Qu.: 3624   3rd Qu.:4.000                     
##  Max.   :71567   Max.   :65133   Max.   :5.000                     
##     genres            movie_age      movie_avg_rat     user_avg_rat   
##  Length:8569002     Min.   :-12.00   Min.   :0.7946   Min.   :0.7059  
##  Class :character   1st Qu.:  2.00   1st Qu.:3.2497   1st Qu.:3.2723  
##  Mode  :character   Median :  7.00   Median :3.6073   Median :3.5422  
##                     Mean   : 12.07   Mean   :3.5382   Mean   :3.5315  
##                     3rd Qu.: 16.00   3rd Qu.:3.8858   3rd Qu.:3.8106  
##                     Max.   :108.00   Max.   :5.0000   Max.   :5.0000  
##    numbVotes    
##  Min.   :    1  
##  1st Qu.: 1692  
##  Median : 4323  
##  Mean   : 6897  
##  3rd Qu.: 9933  
##  Max.   :31362
```

There are fewer rows (8 569 002). Minimum rating is now 1.5, and the mean improved from 3.512 to 3.664. We can also notice that there are fewer disctinct movies (10664). we dropped 13 movies that were probably rate 1.5 or lower there are also fewer users (69870). we dropped 8 users

we will move forward with the below features:

- average movie rating (movie_avg_rat) : this will be the feature that check the movie_effect
- average user rating (User_avg_rat) : this will be the feature that check the user_effect
- the age of the movie at the time of rating (movie_age) : this will be the feature that check the age_effect

MODELLING

What are we trying to predict?

Not all movies were rated in the dataset by all Users. Our goal is to predict which rating a user will give to a particular movie based on other users' ratings, the movieaverage rating or the movie age .

What type of problem is it? Supervised or Unsupervised Learning? Classification or Regression? Binary or Multi-class? Uni-variate or Multi-variate?

This is a multivariate supervised machine learning problem in which we have to predict numeric outcomes. so I will be using linear regression techniques.

I will use edx_ml as the training set. edx_ml is the final dataset without the outliers. I will also train the full edx dataset(with all outliers) and compare the results to see if removing the outliers did have an impact on RMSE.

I will treat validation as new data I don't have any access to until my algorithm is finished. So all my calculations / cross-validations will be on the training dataset edx_ml and edx. In the final step, I will predict user ratings on validation set .

Assessing the Fit of linear Regression Model

A well-fitting regression model results in predicted values close to the observed data values. Three statistics are used in Ordinary Least Squares (OLS) regression to evaluate model fit: R-squared, the overall F-test, and the Root Mean Square Error (RMSE).

I will be using the RMSE statistic here to assess the fit of the model

The RMSE is the square root of the variance of the residuals. It indicates the absolute fit of the model to the data . It measures how close the observed data points are to the model's predicted values. Lower values of RMSE indicate better fit.

LETS CACULATE RMSE

We will use 2 regression models to calculate the RMSE

MODEL 1 (movie_effect + user_effect) : Predicted_rating = mu + b_m + b_u

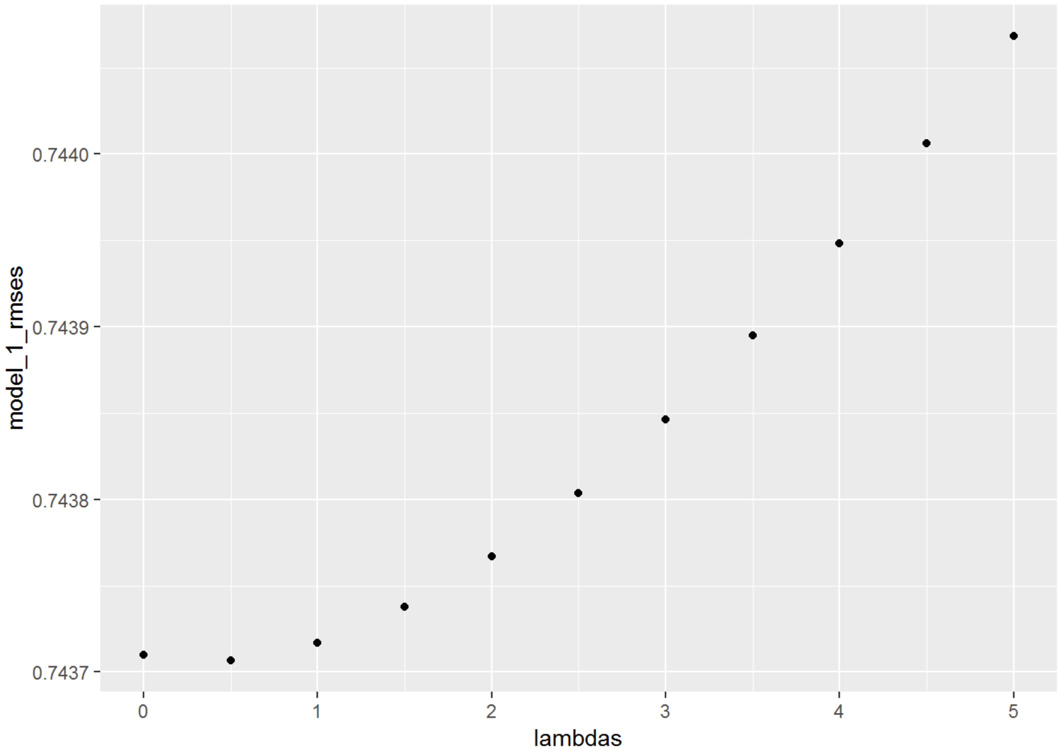
Model 2 (movie_effect + user_effect + age_effect) : predicted_rating = mu + b_m + b_u + b_a

Basically, we will add age_effect to the first model to see if it improves our RMSE

MODEL 1 (movie_effect + user_effect) TRAINING : Predicted_rating = mu + b_m + b_u

Let's use cross validation on the train set EDX_ML to define without using the test set until the final assessment. The test set should

```
# define RMSE function
RMSE <- function(actual_rating, predicted_rating){
  sqrt(mean((actual_rating - predicted_rating)^2))
}

#Choose the tuning value of lambda

lambdas <- seq(0,5,.5)
model_1_rmses <- sapply(lambdas, function(l){
  mu <- mean(edx_ml$rating)

  b_m <- edx_ml %>%
    group_by(movieId) %>%
    summarize(b_m = sum(rating - mu)/(n() + l))

  b_u <- edx_ml %>%
    left_join(b_m, by='movieId') %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_m - mu)/(n() +l))

  predicted_rating <- edx_ml %>%
    left_join(b_m, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    mutate(pred = mu + b_m +  b_u) %>% .$pred

  return(RMSE(predicted_rating, edx_ml$rating))
})


# lets compute lambdas curve to visually assess the optimal lambda
qplot(lambdas, model_1_rmses)
```



Let's find the lambda which minimize model_1_rmse

```
# determine optimal lambda
lambdas[which.min(model_1_rmses)]
```

```
## [1] 0.5
```

Model 1 Validation : Check model 1 against the validation set

```
#Check model 1 againt the validation set Prepare Validation set
mu <- mean(validation$rating)
l <- 0.5
b_m <- validation %>%
    group_by(movieId) %>%
    summarize(b_m = sum(rating - mu)/(n() + l))

b_u <- validation %>%
    left_join(b_m, by='movieId') %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_m - mu)/(n() +l))

predicted_rating <- validation %>%
    left_join(b_m, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    mutate(pred = mu + b_m +  b_u) %>% .$pred

RMSE(predicted_rating, validation$rating)
```

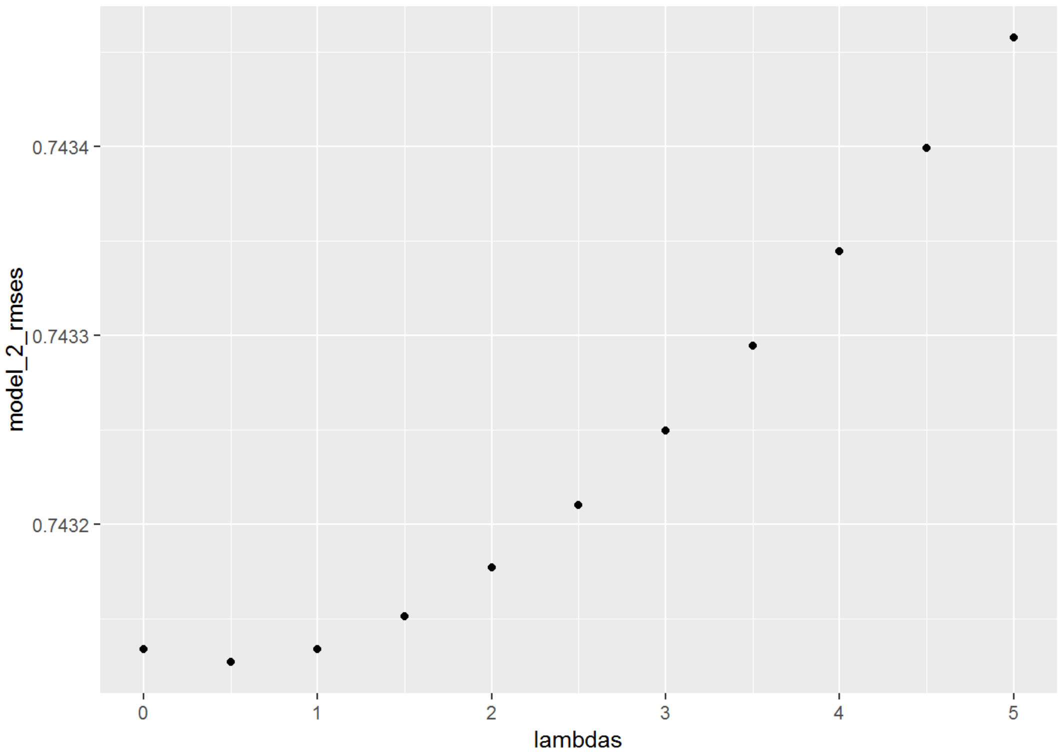
```
## [1] 0.8258487
```

we get **RMSE = 0.8258487**

Model 2 (movie_effect + user_effect + age_effect) : predicted_rating = mu + b_m + b_u + b_a

In this second model, we will add the movie effect and see if it improves RMSE Let's use cross validation on the train set EDX_ML to define the optimal lambda on the training set

```r
# define RMSE2 function
RMSE2 <- function(actual_rating, predicted_rating2){
  sqrt(mean((actual_rating - predicted_rating2)^2))
}

#Choose the tuning value of lambda
lambdas <- seq(0,5,.5)
model_2_rmses <- sapply(lambdas, function(l){
  mu <- mean(edx_ml$rating)

  b_m <- edx_ml %>%
    group_by(movieId) %>%
    summarize(b_m = sum(rating - mu)/(n() + l))

  b_u <- edx_ml %>%
    left_join(b_m, by='movieId') %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_m - mu)/(n() +l))

   b_a <- edx_ml %>%
    left_join(b_m, by='movieId') %>%
     left_join(b_u, by='userId') %>%
     group_by(movie_age) %>%
    summarize(b_a = sum(rating - b_m - b_u - mu)/(n() +l))



  predicted_rating2 <- edx_ml %>%
    left_join(b_m, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    left_join(b_a, by = "movie_age") %>%
    mutate(pred = mu + b_m +  b_u + b_a) %>% .$pred

  return(RMSE2(predicted_rating2, edx_ml$rating))
})


#plot Lambdas
qplot(lambdas, model_2_rmses)
```



Let's find the lambda which minimize model_2_rmse

```r
# define optimal lambda value
lambdas[which.min(model_2_rmses)]
```

```
## [1] 0.5
```

*Check model 2 against the validation set : predicted_rating = mu + b_m + b_u + b_a

Now let begin the real tests

```
#Check model 2 againt the validation set Prepare Validation set
mu <- mean(validation2$rating)
l <- 0.5
b_m <- validation2 %>%
    group_by(movieId) %>%
    summarize(b_m = sum(rating - mu)/(n() + l))

b_u <- validation2 %>%
    left_join(b_m, by='movieId') %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_m - mu)/(n() +l))

b_a <- validation2 %>%
    left_join(b_m, by='movieId') %>%
    left_join(b_u, by='userId') %>%
    group_by(movie_age) %>%
    summarize(b_a = sum(rating - b_m - b_u - mu)/(n() +l))

predicted_rating2 <- validation2 %>%
    left_join(b_m, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    left_join(b_a, by = "movie_age") %>%
    mutate(pred = mu + b_m +  b_u + b_a) %>% .$pred

RMSE2(predicted_rating2, validation2$rating)
```

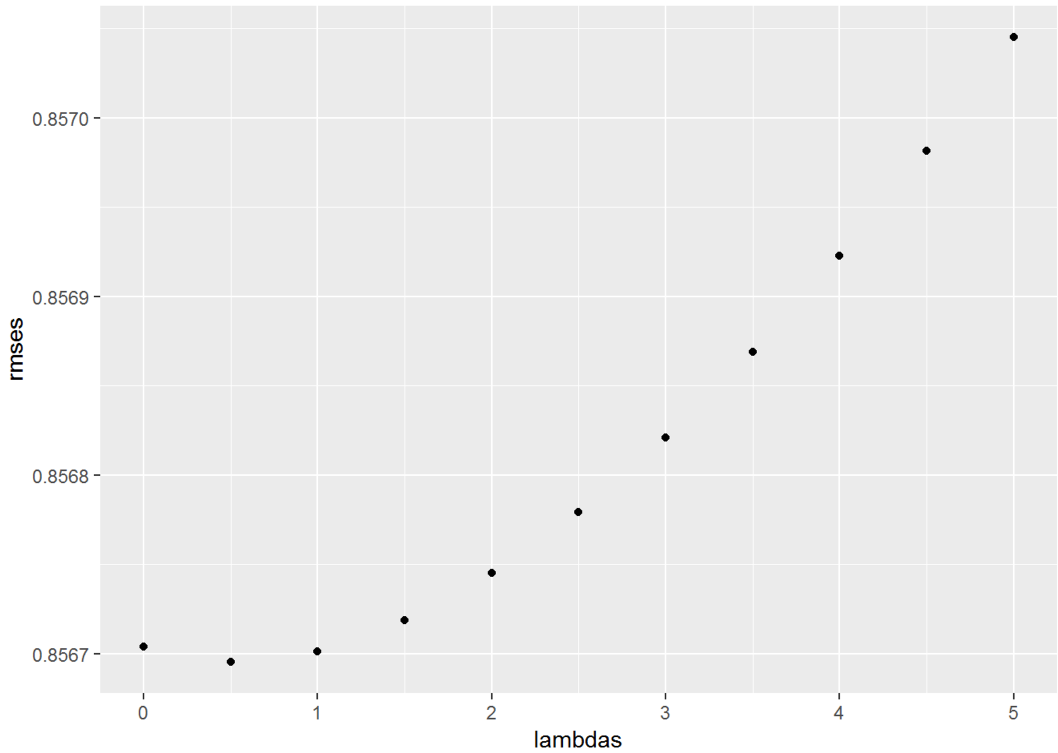
```
## [1] 0.825298
```

We get **RMSE = 0.825298**

MODELING WITH OUTLIERS

Lets check to see if keeping the outliers would have had an impact on our results. we will use the dataset edx (With all the outliers, wich means the movies with ratings < 1.5)

```
edx <- edx %>% select(-numbVotes) # drop non numeric values from edx_ml dataset

# define RMSE function for edx
RMSE_all <- function(actual_rating, predicted_rating){
  sqrt(mean((actual_rating - predicted_rating)^2))
}

#Choose the tuning value of lambda

lambdas <- seq(0,5,.5)
rmses <- sapply(lambdas, function(l){
  mu <- mean(edx$rating)

  b_m <- edx %>%
    group_by(movieId) %>%
    summarize(b_m = sum(rating - mu)/(n() + l))

  b_u <- edx %>%
    left_join(b_m, by='movieId') %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_m - mu)/(n() +l))

  predicted_rating <- edx %>%
    left_join(b_m, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    mutate(pred = mu + b_m +  b_u) %>% .$pred

  return(RMSE_all(predicted_rating, edx$rating))
})


# lets compute lambdas curve to visually assess the optimal lambda
qplot(lambdas, rmses)
```




```
# define optimal lambda value
lambdas[which.min(rmses)]
```

```
## [1] 0.5
```

Do the testing on the validation set

```
#Check  againt the validation
mu <- mean(validation$rating)
l <- 0.5
b_m <- validation %>%
    group_by(movieId) %>%
    summarize(b_m = sum(rating - mu)/(n() + l))

b_u <- validation %>%
    left_join(b_m, by='movieId') %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_m - mu)/(n() +l))

predicted_rating <- validation %>%
    left_join(b_m, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    mutate(pred = mu + b_m +  b_u) %>% .$pred

RMSE_all(predicted_rating, validation$rating)
```

```
## [1] 0.8258487
```

We get **RMSE = 0.8258487** . This is the same RMSE that the one obtained before with model 1(Movie_effect + User_effect). No improvement here as well.

RESULTS

1. Adding the movie age feature as variable is slightly lowering the RMSE which is lowered from 0.8258487 (0.826) to 0.825298 (0.825), a improvement of only 0.06 %. This improvement is not significant. Adding the additional feature Movie age is not improving the model significantly.
2. Using the model with the full edx dataset inculing the outliers is also yielding the same RMSE = 0.8258487 . It did not improve RMSE on model 1 (Movie_effect + User_effect).

PROJECT CONCLUSIONS

lets put the results side to side for comparison

```
# Computing the 2 models results side by side

#model 1:  Predicted_rating = intercept + movie_effect + user_effect =  mu + b_m + b_u
movie_User_effect <- RMSE(predicted_rating, validation$rating)
model_1_results <- data_frame(method = "Movie + User_effects", RMSE = movie_User_effect)

# model 2 : Model 2 : predicted_rating = intercept + movie_effect + user_effect  + age_effect=  mu + b_m +
b_u + b_a
movie_user_age_effect <- RMSE2(predicted_rating2, validation2$rating)
model_2_results <- data_frame(method = "Movie + User + age_effects", RMSE2 = movie_user_age_effect)
```

This yield the below table

```
## # A tibble: 2 x 3
##   method                       RMSE  RMSE2
##   <chr>                       <dbl>  <dbl>
## 1 Movie + User_effects        0.826 NA
## 2 Movie + User + age_effects NA      0.825
```

We will keep the model 1 (movie_effect + user_effect) : **Predicted_rating = mu + b_m + b_u** since adding movie age did not significantly improve the RMSE.

We were able to get a **RMSE = 0.8258487** (rounded to 0.826 in the above computation) using the movie_effect(b_m) and the user_effect (b_u). This is an improvement on the RMSE target assigned.