Movielens Capstone Project Report for HarvardX: PH125.9X

Silvia Elaluf-Calderwood

1/7/2020

1. Introduction: MovieLens Data Set

Motivation Our motivation is to complete the requirements for the final marking for Data Science: Capstone HarvardX - PH125.9x. This assignment using the MovieLens data set was given to us. As a backgroun note this data set was originally collected by GroupLens Research. Access to it is public and widely used by Data Science students and teachers.

Goals The goals of the exercise is to answer the general question:**Can we predict movie ratings based on user preferance, age of a movie?**

Considerations, data set overall factors and methodology

With the MovieLens data set and using penalized least squares, I have proceed to elaborate the following R script, which calculates the RMSE. The RMSE takes into account as primaryu parameters: user ratings, movieId and the age of the movie.

The MovieLens data set contains 10000054 rows, 10677 movies, 797 genres and 69878 users.

Our methodology to proceed to do the data analysis can be listed with the following steps:

In the cleaned and checked data set, taking the data to a movies data set:

Step 1: An age of movie column is created in the movies data set.

Step 2: Building up graphic displays of movie, users and ratings are helpful to identify data patterns and/or insights on data behaviour.

Step 3: Proceed to an exploration of data set using the category Genres to determine if ratings could be predicted by genre.

Step 4: Proceed to explore the Coefficient of Determination R-Squared by two methods:

4.1 Graphically explored the linear correlation coefficient, r-value 4.2 Calculate RMSE based on movieId, userId, and age of the movie.

Analysis and Results

Methology evaluation: post exploration the MovieLens data set through graphical representations and calculating RMSE, there was enough evidence to choose as the best set predictor for ratings two categories: movieId, userId. Experimentally the analysis showsthe age of the movie didn’t change the rmse.

The final RMSE is 0.8252

** Additional libraries used for the analysis**

The following are the libraries were used to explore the data.

Preliminary data explorations that did not seem to lead to an insightful result were taken out of the script.

Preliminary work: DownLoading the MovieLens data set

```
dl <- tempfile()
download.file("http://files.grouplens.org/datasets/movielens/ml-10m.zip", dl)
ratings <- read.table(text = gsub("::", "\t", readLines(unzip(dl, "ml-10M100K/rating
s.dat"))),
                      col.names = c("userId", "movieId", "rating", "timestamp"))
```

Preliminary work: Build the movies data set

```
movies <- str_split_fixed(readLines(unzip(dl, "ml-10M100K/movies.dat")), "\\::", 3)
colnames(movies) <- c("movieId", "title", "genres")
movies <- as.data.frame(movies) %>% mutate(movieId = as.numeric(levels(movieId))[movi
eId],
                                           title = as.character(title),
                                           genres = as.character(genres))
#Explore the size of the data set
movielens <- left_join(ratings, movies, by = "movieId")
nrow(movielens)
```

```
## [1] 10000054
```

```
n_distinct(movielens$movieId)
```

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

```
n_distinct(movielens$genres)
```

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

```
n_distinct(movielens$userId)
```

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

Establishing the Validation: It was set to be 10% of the movieLens data

```
set.seed(1)
test_index <- createDataPartition(y = movielens$rating, times = 1, p = 0.1, list = FA
LSE)
edx <- movielens[-test_index,]
temp <- movielens[test_index,]
```

Making sure userId and movieId in validation set are also in edx set

```
validation <- temp %>%
  semi_join(edx, by = "movieId") %>%
  semi_join(edx, by = "userId")
```

Add rows removed from validation set back into edx set

```
removed <- anti_join(temp, validation)
```

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

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

Data Cleaning and, data exploration and Data Visulization

In order to determine if age of the movie is a factor for predicting rating, the premier date of the movie was extracted, and then calculated the age of the movie.

Additional exploration work was done looking into individual genres for genre effect, as well as, effects of user ratings.

```
head(edx)
```

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

```
glimpse(edx)
```

```
## Observations: 9,000,061
## Variables: 6
## $ userId    <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,…
## $ movieId   <dbl> 122, 185, 231, 292, 316, 329, 355, 356, 362, 364, 370, 377,…
## $ rating    <dbl> 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,…
## $ timestamp <int> 838985046, 838983525, 838983392, 838983421, 838983392, 8389…
## $ title     <chr> "Boomerang (1992)", "Net, The (1995)", "Dumb & Dumber (1994…
## $ genres    <chr> "Comedy|Romance", "Action|Crime|Thriller", "Comedy", "Actio…
```

How many distinct movie, users and genres

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

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

```
n_distinct(edx$genres)
```

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

```
n_distinct(edx$userId)
```

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

```
nrow(edx)
```

```
## [1] 9000061
```

Convert Timestamp to year

```
edx <- mutate(edx, year_rated = year(as_datetime(timestamp)))
head(edx)
```

```
##   userId movieId rating timestamp                         title
## 1      1     122      5 838985046              Boomerang (1992)
## 2      1     185      5 838983525               Net, The (1995)
## 3      1     231      5 838983392          Dumb & Dumber (1994)
## 4      1     292      5 838983421               Outbreak (1995)
## 5      1     316      5 838983392               Stargate (1994)
## 6      1     329      5 838983392 Star Trek: Generations (1994)
##                            genres year_rated
## 1                  Comedy|Romance       1996
## 2           Action|Crime|Thriller       1996
## 3                          Comedy       1996
## 4    Action|Drama|Sci-Fi|Thriller       1996
## 5         Action|Adventure|Sci-Fi       1996
## 6 Action|Adventure|Drama|Sci-Fi       1996
```

Extract the premier date and calculate the age of the movie. Explore whether or not the age of the movie effects predicted ratings

```
#extracting the premier date
premier <- stringi::stri_extract(edx$title, regex = "(\\d{4})", comments = TRUE ) %>%
as.numeric()
#Add the premier date
edx_with_title_dates <- edx %>% mutate(premier_date = premier)
head(edx_with_title_dates)
```

```
##   userId movieId rating timestamp                         title
## 1      1     122      5 838985046               Boomerang (1992)
## 2      1     185      5 838983525                Net, The (1995)
## 3      1     231      5 838983392           Dumb & Dumber (1994)
## 4      1     292      5 838983421                Outbreak (1995)
## 5      1     316      5 838983392                Stargate (1994)
## 6      1     329      5 838983392 Star Trek: Generations (1994)
##                          genres year_rated premier_date
## 1                Comedy|Romance       1996         1992
## 2         Action|Crime|Thriller       1996         1995
## 3                        Comedy       1996         1994
## 4  Action|Drama|Sci-Fi|Thriller       1996         1995
## 5       Action|Adventure|Sci-Fi       1996         1994
## 6 Action|Adventure|Drama|Sci-Fi       1996         1994
```

After extracting the premier date from the title, check for accuracy

```
#drop the timestamp
edx_with_title_dates <- edx_with_title_dates %>% select(-timestamp)
head(edx_with_title_dates)
```

```
##   userId movieId rating                         title
## 1      1     122      5               Boomerang (1992)
## 2      1     185      5                Net, The (1995)
## 3      1     231      5           Dumb & Dumber (1994)
## 4      1     292      5                Outbreak (1995)
## 5      1     316      5                Stargate (1994)
## 6      1     329      5 Star Trek: Generations (1994)
##                          genres year_rated premier_date
## 1                Comedy|Romance       1996         1992
## 2         Action|Crime|Thriller       1996         1995
## 3                        Comedy       1996         1994
## 4  Action|Drama|Sci-Fi|Thriller       1996         1995
## 5       Action|Adventure|Sci-Fi       1996         1994
## 6 Action|Adventure|Drama|Sci-Fi       1996         1994
```

```
#looking at the dates - are they correct?
edx_with_title_dates %>% filter(premier_date > 2018) %>% group_by(movieId, title, pre
mier_date) %>% summarize(n = n())
```

```
## # A tibble: 6 x 4
## # Groups:   movieId, title [6]
##   movieId title                                              premier_date     n
##     <dbl> <chr>                                                     <dbl> <int>
## 1     671 Mystery Science Theater 3000: The Movie (1996)         3000  3266
## 2    2308 Detroit 9000 (1973)                                    9000    22
## 3    4159 3000 Miles to Graceland (2001)                         3000   714
## 4    5310 Transylvania 6-5000 (1985)                             5000   197
## 5    8864 Mr. 3000 (2004)                                        3000   155
## 6   27266 2046 (2004)                                            2046   422
```

```
edx_with_title_dates %>% filter(premier_date < 1900) %>% group_by(movieId, title, pre
mier_date) %>% summarize(n = n())
```

```
## # A tibble: 8 x 4
## # Groups:   movieId, title [8]
##   movieId title                                                premier_date     n
##     <dbl> <chr>                                                       <dbl> <int>
## 1    1422 Murder at 1600 (1997)                                       1600  1552
## 2    4311 Bloody Angels (1732 Høtten: Marerittet Har et Post…         1732     8
## 3    5472 1776 (1972)                                                 1776   184
## 4    6290 House of 1000 Corpses (2003)                                1000   371
## 5    6645 THX 1138 (1971)                                             1138   467
## 6    8198 1000 Eyes of Dr. Mabuse, The (Tausend Augen des Dr…         1000    26
## 7    8905 1492: Conquest of Paradise (1992)                           1492   141
## 8   53953 1408 (2007)                                                 1408   465
```

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

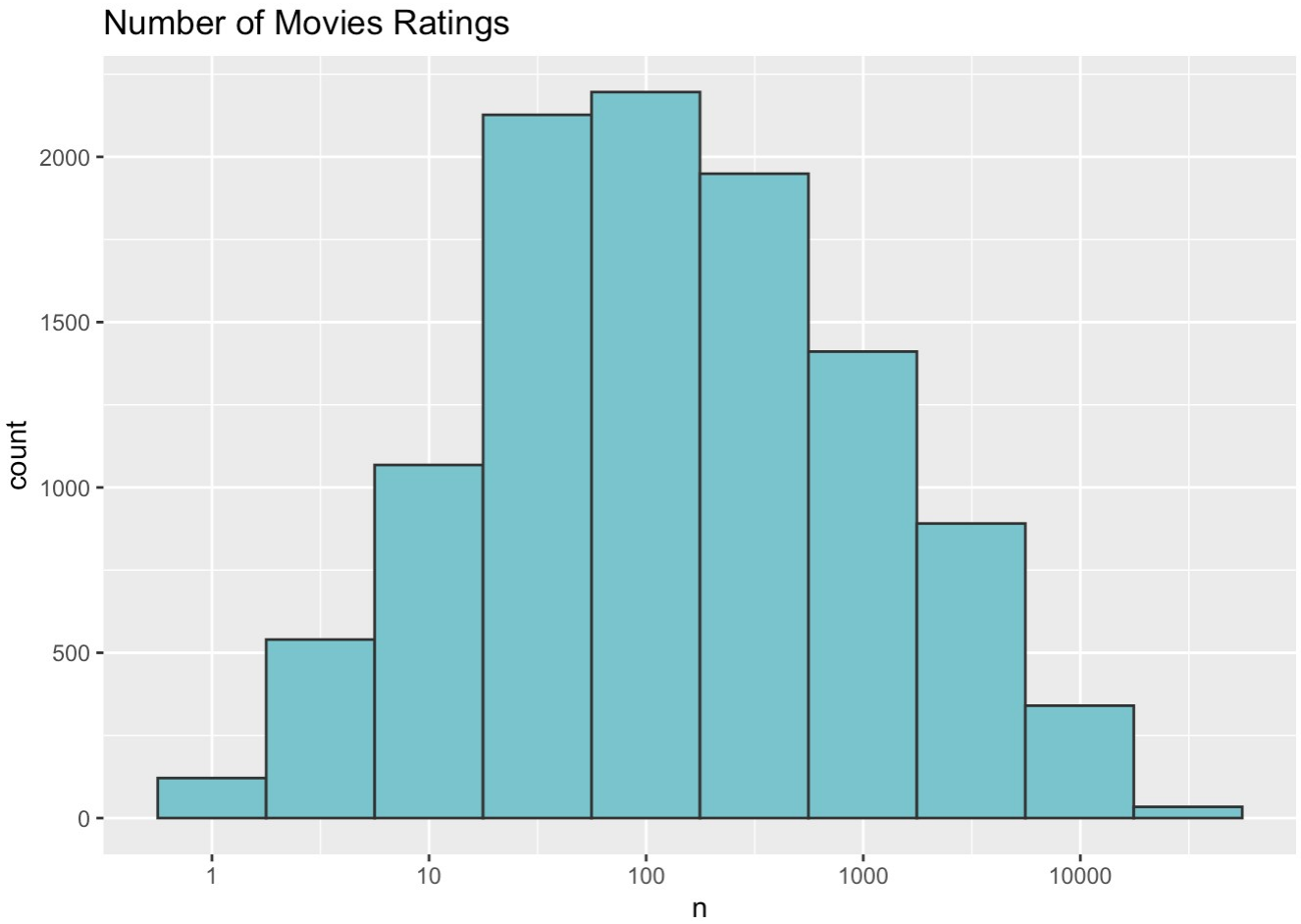
Calculate the age of the movie

```
#Calculate the age of a movie
edx_with_title_dates <- edx_with_title_dates %>% mutate(age_of_movie = 2018 - premier
_date,
                                                        rating_date_range = year_rate
d - premier_date)
head(edx_with_title_dates)
```

```
##   userId movieId rating                         title
## 1      1     122      5              Boomerang (1992)
## 2      1     185      5               Net, The (1995)
## 3      1     231      5          Dumb & Dumber (1994)
## 4      1     292      5               Outbreak (1995)
## 5      1     316      5               Stargate (1994)
## 6      1     329      5 Star Trek: Generations (1994)
##                          genres year_rated premier_date age_of_movie
## 1                Comedy|Romance       1996         1992           26
## 2         Action|Crime|Thriller       1996         1995           23
## 3                        Comedy       1996         1994           24
## 4  Action|Drama|Sci-Fi|Thriller       1996         1995           23
## 5       Action|Adventure|Sci-Fi       1996         1994           24
## 6 Action|Adventure|Drama|Sci-Fi       1996         1994           24
##   rating_date_range
## 1                 4
## 2                 1
## 3                 2
## 4                 1
## 5                 2
## 6                 2
```

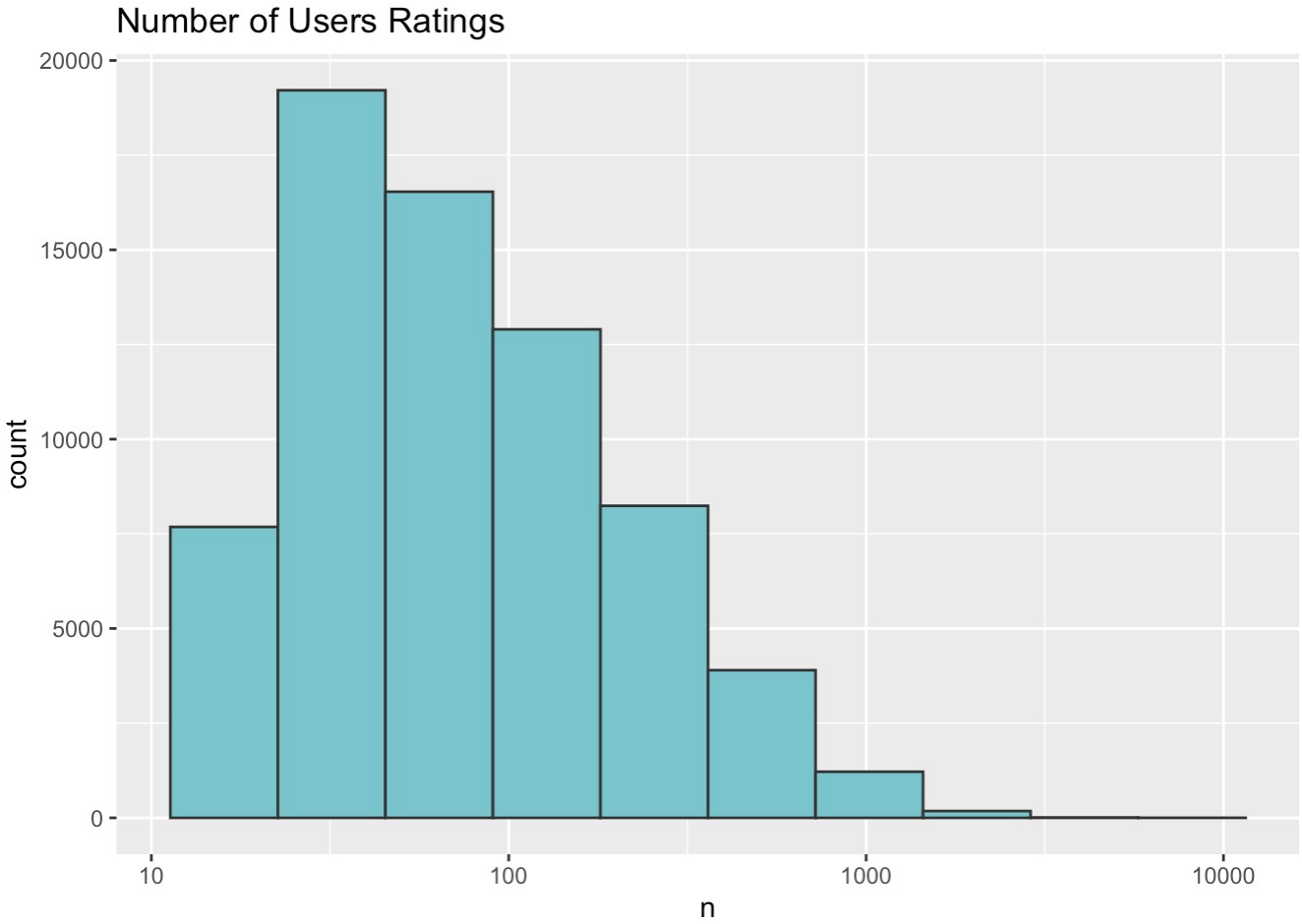
Graph the data

```
#Distribution of Movie Ratings
edx %>% group_by(movieId) %>% summarize(n = n()) %>%
  ggplot(aes(n)) + geom_histogram(fill = "cadetblue3", color = "grey20", bins = 10) +
  scale_x_log10() +
  ggtitle("Number of Movies Ratings")
```



#Number of Ratings by userId

```
#Distribution of Users
edx %>% group_by(userId) %>% summarize(n = n()) %>%
  ggplot(aes(n)) + geom_histogram(fill = "cadetblue3", color = "grey20", bins = 10) +
  scale_x_log10() +
  ggtitle("Number of Users Ratings")
```



Calculate movie rating average, user rating average, average rating by age of movie, average rating by year

```
#Movie rating averages
movie_avgs <- edx_with_title_dates %>% group_by(movieId) %>% summarize(avg_movie_rati
ng = mean(rating))
user_avgs <- edx_with_title_dates %>% group_by(userId) %>% summarize(avg_user_rating
= mean(rating))
year_avgs <- edx_with_title_dates%>% group_by(year_rated) %>% summarize(avg_rating_by
_year = mean(rating)) #year the movie was rated
age_avgs <- edx_with_title_dates %>% group_by(age_of_movie) %>% summarize(avg_rating_
by_age = mean(rating)) #age of movie
head(age_avgs)
```

```
## # A tibble: 6 x 2
##   age_of_movie avg_rating_by_age
##          <dbl>             <dbl>
## 1            8              3.36
## 2           10              3.46
## 3           11              3.53
## 4           12              3.53
## 5           13              3.48
## 6           14              3.53
```

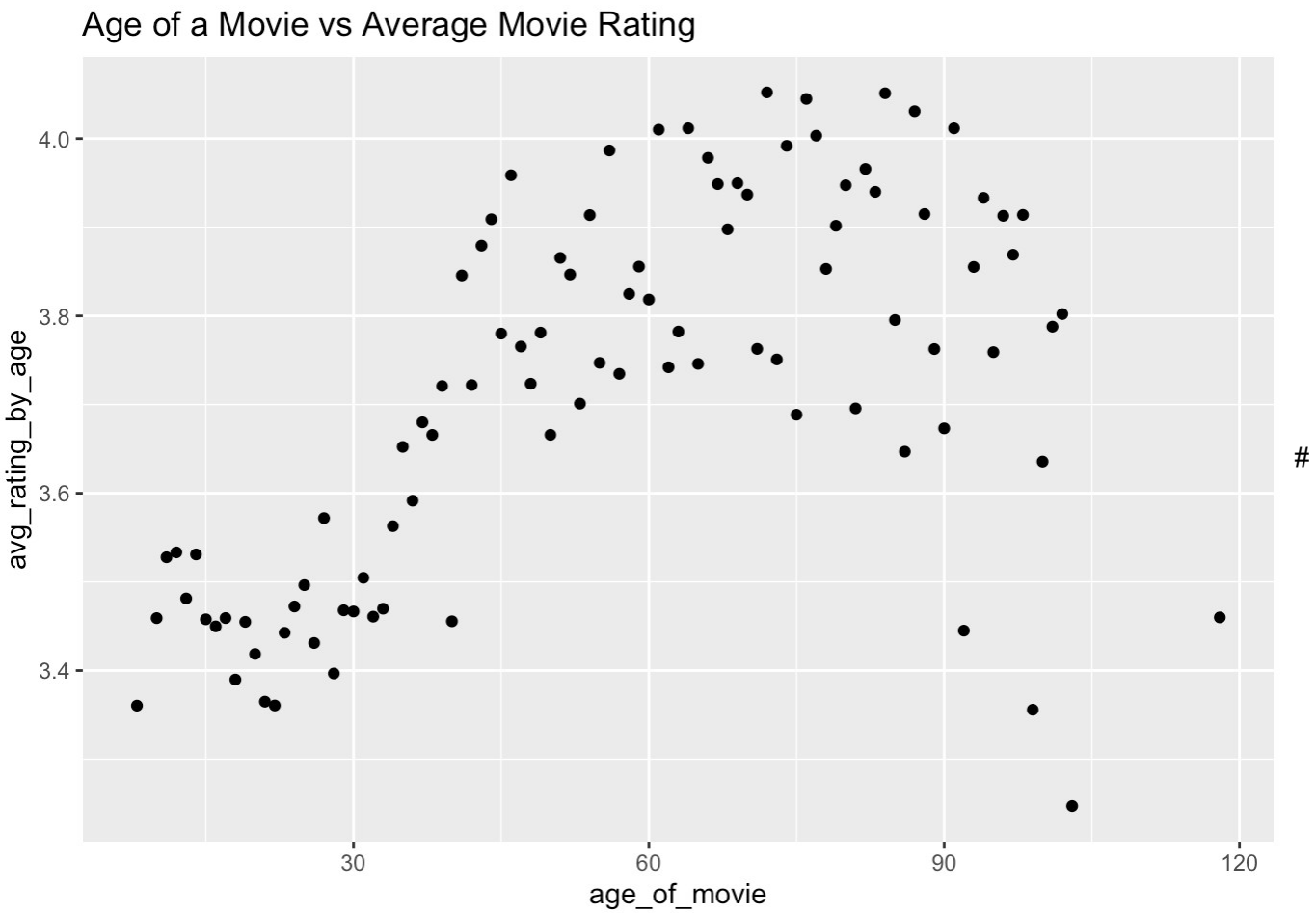
```
head(user_avgs)
```

```
## # A tibble: 6 x 2
##   userId avg_user_rating
##    <int>           <dbl>
## 1      1            5
## 2      2            3.17
## 3      3            3.98
## 4      4            4.06
## 5      5            3.84
## 6      6            3.90
```

What is the relationship to the age of a movie and the movies average rating?

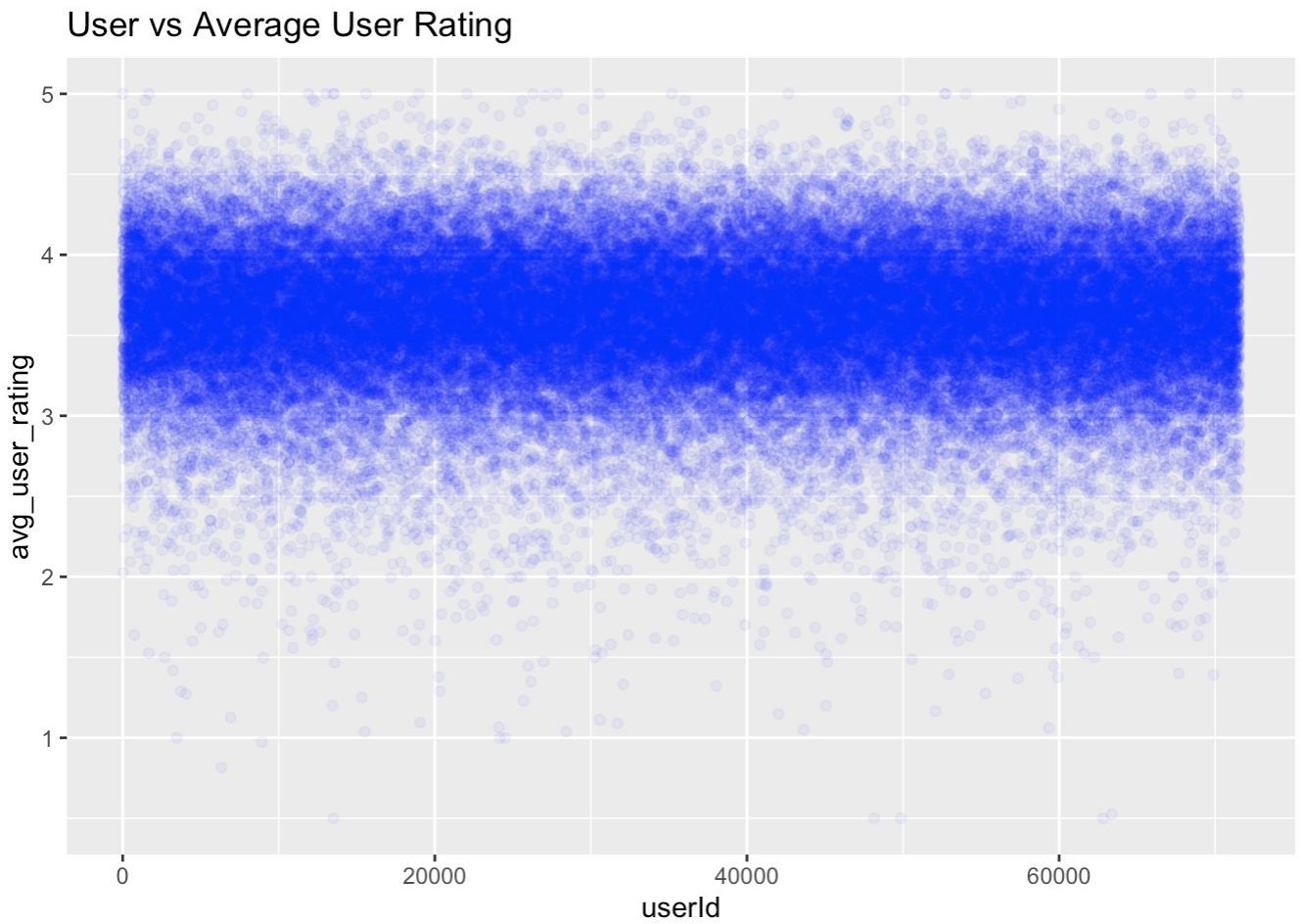
Graph age of movie vs average movie rating

```
# age of movie vs average movie rating
age_avgs %>%
  ggplot(aes(age_of_movie, avg_rating_by_age)) +
  geom_point() +
  ggtitle("Age of a Movie vs Average Movie Rating")
```



The above plot shows more variability as movies age. The plot, also, shows higher ratings the older a movies is up to 90 years old, then the ratings drop.

```
# userId vs average movie rating
user_avgs %>%
  ggplot(aes(userId, avg_user_rating)) +
  geom_point(alpha = 1/20, colour = "blue") +
  ggtitle("User vs Average User Rating")
```



#From the above graph, we can see average ratings by user are pretty consistent between 2.5 and 4.5

Calculating the lm of the age of a movie vs average rating

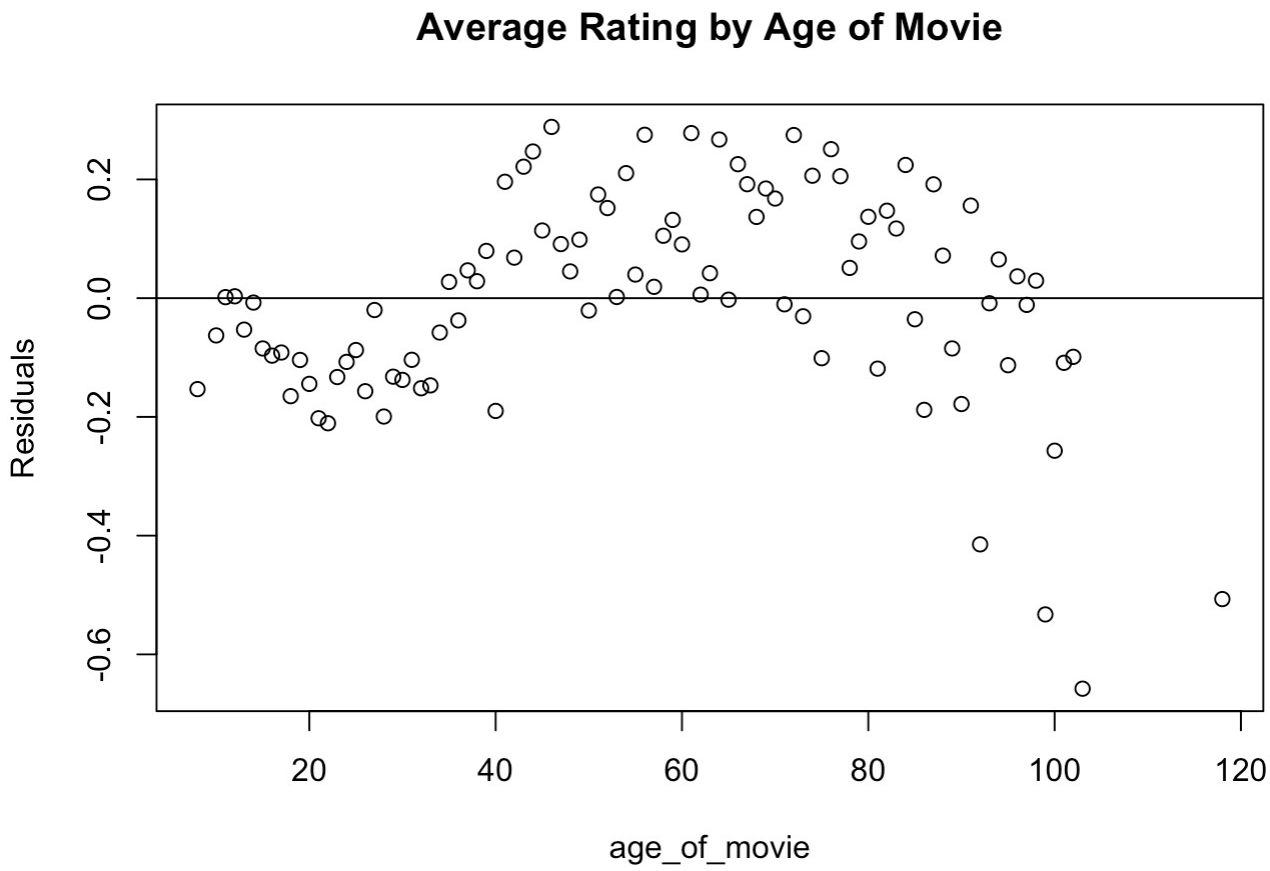
```
summary(lm(avg_rating_by_age ~ age_of_movie, data = age_avgs))
```

```
##
## Call:
## lm(formula = avg_rating_by_age ~ age_of_movie, data = age_avgs)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.65779 -0.10490  0.00254  0.13308  0.28844
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.4807249  0.0411521  84.582  < 2e-16 ***
## age_of_movie 0.0041193  0.0006513   6.325  8.5e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1788 on 94 degrees of freedom
## Multiple R-squared:  0.2985, Adjusted R-squared:  0.2911
## F-statistic:    40 on 1 and 94 DF,  p-value: 8.495e-09
```

We can see that R-square is small at 0.30

Plot the Residuals

```
avg_rating.lm <- lm(avg_rating_by_age ~ age_of_movie, data = age_avgs)
avg_rating.res <- resid(avg_rating.lm)
plot(age_avgs$age_of_movie, avg_rating.res,
     ylab='Residuals', xlab='age_of_movie',
     main = 'Average Rating by Age of Movie') + abline(0,0)
```

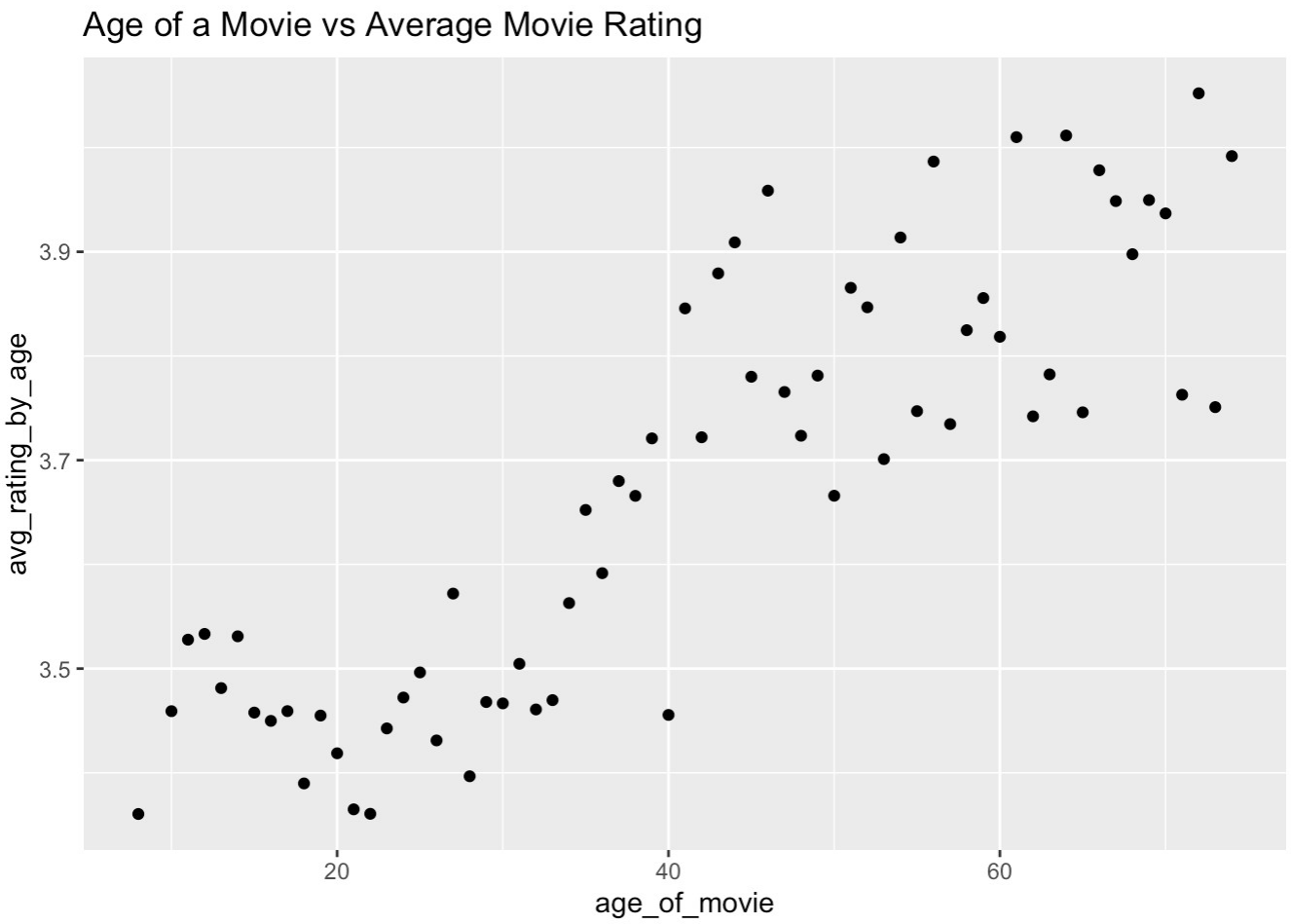



```
## integer(0)
```

The R-squared is fairly small at 0.30; 30% of the variation in movie ratings can be prdicted by

explore the data graphically to see if age of the movie and rating are coorelated

```
#Movies less than 75 years old
age_of_movie_less_than75 <- age_avgs %>% filter(age_of_movie <75)
# age of movie less than 75 years old vs average movie rating
age_of_movie_less_than75 %>%
  ggplot(aes(age_of_movie, avg_rating_by_age)) +
  geom_point() +
  ggtitle("Age of a Movie vs Average Movie Rating")
```



#Calculate the R-squared value

```
age_lessthan75_rating.lm <- lm(avg_rating_by_age ~ age_of_movie, data = age_of_movie_
less_than75)
summary(age_lessthan75_rating.lm)
```

```
##
## Call:
## lm(formula = avg_rating_by_age ~ age_of_movie, data = age_of_movie_less_than75)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.215669 -0.079879  0.002964  0.070316  0.240668
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.2944593  0.0308517  106.78   <2e-16 ***
## age_of_movie 0.0092071  0.0006757   13.63   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1047 on 64 degrees of freedom
## Multiple R-squared:  0.7437, Adjusted R-squared:  0.7397
## F-statistic: 185.7 on 1 and 64 DF,  p-value: < 2.2e-16
```

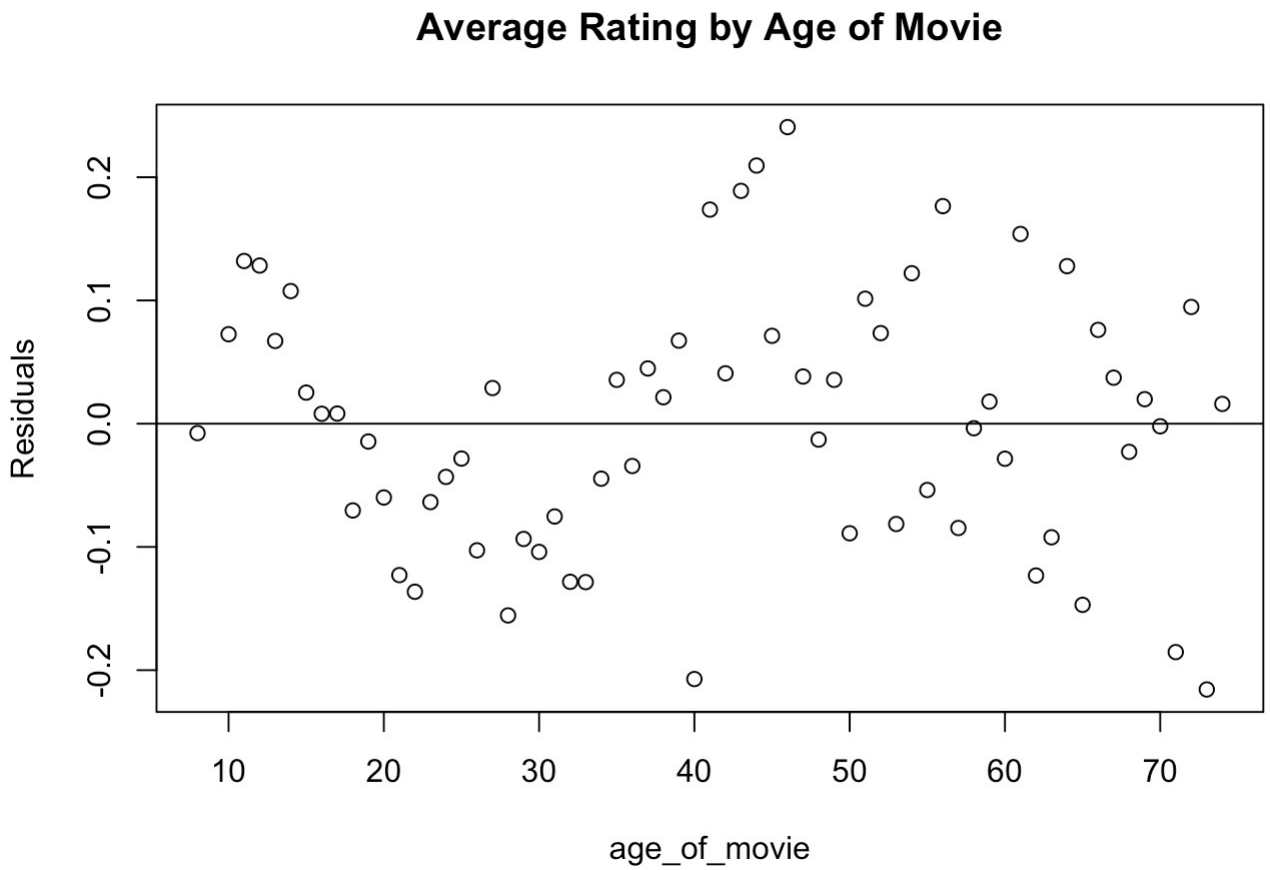
The R-squared increased to 0.745

Plot the residuals

```
head(age_of_movie_less_than75)
```

```
## # A tibble: 6 x 2
##   age_of_movie avg_rating_by_age
##          <dbl>             <dbl>
## 1            8              3.36
## 2           10              3.46
## 3           11              3.53
## 4           12              3.53
## 5           13              3.48
## 6           14              3.53
```

```
age_lessthan75.res <- resid(age_lessthan75_rating.lm)
plot(age_of_movie_less_than75$age_of_movie, age_lessthan75.res,
     ylab='Residuals', xlab='age_of_movie',
     main = 'Average Rating by Age of Movie') + abline(0,0)
```

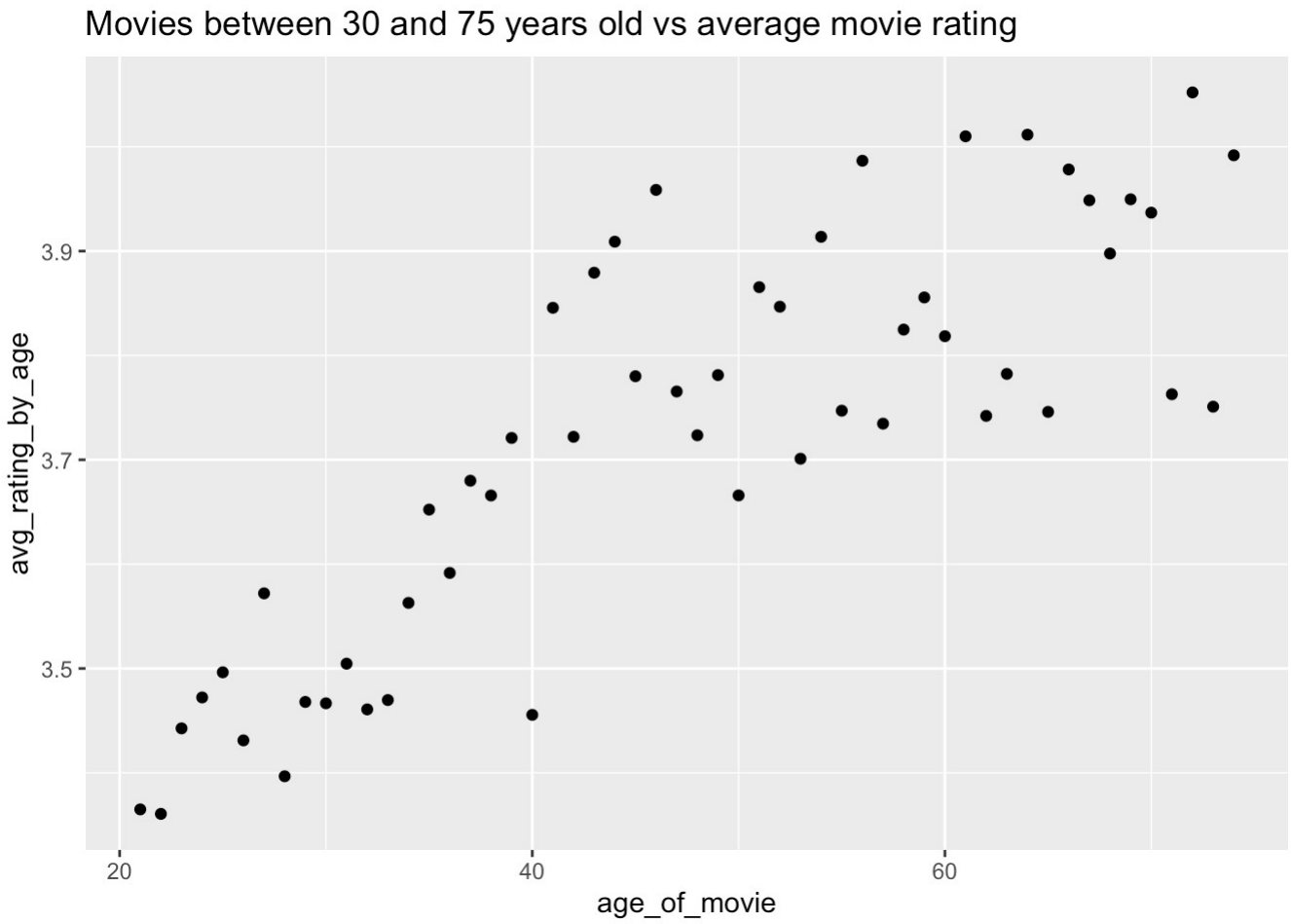


```
## integer(0)
```

Let's look at moveies between 20 and 75 years old as the graph looks more linear in that time frame

```
#Movies between 20 and 75 years old
age_between20_and_75 <- age_avgs %>% filter((age_of_movie > 20) & (age_of_movie < 75))
```

```
# graph the age of movie between 30 and 75 years old
age_between20_and_75 %>%
  ggplot(aes(age_of_movie, avg_rating_by_age)) +
  geom_point() + ggtitle("Movies between 30 and 75 years old vs average movie rating")
```



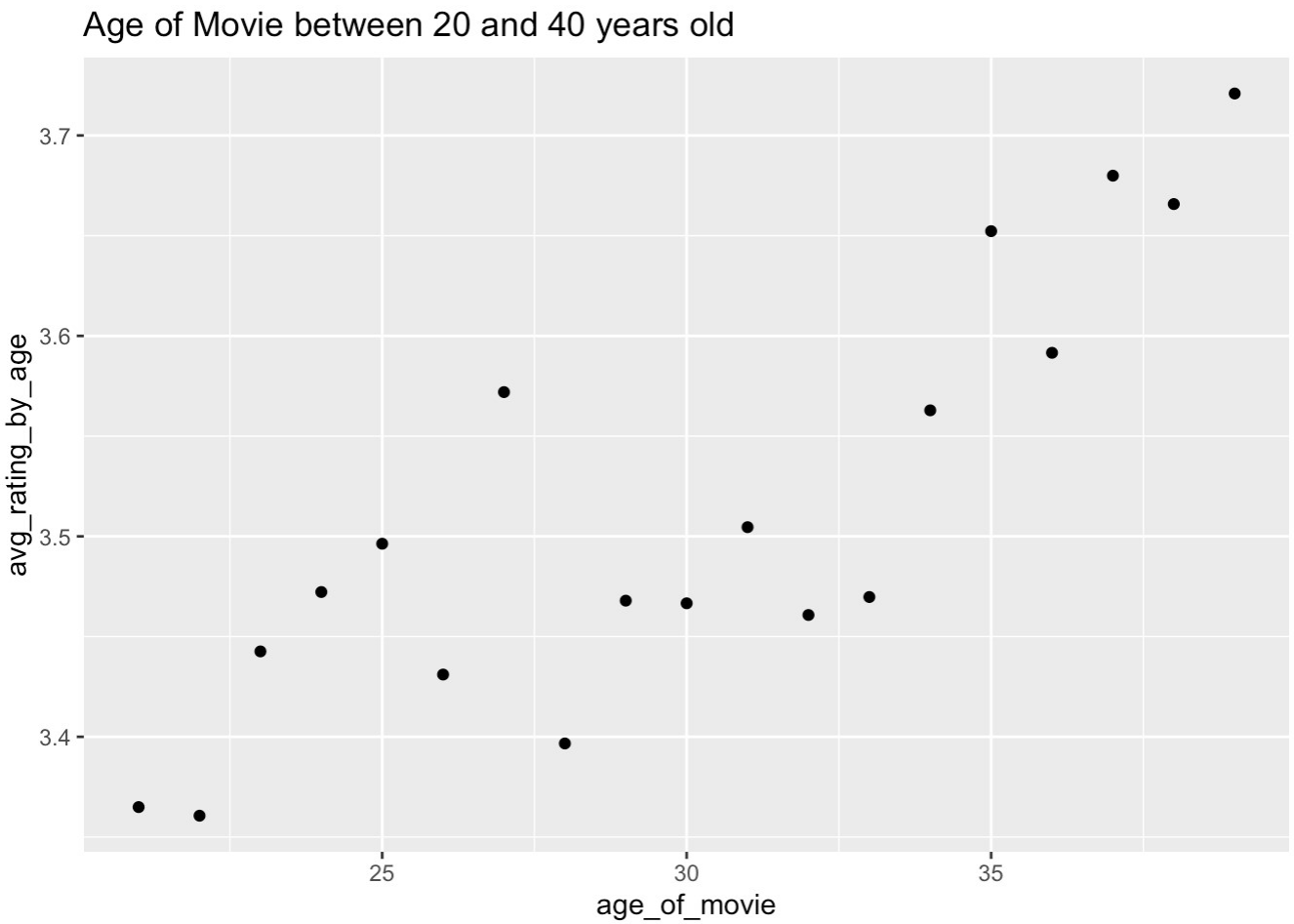
#The plot above appears to be a linear trend; however, the r-square is 0.69

```
summary(lm(avg_rating_by_age ~ age_of_movie, data = age_between20_and_75))
```

```
##
## Call:
## lm(formula = avg_rating_by_age ~ age_of_movie, data = age_between20_and_75)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.23729 -0.07944 -0.00750  0.06368  0.24972
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.2331481  0.0475495   68.00  < 2e-16 ***
## age_of_movie 0.0103432  0.0009511   10.87 5.31e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1089 on 52 degrees of freedom
## Multiple R-squared:  0.6946, Adjusted R-squared:  0.6887
## F-statistic: 118.3 on 1 and 52 DF,  p-value: 5.311e-15
```

The R-squared value is lower at 0.6981

```
# graph the age of movie between 20 and 40 years old
age_between20_and_40 <- age_avgs %>% filter((age_of_movie > 20) & (age_of_movie < 4
0))
age_between20_and_40 %>%
  ggplot(aes(age_of_movie, avg_rating_by_age)) +
  geom_point() + ggtitle('Age of Movie between 20 and 40 years old')
```



#The above graph is displying a linear trend with older movies having higher ratings

```
#calculate a linear model
summary(lm(avg_rating_by_age ~ age_of_movie, data = age_between20_and_40))
```

```
##
## Call:
## lm(formula = avg_rating_by_age ~ age_of_movie, data = age_between20_and_40)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.09332 -0.02845 -0.01631  0.05332  0.10562
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.031165   0.075224  40.295  < 2e-16 ***
## age_of_movie 0.016118   0.002467   6.534  5.1e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05889 on 17 degrees of freedom
## Multiple R-squared:  0.7152, Adjusted R-squared:  0.6985
## F-statistic:  42.7 on 1 and 17 DF,  p-value: 5.101e-06
```

The R-squared value is much higher than at 0.71

```
#Movies between 0 and 30 years old
age_less_than30 <- age_avgs %>% filter((age_of_movie < 30))
#Graph movies less than 30 years old and average movie rating
age_less_than30 %>%
  ggplot(aes(age_of_movie, avg_rating_by_age)) +
  geom_point() + ggtitle('Age of Movie less then 30 years old')
```

#For movies less than 30 years old there appears to be quite a bit of variation. We can see from the linear model that r-squared is nearly zero.

```
summary(lm(avg_rating_by_age ~ age_of_movie, data = age_less_than30))
```

```
##
## Call:
## lm(formula = avg_rating_by_age ~ age_of_movie, data = age_less_than30)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.099410 -0.034380  0.001252  0.024231  0.122832
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.4644023  0.0427367  81.064   <2e-16 ***
## age_of_movie -0.0005653  0.0021453  -0.263    0.795
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06033 on 19 degrees of freedom
## Multiple R-squared:  0.003641,   Adjusted R-squared:  -0.0488
## F-statistic: 0.06942 on 1 and 19 DF,  p-value: 0.795
```

The age of a movie did seem to effect the outcome of the average rating. This is possibly due to a higher number of ratings for older movies.

Do Genres have an effect on ratings?

I extracted the genres from the data with the idea to do an analysis on each genre. Some of the exploration I did here was removed as it didn't appear to effect the RMSE and this analysis keep growing! But I did get some nice graphs pertaining to genres.

```
#Genres split the data into single genres
dat <- edx_with_title_dates %>% separate_rows(genres, sep ="\\|")
head(dat)
```

```
##   userId movieId rating                title   genres year_rated premier_date
## 1      1     122      5     Boomerang (1992)   Comedy       1996         1992
## 2      1     122      5     Boomerang (1992)  Romance       1996         1992
## 3      1     185      5      Net, The (1995)   Action       1996         1995
## 4      1     185      5      Net, The (1995)    Crime       1996         1995
## 5      1     185      5      Net, The (1995) Thriller       1996         1995
## 6      1     231      5 Dumb & Dumber (1994)   Comedy       1996         1994
##   age_of_movie rating_date_range
## 1           26                 4
## 2           26                 4
## 3           23                 1
## 4           23                 1
## 5           23                 1
## 6           24                 2
```

Count the number of movies using movieId in each genre

```
genre_count_by_movieId <- dat %>% group_by(movieId, genres) %>% summarize(n = n())
head(genre_count_by_movieId)
```

```
## # A tibble: 6 x 3
## # Groups:   movieId [2]
##   movieId genres        n
##     <dbl> <chr>     <int>
## 1       1 Adventure 23826
## 2       1 Animation 23826
## 3       1 Children  23826
## 4       1 Comedy    23826
## 5       1 Fantasy   23826
## 6       2 Adventure 10717
```

Total number of movies in each genre

```
number_of_genres <- dat %>% group_by(genres) %>% summarize(n = n())
number_of_genres
```

```
## # A tibble: 20 x 2
##    genres                   n
##    <chr>                <int>
##  1 (no genres listed)       6
##  2 Action             2560649
##  3 Adventure          1908692
##  4 Animation           467220
##  5 Children            737851
##  6 Comedy             3541284
##  7 Crime              1326917
##  8 Documentary          93252
##  9 Drama              3909401
## 10 Fantasy             925624
## 11 Film-Noir           118394
## 12 Horror              691407
## 13 IMAX                  8190
## 14 Musical             432960
## 15 Mystery             567865
## 16 Romance            1712232
## 17 Sci-Fi             1341750
## 18 Thriller           2325349
## 19 War                 511330
## 20 Western             189234
```

List the genres. Movies are either in one genre or multiple genres

```
genre_list <- number_of_genres$genres
genre_list
```

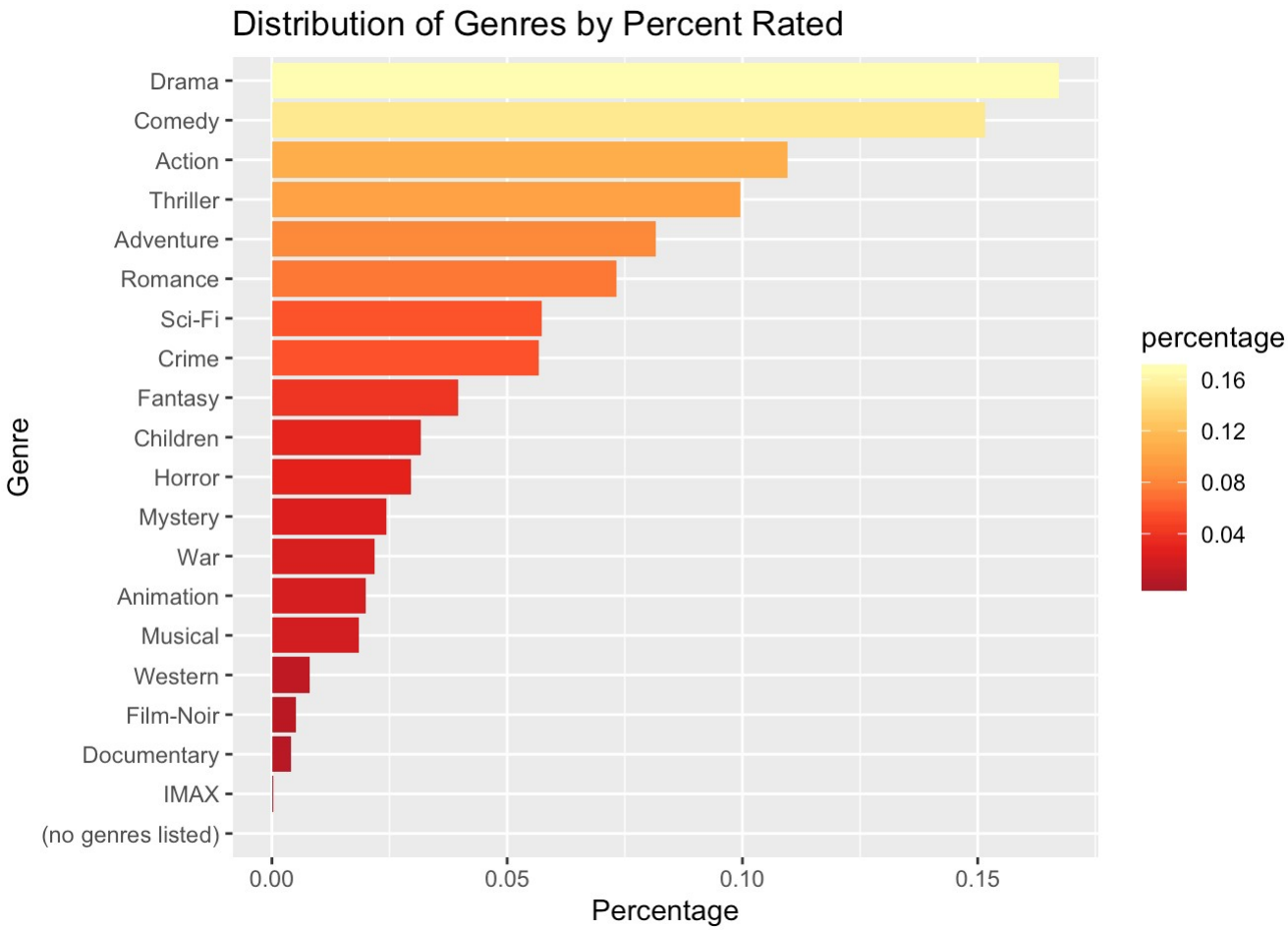
```
##  [1] "(no genres listed)" "Action"             "Adventure"
##  [4] "Animation"          "Children"           "Comedy"
##  [7] "Crime"              "Documentary"        "Drama"
## [10] "Fantasy"            "Film-Noir"          "Horror"
## [13] "IMAX"               "Musical"            "Mystery"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

Explore the distribution of ratings by genre

```
#Distribution of Ratings per Genre
temp <- dat %>%
  group_by(genres) %>%
  summarize(n=n()) %>%
  ungroup() %>%
  mutate(sumN = sum(n), percentage = n/sumN) %>%
  arrange(-percentage)
```

Bar Graph of Genre's

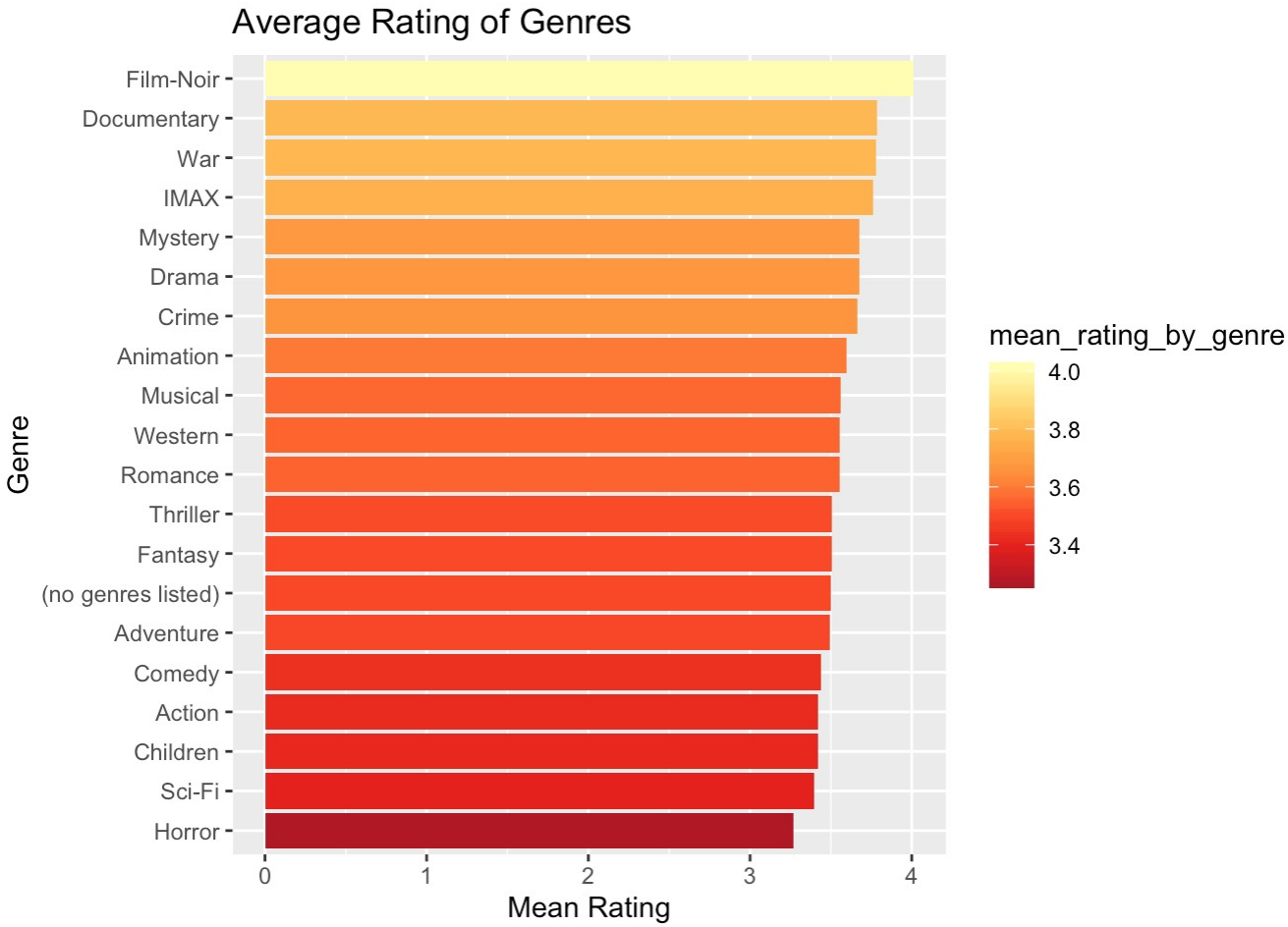
```
temp %>%
  ggplot(aes(reorder(genres, percentage), percentage, fill= percentage)) +
  geom_bar(stat = "identity") + coord_flip() +
  scale_fill_distiller(palette = "YlOrRd") + labs(y = "Percentage", x = "Genre") +
  ggtitle("Distribution of Genres by Percent Rated")
```



#From the above graph, we can see Drama had the highest percentage of ratings.

Genre's Mean rating

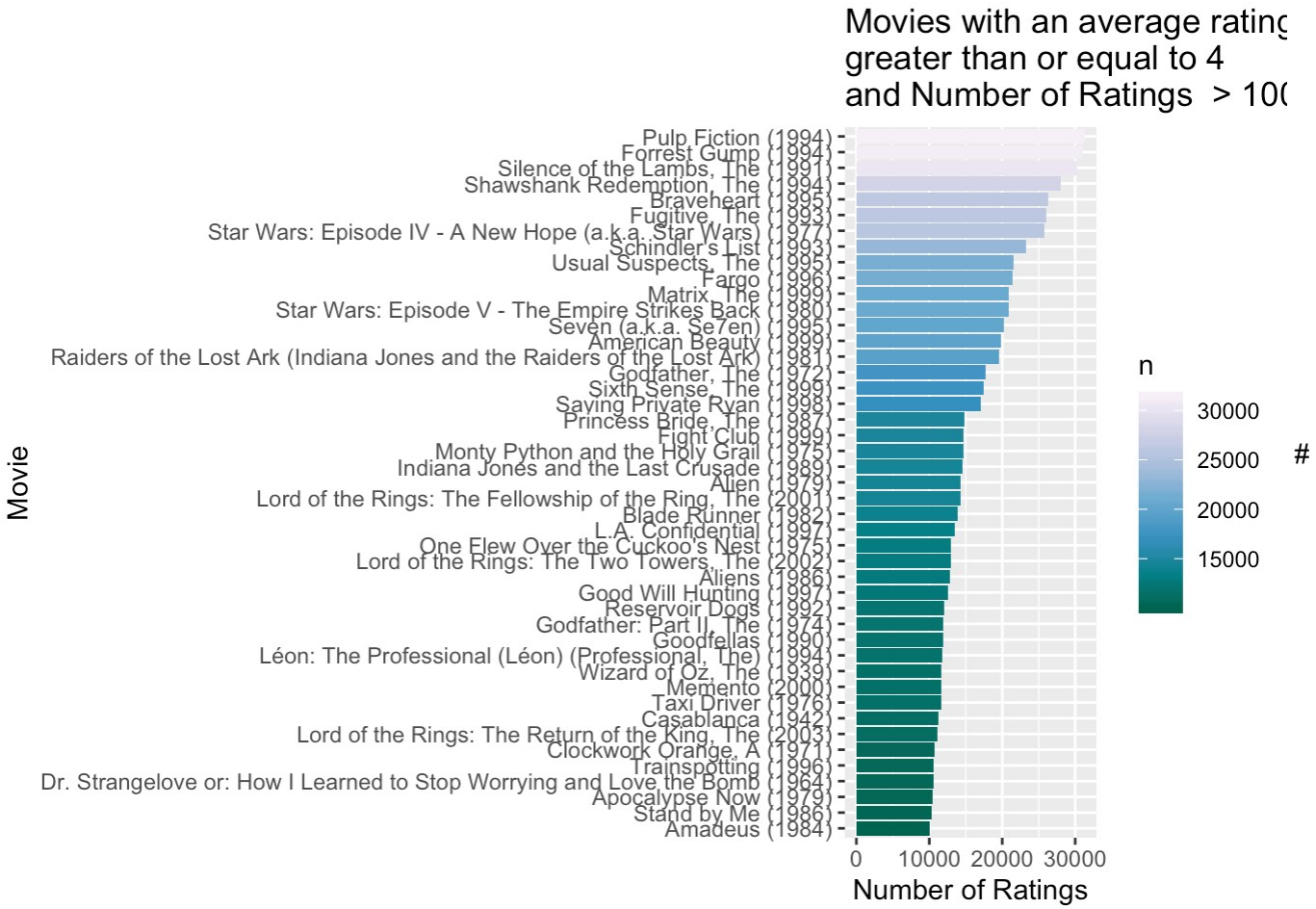
```
temp <- dat %>%
  group_by(genres) %>%
  summarize(mean_rating_by_genre=mean(rating)) %>%
  arrange(-mean_rating_by_genre)
temp %>%
  ggplot(aes(reorder(genres, mean_rating_by_genre), mean_rating_by_genre, fill= mean_
rating_by_genre)) +
  geom_bar(stat = "identity") + coord_flip() +
  scale_fill_distiller(palette = "YlOrRd") + labs(y = "Mean Rating", x = "Genre") +
  ggtitle("Average Rating of Genres")
```



#Film Noir had the highest average rating, while Horror had the lowest average rating.

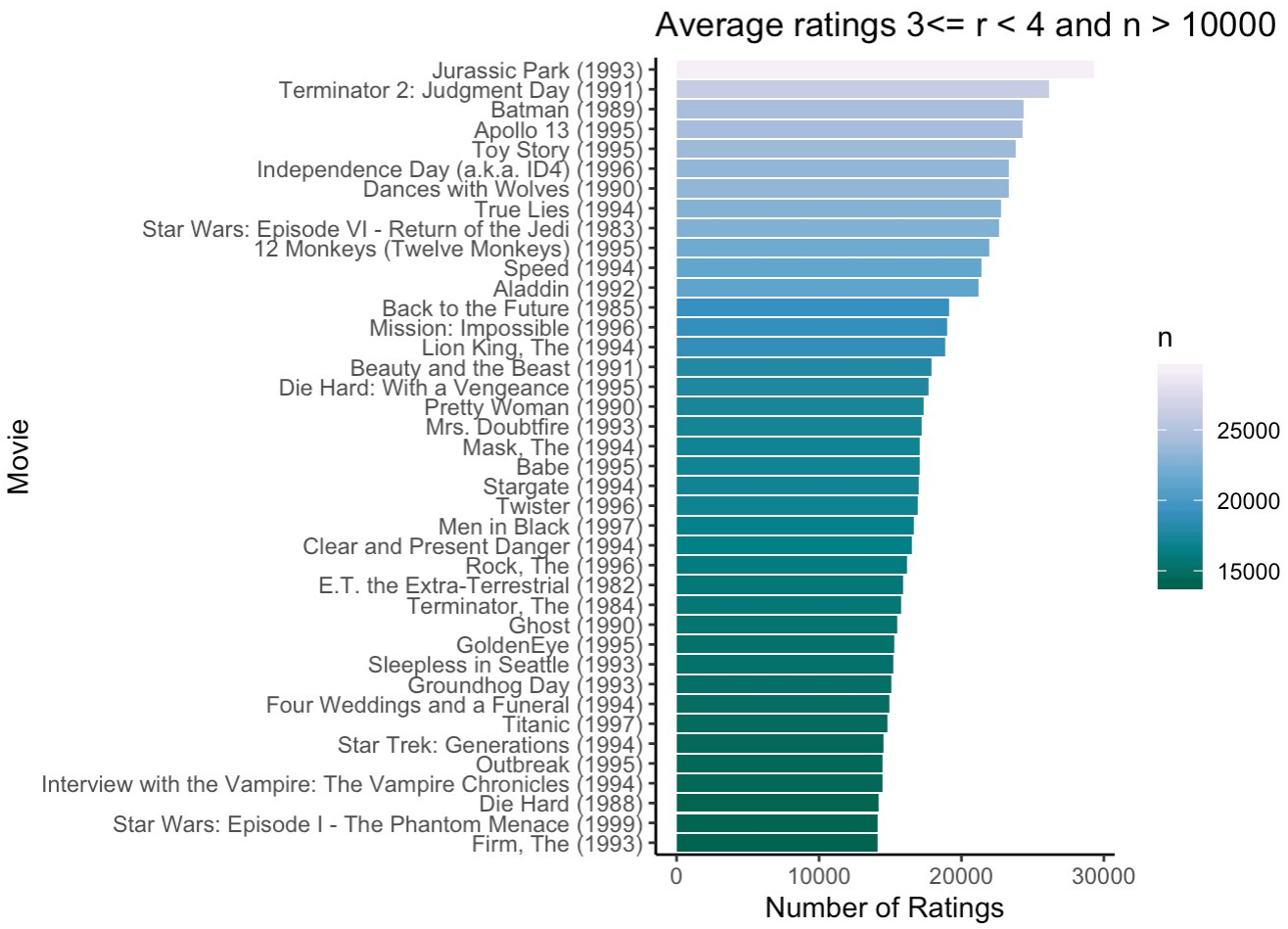
Explore movie ratings based on number of ratings and value of the rating

```
#Graph of movies with more than 10000 ratings and a mean rating greater than 4.
avg_rating_greater_than_4 <- edx %>% group_by(title) %>%
  summarize(mean_rating= mean(rating), n = n()) %>% filter(mean_rating >=4) %>% arran
ge(desc(n, mean_rating))
avg_rating_greater_than_4 %>% filter(n >=10000) %>%
  ggplot(aes(reorder(title, n), n, fill = n)) +
  geom_bar(stat = "identity") + coord_flip() + scale_fill_distiller(palette = "PuBuG
n") + xlab("Movie") +ylab('Number of Ratings') +
  ggtitle("Movies with an average rating\ngreater than or equal to 4\nand Number of R
atings  > 10000")
```



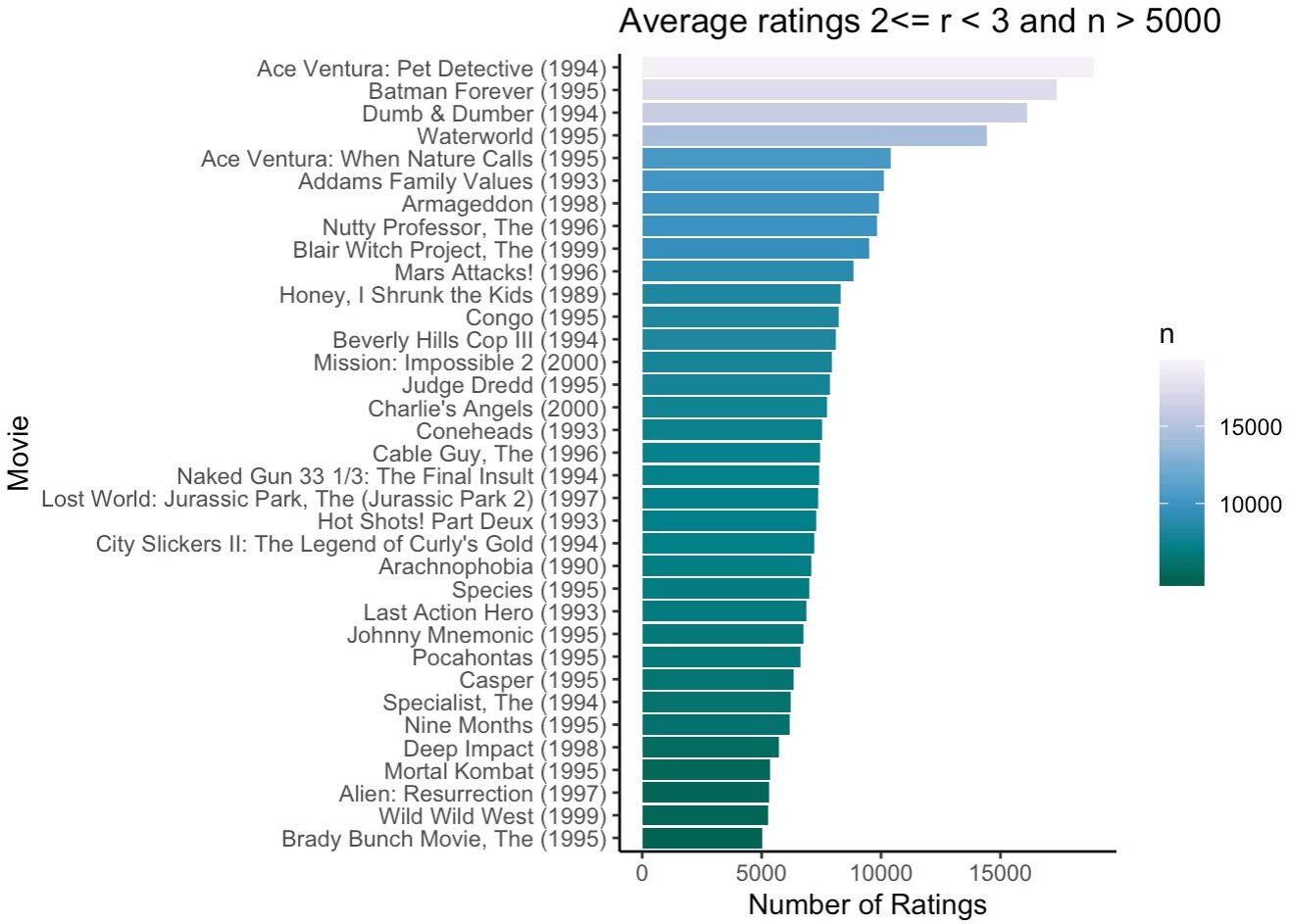
Examine Movies with ratings between 3 and 4 and more than 10000 ratings

```
avg_between3_4 <- edx %>% group_by(title) %>%
  summarize(mean_rating= mean(rating), n = n()) %>% filter(n > 10000, (mean_rating >=
3 & mean_rating < 4)) %>% arrange(desc(n, mean_rating))
p <- avg_between3_4 %>% slice(1:40)
p %>%
  ggplot(aes(reorder(title, n), n, fill = n)) +
  geom_bar(stat = "identity") + coord_flip() + scale_fill_distiller(palette = "PuBuG
n") +
  ggtitle("Average ratings 3<= r < 4 and n > 10000") + xlab('Movie') + ylab('Number o
f Ratings') +
  theme_classic()
```



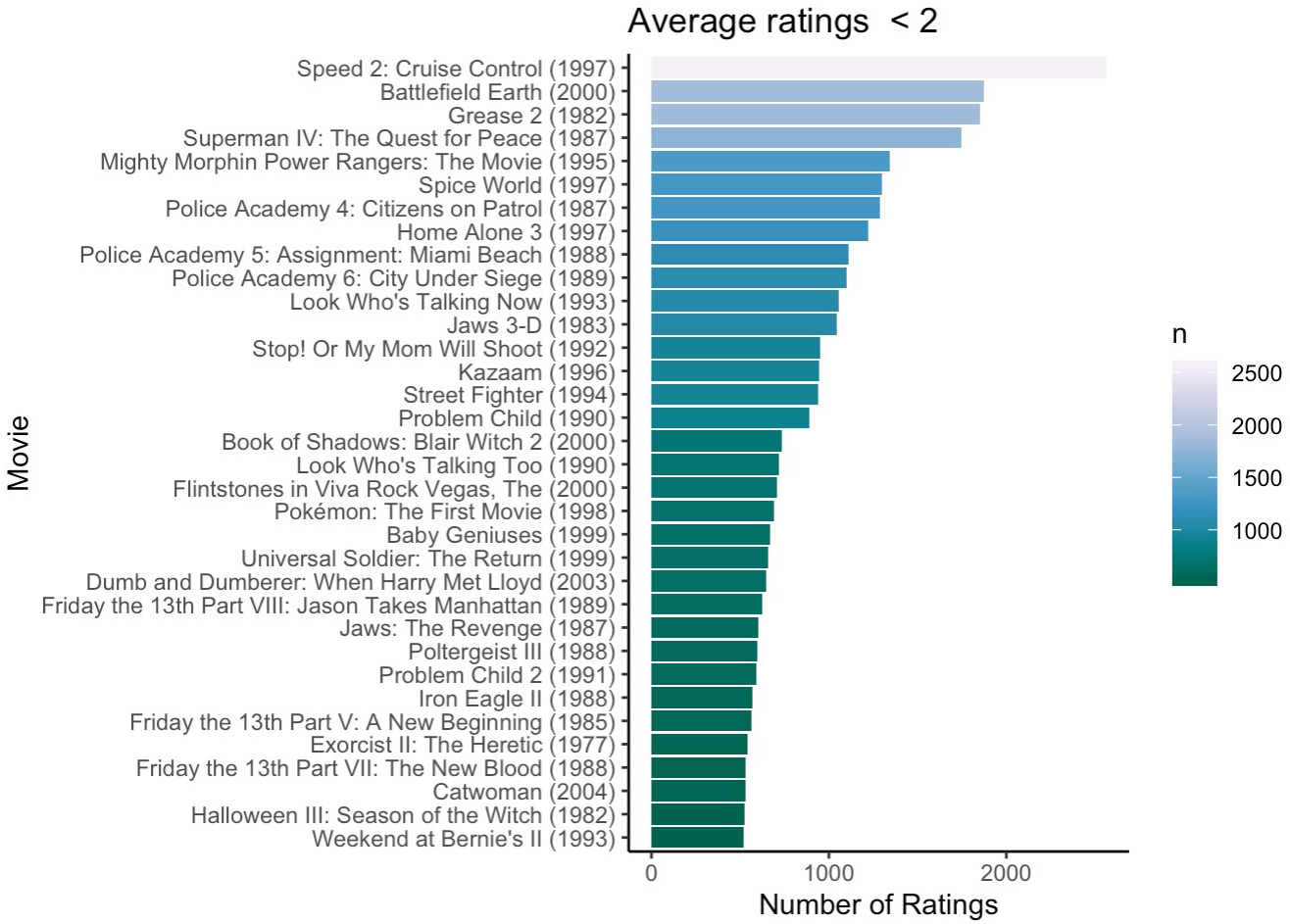
Movies with an average rating between 2 and 3 lets look at number of ratings greater than 5000

```
avg_between2_3 <- edx %>% group_by(title) %>%
  summarize(mean_rating= mean(rating), n = n()) %>% filter(n > 5000, (mean_rating >=
2 & mean_rating < 3)) %>% arrange(desc(n, mean_rating))
avg_between2_3 %>%
  ggplot(aes(reorder(title, n), n, fill = n)) +
  geom_bar(stat = "identity") + coord_flip() + scale_fill_distiller(palette = "PuBuG
n") +
  ggtitle("Average ratings 2<= r < 3 and n > 5000") + xlab('Movie') + ylab('Number of
Ratings') +
  theme_classic()
```



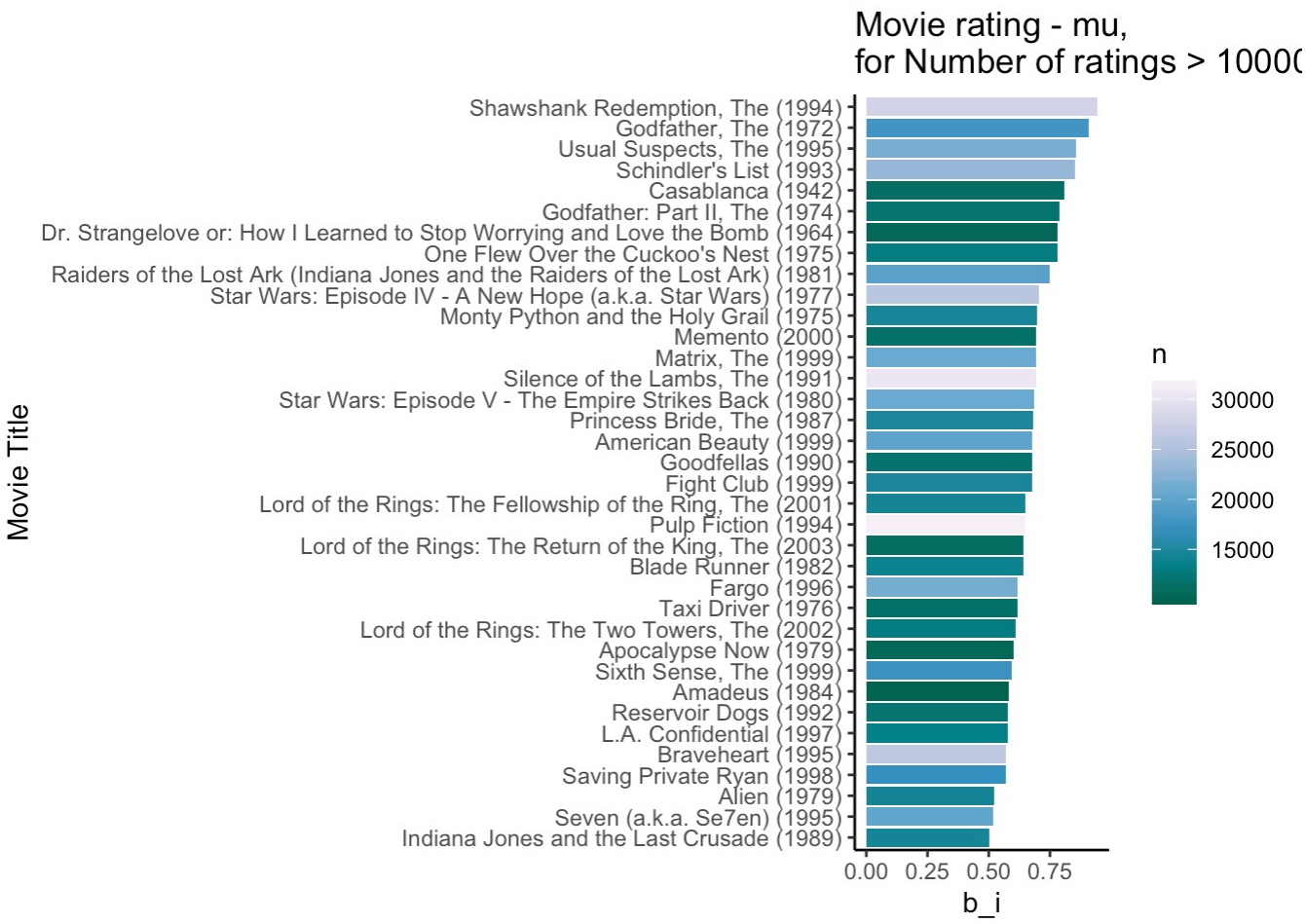
#Less than 10000 ratings and a rating less than 2 and number of ratings greater than 500

```
avg_rating_less_than_2 <- edx %>% group_by(title) %>%
  summarize(mean_rating= mean(rating), n = n()) %>% filter(n > 500, mean_rating < 2)
%>% arrange(desc(n, mean_rating))
avg_rating_less_than_2 %>%
  ggplot(aes(reorder(title, n), n, fill = n)) +
  geom_bar(stat = "identity") + coord_flip() + scale_fill_distiller(palette = "PuBuG
n") +
  ggtitle("Average ratings  < 2") + xlab('Movie') + ylab('Number of Ratings') +
  theme_classic()
```



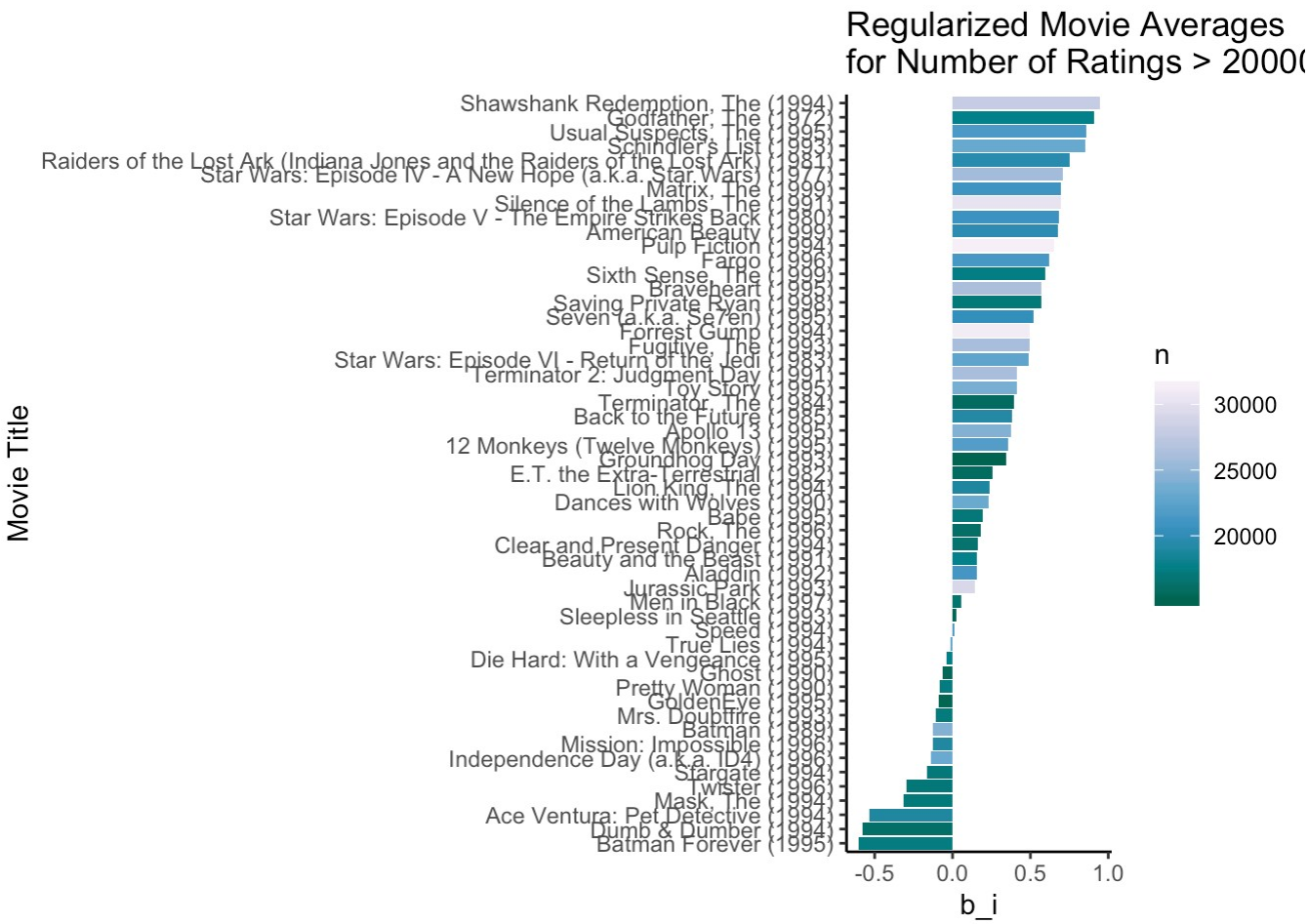
Compute the least squares for movieId

```
#Which movies have a large number of ratings and a rating larger than the average mu
mu <- mean(edx$rating)
edx %>% group_by(title) %>%
  summarize(b_i = mean(rating - mu), n = n()) %>% filter(b_i > 0.5, n > 10000) %>%
  ggplot(aes(reorder(title, b_i), b_i, fill = n)) +
  geom_bar(stat = "identity") + coord_flip() + scale_fill_distiller(palette = "PuBuG
n") +
  ggtitle("") + xlab("Movie Title") +
  ggtitle("Movie rating - mu,\nfor Number of ratings > 10000") +
  theme_classic()
```



#Regularized Movie Averages

```
movie_avgs <- edx %>% group_by(movieId) %>% summarize(b_i = mean(rating - mu))
movie_reg_avgs <- edx %>%
  group_by(movieId) %>%
  summarize(b_i = sum(rating - mu)/(n()+1), n_i = n())
movie_titles <- edx %>% select(movieId, title) %>% distinct()
edx_with_avgs <- edx %>% group_by(title, movieId) %>% summarize(n = n()) %>%
  left_join(movie_reg_avgs, by = "movieId") %>%
  arrange(desc(b_i, n))
edx_with_avgs %>% filter(n > 15000) %>%
  ggplot(aes(reorder(title, b_i), b_i, fill = n)) +
  geom_bar(stat = "identity") + coord_flip() + scale_fill_distiller(palette = "PuBuG
n") +
  ggtitle("") + xlab("Movie Title") + ggtitle('Regularized Movie Averages\nfor Number
of Ratings > 20000') +
  theme_classic()
```



#Regularized Movie Averages for the movies with regularized ratings less than 2

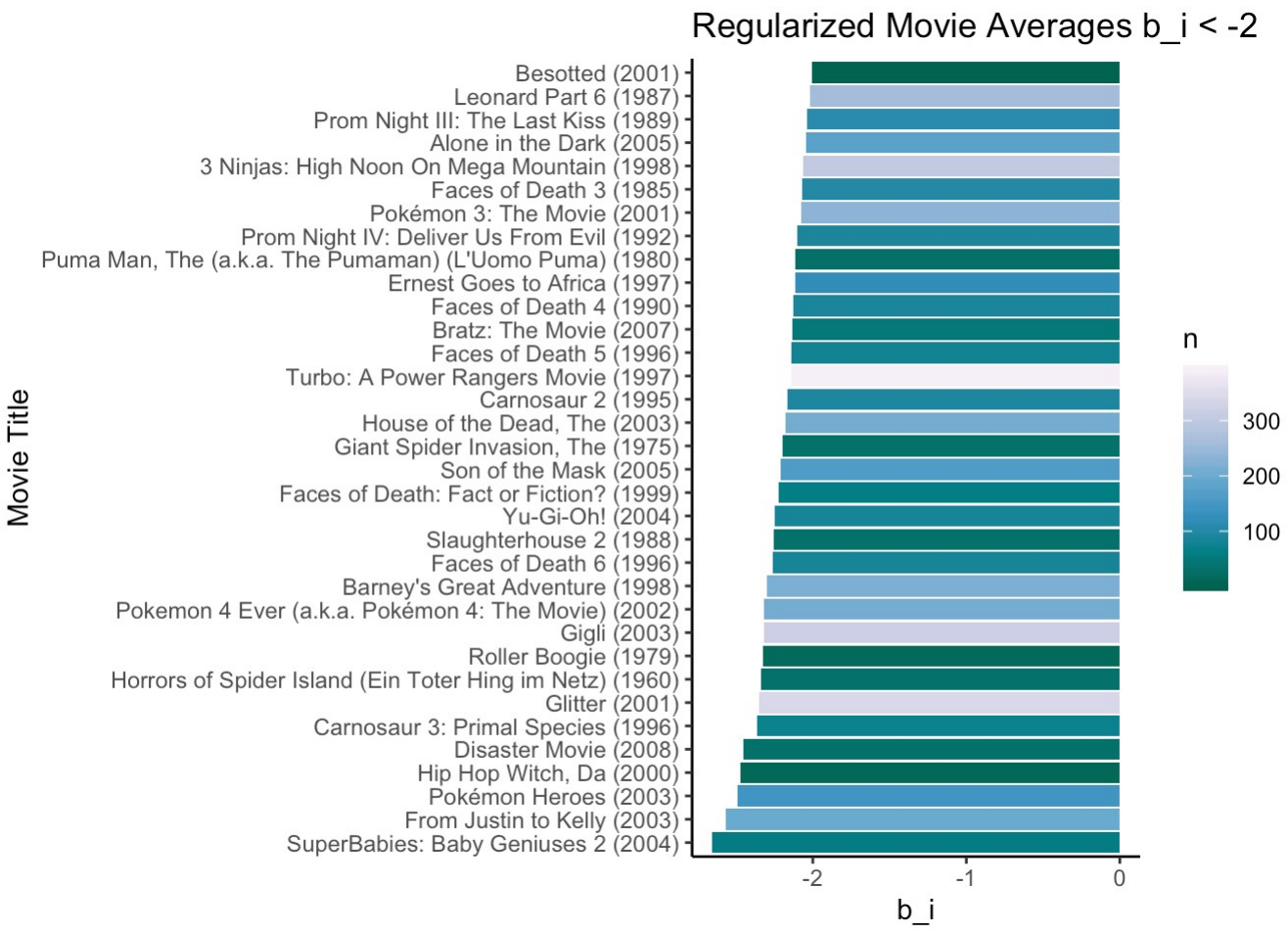
```
head(edx_with_avgs)
```

```
## # A tibble: 6 x 5
## # Groups:   title [6]
##   title                                                movieId     n   b_i   n_i
##   <chr>                                                  <dbl> <int> <dbl> <int>
## 1 More (1998)                                             4454     7 1.05      7
## 2 Satan's Tango (Sátántangó) (1994)                      33264     2 0.992     2
## 3 Human Condition II, The (Ningen no joken II) (1959)    26048     3 0.991     3
## 4 Human Condition III, The (Ningen no joken III) (196…   26073     4 0.990     4
## 5 Who's Singin' Over There? (a.k.a. Who Sings Over Th…    5194     4 0.990     4
## 6 Shawshank Redemption, The (1994)                         318 27988 0.944 27988
```

```
p <- edx_with_avgs %>% arrange(b_i) %>% filter(b_i < -2) %>% arrange((b_i))
p
```

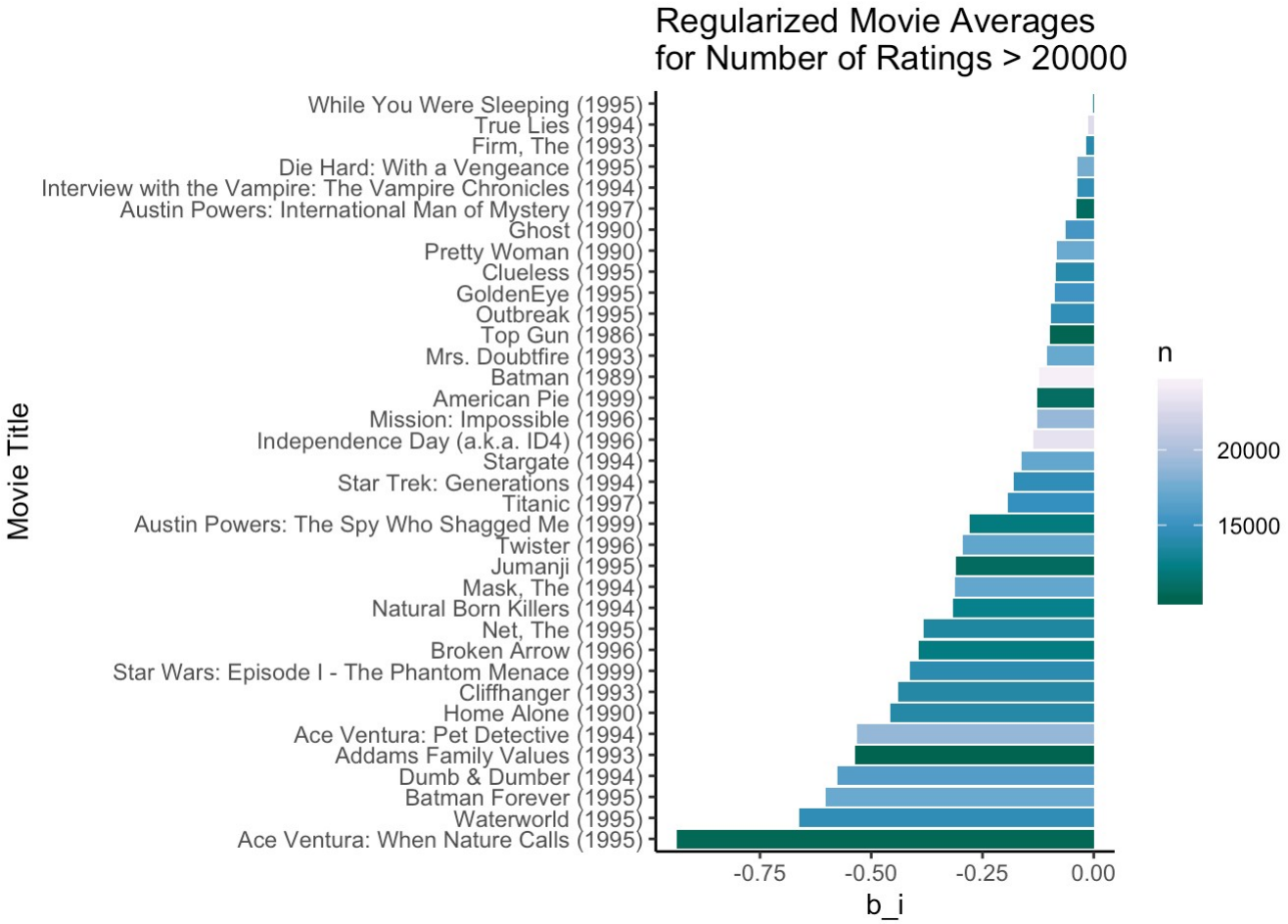
```
## # A tibble: 34 x 5
## # Groups:   title [34]
##    title                                                movieId     n   b_i   n_i
##    <chr>                                                  <dbl> <int> <dbl> <int>
##  1 SuperBabies: Baby Geniuses 2 (2004)                     8859    59 -2.65    59
##  2 From Justin to Kelly (2003)                             6483   198 -2.57   198
##  3 Pokémon Heroes (2003)                                   6371   145 -2.49   145
##  4 Hip Hop Witch, Da (2000)                                7282    12 -2.47    12
##  5 Disaster Movie (2008)                                  61348    31 -2.45    31
##  6 Carnosaur 3: Primal Species (1996)                      3574    73 -2.36    73
##  7 Glitter (2001)                                          4775   340 -2.35   340
##  8 Horrors of Spider Island (Ein Toter Hing im Netz) …     4051    31 -2.34    31
##  9 Roller Boogie (1979)                                    8856    15 -2.32    15
## 10 Gigli (2003)                                            6587   319 -2.32   319
## # … with 24 more rows
```

```
p %>%
  ggplot(aes(reorder(title, b_i), b_i, fill = n)) +
  geom_bar(stat = "identity") + coord_flip() + scale_fill_distiller(palette = "PuBuG
n") +
  ggtitle("") + xlab("Movie Title") + ggtitle('Regularized Movie Averages b_i < -2')
+
  theme_classic()
```



#Movies with number of ratings larger than 1000 and regularized average less than 0.

```
edx_with_avgs %>% filter(n > 10000, b_i < 0.0) %>%
  ggplot(aes(reorder(title, b_i), b_i, fill = n)) +
  geom_bar(stat = "identity") + coord_flip() + scale_fill_distiller(palette = "PuBuG
n") +
  ggtitle("") + xlab("Movie Title") + ggtitle('Regularized Movie Averages\nfor Number
of Ratings > 20000') +
  theme_classic()
```



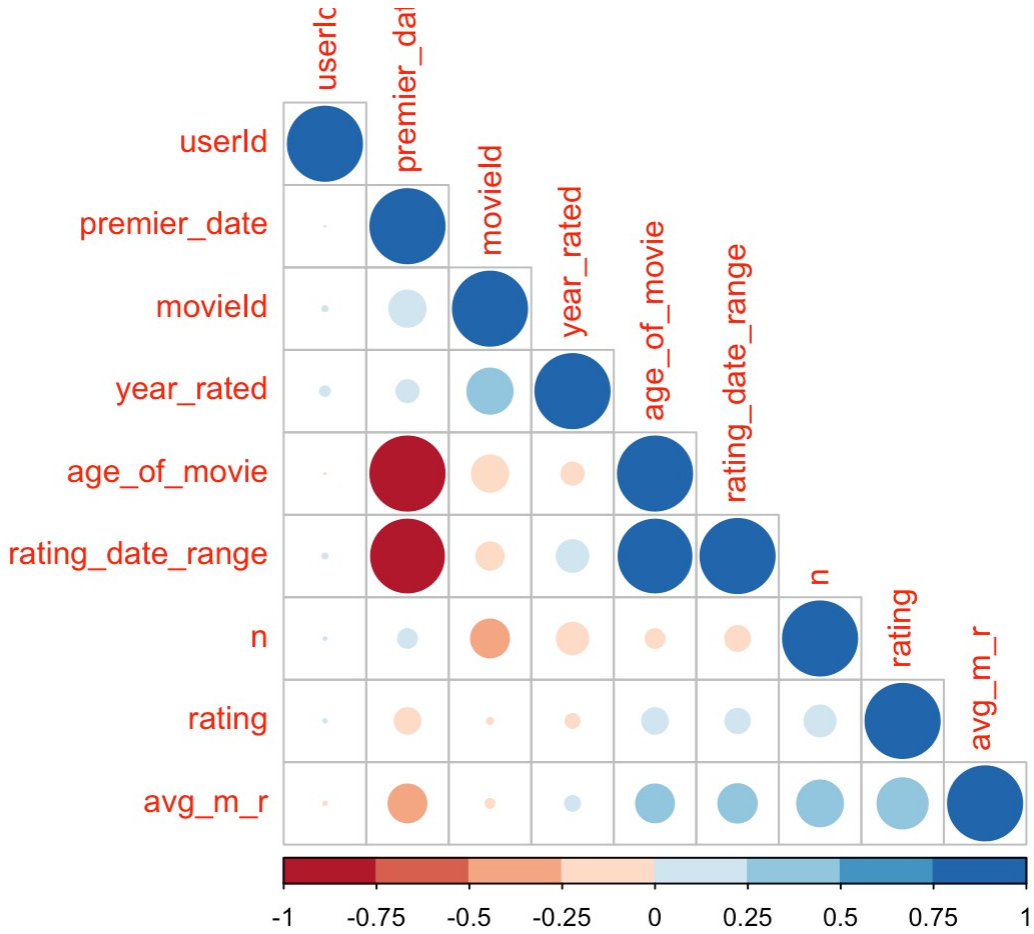
Explore correlation between ratings, users, movieId age of movie and number of ratings

```
#Is there a correlation
#Number of movie ratings per movie
n_movies_ratings <- edx_with_title_dates %>% group_by(movieId) %>% summarize(n = n())
#Average Movie Rating for each movie
avg_movie_rat <- edx_with_title_dates %>% group_by(movieId) %>% summarize(avg_m_r = m
ean(rating))
#Create correlation data
cor_dat <- edx_with_title_dates %>% select(rating, movieId, userId, year_rated, age_o
f_movie, rating_date_range, premier_date) %>%
  left_join(n_movies_ratings, by = "movieId") %>%
  left_join(avg_movie_rat, by = 'movieId')
head(cor_dat)
```

```
##   rating movieId userId year_rated age_of_movie rating_date_range premier_date
## 1      5     122      1       1996           26                 4         1992
## 2      5     185      1       1996           23                 1         1995
## 3      5     231      1       1996           24                 2         1994
## 4      5     292      1       1996           23                 1         1995
## 5      5     316      1       1996           24                 2         1994
## 6      5     329      1       1996           24                 2         1994
##       n  avg_m_r
## 1  2157 2.863236
## 2 13467 3.129984
## 3 16122 2.936825
## 4 14481 3.416580
## 5 17024 3.350417
## 6 14517 3.333678
```

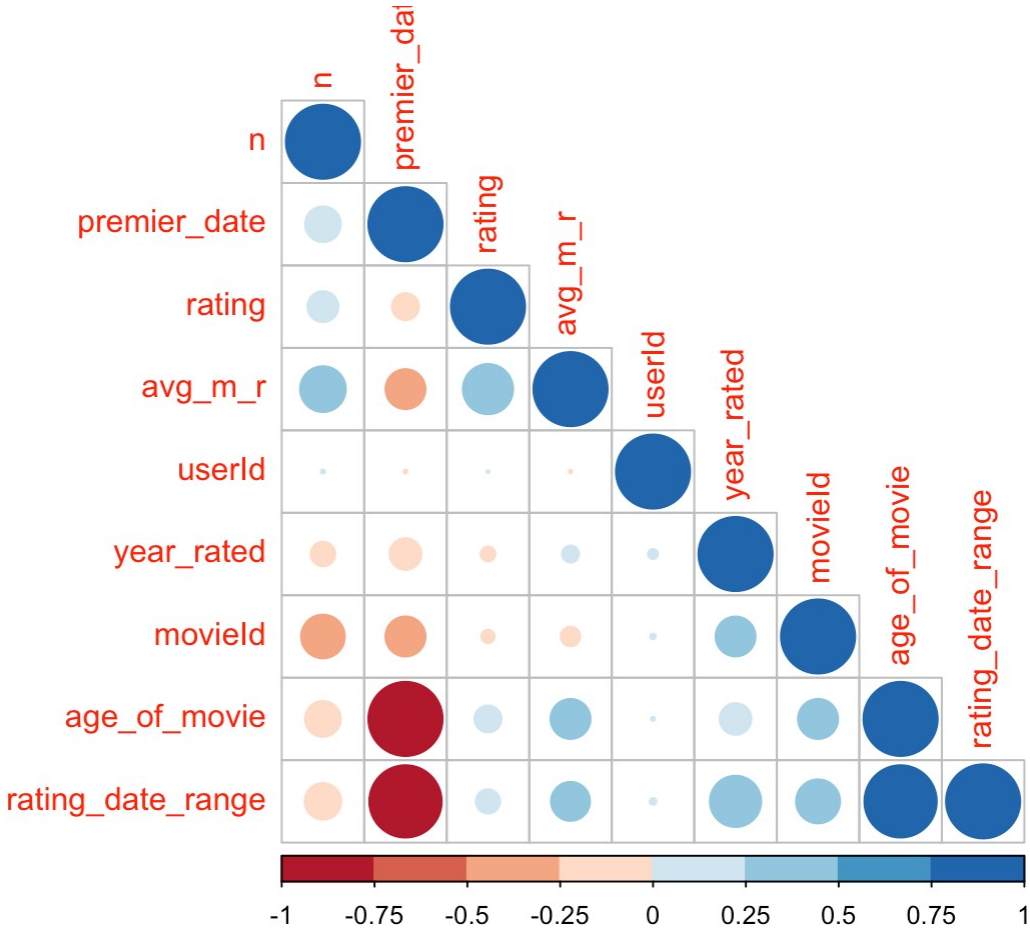
Graph the correlation

```
temp <- cor_dat %>% select(one_of("rating", "movieId", "userId", "year_rated", "age_o
f_movie",
                                  "rating_date_range", "premier_date", "n", "avg_m_
r")) %>% as.matrix()
M <- cor(temp, use = "pairwise.complete.obs")
corrplot(M, order = "hclust", addrect = 2, type = "lower", col = brewer.pal(n = 8, na
me = "RdBu"))
```

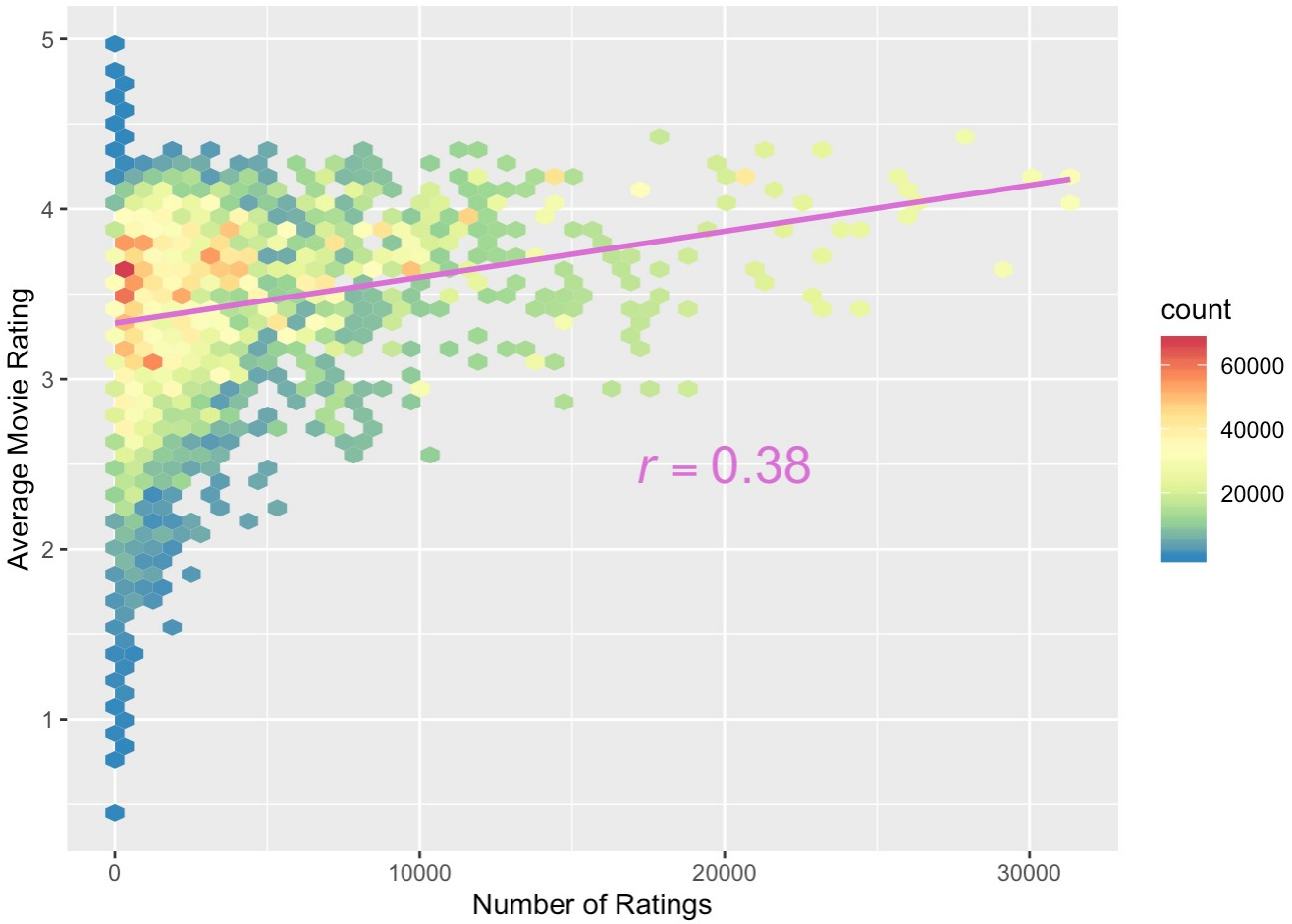


#What is the effect of the age of the movie

```
corr_by_age_of_movie <- cor_dat %>% filter((age_of_movie >20) & (age_of_movie < 70))
temp <- corr_by_age_of_movie %>% select(one_of("rating", "movieId", "userId", "year_r
ated", "age_of_movie",
                                               "rating_date_range", "n", "premier_dat
e", "avg_m_r")) %>% as.matrix()
M <- cor(temp, use = "pairwise.complete.obs")
corrplot(M, order = "hclust", addrect = 2, type = "lower", col = brewer.pal(n = 8, na
me = "RdBu"))
```

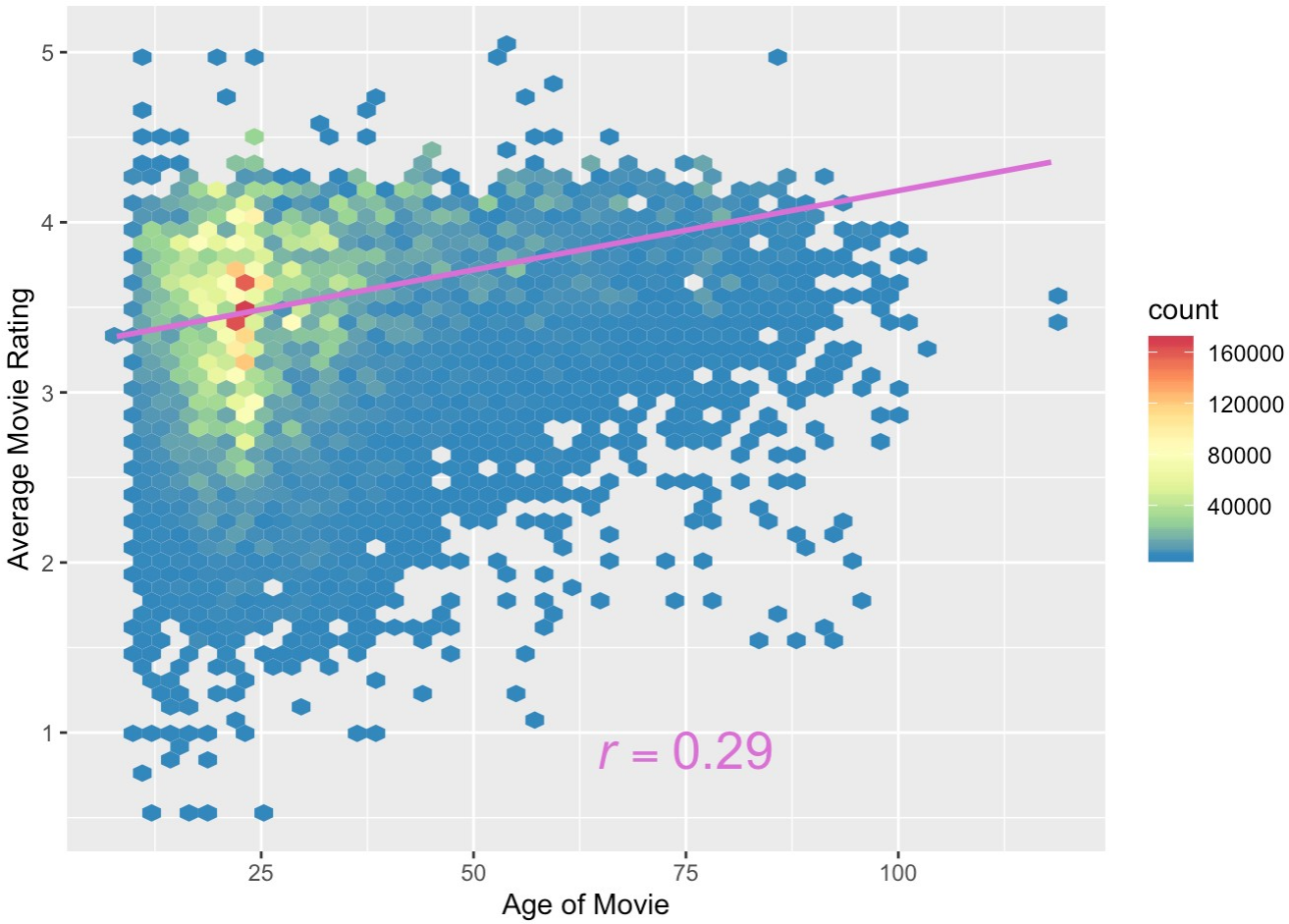



```
#Is there a relationship between number of ratings and the average rating
get_cor <- function(df){
  m <- cor(df$x, df$y, use="pairwise.complete.obs");
  eq <- substitute(italic(r) == cor, list(cor = format(m, digits = 2)))
  as.character(as.expression(eq));
}
#Number of ratings vs avg movie ratings
cor_dat %>%
  ggplot(aes(n, avg_m_r)) + stat_bin_hex(bins = 50) + scale_fill_distiller(palette =
"Spectral") +
  stat_smooth(method = "lm", color = "orchid", size = 1) +
  annotate("text", x = 20000, y = 2.5, label = get_cor(data.frame(x = cor_dat$n, y =
cor_dat$avg_m_r)),
           parse = TRUE, color = "orchid", size = 7) + ylab("Average Movie Rating") +
xlab("Number of Ratings")
```



#Is there an Age Effect on Movie Ratings?

```
cor_dat %>%
  ggplot(aes(age_of_movie, avg_m_r)) + stat_bin_hex(bins = 50) + scale_fill_distiller
(palette = "Spectral") +
  stat_smooth(method = "lm", color = "orchid", size = 1) +
  annotate("text", x = 75, y = 0.9, label = get_cor(data.frame(x = corr_by_age_of_mov
ie$age_of_movie, y = corr_by_age_of_movie$avg_m_r)),
           parse = TRUE, color = "orchid", size = 7) + ylab("Average Movie Rating") +
xlab('Age of Movie')
```



Calculate the RMSE

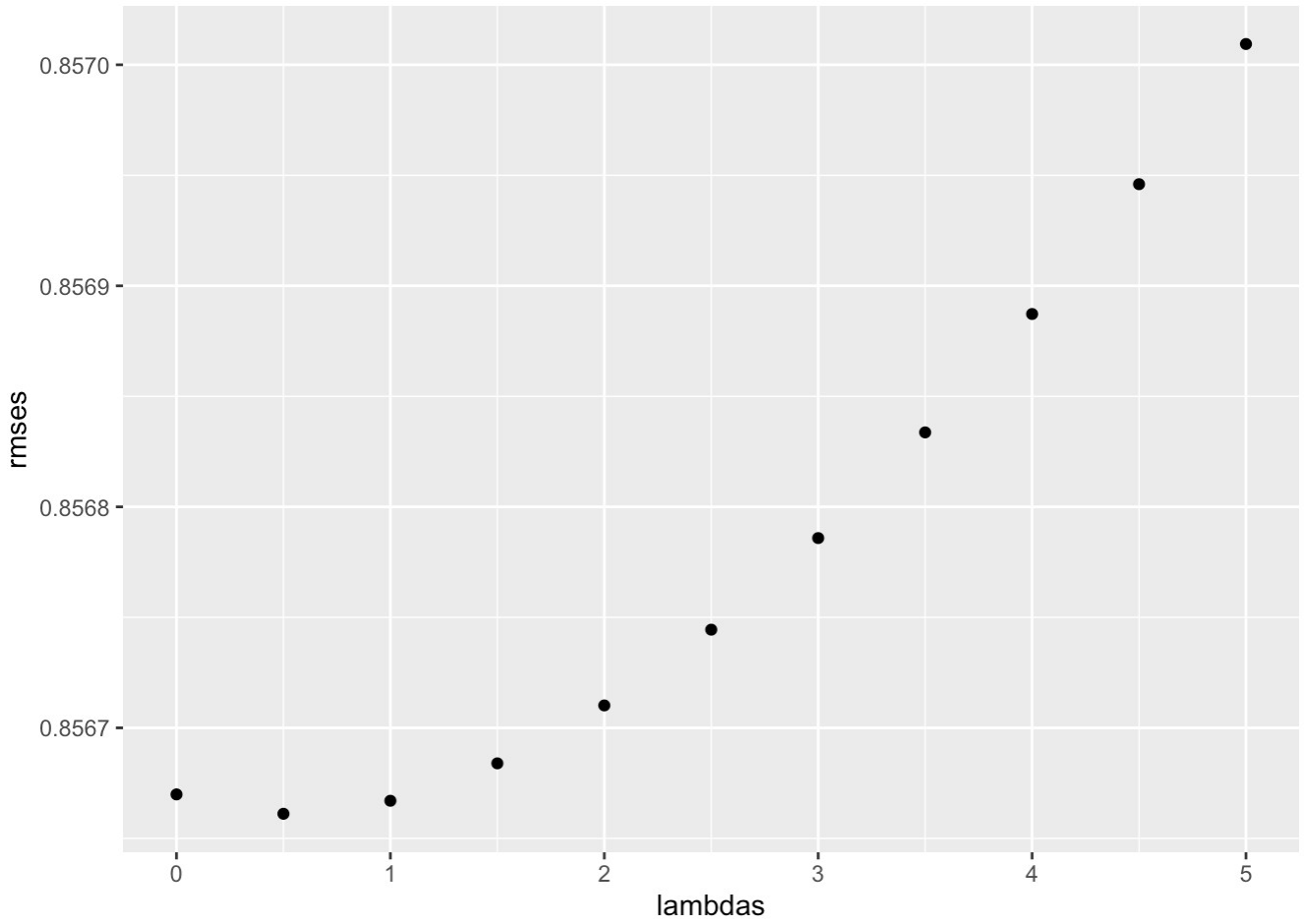
```
#RMSE function
RMSE <- function(true_ratings, predicted_ratings){
  sqrt(mean((true_ratings - predicted_ratings)^2))
}
#Choose the tuning value
lambdas <- seq(0,5,.5)
rmses <- sapply(lambdas, function(l){
  mu <- mean(edx_with_title_dates$rating)

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

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

  predicted_ratings <- edx_with_title_dates %>%
    left_join(b_i, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    mutate(pred = mu + b_i +  b_u) %>% .$pred

  return(RMSE(predicted_ratings, edx_with_title_dates$rating))
})
qplot(lambdas, rmses)
```




```
lambdas[which.min(rmses)]
```

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

Using the model on the Validation data

```
mu <- mean(validation$rating)
l <- 0.15
b_i <- validation %>%
  group_by(movieId) %>%
  summarize(b_i = sum(rating - mu)/(n() + l))

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

predicted_ratings <- validation %>%
  left_join(b_i, by = "movieId") %>%
  left_join(b_u, by = "userId") %>%
  mutate(pred = mu + b_i +  b_u) %>% .$pred
RMSE(predicted_ratings, validation$rating)
```

```
## [1] 0.8253432
```

Notes:

As part of my preliminary exploratory work I did work on calculating a b_a for the age of a movie.

However I found that there was no decrease or a lower RMSE value so I removed these code from the current script.

By changing to use movieId and userId to calculate the RMSE, the script was able to achieve an RMSE = 0.826

Alternative checking option: Using the R package “Metrics” and the code below it is possible to verify the RMSE values, which

resulted in close/same value of RMSE.

```
library(Metrics)
```

```
##
## Attaching package: 'Metrics'
```

```
## The following objects are masked from 'package:modelr':
##
##     mae, mape, mse, rmse
```

```
## The following objects are masked from 'package:caret':
##
##     precision, recall
```

```
rmse(validation$rating, predicted_ratings)
```

```
## [1] 0.8253432
```

Apologies for not optimizing the code. Due to my relative inexperience in using R I have tried to follow all variables to detail - January 2020 - SEC