MOVIELENS RATING PREDICTION

BY

M.HANUMAN SAI 19BPS1066

VIJAY. GOPU 19BPS1078

A project report submitted to

Dr. Tulasi Prasad

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

in partial fulfilment of the requirements for the course of

CSE3505 - Fundamental of Data Analytics

IN

B. Tech. COMPUTER SCIENCE ENGINEERING

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Vandalur – Kelambakkam Road

Chennai – 600127

NOVEMBER 2021

ABSTRACT

To produce precise suggestions, recommendation systems employ ratings that users have provided to things. Companies that sell a large number of items to a large number of consumers and allow those customers to evaluate their products, such as Amazon, can accumulate vast datasets that can be used to forecast what rating a certain user will give to a specific item. Items with a high expected rating for a certain user are then recommended to that person. The same might be said of other products, such as movies in our situation. One of the most often used models in machine learning algorithms is recommendation systems. Netflix's success is claimed to be due to its powerful recommendation algorithm. In reality, the Netflix reward (an open competition for the best collaborative filtering algorithm to forecast user ratings for films based on prior ratings without any other information about the users or films) is representative of this algorithm for products recommendation system.

INTRODUCTION

In this project our plan is to develop a model that can predict how a user will rate a specific movie, similar to a movie recommendation system. Our model will make predictions based on user ratings of other movies and the average rating of the specific movie.

METHODOLOGY

After downloading and unzipping the data file, I will extract its contents and place the data in a data frame.

Next, I will split the data into two sets: a training set and a validation set. The training set will contain 90% of the data (9 million ratings) which the model will learn from. The validation set will contain the remaining 10% of data (1 million ratings) which will be used to evaluate the performance of our model

First, we will run an exploratory data analysis to get a general overview of the data and explore possible predictor variables.

Then, we will include the most important features into numerous models i..e We are planning to test three different regression models to predict each rating .Then, we will select the best model and apply it to the test data set (validation)

Finally, We will deploy the model with the smallest error to the test set and evaluate the results.

DATASETS USED

For this project we using "MovieLens dataset"

- [MovieLens 10M dataset]
https://grouplens.org/datasets/movielens/10m/

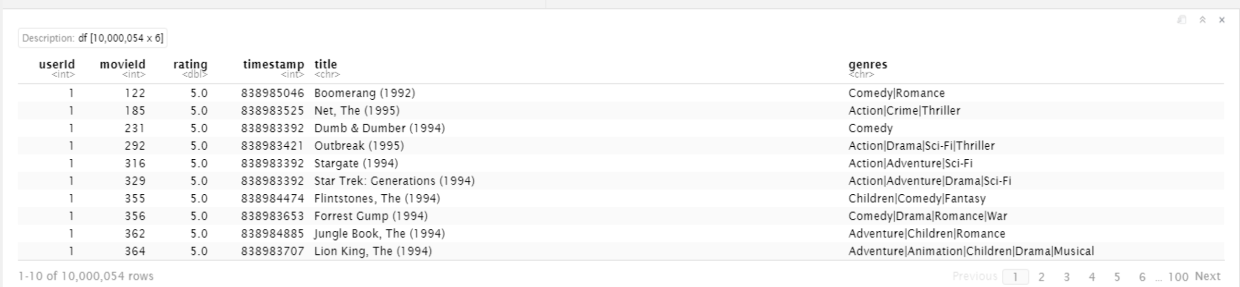
- [MovieLens 10M dataset - zip file]

https://files.grouplens.org/datasets/movielens/ml-10m.zip

DATA DESCRIPTION

For this project, we used the 10M version of the MovieLens dataset which we collected from “https://grouplens.org/datasets/movielens/10m/”.

The dataset presents information about 10 million movie ratings including user id, movie id, user rating of the movie (between 0.5 to 5 stars), timestamp of the rating (seconds since midnight Coordinated Universal Time of January 1, 1970), title of the movie, and movie genre(s): Action, Adventure, Animation, Children’s, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, IMAX, Musical, Mystery, Romance, Sci-Fi, Thriller, War, and/or Western



Description: df [10,000,054 x 6]

userId <int>	movieId <int>	rating <dbl>	timestamp <int>	title <chr>	genres <chr>
1	122	5.0	838985046	Boomerang (1992)	Comedy|Romance
1	185	5.0	838983525	Net, The (1995)	Action|Crime|Thriller
1	231	5.0	838983392	Dumb & Dumber (1994)	Comedy
1	292	5.0	838983421	Outbreak (1995)	Action|Drama|Sci-Fi|Thriller
1	316	5.0	838983392	Stargate (1994)	Action|Adventure|Sci-Fi
1	329	5.0	838983392	Star Trek: Generations (1994)	Action|Adventure|Drama|Sci-Fi
1	355	5.0	838984474	Flintstones, The (1994)	Children|Comedy|Fantasy
1	356	5.0	838983653	Forrest Gump (1994)	Comedy|Drama|Romance|War
1	362	5.0	838984885	Jungle Book, The (1994)	Adventure|Children|Romance
1	364	5.0	838983707	Lion King, The (1994)	Adventure|Animation|Children|Drama|Musical

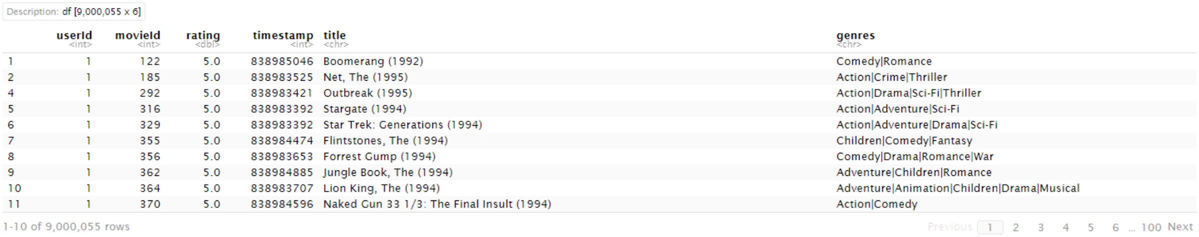
1-10 of 10,000,054 rows

Previous 1 2 3 4 5 6 ... 100 Next

It contains 10M rows and 6 coloumns. Since data is huge we are dividing into training and validation sets earlier than later stages.

SPLIT RAW DATA: TRAIN AND VALIDATION

Train dataset contains 90% of original data:



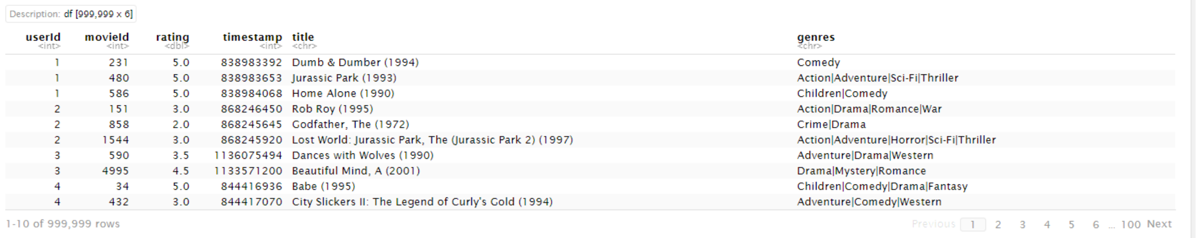
Description: df [9,000,055 x 6]

	userId <int>	movieId <int>	rating <dbl>	timestamp <int>	title <chr>	genres <chr>
1	1	122	5.0	838985046	Boomerang (1992)	Comedy|Romance
2	1	185	5.0	838983525	Net, The (1995)	Action|Crime|Thriller
4	1	292	5.0	838983421	Outbreak (1995)	Action|Drama|Sci-Fi|Thriller
5	1	316	5.0	838983392	Stargate (1994)	Action|Adventure|Sci-Fi
6	1	329	5.0	838983392	Star Trek: Generations (1994)	Action|Adventure|Drama|Sci-Fi
7	1	355	5.0	838984474	Flintstones, The (1994)	Children|Comedy|Fantasy
8	1	356	5.0	838983653	Forrest Gump (1994)	Comedy|Drama|Romance|War
9	1	362	5.0	838984885	Jungle Book, The (1994)	Adventure|Children|Romance
10	1	364	5.0	838983707	Lion King, The (1994)	Adventure|Animation|Children|Drama|Musical
11	1	370	5.0	838984596	Naked Gun 33 1/3: The Final Insult (1994)	Action|Comedy

1-10 of 9,000,055 rows

Previous 1 2 3 4 5 6 ... 100 Next

Validation set contains 10% :



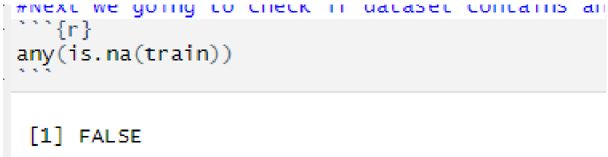
Description: df [999,999 x 6]

userId <int>	movieId <int>	rating <dbl>	timestamp <int>	title <chr>	genres <chr>
1	231	5.0	838983392	Dumb & Dumber (1994)	Comedy
1	480	5.0	838983653	Jurassic Park (1993)	Action|Adventure|Sci-Fi|Thriller
1	586	5.0	838984068	Home Alone (1990)	Children|Comedy
2	151	3.0	868246450	Rob Roy (1995)	Action|Drama|Romance|War
2	858	2.0	868245645	Godfather, The (1972)	Crime|Drama
2	1544	3.0	868245920	Lost World: Jurassic Park, The (Jurassic Park 2) (1997)	Action|Adventure|Horror|Sci-Fi|Thriller
3	590	3.5	1136075494	Dances with Wolves (1990)	Adventure|Drama|Western
3	4995	4.5	1133571200	Beautiful Mind, A (2001)	Drama|Mystery|Romance
4	34	5.0	844416936	Babe (1995)	Children|Comedy|Drama|Fantasy
4	432	3.0	844417070	City Slickers II: The Legend of Curly's Gold (1994)	Adventure|Comedy|Western

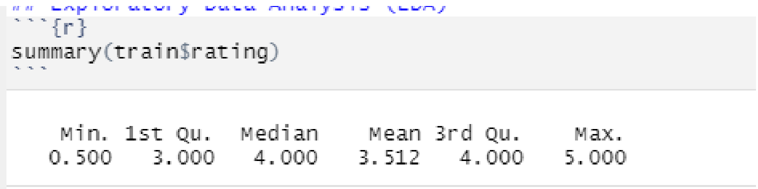
1-10 of 999,999 rows

Check for NA values

Prior to the exploratory data analysis, we will check if the dataset contains any missing values.

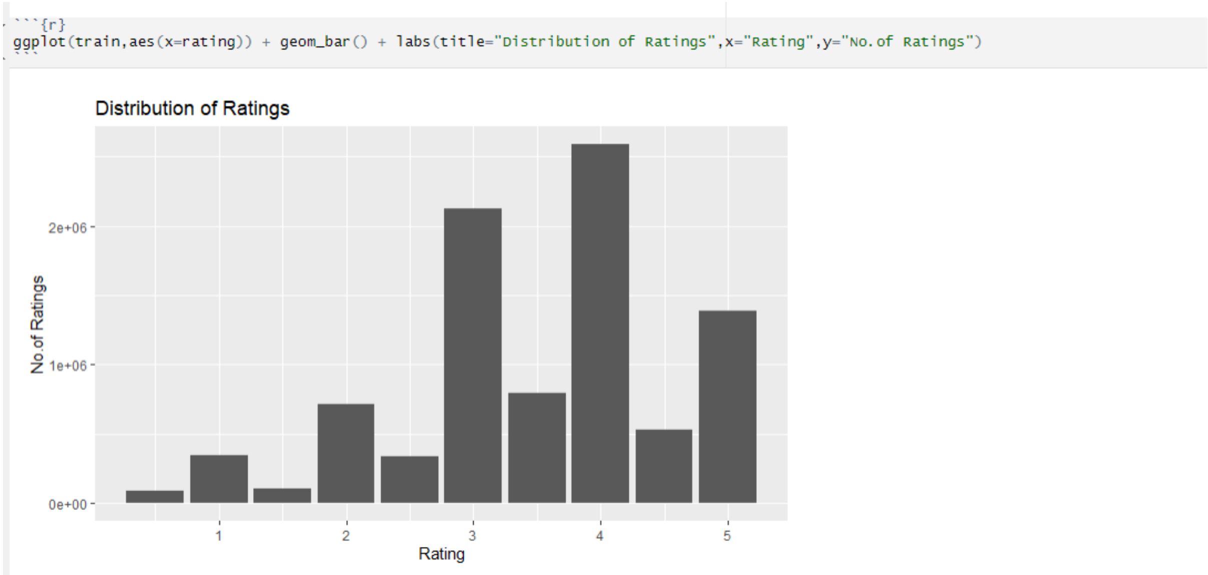


~~~
#Next we going to check if dataset contains an
```{r}
any(is.na(train))
~~~

 [1] FALSE
```

There are no missing values in the dataset.

Exploratory Data Analysis (EDA)



```
## Exploratory Data Analysis (EDA)
```{r}
summary(train$rating)
```

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

so ,the average rating across the 9 million ratings in the training set is 3.51 stars
```

# Distribution of ratings across the dataset:

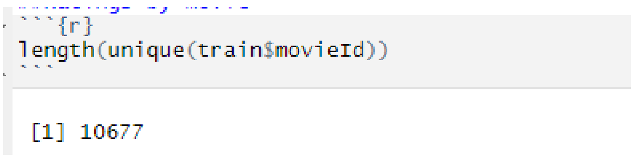


~~~
```{r}
ggplot(train,aes(x=rating)) + geom_bar() + labs(title="Distribution of Ratings",x="Rating",y="No.of Ratings")
```
~~~

As you can see, the ratings appear to be left-skewed since there are few ratings between 0 to 2 stars and many ratings between 3 to 5 stars. It is important to note that users only had the option to select whole number or half ratings. Thus, there were only ten options users could select from (0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5). In general, half star ratings appear to be less common than whole star ratings.

Now, I will perform a deeper analysis of ratings by movie. Specifically, we will compare the average rating of movies and determine whether the number of ratings a movie receives impacts its average rating.

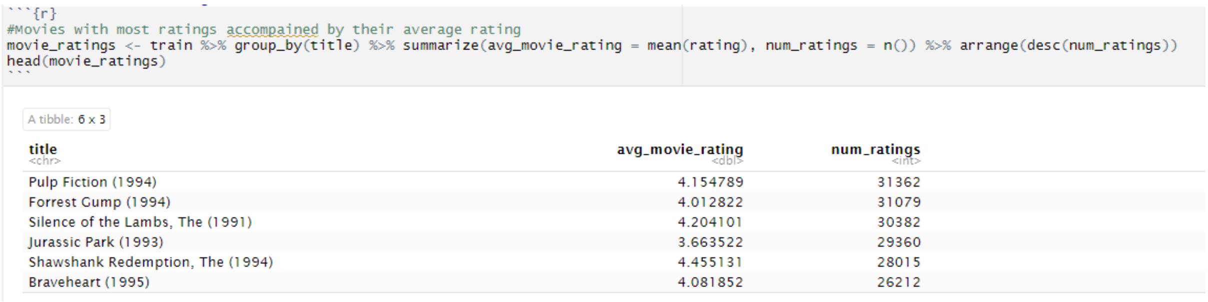
Ratings by movie:



~~~
```{r}
length(unique(train$movieId))
```

 [1] 10677
~~~

There are 10,677 movies in the training dataset.

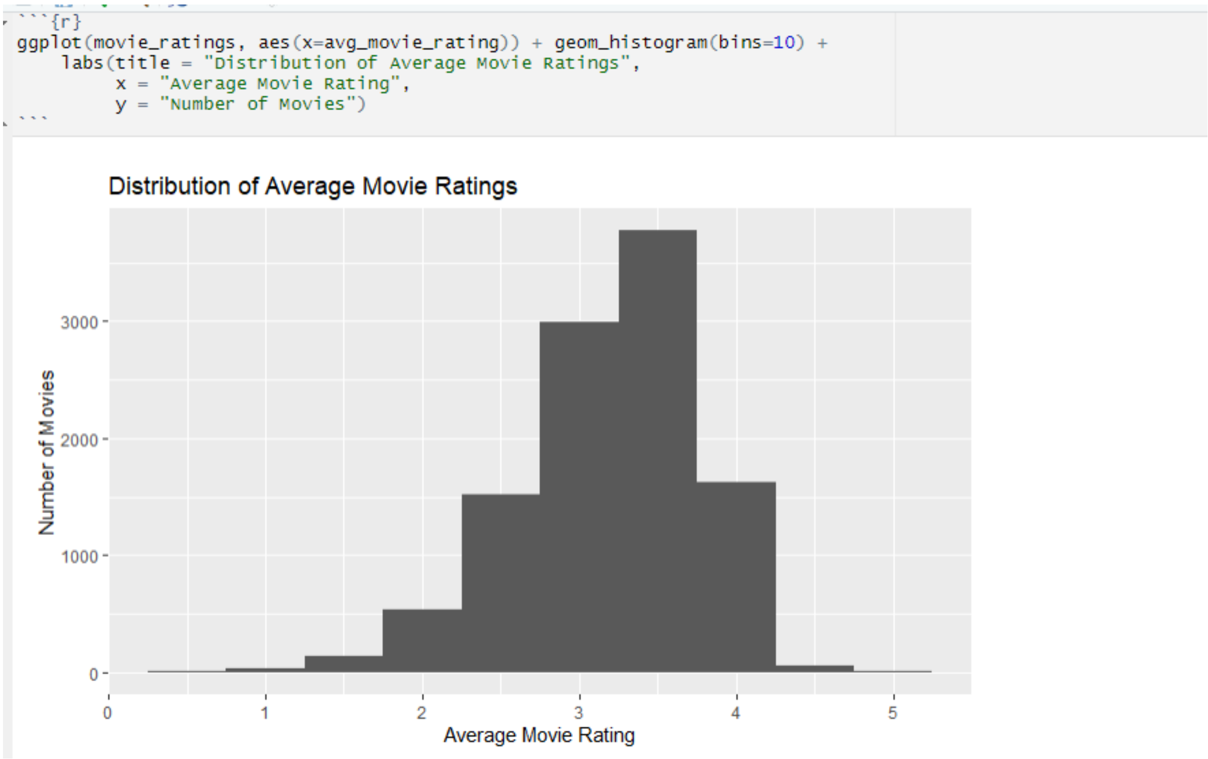
Movies with the Most Ratings:



~~~
```{r}
#Movies with most ratings accompained by their average rating
movie_ratings <- train %>% group_by(title) %>% summarize(avg_movie_rating = mean(rating), num_ratings = n()) %>% arrange(desc(num_ratings))
head(movie_ratings)
```

A tibble: 6 x 3
title                                   avg_movie_rating   num_ratings
<chr>                                              <dbl>         <int>
Pulp Fiction (1994)                             4.154789         31362
Forrest Gump (1994)                             4.012822         31079
Silence of the Lambs, The (1991)                4.204101         30382
Jurassic Park (1993)                            3.663522         29360
Shawshank Redemption, The (1994)                4.455131         28015
Braveheart (1995)                               4.081852         26212
~~~

Pulp Fiction, Forrest Gump, Silence of the Lambs, Jurassic Park, Shawshank Redemption, and Braveheart have the most ratings (about 30,000 each) with an average rating ranging between 3.66 and 4.46.

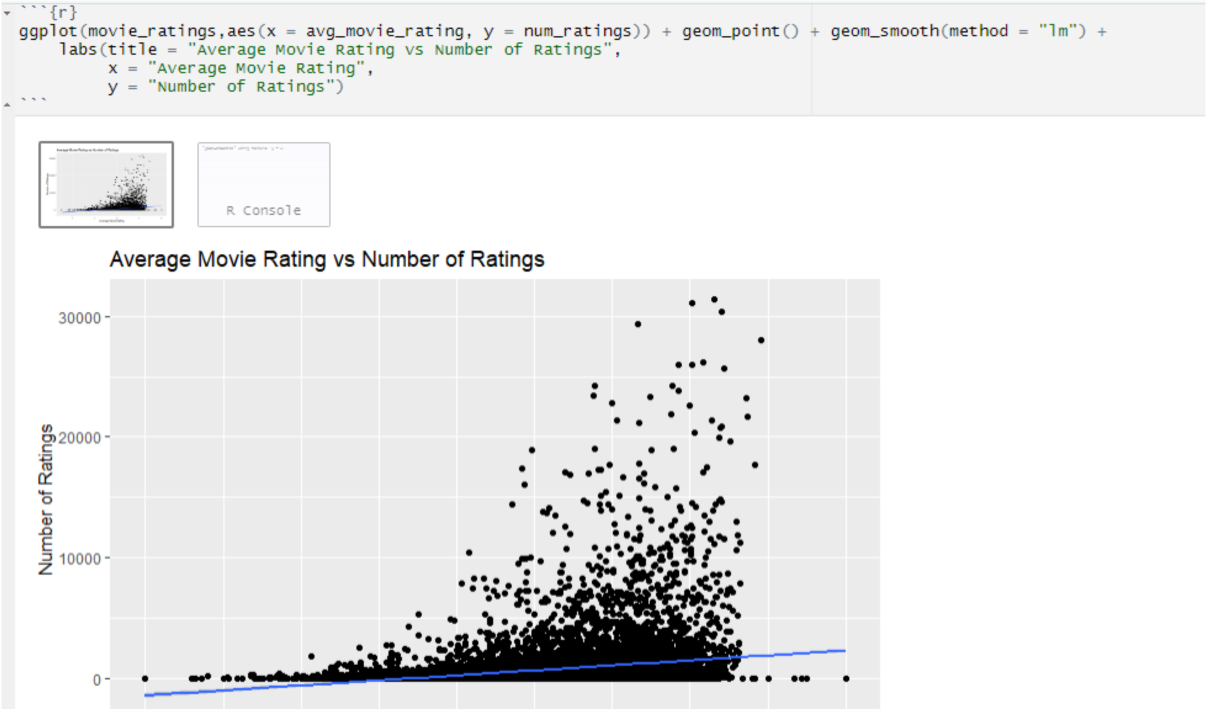
Most common average movie rating:



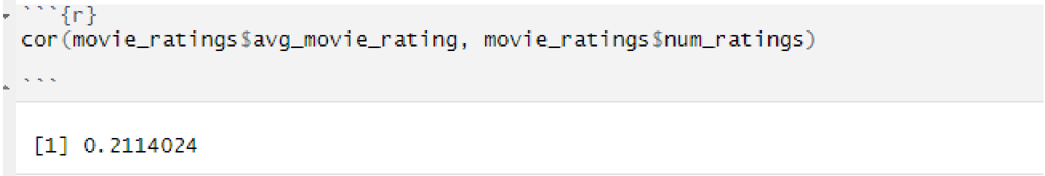
~~~
```{r}
ggplot(movie_ratings, aes(x=avg_movie_rating)) + geom_histogram(bins=10) +
    labs(title = "Distribution of Average Movie Ratings",
         x = "Average Movie Rating",
         y = "Number of Movies")
```
~~~

Based on the histogram above, most movies appear to have an average rating between 2.5 and 4. In addition, there are only be a few movies with an average rating of 0.5 stars (worst possible rating) and 5 stars (perfect rating).

Do movies with many ratings tend to be rated higher than movies with few ratings?



~~~
```{r}
ggplot(movie_ratings,aes(x = avg_movie_rating, y = num_ratings)) + geom_point() + geom_smooth(method = "lm") +
    labs(title = "Average Movie Rating vs Number of Ratings",
         x = "Average Movie Rating",
         y = "Number of Ratings")
```
~~~

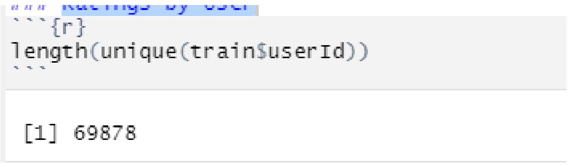


~~~
```{r}
cor(movie_ratings$avg_movie_rating, movie_ratings$num_ratings)
```

[1] 0.2114024
~~~

In general, the more a movie is rated by users, the greater its average rating. However, this relationship is relatively weak.

Now, we will perform a deeper analysis of ratings by user. Similar to my analysis of ratings by movie, I will compare the average rating of users and determine whether the number of ratings they have given in total impact their average rating.

Ratings by User

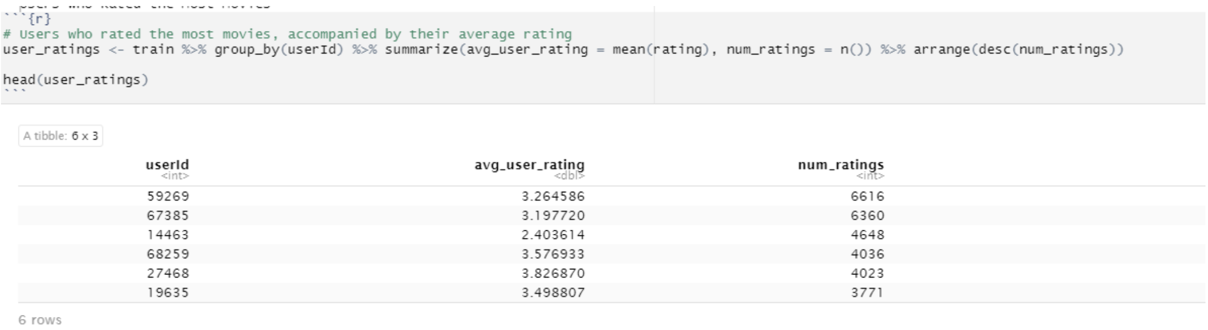


~~~
```{r}
length(unique(train$userId))
```
~~~

```
[1] 69878
```

There are 69,878 users in the training dataset.

Users Who Rated the Most Movies:



~~~
```{r}
# Users who rated the most movies, accompanied by their average rating
user_ratings <- train %>% group_by(userId) %>% summarize(avg_user_rating = mean(rating), num_ratings = n()) %>% arrange(desc(num_ratings))

head(user_ratings)
```
~~~

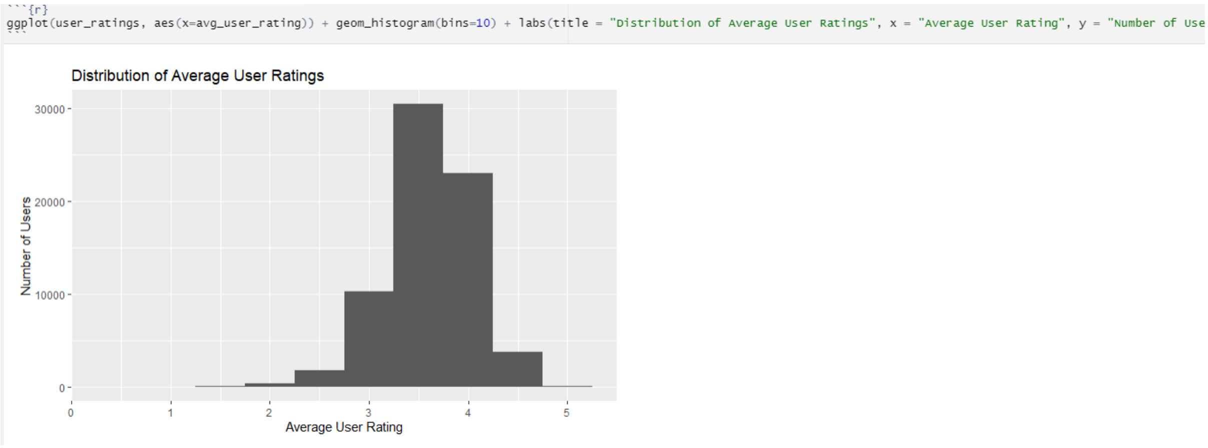
A tibble: 6 x 3

userId <int>	avg_user_rating <dbl>	num_ratings <int>
59269	3.264586	6616
67385	3.197720	6360
14463	2.403614	4648
68259	3.576933	4036
27468	3.826870	4023
19635	3.498807	3771

6 rows

The user that rated the most movies rated a total of 6616 movies with an average rating of 3.26.

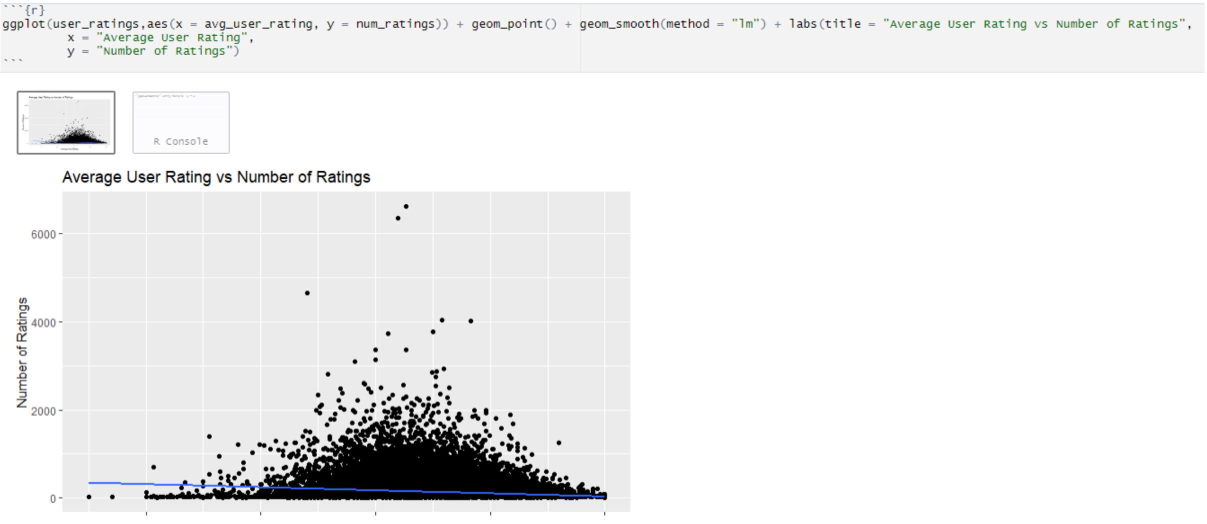
Most Common Average Rating Given by Users:



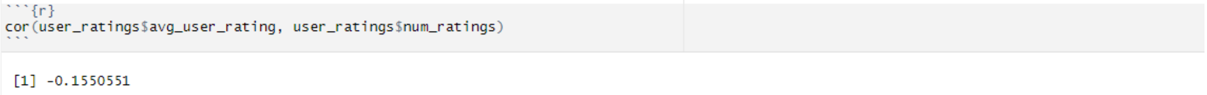
~~~
```{r}
ggplot(user_ratings, aes(x=avg_user_rating)) + geom_histogram(bins=10) + labs(title = "Distribution of Average User Ratings", x = "Average User Rating", y = "Number of Use
```
~~~

Based on the histogram above, most users give an average rating between 3 and 4.5. In addition, only a few users have a very high or very low average rating (i.e. 0 to 2 stars; 5 stars).

Do users who rate many movies tend to rate higher than users who rate few movies?



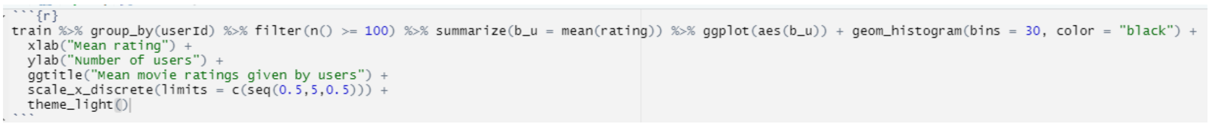
~~~
```{r}
ggplot(user_ratings,aes(x = avg_user_rating, y = num_ratings)) + geom_point() + geom_smooth(method = "lm") + labs(title = "Average User Rating vs Number of Ratings",
         x = "Average User Rating",
         y = "Number of Ratings")
```
~~~



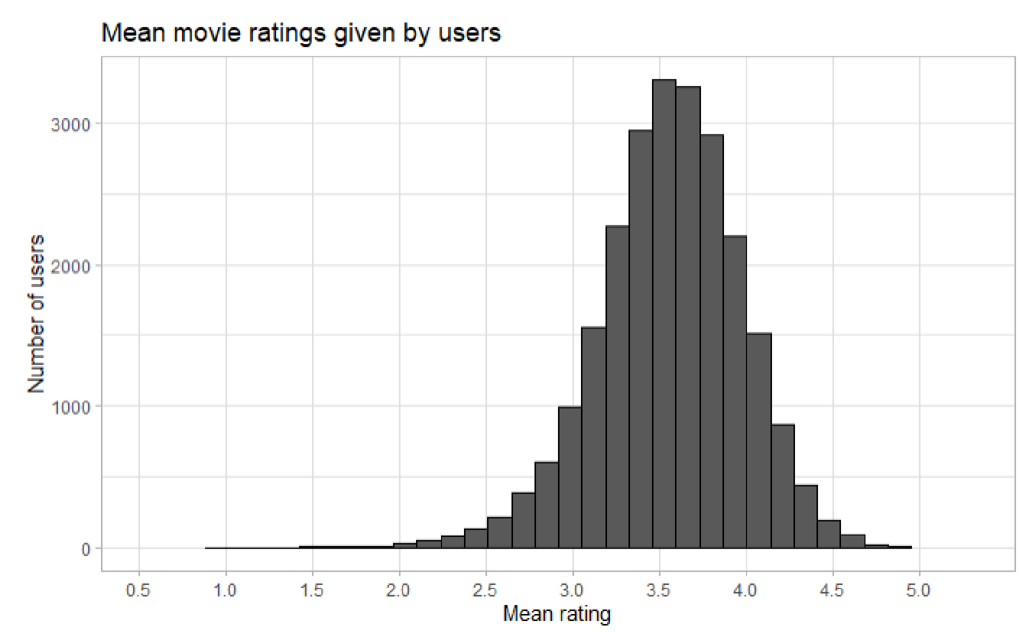
~~~
```{r}
cor(user_ratings$avg_user_rating, user_ratings$num_ratings)
```

[1] -0.1550551
~~~

In general, the more a user rates movies, the lower their average rating tends to be. However, this relationship is relatively weak.

Mean movie ratings given by users:



~~~
```{r}
train %>% group_by(userId) %>% filter(n() >= 100) %>% summarize(b_u = mean(rating)) %>% ggplot(aes(b_u)) + geom_histogram(bins = 30, color = "black") +
  xlab("Mean rating") +
  ylab("Number of users") +
  ggtitle("Mean movie ratings given by users") +
  scale_x_discrete(limits = c(seq(0.5,5,0.5))) +
  theme_light()
```
~~~



We can depict that 3.5 is the highest mean rating given by almost 3000+ users.

Key Insights from EDA

Based on our EDA, we would expect most ratings to be between 3 and 4 stars. In addition, both movie and user averages appear to have an impact on the actual rating, so we will include these features in model development. However, the number of ratings a movie receives and the number of movies a user rates has a small impact on the actual rating. Thus, we will not include either component in model development.

MODEL DEVELOPMENT

we will test three different regression models to predict each rating in the training set . Then, we will select the best model and apply it to the validation set .

Model 1: Predicted Rating = Global Average Rating + Movie Effect

Model 2: Predicted Rating = Global Average Rating + User Effect

Model 3: Predicted Rating = Global Average Rating + Movie Effect + User Effect

The global average rating is the average rating across all entries in the dataset.

The movie effect is the difference between the average rating for the specific movie and the global average rating.

Similarly, the user effect is the difference between the average rating for the specific user and the global average rating.

To evaluate the three models, we will use RMSE (Root Mean Square Error). Ultimately, we will select the model with the lowest RMSE.

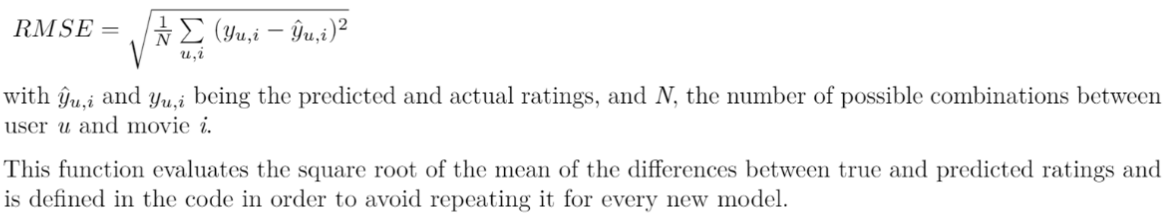
RMSE:

RMSE is used measure of the differences between values predicted by a model and the values observed. RMSE is a measure of accuracy, to compare forecasting errors of different models for a particular dataset, a lower RMSE is better than a higher one.

The effect of each error on RMSE is proportional to the size of the squared error; thus larger errors have a disproportionately large effect on RMSE.

Three models that will be developed will be compared using their resulting RMSE in order to assess their quality.

The function that computes the RMSE for vectors of ratings and their corresponding predictors will be the following



$$RMSE = \sqrt{\frac{1}{N}\sum_{u,i}(y_{u,i} - \hat{y}_{u,i})^2}$$

with $\hat{y}_{u,i}$ and $y_{u,i}$ being the predicted and actual ratings, and N, the number of possible combinations between user u and movie i.

This function evaluates the square root of the mean of the differences between true and predicted ratings and is defined in the code in order to avoid repeating it for every new model.

MODEL EXAMPLE

For example, suppose If am trying to predict User X's rating of Forest Gump and the overall average rating (across all movies/users) is 3 stars, the average rating of Forest Gump is 4 stars, and the average rating User X gives is 2.5 stars.

Thus

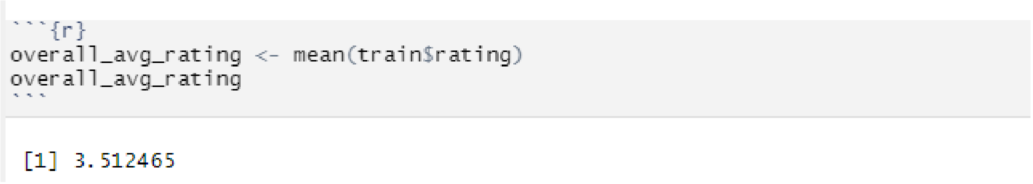
The global average rating is 3, the movie effect is +1 (4-3), and the user effect is -0.5 (2.5-3).

Model 1: Predicted Rating = 3 + 1 = 4

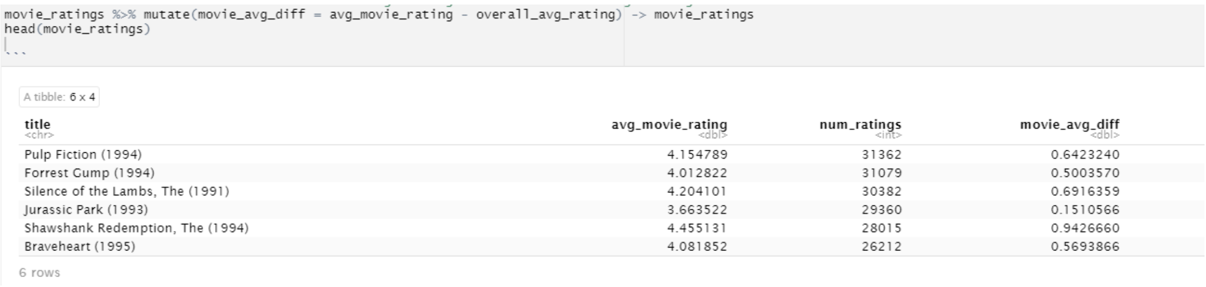
Model 2: Predicted Rating = 3 - 0.5 = 2.5

Model 3: Predicted Rating = 3 + 1 - 0.5 = 3.5

Calculating Global Average Rating:



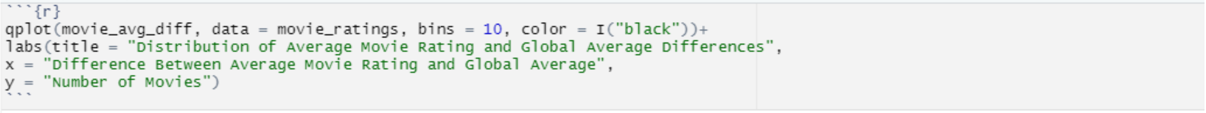
~~~
```{r}
overall_avg_rating <- mean(train$rating)
overall_avg_rating
~~~

[1] 3.512465
```

The global average rating is 3.512465.

# Model 1: Movie Effect

## Calculate difference between each movie's average rating and the overall average rating



```
movie_ratings %>% mutate(movie_avg_diff = avg_movie_rating - overall_avg_rating) -> movie_ratings
head(movie_ratings)
```
```

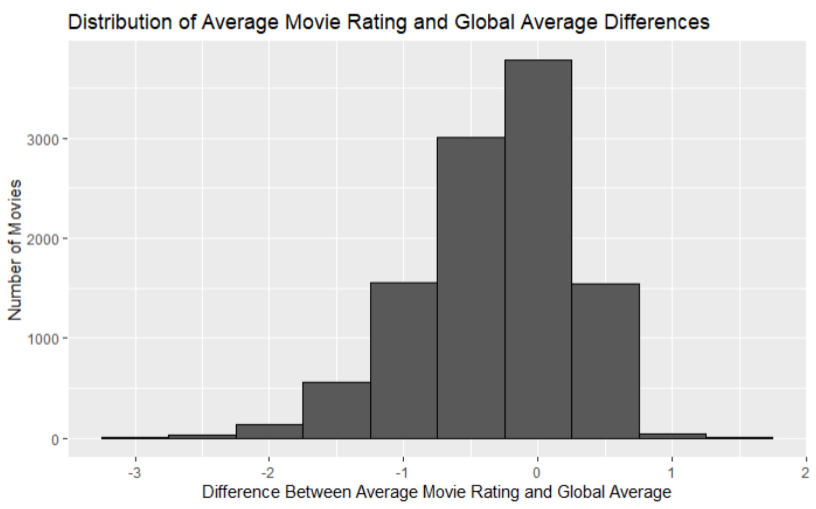
A tibble: 6 x 4

title <chr>	avg_movie_rating <dbl>	num_ratings <int>	movie_avg_diff <dbl>
Pulp Fiction (1994)	4.154789	31362	0.6423240
Forrest Gump (1994)	4.012822	31079	0.5003570
Silence of the Lambs, The (1991)	4.204101	30382	0.6916359
Jurassic Park (1993)	3.663522	29360	0.1510566
Shawshank Redemption, The (1994)	4.455131	28015	0.9426660
Braveheart (1995)	4.081852	26212	0.5693866

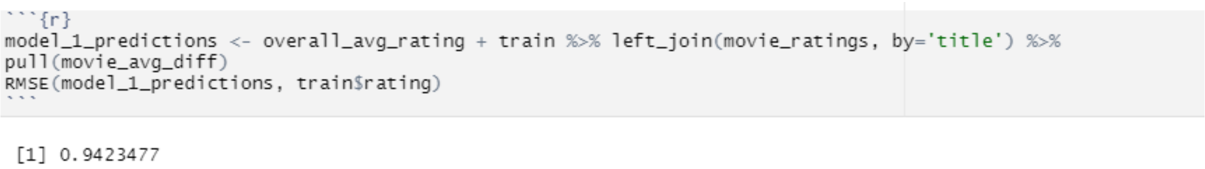
6 rows

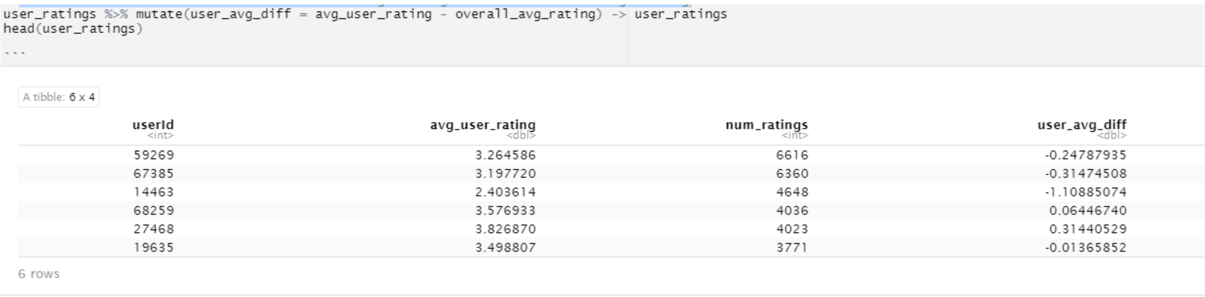


~~~
```{r}
qplot(movie_avg_diff, data = movie_ratings, bins = 10, color = I("black"))+
labs(title = "Distribution of Average Movie Rating and Global Average Differences",
x = "Difference Between Average Movie Rating and Global Average",
y = "Number of Movies")
```
~~~



As you can see, a lot of movies have an average rating close to the global average (difference of 0) and few movies have an average rating that is far from the global average (difference of +/- 2). In addition, the average rating for the specific movie is more likely to have a negative impact on each rating since there are more negative differences.



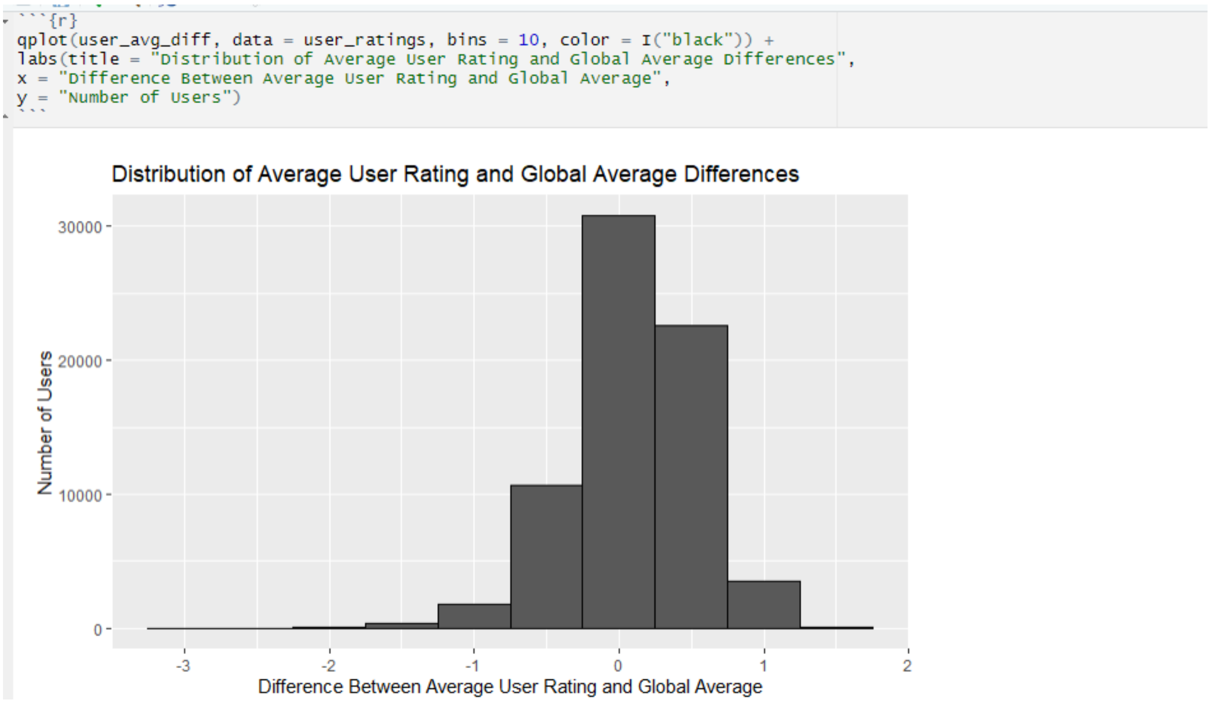
~~~
```{r}
model_1_predictions <- overall_avg_rating + train %>% left_join(movie_ratings, by='title') %>%
pull(movie_avg_diff)
RMSE(model_1_predictions, train$rating)
~~~
```

```
[1] 0.9423477
```

The RMSE taking into account only the movie effect is 0.9423.

Model 2: User Effect

Calculate difference between each user's average rating and the overall average rating.



```
user_ratings %>% mutate(user_avg_diff = avg_user_rating - overall_avg_rating) -> user_ratings
head(user_ratings)

```
```

A tibble: 6 x 4

userId <int>	avg_user_rating <dbl>	num_ratings <int>	user_avg_diff <dbl>
59269	3.264586	6616	-0.24787935
67385	3.197720	6360	-0.31474508
14463	2.403614	4648	-1.10885074
68259	3.576933	4036	0.06446740
27468	3.826870	4023	0.31440529
19635	3.498807	3771	-0.01365852

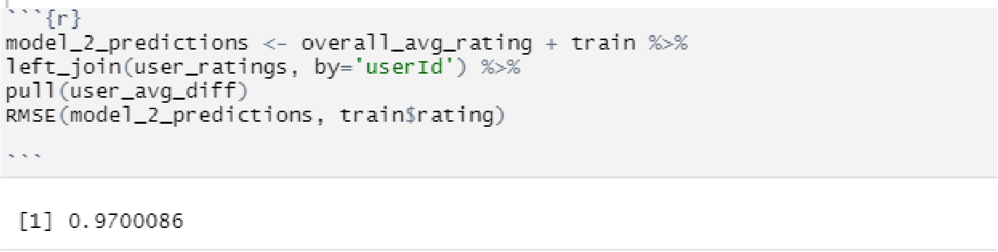
6 rows



~~~~
```{r}
qplot(user_avg_diff, data = user_ratings, bins = 10, color = I("black")) +
labs(title = "Distribution of Average User Rating and Global Average Differences",
x = "Difference Between Average User Rating and Global Average",
y = "Number of Users")
```
~~~~

Distribution of Average User Rating and Global Average Differences
30000
20000
10000
0
Number of Users
-3
-2
-1
0
1
2
Difference Between Average User Rating and Global Average

As you can see, a lot of users have an average rating close to the global average (difference of 0) and few users have an average rating that is far from the global average (difference of +/- 2). In addition, the average rating for the specific user is more likely to have a positive impact on each rating since there are more positive differences



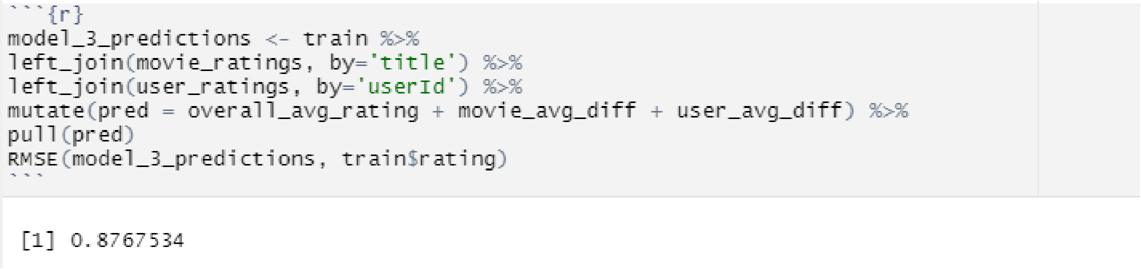
~~~~
```{r}
model_2_predictions <- overall_avg_rating + train %>%
left_join(user_ratings, by='userId') %>%
pull(user_avg_diff)
RMSE(model_2_predictions, train$rating)

```

 [1] 0.9700086
~~~~

The RMSE taking into account only the user effect is 0.9700, which is higher (worse) than model 1.

Model 3: Movie & User Effect

Finally, I will take into account both movie and user effect.



~~~
```{r}
model_3_predictions <- train %>%
left_join(movie_ratings, by='title') %>%
left_join(user_ratings, by='userId') %>%
mutate(pred = overall_avg_rating + movie_avg_diff + user_avg_diff) %>%
pull(pred)
RMSE(model_3_predictions, train$rating)
```

 [1] 0.8767534
~~~

The RMSE taking into account both the movie and user effect is 0.8767.

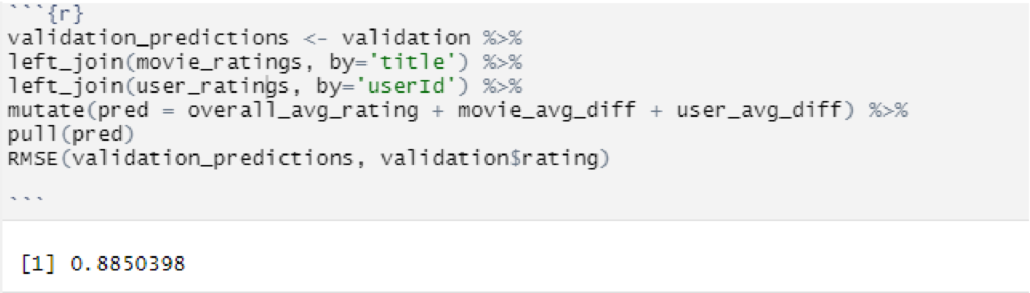
RESULTS

Model Evaluation

MODEL	RMSE
MODEL1: MOVIE EFFECT	0.9423
MODEL2: USER EFFECT	0.9700
MODEL3: MOVIE AND USER EFFECT	0.8767

Model 3 (Movie & User Effect) has the lowest RMSE. Thus, I will deploy this model to the validation dataset.

Model Deployment



~~~
```{r}
validation_predictions <- validation %>%
left_join(movie_ratings, by='title') %>%
left_join(user_ratings, by='userId') %>%
mutate(pred = overall_avg_rating + movie_avg_diff + user_avg_diff) %>%
pull(pred)
RMSE(validation_predictions, validation$rating)

```

[1] 0.8850398
~~~

After deploying model 3, which incorporates movie and user effects, the resulting RMSE is 0.885

CONCLUSION

In conclusion, taking into account user preferences and a movie's average rating does a better job of predicting ratings than simply taking into account only user effects or only movie effects.

Our model does not take into account changes in user preferences which is a limitation of our model. However, a component can be implemented in future iterations to capture this feature. For example, If we want to create a rolling average metric that would determine a user's average rating of their 10 most recent ratings. This may better represent user preferences because it takes into account possible changes in user tendencies. However, it may also create a lot of "noise". For example, a user may decide to watch a lot of highly rated movies back-to-back which would skew predictions.

Ideally, we would also like our model to include a component that would capture similarity scores between users. For example, if user A rates movies similar to user B, and user B rated *Jurassic Park* 3 stars, user A should be likely to rate *Jurassic Park* close to 3 stars. Similarly, I would like my model to integrate an element that would capture similarity scores between movies, based on genres. For example, if Drama and War movies tend to be rated higher than other movies, the model should take it into account. However, considering the dataset contains about 10 million ratings, it is challenging to implement these components without significantly slowing down the time it takes for the model to make predictions.

REFERENCES

- https://rafalab.github.io/dsbook/
- https://mickteaching.wordpress.com/2016/04/19/data-need-to-be-normally-distributed-and-other-myths-of-linear-regression/
- https://stats.stackexchange.com/questions/148803/how-does-linear-regression-use-the-normal-distribution
- https://iovs.arvojournals.org/article.aspx?articleid=2128171
- https://statisticsbyjim.com/regression/interpret-constant-y-intercept-regression/
- https://www.statisticshowto.com/probability-and-statistics/non-normal-distributions/
- https://www.statsdirect.com/help/nonparametric_methods/nonparametric_regression.htm

- https://www.statsdirect.com/help/nonparametric_methods/loess.htm
- https://towardsdatascience.com/regularization-an-important-concept-in-machine-learning-5891628907ea