Edx Project: Movielense

Introduction

The movie industry has been on the rise lately and it has been a challenge for users to find their desired movies. The aim of this project is to create a movie recommendation system for users, which can help them, predict the rating of an unknown movie.
The data used for the project is the MovieLens 10M dataset: (https://grouplens.org/datasets/movielens/10m/, http://files.grouplens.org/datasets/movielens/ml-10m.zip). The data set has 9000055 rows and 6 variables which are: movieId (a number linked to one movie in the data set) , userId (a number linked to one user in the data set) , ratings (are the votes from a range of 1to 5 by users, for each specific movies) , timestamp (represent the date and the times), genres (describes all movies categories) and the title of movies.
In this project we have analyzed and visualized the data, created a recommendation movie model, and finally, validated and optimized that model. The challenge was to create a model with a RMSE < 0.86490.

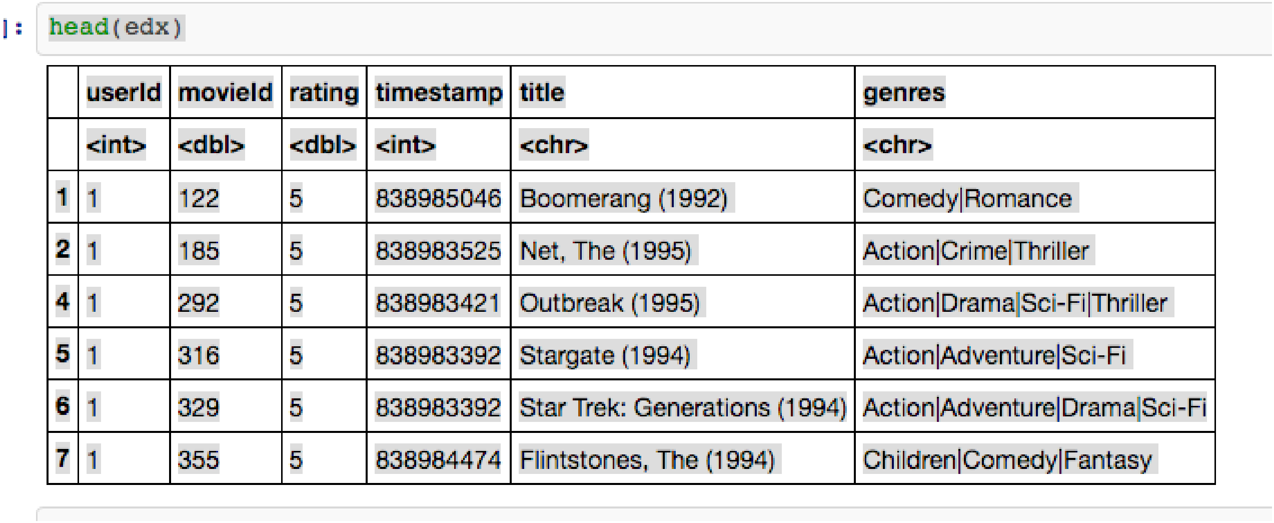
Method/Analysis

-Data cleaning and Partition

We started by downloading all libraries, data set and packages.
We did cleaned the data from the original data set, by rearranging and identifying various variables such as movieid, ratings, movielens and more.
 Out of the new data created (movielens), we extracted ratings to create a data partition for our model. The data partition created was 90 % of training set (edx) and 10 % of that partition was the test set (Validation).

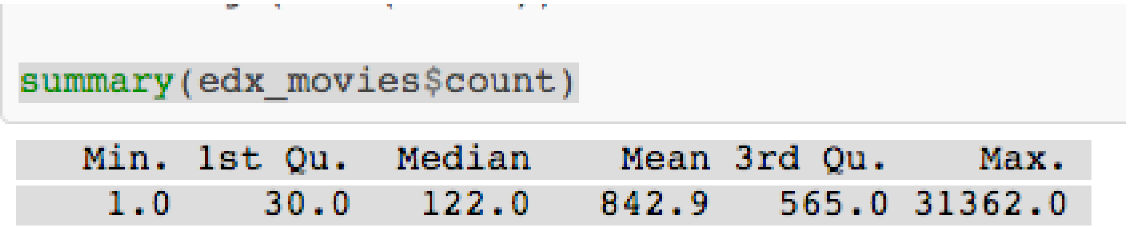
-Data Analysis and Exploration

We pull out the first 6 rows of the data, and summarize the data to have an overview of all the variables (as below).



```
]: head(edx)
```

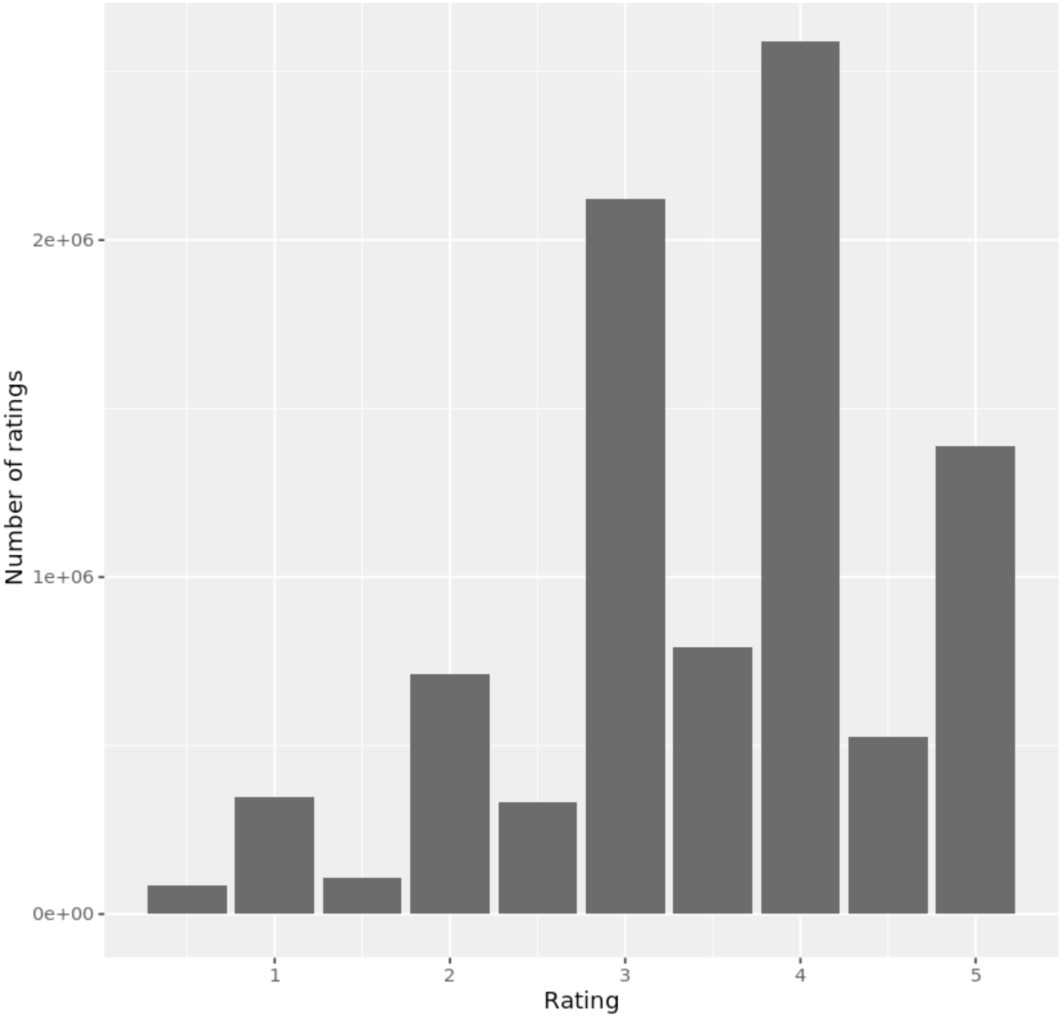
	userId	movieId	rating	timestamp	title	genres
	<int>	<dbl>	<dbl>	<int>	<chr>	<chr>
1	1	122	5	838985046	Boomerang (1992)	Comedy|Romance
2	1	185	5	838983525	Net, The (1995)	Action|Crime|Thriller
4	1	292	5	838983421	Outbreak (1995)	Action|Drama|Sci-Fi|Thriller
5	1	316	5	838983392	Stargate (1994)	Action|Adventure|Sci-Fi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



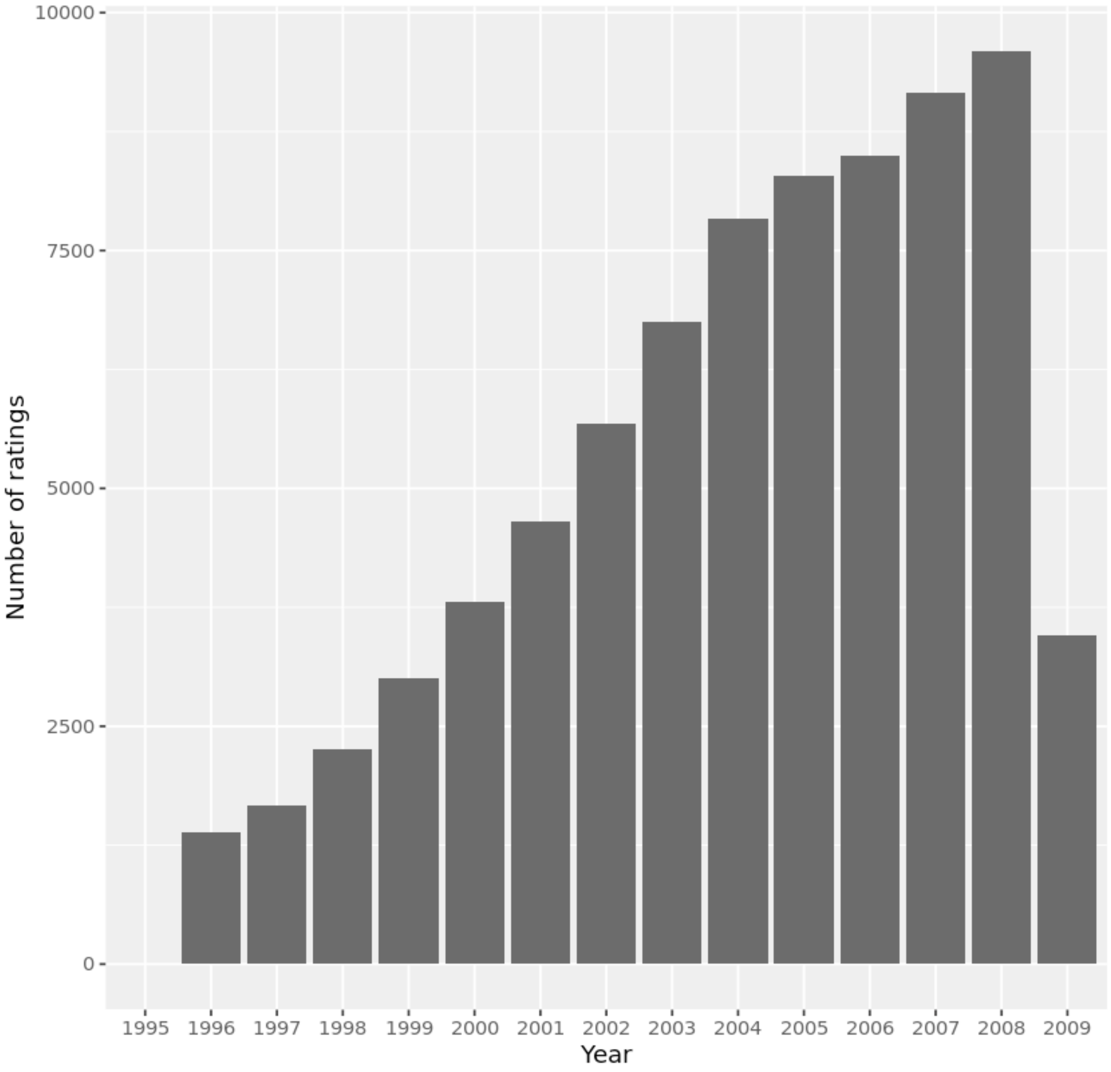
```
summary(edx_movies$count)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
    1.0    30.0   122.0   842.9   565.0 31362.0 
```

By looking at the summary table, we can have some interesting insight; One movie was most rated 31362 times, 122 movies represent half of the ratings, movies was rated 842.9 in average and there is a big gap between the median and the average.

We did the distribution of ratings across the data, we realize that users are more open to give a high rating rate on movies than the opposite. In fact more than 2 millions users gave a rating of 4 compare to less than half a million for rating 1(picture below).



We also analyze the movies ratings distribution per year to see the tendency of the curve. The curve is having a bell shape, similar to a normal curve (picture below).



Insight gained/modeling approach

Since the data set is supervised and the variable to predict (ratings) is quantitative and numeric, we decided to used the linear regression.
We are starting with a model; assuming that all movies in the training set have equal ratings.
Then the formula for that model will be: $Ymu,i= u + \epsilon\ u,i$
Here u represents the average rating for all movies and users in edx, and ϵ represent all errors (in this model we are minimizing ϵ).
Now we can compute the average ratings on edx (u), test it into the validation set and predict the RMSE

Results

*** **All the calculation were completed in R studio (see rmd file)**

-To start we first calculate the overall average rating on the training dataset.

```
u  = mean (edx$rating)
```

-Here is the formula of RMSE

```
RMSE <- function(true_ratings = NULL, predicted_ratings = NULL) {
  sqrt(mean((true_ratings - predicted_ratings)^2))
}
```

- Then we Calculate RMSE using validation ratings

```
  RMSE (validation$rating, u)
```

This model gives us a RMSE of 1.061202, which is not the goal for our ideal model (RMSE < 0.86490.).

Movie effect on RMSE

WE can optimize our model by including the movie effect.
Lets call bi the average rating of movie i.
To calculate the new model we can use the formula:
$Yu, i = u + bi + \epsilon ui$
If we rearrange the formula and isolate bi we will have :
$bi = Yu,i - u$. This means we can calculate bi by subtracting the overall average of each movie rating with the overall average rating of all movies on the training model (edx). Then test it on the validation model.

After compilation of the RMSE via Rstudio, we obtained a RMSE of 0.9439087 which is still is not close to our target

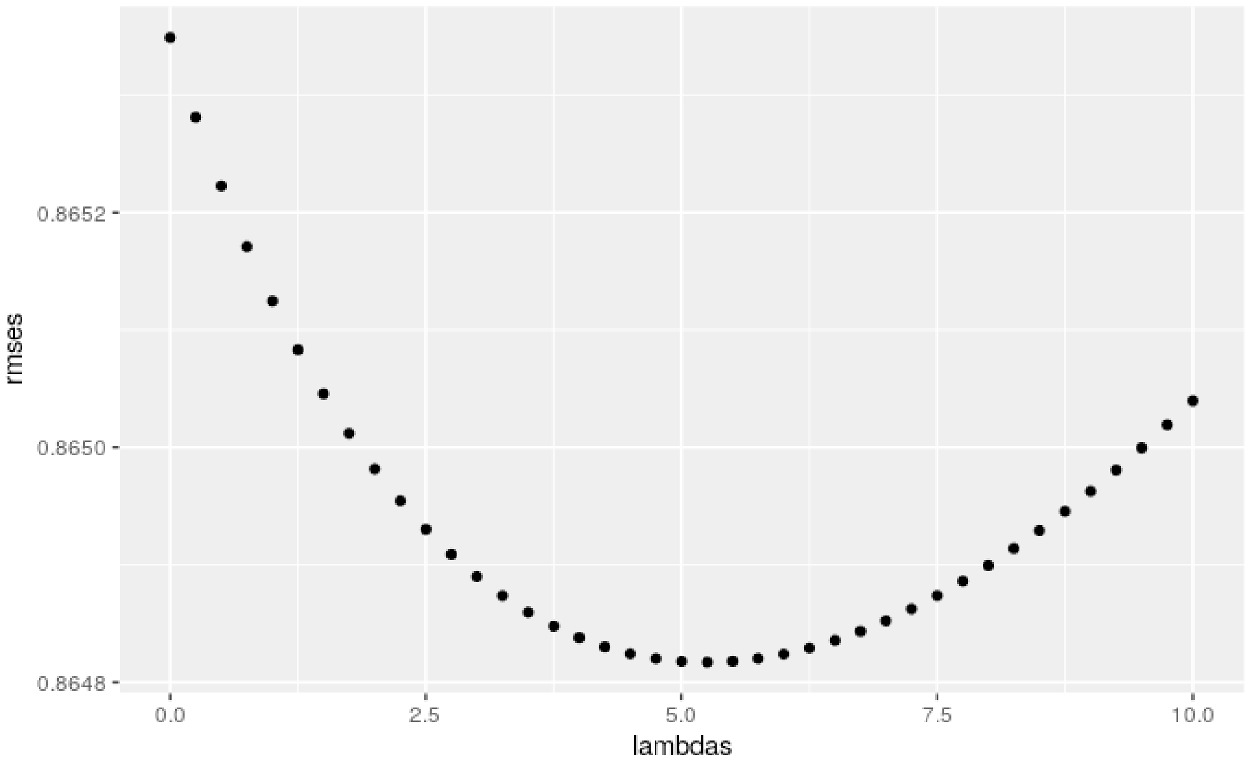
Movie effect and user effect on RMSE.

We can include the user effect (bu) into the previous model to optimize it.
Yu, i = µ + bi + bu + ϵµ, i
We can then compute bu based on the formula above
After findings bu, we can apply it to the train set and test it to the validation set.

After compilation of the RMSE via Rstudio, we obtained a RMSE of 0.8653448 which is getting close to our target

Lets Train the model with the regularization factor lambda (l).

Lets optimized movie and user effect method with the best regularization factor (lamba)
First we will determine the best lambda from a sequence (0:10, by 0.25). and we will use a function to compile lambda in the entire training data set via bi and bu ((bu = sum(rating - bi - u)/(n()+l), bi = sum(rating - u)/(n()+l) to obtain the minimum value of lambda which optimize the RMSE . And we will plot to see the distribution of the curve for visualization.



After compilation in R, we obtained the best regularization lambda = 5.25

Once we have lambda, we can now compute it directly inside bu and bi ((bi = sum(rating - u)/(n()+lambda , bu = sum (rating - bi - u)/(n()+lambda)) .

Then we will Predict the model: pred = u + b_i + b_u

Then compute the RMSE, which is 0.864817

The RMSE of Our final model is 0.864817

Conclusion

We finally meet our target to create a model with a RMSE < 0.86490. Although we didn't take into consideration the errors associate with the data. Including the errors will probably create some variation into our model. For future works, we can train and text another model using other techniques such as Knn to see if we can improve the result. We can also create a tree for this model to have a better visualization.