HarvardX Data Science Capstone Project: MovieLens

Dr. Fatih Uenal

4/14/2021

Overview

In the following, I am presenting the results of the first part of the **HarvardX PH125.9x Data Science: Capstone Project**. The goal of this exercise is to develop and test Machine Learning algorithms to predict movie ratings. The idea for this exercise is based on real-world data challenges faced by commercial enterprises such as Amazon, Facebook, and Netflix who use large data sets to predict specific behaviors of customers (e.g., ratings, purchase, etc.). In this vein, the project idea here is based on the so-called "Netflix Challenge" which took place in October 2006. In this challenge, Netflix approached the data science community asking them to improve their movie recommendation algorithm by 10%. The winner of this challenge would win a Million US Dollars. In September 2009, the winners were announced. A summary of how the winning algorithm was put together **here**, and a more detailed explanation can be found **here**.

Data

The data for this project comes from the GroupLens research lab who generated their own database with over 20 million ratings for over 27,000 movies by more than 138,000 users. The data used here is a subset of this data provided by the course.

Approach

Following the example provided in the textbook, I will use the RMSE (Root Mean Squared Error) as an evaluation metric of the algorithm. The textbook example used here follows the data analysis strategies used by the winning team of the Netflix Challenge. Extending the approach provided by the course, I will run additional algorithms, which take further variables into account. Specifically, in addition to using User IDs and Movie IDs I will also make use of further features contained in the data (e.g., Genre) to better predict movie ratings and compare the RMSE score of each of these increasingly complex models. The lowest RMSE score will determine the best model.

Report structure

1. Exploratory Data Analysis (EDA)

2. Analysis: Modeling & Evaluation

3. Results

4. Conclusion

After running the code snippet provided by the course creators (not-visible in this document), I will start off with an Exploratory Data Analysis section in which I explain the process and techniques I used, including data cleaning, data exploration, and visualization, insights gained, and my modeling approach. Next, I will present my modeling results and discusses the model performance in the results section. Finally, in the conclusion section I will give a brief summary of my findings, its limitations and potential future work.

PLEASE NOTE: That running the code provided here might require up to 75 min ot more to run depending on the processing power of your PC.

1. Exploratory Data Analysis (EDA)

As a starting point, I will first take a closer look at the structure and completeness of the data set provided. As shown below, the data set contains 9,000,055 observations and 6 columns. The columns contain information on individual users (userId), movies (movieId), rating of movies (rating), time of rating (timestamp), title of the movies (title), and the genres (genres) to which the movies belong to. We can also see that the data set contains integer, numeric, and character variable types.

```
## # A tibble: 9,000,055 x 6
##    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
##  3      1     292      5 838983421 Outbreak (1995)       Action|Drama|Sci-Fi|T~
##  4      1     316      5 838983392 Stargate (1994)       Action|Adventure|Sci-~
##  5      1     329      5 838983392 Star Trek: Generation~ Action|Adventure|Dram~
##  6      1     355      5 838984474 Flintstones, The (199~ Children|Comedy|Fanta~
##  7      1     356      5 838983653 Forrest Gump (1994)   Comedy|Drama|Romance|~
##  8      1     362      5 838984885 Jungle Book, The (199~ Adventure|Children|Ro~
##  9      1     364      5 838983707 Lion King, The (1994)  Adventure|Animation|C~
## 10      1     370      5 838984596 Naked Gun 33 1/3: The~ Action|Comedy
## # ... with 9,000,045 more rows
```

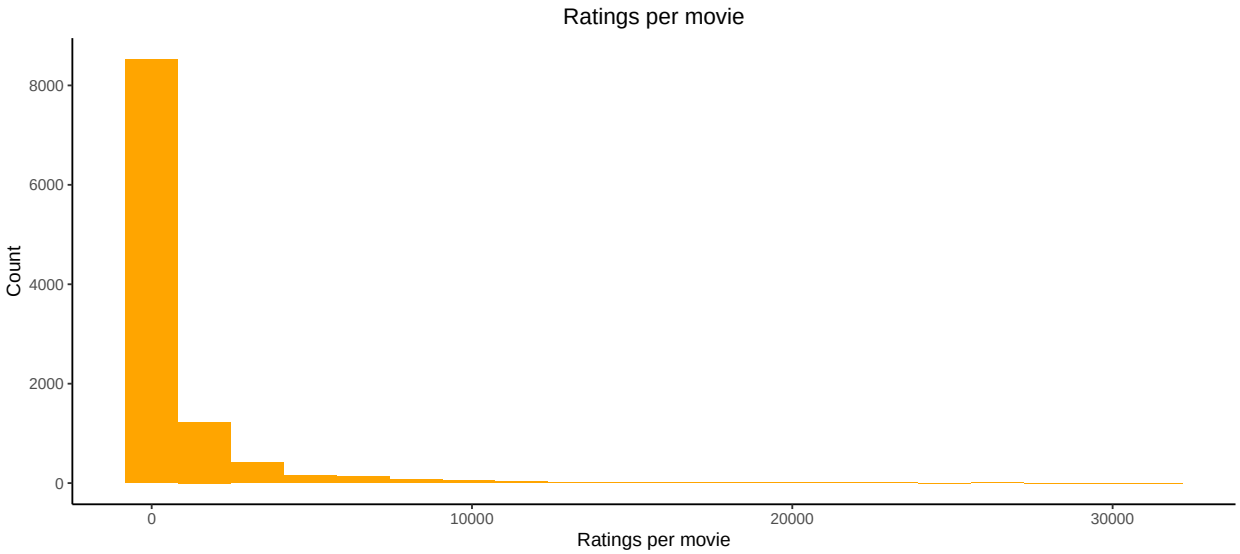
Some Machine Learning approaches require that data sets do not include missing values (NaNs). Hence, as a next step, I check whether the data set contains any missing values. In case of missing values, there are different methods such as replacing NaNs with the mean or median value of a given column or more complex imputation methods which I could apply to prepare the data for the modeling part. However, as seen below, the data set does not contain NaNs and thus no imputation seems required.

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

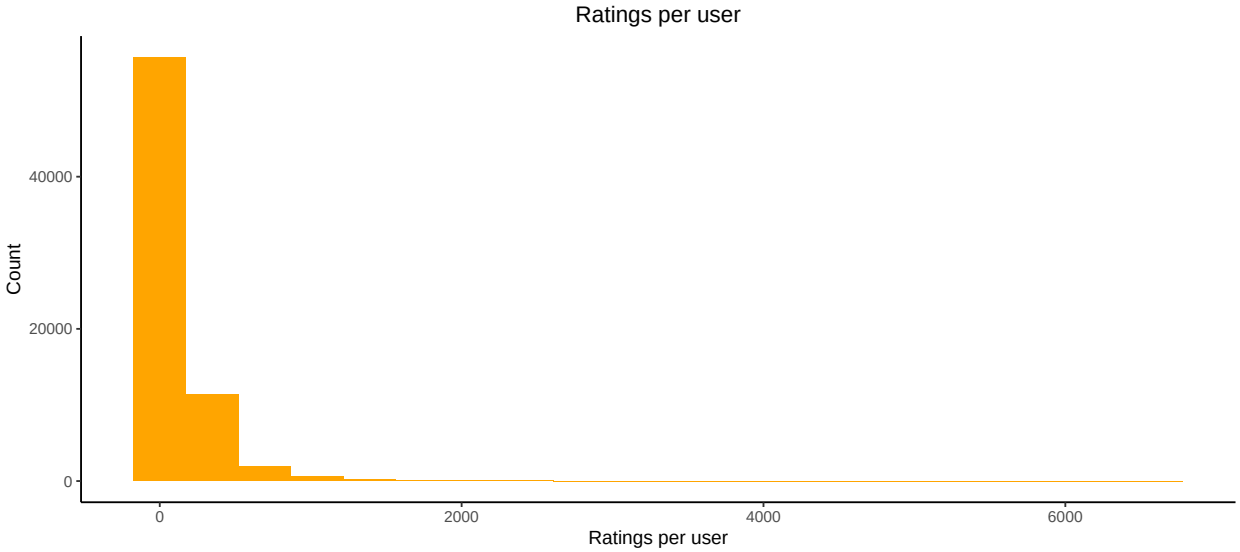
To further drill down in to the data set, I next look at individual user and movie numbers. As shown below, the data set contains 69878 unique users and 10677 in total. Thus, as described in the textbook, not every user seems to have rated every movie in the data set and the task of this challenge can thus be described as to predicting how a User `X` might predict a Movie `Y` for which he/she has not provided a rating. In other words, the task is to 'fill in the *missing values*' by predicting them based on the information present in the data set.

```
##   n_users n_movies
## 1   69878    10677
```

It is probably fair to assume that some movies like **Blockbusters** will have received more ratings compared to for example more **Independent/low-budget production movies**. To see how movie ratings are distributed across movies, I next plot the distribution of ratings across movies below. Indeed, as shown below, some movies have received many more ratings compared to other movies.



Similarly, the number of ratings per user might show variation with some users being more active compared to others. Plotting the number of ratings per user confirms this assumption. The distribution of ratings shows a strong left-skew as seen below, confirming my intuition.



Besides ratings per user and movie, the data also provides information on the specific genres the movies fall into. Similar to the previous distributions of ratings, it might be the case that the distribution of ratings might vary across genres. Again, for example **Horror** movies might receive different ratings compared to **Comedies**. To analyze whether this is the case, I analyze the number of total ratings across genres next.

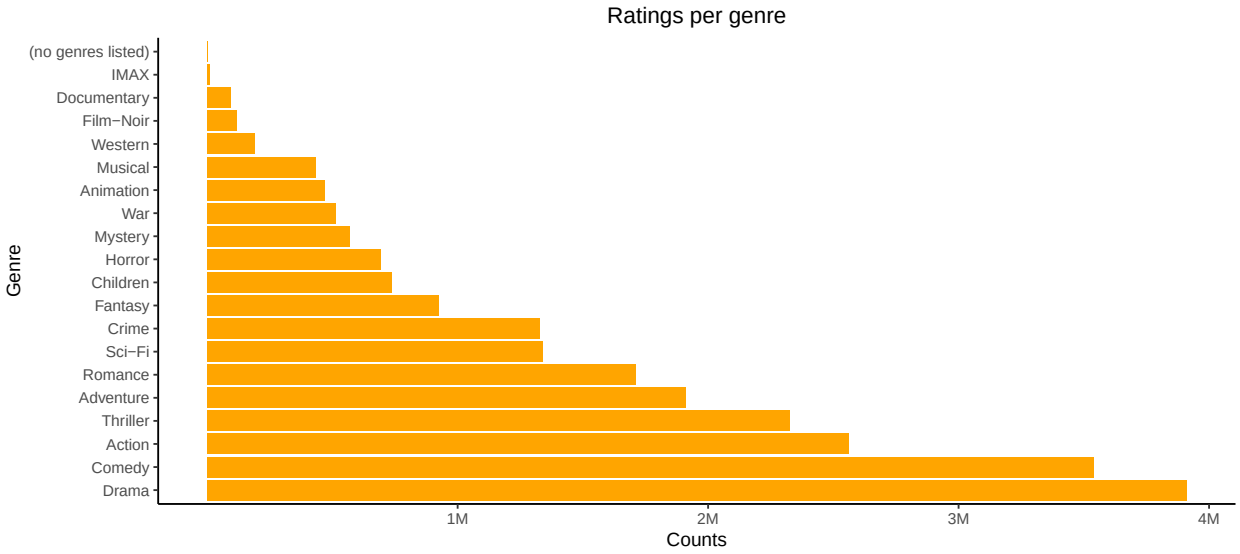
Before plotting the distribution, I first need to check which genres are present in the data set. To be able to plot the data, I also need to apply some data wrangling to the genre column using the `str_extract_all` function.

As seen below, the data set contains 20 unique genre categories.

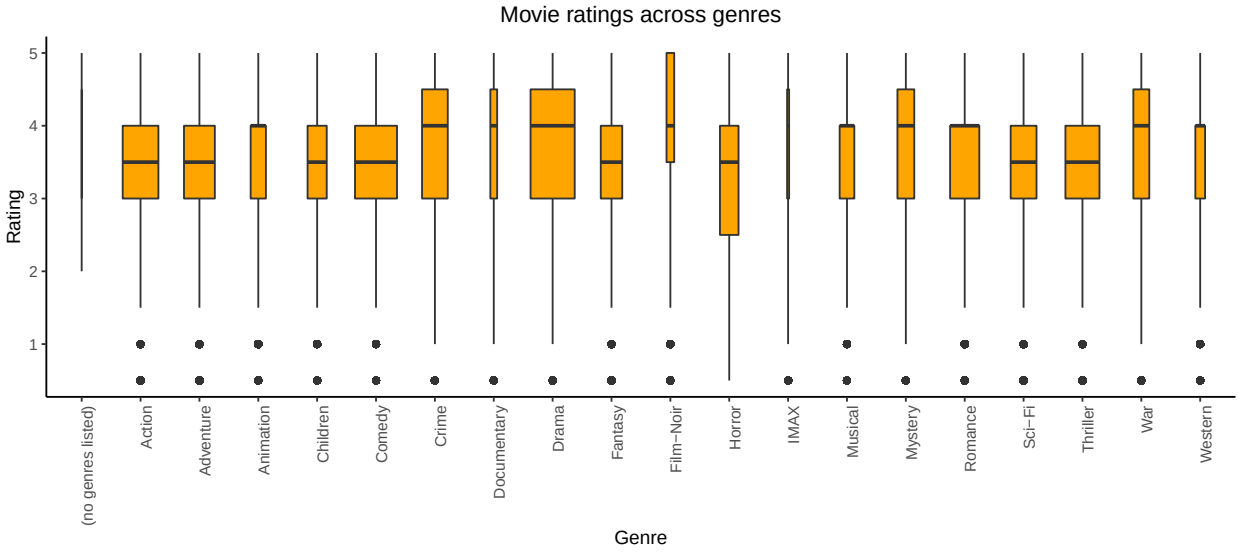
```
##  [1] "Comedy"              "Romance"             "Action"
##  [4] "Crime"               "Thriller"            "Drama"
##  [7] "Sci-Fi"              "Adventure"           "Children"
## [10] "Fantasy"             "War"                 "Animation"
```

```
## [13] "Musical"             "Western"             "Mystery"
## [16] "Film-Noir"           "Horror"              "Documentary"
## [19] "IMAX"                "(no genres listed)"
```

The plot below confirms my assumption that the number of movie ratings varies considerably across genres with **Drama** and **Comedy** showing the highest number of ratings while **IMAX** and **Documentaries** received the lowest total number of ratings.



Similarly, the plot below indicates that the average ratings show some variation across genres as well. For example, the genres **Crime, Documentary**, and **Drama** show higher ratings on average compared to **Action** and **Adventure** movies.



Based on the above EDA, I will use User IDs, Movie IDs, and Genres as predictors of Movie ratings. I will do so incrementally, to ascertain whether adding more predictors improves my prediction. However, before starting to include predictors, I need to determine what I mean by 'improving' our prediction. Following the textbook example which itself follows the aforementioned Netflix Challenge, I will first describe a baseline model and a measure of model-evaluation.

Analysis: Modeling & Evaluation

Loss function: RMSE

In the Netflix challenge which inspired this exercise, the root mean squared error (RMSE) -a typical error loss- was employed to determine the winner. Here, in a similar vein, I define $y_{u,i}$ as the rating for movie i by user u and denote our prediction with $\hat{y}_{u,i}$. The RMSE is then defined as:

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

with N being the number of user/movie combinations and the sum occurring over all these combinations.

To measure how close the predictions are to the true values in the validation set I will also use the RMSE, which is defined by the following function:

Given the relatively large size of the data set, applying a simple *lm* model is not feasible in terms of computational resources. To mitigate this issue, I manually compute the least square estimates first. Moreover, given the sparse nature of the data set, I will also apply regularization (last step).

In line with the course material, as a baseline model, I chose to predict movie ratings irrespective of users, movies or genres as predictive features. This baseline model will serve as point of comparison upon which I want to improve. The formal representation of this model can be formulated as follows:

$$Y_{u,i} = \mu + \varepsilon_{u,i}$$

u represents the index for users, and i movies and with $\varepsilon_{i,u}$ independent errors sampled from the same distribution centered at 0 and μ the "true" rating for all movies. We know that the estimate that minimizes the RMSE is the least squares estimate of μ and, in this case, is the average of all ratings:

```
## [1] 3.527019
```

Baseline Model: Just the average

Predicting unknown ratings based on μ, we obtain the following RMSE score:

Method	RMSE
Just the average	1.052262

This RMSE score can be interpreted similar to the standard deviation. In other words, the baseline model would have an error of:

```
## [1] 1.052262
```

More to the point, the model would make a relatively large error of 1 Rating unit ('Star'). There is certainly room to impove upon this model given that we have seen that some features contained in the data set such as Movie IDs, User IDs, and Genre are differently associated with the outcome. In the following I will therefore use each feature as predictor to improve the prediction model.

First model: Movie Effects Model

As indicated in the EDA section, movie ratings differ considerably with some movies receiving more and/or higher ratings than others. In turn, specific movies with higher/lower overall ratings might have positive/negative impacts on predicting a rating for another movie. Therefore, as a first step, I will model Movie IDs as a predictor of movie ratings and check whether the RMSE score improves compared to the baseline model. As indicated below, the RMSE score for the 'Movie Effect Model' is lower than the baseline model

and thus using Movie IDs as a predictive feature yields a slightly better prediction compared to simply predicting all ratings by 'Just the average' rating of all movies.

Method	RMSE
Movie Effect Model	0.94107

Second model: Movie + User Effects Model

Similar to Movie IDs association with ratings, the EDA also indicated the the relationship between individual users and ratings is also heterogeneous with some users rating more movies positively or negatively while others provide fewer ratings either positively or negatively. Hence, in the second model, besides Movie IDs, I will also include the User IDs as predictor and check whether a 'Movie + User Effects Model' yields a better prediction score than the baseline and the 'Movie Effect Model'.

Indeed, as indicated below, including the User IDs as predictors further improved the prediction score of the model.

Method	RMSE
Movie + User Effects Model	0.863366

Third model: Movie + User + Genre Effects Model

As indicated in the EDA section, the movie ratings also showed variation in conjunction with specific genres. Therefore, in addition to User and Movie effects, I decided to included Genre as further predictor in to the Model and see if this would improve the RMSE score even further. And indeed, as seen below, including Genre as further predictor slightly improved the prediction accuracy even more.

Method	RMSE
Movie + User + Genre Effects Model	0.8632723

Fourth model: Genre*User effects

Another potentially interesting predictor to include would be a combined `User x Genre` predictor. It is reasonable to assume that users cluster around genres with some more partial to independent movies while others being more partial to Blockbuster movies for example. In the next model, I will test a 'User + Movie + Genre + Genre*User' model by further including `User x Genre` as further predictor into the model and see whether the model prediction improves. As can be seen below, this model yielded a slightly higher RMSE value and thus lowered the previous score. Hence, I will not consider this model for the final evaluation on the test set.

Method	RMSE
Movie + User + Genre + Genre-User Effects Model	0.8657195

Fifth model: Genre*Movie effects

Yet another potentially interesting predictor to include would be a combined `Movie x Genre` predictor. Similar to the previous idea of combining the effects of the User and Genre features, it is reasonable to assume that movies cluster around genres with some movies falling into the genre of independent movies while others falling into the Blockbuster genre. Thus, in the next model, I will test a 'User + Movie + Genre + Genre*MOVIE' model by further including `Movie x Genre` as further predictor into the model and see whether the model prediction improves. As can be seen below, this model yielded a slightly lower RMSE

value and thus improved the previous RMSE high-score a bit. Hence, I will consider this model for the final evaluation on the test set.

Method	RMSE
Movie + User + Genre + Genre-Movie Effects Model	0.8621138

Sixth model: Regularization Model

As a further step to improve the prediction score of the models, I now consider applying `Regularization`. The general idea behind regularization is to constrain the total variability of the effect sizes. Constraining the total variability of effect sizes is pertinent given that `Regularization` permits to penalize `biased` effect sizes such as relatively large estimates that are formed using small sample sizes and thus optimizes the results by correcting those. A good explanation for this is provided in the textbook which accompanies the course:

Consider a case in which I have movie $i = 1$ with 100 user ratings and 4 movies $i = 2, 3, 4, 5$ with just one user rating. So we intend to fit the following model:

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

Suppose we know the average rating is, say, $\mu = 3$. If we use least squares, the estimate for the first movie effect b_1 is the average of the 100 user ratings, $1/100 \sum_{i=1}^{100}(Y_{i,1} - \mu)$, which we expect to be a quite precise. However, the estimate for movies 2, 3, 4, and 5 will simply be the observed deviation from the average rating $\hat{b}_i = Y_{u,i} - \hat{\mu}$ which is an estimate based on just one number so it won't be precise at all. Note these estimates make the error $Y_{u,i} - \mu + \hat{b}_i$ equal to 0 for $i = 2, 3, 4, 5$, but this is a case of over-training. In fact, ignoring the one user and guessing that movies 2,3,4, and 5 are just average movies ($b_i = 0$) might provide a better prediction. The general idea of penalized regression is to control the total variability of the movie effects: $\sum_{i=1}^{5} b_i^2$. Specifically, instead of minimizing the least squares equation, we minimize an equation that adds a penalty:

$$\sum_{u,i} \left(y_{u,i} - \mu - b_i\right)^2 + \lambda \sum_i b_i^2$$

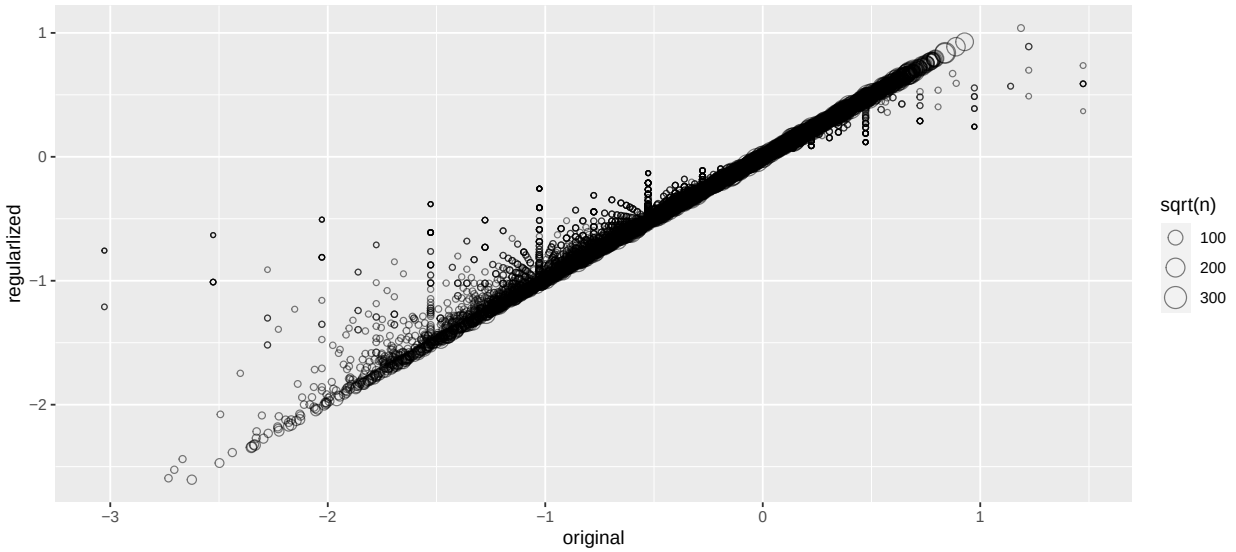
The first term is just the sum of squares and the second is a penalty that gets larger when many b_i are large. Using calculus we can actually show that the values of b_i that minimize this equation are:

$$\hat{b}_i(\lambda) = \frac{1}{\lambda + n_i} \sum_{u=1}^{n_i} (Y_{u,i} - \hat{\mu})$$

where n_i is the number of ratings made for movie i. This approach will have our desired effect: when our sample size n_i is very large, a case which will give us a stable estimate, then the penalty λ is effectively ignored since $n_i + \lambda \approx n_i$. However, when the n_i is small, then the estimate $\hat{b}_i(\lambda)$ is shrunken towards 0. The larger λ, the more we shrink.

Let's compute these regularized estimates of b_i using $\lambda = 3$.

To see how the estimates shrink, let's make a plot of the regularized estimates versus the least squares estimates.




```
## [1] 1.153109
```

As shown below, the penalized estimates provide a large improvement over the least squares estimates.

```
## [1] "Movie Effect Model"
```

```
## [1] 0.94107
```

```
## [1] "Regularized Movie Effect Model"
```

```
## [1] 0.9410386
```

Given the improvements for the movie model shown above, in the final step of this project, I decided to apply regularization to all the models next. To ascertain the best value of the penalty term λ, I am first defining a vector containing a sequence of incrementally increasing λ values. Since λ is treated as a tuning parameter here, I am essentially performing a grid search to choose the optimal λ value.

Method	RMSE
Regularized Movie + User + Genre + Movie-Genre Model	0.8616823

Results section

Below are the RMSE scores for each model used on the test set. As shown below, the final model in which the regularized effects of users, movies, genres, and the genre*movie features were modeled as predictors yielded the best score achieving an RMSE of 0.8616823. Overall, the RMSE score is below RMSE < 0.86490 and thus meets the requirements of the course.

Method	RMSE
Just the average	1.0522618
Movie Effect Model	0.9410700
Movie + User Effects Model	0.8633660
Movie + User + Genre Effects Model	0.8632723
Movie + User + Genre + Genre-User Effects Model	0.8657195
Movie + User + Genre + Genre-Movie Effects Model	0.8621138
Regularized Movie + User + Genre + Movie-Genre Model	0.8616823

Conclusion section

Here, I reported the results of the first part of my **HarvardX PH125.9x Data Science: Capstone Project**. The goal of this exercise was to develop and test Machine Learning Algorithms to predict movie ratings.

The data for this project came from the GroupLens research lab who generated their own database with over 20 million ratings for over 27,000 movies by more than 138,000 users. The data used here is a subset of this data provided by the course.

I programmed several regression algorithms to predict movie ratings using the available features of movies, users and genres. After an exploratory data analysis in which I analyzed the structure, composition, completeness of the data set as well as the associations between the features and the outcome variable, I controlled for missing values and applied some data wrangling techniques to prepare the data for the modeling phase. The EDA indicated that movie ratings might be predicted by a) Users, Movies, and Genres, and that the data set represents a so-called sparse data set. Based on my observations during the EDA, I modeled several algorithms to predict the outcome variable and compared the RMSE scoreof each alogorithm to determine the best model (lowest RMSE value). Due to the sparse nature of the data set, I also applied regularization of my final model to account for potentially `biased` effect sizes. Regularization permits to penalize `biased` effect sizes such as large estimates that are formed using small sample sizes and thus optimizes the results.

The `Regularized Movie + User + Genre + Movie-Genre` Model achieved an RMSE of 0.8616823.

While the final RMSE score meets the requirements of the course, several other methods and techniques could be considered to further improve the results. First, the data set contains further features, time of rating (`timestamp`) and title of the movies (`title`), which I did not consider as predictor variables due to time and resource constrains. Future modeling could test these variables as further predictors to improve the results. Second, one might also consider other approaches and algorithms such as Matrix Factorization, Random Forests, PCA, and SVD to achieve better results. Given the relatively large size of the data set, one should keep in mind though that these approaches require high computational processing power and as such, a regular laptop, such as the one I used in my project requires would require several hours to process the models, without sufficient computing power.

In closing, the project was fun and interesting to accomplish and I hope that my results and approach will be useful for others in their own learning journey. For the second part of my **HarvardX PH125.9x Data Science: Capstone Project**, I will use a publicly available data set to predict climate change threat perceptions employing more sophisticated ML algorithms such as Random Forests with XGBoost and programm Artificial Neural Networks.