MovieLens Project

Craig Hamlin

23/07/2021

INTRODUCTION

In this project we utilize the MovieLens publically available dataset to create a movie recommendation algorithm. This algorithm, similar to the 'Netflix Challenge' algorithm, is intended to predict how any user of the MovieLens site would rate any given movie. The MovieLens database itself is a collection of over 10 million ratings which have been compiled from in excess of 10,000 movies and 72,000 users. The dataset, though vast in its content, is minimal in its features, simply containing user and movie ID's as well as rating, genre, and the timestamp. The minimal features will hopefully provide a less expansive interpretation of the data which will aid our algorithm.

Create training and test sets

After importing the data, to initiate our process of determining a prediction algorithm the data is split into training (edx) and test (validation) sets, whereby the test set comprises 10% of the original data. The train data, comprising the remaining 90% of original data, is used exclusively in the model development. Once the final model is determined we will return to the Validation data to evaluate the model. Determining root mean squared error (RMSE) as our accuracy target, we utilize data wrangling, least squares estimates, and regularization to train and combine our best models to find the highest accuracy.

DATA VISUALIZATION AND ANALYSIS

Through using code supplied from the edx capstone program the data was imported into Rstudio and split into a training set called 'edx' and a test set called 'validation'. The two sets each contain the columns: userID, movieID, rating, timestamp, title and genres.

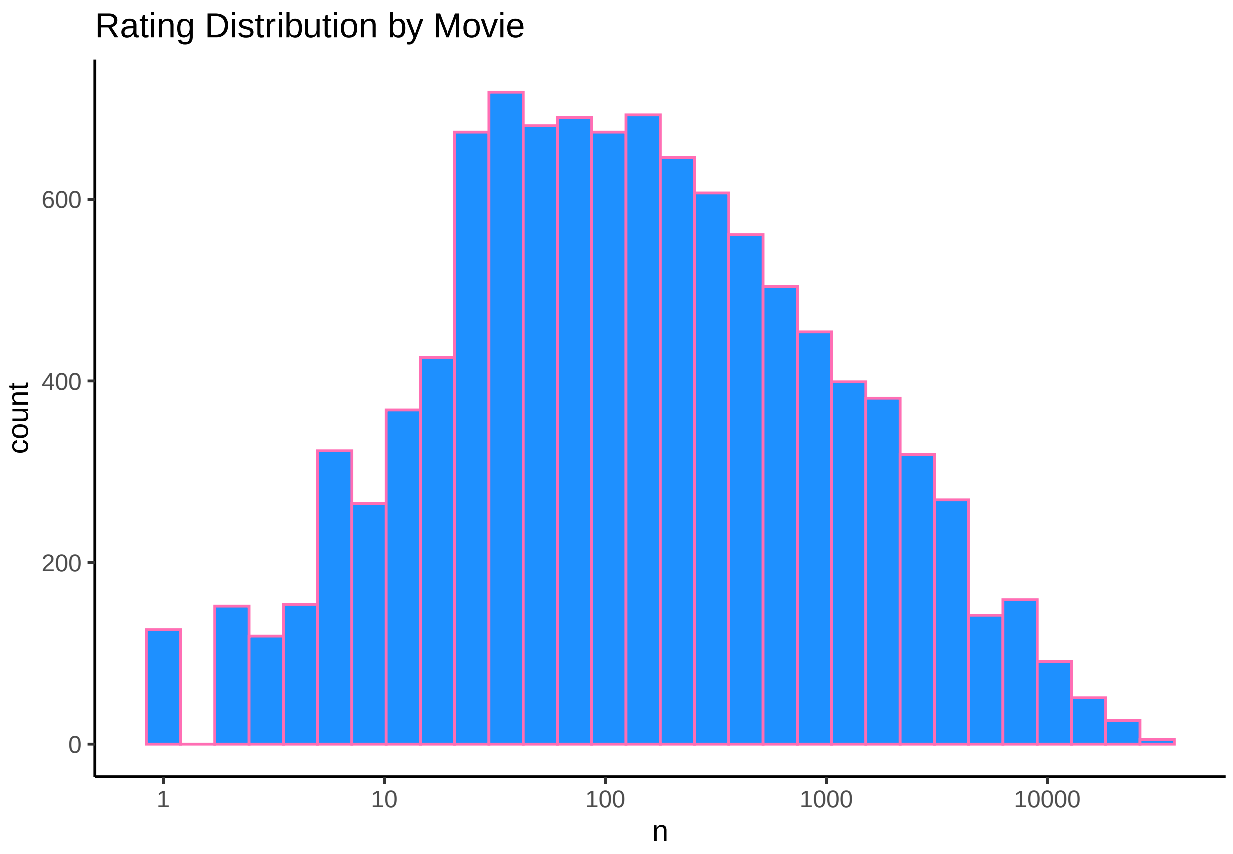
edx

```
## # A tibble: 6 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: Generations~ Action|Adventure|Dram~
## 6      1     355      5 838984474 Flintstones, The (1994) Children|Comedy|Fanta~
```

validation

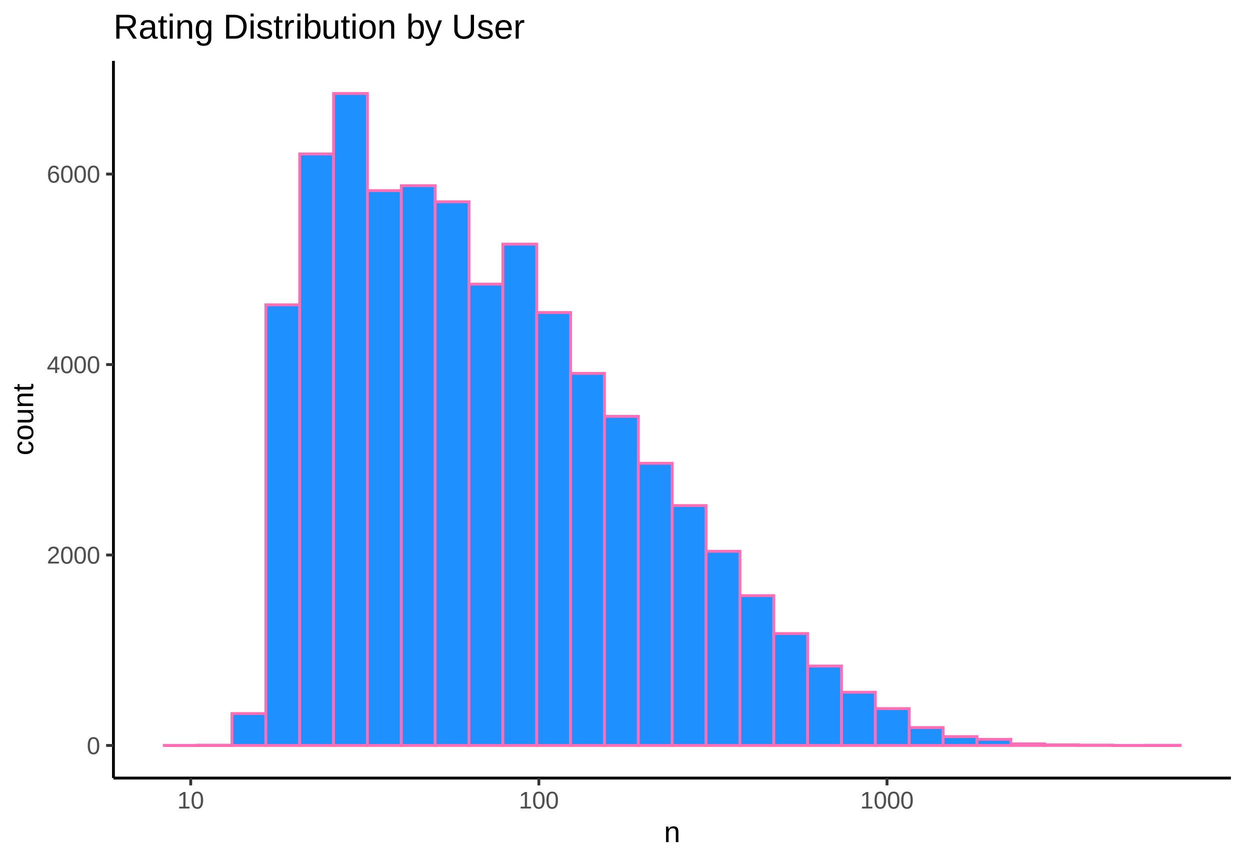
```
## # A tibble: 6 x 6
##   userId movieId rating timestamp title                        genres
##    <int>   <dbl>  <dbl>     <int> <chr>                        <chr>
## 1      1     231      5 838983392 Dumb & Dumber (1994)         Comedy
## 2      1     480      5 838983653 Jurassic Park (1993)         Action|Adventure|S~
## 3      1     586      5 838984068 Home Alone (1990)            Children|Comedy
## 4      2     151      3 868246450 Rob Roy (1995)               Action|Drama|Roman~
## 5      2     858      2 868245645 Godfather, The (1972)        Crime|Drama
## 6      2    1544      3 868245920 Lost World: Jurassic Park~ Action|Adventure|H~
```

Variance in movie review totals



As displayed in the histogram above: there is a great deal of variance in the number of reviews per movie. Some movies get rated more than others. Popular movies and box office successes are rated by many and much less promoted or less popular films are rated by few.

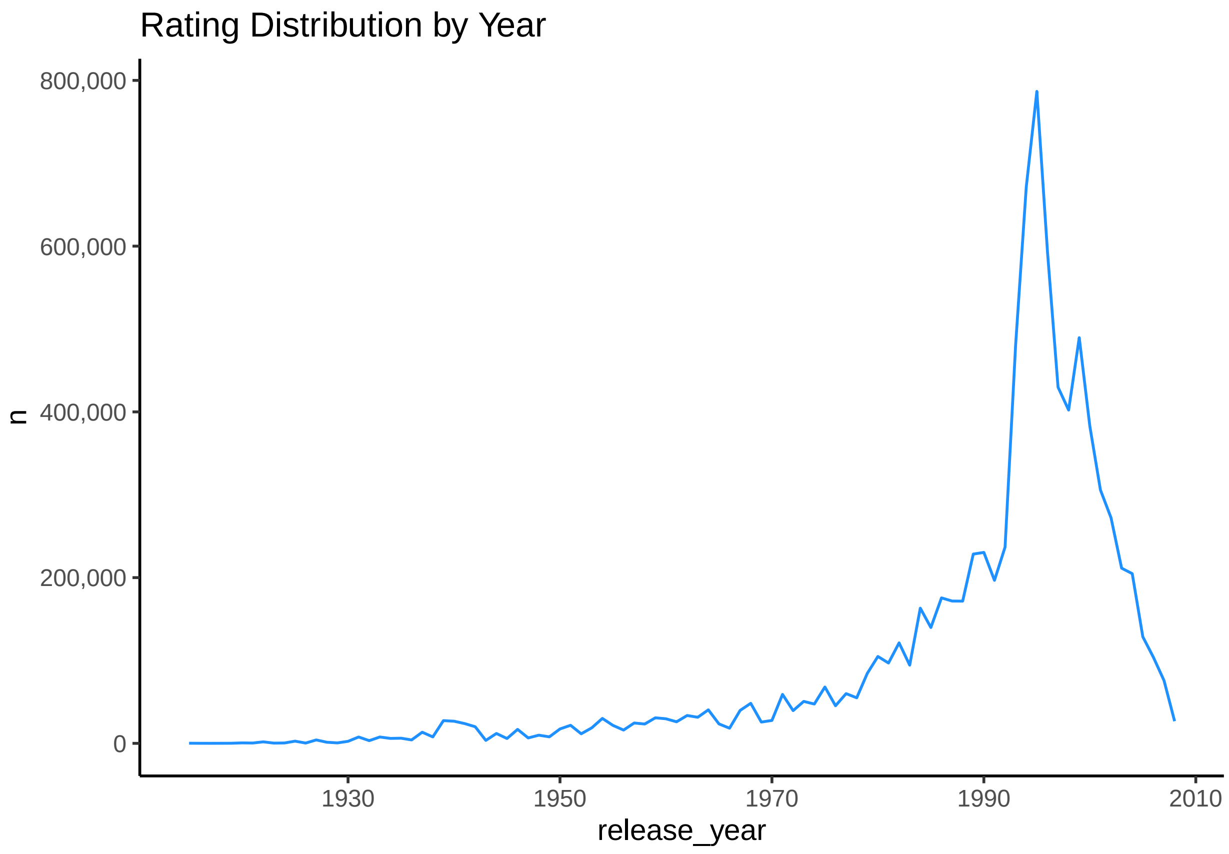
Variance in user review totals



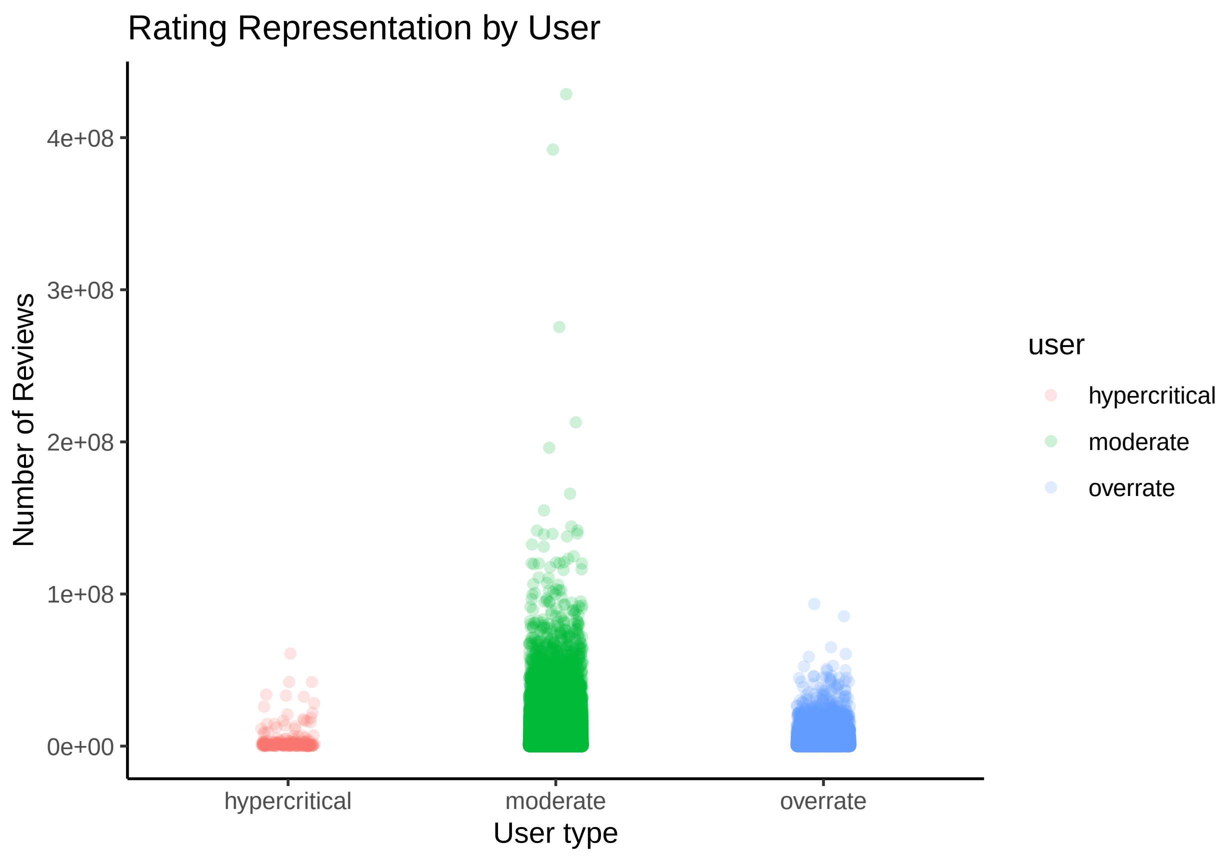
As shown above: some users are considerably more active in creating reviews than others. Some users have rated over 1000 movies while others have only rated a few

Year of release

Further inspecting the data we see that movies range in year from 1915 until 2008 and the number of reviewed movies differs greatly by year.



Identifying differences in user behaviour



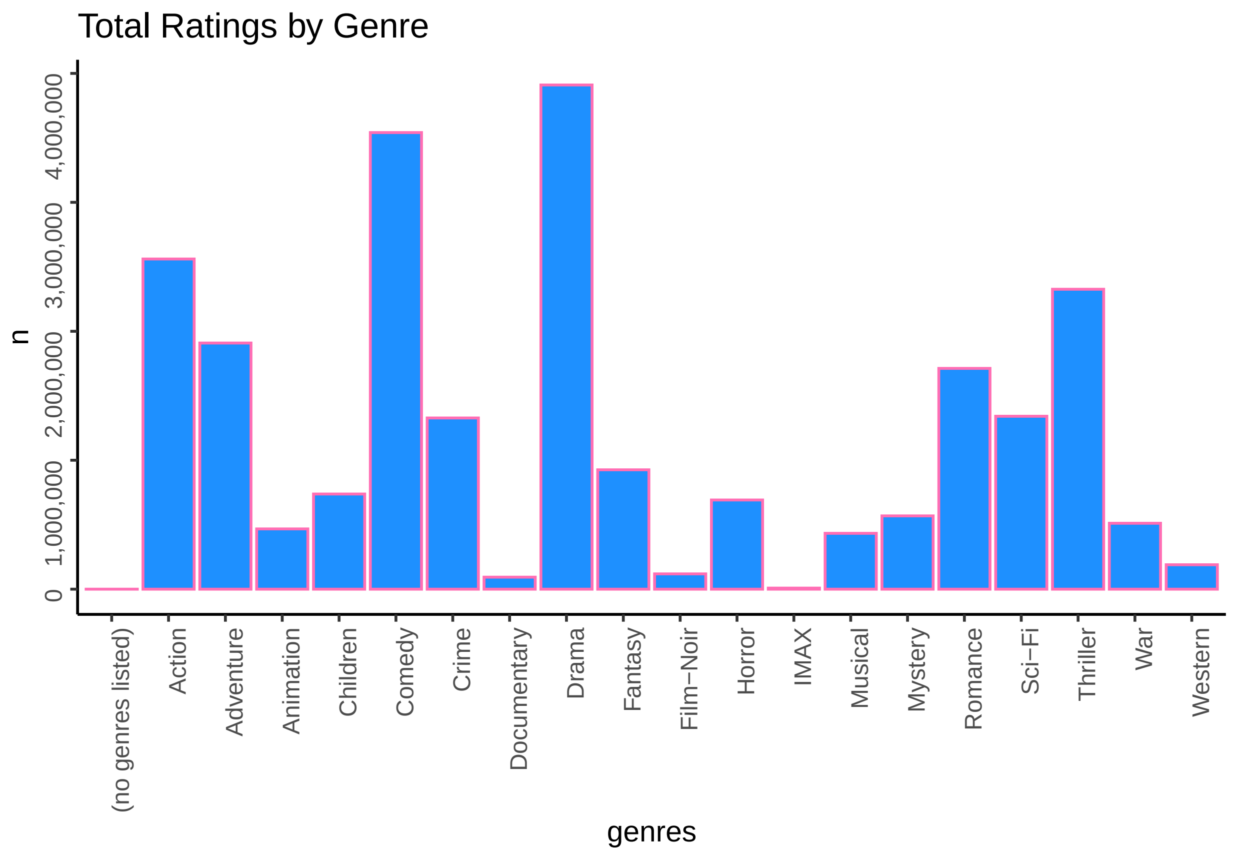
In the representation above we see there is also variance in the critique level of users. There are users that are shown to consistently rate movies highly but there are also other users who are significantly more conservative in their rating and offer few high ratings Note that users that fall into the moderate rating system seem to have more ratings in general. We also see that even though users that overrate films and hypercritical users are the minority there are significantly less hypercritical users. The limited number of hypercritical users could make it difficult to analyze a realistic penalty later due to small sample size.

Film Genres

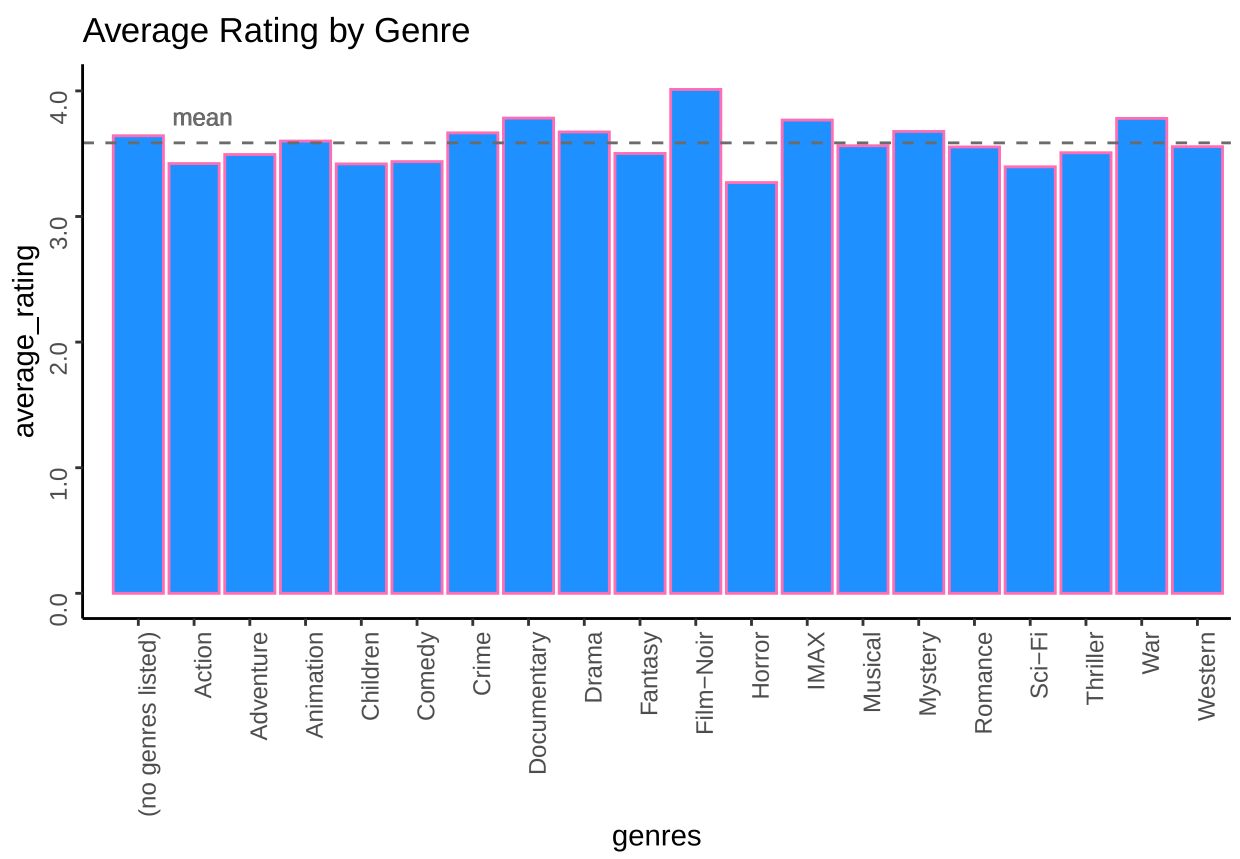
Film genres can be be broken down to twenty separate varieties

```
##  [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)"
```

An examination of genres listed by total number of reviews shows significant variance.



Likewise, we also see that some genres are rated highly while others are rated poorly.



RMSE

The process of establishing a prediction algorithm for this project will focus on obtaining the residual mean squared error.

$$\sqrt{\frac{1}{N}\sum_{e}(\hat{y}_e - y_e)^2}$$

In r programming language this equation is expressed as:

rmse <- function(true_ratings, predicted_ratings){
sqrt(mean((true_ratings - predicted_ratings)^2))
}

MODELING

Step One: The Average

Due to the immense size of the data matrix a function like lm (linear model) would be overwhelming to most cpu's, therefore we will compute the RMSE manually. To provide a baseline we first develop simple model which predicts the same rating for all users regardless of any other feature or effect. The baseline model is represented as such:

$$Y = \mu + \epsilon$$

In which Y is the predicted rating (Y hat), μ is the true rating for all movies and users, and ϵ represents independent errors sampled from the same distribution centered at zero.

method	RMSE
Average	1.061202

In calculating the average rating of all movies across all users we obtained an rmse of 1.06

Step Two: The Movie Effect

The individual movies in this database are rated independently of each other which creates a contrasting series of ratings. Accounting for this variability found in the rating of individual movies we can improve our model by adding a term, b_i , that represents the average rating for movie i:

$$Y = \mu + b_i + \epsilon$$

method	RMSE
Movie Effect Model	0.9439087

in adding the movie effect the model rmse improves to 0.943

Step Three: The User Effect

Similar to how movies are rated differently, individual users have their own different ranking data. Some users might rate movies much higher than others on a consistent basis and vice versa. Accounting for this variability found in the ratings of individual users we can improve our model by adding a term, b_i , that represents the average rating for user u:

$$Y = \mu + b_i + b_u \epsilon$$

method	RMSE
Movie + User Effects Model	0.8653488

in adding the user effect the model rmse improves to 0.865

Step Four: The Genre Effect

As previously shown through our experimental data analysis, different genres are rated differently. As a simple example we see from our earlier data analysis that Drama has a higher overall rating than Horror. In this next step we introduce error that is induced by the specificity of rating based on genre. Accounting for this variability found in the overall ratings of different genres we can improve our model by adding a term, b_g , that represents the average rating for genre g:

$$Y = \mu + b_i + b_u + b_g \epsilon$$

method	RMSE
Movie + User Effects Model + Genre Effects Model	0.8649469

in adding the genre effect the model rmse holds at 0.865

Step Five: Regularization

Within this database there are many cases with movies where there are few reviews. Likewise, there are often cases where a user only has a few reviews. These coincidences can present an unreliable representation of the data as a whole. In order to constrain the total variability of the effect sizes we use regularization to penalize large estimates that come from small sample sizes. Using a sequence of numbers for lambda as a tuning parameter, we then use cross validation to select the lambda value with the best rmse.

method	RMSE
Regularized Movie Model	0.9438521
Regularized Movie + User Effect Model	0.8648170
Regularized Movie + User Effect + Genre Effect Model	0.8644509

Through the process of regularization three versions of the model were created. Each successive model was obtained by adding a new feature to the previous model. Once the third model was run an rmse of 0.864 was obtained.

Final Results

The table below shows our results through all explored versions of our model.

method	RMSE
Average	1.0612018
Movie Effect Model	0.9439087
Movie + User Effects Model	0.8653488
Movie + User Effects Model + Genre Effects Model	0.8649469
Regularized Movie Model	0.9438521
Regularized Movie + User Effect Model	0.8648170
Regularized Movie + User Effect + Genre Effect Model	0.8644509

The final model listed, 'Regularized Movie + User Effect + Genre Effect Model' created the best results for our RMSE.

Conclusion

In this project we conducted data analysis to gain a preliminary understanding and visualization of the MovieLens data set. The data itself was partitioned into training (edx) and test (validation) sets to help provide a most effective working model that wouldn't be overtrained. After training several models and building upon the results we then utitlized regularization to enhance the model through cross validation. In the final model, the version of 'Regularized Movie + User Effect + Genre Effect Model' obtained our best rmse result of 0.864. There certainly are other methods which could be examined to lower the rmse even further: year of release data could be considered and other forms of modeling such as matrix factorization could be utilized. However, as the expectation of the course was to obtain an RMSE < 0.86490 we will stand with the result of 0.8644 obtained in our model since it has achieved the highest expectation level.