MovieLens Project Report

Hannah Puisto

2023-12-05

Introduction

In this project, a movie recommendation system will be created to predict the ratings of movies for given users from the MovieLens 10M dataset. We will do this by training a linear model to predict movie ratings and calculate the Root Mean Square Error (RMSE) of the predicted ratings versus the actual ratings. The provided data is a larger dataset from which the *movielens* subset included in the **dslabs** package that we used for exercises in the previous course for machine learning.

The given script splits this larger dataset into a training dataset `edx` and a test dataset `final_holdout_test` 90% and 10%, respectively. We will use the training set in the construction of our model and will only use the test dataset to validate our predictions against.

Due to the large size of the dataset, certain machine learning algorithms or data wrangling methods would require too much memory to run on a normal laptop machine in a timely manner. We will create our recommendation system using a linear model as described in the *Methods and Analysis* section of this report.

Data Summary

The MovieLens 10M dataset contains 10 million ratings of over 10,000 movies by more than 72,000 users (ref: https://grouplens.org/datasets/movielens/10m/). The given training dataset `edx` contains 9,000,055 records while the test dataset `final_holdout_test` contains 999,999 records. Both the training and test datasets contain the same 6 feature columns:

```
names(edx)
```

```
## [1] "userId"    "movieId"   "rating"    "timestamp" "title"     "genres"
```

The number of ratings for each movie varies greatly, with *Pulp Fiction* being the most rated movie and over 100 titles rated once. The ratings range from 0.5 to 5 in increments of 0.5. There are no ratings of 0 for any movie.

```
# Most rated films
edx %>% group_by(title) %>%
  summarize(n_ratings = n()) %>%
  arrange(desc(n_ratings))
```

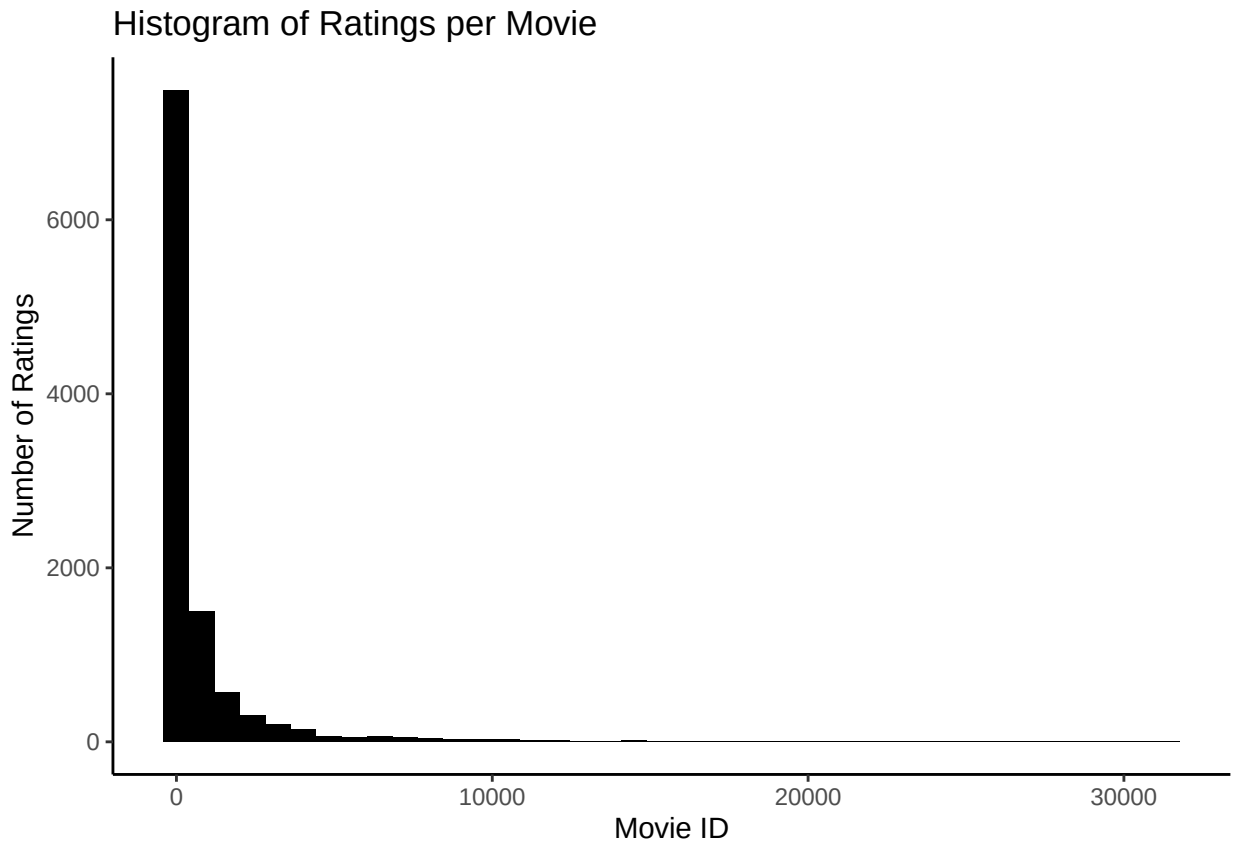
```
## # A tibble: 10,676 x 2
##    title                                                     n_ratings
##    <chr>                                                         <int>
##  1 Pulp Fiction (1994)                                           31362
```

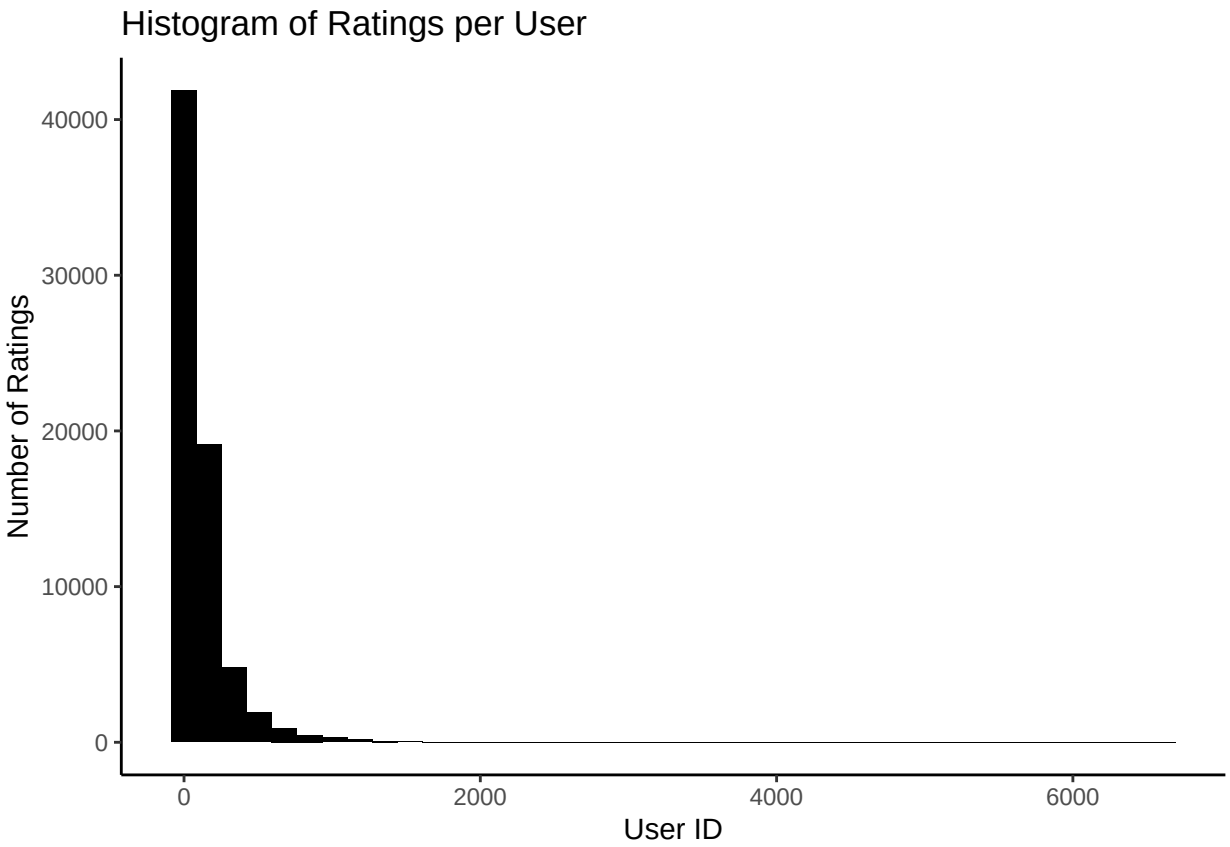
```
##  2 Forrest Gump (1994)                                                31079
##  3 Silence of the Lambs, The (1991)                                   30382
##  4 Jurassic Park (1993)                                               29360
##  5 Shawshank Redemption, The (1994)                                   28015
##  6 Braveheart (1995)                                                  26212
##  7 Fugitive, The (1993)                                               25998
##  8 Terminator 2: Judgment Day (1991)                                  25984
##  9 Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)       25672
## 10 Apollo 13 (1995)                                                   24284
## # i 10,666 more rows
```

```
# Number of movies rated once
edx %>% group_by(title) %>%
  summarize(n_ratings = n()) %>%
  filter(n_ratings==1) %>%
  count() %>% pull()
```

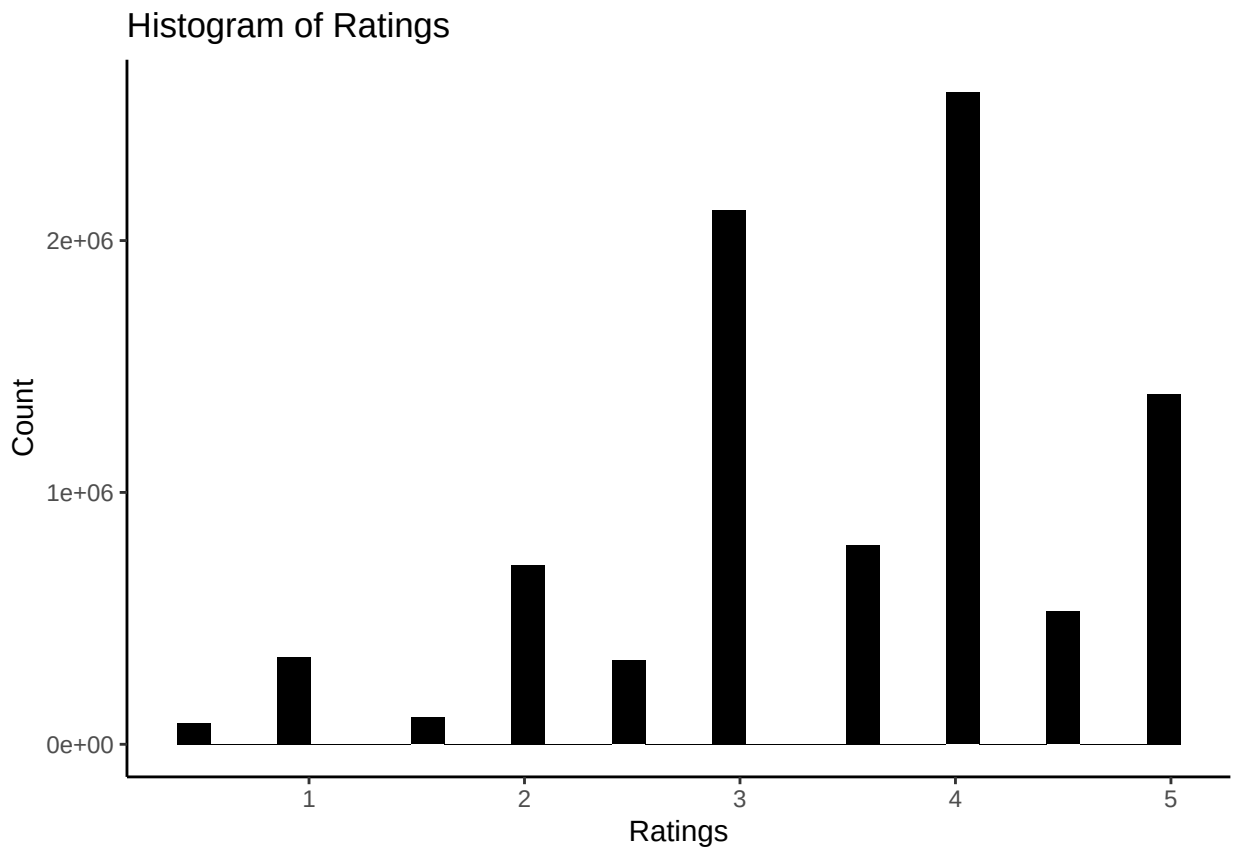
```
## [1] 126
```

Some movies were rated more than others and some users rated more movies than others. This also means that not every movie was rated by every user.





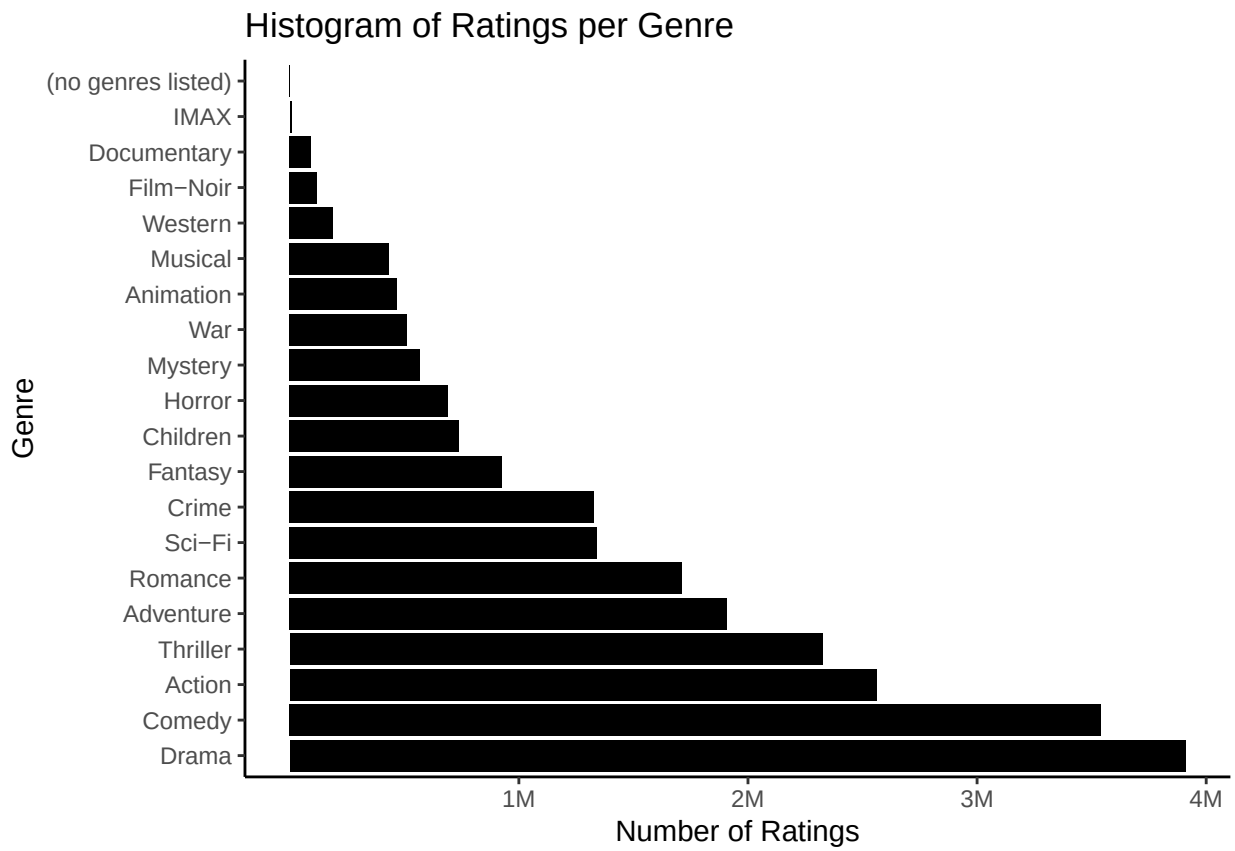
Users were also more likely to provide integer ratings over half-ratings.



The genre variable for a given movie lists all genres the movie fits into. There are twenty distinct genres a movie can be classified with:

```
##  [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)"
```

After breaking out the genres listed for movies in our training dataset (by creating a new row for the movie rating in the dataset for each genre listed), we see that some genres were rated more often than others.



Methods and Analysis

The simplest model is to use the average rating across all users and movies for our predicted ratings. This model uses the equation

$$Y_{u,m} = \mu, \tag{1}$$

where $Y_{u,m}$ is the predicted rating of user u and movie m, and μ is the average rating across all entries, which is 3.5124652.

```
mu <- mean(edx$rating)
RMSE(final_holdout_test$rating, mu)
```

```
## [1] 1.061202
```

To improve our model, we need to account for the rating differences between movies. This bias term, b_m, allows us to consider that some movies are liked or hated more than others. Our new model will be based off the average rating by movie as

$$Y_{u,m} = \mu + b_m. \tag{2}$$

```r
# Create a movie bias term, b_m
b_m <- edx %>%
  group_by(movieId) %>%
  summarize(b_m = mean(rating - mu))

# Predict unknown ratings for each movie on test dataset with mu and b_m
predicted_ratings <- final_holdout_test %>%
  left_join(b_m, by='movieId') %>%
  mutate(pred = mu + b_m) %>%
  pull(pred)

# Calculate RMSE of movie bias effect
RMSE(final_holdout_test$rating, predicted_ratings)
```

```
## [1] 0.9439087
```

Adding in a movie bias improved our predictions, but we can still do better. We now similarly introduce a user bias term, b_u, to account for rating differences between users who gave overall high or low ratings for all movies. The updated model considers both the movie and user effects as

$$Y_{u,m} = \mu + b_m + b_u. \tag{3}$$

```r
# Create user bias term, b_u
b_u <- edx %>%
  left_join(b_m, by='movieId') %>%
  group_by(userId) %>%
  summarize(b_u = mean(rating - mu - b_m))

# Predict new ratings for each movie on test dataset using both movie and user biases
predicted_ratings <- final_holdout_test %>%
  left_join(b_m, by='movieId') %>%
  left_join(b_u, by='userId') %>%
  mutate(pred = mu + b_m + b_u) %>%
  pull(pred)

# Calculate RMSE of movie and user biases effect
RMSE(final_holdout_test$rating, predicted_ratings)
```

```
## [1] 0.8653488
```

One more way to improve our predictions is to account for the genre bias in a similar way we did for both the movie and user biases. This new genre bias term, b_g, will be applied to our model as

$$Y_{u,m} = \mu + b_m + b_u + b_g. \tag{4}$$

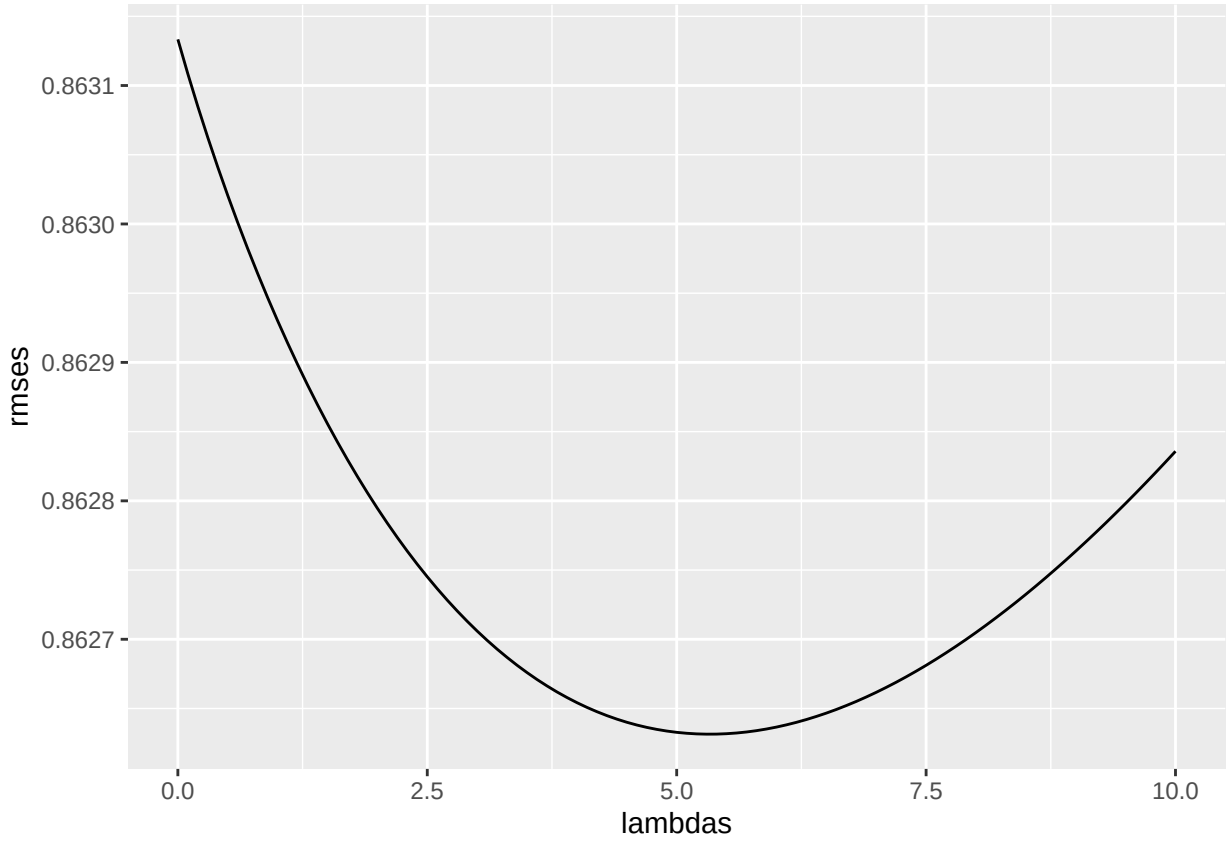
To account for individual ratings by genre, we created an extended version of the training dataset. As mentioned in the *Data Summary* section of this report, the original training dataset has multiple genres listed for each movie. We created a new dataset that creates an individual row for the rating for each unique genre listed in the original dataset. We will use this extended version of the training dataset to apply our genre bias.

```r
# Create genre bias term, b_g, on extended version of edx
b_g <- edx_genres %>%
  left_join(b_m, by='movieId') %>%
  left_join(b_u, by='userId') %>%
  group_by(genres) %>%
  summarize(b_g = mean(rating - mu - b_m - b_u))

# Predict new ratings for each movie on test dataset using both movie and user biases
predicted_ratings <- test_genres %>%
  left_join(b_m, by='movieId') %>%
  left_join(b_u, by='userId') %>%
  left_join(b_g, by='genres') %>%
  mutate(pred = mu + b_m + b_u + b_g) %>%
  pull(pred)

# Calculate RMSE of movie, user, and genre biases effect
RMSE(test_genres$rating, predicted_ratings)
```

```
## [1] 0.8631334
```

The last step of our model is to use regularization to penalize estimates of small samples. As mentioned, some movies have fewer ratings than others, meaning their sample size will be smaller. When the sample size is large, the estimate is more stable, so movies that are reviewed a lot will be better estimated. The regularization parameter, λ, will be applied to our movie, user, and genre bias effects to reduce large anomalies in the ratings across movies, users, and genres. We first need to find the best λ from a sequence for our data:

```r
ggplot(data.frame(lambdas, rmses), aes(lambdas, rmses)) + geom_line()
```



Now we see that the minimizing λ term is

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

```
## [1] 5.3
```

Results

From our analysis above, we chose the final model as

```
# The final linear model with the minimizing lambda
lam <- lambdas[which.min(rmses)]

# Compute final regularized movie bias term
b_m <- edx %>%
  group_by(movieId) %>%
  summarize(b_m = sum(rating - mu)/(n()+lam))

# Compute final regularized user bias term
b_u <- edx %>%
  left_join(b_m, by="movieId") %>%
  group_by(userId) %>%
  summarize(b_u = sum(rating - b_m - mu)/(n()+lam))

# Compute final regularized genre bias term
```

```
b_g <- edx_genres %>%
  left_join(b_m, by='movieId') %>%
  left_join(b_u, by='userId') %>%
  group_by(genres) %>%
  summarize(b_g = sum(rating - mu - b_m - b_u)/(n()+lam))

# Compute final predictions on final_holdout_test set based on the above biases
predicted_ratings <- test_genres %>%
  left_join(b_m, by='movieId') %>%
  left_join(b_u, by='userId') %>%
  left_join(b_g, by='genres') %>%
  mutate(pred = mu + b_m + b_u + b_g) %>%
  pull(pred)

# Output final RMSE of these predictions
RMSE(test_genres$rating, predicted_ratings)
```

```
## [1] 0.8626315
```

With each bias effect we added to our model, we made incremental improvements to the RMSE, with the movie effect having the biggest decrease in RMSE. Our final RMSE was 0.8626315. The RMSE of our final model is considered very good by the course rubric.

Conclusion

Our goal for this project was to predict movie ratings from the MovieLens 10M dataset. To do this, we considered the effect movies, users, and genres had on the rating. Regularization allowed us to account for extreme ratings and to penalize estimates of small samples.

Given the large dataset, the simplicity of our linear model and calculating the least square estimates manually allowed us to predict movie ratings without a serious toll on the computer's memory.

Future work for this dataset would be interesting if there was demographic information included. It would be interesting to see if the age, gender, or location of the user played a factor in ratings. It would also be interesting to see if there were overlaps of actors and directors in the movies and whether that had an impact on the quantity and value of the ratings.