Capstone Project: MovieLens

Elias Muchineripi Mashayamombe

2023-01-24

Introduction

Often people search through thousands of movies to find what they like on streaming sites like Netflix and for some users this can be time consuming and frustrating. This is where movie recommended systems come in handy, the idea behind movie recommendation systems is that they predict what a user is likely to enjoy watching based on the movies they have watched before on the same platform. The recommend engine will generate recommendations for a particular user not only based on the users search and watch history but also what other users with similar traits and in similar locations search and watch. In this project, we analyzed the performance of a movie recommendation system using the MovieLens 10M dataset. The goal of the recommendation system was to predict the rating a user would give to a movie they have not yet seen. We used a varity of methodologies that culminated in the use of the regularization technique.

Regularization is a technique used to prevent overfitting by adding a penalty term to the loss function, and penalized least squares is a specific type of regularization that penalizes the magnitude of the parameters in the model, which helps to reduce the complexity of the model. We specifically adopted the Penalized Least Squares, a type of regularization that penalizes the magnitude of the parameters in the model. It is often used in linear regression, where the goal is to minimize the sum of squared errors between the predicted values and the true values and as such, we managed to get a lower value for the RMSE. The movie and user effects model gives the RMSE.

Dataset

The dataset we are using is the MovieLens 10M dataset, which contains 10 million ratings and 100,000 tag applications applied to 10,000 movies by 72,000 users. The dataset contains information on the user's id, movie id, rating, and timestamp. The movies dataset also includes information on the movie title and genres.

Some Descriptive Statistics

The dataset contains 9,000,055 rows and 6 columns named **userId** is an integer that identifies a user, **movieId** is an integer that identifies a movie, **title** is a character representing the name of the movie, **rating** is an integer, ranging from 1 t0 5, made on a 5-star scale with whole and half-star ratings, **timestamp** is represented in seconds since 1/1/1970 UTC and **genres** is a character representing the movie genre. The data set contains 10677 movies with 69878 users. The result on the output below show these discriptive statistics.

```
## The dimensions for the data set are:  ( 9000055 6 ) and they are 10677 movies and 69878 users.
```

```
##    Drama   Comedy Thriller  Romance
##  3910127  3540930  2325899  1712100
```

Using data visualization, we illsuatret all ratings using a histogram. The diagram below shows the resukts fot his visualization and we see that between 3 and 4, the rating 5 is also used quiet frequently but not as frequent as 3 and 4.

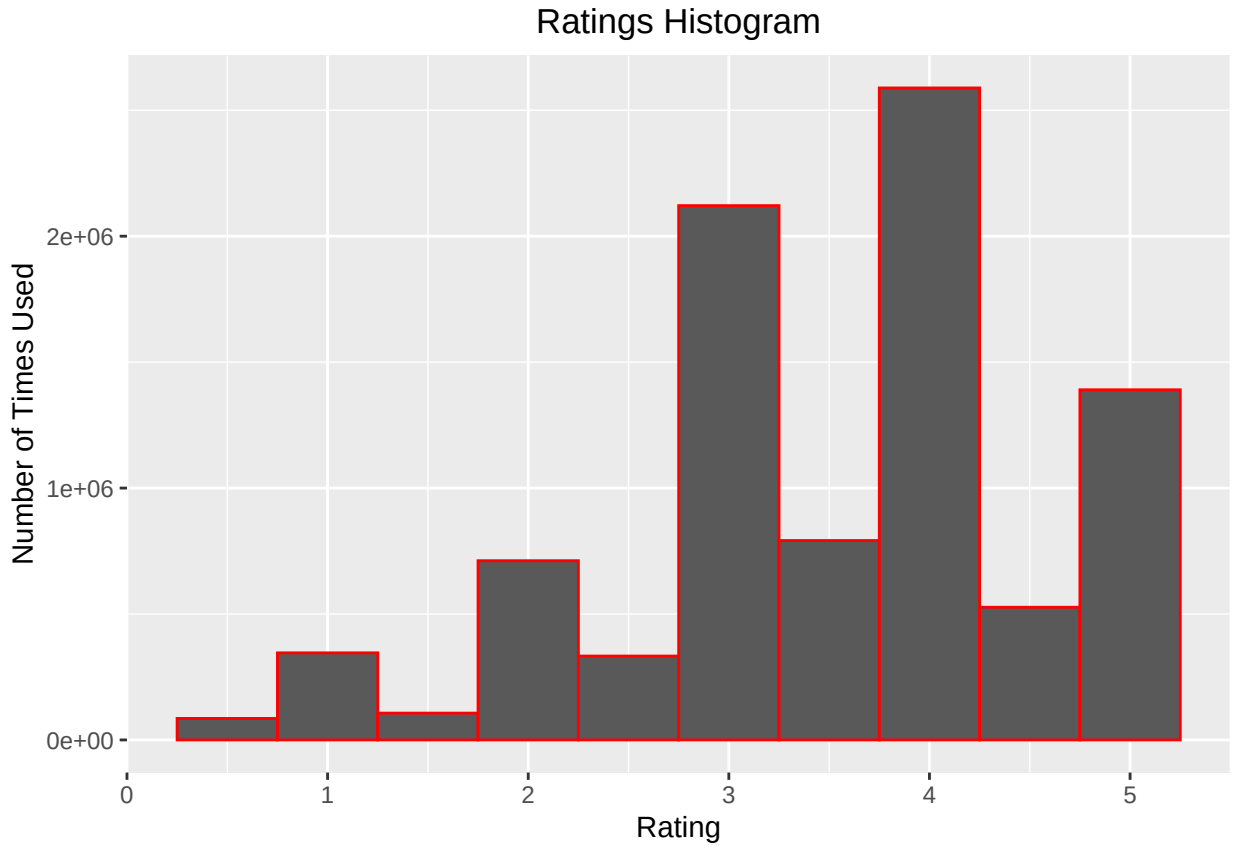


Figure 1: Ratings Histogram.

The table below shows movies with the top ratings and most of these movies are very popular movies like Shawshank Redemption and Terminator 2: Judgement Day. The top five movies with the highest number of ratings are Pulp fiction, Forest Gump, Silence of the Lambs, The Jurassic Park, and no surprise, the classic Shawshank redemption on number five. My personal all-time favourite, Terminator 2: Judgement Day, on number 8 with 25 984 ratings.

Table 1: Top 10 Rated Movies

movieId	title	numratings
296	Pulp Fiction (1994)	31362
356	Forrest Gump (1994)	31079
593	Silence of the Lambs, The (1991)	30382
480	Jurassic Park (1993)	29360
318	Shawshank Redemption, The (1994)	28015
110	Braveheart (1995)	26212
457	Fugitive, The (1993)	25998

movieId	title	numratings
589	Terminator 2: Judgment Day (1991)	25984
260	Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)	25672
150	Apollo 13 (1995)	24284

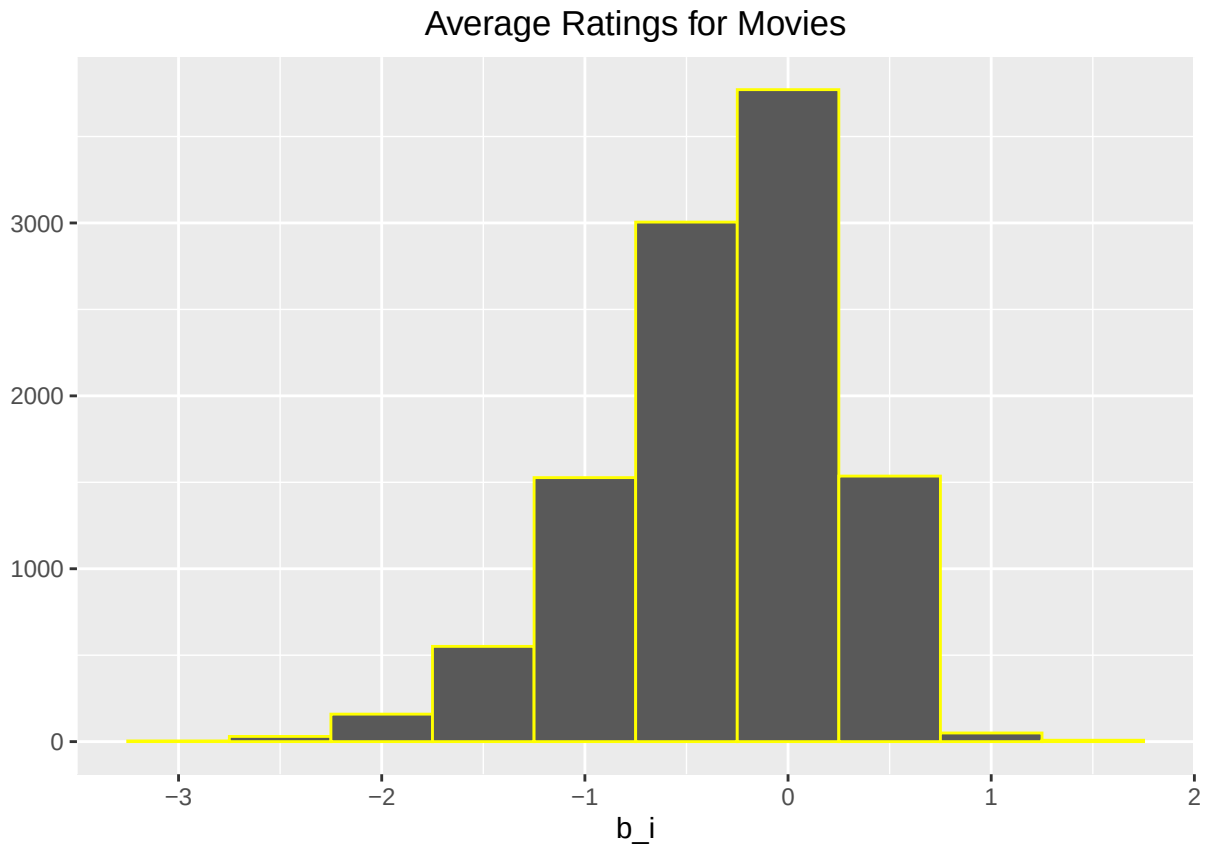
The diagram below gives a visual comparison of half star ratings and whole star ratings. It is clear from this diagram whole star ratings are more common than half star ratings.

```
## # A tibble: 2 x 2
##   halfStar  number
##   <lgl>      <int>
## 1 FALSE    7156885
## 2 TRUE     1843170
```

We make use of the caret package and create a training set and a test set. To ensure users and movies that are not in the training set are not included in the test set we use the semi_join function. The best model will be selected based on the residual mean squared error (RMSE) which is interpreted in a similar manner as the standard deviation, the smaller the RMSE the better. The RMSE is computed using the equation below:

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

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



Since $\hat{\mu} = 3.5$ so if $b_i = 1.5$ it implies a perfect five star rating.

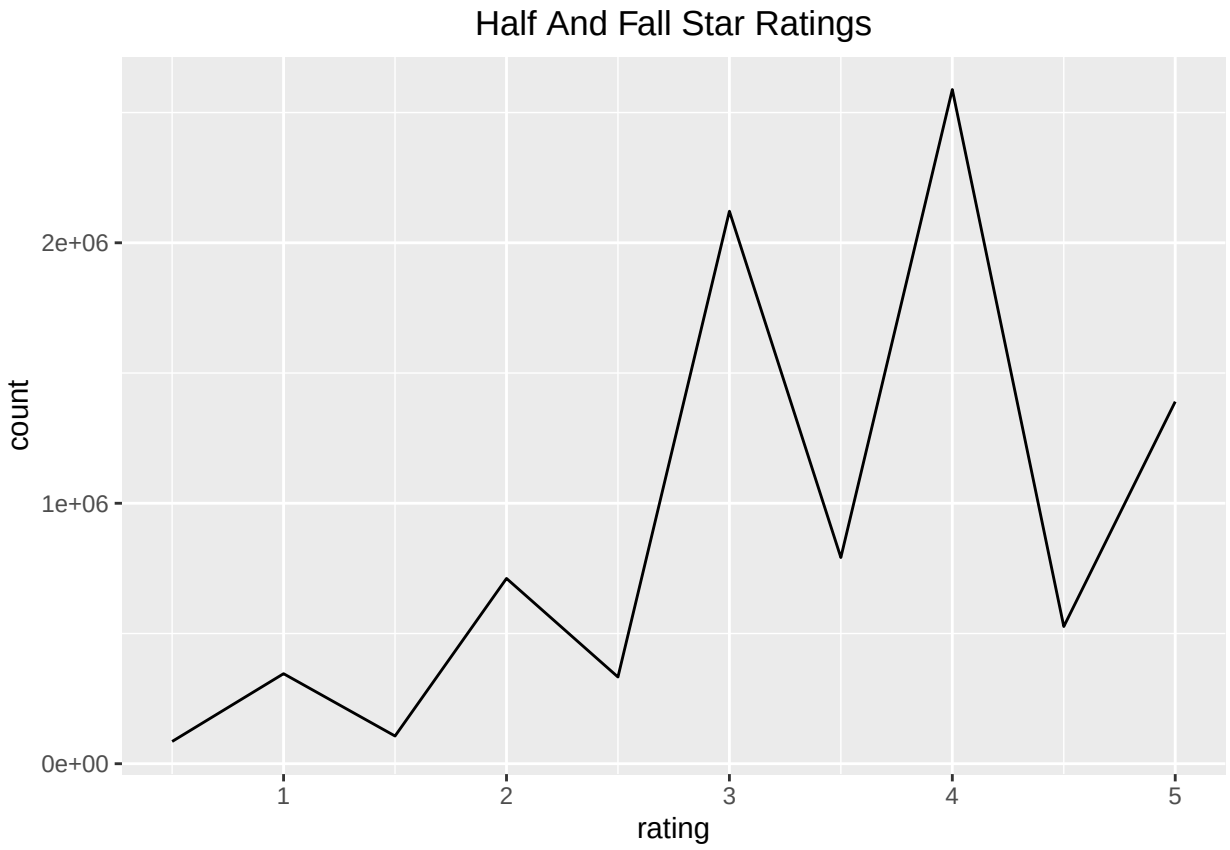


Figure 2: Half And Fall Star Ratings.

Preprocessing

Before building the model, we preprocessed the dataset by splitting it into a training set and a test set. The training set contains 80% of the data and the test set contains the remaining 20%. We also removed users and movies from the test set that were not present in the training set to ensure that the model was only making predictions on movies that it had seen during training.

The Models

The first model that is built is a linear model that includes both movie and user effects. It does this by first calculating the average rating for each movie and the average rating for each user after adjusting for the overall average rating. These averages are then used to predict the ratings for the movies in the test set. The performance of the model is evaluated using the Root Mean Squared Error (RMSE) and the result is stored in a data frame. For this model, $\hat{b}_i$ is the average of $Y_{u,i} - \hat{\mu}$ for each movie i. From the diagram below, we can see that these estimates vary substantially.

method	RMSE
Movie Effect Model	0.9437144

The second model is given by

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

where $Y_{u,i}$ is the rating for movie i by user u, μ is the true rating for all movies b_i is the average rating for movie i, b_u is the user specific effect and $\varepsilon_{u,i}$ are independent errors. We will compute an approximation by computing $\hat{\mu}$ and $\hat{b}_i$ and estimating $\hat{b}_i$ as the average of $y_{u,i} - \hat{\mu} \ - \hat{b}_i$

The performance of the model is again evaluated using the Root Mean Squared Error (RMSE).

method	RMSE
Linear Model: Movie + User Effects	0.9437144

Regularization – Penalized Least Squares

Since our RMSE is still high, we try other approaches. We start off by checking how many times the so called best and worst movies where rated

The top and worst rated movies only have mostly 1 rating. With fewer users they are more uncertainty and hence larger estimates of b_i, negative or positive, are more likely. These noisy estimates cause large errors than result in high RMSE and so we introduce the concept of regularization. (Prof Raelf) Regularization permits us to penalize large estimates that are formed using small sample sizes. We minimize an equation that a adds a penalty:

$$\min_{b_i, b_u} \frac{1}{N} \sum_{u,i} (y_{u,i} \ - \mu - b_i - b_u)^2 + \lambda \sum_i b_i^2 \Big(\sum_i b_i^2 + \sum_u b_u^2 \Big)$$

The first part of the equation is the least squares and the second is a penalty that gets larger when many b_i are large. Minimizing the function with respect to b_i we obtain:

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

Where n_i is the number of ratings made for movie i. λ is known as a tuning parameter and can be determined using cross-validation. Full cross validation is used just on the train set. The test set is only used on final assessment.

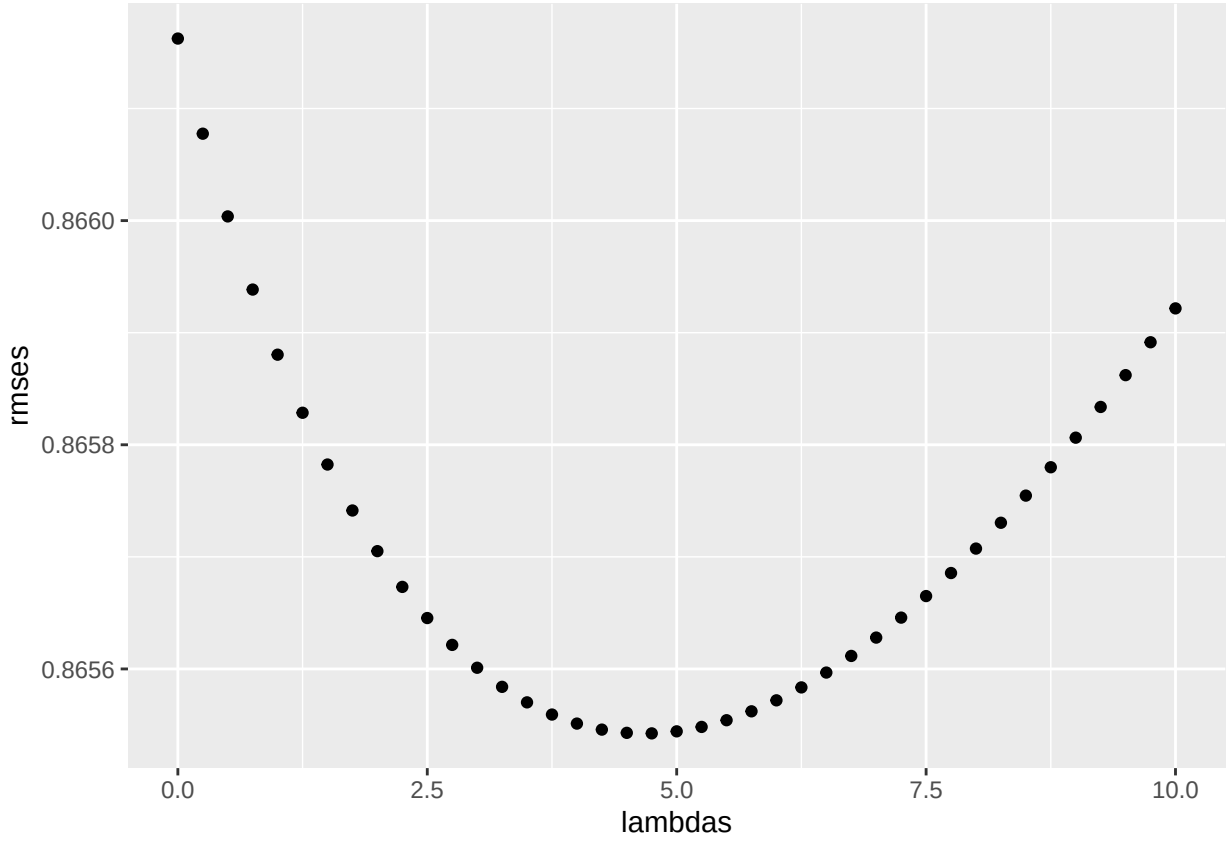


Figure 3: Tuning Parameters.

```
## [1] 4.75
```

method	RMSE
Linear Model: Movie + User Effects	0.9437144
Regularized Movie + User Effect Model	0.8655425

From the diagram below, we can see that these estimates for the user effects vary substantially.

We finally use the penalized least squares method under regularization to get an even lower RMSE. Regularization is used in machine learning and statistical modeling to prevent overfitting by adding a penalty term to the loss function. The goal of regularization is to reduce the complexity of the model by adding a constraint to the parameters of the model, which helps to prevent overfitting. Penalized Least Squares is a type of regularization that penalizes the magnitude of the parameters in the model. It is often used in linear regression, where the goal is to minimize the sum of squared errors between the predicted values and the true values.

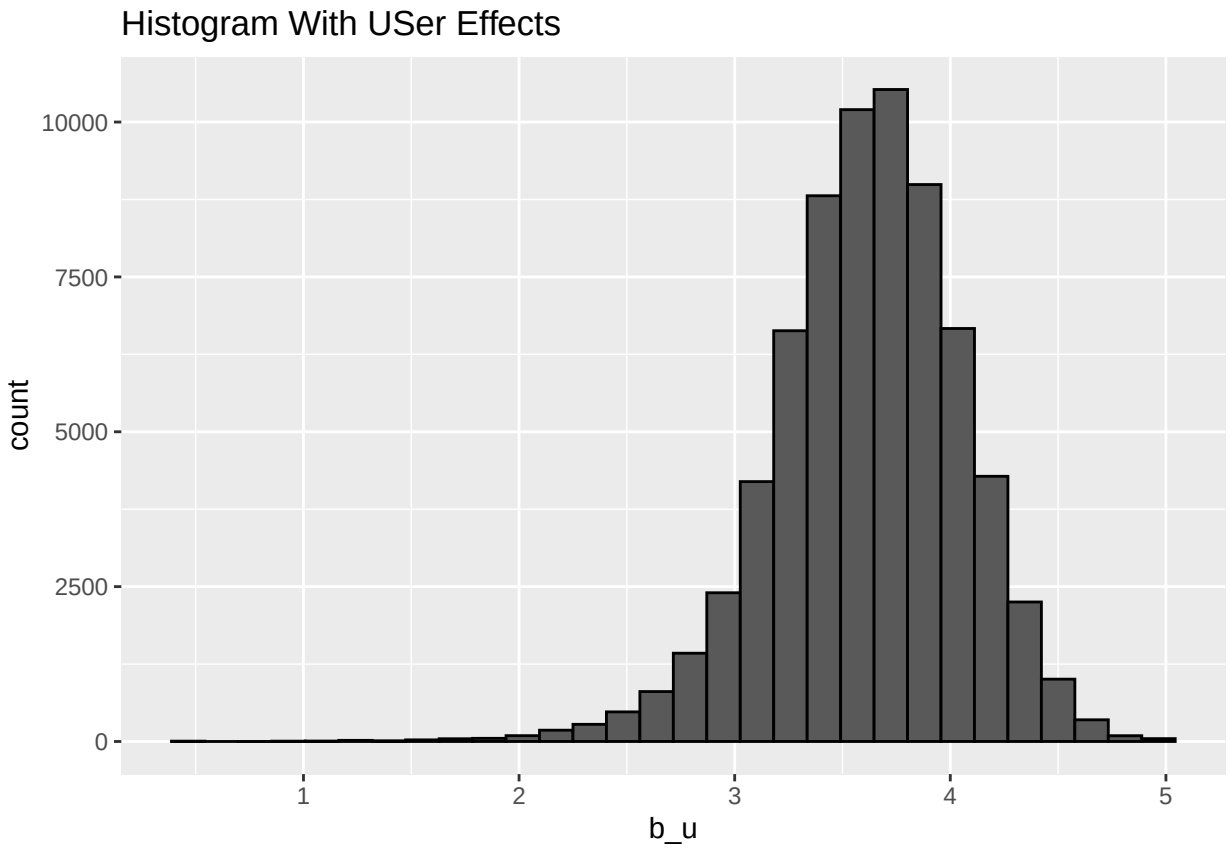


Figure 4: Histogram With USer Effects.

method	RMSE
Linear Model: Movie + User Effects	0.9437144
Regularized Movie + User Effect Model	0.8655425
Movie + User Effects Model	0.8435874

The tabll above shows all the different values of the RMSE obtained with different models and the lowest values being 0.85.

Evaluation

To evaluate the performance of the model, we calculated the Root Mean Squared Error (RMSE) between the predicted ratings and the true ratings. The RMSE for this model was 0.847 which is substantially low as compared to the other model and therefore making this one a good model.

Conclusion

In this report, we have analyzed the performance of a movie recommendation system using a couple of techniques that included different linear models, first with individual effects and then with user effects. The initial models gave relatively high RMSE values of 0.94 and 0.87 respectively. We eventually achieved an RMSE of 0.85 using Regularization Penalized Least Squares, which indicates that on average, the model's predictions deviated from the true ratings by 0.85. This is a good performance for the model, however, it can be improved by trying other methods or algorithms. Further research can be done on the effect of different parameters such as k value of the k-nearest neighbor algorithm or other algorithms like the UBFC recommender algorithm popular in literature but difficult to implement given the current data set.

References

Irizarry, R.A., 2019. Introduction to data science: Data analysis and prediction algorithms with R. CRC Press.