MovieLens Recommendation System Project

HarvardX - PH125.9x: Data Science: Capstone

Ringgold P. Atienza

July 25, 2022

1. Introduction

The recommendation system is one of the most popular machine learning projects to learn. It is widely used in many fields such as education, medicine, movies, music, e-commerce, TV programming, social networking, and tourism to fulfill user needs [1]. Recommendation systems are software tools and techniques that provide suggestions for items most likely of interest to a particular user [2]. In the context of business, a recommendation system ensures that users are always provided with the best possible experience. A good user experience leads to engagement or profit. Thus, recommendation systems derive tremendous value, and companies are always looking to improve their efficiency in this area.

In 2006, the company Netflix held the $1M Netflix Prize challenge, searching for a recommendation system algorithm that could increase their current efficiency by 10% [3]. It was a prominent analytics competition back then not only because of the $1M prize but also because it provided 100 million data of movie ratings, which invigorated many analysts back then.

Nowadays, big data has become very accessible due to the internet. Datasets such as MovieLens, which provides big data for movie ratings, are available for those who want to analyze the data online [4]. For this project, I use the 10 million version of the MovieLens dataset to create a movie recommendation system. The Movielens 10M dataset tags ratings applied to 10,000 movies by 72,000 users, released in 2009. This project aims to create a recommendation system algorithm with a predicted rating that will result in root means square error (RMSE) < 0.86490 in the validation set.

Due to the large dataset, there were limitations regarding the technique I used in this project. A more powerful algorithm such as matrix factorization and neighborhood models were impractical as they required immense computing power that a personal computer cannot provide. While those complex algorithms are terrific, a simpler algorithm that can do the job is always preferred.

2. Methods and Analysis

2.1. Preparing the Data

2.1.1. Training Set and Validation Set

In machine learning, the best practice is to split the data into three independent sets: a training set, a test set, and a validation set. The training dataset is used to fit the model. The test and validation sets are a sample of data taken from the training dataset to estimate the model's performance while tuning the model's hyperparameter [5]. A test dataset is different from the validation dataset as it provides an unbiased evaluation of the model fit on the training dataset. This process is important to cross-validate and refines the final model without the risk of over-fitting. Meanwhile, the validation dataset provides an

unbiased evaluation of the performance of the final tuned model. The validation set is not used for training, developing, or selecting algorithms but only to evaluate the final model. Thus, it may not be used to test the RMSE of different algorithms during model development.

To start, the MovieLens 10M dataset was downloaded from this website: "https://grouplens.org/datasets/movielens/10m/". Then the MovieLens 10M dataset was partitioned into train-validation split, as shown in Figure 1. The training dataset will be used to evaluate the final model against the validation dataset. Further, the final training dataset is partitioned into a train-test split. These two resulting datasets, training and test set, were then used to find the best algorithms for the recommendation system.

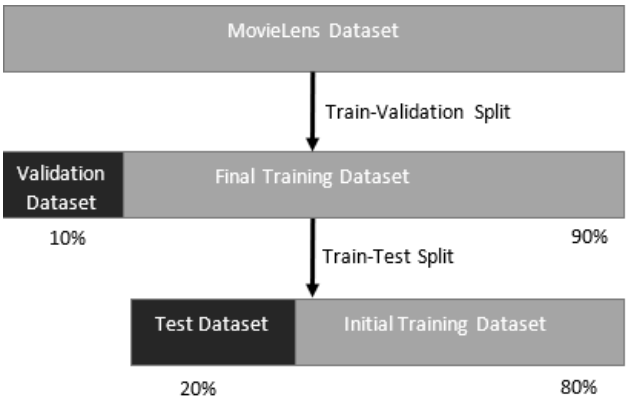

Figure 1: Partitioning of Train, Test, and Validation Datasets

2.1.2. Mutating Dataset

The original data only provides the user ID, movie ID, rating, timestamp of the rating, movie title, and movie genres. Mutations were made to the dataset to flesh out other relevant variables such as the movie's year of release, age of the movie upon the rating, user frequency of rating, and movie frequency of being rated. The final dataset for training includes the following variables:

```
##  [1] "userId"     "movieId"    "rating"     "timestamp"  "title"
##  [6] "movieYear"  "genres"     "reviewDate" "reviewYear" "movieAge"
## [11] "userFreq"   "movieFreq"
```

2.2. Data Visualization

2.2.1. Ratings

The ratings of the movies start with the lowest value of 0.5 up to the highest value of 5.0. The distribution of the total ratings is shown in Figure 2. The distribution tells us that most of the raters rate the movies at 4.0, and very few rate movies at 0.5. It also shows that raters tend to rate movies the whole stars as compared to half stars.

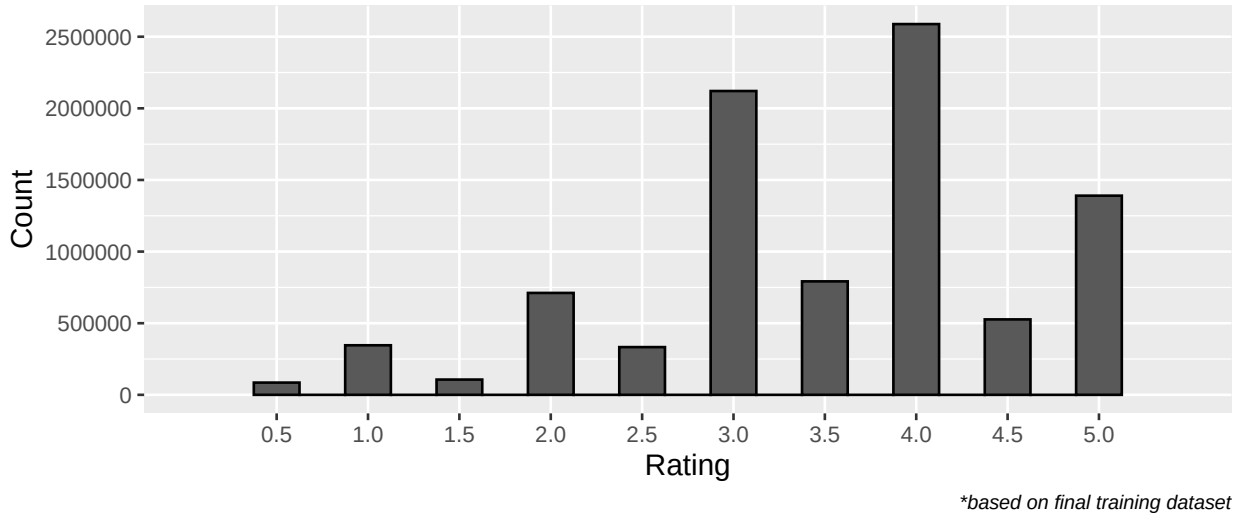

Figure 2. Total Ratings Distribution

2.2.2. Average Rating by MovieID

Most of the average ratings by MovieID are at 3.0 and 3.5, as shown in Figure 3. Also, average movie ratings are between 2.84 (10% percentile) and 3.85 (90% percentile). The result indicates there are very few movies rated averagely very low (less than 10% percentile) and very high (higher than 90 % percentile).

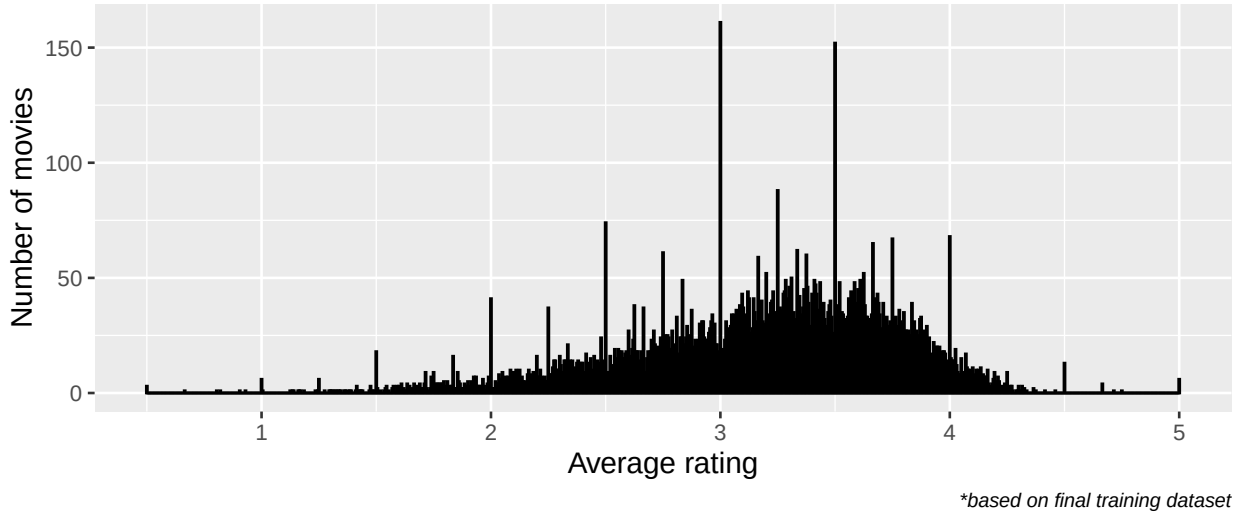

Figure 3. Distribution of Average Ratings by Movie

2.2.3. Average Rating by UserID

In Figure 4, most of the average ratings by UserID are at 4.0 and 3.5, which indicates that many users are inclined to rate movies at 4.0 and 3.5. Also, average movie ratings are between 3.08 (10% percentile) and 4.13 (90% percentile). The result indicates there are very few users rate very low (less than 10% percentile) and very high (higher than 90 % percentile).

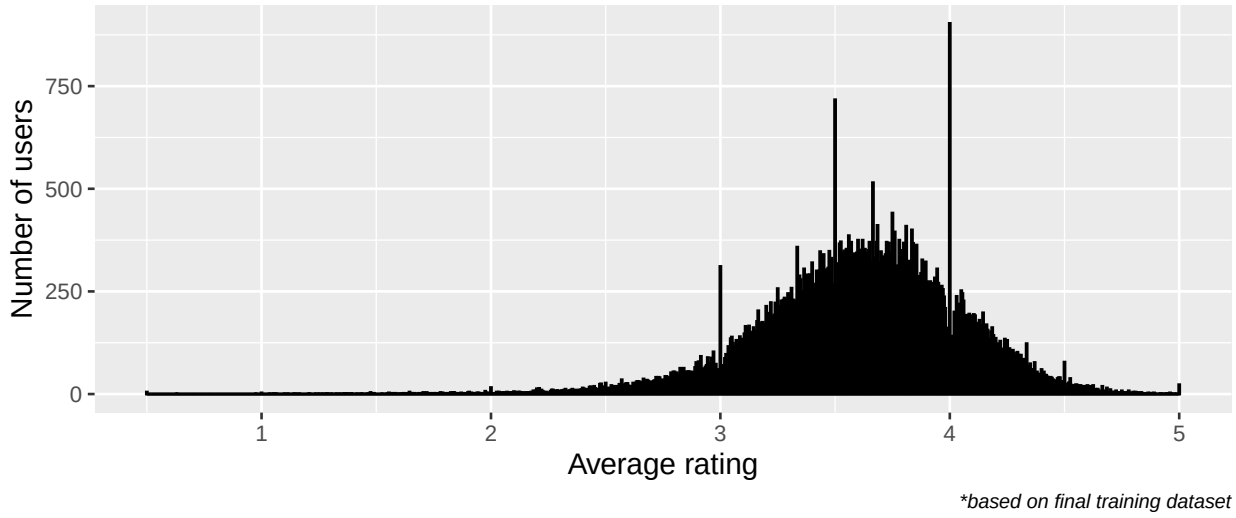

Figure 4. Distribution of Average Ratings by Users

2.2.4. Total Rated Movies by Year

In Figure 5, movies released on 1995 have the highest rating reviews while the fewest is 1917. The total rated movies are between 1973 (10% percentile) and 2002 (90% percentile). The result indicates that few movies were rated from 1917 to 1972 and 2002 to 2008.

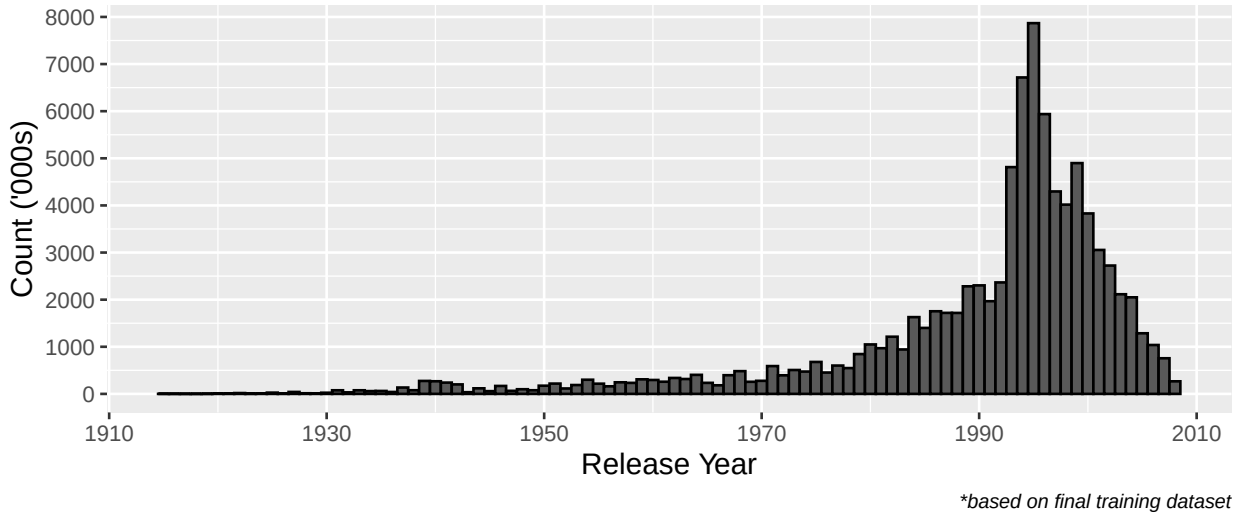

Figure 5. Distribution of Total Rated Movies Per Year

2.2.5. Total Rated Movies by Movie Age

Results in Figure 6 showed that users rate movies more frequently after a year of the movie's release. Users seldom rate movies released during its year (10% percentile) and over 30 years (90% percentile). There were movies rated before the release date because some users rate movies based on the pre-screening of the movie.

Figure 6. Distribution of Total Rated Movies Per Movie Age

2.2.6. Total Rated Movies by Genre

In the MovieLens dataset, most movies are assigned to more than one genre. With this, we have to separate these genres aggregated in one feature to generate the distribution of the most rated genre in the dataset. The result in Figure 7 shows that Drama has the most number of rated movies.

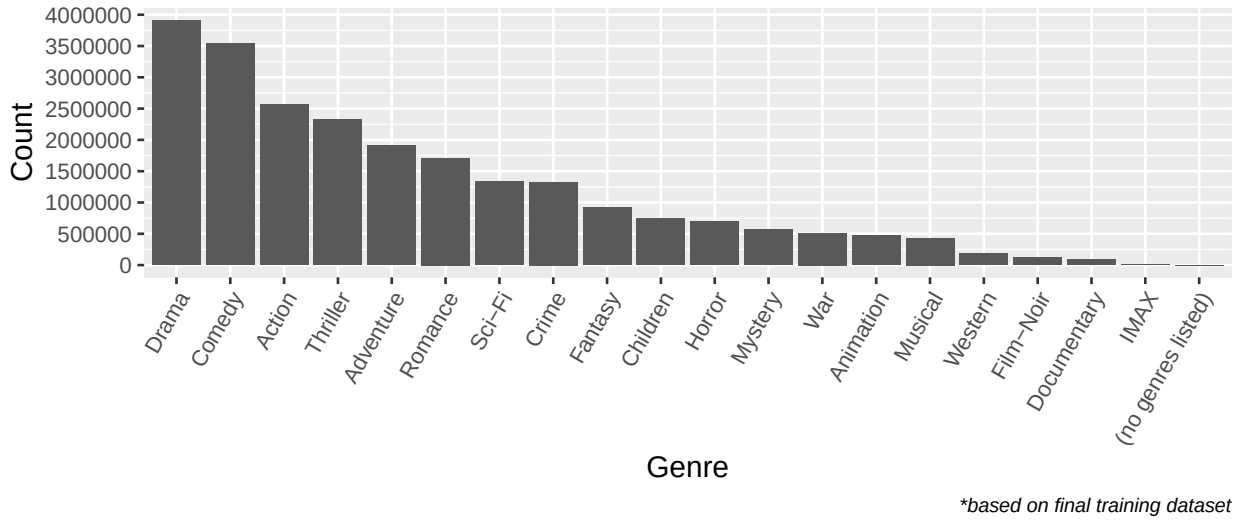

Figure 7. Distribution of Total Rated Movies Per Genre

2.3. Building the Model

2.3.1. Residual Mean Square Error (RMSE) as Loss Function

RMSE is one of the most widely used metrics for evaluating prediction models or algorithms in many fields [5]. In machine learning, we refer to those metrics that evaluate how well the model performs over the

training dataset as loss functions. RMSE is calculated by taking the square root of Mean Squared Errors (MSE):

$$RMSE = \sqrt{1/N \sum_{i=1}^{N} (\hat{x}_i - y_i)^2}$$

In general, creating a prediction model aims to build an algorithm that minimizes the loss so it is as close to 0 as possible [6]. This project aims to build an algorithm that results in an RMSE < 0.86490.

2.3.2. Stepwise Method for Least Square Estimate

This project utilizes the stepwise method using the least squares estimates and RMSE as an evaluation metric. The least squares method is about estimating parameters by minimizing the squared discrepancies [7]. The (1)movieID, (2)userID, (3)movie's year of release, (4)movie's age when rated, (5)genre, (6)frequency the movie is rated, and (7)frequency the user rated movies are the variables considered for testing between observed data

Stepwise method is a technique that utilizes an evaluation metric to select the variables to be used in the model. This technique helps us determine the best model by incrementally examining the variables for some explanatory power, then includes or excludes variables based on particular criteria [8]. The procedure starts with the simplest model as the baseline, using the rating means as a predictor. Subsequently, the method tests the variables individually to find the best predictor, significantly reducing the RMSE. The best predictor for that step will be added to the model and is considered the new baseline. The process is repeated until no single step can improve the model.

2.3.2.1. Baseline Rating

The baseline is the mean of the overall user's movie rating. The mean rating is used to predict all movies. Our model for this is:

$$Y = \mu + \epsilon$$

To run the model, the mean of the rating was computed:

```
mu <- mean(edxTrainSet$rating)
```

```
## The mean rating is = 3.512
```

The RMSE was also computed for the baseline:

Variable	RMSE	Difference
Baseline (mu)	1.059519	0

After setting the baseline rating (μ), the estimates movie effect (β_i), user effect (β_u), movie age effect (β_a), movie year affect (β_y), genre effect (β_g), user frequency effect (β_f), and movie frequency effect (β_m) are tested individually with the baseline. The model is then extended to accommodate the first effect:

$$Y = \mu + \beta + \epsilon$$

The table below shows the values of RMSE when we individually add the first effects in the model. The effect that shows the minimal loss, in this case the lowest RMSE, was chosen for the development of the model.

Variable	RMSE	Difference
Baseline (mu)	1.0595191	0.0000000
Baseline + Movie Effect (mu + b_i)	0.9431919	0.1163272
Baseline + User Effect (mu + b_u)	0.9782496	0.0812695
Baseline + Movie Age Effect (mu + b_a)	1.0506993	0.0088198
Baseline + Movie Year Effect (mu + b_y)	1.0484775	0.0110417
Baseline + Genres Effect (mu + b_g)	1.0172779	0.0422413
Baseline + User Frequency Effect (mu + b_f)	1.0403396	0.0191795
Baseline + Movie Frequency Effect (mu + b_m)	0.9678891	0.0916300

A graph is shown in Figure 8 for visual inspection of the RMSE values of the different effects. The baseline (rating average) with the movie effect ($\mu + \beta_i$) produced minimal loss compared to other variables. The baseline with the movie effect ($\mu + \beta_i$) was then chosen as the new baseline for the next stepwise model.

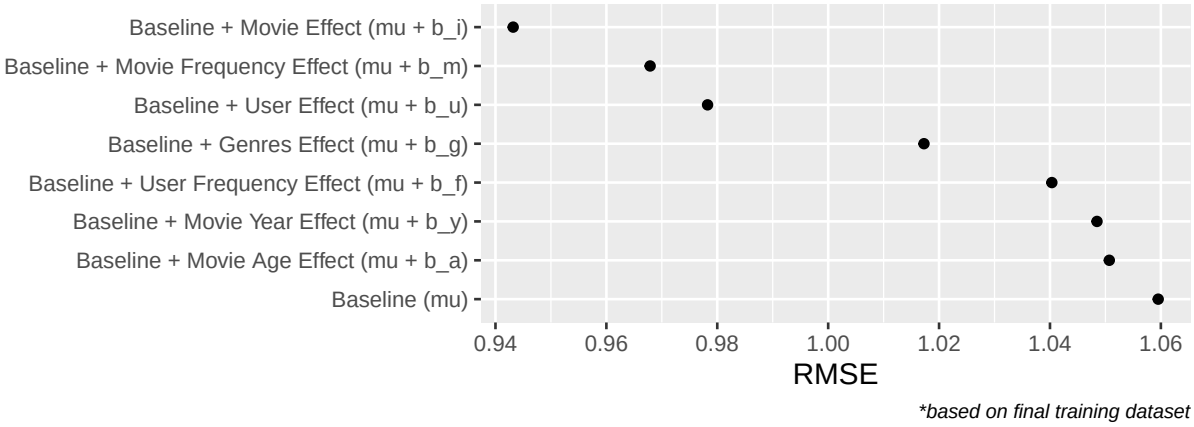

Figure 8. Stepwise RMSE (Baseline)

2.3.2.2. Movie Effect

The rating average + movie effect ($\mu + \beta_i$) became the baseline for the 2nd stepwise reiteration. The stepwise method then considered the user effect (β_u), movie age effect (β_a), movie year affect (β_y), genre effect (β_g), user frequency effect (β_f), and movie frequency effect (β_m). The result of the stepwise method is shown in the table below.

Stepwise	RMSE	Difference
Baseline + Movie	0.9431919	0.0000000
+ Movie Frequency	0.9431919	0.0000000
+ User	0.8657044	0.0774874
+ Genres	0.9431919	0.0000000
+ User Frequency	0.9303621	0.0128298
+ Movie Year	0.9431919	0.0000000

Stepwise	RMSE	Difference
+ Movie Age	0.9416793	0.0015126

As shown in Figure 9, the previous baseline with user effect ($\mu + \beta_i + \beta_u$) produced minimal loss compared to other variables. The baseline with the user effect ($\mu + \beta_i + \beta_u$) was then chosen as the new baseline for the next stepwise model.

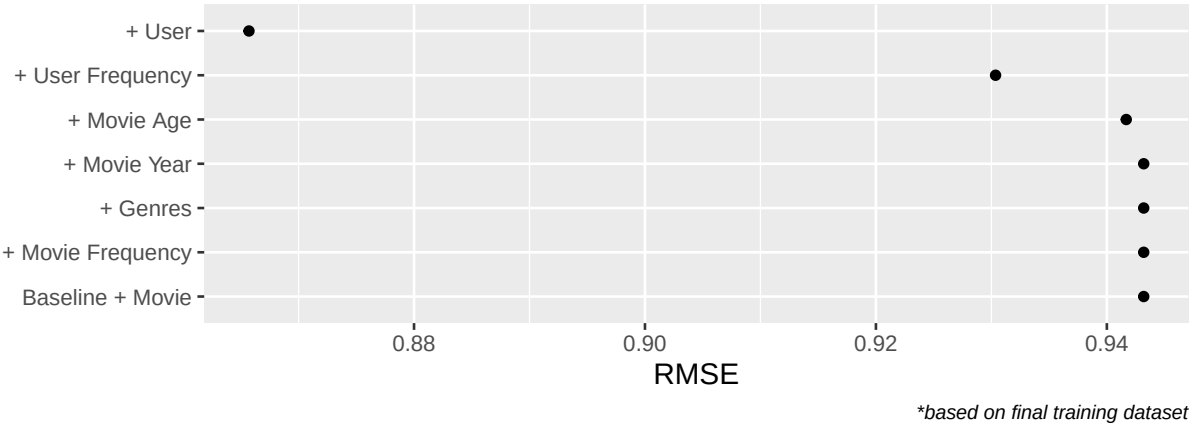

Figure 9. Stepwise RMSE (Movie Effect)

2.3.2.3. User Effect

The rating average + movie effect + user effect ($\mu + \beta_i + \beta_u$) became the baseline for the 3rd stepwise reiteration. The stepwise method then considered the movie age effect (β_a), movie year affect (β_y), genre effect (β_g), user frequency effect (β_f), and movie frequency effect (β_m). The result of the stepwise method is shown in the table below.

Stepwise	RMSE	Difference
Baseline + Movie + User	0.8657044	0.0000000
+ User Frequency	0.8657044	0.0000000
+ Movie Age	0.8651973	0.0005072
+ Movie Year	0.8653602	0.0003442
+ Genres	0.8653730	0.0003314
+ Movie Frequency	0.8648264	0.0008780

As shown in Figure 10, the previous baseline with movie frequency effect ($\mu+\beta_i+\beta_u+\beta_m$) produced minimal loss compared to other variables. The baseline with the movie frequency effect ($\mu + \beta_i + \beta_u + \beta_m$) was then chosen as the new baseline for the next stepwise model.

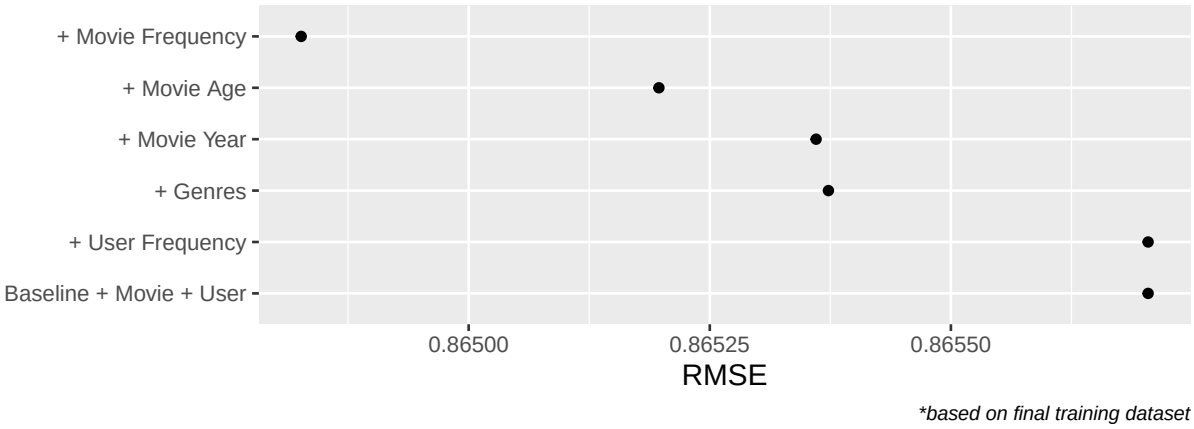

Figure 10. Stepwise RMSE (User Effect)

2.3.2.4. Movie Frequency Effect

The rating average + movie effect + user effect + movie frequency effect ($\mu + \beta_i + \beta_u + \beta_m$) became the baseline for the 4th stepwise reiteration. The stepwise method then considered the movie age effect (β_a), movie year affect (β_y), genre effect (β_g), and user frequency effect (β_f). The result of the stepwise method is shown in the table below.

Stepwise	RMSE	Difference
Baseline + Movie + User + Movie Frequency	0.8648264	0.0000000
+ Movie Age	0.8644310	0.0003955
+ Genres	0.8647961	0.0000303
+ Movie Year	0.8648201	0.0000063
+ User Frequency	0.8647510	0.0000754

As shown in Figure 11, the previous baseline with movie age effect ($\mu + \beta_i + \beta_u + \beta_m + \beta_a$) produced minimal loss compared to other variables. The baseline with the movie age effect ($\mu + \beta_i + \beta_u + \beta_m + \beta_a$) was then chosen as the new baseline for the next stepwise model.

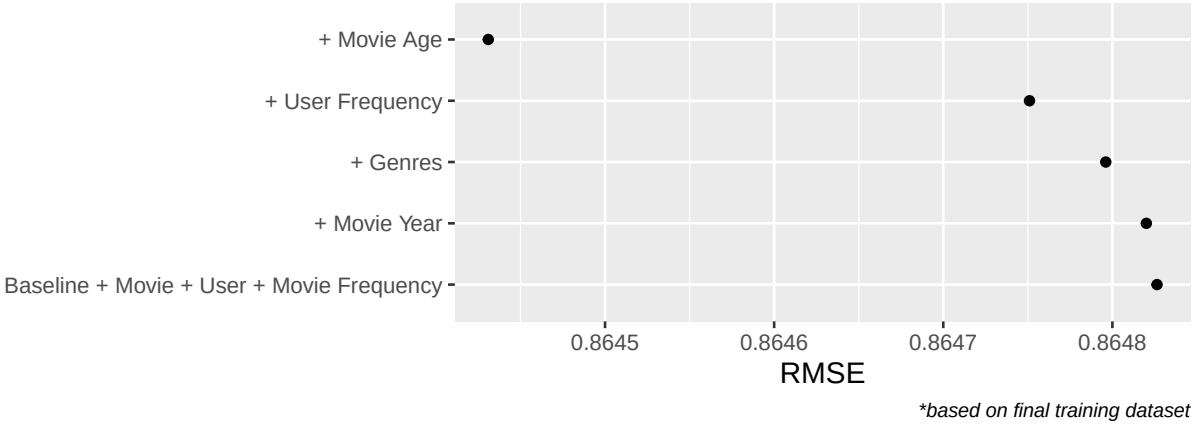

Figure 11. Stepwise RMSE (Movie Frequency Effect)

2.3.2.5. Movie Age Effect

The rating average + movie effect + user effect + movie frequency effect + movie age effect ($\mu + \beta_i + \beta_u + \beta_m + \beta_a$) became the baseline for the 5th stepwise reiteration. The stepwise method then considered the movie year affect (β_y), genre effect (β_g), and user frequency effect (β_f). The result of the stepwise method is shown in the table below.

Stepwise	RMSE	Difference
Baseline + ... + Movie Age	0.8644310	0.00e+00
+ User Frequency	0.8643743	5.67e-05
+ Genres	0.8643765	5.44e-05
+ Movie Year	0.8643562	7.47e-05

As shown in Figure 12, the previous baseline with movie year effect ($\mu + \beta_i + \beta_u + \beta_m + \beta_a + \beta_y$) produced minimal loss compared to other variables. The baseline with the movie year effect ($\mu + \beta_i + \beta_u + \beta_m + \beta_a + \beta_y$) was then chosen as the new baseline for the next stepwise model.

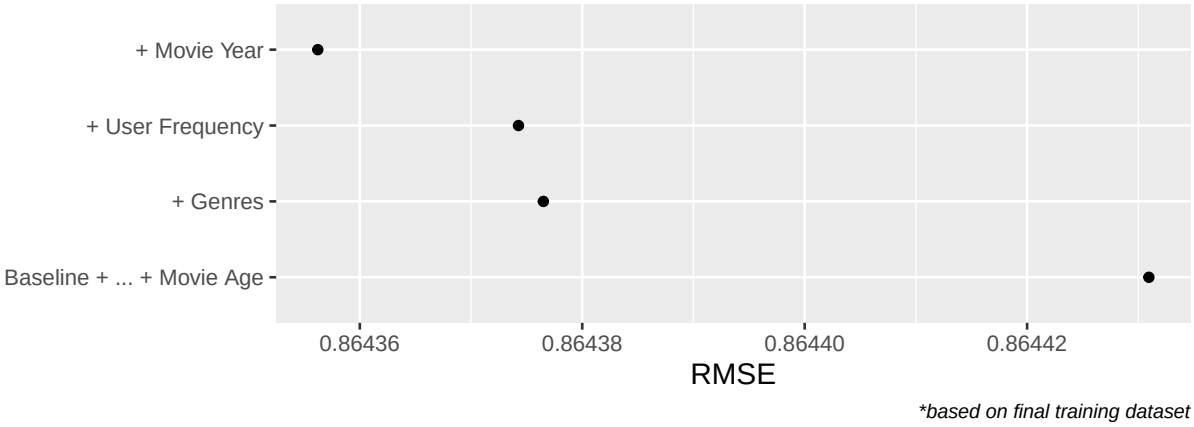

Figure 12. Stepwise RMSE (Movie Age Effect)

2.3.2.6. Movie Year Effect

The rating average + movie effect + user effect + movie frequency effect + movie age effect + movie year effect ($\mu + \beta_i + \beta_u + \beta_m + \beta_a + \beta_y$) became the baseline for the 6th stepwise reiteration. The stepwise method then considered the genre effect (β_g), and user frequency effect (β_f). The result of the stepwise method is shown in the table below.

Stepwise	RMSE	Difference
Baseline + ... + Movie Year	0.8643562	0.00e+00
+ User Frequency	0.8643076	4.87e-05
+ Genres	0.8643153	4.10e-05

As shown in Figure 13, the previous baseline with user frequency effect ($\mu + \beta_i + \beta_u + \beta_m + \beta_a + \beta_y + \beta_f$) produced minimal loss compared to genres. The baseline with the movie year effect ($\mu + \beta_i + \beta_u + \beta_m + \beta_a + \beta_y + \beta_f$) was then chosen as the new baseline for the next stepwise model.

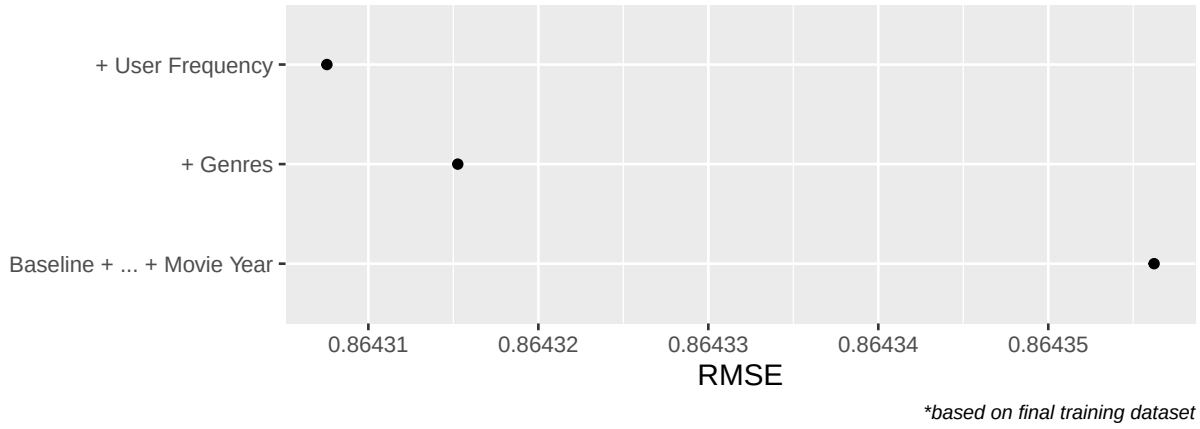

Figure 13. Stepwise RMSE (Movie Year Effect)

2.3.2.7. User Frequency Effect and Genre Effect

The rating average + movie effect + user effect + movie frequency effect + movie age effect + movie year effect + user frequency effect ($\mu + \beta_i + \beta_u + \beta_m + \beta_a + \beta_y + \beta_u$) became the baseline for the 7th stepwise reiteration. The stepwise method then considered the genre effect (β_g). The result of the stepwise method is shown in the table below.

Stepwise	RMSE	Difference
Baseline + Movie + User + Movie Frequency + Movie Age + Movie Year + User Frequency	0.8643076	0.00e+00
+ Genres	0.8642639	4.37e-05

The result shows that the genre effect (β_g) provided improvement in the RMSE, thus it is retain in the final model:

$$Y = \mu + \beta_i + \beta_u + \beta_m + \beta_a + \beta_y + \beta_u + \beta_g + \epsilon$$

2.3.3. Regularization using penalized least squares

In this section, we explore if regularization can still improve our prediction model. By adding a penalty term to our estimates, thus the penalized least squares, the model can constrain the total variability of the effect sizes [9]. Penalizing the least squares estimate helps because some observations, e.g., movie $i = 1$, are rated many times compared to movie $i = 2$, which only one user rates. The situation creates problem estimation since having only one or few raters creates a problem with precision (i.e., lacking sample observation). Instead of ignoring movies with one or few raters, as this creates over-training, it is better to set these movies as having an average rating. This idea is to control for the total variability of the estimates.

To find the best penalty term λ, we run the final model with different penalty terms and find the best one that minimizes the RMSE most. The λ is set to 0 up to 3, with in increment of 0.1. The result in Figure 14 shows that regularization is set to 0. The result means that regularization does not improve the final model. The final model without regularization was retained for the final testing.

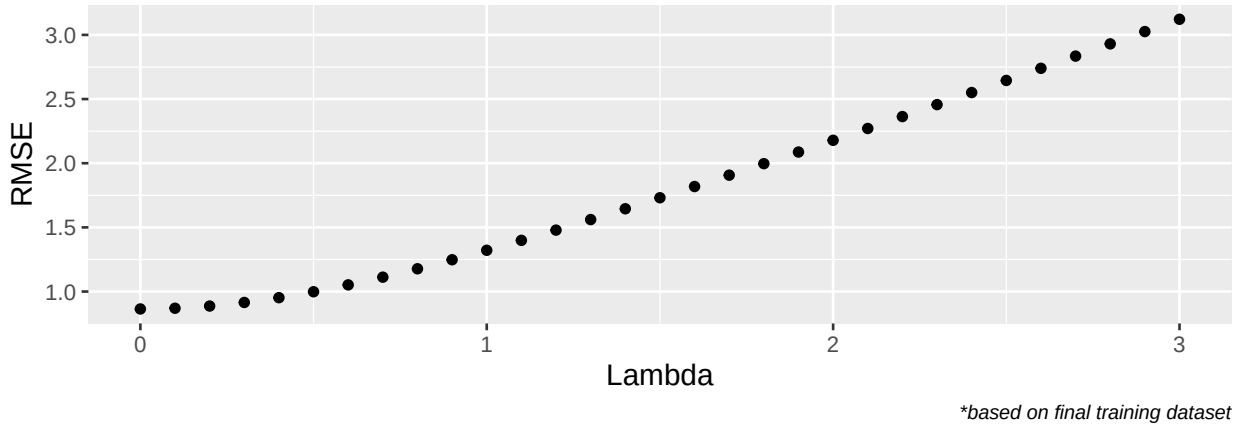

Figure 14. Penalty Terms for Regularization

```
min_lambda <- min(lambda)
```

```
## The lowest penalty (lambda) is = 0.
```

```
rmse_regularized <- min(rmses)
```

```
## The RMSE for the lowest lambda is = 0.864263862450954
```

3. Results

The final evaluation for the model using the final training dataset versus validation has achieved an RMSE of 0.8642, which is lower than the 0.8649, the required RSME. The movie recommendation system algorithm for this project has achieved its objective.

```
## The RMSE for the final model is = 0.8640222
```

Model	RMSE	Difference
Objective	0.8649000	0.0000000
Final Model	0.8640222	0.0008778

4. Conclusion

This project developed a recommendation system using the MovieLens 10M dataset with an RMSE of 0.8640, which has passed the required RMSE of 0.8649. The stepwise method found the best model using the least square for the estimate and RMSE for the evaluation metric. With this, we can find the best possible combination of variables for our prediction model. Specifically, our prediction model comprises several parts. First, a baseline rating pertains to the mean of the overall movie ratings. Second is the movie-specific effect, as some movies are rated higher while some are rated lower than the average. The third is the user-specific effect, as some users tend to rate movies higher or lower than the average. Fourth is the movie's popularity,

as some movies may attract more raters than others and thus may have an effect on the movie's rating. Fifth is the movie age effect, as some users may judge movies based on the age of the movie. Sixth is the year of release effect, as some users may judge based on the year or the generation of the movie it was released. Seventh is the film critic effect, in which some users who rate more may rate differently than those who rate less. Last is the genre effect, as a genre provides different satisfaction to the users depending on their preferences.

Although this project has developed a recommendation system that passed the objectives, there could still be room for improvement in minimizing the error using other techniques such as matrix factorization, neighborhood models, regression, temporal effects, or ensemble methods. Though, most of these techniques require considerable computing power to compute 10 million data. This constraint is why a simpler algorithm is better than a complex one in terms of computing efficiency.

References

[1] https://iopscience.iop.org/article/10.1088/1757-899X/1098/3/032039/pdf

[2] https://link.springer.com/chapter/10.1007/978-1-4899-7637-6_1

[3] https://www.thrillist.com/entertainment/nation/the-netflix-prize

[4] https://grouplens.org/datasets/movielens/

[5] https://gmd.copernicus.org/articles/15/5481/2022/

[6] https://rafalab.github.io/dsbook/introduction-to-machine-learning.html

[7] https://stat.ethz.ch/~geer/bsa199_o.pdf

[8] https://rtmath.net/assets/docs/finmath/html/31a58294-d03b-4f6d-ab26-009e

[9] https://rafalab.github.io/dsbook/large-datasets.html#penalized-least-squares