MovieLens Recommendation System Report

HarvardX Data Science Professional Certificate: PH125.9x
Capstone project(1)

Ahmed G. Alastal

30 March, 2021

Contents

1. Introduction

A recommendation system is a subclass of information filtering system that seeks to predict the "rating" or "preference" that a user would give to an item.

Recommendation systems are utilized in a variety of areas including music, books, news, research articles, search queries, movies, restaurants, garments, financial services, and products in general. Major companies such as Amazon, Twitter, Facebook, Netflix and Spotify utilize recommendation systems.

Netflix uses the recommendation system to predict how many stars a user will give to a specific movie. One star represents a bad rating, whereas five stars represents an excellent one. In 2006, Netflix offered a million dollar prize to anyone who could improve the effectiveness of its recommendation system by 10%.

In this project, I will combine several machine-learning strategies to construct a movie recommendation system using the MovieLens data set, and try to reduce RMSE as much as possible, using skills I learned in the data science program from HarvardX.

1.1 Data set Overview

We can find the full MovieLens dataset here: https://grouplens.org/datasets/movielens/latest/ it is rather large; *(27,000,000 ratings and 1,100,000 tag applications applied to 58,000 movies by 280,000 users, Last updated 9/2018)*

The used dataset in this project is the MovieLens 10M Dataset. You can found and download it from: https://grouplens.org/datasets/movielens/10m/; *(10 million ratings and 100,000 tag applications applied to 10,000 movies by 72,000 users. Released 1/2009)*

The MovieLens 10M Dataset is split into two dataset: edx and validation. The edx data set represents 90% and used for Developing the algorithm and model construction. The validation data set represents 10% and used only for assessing the performance of the final model.

Edx Dataset

The edx dataset consisting of 9,000,055 rows and 6 columns, with ratings provided by a total of 69,878 unique users for a total of 10,677 unique movies. Each row represents a rating given by one user to one movie. The column "rating" is the outcome we want to predict, y.

The following tables show the first 5 rows and the summary statistics in edx dataset:

Table 1: *First 5 rows of edx data set*

userId	movieId	rating	timestamp	title	genres
integer	numeric	numeric	integer	character	character
1	122	5	838985046	Boomerang (1992)	Comedy|Romance
1	185	5	838983525	Net, The (1995)	Action|Crime|Thriller
1	292	5	838983421	Outbreak (1995)	Action|Drama|Sci-Fi|Thriller
1	316	5	838983392	Stargate (1994)	Action|Adventure|Sci-Fi
1	329	5	838983392	Star Trek: Generations (1994)	Action|Adventure|Drama|Sci-Fi
1	355	5	838984474	Flintstones, The (1994)	Children|Comedy|Fantasy

Table 2: *Edx Data set summary*

rows_number	columns_number	users_number	movies_number	average_rating	genres_number	first_rating_Date	last_rating_date
9000055	6	69878	10677	3.512	797	1995-01-09	2009-01-05

The characteristics of edx Dataset features:

Quantitative features.

- userId: integer, Unique ID for the user.
- movieId: numeric, Unique ID for the movie.
- timestamp: integer, Date and time the rating was given.

Qualitative features.

- title: character, movie title (not unique).
- genres: character, genres associated with the movie.

Outcome.

- rating: numeric, a rating between 0 and 5 for the movie.

**** Validation Dataset:**

Validation dataset has the same features of edx dataset and has the following summary:

Table 3: *Validation data set summary*

rows_number	columns_number	users_number	movies_number	average_rating	genres_number	first_rating_Date	last_rating_date
999999	6	68534	9809	3.512	773	1995-01-09	2009-01-05

1.2 Project methodology

After downloading the movielends data set, splitting into edx and validation dataset and taking an overview about dataset, we will follow these steps:

• Analyze and visualize the edx data set, get an overview about and between the features and educe the benefits from the analysis using the cleaning, exploring and visualizing concept.

• Develop concept of modeling using ideas gained in previous step, using the techniques and models introduced in machine learning course.

• Concurrently with previous step, we develop the model using edx data set and evaluate its effectiveness by using RMSE. To do that we will split edx dataset into edx_train and edx_test, various models are constructed using edx_train and their performances are assessed using edx_test.

• Finally, retrain The best performing model using edx and assess using validation

2. Analysis and Exploration data

The main objective of this part is to have a good understanding about each features of the edx data set and find any ideas to develop the recommendation model.

2.1 Ratings Exploration

The rating is an ordinal scale of number from 0.5 to 5. (5 is best value) given by the users who watched the movie. We can see the distribution of rating as the following:

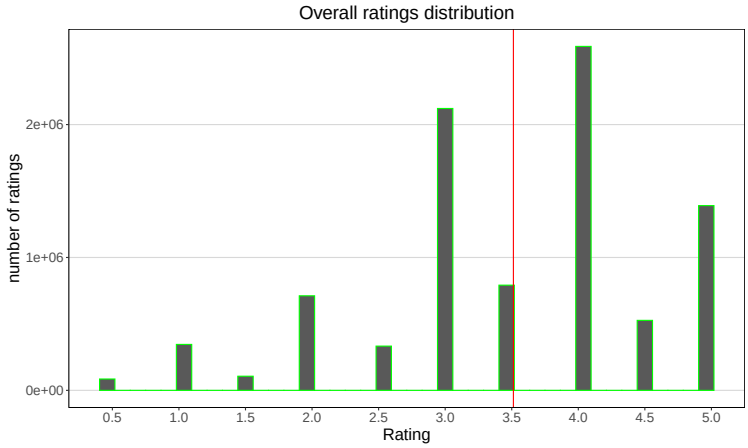

Figure 1: *Overall ratings distribution in Edx dataset*

From the previous figure, we notice that:

- The overall average rating in the edx dataset was 3.51
- The top 3 ratings from most to least are : 4, 3, 5.
- Users desire to rate movies more positively than negatively.
- The histogram shows that the half-star ratings are less common than whole star ratings.

2.2 Movies Exploration

The Edx dataset have total 10,677 movies; represented by a movieId. The following histogram represent the number of ratings by movies:

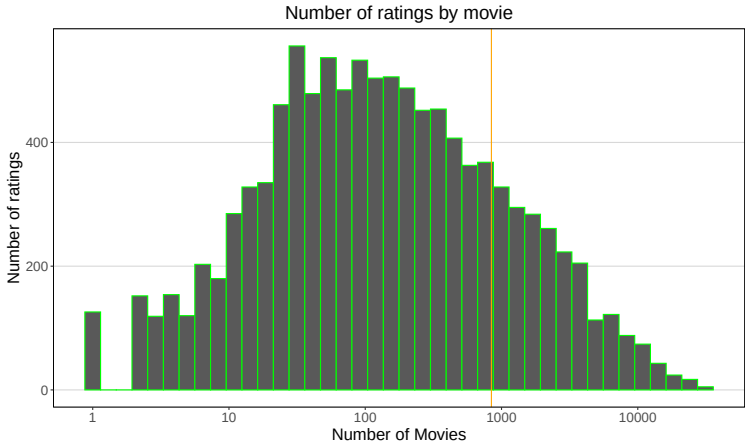

Figure 2: *Number of ratings by movies in Edx Dataset*

Otherwise, the following histogram represent movies distribution by average rating:

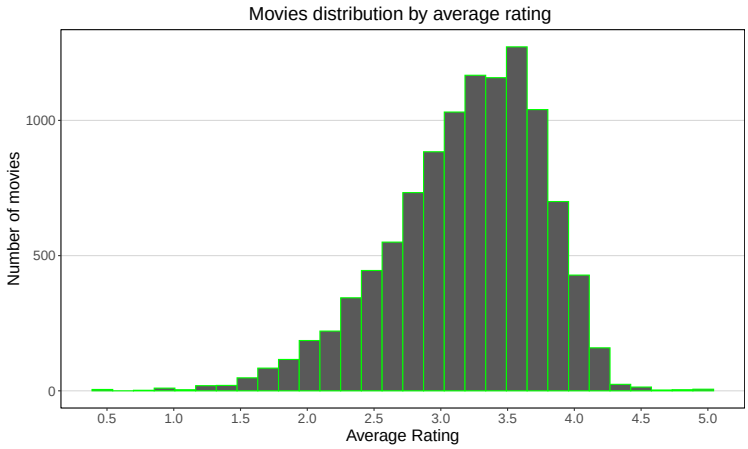

Figure 3: *Movie distribution by average rating in Edx Dataset*

From above, we can conclude:

- Some movies are rated more than other, and the average number of rating is 843.

• Approximate 20 % number of movies have number of rating more than the average, which represent approximate 85% of ratings.
• The average movie rating tends to increase when the number of ratings increases.

2.3 User Exploration

The Edx dataset have 69,878 users represented by userid. The following histogram represent the number of ratings by user:

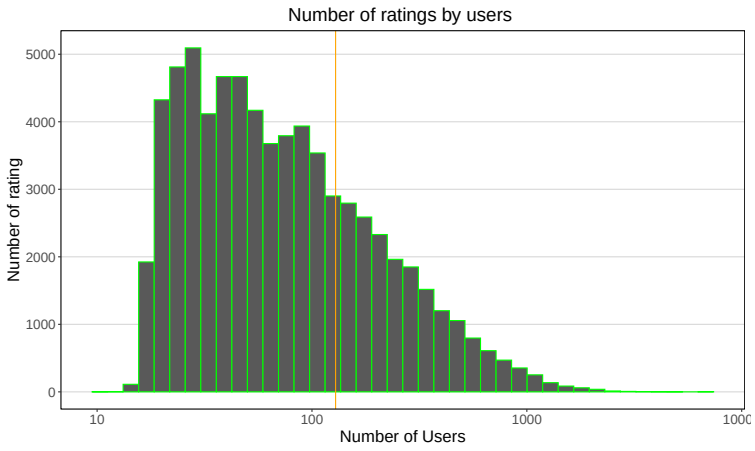

Figure 4: *Number of ratings by users in Edx Dataset*

On the other hand, the following histogram represent users distribution by average rating:

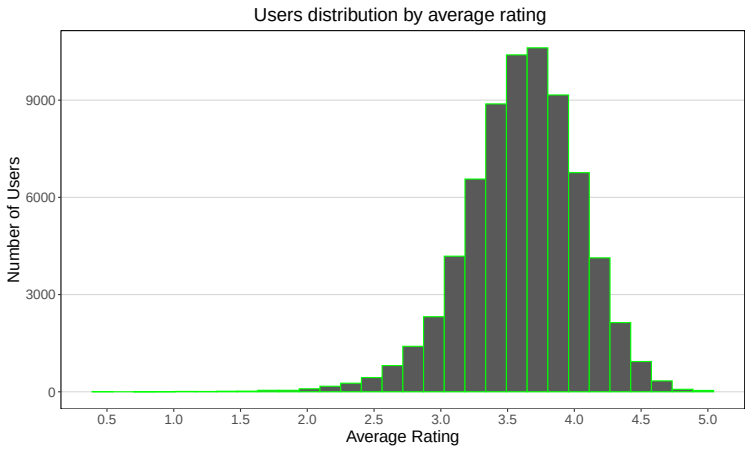

Figure 5: *Users distribution by average rating in Edx Dataset*

From above, we can conclude:
• 30 % of users contribute 70 % of ratings in whole edx dataset.
• Some users rated very few movie and their opinion may bias the prediction results.
• The average user rating tends to increase when the number of ratings increases.

2.4 Time effect Exploration

Time is recorded as the UNIX timestamp, the UNIX timestamp is merely a number of seconds between a particular date and the Unix Epoch. This count starts at the Unix Epoch on January 1st, 1970 at UTC. The following figure represent the average ratings of movies by month:

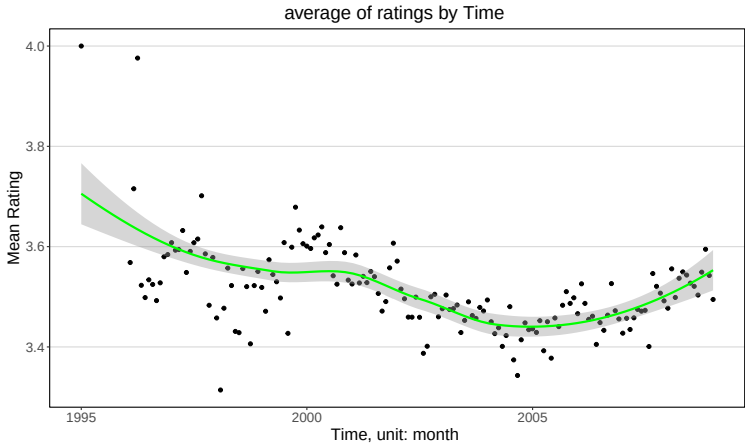

Figure 6: *Average ratings by time/month in Edx Dataset*

From the previous figure, we conclude that:
There are some evidences about time effect on ratings average, but this effect is not a strong.

2.5 Genres Exploration:

A movie could be classified to one or more genres; there are 20 levels of genre.The figure 7 illustrates the number of movies per genres:

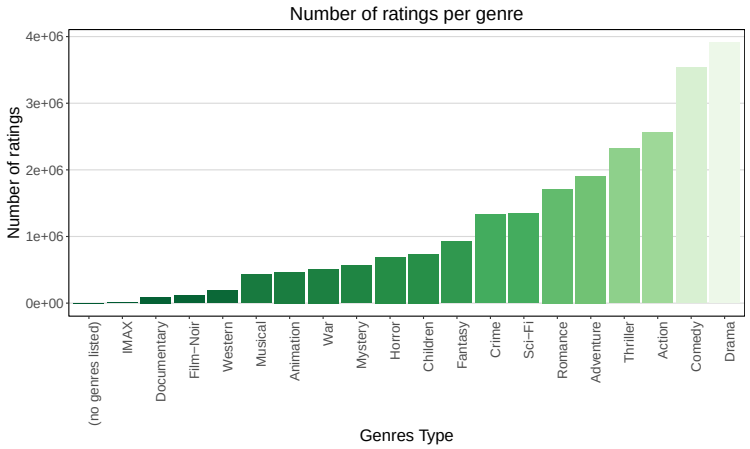

Figure 7: *Number of ratings by genre in Edx Dataset*

On the other hand, we can see the average of ratings per genres as shown in the figure 8.

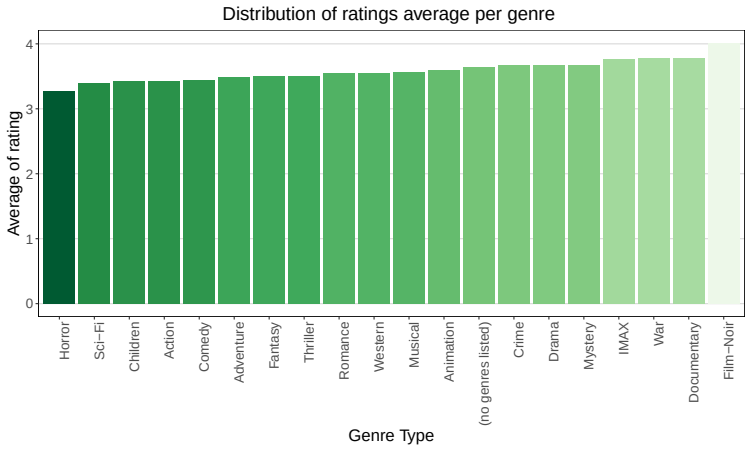

Figure 8: *Distribution of ratings average per genre in Edx Dataset*

We can see the relation between numbers of ratings and rating average as shown in the figure9.

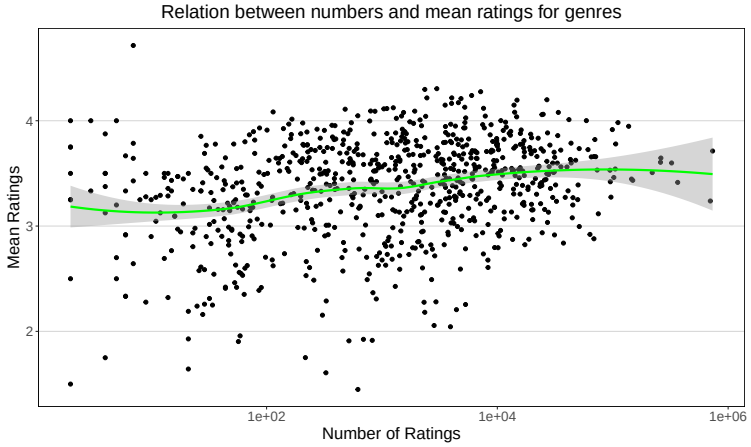

Figure 9: *Relation between Number of Ratings vs Mean Rating for Genres*

From above, we can conclude:

- The number of ratings varies per genre.
- The ratings average for genres are Converging, although the number of ratings varies.
- The genres only slightly affect movie ratings.

3. Modeling approach

In this section, we are going to explain the modeling approach we used to construct our models, and present the metric for the model performance evaluation.

3.1 Model performance evaluation

To compare our models performance, we will use root mean squared error (RMSE) as our loss function. The RMSE represent the error loss between the predicted ratings derived from applying the model and actual ratings in the test set
In the formula shown below, $y_{u,i}$ is defined as the actual rating provided by user i for movie u, $\hat{y}_{u,i}$ is the predicted rating for the same, and N is the total number of user/movie combinations.

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

##3.2 Constructing and developing baseline Model

Based on the exploration of the data described above, genres and Time does not add much information. Therefore, to first build a baseline prediction model, I will consider the movie effect and user effect.

The simplest algorithm for predicting ratings is to apply the same rating to all movies. Here, the actual rating for movie i by user u, $Y_{u,i}$, is the sum of this "true" rating, μ, plus $\epsilon_{u,i}$, the independent errors sampled for the same distribution.

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

- **Movie effects**

since we know that some movies are generally rated higher than others, accounting for this effect will therefore improve the accuracy of the prediction. This implies that an additional improvement to our model may be by taking into account the effect of movie on rating, b_i.

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

The least squares estimate of the movie effects, $\hat{b}_i$, can be derived from the average of $Y_{u,i} - \hat{\mu}$ for each movie i and, thus, the following formula was used to take account of movie effects:

$$\hat{b}_i = mean\left(\hat{y}_{u,i} - \hat{\mu}\right)$$

- **Movie and user effects**

We also know that some users are more active than others at rating movies so further refinements were made to the algorithm to adjust for user effects (b_u).

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

As previously, rather than fitting linear regression models, the least square estimates of the user effect, $\hat{b}_u$ was calculated using the following formulas:

$$\hat{b}_u = mean\left(\hat{y}_{u,i} - \hat{\mu} - \hat{b}_i\right)$$

3.3 Regularization

Regularization permits us to penalize large estimates that come from small sample sizes. It has commonalities with the Bayesian approach that shrunk predictions. The general idea is to add a penalty for large values of bi, bu to the sum of squares equation that we minimize. So having many large bi or bu makes it harder to minimize.

A more accurate estimation of bu and bi will treat them symmetrically, by solving the least squares problem

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

where the first term $\frac{1}{N}\sum_{u,i}\left(y_{u,i} - \mu - b_i - b_u\right)^2$, strives to find b_u's and b_i's that fit the given ratings. The regularizing term, $\lambda\left(\sum_i b_i^2 + \sum_u b_u^2\right)$, avoids overfitting by penalizing the magnitudes of the parameters. This least square problem can be solved fairly efficiently by the method of stochastic gradient descent that will be object in the matrix factorization recommender engine.

At this step, we used cross validation to pick the best λ and using calculus we can show that the values of b_i and b_u that minimize this equation are :

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

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

3.4 Matrix factorization

Matrix factorization is a class of collaborative filtering algorithms used in recommender systems. Matrix factorization algorithms work by decomposing the user-item interaction matrix into the product of two lower dimensionality rectangular matrices. This family of methods became widely known during the Netflix prize challenge due to its effectiveness as reported by Simon Funk in his 2006 blog post, where he shared his findings with the research community. for more detail: (https://en.wikipedia.org/wiki/Matrix_factorization_(recommender_systems))

We will apply Matrix Factorization with parallel stochastic gradient descent. With the help of "recosystem" package it is an R wrapper of the LIBMF library which creates a Recommender System by Using Parallel Matrix Factorization. The main task of recommender system is to predict unknown entries in the rating matrix based on observed values. for more information about recosystem package and the techniques: (https://cran.r-project.org/web/packages/recosystem/vignettes/introduction.html)

4. Results

It should be noted that the main objective is to develop a recommendation system to reduce the RMSE to less than 0.86490.

As the validation dataset was reserved for the final hold-out test, edx is split into edx_train (80%) and edx_test (20%) dataset. Various models are constructed using edx_train and their performances are assessed using edx_test.

4.1 Model 1: BaseLine Model

It's simply a model which ignores all the feathers and calculates mean rating, This model acts as a baseline model and we will try to improve RMSE relative to this baseline standard model.
The average rating is mu = 3.5125, and the RMSE is 1.0599.

Model	RMSE
Just the Average	**1.059904**

4.2 Model 2: Movie effects Model

Since the features of a movie could obviously affect the ratings of a movie, we will add the bias of movies b_i for each movie to the model. The average of the ratings on that specific movie will have a difference from the overall average rating of all movies. We will plot the distribution of the movie bias and calculate the RMSE of this model.

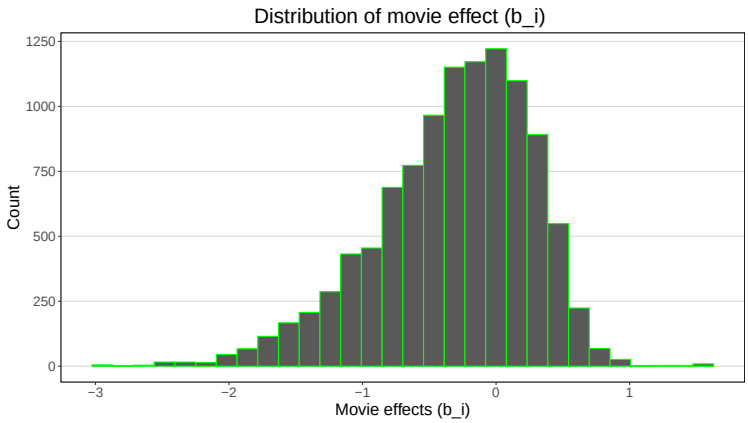

Figure 10: Distribution of movie effect (b_i) Edx_train dataset

Figure 10 shows that the estimate of movie effect (b_i) varies considerably across all of the movies included in the train dataset. Adding the movie effect into the algorithm to improved

the accuracy of the model by 10.96%, yielding an RMSE of 0.9437, but its still above the target.

Model	RMSE
Just the Average	**1.059904**
Movie Effect Model	**0.943700**

4.3 Model 3: Movies and Users effects Model

Similar to the movie effect, features of a given user could also affect the ratings of a movie. a user could give lower scores for all movies watched than rated by other users.
We will add the user bias b_u to the movie effect model and calculate the RMSE of this model

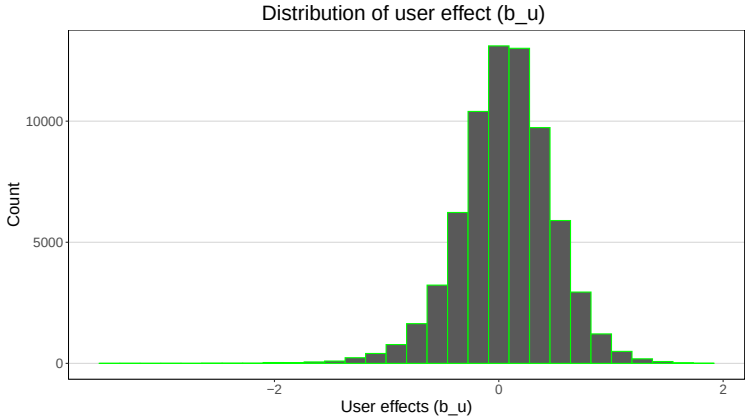

Figure 11: Distribution of movie effect (b_u) in Edx_train dataset

Figure 11 shows the estimated effect of user (b_u) building on the movie effects model above. it was evident that adjusting for user effects enhanced the accuracy of the algorithm.Adjusting for both movie and user effects to improved the RMSE by 18.30% versus the Baseline model.

Model	RMSE
Just the Average	**1.059904**
Movie Effect Model	**0.943700**
Movie and User Effect model	**0.865900**

4.4 Model 4: Regularization of movie and user effects

Regularization technique should be used to take into account on the movie and user effects, by adding a larger penalty to estimates from smaller samples. so we will use parameter λ.

We have noticed in our data exploration, some users are more actively participated in movie reviewing. There are also users who have rated very few movies. On the other hand, some movies are rated very few times. These are basically noisy estimates that we should not trust. Additionally, RMSE are sensitive to large errors. So we must put a penalty term to give less importance to such effect.

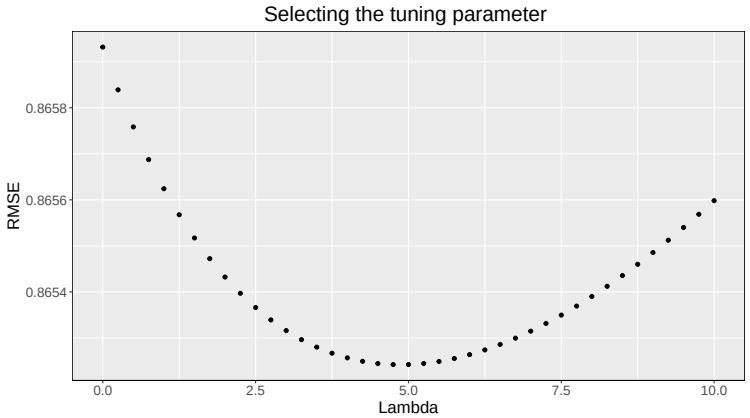

Figure 12: Selecting the tuning parameter in Edx_train dataset

Figure above shows the RMSE delivered across each of the values for lambda tested. The optimum value for λ was 4.75 which minimised RMSE to 0.8652, which was just sufficient to surpass the target RMSE set as the project objective. This represented a total improvement of 18.37% in the accuracy of the baseline.

Model	RMSE
Just the Average	**1.059904**
Movie Effect Model	**0.943700**
Movie and User Effect model	**0.865900**
Regularized Movie and User Effect Model	**0.865200**

4.5 Model 5: Matrix Factorization

By following the required procedure for recosystem library, which was explained in the website, the following result can be obtained:

Performing the Matrix factorization method, we will achieve a RMSE equal 0.7964 If we calculate the percentage decrease of rmse we could see that the matrix factorization method shows a decrease of more than 24.86% with respect to the base line model.

Model	RMSE
Just the Average	**1.059904**
Movie Effect Model	**0.943700**
Movie and User Effect model	**0.865900**
Regularized Movie and User Effect Model	**0.865200**
Matrix Factorization	**0.796400**

4.6 Final Model

The previous sections construct models which are assessed using edx_test that is a subset of edx dataset. The best performing model is amatrix factorization which yields an RMSE of 0.7964. Therefore, the final model takes the same approach. The final model is constructed using the entire edx data set and is then evaluated using validation. Note that validation is not used at any point in this report so it is a fair assessment. The final hold-out test in the validation dataset achieve an RMSE of 0.7867, an improvement of 25.78% versus the baseline model. As shown in Table below:

Model	RMSE
Best Model: Matrix Factorization	**0.7867**

5. Conculusien

The objective of this project is to develop a recommendation system using the MovieLens 10M dataset that predicted ratings with a residual mean square error of less than 0.86490.

This report discusses a few methods used to construct recommendation systems. The best performing model is matrix factorization which yields an RMSE of 0.7867 when trained on edx and tested on validation. This is an improvement on the first model's RMSE by around 25.78%.

Although the algorithm developed here met the project objective it still includes a sizeable error loss, which suggests that there is still scope to improve the accuracy of the recommendation system.

In conclusion, matrix factorization appears to be a very powerful technique for recommendation system. those interested in a more advanced approach to construct recommender systems which should consult the recosystem package. It provides the means to implement matrix factorisation on large data sets.

6. Reference

[1] "Introduction to Data Science - Data Analysis and Prediction Algorithms with R", Dr. Rafael A. Irizarry link

[2] "R Markdown: The Definitive Guide", Yihui Xie, J. J. Allaire, Garrett Grolemund, 2019-06-03 link

[3] "Recommender System Using Parallel Matrix Factorization", Yixuan Qiu, 2021-01-09. link

[4] "E. Winning the Netflix Prize: A Summary", Chen, 2020/10/15. link