Movie Recommendation System

HarvardX Data Science Professional Certificate - PH125.9x Data Science: Capstone (MovieLens Project)

Audrey Karabayinga

2022-09-19

Contents

1 Introduction

1.1 Recommendation systems

Recommendation or recommender systems (RS) are technologies that make predictions and subsequently recommend items to a user (Pu, Chen, and Hu 2011). These recommendations, such as what item to purchase or what movie to watch, may be based on aggregates of a large number of user preferences, e.g., ratings, a recommendation technique known as *collaborative filtering* (Zhou et al. 2008). RSs are used by well-known companies, like Netflix, YouTube, and Amazon, with the goal of increasing user engagement (Silveira et al. 2019).

Measures of a good RS include high predictive accuracy, i.e., how close the RS's predictions are to the true ratings. Accuracy-based metrics for assessing and evaluating an RS's performance include root/residual mean square error (RMSE).

1.2 GroupLens and MovieLens

GroupLens, a research laboratory in the Department of Computer Science and Engineering at the University of Minnesota, was one of the pioneers of research focusing on RSs. This group of researchers created several recommendation services, including MovieLens, a well-known movie recommendation website that uses *collaborative filtering* to make movie recommendations to users. The MovieLens 10M data set, used for this project, consists of 10,000,054 ratings of 10,681 movies by 71,567 users of the MovieLens recommender service.

1.3 Capstone project

This report was created for the MovieLens Project, one of two Capstone projects required to fulfill the HarvardX Data Science Professional Certificate series. The stated aim of the project was to develop a movie recommendation system using the MovieLens 10M data set by training a machine learning algorithm to predict movie ratings. To measure the quality of the fit of the model, RMSE was used. The target for this project was to achieve **RMSE < 0.86490** when comparing predicted movie ratings to true ratings in the final hold-out test (*validation* set).

To achieve this, the course instructor provided a code to download and combine the original 10M MovieLens data sets into a *movielens* data set, which was then split into 2 data sets: i) **edx** data set (*90%* of *movielens*), used to generate the movie recommendation algorithm, and; ii) **validation** data set (*10%* of *movielens*), *only* used with the final algorithm to predict movie ratings and compute the final project RMSE.

Before building the algorithm, data exploration of the *edx* data set was conducted to examine the different components of the data and assess their effects on rating. *edx* was then split into the *train* and *test* sets that were used to train a rating prediction algorithm. Linear regression models were created and trained by adding variable effects, one at a time, until an RMSE close to the target RMSE was obtained. The model with the lowest RMSE was then regularised, resulting in a model with an RMSE value less than the target RMSE. This final, regularised, model was finally used to predict ratings in the *validation* set and to compute the final model RMSE, which was also below the target RMSE.

The documents submitted for assessment of this project are: i) a report in R markdown format; ii) a report in PDF file format (knit from the Rmd file), and; iii) a script in R format that generates predicted movie ratings and RMSE score.

2 Methods

2.1 Data Cleaning, Exploration and Visualization

Using code provided by the course instructor, the original MovieLens 10M data set files were downloaded, and used to generate the **movielens** data set, which was then divided into 2 as shown in Figure 1:

- **edx** data set, which consisted of *90%* of data *(9,000,055* observations) from the *movielens* data set and was used to generate the recommendation algorithm, and;
- **validation** data set, which consisted of *10%* of data *(999,999* observations) from the *movielens* data set and was *only* used with the final algorithm to predict movie ratings and compute final project RMSE.

The above splitting was done in a way that ensured that users and movies in the *validation* set were also present in the *edx* set. The **edx** set was later divided into the **train** and **test** sets in the modeling approach section.

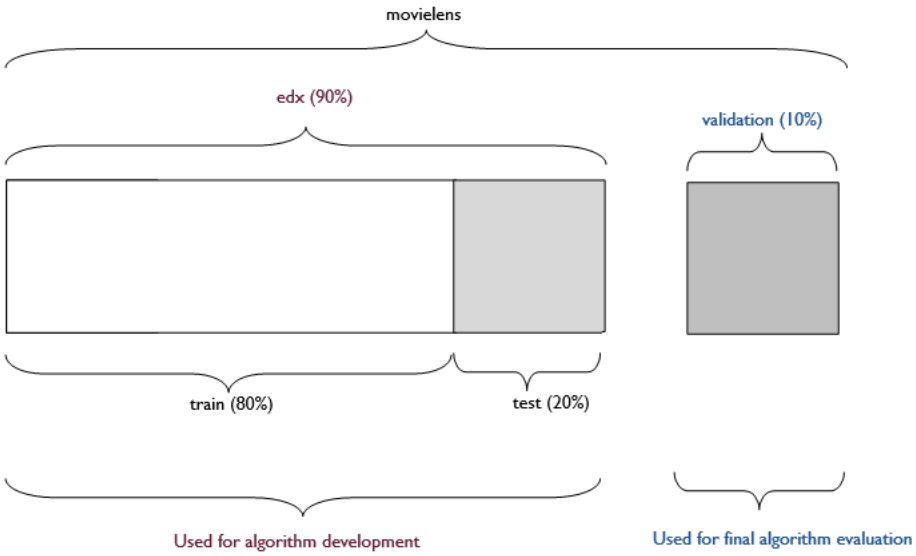


Figure 1: Allocation of data

To get a general idea of the contents of the data, further exploration was carried out **only on *edx* data set** — the *validation* data set was only introduced in the Final algorithm validation section. First, the overall structure of *edx* was examined to learn more about the data set's variables and variable attributes (Table 1). Each row in the data set is a rating given by one user at a particular point in time, for one movie that was released in a particular year and is categorized under one or more genres. The data set consists of 6 variables whose attributes are as follows:

- **userId**: Unique identifier for each user
- **movieId**: Unique identifier for each movie
- **rating**: A 10-scale rating from 0.5 to 5 (in increments of 0.5) made for a movie by a user

Table 1: Overview of edx data set

userId	movieId	rating	timestamp	title	genres
Class					
integer	numeric	numeric	integer	character	character
First 10 observations					
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
1	356	5	838983653	Forrest Gump (1994)	Comedy|Drama|Romance|War
1	362	5	838984885	Jungle Book, The (1994)	Adventure|Children|Romance
1	364	5	838983707	Lion King, The (1994)	Adventure|Animation|Children|Drama|Musical
1	370	5	838984596	Naked Gun 33 1/3: The Final Insult (1994)	Action|Comedy

- **timestamp**: Date and time when rating was made (in seconds since January 1, 1970)
- **title**: Movie title and year of release
- **genres**: Genre(s) associated with a movie

Further examination revealed that *edx* did not contain any missing values. However, as every user in the data set did *not* rate every movie in the data set, the resulting user-movie matrix, whose entries indicate whether the user in a certain row rated the movie in a certain column, was sparse. According to Bøe (2007), sparsity level is defined as

$$1 - \frac{number\ of\ nonzero\ entries}{total\ number\ of\ entries}$$

edx contains 9,000,055 distinct ratings given by 69,878 users for 10,677 movies. The sparsity of *edx* was 0.988 as shown below:

$$1 - \frac{9000055}{69878 * 10677} = 0.988$$

Thus, for every 1000 observations in the user-movie matrix, 988 are missing. Figure 2 shows part of the *edx* user-movie matrix, consisting of a random sample of 50 movies and 50 users, to illustrate its sparsity.

Each of the 6 variables was analysed further below.

2.1.1 Rating

The *rating* column contained the outcome variable, rating score. It used a scale ranging from 0.5 to 5 stars, including half-star ratings. The mean of rating score in *edx* was 3.51 and the standard deviation was 1.06. Figure 3 shows that the most common rating score was 4 and the least common was 0.5. Whole star ratings (1, 2, 3, 4, 5) were more prevalent at 79.5% than half star ratings (0.5, 1.5, 2.5, 3.5, 4.5) at 20.5%. This was likely because the first whole star rating was given in 1995, while the first half-star rating was 8 years later, in 2003.

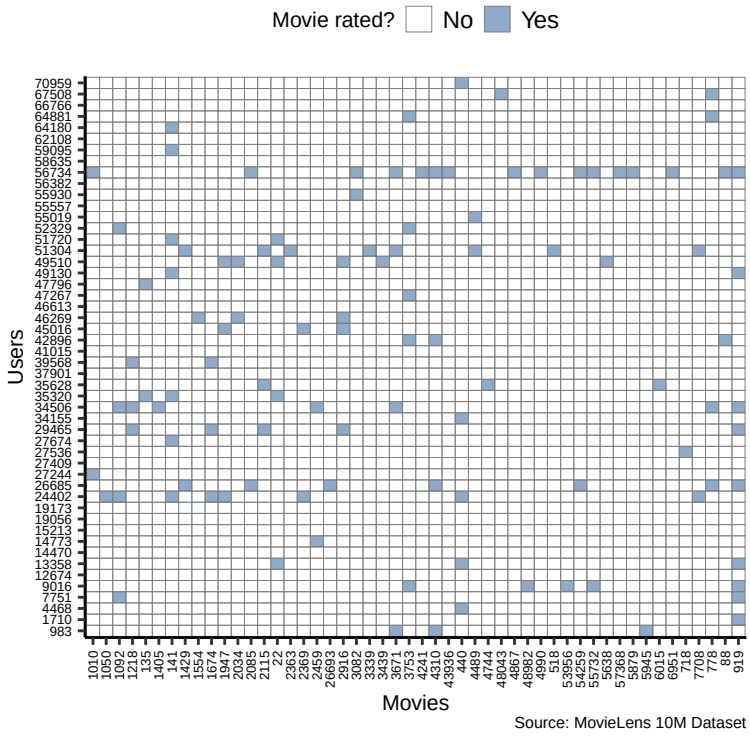


Figure 2: User-movie matrix containing 50 movies and users from edx

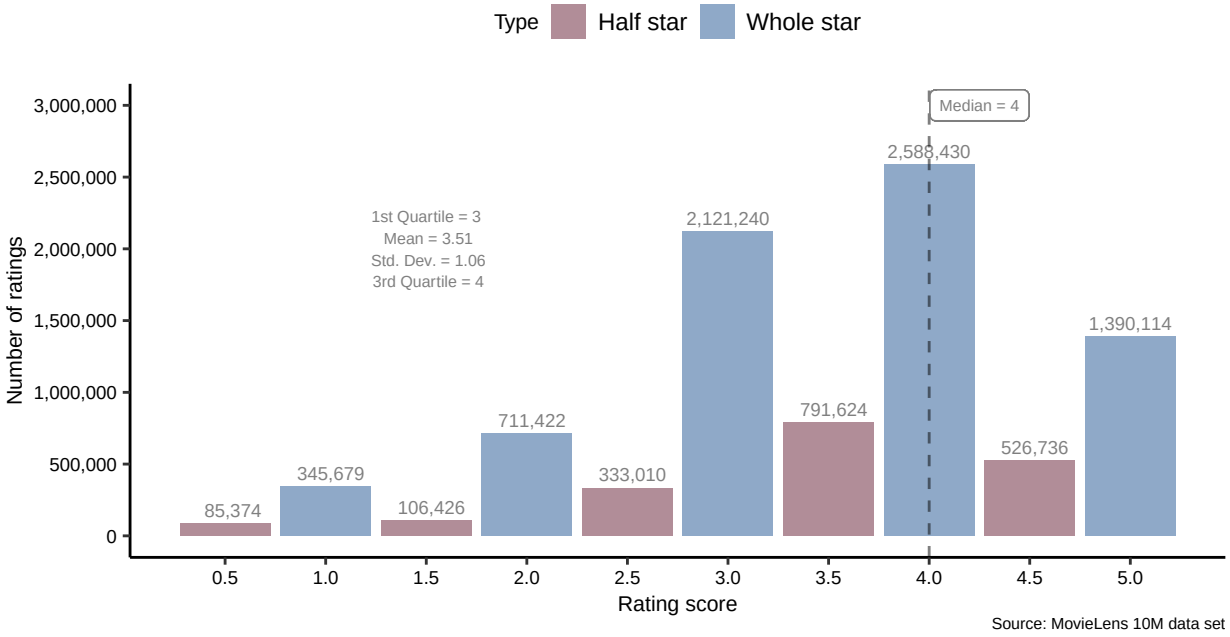


Figure 3: Distribution of rating scores

2.1.2 Movie

The *movieId* column of *edx* contained a total of 10,677 distinct movies. The distributions of number of ratings[1] (Figure 4A) and average ratings scores (Figure 4B) across movies showed slightly positive and negative skewness, respectively. The top 10% of movies, by number of ratings, had 2,154 ratings or more, while the bottom 10% had 10 ratings or fewer. A majority of movies with the highest possible or the lowest possible average rating scores had a very small sample size (i.e., they were only rated by a single user). The mean number of ratings per movie was 843 ($\pm$ 2,238). The average rating score per movie was 3.19 ($\pm$ 0.571). The coefficient of variation (CV) for the number of ratings was 2.66 and the CV for the average rating score was 0.18[2]. Thus, the number of ratings had more variation, relative to their mean, compared to the average rating score.

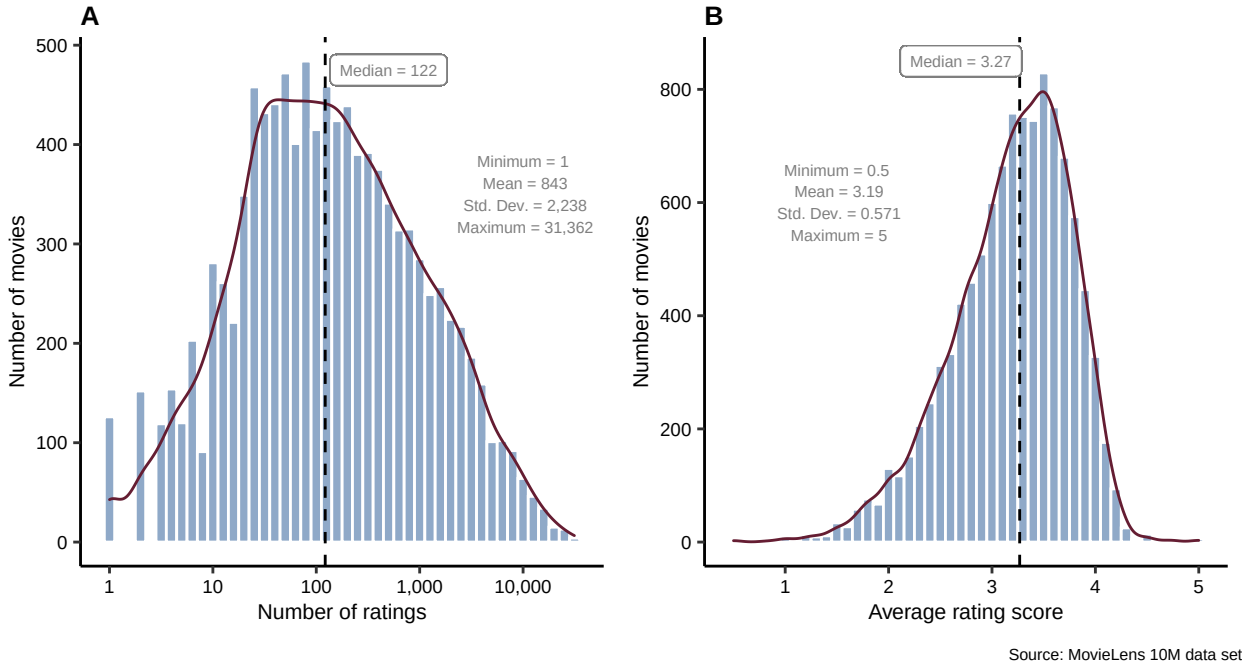


Figure 4: Distribution of number of ratings (A) and average rating scores (B) across movies

Furthermore, only a handful of movies received either very many or very few ratings. For example, the top 5 movies, by number of ratings received, had over 28,000 ratings each (Table 2). The movie with the highest number of ratings was *Pulp Fiction (1994)*, which received 31,362 ratings. 126 movies received the lowest possible number of ratings, a rating of only 1 each. Average rating scores were more variable among movies rated only once compared to movies with the highest number of ratings (Table 2).

2.1.3 User

The *edx* data set constisted of 69,878 distinct users, each uniquely identified by a *userId*. The distributions of the number of ratings[3] (Figure 5A) and average ratings scores (Figure 5B) across users showed positive and

[1] Note that the number of ratings for each movie is equal to the number of users who rated the movie.

[2] The coefficient of variation (CV) is defined as a ratio of the standard deviation to the mean (Lovie 2005). It allows for comparison of the variation in number of ratings and the average rating score.

[3] The number of ratings for each user is equal to the number of movies that the user rated.

Table 2: Top and bottom movies by number of ratings

Title (Release year)	Number of ratings	Average rating score
Most rated		
Pulp Fiction (1994)	31,362	4.15
Forrest Gump (1994)	31,079	4.01
Silence of the Lambs, The (1991)	30,382	4.20
Jurassic Park (1993)	29,360	3.66
Shawshank Redemption, The (1994)	28,015	4.46
Least rated		
Down and Derby (2005)	1	3.50
Double Dynamite (1951)	1	2.00
In Bed (En la cama) (2005)	1	2.50
Demon Lover Diary (1980)	1	4.50
Neil Young: Human Highway (1982)	1	1.50

* The edx data set contained 126 movies rated by a single user (i.e., 'least rated'). Only 5 of these are shown in this table.

negative skewness, respectively. On average, users rated 129 ($\pm$ 195) times. A majority of users (99.1%) rated less than 1,000 movies overall. Only a handful of users rated either very many or very few movies e.g., the top 5 users, by number of ratings, rated over 4,000 movies while the bottom 5 rated 14 movies or less, each (Table 3). The user with the highest number of ratings had 6,616 ratings while the user with the lowest number of ratings had 10 ratings. 5 users gave the lowest possible average rating score, 0.5, while 20 users gave the highest possible average rating score, 5. The CV for the number of ratings was 1.51 and the CV for the average rating score was 0.119. Thus, the number of ratings had more variation, relative to their mean, compared to the average rating score.

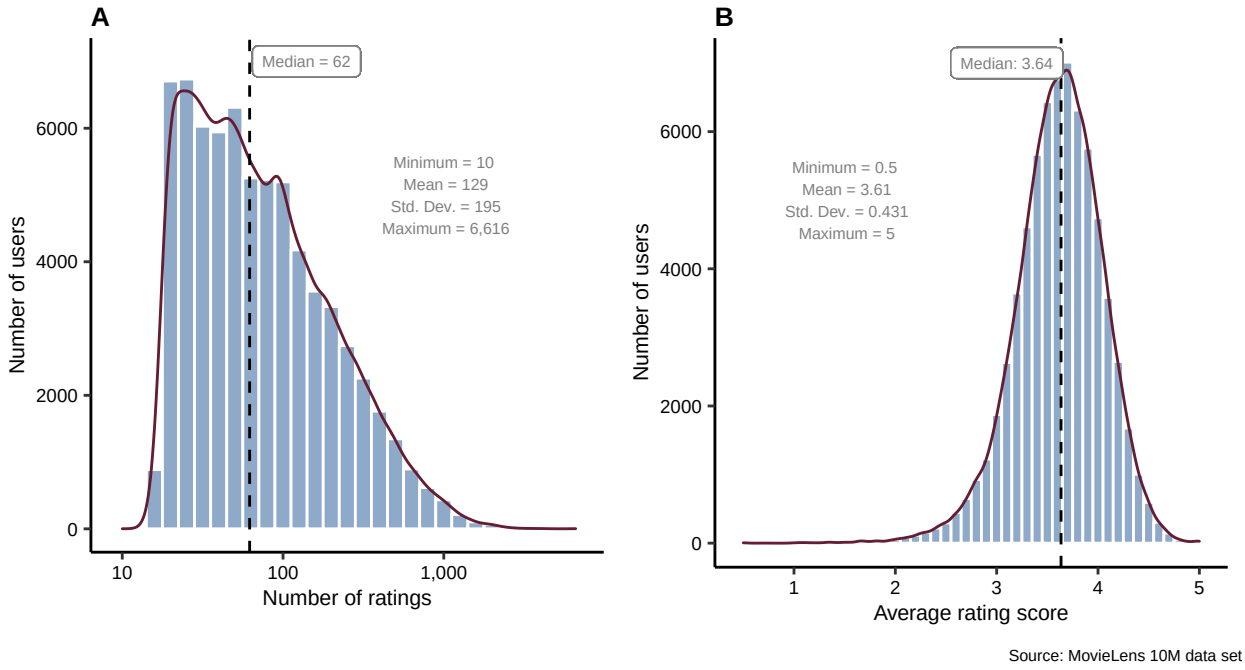


Figure 5: Distribution of number of ratings (A) and average rating scores (B) across users

Table 3: Top and bottom users by number of ratings

User ID	Number of ratings	Average rating score
Most frequent raters		
59269	6,616	3.26
67385	6,360	3.20
14463	4,648	2.40
68259	4,036	3.58
27468	4,023	3.83
Least frequent raters		
62516	10	2.25
22170	12	4.00
15719	13	3.77
50608	13	3.92
901	14	4.71

2.1.4 Genre

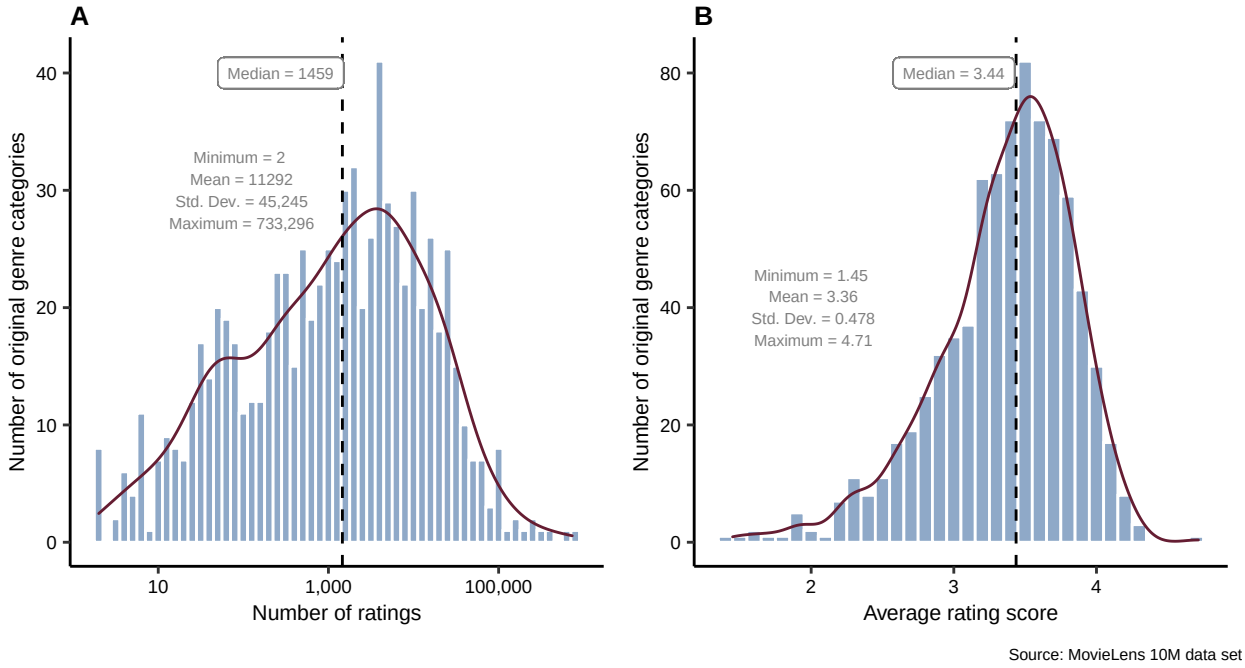


Figure 6: Distributions of number of ratings (A) and average rating scores (B) across original genre categories

The *genre* column contained, for each movie, all the different genres that it could be associated with, resulting in some movies being classified under multiple genres e.g., *Boomerang (1992)* was assigned to the genres *Comedy|Romance* (see Table 1). 3 genre categories were considered.

2.1.4.1 Original genre categories The data set contained 797 of these original genre categories, including a *'no genres listed'* category. As this latter category only had only one movie, it was discarded, during modeling, and only 796 original genre categories were considered. Figure 6 shows the distributions for the number of ratings (A) and average rating scores (B) across the original genre categories. The CV for

the number of ratings across original genre categories was 4.01, while the CV for the average rating scores 0.142.

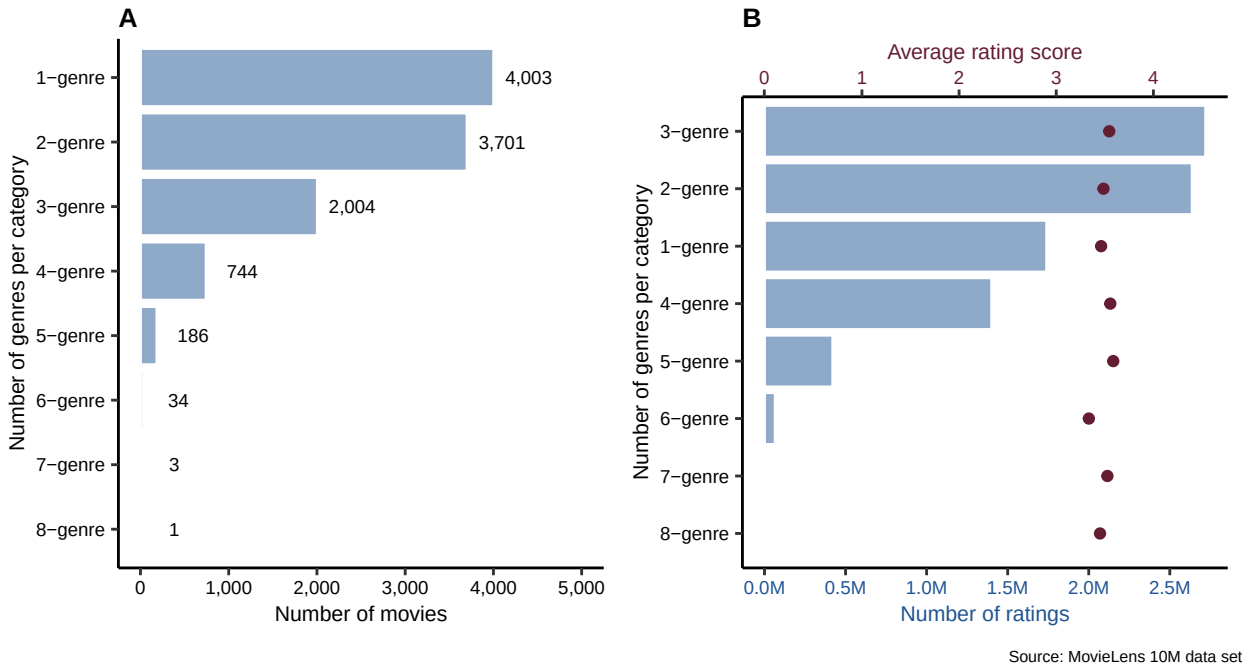


Figure 7: Distributions of number of movies (A) and number of ratings and average rating scores (B) across number-of-genres categories

2.1.4.2 Number-of-genres categories Here, genres were categorized as 1-, 2-, 3-,4-, 5-, 6-, 7-, and 8-genre categories, based on the number of genres. For examples, genres *Comedy|Romance* and *Action|Thriller* were categorised as 2-genre, and *Action|Crime|Thriller* and *Action|Adventure|Sci-Fi*, as 3-genre. Figure 7A shows the distribution of movies across number-of-genres categories. 4,003 movies (37.5% of all movies) were categorized as 1-genre movies and the rest, 6,673 movies (62.5% of all movies) were categorized under multiple genres. 1-, 2- and 3-genre categories contained 9,708 movies, (90.9% of all movies) (Figure 7A).

Data was explored to assess variation in number of ratings and average rating scores across number-of-genre categories (Figure 7B). Large variations in number of ratings were noted, with the 3-genre category having the highest number of ratings, 2,721,164, (30.2% of all ratings) and the 8-genre category the lowest, 256 (0.0028% of all ratings). The CV for the number of ratings across number-of-genre categories was 1.03. Lower variation in average rating scores was noticed, with the 5-genre category having the highest average rating score, 3.59, and the 6-genre category the lowest, 3.33 (Figure 7B). The CV for the average rating score across number-of-genre categories was 0.0227.

2.1.4.3 Individual genres From the original genre categories in *edx*, individual genres were extracted, resulting in 19 categories, including the *'no genres listed'* category.[4] Figure 8A shows the distribu-

[4]It was noted that according to the MovieLens 10M Readme file, the data set consisted of only 18 distinct genres. *IMAX*, included here as the 19th genre, was not listed on the file. 29 movies were found to be assigned to the *IMAX* genre and 1 movie had no genre listed. For this analysis, *IMAX* was retained as the 19th genre to keep the corresponding movies.

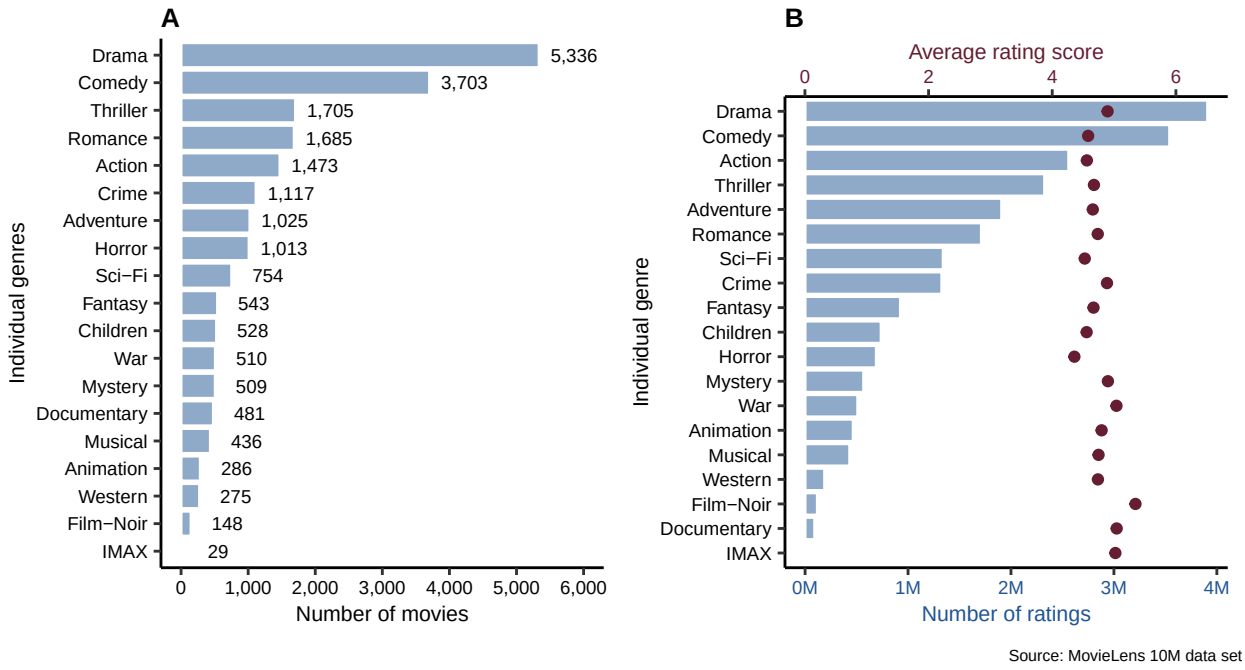


Figure 8: Distributions of number of movies (A) and number of ratings and average rating scores (B) per individual genre

tion of movies across individual genres.[5] The distribution of the number of movies across individual genres revealed that Drama had the most number of movies, 5,336, while IMAX had the least, 29.

Variation in the number of ratings and average rating scores across individual genres was also analysed. Figure 8B showed that *Drama* had the highest number of ratings, 3,910,127, while *IMAX* had the lowest, 8,181. The CV for the number of ratings across individual genres was 0.941. There was less variation in average rating scores across distinct genres. *Film-Noir* had the highest average rating score, 4.01, while *Horror* had the lowest average rating score of 3.27. The CV for the rating scores across individual genres was 0.0488.

2.1.5 Movie title and release year

The *title* column consisted of a movie's title and year of release (e.g, Boomerang (1992)).

Movie release years were extracted from the *title* column and explored. The oldest movie was released in 1915. Only 1 movie was released that year and it received 180 ratings (from 180 users) and had an average rating score of 3.29. The newest movies were released in 2008. This year had 251 movie releases, 26,741 ratings from 4,519 distinct users, and an average rating score of 3.46. Overall, there was a steady increase in number of ratings across years, which peaked in 1995, followed by a decline. The year with the highest possible number of ratings, 1995, had 786,762 ratings, while the year with the lowest number of ratings, 1917, had 32 ratings. Across years, the average rating score did not appear to vary greatly. The lowest

[5]Note that movie-user combinations are represented multiple times in Figure 8. For example, the movie, Boomerang (1992), which is categorized under the genres, Comedy|Romance, and was rated 5 by user 1 will be counted twice, under both individual genres *Comedy* and *Romance,* for this user.

average rating score per year was 3.28 in 1919, while the highest was 4.05 in the years 1934 and 1946 (Figure 9). The CV for the number of ratings was 1.62 and the CV for the average rating score was 0.06.

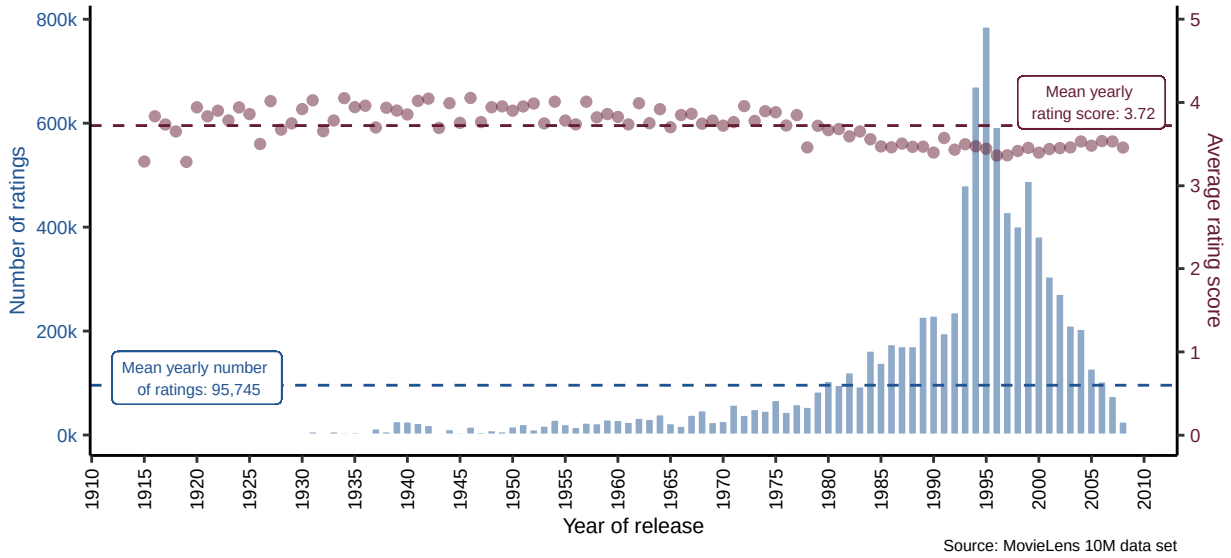


Figure 9: Distribution of number of ratings and average rating scores by movie release year

2.1.6 User rating date and time

The *timestamp* column contained the date and time when a particular user rated a particular movie. The units shown are in seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970 (e.g., 838985046). These values were converted to a more understandable date and time format (i.e., 1996-08-04). The earliest rating occured on 1995-01-08 and the final rating on 2009-01-04.

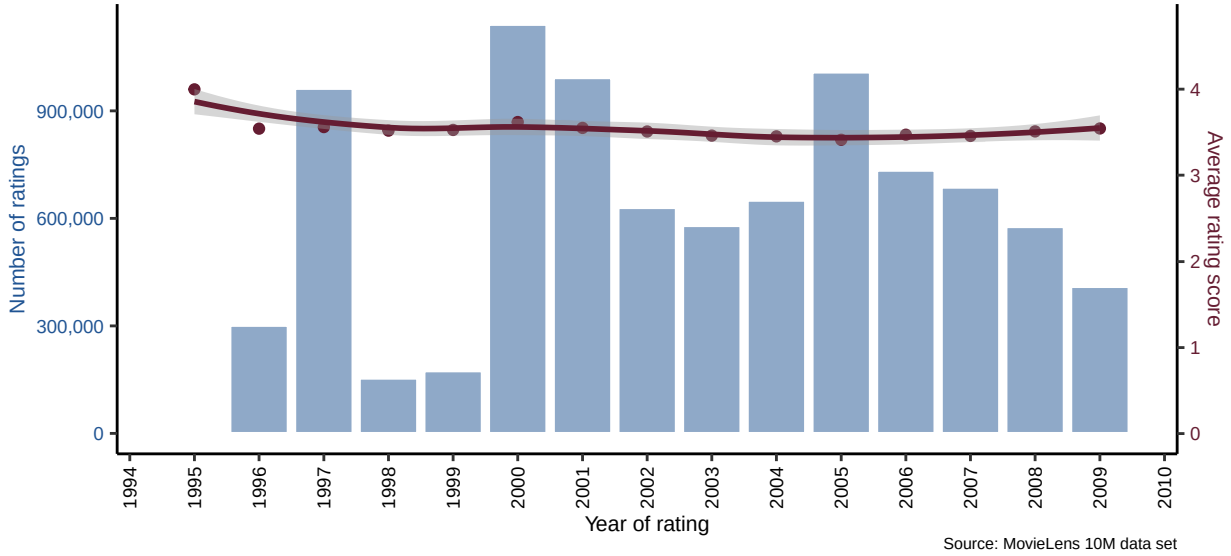


Figure 10: Distribution of number of ratings and average rating scores per rating year

Across rating years the number of ratings varied greatly with the highest number of ratings (1,140,977)

observed in 2000, and the lowest (2) in 1995. Average rating scores across rating years also varied with the highest rating score, 4, observed in 1995 and the lowest, 3.41, in 2005 (Figure 10). The CV for the number of ratings was 0.57 and the CV for the average rating score was 0.04.

2.1.7 Insights gained

The exploratory data analysis section above revealed variation in the number of ratings and the average rating scores across movies, users, genres, movie release year and user rating year. The CV for number of ratings and average rating score for each of these independent variables is shown in Table 4. Based on this table, the highest variability in rating score appeared across *movie*, *user*, and *genre - original combinations*, i.e., these were the variables with the highest CV for average rating score. As a result, these 3 variables were chosen to be added to the modeling approach. As the other variables had relatively lower CV for average rating score, they were excluded from the algorithm.

Table 4: Comparison of CV for number of ratings and average rating scores for all independent variables

Variable	CV (Number of ratings)	CV (Average rating score)
Movie	**2.660**	**0.1800**
User	**1.510**	**0.1190**
Genre - original genre categories	**4.010**	**0.1420**
Genre - number-of-genres categories	1.030	0.0227
Genre - Individual genres	0.941	0.0488
Release year	1.620	0.0600
Rating date	0.570	0.0400

2.2 Modeling Approach

A multivariable linear regression modeling approach was chosen for prediction, given the modest target RMSE of < 0.86490. Because of the sheer size of the MovieLens 10M data set, fitting a linear model in R, using the `lm()` function, was not feasible and was eschewed for all models. Instead, approximations were computed for each of the added variable effects.

2.2.1 *Train* and *test* sets

The algorithm was developed using *edx* data set, which was split into 2: i) *train* set, containing *80%* of *edx* data and used to generate the algorithm, and ii) *test* set, containing *20%* of *edx* data and used for initial testing of the algorithm (see Figure 1). Splitting was done in a manner that ensured that the distribution of rating scores (stored in the *edx* outcome column, *rating)* in the *train* and *test* sets were balanced. An overview of the resulting 2 data sets is shown in Table 5.

Table 5: Dimensions and proportions of train and test sets

	Number of rows	Proportion of edx
Train set	7,200,081	80 %
Test set	1,799,967	20 %

2.2.2 Loss Function

To assess the algorithm's adequacy to model the data, the accuracy-based measure, **root/residual mean square error (RMSE)**, was used as the loss function.

$$RMSE = \sqrt{\frac{1}{N}\sum_{u,m}(\hat{Y}_{u,m} - Y_{u,m})^2}$$

where:

$Y_{u,m}$ = rating for movie, *m*, by user, *u*

$\hat{Y}_{u,m}$ = predicted rating

N = number of user-movie combinations in the *test set* (when developing the algorithm) or *validation set* (when evaluating the final algorithm).

The goal for the project was to create a model which predicted ratings in the *validation* set with an RMSE of < 0.86490.

2.2.3 Developing the algorithm

2.2.3.1 Baseline model The baseline model of the algorithm was a naive model that set the predicted rating for each movie as *the average rating for all users across all movies*.

$$Y_{u,m} = \mu + \varepsilon_{u,m}$$

where:

$Y_{u,m}$ = rating for movie, *m*, by user, *u*

μ = average movie rating for all movies

$\varepsilon_{u,m}$ = independent errors sampled from the same distribution centred at 0.

To minimize the RMSE, the least squares estimate (LSE) of μ - i.e., the average rating across all movies and users - $\hat{\mu}$, was computed as follows:

$$\hat{\mu} = \frac{1}{N}\sum_{u,m} Y_{u,m}$$

where:

N = total number of user-movie combinations

$Y_{u,m}$ = actual rating for movie *m* by user *u*

2.2.3.2 Baseline + Movie effects model A movie effects term was added to the baseline model as follows:

$$Y_{u,m} = \mu + b_m + \varepsilon_{u,m}$$

where:

$Y_{u,m}$ = rating for movie, m, by user, *u*

μ = average movie rating for all movies

b_m = movie-specific effects

$\varepsilon_{u,m}$ = independent errors

The "movie effect" for movie *m*, denoted as $\hat{b}_m$, was estimated as follows:

$$\hat{b}_m = \frac{1}{N_m}\sum_{u=1}^{N_m}(Y_{u,m} - \hat{\mu})$$

where:

N_m = number of users that rated movie *m*

$Y_{u,m}$ = actual rating for movie *m* by user *u*

$\hat{\mu}$ = average rating across all movies and users estimated in the Baseline model section

2.2.3.3 Baseline + Movie + User effects model

A user effects term was incorporated into the model as follows:

$$Y_{u,m} = \mu + b_m + b_u + \varepsilon_{u,m}$$

where:

$Y_{u,m}$ = rating for movie, m, by user, *u*

μ = average movie rating for all movies

b_m = movie-specific effects

b_u = user-specific effects

$\varepsilon_{u,m}$ = independent errors

The "user effect" for user *u*, denoted as b_u, was estimated using the equation:

$$\hat{b}_u = \frac{1}{N_u}\sum_{m=1}^{N_u}(Y_{u,m} - \hat{\mu} - \hat{b}_m)$$

where:

N_u = number of movies rated by user *u*

$Y_{u,m}$ = actual rating for movie *m* by user *u*

$\hat{\mu}$ = average rating across all movies and users estimated in the Baseline model section

$\hat{b}_m$ = movie effect estimates computed in the Baseline + Movie effects model section

2.2.3.4 Baseline + Movie + User + Genre effects model (Original genre categories)

The original genre categories were utilised for this model. The genre effects term was added to the model as follows:

$$Y_{u,m} = \mu + b_m + b_u + b_{g[m]} + \varepsilon_{u,m}$$

where:

$Y_{u,m}$ = rating for movie, m, by user, *u*

μ = average movie rating for all movies

b_m = movie-specific effect

b_u = user-specific effect

$b_{g[m]}$ = genre-specific effect for the genre(s) associated with movie *m*

$\varepsilon_{u,m}$ = independent errors

The "genre effect" for genre *g*, denoted as $\hat{b}_g$, was estimated using the following equation:

$$\hat{b}_g = \frac{1}{N_g} \sum_{r=1}^{N_g} (Y_{u[r],m[r]} - \hat{\mu} - \hat{b}_{m[r]} - \hat{b}_{u[r]})$$

where:

N_g = number of ratings for all movies of genre *g*

$Y_{u[r],m[r]}$ = actual rating for the movie by the user in row *r*

$\hat{\mu}$ = average rating across all movies and users estimated in the Baseline model section

$\hat{b}_{m[r]}$ = movie effect estimates computed in the Baseline + Movie effects model section

$\hat{b}_{u[r]}$ = user effect estimates computed in the Baseline + Movie + User effects model section

2.2.3.5 Regularisation

Exploratory analysis of the *edx* data set revealed wide variation in rating across predictor variables. In some cases, small sample sizes were observed for predictors. For example, some movies were only rated by a single user, some users only rated a few movies, and some genres/genre-combinations only had a handful of movie-user combinations associated with them. These small sample sizes tend to yield larger, more uncertain, effect estimates — both negative and positive (Irizarry 2022). These, in turn, tend to result in a model that is over-fitted, i.e., one that predicts well the data set used to train it but under-performs on the *validation (final hold-out) set* or any other new data set. Regularisation is an approach that can help to mitigate this by penalising such large estimates (Irizarry 2022). Ridge regression is an example of a regularisation approach, which minimizes the penalized residual sum of squares by adding a penalty term to the least squares equation (Hastie et al. 2009). This penalty term includes a tuning parameter, λ, that shrinks the estimates towards zero.

The **Baseline + Movie + User + Genre effects model** was regularised as by minimizing the following:

$$\frac{1}{N} \sum_{u,m} (Y_{u,m} - \mu - b_m - b_u - b_{g[m]})^2 + \lambda (\sum_m b_m^2 + \sum_u b_u^2 + \sum_g b_{g[m]}^2)$$

where:

$Y_{u,m}$ = rating for movie, m, by user, *u*

μ = average movie rating for all movies

b_m = movie-specific effect

b_u = user-specific effect

$b_{g[m]}$ = genre-specific effect for the genre(s) associated with movie *m*

N = number of user-movie combinations

λ = tuning parameter

An example of how small sample sizes result in larger estimates and how this is mitigated by regularisation is shown in the Example of estimate shrinkage section.

2.2.3.5.1 Tuning parameter (λ) The optimal value of the tuning parameter, λ, is commonly selected using cross-validation (Serang et al. 2017). Here the optimal λ was the value that minimized RMSE in the *test set*.

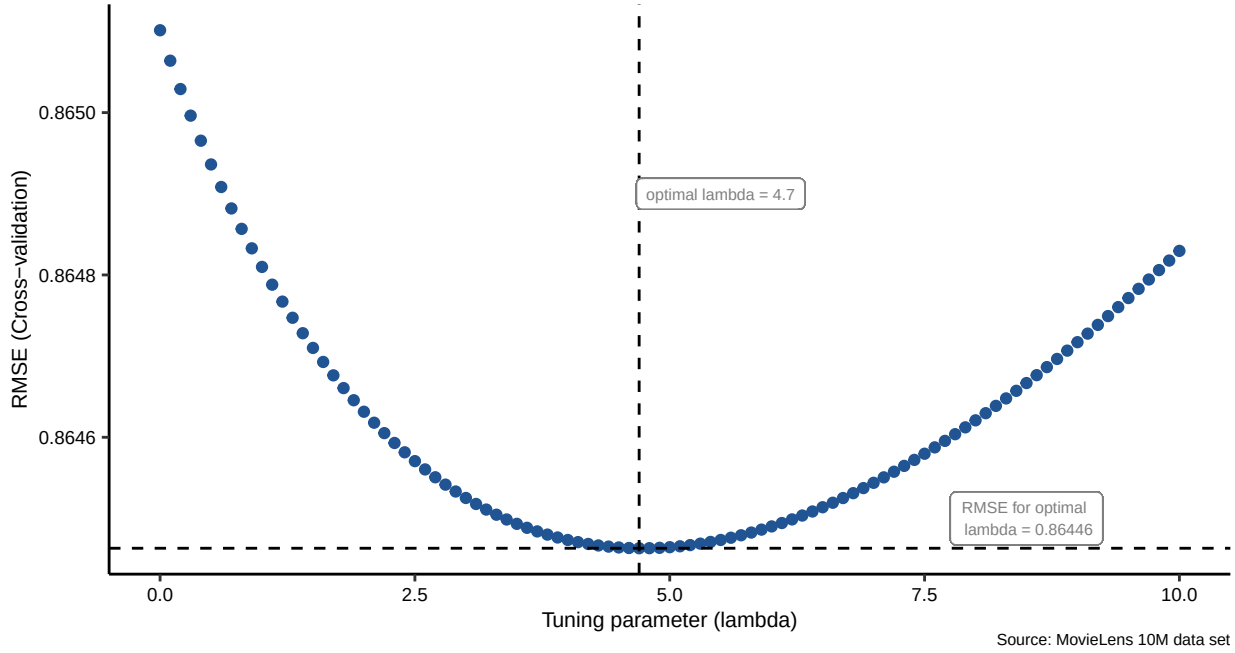


Figure 11: Optimal tuning parameter (lambda)

The optimal λ chosen was 4.7 as shown in Figure 11.

2.2.3.5.2 Example of estimate shrinkage To illustrate how regularisation mitigates large estimates resulting from small samples, non-regularised and regularised movie effect estimates were calculated (for the *train set* data) before and after regularisation, respectively, using the following equations:

Non-regularised movie effects:

$$\hat{b}_m = \frac{1}{N_m}\sum_{u=1}^{N_m}(Y_{u,m} - \hat{\mu})$$

Regularised movie effects:

$$\hat{b}_m(\lambda) = \frac{1}{\lambda + N_m}\sum_{u=1}^{N_m}(Y_{u,m} - \hat{\mu})$$

These values were compared in Figure 12, which shows the regularised and non-regularised movie effect estimates. Movies with higher number of ratings, N_m, (shown as large cirlces) show less sensitivity to regularisation compared to those with lower (smaller circles).

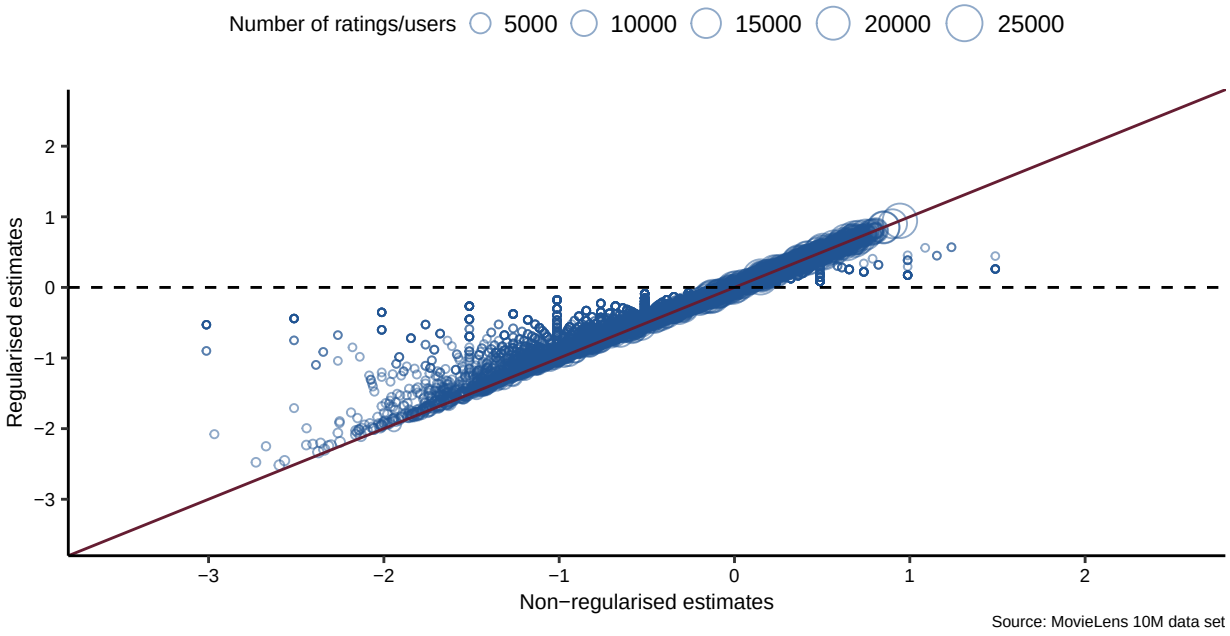


Figure 12: Comparison between regularised and non-regularised movie effect estimates by sample sizes (number of ratings/users)

2.2.3.6 Final algorithm validation The algorithm was evaluated on the *validation* data set to assess how well the final algorithm performed on new data. This entailed:

1. estimating the **selected model's** effects using the full *edx* data set;
2. predicting ratings in the *validation* set and;
3. calculating a final RMSE comparing the predicted ratings to the actual user ratings in the *validation* set.

3 Results

3.1 Modeling results

Movie, user and genre effect estimates were computed and graphed to illustrate the substantial variability (Figure 13). Examples of effect estimates calculations and the resulting variability are shown below[6].

Movie effects:

Movie effect estimate calculations for movies *Titanic* and *Tokyo!* were compared as follows:

Titanic: Rated by 14,766 users.

$$\hat{b}_{Titanic} = \frac{1}{N_{Titanic}} \sum_{u=1}^{N_{Titanic}} (Y_{u,Titanic} - \hat{\mu}) = \frac{1}{14766} \sum_{u=1}^{14766} (Y_{u,Titanic} - 3.512) = -0.195$$

Tokyo!: Rated 4.5 by only 1 user (*userId* 27006).

$$\hat{b}_{Tokyo!} = \frac{1}{N_{Tokyo!}} \sum_{u=1}^{N_{Tokyo!}} (Y_{27006,Tokyo!} - \hat{\mu}) = \frac{1}{1} \sum_{u=1}^{1} (4.5 - 3.512) = 0.988$$

Variability in movie effects reflects variability in rating scores around the overall mean. Note that given the average overall movie rating (for *train set*) is 3.51, for *Tokyo!*, a movie effect, $b_{Tokyo!}$, of 0.988 means a 4.5-star rating (i.e., 3.51 + 0.988 = 4.5).

User effects:

User effect estimate calculations for 2 users (*userIds* 59269 and 22170) were compared as follows:

59269: Rated 6,616 movies.

$$\hat{b}_{59269} = \frac{1}{N_{59269}} \sum_{m=1}^{N_{59269}} (Y_{59269,m} - \hat{\mu} - \hat{b}_m) = \frac{1}{6616} \sum_{m=1}^{6616} (Y_{59269,m} - 3.512 - \hat{b}_m) = -0.0000756$$

22170: Rated 12 movies.

$$\hat{b}_{22170} = \frac{1}{N_{22170}} \sum_{m=1}^{N_{22170}} (Y_{22170,m} - \hat{\mu} - \hat{b}_m) = \frac{1}{12} \sum_{m=1}^{12} (Y_{22170,m} - 3.512 - \hat{b}_m) = 0.0417$$

Genre effects (Original genre categories):

Genre effect estimate calculations for genres *Action|Animation|Comedy|Horrror* and *Drama* were compared below:

Drama: Consists of 1,815 distinct movies and 62,724 distinct users.

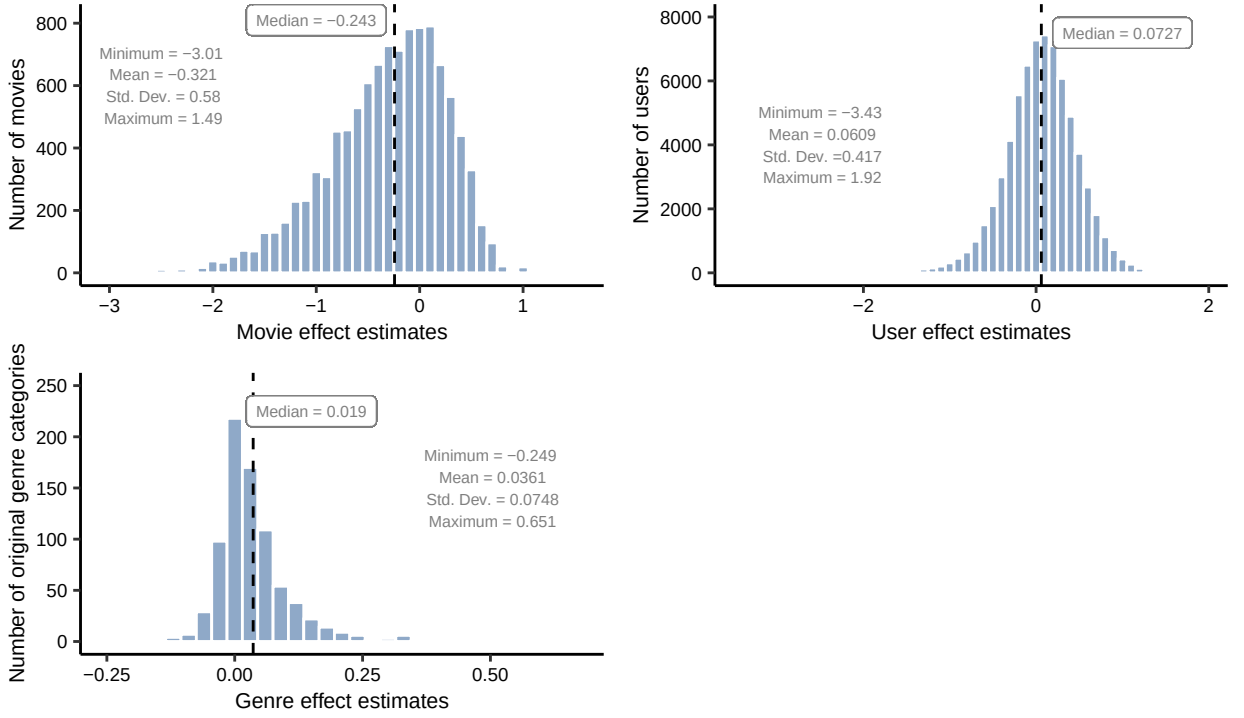
[6] These calculations were performed using data from e*dx* and without regularisation.

$$\hat{b}_{Drama} = \frac{1}{N_{Drama}} \sum_{m=1}^{N_{Drama}} (Y_{u,m} - \hat{\mu} - \hat{b}_m - \hat{b}_u) = \frac{1}{1815} \sum_{m=1}^{1815} (Y_{u,m} - 3.512 - \hat{b}_m - \hat{b}_u) = -0.000000828$$

Action|Animation|Comedy|Horrror: Consists of 1 distinct movie (*movieId* 62008) and 2 users (*userId* 3149 and 27946). The genre is denoted as '*aach*' in the formula below.

$$\hat{b}_{aach} = \frac{1}{N_{aach}} \sum_{m=1}^{N_{aach}} (Y_{u,m} - \hat{\mu} - \hat{b}_m - \hat{b}_u) = \frac{1}{2} \sum_{m=1}^{2} (Y_{u,m} - 3.512 - \hat{b}_m - \hat{b}_u) = 0.304$$

Note that the genre with the smaller sample size of 2 users/ratings, *Action|Animation|Comedy|Horrror*, resulted in a larger estimate compared to the genre with the larger sample size, *Drama*.



Source: MovieLens 10M data set

Figure 13: Distribution of effect estimates for movies, users and original genre categories

For each of the models used to develop the final algorithm, an RMSE value was computed. The results are shown in Table 6. The final regularised model was optimized using the tuning parameter (λ) of 4.7.

3.2 Final algorithm validation

The final model, **regularised baseline + movie + user + genre effects model (original genre categories)**, optimized using the optimal λ of 4.7, underwent a final hold-out test, where it was used to predict ratings in the *validation* set. The resulting RMSE is shown in Table 7.

Table 6: Comparison of RMSE results from different modeling approaches

Modeling Approach	RMSE
Target	**0.86490**
Baseline model	1.05956
Baseline + Movie Effect Model	0.94293
Baseline + Movie + User Effect Model	0.86545
Baseline + Movie + User + Genre Effect Model	0.86510
Regularised Baseline + Movie + User + Genre Effect Model	**0.86446**

Table 7: Comparison of RMSEs from final model RMSE to project target

Modeling Approach	RMSE	Difference
Target	0.86490	0
Final (Validation) Model	0.86445	-0.052%

The RMSE for the final model was 0.86445, which translated into a 0.052% improvement over the project target.

4 Conclusion

This project aimed to develop a movie recommendation system for the MovieLens 10M data set by using e*dx* data set — divided into *train* (80%) and *test* (20%) sets — to train a machine learning algorithm and using the final algorithm to predict movie ratings in the *validation* data set. The final algorithm, a regularised multivariable linear regression model, which adjusted for movie, user and genre effects (original genre categories), achieved RMSEs of 0.86446 in the *edx test* set and 0.86445 in the *validation* (final hold out) set. The latter was 0.052% below the project objective.

Given that the aim of this project was to target an RMSE below 0.86490, a simple regularised multivariable linear regression model was utilized. However, because of the current extensive use of different types of recommender systems, several more complex algorithms, for example matrix factorisation, which generate much lower RMSE values have been implemented (Koren, Bell, and Volinsky 2009) (Takács et al. 2008) (Fathan, Adji, and Ferdiana 2018).

4.1 Limitations and future work

The greatest limitation faced during this project was the inability to run some operations in R on a standard laptop given the sheer size of the data set. For example, fitting a linear model in R, e.g., using the `lm()` function, for each of the above-mentioned models would have been very time consuming and likely impossible in terms of memory for a standard personal computer.

The final model contained only the movie, user, and genre variables. Future work could explore inclusion of all features present in the data set used in this report (year of movie release and date and time of rating), using predictors derived from these features, as well as including other features related to this data that were not included in the data set (e.g., tags — words or phrases about movies created by users). Additionally, future projects could explore the use of other measures of accuracy other than RMSE, such as Mean of absolute value of errors (MAE), or even combinations of these measures.

This report outlines a simple model that can predict the rating, of a particular movie in the MovieLens 10M data set, by a particular user, also in the data set. Thus, it does not provide actual movie recommendations. Future work can explore how a model such as this one could be implemented to provide movie recommendations.

References

Allaire, JJ, Yihui Xie, Jonathan McPherson, Javier Luraschi, Kevin Ushey, Aron Atkins, Hadley Wickham, Joe Cheng, Winston Chang, and Richard Iannone. 2022. *Rmarkdown: Dynamic Documents for r*. https://github.com/rstudio/rmarkdown.

Auguie, Baptiste. 2017. *gridExtra: Miscellaneous Functions for "Grid" Graphics*. https://CRAN.R-project.org/package=gridExtra.

Bøe, Cecilie. 2007. "Collaborative Filtering for Recommending Movies." Master's thesis, Institutt for datateknikk og informasjonsvitenskap.

Bühlmann, Peter, and Sara Van De Geer. 2011. *Statistics for High-Dimensional Data: Methods, Theory and Applications*. Springer Science & Business Media.

Dowle, Matt, and Arun Srinivasan. 2021. *Data.table: Extension of 'Data.frame'*. https://CRAN.R-project.org/package=data.table.

Fathan, Gess, Teguh Bharata Adji, and Ridi Ferdiana. 2018. "Impact of Matrix Factorization and Regularization Hyperparameter on a Recommender System for Movies." In *2018 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, 113–16. IEEE.

Garbett, Shawn P, Jeremy Stephens, Kirill Simonov, Yihui Xie, Zhuoer Dong, Hadley Wickham, Jeffrey Horner, et al. 2022. *Yaml: Methods to Convert r Data to YAML and Back*. https://CRAN.R-project.org/package=yaml.

Grolemund, Garrett, and Hadley Wickham. 2011. "Dates and Times Made Easy with lubridate." *Journal of Statistical Software* 40 (3): 1–25. https://www.jstatsoft.org/v40/i03/.

Hastie, Trevor, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Vol. 2. Springer.

Irizarry, Rafael. 2022. *Introduction to Data Science: Data Analysis and Prediction Algorithms with r*. https://rafalab.github.io/dsbook/.

Koren, Yehuda, Robert Bell, and Chris Volinsky. 2009. "Matrix Factorization Techniques for Recommender Systems." *Computer* 42 (8): 30–37.

Kuhn, Max. 2022. *Caret: Classification and Regression Training*. https://CRAN.R-project.org/package=caret.

Lovie, Pat. 2005. "Coefficient of Variation." *Encyclopedia of Statistics in Behavioral Science*.

Pu, Pearl, Li Chen, and Rong Hu. 2011. "A User-Centric Evaluation Framework for Recommender Systems." In *Proceedings of the Fifth ACM Conference on Recommender Systems*, 157–64.

R Core Team. 2022. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Serang, Sarfaraz, Ross Jacobucci, Kim C Brimhall, and Kevin J Grimm. 2017. "Exploratory Mediation Analysis via Regularization." *Structural Equation Modeling: A Multidisciplinary Journal* 24 (5): 733–44.

Silveira, Thiago, Min Zhang, Xiao Lin, Yiqun Liu, and Shaoping Ma. 2019. "How Good Your Recommender System Is? A Survey on Evaluations in Recommendation." *International Journal of Machine Learning and Cybernetics* 10 (5): 813–31.

Takács, Gábor, István Pilászy, Bottyán Németh, and Domonkos Tikk. 2008. "Matrix Factorization and Neighbor Based Algorithms for the Netflix Prize Problem." In *Proceedings of the 2008 ACM Conference on Recommender Systems*, 267–74.

Tierney, Nicholas, Di Cook, Miles McBain, and Colin Fay. 2021. *Naniar: Data Structures, Summaries, and Visualisations for Missing Data*. https://CRAN.R-project.org/package=naniar.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. "Welcome to the tidyverse." *Journal of Open Source Software* 4 (43): 1686. https://doi.org/10.21105/joss.01686.

Wickham, Hadley, and Dana Seidel. 2022. *Scales: Scale Functions for Visualization*. https://CRAN.R-project.org/package=scales.

Xie, Yihui. 2022a. *Knitr: A General-Purpose Package for Dynamic Report Generation in r*. https://yihui.org/knitr/.

———. 2022b. *Tinytex: Helper Functions to Install and Maintain TeX Live, and Compile LaTeX Documents*. https://github.com/rstudio/tinytex.

Zhou, Yunhong, Dennis Wilkinson, Robert Schreiber, and Rong Pan. 2008. "Large-Scale Parallel Collaborative Filtering for the Netflix Prize." In *International Conference on Algorithmic Applications in Management*, 337–48. Springer.

Zhu, Hao. 2021a. *kableExtra: Construct Complex Table with 'Kable' and Pipe Syntax*. https://CRAN.R-project.org/package=kableExtra.

———. 2021b. "Create Awesome Latex Table with Knitr::kable and Kableextra." https://cran.rstudio.com/web/packages/kableExtra/vignettes/awesome_table_in_pdf.pdf.