MovieLens Recommendation System

Capstone Project 1

Author – Amol Chaudhari

Date – 22/06/2025

Contents

1 Introduction 3

2 ExploratoryAnalysis 4

3 Methods 11

4 Results 21

5 Conclusions 22

References

1 Introduction

The MovieLens datasets have provided a popular environment for experimentation with machine learning since their launch in 1997 (Harper and Konstan 2015).
The goal of this project was to develop a recommendation system using one of the MovieLens datasets which consists of 10 million movie ratings. To facilitate this work, the dataset was split into a training set (edx) and a final hold-out test set (validation) using code provided by the course organisers. The objective was for the final algorithm to predict ratings with a root mean square error (RMSE) of less than 0.86490 versus the actual ratings included in the validation set.

This report sets out the exploratory analysis of the data using common visualisation techniques followed by the methods used to develop, train and test the algorithm before providing and discussing the results from each iteration of the algorithm and concluding on the outcome of the final model, its limitations and potential for future work.

The report was compiled using R Markdown in RStudio, an integrated development environment for programming in R, a language and software environment for statistical computing.

2 Exploratory Analysis

Table 1: edx dataset: variable class and first 5 rows

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

2.1 Ratings ($rating)

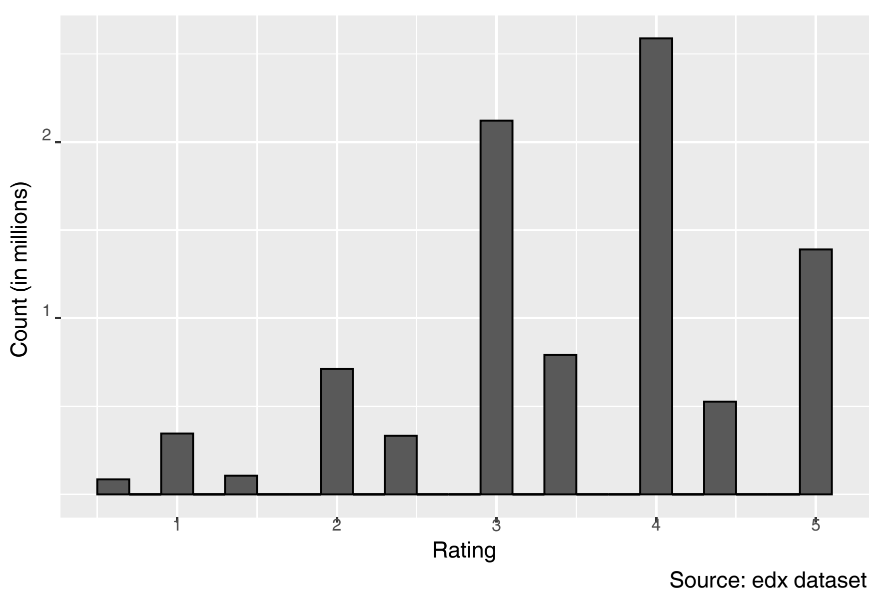

Figure 1: Overall ratings distribution

2.2 Movies($movieId)

Unsurprisingly, some movies are more highly rated than others (see Figure 2). Further analysis reveals significant variation in the number of ratings received by each movie (see Figure 3), with the movie with the most ratings, Pulp Fiction (1994), receiving a total of 31362 ratings whereas as many as 126 movies were only rated once. There is clearly a movie effect on the rating awarded and, as such, adjusting for this effect (or bias) was considered worthwhile for inclusion in the training algorithm.

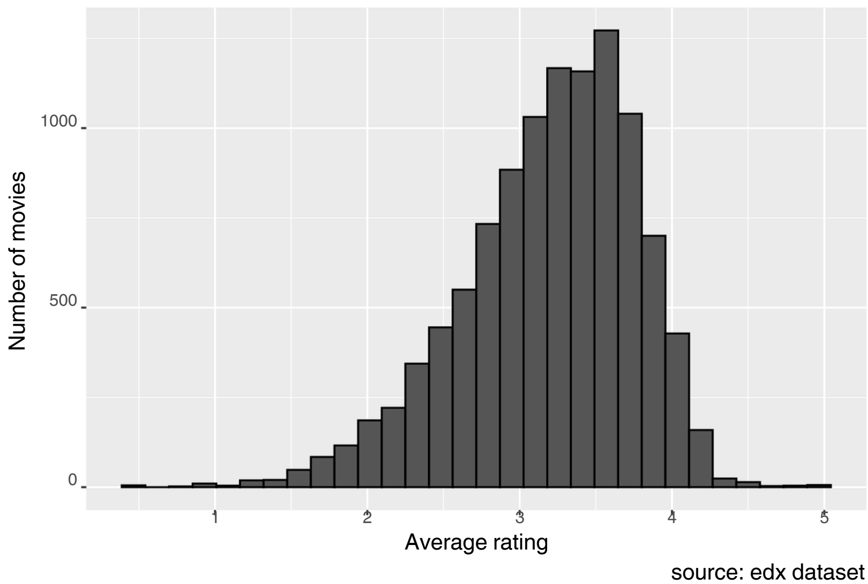

Figure 2: Movie distribution by average rating

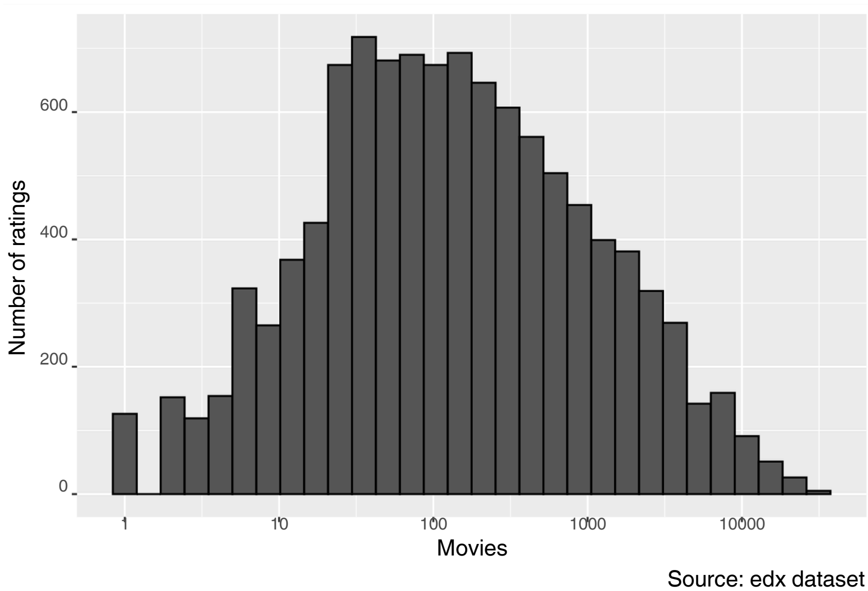

Figure 3: Number of ratings by movie

2.3 Users($userId)

Exploration of user data revealed a similar pattern to that observed for movies, with some users appearing more generous in the way they assessed movies, having provided higher ratings than others (see Figure 4). Some users contributed many more ratings than other users (Figure 5). For example, one user provided a total of 6616 ratings whereas as many as 1059 provided fewer than 10 movie ratings each. This analysis identifies a clear user effect (or bias) which, if adjusted for, may further improve the accuracy of a movie recommendation system.

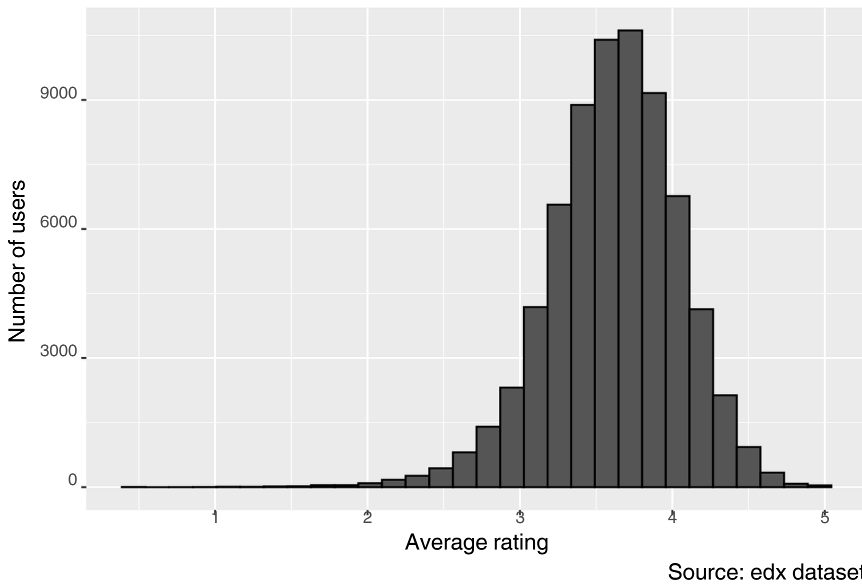

Figure 4: User distribution by average rating

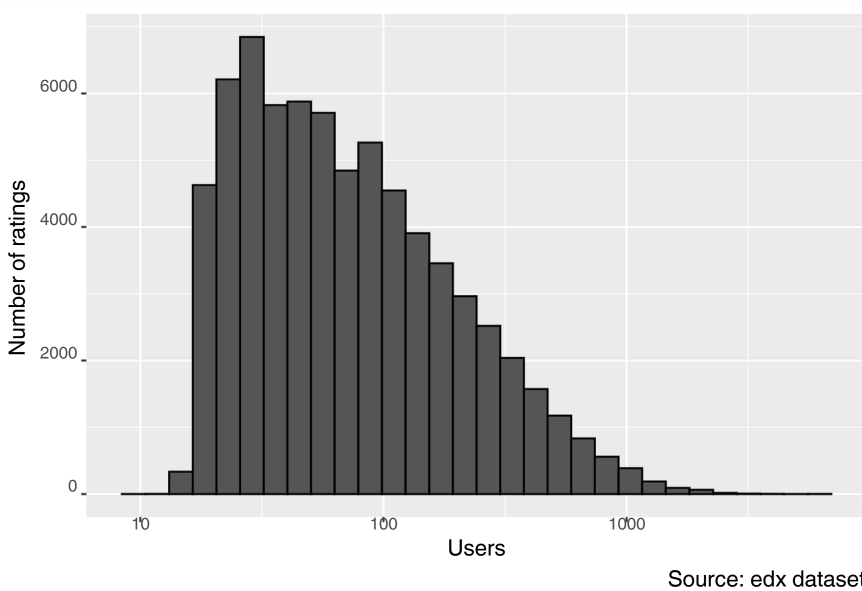

Figure 5: Number of ratings by user

2.4 MovieGenre($genres)

As was shown in Table 1 above, the genres variable provides the names of the genres which each movie is identified with. For example, Boomerang (1992) in the first row of the edx dataset includes Comedy|Romance in the genre variable. Some movies were assigned to more than one genre and there were a total of 797 unique genre combinations included in the dataset. Separating these combinations into rows with single genres, it was possible to identify 20 different genre categories (including “no genre listed”) and to rank these by the number of ratings (Table 2).

Table 2: Individual genres ranked by number of ratings

Genre	No.ofRatings	Ave.Rating
Drama	3910127	3.67
Comedy	3540930	3.44
Action	2560545	3.42
Thriller	2325899	3.51
Adventure	1908892	3.49
Romance	1712100	3.55
Sci-Fi	1341183	3.40
Crime	1327715	3.67
Fantasy	925637	3.50
Children	737994	3.42
Horror	691485	3.27
Mystery	568332	3.68
War	511147	3.78
Animation	467168	3.60
Musical	433080	3.56
Western	189394	3.56
Film-Noir	118541	4.01
Documentary	93066	3.78
IMAX	8181	3.77
(nogenreslisted)	7	3.64

Drama and comedy movies had the most ratings whereas Documentary and IMAX movies had the fewest ratings. Seven ratings were provided for movies for which no genre was listed. Table 2 also shows a variation in average rating by genre. Grouping the data by unique genre combinations and filtering only those genre combinations with at least 100,000 ratings in order to simplify the analysis for the purposes of visualisation, shows a clear effect of genre with ‘Comedy’ movies achieving the lowest average rating whereas ‘Crime|Drama’ and ‘Drama|War’ films achieving the highest rating (Figure 6). Clearly, there is merit in seeking to address this effect in training the algorithm for the movie recommendation system.

2.5 MovieTitle($title)

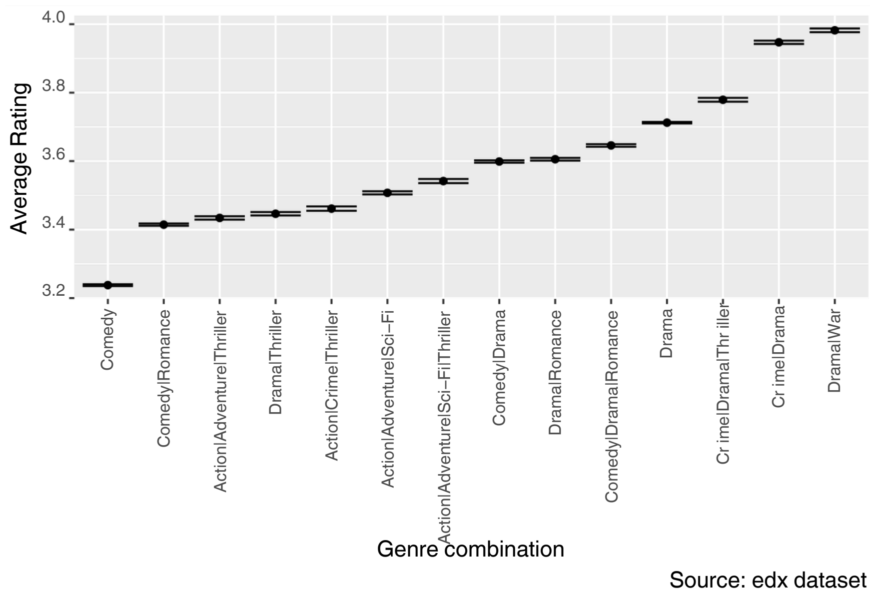

Figure 6: Average rating by genre

Table 3: Top 10 Movies by Number of Ratings

title	n
Pulp Fiction (1994)	31362
Forrest Gump (1994)	31079
Silence of the Lambs, The (1991)	30382
Jurassic Park (1993)	29360
Shawshank Redemption, The (1994)	28015
Braveheart (1995)	26212
Fugitive, The (1993)	25998
Terminator 2: Judgment Day (1991)	25984
Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)	25672
Apollo 13 (1995)	24284

However, as shown in Figure 8, there are very few ratings within the dataset assigned to movies released prior to 1970. The movies with the most ratings were released during the 1990s, peaking in 1995 with approximately 9% of the total number of ratings included in the edx dataset. Thus, as with other variables adjusting for the effect of release year should improve the accuracy of the training algorithm but taking account of the greater uncertainty in point estimates created by small sample sizes in some years would also be important.

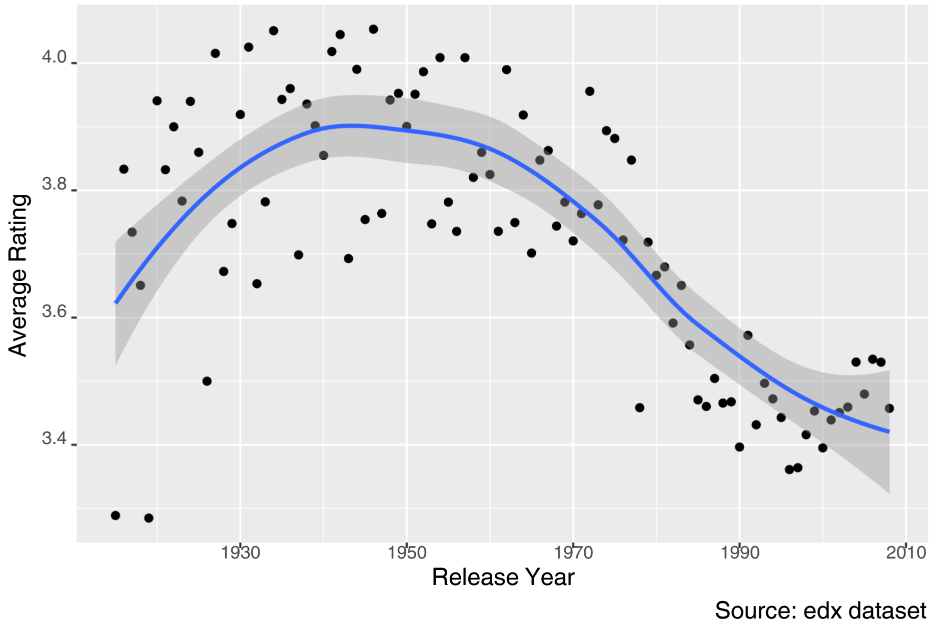

Figure 7: Average rating by year of release

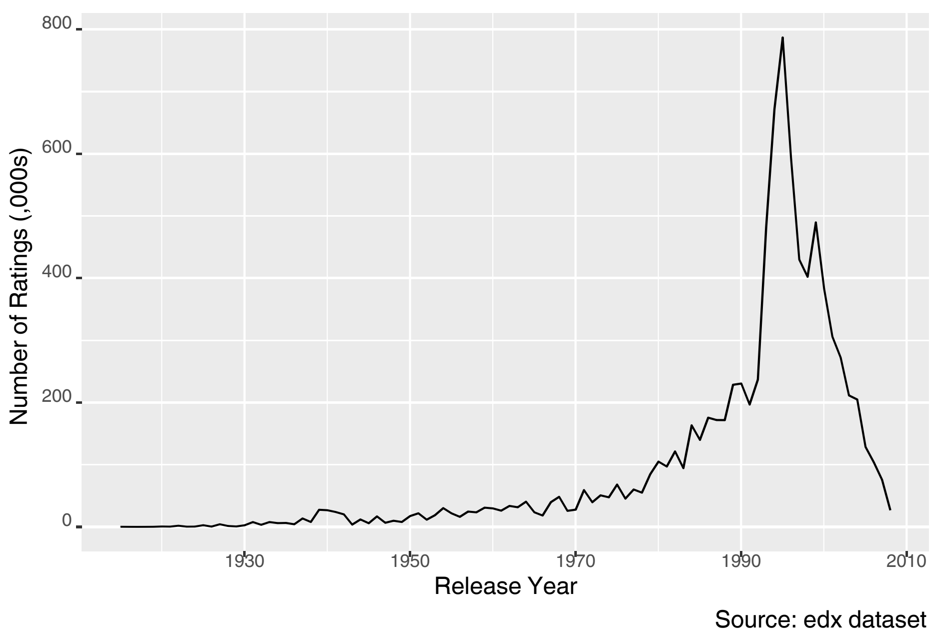

Figure 8: Number of ratings by year of release

Author - Amol Chaudhari

2.6 Dateofreview($timestamp)

The earliest review included in the dataset was completed in 1995. This was also when the average rating was highest, with a gradual decline in ratings observed until around 2005 after which average ratings began to increase. The effect of review date on the average rating was modest relative to that observed for movies and users but there was still some variation over time (see Figure 9), justifying its inclusion in the development of the recommendation algorithm.

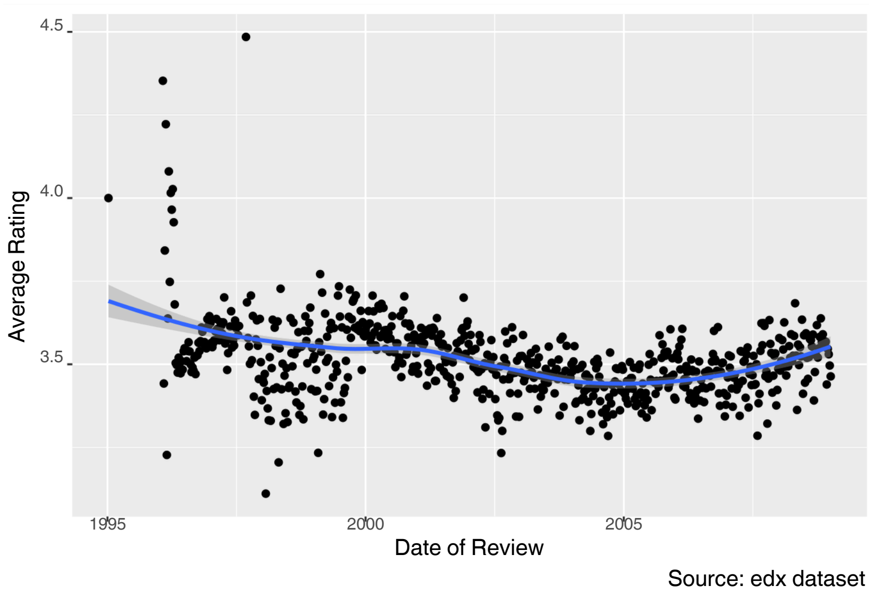

Figure 9: Average rating by date of review

3 Methods

3.1 Splitting edx out into train and test sets

This is important to allow for cross- validation and refinement of the final model without the risk of over-training. Other methods for cross-validation include K-fold cross validation and bootstrapping but were not utilised here (see Irizarry (2020) for further information).

‘

3.2 Calculating the error loss

The residual mean square error (RMSE) is defined as the standard deviation of the residuals (prediction errors) where residuals are a measure of spread of data points from the regression line (Glen 2020). The RMSE was calculated to represent the error loss between the predicted ratings derived from applying the algorithm 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.

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

Method	RMSE	Difference
Projectobjective	0.86490	-

3.3 Developingthealgorithm

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 $\varepsilon_{u,i}$, the independent errors sampled for the same distribution.

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

The average of all ratings is the estimate of μ that minimises the RMSE. Thus, $\hat{\mu}$ = mean(train_set$rating) was the simple formula used to train the first algorithm.

The exploratory analysis detailed in the previous section showed that ratings were not assigned equally across all movies. That is, some movies achieved a higher average rating than others and, accounting for this effect (or bias) will therefore improve the accuracy of the prediction. Thus, the training algorithm was further refined by taking into account the effect of movie on rating, bi.

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

A linear regression model would take some time to run given the large dataset involved. Instead, 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 within the training algorithm.

$$\hat{y}_{u,i} = \hat{\mu} + \hat{b}_i$$

.

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

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

Movie ratings were also dependent on genre, with some genres achieving higher average ratings than others. This effect was observed even when movies were allocated to multiple genres, as in the original dataset. Therefore, the rating for each movie and user was further refined by adjusting for genre effect, bg, and the least squares estimate of the genre effect, $\hat{b}_g$ calculated using the formula shown below.

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

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

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

$$\hat{b}_y = \text{mean}(\hat{y}_{u,i} - \hat{\mu} - \hat{b}_i - \hat{b}_u) - \hat{b}_g$$

There was also a small effect of the date of review (br)on the average rating awarded for each movie and user. The most appropriate way to incorporate this into the model would be to apply a smooth function to the day of release for each rating by movie and user. Rounding the date of review to the nearest week served to effectively smooth the data. The least squares estimate, taking into account the date of review effect, $\hat{b}_r$ was calculated using the formula shown below.

$$Y_{u,i} = \mu + (b_i + b_u + b_g + b_y + b_r + \varepsilon_{u,i}$$

$$\hat{b}_r = \text{mean}(\hat{y}_{u,i} - \hat{\mu} - \hat{b}_i - \hat{b}_u - \hat{b}_g - \hat{b}_y)$$

3.4 Regularising the algorithm

Regularisation is an effective method for penalising large effect estimates that are based on small sample sizes (Irizarry 2020). The penalty term, λ, is a tuning parameter chosen using cross-validation within the edx dataset. Thus, the movie effect, bi can be regularised to penalise these large effects as follows.

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

Based on the above, the least squares estimate for the regularised effect of movies can be calculated

as below, where n_i is the number of ratings made for movie i. The effect of $\lambda + n_i$ is such that when the sample size is large, i.e. ni is a big number, λ has little impact on the estimate, $\hat{b}_i(\lambda)$. On the other hand, where the sample size is small, i.e. ni is small, the impact of λ increases and the estimate shrinks towards zero.

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

Here, the regularisation model was developed to adjust for all of the effects previously described, as shown below. A range of values for λ (range: 4-6, with increments of 0.1) was applied in order to tune the model to minimise the RMSE value. As before, all tuning was completed within the edx dataset, using the train and test sets, so as to avoid over-training the model in the validation set.

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

3.5 Validating the final model

the final stage of the project was to train the algorithm using the full edx dataset and then to predict ratings within the validation dataset. Prior to doing this, it was necessary to incorporate the date of review and year of release variables in the validation set using the mutate function from the dplyr package.

The final model, adjusting for biases introduced by movie, user, genre, release year, review date,

4 Results

4.1 Simple average

Predicting the average rating from the train set (3.51) for every entry in the test set resulted in a RMSE of 1.06, substantially above the project objective. Moreover, an RMSE of 1.06 means that predicted ratings are more than 1 star away from the actual rating, an unacceptable error loss for a movie recommendation system.

Method	RMSE	Difference
Projectobjective	0.86490	-
Simpleaverage	1.06057	0.19567

4.2 Adjustingformovieeffects

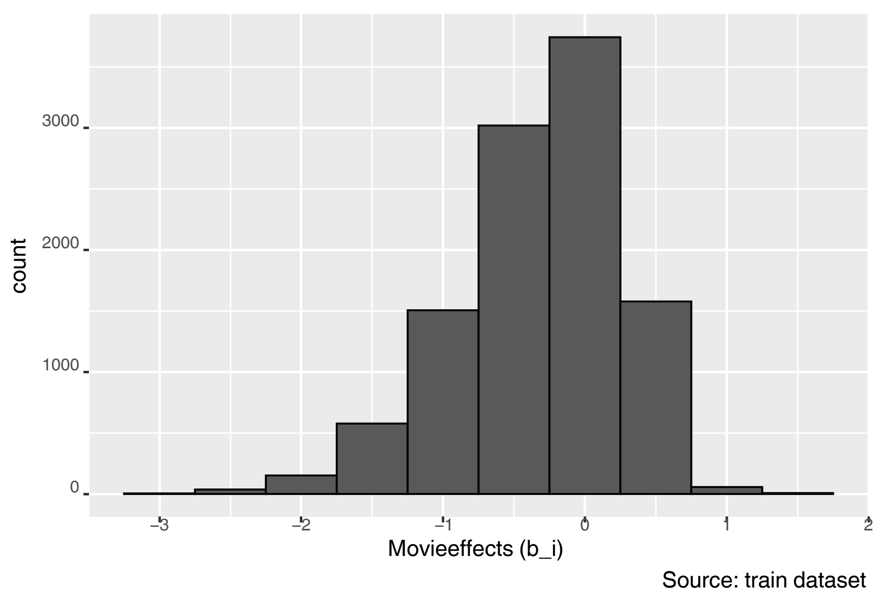

Figure 10: Distribution of movie effects

Metho d	RMSE	Difference
Project objective	0.86490	-
Simple average	1.06057	0.19567
Movie effects (b_i)	0.94371	0.07881

4.3 Adjusting for user effects

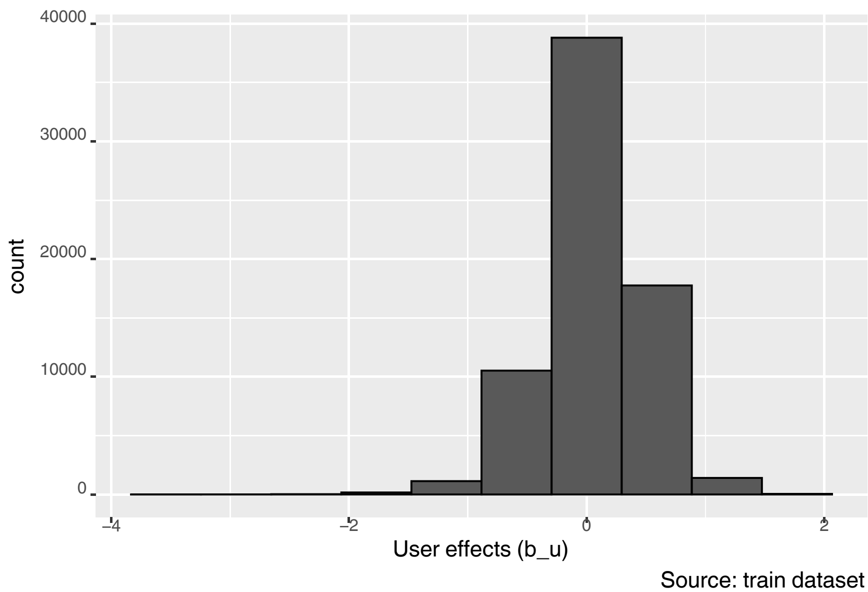

Figure 11: Distribution of user effects

Metho d	RMSE	Difference
Project objective	0.86490	-
Simple average	1.06057	0.19567
Movie effects (b_i)	0.94371	0.07881
Movie + User effects (b_u)	0.86617	0.00127

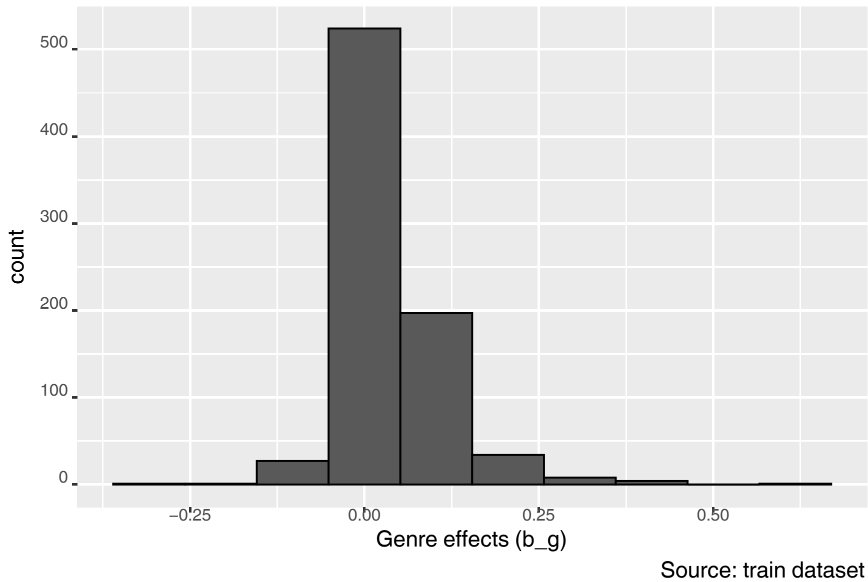

Figure 12: Distribution of genre effects

Method	RMSE	Difference
Projectobjective	0.86490	-
Simpleaverage	1.06057	0.19567
Movieeffects(b_i)	0.94371	0.07881
Movie+Usereffects(b_u)	0.86617	0.00127
Movie,UserandGenreeffects(b_g)	0.86582	0.00092

4.5 Adjusting for release year effects

incorporating this into the training algorithm yielded a modest incremental improvement of 0.02% in the accuracy of ratings prediction bringing the RMSE slightly closer to meeting the project objective at 0.86567.

Method	RMSE	Difference
Projectobjective	0.86490	-
Simpleaverage	1.06057	0.19567
Movieeffects(b_i)	0.94371	0.07881
Movie+Usereffects(b_u)	0.86617	0.00127
Movie,UserandGenreeffects(b_g)	0.86582	0.00092
Movie, User, Genre and Year effects (b_y)	0.86567	0.00077

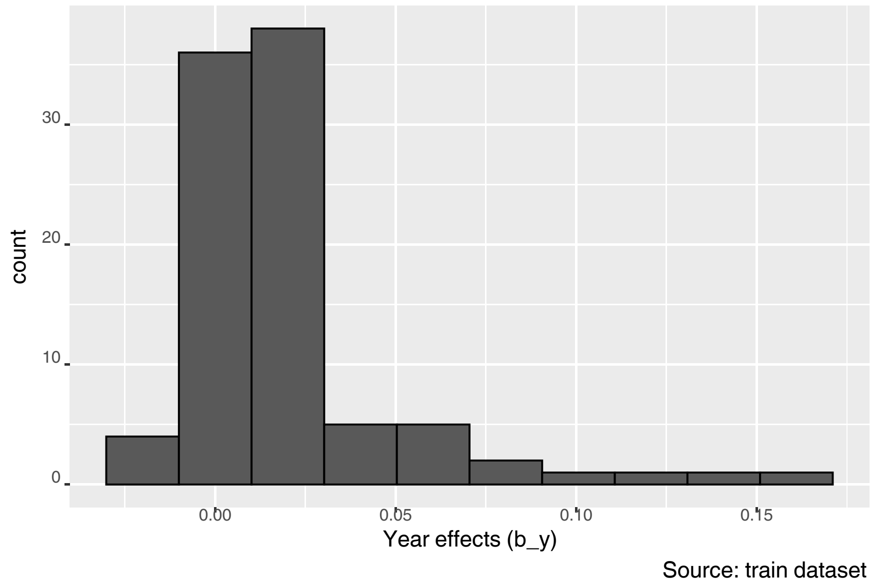

Figure 13: Distribution of release year effects

4.6 Adjustingforreviewdateeffects

The final bias to be adjusted for was the date of review. The exploratory analysis had shown that this had a small impact on ratings and this was confirmed by visualising the distribution of br in Figure 14.

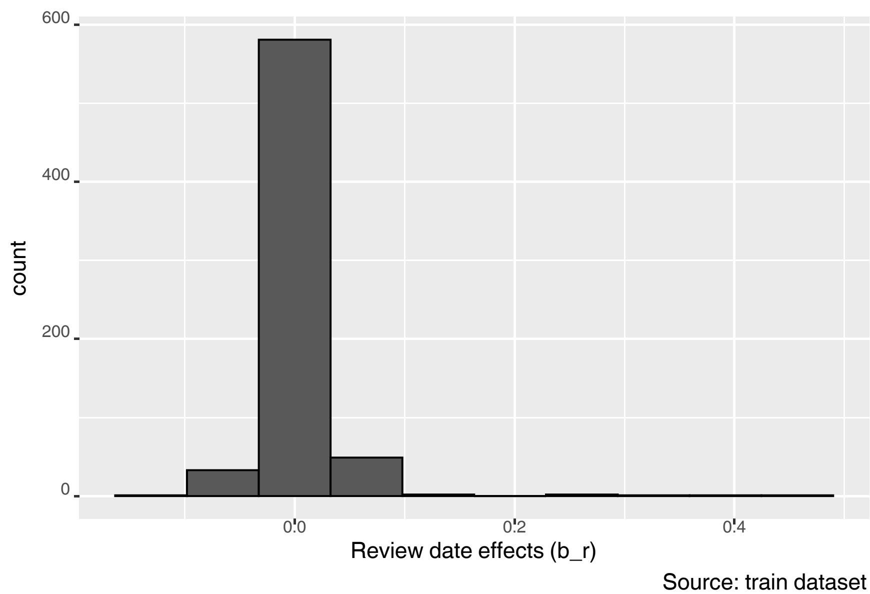

Figure 14: Distribution of review date effects

Adding the effect of review date into the algorithm delivered an RMSE of 0.86549, an improvement of 18.39% versus the original model but still not quite enough to meet the project objective.

Metho d	RMSE	Difference
Project objective	0.86490	-
Simple average	1.06057	0.19567
Movie effects (b_i)	0.94371	0.07881
Movie + User effects (b_u)	0.86617	0.00127
Movie, User and Genre effects (b_g)	0.86582	0.00092
Movie, User, Genre and Year effects (b_y)	0.86567	0.00077
Movie, User, Genre, Year and Review Date effects (b_r)	0.86549	0.00059

4.7 Effect of regularisation

The final step in developing the algorithm was to apply regularisation. Figure 15 shows the RMSE delivered across each of the values for λ tested. The optimum value for λ was 5.1 which minimised RMSE to 0.86483, which was just sufficient to surpass the target RMSE set as the project objective. This represented a total improvement of 18.46% in the accuracy of the model by adjusting for movie, user, genre, year of release and review date effects and applying regularisation to the combination of these effects.

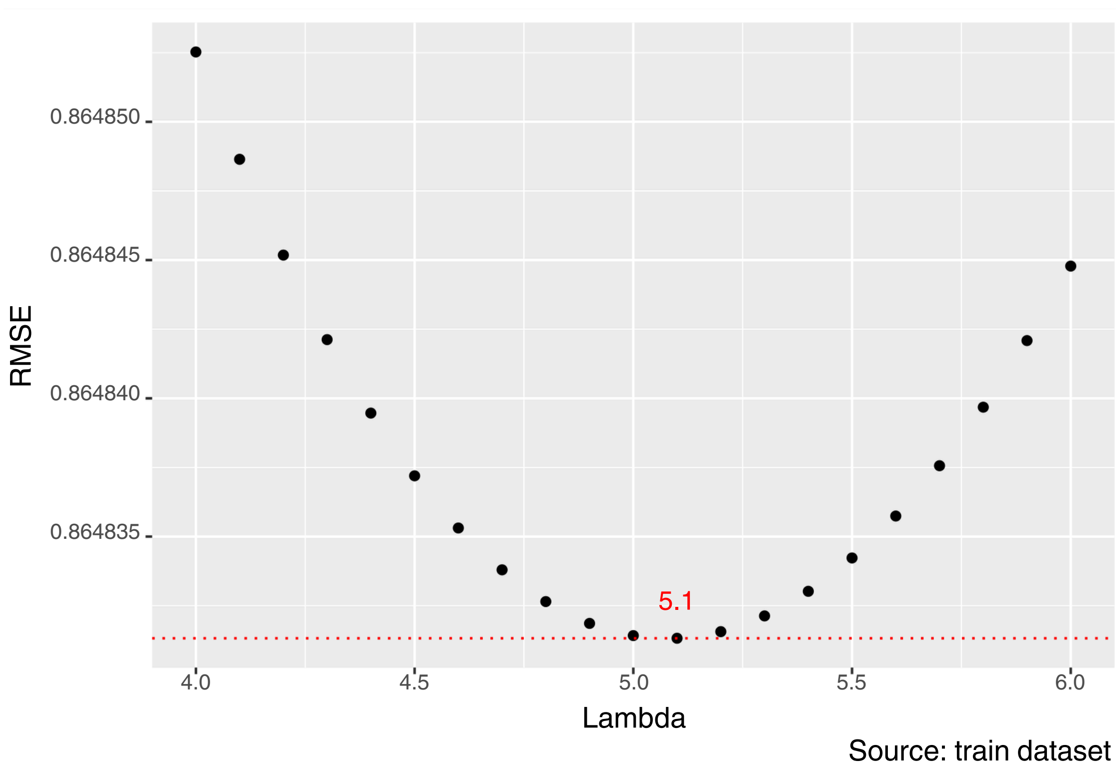

Figure 15: Selecting the tuning parameter

Method	RMSE	Difference
Project objective	0.86490	-
Simple average	1.06057	0.19567
Movie effects (b_i)	0.94371	0.07881
Movie + User effects (b_u)	0.86617	0.00127
Movie, User and Genre effects (b_g)	0.86582	0.00092
Movie, User, Genre and Year effects (b_y)	0.86567	0.00077
Movie, User, Genre, Year and Review Date effects (b_r)	0.86549	0.00059
Regularised Movie, User, Genre, Year and Review Date effects	0.86483	-0.00007

4.8 Final hold-out test in validation dataset

The final hold-out test in the validation dataset achieved an RMSE of 0.86405, an improvement of 18.53% versus the simple model based on the overall average rating and 0.00085 below the target RMSE set as project objective.

Method	RMSE	Difference
Project objective	0.86490	-
ValidationofFinalModel	0.86405	-0.00085

5 Conclusions

The objective of this project was to develop a recommendation system using the MovieLens 10M dataset that predicted ratings with a residual mean square error of less than 0.86490. Adjusting for a number of estimated biases introduced by the movie, user, genre, release year and review date, and then regularising these in order to constrain the variability of effect sizes, met the project objective yielding a model with an RMSE of 0.86483. This was confirmed in a final test using the previously unused validation dataset, with an RMSE of 0.86405.

Although the algorithm developed here met the project objective it still includes a sizeable error loss, not all of which may be considered truly independent, which suggests that there is still scope to improve the accuracy of the recommendation system with techniques that can account for some of this non-independent error. One such approach is matrix factorisation, a powerful technique for user or item-based collaborative filtering based machine learning which can be used to quantify residuals within this error loss based on patterns observed between groups of movies or groups of users such that the residual error in predictions can be further reduced (Irizarry 2020; Koren, Bell, and Volinsky 2009).

The techniques used in this project were limited due to the impracticality of using some powerful tools to train such a large dataset on a personal computer. One of the key advantages of matrix factorisation which has contributed to its popularity in recommendation systems is that it is both scalable and compact which makes it memory efficient and compatible with use on personal computers (Koren, Bell, and Volinsky 2009). Thus, further work on the recommendation system developed here would focus on the use of matrix factorisation.