Data Science: Capstone
(HarvardX PH125.9x)

Capstone Project I

MovieLens

Build a movie recommendation system

January 2021

submitted by Bela Koch

Contents

1 Introduction

Within this project, a recommendation system is developed that attempts to predict the rating an user would give a movie in terms of number of stars, whereby one to five starts can be awarded. A one-star rating indicates the worst possible rating, while a five-star rating corresponds to the best possible rating.

The data used in this project is provided from the GroupLens research lab, who have generated their own database with over 20 million ratings for over 27'000 movies by more than 138'000 users as part of their MovieLens project. For computational reasons, the 10M version of the MovieLens dataset is used for the development of the model. Following data is provided in the downloaded dataset:

variable	class	first_values
userId	integer	1, 1, 1
movieId	double	122, 185, 292
rating	double	5, 5, 5
timestamp	integer	838985046, 838983525, 838983421
title	character	Boomerang (1992), Net, The (1995), Outbreak (1995)
genres	character	Comedy|Romance, Action|Crime|Thriller, Action|Drama|Sci-Fi|Thriller
year_of_pub	double	1992, 1995, 1995
decade	double	1990, 1990, 1990
year_of_rating	double	2000, 2000, 2000

2 Analysis: developing the model

2.1 Residual mean squared error (RMSE)

The metric used to evaluate the model while developing is the residual mean squared error (RMSE), which is defined as

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

whereby $y_{u,i}$ denotes the rating of movie i by user u while $\hat{y}_{u,i}$ corresponds to the prediction of $y_{u,i}$. N stands for the number of user/movie combination and it is summed over all these combinations. The RMSE is the typical error that is made when predicting a movie rating and can therefore be interpreted similarly to a standard deviation. If the RMSE exceeds a value of one, a typical error of one star is made when estimating the rating of a movie. Consequently, the goal when developing the model used for the recommendation system is to minimize the RMSE.

2.2 Benchmark model

As reference value the simplest possible model is built which predicts the same rating for each movie regardless of the user and explains all the differences between ratings by random variation. Mathematically, the model is defined as follows:

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

whereby $\epsilon_{u,i}$ indicates independent errors sampled from the same distribution centered at 0 and μ indicates the *true* rating for all movies. Since the least squares estimate of μ which minimizes RMSE is the arithmetic mean, the predicted μ ($\hat{\mu}$) is defined as:

$$\hat{\mu} = \frac{1}{N}\sum_{u,i}^{N} \text{rating}_{u,i}$$

When applying the benchmark model on the test set, a RMSE of 1.06114 is reached:

model	rmse
benchmark model	1.06114

2.3 Adding movie effects

The relatively high RMSE in the benchmark model is expected, as it is assumed that there are no differences between all movies in all aspects. Purely intuitively and also from personal experience, however, it can be assumed that movies are of varying quality. This assumption is consistent with the data used for the analysis:

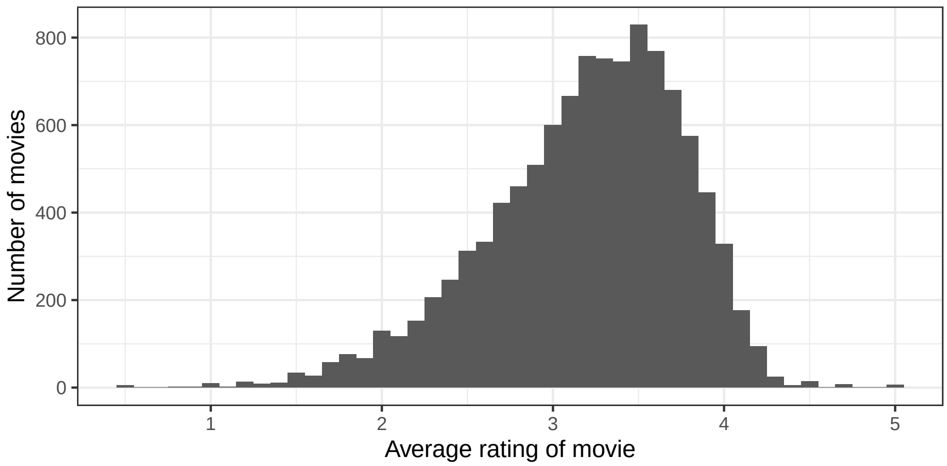

Figure 1: Average rating of a movie

Hence, the benchmark model can be extended by including a movie specific effect, i.e. by adding the average rating of the corresponding movie (b_i):

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

As we can see, the movie effect varies greatly for the different movies:

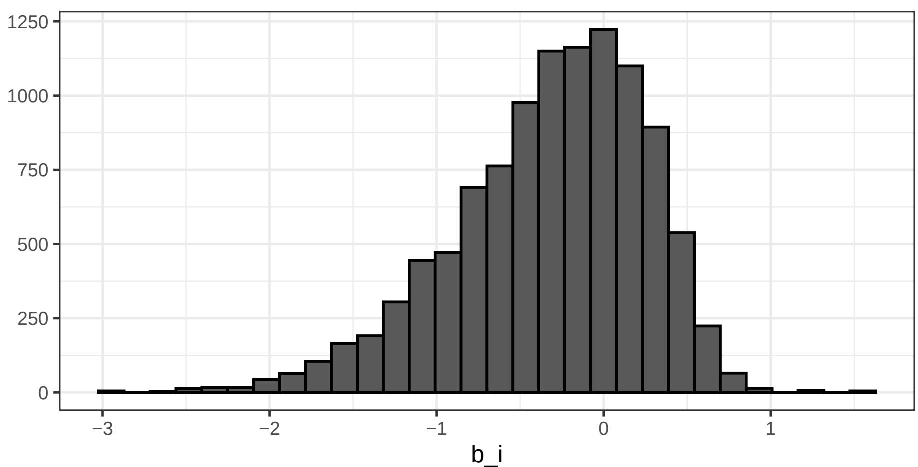

Figure 2: Movie effects

After including the movie effects, a RMSE of 0.94416 is reached:

model	rmse
benchmark model	1.06114
movie effect	0.94416

2.4 Adding user effects

As we have seen in previous section, RMSE could be reduced by considering the variability in quality between different movies. However, it was assumed that all users rate the movies in the same manner. But it is likely that the rating pattern is user-dependent - some users will be more critical and will on average rate movies lower than other users. The data used for the analysis supports this assumption:

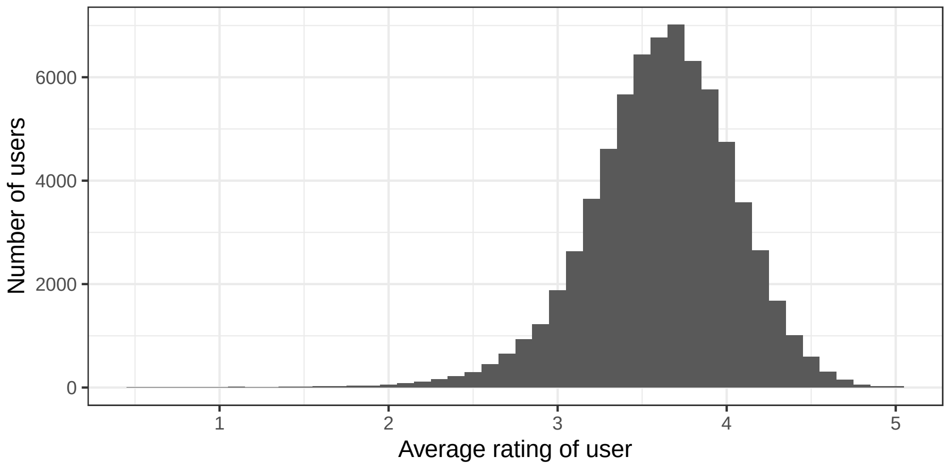

Figure 3: Average rating given by user

Thus, in a next step, the model was extendend to include user effects (b_u):

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

Again, a variation in the rating of the different users can be seen:

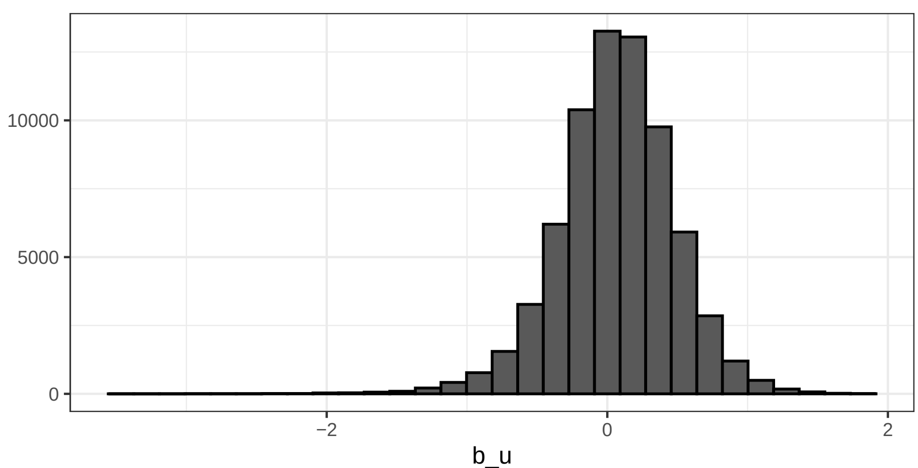

Figure 4: User effects

After including the user effects, a RMSE of 0.86597 is reached:

model	rmse
benchmark model	1.06114

model	rmse
movie effect	0.94416
movie+user effect	0.86597

2.5 Adding decade of publication effects

Many movie fans have a preferred decade in which they claim that the best movies were released. Furthermore, although no demographic data on users is available, it is possible that a certain group of users is over-represented in terms of year of birth. It could be that this group rates the movies of their childhood and youth higher for reasons of nostalgia, what would lead to biases. Another possible reason for varying quality in movies over the decades could be, that in earlier decades the production of a film was associated with higher entry barriers, which is why only large movie studios with sufficient resources could produce movies. As technology has progressed, the barriers to entry for movie production have been lowered, which could also potentially have a negative impact on the average quality of movies. The data used for this analysis shows, that movies from earlier decades are actually rated higher on average:

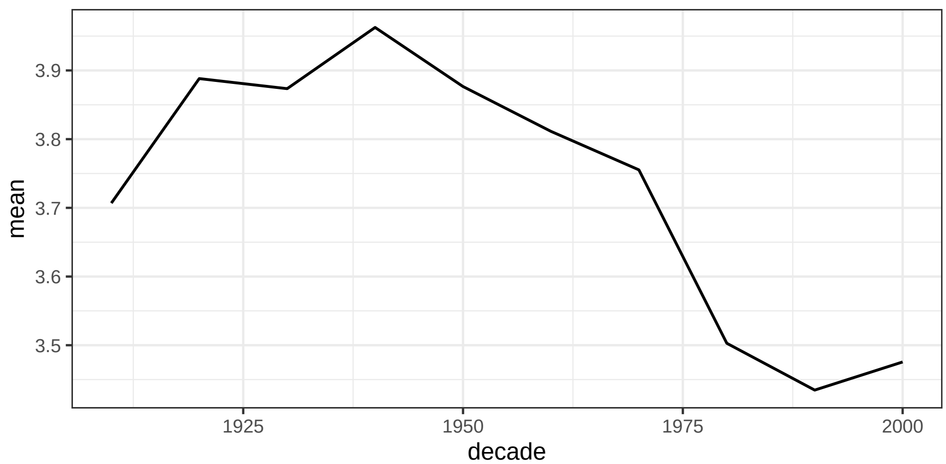

Figure 5: Average rating over decades

Note that the movies are classified by decade rather than by the exact year of publication, as it is assumed that the possible factors influencing the rating of a movie remain constant over a decade and thus data with less noise can be used to estimate the ratings. This should also reduce the possibility of overfitting.

The average ratings of the movies are shown below, classified according to the exact date of publication. The same trend is discernible, but with considerably more noise.

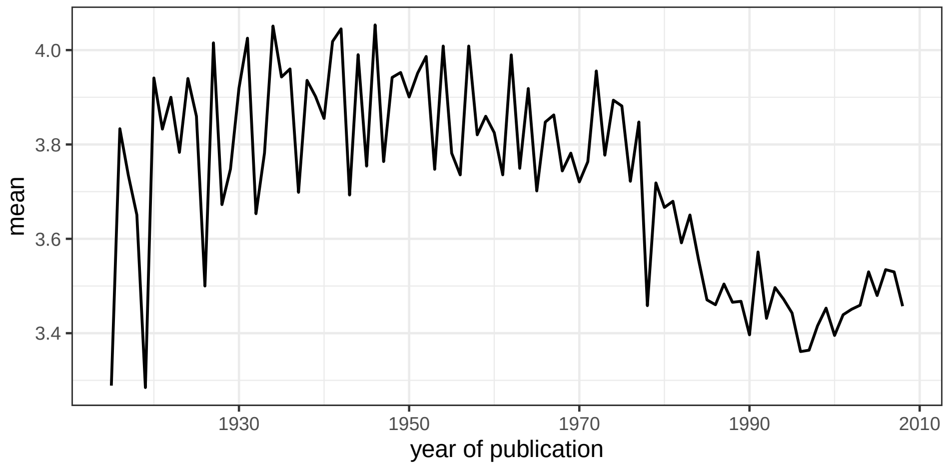

Figure 6: Average rating over years of publication

With the inclusion of the effect of the decade (b_d), the model can now be represented as follows:

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

The following graph shows the distribution of the decade effect:

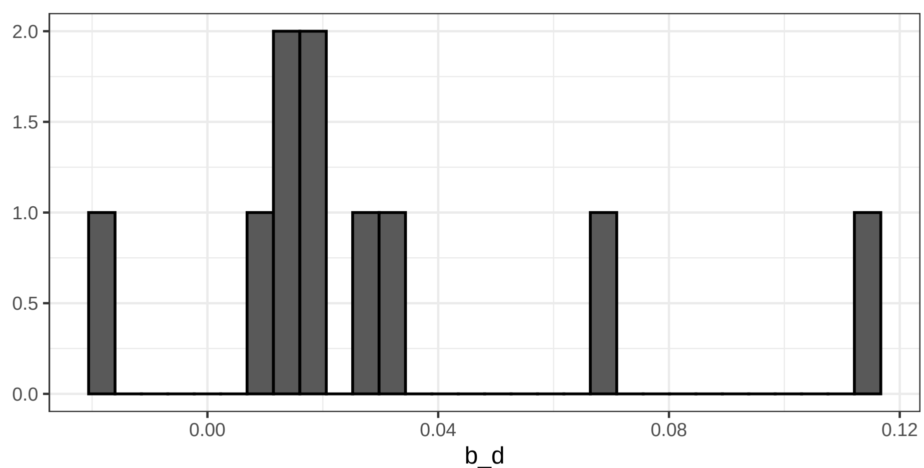

Figure 7: Decade effects

After including the decade effects, a RMSE of 0.86575 is reached:

model	rmse
benchmark model	1.06114
movie effect	0.94416
movie+user effect	0.86597
movie+user+decade effect	0.86575

2.6 Adding genre effects

Another influential effect could be the genre to which a movie belongs. For example, there may be genres that are generally more popular and therefore movies belonging to this genre receive a higher rating in average. Indeed, as we can see in the data, there is variability of the average rating per genre:

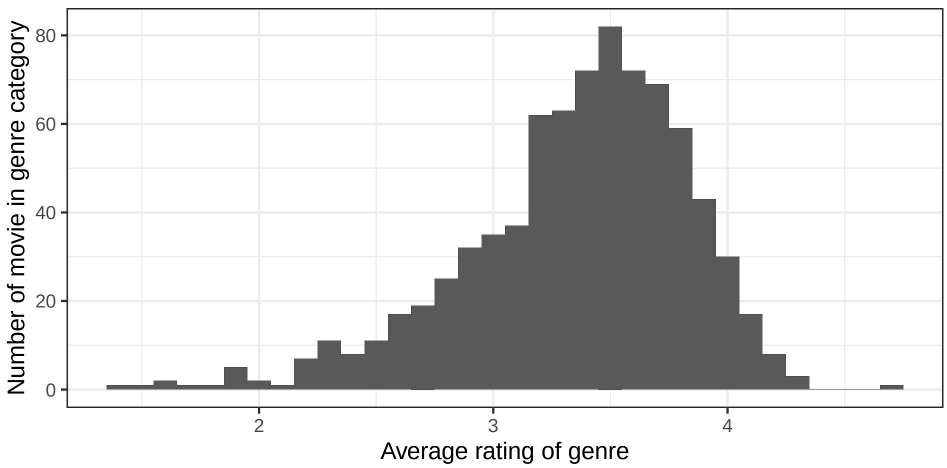

Figure 8: Average rating of genre

For this reason, the genre effect (b_g) is included in the model:

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

The distribution of the genre effect can be seen in the graph below.

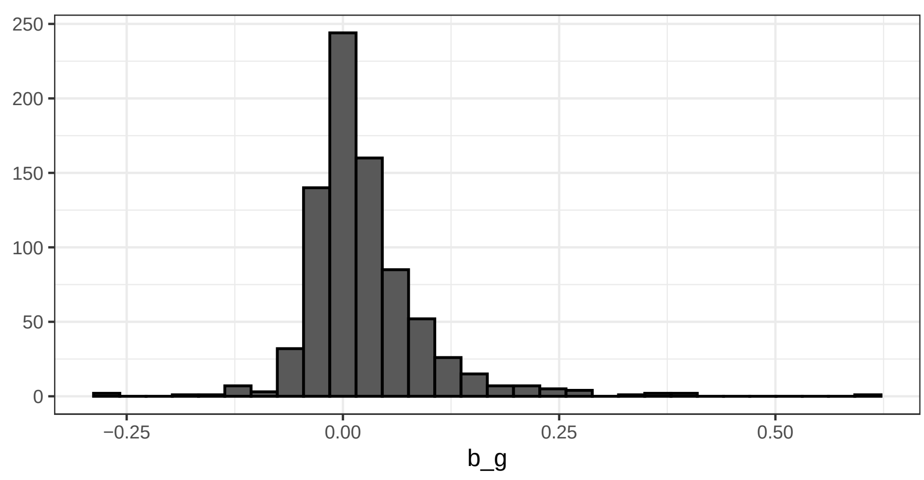

Figure 9: Genre effects

After including the decade effects, a RMSE of 0.86543 is reached:

model	rmse
benchmark model	1.06114
movie effect	0.94416

model	rmse
movie+user effect	0.86597
movie+user+decade effect	0.86575
movie+user+decade+genre effect	0.86543

2.7 Regularization

A major weakness of the previous model is that it does not take into account how much ratings where made per movie. Some movies have been rated by very few users,occasionally by only one user. This leads to more uncertainty and thus also to biases. More concretely, large values of the effect of b_i (positive and negative) due to biases are more likely with small sample sizes per movie. Consequently, an estimator based on a small sample size should be less trusted.

As Figure 10 confirms, the relative proportions of the group of movies with fewer ratings are greater when the average rating is more extreme:

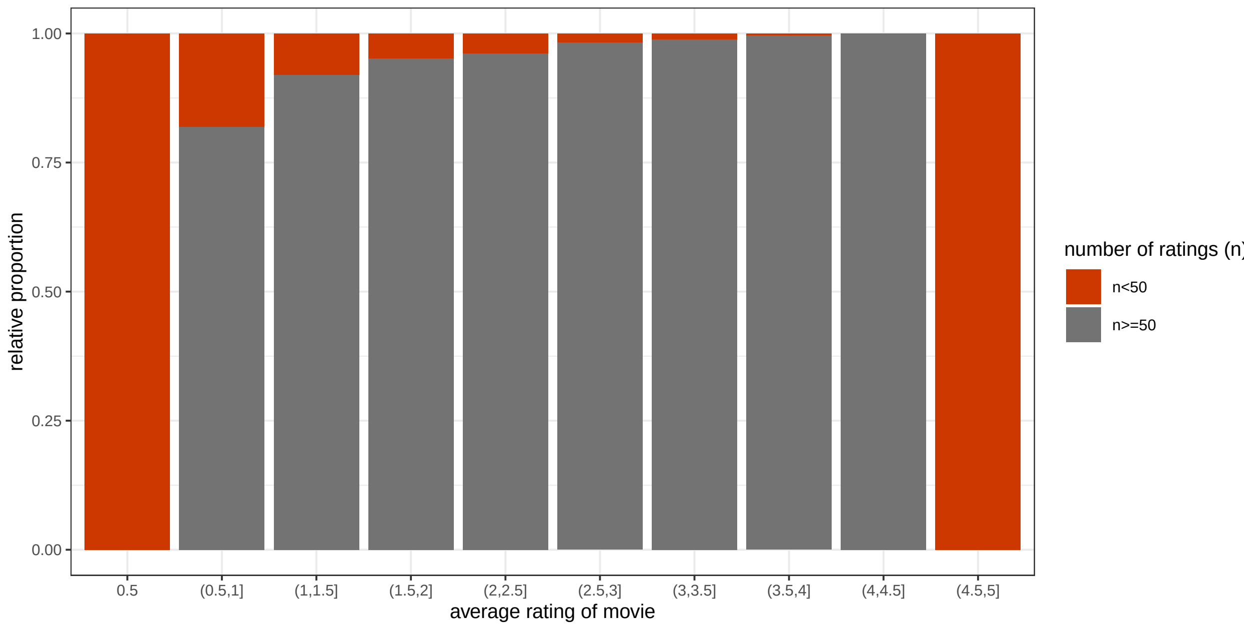

Figure 10: Average rating by number of ratings

For this reason, regularization is applied in a next step, which allows to penalize large estimates (in both directions) that are formed using small sample sizes to constrain the total variability of the effect sizes. More concretely, instead of minimizing the least squares equation, we minimize the least squares equation extended by a penalty term. Therefore, the following formula must be minimized for the estimation of the movie effects:

$$\frac{1}{N}\sum_{u,i}(y_{u,i}-\mu-b_i)^2 + \underbrace{\lambda\sum_{i} b_i^2}_{\text{penalty term}}$$

By means of calculus it can be shown that the values of b_i which minimize the above formula are:

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

whereby n_i denotes the number of ratings made for movie i. If the dependence of this formula on n_i and the penalty λ, it can be seen that the desired effect is achieved. With a large value of n_i, which is associated with a stable estimate of b_i, the penalty λ has little effect since $n_i + \lambda \approx n_i$. With a small value of n_i, which is associated with a biased or a relatively uncertain estimate of b_i, the penalty λ shrinks the estimate of b_i towards 0, resulting in a more conservative estimate. Note that when increasing the penalty λ, the estimate of b_i shrinks more leading to a more conservative estimate. The penalty λ is a tuning parameter and is chosen using cross validation.

Regularization can also be applied to the previously mentioned effects, resulting in the following minimization problems:

user effects:

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

decade effects:

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

Using regularization following RMSEs are reached:

	rmse		rmse
movie effect	0.94416	regularized movie effect	0.94412
movie+user effect	0.86597	regularized movie+user effect	0.86547
movie+user+decade effect	0.86575	regularized movie+user+decade effect	0.86527
movie+user+decade+genre effect	0.86543	regularized movie+user+decade+genre effect	0.86498

2.8 Matrix factorization

So far, no attention has been paid to the fact that both certain groups of movies and certain groups of users have similar rating patterns. With the use of matrix factorization it is attempted to capture these patterns and describe them by factors. The idea behind matrix factorization is to approximate the whole rating matrix $\boldsymbol{R}_{m \times n}$ by the product of two matrices of lower dimensions ($\boldsymbol{P}_{n \times k}$ and $\boldsymbol{Q}_{n \times k}$) so that

$$\boldsymbol{R} \approx \boldsymbol{P}\boldsymbol{Q}'$$

whereby $\boldsymbol{P}$ stores the user factors while $\boldsymbol{Q}$ stores the item factors, i.e. the movie factors. The rating matrix $\boldsymbol{R}$ can subsequently be used to predict the rating user u would give to movie i. In more formal terms, if p_u is the u-th row of $\boldsymbol{Q}$ and if q_i is the i-th row of $\boldsymbol{Q}$, then the prediction of the rating user u would give to movie i would be $p_u q_i'$. A typical solution for $\boldsymbol{P}$ and $\boldsymbol{Q}$ results from the following optimization problem:

$$\min_{P,Q} \sum_{(u,i) \in R} \left[f(p_u, q_i; r_{u,i}) + \mu_P ||p_u||_1 + \mu_Q ||q_i||_1 + \frac{\lambda_P}{2} ||p_u||_2^2 + \frac{\lambda_Q}{2} ||p_i||_2^2 \right]$$

whereby (u, i) are the locations of observed entries in $\boldsymbol{R}$ while $(u, i) \in R$ indicates that rating r_i is available, f is the loss function used, $||.||$ is the euclidean norm, $r_{u,i}$ is the observed rating and μ_P, μ_Q, λ_P and λ_Q are penalty parameters used as regularization to avoid overfitting.

To apply matrix factorization for this project, the `recosystem` package is used, which is a R package for recommender systems using parallel matrix factorization. The package allows to execute parameter optimization within a reasonable amount of time and with the relevant data stored in the RAM.

After tuning the model parameters with all default settings - except increasing the number of threads to four for parallel computing to increase performance - following parameters were selected to minimize the optimization problem:

Parameter	Selected value
μ_P	0.01
λ_P	0.1
μ_Q	0.01
λ_Q	0.01

Please note that when increasing the number of threads for parallel computing (setting `nthread > 1`) the training result is not guaranteed to be reproducible, even if a random seed is set. Since the default loss function for real-valued matrix factorization is the squared error which is the sum of all the squared differences between the observed values and the estimated values. Thus, the following minimisation problem with the parameters listed in the table above is relevant for the process of solving for $\boldsymbol{P}$ and $\boldsymbol{Q}$, that corresponds to model training:

$$\min_{P,Q} \sum_{(u,i)\in R} \left[(p_u q_i' - r_{u,i})^2 + \mu_P ||p_u||_1 + \mu_Q ||q_i||_1 + \frac{\lambda_P}{2} ||p_u||_2^2 + \frac{\lambda_Q}{2} ||p_i||_2^2 \right]$$

After training the model with 100 iterations, a RMSE of 0.79519 is reached:

model	rmse
benchmark model	1.06114
movie effect	0.94416
movie+user effect	0.86597
movie+user+decade effect	0.86575
movie+user+decade+genre effect	0.86543
regularized movie effect	0.94412
regularized movie+user effect	0.86547
regularized movie+user+decade effect	0.86527
regularized movie+user+decade+genre effect	0.86498
matrix factorization	0.79519

3 Results: applying the model

Since matrix factorization clearly outperforms the linear models and the regularized linear models according to the `edx` dataset, the matrix factorization model with the tuned parameters is chosen to be applied to the `validation` dataset.

4 Conclusion