Movie Recommendation System

Norbert Hartkamp, M.D., M.Sc.

01.10.2022

Contents

1 Executive Summary

An analysis is done of the 10M subset of the **MoviLens** database with the aim of developing a predictive algorithm for movie ratings, based on a division of the original data set that serves as a *training* data set.

The resulting algorithm is then tested against another, independent division from the original data set (*test* data).

The final result demonstrates that by including several predictive markers in a simple model, a considerable gain in predictive accuracy can be achieved.

2 Introduction and Overview

As part of the HarvardX PH125.9x course on *Data Science* this document attempts at working out a movie recommendation system. The system is based on the 10M-version the large **movielens**-database encompassing 10,000,054 ratings from 1/1995 to 1/2009, which was provided for this course by the edx-team.

I first will try to *describe* the data to some extent, and then to detect *patterns* regarding groups of *movies*, and *users*, and their respective *ratings*.

I'll also take into account possible effects of movie-*genres* and of the *age of the movie at the time of rating*.

2.1 The MoviLens data

The MovieLens system has first been released 1998 by a team from the Univ. of Minnesota (cf. Harper, F. M., & Konstan, J. A. (2015). The movielens datasets: History and context. ACM tiis, 5(4), 1-19.). The data that have been collected by MovieLens are a result of people interacting with an online movie-recommendation system, a process in which the users are required to input their movie preferences into the system.

Although MovieLens was only launched in the fall of 1997 it includes ratings from before that time. Those ratings had been made with the predecessor of MoviLens, "EachMovie" by DEC. DEC had decided to discontinue their system, and transfered an anonymized dataset of ratings to the makers of MovieLens, who had already developed a community based rating and filtering machanism for usenet-articles (Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R., & Riedl, J. (1997). Grouplens: Applying collaborative filtering to usenet news. Communications of the ACM, 40(3), 77-87.).

In MovieLens *collaborative filtering* has been used from the start (Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994, October). Grouplens: An open architecture for collaborative filtering of netnews. In Proceedings of the 1994 ACM conference on Computer supported cooperative work (pp. 175-186).). As Resnick et al. (1994) state: "Collaborative filters help people make choices based on the opinions of other people."

In the MovieLens framework this approach had been realized at first by presenting users movies that the system predicted the user would rate highest to be rated by that particular user. New users for which a prediction could not yet been made, were required to rate a given list of randomly and of hand-selected movies, before they could enter the MovieLens system (Rashid, A. M., Albert, I., Cosley, D., Lam, S. K., McNee, S. M., Konstan, J. A., & Riedl, J. (2002, January). Getting to know you: learning new user preferences in recommender systems. In Proceedings of the 7th international conference on Intelligent user interfaces (pp. 127-134).).

This system has prevailed until vers. 4 of MovieLens, although the underlying algorithm underwent changes a couple of times (Harper & Konstan 2015, p. 19:6). The early versions of MovieLens simply presented to the user one list of movies to be rated, later versions 2 and 3 included possibilities to select the genre and the release dates of the movies to be rated. In version 4 the ratings of so-called *buddies* were optionally included in the selection of movies to be rated.

2.2 R libaries being used

I will be using the following libraries in the process:

```
library(knitr)
library(data.table)
library(tidyverse)
library(ggthemes)
library(gridExtra)
library(caret)
library(lubridate)
library(kableExtra)
library(psych)
library(latex2exp)
library(compiler)
```

The basic data I'll be using are stored and loaded from the file *movie_recommendation.RData* out of the working directory of the present project. The data have been downloaded and prepared according to the instructions and R-code given under the heading "Create train and validation sets" in the course materials, which is reproduced in the box below and also included in the in the accompanying *movie_recommendation.r* file.

```
############################################################
# Create edx set, validation set (final hold-out test set)
############################################################

# Note: this process could take a couple of minutes

repository = "http://cran.us.r-project.org"
if(!require(tidyverse)) install.packages("tidyverse", repos = repository)
if(!require(caret)) install.packages("caret", repos = repository)
if(!require(data.table)) install.packages("data.table", repos = repository)

library(tidyverse)
library(caret)
library(data.table)

# MovieLens 10M dataset:
# https://grouplens.org/datasets/movielens/10m/
# http://files.grouplens.org/datasets/movielens/ml-10m.zip

dl <- tempfile()
download.file("https://files.grouplens.org/datasets/movielens/ml-10m.zip", dl)

ratings <- fread(text = gsub("::", "\t", readLines(unzip(dl, "ml-10M100K/ratings.dat"))),
                 col.names = c("userId", "movieId", "rating", "timestamp"))

movies <- str_split_fixed(readLines(unzip(dl, "ml-10M100K/movies.dat")), "\\::", 3)
colnames(movies) <- c("movieId", "title", "genres")

# if using R 3.6 or earlier:
movies <- as.data.frame(movies) %>% mutate(movieId = as.numeric(levels(movieId))[movieId],
                                           title = as.character(title),
                                           genres = as.character(genres))
# if using R 4.0 or later:
movies <- as.data.frame(movies) %>% mutate(movieId = as.numeric(movieId),
                                           title = as.character(title),
                                           genres = as.character(genres))


movielens <- left_join(ratings, movies, by = "movieId")

# Validation set will be 10% of MovieLens data
set.seed(1, sample.kind="Rounding") # if using R 3.5 or earlier, use `set.seed(1)`
test_index <- createDataPartition(y = movielens$rating, times = 1, p = 0.1, list = FALSE)
edx <- movielens[-test_index,]
temp <- movielens[test_index,]

# Make sure userId and movieId in validation set are also in edx set
validation <- temp %>%
```

```
  semi_join(edx, by = "movieId") %>%
  semi_join(edx, by = "userId")

# Add rows removed from validation set back into edx set
removed <- anti_join(temp, validation)
edx <- rbind(edx, removed)

rm(dl, ratings, movies, test_index, temp, movielens, removed)
```

2.3 Describing the data

2.3.1 The data-sets

The **edx**-data set consists of 9,000,055 records of 6 variables (*userId*, *movieId*, *rating*, *timestamp*, *title*, *genres*). The **validation**-set consists of 999,999 records of the same variables. Each record represents a rating of *one movie* by *one user*. The following table Tab. 1. gives an impression of the original structure of the movielens-data set as it is being used in the present task.

Table 1: Structure of records in edx-dataset

userId	movieId	rating	timestamp	title	genres
19	5	3	877088900	Father of the Bride Part II (1995)	Comedy
19	10	4	877082417	GoldenEye (1995)	Action|Adventure|Thriller
19	13	4	877088233	Balto (1995)	Animation|Children
19	17	5	877080437	Sense and Sensibility (1995)	Comedy|Drama|Romance

Each record has a unique **userId**, a unique **movieId**, a **timestamp** representing the date and time of the rating in the form of an **integer** (which can be converted into a human-readable date-time-string using the `as_datetime`-function from the *lubridate*-package, the **title** of the movie including the release-date in parentheses, and a **genres**-string that holds the labels for genres separated by the **pipe**-symbol `|`.

In the present analysis the **validation**- and **edx**-datasets have been modified by extracting the release date from the **title**-field into an additional column **releaseDate**.

Seemingly there are, however, a few problems with the **timestamp**-field: In a few cases the decoding of the **timestamp** results in a date *before* the release of the movie, which obviously indicates some kind of error.

E.g. in the **edx**-data set the command `edx[which(edx$userId==8010 & movieId==887), ]` results in a record with the timestamp `844937869` (see Tab. 2). Decoding this timestamp results in: `1996-10-10 08:57:49`, which obviously is wrong, as the date is *before* the release date of the movie.

Table 2: Possible error in the edx-dataset

userId	movieId	rating	timestamp	title	genres
8010	887	3	844937869	Talk of Angels (1998)	Drama

2.4 Further insights into the structure of the data

2.4.1 The movies

All in all there are 10,677 movies in the **edx**-set, and 9,809 movies in the **validation**-set. In the complete data set there are as well 10677 movies (four less than the number reported in Harper & Konstan (2015), Tab. II).

There is a weak association of $r = 0.213$ between the average rating of a movie and the number of ratings of a movie which means that movies with an on average higher rating are also rated more often, as shown in figure 1. This association might result from the fact that within the framework of the MovieLens system positively rated movies are more often presented to users, who then more often rate those, accordingly.

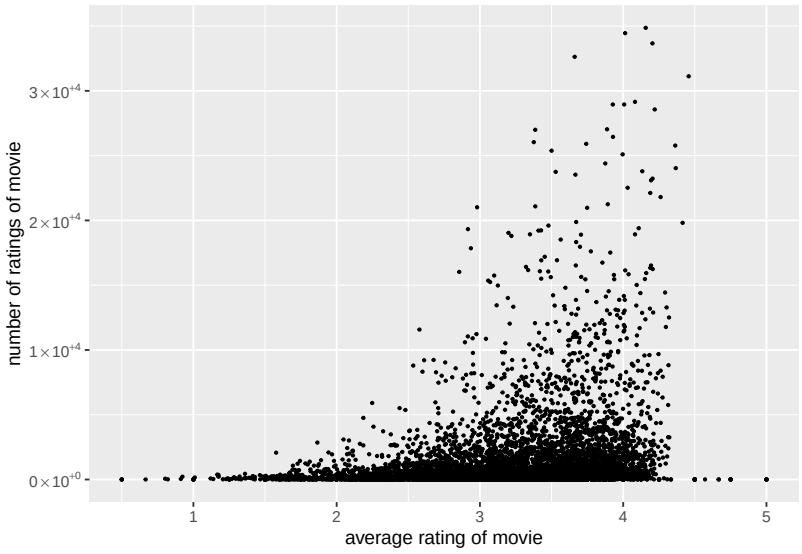

Figure 1: Average ratings of movies against number of ratings of movies

The fact that some movies are more often rated than others can also be seen from the distribution of the *count* of ratings for individual movies as shown in figure 2. Here the *median* number of rating per movie is 135, meaning that half of the movies in the database gets **135** or less ratings. On the other hand there is one movie (*Pulp Fiction (1994)*) having received **34864** ratings.

2.4.2 The genres

The different genres of the movies are *Action*, *Adventure*, *Animation*, *Children*, *Comedy*, *Crime*, *Documentary*, *Drama*, *Fantasy*, *Film-Noir*, *Horror*, *IMAX*, *Musical*, *Mystery*, *Romance*, *Sci-Fi*, *Thriller*, *War*, *Western* both in the **edx**- and in the **validation**-data set, with the exception of the **edx**-data set also having entries denoted as "*(no genres listed)*".

The following table 3 lists the number of ratings of movies to which each category is assigned both in the **edx**- and in the **validation**-sample.

The distribution of ratings of each movie among the different genres is quite similar with an average of the mean ratings of movies grouped by *genres* of $\overline{\mu} = 3.192$. (Interestingly the overall mean of all ratings in the complete database (not being grouped by *movieId*) is somewhat higher with $\overline{\mu}' = 3.512$, which again is the consequence of higher-ranking movies being rated more often.) The following graph (Figure 3) depicts the distribution of mean ratings of individual movies for the different genres.

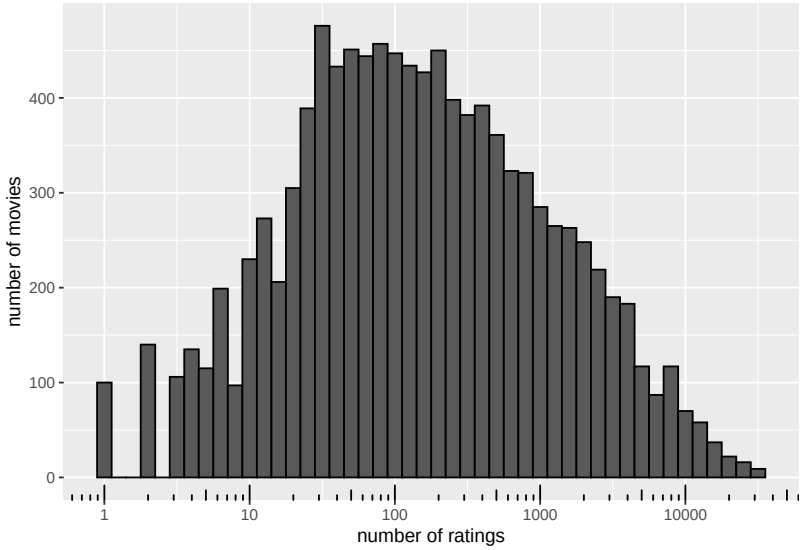

Figure 2: Count of ratings for individual movies

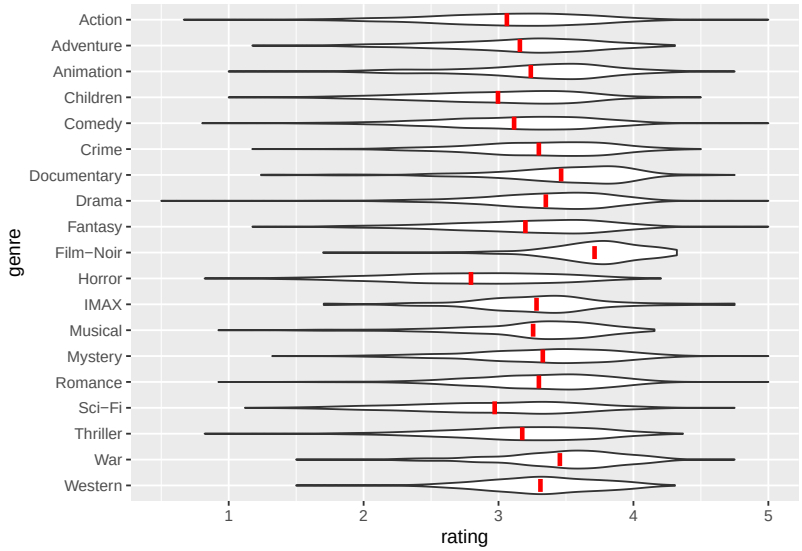

Figure 3: Distribution of mean ratings of movies assigned to each genre (red bar indicates the overall mean for each genre)

Table 3: Count of ratings of movies assigned to each genre

Genre	edx	validation
Drama	3,910,127	434,071
Comedy	3,540,930	393,138
Action	2,560,545	284,804
Thriller	2,325,899	258,536
Adventure	1,908,892	212,182
Romance	1,712,100	189,783
Sci-Fi	1,341,183	149,306
Crime	1,327,715	147,242
Fantasy	925,637	102,845
Children	737,994	82,155
Horror	691,485	76,740
Mystery	568,332	62,612
War	511,147	56,916
Animation	467,168	51,944
Musical	433,080	48,094
Western	189,394	21,065
Film-Noir	118,541	13,051
Documentary	93,066	10,388
IMAX	8,181	899
(no genres listed)	7	0

It can be noted that the **film-noir** movies on average get ratings that are somehat higher than the other movies, while the **horror**-movies on average are rated somewhat lower, as shown by the next box-plots, where the non-overlapping of the notches indicates the statistical significance of differences in mean ratings (see Figure 4).

2.4.3 The users

The number of distinct users in the **edx**-set is 69,878, and in the **validation**-set 68,534, respectively. However, there are obvious differences between the **edx**- and the **validation**-data set: In the **validation**-data set the the majority of users rates 7 or less movies (7 being the *median*), while in the **edx**-data set the median is 62.

In the **validation**-data set the minimum number of ratings of a single user is 1, and the maximum number is 743. In the **edx**-data set the minimum number of ratings of a single user is 10, and the maximum number is 6616. These differences simply result from the **validation**-set being much smaller than the **edx**-set (e.g. the **edx**-data set contains 19 ratings by user 1 (`userId == 1`), while the **validation**-set contains only 3 ratings by the same user).

However the mean rating each user gives is identical in both data sets (3.61), with just small differences of variability within users (**edx**: $sd = 0.96$, **validation**: $sd = 0.90$).

A small number of users have given ratings to a fairly huge number of movies, e.g. users *59269* and *67385*. Taking together the **edx**- and the **validation**-data set user *59269* has rated **7359** movies, and user *67385* has rated **7047** movies.

Figure 5 shows how many users rate how many movies. It can be seen that the majority of users rates less than 70 movies ($median = 69$).

In the case of user *59269* the difference between the dates of the first and the last rating is 2708 days. Given

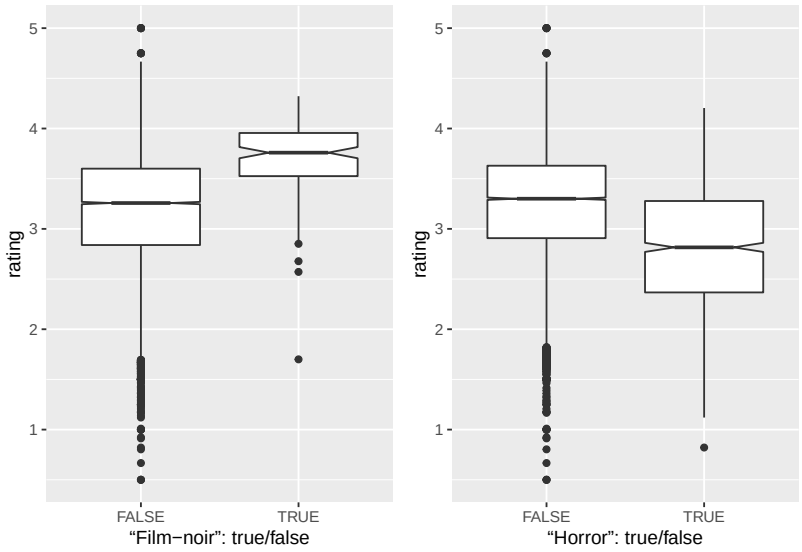

Figure 4: Comparison of film-noir- and of horror-movies to movies having been tagged with other genres

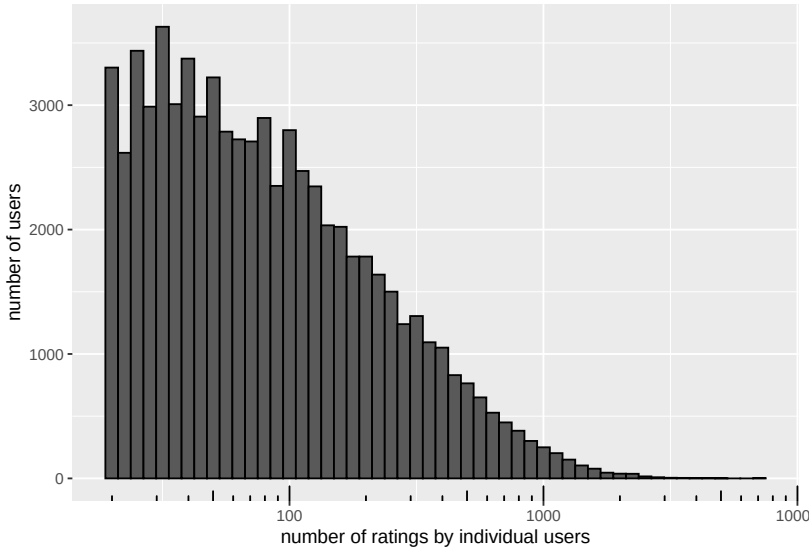

Figure 5: Distribution of number of ratings of individual users

the total number of ratings of this user, it means that this user, over a period of almost 387 weeks, has perhaps watched, and at least rated, every day (on average) a number of 2.7 movies.

It being unlikely that a single person has consumed so many movies over such an extended period of time, it is possible that this user (and maybe others like user *67385*) has made his huge amount of ratings based solely on a superficial impression of the respective movies.

It remains unclear, whether or not the inclusion of users with possibly not so well founded ratings has implications for the generation of movie recommendations. It is likely, however, that some kind of bias is being introduced.

2.4.4 The ratings

The range of ratings extends from min = 0.5 to max = 5. "Half-star" ratings have not always been present in the MovieLens database. As Harper & Konstan (2015, p. 19:8) state: "The biggest change to the ratings interface came with the launch of v3 (February 18th, 2003) when the interface shifted from "whole star" to "half star" ratings, the most-requested feature in a user survey, doubling the range of preference values from five (1–5) to ten (0.5–5.0)."

This change can be seen in the data. To visualize this the combined data set of **validation**-data and **edx**-data has been divided into one part having only the ratings from before Feb. 18th, 2003, and one part having the ratings from Feb. 18th, 2003 onwards. Figure 6 shows that in the former part there are only *full star*-ratings present, while in the latter part there are also *half-star*-ratings.

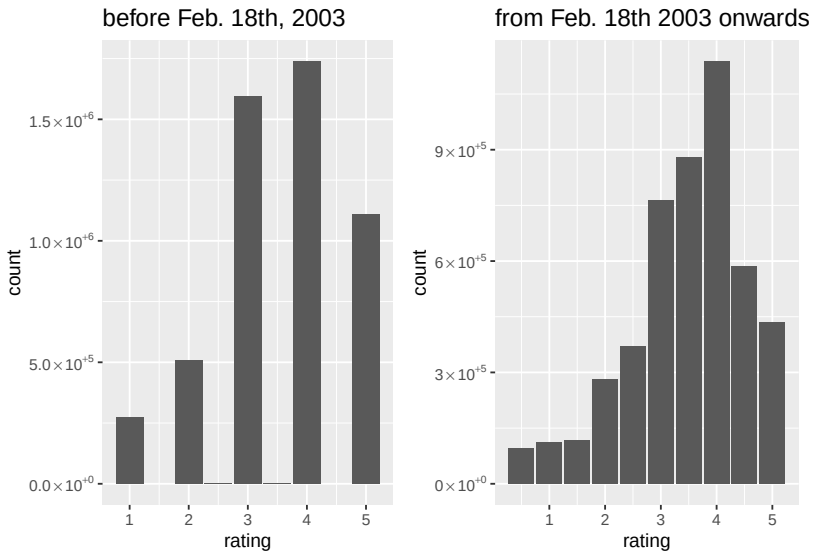

Figure 6: Distribution of ratings in ver.0 to ver.2, and ver.3 and after of MovieLens database

Figure 7 shows that, with the exception of the oldest movies in the database, the newer movies tend to be rated somewhat lower than those that were 25+ years old at the time of rating, which might show some influence of a "nostalgia"-effect among the users. Here the above mentioned possible error with some entries in the `timestamp`-field (see: 2.3.1) has been corrected by setting *negative differences* of *rating date* and *release date* (*"movie age"*) to `0`.

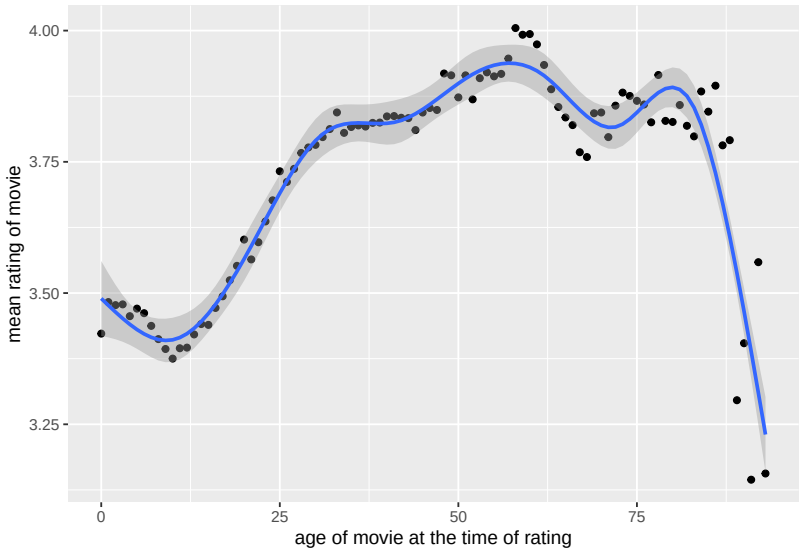

Figure 7: Mean ratings of movies against difference between release year and year the movies was rated

3 Methods

3.1 Defining the *root mean squared error* (RMSE)

The RMSE is a measure of accuracy of a prediction. In the given case this means that the larger the RMSE, the lesser the accuracy of the prediction of movie ratings.

The RMSE is mathematically related to the *standard deviation* in that it is the root of the mean squared differences of each rating and the prediction of such rating resulting from the model.

The formula of the RMSE is as follows:

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

In *R* this boils down to a (pseudo-)code like: `RMSE = sqrt(mean((predicted_ratings_from_the_model - actual_ratings_from_validation_set)^2))`. This function is implemented as `RMSE()` in the *caret*-package which has been loaded for the present analysis.

To be able to compute the RMSE it is first necessary to determine from the **edx**-data set (**train**-set in the following) a function to calculate predictions.

The results of this procedure then have to be checked with the **validation**-data set, in the present case by calculating the RMSE.

3.2 Modeling approach

3.2.1 grand mean of ratings as a predictor

The easiest approach to predicting movie ratings would be to assume that all movies get a rating as high as the mean rating in the **train**-set ($\hat{\mu} = 3.51$).

The formula in this case would be:

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

where μ is the grand mean of all ratings and ϵ is an error-term. The indices u and i refer to the *users*, and the *movies* respectively. Other features (besides *users* and the *movies*) might be included as well, however.

The calculation of the RMSE results in the **train**-set in a value of RMSE = **1.0603**, meaning that the average deviation of the prediction from the actual value is somewhat larger than **1**. In the **test**-set the result is RMSE = **1.0612**, which is even a bit worse. With this model on average the ratings would be "one star" off, which is not a good result.

3.2.2 getting more specific by including movie-information

To get to better results it is required in a first step to include more specific information on particular movies.

Here the formula is:

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

where b_i represents the deviation of the mean rating of a specific movie from the overall mean μ. The distribution of these b_i-values is skewed as can be seen from the following graph (Fig. 8).

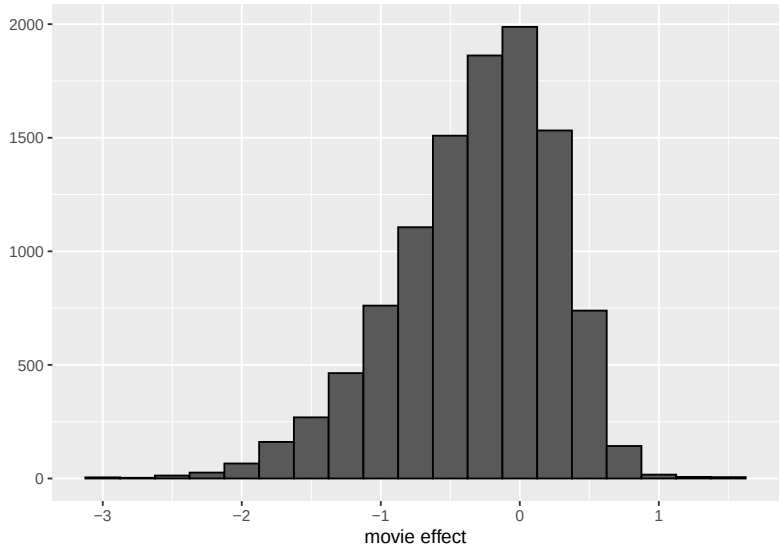

Figure 8: Distribution of *movie-effects* b_i

The predictions are now computed by *adding* for each movie the specific movie effect b_i (which can be positive *or* negative) to the estimated overall mean $\hat{\mu}$ and using the resulting vector to calculate the *RSME*.

The deviation of predicted and actual ratings in the **train**-set is now lower with a RMSE = **0.9423**, meaning the prediction error is now somewhat less than **1.0**.

3.2.3 adding user-information

It is likely that different users follow different patterns in their ratings. There might be users who are almost always critical of the movies they rate, while others might like almost everything they are confronted with.

This variability can be seen from the following graph (Fig. 9). It shows that the variation of users' ratings is roughly normal distributed around $m = 3.614$ with a *standard deviation* of $s = 0.431$

To further improve the predictions of movie ratings a user-term can be included in the equation, which now is:

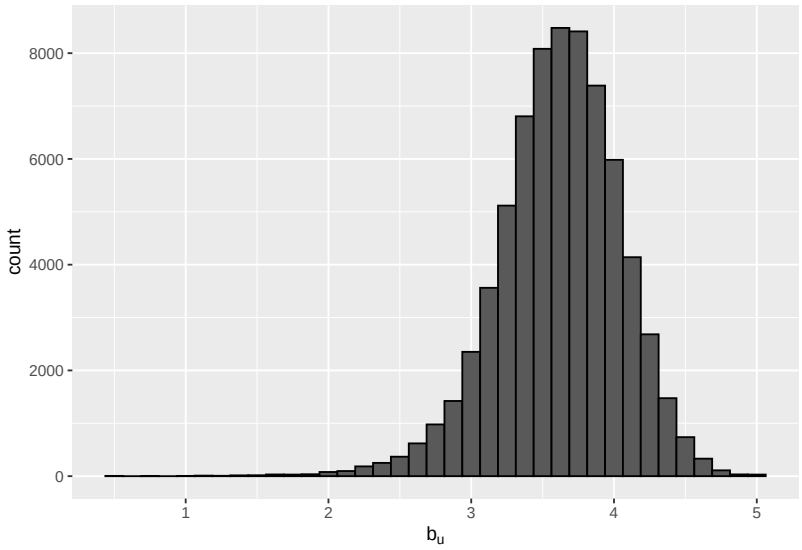

Figure 9: Distribution of *user-effects* b_u

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

where b_u represents the deviation of the mean rating of an individual *user* from the mean rating of one movie $b_i + \mu$.

By including the *user*-term the result in the **train**-set is RMSE = **0.8567**, which is considerably better than the initial model.

3.2.4 effects of movie-genres

As was reported in the section on genres (see 2.4.2) there is a difference in mean ratings for movies from different genres. This difference can be included in the model according to formula:

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

The question is, however, how to calculate the influence of the *genres* b_g. One way to do this is to group all ratings by their *genres*-string (which results into 797 combinations), and then calculate the *mean*-rating m_g for each of those combinations.

A different way is to take the list of all *genre*-tags (including the "(no genres listed)"-tag which ist present only in the **edx**-data set), and then to subsequently calculate mean ratings for all movies, to which the *genre*-tags from this list apply. In the case of multiple *genre*-tags per movie then the sum of these means is divided by the number of *genre*-tags for the respective movie.

As a formula it is:

$$m_{g(comb)} = \frac{1}{G} \sum_{g=1}^{G} x_g$$

where G is the number of *genres* with which a movie is tagged, and x_g is the mean rating of all movies having been tagged with *genre* g.

If m_g and $m_{g(comb)}$ are plotted against each other it can be seen that both are not linear associated, and so they are not really equivalent, as shown in the following graph (Fig. 10).

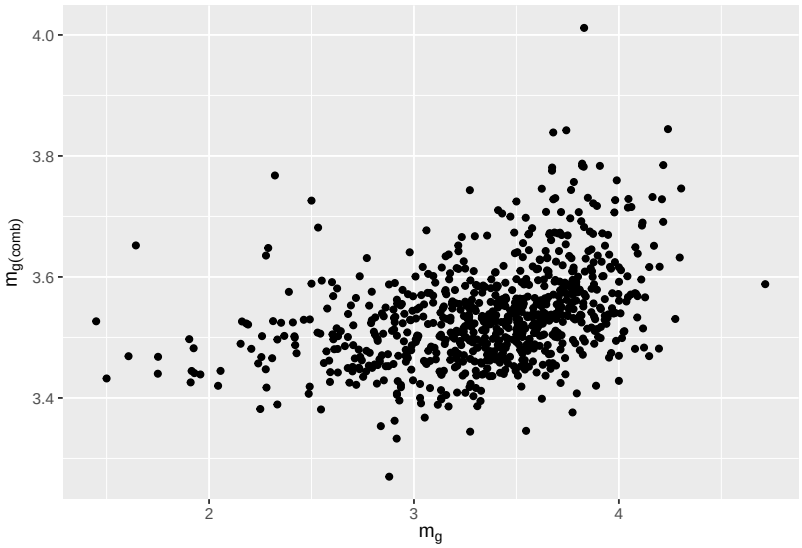

Figure 10: Plot of combination of mean genre ratings $m_{g(comb)}$ and mean ratings grouped by *genres*-string m_g

To decide which of the two approaches should be chosen, a RMSE is calculated with the resulting b_g of both. In the case of using m_g results a RMSE = **0.8564**, in the case of using $m_{g(comb)}$ the result is RMSE = **0.8913**, which indicates that m_g is a better choice to be included in the predictive model. However, the gain in prediction accuracy by including genre-information seems to be only marginal.

3.2.5 does movie-age at the time of rating play a part?

As it was shown in the section on ratings (see 2.4.4) there is an association between the age of the movie at the time of rating and the ratimg it gets. It might be useful then to include this factor into the prediction. With b_a indicating the age of the movie at the time of rating as a formula it would then be:

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

Including this factor, however, again contributes only marginally to an improvement of the prediction, which (in the **train**-set) is now RMSE = **0.8559**.

3.2.6 is it possible to still get to better results using regularization?

It can be assumed that some movies with only a few ratings get an overly strong influence on the predictions, particularly if the deviation of a movie's *actual* from the *predicted* rating is high. To account for this a *regularization* can be done by adding a term λ which is construed such, that, say, a sum of n deviations of *actual* and *predicted* ratings is divided by $\lambda + n$. If n is *large* this term basically equals the mean deviation of *actual* and *predicted* ratings, and is counted as such. If, however, n is small (e.g.: 1), then the mean deviation of *actual* and *predicted* ratings shrinks considerably (of course depending on the size of λ), as λ is added to the demnominator when averaging the ratings.

As a formula this can be written as:

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

In the present case a λ was picked by cross-validation using a sequence of possible λs from `0` to `10` with smaller intervals from `4.1` to `5.2`, where the regularization was done with *movie* and *user* variables only.

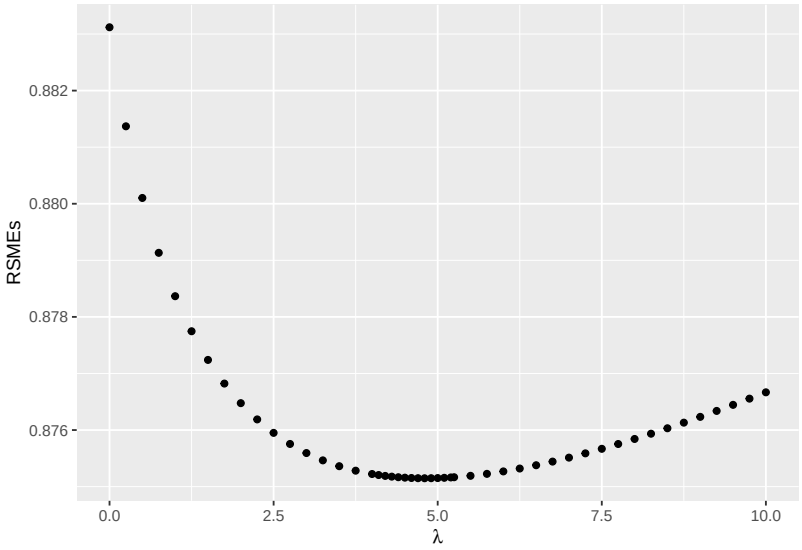

Figure 11: Plot of λ and RMSE from the present model

From Fig. 11 it can be seen, that a minimum *lambda* of 4.8 can be determined which might increase the accuracy when include in the calculation of the prediction.

4 Results

The results of predictions using the present modeling approach (see 3.2) are presented in the below table:

Table 4: Results of predictions in the validation-data set

prediction.model	RMSE
overall mean	1.06120
movie effects	0.94391
movie and user effects	0.86535
movie, user, and genres effects	0.86495
movie, user, genres, and movie age effects	0.86454
reularized movie & user plus genres & movie age effects	0.86402

The final result of using *movie*, *user*, combination of *genres* for movies, and *age of movie* at the time of rating results in the RMSE = **0.86402**, which in the context of the present task seems acceptable.

5 Concluding Remarks

While the first estimate using only the overall mean resulted in an RSME larger than 1.0, the final model resulted in a RMSE that was improved approx. 19.7 %. Further methods could maybe be applied, like matrix factorization to get to even better results, that anyway weren't aimed at in the context of the present task.