HarvardX Data Science Capstone Project

Philippe Landucci

January 8th, 2021

Contents

1 Executive Summary

The goals of this project are to build a movie recommendation system. The main task of a such a system is to predict which ratings would users give to items, in our case movies, based on observed values.[1]

The recommendation system in this project is based on the **MovieLens** data set and shall achieve a model performance of RMSE lower than 0.86490.

Various MovieLens data sets are available at the web site *(http://grouplens.org)* of the **GroupLens Research Team** at the **University of Minnesota**. They have been collected over various periods of time from the **MovieLens** web site *(http://movielens.org)*. The "10M" data set used in this project contains approximately 10 millions of movies ratings which have been split into a training set of 9 millions and a test or validation set of one million of movie ratings.

The following approach has been used:
- Data exploration, including numerous visualizations, data cleaning and insights
- Fitting models and comparing their performance
- Summary comparison of models performance

Special attention has been given to the insight, that the "10M" data set contains approximately half of the observations based on a "full stars" rating scale and the other half on a finer "half stars" rating scale, with a switch to "half stars" ratings mid February 2003. Therefore, all models have been fitted and their performance evaluated not only on the "original" data, but also on the "full stars" and "half stars" data.

The best performance on the original data has been achieved using a "Matrix Factorization" model. RMSE could be reduced to approximately 0.785, with some lower level residual variation due to the randomized algorithm.

2 Data Exploration

Data exploration is conducted along the following steps:
- High-level exploration of the training and test data sets
- Extraction of movie release year into a separate column
- Oldest and newest movies
- Extraction of year of rating and movie age - when rated - into separate columns
- Sanity check - find/remove observations where movie rated before launch
- Exploration of movie age on mean of ratings
- Exploration of ratings
- Further exploration of movies
- Exploration of users
- Exploration of genres
- Structure of data sets after data exploration
- Additional data preparation due to change in ratings scale in February 2003

2.1 High-level exploration of the training and test data sets

2.1.1 Training data set

The training data set "edx" has 9000055 observations of 6 variables.

The variables are:
- **userId** : User identification key *(integer)*
- **movieId** : Movie identification key *(numeric)*
- **rating** : "Five Stars" type rating, ranging from 0.5 to 5 *(numeric)*
- **timestamp** : Time stamp of rating, in Posix format *(integer)*
- **title** : Title and release year of movie *(character string, release year in parentheses)*
- **genres** : Genre(s) of movie *(character string, multiple genres delimited by vertical bar(s)*

In the training data set, 69878 users have rated 10677 movies.

2.1.2 Test data set

The test data set "validation" has 999999 observations of 6 variables.

In the test data set, 68534 users have rated 9809 movies.

2.2 Extraction of movie release year into a separate column

The training data set now has the following structure:
- **userId**: User identification key *(integer)*
- **movieId**: Movie identification key *(numeric)*
- **rating**: "Five Stars" type rating, ranging from 0.5 to 5 *(numeric)*
- **timestamp**: Time stamp of rating, in Posix format *(integer)*
- **title**: Title and release year of movie *(character string, release year in parentheses)*
- **genres**: Genre(s) of movie *(character string, multiple genres delimited by vertical bar(s)*
- **year_released**: Release year of movie *(integer)*

2.3 Oldest and newest movies

The next two tables show the oldest and latest movies in the training data set.

Table #1 - Oldest movies

Year released	Movie title
1915	Birth of a Nation, The
1916	Intolerance
1916	20,000 Leagues Under the Sea
1917	Immigrant, The
1917	Father Sergius (Otets Sergiy)
1918	Dog's Life, A

Table #2 - Latest movies

Year released	Movie title
2008	Witless Protection
2008	Forbidden Kingdom, The
2008	27 Dresses
2008	Rambo
2008	Iron Man
2008	Kung Fu Panda

2.4 Extraction of year of rating and movie age - when rated - into separate columns

The training data set now has the following structure:
- **userId**: User identification key *(integer)*
- **movieId**: Movie identification key *(numeric)*
- **rating**: "Five Stars" type rating, ranging from 0.5 to 5 *(numeric)*
- **timestamp**: Time stamp of rating, in Posix format *(integer)*
- **title**: Title and release year of movie *(character string, release year in parentheses)*
- **genres**: Genre(s) of movie *(character string, multiple genres delimited by vertical bar(s)*
- **year_released**: Release year of movie *(integer)*
- **year_rated**: Year of rating *(integer)*
- **movie_age**: Movie age, in years, when rated *(integer)*

2.5 Sanity check - find/remove observations where movie rated before launch

Warning: 175 movies have been rated before launch…

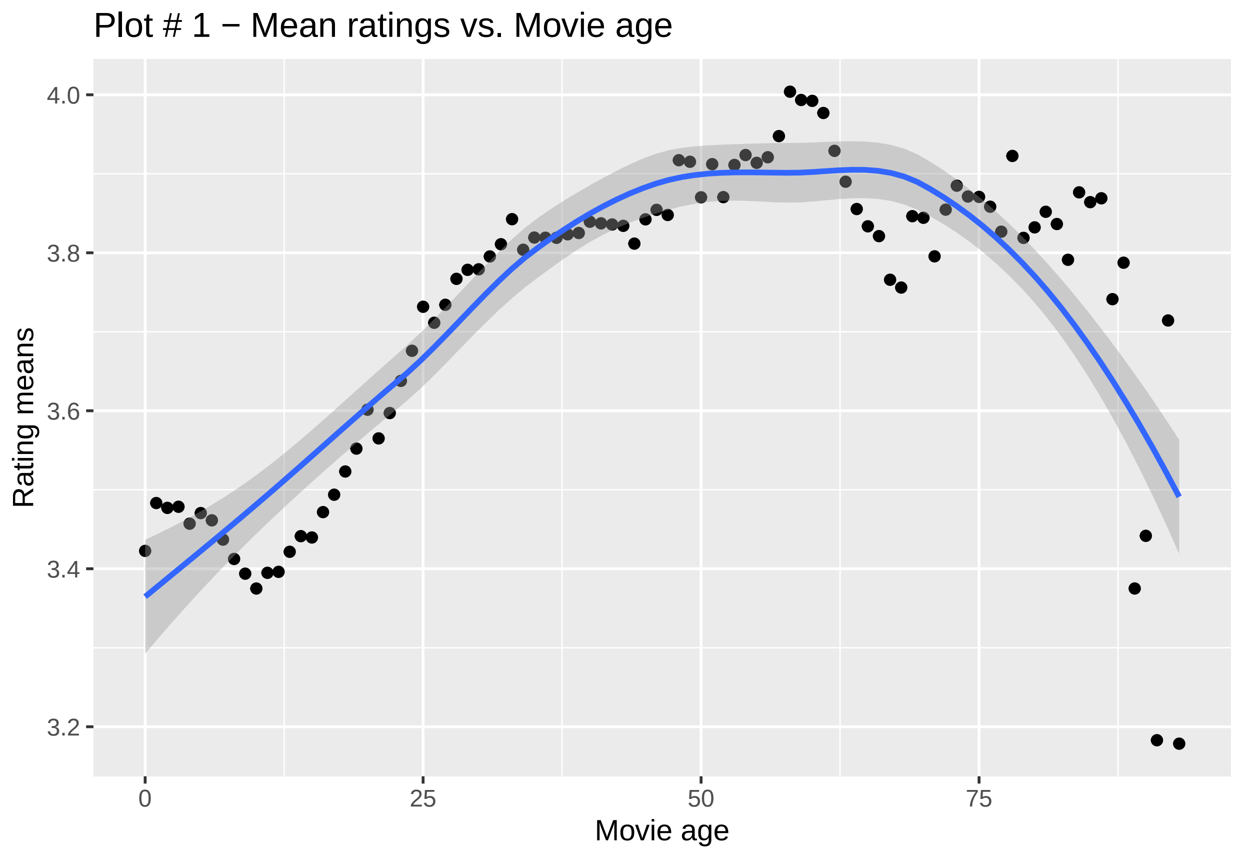
The number of observations in the training data set has been reduced by 175 to `toString(dim(edx)[1])`

2.6 Exploration of movie age on mean of ratings

2.6.1 Mean ratings per movie age

Table #3 - Ratings means vs. movie age

Movie age	Ratings means (highests)	Movie age	Ratings means (lowests)
58	4.003894	93	3.178571
59	3.993420	91	3.182927
60	3.992289	10	3.374975
61	3.976950	89	3.375000
57	3.947688	9	3.393785
62	3.929015	11	3.394863



2.6.2 Modeling mean ratings as a function of movie age

The plot shows kind of a parabolic (2nd order) relationship between movie age and mean rating. This assumption will be verified by fitting a linear model with a polynomial function. Even though curve fitting can be - very slightly - improved with high orders, we stop at order 4 to avoid over-fitting and rank deficiency

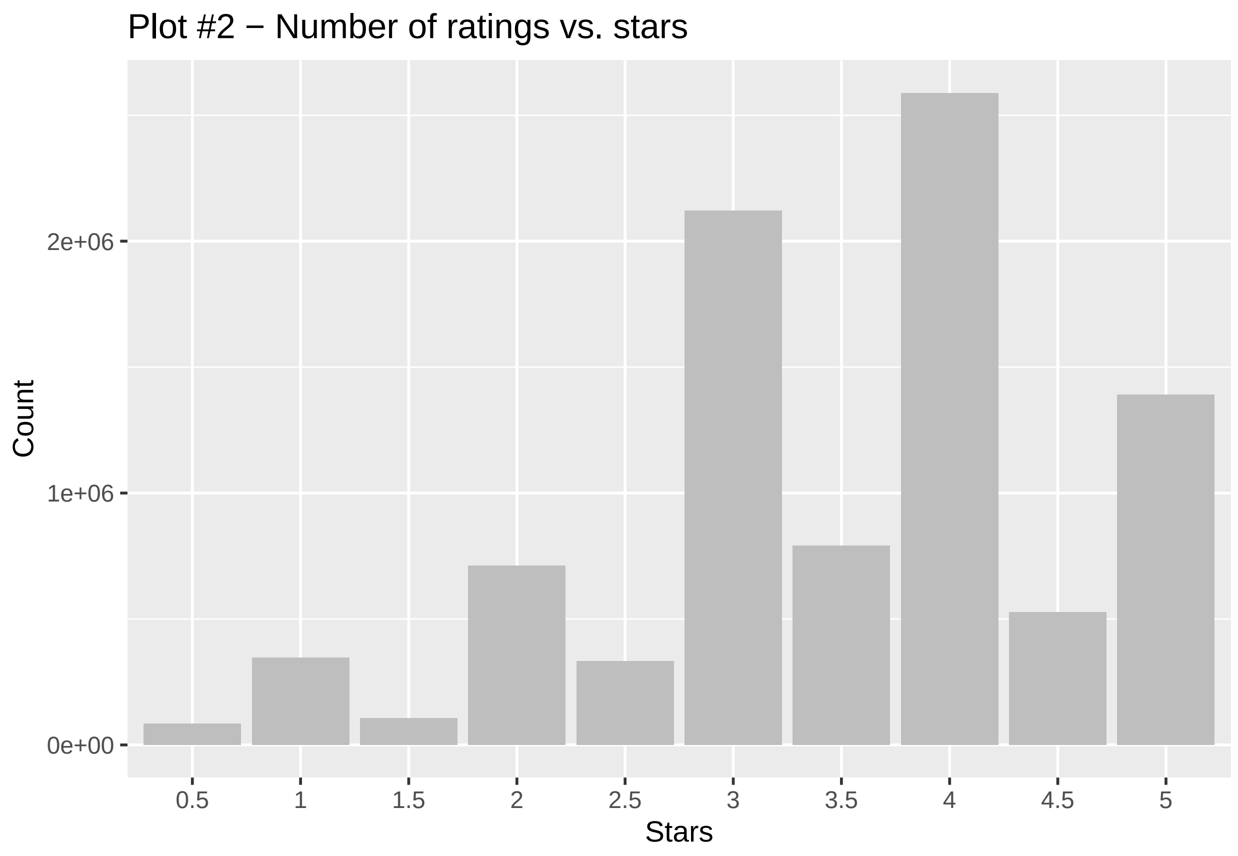
Best fitting result with order: 4, rmse: 0.0897712

2.7 Exploration of ratings

2.7.1 Distribution of number of ratings per number of stars

Table #4 - Ratings means vs. movie age

Rating	**Count**
4.0	2588399
3.0	2121185
5.0	1390077
3.5	791624
2.0	711402
4.5	526736
1.0	345649
2.5	333008
1.5	106426
0.5	85374

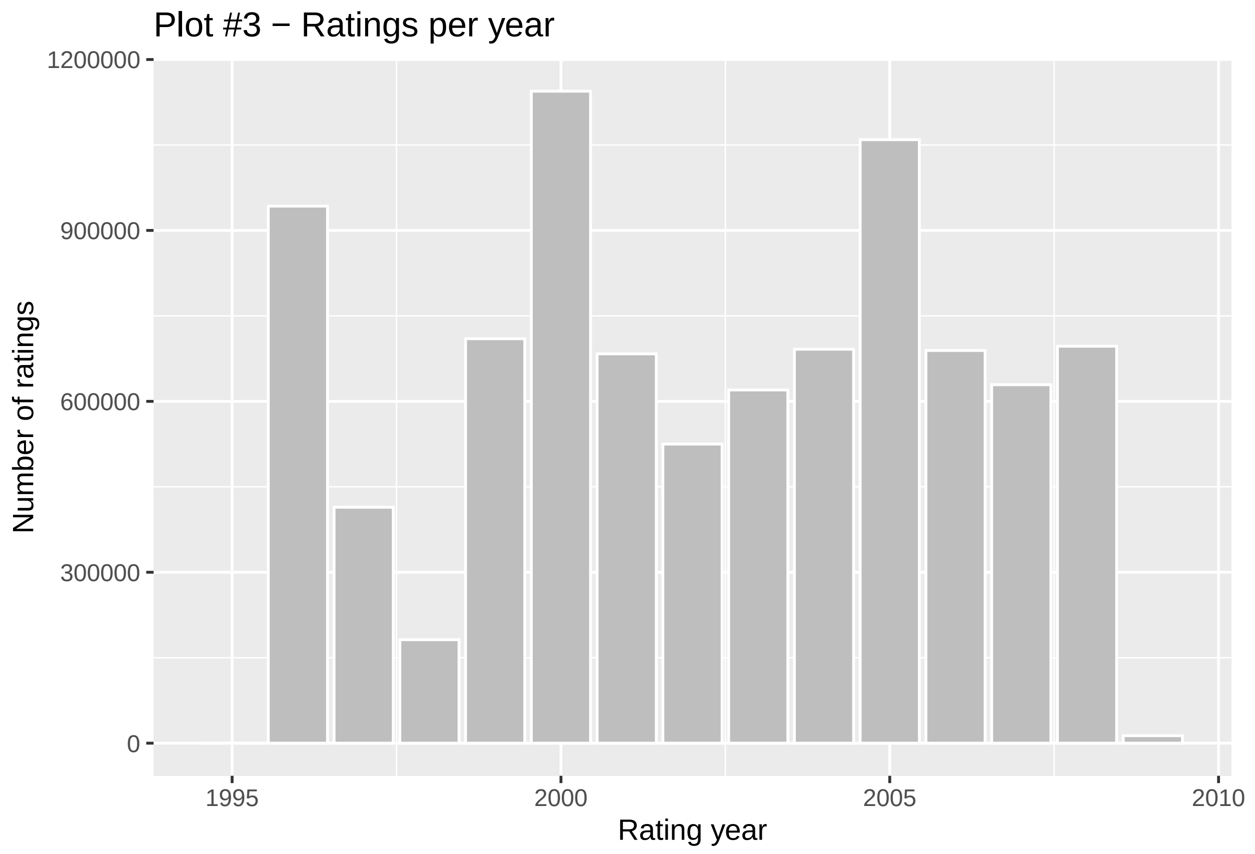


The plot shows that there are systematically less half star that full star ratings, needs further exploration. . .

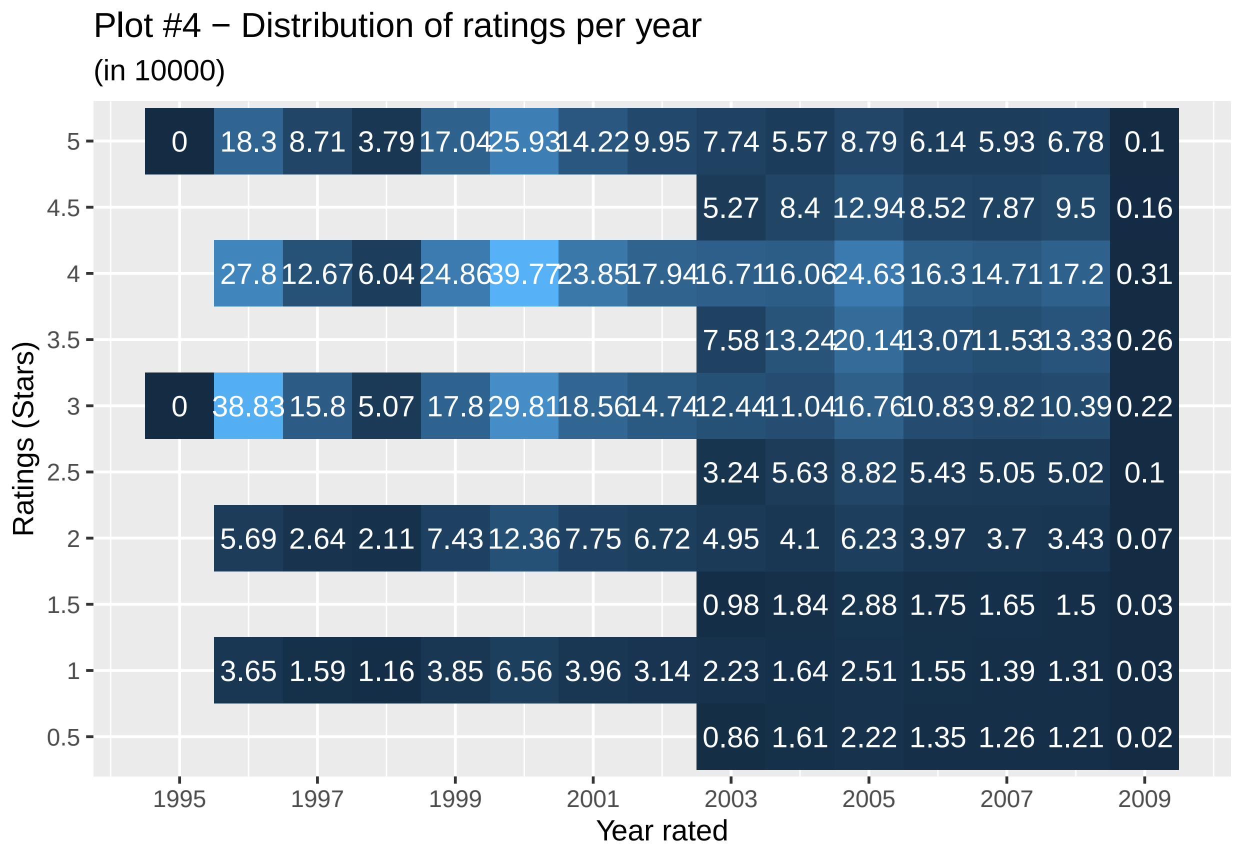
2.7.2 Distribution of number of ratings per year

Table #5 - Ratings means vs. movie age

Year rated	Ratings
1995	2
1996	942643
1997	414072
1998	181628
1999	709889
2000	1144349
2001	683354
2002	524955
2003	619938
2004	691429
2005	1059275
2006	689315
2007	629168
2008	696740
2009	13123



2.7.3 Distribution of ratings per year



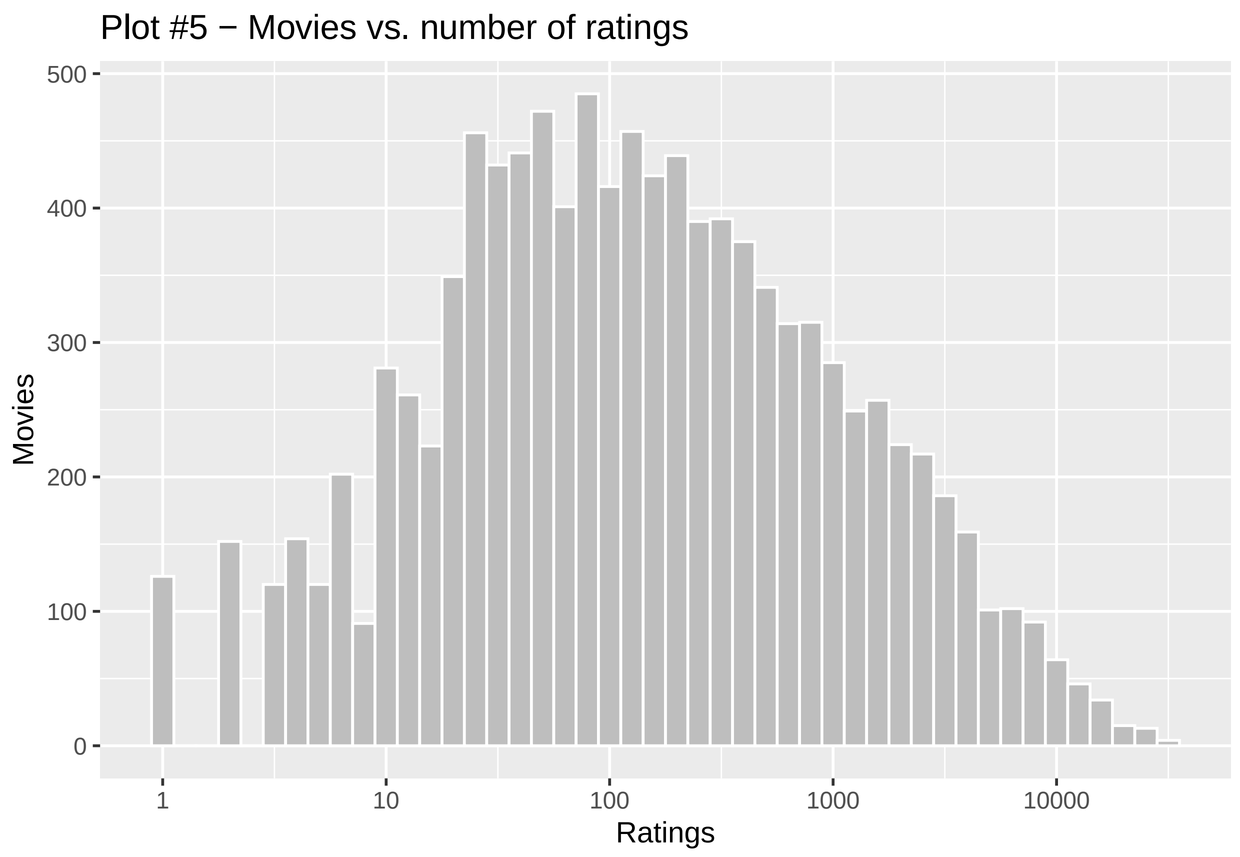
There is a clear change in the ratings scale in 2003, next step is to find the exact date.

Half stars ratings are in the training set since February 12, 2003.

Half stars ratings are in the validation set since February 18, 2003.

2.8 Further exploration of movies

2.8.1 Distribution of movies vs. number of ratings



2.8.2 Most and least rated movies

Table #6 - Most rated movies

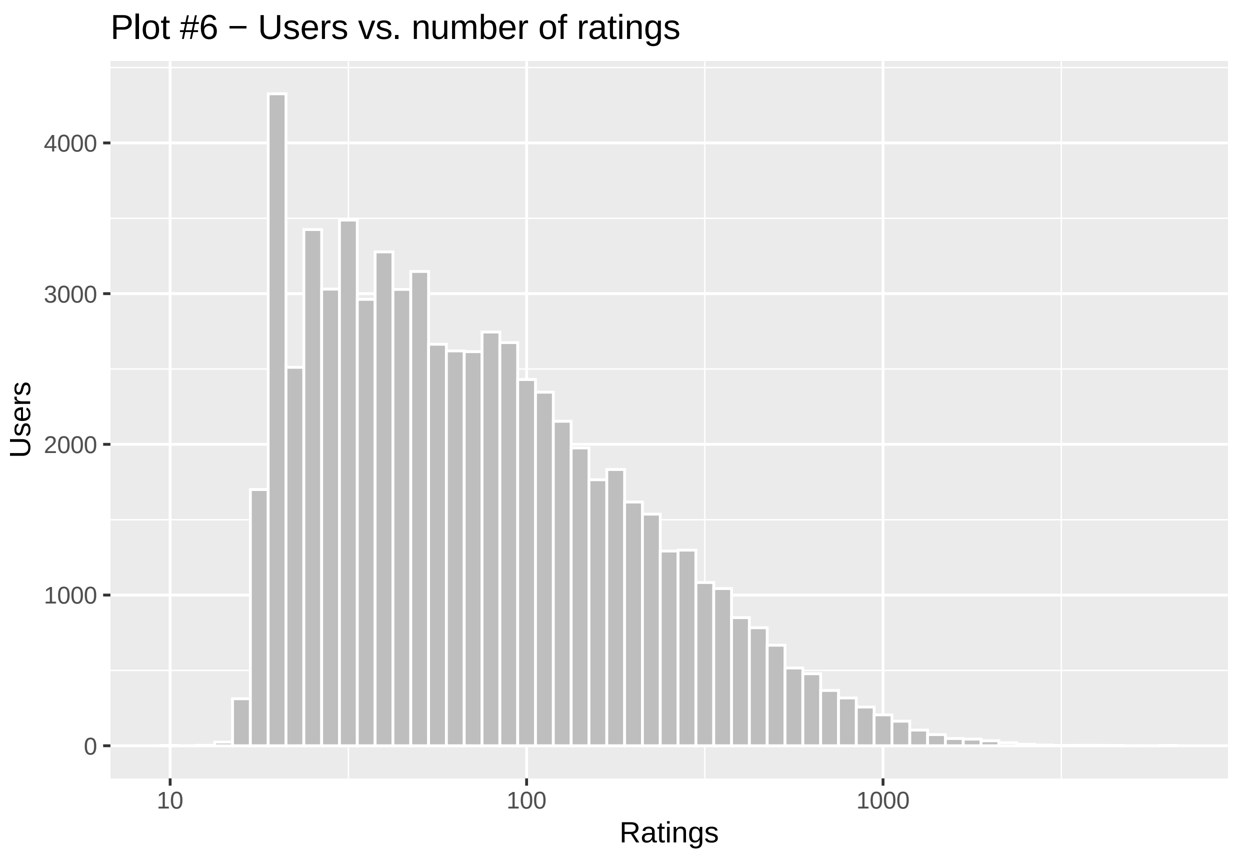
Movie title	Ratings
Pulp Fiction	31362
Forrest Gump	31079
Silence of the Lambs, The	30382
Jurassic Park	29360
Shawshank Redemption, The	28015
Braveheart	26212
Fugitive, The	25998
Terminator 2: Judgment Day	25984
Star Wars: Episode IV - A New Hope (a.k.a. Star Wars)	25672
Apollo 13	24284

Table #7 - Least rated movies

Movie title	Ratings
Quarry, The	1
Hellhounds on My Trail	1
Train Ride to Hollywood	1
Condo Painting	1
Big Fella	1
Stacy's Knights	1
Black Tights (1-2-3-4 ou Les Collants noirs)	1
Dog Run	1
Monkey's Tale, A (Les ChÃ¢teau des singes)	1
Won't Anybody Listen?	1

2.9 Exploration of Users

2.9.1 Distribution of users vs. number of ratings



2.9.2 Users having given the lowest and highest numbers of rankings

The next lines show which users have given the lowest and highest numbers of rankings:

```
## User with Id 62516 has given 10 ratings in 2005
```

```
## User with Id 59269 has given 6616 ratings from 2001 to 2009
```

2.10 Exploration of genres

2.10.1 Distribution of ratings vs. number of genres

Table #8 - Ratings per number of genres

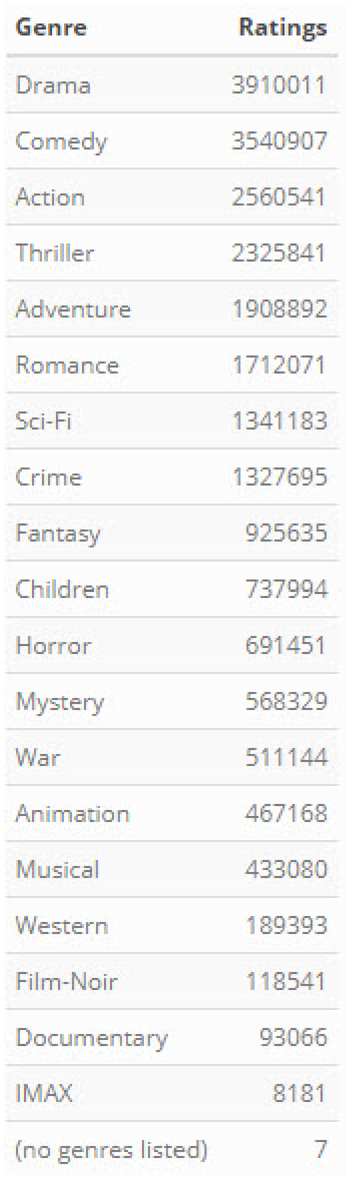
# Genres	# Ratings
3	2721142
2	2637701
1	1740490
4	1401379
5	421069
6	65998
7	11845
8	256

The following tables and plots "Ratings per genre" and "Rating means per genre" are screen shots from the Movielens.R output as the Movielens.Rmd execution of the corresponding code was not possible due to memory limitations.

For details, please refer to the code sections 1.10.2, 1.10.3 and 1.10.5 of Movielens.R code.

2.10.2 Ratings per genre

Table #9 - "Ratings per genre"



Genre	Ratings
Drama	3910011
Comedy	3540907
Action	2560541
Thriller	2325841
Adventure	1908892
Romance	1712071
Sci-Fi	1341183
Crime	1327695
Fantasy	925635
Children	737994
Horror	691451
Mystery	568329
War	511144
Animation	467168
Musical	433080
Western	189393
Film-Noir	118541
Documentary	93066
IMAX	8181
(no genres listed)	7

Figure 1: Ratings per genre

Plot #7

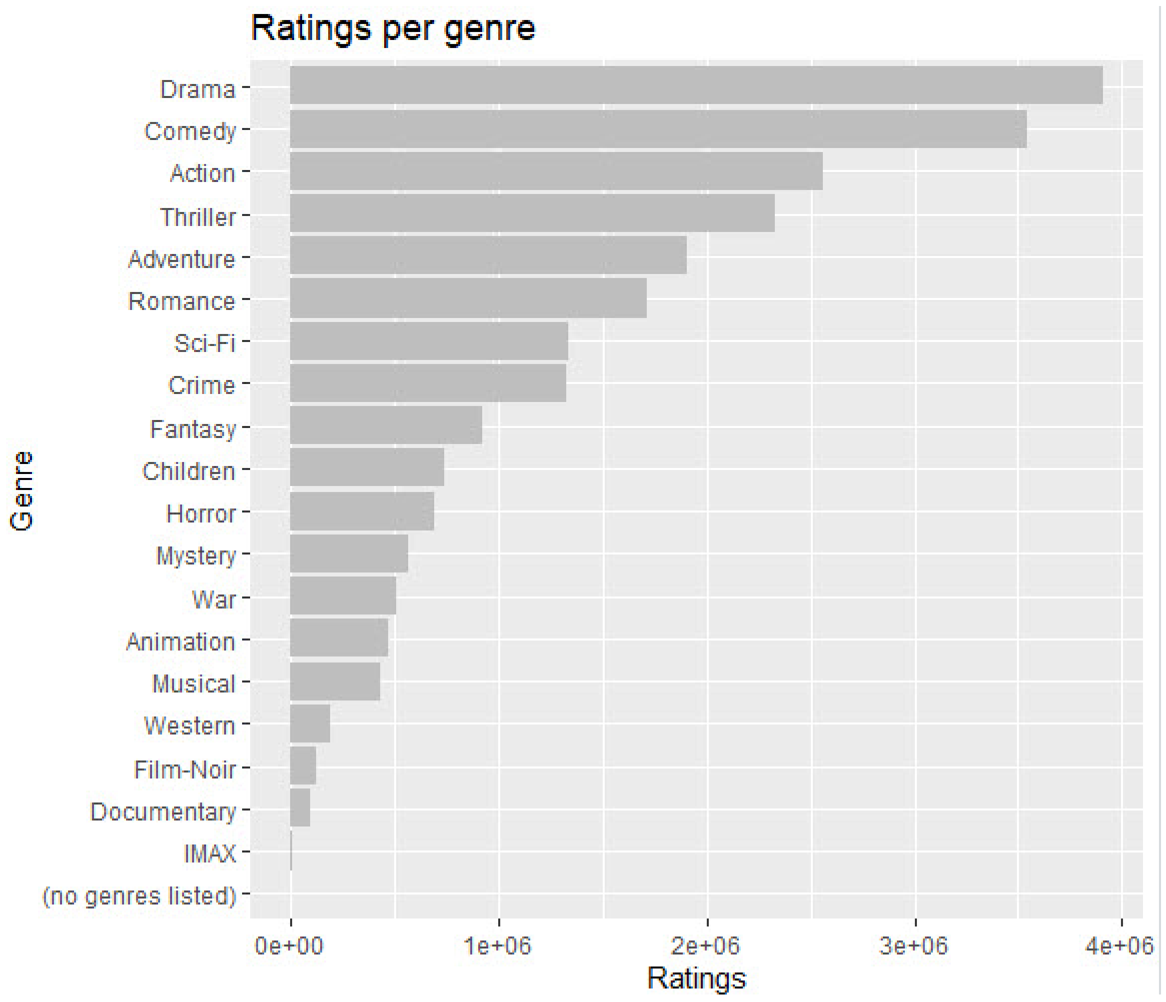
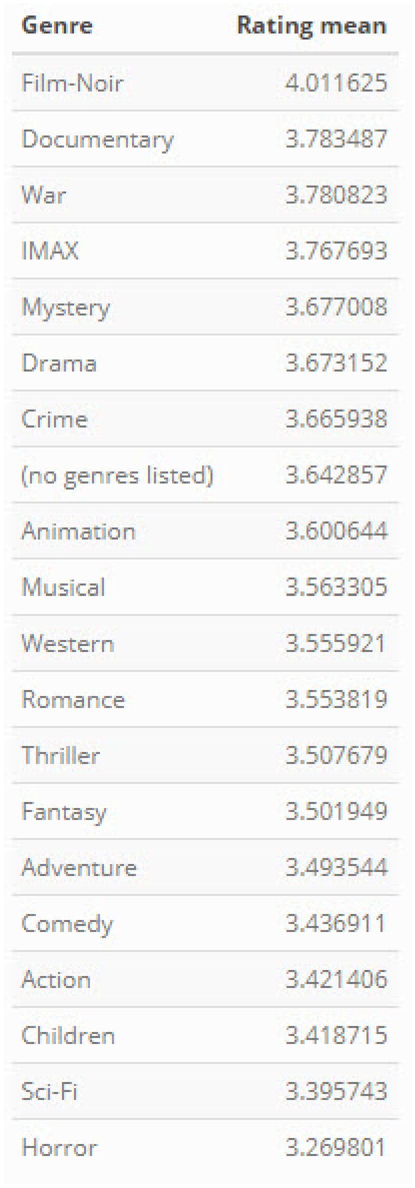


Figure 2: Ratings per genre

2.10.3 Rating means per genre

Table #10 - “Ratings means per genre”



Genre	Rating mean
Film-Noir	4.011625
Documentary	3.783487
War	3.780823
IMAX	3.767693
Mystery	3.677008
Drama	3.673152
Crime	3.665938
(no genres listed)	3.642857
Animation	3.600644
Musical	3.563305
Western	3.555921
Romance	3.553819
Thriller	3.507679
Fantasy	3.501949
Adventure	3.493544
Comedy	3.436911
Action	3.421406
Children	3.418715
Sci-Fi	3.395743
Horror	3.269801

Figure 3: Rating means per genre

Plot #8

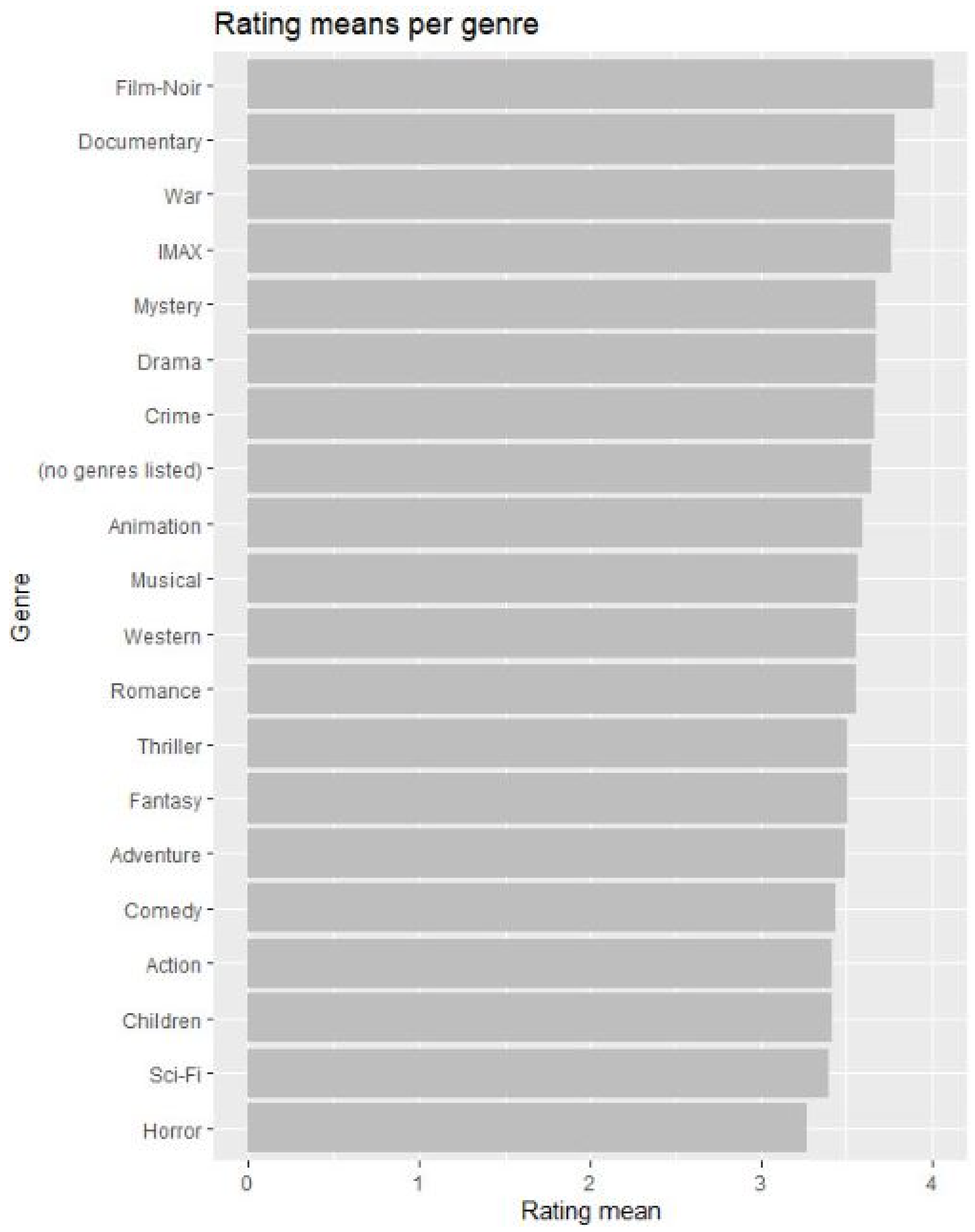


Figure 4: Rating means per genre

2.10.4 Handling of observations with "(no genres listed)"

The genre "(no genres listed)" is a candidate outlier: there is only one movie without genre - and this only in the training set, not in the test set.

The number of observations in the training data set has been reduced by 7 to 8999873

2.10.5 Conclusion on genre(s)

Exploration of genres indicates a very low correlation to ratings: 0.0220456

Variables related to genres can be dropped to reduce memory usage.

2.11 Structure of data sets after Data Exploration

2.11.1 Training data set

The training data set now has the following structure:
- **userId** : User identification key *(integer)*
- **movieId** : Movie identification key *(numeric)*
- **rating** : "Five Stars" type rating, ranging from 0.5 to 5 *(numeric)*
- **timestamp** : Time stamp of rating, in Posix format *(integer)*
- **year_released** : Release year of movie *(integer)*
- **year_rated** : Year of rating *(integer)*
- **movie_age** : Movie age, when rated *(integer)*

2.11.2 Test data set

The test data set structure is made identical to the training set:
- new movie age variable (year_rated - year_released)
- removal of the genre and title variables (they won't be needed anymore)

3 Models and their performance

Following an explanation on the performance metric "RMSE", the original "edx" data set will be split into training and test sets.

Additional splits will also be done to further analyze models before and after the change in ratings scale from "full stars (fs)" to "half stars (hs)" in February 2003.

The performance of the following model types will be evaluated
- Baseline "Naive" (just using the mean) models
- "Movie effect" models
- "Movie Age effect" models
- "Movie and User effects" models
- "Regularized Movie and User effects" models
- "Matrix Factorization" models

3.1 Performance evaluation metric: RMSE

The root-mean-square error error (RMSE) or root-mean-square deviation (RMSD) is a frequently used measure of the differences between values predicted by a model and the values observed.

RMSE is the square root of the average of squared errors and is therefore always non-negative. A value of 0 (almost never achieved in practice) would indicate a perfect fit to the data. In general, a lower RMSE is better than a higher one.

In our project, if N is the number of user-movie-rating combinations, $y_{u,i}$ is the rating for movie i by user u and $\hat{y}_{u,i}$ is our prediction, then the following formula applies for RMSE:

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

The effect of each error on RMSE is proportional to the size of the squared error; thus larger errors have a disproportionately large effect on RMSE. Consequently, RMSE is sensitive to outliers.[3]

3.2 Additional data preparation incl. splitting "edx" data into training and test sets

3.2.1 Split of "edx" data into training and test sets

The original "training_set" has 7199897 observations of 6 variables. This is 80.00 % of the original training set before splitting ("edx").

The original "test_set" has 1799947 observations of 6 variables. This is 20.00 % of the original training set before splitting ("edx").

3.2.2 Additional data preparation due to change in ratings scale in February 2003

In addition to the original training and test data sets,those will be split
into data sets reflecting the change in ratings scale in 2003
- "full stars (fs)" data sets from 1st rating date to date of change (excluded)
- "half stars (hs)" data sets from date of change (included) to last rating date

"full stars" data sets
- "edx_fs" has 4694894 observations of 6 variables. This is 52.17% of the original training set before splitting ("edx") around the rating scale change date.
- "training_set_fs" has 3755914 observations of 6 variables. This is 80.00 % of the "edx_fs" data set.
- "test_set_fs" has 938959 observations of 6 variables. This is 20.00 % of the "edx_fs" data set.
- "validation_fs" has 522993 observations of 6 variables. This is 52.30% of the original validation set ("validation").

"half stars" data sets
- "edx_hs" has 4304979 observations of 6 variables. This is 47.83% of the original training set before splitting ("edx") around the rating scale change date.
- "training_set_hs" has 3443982 observations of 6 variables. This is 80.00% of the "edx_hs" data set.
- "test_set_hs" has 860939 observations of 6 variables. This is 20.00 % of the "edx_hs" data set.
- "validation_hs" has 477006 observations of 6variables. This is 47.70% of the original validation set ("validation").

All training, test and validation data sets now have the following structure:
- **userId** : User identification key *(integer)*
- **movieId** : Movie identification key *(numeric)*
- **rating** : "Five Stars" type rating, ranging from 0.5 to 5 *(numeric)*
- **year_released** : Release year of movie *(integer)*
- **year_rated** : Year of rating *(integer)*
- **movie_age** : Movie age, when rated *(integer)*

3.3 Baseline "Naive" (just using the mean) models

This very simple models predict the same rating for all movies, independently of the user and movie. They uses the average of all ratings $\hat{\mu}$ as a constant rating for all movies and all users, and assume that all differences are explained by the random variation $\varepsilon_{u,i}$.

The corresponding formula is:

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

3.3.1 Get the mean ratings of the original and split training sets

Mean ratings of the original training sets "edx", "edx_fs" and "edx_hs" are: 3.5124724, 3.5558869 and 3.4651257.

Mean ratings of the split training sets "training_set", "training_set_fs" and "training_set_hs" are: 3.5123848, 3.5559893and 3.4652699.

3.3.2 "Naive" model's RMSE on the original data sets

The "Naive" model achieves an RMSE of 1.0612018 on the original data sets.

3.3.3 "Naive" model's RMSE on the "full stars" data sets

The "Naive" model achieves an RMSE of 1.0864602 on the "full stars" data sets.

3.3.4 "Naive" model's RMSE on the "half stars" data sets

The "Naive" model achieves an RMSE of 1.0306833 on the "half stars" data sets.

3.3.5 "Naive" model's RMSE on all three data sets and first observations

Table #11 - Results of "Naive" models

Model	**Data Sets**	**RMSE**
Naive (just the average)	original	1.0612018
Naive (just the average)	full stars	1.0864602
Naive (just the average)	half stars	1.0306833

The "Naive" model based on data after switch to "half stars" ratings has a lower RMSE than those based on data before this switch or based on original data.

We will see if the same applies to the next models' results.

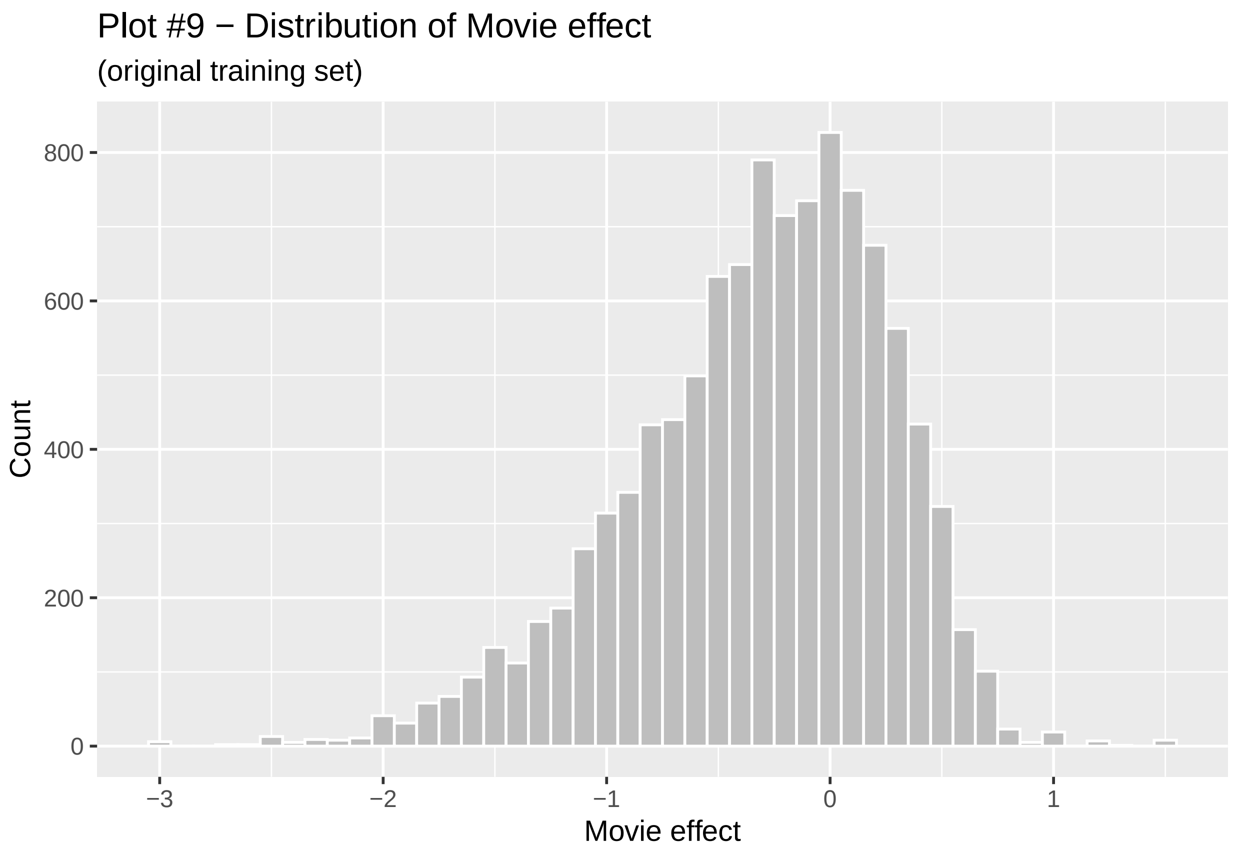
3.4 "Movie effect" models

Under the assumption (and experience, e.g. with IMDB) that some movies are usually better rated, we can model such a "Movie effect" by including a "movie bias" b_i *("b_movie" in my code)*, representing the average rating for movie i into the model.

The new formula is:

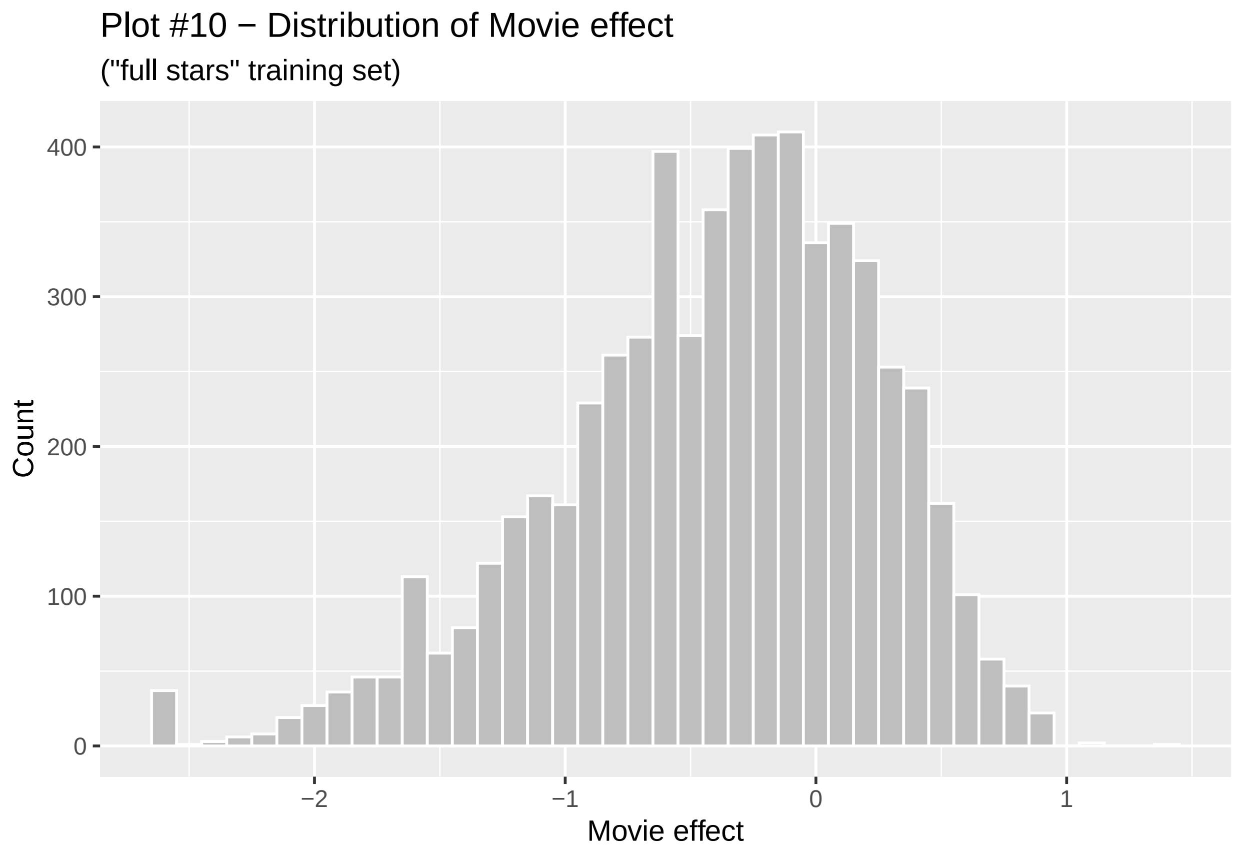
$$Y_{u,i} = \hat{\mu} + b_i + \epsilon_{u,i}$$

3.4.1 "Movie effect" model and its RMSE on the original data sets



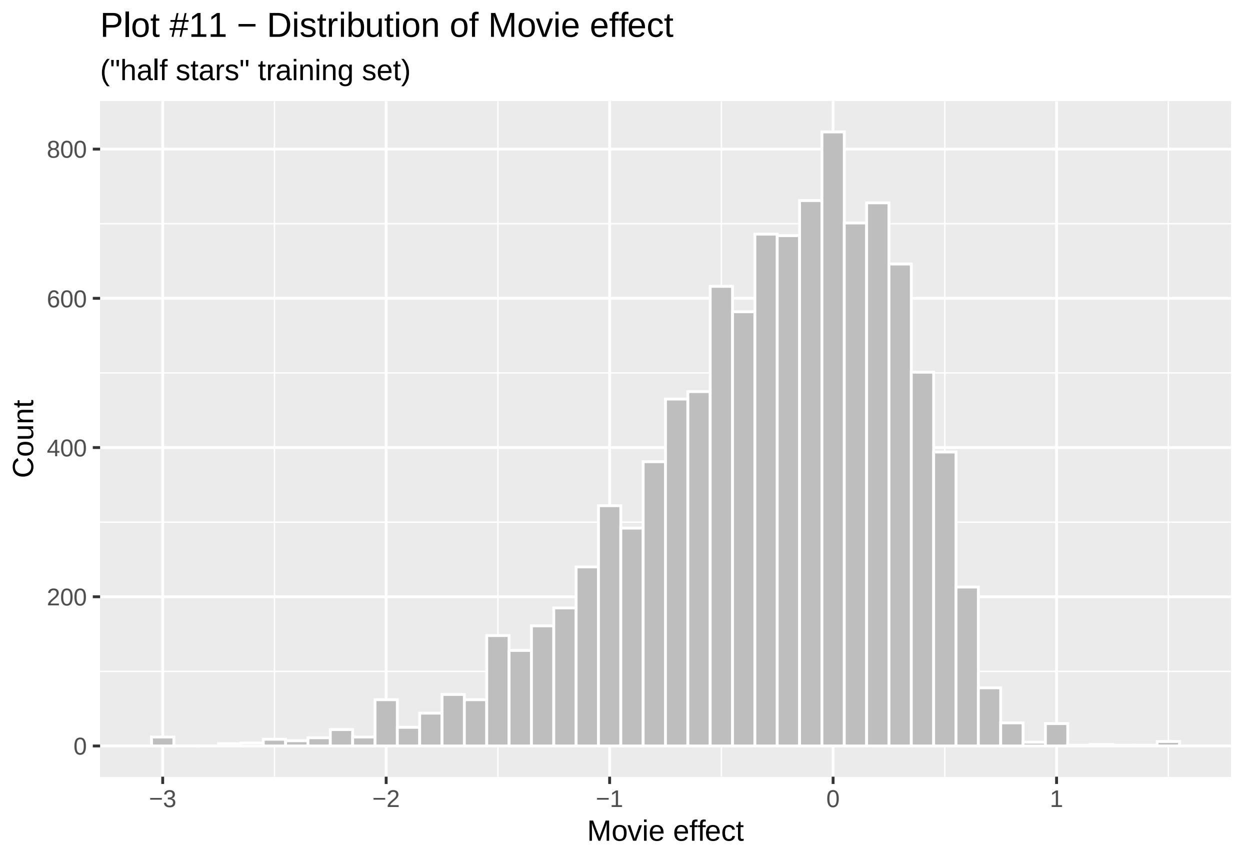
The "Movie effect" model achieves an RMSE of 0.9439100 on the original data sets.

3.4.2 “Movie effect” model and its RMSE on the “full stars” data sets



The “Movie effect” model achieves an RMSE of 0.9680030 on the “full stars” data sets.

3.4.3 "Movie effect" model and its RMSE on the "half stars" data sets



The "Movie effect" model achieves an RMSE of 0.9060957 on the "half stars" data sets.

3.4.4 "Movie effect" models' RMSE on all three data sets

Table #12 - Results of "Movie effect" models

Model	**Data Sets**	**RMSE**
Movie effect	original	0.9439100
Movie effect	full stars	0.9680030
Movie effect	half stars	0.9060957

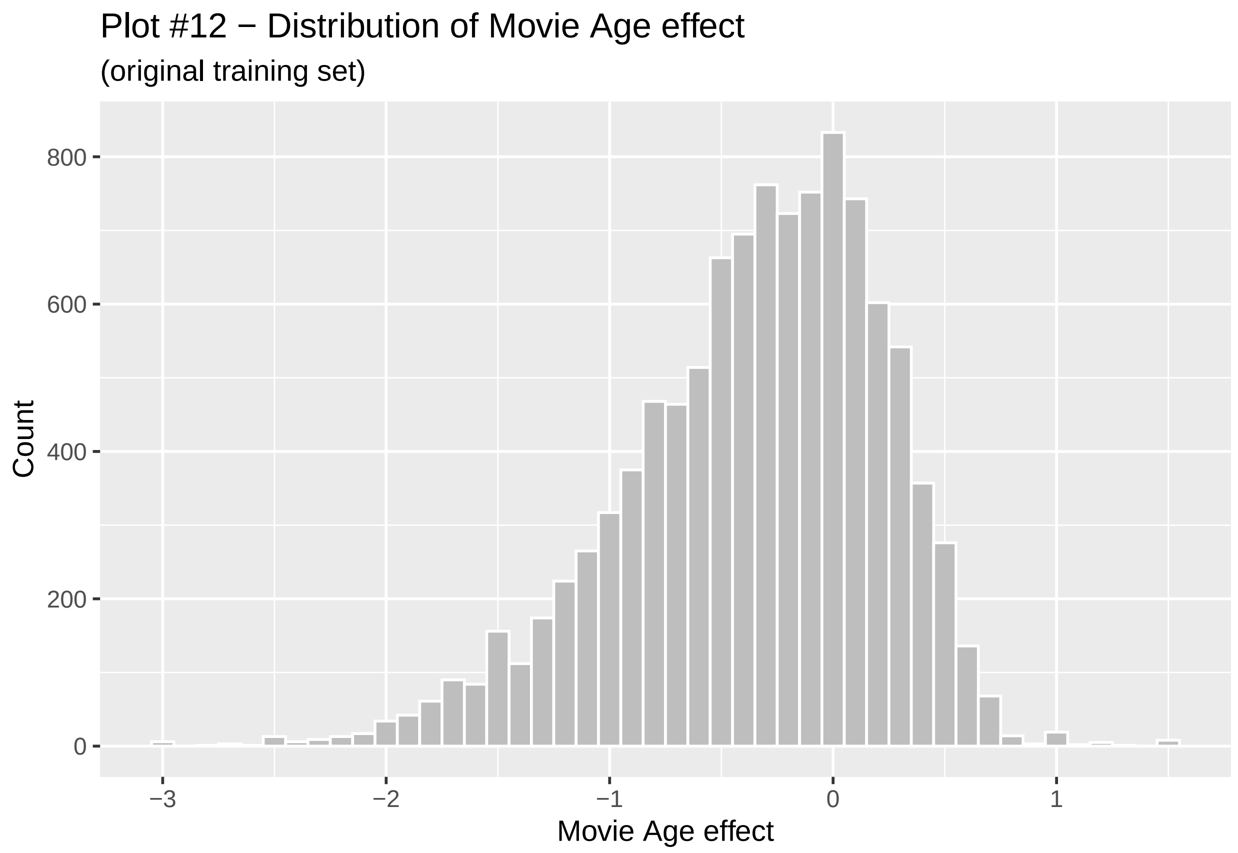
3.5 “Movie Age effect” models

Under the assumption that there is a relationship between the “age” of a movie *(the number of years between the movie release and its rating)*, we can model such a “Movie Age effect” by including a “movie bias” b_a *(“b_movie_age” in my code)*, representing the average rating for movies i of a given age into the model.

The new formula is:

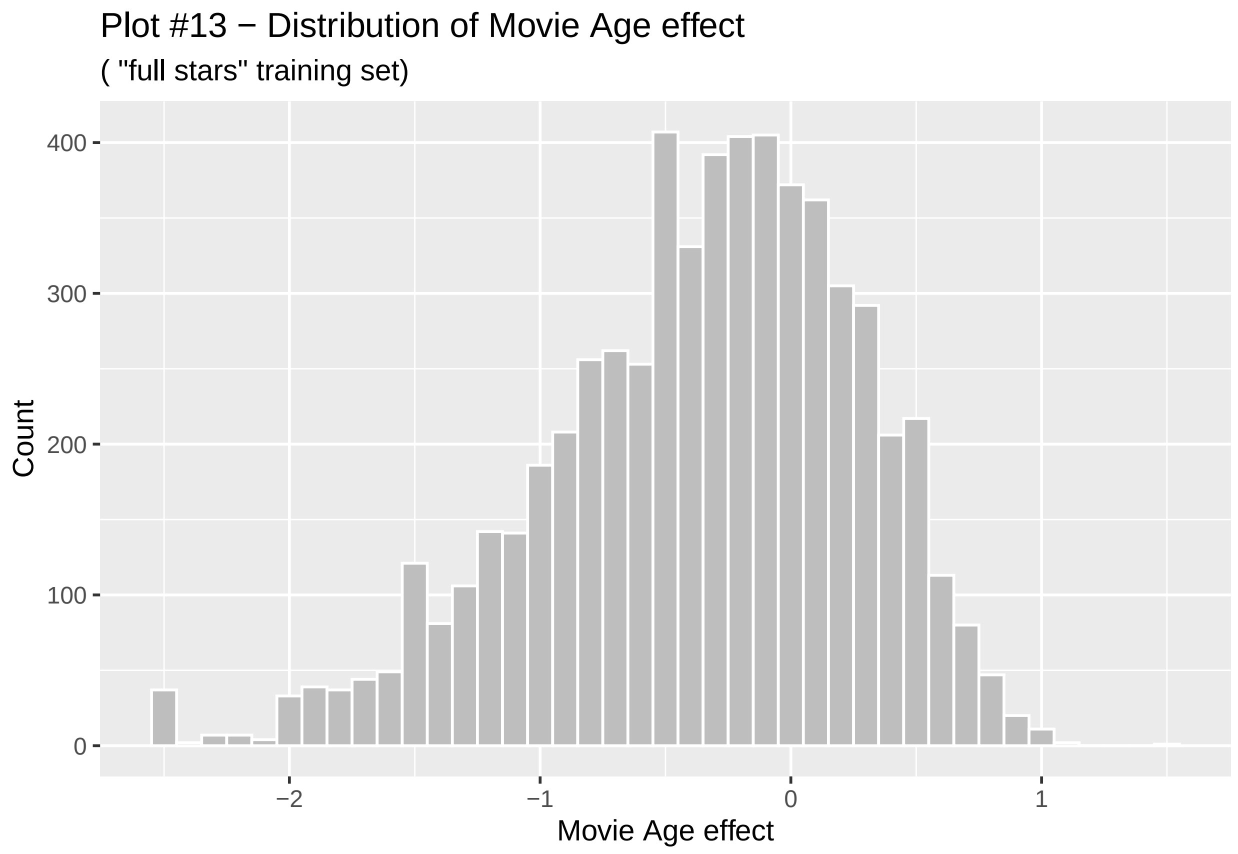
$$Y_{u,i} = \hat{\mu} + b_a + \epsilon_{u,i}$$

3.5.1 “Movie Age effect” model and its RMSE on the original data sets



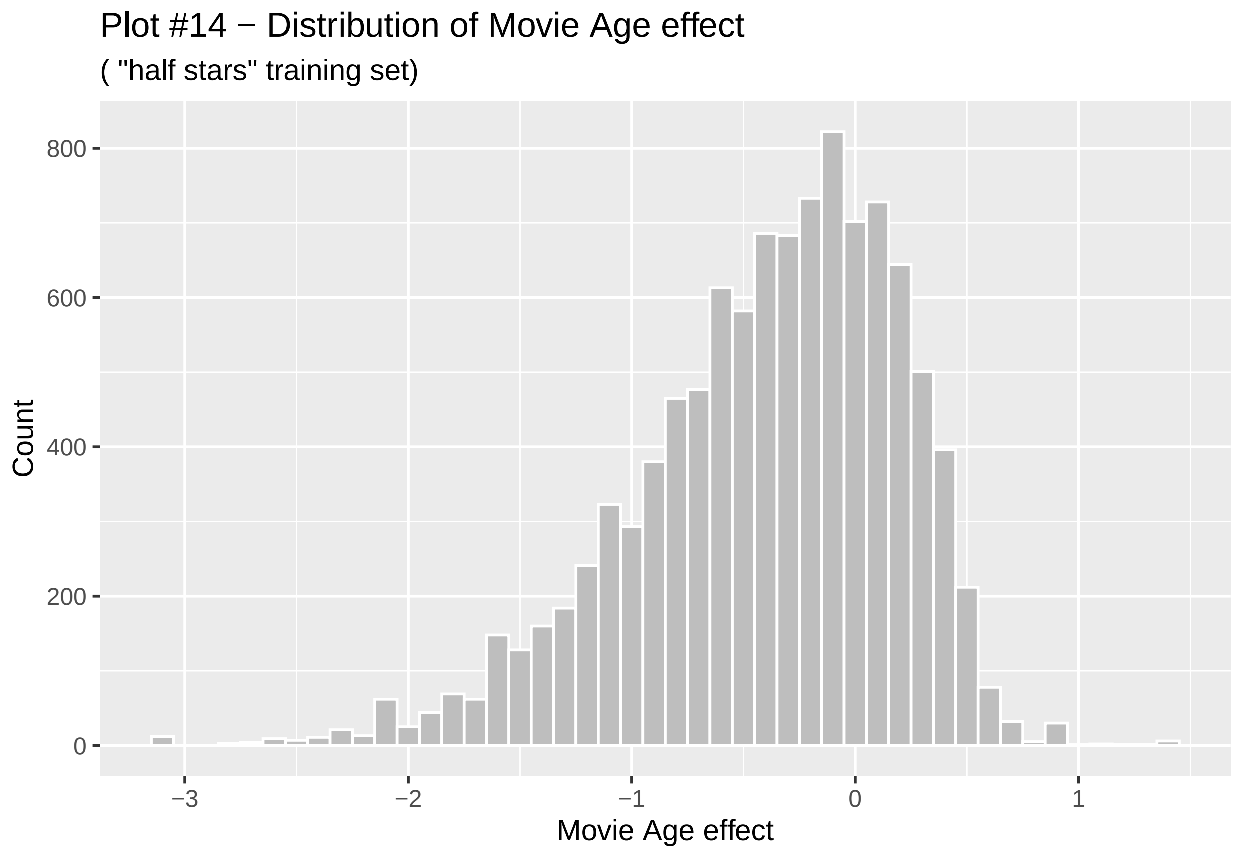
The “Movie Age effect” model achieves an RMSE of 0.9446426 on the original data sets.

3.5.2 “Movie Age effect” model and its RMSE on the “full stars” data sets



The “Movie Age effect” model achieves an RMSE of 0.9685611 on the “full stars” data sets.

3.5.3 "Movie Age effect" model and its RMSE on the "half stars" data sets



The "Movie Age effect" model achieves an RMSE of 0.9115944on the "half stars" data sets.

3.5.4 Observations on "Movie Age" models' performance and conclusions

Table #13 - Results of "Movie effect" and "Movie Age effect" models

Model	**Data Sets**	**RMSE**
Movie effect	original	0.9439100
Movie effect	full stars	0.9680030
Movie effect	half stars	0.9060957
Movie Age effect	original	0.9446426
Movie Age effect	full stars	0.9685611
Movie Age effect	half stars	0.9115944

Models based on data after switch to "half stars" ratings, have again lower RMSEs than those based on data before this switch or based on original data.

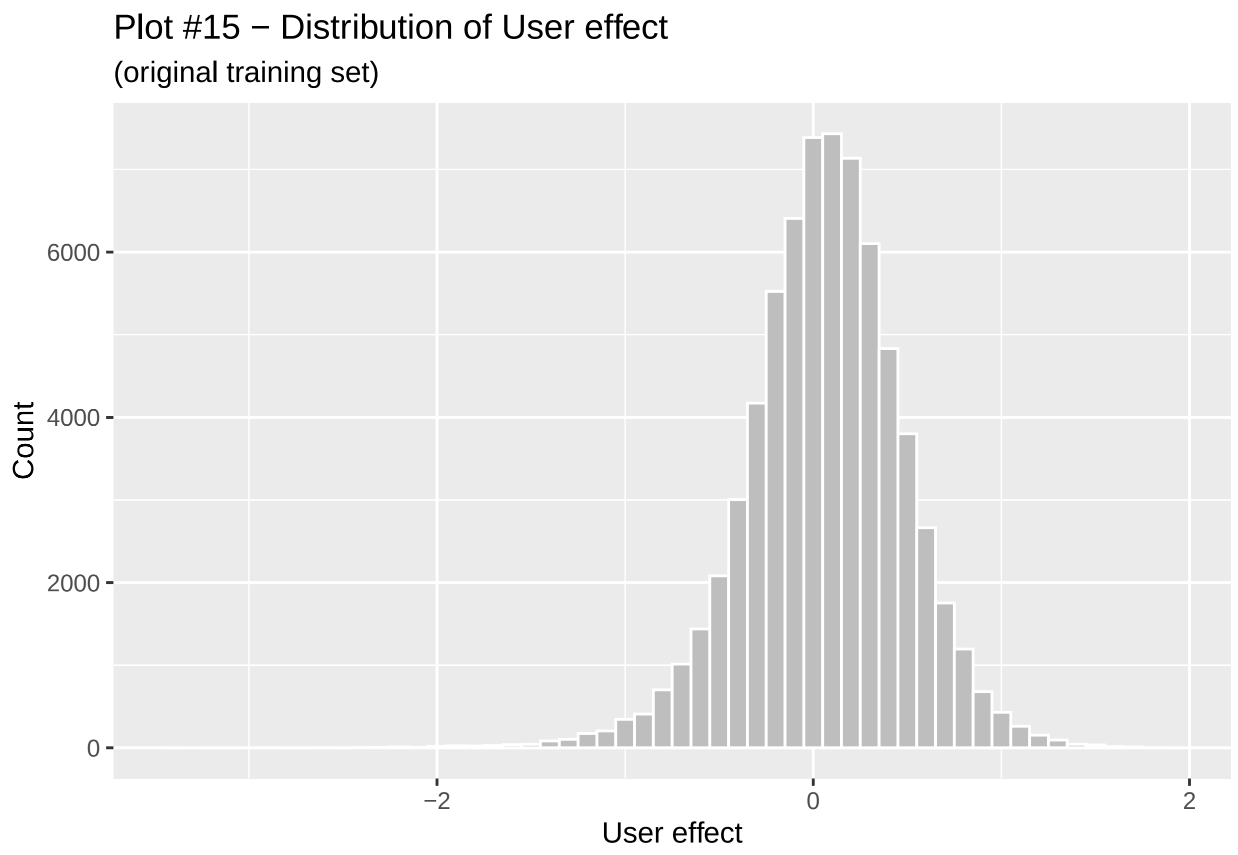
Modeling the "Movie Age" effect does not provide better results than modeling the "Movie" effect. Models including the "movie_age" variable won't therefore be further investigated.

3.6 “Movie and User effects” models

A further assumption (and experience no farther than within family and friends) is that users’ preferences can be very different and that this has an effect on the ratings. In addition to the “movie bias” b_i, we now include an “user bias” b_u, which leads to this new formula:

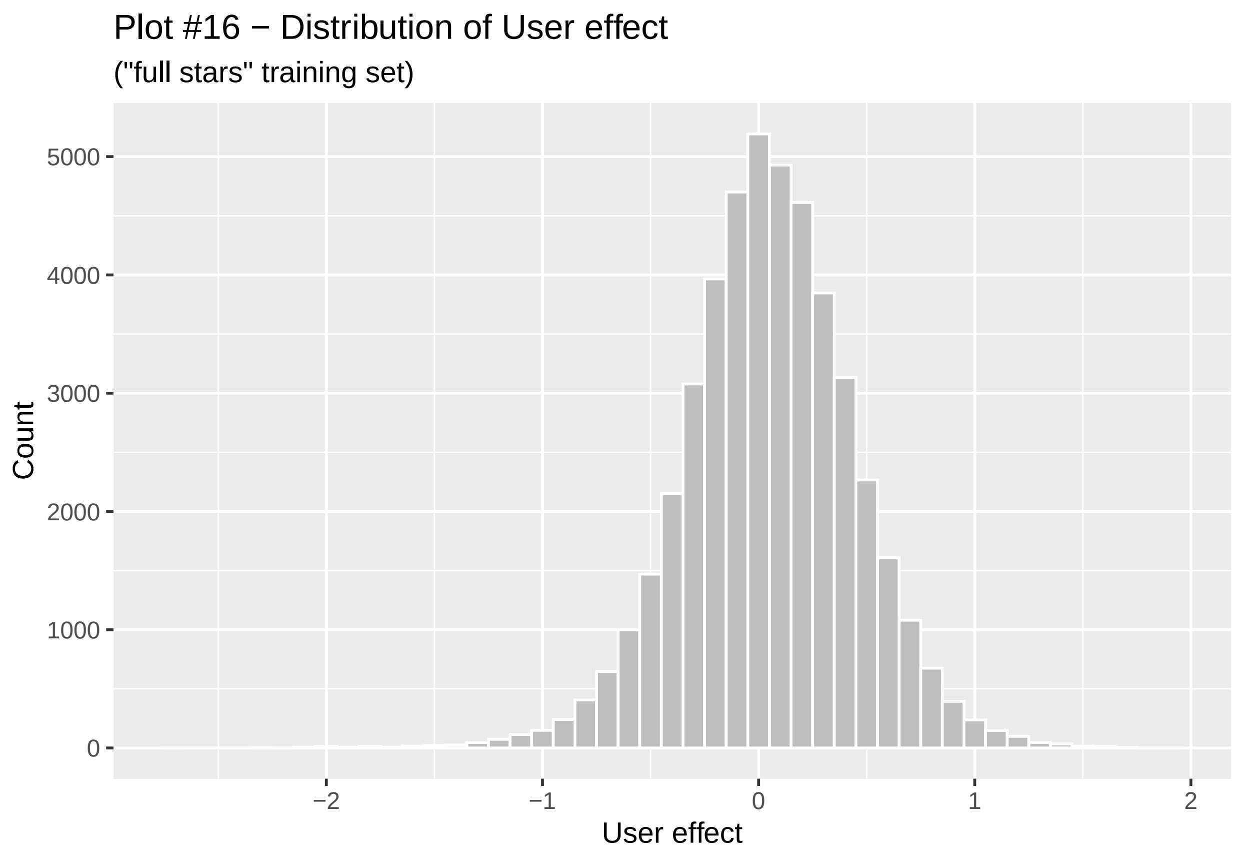
$$Y_{u,i} = \hat{\mu} + b_i + b_u + \epsilon_{u,i}$$

3.6.1 “Movie and User effects” model and its RMSE on the original data sets



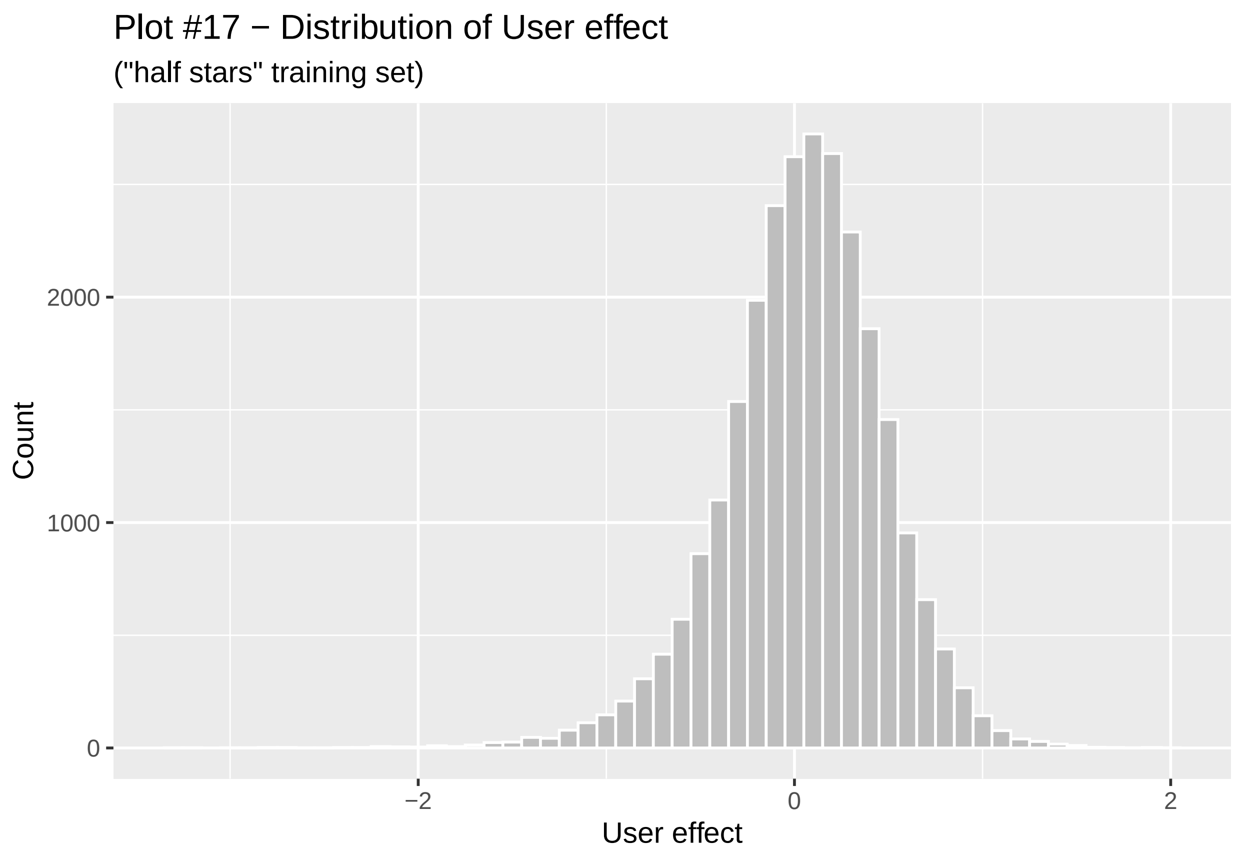
The “Movie and User effects” model achieves an RMSE of 0.8653498 on the original data sets.

3.6.2 "Movie and User effects" model its RMSE on the "full stars" data sets



The "Movie and User effects" model achieves an RMSE of 0.8982483on the "full stars" data sets.

3.6.3 "Movie and User effects" model its RMSE on the "half stars" data sets



The "Movie and User effects" model achieves an RMSE of 0.8214449 on the "half stars" data sets.

3.6.4 "Movie and User effects" models' RMSE on all three data sets and next observations on models' performance

Table #14 - Results of "Movie and User effects" models

Model	Data Sets	RMSE
Movie and User effects	original	0.8653498
Movie and User effects	full stars	0.8982483
Movie and User effects	half stars	0.8214449

Once again, the model based on data after switch to "half stars" ratings, has a lower RMSE than those based on data before this switch or based on original data.

The "Movie and User" effect model achieves an RMSE slightly under the 2nd goal RMSE value of 0.8654900. Let's see if those results can be further improved by applying Regularization. . .

3.7 “Regularized Movie and User effects” models

As mentioned before, larger errors have a disproportionately large effect on RMSE and *noisy estimates* (large estimates that come from small sample sizes) can cause such errors. The regularization permits to penalize such noisy estimates.

Regularization adds a penalty λ to movies with large estimates from a small sample size, i.e. to large values of b_i. This means, minimizing the following equation (with the second term being the penalty):

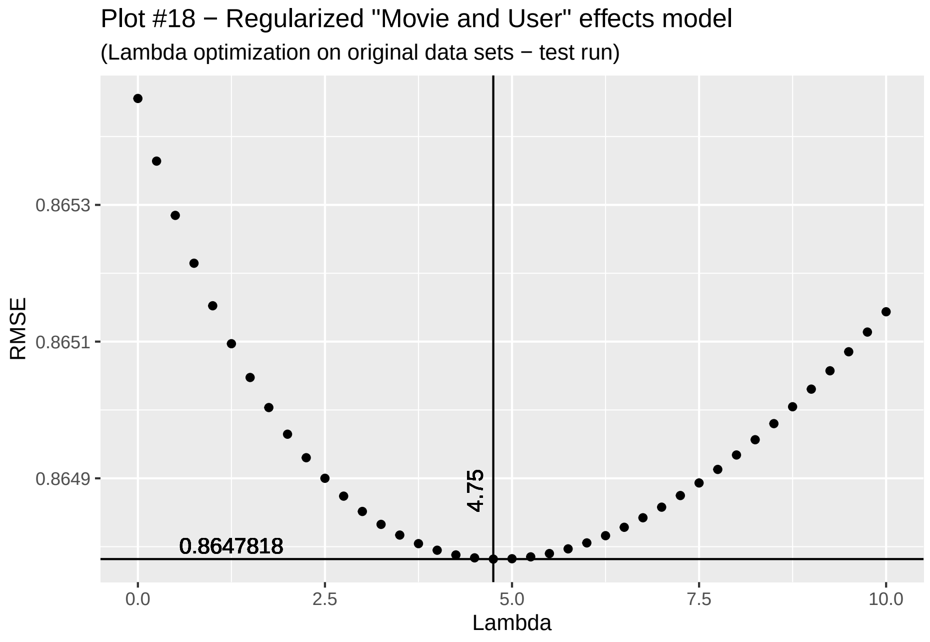
$$\frac{1}{N}\sum_{u,i}(y_{u,i} - \mu - b_i)^2 + \lambda \sum_{i} b_i^2$$

calculus reduces this problem to this equation, where n_i is a number of ratings b for movie i:

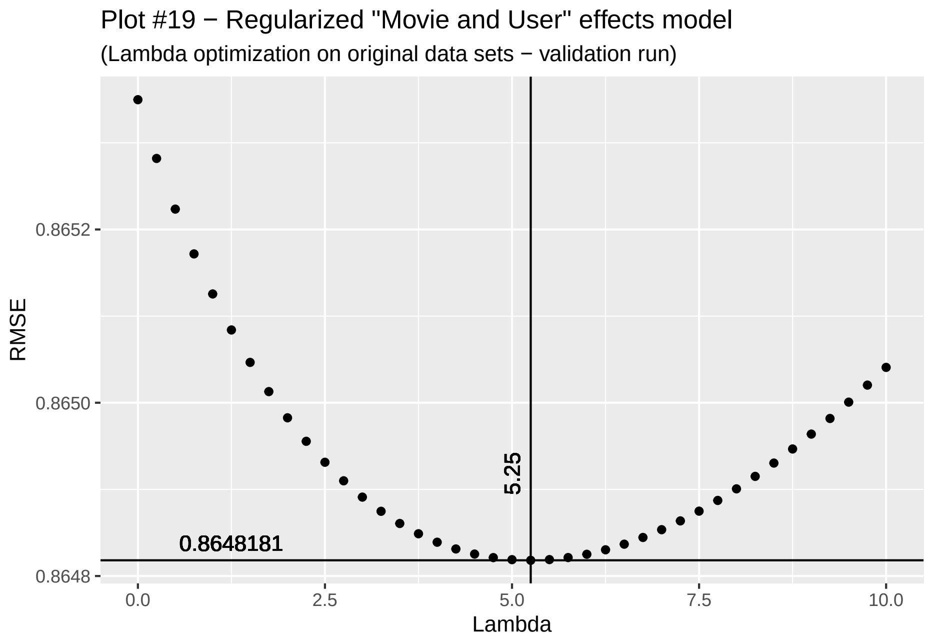
$$\hat{b_i}(\lambda) = \frac{1}{\lambda + n_i}\sum_{u=1}^{n_i}(Y_{u,i} - \hat{\mu})$$

Furthermore, λ is a tuning parameter that needs to be determined.

3.7.1 "Regularized Movie and User effects" model and its RMSE on the original data sets



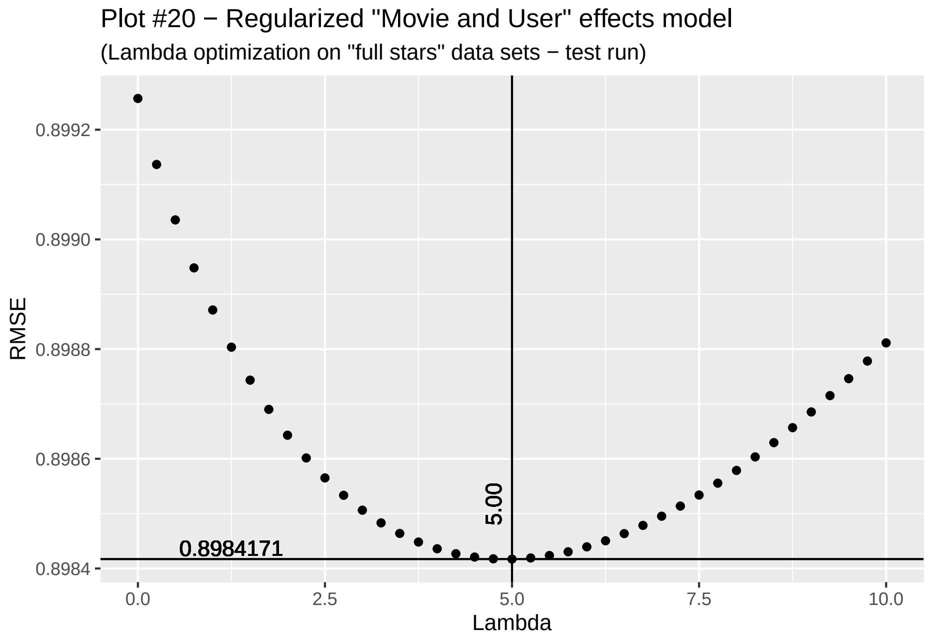
Test run: optimal regularization lambda: 4.75 with RMSE: 0.8647818.



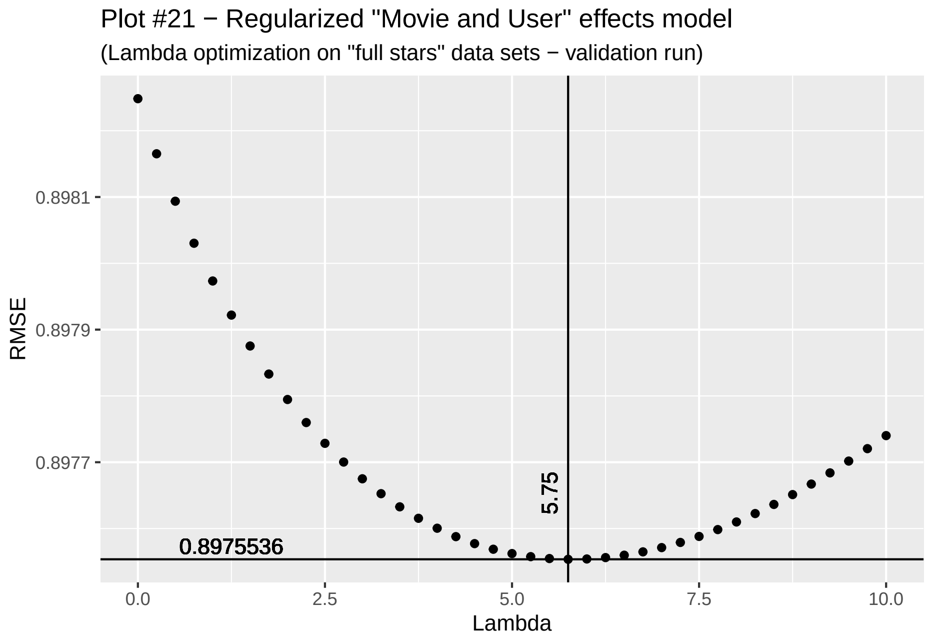
Validation run: optimal regularization lambda: 5.25 with RMSE: 0.8648181

The "Regularized Movie and User effects" model achieves an RMSE of 0.8648181 on the original data sets.

3.7.2 "Regularized Movie and User effects" model and its RMSE on the "full stars" data sets



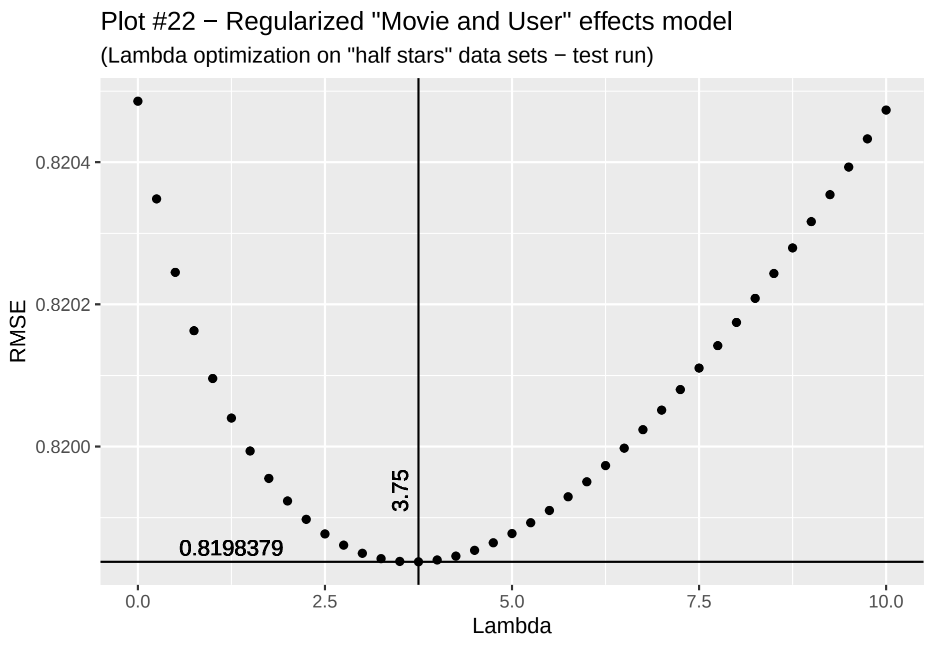
Test run: optimal regularization lambda: 5.00 with RMSE: 0.8984171.



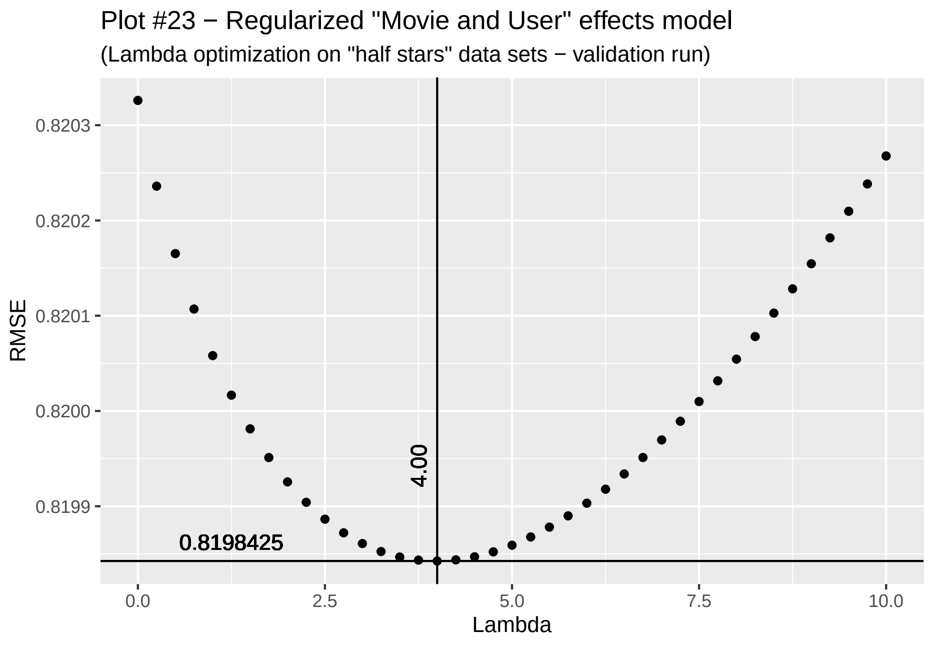
Validation run: optimal regularization lambda: 5.75 with RMSE: 0.8975536.

The "Regularized Movie and User effects" model achieves an RMSE of 0.8975536 on the "full stars" data sets.

3.7.3 "Regularized Movie and User effects" model and its RMSE on the "half stars" data sets



Test run: optimal regularization lambda: 3.75 with RMSE: 0.8198379.



Validation run: optimal regularization lambda: 4.00 with RMSE: 0.8198425.

The "Regularized Movie and User effects" model achieves an RMSE of 0.8198425 on the "half stars" data sets.

3.7.4 “Regularized Movie and User effects” models’ RMSE on all three data sets and next observations on models’ performance

Table #15 - Results of “Regularized Movie and User effects” models

Model	**Data Sets**	**RMSE**
Regularized Movie and User effects	original	0.8648181
Regularized Movie and User effects	full stars	0.8975536
Regularized Movie and User effects	half stars	0.8198425

As expected, the model based on data after the switch to “half stars” ratings, has a lower RMSE than those based on data before this switch or based on original data.

The “Regularized Movie and User effects” model achieves an RMSE slightly slightly better than the 4th and final goal RMSE value of 0.8649000 on both the original data sets (as required) and the “half stars” data sets.

That would be good enough, but let’s see if those results can be further improved by Matrix Factorization models…

3.8 “Matrix Factorization” models

The following “Matrix Factorization” models are built and evaluated according to the documentation of the “recosystem” package.[1], [2]

3.8.1 Generation of training, test and validation rating matrices for all data sets

Generate training, test and validation rating matrices based on observed values:
- Referring to [1], “Each cell with number in it is the rating given by some user on a specific item”
- The data file for training set needs to be arranged in sparse matrix triplet form, i.e., each line in the file contains three numbers: “user_index”, “item_index”, and “rating”

In order to avoid very long compute time(s), the models will be run on the training sets before split (in training and test) and validation sets.

3.8.2 2.8.2 “Matrix factorization” model and its RMSE on the original data sets

Tuning run #1, loss function minimum: 0.8061254.

Tuning run #2, loss function minimum: 0.8061113.

Table #16 - Results of “Matrix factorization” models on original data sets

Model	**Data Sets**	**RMSE**
Matrix Factorization, tuning run #1	original	0.7979920
Matrix Factorization, tuning run #2	original	0.7857343

Matrix Factorization allowed to substantially reduce RMSE. The best RMSE result (0.7857343) is quite better than the final goal of this exercise (RMSE < 0.8649000). However, this is at the cost of a very long processing time…

3.8.3 “Matrix factorization” model and its RMSE on the “full stars” data sets

Tuning run fs#1, loss function minimum: 0.8474256.

Tuning run fs#2, loss function minimum: 0.8457675.

Best RMSE result on “full stars” data sets: 0.8255740.

3.8.4 Matrix factorization" model and its RMSE on the “half stars” data sets

Tuning run hs#1, loss function minimum: 0.7626520.

Tuning run hs#2, loss function minimum: 0.7634794.

Best RMSE result on “half stars” data sets: 0.7505570.

3.8.5 Results of “Matrix Factorization” models on “full stars” and “half stars” data sets

Table #17 - Results of “Matrix factorization” models on “full stars” and “half stars” data sets.

Model	**Data Sets**	**RMSE**
Matrix Factorization, tuning run #1	full stars	0.8264217
Matrix Factorization, tuning run #2	full stars	0.8255740
Matrix Factorization, tuning run #1	half stars	0.7505570
Matrix Factorization, tuning run #2	half stars	0.7519677

4 Results

4.1 Summary of model results on original data sets

Table #18 - Results of all models on original data sets

Model	RMSE
Naive (just the average)	1.0612018
Movie effect	0.9439100
Movie Age effect	0.9446426
Movie and User effects	0.8653498
Regularized Movie and User effects	0.8648181
Matrix Factorization, tuning run #1	0.7979920
Matrix Factorization, tuning run #2	0.7857343

The RMSE goal of less than 0.8649000 is achieved with the "Regularized Movie and User effects" model and outperformed by the "Matrix Factorization" model, this however at the cost of a very long processing time.

4.2 Summary of model results on the "full stars" and "half stars" data sets

The original data sets contain ratings done on a "full stars" scale until February 12, 2003 and a "half stars" scale after this date. It seemed therefore appropriate to test the models' performance not only on the original data sets, but also on data sets before / after this change in the ratings scale.

Table #19 - Results of all models on "full stars" and "half stars" data sets

Model	RMSE "full stars"	RMSE "half stars"
Naive (just the average)	1.0864602	1.0306833
Movie effect	0.9680030	0.9060957
Movie Age effect	0.9685611	0.9115944
Movie and User effects	0.8982483	0.8214449
Regularized Movie and User effects	0.8975536	0.8198425
Matrix Factorization, tuning run #1	0.8264217	0.7505570
Matrix Factorization, tuning run #2	0.8255740	0.7519677

All models used in this script show, that performance is consistently better on "half stars" data sets. This could - intuitively - be related to the finer granularity of the rating scale.

5 Conclusion

This project was a challenging but very productive learning experience. It required to go back to a number of course material and also to look on the internet for help on what to install and configure my personal computing environment (e.g. Latex) to be able to produce this report…

If time, and most of all compute power, would have allowed, I would have liked to experiment with additional packages, such as e.g. "h2o", at least to see which baseline "AutoML" would have produced and which level of performance clustering algorithms or even deep learning algorithms would attain in the context of this project.

6 References

[1] https://cran.r-project.org/web/packages/recosystem/vignettes/introduction.html

[2] https://cran.r-project.org/web/packages/recosystem/recosystem.pdf

[3] https://en.wikipedia.org/wiki/Root-mean-square_deviation