MovieLens

Matej Sebenik

2022-05-29

1. Introduction:

The goal of this paper is the creation of a predictive algorithm, based on the "movielens" data set, that will predict the rating of movies. The metrics used to evaluate the algorithm is the RMSE (root mean square error).

The data set is available at **this link**.

The data set is previewed below:

```
##    userId movieId rating timestamp                         title
## 1:      1     122      5 838985046              Boomerang (1992)
## 2:      1     185      5 838983525               Net, The (1995)
## 3:      1     292      5 838983421               Outbreak (1995)
## 4:      1     316      5 838983392               Stargate (1994)
## 5:      1     329      5 838983392 Star Trek: Generations (1994)
## 6:      1     355      5 838984474       Flintstones, The (1994)
##                            genres
## 1:                Comedy|Romance
## 2:         Action|Crime|Thriller
## 3:  Action|Drama|Sci-Fi|Thriller
## 4:       Action|Adventure|Sci-Fi
## 5: Action|Adventure|Drama|Sci-Fi
## 6:       Children|Comedy|Fantasy
```

The data set is divided into 6 columns. These contain:
1. The Id of the user (numeric).
2. The Id the movie (numeric).
3. The rating (numeric, from 0.5 to 5, with 10 possible ratings (0.5, 1, 1.5,. . . , 4.5, 5)).
4. The timestamp (numeric, the time of the rating creation, expressed in seconds elapsed since 01.01.1970).
5. The title (character, also contains the year the movie was made).
6. Genres (character, can be a combination of many individual genres).

The data set has 10000054 entries/rows.

Key steps in the calculation of the predictive algorithm are as follows:
1. Data set download.
2. Initial data preparation and data cleaning.
3. Additional data modification.
4. Data visualizations.
5. Algorithm training.
6. Algorithm verification.

2. Methods, analysis, data preparation and visualizations:

2.1. Methods, analysis, data preparation

For the data set download, as well as initial data preparation and data cleaning, the R code was (thoughtfully) provided by the edx/Harvard course instructors/assistants. The compressed data was downloaded, unzipped, and the columns named. Additionally, the data set was split into the edx (training/test) set (90% of the original data) and validation set (only to be used for algorithm verification) (10% of the original data).

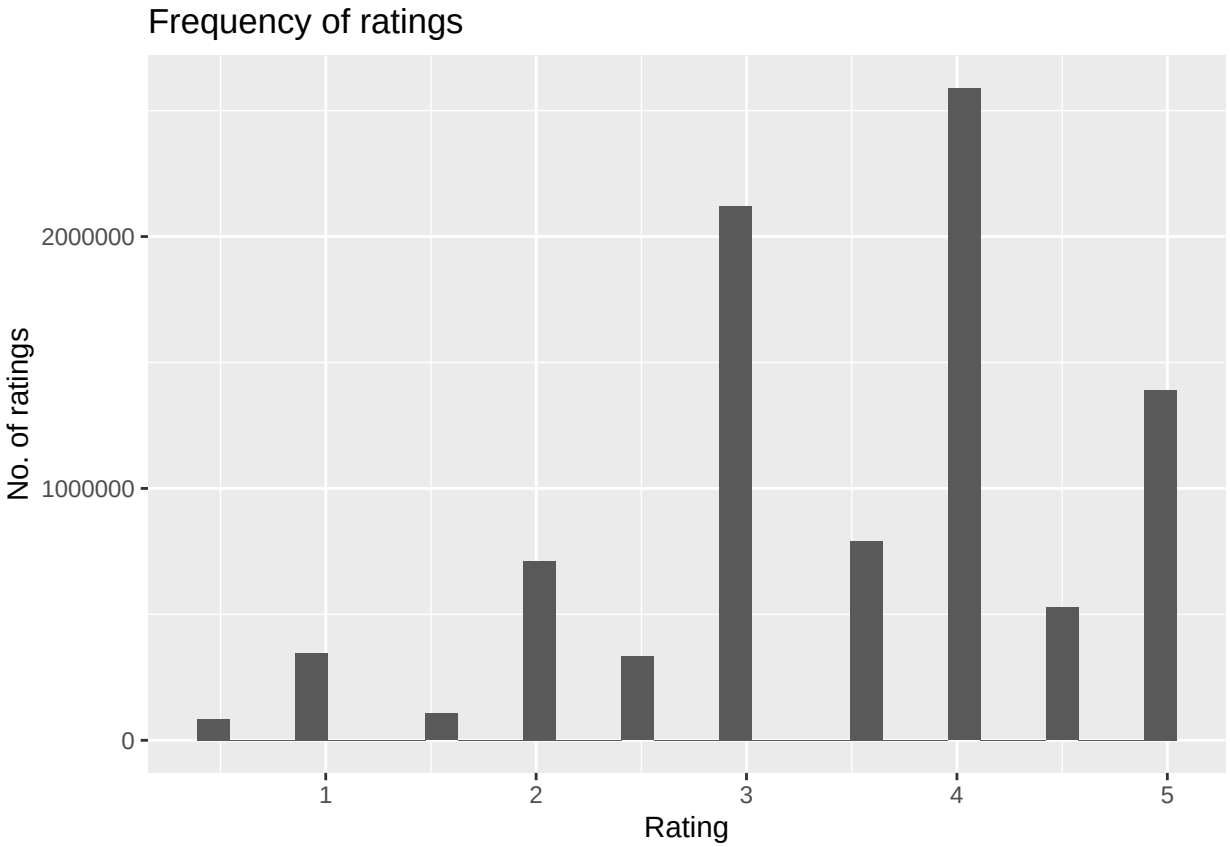
Through work on the algorithm, additional steps were taken with the goal of improving (lowering) the RMSE. These steps were applied equally to the edx and validation data sets described above. These steps (broadly) included:
1. Determining if an entry had only one genre assigned to it (e.g. "Action") as opposed to multiple genres (e.g. "Action|Adventure"). Single genres might get higher or lower ratings than multiple genres.
2. Genres are presented as words. This makes them unnecessarily long and unwieldy for use. Regressions using the original genres column would usually not run due to memory limitations. As such, the genres column was recoded to numeric values, which avoid the pitfalls listed above.
3. As stated above, each title of the movie also contains the year it was filmed/created. This value was extracted into a separate column, as movies created in different years might get different ratings (alluding to a "Golden Age" of movie making, when movies were simply better than at other times).
4. For each genre, the number of ratings, mean of ratings and standard deviation of ratings was calculated. Certain genres might be more popular than other, getting more/higher ratings than others.
5. Similarly, the number of ratings, mean of ratings and standard deviation was calculated for individual movies, at the risk of increased overtraining.
6. The mean of ratings and number of ratings was obtained per user (at risk of overtraining).
7. In addition, the mean of ratings and number of ratings was calculaterd per the user and genre (to take into account the possibility that individual users prefer certain genres to other. . .).
8. Lastly the year of rating was extracted from the timestamp column, roughly following the same logic as the year of movie creation mentioned under the third point (above).

After further data preparation, the final data set used looks like this:

```
##    userId genre_number movieId rating timestamp single_genre single_genre_title
## 1:      1           49     370      5 838984596            0               <NA>
## 2:      1          139     420      5 838983834            0               <NA>
## 3:      1          160     329      5 838983392            0               <NA>
## 4:      1          163     377      5 838983834            0               <NA>
## 5:      1          212     122      5 838985046            0               <NA>
## 6:      1          297     185      5 838983525            0               <NA>
##    movie_year n_genre mean_genre sd_genre n_movie mean_movie  sd_movie
## 1:       1994   51289   2.995379 1.086490    7331   2.957032 1.0270864
## 2:       1994   32549   3.234769 1.063535    8079   2.752568 0.9778991
## 3:       1994   20366   3.154399 1.003706   14550   3.337457 0.9424496
## 4:       1994   41025   3.334247 1.023317   21361   3.526871 0.9275253
## 5:       1992  365468   3.414486 1.034261    2178   2.858586 0.9324804
## 6:       1995  102259   3.461289 1.020066   13469   3.129334 0.9548463
##    year_timestamp n_user mean_user n_user_genre mean_user_genre
## 1:           1997     19         5            1               5
## 2:           1997     19         5            1               5
## 3:           1997     19         5            1               5
## 4:           1997     19         5            1               5
## 5:           1997     19         5            1               5
## 6:           1997     19         5            1               5
```

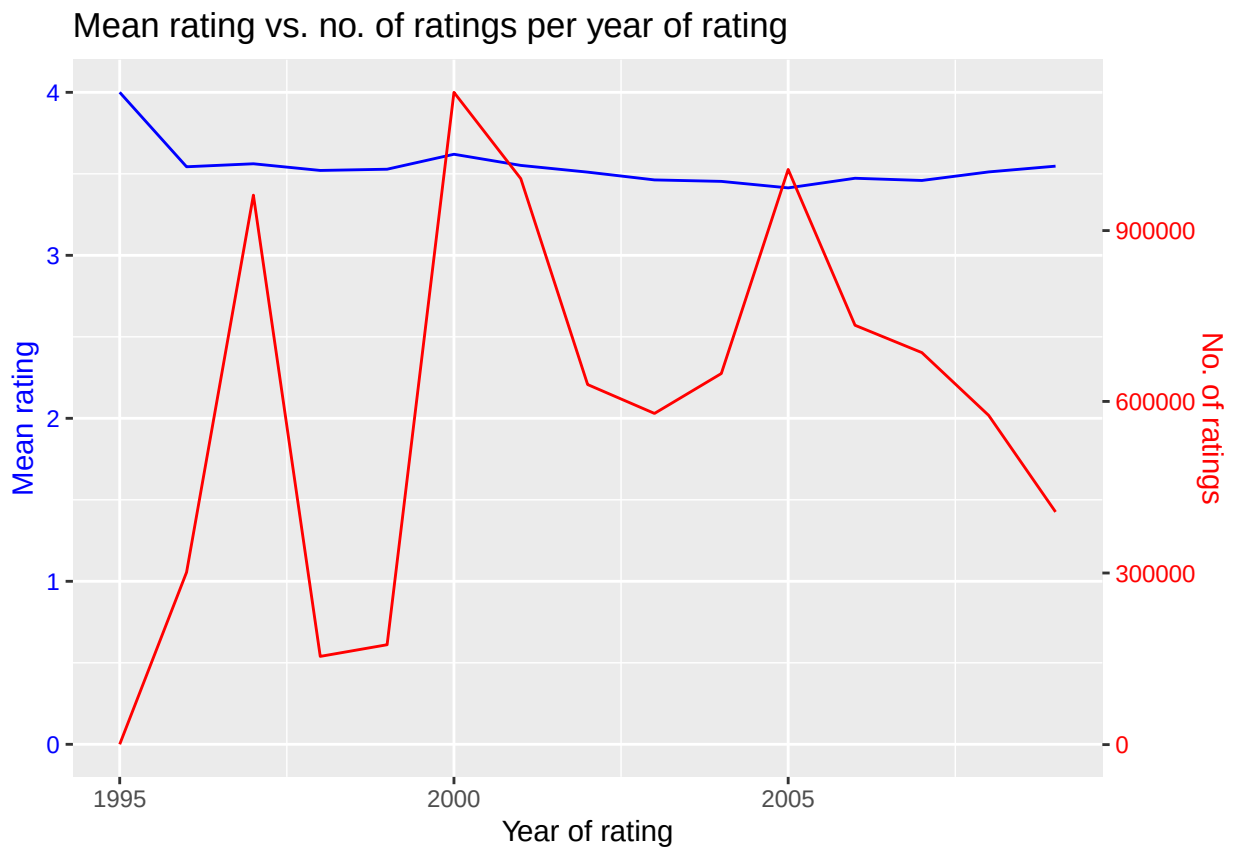
2.2. Data visualizations:
Several data visualizations were considered and prepared, with the aim of getting acquainted with the data. These visualizations are provieded below, along with the relevant descriptions and insights.

1. Histogram of ratings:



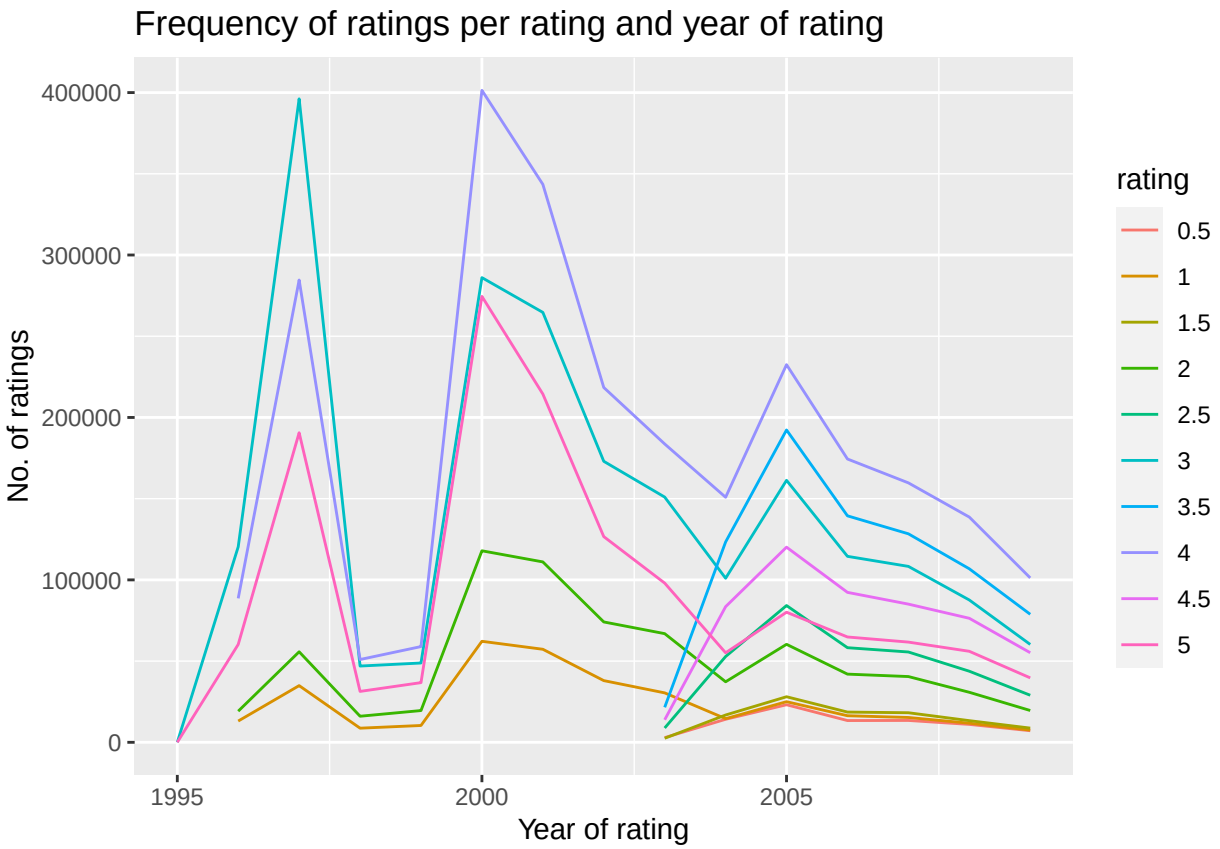
This histogram shows the number of ratings per each rating level, useful for general rating distribution overview. Two findings stand out; the "whole number" ratings (e.g. 1, 3) are much more likely to be chosen than those with decimal points (e.g. 1.5, 3.5). The distribution also does not seem to be entirely normal; for that the peak of ratings should be at rating value 3, not 4 (the rating curve is skewed to the right).

2. Number of ratings and mean rating per year of rating:



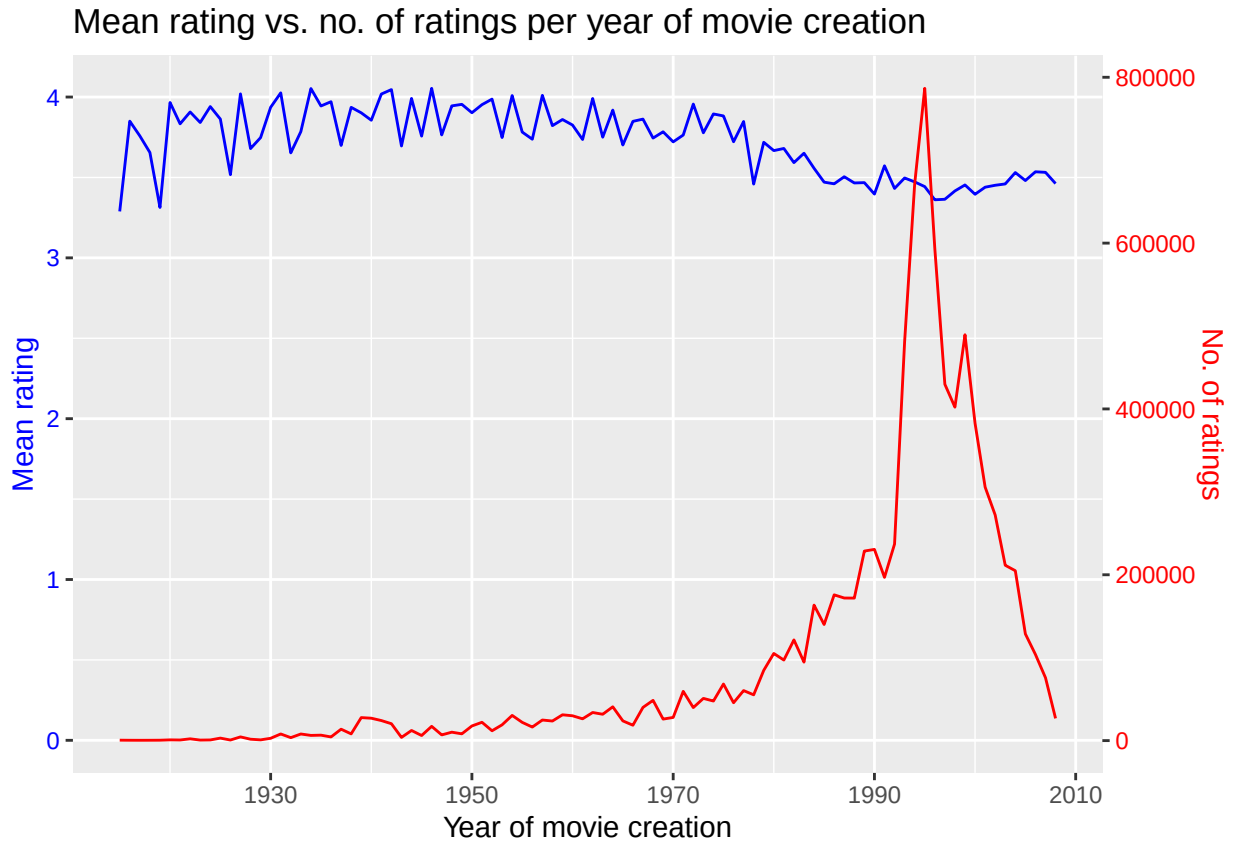
This graph shows the number of ratings and mean rating per year of rating. Curiously, it shows that the number of ratings per year of rating is highly irregular, swinging wildly without any obvious pattern. The mean rating, in contrast, remains fairly constant (aside from an initial high point, which corresponds to extremely low number of ratings). There is also no obvious connection between the no. of ratings and mean of ratings - sometimes higher number of ratings corresponds to higher ratings (year 2000), sometimes to lower ratings (year 2005), and sometimes there is no visible impact (year 1997).

3. Number of ratings per year of rating and individual rating:



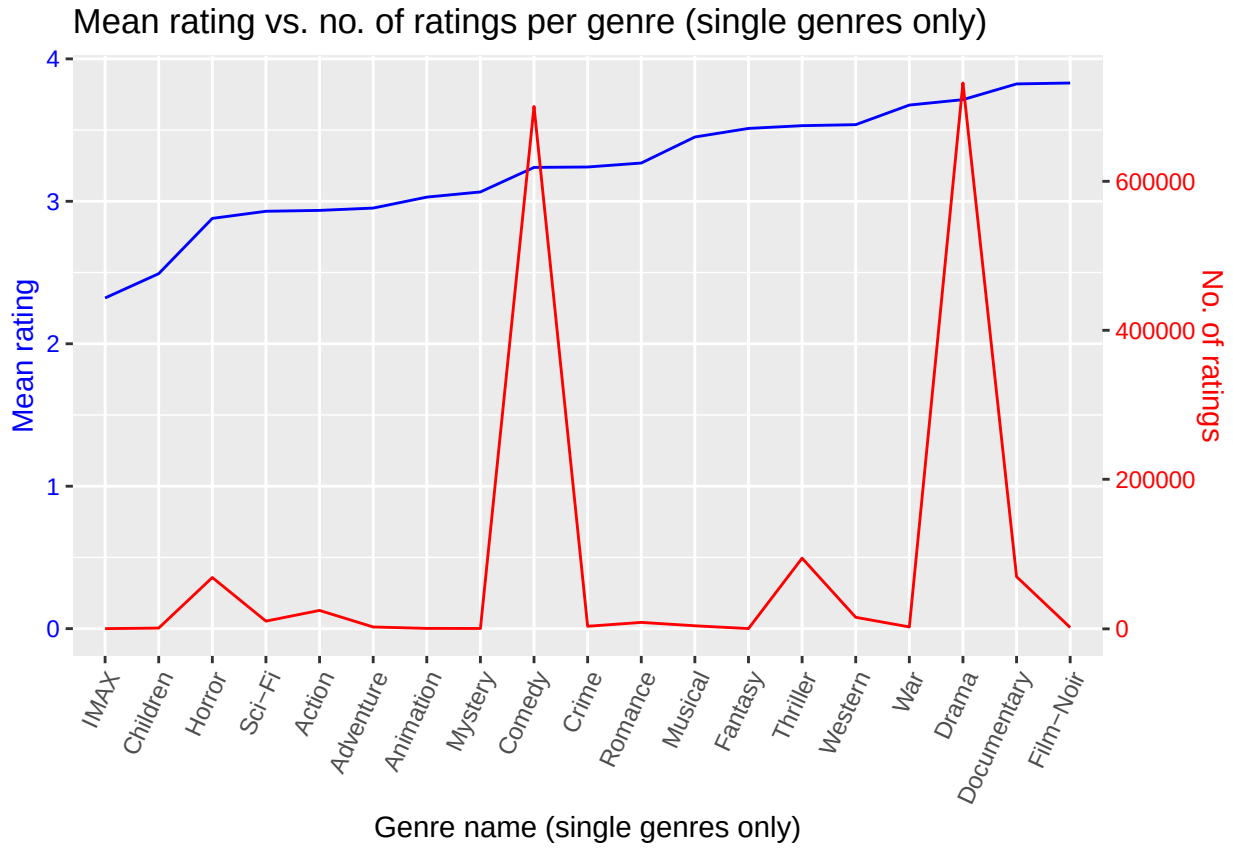
This graph shows the number of ratings per year of rating, divided into lines for individual ratings. Interestingly, this graph shows that the "decimal point ratings" (e.g. 2.5) only came into use in 2003. The year 2004 was the only year when the "decimal point ratings" and "whole number ratings" did not move in concert. Otherwise the individual rating numbers all follow the same dynamic of rises/falls in frequency of ratings.

4. Number of ratings and mean rating per year of movie creation:



This graph shows the number of ratings and mean rating per year of movie creation. This graph shows a connection between the year of movie creation and the number and mean of associated ratings. Up to cca. year 1970 of movie creation, the number of ratings per year was relatively low, and the mean rating was relatively high. After 1970, the dynamic changes, with a huge spike of ratings for newer movies and corresponding lower mean ratings. Speculatively speaking, users might have rated their favorite old movies, giving them higher ratings. They also rated current (possibly trending) movies, finding them less appealing. This graph might provide some support for the "Golden Age" theory described above (second chapter, third line).

5. Number of ratings and mean rating per genre name (single genres only):



This graph shows the number of ratings and mean rating per genre name (only for movies which have only one genre assigned). Sorted by the mean rating.

The graph shows little association of no. of ratings with ratings per genre name. Comedy has a large number of ratings but average mean ratings, while Drama also has high number of ratings and a high mean rating.

3. Modelling approach and results:

3.1. Modelling approach:

The goal of the algorithm is to predict the rating of movies. As noted above, ratings fall into 10 distinct "categories", e.g. 0.5, 1, 1.5 etc. Crucially, users cannot rate an individual movie outside these ten categories, e.g. 1.73. This makes the rating a categorical type of variable, and not a continuous variable.

The distinction mentioned above is relevant since each type of variable has specific machine learning approaches associated with them. Thus, categorical types of variables generally imply the use of classification methods, such as LDA, QDA, decision trees and random forests. Crucially, they also imply **accuracy** as the metric used to evaluate the performance of the algorithm. On the other hand, continuous variables are associated with various regressions and use the RMSE for performance evaluation.

According to the facts above, the correct metric to evaluate our model should be accuracy, but is actually RMSE. As the staff/assistants of the course helpfuly point out, RMSE actually has benefits as opposed to accuracy, in that if we predict the value of 3.99 for an actual number of 4, the accuracy metric would still report 0 accuracy (as the numbers are not identical). This accuracy value of 0 would be equal for values of 3.99 and 0.5, even though the value 3.99 is much closer to 4 than 0.5. The RMSE, on the other hand, would return a more favorable assessment of value 3.99 as opposed to the value 0.5.

Accordingly, and in line with the demands of the course, RMSE was used for evaluation and associated regressions were used to build the algorithm. Having insufficient time and knowledge to test all the regression methods available in the caret package (there are over a hundred), the most popular were tested, these being the linear regression, general linear model, localized regression and the k nearest neighbor method. Of

these, localized regression (loess) slightly outperformed the linear regression and the general linear model. K nearest neighbor performed worst, taking huge amounts of time to complete and often returning errors or empty results (no doubt due to the lack of knowledge on the part of the data scientist involved). Thus, localized regression method was chosen for the finalization of the algorithm.

The data set was not further divided into training and test sets. Instead, cross validation was used for initial model evaluation. The reason for this was two fold; firstly, cross validation returned slightly better RMSE values. Secondly and more importantly, it was also significantly faster than conventional approach with training and test sets. Spans and degrees were not regulated. Lowering the span toward 0 and setting the degree to 1 improved RMSE marginally, but occasionally caused errors during subsequent work. Constant reductions of the span also fueled concerns of overtraining.

As for the data used in the algorithm, the following were used:
1. genre_number (recoded name of genre),
2. userId,
3. movieId (recoded name of movie),
4. timestamp,
5. n_genre (number of ratings for the genre),
6. mean_genre (mean rating for the genre),
7. single_genre (1 if the movie only has one genre associated with it),
8. movie_year (the year of the movie creation),
9. n_movie (how many times the movie was rated),
10. mean_movie (the mean of ratings for the movie),
11. n_user (how many times the user rated),
12. mean_user (the mean of ratings for each user),
13. n_user_genre (how many times the combination of user and genre was rated).

Several other variables were considered and discarded, such as:
1. year_timestamp (using the timestamp yields a better RMSE),
2. sd_genre (standard deviation of the ratings in associated genre, discarded since the improvement to the RMSE using it was marginal at best),
3. sd_movie (standard deviation of the ratings in associated movie, same reason as for sd_genre),
3. mean_user_genre (a huge contributor of overtraining).

3.2. Results: The initial RMSE, calculated on the training/test data set, is a fine **0.86892**. The final RMSE, calculated by predicting the rating of the validation data set with the developed algorithm, revealed an equally fine result at ***0.84467***.

Unfortunately, the time and resources it takes to develop the algorithm leave much to be desired. The runtime of the final model training takes at least 30 minutes, on a very high end personal computer. The calculation itself is also memory demanding, at cca. 25 GB of RAM, though the prediction itself on the validation (final) data set was more tolerable.

4. Conclusion: This report presented the development and results of the algorithm used to predict the ratings of movies in the MovieLens data set. The results are fine, although they do raise the question of overtraining (this question was raised in the comments with the staff and left unanswered).

If this project should ever be revisited, aditional effort should be levied at regularisation, while attempting to minimise or remove the importance/use of means of genre, movie and user in the algorithm.

Best regard,

Matej Sebenik
Ljubljana
Slovenia
EU