Data Science Capstone MovieLens project

Ulises Bonilla

21/5/2021

Introduction

For this project, I will create a movie recommendation system using the `MovieLens` dataset. The objective of this project is to train a machine learning algorithm using a large data set, the training set, to predict the rating of movies. The model is ultimately tested on a second data set, the validation set. However, cross validation is recommended prior to making use of the validation set. The metric employed for this task is the RMSE. That is, the RMSE will be used to evaluate how close the predictions are to the true values in the validation set.

The report of this project is organized as follows. First, the present section, the Introduction, describes the data and the goal of the project. Additionally, the same section includes a quiz about the data, which helped me familiarize with the data. Second, the Analysis section addresses data cleaning, data wrangling and feature engineering, the main exploration and visualizations of the data, and finally the models. Third, the Results section simply provide the outcome of the project in terms of the chosen metric. And, lastly, the Conclusion section gives a brief summary of the report, its limitations and the work that could be undertaken in the future.

Data

Here is a glimpse of how the original data looks:

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

As observed above, the columns in the data are:

- `userId` that contains the unique identification number for each user.
- `movieId` that contains the unique identification number for each movie.
- `rating` that contains the rating of one movie by one user. Ratings are made on a 5-Star scale with half-star increments.
- `timestamp` that contains the timestamp for one specific rating provided by one user.
- `title` that contains the title of each movie including the year of the release.
- `genres` that contains a list of pipe-separated of genre of each movie.

At a first glance, what stands out is that the `genres` column is not one-to-one, in the sense that a single movie

may have more than one genre. Another thing that I notice is that the `title` column actually contains two pieces of information, the title and the date of the movie in parenthesis. Lastly, I also take into consideration that the `timestamp` column contains data such as the date and time at which each review was done.

Quiz

To get to know the data better I answer the following quiz of 8 questions.

1. How many rows and columns are there in the edx dataset?

   The data set has 9,000,055 rows and 7 columns.

   In principle, this fact could represent a problem due to computing limitations.

2. How many zeros were given as ratings in the edx dataset? How many threes were given as ratings in the edx dataset?

   There are no zeros given as ratings in the data set. There are 2,121,240 threes given as ratings in the data set.

   These are good news because zeros may create issues with the data. Also, that many threes could hint towards the location of the mean in the data.

3. How many different movies are in the edx dataset?

   There are 10,677 different movies in the data.

4. How many different users are in the edx dataset?

   There are 69,878 different users in the data.

   Despite that movies and users are not a continuous variable, but a factor, there are way too many of each to be taken as factor/categorical features. Instead, a mean deviation approach would be a better choice.

5. How many movie ratings are in each of the following genres in the edx dataset?

- *Drama: there are 3,910,127 movie ratings in the Drama genre.*
- *Comedy: there are 3,540,930 movie ratings in the Comedy genre.*
- *Thriller: there are 2,325,899 movie ratings in the Thriller genre.*
- *Romance: there are 1,712,100 movie ratings in the Romance genre.*

   In principle, it seems that, unlike users or movies, genres could be treated as categorical. However, this is not certain.

6. Which movie has the greatest number of ratings?

   The movie which has the greatest number of ratings among Forrest Gump,Jurassic Park, Pulp Fiction, Shawshank Redemption and Speed 2: Cruise Control is Pulp Fiction (1994).

7. What are the five most given ratings in order from most to least?

   The most given rating is 4, followed by 3,5,3.5 and 2 in that order.

   These are not necessarily good news. If the ratings in `rating` are too unbalanced, the models might be biased towards the most common ratings in the sample.

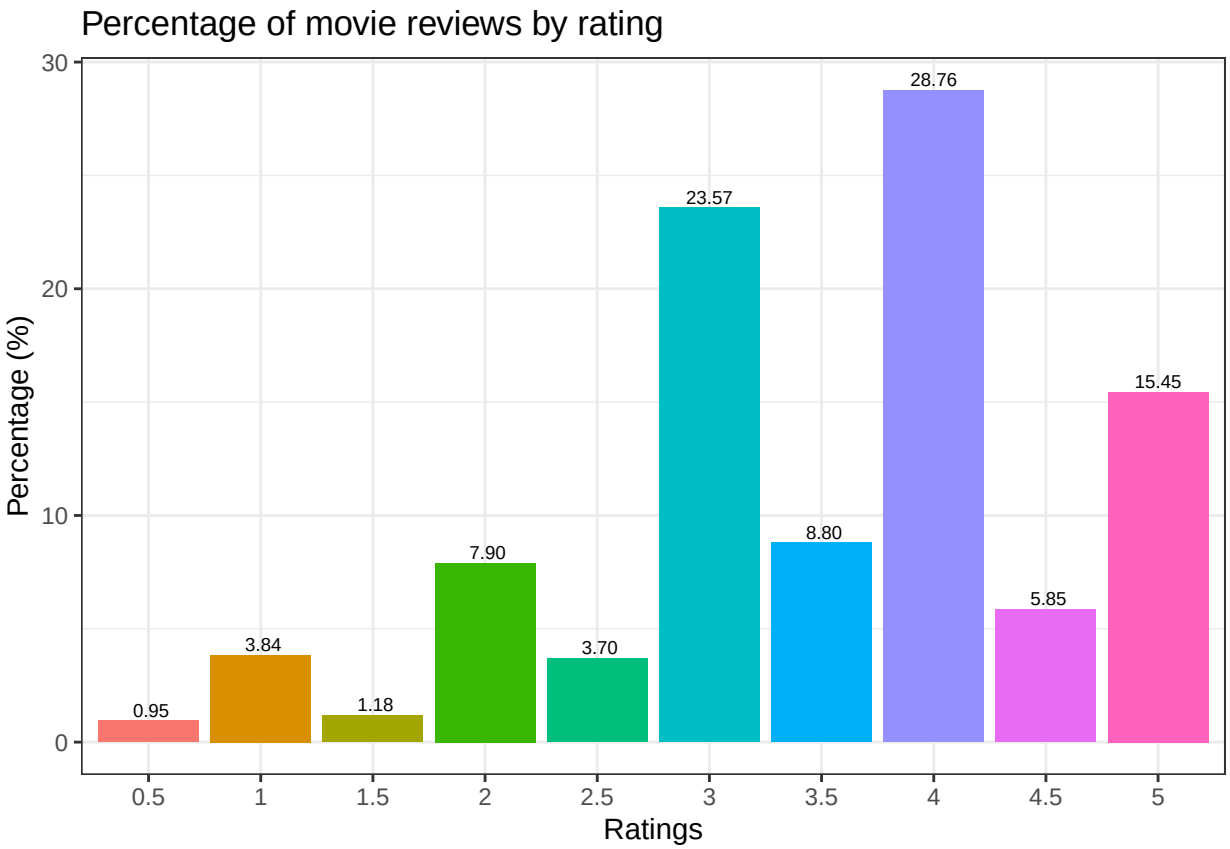
8. True or False: In general, half star ratings are less common than whole star ratings.

   It is True. There are 1,843,170 half star ratings and 7,156,885 whole star ratings.

   The answer to this last question might suggest a heuristic bias. People are naturally drawn towards whole numbers. Yet, I do not think this observation could result into a significant feature for the models.

Analysis

I start this section by exploring the target variable `rating`. I explore this variable first because others may require some manipulation, which has not been done yet. The following graph shows the percentage of reviews by rating. It confirms question 8 in that there are less half star ratings than whole star ratings. Moreover, it is also observed, as suggested by the answer in question 7, that higher ratings are more common than lower ratings. In particular, the most common rating is 4.



Given the insights obtained so far, i first proceed to do feature engineering using all the columns in the data, with the exception of `genre` (later, I will also make use of it). In the next sub-section I will discuss the models. But for now I will only say that the first model is a linear model using four regressors for rating, without using the data on genre. The second and third models do require to make use of the data in the `genre` column. To do so, i will create a data set of dummy variables for each of the genres (as columns).[1] But, more on these later. Finally, the third model is an variation from the first model, but adding a summarized version of the genres data.

Feature engineering

The features used in the first model are:

- `movie_mean` , is the difference between each movie´s mean rating and the overall average rating
- `user_mean` , is the difference between each user´s mean rating and the overall average rating

[1]Note: Due to computing restrictions, I have written the code in a way in which chunks are self-contained. In particular, code that generates new objects is evaluated only once. Since the objects are saved, they can be imported when needed, without using memory unnecessarily. Unfortunately, this slows down the code because importing and exporting is slow. However, this strategy saves time when publishing the report.

- `year_dif_mean` , is the mean difference in years between the year of the review and the year of the movie
- `year_mean` , is the difference between each movie´s mean rating at each (release) year and the overall average rating.
- `movie_user` , is simply the interaction (multiplication) between the `movie_mean` and `user_mean`.

The approach is to use the least number of features as possible, that could still maximize the predictability. The proposed features are justified as follows. The selection of the first and second features (`movie_mean` and `user_mean`) is straight forward because they aim to capture user and movie specific biases, respectively. The third feature (`year_dif_mean`) is meant to capture a generational bias towards movies. That is, tries to capture differences in the preferences of movies across time. The fourth feature (`year_mean`) aims to capture a year specific bias. These last could be relevant when the film industry as a whole under performs during a specific year. Finally, the last feature (`movie_user`) is meant to capture the interaction between users and movies specific biases. I cannot compute user-movie mean because i will not be able to match both user and movie in the validation data. However, I can match user and movie separately and then calculate their interaction, which is the next best thing.

The features that i created so far do not make use of the `genres` column. As indicated earlier, this column contains a list of pipe-separated of genre of each movie. Because a single movie can have several genres, the column cannot be used as a feature directly. Thus, i first proceed to unravel it into categorical columns, one for each genre. The total number of genres in the data are 20. Furthermore, I notice that, the training data has 20 number of genres. Because all the movies in the testing data are also contained in the training data, i know 20 is also the maximum number of genres needed in this analysis. In the second model I convert this data into dummy columns, one column for each genre. Notice that because the genres are not one-to-one the issue of multi-collinearity does not apply.

Finally, for the third and last model I also make use of the `genre` column, but because having that many columns for genres is not efficient I attempted to summarize the information. My approach was to obtain a "genre effect", as i did previously to construct other features. Then, because one movie may have more than one genre, I also took the average of the genre effects across all genres reported for each movie. These reduced the number of features considerably.

Further exploration

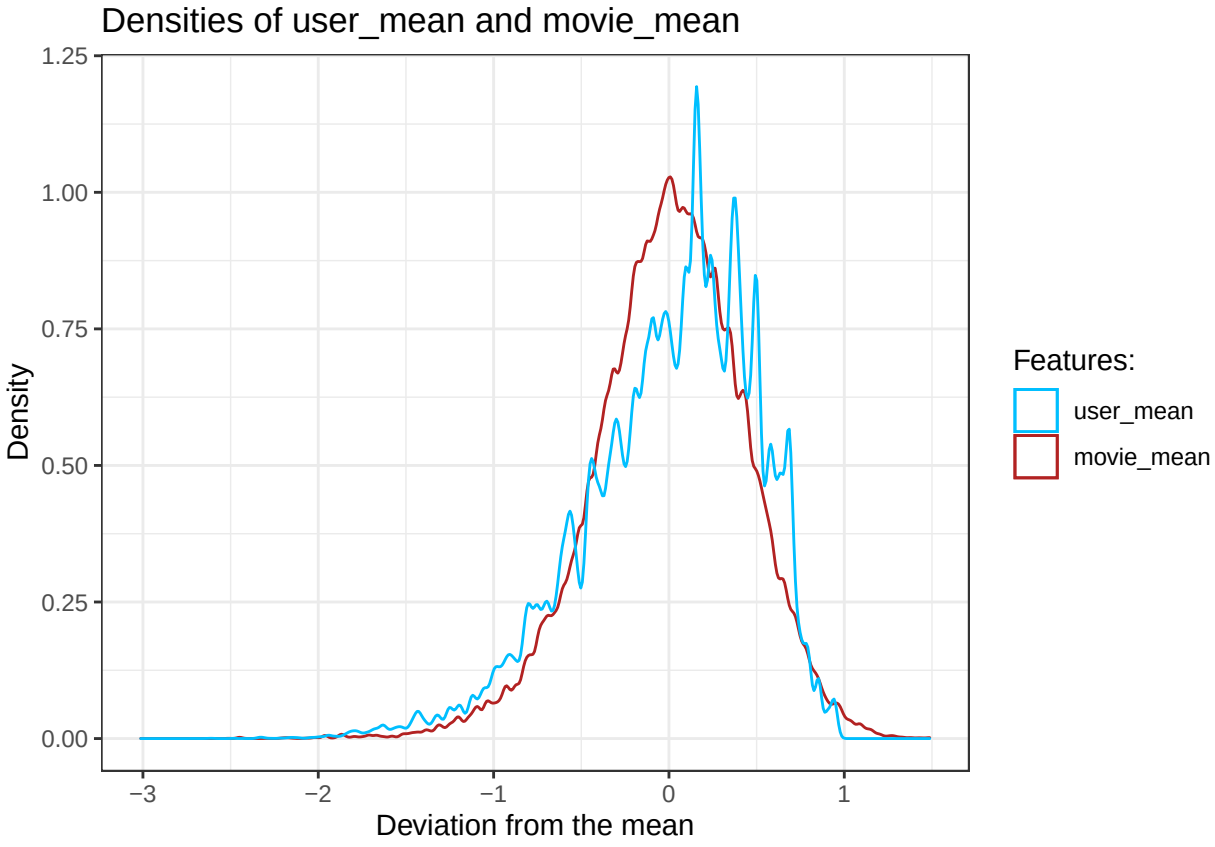
In the Introduction section I explored the original data to a certain extend. However, it is now worth it to also explore at least some of the features that I created afterwards. The following table shows the correlations between the `rating` and the five features first created.

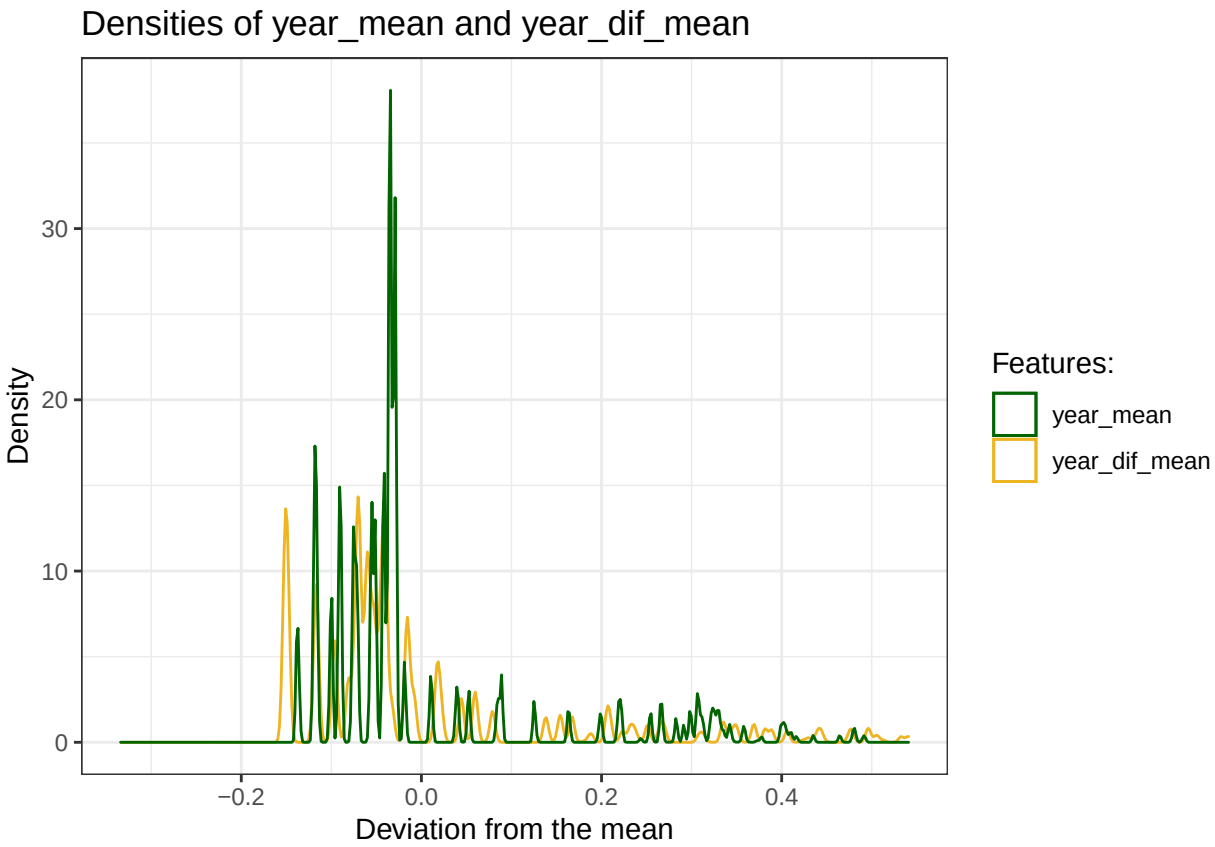
	rating	user_mean	movie_mean	year_dif_mean	year_mean	movie_user
rating	1.0000000	0.4038697	0.4584323	0.1280014	0.1435860	-0.0901612
user_mean	0.4038697	1.0000000	0.1538793	0.0435060	0.0506321	-0.0379854
movie_mean	0.4584323	0.1538793	1.0000000	0.2714144	0.3132105	-0.0964998
year_dif_mean	0.1280014	0.0435060	0.2714144	1.0000000	0.8690088	0.0117668
year_mean	0.1435860	0.0506321	0.3132105	0.8690088	1.0000000	0.0250318
movie_user	-0.0901612	-0.0379854	-0.0964998	0.0117668	0.0250318	1.0000000

As observed above, the features with the highest correlation with `rating` are `movie_mean` and `user_mean`, in that order. It is also observed that `year_dif_mean` and `year_mean` are highly correlated, which is not ideal. This last could be because most reviews are about recent movies, and thus the variable `year_dif_mean` may be unbalanced towards low differences between the movie and the review. I can explore these (and more) ahead. First, I address the features that are directly comparable to the mean of `rating`. Thus, in the following table of statistics (and following graphs) I exclude `movie_user`.

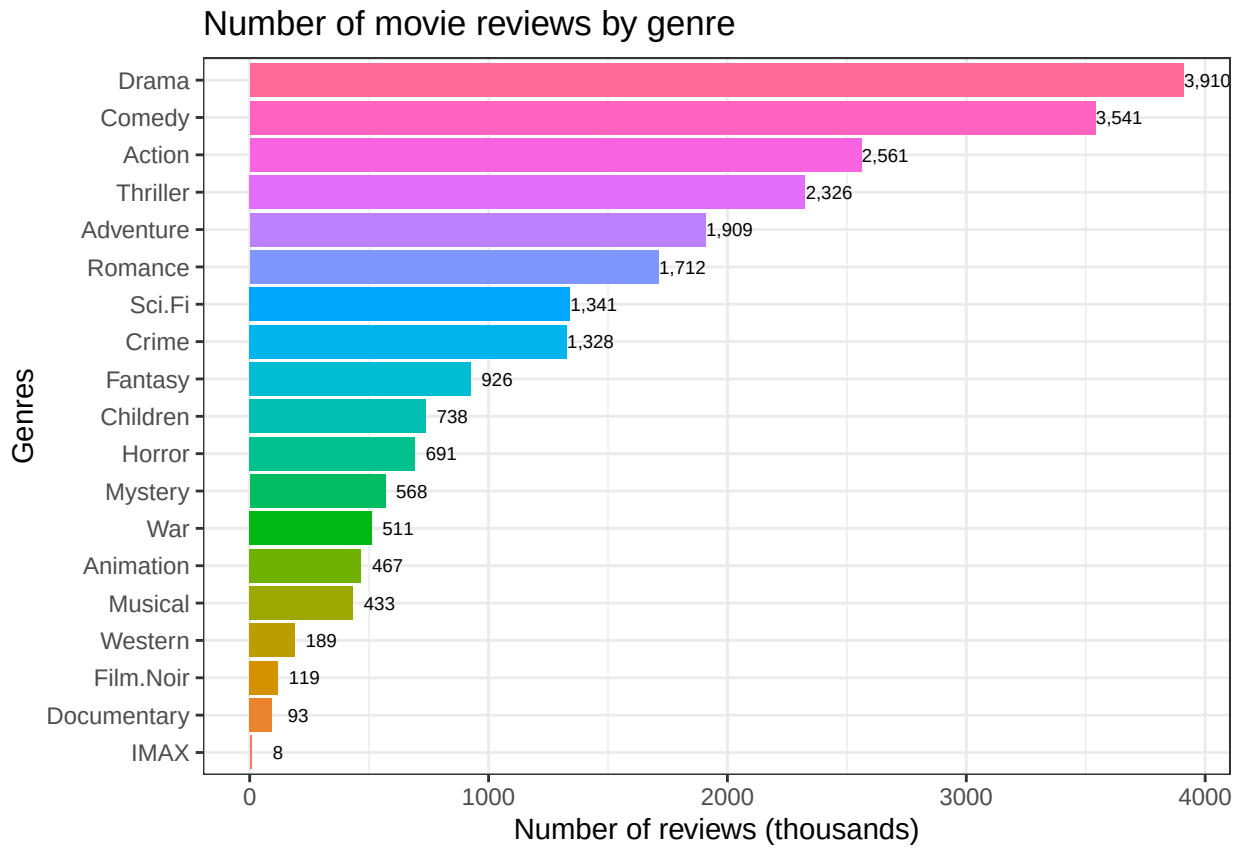
Statistic	rating	user_mean	movie_mean	year_dif_mean	year_mean
Min.	0.500	-3.013	-3.013	-0.334	-0.228
1st Qu.	3.000	-0.261	-0.295	-0.076	-0.081
Median	4.000	0.017	0.078	-0.035	-0.052
Mean	3.512	0.000	0.000	0.000	0.000
3rd Qu.	4.000	0.288	0.364	-0.029	0.018
Max.	5.000	1.488	1.488	0.491	0.541
Skewness	-0.596	-0.409	-0.755	1.782	1.729
Kurtosis	0.008	0.851	0.615	2.175	2.316
Std. Dev	1.060	0.428	0.486	0.136	0.152

Notice above that all features have mean zero. That is because they all are mean deviations. Then, notice that `user_mean` and `movie_mean` vary considerably more than `year_dif_mean` and `year_mean`. In fact, `year_dif_mean` and `year_mean` high coefficient of kurtosis indicate a high concentration around the mean. Finally, i also point out that the skewness coefficients indicate `user_mean` and `movie_mean` have a longer tail to the left of their mean, and conversely, `year_dif_mean` and `year_mean` have a longer tail to the right of their mean. Now, all these information is graphically represented bellow:





Notice in the two graphs above that the ranges in the axes are considerably different. In fact, the two pairs of features cannot be represented together because the latter pair limits the visualization of the former pair. So far, I explored the features I obtained from other columns, with the exception of the column `genres`. To explore further the information contained in the `genres` column I present the following graph.



As observed, the sum of all genres is higher than the original number of reviews. This again has to do with the fact that a single movie can have more than one genre. The most common genres are Drama, Comedy and Action, while IMAX and Documentary are the least common ones. One of my concerns so far is that the distribution of genres is quite unbalanced. Genres such as IMAX or Documentary are filled almost entirely by zeros, which is not efficient.

Models

- The first model I propose is a simple linear model using the 5 variables created initially, these are: `movie_mean`, `user_mean`, `year_dif_mean`, `year_mean` and `movie_user`. To have an idea of the performance of the model I first use the `caret` package for cross validation. The results indicate the RMSE of the simple linear model would be approximately 0.8710405.

- The second model I propose is also a linear model, but this time including 25 variables: the 5 from the previous model, as well as the 20 dummy variable columns from the genre data. However, due to insurmountable computing limitations, I will not estimate the RMSE using cross validation.[2]

- Finally, the third and last model in this project once more takes the features `movie_mean`, `user_mean`, `year_dif_mean`, `year_mean` and `movie_user`, but also adds the feature `genre_mean`, which is an average of the genre effect described earlier. Despite that the computation constrains are lower, the improvement is not as large as expected. The results from cross validation indicate the RMSE of this model would be approximately 0.8708461.

[2]I am aware that this project asks to use cross validation on all models. However, it simply was not an option for me. I was not able to find a computer with sufficient memory to run the models, and certainly not enough to do cross validation. I eventually was able to borrow a computer with sufficient memory to run a simple model and save the results. I hope this suffices and is taken into consideration for this project´s evaluation.

Results

Because the objective of this project is to predict the ratings from the validating data, i cannot obtain the features in the same way as i did with the training data. Doing so would imply using the result (rating column) before testing the model. Instead, i can use the other columns in the validation set together with the training set to create the features for the model. This certainly is not usual, but it is a way of ensuring the validity of the results.

To be clear, what I did is, for example, to match the variable `movie_mean` in the validation data and the training data to get the feature `movie_mean` into the validation data. This process can be repeated to `user_mean`, `year_dif_mean`, and `year_mean`. The last feature in the first model, `movie_user`, is obtained as before, by multiplying `movie_mean` and `user_mean`.

The features `user_mean`, `movie_mean`, `year_diff_mean`, `year_mean` and `movie_user` are sufficient for the first model. However, recall that the other two models require making use of the genres column. This time, the `genres` column in the validation data is sufficient and I can use the exact same functions to unravel the genres into dummy variables.

As for the final results, recall that the RMSE obtained from cross validation was around 0.871. So, I expect values close to that number. Indeed, the outcomes from the first and third models are RMSEs of 0.8781892 and 0.8781858, respectively, which is good, but could be better. The second model improved the RMSE, but only marginally to 0.8780921. Given that the second model required considerable additional time and computing resources, and provided little benefit, I consider the third model a better option, and thus my selected model.

	RMSE	Number of variables
Linear model 1	0.8781892	5
Linear model 2	0.8780921	25
Linear model 3	0.8781858	6

Conclusion

In this project I was given a large data set and was tasked to run a machine learning model to predict ratings of movies. After data exploration and manipulation, I came up with three models, one with 5, one with 6 and anoother with 25 features. The final hold-out test in this project resulted into a RMSE close to 0.88, which was not enough to reach the target mark of 0.86490. However, I believe that this project has solid insights and attempted to follow innovative ideas.

Regarding my final considerations, I believe that if I had at my disposal a more powerful computer I might have attempted feature selection methods or even attempted to reduce the dimensions of the data in the second model. Unfortunately, that was not feasible. I also believe that more sophisticated ML models such as Support vector Machine or Random Forest could considerably improve the outcome. Likewise, such alternatives were not feasible. Yet, I also conclude that this project allowed me to put into practice my skills in R and motivated me to continue learning. In particular the project got me interested about the topic of distributed data algorithms and machine learning on the cloud. I hope that in the near future, the specifications of my personal computer will not be a true limitation for my data science projects.