Movielens Recomendation System

HarvardX: PH125.9x Data Science: Capstone

Álvaro Amorós Rodríguez

16/4/2021

1.Introduction.

1.1 Summary.

The objective of this project is to put in to practice some of the techniques and skills learned during the HarvardX Data Science Professional Certificate courses, and apply them to generate a movie recommendation system. To do so, I will use the movielens dataset provided by the course, which contains 9 million observations provided by 69878 users on 10677 movies. The data has 6 dimensions, userId, movieId, rating, tiemstamp, title, and genders. My is goal is to use this dimensions to predict the future ratings that viewers will give to movies. The course also provides a 1 million observations validation dataset in which I will test the validity of my final model. In the first part of the project, I will conduct descriptive and visual analysis of the data, and I will prepare it to tests or subsequent predictive models. In the second part, I will explore de advantages and limitations of different models, and apply them on the *validation* dataset, to asses the overall validity of those. The third and final part will consist of a small summary of my conclusions.

1.2 Data.

There are 69878 users in the dataset, which rated 10677 movies, with and average rating of 3.51 and a standard deviation of 1.06, being the highest rating 5 and the lowest 0.5, each movie has and average number of 5429 ratings and each user has rated and average number of 340 movies.

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

Table 1: Summary statistics

Users	Movies	Ratings	Avg.	Sd	Max	Min	Avg. N. movie	Avg. N. user
69878	10661	9000055	3.512457	1.060362	5	0.5	6108	382

Table 2: Data table

userId	movieId	rating	timestamp	title	genres
1	122	5	838985046	Boomerang (1992)	Comedy|Romance
1	292	5	838983421	Outbreak (1995)	Action|Drama|Sci-Fi|Thriller
1	316	5	838983392	Stargate (1994)	Action|Adventure|Sci-Fi

userId	movieId	rating	timestamp	title	genres
1	329	5	838983392	Star Trek: Generations (1994)	Action|Adventure|Drama|Sci-Fi
1	355	5	838984474	Flintstones, The (1994)	Children|Comedy|Fantasy
1	356	5	838983653	Forrest Gump (1994)	Comedy|Drama|Romance|War

2. Analysis

##2.1 Preparation To test different models before applying them to the *validation* dataset, I will divide the data in two sets, *train set* and *test set*, with the first containing 90% of the observations and the second 10%.

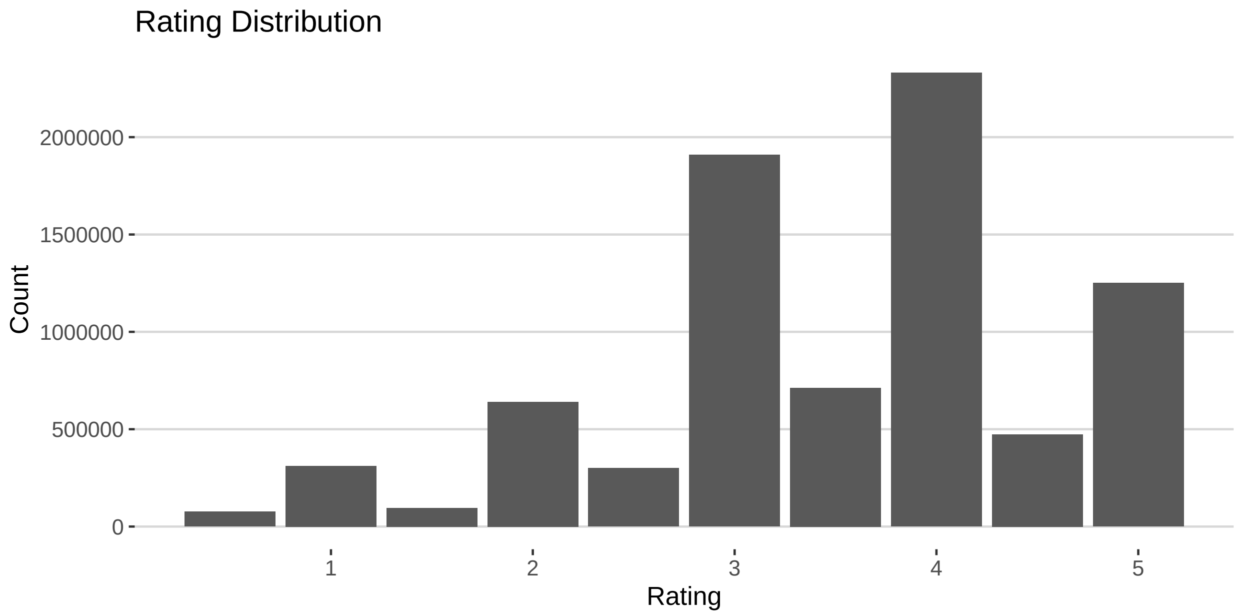
Besides the information a about the rating that each user gave to each movie, we have data on when this rating was given and on the genre or genres of the each movie. One can expect a certain time effect in the data, with users giving higher ratings in specific periods and lower in others. Genre should be also a valid predictor for rating as certain users will have preferences towards certain genders. But given the size of the dataset, at the limited memory and computing power of my laptop, this last two dimensions will not be used.

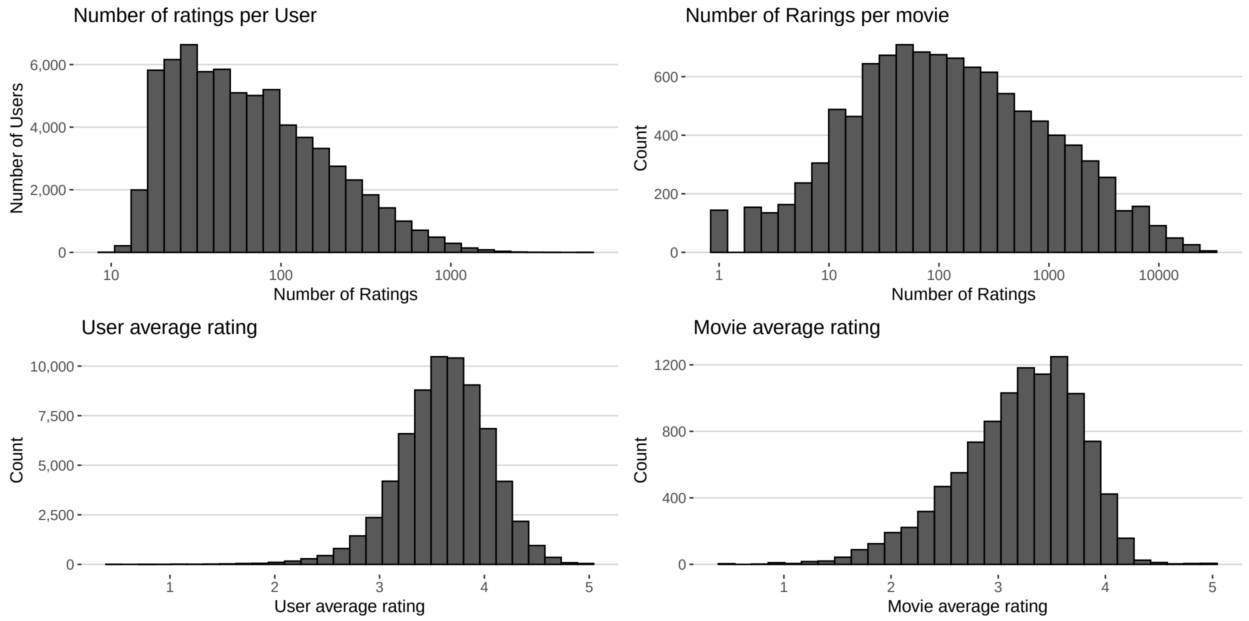
2.2 Visual analysis.

A preliminary visual inspection of the data can be useful to get a glance of the predictive capacity of different dimensions. From the five graphs generated we can extract a set of observations:

- Higher ratings and prevalent.
- The user average rating distributions is almost perfectly normal.
- The movie average rating is skewed to the right.
- The distribution of number of ratings per user is skewed to the left.
- The distribution of number of ratings per movie is close to normal.

Regarding the users,on one hand, we can see that most of them have rated a small number of movies, while a small group of them have rated a significantly high number. On the other hand, the distribution of the mean rating of users is almost perfectly normal, with most of the mean ratings concentrating around the mean. If we look at the number of ratings per movie, we can see that some block busters have very high number of ratings while another group of movies has almost none. The average movie rating is skewed to the right, with only a few movies having a rating higher than 4. This information about the data will be useful as we start testing or different models.





3 Results

3.1 Model testing

Before applying the final models to the *validation* dataset, I will test a set of models using the train and test datasets. I will start wit a set of simple linear models to conclude with a more complex collaborative filtering algorithm based on matrix factorization. To asses the validity of each model I will use the *Root-mean-square deviation*, which according to Wikipedia *“represents the square root of the second sample moment of the differences between predicted values and observed values or the quadratic mean of these differences.”*

RMSE = sqrt(mean((true_ratings - predicted_ratings)^2))

3.1.1 Linear Model

Naive model The simplest model possible in order to minimize mean squared error is to use the mean rating of our sample to predict ratings. With this model we obtain a RMSE of 1.06, which is the same as the standard deviation. I will create a table in which to store the results as well as the objective RMSE given by the exercise (0.864900) in order to be able to better compare the different models.

Table 3: Models

method	RMSE
Objective	0.864900
Mean	1.060054

Movie effect As shown by the graphical analysis conducted in the previous part, different movies have different mean ratings, as this average ratings follow a specific distribution, we can use this to further improve our model. The simplest way to incorporate this to the model is to calculate the standard deviation from the mean of each movie individually, and take into account this deviation when predicting ratings. With this model we improve the RMSE to 0.943.

Table 4: Models

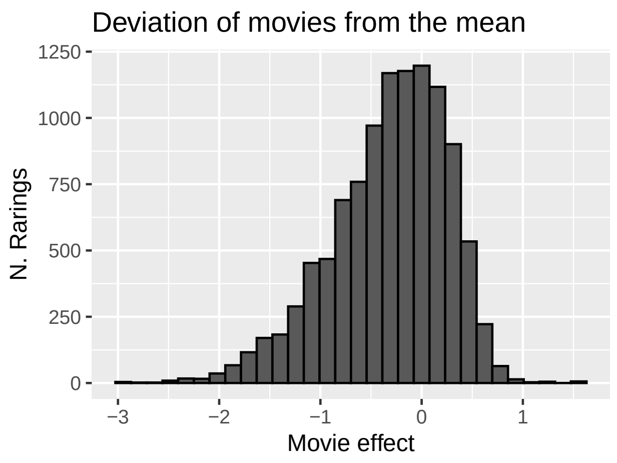
method	RMSE
Objective	0.8649000
Mean	1.0600537
Movie Effect	0.9429615

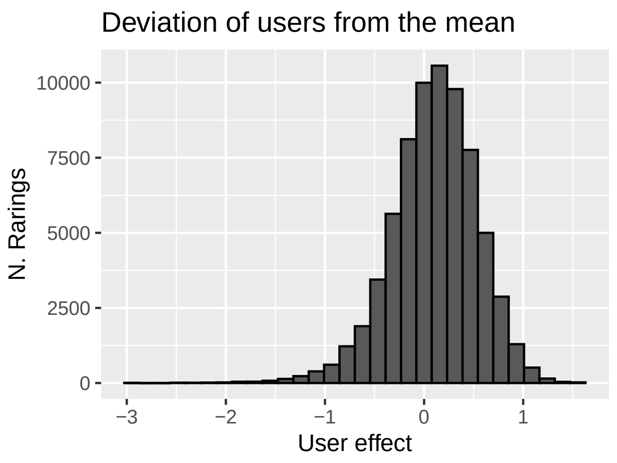
User effect. As it happens with the mean rating of movies, each specific user has its own mean rating, altogether, this average ratings form a normal distribution around the overall mean. To incorporate this to our model, we will incorporate the deviation of the average of each user to the overall mean (ones the movie effect has been already incorporated). By adding the user effect to the model, the RMSE improves to 0.864.

Table 5: Models

method	RMSE
Objective	0.8649000
Mean	1.0600537
Movie Effect	0.9429615
Movie and User Effects	0.8646844

Regilarizartion. The RMSE has reached the desired objective of 0.8649 by very little. We can analyze which are the predictions that have deviated more from the data, for that we can look at the margins of our users and movies distribution. As we can see in Tables 6 and 7, the movies and users that deviate more from the mean are those with a very low number of ratings, as a low number of observations increases the standard error. The top 10 movies that deviate more from our predictions have between 1 and 4 ratings, white the 10 users that deviate more have between 15 and 28 ratings. This numbers ar far away from the average number of ratings per movie and user, 6108 and 382 respectively. This movies and users correspond to the tails of our distribution. To minimize the effect those users and movies in our predictions, we can use a penalization term which will give those movies and users with a low number of ratings less weight in our model.





```
## Joining, by = "movieId"
```

Table 6: Movies with the highest deviation from the mean

Title	Deviation	N. Votes	Rating
Hellhounds on My Trail (1999)	1.487543	1	5
Satan's Tango (SÃ¡tÃ¡ntangÃ³) (1994)	1.487543	1	5
Shadows of Forgotten Ancestors (1964)	1.487543	1	5
Fighting Elegy (Kenka erejii) (1966)	1.487543	1	5
Sun Alley (Sonnenallee) (1999)	1.487543	1	5
Blue Light, The (Das Blaue Licht) (1932)	1.487543	1	5
Who's Singin' Over There? (a.k.a. Who Sings Over There) (Ko to tamo peva) (1980)	1.237543	4	5
Who's Singin' Over There? (a.k.a. Who Sings Over There) (Ko to tamo peva) (1980)	1.237543	4	4
Life of Oharu, The (Saikaku ichidai onna) (1952)	1.237543	2	5
Life of Oharu, The (Saikaku ichidai onna) (1952)	1.237543	2	4

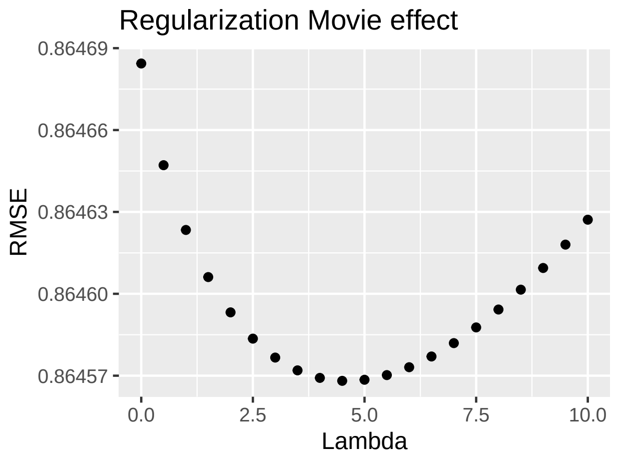
```
## Joining, by = "userId"
```

Table 7: users with the highest deviation from the mean

User	Deviation	N. Votes	Rating
13496	-3.418827	15	0.5
48146	-3.233854	21	0.5
49862	-3.163456	16	0.5
63381	-3.020395	16	0.5
62815	-2.928849	19	0.5
6322	-2.847928	16	0.5
6322	-2.847928	16	4.0
15515	-2.654739	28	0.5

User	Deviation	N. Votes	Rating
15515	-2.654739	28	5.0
15515	-2.654739	28	2.0

To find the optimal penalization term we will use an algorithm that will test different penalization between 0 and 10. The lowest RMSE is obtained with a penalization of 4.5 for the movie effect and 5 for user effect. The final linear model will include the movie effect, the user effect and the realizations Lambda outputs, this will yield a RMSE of 0.8641359, which meets the requirements of the exercise.



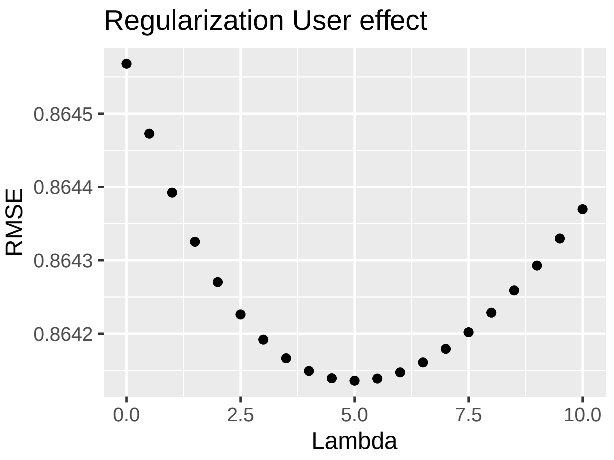


Table 8: Models

method	RMSE
Objective	0.8649000
Mean	1.0600537
Movie Effect	0.9429615
Movie and User Effects	0.8646844
Movie + User Effects + Regularization	0.8641359

3.1.2 Matrix factorization.

Matrix factorization is a class of collaborative filtering algorithm used to build recommendation systems. Recommendation systems based on collaborative filtering rely on the assumption that users with similar tastes will rate the same items in similar ways. Hence, the missing ratings of user A can be inferred from other similar users, which are refer to as a *“neighborhood”*. This rating patters of users are also present in between items. With matrix factorization one can extract this “hidden” structure of the data and use it to make predictions. To do so, in this project we will use the *recosystem* package.

Ones our data has been transformed to use with the recommended package, we can use the *tune* function to fit the best parameters: number of latent factors, gradient descent rate and penalty parameter to avoid overfiting. Ones the best parameters have been defined we can train the mode. With the collaborative filtering model we obtain a significantly imported RMSE that meets the objective of the exercise.

```
## iter      tr_rmse          obj
##    0       0.9824   1.1032e+07
##    1       0.8764   8.9984e+06
##    2       0.8430   8.3548e+06
##    3       0.8200   7.9526e+06
##    4       0.8039   7.6859e+06
##    5       0.7919   7.4982e+06
##    6       0.7817   7.3530e+06
##    7       0.7730   7.2371e+06
##    8       0.7654   7.1394e+06
##    9       0.7588   7.0592e+06
##   10       0.7531   6.9918e+06
##   11       0.7480   6.9354e+06
##   12       0.7435   6.8854e+06
##   13       0.7394   6.8419e+06
##   14       0.7357   6.8036e+06
##   15       0.7323   6.7701e+06
##   16       0.7293   6.7389e+06
##   17       0.7263   6.7095e+06
##   18       0.7237   6.6837e+06
##   19       0.7213   6.6618e+06
```

```
## [1] 0.7855709
```

Table 9: Models

method	RMSE
Objective	0.8649000
Mean	1.0600537
Movie Effect	0.9429615
Movie and User Effects	0.8646844
Movie + User Effects + Regularization	0.8641359
Matrix Factorization	0.7855709

3.2 Final model and results.

To asses the final validity of our models we will test them with the *edx* and *validation* datasets. For our linear model with regularization we obtain a RMSE of 0.8648177, which meets our required 0.8649, while for the collaborative filtering algorithm we obtain significant improvements with a RMSE fo 0.7824548

```
## iter      tr_rmse          obj
##    0       0.9724   1.2010e+07
##    1       0.8720   9.8823e+06
##    2       0.8381   9.1582e+06
##    3       0.8166   8.7475e+06
##    4       0.8018   8.4726e+06
##    5       0.7901   8.2813e+06
##    6       0.7805   8.1263e+06
##    7       0.7724   8.0105e+06
##    8       0.7655   7.9099e+06
##    9       0.7596   7.8326e+06
##   10       0.7544   7.7649e+06
##   11       0.7497   7.7054e+06
##   12       0.7456   7.6545e+06
##   13       0.7418   7.6104e+06
##   14       0.7384   7.5740e+06
##   15       0.7352   7.5365e+06
##   16       0.7322   7.5047e+06
##   17       0.7294   7.4781e+06
##   18       0.7269   7.4531e+06
##   19       0.7246   7.4293e+06
```

```
## [1] 0.7826846
```

Table 10: Models

method	RMSE
Objective	0.8649000
Movie + User + Regularization (edx-validation)	0.8648177
Matrix Factorization (edx-validation)	0.7826846

4. Conclusions.

The collaborative filtering model has proven to be by far the best model to predict ratings, and supposes a significant increase in the RMSE when compared to linear models, besides that, the model only needs a feedback matrix to be trained as it does not require any type contextual features which can be useful in situations where we lack labeled information. A last advantage of this approach is that it can be used to discover new interests of the users and hidden structures in the data (although this is not done in this project.)Regarding the limitations, the most important one is the incapacity of the model to rate new items, as if the item was not present in the train set, it will not be able to generate any prediction. The second limitation comes from the incapacity of the model to use secondary information besides it meId and rating, in the case of this dataset, other type of model could use gender or timestamp as predictors, but there is not straightforwards method to incorporate this information to a collaborative filtering model. Regarding the linear model, it yields a significantly weaker RMSE when compared to matrix factorization. Even so, a simple linear model which includes user and movie effects is enough to meet the exercise requirements. This model has the advantages of being straightforward to interpret and easy to implement. By using regularization techniques the model improves even further, but there is not a significant difference. Finally, the linear model could be improved by using the genre information and to some extend the timestamp information, but due to the memory limitations of this computer this could not be done. Both approaches yielded satisfactory results and met the required RMSE of the exercise being both valid methods to predict ratings. The linear model is computationally more efficient and is straightforward to understand an implement, but yields the worst RMSE, on the contrary, the collaborative filtering model is computationally demanding, and difficult to interpret, but with it we obtain the best RMSE.