MovieLens Recommendation System Report

Mohammad S. Sammoudi

19 July, 2021

Introduction

This project aims to build a movie recommendation system using the machine learning. We will use a 10M versino of MovieLens dataset included in dslabs package. Movie recommendation system is used to predict the movies suitable for some user based on ratings that we have from other users. Firstly we will look at the structure of the data, make a visualization plots and then build a model step by step to reach the required accuracy.

MovieLens dataset:

The data was collected and provided by GroupLens. A research lab at the University of Minnesota that specializes in recommender systems, online communities, mobile and ubiquitoustechnologies, digital libraries and local geographic information systems. They have collected millions of movie reviews and offer these data sets in a range of sizes. For this project the 10M version will be used. It contains 10 million ratings on 10,000 movies by 72,000 users. It was released in 2009.

You can find the latest MovieLens dataset in the follwing URL: https://grouplens.org/datasets/movielens/latest/

The MovieLens 10M Dataset is split into two dataset: edx and validation. The edx data set represents 90% and used for Developing the algorithm and model construction. The validation data set represents 10% and used only for assessing the performance of the final model.

Methodology and exploratory data analysis

In this section we will explain the process and techniques used, including data cleaning, data exploration and visualization, insights gained, and the modeling approach that will be used.

The first step in data analysis is to explore the structure of the data. The MovieLens version dataset that will be used is called "edx" and it contains 9000055 obs. of 6 variables. Each observation represents a rating given by one user for one movie.Columns include userId, movieId, rating, timestamp, title and genres. Timestamps represent seconds since midnight UTC January 1, 1970.

There are 6 variables in the edx dataset which are the follwing:

1. userId: an integer variable which represent a unique id for the user.
2. movieId: a numeric value which is unique id for the movie.
3. rating: numeric, rating between 0 and 5, where 0 is bad rating and 5 is the best rating.
4. timestamp: integer, represent the time of rating.
5. title: character, represent the name of the movie.
6. genres: character, represents the movie category(comedy, romance, drama,...etc).

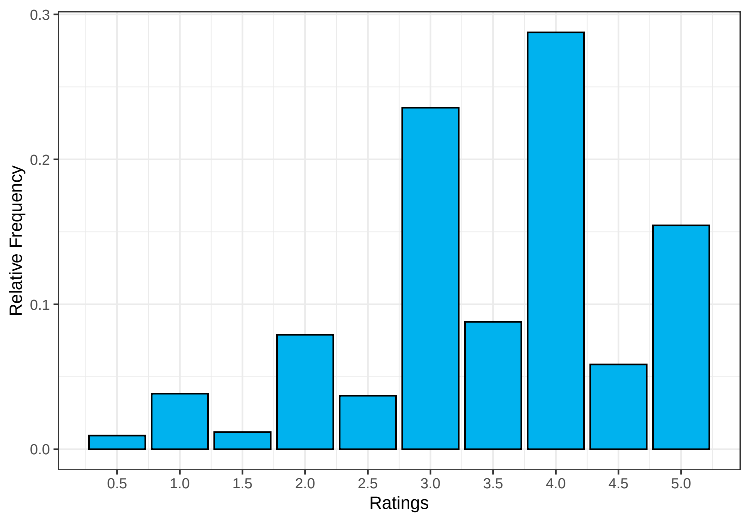
Table blow shows the structure of the data

```
## Classes 'data.table' and 'data.frame':   9000055 obs. of  6 variables:
##  $ userId   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ movieId  : num  122 185 292 316 329 355 356 362 364 370 ...
##  $ rating   : num  5 5 5 5 5 5 5 5 5 5 ...
##  $ timestamp: int  838985046 838983525 838983421 838983392 838983392 838984474 838983
##  $ title    : chr  "Bird of Prey (1996)" "Bad Moon (1996)" "Arsenic and Old Lace (194
##  $ genres   : chr  "Action" "Action|Adventure|Horror" "Comedy|Mystery|Thriller" "Dram
##  - attr(*, ".internal.selfref")=<externalptr>
```

There are 69,878 unique users and 10,677 movies as shown below.

```
##   n_users n_movies
## 1   69878    10677
```

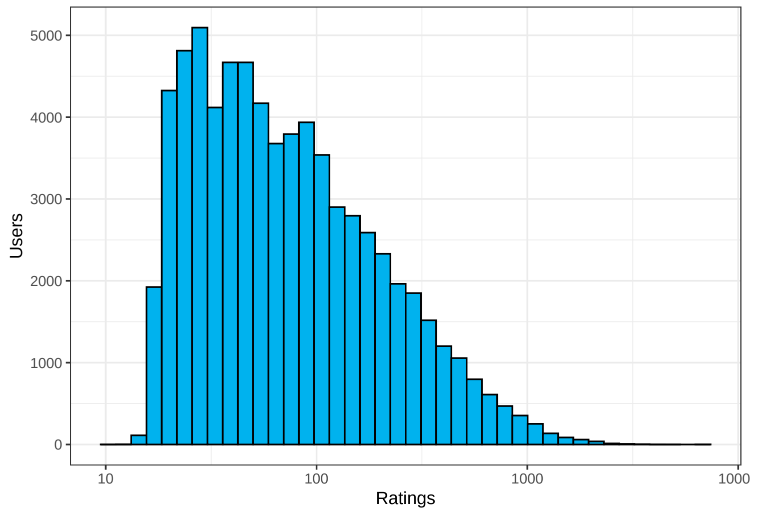
The distribution of the movie ratings shows a range of 0.5 to 5 with whole numbers used more often as shown in the figure below.



From the previous figure, we notice that:

- The overall average rating in the edx dataset was 3.51
- The top 3 ratings from most to least are : 4, 3, 5.
- Users desire to rate movies more positively than negatively.
- The histogram shows that the half-star ratings are less common than whole star ratings.

The number of ratings VS users shows a right skew in its distribution as shwin the below figure.



From above, we can conclude:

- 30 % of users contribute 70 % of ratings in whole edx dataset.
- Some users rated very few movie and their opinion may bias the prediction results.
- The average user rating tends to increase when the number of ratings increases.

We can also explore the timestamp variable. Time is recorded as the UNIX timestamp, the UNIX timestamp is merely a number of seconds between a particular date and the Unix Epoch. This count starts at the Unix Epoch on January 1st, 1970 at UTC.

The following figure represent the average ratings of movies by week:

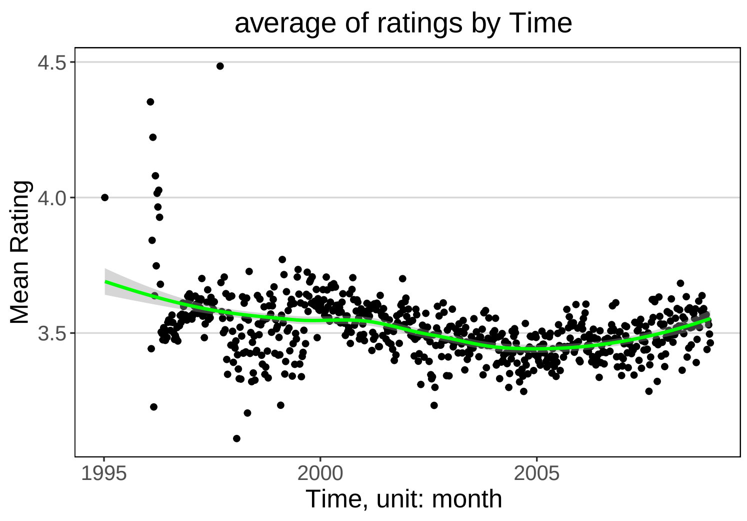


Figure 1: *Average ratings by time/month in Edx Dataset*

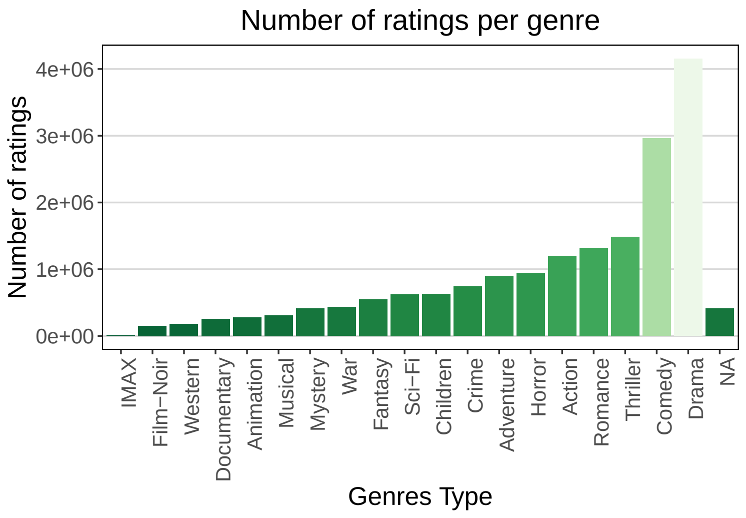
From the previous figure, we conclude that:

There are some evidences about time effect on ratings average, but this effect is not a strong.

We also have genres classification for the movies, and we need to explore this variable as it may be effective in reducing RMSE.

A movie could be classified to one or more genres; there are 20 levels of genre.

we can see the average of ratings per genres as shown in the following figure:



From above, we can conclude:

- The number of ratings varies per genre.
- The ratings average for genres are Converging, although the number of ratings varies.
- The genres only slightly affect movie ratings.

Evaluation Approach:

Several models will be assessed starting with the simplest. Accuracy will be evaluated using the residual mean squared error (RMSE) as requested:

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

N is defined as the number of user/movie combinations, yu,i as the rating for movie i by user u with the prediction as y^u,i. The RMSE is a commonly used loss function that simply measures the differences between predicted and observed values. It can be interpreted similarly to a standard deviation. For this project if the number is larger than 1 it means our typical error is larger than one star. The goal is to make RMSE as minimum as we can.

Accuracies will be compared across all models using the code below.

```r
RMSE <- function(true_ratings, predicted_ratings){
  sqrt(mean((true_ratings - predicted_ratings)^2))
}
```

Project Methodology:

We will follow the following steps to analyze the data and reach our goal of a minumum accuracy:-

- Firstly we need to download data and explore its observations and variables, then we'll make some visualizations to better understanding the data and this will help us later in choosing the appropriate model.

- Then We'll start building models with the ideas gaind from the first step using machine learning algorithms.

- Concurently with the first step, we will check and evaluage the effectiveness of each model by using RMSE, and this will be done by splitting the edx dataset to edx_train and edx_test datasets and the performance will be assessed using the edx_test.

- We will then apply regularization in our final model which will add a penalize term on our model.

- Finally, retrain the best performance model and assess (evaluate) using the Validation test.

Building Models

We will start building different models and after each model built, we will check the RMSE value, st at the end we will have our final model with the lowest RMSE.

Model 1: Same Rating for all movies and users(just the average)

The first model, is the very basic model wich will assume the same rating for all movies and users, with all differences explained by random variatino. If μ represents the true rating for all movies and users and ϵ represents independent errors sampled from the same distribution centered at zero, then:

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

Model 2: Movie effects

This model accounts for rating of the movies, as we know some movies have rating greater than others, so this model will give better accuracy of the prediction.

so here we will define a new bias effect called b_i which represents the average rating of movie i.

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

Model 3: User effect model

We know that users are different between others in rating, some users are more active in rating movies than others, and this of course affects the prediction.

We will add a new abis term called user effect b_u which represents the user-specific effect.

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

Model 4: Movie and user effects.

In previous models, we take in our calculations the movie effect and the user effect separately, and now we can imporove our model more by taking both of them together and see the resutls.

We will add both abis terms (movie bias b_i and user effect b_u), so our model will as the follwing:

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

Movie, user and time effects.

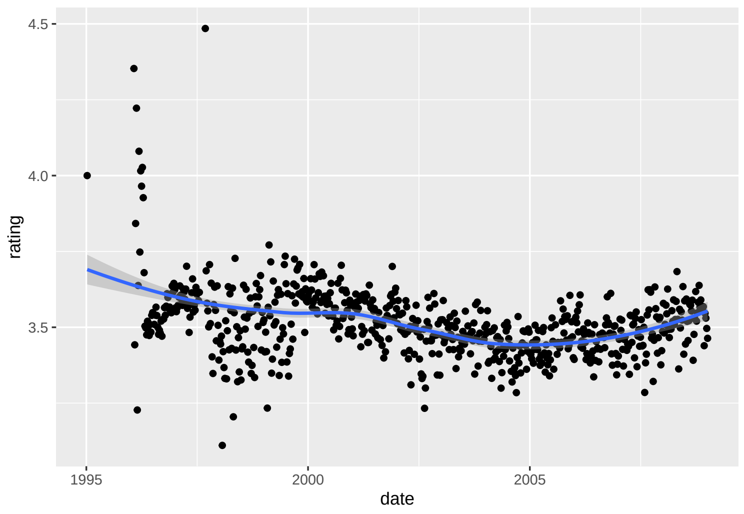
In this model, we will add a third bias term, which will the time effect.

The time and date of rating sometimes affect the rating vlaue for the movies, So it worth to check this bias effect.

The edx dataset has variable called timestamp. this variable represents the time and data in which the rating was provided. The units are seconds since January 1, 1970. We can create a new column in edx dataset called date with the follwing command:

```r
edx <- mutate(edx, date = as_datetime(timestamp))
```

In this model, we will compute the average rating for each week and plot this average againt date, and the output is shown in the below figure.



From the above plot, we notice that there is some evidence of a time effect on average rating, but in fact it not as that strong so we'll ignore it.

Note for information only: we take the time effect in our account, then the model will be like the follwing:

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

with f a smooth function of $d_{u,i}$

Model 5: Regularization

Regularization constrains the total variability of the effect sizes by penalizing large estimates that come from small sample sizes.

To estimate the b's, we will now minimize this equation, which contains a penalty term. We use regularization to estimate movie and user effects.

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

The larger λ is, the more we shrink.λ is a tuning parameter, so we can use cross-validation to choose it. We should be using full cross-validation on just the training set, without using the test set until the final assessment.

Results:

We'll now start testing models one by one using R code. and the follwing steps summarize what we will do step by step:

1. Split the edx data into two datasets: edx_train with 80% of the data and edx_test with 20% of data.
2. build the models mentioned above and check the RMSE in each model to test preformance using the edx_test dataset.
3. apply regularization to maximize performance and minimize RMSE.
4. Identifying the best mode.
5. Rerun the optimal model using the edx dataset as trainig dataset and validation dataset as a test dataset.

create train and test set from edx dataset

This is the first step. We partition the edx dataset into two datasets: the first one is the edx_train set which will contain 80% of the data and used to train the models, the other set is the edx_test which will be used to evaluate each mode.

define the RMSE (Risidual Mean Sqaure Error)

```r
RMSE <- function(true_ratings, predicted_ratings){
  sqrt(mean((true_ratings - predicted_ratings)^2))
}
```

Model 1: base model (just the average)

This model ignores all features and calculate the average rating. We will consider this model as baseline and try to imporve it as much as we can to mimize the RMSE.

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

The average rating is the follwing:

```
## [1] 3.512417
```

And the RMSE for this model is shown in the next table:-

method	RMSE
Model 1:Just the average	1.060426

Model 2: Movie effect model

Here we will add the bias of movie effect.

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

And the RMSE is shown in the next table:-

method	RMSE
Model 1:Just the average	1.0604262
Model 2:Movie Effect	0.9437355

Model 3: User effect model

We will now see the results of user effect bias on the performance of system and how this bias can lower the value of RMSE.

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

The next table shows the RMSE for this model:-

method	RMSE
Model 1:Just the average	1.0604262
Model 2:Movie Effect	0.9437355
Model 3:User Effect	0.9545522

Model 4: Movie + User effects model

Now, we will add the user bias effect to our model in addition to movie effect from model 2(both of them).

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

The next table shows the RMSE for this model:-

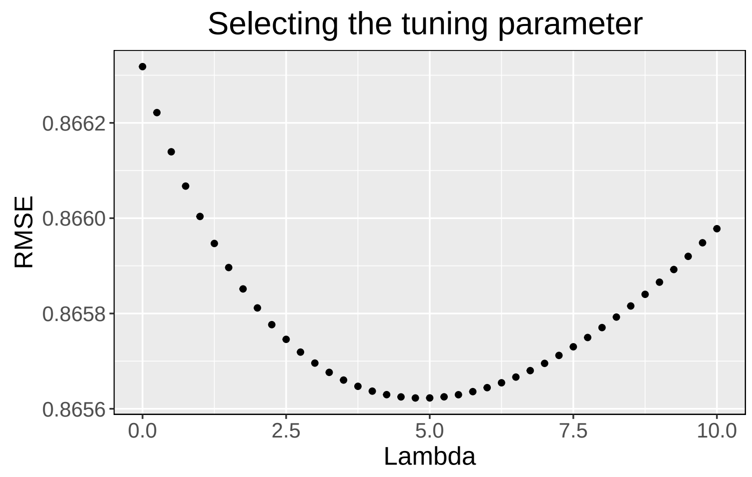
method	RMSE
Model 1:Just the average	1.0604262
Model 2:Movie Effect	0.9437355
Model 3:User Effect	0.9545522
Model 4:Movie + User Effects	0.8663181

Model 5: Regularization of movie and user effects

Regularization technique should be used to take into account on the movie and user effects, by adding a larger penalty to estimates from smaller samples. so we will use parameter λ.

Note: λ is a tuning parameter, that we need to choose an optimal value of it to return a minimum RMSE value.

We will use k fold cross validation to find the optimal λ.



Selecting the tuning parameter in Edx_train da

After we choose the optimal λ from the above k-fold cross validation method, we can use it in the model to get the RMSE results:

method	RMSE	Model
Model 1:Just the average	1.0604262	NA
Model 2:Movie Effect	0.9437355	NA
Model 3:User Effect	0.9545522	NA
Model 4:Movie + User Effects	0.8663181	NA
NA	0.8656000	Regularized Movie,User

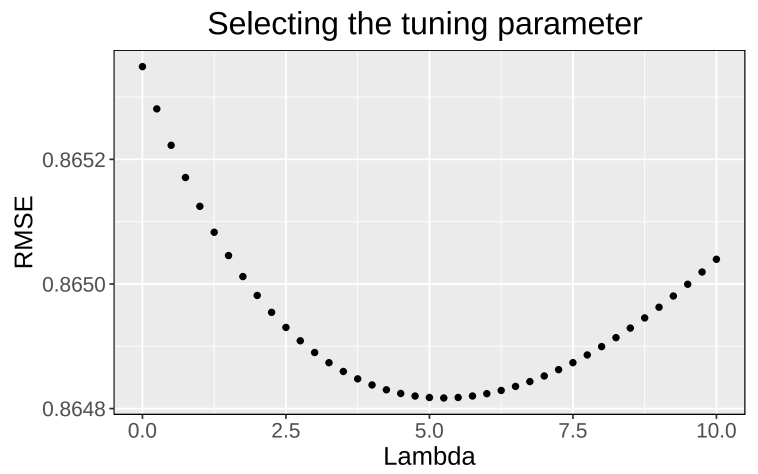
Final Model

In the previous section, we build and construct different models using edx_train and edx_test which are subsets from the edx dataset.

We found that the best model in performance is model number 5 which is regularized movie and user effects model. We found the RMSE to be 0.8656200.

Now we need to build the final model with entire edx set as training dataset and evaluate this model by validation set (which we don't use in any model).

Now, let's apply our final model which is the regularized movie and user effect model with edx dataset as training dataset and validation dataset as testing dataset. Firstly, we need to fine λ that minimizes the RMSE as in the follwing plot:



Selecting the tuning parameter in Edx_train da

after finding the optimal λ. Let us apply it to our model. The results are shown below:

method	RMSE	Model
Model 1:Just the average	1.0604262	NA
Model 2:Movie Effect	0.9437355	NA
Model 3:User Effect	0.9545522	NA
Model 4:Movie + User Effects	0.8663181	NA
NA	0.8656000	Regularized Movie,User
NA	0.8648000	Final Model

Conclusion

The objective of this project is to develop a recommendation system using the MovieLens 10M dataset that predicted ratings with a residual mean square error of less than 0.86490.

This report discusses a few methods used to construct recommendation systems. The best performing model is regularized movie and user effects, which yields an RMSE of 0.8648000 when trained on edx and tested on validation.

Finally, our final model can be improved more if we explore more techniques like matrix factorization, which may give better results.