# Harvardx Data Science: Capstone - Project MovieLens

Mathieu Labelle

February 2020

---

# 1 INTRODUCTION

You will find here a document that is part of my submission for the MovieLens Project (*"the Project"*) of HarvardX PH125.09x: DATA SCIENCE: Capstone.
This *Project* will show how to apply some of the knowledge base and skills learned throughout the "Data Science" Series.

## 1.1 Summarize goal

The aim of this *Project* is to create a recommendations system ("*the System*", "*the Algorithm*" or *the Model*) using a small subset of MovieLens dataset (*"the* Dataset*"*), this subset is given as input for the *Project*. This *Dataset* shows ten million entries of movie rating. The *Dataset* is split in two, the first set (*"Edx set"*) will be used to train a machine learning *Algorithm* to predict movies ratings, and we will apply the *Algorithm* to the second *Dataset* (*"Validation set"*) as if ratings were unknown, to evaluate how close our predictions are from the true values, Root Mean Square Error (*"RMSE"*) will be used, with maximum points given for:

$$RMSE < 0.86490$$

## 1.2 Key Steps

To create our *Algorithm*, we will use the methodology we saw in previous course: "Machine Learning", you can find the [link] (https://rafalab.github.io/dsbook/large-datasets.html) and specifically under section 33.7. This methodology use the average rating and adds bias (or effects), it also use Regularization in order to constrains the total variability of the effects sizes by penalizing large bias that come from small sample sizes.

- Find here the keys step of the *Project*:

a. Create a *Train set* and a *Validation set*, with the code provided
b. Explore the *Dataset*, clean and wrangle Data to prepare our analysis
c. Create a Train and Test set for Cross validation
d. Train our *Algorithm* to tuned or parameters using **RMSE** on test set
e. Apply our *Algorithm* to the *Validation set*, and compute our final **RMSE**

## 1.3 Explore *Dataset*

We will briefly explore here the *Dataset* to latter do some cleaning and wrangling. Starting with the summary of the *Dataset*:

Table 1: Summary of the Datasets

| userId | movieId | rating | timestamp | title | genres |
|---|---|---|---|---|---|
| Min. : 1 | Min. : 1 | Min. :0.500 | Min. :7.897e+08 | Length:10000054 | Length:10000054 |
| 1st Qu.:18123 | 1st Qu.: 648 | 1st Qu.:3.000 | 1st Qu.:9.468e+08 | Class :character | Class :character |
| Median :35740 | Median : 1834 | Median :4.000 | Median :1.035e+09 | Mode :character | Mode :character |
| Mean :35870 | Mean : 4120 | Mean :3.512 | Mean :1.033e+09 | NA | NA |
| 3rd Qu.:53608 | 3rd Qu.: 3624 | 3rd Qu.:4.000 | 3rd Qu.:1.127e+09 | NA | NA |
| Max. :71567 | Max. :65133 | Max. :5.000 | Max. :1.231e+09 | NA | NA |

Then looking at the first 10 entries and headers:

Table 2: Head of Edx

| | userId | movieId | rating | timestamp | title | genres |
|---|---|---|---|---|---|---|
| 1 | 1 | 122 | 5 | 838985046 | Boomerang (1992) | Comedy\|Romance |
| 2 | 1 | 185 | 5 | 838983525 | Net, The (1995) | Action\|Crime\|Thriller |
| 4 | 1 | 292 | 5 | 838983421 | Outbreak (1995) | Action\|Drama\|Sci-Fi\|Thriller |
| 5 | 1 | 316 | 5 | 838983392 | Stargate (1994) | Action\|Adventure\|Sci-Fi |
| 6 | 1 | 329 | 5 | 838983392 | Star Trek: Generations (1994) | Action\|Adventure\|Drama\|Sci-Fi |
| 7 | 1 | 355 | 5 | 838984474 | Flintstones, The (1994) | Children\|Comedy\|Fantasy |

The headers here are relatively clear and do not need further explanation. We can work directly on userId, movieId, rating, but we need to extract release date from the title and timestamp need to be convert in Date format. We will need also to create 2 new variables from timestamp, to aggregate year and aggregate month of rating.

We can also look at the rating distribution:

Table 3: Given Ratings

| rating | count |
|---|---|
| 4.0 | 2588430 |
| 3.0 | 2121240 |
| 5.0 | 1390114 |
| 3.5 | 791624 |
| 2.0 | 711422 |
| 4.5 | 526736 |
| 1.0 | 345679 |
| 2.5 | 333010 |
| 1.5 | 106426 |
| 0.5 | 85374 |

Distribution of rating looks a bit erratic as users tempt to give rounded rating.

# 2 ANALYSIS

## 2.1 *Dataset* cleaning and wrangling

As per our conclusion on the brief exploration we did, we will clean and wrangle the *Dataset*. So we will modify our *edx set* and *validation set* into tidy/clean ones (named edx_Tidy and validation_Tidy). Here is how the first ten entries look like:

Table 4: Head of Edx_Tidy

| userId | movieId | rating | time_stamp | RY_TS | released | genres |
|---|---|---|---|---|---|---|
| 1 | 122 | 5 | 1996-08-02 | 1997-01-01 | 1992 | Comedy\|Romance |
| 1 | 185 | 5 | 1996-08-02 | 1997-01-01 | 1995 | Action\|Crime\|Thriller |
| 1 | 292 | 5 | 1996-08-02 | 1997-01-01 | 1995 | Action\|Drama\|Sci-Fi\|Thriller |
| 1 | 316 | 5 | 1996-08-02 | 1997-01-01 | 1994 | Action\|Adventure\|Sci-Fi |
| 1 | 329 | 5 | 1996-08-02 | 1997-01-01 | 1994 | Action\|Adventure\|Drama\|Sci-Fi |
| 1 | 355 | 5 | 1996-08-02 | 1997-01-01 | 1994 | Children\|Comedy\|Fantasy |

## 2.2 Insights with Data visualization

### 2.2.1 Summary

Let's look at some statistics on the effects we want to study:

Table 5: Statistics

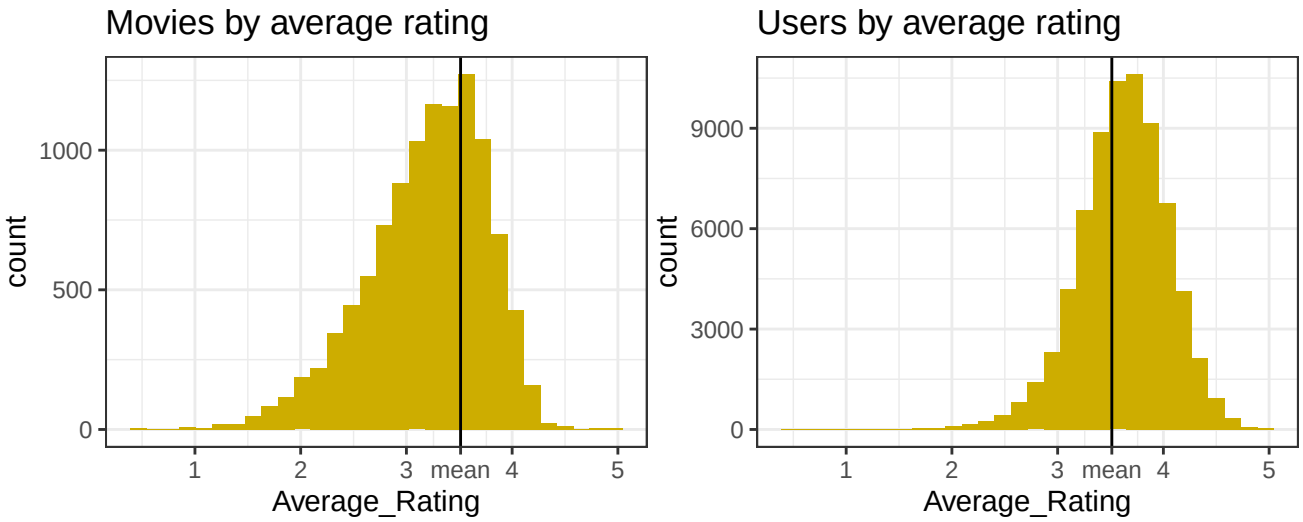
| | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. |
|---|---|---|---|---|---|---|
| rating | 0.5 | 3 | 4 | 3.512465 | 4 | 5 |

```
## [1] "Standard Deviation for the rating is: 1.06033139986467"
```

```
## [1] "Mean of ratings is: 3.51246520160155"
```

Minimum rating is 0.5 and maximum 5, our predictions should not be below or above.

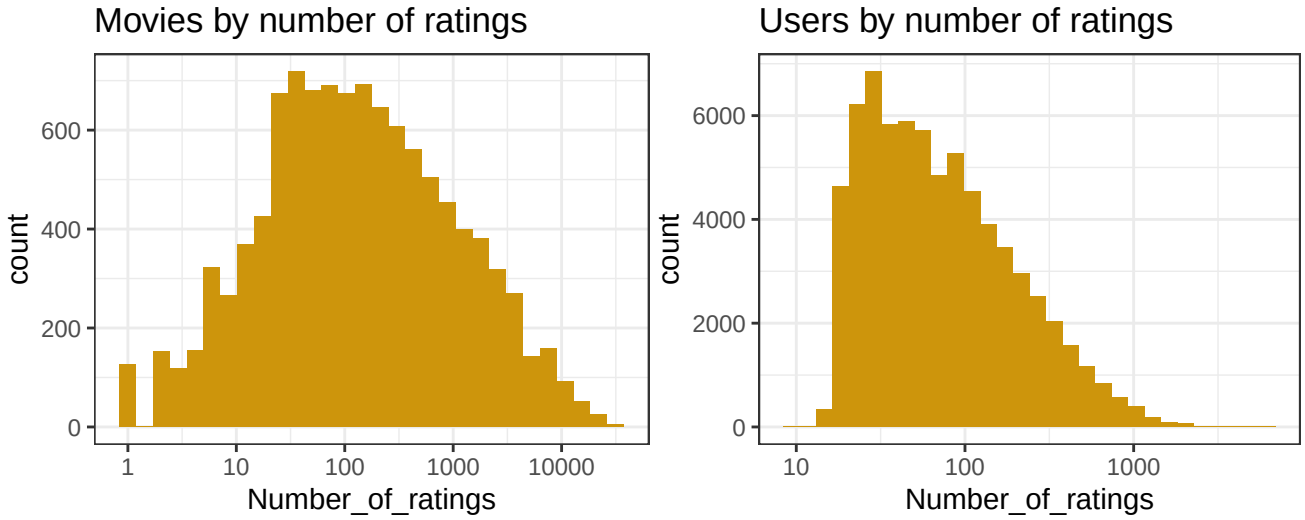
### 2.2.2 Entries by Average rating




Movies and users distribution by Average Rating look "Normal". We can see that there is movie and user

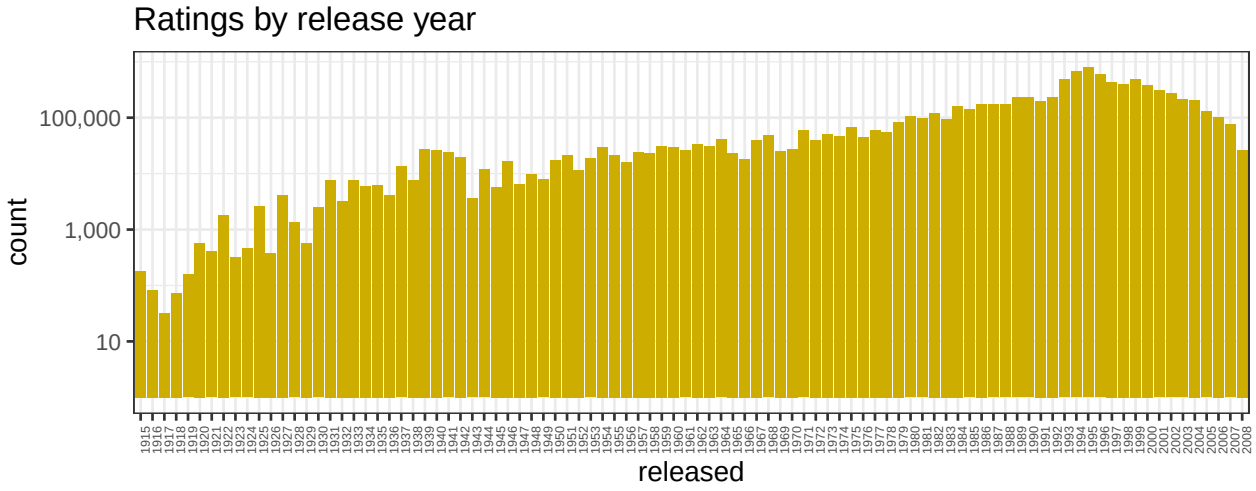
effects/bias, as "good" movies tempt to be rated higher than others, as well as some users are more easygoing than others.

### 2.2.3 Movies and users by number of ratings

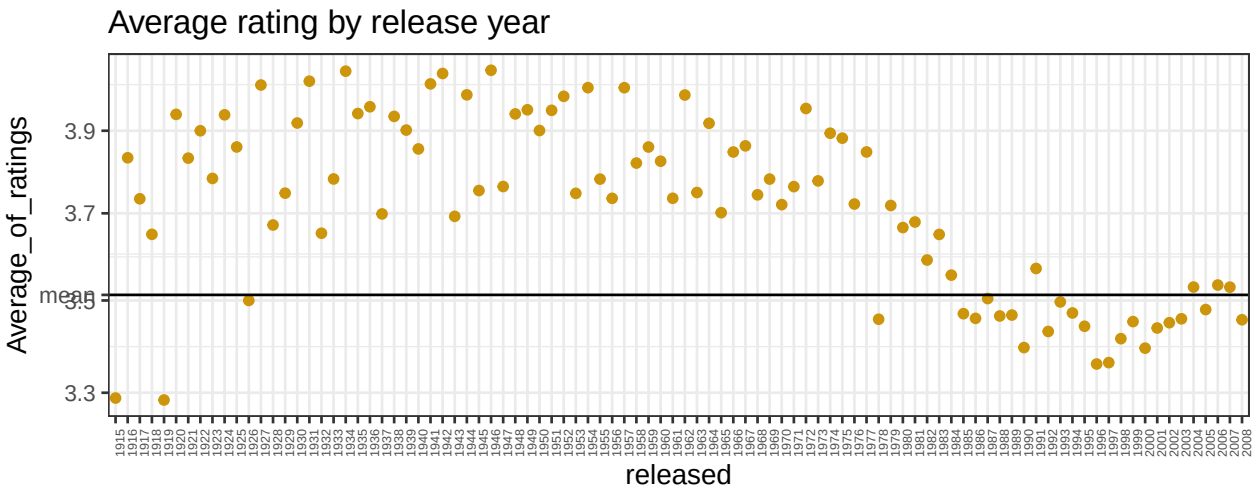



There are movies that are rated rarely, and users who rate very few movies.

### 2.2.4 Numbers and average rating by rating year

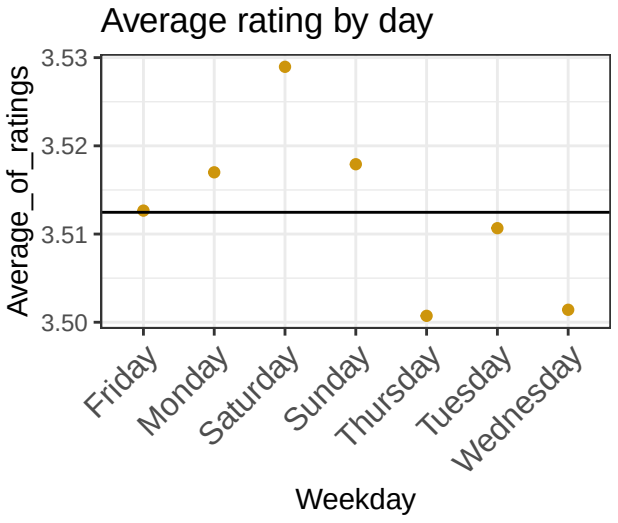


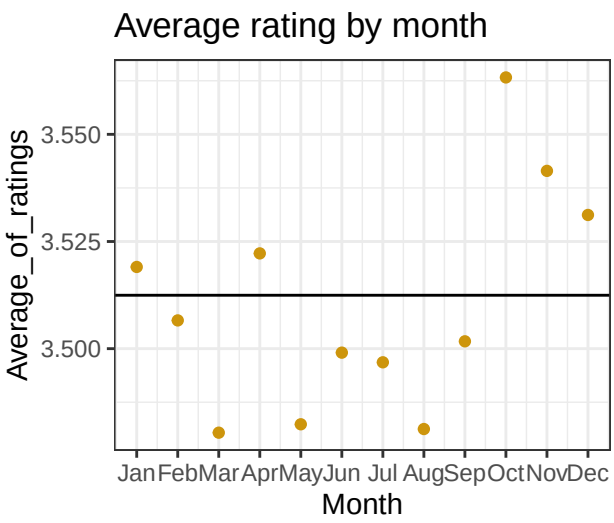

Old movies (before 1930) are rated less than 1,000 times, and older one even less.




There is a release year effect, with older movies less rated and rated higher.

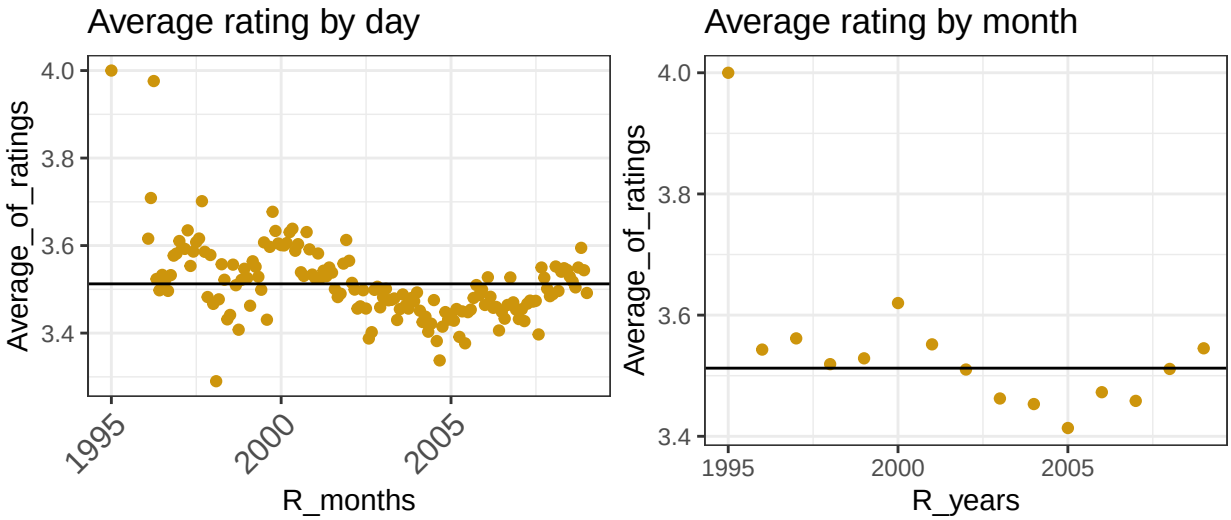
### 2.2.5 Seosonality of rating date




If you look at the scale the seasonality effect is very limited and therefore will not be taken into account to built our model.

#### 2.2.6 Average rating by month and year of rating




If we look at the pattern of both effects (month and year), it seems that month effect is included in year effect.

#### 2.2.7 Bias of genre

We can also look at the top 10 and bottom 10 "genres" bias (mean of rating by genre minus $\mu$) for genres that have more than 10K ratings.

| genres | Average_Rating | genres | Average_Rating |
|---|---|---|---|
| Crime\|Mystery\|Thriller | 0.69 | Adventure\|Children\|Drama | -0.72 |
| Action\|Adventure\|Comedy\|Fantasy\|Romance | 0.68 | Action\|Crime\|Sci-Fi | -0.69 |
| Adventure\|Drama\|Film-Noir\|Sci-Fi\|Thriller | 0.64 | Comedy\|Horror | -0.67 |
| Adventure\|Drama\|War | 0.57 | Children\|Comedy\|Fantasy | -0.64 |
| Crime\|Horror\|Thriller | 0.57 | Comedy\|Thriller | -0.64 |
| Comedy\|Crime\|Drama\|Thriller | 0.55 | Horror | -0.63 |
| Crime\|Film-Noir\|Mystery\|Thriller | 0.55 | Action\|Horror\|Sci-Fi | -0.61 |
| Adventure\|Mystery\|Thriller | 0.52 | Horror\|Sci-Fi | -0.60 |
| Comedy\|Drama\|Romance\|War | 0.50 | Children\|Comedy | -0.59 |
| Comedy\|Crime\|Drama | 0.49 | Action | -0.58 |
| Crime\|Drama\|Sci-Fi\|Thriller | 0.49 | Sci-Fi | -0.58 |

Genres have an effect on the rating.

### 2.3 Process

A supervised machine learning *Algorithm*, needs to be constructed step by step and evaluate at each step. Starting with a simple model, adding features and assess the performance, to improve our *Algorithm*. Here for our *System* we will use **RMSE** to evaluate our predictions against the actual values.

#### 2.3.1 Root Mean Square Error

**RMSE** (Root Mean Square Error) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; **RMSE** is a measure of how spread out these residuals are.

$$RMSE = \sqrt{\frac{1}{N}\sum(r - \hat{r})^2} \tag{1}$$

Where $r$ is the actual rating and $\hat{r}$ is our predicted rating

#### 2.3.2 Cross-Validation

We will also use cross-validation to built our *System*, as we can't use the *Validation set* (Nor the Tidy one) during our training, we will need to create two sets within edx_Tidy, one to train our models, another one to evaluate its predictions with **RMSE**, respectively edx_Tidy_train and edx_Tidy_test. For the partition of our *Edx set* will set the seed to 1, in order to constantly get the same partition. The partition will be 80% and 20% for train set and test set respectively.

### 2.4 Modelling approach

#### 2.4.1 Statistical learning approach

Let's suppose that we observe a quantitative response $Y$ and $p$ different predictors $X_1, X_2, X_3, ....X_p$. We will assume that there is some relationship between $Y$ and $X = (X_1, X_2, X_3, ....X_p)$, which can be written as:

$$Y = f(X) + \varepsilon. \tag{2}$$

With $\varepsilon$ random *error term* with a mean equals to 0.

#### 2.4.2 Following previous HarvardX courses approach

We are familiar with Movielens *Dataset* and with a recommendation system based on it. We already worked on it in HarvardX PH125.09x: DATA SCIENCE: Machine Learning. we will use a similar process/approach as [link] (https://rafalab.github.io/dsbook/large-datasets.html) under section 33.7 to 33.9.

#### 2.4.3 Parameters taken into account to built our model

- The insights we gained from the data exploration and visualization are the following:

a. Ratings are distributed around 4, but as users tempt to gave rounded ratings the distribution looks erratic.
b. We have detect 5 bias: movie, user, release date, rating year and genres
c. Ratings are always between 0 to 5, therefore all bias effect added to the average can't be out of theses bounds
d. As they are variability due to small sample size for bias, we will need to use regularization method

#### 2.4.4 Model progression steps

- We will progress with a supervised machine learning *Algorithm*:

a. We will first consider a simple model, where predicted ratings are just the average rating
b. We will then use a model with movie bias
c. We will compare with a model with two bias: movie and user
d. We will regularized the bias of previous model
e. We will use the year movies were released, compute its bias and add it to the previous model
f. We will add the genres regularized bias and add it to the previous modela
g. Finally we will aggregate timestamp by year and add it to the previous model

## 2.5 Developing and training our model.

### 2.5.1 Simple Average model

So for our first *model*, we will consider $f(X)$ as the mean of ratings (of our train set), we can write it as:

$$Y = \mu + \varepsilon \tag{3}$$

Practically we will just use $\mu$ as our predictions in the test set, and compute the **RMSE** (see (1)) Here are the result of our first simple model:

| Model | RMSE |
|---|---|
| Simple model using average | 1.0599 |

### 2.5.2 Model with movie bias

Now we will introduce a bias or effect. We will use movie bias ($b_i$), that is simply the difference between $\mu$ and the average rating by movie. we can write:

$$b_i = \mu - \frac{1}{N_i} \sum (rating_i) \tag{4}$$

With $N_i$ the number of entries for $movie_i$ and $rating_i$ each rating entry for $movie_i$.
We can write:

$$Y_i = \mu + b_i + \varepsilon_i \tag{5}$$

Here is the result:

| Model | RMSE |
|---|---|
| Simple model using average | 1.05990 |
| Model with movie bias | 0.94374 |

It is already better than our simple model, showing that our intuition about effects/bias are right

### 2.5.3 Model with two bias

In this model, we just add two bias, movie ($b_i$) and user ($b_u$) to the average. we can write is as:

$$Y_{i,u} = \mu + b_i + b_u + \varepsilon_{i,u} \tag{6}$$

Here is the result:

| Model | RMSE |
|---|---|
| Simple model using average | 1.05990 |
| Model with movie bias | 0.94374 |
| Model with Movie & user bias | 0.86572 |

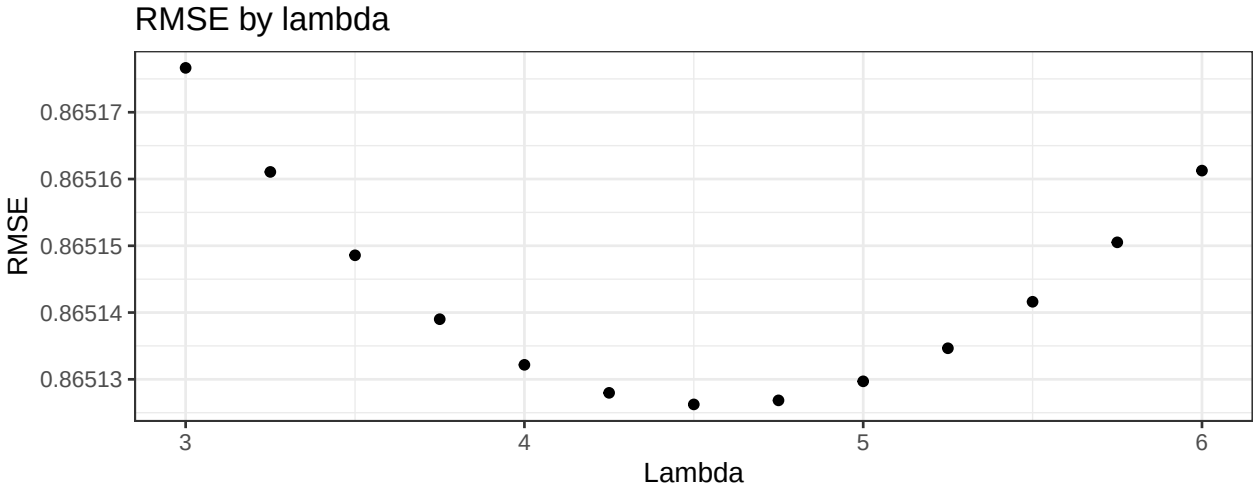
Using two bias improve dramatically our **RMSE**

### 2.5.4 Model with two bias regularized by a single parameter lambda

Using same model as previously (see (6)). We will now add the regularization of $b_i$ and $b_u$. As an example we can write the regularization of $b_i$ as:

$$\hat{b_i}(\lambda) = \frac{1}{n_i + \lambda} \sum_{1}^{n_i} (Y_i - \hat{\mu}) \tag{7}$$

The regularization have a parameter $\lambda$, the greater it is compare to number of sample $n_i$, the smallest the bias is. We will train our model to find $\lambda$ that minimize **RMSE**.
Here are the results:

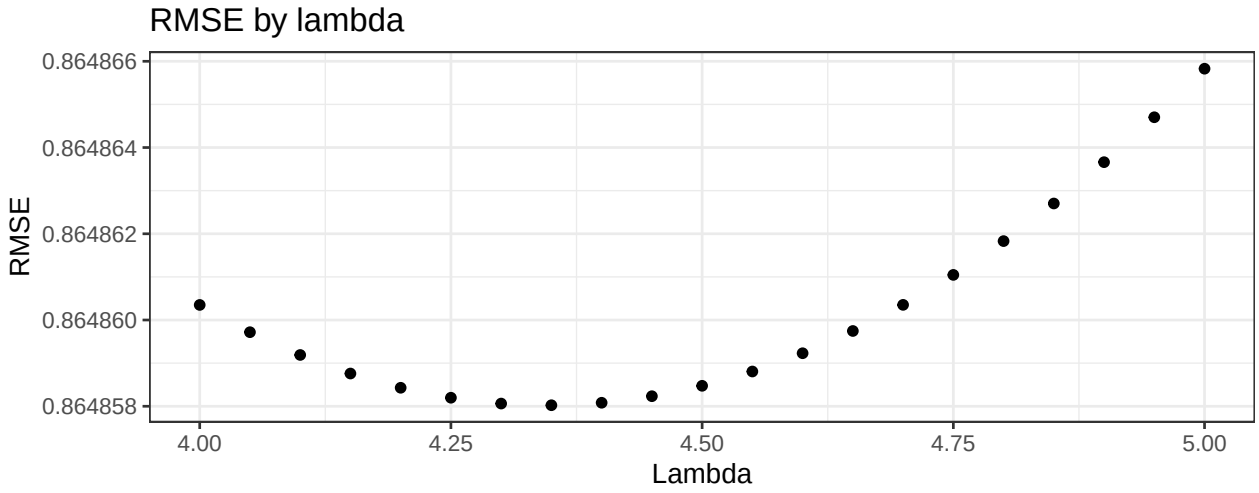



| Model | RMSE |
|---|---|
| Simple model using average | 1.05990 |
| Model with movie bias | 0.94374 |
| Model with Movie & user bias | 0.86572 |
| Model with 2 bias regularized | 0.86513 |

We are getting closer to our target in term of **RMSE**

### 2.5.5 Model with 3 bias regularized

On top of $b_i$, $b_u$, we will add the release year date bias: $b_y$.
Here are the result of this model:

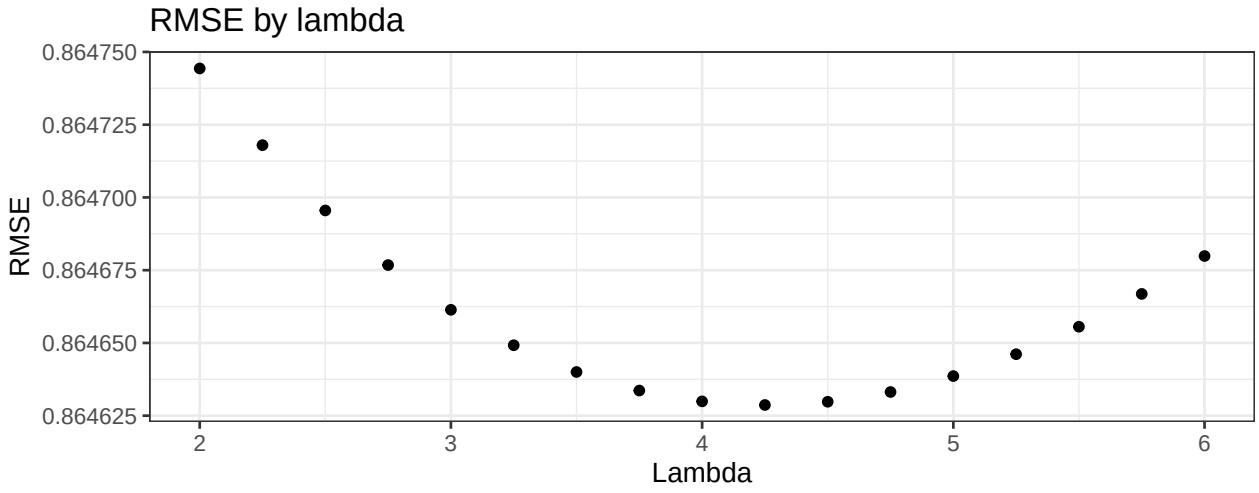



| Model | RMSE |
|---|---|
| Simple model using average | 1.05990 |
| Model with movie bias | 0.94374 |
| Model with Movie & user bias | 0.86572 |
| Model with 2 bias regularized | 0.86513 |
| Model with 3 bias regularized | 0.86486 |

That's it we are below 0.8649.

#### 2.5.6 Model with 4 bias

We will add the genre regularized bias $b_g$ to the previous model.

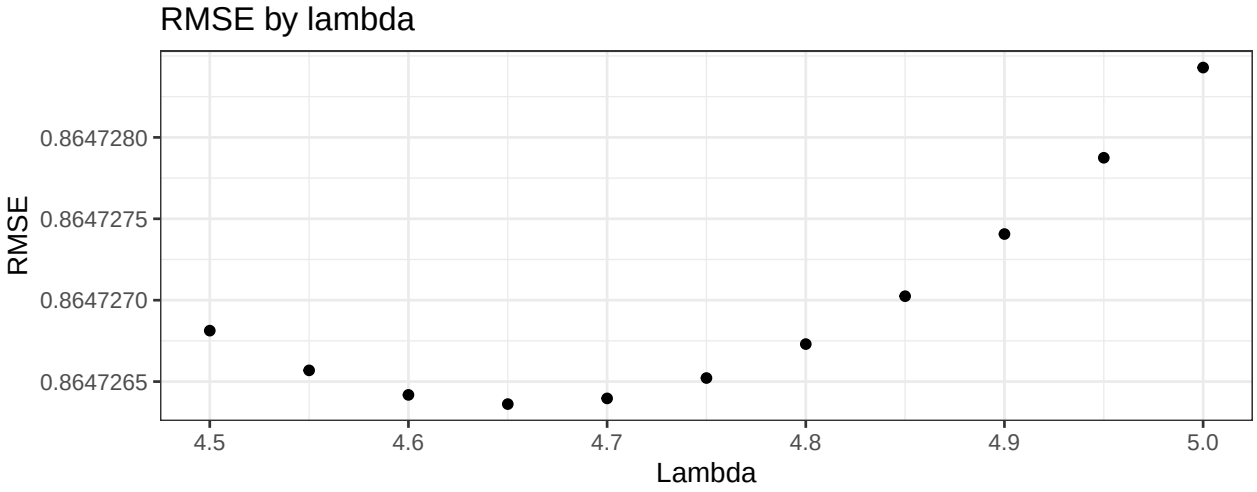



| Model | RMSE |
|---|---|
| Simple model using average | 1.05990 |
| Model with movie bias | 0.94374 |
| Model with Movie & user bias | 0.86572 |
| Model with 2 bias regularized | 0.86513 |
| Model with 3 bias regularized | 0.86486 |
| Model with 4 bias regularized | 0.86463 |

It improves our prediction.

### 2.5.7 Model with 5 bias regularized by a single parameter lambda

To finish, We will aggregate timestamp by year, and add it as bias $b_{ry}$.




| Model | RMSE |
|---|---|
| Simple model using average | 1.05990 |
| Model with movie bias | 0.94374 |
| Model with Movie & user bias | 0.86572 |
| Model with 2 bias regularized | 0.86513 |
| Model with 3 bias regularized | 0.86486 |
| Model with 4 bias regularized | 0.86463 |
| Model with 5 bias regularized | 0.86473 |

We have no improvement by adding the rating year, so we will drop this bias.
Anyway using rating date for a recommendation system, is a bit tricky except for it seasonality (i.e people to gave better rating on Saturday or in winter's month...), bias we saw have no real effect.
A recommendation system should not give rating as if you had rated the film in the past, recommendation is a good rating you are likely to gave if you see the film now, so rating date likely to be in the future.

## 2.6 Test our *Algorithm* with the validation_Tidy set for the final assessment.

Now that our *Algorithm* is trained, will need to pass our final hurdle: apply it on edx_Tidy set and evaluate it against the validation_Tidy set. Please find here the result:

```
## [1] "Using the mean of Edx_Tidy, equals to: 3.51246520160155"
```

| Model | RMSE |
|---|---|
| Simple model using average | 1.05990 |
| Model with movie bias | 0.94374 |
| Model with Movie & user bias | 0.86572 |
| Model with 2 bias regularized | 0.86513 |
| Model with 3 bias regularized | 0.86486 |
| Model with 4 bias regularized | 0.86463 |
| Model with 5 bias regularized | 0.86473 |
| Final assessment with Validation set | 0.86414 |

We are at 0.8641 below the targeted **RMSE** of 0.8649.

### 2.7 Discussing performance

The *algorithm* we use is relatively simple, and only compute average and reorganize data. So it is easy to run on a home personal computer, and it doesn't take ages to run.
The RMSE we found at 0.8641 is 18% better than the simple model using average as predictions.
The model use average rating and add bias that are intuitive, even regularization is intuitive, so this *model* is also easy to understand for someone with no prior statistical skills. Our model have a problem with small sample, in fact by using regularization we do minimise the weight of small sample, it is good for our RMSE but we are losing informations, we are not able to predict new film or predict rating by new user, for this we are blind.

## 3 CONCLUSION

We had follow instructions given in this course and process/model/algorithm saw in previous courses. We trained a machine learning algorithm, that produce a better level than the targeted RMSE.
But still the winner (The ensemble) in July 2009 of the Netflix challenges obtain 0.8558 (dataset was with 100 million entries). And today you can use package or code that give an even lower RMSE. So it shows us, that the road is still long, but at least we are engage on it.
One of the tool that can improve our RMSE would have been Matrix factorization , it is related to factor analysis, singular value decomposition (SVD), and principal component analysis (PCA) (see section 33.11 link). With this model, we can spot similar rating pattern within users or/and film.

Now if we get more predictors like for users: sex, age, location, education. . . , RMSE could be dramatically improved.

As my first project in data science, it is a small one, but I learn a lot from it. I use all learnings obtain during the previous eight courses of Havardx "Data Science", honestly it was great fun. And despite its modesty, I am relatively proud of my first long R code, my first use of R Markdown and also my first publish in LaTeX.