Movie recommendation system

Ltaief Mohamed

5/14/2021

Contents

1 Introduction

Streaming industry is one of the most important modern entertainment services. As a matter of fact, it is a growing business reaching a value of 50,11 billion USD in 2020. Numerous streaming services are drawing the attention of a film consumers providing a myriad of options. However, the abundance of material can make a movie choice not an easy task. Therefore a recommendation system can be of value.

2 Project objective

We will be creating a movie recommendation system using a 10 million version of the Movielens dataset. Our goal is to predict the values of the ratings which range on a scale from 0 to 5.

2.0.1 RMSE:

To describe the behavior of our rating outcome, our approach is to define the loss function. A function that quantifies the deviation of the observed outcome from the prediction (residual). In our case we used the root square error(RMSE) known as the standard deviation of the residuals. It has the same units as the measured and calculated data. Smaller values indicate a better performance of our recommendation system.

$$RMSE = \sqrt{}(\sum_{i=1}^{n}(X\ observation, i\ \ - X\ model, i\ )^2)$$

3 Exploratory data analysis

3.1 Features and processing

3.1.1 The dataset

We are given a dataset that contains a set of movie ratings from the `Movielens` ***website***, a movie recommendation service. This was made available by the `Grouplens`, a research group at the University of Minnesota.

Working with a big dataset is a challenging task. To make the computation easier we made two subsets; one for implementing the algorithm and one for testing its effectiveness. We called them respectively *edx* and *validation*.

The edx dataset assigns unique identification number to 69878 movies (movieId) and a unique identification number to 10677 unique users (userId). In total we have 9000055 ratings distributed between train set and test set with ratios of 90% to 10%. The validation dataset has 999999 ratings used to measure the RMSE of our model. Let's take a look at the first lines of our edx table:

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

Each row represents a rating of one user to a specific movie.

3.1.2 Data wrangling:

Our timestamp variable is in date-time format. It represents the time in which the rating was provided since January 1, 1970. We have to make an approximation. Since we don't have the exact movie release date, we will compute the rating delay for each movie (The difference between the rating time and the first rating time) in weeks.

4 Model selection:

4.1 Methodology:

In the course of our project we adopted a matrix factorization method to construct our algorithm. We assume the rating y is the same to all entries with the difference explained by random variations (bias). Thereby the goal is to minimize the residual ϵ for each observation k with b the biases total.

$$\epsilon \ k \ = y \ k \ - b \ k$$

Given that the average of all rating as a value of μ minimize the residual ϵ we will start by identifying the first element of our formula $\hat{y} \ k$. The idea is to work with the average rating $y \ k$ and gradually add the different biases caused by the main features. The default model performance (without considering the bias) is characterized by the following RMSE:

```
## [1] 1.060355
```

Let's have some graphical insight into our data. we made a distribution representation of a sample according to the main features:

4.1.1 MovieId:

lets start by looking at the graphic representing the mean rating of movies (Fig.1 a)). Ratings are rounded to nearest 10%. With the mean rating ranging from 1 to 5 Some movies clearly outperform each other. Some movies are more rated than others (Fig.1 b). In fact, the number of ratings varies from 1 to 31362. For a low number of rating, the rating of the film is not reliable since it is based on a judgment of few people. That will be discussed it the tuning chapter.

4.1.2 UserId:

Not all the movies are rated by every user. To be exact users' number of ratings fluctuate between 10 and 6616(Figure 2.a). Some users are clearly more active than others and thus more fastidious.
The standards for a good movie varies from person to another(Figure 2.b). The average rating varies from a user to the other ranging from 2 to 5 stars. Some users love every movie they watch, others less.

4.1.3 Rating time:

The rating time metric is ranging between 1 week and 731 weeks(Figure 3.a). Blockbuster movies get so much attention in the early stage they get released. In fact some movies get rated up to 694 times during the first month of its release (case of the movie Ghost). On the other hand some independent movies get rated less. In Figure 3.b we can clearly see that older movie ratings -more than 600 weeks (about 13 years)- have better overall rating.

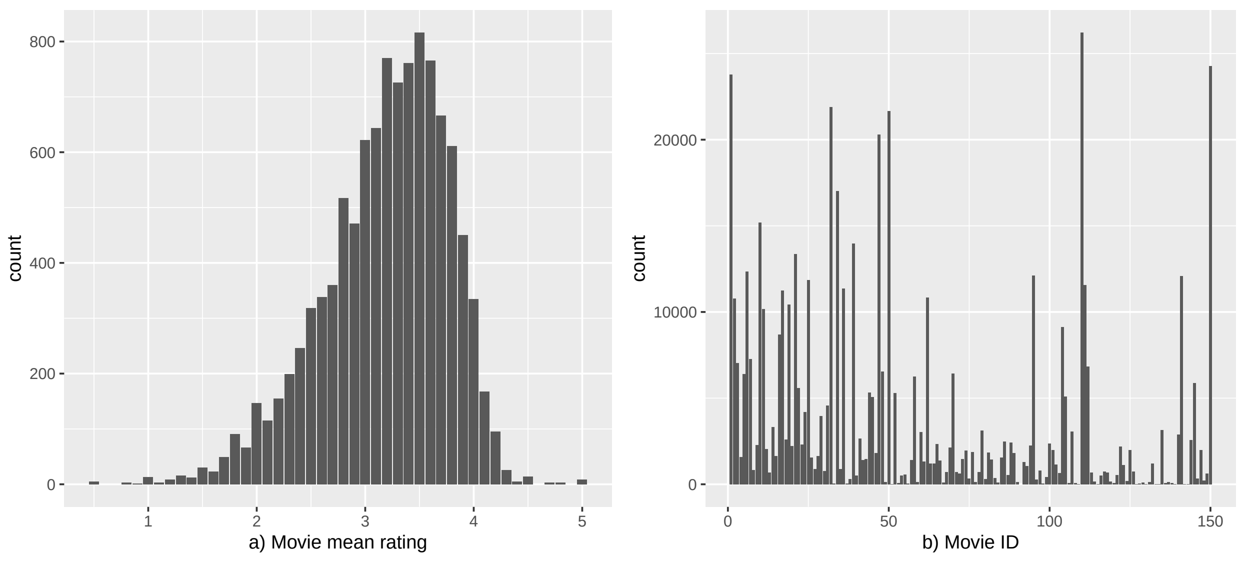


Figure 1: Movie rating distribution

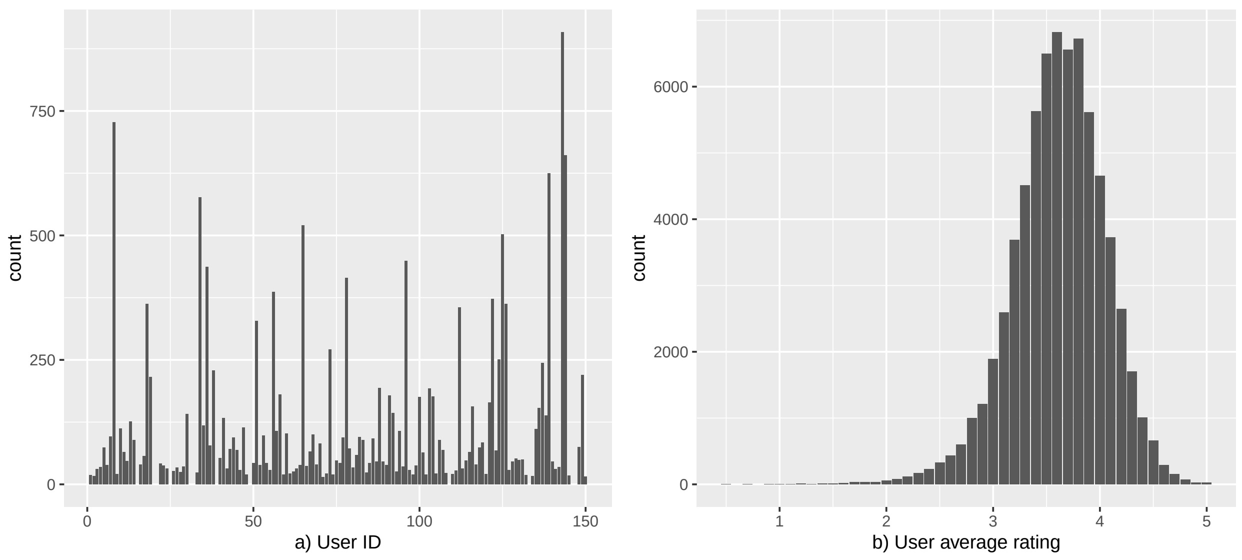


Figure 2: User rating distribution

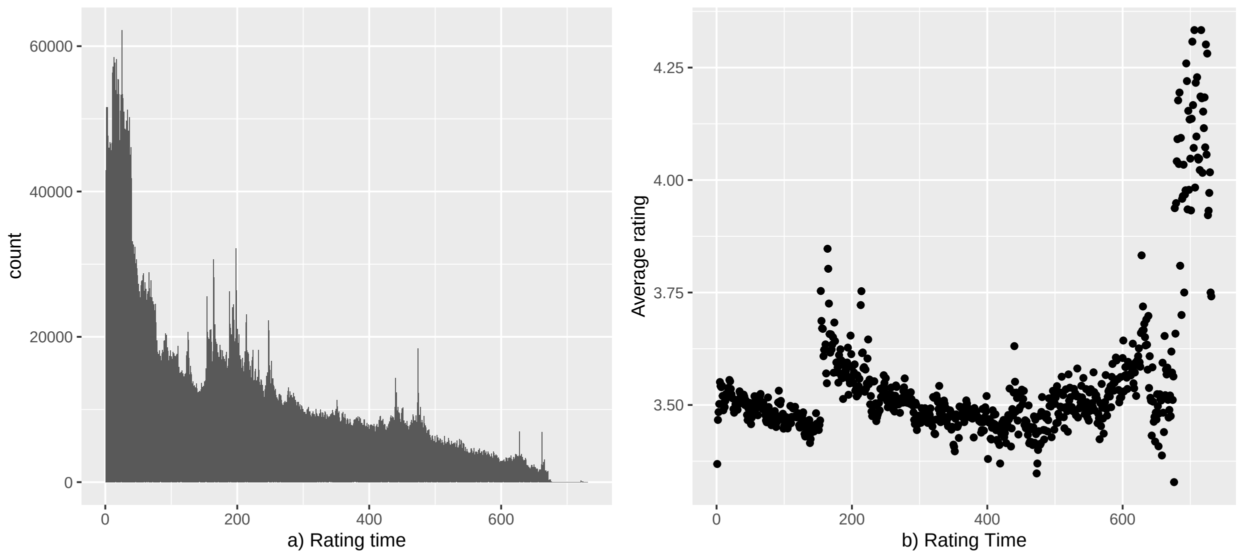


Figure 3: Rating time distribution

Each movie rating will have a different set of predictors Based on these observations we are building our model characterized by:
* movie's overall performance
* user's overall evaluation
* Rating time

Our method is analytically described by the formula:

$$Y\ i,u,t\ = \hat{\mu} + \hat{b}\ u\ + \hat{b}\ i\ + \hat{b}\ t\ + \epsilon$$

where
** $\hat{b}\ i$ is a movie specific effect
** $\hat{b}\ u$ is a user specific effect
** $\hat{b}\ t$ is a time specific effect
penalized with the independent error ϵ

4.1.4 The combination effect:

In order to determine these effects we first calculate the mean of the ratings $\hat{\mu}$ and the movie specific effect $\hat{b}\ i$, estimate the user specific effect $\hat{b}\ u$ as the mean of $y\ i,u -\ \hat{b}\ i$ than estimate the time effect $\hat{b}\ t$ as the mean of $y\ i,u -\ \hat{b}\ i\ -\ \hat{b}\ u$.

4.2 Tuning:

Having movies with variable rating times can alter our judgment of the rating. Let's take a look at the best 5 movies according to our model:

```
## # A tibble: 6 x 3
##   movieId n_ratings average_rating
##     <dbl>     <int>          <dbl>
## 1    5805         2          0.5
## 2    8394         1          0.5
## 3   61768         1          0.5
## 4   63828         1          0.5
## 5   64999         2          0.5
## 6   61348        29          0.776
```

and the 5 worst ones:

```
## # A tibble: 6 x 3
##   movieId n_ratings average_rating
##     <dbl>     <int>          <dbl>
## 1   33264         2              5
## 2   42783         1              5
## 3   51209         1              5
## 4   53355         1              5
## 5   64275         1              5
## 6   64280         1              5
```

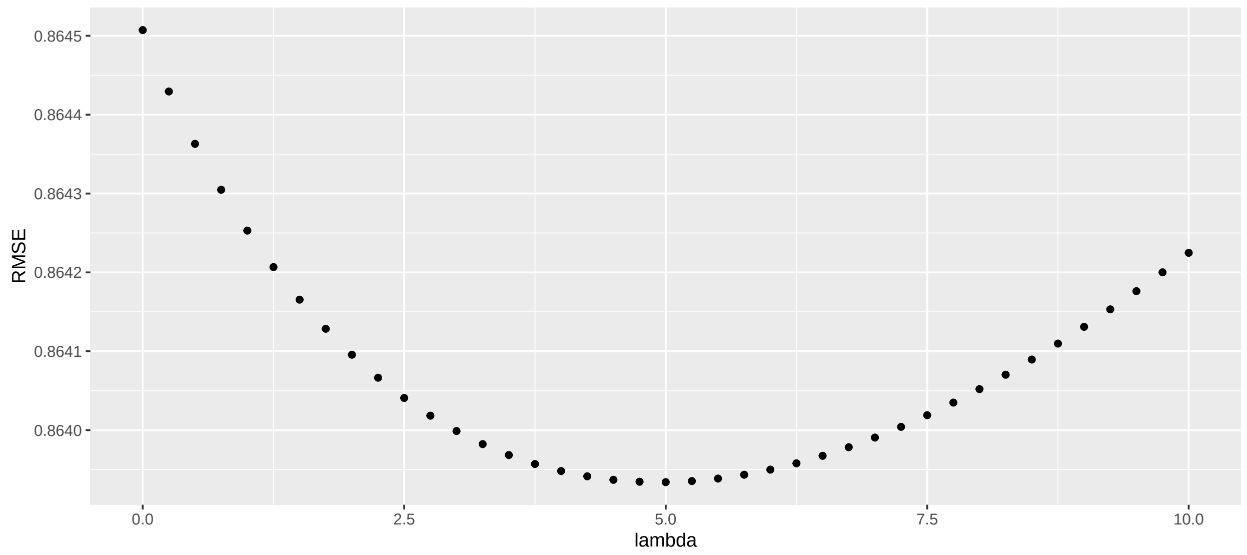
As we can see the number of ratings is very low (less than 5 for the most part). We have to penalize the extreme (large and low) estimates that have low sample size. To tackle this issue we are going to regularize the average rating depending on the number of ratings adding a constraint λ (tuning parameter) in the equation. λ represents the amount of shrinkage: The larger the value of λ, the greater the amount of shrinkage Our bias equations will look like this:

$$\hat{b}\ i\ (\lambda) = \frac{1}{\lambda + n\ i} \sum_{u=1}^{n\ i} (Y\ i, u, t\ - \hat{\mu})$$

$$\hat{b}\ u\ (\lambda) = \frac{1}{\lambda + n\ u} \sum_{u=1}^{n\ i} (Y\ i, u, t\ - \hat{\mu} - \hat{b}\ i\ )$$

$$\hat{b}\ t\ (\lambda) = \frac{1}{\lambda + n\ t} \sum_{u=1}^{n\ i} (Y\ i, u, t\ - \hat{\mu} - \hat{b}\ i\ - \hat{b}\ u\ )$$

With $n\ i$, $n\ u$, $n\ t$ are respectively number of ratings made for a movie i, number of rating made by a user u, and number of ratings made after a time duration t. If these terms are negligibly small with respect to λ, the bias value will be close to 0. Selecting the optimal tuning parameter is done through cross validation to minimize the RMSE of our model.



In this case the parameter λ equals 5.

5 Results:

In this chapter we are going to quantify the performance of our model by testing it on a separate validation data set. We will proceed by calculating the RMSE we get in each step of our analysis.

method	RMSE
Average	1.060355
Movie effect	0.943953
Movie+User effect	0.885521
Movie+User+Time effect	0.888258
Movie+User+regulerization effect	0.865171
Movie+User+Time+regulerization effect	0.864724

6 Conclusions:

In order to test movie performances across a full spectrum of users from the Movielens website we selected a 10 million version of a rating dataset and performed regularized matrix factorization to predict movie ratings. Our final model accomplished a final root mean square error of 0.864724.
The most important of future work is to expand the knowledge acquired during the execution of this project working on a chosen set of data.

7 References:

https://grouplens.org/datasets/movielens/

Trevor Hastie, Robert Tibshirani, Jerome Friedman. The elements of statistical learning, Data mining, inference and prediction. second edition.

https://leanpub.com/datasciencebook