Data Science Capstone - MovieLens Report

HarvardX 125.9x Capstone Project - Part 1

Julian Fuchs

October 10, 2024

Contents

1 Introduction

1.1 Motivation

The goal of this project is to train an algorithm to predict the rating of a movie. The dataset used is the Movielens 10M dataset [1]. As instructed, we use a slightly modified version of the dataset, the *edx* dataset. The code for this was provided by the instructor in the course materials. The performance of our model is evaluated using the **RSME** on a course provided subset, the *final_holdout_set*. Our aim is to build a model with an RMSE < 0.8649.

1.2 The Dataset

The Dataset contains 9000055 observations and consists of 6 columns:

- **rating -** The value we are trying to predict. Ratings are made on a 0 to 5 star scale, with half-star increments.
- **userId -** anonymized user identifier
- **movieId -** The movie Id
- **timestamp -** timestamp of submission
- **title -** The movie title
- **genres -** The genres associated with the movie

	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 following sections will guide you threw Data Exploration, Modelbuilding and validation and a Conclusion.

2 Analysis

2.1 Exploratory Data Analysis

Because of the size of the dataset, extensive data wrangling/exploration and standard algorithms like the *lm* are not possible on this dataset due to the immense computing time this would need on most machines. However we can draw some assumptions to build a linear model.

- **rating**

Before we begin with the analysis we have to understand the value we are trying to predict. Therefore we can take a look at the mean (3.5124652) and the histogram of the ratings.

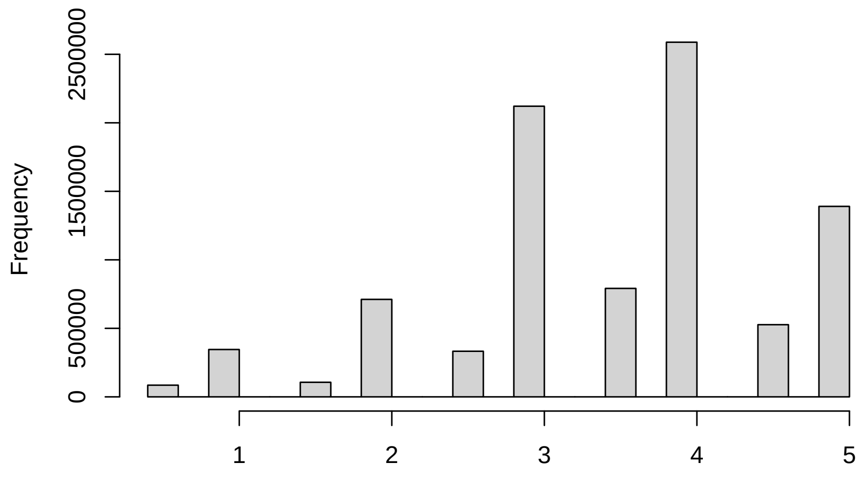

Figure 1: Ratings

As you can see in Figure 1 the ratings in the dataset are not centered. The Distribution shows that there are more positive ratings (≥ 3) than rather negative ratings (< 3).

- **userId & movieId**

There are 9081 different Movies in the dataset. However there are 1596 movies in the dataset which are rated 15 times or even less. These ratings are not really helpful to our analysis as they introduce more variability.

Having removed these ratings there are now 8988466 ratings left.

As there are 9081 different Movies in the dataset, but just 69878 users suggests that not every user has rated every movie. You can simply explain that if you multiply those two together you get $634562118 < 8988466$, which is the number of observations recorded in the dataset.

- **timestamp**

A Timestamp is representing the seconds elapsed since January 01, 1970 UTC [2]. This means we can convert the timestamp to the specific date of the submission.

	movieId	userId	timestamp	date
1	122	1	838985046	1996-08-02
2	185	1	838983525	1996-08-02
4	292	1	838983421	1996-08-02
5	316	1	838983392	1996-08-02
6	329	1	838983392	1996-08-02
7	355	1	838984474	1996-08-02

- **genres**

According to Dataset documentation [3], there are 18 genres available to be assigned to a movie:

genre	count
Action	2559603
Adventure	1908207
Animation	467037
Children	737795
Comedy	3537704
Crime	1326623
Documentary	92357
Drama	3903567
Fantasy	925345
Film.Noir	118434
Horror	690494
Musical	432570
Mystery	567935
Romance	1710482
Sci.Fi	1340675
Thriller	2324602
War	510452
Western	188956

As you can see not all Genres are equally represented.

2.2 RMSE

The Root Mean Squared Error is defined as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(\hat{y}_k - y_k)^2}$$

where $\hat{y}$ are the actual and y are the predicted values. More information [4].

In code it is calculated like this:

```r
rmse <- function(y_actual, y_predicted) {
  sqrt(mean((y_actual - y_predicted)^2))
}
```

2.3 Train /Test set

In order to train our Algorithm and validate its accuracy, we will split the edx dataset in:

- **Training set** - will be used to train the algorithm *(75%, 6741348 observations)*
- **Test set** - will be used to validate the algorithms accuracy *(25%, 2247118 observations)*

2.4 Modelbuilding

The attempts used in this section (2.4) are linear models to predict the ratings presented by **Rafael A. Irizarry** in his book *Introduction to Data Science* [5]. However we are trying to extend on this ideas in a later section (2.5).

2.4.1 Trivial Model

The most basic model we can think of is predicting the rating as just the mean.

$$y_{predicted} = \mu \quad \text{where} \quad \mu = \frac{1}{n}\sum_{k=1}^{n} \hat{y}_k \qquad \hat{y} \text{ are the actual values}$$

So, using this model by prediction the rating with just the mean μ of the actual ratings ($\mu = 3.5132928$), we get:

Model	RMSE	Goal
Trivial Model	1.060647	FALSE

This should be seen as a sort of benchmark because it is the most basic we can get and we are far away from our goal.

2.4.2 Movie Effect

However as indicated earlier some movies (e.g. Blockbusters) are just higher rated then other movies. So to improve our model and to compensate for this effect, we adjust our model like this:

$$y_{predicted} = \mu + b_i \quad \text{where} \quad b_i = \frac{1}{n}\sum \hat{y}_i - \mu \qquad \hat{y} \text{ are the actual values}$$

Model	RMSE	Goal
Trivial Model	1.0606472	FALSE
Movie Effect	0.9435642	FALSE

You can see that this has improved our model quite a bit, but still not good enough.

2.4.3 User Effect

As with movies, it is understandable that not all users rate films the same way. Some may be more critical, others more enthusiastic. So we adjust out model like this:

$$y_{predicted} = \mu + b_i + b_u \quad \text{where} \quad b_u = \frac{1}{n} \sum \hat{y}_{u,i} - \mu - b_i \qquad \hat{y} \text{ are the actual values}$$

Model	RMSE	Goal
Trivial Model	1.0606472	FALSE
Movie Effect	0.9435642	FALSE
Movie + User Effect	0.8664191	FALSE

We were able to reduce the RMSE again quite substantially but to reach our goal we still have to improve it.

2.4.4 Movie & User Regularization

As User & Movie Effect was quite an improvement to our RMSE, we can do some more with a technique called Regularization. With Regularization we are trying to reduce the impact of insignificant observations, like we did before when removing the less rated movies. But there is still some room for improvement.

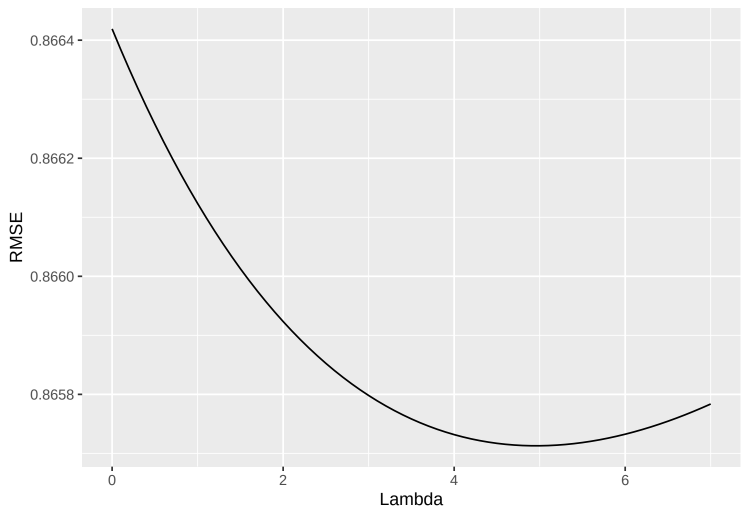

Figure 2: Minimizing Lambda

After using cross validation we see that $\lambda = 5$ minimizes the RMSE. Using the selected λ we can take a look at the RMSE.

Model	RMSE	Goal
Trivial Model	1.0606472	FALSE
Movie Effect	0.9435642	FALSE
Movie + User Effect	0.8664191	FALSE
Regularization	0.8657129	FALSE

This improved the RMSE again by a small amount, but we still have room for improvement.

2.5 Extending the Model

2.5.1 Matrix Factorization

However up to this point our models were provided by our Instructor **Rafael A. Irizarry** in the course materials and in his book [5]. But in order to reach our goal we must go a step further. To further enhance our predictions we turn to a concept called **Matrix Factorization**.

Matrix Factorization is an algorithm that constructs two matrices by decomposing an matrix called the user-interaction-matrix[6].

In our example the user-interaction-Matrix is constructed by the ratings, userId and the movieId columns.

We will use a built-in function to find the best tune options for the algorithm:

dim	costp_l1	costp_l2	costq_l1	costq_l2	lrate	loss_fun
40	0	0.01	0	0.1	0.1	0.8060556

Using these Parameters we are training our Matrix factorization algorithm.

Model	RMSE	Goal
Trivial Model	1.0606472	FALSE
Movie Effect	0.9435642	FALSE
Movie + User Effect	0.8664191	FALSE
Regularization	0.8657129	FALSE
Matrix Factorization	0.7982787	TRUE

We are able to reach our goal with an RMSE of 0.7982787 using Matrix factorization on our test set.

3 Results

Model	RMSE	Goal
Trivial Model	1.0606472	FALSE
Movie Effect	0.9435642	FALSE
Movie + User Effect	0.8664191	FALSE
Regularization	0.8657129	FALSE
Matrix Factorization	0.7982787	TRUE

As you can see above we have reached our goal using the Matrix Factorization technique. To validate our result and fulfill the course objective we are validating the result using the *final_holdout_set* mentioned at the beginning.

Model	RMSE
Matrix Factorization - Final	0.7982615

As you can see the Algorithm does perform with quite simmilar accuracy on the test and the final_holdout set.

4 Conclusion

Our goal in this project was to train an algorithm that could predict Movieratings using the MovieLens 10M[1] Dataset. We explored the dataset and constructed different linear models to make more accurate predictions. As our final model we use the Matrix Factorization algorithm, as it could make the most accurate predictions.

There is however still room for improvement. Future work could address more extensive data exploration, data preprocessing and optimizing the algorithm parameters.

5 References

[1] : https://grouplens.org/datasets/movielens/10m/

[2] : **Wikipedia contributors** (2024, October 5) *Unix time* In Wikipedia, The Free Encyclopedia. Available at: https://en.wikipedia.org/wiki/Unix_time (Last Accessed: 09 October 2024).

[3] : **GroupLens** (no date), *MovieLens 10M/100k Data Set README* Available at: https://files.grouplens.org/datasets/movielens/ml-10m-README.html (Last Accessed: 10 October 2024).

[4]: **Wikipedia contributors** (2024, October 10) *Root mean square deviation* In Wikipedia, The Free Encyclopedia. Available at: https://en.wikipedia.org/wiki/Root_mean_square_deviation (Last Accessed: 10 October 2024).

[5] : **Irizarry, R.A.** (no date) 2024-05–16th edn, *Introduction to Data Science*. 2024-05–16th edn. Available at: http://rafalab.dfci.harvard.edu/dsbook/ (Last Accessed: 09 October 2024).

[6] : **WIkipedia contributors** (2024, August 23) *Matrix factorization (recommender systems)* In Wikipedia, The Free Encyclopedia. Available at: https://en.wikipedia.org/wiki/Matrix_factorization_(recommender_systems) (Last Accessed: 09 October 2024).