MovieLens Project Report

Victor Tapia

30/5/2019

Introduction

This is the first of two Capstone Projects that will be presented in order to complete the Data Science Professional Certificate.

This project is based on the Netflix Prize which was held by the company from 2006 to 2011 or until the Grand Prize was awarded. They were seeking to improve their collaborative filtering algorithm for rating prediction based on previous rating without any other information about users or films but number assigned for the contest. (Wikipedia 2018).

The prize was given to the team who can improve the RMSE by 1% over the previous year's result. The Progress Prizes for 2007 and 2008 were won respectively by the teams called "BellKor" and "BellKor in BigChaos" and the Grand Prize was winned by a team called "BellKor's Pragmatic Chaos" on July 26, 2009 with a RMSE of 0.856704 which represented an 10.06% improvement over the contest's baseline. The team "The Ensemble" was able to get the a RMSE of 0.856714, but the prize was given to the first team because the rules specified that the RMSE was limited to 4 decimal places and the prize was given to the first entry that was received (Feuerverger, He, and Khatri 2012).

The RMSE (Residual Mean Error) can be expressed using the following formula:

As the Netflix datasets are not publicly availables, we will be using the 10M version of the MovieLens dataset. This dataset consist of a table with 9000055 rows, each one containing a rate made by a specific user for a specific movie and a validation set with 999999 observations (10% of the data).

The dataset contains rates for 10677 different movies made by 69878 different users. The movies with more rates are the following:

movieId	title	count
296	Pulp Fiction (1994)	31362
356	Forrest Gump (1994)	31079
593	Silence of the Lambs, The (1991)	30382
480	Jurassic Park (1993)	29360
318	Shawshank Redemption, The (1994)	28015
110	Braveheart (1995)	26212
457	Fugitive, The (1993)	25998
589	Terminator 2: Judgment Day (1991)	25984

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

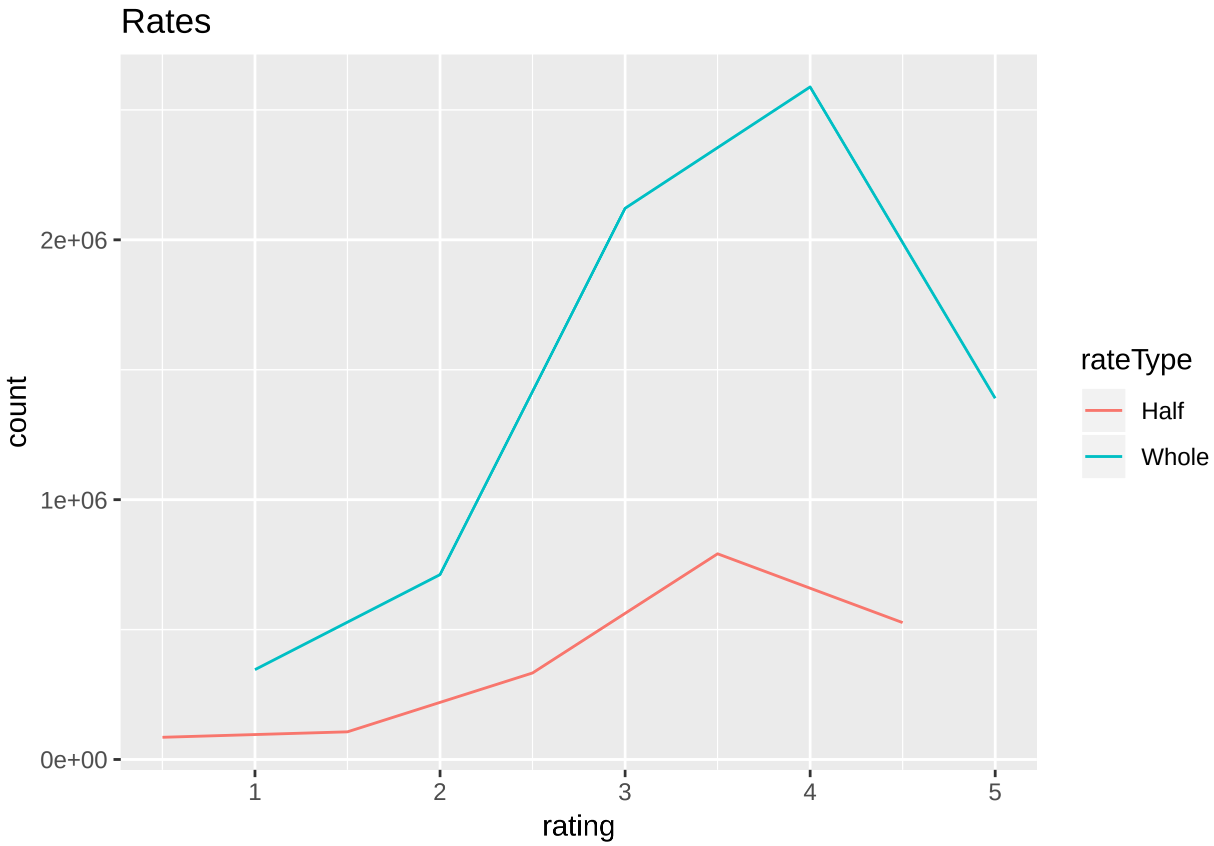
Figure 1: RMSE formula

movieId	title	count
260	Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)	25672
150	Apollo 13 (1995)	24284

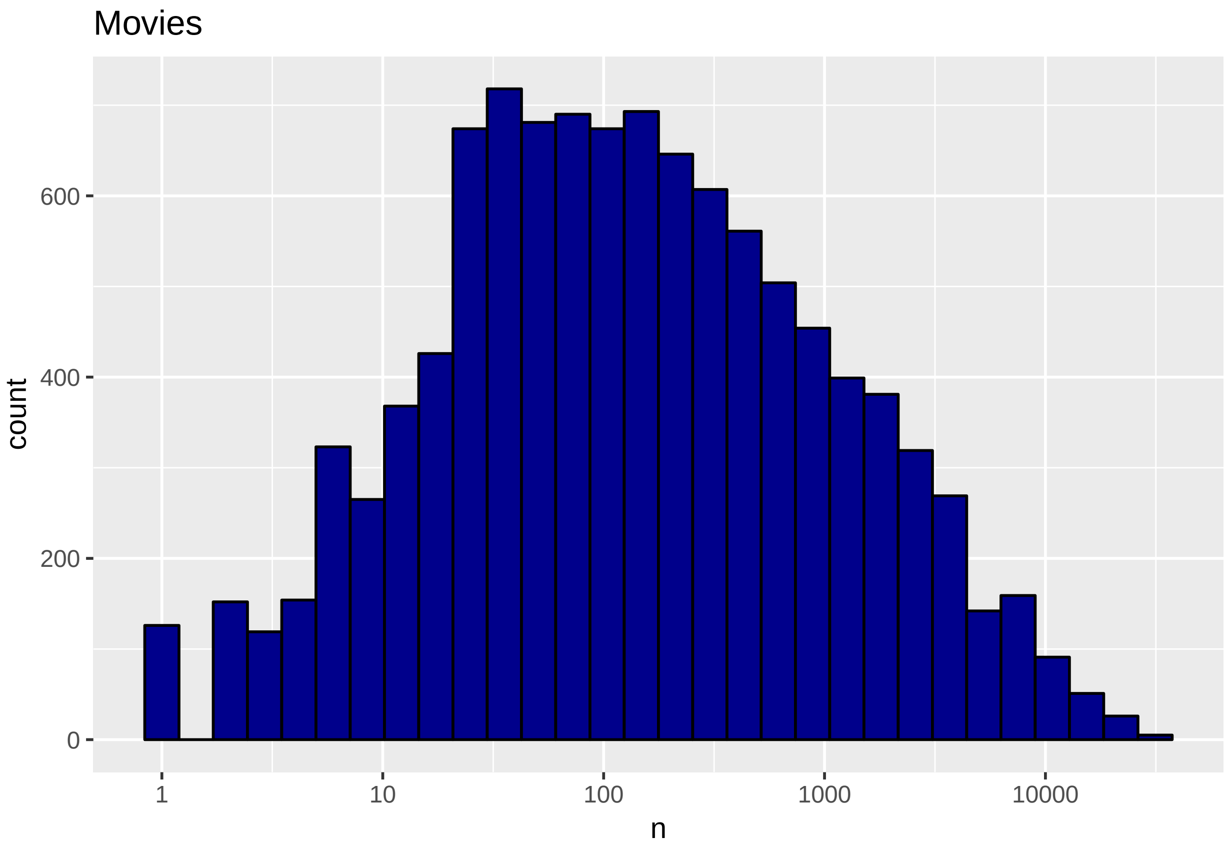
And the less rated movies are the following:

movieId	title	count
3191	Quarry, The (1998)	1
3226	Hellhounds on My Trail (1999)	1
3234	Train Ride to Hollywood (1978)	1
3356	Condo Painting (2000)	1
3383	Big Fella (1937)	1
3561	Stacy's Knights (1982)	1
3583	Black Tights (1-2-3-4 ou Les Collants noirs) (1960)	1
4071	Dog Run (1996)	1
4075	Monkey's Tale, A (Les Château des singes) (1999)	1
4820	Won't Anybody Listen? (2000)	1

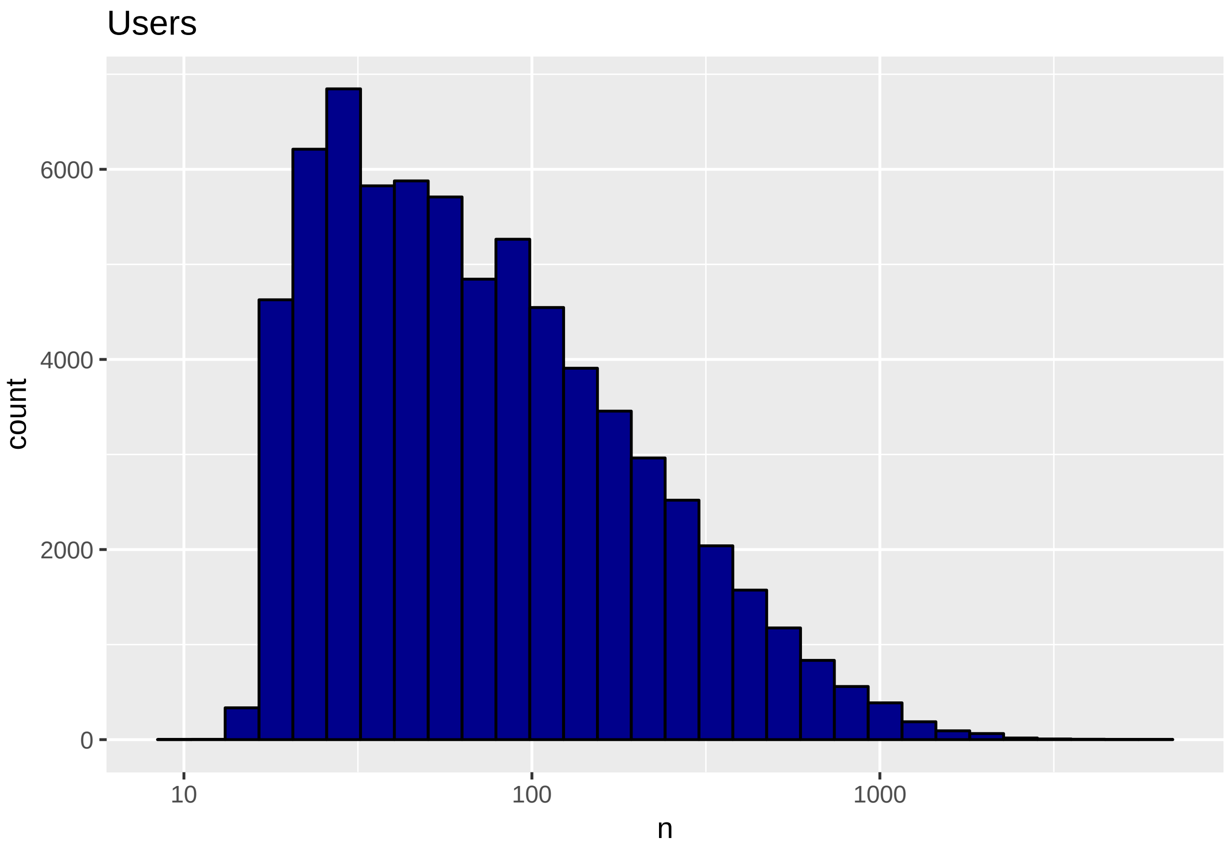
This plot shows how the data is distributed by rating. The rates were separated in half and whole star points because the latest seems to be more frequent.



This plot shows how the rates are distributed by movies.



In this plot we can see that some users are more active than others.



The following table shows how the ratings are distributed by genre.

genres	count
Drama	3910127
Comedy	3540930
Action	2560545
Thriller	2325899
Adventure	1908892
Romance	1712100
Sci-Fi	1341183
Crime	1327715
Fantasy	925637
Children	737994
Horror	691485
Mystery	568332
War	511147
Animation	467168
Musical	433080
Western	189394
Film-Noir	118541
Documentary	93066
IMAX	8181
(no genres listed)	7

The goal of this project is to emulate the work done by the "BellKor's Pragmatic Chaos" team during the Netflix Prize and build a algorithm as close as possible to the one they made.

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

Figure 2: Average Model

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

Figure 3: Adjusting Movie

For this project, the algorithms reviewed during the course PH125.8x Data Science: Machine Learning will be tested and we will choose the best alternative.

Methodology

For this project all algorithms are based in the course PH125.8x Data Science: Machine Learning from this program.

Some algorithms like knn, Rborist, glm, lda and qda were tried previous to the final procedure, however, the time that the algorithms required to complete the processing was very high and in some cases, the GUI crashed before the process finished.

The first approach to solve the issue was predicting the same rating for all movies, regardless of user and movie. This model can be represented using the following formula

Here epsilon represents independent errors sampled from the same distribution centered at zero, and mu represents the true rating for all movies and users. For the dataset provided, the value of mu is 3.51.

Using this value as predictor, the algorithm is able to get a RMSE value of 1.06, which will be reported as "Just the average".

As seen before, different movies are rated differently, by adding the term bi to represent the average rating for movie i, it is expected to get an improvement in the model.

Using this model, we got a RMSE value of 0.94, which will be reported as "Movie Effect Model".

One of the concept used by the winning team to improve the algorithm was regularization. Regularization permits to penalize large estimates that come from small sample sizes. To estimate this, we minimize the following equation:

Using calculus, the following formula shows the values of b that minimize this equation, where ni is a number of ratings for movie i. Using this model, when n is large enough, the estimate will be stable.

After regularization, using a value of lambda 2.5, the new model got a RMSE value of 0.94 which will be reported as "Regularized Movie Effect Model".

But Movie it's not the only bias, it is also clear that there is a substancial variability across users as well, by adding the term bu to represent the average rating for user u. The model now looks like this:

This new model have a RMSE value of 0.87 which will be reported as "Movie + Users Effect Model".

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

Figure 4: Movie Regularization

$$\hat{b}_i(\lambda)=\frac{1}{\lambda+n_i}\sum_{u=1}^{n_i}\left(Y_{u,i}-\hat{\mu}\right)$$

Figure 5: Movie Regularization Formula

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

Figure 6: Adjusting User

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

Figure 7: Movie User Regularization Formula

Finally, it is required to use regularization at both Movie and User levels, this is the equation that would be minimized:

With this final model, the RMSE value is 0.864817 using a value of lambda 5.25, which is a 18.51% of improvement compared to the average method.

Results

After testing several methods and algorithms and discarding some specific algorithms like knn, Rborist, glm, lda and qda, among others, the following table shows the specific results for all the final models reviewed during the course:

method	RMSE
Just the average	1.0612018
Movie Effect Model	0.9439087
Regularized Movie Effect Model	0.9394544
Movie + User Effects Model	0.8653488
Regularized Movie + User Effect Model	0.8648170

It's clear that even though, the Movie + User Effects model is very close to the goal, after regularization the algorithm is improved.

Conclusions

For this specific problem, it's almost impossible to use most of the common machine learning algorithms such as knn, Rborist, glm, lda and qda, among others. Some of them were tried, but in most of the cases R crashed and in other the process taked more than 2 or 3 hours to complete.

The approach and models provided in the course PH125.8x Data Science: Machine Learning proven to be very effective for this kind of tasks.

The experience gained during the previous eight courses proven to be very useful when solving Data Science problems, however, as all courses are introductories, there are a road of kwnoledge ahead.

References

Feuerverger, Andrey, Yu He, and Shashi Khatri. 2012. "Statistical Significance of the Netflix Challenge." *Statistical Science* 27 (2). Institute of Mathematical Statistics: 202–31.

Wikipedia. 2018. "Netflix Prize." https://en.wikipedia.org/wiki/Netflix_Prize.