MovieLens_Project

Jesus Gastanaduy

18 de mayo de 2020

Introduction

Users usually consume a new product or service based on recommendations made by other users. This is clearly seen when deciding whether to watch or not to watch a movie. Companies such as Netflix use recommendation algorithms to predict how many stars a user will give a specific movie. Unfortunately, their data is not publicly available. However, the GroupLens research lab generated a dataset with over 10 million ratings for over 10,000 movies by more than 69,000 users. We used this dataset to create a movie recommendation algorithm.

We tried different models with different tuning parameters with the goal of minimizing as much as we could the error in our predictions. We started with some data exploration and construction of simple models; so then we could build on them to improve our prediction accuracy. The next sections will describe in detail our methodology as well as results, conclusions and future work recommendations.

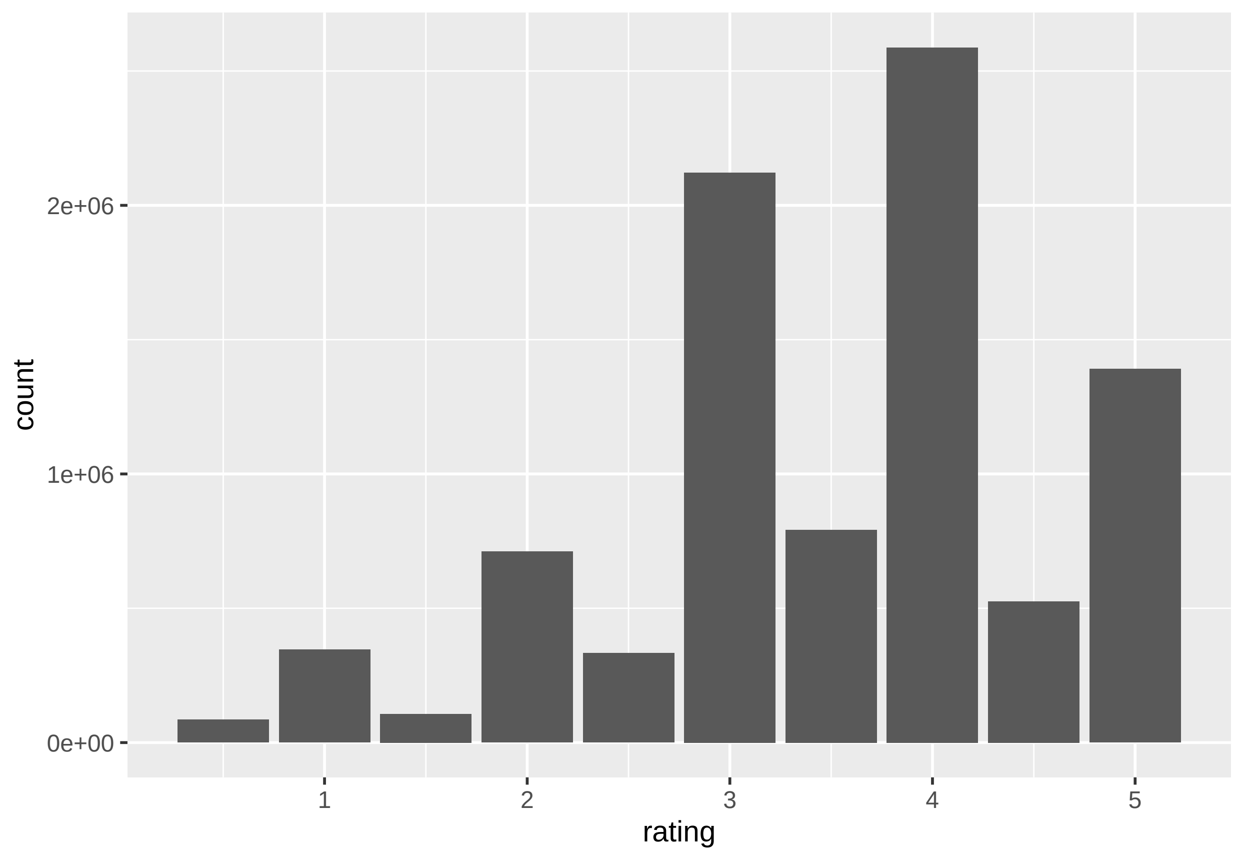
Methods

The MovieLens dataset contains two databases which match by movie ID. The first database contains the ratings given to a specific movie by a specific user. The data is in long format, so each row corresponds to a specific rating given to a movie by a user id. Similarly, the second database contains the specific movie title as well as the movie genre a specific movie id corresponds. Both databases were merged by movie id and stored in the "movielens" object, while keeping the long format.

Train and validation sets were created by randomly splitting the data into two: "edx" (90% of the original data) and "temp" (10% of the original data). To make sure only users and movies in the test set that appeared in the training set were included, a new object called "validation" was created; in which entries that did not have a match in the training set were removed and binded to the edx set.

As we wanted to create an algorithm that predicts ratings a specific user gives to a specific movie (a continuous variable), we wanted to minimize as much as we could the Residual Mean Squared Error (RMSE) of the algorithm on a test set. We created three main models by aggregation (i.e., by optimizing on the previous one).

We started with some data exploration about ratings in general by computing the rating mean of movies regardless of everything. This one was our first model. The plot below shows the distribution of all ratings. As it can be seen from the plot, it follows an asymmetric distribution to the left, so many more movies were rated with higher scores than lower ones.



We then explored the average movie rating by movie title. By seeing the titles ordered by mean rating, we could see that some movies had a perfect rating given by very few people. On the other hand, by sorting the movies by number of ratings received, we could see that some movies had been rated by many people while not achieving a perfect score. For this reason, we modeled the regularized movie effect; to penalize large estimates that are formed using small samples sizes.

```
## # A tibble: 10,676 x 3
##    title                                                              mean     n
##    <chr>                                                             <dbl> <int>
##  1 Blue Light, The (Das Blaue Licht) (1932)                           5        1
##  2 Constantine's Sword (2007)                                         5        1
##  3 Fighting Elegy (Kenka erejii) (1966)                               5        1
##  4 Hellhounds on My Trail (1999)                                      5        1
##  5 Satan's Tango (SÃ¡tÃ¡ntangÃ³) (1994)                                5        2
##  6 Shadows of Forgotten Ancestors (1964)                              5        1
##  7 Sun Alley (Sonnenallee) (1999)                                     5        1
##  8 Human Condition II, The (Ningen no joken II) (1959)                4.83     3
##  9 Human Condition III, The (Ningen no joken III) (1961)              4.75     4
## 10 Who's Singin' Over There? (a.k.a. Who Sings Over There) (Ko to t~  4.75     4
## # ... with 10,666 more rows
```

```
## # A tibble: 10,676 x 3
##    title                                                         mean     n
##    <chr>                                                        <dbl> <int>
##  1 Pulp Fiction (1994)                                           4.16 31336
##  2 Forrest Gump (1994)                                           4.01 31076
##  3 Silence of the Lambs, The (1991)                              4.21 30280
##  4 Jurassic Park (1993)                                        3.66 29291
##  5 Shawshank Redemption, The (1994)                            4.46 27988
##  6 Braveheart (1995)                                           4.08 26258
##  7 Terminator 2: Judgment Day (1991)                           3.93 26115
##  8 Fugitive, The (1993)                                        4.01 26050
##  9 Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977) 4.22 25809
## 10 Batman (1989)                                               3.39 24343
## # ... with 10,666 more rows
```

For our second model we had to control the total variability of the movie effects by tuning a penalty parameter lambda. To do this, we used cross validation on the edx set with a sequence of values from 0 to 10, which increased by 0.25 points. We calculated the RMSE of our predictions on the validation set. The lambda parameter which minimized the RMSE of this model was 2.75. We used this value as our penalty term.

Our third and final model included a user-specific effect. By doing some data exploration, we could see that some users gave movies very high ratings, while others were very critical about every movie. We modeled this user-specific effect after modelling the movie effect. Also, we included a penalization term to control the total variability of the user effect; the same as we did with our second model. The lambda parameter which minimized the RMSE of our third model was 5.5.

Results

We found that our third model was the one which minimized the RMSE the most. The table below shows the typical error when predicting a movie rating with each of our three models. The RMSE of our first model was above 1, which meant that our typical error in this case was larger than one star. This model, however, was created only for illustration purposes.

method	RMSE
Just the average	1.0606506
Regularized Movie Effect Model	0.9436515
Regularized Movie + User Effect Model	0.8649857

As it was mentioned, our second and third model both included a penalization term for regularization purposes. In other words we wanted to control for the variability in both, the movie and user effects. The final results show that our third model improved its performance on the validation set in comparison with the second one; and our second model, in comparison with the first one.

Conclusion

To sum up, we managed to create an algorithm which was able to make movie recommendations based on regularized movie and user effects. Our final typical error when predicting a specific rating was below 0.86499 stars, which is quite decent. Future work may include groups of movies and users with similar rating patterns in their modeling approach by using matrix factorization.