Capstone: Movie Recommendation System

Juhe Nie

2020/2/10

Executive Summary

The objective of this project is to create a recommendation system using the Movielens dataset. The Movielens dataset has been divided into two subsets "**edx**" and "**validation**". We train and modify machine learning algorithm in the **edx** set and predict movie ratings in the **validation** set. We then split the edx dataset into a train set "**train_set**" and a test set "**test_set**", where **train_set** is used to build algorithm and **test_set** is used to test.

In our project, the root mean squared error (RMSE) is used as loss function. We start our prediction model with assuming the same rating for every data entry. We then use bias information of each user, movie, genre and time to learn the effect of users and three items (movie, genre and time) independently. To improve the results, we use regularization to penalize large estimates that come from small sample sizes. The best tuning parameter of regularization is equal to 5, which is chosen by using full-cross validation. After training and revising, we get RMSE of **test_set** as 0.864372. Finally, we use our recommendation model to predict ratings in the **validation** set and RMSE is **0.864568** (< 0.8649).

Methods

1. Basic information of edx dataset

At first, we need to observe some basic information from edx. Dataset edx has 6 objects (userId, movieId, rating, timestamp, title and genres) and 9,000,055 entries. Information from these objects will be used to create algorithm model and train data.

```
edx_names <- names(edx)
edx %>% summarize (n_users=n_distinct(userId), n_movies = n_distinct(movieId))
```

2. Divide edx set

We then partition edx set into two subsets. We keep 95% of data in **train_set** to create and train the algorithm, and we use the other 5% of data to test the effect of our algorithm. This division ratio looks large, however, the edx set is a large dataset that has around nine million entries, and its **test_set** also has around 450 thousand entries which are enough to make a test. Therefore, this division is reasonable.

```
set.seed(100)
edx_testindex <- createDataPartition(y = edx$rating, times = 1, p = 0.05, list = FALSE)
train_set <- edx[-edx_testindex,]
edx_temp <- edx[edx_testindex,]
```

```
test_set <- edx_temp %>%
  semi_join(train_set, by = "movieId") %>%
  semi_join(train_set, by = "userId")
edx_removed <- anti_join(edx_temp, test_set)
train_set <- rbind(train_set, edx_removed)
```

3. RMSE

We use root mean squared error (RMSE) as our loss function to compare our prediction models to the true rating values. We denote N as the number of data entries used, $y_{u,i}$ as the rating for movie i by user u, and $\hat{y}_{u,i}$ as our prediction, then RMSE is defined as follows:

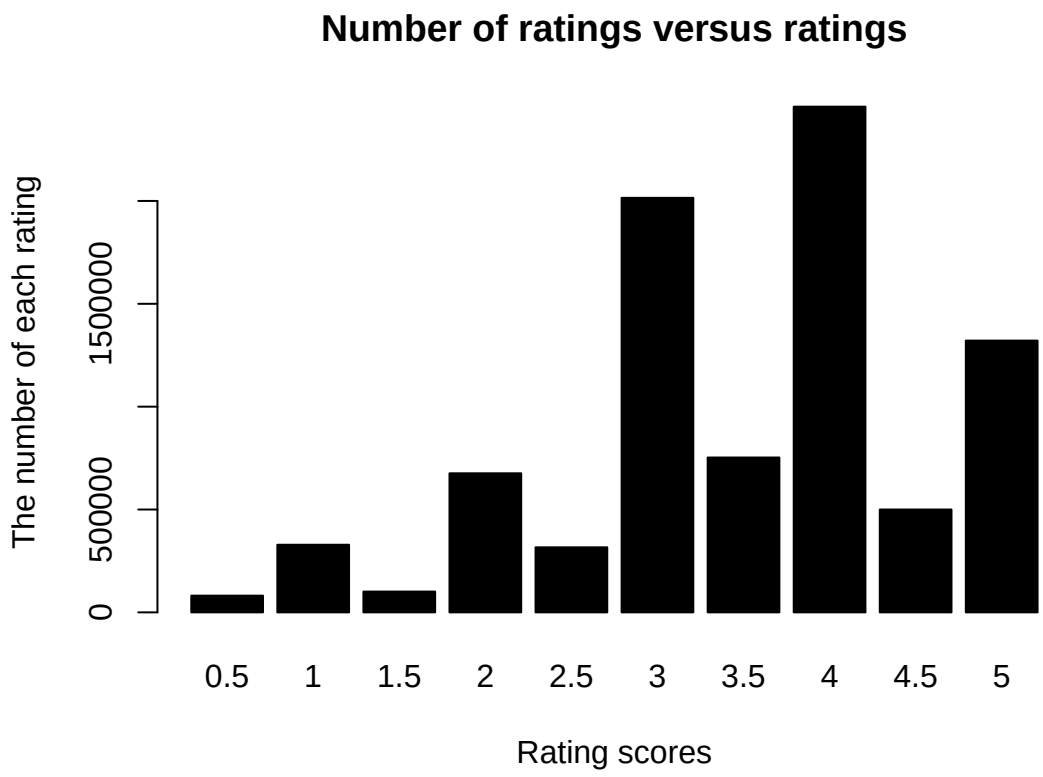
$$\sqrt{\frac{1}{N}\sum_{u,i}(\hat{y}_{u,i} - y_{u,i})^2}$$

We then create a function to calculate RMSE:

```
RMSE <- function(true_ratings, predicted_ratings){sqrt(mean((true_ratings-predicted_ratings)^2))}
```

4. Model 1: Start with the average rating

To start with evaluating ratings, we firstly want to observe the overall rating distributions. As figure below shows, more ratings are clustered between 3 and 4.



We then start with a model that assumes the same rating μ for all data entries, with all the differences explained by random variation $\epsilon_{u,i}$:

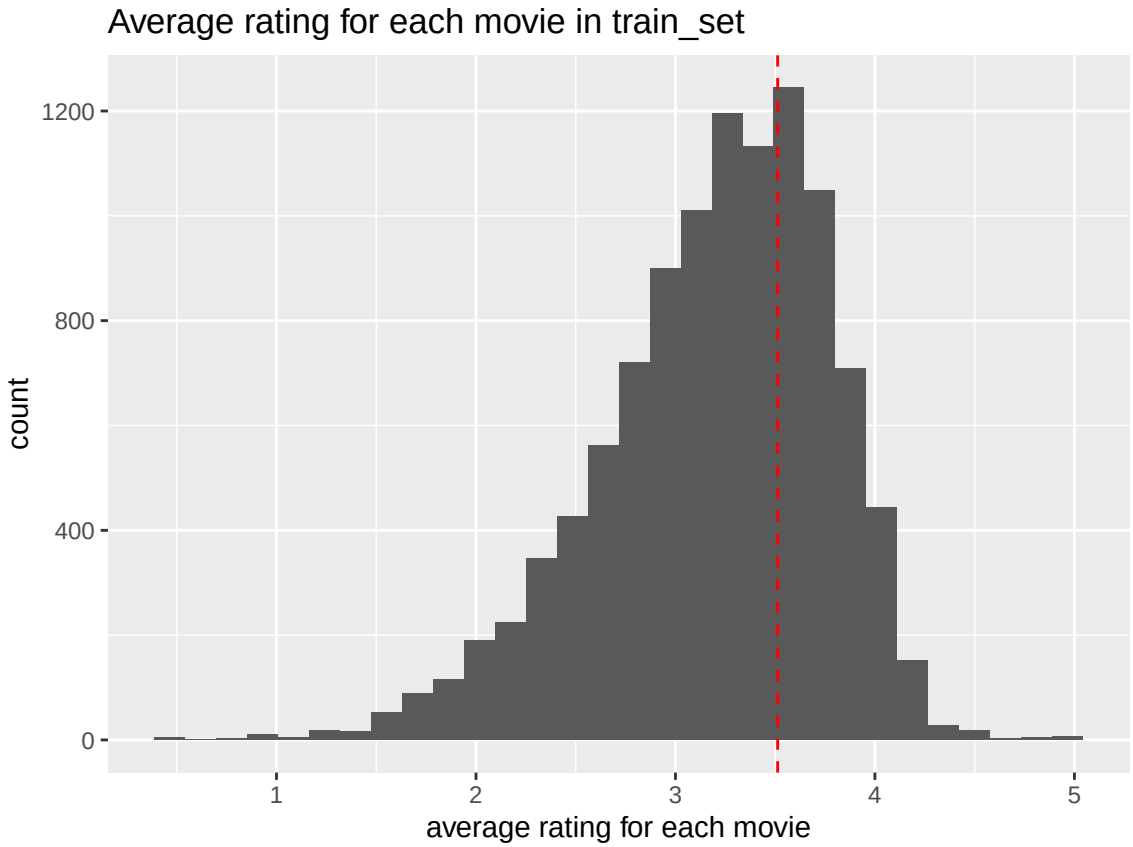
$$y_{u,i} = \mu + \epsilon_{u,i} \tag{1}$$

In this project, the estimate of rating μ, which can minimize the squared error, is the average rating across all use and the value of μ equals to 3.512452. This value is reasonable since it is in the range from 3 to 4.

We regard Equation (1) as our naive model and use it to predict ratings in **test_set**. The RMSE result is 1.059933.

5. Model 2: Add specific effects of user, movie, date and genre

We then observe the effects of users, movies, dates and genres. In real life, some movies tend to have higher marks than average, while some movies tend to get more low rating scores. This can be proven by the figure below:



This phenomenon leads to a movie bias between different movies, thus we add a new term b_i to Equation (1) to represent the specific effect of each movie i. The value of b_i is the mean value of the difference between true ratings of movie i and μ:

$$b_i = \frac{1}{n_i} \sum_{u=1}^{n_i} (y_{u,i} - \mu),$$

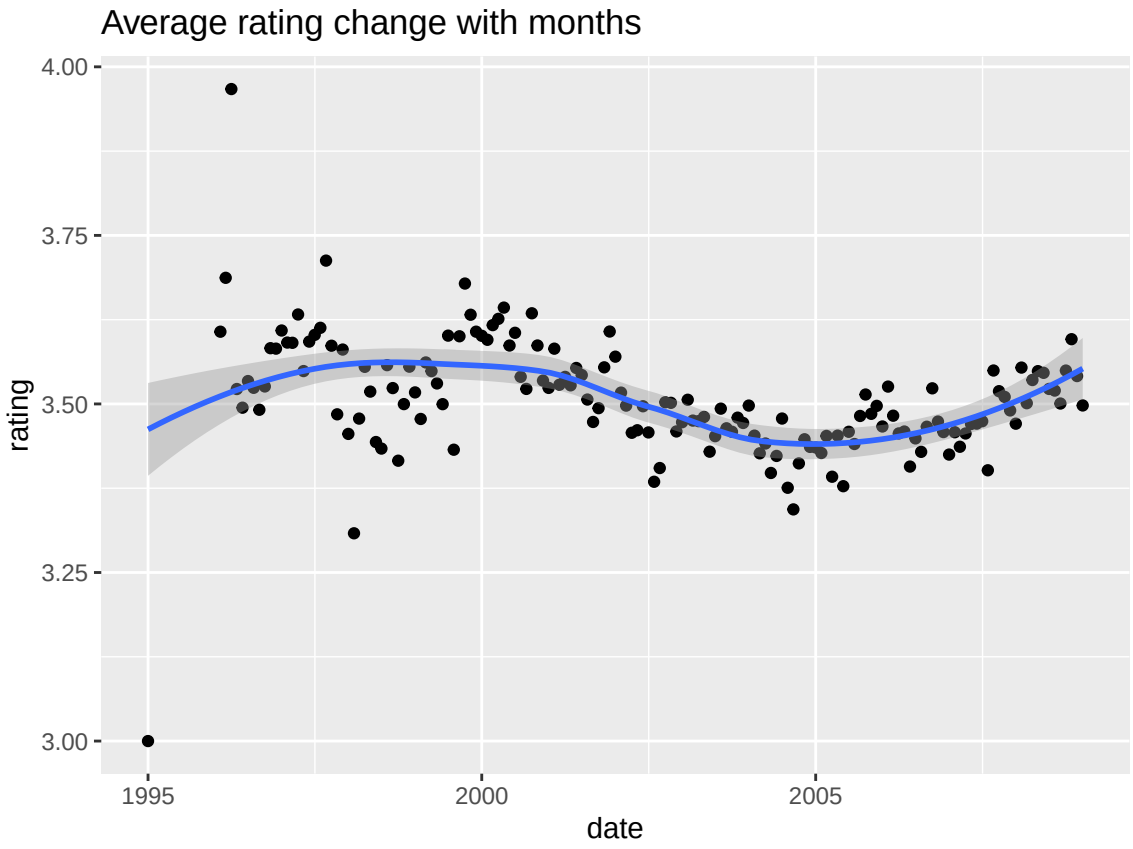
where n_i is a number of rating for movie i. The code for calculating b_i is

```
movie_avgs <- train_set %>% group_by(movieId) %>% summarize(b_i=mean(rating-mu))
```

Due to the same reasons, we add a term b_u for each user u and a term $g_{u,i}$ for each genre combination.

```
user_avgs <- train_set %>% left_join(movie_avgs,by="movieId")%>% group_by(userId) %>%
  summarize(b_u = mean(rating-mu - b_i))
genre_avgs <- train_set %>% left_join(movie_avgs,by="movieId") %>%
  left_join(user_avgs,by = "userId") %>% group_by(genres) %>%
  summarize(g_ui = mean(rating- mu - b_i - b_u))
```

About the impact of time, we define $d_{u,i}$ as the timestamp for user's u rating of movie i. We convert timestamp to a form of standard time and group them in months. As figure below shows, there is some evidence of a time effect on average rating. Therefore, we add a new function term $f(d_{u,i})$ to present the effect of each timestamp.



Taking effects of users, movies, dates and genres all into consideration, our model is written as:

$$y_{u,i} = \mu + b_i + b_u + f(d_{u,i}) + g_{u,i} + \epsilon_{u,i}$$

We use this model to predict rating in **test_set** again and get RMSE as 0.864962.

6. Model 3: Regularization

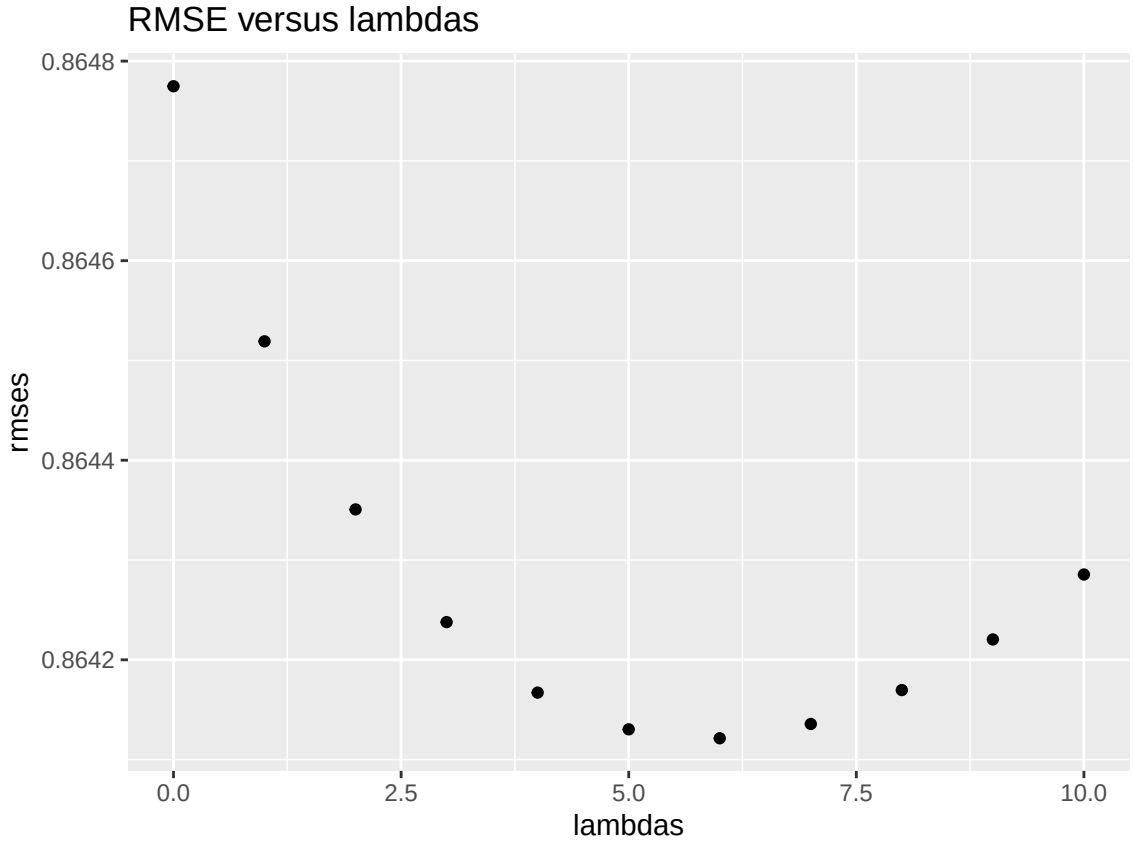
To improve our results, we will use regularization. Regularization constrains the total variability of the effect sizes by penalizing large estimates that come from small sample sizes. By using regularization, the formula

of b_i is written as

$$b_i = \frac{1}{n_i + \lambda} \sum_{u=1}^{n_i} (y_{u,i} - \mu),$$

where λ is a tuning parameter. In the same way, we can get new expressions of b_u, $f(d_{u,i})$ and $g_{u,i}$.

We then use cross-validation to find the optimal λ. Figure below shows RMSE of test_set in edx when predicting with different values of λ and we see that the root mean square error reaches its minimum value 0.864372 when $\lambda = 5$. The establishment of algorithm model is done.



Results

Table below lists the RMSE result of predicting ratings in test_set of edx with different machine learning models. We can see significant improvements every time we modify the model.

Model	RMSE
Naive model	1.059933
Add bias effects	0.864962
Regularization	0.864372

We finally use our algorithm to predict ratings of **validation** set and the RMSE value is **0.864568**, which is smaller than 0.8649.

Conclusion

To create a movie recommendation system, we start with finding the mean rating value for all the data entries. We then study bias effect of users, movies, dates and genres and add these effects to the naïve model. To get a better result, we use regularization to modify the effect of each object. The best tuning parameter of regularization is chosen by using full-cross validation. In this project, the root mean squared error (RMSE) is used to compare the predicted results and true rating values. Finally, we predict ratings in the **validation** set and RMSE is **0.864568** (< 0.8649).

The effect of each object in our model is studies independently. Our recommendation system can make further improvements by analyzing the correlations between different objects, such as analyzing the correlation between users and genres: whether a group of users tend to give similar scores for a type of genres and so on.