Movie Recommendation System

Capstone Project of

Ezio Tarquilio

7/15/2021

Overview

This short thesis is my attempt to improve the **Movie Recommendation System** studied in the course textbook. To the model based on **movie and user effect with regularization**, I added the genre effect. I tested two ways to add the genre effect: one based on the sum of the effects of single genres under which a movie falls; the other based on the combination of genre effects defined as a category.

I will use the full MovieLens dataset divided into the **edx set** (90% in size) and **validation set** (remaining 10%). The overall size of the dataset is about 10 million observations and has six columns: userId, movieId, ratings, timestamp, title, and genres. To create those sets, run the script and follow instructions inside before running any other code: **./scripts/Create_edx&validation_sets_from_dat_files.R**.

Methods

I want to improve the **Movie Recommendation System** model analyzed in the textbook, considering the genre effect. Once assumed an improved model, I will first estimate the new parameters and train the algorithm using the edx set. Then I will predict and assess the results using the validation set.

Best results coincide with lower RMSE defined as:

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

where

$y_{u,i}$ is the actual rating for movie i given by user u;

$\hat{y}_{u,i}$ is the prediction for the same observation and N the total number of observations.

Before going on, let's load packages, data and define the RMSE function I will need next.

```
if(!require(tidyverse))
  install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret))
  install.packages("caret", repos = "http://cran.us.r-project.org")

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

load("./data/edx&validation_sets.RData")
```

The initial model

The textbook model I will start from, is movie and user effects plus regularization

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

where

- $Y_{u,i}$ is the rating prediction for movie i given by user u;
- μ is true average;
- $b_i$is the movie effect;
- b_u is the user effect;
- $\varepsilon_{u,i}$ is a random error from a distribution centered at 0;

and the estimates are

- $\hat{\mu} = \frac{1}{N} \sum_{i,u}^{N} y_{i,u}$ with N num of overall ratings (size of edx set)
- $\hat{b_i}(\lambda) = \frac{1}{\lambda + N_i} \sum_{u}^{N_i} (y_u - \hat{\mu})$ with N_i num of ratings for each movie i
- $\hat{b_u}(\lambda) = \frac{1}{\lambda + N_u} \sum_{i}^{N_u} (y_i - \hat{\mu} - \hat{b_i})$ with N_u num of ratings of each user u

where λ is the tuning parameter to choose with cross-validation.

Tuning parameters should never use the **validation set**, so I have to split the **edx set** into **train set** and **test set**.

```
create_train_and_test_sets <- function(seed=1993){
  ## Create train and test sets from edx set and assign
  ## them to global variables train_set & test_set

  set.seed(seed, sample.kind = "Rounding")

  test_index <-
    createDataPartition(y = edx$rating, times = 1, p = 0.2, list = FALSE)
  train_set <<- edx[-test_index]
  test_set <<- edx[test_index]

  test_set <<- test_set %>%
    semi_join(train_set, by = 'movieId') %>%
    semi_join(train_set, by = 'userId')
}

create_train_and_test_sets(1861)
```

Now I am ready to choose the best λ

```
mu_train <- mean(train_set$rating)
lambdas <- seq(4.5, 5.5, 0.1)

# tuning of lambda
rmses <- sapply(lambdas, function(l){

  bi_df <- train_set %>%
    group_by(movieId) %>%
    summarise(b_i = sum(rating - mu_train)/(n()+l))

  bu_df <- train_set %>%
    left_join(bi_df, by = 'movieId') %>%
    group_by(userId) %>%
    summarise(b_u = sum(rating - b_i - mu_train)/(n()+l))

  predictions <- test_set %>%
    left_join(bi_df, by = 'movieId') %>%
    left_join(bu_df, by = 'userId') %>%
    mutate(pred = mu_train + b_i + b_u) %>%
    pull(pred)

  return(RMSE(predictions, test_set$rating))
})

best_lambda <- lambdas[which.min(rmses)]
```

With the best-found $\lambda = 5$, I can estimate the parameters, make the predictions, and finally evaluate the RMSE.

```r
mu_hat <- mean(edx$rating)

# penalized least squares estimate of movie effect b_is
penalized_bi_df <- edx %>%
  group_by(movieId) %>%
  summarise(b_i = sum(rating - mu_hat)/(n()+best_lambda))

# penalized least squares estimate of user effect b_us
penalized_bu_df <- edx %>%
  left_join(penalized_bi_df, by='movieId') %>%
  group_by(userId) %>%
  summarise(b_u = sum(rating - mu_hat - b_i)/(n()+best_lambda))

# prediction and assessment
predictions <- validation %>%
  left_join(penalized_bi_df, by = 'movieId') %>%
  left_join(penalized_bu_df, by = 'userId') %>%
  mutate(pred = mu_hat + b_i + b_u) %>%
  pull(pred)

rmse_starting_point <- RMSE(predictions, validation$rating)
```

Results

I found an $RMSE$ = 0.8648177 to improve.

The genre effect as sum of the effects of single genres

I will try to improve the previous model by adding the sum of the effects of single genres. MovieLens data has a genres column that is a combination of several genres under which a movie falls. The new model is

$$Y_{u,i} = \mu + b_i + b_u + \sum_{k=1}^{K} x_{i,k} b_k + \varepsilon_{u,i}$$

where

$$x_{i,k} = \begin{cases} 1, & \text{if movie } i \text{ is genre } k \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

b_k is the single genre effect

I will estimate the above parameter like this

$$\hat{b_k} = \frac{1}{N_k} \sum_{j=1}^{N_k} (y_j - \hat{\mu} - \hat{b_i} - \hat{b_u}) \text{ with } N_k \text{ n. of rated movies falling under genre } k$$

The first step is to calculate every single genre effect.

```r
# vector of all genres-combinations from edx set
genres_combinations <- edx %>%
  group_by(genres) %>%
  summarise(genres = first(genres)) %>%
  pull(genres)

# split genres-combinations and collect all single genres in a named vector
genres_vec <- NULL
for (gc in genres_combinations) {
  # remove 'no geners listed'
  if (gc != '(no genres listed)') {
    a <- unlist(strsplit(gc, '|', fixed = TRUE))  # split
    genres_vec <- unique(append(genres_vec, a))   # collect
  }
}

# genre effects and n. of observations in a named matrix
genre_effects_matrix <- sapply(genres_vec, function(g){
  pattern <- paste('^.*', g, '.*?', sep ='')

  # filter edx entries having the given genre
  edx %>%
    filter(grepl(pattern, edx$genres, perl = T)) %>%
    left_join(penalized_bi_df, by = "movieId") %>%
    left_join(penalized_bu_df, by = "userId") %>%
    summarize(beta_k = mean(rating - mu_hat - b_i - b_u), n =n()) %>%
    unlist()
})
```

Having the singles genre effect in the vector **genre_effects_matrix['beta_k',]**, I define the function that makes the sum based on the genre column of the MovieLens dataset

```r
sum_genre_effects <- function(genres_combination, genre_effects_vec){
  ## Sum genre effects for a given movie, take in input
  ## 1. movie genres combination; 2. all single genre effects vector

  sapply(genres_combination, function(gc){
    if (gc == '(no genres listed)') {return(0)
    } else {
      a <- strsplit(gc, '|', fixed = TRUE)
      a <- unlist(a)
      a <- genre_effects_vec[a]
      return(sum(a))
    }})
}
```

Now I can make predictions and evaluate the RMSE with the new model

```
predictions <- validation %>%
  left_join(penalized_bi_df, by='movieId') %>%
  left_join(penalized_bu_df, by='userId') %>%
  mutate(pred = mu_hat + b_i + b_u +
           sum_genre_effects(genres, genre_effects_matrix['beta_k',])) %>%
  pull(pred)

rmse_genre_sum_effect <- RMSE(predictions, validation$rating)
```

Results

I got a better result, being the previous $RMSE$ = 0.8648177 and the current $RMSE$ = 0.8647099.

Can regularization improve this result?

As we know from the textbook, regularization penalizes those effects whose sample sizes are small because their estimates are less precise. In this case, regularization makes no sense because all samples are large.

Previously together with the singles genre effect, I collected the sample sizes in the vector

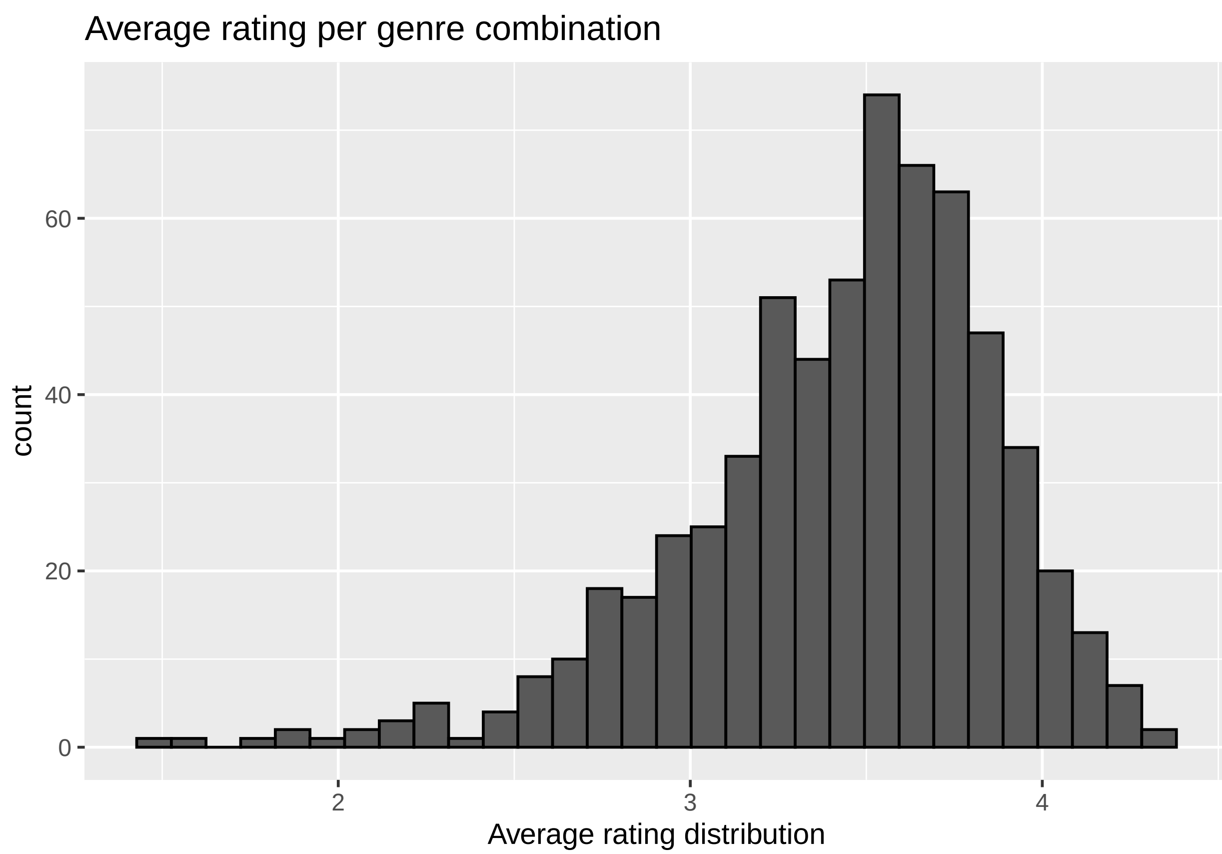
genre_effects_matrix['n',]:

Action: 2560545, Adventure: 1908892, Animation: 467168, Children: 737994, Comedy: 3540930, Fantasy: 925637, IMAX: 8181, Sci-Fi: 1341183, Drama: 3910127, Horror: 691485, Mystery: 568332, Romance: 1712100, Thriller: 2325899, Crime: 1327715, War: 511147, Western: 189394, Musical: 433080, Documentary: 93066, Film-Noir: 118541

The genre effect as genre-combination effect

As suggested in an exercise from the textbook (n. 8 chp 33.8), there is considerable variability across genres-combinations as shown by the following plot some are very disliked and others very popular.

```
# histogram plot
edx %>% group_by(genres) %>%
  filter(n()>100) %>%
  summarise(genre_avg_rating = mean(rating)) %>%
  ggplot(aes(genre_avg_rating)) + geom_histogram(bins=30, color="black") +
  labs(title = "Average rating per genre combination",
       x = "Average rating distribution")
```



So my idea is to fit a new model that add genres-combinations defined as a category

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

where

$\hat{b_z} = \frac{1}{N_z} \sum_{j=1}^{N_z} (y_j - \hat{\mu} - \hat{b_i} - \hat{b_u})$ with N_z n. of rated movies falling under genres-combinations z

making prediction and evaluating the RMSE

```
bz_df <- edx %>%
  left_join(penalized_bi_df, by = "movieId") %>%
  left_join(penalized_bu_df, by = "userId") %>%
  group_by(genres) %>%
  summarise(b_z = mean(rating - mu_hat - b_i - b_u))

predictions <- validation %>%
  left_join(penalized_bi_df, by='movieId') %>%
  left_join(penalized_bu_df, by='userId') %>%
  left_join(bz_df, by='genres') %>%
  mutate(pred = mu_hat + b_i + b_u + b_z) %>%
  pull(pred)

rmse_genre_combo_effect <- RMSE(predictions, validation$rating)
```

Results

I got an even better result, being the previous $RMSE$ = 0.8647099 and the current $RMSE$ = 0.8644509.

Can regularization improve this result?

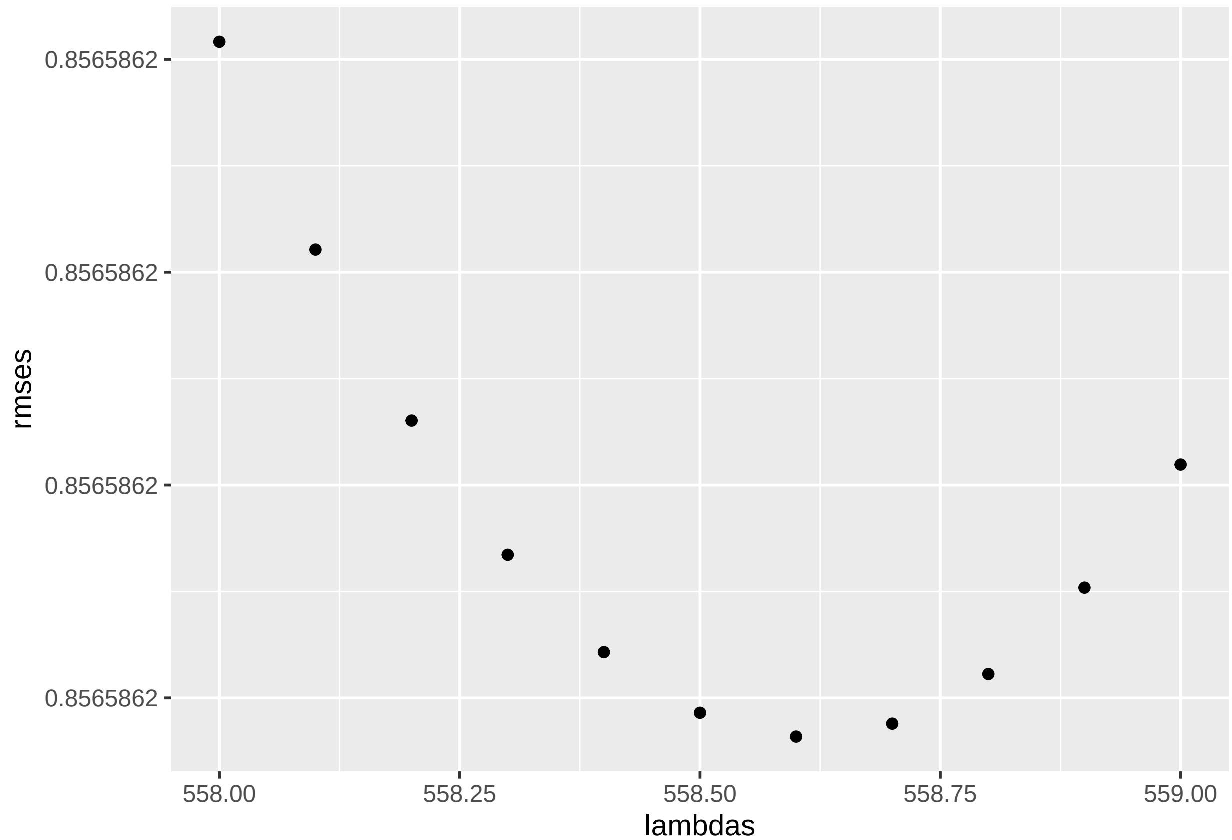
I note several genres-combinations categories with very few movies (observations) and several with many movies.

```
# n.of genres-combinations with more then 1000 movies
a <- edx %>%
  group_by(genres) %>%
  summarise(n = n()) %>%
  filter(n > 1000) %>%
  pull(n) %>% length()

# n.of genres-combinations with less then 100 movies
b <- edx %>%
  group_by(genres) %>%
  summarise(n = n()) %>%
  filter(n < 100) %>%
  pull(n) %>%
  length()
```

As shown above, there are 167 genres-combinations with a sample size smaller than 100 observations compared to 444 genres-combinations with a sample size greater than 1000 observations. Penalizing those small-size categories could make sense, so I will use cross-validation to choose the best lambda.

```
#cross-validation to choose lambda
lambdas <- seq(558, 559, 0.1)

rmses <- sapply(lambdas, function(l){

  bk_df <- train_set %>%
    left_join(penalized_bi_df, by = "movieId") %>%
    left_join(penalized_bu_df, by = "userId") %>%
    group_by(genres) %>%
    summarise(b_k = sum(rating - mu_train - b_i - b_u)/(n()+l))

  predictions <- test_set %>%
    left_join(penalized_bi_df, by='movieId') %>%
    left_join(penalized_bu_df, by='userId') %>%
    left_join(bk_df, by = 'genres') %>%
    mutate(pred = mu_train + b_i + b_u + b_k) %>%
    pull(pred)

  return(RMSE(predictions, test_set$rating))
})
```

```r
qplot(lambdas, rmses)
```




```r
best_lambda <- lambdas[which.min(rmses)]
```

making prediction and evaluating the RMSE

```r
# penalized least squares estimate of bk_df
penalized_bz_df <- edx %>%
  left_join(penalized_bi_df, by = "movieId") %>%
  left_join(penalized_bu_df, by = "userId") %>%
  group_by(genres) %>%
  summarise(b_z = sum(rating - mu_hat - b_i - b_u)/(n()+best_lambda))

# prediction and assessment
predictions <- validation %>%
  left_join(penalized_bi_df, by = 'movieId') %>%
  left_join(penalized_bu_df, by = 'userId') %>%
  left_join(penalized_bz_df, by='genres') %>%
  mutate(pred = mu_hat + b_i + b_u + b_z) %>%
  pull(pred)
```

```r
rmse_genre_combo_penalized_effect <- RMSE(predictions, validation$rating)
```

I got a worse result, being the previous $RMSE$ = 0.8644509 and the current $RMSE$ = 0.8644672.

In this case, regularization makes sense but does not improve the result.

Conclusion

At the end of this study, I got the best improvement adding to the initial model the genre effect as a combination of genre effects defined like a category. Regularization applied to the genres-combinations effect does not further improve the result.

Starting from an $RMSE$ = 0.8648177, my best improvement is $RMSE$ = 0.8644509.