MovieLens Project HarvardX PH125.9x

Muzukhona Magagula

13 June 2020

MovieLens Introduction

The purpose of the project is to explore different skills which learned on the series of courses which were undertaken. The task main

task was to analyse movielens dataset which contains 10000054 rows, 10677 movies, 797 genres and 69878 users.Using penealized least squares approach i was able to calculate

the final RMSE is 0.8252.

#Step by step guide

```
##########################################################
# Create edx set, validation set, and submission file
##########################################################

  # Note: this process could take a couple of minutes

  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")

  # MovieLens 10M dataset:
  # https://grouplens.org/datasets/movielens/10m/
  # http://files.grouplens.org/datasets/movielens/ml-10m.zip

  #To speed up data loading, the final result was already saved as 'movielens.csc'
  step <- 'load_data'#new_analysis
```

```
  if (step == 'new_analysis') {


    dl <- tempfile()

    download.file("http://files.grouplens.org/datasets/movielens/ml-10m.zip", dl)


    ratings <- read.table(text = gsub("::", "\t", readLines(unzip(dl, "ml-
10M100K/ratings.dat"))),
                          col.names = c("userId", "movieId", "rating", "timestamp"))


    movies <- str_split_fixed(readLines(unzip(dl, "ml-10M100K/movies.dat")), "\\::", 3)

    colnames(movies) <- c("movieId", "title", "genres")

    movies <- as.data.frame(movies) %>% mutate(movieId =
as.numeric(levels(movieId))[movieId],
                                               title = as.character(title),

                                               genres = as.character(genres))


    movielens <- left_join(ratings, movies, by = "movieId")


    #Shortcut for testing purposes:

  } else {

    movielens <- read.csv("ml-10M100K/movielens.csv", row.names = 1)

  }


  # Validation set will be 10% of MovieLens data


  set.seed(1)

  test_index <- createDataPartition(y = movielens$rating, times = 1, p = 0.1, list = FALSE)

  edx <- movielens[-test_index,]

  temp <- movielens[test_index,]
```

```
# Make sure userId and movieId in validation set are also in edx set

validation <- temp %>%
  semi_join(edx, by = "movieId") %>%
  semi_join(edx, by = "userId")

# Add rows removed from validation set back into edx set

removed <- anti_join(temp, validation)
edx <- rbind(edx, removed)

# Learners will develop their algorithms on the edx set
# For grading, learners will run algorithm on validation set to generate ratings

validation <- validation %>% select(-rating)
```

Data Analysis based on Capestone QUIZ

```
#Q1.How many rows and columns are there in the edx dataset?
nrow(edx)
ncol(edx)
#Answer
[1]9000055
[1]6
#Q2.How many zeros and threes were given in the edx dataset?
sum(edx$rating == 0)
edx$rating == 3)
#Answer
[1]0
[1]2121240
#Q3.How many different movies are in the edx dataset?
```

```
edx %>% summarize(n_movies = n_distinct(movieId))
#Answer
[1]10677
#Q4.How many different users are in the edx dataset?
edx %>% summarize(n_users = n_distinct(userId))
#Answer
[1]69878
#Q5.How many movie ratings are in cear of the following genres in the edx dataset?
 drama <- edx %>% filter(str_detect(genres,"Drama"))
 comedy <- edx %>% filter(str_detect(genres,"Comedy"))
 thriller <- edx %>% filter(str_detect(genres,"Thriller"))
 romance <- edx %>% filter(str_detect(genres,"Romance"))

 nrow(drama)
 nrow(comedy)
 nrow(thriller)
 nrow(romance)
#Answer
[1]4151718
[1]2962038
[1]1485456
[1]1312948

#Q6.Which movie has the greatest number of ratings?
edx %>% group_by(title) %>% summarise(number = n()) %>%
 arrange(desc(number))
#Answer
[1]Pulb Fiction
#Q7.What are the five most given ratings in order from most to least?
head(sort(-table(edx$rating)),5)
```

```
[1] 4   3   5   3.5  2
 2588430  2121240  1390114  791624  711422
#Q8.True or False: In general, half star ratings are less common than whole star ratings
(e.g., there are fewer ratings of 3.5 than there are ratings of 3 or 4, etc.).
 table(edx$rating)
 #Answer
 [1]True

 MovieLens Data Analysis

 str(movielens)

 #The movielens dataset has more than 10 million ratings with columns; userId,
movieId,rating, timestamp,title and genre.
 hist(movielens$rating,
    col = "#2E9FDF")

  summary(movielens$rating)

  Min.  1st Qu. Median    Mean  3rd Qu.
0.500  3.000  4.000  3.512  4.000

#Ratings range from 0.5 to 5.0

 movielens$year <-
as.numeric(substr(as.character(movielens$title),nchar(as.character(movielens$title))-
4,nchar(as.character(movielens$title))-1))

 plot(table(movielens$year),
   col = "#2E9FDF")
```

```
#As years elapse the ratings decrease this shows most recent movies are the most high rated than old ones
 avg_ratings <- movielens %>% group_by(year) %>% summarise(avg_rating = mean(rating))
 plot(avg_ratings,
    col = "#2E9FDF")
#Aged have more volatile ratings, this is dipicted by lower frequence of movieratings

# Results

#Calculate the RMSE using penealized least squares approach

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

#Choose the tuning value
lambdas <- seq(0,5,.5)
rmses <- sapply(lambdas, function(l){
  mu <- mean(edx_with_title_dates$rating)

  b_i <- edx_with_title_dates %>%
    group_by(movieId) %>%
    summarize(b_i = sum(rating - mu)/(n() + l))

  b_u <- edx_with_title_dates %>%
    left_join(b_i, by='movieId') %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_i - mu)/(n() +l))

  predicted_ratings <- edx_with_title_dates %>%
```

```
    left_join(b_i, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    mutate(pred = mu + b_i +  b_u) %>% .$pred

  return(RMSE(predicted_ratings, edx_with_title_dates$rating))
})

qplot(lambdas, rmses)



lambdas[which.min(rmses)]
```

```
[1] 0.5
```

```
#Predictions will be done using this value 0.5

#Validation data
lambda <- 0.5

pred_y_lse <- sapply(lambda,function(l){


  mu <- mean(edx$rating)


  b_i <- edx %>%
    group_by(movieId) %>%
    summarize(b_i = sum(rating - mu)/(n()+l))
```

```
  b_u <- edx %>%
    left_join(b_i, by="movieId") %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_i - mu)/(n()+l))



  predicted_ratings <-
    validation %>%
    left_join(b_i, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    mutate(pred = mu + b_i + b_u) %>%
    .$pred #validation

  return(predicted_ratings)

})

write.csv(validation %>% select(userId, movieId) %>% mutate(rating = pred_y_lse),
          "submission.csv", na = "", row.names=FALSE)
```

Conclusion

The project has brought not only excitement but insight of the whole course "Professional Data Science". A significant skill was attained while applying all skills learned.