movieLens Project

Silpa Velagapudi

November 30, 2019

Downloading the data file

```r
#################################
# Create edx set, validation set
#################################

# Note: this process could take a couple of minutes

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

```
## Loading required package: tidyverse

## Warning: package 'tidyverse' was built under R version 3.5.3

## -- Attaching packages ------------------------------------------------------------------------

## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## Warning: package 'ggplot2' was built under R version 3.5.3

## Warning: package 'tibble' was built under R version 3.5.3

## Warning: package 'tidyr' was built under R version 3.5.3

## Warning: package 'readr' was built under R version 3.5.3

## Warning: package 'purrr' was built under R version 3.5.3

## Warning: package 'dplyr' was built under R version 3.5.3

## Warning: package 'stringr' was built under R version 3.5.3

## Warning: package 'forcats' was built under R version 3.5.3

## -- Conflicts ---------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

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

```
## Loading required package: caret

## Warning: package 'caret' was built under R version 3.5.3

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

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

```
## Loading required package: data.table

## Warning: package 'data.table' was built under R version 3.5.3

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose
```

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

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

ratings <- fread(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")
```

Cleaning data

```r
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")

# Validation set will be 10% of MovieLens data

set.seed(1)#, sample.kind="Rounding")
# if using R 3.5 or earlier, use `set.seed(1)` instead
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
#Converting timestamp to Year, month, date, week, hour, minutes, sec for the validation set with differ
validation <- temp %>%
     semi_join(edx, by = "movieId") %>%
     semi_join(edx, by = "userId") %>% mutate(n_week=format(weekdays(as.Date(format(as.POSIXct(timestam
  mutate(n_hour=format(as.POSIXct(timestamp,origin="1970-01-01"),"%H")) %>%
  mutate(n_min=format(as.POSIXct(timestamp,origin="1970-01-01"),"%M")) %>%
  mutate(n_secs=format(as.POSIXct(timestamp,origin="1970-01-01"),"%OS"))

# Add rows removed from validation set back into edx set
```

```r
removed <- anti_join(temp, validation)
```

```
## Joining, by = c("userId", "movieId", "rating", "timestamp", "title", "genres")
```

```r
edx <- rbind(edx, removed)

rm(dl, ratings, movies, test_index, temp, movielens, removed)
```

Using subset of edx data for analysing across different algorithms

```r
set.seed(1)
#subsetting the data using filter with row greater than 25000
edx_subset <- edx %>% group_by(movieId)  %>%
     filter(n()>25000)

#viewing the subset data
head(edx_subset)
```

```
## # A tibble: 6 x 6
## # Groups:   movieId [4]
##   userId movieId rating  timestamp title                          genres
##    <int>   <dbl>  <dbl>      <int> <chr>                          <chr>
## 1      1     356    5    838983653 Forrest Gump (1994)            Comedy|Drama|~
## 2      1     589    5    838983778 Terminator 2: Judgment Day (1~ Action|Sci-Fi
## 3      2     110    5    868245777 Braveheart (1995)              Action|Drama|~
## 4      2     260    5    868244562 Star Wars: Episode IV - A New~ Action|Advent~
## 5      3     110    4.5 1136075500 Braveheart (1995)              Action|Drama|~
## 6      4     110    5    844416866 Braveheart (1995)              Action|Drama|~
```

GLM method results:

```r
#using glm method for analysis
train_glm <- train(rating ~  movieId , method = "glm", data = edx_subset)
#created RMSE function for using it multiple times
RMSE <- function(true_ratings, predicted_ratings){
     sqrt(mean((true_ratings - predicted_ratings)^2))
}
#finding mean of the rating
mu_hat <- mean(edx_subset$rating)
#Finding RMSE result for the glm method results
naive_rmse <- RMSE(validation$rating, mu_hat)
#loading the RMSE result of glm method in table for future analysis
rmse_results <- data_frame(method="glm",
                                     RMSE = naive_rmse )
```

```
## Warning: `data_frame()` is deprecated, use `tibble()`.
## This warning is displayed once per session.
```

```r
#viewing the RMSE results
rmse_results
```

```
## # A tibble: 1 x 2
##   method  RMSE
##   <chr>  <dbl>
## 1 glm     1.20
```

Rpart method results:

```
#using rpart method for analysis
train_rpart <- train(rating ~ ., method = "rpart", data = edx_subset, tuneGrid=data.frame(cp= seq(0, 0.
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```
#finding mean of the rating
mu_hat <- mean(edx_subset$rating)
#Finding RMSE result for the rpart method results
rpart_rmse <- RMSE(validation$rating, mu_hat)
#loading the RMSE result of glm method in table for future analysis
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="rpart",
                                     RMSE = rpart_rmse ))
#viewing the RMSE results
rmse_results
```

```
## # A tibble: 2 x 2
##   method  RMSE
##   <chr>  <dbl>
## 1 glm     1.20
## 2 rpart   1.20
```

RF method results:

```
#using rf method for analysis
train_rf <- train(rating ~ ., method = "rf", data = edx_subset, tuneGrid=data.frame(mtry= seq(1, 7, 1))
#finding mean of the rating
mu_hat <- mean(edx_subset$rating)
#Finding RMSE result for the rpart method results
rf_rmse <- RMSE(validation$rating, mu_hat)
#loading the RMSE result of glm method in table for future analysis
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="rf",
                                     RMSE = rf_rmse ))
#viewing the RMSE results
rmse_results
```

```
## # A tibble: 3 x 2
##   method  RMSE
##   <chr>  <dbl>
## 1 glm     1.20
## 2 rpart   1.20
## 3 rf      1.20
```

Converting the timestamp into different columns for Year, Month, Date, Day, Hours, Minutes, Seconds. Reducing the number of records for pdf file generations. Results might not be the same as that of orirignal data.

```
#converting the timestamp to different columns of year, month, date, week, hour, minutes, seconds.
 edx_updated <- edx  %>% mutate(n_year=format(as.POSIXct(timestamp,origin="1970-01-01"),"%Y")) %>% muta
  mutate(n_hour=format(as.POSIXct(timestamp,origin="1970-01-01"),"%H")) %>%
   mutate(n_week=format(weekdays(as.Date(format(as.POSIXct(timestamp,origin="1970-01-01"),"%Y-%m-%d")))
  #Please add n_day while running the script and remove it if pdf file needs to be generated
  select(userId,movieId,rating,n_year,n_month,n_week,n_day,n_hour)
head(edx_updated)
```

```
##   userId movieId rating n_year n_month    n_week n_day n_hour
## 1      1     122      5   1996      08 Friday       02     07
## 2      1     185      5   1996      08 Friday       02     06
## 3      1     292      5   1996      08 Friday       02     06
## 4      1     316      5   1996      08 Friday       02     06
## 5      1     329      5   1996      08 Friday       02     06
## 6      1     355      5   1996      08 Friday       02     07
```

Finding the relation between the movie biased and rating

```
#calculating the overall mean of rating
mu <- mean(edx_updated$rating)
#checking the ratings and the movieId dependencies
     edx_updated %>%
          group_by(movieId) %>%
          summarize(b_i = mean(rating - mu)) %>% slice(1:10)
```

```
## # A tibble: 10 x 2
##    movieId     b_i
##      <dbl>   <dbl>
##  1       1  0.415
##  2       2 -0.307
##  3       3 -0.365
##  4       4 -0.648
##  5       5 -0.444
##  6       6  0.303
##  7       7 -0.154
##  8       8 -0.378
##  9       9 -0.515
## 10      10 -0.0866
```

Finding the relation between rating and user biased

```
#checking the ratings and the userId dependencies
     edx_updated %>%
          group_by(userId) %>%
          summarize(b_u = mean(rating - mu)) %>% slice(1:10)
```

```
## # A tibble: 10 x 2
##    userId    b_u
##     <int>  <dbl>
##  1      1  1.49
##  2      2 -0.218
##  3      3  0.423
##  4      4  0.545
##  5      5  0.406
##  6      6  0.436
##  7      7  0.352
##  8      8 -0.126
##  9      9  0.535
## 10     10  0.318
```

Finding the relation between rating and year rating was provided

```
#checking the ratings and the userId dependencies
     edx_updated %>%
          group_by(n_year) %>%
          summarize(b_y = mean(rating - mu)) %>% slice(1:10)
```

```
## # A tibble: 10 x 2
##    n_year      b_y
##    <chr>     <dbl>
##  1 1995    0.488
##  2 1996    0.0328
##  3 1997    0.0734
##  4 1998   -0.00639
##  5 1999    0.105
##  6 2000    0.0657
##  7 2001    0.0237
##  8 2002   -0.0401
##  9 2003   -0.0400
## 10 2004   -0.0872
```

Finding the best optimum value for the RMSE

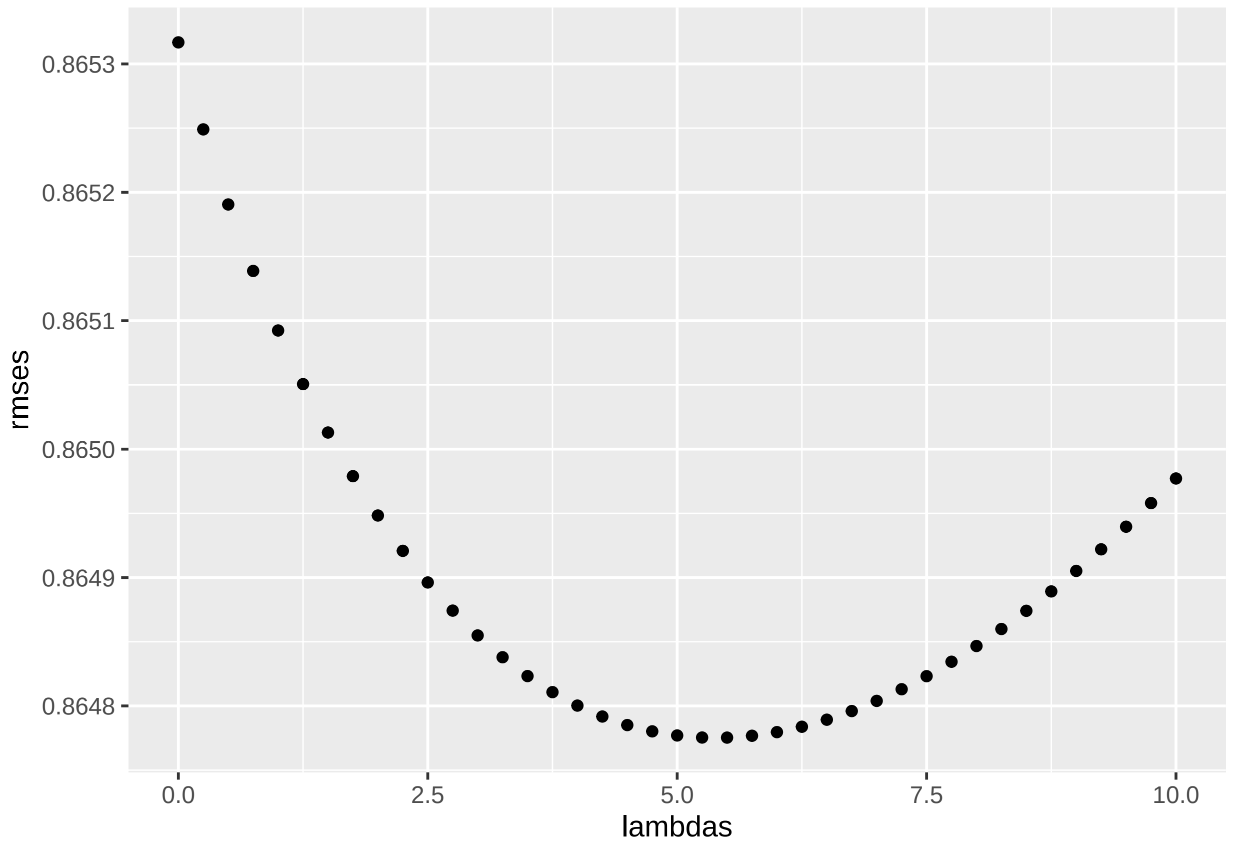
```
#RMSE function for caalculating different lambdas.

RMSE <- function(true_ratings, predicted_ratings){
     sqrt(mean((true_ratings - predicted_ratings)^2))
}
#using different lambdas for our analysis
lambdas <- seq(0, 10, 0.25)
#l<-5.5
rmses <- sapply(lambdas, function(l){
  # finding mean of the rating
mu <- mean(edx_updated$rating)
# Finding the movie biased
     b_i <- edx_updated %>%
          group_by(movieId) %>%
          summarize(b_i = sum(rating - mu)/(n()+l))
     # Finding the user biased
     b_u <- edx_updated %>%
          left_join(b_i, by="movieId") %>%
          group_by(userId) %>%
          summarize(b_u = sum(rating - b_i - mu)/(n()+l))
     #Finding the year biased
     b_y <- edx_updated %>%
          left_join(b_i, by="movieId") %>%
          left_join(b_u,by="userId") %>%
          group_by(n_year) %>%
          summarize(b_y = sum(rating - b_i - mu - b_u)/(n()+l))
     #finding the month biased
      b_m <- edx_updated %>%
          left_join(b_i, by="movieId") %>%
          left_join(b_u,by="userId") %>%
          left_join(b_y,by="n_year") %>% group_by(n_month) %>%
          summarize(b_m = sum(rating - b_i - mu - b_u - b_y)/(n()+l))
      #Finding the Week biased
      b_w <- edx_updated %>%
          left_join(b_i, by="movieId") %>%
          left_join(b_u,by="userId") %>%
          left_join(b_y,by="n_year") %>%
          left_join(b_m,by="n_month") %>%
          group_by(n_week) %>%
          summarize(b_w = sum(rating - b_i - mu - b_u - b_y - b_m)/(n()+l))
      #Finding the date biased
      b_d <- edx_updated %>%
          left_join(b_i, by="movieId") %>%
          left_join(b_u,by="userId") %>%
          left_join(b_y,by="n_year") %>%
          left_join(b_m,by="n_month") %>%
          left_join(b_w,by="n_week") %>%
          group_by(n_day) %>%
          summarize(b_d = sum(rating - b_i - mu - b_u - b_y - b_m - b_w)/(n()+l))
      #Finding the hour biased. Please add this while running the script and remote this code while run
      b_h <- edx_updated %>%
          left_join(b_i, by="movieId") %>%
          left_join(b_u,by="userId") %>%
          left_join(b_y,by="n_year") %>%
          left_join(b_m,by="n_month") %>%
          left_join(b_w,by="n_week") %>%
          left_join(b_d, by="n_day") %>%
          group_by(n_hour) %>%
          summarize(b_h = sum(rating - b_i - mu - b_u - b_y - b_m - b_w - b_d)/(n()+l))
      #Finding the predicted rating based on Year, Month, Movie id, User id, date, week
     predicted_ratings <-
          validation %>%
          left_join(b_i, by = "movieId") %>%
          left_join(b_u, by = "userId") %>%
       left_join(b_y, by = "n_year") %>%
       left_join(b_m, by = "n_month") %>%
       left_join(b_w, by = "n_week") %>%
       left_join(b_d,by="n_day") %>%
       left_join(b_h, by="n_hour") %>%
       #please removee b_h and b_d. Above left_join for hour while running the pdf file and use it whil
          mutate(pred = mu + b_i + b_u + b_y + b_m + b_w + b_d + b_h  ) %>%
          .$pred
# Finding the RMSE for the predicted rating
  return(RMSE(predicted_ratings, validation$rating))

})
```

Plotting lambas and rmse values

```
#creating the plot with the lamdas and RMSEs
qplot(lambdas, rmses)
```



Finding minimum rmse value

```
#find the best lamda value for the RMSE values
lambda <- lambdas[which.min(rmses)]
#displaying min lambda value
lambda
```

```
## [1] 5.5
```

Adding the min RMSE value and showing all the results we have done so far

```
#Adding the table with the RMSE values which are the best
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Overall effect",
                                     RMSE = min(rmses)))

#displaying all the RMSE values calculated so far
rmse_results %>% knitr::kable()
```

method	RMSE
glm	1.2039893
rpart	1.2039893
rf	1.2039893
Overall effect	0.8647754