edx movielens capstone

Bush Daniel Kwajaffa

1/4/2020

```
################################
# 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
```

```
##  Attaching packages  tidyverse 1.3.0
```

```
##  ggplot2 3.2.1      purrr   0.3.3
##  tibble  2.1.3      dplyr   0.8.3
##  tidyr   1.0.0      stringr 1.4.0
##  readr   1.3.1      forcats 0.4.0
```

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

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

```
## Loading required package: caret
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

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

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

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

```
##
## 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")
movies <- as.data.frame(movies) %>% mutate(movieId = as.numeric(levels(movieId))[movieId], title = as.ch
movielens <- left_join(ratings, movies, by = "movieId")


# Validation set will be 10% of MovieLens data

set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```r
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)
```

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

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

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

#setting the ramdom number generato.
set.seed(1998, sample.kind = "Rounding")
```

```
## Warning in set.seed(1998, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```r
# creating a serie of test/traininh partions
test_index <- createDataPartition(y = edx$rating, times = 1, p = 0.2, list = FALSE)
train_set <- edx[-test_index,]
test_set <- edx[test_index,]

#matching the test set to train set
test_set <- test_set %>% semi_join(train_set, by = "movieId") %>% semi_join(train_set, by = "userId")

#creating RMSE function
RMSE <- function(true_ratings, predicted_ratings){
```

```
      sqrt(mean((true_ratings - predicted_ratings)^2))
}

#average of all rates and bias
mu <- mean(train_set$rating)
movie_avgs <- train_set %>% group_by(movieId) %>% summarize(b_i = mean(rating - mu))
predicted_ratings <- mu + test_set %>% left_join(movie_avgs, by='movieId') %>% pull(b_i)


# fit <- lm(rating ~ as.factor(movieId) + as.factor(userId))
user_avgs <- test_set %>% left_join(movie_avgs, by='movieId') %>%
 group_by(userId) %>% summarize(b_u = mean(rating - mu - b_i))

predicted_ratings <- test_set %>% left_join(movie_avgs, by='movieId') %>% left_join(user_avgs, by='user
  mutate(pred = mu + b_i + b_u) %>% .$pred

model_rmse <- RMSE(predicted_ratings, test_set$rating)
model_rmse
```

```
## [1] 0.8430817
```