MovieLens Recommender Report

Ragini

June 14, 2019

R Markdown

[1. Executive Summary]

This project is a movie recommendation system which has been created using MovieLens dataset. I have used the edx and validation dataset from the Google drive location https://drive.google.com/drive/folders/1IZcBBX0OmL9wu9AdzMBFUG8GoPbGQ38D (https://drive.google.com/drive/folders/1IZcBBX0OmL9wu9AdzMBFUG8GoPbGQ38D)

Goal - Through this project, I have learnt to build a model through a machine learning algorithm on a training dataset and apply the model on an unknown dataset to predict its parameters. Here, movie ratings are predicted.

Key Steps - edx data was explored and visualized. It was partitioned into train and test sets to build a linear regression model. Effect of movies and users both have been considered while predicting rating. The predicted ratings are compared with actual ratings to calculate RMSE and hence evaluate the performance of the model. This model is then applied on validation dataset and again RMSE is calculated.

[2. Analysis]

The analysis starts with loading the libraries, reading data and visualizing data.

```
############################################################
#        Movie Recommendation system (MovieLens Dataset)   #
############################################################

# Library for data operations
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

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

```
## Loading required package: tidyverse
```

```
## -- Attaching packages ------------------------------------------------------- tidyverse 1.2.1
--
```

```
## v ggplot2 3.1.1     v readr   1.3.1
## v tibble  2.1.3     v purrr   0.3.2
## v tidyr   0.8.3     v stringr 1.4.0
## v ggplot2 3.1.1     v forcats 0.4.0
```

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

```
# Library for Classification And REgression Training
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
```

```
# Reading data. Used dataset provided by instructor on Google drive as
# code of building dataset was taking too much time and not ending
# Dataset location https://drive.google.com/drive/folders/1IZcBBX0OmL9wu9AdzMBFUG8GoPbGQ38D
edx <- readRDS("C:/Ragini/MovieLens-Recommender/edx.rds")
validation <- readRDS("C:/Ragini/MovieLens-Recommender/validation.rds")

# Data exploration
edx %>% summarize(n_users = n_distinct(userId), n_movies = n_distinct(movieId))
```

```
##   n_users n_movies
## 1   69878    10677
```

```
#Data visualization
head(edx)
```

```
##   userId movieId rating timestamp                         title
## 1      1     122      5 838985046              Boomerang (1992)
## 2      1     185      5 838983525               Net, The (1995)
## 4      1     292      5 838983421               Outbreak (1995)
## 5      1     316      5 838983392               Stargate (1994)
## 6      1     329      5 838983392 Star Trek: Generations (1994)
## 7      1     355      5 838984474       Flintstones, The (1994)
##                          genres
## 1                Comedy|Romance
## 2         Action|Crime|Thriller
## 4  Action|Drama|Sci-Fi|Thriller
## 5       Action|Adventure|Sci-Fi
## 6 Action|Adventure|Drama|Sci-Fi
## 7       Children|Comedy|Fantasy
```

Building a model.

```
##Partitioning edx data to create a model
set.seed(755)
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")

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

### Building the Recommendation System using edx dataset
# lm(rating ~ as.factor(movieId) + as.factor(userId))

mu_hat <- mean(train_set$rating)
mu_hat
```

```
## [1] 3.512527
```

```
mu <- mean(train_set$rating)

#Calculating movie averages
movie_avgs <- train_set %>%
  group_by(movieId) %>%
  summarize(b_i = mean(rating - mu))

#Calculating user averages
user_avgs <- train_set %>%
  left_join(movie_avgs, by='movieId') %>%
  group_by(userId) %>%
  summarize(b_u = mean(rating - mu - b_i))

#Predicting ratings on test set
predicted_ratings <- test_set %>%
  left_join(movie_avgs, by='movieId') %>%
  left_join(user_avgs, by='userId') %>%
  mutate(pred = mu + b_i + b_u) %>%
  .$pred

model_rmse <- RMSE(predicted_ratings, test_set$rating)
rmse_results <- data_frame(method="Movie + User Effects Model on edx test set",
                           RMSE = model_rmse )
```

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

```
rmse_results %>% knitr::kable()
```

method	RMSE
Movie + User Effects Model on edx test set	0.8666408

Evaluation of the model on Validation set

```
### Validating Recommendation System on Validation set
#Using the model built in previous steps to predict rating for validation set
# lm(rating ~ as.factor(movieId) + as.factor(userId))

mu_hat_v <- mean(validation$rating)
mu_hat_v
```

```
## [1] 3.512033
```

```
mu_v <- mean(validation$rating)


movie_avgs_v <- validation %>%
  group_by(movieId) %>%
  summarize(b_i_v = mean(rating - mu_v))


user_avgs_v <- validation %>%
  left_join(movie_avgs_v, by='movieId') %>%
  group_by(userId) %>%
  summarize(b_u_v = mean(rating - mu_v - b_i_v))

predicted_ratings <- validation %>%
  left_join(movie_avgs_v, by='movieId') %>%
  left_join(user_avgs_v, by='userId') %>%
  mutate(pred = mu_v + b_i_v + b_u_v) %>%
  .$pred

model_rmse_v <- RMSE(predicted_ratings, validation$rating)
rmse_results <- bind_rows(rmse_results, data_frame(method="Movie + User Effects Model on Validat
ion Set",
                                                   RMSE = model_rmse_v ))
```

[3. Results]

Below section displays the result of the code.

```
rmse_results %>% knitr::kable()
```

method	RMSE
Movie + User Effects Model on edx test set	0.8666408
Movie + User Effects Model on Validation Set	0.8251770

[4. Conclusion]

The program resulted in a RMSE of 0.8251 which indicates good prediction performance. Hence, it can be concluded that linear regression analysis is optimal for MovieLens dataset evaluation.