MovieLens Project Report

Thomas D. Pellegrin

October 1, 2023

Abstract

We report the methods and results from the Capstone module of the HarvardX PH125.9 “Data Science” course from edx.org. A machine learning algorithm was developed using the R statistical computing environment to predict user ratings of movies against an archival dataset of historical ratings. The resulting root mean square error (RMSE) is…

Introduction

The final module of the HarvardX PH125.9 “Data Science” course from edx.org requires students to independently develop and submit a capstone project. The project’s goal is to develop a machine learning algorithm that can predict movie ratings with an RMSE lower than 0.8649.

Methods

Data preparation

The *edx* dataset was generated using R code adapted from the course instructions. First, the dataset was downloaded from the *grouplens.org* website as a 65.6 MB zipped file.

```
dl <- "ml-10M100K.zip"
if(!file.exists(dl)) {
  download.file("https://files.grouplens.org/datasets/movielens/ml-10m.zip", dl)
}
```

Next, the *ratings* and *movies* data were extracted from the zipped file and their respective data frames joined into one *movieLens* data frame.

```
movielens <- left_join(

  ratings <- read.table(
    text = gsub(
      x           = readLines(con = unzip(dl, "ml-10M100K/ratings.dat")),
      pattern     = "::",
      replacement = ";",
      fixed       = TRUE
    ),
```

```r
    sep        = ";",
    col.names  = c("userId", "movieId", "rating", "timestamp"),
    colClasses = c("integer", "integer", "numeric", "integer")
  ),

  movies <- read.table(
    text = gsub(
      x           = readLines(con = unzip(dl, "ml-10M100K/movies.dat")),
      pattern     = "::",
      replacement = ";",
      fixed       = TRUE
    ),
    sep        = ";",
    col.names  = c("movieId", "title", "genres"),
    colClasses = c("integer", "character", "character")
  ),

  by = "movieId"
)
```

```
## Warning in scan(file = file, what = what, sep = sep, quote = quote, dec = dec,
## : EOF within quoted string
```

```
## Warning in scan(file = file, what = what, sep = sep, quote = quote, dec = dec,
## : number of items read is not a multiple of the number of columns
```

Then, the dataset was partitioned into a training set (*edx*) and a test set (*final_holdout_test*). The training set was used to develop the machine learning algorithm. The test set was used to evaluate the algorithm's performance against the course's grading rubric.

```r
# Set a seed for reproducibility
set.seed(1, sample.kind = "Rounding") # if using R 3.6 or later
```

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

```r
# Partition the data. Final hold-out test set will be 10% of MovieLens data
test_index <- createDataPartition(
  y     = movielens$rating,
  times = 1,
  p     = 0.1,
  list  = FALSE
)

# Separate the test set from the edx dataset
edx <- movielens[-test_index,]
temp <- movielens[test_index,]

# Make sure userId and movieId in final hold-out test set are also in edx set
final_holdout_test <- temp %>%
  semi_join(edx, by = "movieId") %>%
  semi_join(edx, by = "userId")
```

```
# Add rows removed from final hold-out test set back into edx set
removed <- anti_join(temp, final_holdout_test)
```

```
## Joining with ‘by = join_by(userId, movieId, rating, timestamp, title, genres)‘
```

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

# Remove unnecessary objects from memory
rm(dl, ratings, movies, test_index, temp, movielens, removed)
```

Exploratory analysis

Next, the *edx* dataset was described. The data consist of individual ratings assigned by a user to a movie. There are approximately 9.10^6 observations of 6 variables, which are the unique sequential IDs of the user and movie, the rating given to the movie (on a scale of 0.0 to 5.0 in 0.5 increments), a Unix timestamp of the review, the title of the movie including its year of release between brackets, and one or more cinematographic genres associated with the movie separated by a delimiter character.

```
str(edx)
```

```
## 'data.frame':    9000055 obs. of  6 variables:
##  $ userId   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ movieId  : int  122 185 292 316 329 355 356 362 364 370 ...
##  $ rating   : num  5 5 5 5 5 5 5 5 5 5 ...
##  $ timestamp: int  838985046 838983525 838983421 838983392 838983392 838984474 838983653 838984885 83
##  $ title    : chr  "Boomerang (1992)" "Net, The (1995)" NA "Stargate (1994)" ...
##  $ genres   : chr  "Comedy|Romance" "Action|Crime|Thriller" NA "Action|Adventure|Sci-Fi" ...
```

From this description, six hypotheses were formed about the factors that might influence movie ratings:

1. How old the movie is, as measured by its year of release (i.e., older movies might be rated differently than newer ones);
2. How old the rating is, as measured by its year of assignment (i.e., older ratings might be different than newer ones);
3. How much time passed between the release of the movie and the rating, as measured by the difference between the years of both (i.e., ratings assigned closer in time to the movie release might be different than those assigned further in time);
4. The genre of the movie (i.e., movies of different genres might be rated differently);
5. The user who assigned the rating (i.e., users might have different rating habits);
6. The movie itself (i.e., individual movies might be rated differently).

To test these hypotheses, the *edx* dataset was transformed using the following R code. The *timestamp* variable was converted to a human-readable date. The *year_movie* variable was extracted from the movie title, and the latter was shortened correspondingly. The *year_dist* variable was created by calculating the absolute difference between the years of the movie release and of the rating.

```
edx <- edx %>%
    mutate(
      year_movie  = as.integer(str_sub(title, start = -5L, end = -2L)),
      year_rating = as.integer(as.POSIXlt(timestamp, origin = "1970-01-01")$year + 1900L),
      year_dist   = as.integer(abs(year_movie - year_rating)),
      title       = str_sub(title, end = -8L),
      timestamp   = NULL
    )
```

```
## Warning: There was 1 warning in ‘mutate()‘.
## i In argument: ‘year_movie = as.integer(str_sub(title, start = -5L, end =
##   -2L))‘.
## Caused by warning:
## ! NAs introduced by coercion
```

Each of these six hypotheses was then tested against the *edx* dataset.

Ratings by year of movie release

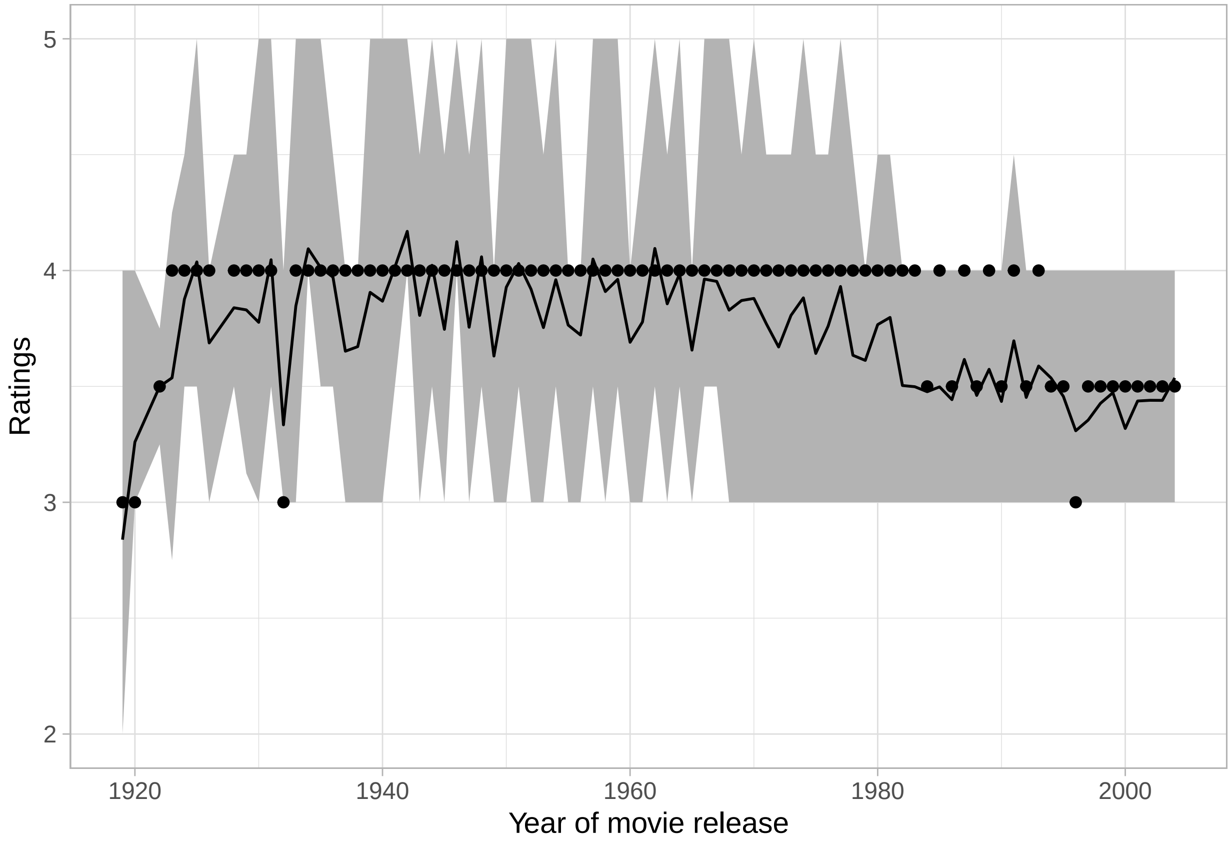
```
# Table
edx %>%
  group_by(year_movie) %>%
  summarize(
    mean_rating   = mean(rating),
    median_rating = median(rating),
    n_rating      = n()
  )
```

```
## # A tibble: 85 x 4
##    year_movie mean_rating median_rating n_rating
##         <int>       <dbl>         <dbl>    <int>
##  1       1919        2.84           3         31
##  2       1920        3.26           3         25
##  3       1922        3.5            3.5        2
##  4       1923        3.54           4         27
##  5       1924        3.88           4        162
##  6       1925        4.04           4       1007
##  7       1926        3.69           4         64
##  8       1928        3.84           4        137
##  9       1929        3.83           4        262
## 10       1930        3.78           4        660
## # i 75 more rows
```

```
# Plot
edx %>%
  group_by(year_movie) %>%
  summarize(
    mean   = mean(rating),
    median = median(rating),
    low    = quantile(rating, .25), # 25th percentile of the ratings for that year
    high   = quantile(rating, .75)  # 75th percentile of the ratings for that year
  ) %>%
  ggplot(aes(x = year_movie)) +
  geom_ribbon(
    mapping = aes(y = mean, ymin = low, ymax = high),
    fill    = "grey70"
  ) +
  geom_line(mapping = aes(y = mean)) +
  geom_point(mapping = aes(y = median)) +
  theme_light() +
  xlab("Year of movie release") +
  ylab("Ratings")
```

```
## Warning: Removed 1 row containing missing values (‘geom_line()‘).
```

```
## Warning: Removed 1 rows containing missing values (‘geom_point()‘).
```



Results

Conclusion