Code for The Social Web

M.M

13 March 2020

In order not to be bothered with rounding the numbers, set `options(digits=3)` .

Exercise 5

Data preprocessing First I extract into 11 files named "output/ratings_sortedXXX-YYY.csv" ratings newer than 01.01.2015 00:00. Then I load them into the dataset in R:

```
rating_scale = c(0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0);
sum_ratings <- function(data) {
  result_sum_ratings = c();
  for (i in rating_scale) {
    result_sum_ratings[i * 2] = length(data[data == i]);
  }
  result_sum_ratings
}


preprocess_movie_data_for_sample <- function(data) {
  c(sum_ratings(data[1:(length(data) / 2)]),
    sum_ratings(data[(length(data) / 2 + 1): length(data)]))
}

preprocess_movie_data <- function(data) {
  random_indices = sample(1:nrow(data), nrow(data) / 2);
  preprocess_sample_1 <- c(data[random_indices,2], data[-random_indices,2]);
  random_indices = sample(1:nrow(data), nrow(data) / 2);
  preprocess_sample_2 <- c(data[random_indices,2], data[-random_indices,2]);
  random_indices = sample(1:nrow(data), nrow(data) / 2);
  preprocess_sample_3 <- c(data[random_indices,2], data[-random_indices,2]);
  c(preprocess_movie_data_for_sample(data[,2]),
    preprocess_movie_data_for_sample(preprocess_sample_1),
    preprocess_movie_data_for_sample(preprocess_sample_2),
    preprocess_movie_data_for_sample(preprocess_sample_3))
}

preprocess_file_data <- function(fileName) {
  data = read.table(fileName, header=TRUE, sep=",");
  movieIds = unique(data[,1]);
  res = matrix(-1,length(movieIds),81);
  counter = 1;
  for (i in movieIds) {
    temp = subset(data,data['movieId'] == i)
```

```
    if (nrow(temp) < 50) {
      next
    }
    res[counter,] = c(i, preprocess_movie_data(temp));
    counter = counter + 1;
  }
  res = subset(res, res[,1] != -1);
  res
}
```

sum_ratings function for given ratings produces a vector of 10 values. i-th value represents number of (i/2) ratings in the given data.

preprocess_movie_data_for_sample function for given sample of ratings divides it into 2 halves. As an output vector of 2 x 10 values is produced. First 10 values describes the distribution of ratings in the first half. Next 10 values describes distribution of ratings in the second half.

preprocess_movie_data function for data consisting ratings ordered in time creates a vector of 4 x 20 values. First 20 values describe distribution of ratings in the first half and the second half. Next 3 x 20 values describes distribution of values in the first and second halves for 3 random permutation. These 3 samples are used as a control group. I used 3 samples instead of 1 to avoid unlucky pick. I expect that the median of test results for 3 samples to be >0.05 almost everytime.

preprocess_file_data function for a given fileName reads the data, extracts unique movieIds and then preprocess data from the file. It removes any movie that has less than 50 ratings from further investigation. As an output it produces matrix of size numberOfMovies x 81. Each row represents a movie that has at least >50 ratings. First column is movieId, then 4 x 20 preprocessed values as described above (4 samples). **Data testing** For each row we compute 4 times (one time for each sample) chisq_test. This test is most reliable when it comes to measure difference between 2 discrete distributions:

[linked phrase]https://stats.stackexchange.com/questions/298467/are-two-grade-distributions-significantly-different-in-r

[linked phrase]https://rcompanion.org/handbook/H_09.html?fbclid=IwAR0YsHHx7ORNq2qlIrefuW-zRsQgHvjpnXKb6u4qTLLRFeg3ab3RTufWNSQ

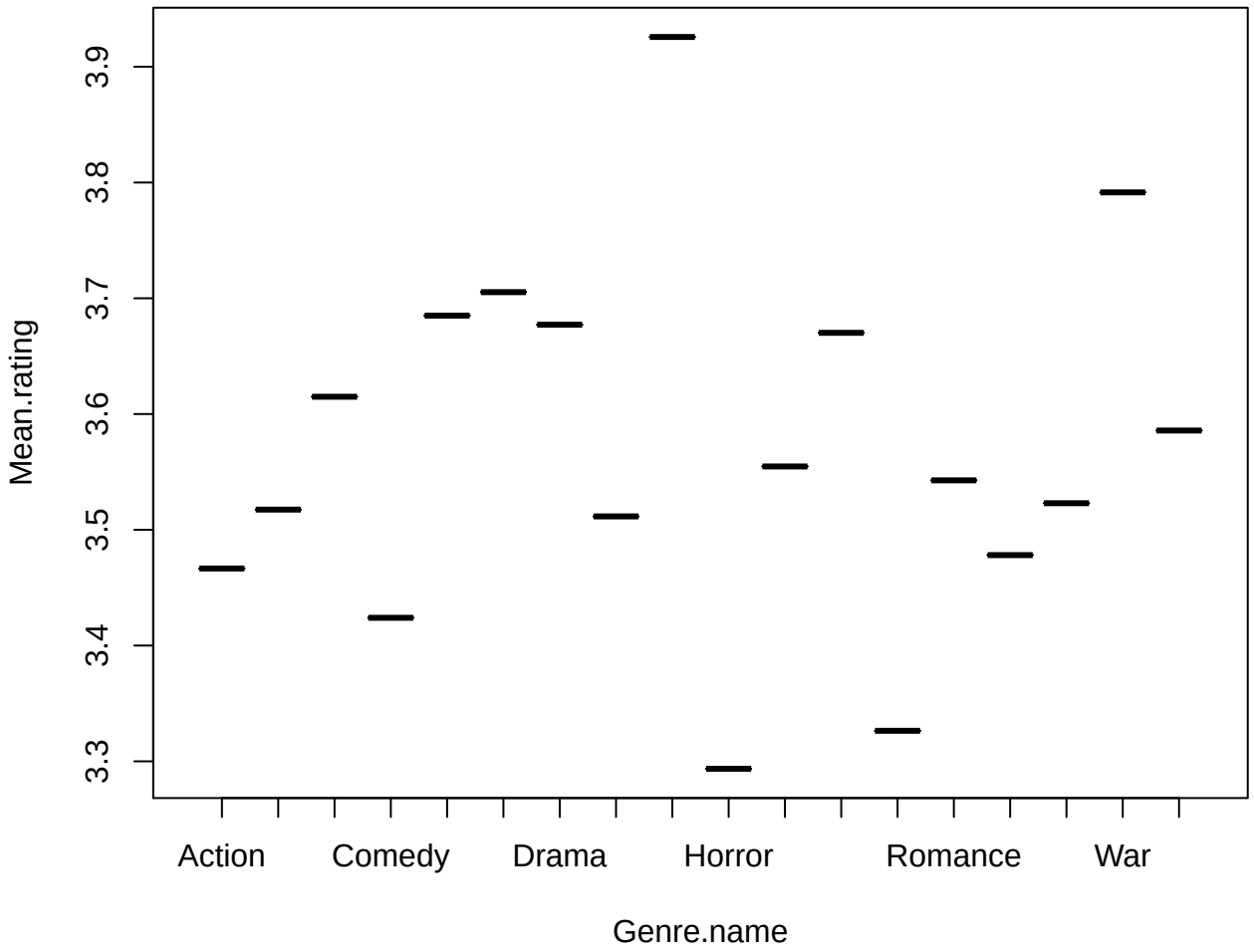
```
chiTest <- function(data, offset) {
  x <- as.table(matrix(data[(offset+1):(offset+20)], nrow=2, byrow=TRUE));
  pvalue(chisq_test(x, scores=list("Var2" = c(1,2,3,4,5,6,7,8,9,10))))
}

library(survival)
library(coin)
test_data <- function(data) {
  movieIds = unique(data[,1]);
  test_results = matrix(-1, length(movieIds), 5);
  for (i in (1:nrow(data))) {
    test_results[i,] = c(data[i,1],
                         chiTest(data[i,], 1),
                         chiTest(data[i,], 21),
                         chiTest(data[i,], 41),
                         chiTest(data[i,], 61));
  }
  test_results = subset(test_results, test_results[,1] != -1)
}
```

Data computation and save to file

```
run_tests <- function() {
  for (i in (1:11)) {
    namePrefix = "output/ratings_sorted";
    inputFilename = paste(namePrefix,
                          format(((i-1) * 20000), scientific=FALSE),
                          "-",
                          format(i*20000, scientific=FALSE),
                          "__.csv",
                          sep="");
    outputFilename = paste(namePrefix,
                           format(((i-1) * 20000), scientific=FALSE),
                           "-",
                           format((i*20000), scientific=FALSE),
                           "__output.csv",
                           sep="");
    preprocessed_data <- preprocess_file_data(inputFilename);
    tested_data <- test_data(preprocessed_data);
    write.csv(tested_data, outputFilename)
  }
}
```

Additional Info To investigate potential points and find any collinearity in our explanatory variables we take a look at plots between all the pairs of explanatory variables.

```
library(knitr)
genreMeansData = read.table("output/genres_mean_ratings.csv", header=TRUE, sep=",")
plot(genreMeansData)
```




```
kable(genreMeansData)
```

Genre.name	Mean.rating
Action	3.466592
Adventure	3.517445
Animation	3.614946
Comedy	3.423993
Crime	3.685044
Documentary	3.705281
Drama	3.677185
Fantasy	3.511589
Film-Noir	3.925728
Horror	3.293563
Musical	3.554716
Mystery	3.670169
Romance	3.542712
Sci-Fi	3.478143
Thriller	3.522964
War	3.791466
Western	3.585755
(no genres listed)	3.326379