DATA 607, Assignment Three - Character Manipulation

Kevin Kirby

2024-09-14

Overview

This is the third weekly assignment for the Fall 2024 edition of DATA 607. This week covers the interpretation and manipulation of regular expressions.

1. Majors with "DATA" or "STATISTICS" in their names

The first question states:

Using the 173 majors listed in [fivethirtyeight.com's College Majors dataset] (https://fivethirtyeight.com/features/the-economic-guide-to-picking-a-college-major/), provide code that identifies the majors that contain either "DATA" or "STATISTICS"

To do so, I downloaded the majors-list_csv from the Github linked in the story and then uploaded the file to my GCP instance.

To answer the question, I will:

- Drop missing/NA values
- Use a filter and knitr to find the right majors

```
library(knitr)
library(readr)
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
```

```
majors_csv <- "https://storage.googleapis.com/data_science_masters_files/2024_fall/data_607_data_manage
majors_data <- read_csv(url(majors_csv))
```

```
## Rows: 174 Columns: 3
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): FOD1P, Major, Major_Category
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
stats_data_majors <- majors_data %>%
  filter(
    !is.na(Major),
    Major != "N/A",
    grepl("DATA|STATISTICS", Major, ignore.case = TRUE)
  )

kable(stats_data_majors, caption = "Majors with 'STATS' or 'DATA' in Name")
```

Table 1: Majors with 'STATS' or 'DATA' in Name

FOD1P	Major	Major_Category
6212	MANAGEMENT INFORMATION SYSTEMS AND STATISTICS	Business
2101	COMPUTER PROGRAMMING AND DATA PROCESSING	Computers & Mathematics
3702	STATISTICS AND DECISION SCIENCE	Computers & Mathematics

The three majors with "STATS" or "DATA" in the title are:

- Management Information Systems And Statistics
- Computer Programming And Data Processing
- Statistics And Decision Science

2. String transformations

This will take the provided list of words and transform them into an output that can later be used to get the list back to its original state.

```r
fruits_before <- '[1] "bell pepper"  "bilberry"     "blackberry"   "blood orange"
[5] "blueberry"    "cantaloupe"   "chili pepper" "cloudberry"
[9] "elderberry"   "lime"         "lychee"       "mulberry"
[13] "olive"        "salal berry"'

text_clean <- gsub("\\[\\d+\\]", "", fruits_before)
text_clean <- gsub("\n", " ", text_clean)

elements <- regmatches(text_clean, gregexpr('"(.*?)"', text_clean))[[1]]

fruits_after <- gsub('"', '', elements)

dput(fruits_after, file = "fruits_after.R")

fruits_before_two <- dget("fruits_after.R")
```

3. Regular expressions explanations

- (.)\1\1
  - This matches any character repeated three times in a row, IE: "aaa" or "bbb".
  - (.) matches any character as long as its not a new line so this can functionally check a whole text
- "(.)(.)\2\1"

  - This looks like for four straight characters that acts as a mirror image of itself, IE: “3443” or “5665”

  - (.)(.) matches the any two consecutive characters, then the \2 looks to see if the third character matches the second and \1 looks to see if the fourth character matches the first

- (..)\1
  - This matches any two characters in a row that are then repeated once, IE: “soso”
  - (..) looks for any two characters and \1 checks for the single repeat
- “(.).\1.\1”
  - This matches five straight characters where the first, third, and fifth characters are the same, with the second and fourth being any character, IE “r7r9r” (.) allows the character to be used in capturing, which is how \1 and \1 at the third and fifth positions can be matched to it
  - the single “.” can also match any character but can’t be used in capturing
- “(.)(.)(.).*\3\2\1”
  - Matches any three straight characters with any characters (including none at all) in between them and those same three characters flipped in reverse, IE “123thankgod321” or “123321” (.)(.)(.) captures the characters for the matching, with .* allowing for zero or more characters in between the \3\2\1 that looks for those same characters in reverse

4. Constructing regular expressions

This question asks me to construct regular expressions that match the specified criteria.

Words that start and end with the same character:

```
regexers <- "^(.).*\\1$"
```

Words that contain a repeated pair of letters:

```
double_oh <- "(..).*\\1"
```

Words that contain one letter repeated in at least three places:

```
thirds <- "([a-zA-Z]).*\\1.*\\1"
```