Lab.4 - Odpowiednie Przygotowanie Danych

Jakub Bryl

11 11 2019

Przygotowanie bibliotek

```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```
require(dplyr)
```

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

```
require(readr)
```

```
## Loading required package: readr
```

```
require(tidyr)
```

```
## Loading required package: tidyr
```

```
require(stats)
require(stringr)
```

```
## Loading required package: stringr
```

Zadanie 1: Proszę wczutać plik pomiaryZapylenia.txt oraz doprowadzić do otrzymania poprawnego technicznie zbioru danych.

```
dane <- read.csv(file="pomiaryZapylenia.txt", header = F)
summary(dane)
```

```
##                 V1       V2
##  al. Krasinskiego:1    22 :1
##  Marek           :1    48 :1
##  Marta           :1    56*:1
##  Monika          :1    68*:1
##  Nowe Huta       :1    9  :1
```

```
#Tutaj przy Col_names = c() podajemy nowe nazwy kolumn, a przy col_types podajemy typ zmiennych kazdej
dane <- read_csv(file="pomiaryZapylenia.txt", col_names =  c("Miejsce", "Zapylenie"), col_types = "cn")
summary(dane)
```

```
##    Miejsce            Zapylenie
##  Length:5           Min.   : 9.0
##  Class :character   1st Qu.:22.0
##  Mode  :character   Median :48.0
##                     Mean   :40.6
##                     3rd Qu.:56.0
##                     Max.   :68.0
```

Zadanie 2: Oczyszczenie danych i przerobienie na dany typu Tidy

```
data <- read_csv(file = "IRCCyN_IVC_1080i_Database_Score.csv", skip = 1)
```

```
## Warning: Missing column names filled in: 'X1' [1], 'X2' [2], 'X42' [42],
## 'X44' [44]
```

```
## Warning: Duplicated column names deduplicated: '1' => '1_1' [45], '2' =>
## '2_1' [46], '3' => '3_1' [47], '4' => '4_1' [48], '5' => '5_1' [49], '6'
## => '6_1' [50], '7' => '7_1' [51], '8' => '8_1' [52], '9' => '9_1' [53],
## '10' => '10_1' [54], '11' => '11_1' [55], '12' => '12_1' [56], '13' =>
## '13_1' [57], '14' => '14_1' [58], '15' => '15_1' [59], '16' => '16_1' [60],
## '17' => '17_1' [61], '18' => '18_1' [62], '19' => '19_1' [63], '20' =>
## '20_1' [64], '21' => '21_1' [65], '22' => '22_1' [66], '23' => '23_1' [67],
## '24' => '24_1' [68], '25' => '25_1' [69], '26' => '26_1' [70], '27' =>
## '27_1' [71], '28' => '28_1' [72], '29' => '29_1' [73], '30' => '30_1' [74],
## '31' => '31_1' [75], '32' => '32_1' [76], '33' => '33_1' [77], '34' =>
## '34_1' [78], '35' => '35_1' [79], '36' => '36_1' [80], '37' => '37_1' [81],
## '38' => '38_1' [82], '39' => '39_1' [83]
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   X2 = col_character(),
##   `30` = col_logical(),
##   X42 = col_logical(),
##   X44 = col_logical(),
```

```
##   `2_1` = col_logical(),
##   `4_1` = col_logical(),
##   `5_1` = col_logical(),
##   `6_1` = col_logical(),
##   `17_1` = col_logical(),
##   `20_1` = col_logical(),
##   `22_1` = col_logical(),
##   `23_1` = col_logical(),
##   `24_1` = col_logical(),
##   `25_1` = col_logical(),
##   `26_1` = col_logical(),
##   `27_1` = col_logical(),
##   `28_1` = col_logical(),
##   `29_1` = col_logical(),
##   `31_1` = col_logical(),
##   `32_1` = col_logical()
##   # ... with 7 more columns
## )

## See spec(...) for full column specifications.
```

```
#Zczytujemy tylko do 41 kolumny
data_2 <- data[,1:41]
#Grupujemy dane po kolumnie tester od kolumny ocena z wyłączeniem kolumn X1 oraz X2
data_2 <- data_2 %>% gather( key = "Tester", value = "Ocena", -X1, -X2)
#Zmiany kosmetyczne odnosnie nazewnictwa kolumny X1 oraz X2, dodatkowo czyscimy z typu NA
colnames(data_2)[2] <- "Zrodlo"
data_2$X1[is.na(data_2$X1)] = 0
data_2$X1[data_2$X1 > 0] = 1
colnames(data_2)[1] <- "Brak Kompresji"

#Inicjalizacja nowych kolumn
data_2$`Zlozonosc Kompresji` = 0
data_2$`Typ` = 0

for (y in seq(1, length(data_2$Zrodlo))){
  #Wyciagamy wspolczynnik kompresji z danego wiersza i dodajemy go do odpowiedniej kolumny
  data_2$`Zlozonosc Kompresji`[y] <- stringr::str_extract(data_2$Zrodlo[y], "\\d*M")
  #Wyciagamy typ filmu(jego rozszerzenie).
  data_2$Typ[y] <- stringr::str_extract(data_2$Zrodlo[y], "\\.(\\w{3})")
  data_2$Typ[y] <- unlist(strsplit(data_2$Zrodlo[y], "\\."))[2]

  #Oczyszczamy kolumne zrodlo w wartosci ktore juz wczesniej wyciagnelismy i
  #uporzadkolismy w dedykowanych kolumnach - usuwanie redundantnych informacji
  if (data_2$`Brak Kompresji`[y] == 1) {
    data_2$`Zlozonosc Kompresji`[y] = 0
    data_2$Zrodlo[y] <- unlist(strsplit(data_2$Zrodlo[y], "\\."))[1]
  } else {
      data_2$Zrodlo[y] <- unlist(strsplit(data_2$Zrodlo[y], ".\\d*M"))[1]
  }
}

summary(data_2)
```

```
##  Brak Kompresji     Zrodlo             Tester              Ocena
##  Min.   :0.000   Length:7488        Length:7488        Min.   :  0.00
##  1st Qu.:0.000   Class :character   Class :character   1st Qu.: 34.00
##  Median :0.000   Mode  :character   Mode  :character   Median : 55.00
##  Mean   :0.125                                         Mean   : 52.04
##  3rd Qu.:0.000                                         3rd Qu.: 70.00
##  Max.   :1.000                                         Max.   :100.00
##                                                        NA's   :3328
##  Zlozonosc Kompresji     Typ
##  Length:7488         Length:7488
##  Class :character    Class :character
##  Mode  :character    Mode  :character
##
##
##
##
```

```
show(data_2)
```

```
## # A tibble: 7,488 x 6
##    `Brak Kompresji` Zrodlo  Tester Ocena `Zlozonosc Kompresji` Typ
##               <dbl> <chr>   <chr>  <dbl> <chr>                 <chr>
##  1                1 credits 1         80 0                     yuv
##  2                0 credits 1         20 4M                    yuv
##  3                0 credits 1         60 6M                    yuv
##  4                0 credits 1         40 7M                    yuv
##  5                0 credits 1         60 8M                    yuv
##  6                0 credits 1         60 9M                    yuv
##  7                0 credits 1         60 10M                   yuv
##  8                0 credits 1        100 14M                   yuv
##  9                1 golf    1        100 0                     yuv
## 10                0 golf    1         20 1M                    yuv
## # ... with 7,478 more rows
```