인문학 텍스트 마이닝

Regular Expression

- 기초 텍스트 분석 방법

```
> x <- c("16_24cat", "25_34cat", "35_44catch", "45_54Cat", "55_104fat")
```

- 더미 데이터 생성

```
> grep(pattern = "cat", x = x)
[1] 1 2 3
```

- cat이 속해 있는 데이터를 찾는다. (소문자)

```
> grep("cat$", x, ignore.case = T)
[1] 1 2 4
```

- 대소문자에 관계없이 cat으로 끝나는 문자를 찾는다.

```
> grepl("cat$", x, ignore.case = T)
[1]  TRUE  TRUE FALSE  TRUE FALSE
```

- 대소문자에 관계없이 cat으로 끝나는 문자를 찾고 논리형으로 나타낸다.

```
> strsplit(x, split = "_")
[[1]]
[1] "16"    "24cat"

[[2]]
[1] "25"    "34cat"

[[3]]
[1] "35"      "44catch"

[[4]]
[1] "45"    "54Cat"

[[5]]
[1] "55"     "104fat"
```

- _를 기준으로 데이터를 나눈다.

```
> sapply(strsplit(x, split = "_"), "[", 2)
[1] "24cat"   "34cat"   "44catch" "54Cat"   "104fat"
```

- _를 기준으로 데이터를 나누고 배열을 무시한 뒤 2번째의 데이터를 추출한다.

```
> gsub(pattern = "cat$", replacement = "fat", x = x, ignore.case = T)
[1] "16_24fat"   "25_34fat"   "35_44catch" "45_54fat"   "55_104fat"
```

- 대소문자에 관계없이 cat으로 끝나는 문자를 찾고 참일 경우 fat으로 문자를 대체한다.

```
> gsub(pattern = "cat", replacement = "fat", x = x, ignore.case = T)
[1] "16_24fat"   "25_34fat"   "35_44fatch" "45_54fat"   "55_104fat"
```

- 대소문자에 관계없이 cat이 속하는 문자를 찾고 참일 경우 fat으로 문자를 대체한다.

Regular Expression

- 정규 표현식을 활용한 텍스트 분석 방법

```
> setwd("/Users/Seongmin_M/Desktop/Class")
```

- 경로지정

```
> fruits <- readLines("fruits.txt")
```

- 데이터 불러오기

```
> install.packages("stringr")
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0100 77633  100 77633
0     0   204k      0 --:--:-- --:--:-- --:--:--  205k

The downloaded binary packages are in
        /var/folders/28/g8cf_pvx46s5phqgwr6qq7jw0000gn/T//RtmptGMrmX/downloaded_packages
>
>
> library(stringr)
```

- 라이브러리 설치 및 불러오기

Regular Expression

```
> matches <- str_match(fruits,"\\w+:\\s\\d+")

> matches
     [,1]
[1,] "apple: 20"
[2,] NA
[3,] "banana: 30"
[4,] NA
[5,] "watermelon: 2"
[6,] "blueberry: 12"
[7,] NA
```

- ₩₩w+:₩₩s₩₩d+의 형태의 데이터를 추출한다.

```
> matches2 <- str_match(fruits,"(\\w+):\\s(\\d+)")

> matches2
     [,1]            [,2]         [,3]
[1,] "apple: 20"     "apple"      "20"
[2,] NA              NA           NA
[3,] "banana: 30"    "banana"     "30"
[4,] NA              NA           NA
[5,] "watermelon: 2" "watermelon" "2"
[6,] "blueberry: 12" "blueberry"  "12"
[7,] NA              NA           NA
```

- 문자와 숫자의 데이터를 추출한다.

```
> fruits.df <- data.frame(na.omit(matches2[,-1]),stringsAsFactors=FALSE)
>
>
> fruits.df
          X1 X2
1      apple 20
2     banana 30
3 watermelon  2
4  blueberry 12
```

- 결측치를 제거 후 데이터형을 변환한다.

```
> names(fruits.df) <- c("fruit","quantity")
>
>
> fruits.df
       fruit quantity
1      apple       20
2     banana       30
3 watermelon        2
4  blueberry       12
```

- 열의 이름을 지정한다.

```
> str(fruits.df)
'data.frame':   4 obs. of  2 variables:
 $ fruit   : chr  "apple" "banana" "watermelon" "blueberry"
 $ quantity: chr  "20" "30" "2" "12"
```

```
> fruits.df$quantity <- as.integer(fruits.df$quantity)
>
>
> str(fruits.df)
'data.frame':   4 obs. of  2 variables:
 $ fruit   : chr  "apple" "banana" "watermelon" "blueberry"
 $ quantity: int  20 30 2 12
```

- quantity변수를 숫자형으로 형변환한다.(integer or numeric)

```
> write.csv(fruits.df,file="fruits.df.csv")
```

- 데이터를 아웃풋한다.

● 정규 표현식(정리 표)

- http://www.endmemo.com/program/R/grep.php