R Basics

Haewon Lee

Table of contents

1 Basics

1.1 R data types

1.1.1 variable type 종류

- numeric, complex

```
(numeric_value <- pi)
```

```
[1] 3.141593
```

```
(1+sqrt(2)*1i)*(1-sqrt(2)*1i)  # complex 연산
```

```
[1] 3+0i
```

```
options(digits=20) # 20자리 표현 (defualt=7)
pi
```

```
[1] 3.1415926535897931159
```

- integer

```
options(digits=10)
(integer_value <- 42L) #반드시 정수라야 하는 경우에는 뒤에 L을 붙인다.
```

```
[1] 42
```

```
typeof(1+1); typeof(1L+1L); typeof(1:3)
```

```
[1] "double"
```

```
[1] "integer"
```

```
[1] "integer"
```

- logical 논리 값 TRUE | FALSE 두가지 값중 하나를 갖는다. 비교를 할 때에는 == 를 사용 (=는 input 명령이 됨)

```
1<0 ; 1>0 ; 1<"a"; "a">"A"; 3==6 #한줄에 여러 명령을 쓸 때에는 semicolon ; 로 구분
```

```
[1] FALSE
```

```
[1] TRUE
```

```
[1] TRUE
```

```
[1] FALSE
```

```
[1] FALSE
```

- charactor : “abc”, “a”, “a123xz” 등 quotation mark로 된 문자열 ” = \” 로 표기 줄바꿈 = \n

```
letters[5:10]; paste("ab","cde", sep = "")
```

```
[1] "e" "f" "g" "h" "i" "j"
```

```
[1] "abcde"
```

```
as.character(345); as.numeric("23.5")
```

```
[1] "345"
```

```
[1] 23.5
```

```
sub("a","x", "father and grandpa"); gsub("a","x", "father and grandpa")
```

```
[1] "fxther and grandpa"
```

```
[1] "fxther xnd grxndpx"
```

```
(ex2 <- 'The "R" project for statistical computing')
```

```
[1] "The \"R\" project for statistical computing"
```

- Escape characters in R :
  - `\t` Insert a tab in the text at this point.
  - `\b` Insert a backspace in the text at this point.
  - `\n` Insert a newline in the text at this point.
  - `\r` Insert a carriage return in the text at this point.
  - `\f` Insert a formfeed in the text at this point.
  - `\s` Insert a space in the text at this point.
  - `\'` Insert a single quote character in the text at this point.
  - `\"` Insert a double quote character in the text at this point.
  - `\\` Insert a backslash character in the text at this point.
- raw : used for binary data
- time : r 에서는 시간을 다루는 방법이 매우 다양함 POSIX 시간변수는 복잡한 list 형태로 되어 있음

```r
(time1 <- as.POSIXlt("1960-01-01")); class(time1); typeof(time1)
```

```
[1] "1960-01-01 KST"
```

```
[1] "POSIXlt" "POSIXt"
```

```
[1] "list"
```

```r
first <- "2022-08-20 08:15:22" ; second <- "2022-01-01 20:04:48"
difftime(first, second); difftime(first, second, units = "hours")
```

```
Time difference of 230.507338 days
```

```
Time difference of 5532.176111 hours
```

```r
first2 <- as.POSIXlt(first); second2 <- as.POSIXlt(second)
second2 - first2
```

```
Time difference of -230.507338 days
```

```r
## difftime(first, second, units = "months")
## match.arg(units)에서 다음과 같은 에러가 발생했습니다:
##  'arg' should be one of "auto", "secs", "mins", "hours", "days", "weeks"
```

1.1.2 data types

- vector : R에서는 모든 변수가 벡터 (열) 로 되어 있다. 다음 연산결과를 예상해 보시오

```r
1:3 + 2:4 ; 1:10 + 1:2
```

```
[1] 3 5 7
```

```
 [1]  2  4  4  6  6  8  8 10 10 12
```

```r
paste(LETTERS[1:10],1:3,sep = "-"); paste(LETTERS[1:3],1:10)
```

```
 [1] "A-1" "B-2" "C-3" "D-1" "E-2" "F-3" "G-1" "H-2" "I-3" "J-1"
```

```
 [1] "A 1"  "B 2"  "C 3"  "A 4"  "B 5"  "C 6"  "A 7"  "B 8"  "C 9"  "A 10"
```

vector의 특징은 모든 요소가 단일한 것이라는 점이다. NA 값을 제외하고는 모든 요소가 같아야 하기 때문에 서로 다른 성질의 것을 입력하게 되면 에러가 생기거나 변형된다.

```r
c(1,2,3); c(1,2,3,"a")
```

```
[1] 1 2 3
```

```
[1] "1" "2" "3" "a"
```

- array : multidimensional vector

```r
(arr1 <- array(data=1:90, dim = c(6,5,3))) # 3Dimensional array
```

```
, , 1

     [,1] [,2] [,3] [,4] [,5]
[1,]    1    7   13   19   25
[2,]    2    8   14   20   26
[3,]    3    9   15   21   27
[4,]    4   10   16   22   28
[5,]    5   11   17   23   29
[6,]    6   12   18   24   30

, , 2

     [,1] [,2] [,3] [,4] [,5]
[1,]   31   37   43   49   55
[2,]   32   38   44   50   56
[3,]   33   39   45   51   57
[4,]   34   40   46   52   58
[5,]   35   41   47   53   59
[6,]   36   42   48   54   60

, , 3

     [,1] [,2] [,3] [,4] [,5]
[1,]   61   67   73   79   85
[2,]   62   68   74   80   86
[3,]   63   69   75   81   87
[4,]   64   70   76   82   88
[5,]   65   71   77   83   89
[6,]   66   72   78   84   90
```

```r
arr1[6,4,2]  # 3Dimensional indexing
```

```
[1] 54
```

```r
which(arr1==54, arr.ind = TRUE )
```

```
     dim1 dim2 dim3
[1,]    6    4    2
```

- matrix : 2dimensional vector

```
matrix(data = c(3,4,5,6,7,8),
       nrow=2,
       ncol=3,  # nrow=2 하나만 지정해도 ncol=3은 내부적으로 결정됨
       byrow = TRUE,  # data assign 하는 방향
       dimnames = list(c("pt_1", "pt_2"),        # row names
                       c("var1","var2","var3"))  # col names
              )
```

```
     var1 var2 var3
pt_1    3    4    5
pt_2    6    7    8
```

```
x <- 2:9 ; names(x) <- x  # x의 이름을 부여
x %o% x  # = outer function : outer(x,x, FUN="*")
```

```
   2  3  4  5  6  7  8  9
2  4  6  8 10 12 14 16 18
3  6  9 12 15 18 21 24 27
4  8 12 16 20 24 28 32 36
5 10 15 20 25 30 35 40 45
6 12 18 24 30 36 42 48 54
7 14 21 28 35 42 49 56 63
8 16 24 32 40 48 56 64 72
9 18 27 36 45 54 63 72 81
```

- data frame : vector를 구성요소로 한 list의 형태 (외형적으로 보면 2dimension으로 보인다)

  dataframe의 구성요소는 vector들 (각각의 vector는 동일한 데이터 타입이라야 함)

```
## dataframe 만들기
df1 <- data.frame( col1 = c("A", "B", "Anyone", "None"),
                   col2 = c(160, 170, 180, 200),
                   col3 = c(TRUE, FALSE, FALSE, TRUE)
                   )
df1
```

```
    col1 col2  col3
1      A  160  TRUE
2      B  170 FALSE
3 Anyone  180 FALSE
4   None  200  TRUE
```

데이터프레임 이름 <- data.frame(컬럼이름= c(data_1, ···. , data_n), ···..) 이런 형식으로 데이터 프레임을 만들 수 있다. 데이터프레임이 R의 기본적인 데이터 양식이기 때문에 이를 다루는 방법이 다양하게 존재함

```
## dataframe cell 찾기
df1[3,2] #3행 2열의 데이터
```

```
[1] 180
```

```
## column 이름으로 찾기
df1$col1 ; df1[, "col1"]; df1["col1"] ### df1의 col1 열을 찾는 방법들
```

```
[1] "A"      "B"      "Anyone" "None"
```

```
[1] "A"      "B"      "Anyone" "None"
```

```
    col1
1      A
2      B
3 Anyone
4   None
```

```
df1[,1]
```

```
[1] "A"      "B"      "Anyone" "None"
```

- list : R에만 있는 독특한 데이터타입이다. 이것은 모든 데이터 타입을 담을 수 있는 형태이고 자료의 길이가 달라도 같이 담을 수가 있게 되어 있다. 또한 리스트 속에 리스트를 넣을 수 있기에 다단계로 nesting 되는 구조로 만들 수 있다.

```
sample_list <- list(data1=df1, data2 = arr1, data3 = x%o%x)
str(sample_list)
```

```
List of 3
 $ data1:'data.frame':  4 obs. of  3 variables:
  ..$ col1: chr [1:4] "A" "B" "Anyone" "None"
  ..$ col2: num [1:4] 160 170 180 200
  ..$ col3: logi [1:4] TRUE FALSE FALSE TRUE
 $ data2: int [1:6, 1:5, 1:3] 1 2 3 4 5 6 7 8 9 10 ...
 $ data3: num [1:8, 1:8] 4 6 8 10 12 14 16 18 6 9 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:8] "2" "3" "4" "5" ...
  .. ..$ : chr [1:8] "2" "3" "4" "5" ...
```

```
sample_list$data1
```

```
    col1 col2  col3
1      A  160  TRUE
2      B  170 FALSE
3 Anyone  180 FALSE
4   None  200  TRUE
```

```
  sample_list$data1[,3]
```

```
[1]  TRUE FALSE FALSE  TRUE
```

1.2 Built-in Data sets in R

1.2.1 소개 및 개요 : R에는 내장된 데이터세트가 있다. 테스트용, 교육용 및 연습용으로 이러한 데이터세트를 사용하면 좋다.

- 사용법

```
data("volcano") ## built-in dataset 중에서 volcano 사용
library(survival)
data(package="survival") ## survival package에 어떤 데이터 세트들이 있는지 확인
data(cancer)  ## data(cancer, package="survival") 와 같이 사용해도 된다.
str(lung) ## cancer dataset에는 다양한 암종류의 생존분석용 데이터가 들어가 있다.
```

```
'data.frame':   228 obs. of  10 variables:
 $ inst     : num  3 3 3 5 1 12 7 11 1 7 ...
 $ time     : num  306 455 1010 210 883 ...
 $ status   : num  2 2 1 2 2 1 2 2 2 2 ...
 $ age      : num  74 68 56 57 60 74 68 71 53 61 ...
 $ sex      : num  1 1 1 1 1 1 2 2 1 1 ...
 $ ph.ecog  : num  1 0 0 1 0 1 2 2 1 2 ...
 $ ph.karno : num  90 90 90 90 100 50 70 60 70 70 ...
 $ pat.karno: num  100 90 90 60 90 80 60 80 80 70 ...
 $ meal.cal : num  1175 1225 NA 1150 NA ...
 $ wt.loss  : num  NA 15 15 11 0 0 10 1 16 34 ...
```

- rotterdam : breast cancer dataset in survival package

```
library(moonBook)
mytable(grade~. , data=rotterdam)
```

```
      Descriptive Statistics by 'grade'
-------------------------------------------------
                  2              3          p
               (N=794)        (N=2188)
-------------------------------------------------
 pid       1328.4 ± 865.1  1569.0 ± 860.9  0.000
 year       1987.9 ±  3.1   1988.3 ±  3.0  0.004
 age         54.4 ± 12.7     55.3 ± 13.1   0.086
 meno                                      0.000
   - 0       392 (49.4%)     920 (42.0%)
   - 1       402 (50.6%)    1268 (58.0%)
 size                                      0.000
   - <=20    462 (58.2%)     925 (42.3%)
   - 20-50   290 (36.5%)    1001 (45.7%)
   - >50      42 ( 5.3%)     262 (12.0%)
 nodes        2.0 ±  3.7      3.0 ±  4.6   0.000
 pgr        236.2 ± 385.8   134.9 ± 242.8  0.000
 er         179.8 ± 291.9   161.8 ± 265.0  0.127
```

```
hormon                                          0.000
  - 0       735 (92.6%)    1908 (87.2%)
  - 1       59 ( 7.4%)      280 (12.8%)
chemo                                           1.000
  - 0       640 (80.6%)    1762 (80.5%)
  - 1       154 (19.4%)     426 (19.5%)
rtime     2458.3 ± 1408.6 1967.1 ± 1370.8 0.000
recur                                           0.000
  - 0       480 (60.5%)     984 (45.0%)
  - 1       314 (39.5%)    1204 (55.0%)
dtime     2908.9 ± 1278.6 2495.2 ± 1287.8 0.000
death                                           0.000
  - 0       532 (67.0%)    1178 (53.8%)
  - 1       262 (33.0%)    1010 (46.2%)
-----------------------------------------------------
```

```
mytable(grade~. , data=rotterdam) %>% mylatex() %>% cat
```

Descriptive Statistics by grade

	2 (N=794)	3 (N=2188)	p
pid	1328.4 ± 865.1	1569.0 ± 860.9	0.000
year	1987.9 ± 3.1	1988.3 ± 3.0	0.004
age	54.4 ± 12.7	55.3 ± 13.1	0.086
meno			0.000
- 0	392 (49.4%)	920 (42.0%)	
- 1	402 (50.6%)	1268 (58.0%)	
size			0.000
- <=20	462 (58.2%)	925 (42.3%)	
- 20-50	290 (36.5%)	1001 (45.7%)	
- >50	42 (5.3%)	262 (12.0%)	
nodes	2.0 ± 3.7	3.0 ± 4.6	0.000
pgr	236.2 ± 385.8	134.9 ± 242.8	0.000
er	179.8 ± 291.9	161.8 ± 265.0	0.127
hormon			0.000
- 0	735 (92.6%)	1908 (87.2%)	
- 1	59 (7.4%)	280 (12.8%)	
chemo			1.000
- 0	640 (80.6%)	1762 (80.5%)	
- 1	154 (19.4%)	426 (19.5%)	
rtime	2458.3 ± 1408.6	1967.1 ± 1370.8	0.000
recur			0.000
- 0	480 (60.5%)	984 (45.0%)	
- 1	314 (39.5%)	1204 (55.0%)	
dtime	2908.9 ± 1278.6	2495.2 ± 1287.8	0.000
death			0.000
- 0	532 (67.0%)	1178 (53.8%)	
- 1	262 (33.0%)	1010 (46.2%)	

```
## LaTeX을 이용하여 깔끔한 논문형식의 테이블을 만들 수 있다.
```

1.3 Basic statistics functions

1.3.1 t-test

R function : t.test -

option arguments : alternative = c("two.sided", "less", "greater"), formula (종속변수~ 독립변수)

help files : ?t.test 를 치면 함수의 argument, values(results), detail에 대해서 설명이 나옴

```
group1 <- rotterdam[ rotterdam$grade == 2, "age"]
group2 <- rotterdam[ rotterdam$grade != 2, "age"]
t.test(group1, group2)   ## unmatched 임의의 두개의 vector로 비교
```

```
    Welch Two Sample t-test

data:  group1 and group2
t = -1.7436947, df = 1444.4213, p-value = 0.08142509
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -1.9599008823  0.1152640033
sample estimates:
  mean of x   mean of y
54.38161209 55.30393053
```

```
t.test(age~meno,data=rotterdam)  ## matched 한개의 데이터프레임에서 paired t-test
```

```
    Welch Two Sample t-test

data:  age by meno
t = -76.545414, df = 2972.8397, p-value < 2.2204e-16
alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
95 percent confidence interval:
 -21.53192159 -20.45636342
sample estimates:
mean in group 0 mean in group 1
    43.30106707     64.29520958
```

1.3.2 χ^2 (chi-square) test

R function : chisq.test

```
table(rotterdam[,c("hormon","size")])
```

```
      size
hormon <=20 20-50  >50
     0 1283  1119  241
     1  104   172   63
```

```
chisq.test(table(rotterdam[,c("hormon","size")]), correct = TRUE)
```

```
    Pearson's Chi-squared test

data:  table(rotterdam[, c("hormon", "size")])
X-squared = 51.920064, df = 2, p-value = 5.317424e-12
```

```
chisq.test(rotterdam$hormon, rotterdam$chemo)
```

```
    Pearson's Chi-squared test with Yates' continuity correction

data:  rotterdam$hormon and rotterdam$chemo
X-squared = 29.771106, df = 1, p-value = 4.861842e-08
```

```
x <- matrix(c(12, 5, 7, 7), ncol = 2)  ## matrix를 만들어서 검정하는 방법
x
```

```
     [,1] [,2]
[1,]   12    7
[2,]    5    7
```

```
chisq.test(x)$p.value   ## chisq test의 결과물은 list이다 여기서 p.value 부분만 출력
```

```
[1] 0.4233054243
```

```
chisq.test(x, simulate.p.value = TRUE, B = 10000)$p.value
```

```
[1] 0.2919708029
```

1.3.3 generalized linear regression and Loess smoothing (LOcal regrESSion)

R function : glm (generalized linear models) 다중 선형회귀

```
data(economics, package="ggplot2")
economics$index <- 1:nrow(economics) # create index variable
glm_model1 <- glm(psavert~pop, data = economics)
summary(glm_model1)
```

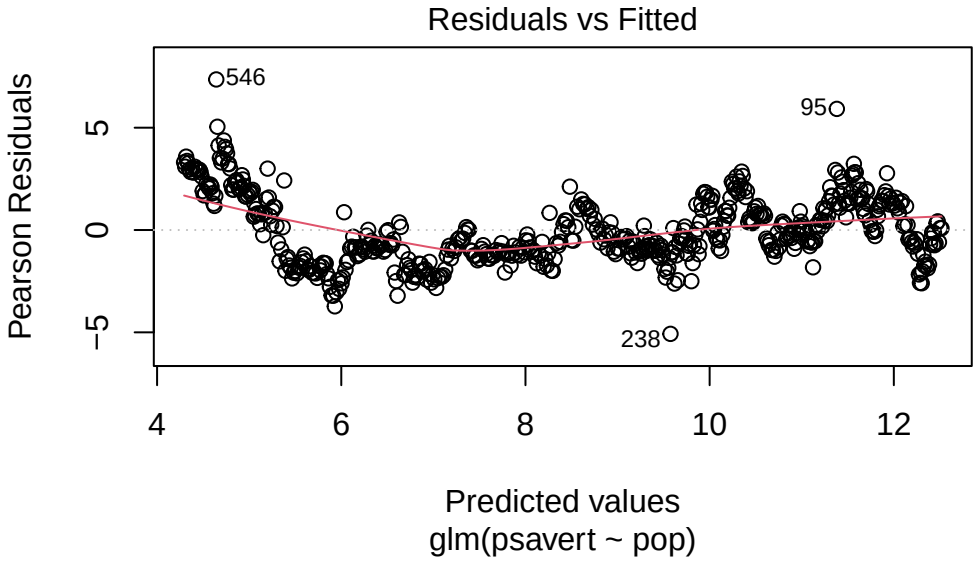
```
Call:
glm(formula = psavert ~ pop, data = economics)

Coefficients:
                Estimate    Std. Error   t value   Pr(>|t|)
(Intercept)  2.594603e+01  4.811677e-01  53.92305 < 2.22e-16 ***
pop         -6.757974e-05  1.852367e-06 -36.48290 < 2.22e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 2.645600898)
```

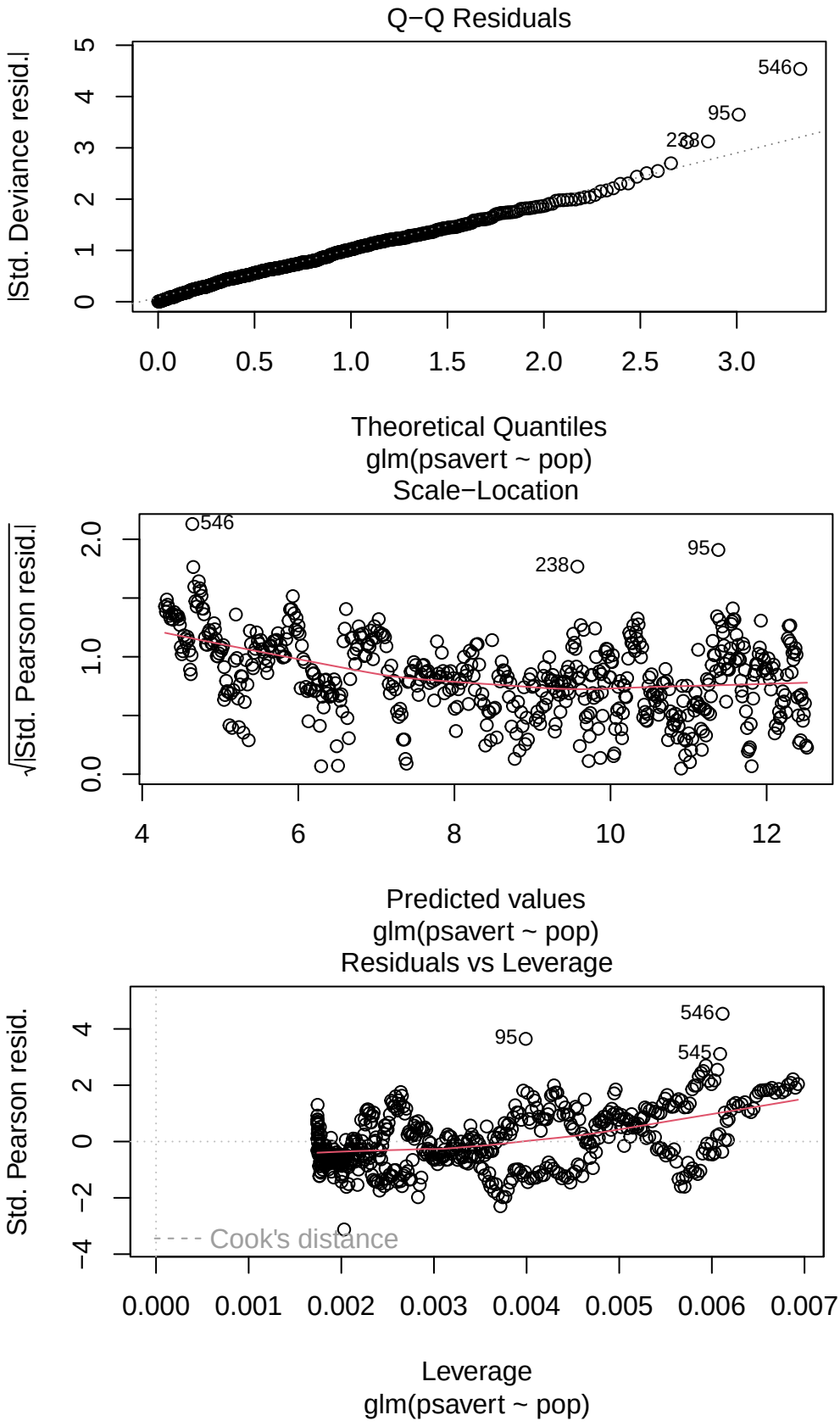
```
    Null deviance: 5034.5843  on 573  degrees of freedom
Residual deviance: 1513.2837  on 572  degrees of freedom
AIC: 2191.3815

Number of Fisher Scoring iterations: 2
```

```
anova(glm_model1)
```

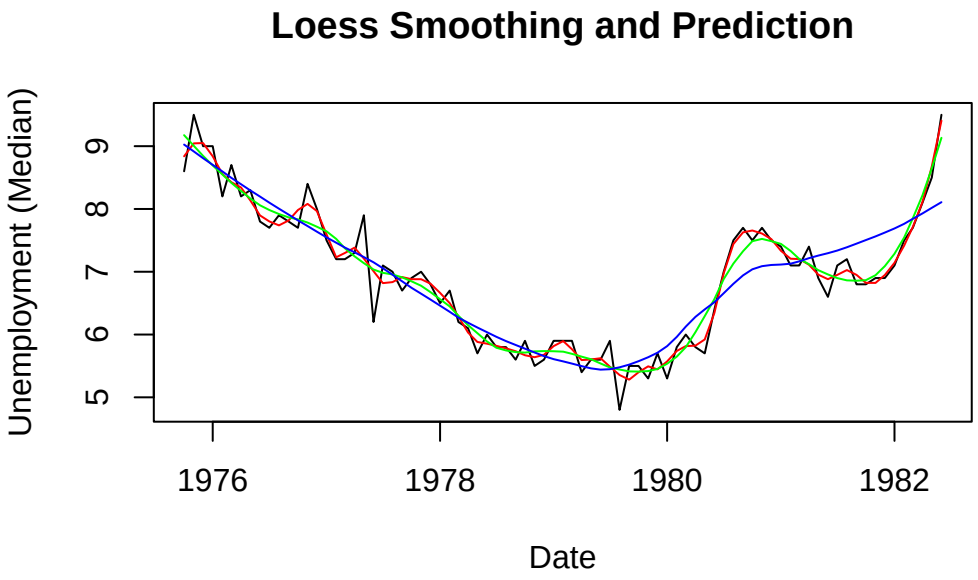
```
Analysis of Deviance Table

Model: gaussian, link: identity

Response: psavert

Terms added sequentially (first to last)

     Df  Deviance Resid. Df Resid. Dev         F     Pr(>F)
NULL                   573  5034.5843
pop   1 3521.3005       572  1513.2837 1331.0022 < 2.22e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(glm_model1)
```

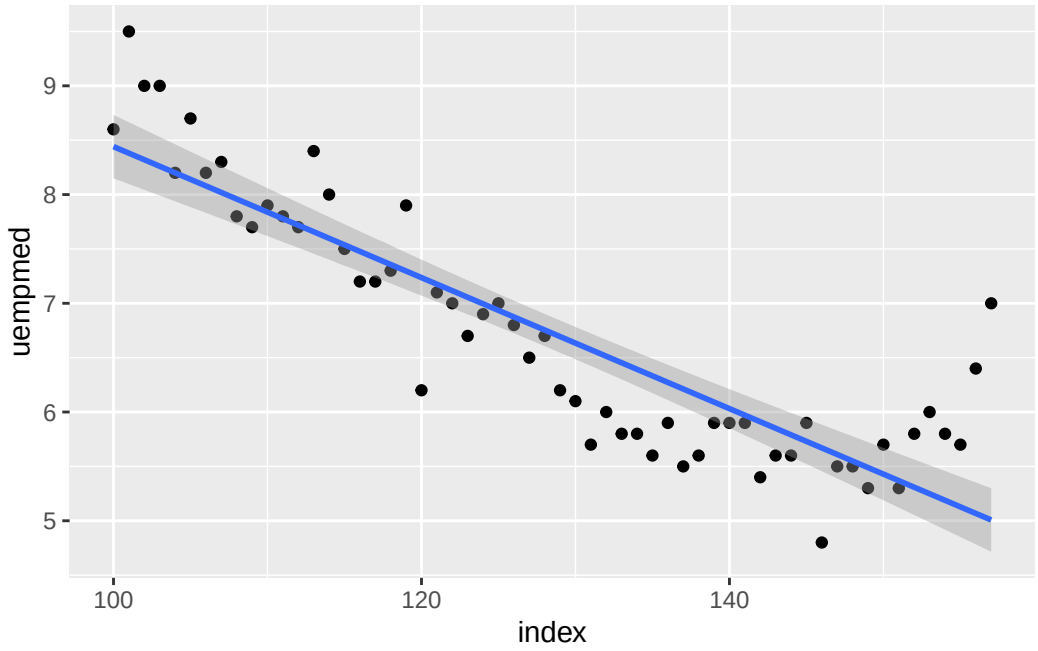



Q−Q Residuals
|Std. Deviance resid.|
546
95
238
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Theoretical Quantiles
glm(psavert ~ pop)
Scale−Location
√|Std. Pearson resid.|
546
238
95
4
6
8
10
12
Predicted values
glm(psavert ~ pop)
Residuals vs Leverage
Std. Pearson resid.
546
95
545
Cook's distance
0.000
0.001
0.002
0.003
0.004
0.005
0.006
0.007
Leverage
glm(psavert ~ pop)

```r
economics <- economics[100:180, ] ### narrow span
loessMod10 <- loess(uempmed ~ index, data=economics, span=0.10) # 10% smoothing span
loessMod25 <- loess(uempmed ~ index, data=economics, span=0.25)
loessMod50 <- loess(uempmed ~ index, data=economics, span=0.50)
smoothed10 <- predict(loessMod10)
smoothed25 <- predict(loessMod25)
smoothed50 <- predict(loessMod50)
plot(economics$uempmed, x=economics$date, type="l", main="Loess Smoothing and Prediction", xlab="Date",
lines(smoothed10, x=economics$date, col="red")
lines(smoothed25, x=economics$date, col="green")
lines(smoothed50, x=economics$date, col="blue")
```




```r
economics <- economics[1:58,]
library(ggplot2)
ggplot(data=economics, aes(x=index, y=uempmed))+
  geom_point()+
  geom_smooth(method = "lm")
```



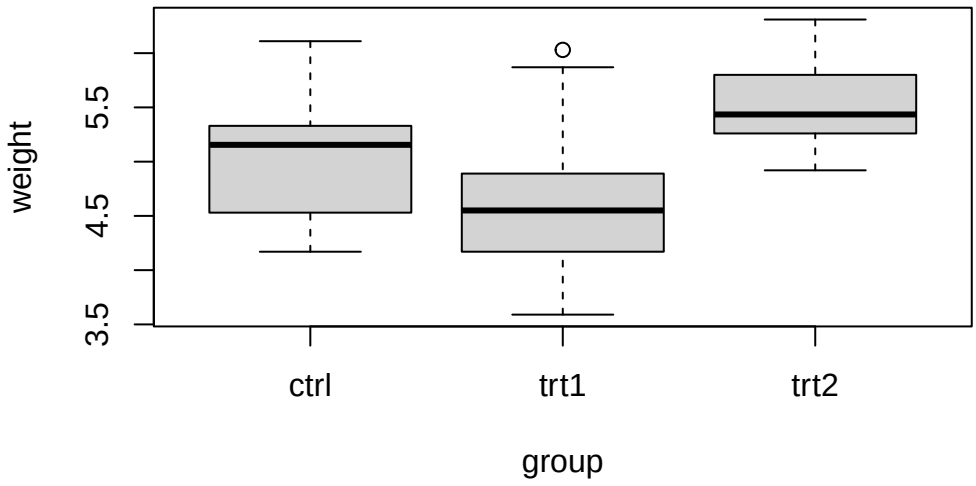
1.3.4 One-way ANOVA

```
library(psych)
PlantGrowth  ## 내장 dataset
```

```
   weight group
1    4.17  ctrl
2    5.58  ctrl
3    5.18  ctrl
4    6.11  ctrl
5    4.50  ctrl
6    4.61  ctrl
7    5.17  ctrl
8    4.53  ctrl
9    5.33  ctrl
10   5.14  ctrl
11   4.81  trt1
12   4.17  trt1
13   4.41  trt1
14   3.59  trt1
15   5.87  trt1
16   3.83  trt1
17   6.03  trt1
18   4.89  trt1
19   4.32  trt1
20   4.69  trt1
21   6.31  trt2
22   5.12  trt2
23   5.54  trt2
```

```
24   5.50  trt2
25   5.37  trt2
26   5.29  trt2
27   4.92  trt2
28   6.15  trt2
29   5.80  trt2
30   5.26  trt2
```

```
plot(weight~group, data = PlantGrowth)  ## Boxplot으로 자동으로 그려준다.
```




```
with(PlantGrowth, describeBy(weight,group))
```

```
 Descriptive statistics by group
group: ctrl
   vars  n mean   sd median trimmed  mad  min  max range skew kurtosis   se
X1    1 10 5.03 0.58   5.15       5 0.72 4.17 6.11  1.94 0.23    -1.12 0.18
------------------------------------------------------------
group: trt1
   vars  n mean   sd median trimmed  mad  min  max range skew kurtosis   se
X1    1 10 4.66 0.79   4.55    4.62 0.53 3.59 6.03  2.44 0.47     -1.1 0.25
------------------------------------------------------------
group: trt2
   vars  n mean   sd median trimmed  mad  min  max range skew kurtosis   se
X1    1 10 5.53 0.44   5.44     5.5 0.36 4.92 6.31  1.39 0.48    -1.16 0.14
```

```
bartlett.test(PlantGrowth$weight ~ PlantGrowth$group)  ## 등분산 가정을 체크함
```

```
    Bartlett test of homogeneity of variances

data:  PlantGrowth$weight by PlantGrowth$group
Bartlett's K-squared = 2.8785738, df = 2, p-value = 0.2370968
```

```r
aov_model <- aov(weight~group, data = PlantGrowth)
summary(aov_model)
```

```
            Df   Sum Sq   Mean Sq F value  Pr(>F)
group        2  3.76634 1.8831700 4.84609 0.01591 *
Residuals   27 10.49209 0.3885959
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

1.3.5 Correlation tests

Pearson correlation formula

$$r = \frac{\sum(x - m_x)(y - m_y)}{\sqrt{\sum(x - m_x)^2 \sum(y - m_y)^2}}$$

Spearman correlation formula : non-parametric

$$\rho = \frac{\sum(x' - m_{x'})(y' - m_{y'})}{\sqrt{\sum(x' - m_{x'})^2 \sum(y' - m_{y'})^2}}$$

$$\text{where } x' = rank(x) \quad \text{and} \quad y' = rank(y)$$

Kendall correlation formula : non-parametric

$$\tau = \frac{n_c - n_d}{\frac{1}{2}n(n-1)}$$

where n_c : number of concordant pairs, n_d : number of concordant pairs, n : size of $x + y$

```r
res <- cor.test(economics$index, economics$uempmed, method = "pearson")
res
```

```
    Pearson's product-moment correlation

data:  economics$index and economics$uempmed
t = -13.654567, df = 56, p-value < 2.2204e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.9255815657 -0.7998071541
sample estimates:
          cor
-0.8769389168
```

```r
res$p.value  ## res는 리스트형태로 나오는 결과물이다. 여기에서 필요한 값만 골라냄
```

```
[1] 1.822639345e-19
```

```
res$estimate
```

```
         cor
-0.8769389168
```

```
res2 <- cor.test(economics$index, economics$uempmed, method = "spearman")
res2
```

```
    Spearman's rank correlation rho

data:  economics$index and economics$uempmed
S = 60517.721, p-value < 2.2204e-16
alternative hypothesis: true rho is not equal to 0
sample estimates:
         rho
-0.861568223
```

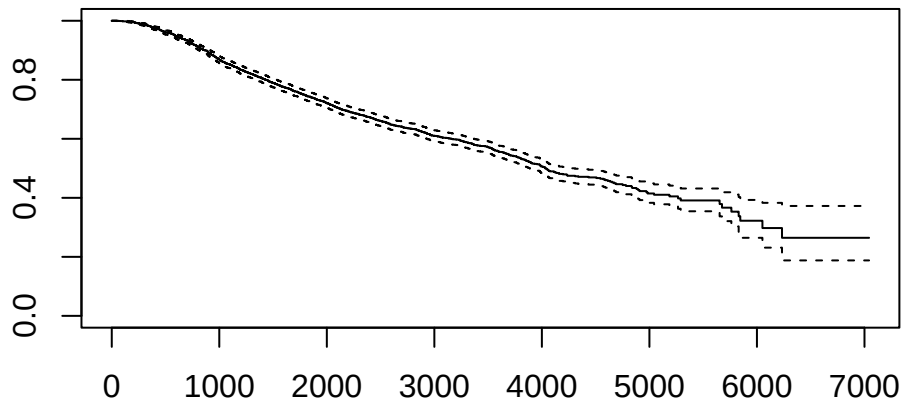
```
res3 <- cor.test(economics$index, economics$uempmed, method = "kendall")
res3
```

```
    Kendall's rank correlation tau

data:  economics$index and economics$uempmed
z = -7.8580733, p-value = 3.90087e-15
alternative hypothesis: true tau is not equal to 0
sample estimates:
          tau
-0.7167487038
```

1.3.6 Survival analysis

- Kaplan Meier Analysis - Basic survival model survival::Surv

```
km <- Surv(rotterdam$dtime, event = rotterdam$death)  ## default type : "right"
plot(km)  ## km - Surv class (time, status) 가지고 있는 리스트
```




```
median(km); mean(km)  ## Surv 객체에 대한 method 함수들이 있다. plot.Surv포함
```

```
$quantile
  50
4033

$lower
  50
3888

$upper
  50
4309

[1] 1302.8833
```

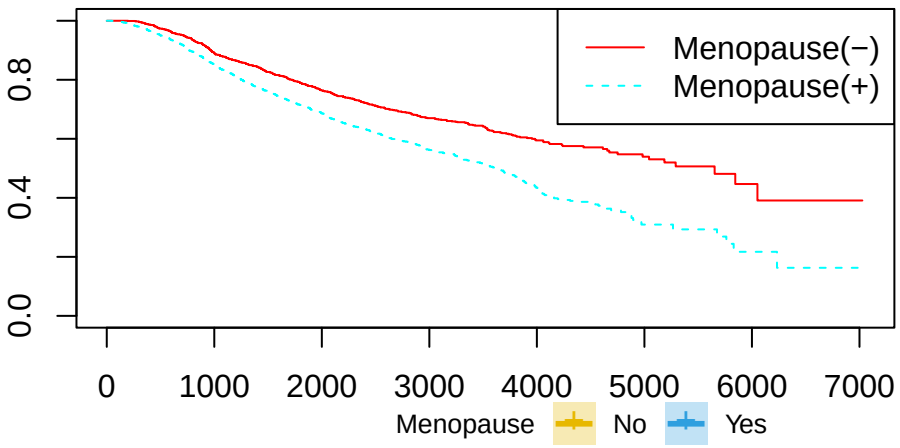
- Kaplan Meier Analysis - survfit model

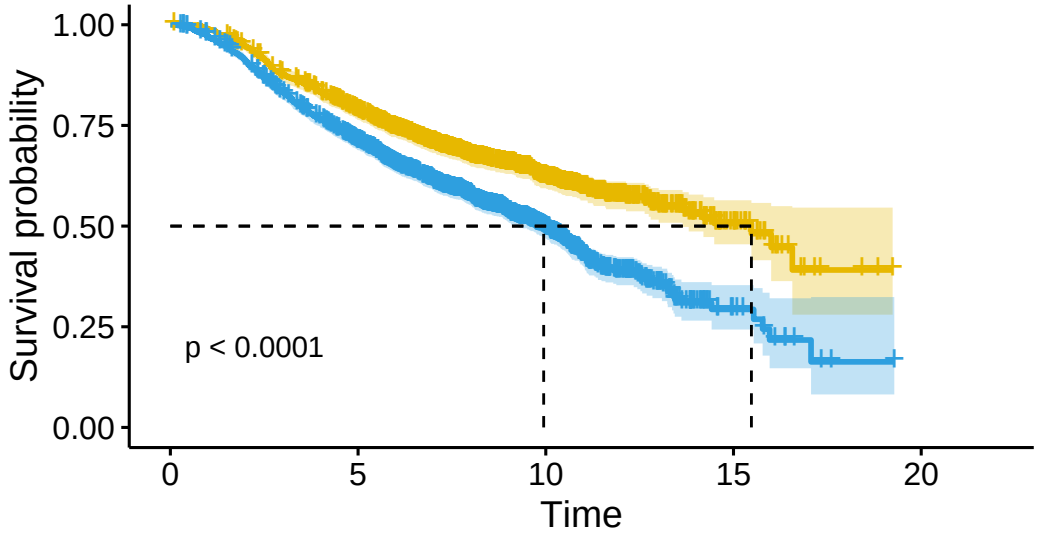
```
km_fit <- survfit(km~rotterdam$meno)
summary(km_fit, c(365*1:19))   ### 정해진 time에 맞는 생존테이블표를 만든다.
```

```
Call: survfit(formula = km ~ rotterdam$meno)

                rotterdam$meno=0
 time n.risk n.event survival    std.err lower 95% CI upper 95% CI
  365   1298      13 0.990084 0.00273656     0.984735     0.995462
  730   1236      56 0.947311 0.00617381     0.935287     0.959489
 1095   1140      90 0.878196 0.00905336     0.860630     0.896121
 1460   1071      59 0.832587 0.01034896     0.812549     0.853120
 1825    973      59 0.786049 0.01140771     0.764005     0.808729
 2190    865      50 0.744541 0.01222354     0.720964     0.768888
 2555    754      43 0.706203 0.01291841     0.681332     0.731982
 2920    611      31 0.674528 0.01353781     0.648509     0.701590
 3285    480      15 0.656234 0.01397343     0.629410     0.684201
 3650    345      21 0.622823 0.01505424     0.594005     0.653039
 4015    217      13 0.594613 0.01631412     0.563482     0.627463
```

```
4380    138       6 0.575323 0.01759653     0.541848     0.610866
4745     88       4 0.553799 0.01999709     0.515960     0.594412
5110     54       3 0.530422 0.02334386     0.486587     0.578207
5475     29       2 0.506485 0.02783275     0.454769     0.564082
5840     14       1 0.481161 0.03617160     0.415241     0.557545
6205      5       2 0.390943 0.06657871     0.279996     0.545853
6570      3       0 0.390943 0.06657871     0.279996     0.545853
6935      1       0 0.390943 0.06657871     0.279996     0.545853

               rotterdam$meno=1
time n.risk n.event survival    std.err lower 95% CI upper 95% CI
 365   1616      46 0.972378 0.00401599    0.9645389     0.980281
 730   1508     103 0.910256 0.00701496    0.8966099     0.924109
1095   1366     129 0.832077 0.00918891    0.8142608     0.850283
1460   1245     111 0.764188 0.01045754    0.7439639     0.784962
1825   1111      87 0.709950 0.01121688    0.6883018     0.732278
2190    944      82 0.655456 0.01186297    0.6326122     0.679124
2555    819      58 0.613767 0.01230810    0.5901113     0.638371
2920    642      45 0.577382 0.01272281    0.5529763     0.602864
3285    474      42 0.535877 0.01333692    0.5103646     0.562665
3650    342      31 0.495578 0.01418038    0.4685496     0.524165
4015    188      35 0.430288 0.01613537    0.3997973     0.463104
4380    113      17 0.386353 0.01771621    0.3531444     0.422684
4745     62       6 0.357899 0.01988720    0.3209686     0.399079
5110     28       7 0.309356 0.02431136    0.2651946     0.360871
5475     14       1 0.293074 0.02795732    0.2430961     0.353326
5840      8       3 0.217092 0.04323095    0.1469392     0.320737
6205      4       0 0.217092 0.04323095    0.1469392     0.320737
6570      1       1 0.162819 0.05710016    0.0818823     0.323757
6935      1       0 0.162819 0.05710016    0.0818823     0.323757
```

```r
plot(km_fit, col = rainbow(2), lty=1:2)
legend("topright", legend = c("Menopause(-)","Menopause(+)"),
       col= rainbow(2), lty=1:2)
library(survminer)
ggsurvplot(km_fit, data = rotterdam,
           conf.int = T, xscale = 365.2425,  ## xscale can be "d_y"
           break.x.by = 5*365.2425,
           pval = T, pval.size =4, surv.median.line = "hv",
           risk.table = FALSE, ## if TRUE, risk table is displayed under graph
           legend.title="Menopause", legend.labs=c("No","Yes"),
           palette = c("#E7B800", "#2E9FDF"),)
```






```
## ggplot + survminer package
```

- Cox Proportional model

$$hazard\ function\ h(t) = \lim_{\Delta t \to 0} \frac{\Pr[(t \leq T < t + \Delta t | T \geq t)]}{\Delta t} \quad = \quad \frac{p(t)}{S(t)}$$

$$\log h_i(t) = \alpha + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik}$$

```
args(coxph)
```

```
function (formula, data, weights, subset, na.action, init, control,
    ties = c("efron", "breslow", "exact"), singular.ok = TRUE,
    robust, model = FALSE, x = FALSE, y = TRUE, tt, method = ties,
    id, cluster, istate, statedata, nocenter = c(-1, 0, 1), ...)
NULL
```

```r
library(carData)   ## Rossi data set 이용하기 위해서 사용
library(car)       ## Anova function
colnames(Rossi)  # emp1-52 : factor (yes or no)
```

```
 [1] "week"   "arrest" "fin"    "age"    "race"   "wexp"   "mar"    "paro"
 [9] "prio"   "educ"   "emp1"   "emp2"   "emp3"   "emp4"   "emp5"   "emp6"
[17] "emp7"   "emp8"   "emp9"   "emp10"  "emp11"  "emp12"  "emp13"  "emp14"
[25] "emp15"  "emp16"  "emp17"  "emp18"  "emp19"  "emp20"  "emp21"  "emp22"
[33] "emp23"  "emp24"  "emp25"  "emp26"  "emp27"  "emp28"  "emp29"  "emp30"
[41] "emp31"  "emp32"  "emp33"  "emp34"  "emp35"  "emp36"  "emp37"  "emp38"
[49] "emp39"  "emp40"  "emp41"  "emp42"  "emp43"  "emp44"  "emp45"  "emp46"
[57] "emp47"  "emp48"  "emp49"  "emp50"  "emp51"  "emp52"
```

```r
cox_model1 <- coxph(Surv(week, arrest) ~
                    fin + age +  race + wexp + mar + paro + prio,
                    data = Rossi)
summary(cox_model1)
```

```
Call:
coxph(formula = Surv(week, arrest) ~ fin + age + race + wexp +
    mar + paro + prio, data = Rossi)

  n= 432, number of events= 114

                     coef   exp(coef)    se(coef)        z  Pr(>|z|)
finyes         -0.37942217  0.68425668  0.19137948 -1.98256 0.0474161 *
age            -0.05743774  0.94418067  0.02199947 -2.61087 0.0090312 **
raceother      -0.31389979  0.73059224  0.30799278 -1.01918 0.3081180
wexpyes        -0.14979570  0.86088384  0.21222430 -0.70584 0.4802897
marnot married  0.43370388  1.54296190  0.38186806  1.13574 0.2560642
paroyes        -0.08487108  0.91863070  0.19575667 -0.43355 0.6646124
prio            0.09149708  1.09581358  0.02864855  3.19378 0.0014042 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

               exp(coef) exp(-coef) lower .95 upper .95
finyes         0.6842567  1.4614399 0.4702367 0.9956841
age            0.9441807  1.0591193 0.9043345 0.9857825
raceother      0.7305922  1.3687526 0.3994948 1.3361001
wexpyes        0.8608838  1.1615969 0.5679354 1.3049390
marnot married 1.5429619  0.6481041 0.7299759 3.2613836
paroyes        0.9186307  1.0885767 0.6259110 1.3482466
prio           1.0958136  0.9125640 1.0359791 1.1591039

Concordance= 0.64  (se = 0.027 )
Likelihood ratio test= 33.27  on 7 df,   p=2.36e-05
Wald test            = 32.11  on 7 df,   p=3.87e-05
```

```
Score (logrank) test = 33.53  on 7 df,   p=2.11e-05
```

```
  Anova(cox_model1)
```

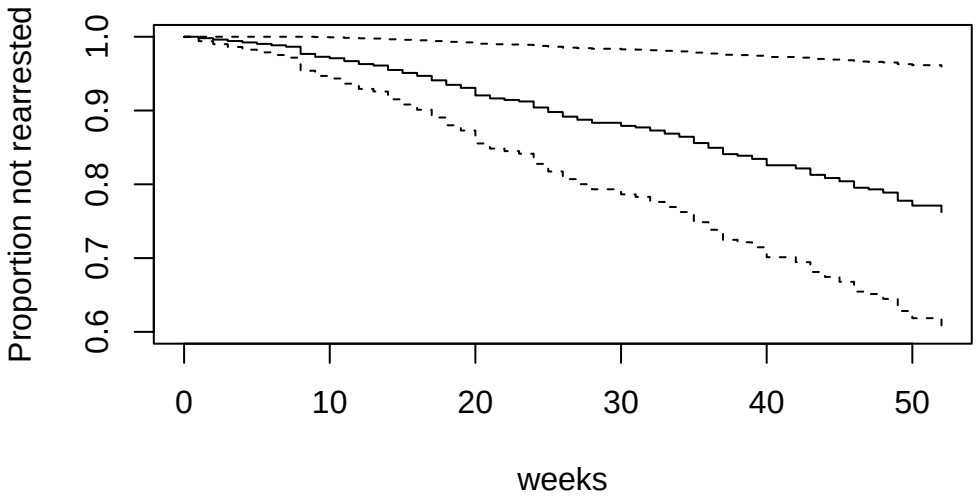
```
Analysis of Deviance Table (Type II tests)
      LR Chisq Df Pr(>Chisq)
fin  3.9862101  1  0.0458741 *
age  7.9880173  1  0.0047088 **
race 1.1251518  1  0.2888118
wexp 0.5003372  1  0.4793520
mar  1.4311793  1  0.2315721
paro 0.1869702  1  0.6654503
prio 8.9765972  1  0.0027346 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
  anova(cox_model1)
```

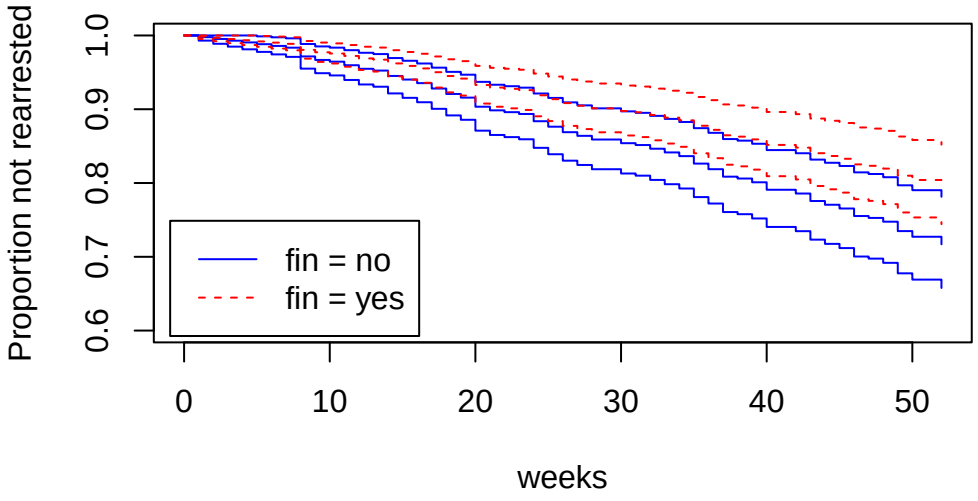
```
Analysis of Deviance Table
 Cox model: response is Surv(week, arrest)
Terms added sequentially (first to last)

         loglik    Chisq Df Pr(>|Chi|)
NULL -675.38063
fin  -673.46210  3.83706  1 0.05013146 .
age  -666.23582 14.45257  1 0.00014373 ***
race -665.84148  0.78867  1 0.37450208
wexp -664.21789  3.24717  1 0.07154674 .
mar  -663.57584  1.28411  1 0.25713587
paro -663.23596  0.67976  1 0.40966904
prio -658.74766  8.97660  1 0.00273459 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

모델의 전체적인 생존곡선을 알고 싶으면 survfit 함수를 이용해서 생존곡선을 그릴 수 있다.

```
  plot(survfit(cox_model1), ylim = c(0.6,1),xlab = "weeks",
       ylab = "Proportion not rearrested")
```




```
Rossi.fin <- with(Rossi, data.frame(fin=c(0, 1),
        age=rep(mean(age), 2), race=rep(mean(race == "other"), 2),
        wexp=rep(mean(wexp == "yes"), 2), mar=rep(mean(mar == "not married"), 2),
        paro=rep(mean(paro == "yes"), 2), prio=rep(mean(prio), 2)))
## fin = 0,1 이것을 두그룹으로 나누고 나머지 변수들은 평균적인 값으로 고정해 버림
plot(survfit(cox_model1, newdata = Rossi.fin), conf.int = T,
     lty = c(1,2), ylim = c(0.6,1),xlab = "weeks",
     ylab = "Proportion not rearrested", col = c("blue","red")
     )
legend("bottomleft", legend=c("fin = no", "fin = yes"), lty=c(1 ,2),
       col=c("blue","red") , inset=0.02)
```



1.4 시간과 날짜 변수 다루기 : Management of date-time variables

1.4.1 Date 클래스의 변수 타입

`as.Date(x, format, tryFormats = c("%Y-%m-%d", "%Y/%m/%d"), optional = FALSE, ...)`

format에 사용되는 코드

code	의미
%d	일(day) : (1부터 31까지의 값) 01-31
%D	시스템 날짜 형식 (mm/dd/yy)
%a	축약형 weekday : (Mon,···)
%A	축약되지 않은 weekday : (Monday,..)
%m	월(month) : (1부터 12까지의 값) 01-12
%b	축약형 월 : (Jan,···)
%B	축약되지 않은 월 : (January,..)
%y	두 자리수 연도 : (19,20,···)
%Y	네 자리수 연도 : (2019,2020,···)

```
library(magrittr)
as.Date("2014-01-24", format = "%Y-%m-%d") -> date01
date01
```

```
[1] "2014-01-24"
```

```
class(date01)
```

```
[1] "Date"
```

```
typeof(date01)
```

```
[1] "double"
```

```
x <- c("May 6 2018", "Aug 20 2019")
as.Date(x, format="%b %d %Y")  ## error 발생
```

```
[1] NA NA
```

```
format(Sys.Date(), format = "%b %B %d %D %Y %y %A")
```

```
[1] "12 12월  22 12/22/24 2024 24 일요일"
```

```
x <- c("5 6 2018", "8 20 2019")
as.Date(x, format="%b %d %Y")
```

```
[1] "2018-05-06" "2019-08-20"
```

```r
x <- c("20180506", "20190820")
as.Date(x, format="%Y%m%d")
```

```
[1] "2018-05-06" "2019-08-20"
```

```r
date02 <- as.Date(x, format="%Y%m%d")
date02[,2] - date02[,1] ## error
```

```
Error in `[.default`(date02, , 2): incorrect number of dimensions
```

```r
diffdays <- diff(date02, units = "days")
diffdays
```

```
Time difference of 471 days
```

```r
class(diffdays)
```

```
[1] "difftime"
```

```r
str(diffdays)
```

```
 'difftime' num 471
 - attr(*, "units")= chr "days"
```

```r
as.numeric(diffdays)
```

```
[1] 471
```

```r
diff(date02, units = "month") ## error
```

```
Error in match.arg(dots$units, choices = setdiff(eval(formals(difftime)$units), : 'arg' should be one of "
```

```r
##units should be one of “secs”, “mins”, “hours”, “days”, “weeks”
as.numeric(diffdays)/365.24219 *12
```

```
[1] 15.47466354
```

1.4.2 POSIX class

POSIXlt, POSIXct

뒤에 있는 “lt”는 list time, “ct”는 continuous time이고 정리하면 POSIX + lt -> POSIXlt, POSIX + ct -> POSIXct

```r
time.list <- as.POSIXlt(Sys.time())
time.list
```

```
[1] "2024-12-22 08:59:02 KST"
```

```
  unlist(time.list)
```

```
               sec                min               hour               mday
"2.64352989196777"               "59"                "8"               "22"
               mon               year               wday               yday
              "11"              "124"                "0"              "356"
             isdst               zone             gmtoff
               "0"              "KST"            "32400"
```

```
  unlist(time.list)%>%t%>%t
```

```
       [,1]
sec    "2.64352989196777"
min    "59"
hour   "8"
mday   "22"
mon    "11"
year   "124"
wday   "0"
yday   "356"
isdst  "0"
zone   "KST"
gmtoff "32400"
```

```
  class(time.list)
```

```
[1] "POSIXlt" "POSIXt"
```

```
  str(time.list)
```

```
 POSIXlt[1:1], format: "2024-12-22 08:59:02"
```

```
  time.cont <- as.POSIXct(Sys.time())
  time.cont
```

```
[1] "2024-12-22 08:59:02 KST"
```

```
  class(time.cont)
```

```
[1] "POSIXct" "POSIXt"
```

```
  str(time.cont)
```

```
 POSIXct[1:1], format: "2024-12-22 08:59:02"
```

POSIX.lt 내용을 살펴보면,

- sec : 그냥 초
- min : 그냥 분
- hour : 그냥 시간
- mday: 1일을 1로
- mon: 1월을 0으로
- year : 1900년을 0으로
- wday: 일요일이 0으로
- yday : 1월 1일이 0으로
- isdst : 서머타임

```
diff(c(time.list, Sys.time() ))
```

```
Time difference of 0.02681708336 secs
```

```
## units should be one of "secs", "mins", "hours", "days", "weeks"
```

원래의 시간에 시간을 더하거나 빼는 것은 어떻게 하나?
R에서는 시간이 전부 초로 계산됨

```
Sys.time() + 3600*24*15  ## 오늘 날짜에 15일을 더함
```

```
[1] "2025-01-06 08:59:02 KST"
```

```
library(lubridate)
Sys.time() + days(15)
```

```
[1] "2025-01-06 08:59:02 KST"
```

1.4.3 Lubridate package

lubridate라는 패키지를 사용하면 시간 연산이 편리함 "period" class 사용

Order of elements in date-time	Parse function
year, month, day	ymd()
year, day, month	ydm()
month, day, year	mdy()
day, month, year	dmy()
hour, minute	hm()
hour, minute, second	hms()
year, month, day, hour, minute, second	ymd_hms()

```
library(lubridate)
ymd("20110604")
```

```
[1] "2011-06-04"
```

```
mdy("06-04-2011")
```

```
[1] "2011-06-04"
```

```
dmy("04/06/2011")
```

```
[1] "2011-06-04"
```

```
ymd(20240105)
```

```
[1] "2024-01-05"
```

```
#### period 는 연산이 가능함
period(second = 3, minute = 1, hour = 2, day = 13, week = 1)-days(1)
```

```
[1] "19d 2H 1M 3S"
```

```
dmy("04/06/2011") + period(second = 3, minute = 1, hour = 2, day = 13, week = 1)
```

```
[1] "2011-06-24 02:01:03 UTC"
```

```
#### _h, _hm, _hms 등을 붙이면 시간:분:초 까지 시간을 만들 수 있다.
arrive <- ymd_hms("2011-06-04 12:00:00", tz = Sys.timezone(location = TRUE))
arrive
```

```
[1] "2011-06-04 12:00:00 KST"
```

```
  wday(arrive)
```

```
[1] 7
```

```
  wday(arrive, label = TRUE)
```

```
[1] 토
Levels: 일 < 월 < 화 < 수 < 목 < 금 < 토
```

```
  arrive %>% class
```

```
[1] "POSIXct" "POSIXt"
```

```
  ymd_hms("2011-06-04 12:00:00") ### time zone을 설정하지 않으면 오류가 생길 수 있다
```

```
[1] "2011-06-04 12:00:00 UTC"
```

```
  ### "UTC" = GMT Coordinated Universal Time
  Sys.timezone(location = TRUE) # 현재 사용하는 시스템의 time zone
```

```
[1] "Asia/Seoul"
```

```
  #### OlsonNames() : 사용가능한 모든 time zone의 이름들
  now("GMT")
```

```
[1] "2024-12-21 23:59:02 GMT"
```

```
  leave <- ymd_hms("2024-11-30 18:30:00", tz="Asia/Seoul")
  arrive <- ymd_hms("2024-11-30 12:20:00", tz="US/Pacific")
  difftime(arrive,leave)
```

```
Time difference of 10.83333333 hours
```

```
  ### 시간 더하기
  now() + minutes(5)
```

```
[1] "2024-12-22 09:04:02 KST"
```

```
  now() + months(-5)
```

```
[1] "2024-07-22 08:59:02 KST"
```

```
  now() + years(5)
```

```
[1] "2029-12-22 08:59:02 KST"
```

```
  jan31 <- ymd("2013-01-31")
  jan31 + months(0:11)
```

```
 [1] "2013-01-31" NA           "2013-03-31" NA           "2013-05-31"
 [6] NA           "2013-07-31" "2013-08-31" NA           "2013-10-31"
[11] NA           "2013-12-31"
```

Date component	Accessor
Year	year()
Month	month()
Week	week()
Day of year	yday()
Day of month	mday()
Day of week	wday()
Hour	hour()
Minute	minute()
Second	second()
Time zone	tz()

```
  library(readxl)
  dir(pattern = "*.xlsx")
```

```
[1] "datasummary.xlsx"    "operation.xlsx"      "patients.xlsx"
[4] "폐암-항암.xlsx"      "폐암 -환자정보.xlsx" "폐암 op.xlsx"
[7] "폐암 RT.xlsx"        "폐암환자 통계.xlsx"
```

```
  opdata <- read_xlsx(dir(pattern = "*.xlsx")[2], sheet = 2)
  ptdata <- read_xlsx(dir(pattern = "*.xlsx")[3], sheet = 1)
  str(opdata)
```

```
tibble [2,657 x 2] (S3: tbl_df/tbl/data.frame)
 $ 등록정보: chr [1:2657] "00679325" "00249356" "00251488" "00250179" ...
 $ 수술일   : chr [1:2657] "1993-08-19" "1982-06-02" "1982-06-22" "1982-06-08" ...
```

```
  str(ptdata)
```

```
tibble [16,921 x 5] (S3: tbl_df/tbl/data.frame)
 $ 등록번호   : chr [1:16921] "00000000" "00000001" "00000039" "00000103" ...
 $ 성별       : chr [1:16921] "M" "F" "F" "M" ...
 $ 나이       : chr [1:16921] NA "48" "54" "-29" ...
 $ 진단일     : chr [1:16921] NA "1963-12-26" "1963-11-14" "1963-06-01" ...
 $ 최종추적일: chr [1:16921] NA "1999-07-29" "1964-11-14" "1964-06-01" ...
```

```r
  opdf <- merge(ptdata,opdata,by.x="등록번호", by.y="등록정보", all=FALSE )
  str(opdf)
```

```
'data.frame':   2654 obs. of  6 variables:
 $ 등록번호   : chr  "00000636" "00001671" "00002686" "00002793" ...
 $ 성별       : chr  "M" "M" "M" "M" ...
 $ 나이       : chr  "39" "48" "41" "48" ...
 $ 진단일     : chr  "1964-05-19" "1964-02-04" "1965-09-29" "1965-08-23" ...
 $ 최종추적일: chr  "1964-05-19" "1965-10-11" "1966-09-29" "1966-08-23" ...
 $ 수술일     : chr  "1964-05-22" "1964-02-04" "1965-09-21" "1965-08-23" ...
```

```r
  colnames(opdf) <- c("chartno","sex","age","dxdate", "fudate", "opdate")
```

```r
  opdf2 <- opdf
  colnames(opdf2)
```

```
[1] "chartno" "sex"     "age"     "dxdate"  "fudate"  "opdate"
```

```r
  date.varname <- grep( "date",colnames(opdf2) , value=T )
  date.varname
```

```
[1] "dxdate" "fudate" "opdate"
```

```r
  for (c in date.varname) {
    opdf2[,c] <- as.POSIXct(opdf2[,c], format="%Y-%m-%d",
                                tz=  "Asia/Seoul" )
  }
  str(opdf2)
```

```
'data.frame':   2654 obs. of  6 variables:
 $ chartno: chr  "00000636" "00001671" "00002686" "00002793" ...
 $ sex    : chr  "M" "M" "M" "M" ...
 $ age    : chr  "39" "48" "41" "48" ...
 $ dxdate : POSIXct, format: "1964-05-19" "1964-02-04" ...
 $ fudate : POSIXct, format: "1964-05-19" "1965-10-11" ...
 $ opdate : POSIXct, format: "1964-05-22" "1964-02-04" ...
```

```r
  head(opdf2$fudate)
```

```
[1] "1964-05-19 KST" "1965-10-11 KST" "1966-09-29 KST" "1966-08-23 KST"
[5] "1965-12-27 KST" "1967-01-25 KST"
```

```r
  opdf2$surv.period <- opdf2$fudate - opdf2$opdate
  str(opdf2$surv.period)
```

```
 'difftime' num [1:2654] -259200 53136000 32227200 31536000 ...
 - attr(*, "units")= chr "secs"
```

```
  opdf2$surv.period <- difftime(opdf2$fudate, opdf2$opdate, units = "days")
  str(opdf2$surv.period)
```

```
 'difftime' num [1:2654] -3 615 373 365 ...
 - attr(*, "units")= chr "days"
```

```
  opdf2$survival.month <- opdf2$surv.period /365.24219 *12
  attr(opdf2$survival.month, "units") <- "months"
  str(opdf2)
```

```
'data.frame':   2654 obs. of  8 variables:
 $ chartno       : chr  "00000636" "00001671" "00002686" "00002793" ...
 $ sex           : chr  "M" "M" "M" "M" ...
 $ age           : chr  "39" "48" "41" "48" ...
 $ dxdate        : POSIXct, format: "1964-05-19" "1964-02-04" ...
 $ fudate        : POSIXct, format: "1964-05-19" "1965-10-11" ...
 $ opdate        : POSIXct, format: "1964-05-22" "1964-02-04" ...
 $ surv.period   : 'difftime' num  -3 615 373 365 ...
  ..- attr(*, "units")= chr "days"
 $ survival.month: 'difftime' num  -0.0985647359085214 20.2057708612469 12.2548821646262 11.9920428688701
  ..- attr(*, "units")= chr "months"
```

1.4.4 ID 주민번호에서 생일 추출하기

주민번호는 연도, 월, 일 두자리씩 + 1또는2는 1900년대 남녀, 3또는4는 2000년대 남녀 구분

Regular expression으로 정상적인 ID 인지 체크하고 날짜 형식으로 변형

```
  library(lubridate)
  library(stringr)
  IDno <- "120923-3012123"
  patternid <- "(\\d{2})([01]\\d)(([0-2]\\d)|(3[01]))-([1-4])"
  match_id <- str_match(IDno, patternid)
  match_id
```

```
     [,1]       [,2] [,3] [,4] [,5] [,6] [,7]
[1,] "120923-3" "12" "09" "23" "23" NA   "3"
```

```
  ymd_str <-paste(match_id[,2:4], collapse = "" )
  c_str <- ifelse(match_id[,7]<"3","19","20")
  as.Date(paste(c_str,ymd_str,sep = ""), format = "%Y%m%d", tz=  "Asia/Seoul" )
```

```
[1] "2012-09-23"
```

1.4.5 PFT report에서 검사날짜와 검사치 추출하기

```
library(tesseract)
setwd(paste(getwd(),"/TMP", sep=""))
setwd("G:/R project/nomogram/TMP")
jpgfiles <- dir(pattern = ".jpg")
jpgfiles
```

```
 [1] "00872290_20120803_2001_F6001_001.jpg"
 [2] "00872290_20150511_2024_F60021_001.jpg"
 [3] "00872290_20170324_2022_F60021_001.jpg"
 [4] "00872290_20200519_2019_F60021_001.jpg"
 [5] "00872290_20210601_2016_F60021_001.jpg"
 [6] "00872290_20220329_2004_F6002_001.jpg"
 [7] "00872290_20241121_2014_F60021_001.jpg"
 [8] "01037742_20100810_2001_F6001_001.jpg"
 [9] "01037742_20110512_2015_F6001_001.jpg"
[10] "01043414_20240926_2014_F60021_001.jpg"
[11] "01220463_20180626_2010_F6002_001.jpg"
[12] "01220463_20200326_2032_F60021_001.jpg"
[13] "01220463_20210506_2006_F60021_001.jpg"
[14] "01220463_20230615_2003_F60021_001.jpg"
[15] "01320646_20241118_2007_F60021_001.jpg"
[16] "01328776_20241115_2002_F60021_001.jpg"
[17] "01345136_20241031_2013_F6010_001.jpg"
[18] "01345136_20241031_2014_F60021_001.jpg"
[19] "01367827_20241101_2014_F60021_001.jpg"
```

```
pfttext <- NULL
for (files in jpgfiles) {
  pfttext <- c(pfttext, ocr(files, engine = tesseract("kor+eng") ))
}
setwd("G:/R project/nomogram")
str(pfttext)
```

```
chr [1:19] "Name: 이 금 주 , 10: 00872290\nGender: Female Room: C7 Date: 08/03/12\nAge: 51 Race: Asian Te
```

```
pattern1 <- "(\\d{2}/\\d{2}/\\d{2})|(\\d{2,4}[ -]\\d{2}[ -]\\d{2})"
match_date0 <- str_match(pfttext, pattern1)
match_date0
```

```
     [,1]         [,2]       [,3]
[1,] "08/03/12"   "08/03/12" NA
[2,] "05/11/15"   "05/11/15" NA
[3,] "03/27/17"   "03/27/17" NA
[4,] "05/19/20"   "05/19/20" NA
```

```
 [5,] "06/01/21"   "06/01/21" NA
 [6,] "03/29/22"   "03/29/22" NA
 [7,] "2024-11-21" NA         "2024-11-21"
 [8,] "2010 08 10" NA         "2010 08 10"
 [9,] "2011 05 12" NA         "2011 05 12"
[10,] "2024-10-29" NA         "2024-10-29"
[11,] "06/26/18"   "06/26/18" NA
[12,] "03/27/20"   "03/27/20" NA
[13,] "08/12/21"   "08/12/21" NA
[14,] "2023-12-21" NA         "2023-12-21"
[15,] "2024-12-05" NA         "2024-12-05"
[16,] "2024-11-15" NA         "2024-11-15"
[17,] "2024-11-06" NA         "2024-11-06"
[18,] "2024-11-06" NA         "2024-11-06"
[19,] "2024-11-01" NA         "2024-11-01"
```

```
match_date1 <- as.POSIXct(match_date0[,1], tz="Asia/Seoul", "%Y-%m-%d")
match_date2 <- as.POSIXct(match_date0[,1], tz="Asia/Seoul", "%Y %m %d")
match_date3 <- as.POSIXct(match_date0[,1], tz="Asia/Seoul", "%m/%d/%y")
match_date <- cbind(match_date1,match_date2,match_date3)
match_date <- apply(match_date, MARGIN = 1, function(x) {sum(x,na.rm = TRUE)})
match_date <- as.POSIXct(match_date)
match_date
```

```
 [1] "2012-08-03 KST" "2015-05-11 KST" "2017-03-27 KST" "2020-05-19 KST"
 [5] "2021-06-01 KST" "2022-03-29 KST" "2024-11-21 KST" "2010-08-10 KST"
 [9] "2011-05-12 KST" "2024-10-29 KST" "2018-06-26 KST" "2020-03-27 KST"
[13] "2021-08-12 KST" "2023-12-21 KST" "2024-12-05 KST" "2024-11-15 KST"
[17] "2024-11-06 KST" "2024-11-06 KST" "2024-11-01 KST"
```

```
pattern_fev1 <- "FEV1[ ]+[\\w\\W]{3,6}[ ]+\\d?[\\d.]\\d{0,2}[ ]+(\\d?[\\d.]\\d{0,2})[ ]+(\\d?[\\d.]\\d{0
str_match(pfttext, pattern_fev1)
```

```
      [,1]                        [,2]   [,3]
 [1,] "FEV1 Liters 2.40 2.17 90"  "2.17" "90"
 [2,] "FEV1 Liters 2.29 253 110"  "253"  "110"
 [3,] "FEV1 Liters 2.23 2.39 107" "2.39" "107"
 [4,] "FEV1 Liters 2.13 2.07 97"  "2.07" "97"
 [5,] "FEV1 Liters 2.14 1.96 92"  "1.96" "92"
 [6,] "FEV1 Liters 2.10 1.97 94"  "1.97" "94"
 [7,] "FEV1 [L] 2.06 1.92 93"     "1.92" "93"
 [8,] "FEV1 Liters 251 3.16 126"  "3.16" "126"
 [9,] "FEV1 Liters 251 293 116"   "293"  "116"
[10,] "FEV1 [L] 1.69 1.97 116"    "1.97" "116"
[11,] "FEV1 Liters 2.79 3.12 112" "3.12" "112"
[12,] "FEV1 Liters 2.65 2.98 113" "2.98" "113"
[13,] "FEV1 Liters 2.59 2.76 107" "2.76" "107"
```

```
[14,] "FEV1 (L) 2.53 2.79 111"    "2.79" "111"
[15,] "FEV1 [L] 2.65 2.17 82"     "2.17" "82"
[16,] "FEV1 [L] 1.43 0.85 59"     "0.85" "59"
[17,] NA                          NA     NA
[18,] "FEV1 [ 니 2.89 2.11 73"    "2.11" "73"
[19,] "FEV1 [L] 2.34 2.80 120"    "2.80" "120"
```

```
match_fev1 <- str_match(pfttext, pattern_fev1)[,2]
match_fev1p <- str_match(pfttext, pattern_fev1)[,3]
pattern_chartno <- "\\d{8}"
match_chartno <- str_match(pfttext ,pattern_chartno)
df <- data.frame(chartno=match_chartno, date = match_date, fev1 = match_fev1, fev1p = match_fev1p)
df$fev1 <- as.numeric(df$fev1)
df$fev1p <- as.numeric(df$fev1p)
df$fev1[df$fev1>10&!is.na(df$fev1)] <- df$fev1[df$fev1>10&!is.na(df$fev1)]/100
df
```

```
    chartno       date fev1 fev1p
1  00872290 2012-08-03 2.17    90
2  00872290 2015-05-11 2.53   110
3  00872290 2017-03-27 2.39   107
4  00872290 2020-05-19 2.07    97
5  00872290 2021-06-01 1.96    92
6  00872290 2022-03-29 1.97    94
7  00872290 2024-11-21 1.92    93
8  01037742 2010-08-10 3.16   126
9  01037742 2011-05-12 2.93   116
10 01043414 2024-10-29 1.97   116
11 01220463 2018-06-26 3.12   112
12 01220463 2020-03-27 2.98   113
13 01220463 2021-08-12 2.76   107
14 01220463 2023-12-21 2.79   111
15 01320646 2024-12-05 2.17    82
16 01328776 2024-11-15 0.85    59
17 01345136 2024-11-06   NA    NA
18 01345136 2024-11-06 2.11    73
19 01367827 2024-11-01 2.80   120
```

1.4.6 Interval class

lubridate package에 있는 함수

```
interval_fu <- interval(opdf2$opdate, opdf2$fudate)
(test_interval <- interval_fu[1000])
```

```
[1] 1997-08-13 KST--2000-09-14 KST
```

```r
  int_start(test_interval)
```

```
[1] "1997-08-13 KST"
```

```r
  int_end(test_interval)
```

```
[1] "2000-09-14 KST"
```

```r
  int <- interval(ymd("2001-01-01"), ymd("2002-01-01"))
  int_length(int)
```

```
[1] 31536000
```

```r
  int <- interval(ymd("2001-01-01"), ymd("2002-01-01"))
  int_flip(int)
```

```
[1] 2002-01-01 UTC--2001-01-01 UTC
```

```r
  int <- interval(ymd("2001-01-01"), ymd("2002-01-01"))
  int_shift(int, duration(days = 11))
```

```
[1] 2001-01-12 UTC--2002-01-12 UTC
```

```r
  int_shift(int, duration(hours = -1))
```

```
[1] 2000-12-31 23:00:00 UTC--2001-12-31 23:00:00 UTC
```

```r
  int1 <- interval(ymd("2001-01-01"), ymd("2002-01-01"))
  int2 <- interval(ymd("2001-06-01"), ymd("2002-06-01"))
  int3 <- interval(ymd("2003-01-01"), ymd("2004-01-01"))

  int_overlaps(int1, int2) # TRUE
```

```
[1] TRUE
```

```r
  int_overlaps(int1, int3) # FALSE
```

```
[1] FALSE
```

2 Advanced Techniques

2.1 Data Manipulation

2.1.1 Data reading

data file이 존재하는 디렉토리를 먼저 설정해주어야 한다. 이를 위한 명령어는 setwd() = set working directory 라는 의미 setwd("G:/R project/nomogram") 와 같이 디렉토리를 설정해줄 수도 있지만,

만약 디렉토리를 찾기 어렵다면 setwd(choose.dir()) 와 같은 명령으로 파일탐색기를 열어서 디렉토리를 선택할 수 있다. 현재 사용할 xlsx 파일들이 다음 디렉토리에 있다고 가정하자

```
setwd("G:/R project/nomogram")
library(readxl)
dir(pattern = "*.xls")
```

```
 [1] "datasummary.xlsx"    "operation.xlsx"      "Patient_info.xls"
 [4] "patients.xlsx"       "Survdata.xls"        "폐암-항암.xlsx"
 [7] "폐암 -환자정보.xlsx" "폐암 op.xlsx"       "폐암 RT.xlsx"
[10] "폐암_OP.xls"         "폐암_RT.xls"         "폐암_추적조사.xls"
[13] "폐암_항암치료.xls"   "폐암환자 통계.xlsx"
```

```
xlsxfiles <- dir(pattern = "*.xls")
ptinfo <- read_xls(xlsxfiles[5])
```

2.1.2 Binding tables

데이터프레임 결합 방법들

rbind(), cbind(), merge()

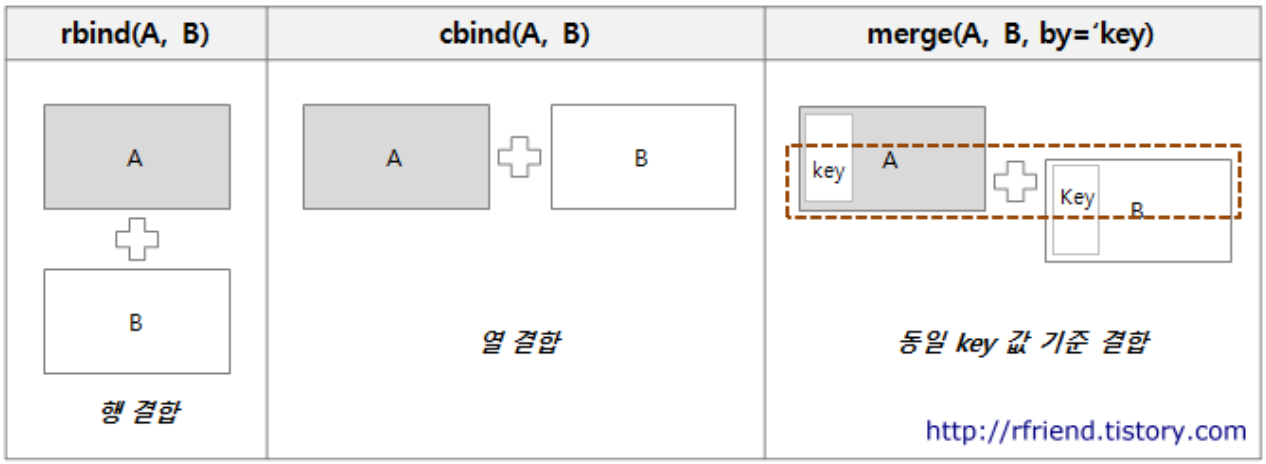


Figure 1: 데이터프레임 결합방법

** 당연한 이야기지만 rbind는 컬럼의 갯수가 같아야 하고, cbind는 행의 갯수가 같아야 함

2.1.3 Join (Merge) tables

merge function

merge(x, y, by = intersect(names(x), names(y)), ## 공통된 컬럼하나를 결합용 키로 선택

by.x = by, by.y = by, all = FALSE, all.x = all, all.y = all, ## x와 y의 결합용 키의 이름이 서로 다를 경우에는 독립적으로 지정

sort = TRUE, suffixes = c(".x",".y"), no.dups = TRUE,

incomparables = NULL, ···)

```
df1 <- data.frame( ID = 1:10, Name = c("Lee","Kim","Park", "Kang",
                   "Shin", "Lim", "Kwon", "Choi", "Nam", "Baek" ),
                   Score = as.integer(rnorm(10, 80,6 ))
                   )
df2 <- data.frame( ID = sample(1:10, 9, replace = F),
                   Department = sample( c("IM","GS","GY","PD" ),9, replace = T),
                   Age = as.integer(rnorm(9, 40,6 )) )
df1
```

```
   ID Name Score
1   1  Lee    76
2   2  Kim    78
3   3 Park    76
4   4 Kang    77
5   5 Shin    77
6   6  Lim    74
7   7 Kwon    89
8   8 Choi    84
9   9  Nam    87
10 10 Baek    82
```

```
df2
```

```
  ID Department Age
1  3         GS  27
2  8         GY  32
3 10         IM  46
4  1         PD  40
5  5         PD  43
6  7         GY  39
7  6         IM  41
8  9         PD  43
9  2         GY  46
```

```
merged_df <- merge(df1,df2, by="ID", all = TRUE)  # full join
merged_df
```

```
   ID Name Score Department Age
1   1  Lee    76         PD  40
2   2  Kim    78         GY  46
3   3 Park    76         GS  27
4   4 Kang    77       <NA>  NA
5   5 Shin    77         PD  43
6   6  Lim    74         IM  41
7   7 Kwon    89         GY  39
8   8 Choi    84         GY  32
9   9  Nam    87         PD  43
10 10 Baek    82         IM  46
```

2.1.4 Types of Join

merge 함수를 실행하여 데이터를 결합할 때에는 데이터 join 방법이 다음과 같이 4가지가 있다.

두개의 df에서 모든 데이터가 완전하게 존재하지 않기 때문에 일치하지 않는 부분에 대한 처리규칙이 중요하다.

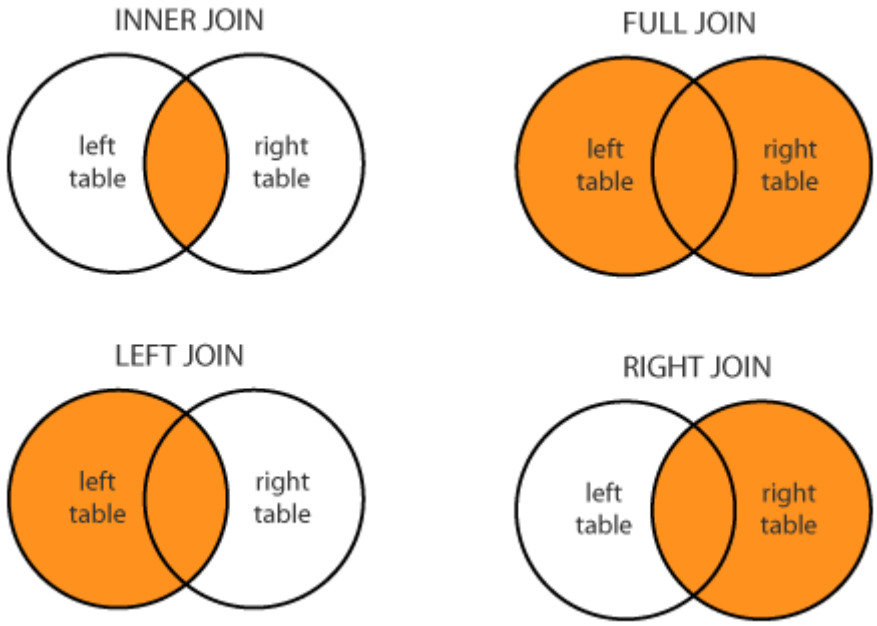


Figure 2: Types of Join

merge 함수의 옵션에서 all = TRUE 를 선택하면 full join, all.x 는 left join, all.y는 right join이 된다.

all= FALSE 인 경우에는 당연히 inner join

dplyr package에는 개별적인 join 함수가 있는데 그것을 사용해도 됨

inner_join(df1, df2), left_join(df1, df2), right_join(df1, df2), full_join(df1, df2)

left_join(df2, df1) : alternative right join

2.1.5 Reshape data

2.2 Pipeline operator

library magrittr 를 사용하면 pipeline 연산자를 쓸 수 있게 된다. %>% 형식이다.

만약 c("A","B","C") 라는 데이터를 "ABC" 로 paste 한 다음에 다시 tolower 함수를 적용하여 "abc"로 변환하는 작업을 한다고 하자. 그런 경우에는 다음과 같이 코딩을 해야 한다. 하지만 pipeline operator를 사용하면 함수 중첩을 줄이고 코드를 이해하기 쉽게 사용할 수 있다.

```
library(magrittr)
tolower(paste(c("A","B","C"), collapse = ""))
```

```
[1] "abc"
```

```
c("A","B","C") %>% paste(., collapse = "") %>% tolower  ## paste 함수에는 여러 인자가 들어가는데
```

```
[1] "abc"
```

```
## 첫번째 인자로 들어가기 위해서 . 을 사용함
```

2.3 Regular Expression

정규표현식의 정의 : 특정 문자열을 가리키는 패턴

아주 간단하고 쉬운 정규표현식의 한가지 예를 들어보면

```
\d
```

이것은 0부터 9까지의 숫자와 매칭된다. 그리고 다음처럼 제법 복잡하게 발전시킬 수도 있다.

```
^0\d{1,2}-?\d{2,4}-?\d{4}
```

이 패턴은 한국의 전화번호에 대응하는 패턴인데 0으로 시작하는 핸드폰번호 또는 지역번호 - 0으로 시작하지 않는 2~4자리 번호 - 4자리 번호와 매칭이 되는 패턴이다.

2.3.1 RegexPal 연습하기

http://www.regexpal.com

이 웹사이트에 들어가면 위에 정규표현식 입력영역이 있고 아래에 목표 텍스트 영역이 있다.

https://regexr.com/ : 설명이 같이 있어서 복잡한 패턴을 공부하는데 도움이 된다.

일단 RegexPal로 숫자 패턴 연습을 해보자. 아래 영역에는 다양한 전화번호 또는 여러가지 번호나 텍스트를 붙여넣어놓자

위의 영역을 지우고 3이라는 숫자만 입력해보자

[0-9] 라고 입력해보자

숫자의 범위를 제한해보자 0,1,2,3,5,8 의 번호만 찾고 싶으면 [012358] 을 입력한다

11개의 숫자를 찾는 방법은? [0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]

[0-9] 표현은 다음과 같은 단축문자 로 바꿀 수 있다.

```
\d
```

다음과 같이 전화번호를 찾아볼 수 있다

```
\d\d\d-\d\d\d\d-\d\d\d\d
```

숫자 이외의 문자를 찾는데 쓰는 단축문자를 쓰면..

```
\d\d\d\D\d\d\d\d\D\d\d\d\d
```

```
\d\d\d.\d\d\d\d.\d\d\d\d
```

점 (.) 을 사용하면 대부분의 아무 문자나 찾을 수 있는 와일드카드 역할을 한다. (개행문자 제외)

수량자를 사용해보자

```
\d{3}-?\d{3}-?\d{4}
```

{3} 중괄호 안의 숫자는 몇번 나오는지 숫자를 지정한다. {3,9}는 3~9회, {2,}는 2회 이상, {,99}는 99회 이하, ..

물음표(?) 도 수량자인데 한개 이하를 의미 (0 또는 1)

덧셈기호 (+)는 하나 이상

별표 (*)는 0 이상을 의미

```
(\d{3,4}[.-]?)+
```

위의 정규식은 숫자 세개 또는 네개 다음에 점이나 하이픈이 한번 이하로 등장하는 패턴을 묶은 그룹이 한번 이상 등장한다는 뜻이다. (괄호는 참조그룹의 의미)

정규표현식 기호들 (1)

표현	설명
*	0개 또는 그 이상 예) x* = x가 0번또는 그 이상 반복
+	1개 또는 그 이상 예) x+
?	0 또는 1개 (1개 이하) 예) x?
.	무엇이든 한글자
^	시작 문자를 지정, 예) ^[abc] = abc중 한글자 포함해서 시작
[^]	해당 문자를 제외한 모든 것, 예) [^abc] = a,b,c만 빼고 모두
$	끝 문자 예) x$ = x로 종료
[가-힣]	모든 한글 글자 중 1개
|	또는
()	소괄호에 묶인 문자는 한개의 그룹으로 표현되거나 and로 됨

정규표현식 기호들 (2)

\d - 숫자와 매치, [0-9]와 동일한 표현식이다.

\D - 숫자가 아닌 것과 매치, [^0-9]와 동일한 표현식

\s - whitespace 문자와 매치, [\t\n\r\f\v]와 동일한 표현식이다.
맨 앞의 빈 칸은 공백문자(space)를 의미

\S - whitespace 문자가 아닌 것과 매치, [^\t\n\r\f\v]와 동일한 표현식

\w - 문자+숫자(alphanumeric)와 매치, [a-zA-Z0-9_]와 동일한 표현식

\W - 문자+숫자(alphanumeric)가 아닌 문자와 매치, [^a-zA-Z0-9_]와 동일한 표현식

\n - 줄바꿈 문자

\t - 탭 문자

\v - 수직 탭 문자

\b - 단어 경계 : 백스페이스는 반드시 [\b]로 입력해야 한다.

\c - 제어 문자

\f - 폼 피드 문자

\x - 16진수값

\0 - 8진수 값

특수표현 : "[: :]" 형태로 존재하는 특수한 표현식이 존재

[::]	의미
[:digit:]	숫자
[:alpha:]	문자
[:lower:]	소문자
[:upper:]	대문자
[:alnum:]	문자＋숫자
[:punct:]	기호
[:graph:]	문자＋숫자＋기호
[:space:]	띄어쓰기
[:blank:]	띄어쓰기＋탭

이어지거나 이어지지 않는 문자열

이어지는 문자열 (?=)

- 가나다(?=a) : 뒤에 “a”가 있는 “가나다”

이어지지 않는 문자열 (?!=)

- 가나다(?!=b) : 뒤에 “b”로 이어지지 않는 “가나다”

앞에서 이어지는 문자열 (?<=)

- (?<=c) 가나다 : 앞에 “c”가 있는 “가나다”

앞에서 이어지 않는 문자열 (?<!)

- (?<!c) 가나다 : 앞에 “c”가 없는 “가나다”

E-mail

```
^[0-9a-zA-Z]([-_\.]?[0-9a-zA-Z])*@[0-9a-zA-Z]([-_\.]?[0-9a-zA-Z])*\.[a-zA-Z]{2,3}$
```

‘시작을’ 0~9 사이 숫자 or a-z A-Z 아무거나로 시작하고
중간에 - _ . 같은 문자가 있을 수도 있고 없을 수도 있으며
그 후에 0~9 사이 숫자 or a-z A-Z 중 하나의 문자가 없거나 연달아 나올수 있으며
@ 가 반드시 존재하고
0-9a-zA-Z 여기서 하나가 있고
중간에 - _ . 같은 문자가 있을수도 있고 없을수도 있으며,
그 후에 0~9 사이 숫자 or a-z A-Z 중 하나의 문자가 없거나 연달아 나올수 있으며
반드시 . 이 존재하고, [a-zA-Z] 의 문자가 2 개나 3 개가 존재하면서 종료

탐욕적 및 게으른 수량자

정규 표현식에서는 일치하는 패턴을 찾는 횟수 제한이 없어 필요 이상의 상황을 연출하기도 하는데 이것은 의도적으로 수량자를 탐욕적으로 만들었기 때문이다. 문법에서 말하는 탐욕적 수량자(Greedy Quantifier)란 가능하면 가장 큰 덩어리를 찾는다는 뜻이다. 반대의 개념인 게으른 수량자(Lazy Quantifier)는 패턴에 근접하는 최소한의 덩어리를 찾는다.

- 탐욕적 수량자: `*`, `+`, `{n,}`
- 게으른 수량자: `*?`, `+?`, `{n,}?`

실제 사용례 - Pathology report

병리결과지에서 추출

`\d{8}[\w\W]*?(?=\d{8}|$)`

8개의 숫자(병록번호)로 시작하는 텍스트 …… 8개의 병록번호 전까지 또는 마지막 전까지 모든 문자 선택

2.3.2 R에서 사용하는 함수들

grep

정규표현식을 사용해 패턴에 일치하는 것을 찾아오는 함수

grep(pattern, x, ignore.case = FALSE, perl = FALSE, value = FALSE, fixed = FALSE, useBytes = FALSE, invert = FALSE)

pattern : 정규표현식

x : 데이터

해당 인덱스로 찾아주기도 하고

```
text <- c('a','ab','acb','accb','acccb','accccb')
grep('a',text)
```

```
[1] 1 2 3 4 5 6
```

```
grep('acb',text)
```

```
[1] 3
```

해당 값으로 찾아주기도 함

```
grep('ac?b',text, value=T)
```

```
[1] "ab"  "acb"
```

```
grep('ac{2,3}',text,value=T)
```

```
[1] "accb"   "acccb"  "accccb"
```

gsub

문자열을 바꿔주는 함수

gsub(pattern, replacement, x, ignore.case = FALSE, perl = FALSE, fixed = FALSE, useBytes = FALSE)

regexpr, gregexpr(regexpr의 global 버전)

grep()과 grepl()의 한계점 보완: 특정 문자 패턴의 일치여부에 대한 정보를 제공하지만 위치 및 정규식의 일치 여부를 알려주지는 않음

```
regexpr(pattern, text, ignore.case = FALSE, perl = FALSE,  fixed = FALSE, useBytes = FALSE)
```

```
x <- c("Darth Vader: If you only knew the power of the Dark Side.
       Obi-Wan never told you what happend to your father",
       "Luke: He told me enough! It was you who killed him!",
       "Darth Vader: No. I'm your father")
regexpr("you", x) # 각 x의 문자열에서 you가 처음 나타난 위치 및 길이 반환
```

```
[1] 17 33 22
attr(,"match.length")
[1] 3 3 3
attr(,"index.type")
[1] "chars"
attr(,"useBytes")
[1] TRUE
```

```
regexpr("father", x) # 패턴을 포함하지 않은 경우 -1 반환
```

```
[1] 111  -1  27
attr(,"match.length")
[1]  6 -1  6
attr(,"index.type")
[1] "chars"
attr(,"useBytes")
[1] TRUE
```

```
gregexpr("you", x) # 각 x의 문자열에서 you가 나타난 모든 위치 및 길이 반환
```

```
[[1]]
[1]  17  86 106
attr(,"match.length")
[1] 3 3 3
attr(,"index.type")
[1] "chars"
attr(,"useBytes")
[1] TRUE

[[2]]
```

```
[1] 33
attr(,"match.length")
[1] 3
attr(,"index.type")
[1] "chars"
attr(,"useBytes")
[1] TRUE

[[3]]
[1] 22
attr(,"match.length")
[1] 3
attr(,"index.type")
[1] "chars"
attr(,"useBytes")
[1] TRUE
```

2.3.3 stringr package

str_match

str_match(string, pattern) Return the first pattern match found in each string, as a matrix with a column for each () group in pattern. 일치하는 단어를 그룹별로 매트릭스 형식으로 돌려주기 때문에 실제 사용시에 편리함

stringr cheat sheet 참고

3 LaTeX codes in quarto

3.1 Basic LaTeX code

$$f(r) = \int_0^\infty e^{-\frac{x^2+y^2}{2}} dx$$

```
cat("$$ f(x)=e^{\\pi i} $$")
```

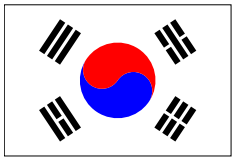
$$f(x) = e^{\pi i}$$

3.2 Tikz pictures : Latex의 벡터 그래픽 도구

3.2.1 태극기 그리기

```
\begin{tikzpicture}[very thin]
        \definecolor{t_red}{RGB}{199,32,50}
        \definecolor{t_blue}{RGB}{34,60,117}
        \definecolor{t_black}{RGB}{0,0,0}
        \definecolor{t_white}{RGB}{255,255,255}
        \fill [white] (-1.5,-1) rectangle (1.5,1);
%       \draw (0,0) circle[radius=0.5];
        \begin{scope}[rotate=-atan(2/3)]
        \fill [blue] (-0.5,0) arc[start angle=-180, end angle=0, radius=0.25]
            (0,0) arc[start angle=180, end angle=0, radius=0.25]
            (0.5,0) arc[start angle=0, end angle=-180, radius=0.5]--cycle;
        \fill [red] (0.5,0) arc[start angle=0, end angle=180, radius=0.25]
            (0,0) arc[start angle=0, end angle=-180, radius=0.25]
            (-0.5,0) arc[start angle=180, end angle=0, radius=0.5]--cycle;
        \end{scope}
        \begin{scope} [rotate=-atan(2/3)]
%       \draw (-1/2-1/3-1/4, -1/4) rectangle (-3/4,1/4);
        \fill [black] (-1/2-1/3-1/4, 1/4)
            -- ++(0,-1/2) -- ++(1/12,0) --++(0,1/2) ++(1/24,0)
            -- ++(0,-1/2) -- ++(1/12,0) --++(0,1/2) ++(1/24,0)
            -- ++(0,-1/2) -- ++(1/12,0) --++(0,1/2);
%       \draw (1/2+1/3+1/4, 1/4) rectangle (3/4,-1/4);
        \fill[black] (3/4, 1/4)
            --++(0,-11/48)--++(1/12,0)--++(0,+11/48)++(-1/12,-1/2)
            --++(0,11/48)--++(1/12,0)--++(0,-11/48) ++(1/24,1/2) % parted
            --++(0,-11/48)--++(1/12,0)--++(0,+11/48)++(-1/12,-1/2)
            --++(0,11/48)--++(1/12,0)--++(0,-11/48) ++(1/24,1/2)
            --++(0,-11/48)--++(1/12,0)--++(0,+11/48)++(-1/12,-1/2)
            --++(0,11/48)--++(1/12,0)--++(0,-11/48) ;
        \end{scope}
        \begin{scope} [rotate=atan(2/3)]
%       \draw (-1/2-1/3-1/4, 1/4) rectangle (-3/4,1/4);
        \fill [black] (-1/2-1/3-1/4, 1/4)
            -- ++(0,-1/2) -- ++(1/12,0) --++(0,1/2) ++(1/24,0)
            --++(0,-11/48)--++(1/12,0)--++(0,+11/48)++(-1/12,-1/2)
            --++(0,11/48)--++(1/12,0)--++(0,-11/48) ++(1/24,1/2)
            -- ++(0,-1/2) -- ++(1/12,0) --++(0,1/2)     ;
```

```
%       \draw (1/2+1/3+1/4, 1/4) rectangle (3/4,-1/4);
        \fill [black] (3/4, 1/4)
            --++(0,-11/48)--++(1/12,0)--++(0,+11/48)++(-1/12,-1/2)
            --++(0,11/48)--++(1/12,0)--++(0,-11/48) ++(1/24,1/2)
            -- ++(0,-1/2) -- ++(1/12,0) --++(0,1/2) ++(1/24,0)
            --++(0,-11/48)--++(1/12,0)--++(0,+11/48)++(-1/12,-1/2)
            --++(0,11/48)--++(1/12,0)--++(0,-11/48) ;
        \end{scope}
    \draw[black] (-1.5,-1) rectangle (1.5,1);
    \end{tikzpicture}
```



3.2.2 Tikz picture : membrane like surface

```
\begin{tikzpicture}
  \def\nuPi{3.1459265}
  \foreach \i in {11,10,...,0}{% This one doesn't matter
    \foreach \j in {5,4,...,0}{% This will crate a membrane
                               % with the front lipids visible
      % top layer
      \pgfmathsetmacro{\dx}{rand*0.1}% A random variance in the x coordinate
      \pgfmathsetmacro{\dy}{rand*0.1}% A random variance in the y coordinate,
                                     % gives a hight fill to the lipid
      \pgfmathsetmacro{\rot}{rand*0.1}% A random variance in the
                                      % molecule orientation
      \shade[ball color=red] ({\i+\dx+\rot},{0.5*\j+\dy+0.4*sin(\i*\nuPi*10)}) circle(0.45);
      \shade[ball color=gray] (\i+\dx,{0.5*\j+\dy+0.4*sin(\i*\nuPi*10)-0.9}) circle(0.45);
      \shade[ball color=gray] (\i+\dx-\rot,{0.5*\j+\dy+0.4*sin(\i*\nuPi*10)-1.8}) circle(0.45);
      % bottom layer
      \pgfmathsetmacro{\dx}{rand*0.1}
      \pgfmathsetmacro{\dy}{rand*0.1}
      \pgfmathsetmacro{\rot}{rand*0.1}
      \shade[ball color=gray] (\i+\dx+\rot,{0.5*\j+\dy+0.4*sin(\i*\nuPi*10)-2.8}) circle(0.45);
      \shade[ball color=gray] (\i+\dx,{0.5*\j+\dy+0.4*sin(\i*\nuPi*10)-3.7}) circle(0.45);
      \shade[ball color=red] (\i+\dx-\rot,{0.5*\j+\dy+0.4*sin(\i*\nuPi*10)-4.6}) circle(0.45);
    }
  }
\end{tikzpicture}
```