2장. 데이터 준비하기

박영식(youngsik.park@bsl-lausanne.ch)

날짜

◎ 시간의 흐름에 따라 데이터를 얻을 경우, 중요한 변수
- 생성: 문자형 벡터에 함수 as.Date ( ) 적용
- 문자형 벡터의 디폴트 형태: yyyy-mm-dd

```
> x <- as.Date(c("2010-01-01", " 2018-01-01"))
> x
[1] "2010-01-01" "2018-01-01"
```

◎ 유형 : 숫자형, 1970년 1월 1일부터의 날수

```
> typeof(x)
[1] "double"

> x[2]-x[1]
Time difference of 2922 day
```

행렬(Matrix) 및 배열(array)

◎ 행렬(2차원 구조) 및 배열(2차원 이상의 구조)
- 1차원 구조인 벡터에 dim(차원) 속성이 추가됨
- 구성요소는 같은 유형(숫자형, 문자형, 논리형)의 데이터

◎ 행렬을 생성하는 방법 1:
- 함수 matrix ( ) : 행과 열의 개수 nrow= 또는 ncol= 로 지정(둘 中 택 1)

```
> x <- matrix(1:12, nrow=4, ncol=3)
> x
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
```

- 1:12는 c(1,2,3,4,5,6,7,8,9,10,11,12)와 동일
- c(1:12)로 입력도 가능
- 자료는 열을 기준으로 입력됨

행렬(Matrix) 및 배열(array)

- 자료를 행 단위로 입력: 옵션 byrow= TRUE를 추가

```
> y <- matrix(1:12, nrow=4, byrow=TRUE)
> y
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
[4,]   10   11   12
```

◎ 행렬의 생성 2:

- 함수 dim ( )

```
> x <- 1:12
> dim(x) <- c(2,6)
> x
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    3    5    7    9   11
[2,]    2    4    6    8   10   12
```

행렬(Matrix) 및 배열(array)

◎ 행렬의 행과 열에 이름 붙이기

▪ 함수 rownames ( )와 colnames ( )

```
> x
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    3    5    7    9   11
[2,]    2    4    6    8   10   12

> rownames(x) <- c("one","two")
> colnames(x) <- c("a","b","c","d","e","f")

> x

     a b c d  e  f
one  1 3 5 7  9 11
two  2 4 6 8 10 12
```

행렬(Matrix) 및 배열(array)

◎ 행렬의 요소 파악하기

```
> length(x)
[1] 12

> nrow(x); ncol(x)
[1] 2
[1] 6

> dim(x)
[1] 2 6
```

행렬(Matrix) 및 배열(array)

◎ 함수 array ( ) 와 함수 c ( ) 로 배열 만들기

```
> abc <- array(1:24, c(3, 4, 2))
> abc
, , 1

     [,1] [,2] [,3] [,4]

[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12

, , 2

     [,1] [,2] [,3] [,4]
[1,]   13   16   19   22
[2,]   14   17   20   23
[3,]   15   18   21   24
```

행렬(Matrix) 및 배열(array)

◎ 각 차원에 이름을 부여!

- 함수 dimnames ( ) 를 활용: 단, 제일 포괄적 구조인 list도 활용

```
> dimnames(abc) <- list(X=c("a1","a2","a3"),
                        Y=c("b1","b2","b3","b4"),
                        Z=c("c1","c2"))

> abc
, , Z= c1

   b1 b2 b3 b4
a1  1  4  7 10
a2  2  5  8 11
a3  3  6  9 12

, , Z= c2

   b1 b2 b3 b4
a1 13 16 19 22
a2 14 17 20 23
a3 15 18 21 24
```

행렬(Matrix) 및 배열(array)의 인덱싱

◎ 행렬 인덱싱

x[i,j] : 행렬 x의 i번째 행, j번째 열의 요소를 호출
x[i,] : 행렬 x의 i번째 행 전체 요소를 호출
x[, j] : 행렬 x의 j번째 열 전체 요소를 호출

```
> abc
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12

> abc [1,3]
 [1] 7
> abc [3,]
 [1] 3  6  9  12
```

```
> abc [,3]
 [1] 7 8 9

> abc [1:2,]

  [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
```

행렬(Matrix) 및 배열(array)의 인덱싱

◎ 배열 인덱싱
- 차원 수만큼의 첨자가 필요함

```
> xyz <- array(1:24, c(4,3,2))

> xyz [,1,1]
[1]  1  2  3  4

> xyz [,,1]
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
```

막간의 퀴즈

◎ 다음의 행렬을 생성하라.

	var1	var2	var3
Case #1	12	21	32
Case #2	17	22	34
Case #3	19	25	35

1) 행렬의 두 번째 열을 다음과 같이 출력하라.

Case #1	Case #2	Case #3
21	22	25

데이터 프레임(Data Frame)

◎ 특성

- 행렬과 같은 2차원 형태
- 하나의 열에는 같은 유형의 자료
- 각각의 열은 서로 다른 유형의 자료가 올 수 있음
- 통계 데이터 세트에 적합한 구조

◎ 데이터 프레임의 생성

- 함수 data.frame ( )

```
> df1 <-data.frame(x=c(2,4,6),y=c("a","b","c"))
> df1
  x y
1 2 a
2 4 b
3 6 c
```

데이터 프레임(Data Frame)

◎ 함수 data.frame()의 편리성:

- 입력된 문자열 벡터를 요인으로 변환시킴
- 요인: level이 있는 순서형 척도로 활용이 가능했던 데이터 factor

```
> str(df1)
'data.frame': 3 obs. of  2 variables:
 $ x: num  2 4 6
 $ y: Factor w/ 3 levels "a","b","c": 1 2 3
```

- 문자열 벡터를 요인으로 변환시키기 싫다면?
  함수 내에 stringsAsFactors=FALSE 입력

```
> df2 <-data.frame(x=c(2,4,6),y=c("a","b","c"),
                   stringAsFactors=FALSE)
> str(df2)
'data.frame': 3 obs. of  2 variables:
 $ x: num  2 4 6
 $ y: chr "a","b","c"
```

데이터 프레임(Data Frame)

◎ 데이터 프레임의 행과 열 이름, 변수 개수 확인법!

- 행 이름: 함수 rownames ( )
- 열 이름: 함수 colnames ( ), names( )
- 열의 자료는 변수를 나타내며 이는 variable이라 불림
- 변수의 개수를 확인하는 방법: 함수 legnth ( )

```
df3<-data.frame(a=c(1,2,3),b=c("x","y","z"))
> df3
  a b
1 1 x
2 2 y
3 3 z

> colnames(df3)
[1] "a" "b"
> names(df3)
[1] "a" "b"

> rownames(df3)
[1] "1" "2" "3"
> length(df3)
[1] 2
```

데이터 프레임(Data Frame)

◎ 데이터 프레임의 인덱싱1

- 열(변수) 선택
- df[[a]] 또는 df[a]의 형식: 벡터 a는 숫자형 혹은 문자형
- df[[a]]: 한 변수의 선택. 결과는 벡터
- df [a] : 하나 또는 그 이상의 변수 선택. 결과는 데이터 프레임

```
> df3
  a b
1 1 x
2 2 y
3 3 z

> df3[1]
  a
1 1
2 2
3 3

> df3[[1]]
[1] 1 2 3
```

```
> df3 [“a”]
  a
1 1
2 2
3 3

> df3[[“a”]]
[1] 1 2 3
```

데이터 프레임(Data Frame)

◎ 데이터 프레임의 변수 선택

- 벡터 형태로 선택하는 것이 일반적
- df[[a]]의 형태가 더 많이 사용됨
- 조금 더 활용하기 편한 방법: $기호 사용

```
> df3[["a"]]
[1] 1 2 3

> df3$a
[1] 1 2 3
```

데이터 프레임 이름$변수 이름

데이터 프레임(Data Frame)

◎ 데이터 프레임의 인덱싱2

- 행렬의 인덱싱 방법 사용
- df[i, j]의 형태
- 선택된 변수가 하나이면 결과는 벡터
  하나 이상이면 결과는 데이터 프레임

```
> df3 [c(1,2),1]
[1] 1 2

> df3[c(1,2),]

  a b
1 1 x
2 2 y
```

막간의 퀴즈

◎ 다음의 데이터 프레임을 생성하라.

	var1	var2	var3
1	12	21	32
2	17	22	34
3	19	25	35

1) 데이터 프레임의 세 번째 열을 다음과 같이 출력하라.

	var3
1	32
2	34
3	35

2) 데이터 프레임의 세 번째 열을 벡터로 출력하라.

막간의 퀴즈

7. base 패키지 중 하나인 datasets에 데이터 프레임 iris가 있다.
   1) iris를 구성하고 있는 변수와 관찰값의 개수를 확인하라.
   2) 변수의 이름을 확인하라.
   3) 함수 head( )를 사용하여 데이터 프레임 iris의 처음 세 줄을 출력하라.

8. 패키지 datasets에 데이터 프레임 trees가 있다.
   1) trees를 구성하고 있는 변수와 관찰값의 개수를 확인하라.
   2) 변수의 이름을 확인하라.
   3) 함수 tail( )을 사용하여 데이터 프레임 trees의 마지막 세 줄을 출력하라.

리스트 (list)

◎ 구조의 특성

- 가장 포괄적인 구조
- 구성요소: 벡터, 배열, 데이터 프레임, 함수, 다른 리스트
- 서로 다른 유형의 객체를 한데 묶은 또 다른 객체

◎ 리스트 생성

- 함수 list ( )를 활용

리스트 (list)

```
> x <- list(a=c("one","two","three"), b=1:3, c=list(-1,-5),
            d=data.frame(x1=c("s1","s2"),x2=1:2))
> x
$a
[1] "one"   "two"   "three"

$b
[1] 1 2 3

$c
$c[[1]]
[1] -1

$c[[2]]
[1] -5

$d
  x1 x2
1 s1  1
2 s2  2
```

리스트 (list)의 인덱싱

◎리스트의 인덱싱

- list [[a]] 또는 list [a]의 형태
- list [a] : 결과는 리스트
- list [[a]] 또는 list $ a : 해당되는 구성요소의 객체구조

```
> x[1]
$a
[1] "one"   "two"   "three"

> x[[1]]
[1] "one"   "two"   "three"

> str(x[1])
List of 1
$ a: chr [1:3] "one" "two" "three"

> str(x[[1]])
chr [1:3] "one" "two" "three
```

리스트 (list)의 인덱싱

◎리스트 x의 4번째 요소를 데이터 프레임의 형태로 선택

```
> x[[4]]
   x1 x2
1  s1  1
2  s2  2
```

```
> x[["d"]]
   x1 x2
1  s1  1
2  s2  2
```

```
> x$d
   x1 x2
1  s1  1
2  s2  2
```

◎리스트 x의 4번째 요소의 두번째 열을 데이터 프레임의 형태로 선택

```
> x[[4]] [2]
   x2
1   1
2   2
```

◎리스트 x의 4번째 요소의 두번째 열을 벡터 형태로 선택

```
> x[[4]] [[2]]
 [1] 1 2
> x$d$x2
 [1] 1 2
```

R에서의 리스트를 활용할 때:

◎ 리스트 형태의 편리성:

- 산만하게 흩어진 정보를 간단히 묶기가 가능
- 많은 R 함수들의 수행결과가 리스트의 형태
- 많은 수행결과들 중 원하는 결과만 인덱싱해서 사용

2-1장. 데이터 생성

박영식(youngsikrex@naver.com)

데이터의 입력

◎데이터 생성법

1) 자판에서 데이터를 직접 생성
2) 외부 파일에서 데이터 가져오기
   - 텍스트 파일 (txt)
   - 엑셀 파일(xlsx)
   - SAS 파일 (등)
   - 웹에서 데이터 불러오기

웹에서의 데이터 불러오기

◎웹에 있는 3v적 데이터(Volume, Velocity, Variety)

◎두 가지 형태의 데이터

1) 웹 서버에 저장된 데이터 파일
- 함수 read.table( ), read.csv( )로 불러올 수 있음

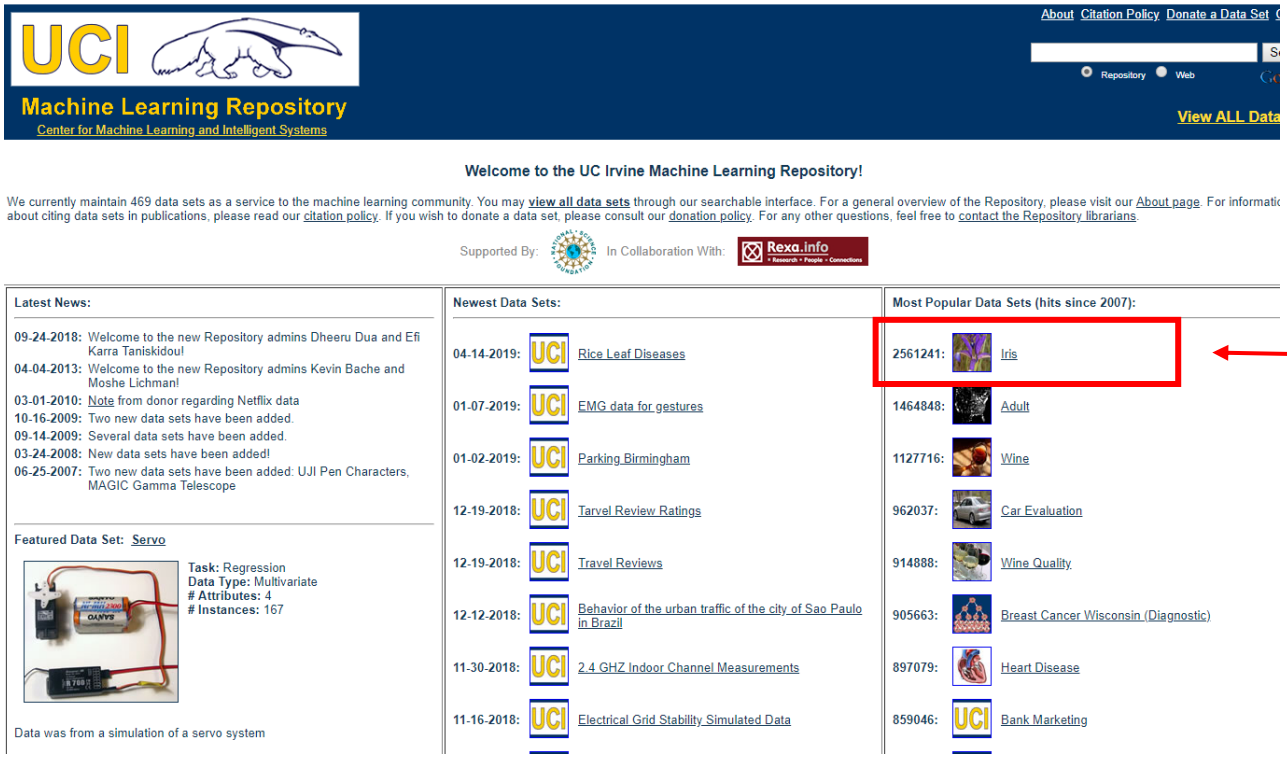
2) HTML 테이블
- 패키지 rvest
- 함수 read_html( ), html_nodes( ), html_table( )

◎웹 서버의 데이터 파일 불러오기
- 함수 read.table( ), read.csv( ), scan ( ) 등으로 파일 이름을 지정하는 대신 URL을 지정하므로써 로딩이 가능

- 예제: UCI Machine Learning Repository 사이트에서 Iris 데이터 셋 불러오기!

웹에서의 데이터 불러오기

◎ 사이트 URL: https://archive.ics.uci.edu/ml/index.php



클릭

웹에서의 데이터 불러오기

◎ Iris 데이터 셋에 대한 설명이 있는 폴더

UCI Machine Learning Repository
Center for Machine Learning and Intelligent Systems

Iris Data Set

Download: Data Folder, Data Set Description

Abstract: Famous database; from Fisher, 1936

Data Set Characteristics:	Multivariate	**Number of Instances:**	150	**Area:**	Life
Attribute Characteristics:	Real	**Number of Attributes:**	4	**Date Donated**	1988-07-01
Associated Tasks:	Classification	**Missing Values?**	No	**Number of Web Hits:**	2561255

Source:

Creator:

웹에서의 데이터 불러오기

◎ Iris 데이터 셋이 있는 URL 확인 및 기억

https://archive.ics.uci.edu/ml/machine-learning-databases/iris/

Index of /ml/machine-learning-databases/iris

- Parent Directory
- Index
- bezdekIris.data
- iris.data
- iris.names

Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips SVN/1.7.14 Phusion_Passenger/4.0.53 mod_perl/2.0.10 Perl/v5.16.3 Server at archive.ics.uci.edu Port 443

웹에서의 데이터 불러오기

◎ Iris 데이터 파일의 형태 확인: CSV

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.4,3.7,1.5,0.2,Iris-setosa
4.8,3.4,1.6,0.2,Iris-setosa
4.8,3.0,1.4,0.1,Iris-setosa
4.3,3.0,1.1,0.1,Iris-setosa
5.8,4.0,1.2,0.2,Iris-setosa
5.7,4.4,1.5,0.4,Iris-setosa
```

웹에서의 데이터 불러오기

◎ Iris 데이터 파일의 형태 확인: CSV

```
> iris.url <-  "https://archive.ics.uci.edu/ml/machine-
              learning-databases/iris/iris.data"

> iris.data <- read.csv(iris.url, header=FALSE,
                        stringsAsFactors=FALSE)

> names(iris.data) <- c("S.length","S.width","P.length",
                        "P.width","Species")

> head(iris.data)
  S.length S.width P.length P.width     Species
1      5.1     3.5      1.4     0.2 Iris-setosa
2      4.9     3.0      1.4     0.2 Iris-setosa
3      4.7     3.2      1.3     0.2 Iris-setosa
4      4.6     3.1      1.5     0.2 Iris-setosa
5      5.0     3.6      1.4     0.2 Iris-setosa
6      5.4     3.9      1.7     0.4 Iris-setosa
```

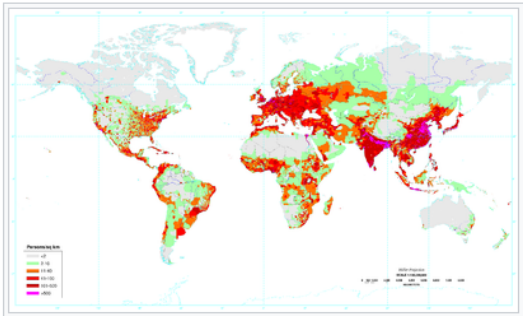
HTML table 가져오기

◎ 예: Wikipedia 웹페이지의 HTML 테이블 데이터 불러오기

https://en.wikipedia.org/wiki/World_population

10 most densely populated countries (with population above 5 million)

Rank	Country	Population	Area (km²)	Density (Pop. per km²)
1	Singapore	5,638,700	710	7,942
2	Bangladesh	166,430,000	143,998	1,156
3	Taiwan	23,577,488	36,193	651
4	Lebanon	6,093,509	10,452	583
5	South Korea	51,635,256	99,538	519
6	Rwanda	12,001,136	26,338	456
7	Netherlands	17,310,000	41,526	417
8	Haiti	11,112,945	27,065	411
9	India	1,346,310,000	3,287,240	410
10	Israel	9,020,000	22,072	409



Population density (people per km²) map of the world in 1994.
Red and pink areas denote regions of highest population density.

HTML table 가져오기

◎ 패키지 rvest의 함수 사용

1) URL을 함수 read_html ( )에 입력

```
> library(rvest)
> URL <- "https://en.wikipedia.org/wiki/World_population"
> web <- read_html (URL)
```

2) 읽어온 웹 페이지에서 함수 html_nodes ( )로 <table> 노드 추출

```
> tbl <- html_nodes(web, "table")
> length(tbl)
[1] 23
> head(tbl)
```

HTML table 가져오기

3) 함수 html_table( )로 테이블 데이터 불러오기

① 원하는 테이블의 노드를 아는 경우:

```
> tbl_1 <- html_table(tbl[7])
> tbl_1
[[1]]
```

	Rank	Country	Population	Area (km2)	Density (Pop. per km2)
1	1	Singapore	5,638,700	710	7,942
2	2	Bangladesh	165,540,000	143,998	1,150
3	3	Taiwan	23,577,488	36,193	651
4	4	Lebanon	6,093,509	10,452	583
5	5	South Korea	51,635,256	99,538	519
6	6	Rwanda	12,001,136	26,338	456
7	7	Netherlands	17,270,000	41,526	416
8	8	Haiti	11,112,945	27,065	411
9	9	India	1,339,630,000	3,287,240	408
10	10	Israel	8,950,000	22,072	405

HTML table 가져오기

② 원하는 테이블이 몇 번째 노드인지 모르는 경우

- 해당 테이블 노드의 Xpath를 알아야 함

1) Chrome을 통한 해당 웹페이지 (https://en.wikipedia.org/wiki/World_population) 접속

2) F12 키를 클릭 (혹은 마우스 우클릭 ->검사 클릭) 개발자 도구 실행

3) Ctrl + F 키를 눌러서 나타난 찾기 창에 "< table"을 입력하고 찾기 실행

4) 원하는 테이블의 해당 노드를 찾게 되면 해당 테이블이 하이라이트(형광처리) 마우스 우클릭

5) 나타난 별도 메뉴에서 'Copy > Copy xpath'를 클릭

6) //*[@id="mw-content-text"]/div/table[6]라는 값을 찾게 됨.

결과 확인

- 찾게 된 값을 적용하여 코딩 시작

```
> node_1 <- html_nodes(web,
                       xpath='//*[@id="mw-content-text"]/div/table[6]')

> tbl_1 <-  html_table(node_1)
> tbl_1
[[1]]
```

	Rank	Country	Population	Area (km2)	Density (Pop. per km2)
1	1	Singapore	5,638,700	710	7,942
2	2	Bangladesh	165,540,000	143,998	1,150
3	3	Taiwan	23,577,488	36,193	651
4	4	Lebanon	6,093,509	10,452	583
5	5	South Korea	51,635,256	99,538	519
6	6	Rwanda	12,001,136	26,338	456
7	7	Netherlands	17,270,000	41,526	416
8	8	Haiti	11,112,945	27,065	411
9	9	India	1,339,630,000	3,287,240	408
10	10	Israel	8,950,000	22,072	405

결과 확인

- 입력된 결과

```
> top_pop <- tbl_1[[1]]

> names(top_pop) <- c("rank“ , "country“ , "pop“ , "area“ , "density")

> str(top_pop)
'data.frame':     10  obs.  of    5   variables:
$ rank :     int      1  2  3  4  5  6  7  8  9  10
$ country: chr      "Singapore"   "Bangladesh"  "Taiwan"  "Lebanon" ...
$ pop :      chr      "5,638,700"   "165,540,000"    "23,577,488"    "6,093,509" ...
$ area :     chr      "710"    "143,998"    "36,193"    "10,452" ...
$ density:  chr      "7,942"     "1,150"    "651"    "583" ...
```

결과의 출력

◎ 분석 결과 외부 파일로 저장
- 함수 sink ( ): 1장에서 살펴보았음. 한계가 존재
- 함수 cat ( )의 활용

◎ 데이터 객체를 텍스트 파일로 저장
- 함수 write.table ( ) 의 이용

◎ 데이터 객체를 Excel 파일로 저장
- 패키지 xlsx의 함수 write.xlsx ( )

분석결과를 외부 파일로 출력

◎ 함수 cat ( )과 print( )의 기본기능: 객체에 할당된 값을 화면에 나타내는 것

```
> pi; print(pi); cat(pi)
[1] 3.141593
[1] 3.141593
3.141593
```

◎ 함수 print( )의 한계: 한 번에 한 객체만 출력 가능

```
> print("원주율은",pi,"이다")
Error in print. default("원주율은", pi, "이다") : invalid 'quote' argument

> cat("원주율은",pi,"이다")
원주율은 3.141593 이다
```

분석결과를 외부 파일로 출력

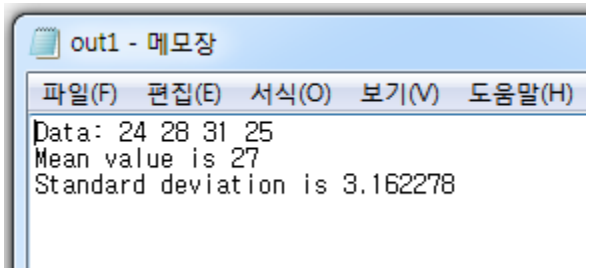
◎ 함수 cat ( )의 한계: 객체에 할당된 값을 벡터로만 나타내는 것

```
> y <- matrix(1:4, nrow=2)

> print ( y )
     [,1] [,2]
[1,]    1    3
[2,]    2    4

> cat(y)
1 2 3 4
```

분석결과를 외부 파일로 출력

◎ 함수 cat ( )으로 작업결과 외부로 저장: 옵션 file

```
> x <- c(24,28,31,25)
> m.x <- mean(x); s.x <- sd(x)

> cat("Data:", x, "\n", file="D:/Data/out1.txt")

> cat("Mean value is", m.x, "\n", file="D:/Data/out1.txt", append=TRUE)

> cat("Standard deviation is", s.x, file="D:/Data/out1.txt", append=TRUE)
```



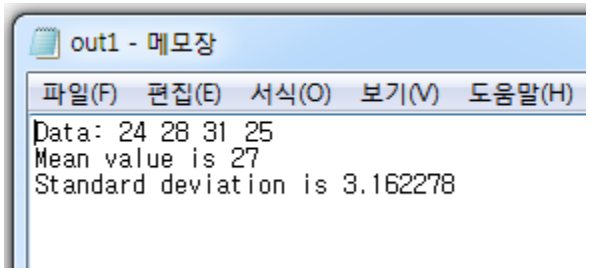
- 옵션 append: 이어 쓰기 여부 지정
  append=FALSE (디폴트): 덮어쓰기
  append=TRUE : 이어 쓰기

- "\n" : 줄 바꾸기

분석결과를 외부 파일로 출력

◎ 함수 cat ( )으로 작업결과 외부로 저장: 옵션 file

```
> x <- c(24,28,31,25)
> m.x <- mean(x); s.x <- sd(x)

> cat("Data:", x, "\n", file="D:/Data/out1.txt")

> cat("Mean value is", m.x, "\n", file="D:/Data/out1.txt", append=TRUE)

> cat("Standard deviation is", s.x, file="D:/Data/out1.txt", append=TRUE)
```



- 옵션 append: 이어 쓰기 여부 지정
  append=FALSE (디폴트): 덮어쓰기
  append=TRUE : 이어 쓰기

- "\n" : 줄 바꾸기

데이터 객체를 텍스트 파일로 저장

◎ 함수 write.table ( ) 이용:
추가 옵션에 따라 저장되는 형태가 다름

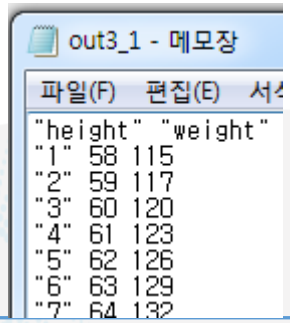
◎ 예제: 데이터 프레임 women
30대 미국 여성 15명의 몸무게(파운드 단위)와 키 (인치 단위)의 데이터

```
> head(women)
  height weight
1     58    115
2     59    117
3     60    120
4     61    123
5     62    126
6     63    129
```

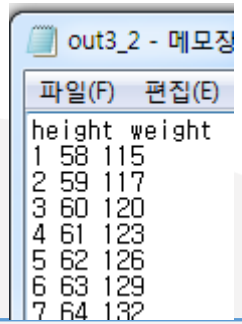
데이터 객체를 텍스트 파일로 저장

```
> write.table(women,“D:/Data/out3_1.txt")                          # 1

> write.table(women, “D:/Data/out3_2.txt", quote=FALSE)            # 2

> write.table(women, “D:/Data/out3_3.txt", quote=FALSE,
      row.names=FALSE)                                             # 3

> write.table(women, “D:/Data/out3_4.txt", quote=FALSE,
      row.names=FALSE, sep=",")                                    # 4

> write.csv(women,"D:/Data/out3_4.txt", quote=FALSE,
      row.names=FALSE)                                             # 4
```

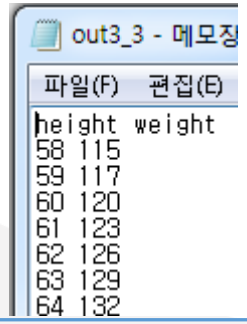
1



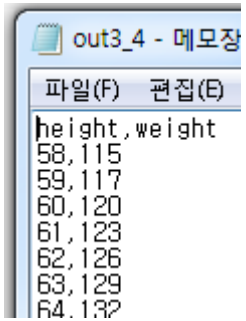
2



3



4



이번 강의를 마치며…

◎ 다양한 구조의 R 데이터 객체
- 벡터, 요인, 행렬, 배열, 데이터 프레임, 리스트

◎ 외부 파일 불러오기 (URL 등)

◎ R 데이터 객체를 외부 파일로 저장하기

◎ 3장: 데이터 다루기 기법
- 입력된 데이터를 분석이 가능하도록 다듬고 변환하는 기법

주요 이력
現) ㈜RTMC 전략기획실장
前) ㈜B사 웹로그분석 및 DP사업 完
前) ㈜H금속사 회계팀
前) ㈜B건설사 회계팀
前) K문고 CRM VIP 군집전략 CRM프로젝트 보조연구원
前) L백화점 CRM Alert 전략 CRM프로젝트 보조연구원

BSL(스위스 로잔 비즈니스 스쿨) MBA
ASSIST 빅데이터경영통계 MBA

국가공인 ADSP(빅데이터 준전문가)
現 **코리아IT아카데미 빅데이터 R 강사**
現 **코리아IT아카데미 빅데이터 기초 파이썬 강사**
現 **코리아IT아카데미 빅데이터 기초통계 전담강사**

[박영식] 완성에 이르기까지