

R컴퓨팅

3강

데이터 입력과 출력

정보통계학과 장영재 교수

1 데이터의 입력

2 데이터의 출력

1

데이터의 입력

1 데이터의 입력

1 키보드를 통한 데이터의 입력

➤ [1] c() 함수를 이용한 입력

```
> v1 = c(51, 56, 80, 78, 99 )
```

```
> v1
```

```
[1] 51 56 80 78 99
```

1 데이터의 입력

1 키보드를 통한 데이터의 입력

- [2] scan() 함수를 이용한 입력
- 보기 3-1 : scan() 함수를 이용하여 (63,73,27,32,69,98)과 같은 벡터 v2 생성

```
> v2 = scan()
```

```
1: 63
```

```
2: 73 27
```

```
4: 32 69 98
```

```
7:
```

```
Read 6 items
```

```
> v2
```

```
[1] 63 73 27 32 69 98
```

1 데이터의 입력

1 키보드를 통한 데이터의 입력

➤ 보기 3-2 : scan() 함수를 이용하여 (A,B,C)와 같은 벡터 v3를 생성

```
> v3 = scan( what=" " )      # what="character"와 같은 효과
1: A B C
4:
Read 3 items
> v3
[1] "A" "B" "C"
```

함수의 인수에 what="character" 라고 하거나,
what=" "라고 하면 문자형 데이터로 인식

1 데이터의 입력

1 키보드를 통한 데이터의 입력

- [3] edit() 함수를 이용한 입력
- edit() 함수를 실행시키면 나타나는 데이터 편집기 창에 데이터를 직접 입력하는 방식

편집기 창은 마치 엑셀과 같은 스프레드시트 형태이지만 데이터프레임이 이미 생성되어 있어야 사용할 수 있기 때문에 미리 빈 데이터프레임을 생성시켜 놓아야 함(6장 참고)

1 데이터의 입력

1 키보드를 통한 데이터의 입력

- 보기 3-3 : data.frame() 함수로 데이터프레임을 열고 dat1으로 저장한 뒤 dat1 = edit(dat1) 명령문으로 편집기를 열어 데이터를 입력

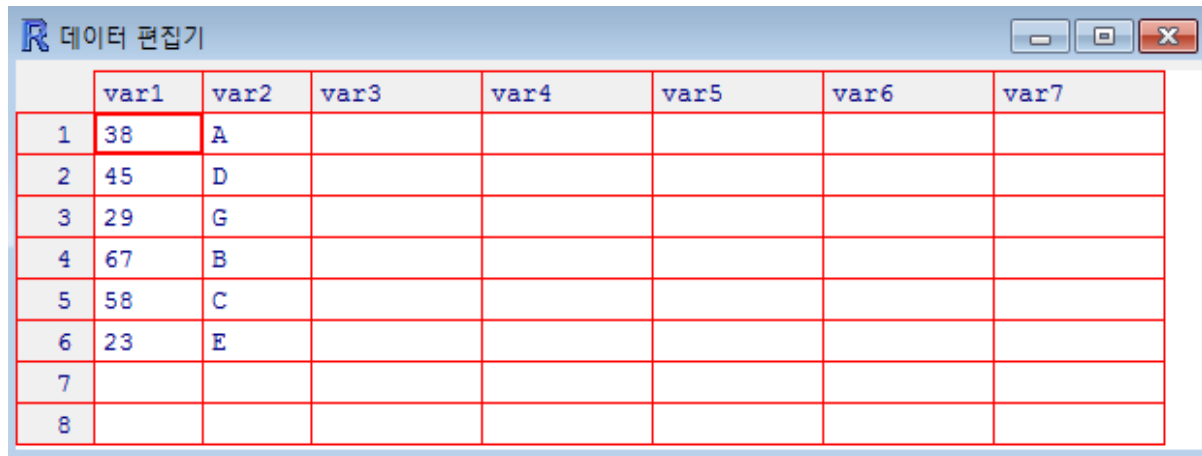
```
> dat1 = data.frame()           # 빈 데이터프레임 생성
> dat1 = edit(dat1)             # 데이터 편집기 창 만들기
> dat1                          # 데이터 입력 종료 후 내용 확인
```

	var1	var2
1	38	A
2	45	D
3	29	G
4	67	B
5	58	C
6	23	E

1 데이터의 입력

1 키보드를 통한 데이터의 입력

※ 데이터 편집기 창 입력



	var1	var2	var3	var4	var5	var6	var7
1	38	A					
2	45	D					
3	29	G					
4	67	B					
5	58	C					
6	23	E					
7							
8							

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

[1] read.table() 함수를 이용한 데이터 불러오기

➤ read.table() 함수를 이용하기 위해서는 우선 스프레드시트 형태의 외부파일이 존재해야 함

○ 스프레드시트 형태를 가지는 데이터는 행과 열이 있는 2차원의 구조를 갖는데, 각 행은 관찰치를 의미하고 각 열은 변수 값을 의미

○ 한 관찰치 내에서 변수 값들 사이에는 빈칸, 탭, 콜론(:), 콤마(,) 와 같은 구분기호가 사용되어 변수 값들을 분리

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

- 보기 3-4 : 'D:/R files' 폴더에 아래와 같이 탭으로 분리된 텍스트 형태인 example2-1.txt 파일이 저장되어 있다고 할 때, 이를 dat2_1라는 이름의 데이터프레임으로 불러오기

<example2-1.txt 파일>

153	150	A
151	149	B
153	155	C
158	145	D
142	153	E
146	153	F

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

```
> setwd("D:/R files")    # 작업폴더의 정의
> dat2_1 = read.table(file="example2-1.txt")
> dat2_1
V1 V2 V3    # 기본값으로 변수명 부여
1 153 150 A
2 151 149 B
3 153 155 C
4 158 145 D
5 142 153 E
6 146 153 F
```

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

- 보기 3-5 : 'D:/R files' 폴더에 다음과 같이 탭으로 분리되어 있고 맨 윗줄에 변수명이 존재하는 텍스트 형태인 example2-2.txt 파일이 저장되어 있을 때, `read.table()` 함수를 이용하여 데이터를 읽고 이를 `dat2_2`라는 이름으로 저장하기

<example2-2.txt 파일>

Current	Innov	Loc
153	150	A
151	149	B
153	155	C
158	145	D
142	153	E
146	153	F

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

```
> dat2_2 = read.table(file="example2-2.txt", header=TRUE)
```

```
> dat2_2
```

```
Current Innov Loc
```

```
1 153 150 A
```

```
2 151 149 B
```

```
3 153 155 C
```

```
4 158 145 D
```

```
5 142 153 E
```

```
6 146 153 F
```

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

- 보기 3-6 : 아래 example2-3.txt 데이터 파일과 같이 결측치가 NA와 miss로 표시되어 있다고 가정할 때, `read.table()` 함수를 이용하여 데이터를 읽고 이를 `dat2_3`이라는 이름으로 저장하기

<example2-3.txt 파일>

Current Innov		Loc
153	150	A
151	149	B
153	NA	C
158	145	D
142	153	E
miss	153	F

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

- 위의 데이터에서 Current 변수내의 miss도 결측치라고 인식하도록 하기 위해서는, `na.strings="miss"`라는 옵션을 추가

```
> dat2_3 = read.table(file="example2-3.txt", header=TRUE,  
  na.strings="miss")
```

```
> dat2_3
```

```
Current Innov Loc
```

```
1 153 150 A
```

```
2 151 149 B
```

```
3 153 NA C
```

```
4 158 145 D
```

```
5 142 153 E
```

```
6 NA 153 F
```

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

- R에서는 NA를 결측치로 인식하게 된다. 그런데 NA로 표현되어 있지 않은 다른 문자도 결측치라면 `na.strings` 옵션을 이용
- 각 관찰치 내의 변수 값들 사이에는 빈칸이나 탭이 아니라 콜론이나 콤마로 구분되어 있다면 `sep` 옵션을 활용하여 구분자를 지정(`sep=","`)
 - 빈칸이나 탭인 경우에는 `sep` 옵션을 사용하지 않아도 디폴트로 변수 값을 구분하지만 만약 `sep` 옵션을 사용하고자 한다면, 빈칸은 `sep=" "`으로, 탭은 `sep="\t"`으로 지정

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

- 보기 3-7 : 아래 example2-4.txt 파일과 같이 콤마로 변수 값들이 구분되어 있을 때, sep옵션을 사용하여 dat2_4라는 데이터프레임으로 읽기

```
<example2-4.txt 파일>
Current, Innov, Loc
153, 150, A
151, 149, B
153, 155, C
158, 145, D
142, 153, E
146, 153, F
```

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

```
> dat2_4 = read.table(file="example2-4.txt", header=TRUE,  
  sep=",")  
> dat2_4  
Current Innov Loc  
1 153 150 A  
2 151 149 B  
3 153 155 C  
4 158 145 D  
5 142 153 E  
6 146 153 F
```

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

[2] read.csv() 함수를 이용한 데이터 불러오기

- CSV이란 comma separated values 란 영문의 약어로서, CSV 파일은 콤마가 구분기호로 사용된 데이터를 의미하며 CSV 형태의 데이터 파일을 간편하게 불러오려면 read.csv() 함수 이용

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

- 보기 3-8 : CSV 형식의 데이터 파일 example2-4.txt 파일을 read.csv() 함수를 이용하여 불러오기

```
> dat2_5 = read.csv(file="example2-4.txt")
```

```
> dat2_5
```

```
Current Innov Loc
```

```
1 153 150 A
```

```
2 151 149 B
```

```
3 153 155 C
```

```
4 158 145 D
```

```
5 142 153 E
```

```
6 146 153 F
```

- read.csv() 함수는 header=TRUE 와 sep="," 옵션이 디폴트

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

[2] read.csv() 함수를 이용한 데이터 불러오기

- CSV이란 comma separated values 란 영문의 약어로서, CSV 파일은 콤마가 구분기호로 사용된 데이터를 의미하며 CSV 형태의 데이터 파일을 간편하게 불러오려면 read.csv() 함수 이용

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

[3] scan() 함수를 이용한 데이터 불러오기

- scan() 함수는 what이라는 옵션을 이용하여 입력될 변수의 명칭과 변수값의 유형을 정의
- numeric 유형이라면 0과 같은 숫자를 입력하면 되고, character 유형이라면 “”을 사용

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

- 보기 3-9 : scan() 함수를 통해 결측치가 있었던 example 2-3.txt 파일을 읽어오는 명령문

```
> dat2_6 = scan(file="example2-3.txt", what=list(Current=0, Innov=0,  
+ Loc=""), skip=1, sep=" ", na.strings="miss")
```

```
Read 6 records
```

```
> dat2_6
```

```
$Current
```

```
[1] 153 151 153 158 142 NA
```

```
$Innov
```

```
[1] 150 149 NA 145 153 153
```

```
$Loc
```

```
[1] "A" "B" "C" "D" "E" "F"
```

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

- ▶ 첫 번째 변수의 명칭을 Current로 하고 그 유형을 numeric으로 했고, 두 번째 변수의 명칭을 Innov으로 하고 그 유형을 numeric으로 정의 하였으며 세 번째 변수의 명칭은 Loc로 하고 character 유형으로 정의
- ▶ scan() 함수에는 header 옵션이 존재하지 않으므로 example2-3.txt 과 같은 첫째 줄에 변수명이 나오는 파일에서는 둘째 줄부터 데이터를 읽을 필요
 - skip=1 옵션은 파일내의 첫째 줄을 건너뛰고 둘째 줄부터 데이터를 읽으라는 명령

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

- 위 R 결과에서 dat2_6는 데이터프레임이 아니고, 리스트(3장에서 소개)인데 리스트를 데이터프레임으로 변환하고자 한다면 다음과 같이 수행

```
> dat2_7 = as.data.frame(dat2_6)
```

```
> dat2_7
```

```
Current Innov Loc
```

```
1 153 150 A
```

```
2 151 149 B
```

```
3 153 NA C
```

```
4 158 145 D
```

```
5 142 153 E
```

```
6 NA 153 F
```

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

[4] `read.xlsx()` 함수를 이용한 데이터 불러오기

- R은 다른 Excel 등 다른 소프트웨어에서 저장된 파일을 데이터로 불러 읽을 수 있음
- Excel 2007 이후의 형식으로 저장되어 있는 데이터를 R로 바로 불러들이려면 우선 `xlsx`라는 패키지를 설치하고 활성화한 뒤 `read.xlsx()` 이용

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

```
> library(xlsx)
```

```
필요한 패키지를 로딩중입니다: rJava
```

```
필요한 패키지를 로딩중입니다: xlsxjars> dat2_8 =
```

```
read.xlsx(file="example2-2.xlsx", sheetIndex=1)
```

```
> dat2_8
```

```
Current Innov Loc
```

```
1 153 150 A
```

```
2 151 149 B
```

```
3 153 155 C
```

```
4 158 145 D
```

```
5 142 153 E
```

```
6 146 153 F
```

➤ sheetIndex=1 옵션은 Excel 파일의 첫 번째 워크시트를 의미

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

- Excel 97-2003 버전의 파일형식인 xls 형식을 사용하고자 한다면, XLConnect라는 패키지를 설치하고 활성화한 뒤 `readWorksheetFromFile()` 함수를 사용
- 보기 3-10 : example2-2.xlsx와 같은 내용이 Excel 97-2003 버전 으로 저장된 example2-2.xls 파일이 있을 때, `readWorksheetFromFile()` 함수를 이용하여 데이터를 불러들이기

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

```
> library(XLConnect)
> dat2_9 = readWorksheetFromFile('example2-2.xls', sheet=1)
> dat2_9
Current Innov Loc
1 153 150 A
2 151 149 B
3 153 155 C
4 158 145 D
5 142 153 E
6 146 153 F
> dat2_10 = readWorksheetFromFile('example2-2.xlsx', sheet=1)
```

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

- 참고로 readWorksheetFromFile() 함수는 Excel 모든 버전 파일에 적용 가능하여 Excel 2007 이후 버전인 example2-2.xlsx 파일도 읽음

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

[5] 웹사이트에서 데이터 불러오기

- 웹사이트에는 파일의 형태로 저장된 많은 공개된 데이터가 있으며 R에서는 이러한 공개 데이터를 손쉽게 불러올 수 있음
- `read.table()` 함수를 사용할 때 파일의 이름 대신 웹사이트 상의 파일의 주소를 적어 주면 가능함

1 데이터의 입력

2 외부파일을 통한 데이터 불러오기

※ OzDASL 사이트에서 auction.txt 라는 파일을 읽어 들이는 방법

```
> website = "http://www.statsci.org/data/general/auction.txt"
```

```
> dat2_11 = read.table(website, header=TRUE)
```

```
> dat2_11
```

```
Age Bidders Price
```

```
1 127 13 1235
```

```
2 115 12 1080
```

```
⋮
```

```
8 132 10 1253
```

```
9 137 9 1297
```

```
10 113 9 946
```


2 데이터의 출력

2 데이터의 출력

[1] sink() 함수를 이용한 결과 저장하기

- R의 Console 창에 나오는 결과를 외부파일로 저장하기 위해 사용하는 함수는 sink()
- sink("파일명") 과 같은 형식으로 구동되며 결과를 저장하고자 하는 작업이 모두 끝나면 sink() 함수를 다시 한 번 실행시켜서 외부파일로 저장하는 작업을 멈추게 해야 함

2

데이터의 출력

- 보기 3-11 : `setwd()` 및 `sink()` 함수를 이용하여 v1이라는 이름으로 저장된 (1, 3, 5, 7, 9, 11, 13, 15, 17, 19, NA) 벡터의 평균값을 D:/R files/output1.txt 파일로 저장하기

```
> setwd("D:/R files")
```

```
> sink("output1.txt")
```

```
> v1 = c(1, 3, 5, 7, 9, 11, 13, 15, 17, 19, NA)
```

```
> mean(v1,na.rm=TRUE)
```

```
> sink()
```

2

데이터의 출력

- ▶ 앞 장에 소개했던 `source()` 함수를 `sink()` 함수와 함께 사용하면 매우 간편하게 R 프로그램을 사용할 수 있음

```
> setwd("D:/R files")  
> sink("output2.txt")  
> source("example1.R", echo=TRUE)  
> sink()
```

- D:/R files/example1.R 파일내의 R 명령문들을 한 번에 수행하고 그 결과를 D:/R files/output2.txt에 저장

2 데이터의 출력

[2] cat() 함수를 이용한 결과 저장하기

- cat() 함수는 여러 개의 객체를 동시에 출력하여 파일로 저장시키는 방법
- sink() 함수와는 달리 cat() 함수는 줄 단위로 결과를 저장
 - file 옵션을 사용하여, 저장하고자 하는 외부파일의 명칭을 지정하며 append=TRUE 옵션을 사용하여 기존 파일의 맨 아래 줄에 결과를 이어 붙이고자 할 때 사용
 - 만약 append 옵션을 사용하지 않으면, 기존 파일의 내용을 덮어쓰게 되어 이전 결과를 모두 지우게 된다.

2

데이터의 출력

- 보기 3-12 : cat() 함수를 통해 ① 벡터 $v1 = c(1, 3, 5, 7, 9)$ 을 output3.txt라는 이름으로 저장하고, 벡터 $v2 = c(2, 4, 6, 8, 10)$ 를 이어 붙이기 ② 두 벡터의 원소간 곱과, 두 벡터의 합을 구하여 각각 v3, v4로 할당하고 이를 앞의 결과에 이어서 붙이기

```
> setwd("D:/R files")  
> v1 = c(1, 3, 5, 7, 9)  
> v2 = c(2, 4, 6, 8, 10)  
> v3 = v1 * v2  
> v4 = v1 + v2
```

2

데이터의 출력

```
> cat( "Vector v1= (", v1, ")", "\n", file="output3.txt" )  
> cat( "Vector v2= (", v2, ")", "\n", file="output3.txt", append=TRUE )  
> cat( "v1 * v2 = (", v3, ")", "\n", file="output3.txt", append=TRUE )  
> cat( "v1 + v2 = (", v4, ")", "\n", file="output3.txt", append=TRUE )
```

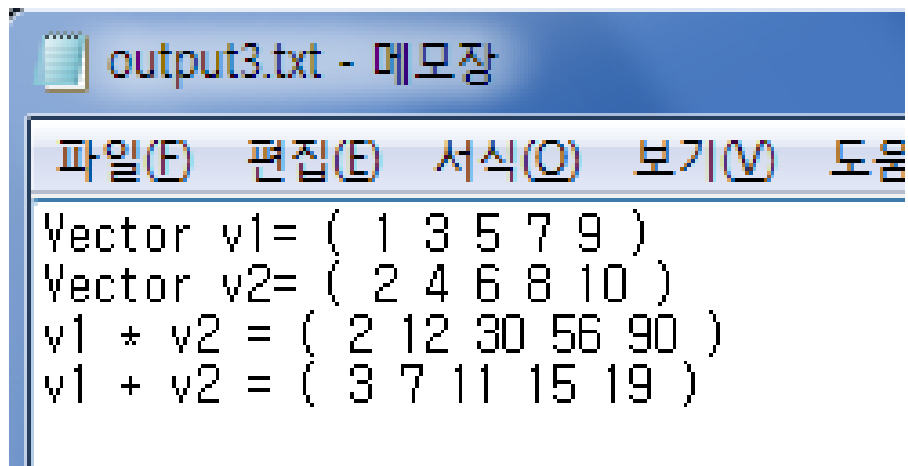
- `cat()` 함수의 명령부분의 ” “ 사이의 문장은 단순한 문자열을 의미하여 R에서는 주어진 대로 출력시키고 v1은 v1 벡터의 실제 값들을 출력

2

데이터의 출력

- “Wn”은 새로운 줄을 의미하고 append=TRUE 옵션에 의해 그 다음 부분의 결과는 두 번째 줄로 이어서 출력

※ cat() 함수를 이용한 파일의 생성결과



```
output3.txt - 메모장
파일(F)  편집(E)  서식(O)  보기(V)  도움
Vector v1= ( 1 3 5 7 9 )
Vector v2= ( 2 4 6 8 10 )
v1 * v2 = ( 2 12 30 56 90 )
v1 + v2 = ( 3 7 11 15 19 )
```


2 데이터의 출력

[3] write.table() 함수를 이용한 결과 저장하기

- R 작업으로 인해 생성된 데이터를 외부파일로 저장시키기 위해서 write.table() 함수를 사용
- 보기 3-13 : R의 기본 데이터로 제공되는 iris 데이터를 write.table() 함수를 이용하여 D:/R files/iris1.txt라는 외부파일로 저장하고 row.names, quote, sep 옵션에 따라 결과가 어떻게 달라지는지 살펴보기

2

데이터의 출력

```
> setwd("D:/R files")  
> write.table( iris, "iris1.txt" )  
> write.table( iris, "iris2.txt", row.names=FALSE )  
> write.table( iris, "iris3.txt", row.names=FALSE, quote=FALSE )  
> write.table( iris, "iris4.txt", row.names=FALSE, quote=FALSE, sep="\t" )  
> write.table( iris, "iris5.txt", row.names=FALSE, quote=FALSE, sep="," )
```

※ `write.table()` 함수의 디폴트

iris1.txt - 메모장

파일(F)	편집(E)	서식(O)	보기(V)	도움말(H)
"Sepal.Length"	"Sepal.Width"	"Petal.Length"	"Petal.Width"	"Species"
"1"	5.1	3.5	1.4	0.2 "setosa"
"2"	4.9	3	1.4	0.2 "setosa"
"3"	4.7	3.2	1.3	0.2 "setosa"
"4"	4.6	3.1	1.5	0.2 "setosa"
"5"	5.3	6	1.4	0.2 "setosa"
"6"	5.4	3.9	1.7	0.4 "setosa"
"7"	4.6	3.4	1.4	0.3 "setosa"
"8"	5.3	4	1.5	0.2 "setosa"
"9"	4.4	2.9	1.4	0.2 "setosa"
"10"	4.9	3.1	1.5	0.1 "setosa"
"11"	5.4	3.7	1.5	0.2 "setosa"
"12"	4.8	3.4	1.6	0.2 "setosa"

※ `write.table(iris, "iris2.txt", row.names=FALSE)`

iris2.txt - 메모장

파일(F)	편집(E)	서식(O)	보기(V)	도움말(H)
"Sepal.Length"	"Sepal.Width"	"Petal.Length"	"Petal.Width"	"Species"
5.1	3.5	1.4	0.2	"setosa"
4.9	3	1.4	0.2	"setosa"
4.7	3.2	1.3	0.2	"setosa"
4.6	3.1	1.5	0.2	"setosa"
5.3	6	1.4	0.2	"setosa"
5.4	3.9	1.7	0.4	"setosa"
4.6	3.4	1.4	0.3	"setosa"
5.3	4	1.5	0.2	"setosa"
4.4	2.9	1.4	0.2	"setosa"
4.9	3.1	1.5	0.1	"setosa"

※ `row.names=FALSE` 와 `quote=FALSE`

iris3.txt - 메모장

파일(F)	편집(E)	서식(O)	보기(V)	도움말(H)
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.3	6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.3	4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa

※ `write.table()` 옵션 `sep= "\t"`

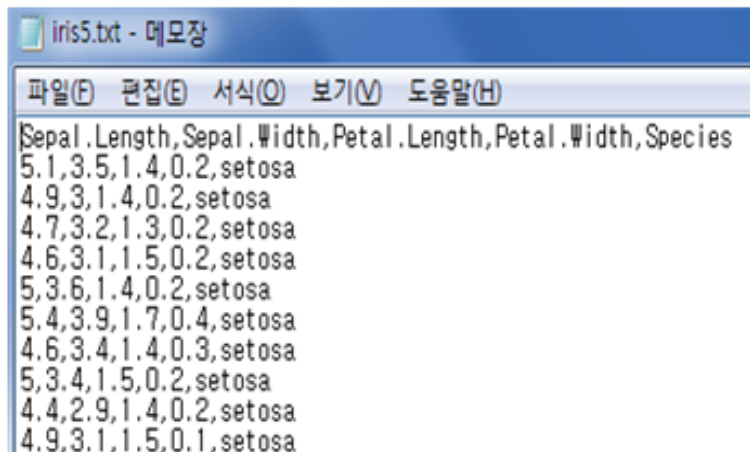
iris4.txt - 메모장

파일(F)	편집(E)	서식(O)	보기(V)	도움말(H)
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa

2

데이터의 출력

※ `write.table()` 함수의 `sep=“.”`



```
iris5.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
Sepal.Length,Sepal.Width,Petal.Length,Petal.Width,Species
5.1,3.5,1.4,0.2,setosa
4.9,3.1,1.4,0.2,setosa
4.7,3.2,1.3,0.2,setosa
4.6,3.1,1.5,0.2,setosa
5.3,6.1,4.0,0.2,setosa
5.4,3.9,1.7,0.4,setosa
4.6,3.4,1.4,0.3,setosa
5.3,4.1,1.5,0.2,setosa
4.4,2.9,1.4,0.2,setosa
4.9,3.1,1.5,0.1,setosa
```

2 데이터의 출력

[4] write.xlsx() 함수를 이용한 결과 저장하기

- Excel 2007 이후 버전의 파일로 데이터를 저장하기 위한 함수는 `xlsx` 패키지내의 `write.xlsx()` 함수
- 패키지가 설치된 상태에서 `library(xlsx)`로 활성화한 뒤 내장된 `iris` 데이터 저장

```
> library(xlsx)
```

```
필요한 패키지를 로딩중입니다: rJava
```

```
필요한 패키지를 로딩중입니다: xlsxjars
```

```
다음의 패키지를 부착합니다: 'xlsx'
```

```
> setwd("D:/R files")
```

```
> write.xlsx( iris, "iris.xlsx" )
```

2 데이터의 출력

※ write.xlsx() 함수에 의해 생성된 iris.xlsx 파일의 내용

	A	B	C	D	E	F
1		Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
2	1	5.1	3.5	1.4	0.2	setosa
3	2	4.9	3	1.4	0.2	setosa
4	3	4.7	3.2	1.3	0.2	setosa
5	4	4.6	3.1	1.5	0.2	setosa
6	5	5	3.6	1.4	0.2	setosa
7	6	5.4	3.9	1.7	0.4	setosa
8	7	4.6	3.4	1.4	0.3	setosa
9	8	5	3.4	1.5	0.2	setosa
10	9	4.4	2.9	1.4	0.2	setosa
11	10	4.9	3.1	1.5	0.1	setosa
12	11	5.4	3.7	1.5	0.2	setosa
13	12	4.8	3.4	1.6	0.2	setosa

- XLConnect 패키지는 Excel 형식의 데이터를 불러들이는 패키지이며 Excel 형식의 데이터로 저장하는 writeWorksheetToFile() 함수도 포함

2

데이터의 출력

- 보기 3-14 : writeWorksheetToFile() 함수를 이용하여 iris 데이터를 D:/R files/iris.xls 파일로 저장하기

```
> library(XLConnect)
```

```
XLConnect 0.2-7 by Mirai Solutions GmbH
```

```
http://www.mirai-solutions.com ,
```

```
http://miraisolutions.wordpress.com
```

```
> setwd("D:/R files")
```

```
> writeWorksheetToFile( "iris.xls", iris, sheet="FirstSheet" )
```

- write.xlsx() 함수와의 차이점은 iris 데이터와 파일이름을 지정하는 순서가 서로 바뀌어 있다는 점

2 데이터의 출력

- `write.xlsx(iris, "iris.xlsx")`와
`writeWorksheetToFile("iris.xlsx", iris)`의 차이
- `writeWorksheetToFile` 옵션 중에서 `sheet="FirstSheet"` 부분은 첫 번째 워크시트로 저장하라는 것

※ `writeWorksheetToFile()` 함수에 의해 생성된 `iris.xls` 파일

	A	B	C	D	E
1	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
2	5.1	3.5	1.4	0.2	setosa
3	4.9	3	1.4	0.2	setosa
4	4.7	3.2	1.3	0.2	setosa
5	4.6	3.1	1.5	0.2	setosa
6	5	3.6	1.4	0.2	setosa
7	5.4	3.9	1.7	0.4	setosa
8	4.6	3.4	1.4	0.3	setosa
9	5	3.4	1.5	0.2	setosa
10	4.4	2.9	1.4	0.2	setosa
11	4.9	3.1	1.5	0.1	setosa
12	5.4	3.7	1.5	0.2	setosa

R컴퓨팅

