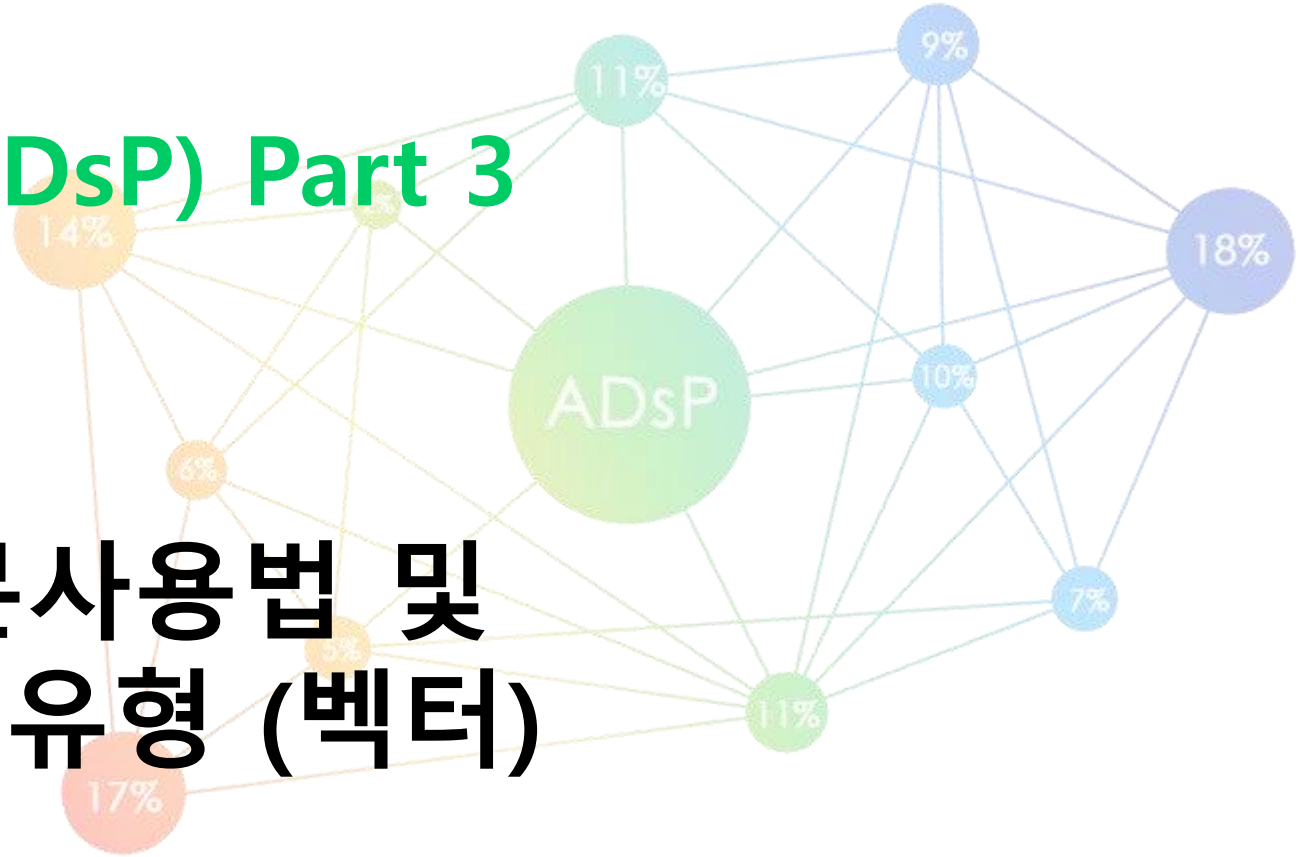
데이터분석준전문가(ADsP) Part 3

데이터분석



01 R의 주요특징,기본사용법 및 데이터의 구조 및 유형 (벡터)

1.R의 주요 특징

(1)R프로그램 역사

-R 프로그램은 미국 벨 연구소의 John Chambers 개발한 s언어를 기반

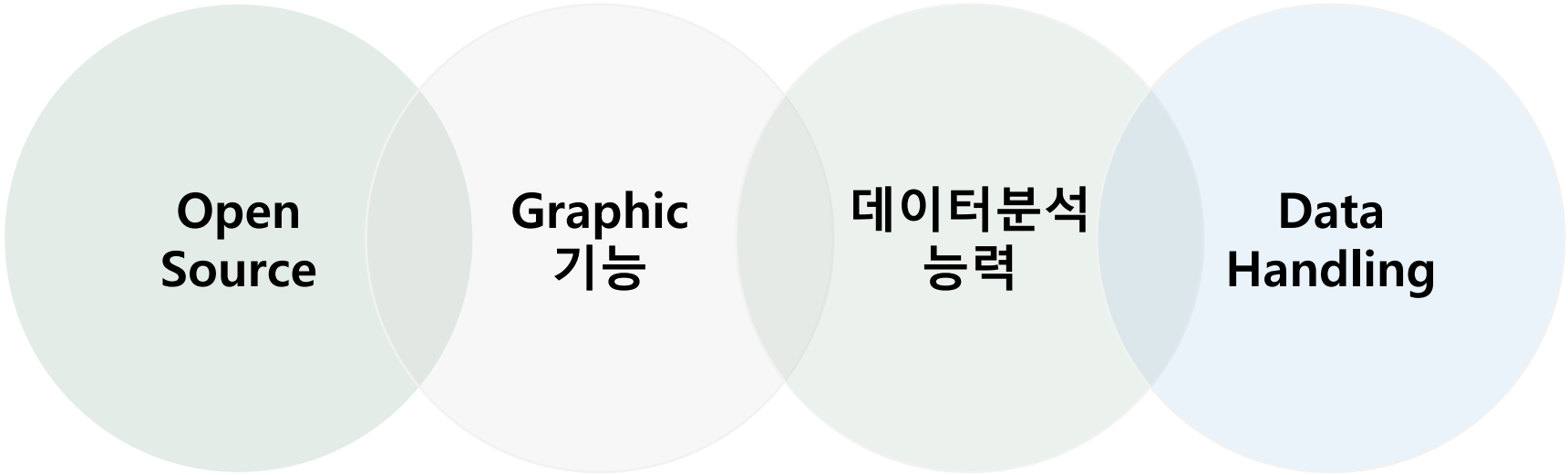
-뉴질랜드 오클랜드 대학교의 로스 이하카와 로버트 젠틀맨에 의해서 개발

-현재는 상용버전과 무료버전으로 운영

-R 프로그램의 버전 2000.02 1 버젼 시작으로 현재 3.5.1 버전

1.R의 주요 특징

(2)R프로그램의 특징



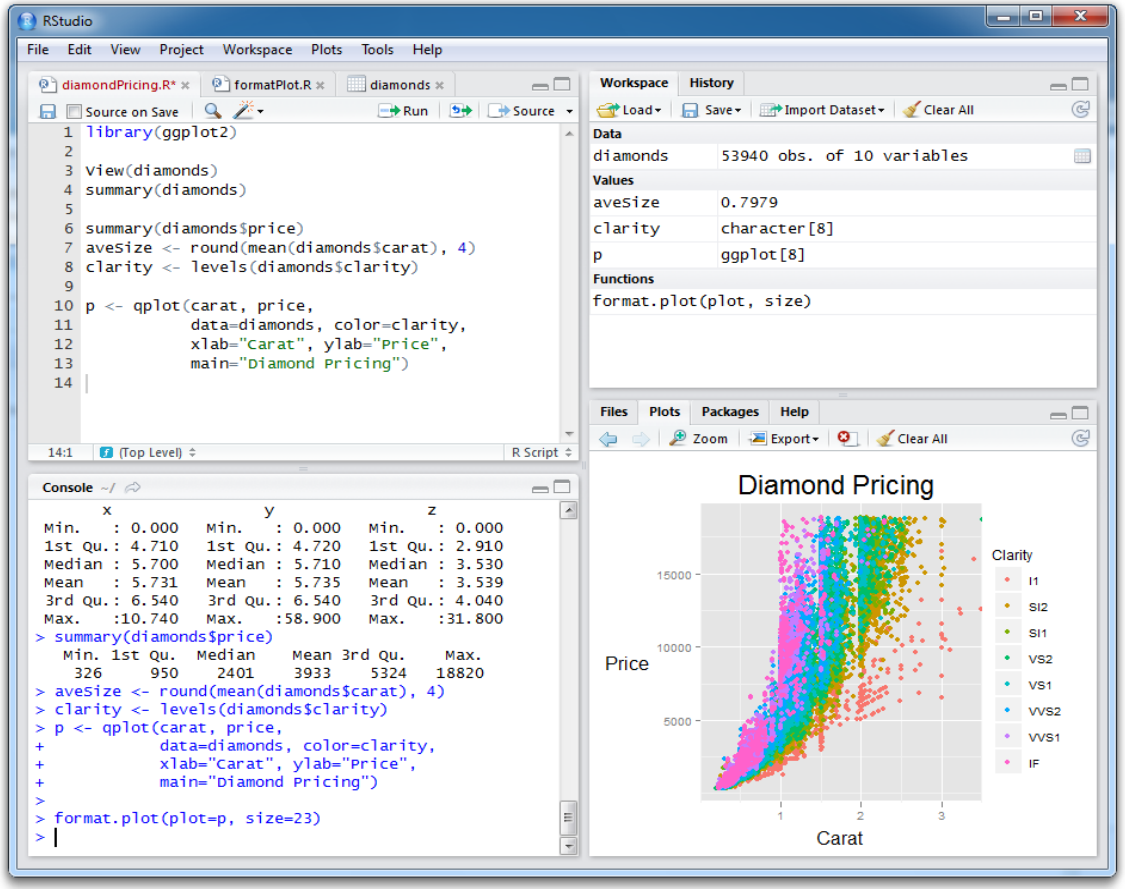
1.R의 주요 특징

(3)R프로그램 설치 (유의사항)

- 윈도우 운영체계 32bit/64bit 체크
- R-> RStudio 설치 순서

1.R의 주요 특징

(4)RStudio 구성



이미지출처 : https://www.rstudio.com

1.R의 기본 사용법

#(해시기호)

주석의 기능, Rcoding 내용이
무엇인지를 알 수 있도록 사용자가
설명을 달아주는 기능

Ctrl+Enter

- R의 명령어를 실행하는 기능
- 명령어가 두 줄 이상인 경우
  블록을 잡고 실행
- RUN 버튼

R은 대소문자 구별

소문자 'a',대문자 'A'
전혀 다른 의미

R도움말 보기

'?' 이나 help
예) ?summary, help(summary)

1.R의 기본 사용법

(1)R패키지 사용하기

- R에서 패키지는 함수,데이터,코드,문서 등을 묶은 것을 의미
- R은 오픈 소스 프로그램으로 다양한 기능이 패키지가 존재
- 패키지를 사용하는 데 필요한 함수
- install.packages("패키지 이름"): 패키지를 다운로드해서 설치한다
- library(패키지 이름):패키지를 로드하여 사용할 준비를 한다.

1.R의 기본 사용법

(2)R연산자(산술,할당,비교,논리)

○ R산술 연산자

- 산술 연산자의 우선순위: 괄호-> 거듭제곱-> 곱하기,나누기-> 더하기,빼기

연산자	설명	입력내용	결과
+	더하기	3+2	5
-	빼기	3-2	1
*	곱하기	3*2	6
/	나누기	3/2	1.5
^ or **	제곱	3^2	9

*주의 : 동일한 우선순위의 경우 왼쪽에서 오른쪽 순서를 가짐 예) 2*3/6=1

1.R의 기본 사용법

ㅇR할당 연산자

- 어떤 객체의 이름을 특정한 값에 저장할 때 사용

연산자	설명	입력내용	결과
<-, =	오른쪽의 값을 왼쪽의 이름에 저장	x<-3,x=3	동일한 값 3

*주의 : '=' R에서는 같다는 의미가 아님

1.R의 기본 사용법

ㅇ R비교연산자

- 두 개 값에 대한 비교로서 맞은면 TRUE, 맞지 않으면 FALSE 값을 갖는다.

연산자	설명	입력내용	결과
>	크다	3>4	FALSE
>=	크거나 같다	3>=4	FALSE
==	같다	3==4	FALSE
!	부정	!(3==4)	TRUE
>	크다	3>4	FALSE

*주의 : 연산자 안에 공백이 있으면 에러 예) ">(공백)="

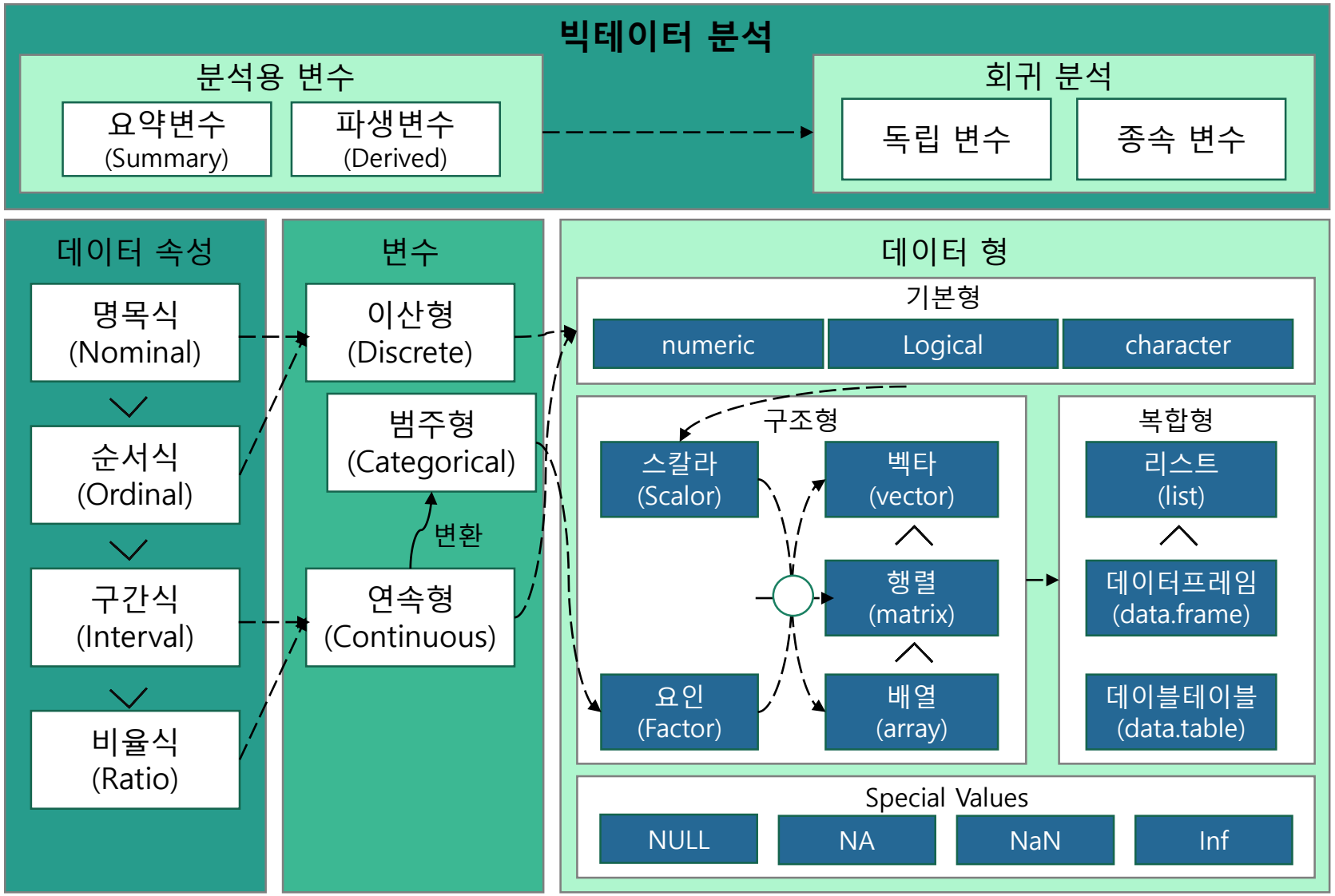
1.R의 기본 사용법

ㅇR논리 연산자

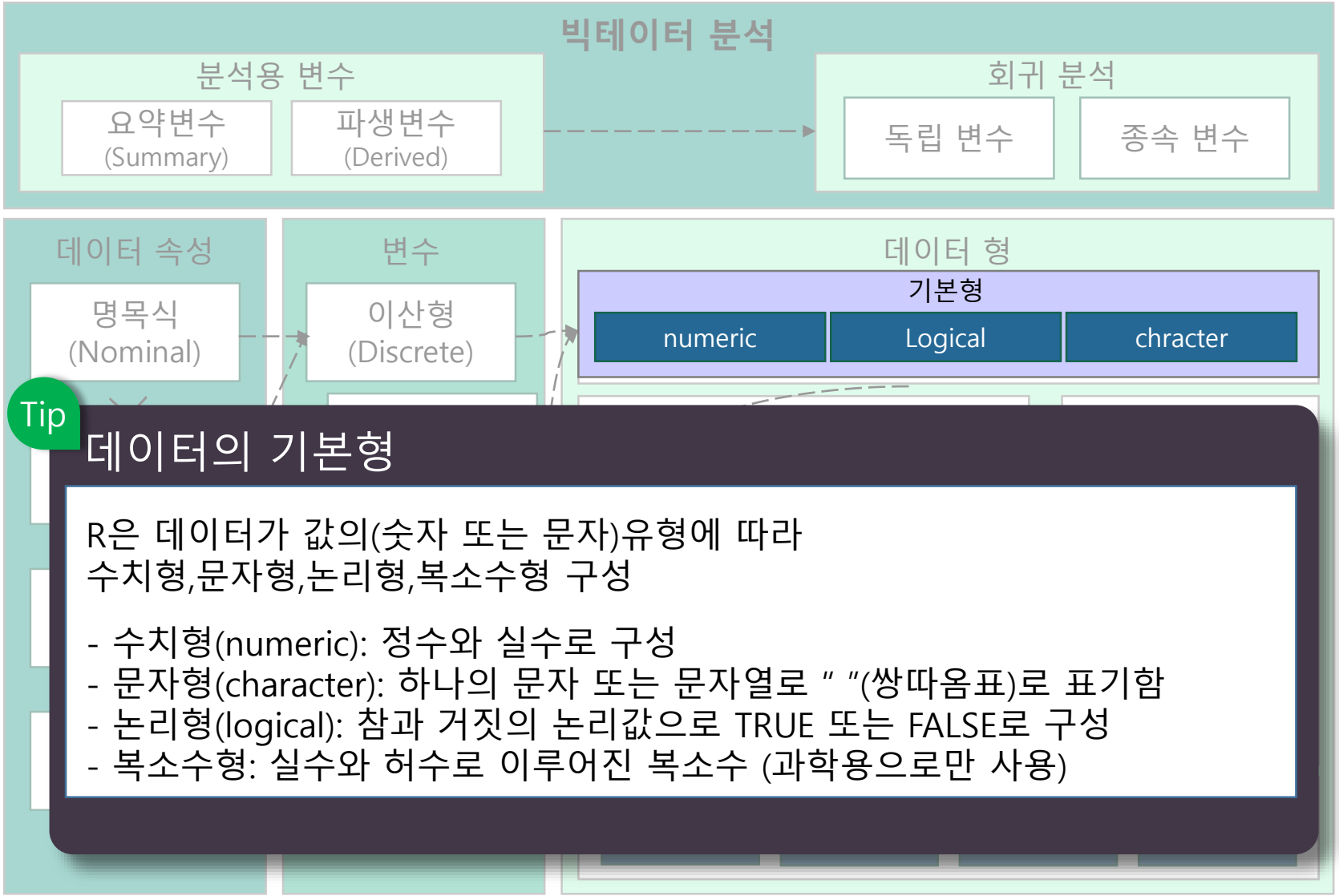
- 두 개 이상의 조건을 비교하여 결과값을 출력

연산자	설명	입력내용	결과
&	**AND의 개념** 두 개의 조건을 동시에 만족할 때만 TRUE가 되는 논리 연산	TRUE&FALSE	FALSE
|	**OR의 개념** 두 개의 조건 중에서 하나만 만족하여도 TRUE가 되는 논리 연산	TRUE&FALSE	TRUE

1.R데이터 종류


빅테이터 분석
분석용 변수
요약변수
(Summary)
파생변수
(Derived)
회귀 분석
독립 변수
종속 변수
데이터 속성
명목식
(Nominal)
순서식
(Ordinal)
구간식
(Interval)
비율식
(Ratio)
변수
이산형
(Discrete)
범주형
(Categorical)
변환
연속형
(Continuous)
데이터 형
기본형
numeric
Logical
character
구조형
스칼라
(Scalor)
벡타
(vector)
행렬
(matrix)
요인
(Factor)
배열
(array)
복합형
리스트
(list)
데이터프레임
(data.frame)
데이블테이블
(data.table)
Special Values
NULL
NA
NaN
Inf

1.R데이터 종류



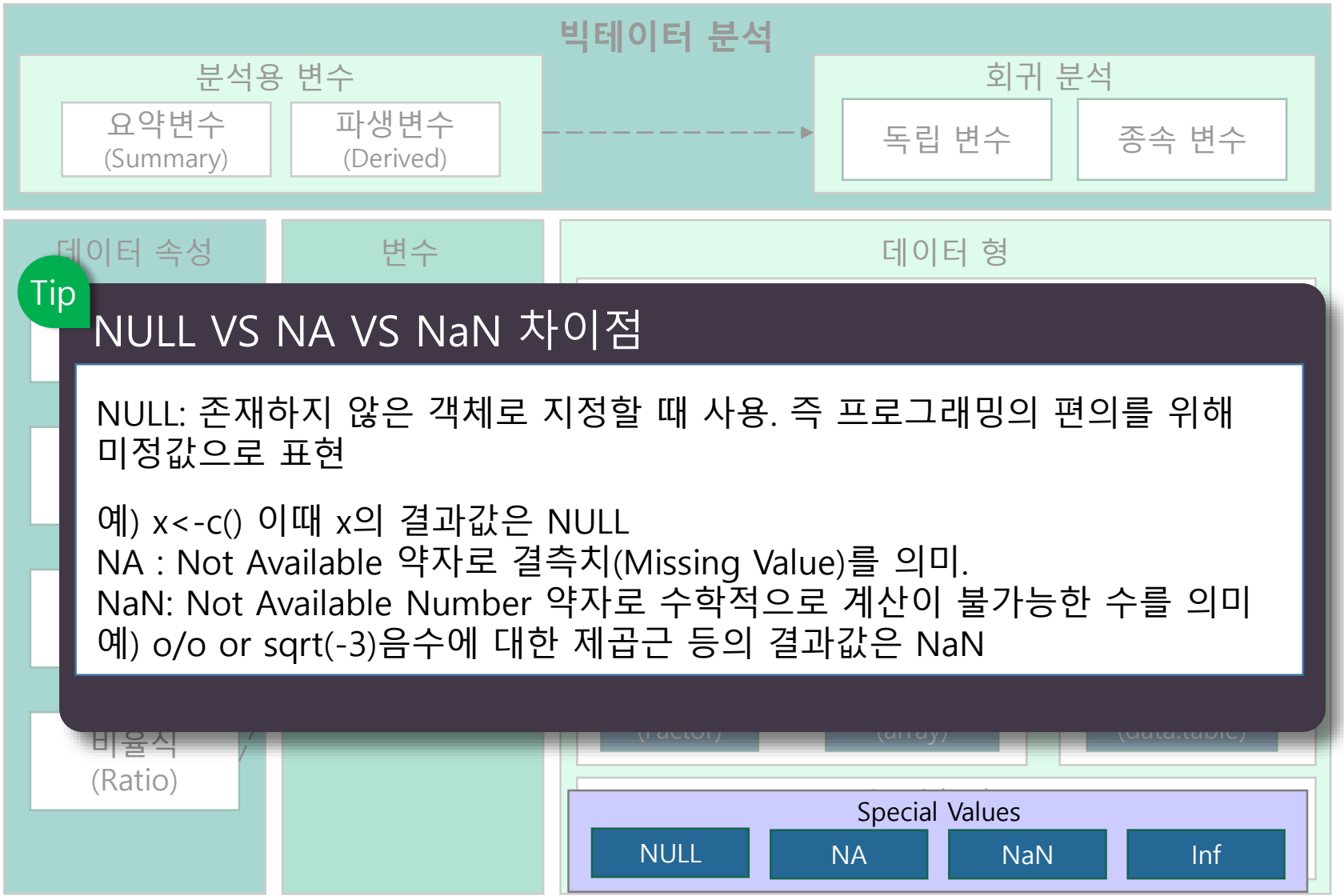
Tip

데이터의 기본형

R은 데이터가 값의(숫자 또는 문자)유형에 따라
수치형,문자형,논리형,복소수형 구성

- 수치형(numeric): 정수와 실수로 구성
- 문자형(character): 하나의 문자 또는 문자열로 " "(쌍따옴표)로 표기함
- 논리형(logical): 참과 거짓의 논리값으로 TRUE 또는 FALSE로 구성
- 복소수형: 실수와 허수로 이루어진 복소수 (과학용으로만 사용)

1.R데이터 종류



Tip

NULL VS NA VS NaN 차이점

NULL: 존재하지 않은 객체로 지정할 때 사용. 즉 프로그래밍의 편의를 위해 미정값으로 표현

예) x<-c() 이때 x의 결과값은 NULL
NA : Not Available 약자로 결측치(Missing Value)를 의미.
NaN: Not Available Number 약자로 수학적으로 계산이 불가능한 수를 의미
예) o/o or sqrt(-3)음수에 대한 제곱근 등의 결과값은 NaN

1.R데이터 종류

Q1 **y<-c(1,2,3,NA) 일 때 3*y ?**

① 3 6 9 NA

② NA NA NA NA

③ 3 6 9 NaN

④ 3 6 9 3NA

2.R데이터 종류

(1)벡터

- 벡터는 동일한 데이터 유형으로 이루어진 한 개 이상의 값들로 구성
- 하나의 열(Column) 구성
- 벡터는 데이터 분석의 가장 기본 단위임

1)벡터 만들기

-하나의 값의 벡터

x<-5(1개의 수치형 벡터), x<-"사과"(1개의 문자형 벡터)

x<-TRUE(1개의 논리 연산자 벡터), x<-1-3i(1개의 복소수형 벡터)

2.R데이터 종류

-두 개 이상의 값의 벡터는 c(),:,seq(),sequece(),rep()

c(): combine(연결) 약자로 4가지 유형 가능(수치형,문자형,논리형,복소수형)

예) x<-c(1,10),y<-c("apple","melon")

xy<-c(x,y)의 결과값은 문자형임 "1","10","apple","melon"

: 콜론(:) 수치형 적용

-start:end 형태,start>end이면 1씩 감소, start<end이면 1씩 증가

예)1:5 의 값은 1,2,3,4,5

2.R데이터 종류

seq()
- from(시작값),to(끝값),by(증가할 때 양수, 감소할 때 음수)
- length=m 옵션을 추가하면 전체 수열 개수가 m개 되도록 자동 증가

예)seq(from=1,to=5,by=1) 실행하면 결과값은 1,2,3,4,5이면 =seq(1,5,1)
seq(1,11,length=6) 실행하면 결과값은 1 3 5 7 9 11

sequence()
- 1과 지정한 숫자사이의 정수로 구성

예)sequence(3), 결과값은 1,2,3, sequence(3.3) 정수값이므로 결과는 1.2.3

rep()
- rep 함수는 replicate의 약자 (복사하다 의미)

예)rep(c("a","b"),times=3), 결과값은 "a" "b" "a" "b" "a" "b"
rep(c("a","b"), each=3), 결과값은 "a" "a" "a" "b" "b" "b"

2.R데이터 종류

Q2 **다음 중 나머지 세 개의 명령과 다른 결과를 주는 명령은 무엇인가?**

① seq(1,10,2)

② seq(from=1,to=10,by=2)

③ seq(from=1,to=10,length=5)

④ 1:5*2-1

2.R데이터 종류

2)벡터의 속성을 확인하는 함수

- 벡터가 가지는 각각의 값을 '원소(Element)'
- 원소가 어떤 데이터 유형, 갯수, 이름을 나타내는 것을 속성이라 말함
- class( ),mode( ) : mode() 클래스 안의 더 작은 단위의 속성
- is.numeric(벡터) : 데이터 유형이 numeric이 맞으면 TRUE 아니면 FALSE
- is.logical(벡터):데이터 유형이 logical이 맞으면 TRUE 아니면 FALSE
- is.character(벡터):데이터 유형이 character가 맞으면 TRUE 아니면 FALSE
- length( ) : 원소의 개수. length와 nrow() 함수도 행렬과 데이터프레임에서 행의 수를 알려준다. 단 벡터에서는 대문자 NROW() 사용한다.

예) x<-c(1,2,3)

함수	mode(x)	is.numeric(x)	is.logical(x)	is.charater(x)	length(x)
결과값	"numeric"	TRUE	FLASE	FALSE	3

2.R데이터 종류

names()
- 원소 또는 변수의 이름이 무엇인지 알 수 있음
- 원소 또는 변수의 이름을 새롭게 부여할 수 있음
- 원소 또는 변수값에 원하는 이름을 문자열 벡터로 할당

예) x<-c(1,2,3)

names(x)<-c("kim","kwan","park")
x
kim kwan park
1 2 3

2.R데이터 종류

3)변수이름 규칙

- R의 변수명(객체이름)은 알파벳,숫자,_(언더스코어), .(마침표) 가능, - (하이픈)사용할 수 없다
- 첫 글자는 알파벳 또는 .(마침표)로 시작.(마침표) 뒤에는 숫자가 올 수 없다.

올바른 변수명(객체이름)	올바르지 않은 변수명(객체이름)
a b a1 a2 .x	2a .2 a-b

2.R데이터 종류

Q3 다음 중 R에서 사용하는 객체의 이름으로 적당하지 않은 것은?

① .3ab

② abc

③ a.b

④ a2b

2.R데이터 종류

4)벡터의 접근(인덱싱)

- 벡터가 가지는 원소들 중에서 일부의 원소를 추출할 때는 대괄호[]를 사용
- 대괄호 안에 추출하기 원하는 원소의 위치(인덱스)를 수치로 입력

예)x<-c(“a”,“b”,“c”)

적용	x[1]	x[-1]	x[c(1,2)]
의미	[]안에 원소를 가져온다	[]음의 인데스를 사용해 특정요소만 제거	여러 위치에 저장된 값을 한번에 가져올 수 있음
결과	“a”	“b” “c”	“a” “b”

Tip 다른 프로그램 언어는 첫 번째 원소에 대한 위치를 ‘0’으로 인식

2.R데이터 종류

5)벡터의 연산

벡터의 길이가 동일한 경우

- 벡터들간의 사칙연산이 가능 최종 결과는 벡터
- 벡터들간의 연산이 될 때 동일한 위치의 값들 간에 연산

예) x<-c(1,2,3) y<-c(4,5,6)

적용	x+y	x*y	x/y
의미	5 7 9	4 10 18	0.25 0.40 0.50

2.R데이터 종류

벡터들 길이가 동일하지 않은 경우

- 두 벡터가 원소가 개수가 다르더라도 연산과정에서 원소의 개수가 적은 쪽의 벡터는 원소 개수가 많은 쪽의 벡터와 동일하게 원소의 개수를 맞춘다

예) x<-c(1,2,3) y<-c(1,2,3,4,5,6)

적용	결과	의미
x+y	2 4 6 5 7 9	R은 재사용 규칙에 따라 x에 3개 y에 6개의 데이터가 있다면 x1의 데이터를 추가적으로 3개가 더 생성되어 연산을 한다.

2.R데이터 종류

Q4 아래의 R 스크립트를 실행하여 얻게 되는 결과로 가장 적절한 것은?

```
x<-c(1,2,3)
y<-c(2,x,3)
x+y
```

2.R데이터 종류

Q5 다음 보기의 명령 중 에러가 발생되는 것은 무엇인가?

```
fruit<-c(1,2,3,4)
names(fruit)<-c("apple","peach","orange","banana")
```

① fruit[c("apple","peach")]

② fruit[1:2]

③ fruit[2:3]

④ fruit[-1:3]

3.R데이터 종류

(2)행렬

- 행렬(matrix)은 데이터의 형태가 2차원으로 행(row)과 열(column) 구성
- 벡터와 동일하게 하나의 데이터 유형만 가능

1)생성 함수

-matrix(): nrow(행의갯수지정),ncol(열의갯수지정)
byrow(행렬에 값이 입력할 때 열방향을 먼저 입력, 값이 입력방향을 행방향으로 수정할 때 byrow=TRUE 지정하고 사용함)

-rbind() : 벡터를 행 방향으로 합치는 방법. 'r'=row(행)

-cbind() : 벡터를 열 방향으로 합치는 방법. 'c'=column(열)

3.R데이터 종류

예) mx<-matrix(c(1,2,3,4,5,6),ncol=2)

```
mx
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
r1<-c(10,10)
rbind(mx,r1)
    [,1]   [,2]
       1    4
       2    5
       3    6
r1    10   10
```

3.R데이터 종류

Q6 다음 R 코드의 결과로서 옳지 않은 것은?

```
x<-matrix(1:5,nrow=1) y<-seq(10,50,10)
d<-rbind(x,y)
```

① d는 2*5행렬이다.

② d[1,]은 x와 같다.

③ d[,1]은 y와 같다.

④ d[1,1]의 값은 1이다.

3.R데이터 종류

예) matrix() 사용해 1~9로 구성된 3*3 행렬 만들기

	행렬 생성	행렬 접근
적용	x<-matrix(c(1,2,3,4,5,6,7,8,9),ncol=3) x	x[1,3]
결과	[,1] [,2] [,3] [1,] 1 4 7 [2,] 2 5 8 [3,] 3 6 9	7
적용	x<- matrix(c(1,2,3,4,5,6,7,8,9),ncol=3,byrow=TRUE) x	x[1.3]
결과	[,1] [,2] [,3] [1,] 1 2 3 [2,] 4 5 6 [3,] 7 8 9	3

Tip 다른 프로그램 언어는 첫 번째 원소에 대한 위치를 '0'으로 인식

3.R데이터 종류

Q7 다음 중 아래의 R코드 출력 결과?

```
m<-matrix(1:6,nrow=3)
m[m[,1]>1 & m[,2]>5] ?
```

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

3.R데이터 종류

Q7 다음 중 아래의 R코드 출력 결과?

```
 m<-matrix(1:6,nrow=3)
 m[m[,1]>1 & m[,2]>5,] ?
```

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

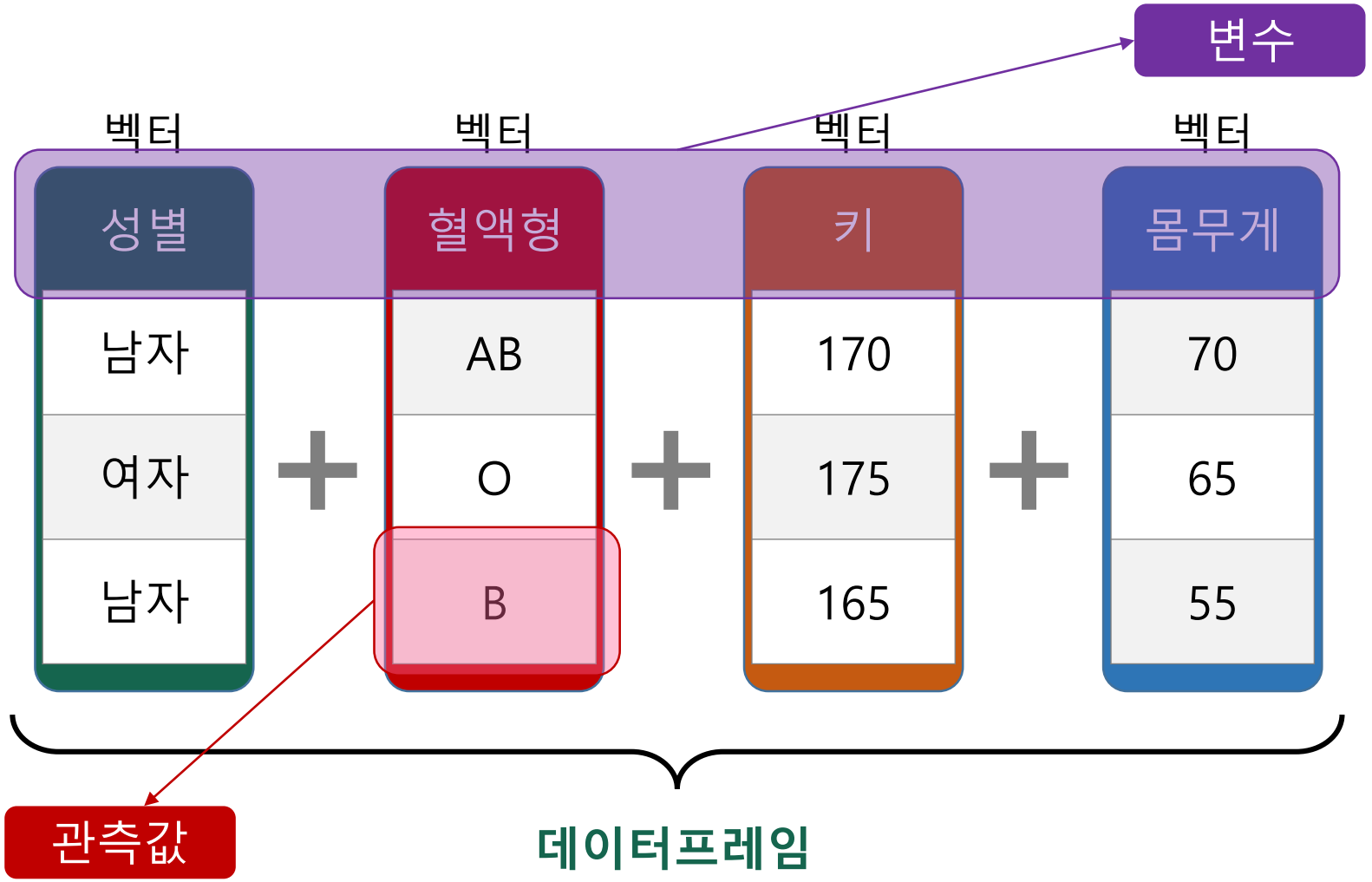
m[,1]>1 의 논리연산자 값은 ->[1] FALSE TRUE TRUE
m[,2]>5 의 논리연산자 값은->[1] FALSE FALSE TRUE
m[,1]>1&m[,2]>5 의 논리연산자 값은->[1] FALSE FALSE TRUE
이것은 다음과 같습니다.m[c(FALSE,FALSE,TRUE),] -> 이것은 m의 구성요소 중 TRUE가 있는 위치 요소를 의미합니다. 결국 m[3,] 를 의미합니다.

4.R데이터 종류

(3)데이터 프레임

- 엑셀의 스프레드시트와 같은 2차원 데이터 구조
- R에서 가장 중요한 데이터 타입
- 여러 가지 데이터 유형을 가질 수 있음
- 통계에서 변수가 모여 집합자료가 되듯이 R에서도 벡터가 모여 데이터 프레임이 된다. 데이터 분석은 변수의 종류에 따라 분석방법이 다르다.

4.R데이터 종류


변수
벡터
벡터
벡터
벡터
성별
혈액형
키
몸무게
남자
여자
남자
AB
O
B
170
175
165
70
65
55
관측값
데이터프레임

4.R데이터 종류

1)데이터프레임 생성함수

- data.frame( ) : stringsAsFactors(문자열을 factor로 저장할것인지,
  문자열로 저장할지 정하는 인자,기본값은 TRUE)
- str() : R 데이터 구조를 확인한다.
  (행의 개수, 열의 개수, 변수명 데이터의 유형을 확인 가능)

```
예) ID<-c(1,2,3,4)
    age<-c(29.30.31.32)
    gender<-c("f","m","f","m")
```

4.R데이터 종류

	데이터 프레임 생성	
적용	df1<-data.frame(ID,age,gender) df1	stringsAsfactors을 지정하지 않으면 문자열이 팩터(factor)로 저장
결과값	ID age gender 1 1 29 f 2 2 30 m 3 3 31 f 4 4 32 m	

4.R데이터 종류

	데이터 프레임 생성	
적용	str(df1)	데이터 구조 확인하기

결과값	data.frame': 4 obs. of 3 variables:
	$ ID : num 1 2 3 4
	$ age : num 29 30 31 32
	$ gender: Factor w/ 2 levels "f","m": 1 2 1 2	

4.R데이터 종류

	데이터 프레임 생성	
적용	df2<-data.frame(ID,age,gender,stringsAsfactors=FALSE) df2 str(df2)	

결과값	data.frame 4 obs. of 3 variables:
	$ ID : num 1 2 3 4
	$ age : num 29 30 31 32
	$ gender: chr "f" "m" "f" "m"
		stringsAsfactors= FALSE 문자열로 저장됨

Tip 데이터 프레임에 들어가 벡터의 길이가 다르면 에러 발생

4.R데이터 종류

2) 데이터 프레임 변수 선택하기

데이터 프레임 객체에 '$'를 붙이고 불러오고 싶은 변수의 이름을 적어주면 된다.

예) df$age

```
   ID  age  gender      -> data.frame을 df에  저장
1  1   29     f
2  2   30     m
3  3   31     f
4  4   32     m
```

df[[2]],df[["age"]],df$age 벡터 추출 -> [1] 29 30 31 32

df[2],df["age"] 데이터 프레임 추출

4.R데이터 종류

Q8 아래의 R 스크립트 결과에 대한 설명으로 가장 부적절한 것은

```
> str(airquality)
'data.frame':          153 obs. of 6 variables:
$ Ozone : int 41 36 12 18 NA 28 23 19 8 NA ...
$ Solar.R: int 190 118 149 313 NA NA 299 99 19 194 ...
$ Wind : num 7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
$ Temp : int 67 72 74 62 56 66 65 59 61 69 ...
$ Month : int 5 5 5 5 5 5 5 5 5 5 ...
$ Day : int 1 2 3 4 5 6 7 8 9 10 ..
```

① 6개의 변수와 153개의 관측치를 포함하고 있다.

② Day 변수를 벡터로 추출하기 위해서는 airquality$Day를 실행한다.

③ Solar.R 변수를 벡터로 추출하기 위해서는 airquality[[2]]를 실행한다.

④ Temp 변수를 데이터프레임으로 추출하기 위해서는 airquality[["Temp"]]를 실행한다.

5.R데이터 종류

(4)리스트

- 서로 다른 데이터 타입을 담을 수 있음
- list()함수 이용, '(KEY,값)' 형태의 연관배열

리스트	x<-list(name="kim",age=c(23)) x	의미
실행	$`name` [1] "kim" $age [1] 23	'x$key' 형태로 접근한다. 여기서 key='name' 의 값은 kim"
x[[1]]	[1] "kim"	key의 값만 출력
x[1]	$`name` [1] "kim"	(key,값)을 담고 서브리스트 출력

5.R데이터 종류

(5)array(배열)

- 행렬이 2차원 데이터라면 배열은 다차원 데이터이다.
- array(),dim=차원 지정 인자
- array(1:12, dim=c(3,4))

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

5.R데이터 종류

(5)array(배열) 인덱싱

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

x[1,1,1,] 값은 1

x[1,2,3] 값은 11

x[,,3] 값은

```
 [,1] [,2]
[1,]    9   11
[2,]   10   12
```

dim(x) -> 2 2 3 *dim() 차원을 알수 있음

5.R데이터 종류

(6)factor(범주형)

factor(x,levels,ordered) # x :팩터로 표현하고자 하는 값 (문자열 저장)
#levels : 값의 레벨
orederd : TURE 순서형, FALSE 명목형

```
gender<-factor("m",c("m","f"))
gender
[1] m
Levels: m f
levels(gender)<-c("male","female")
gender
[1] male
Levels: male female
```

5.R데이터 종류

Q9 **R에서 가능한 데이터 유형에 대한 설명으로 적절하지 않은 것은?**

① data frame은 숫자형과 문자형의 변수를 함께 포함할 수 없다.

② matrix는 차원을 가진 벡터로 숫자형 원소와 문자형 원소를 함께 포함할 수 없다.

③ list의 각 요소는 서로 다른 모드의 객체를 포함할 수 있다.

④ list의 각 요소는 [[]]로 접근 가능하다.

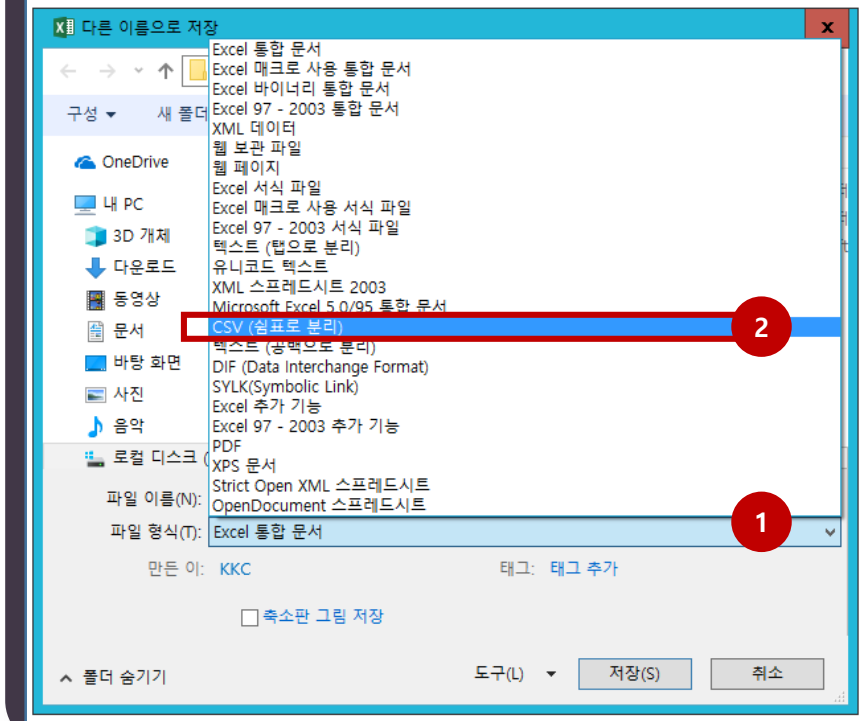
4. 외부데이터 불러오기와 저장하기

-데이터를 저장하는 파일형식은 다양함

-csv파일, txt 파일, 엑셀, spss, sas

Tip

setwd("c: \ \ data"), #getwd( ) 작업환경 설정 및 확인하기



① 파일형식 클릭

② *(파일이름)* adsp_name

(외부데이터) 다운 c의 data 폴더에 csv 파일로 저장

4. 외부데이터 불러오기와 저장하기

1)csv 파일 불러오기

- 엑셀 데이터의 특별한 형태로 콤마로 구분된 형태의 데이터
- csv 데이터는 read.csv() 함수 이용

```
df<-read.csv("c:/data/adsp_name.csv")  # 동일한 결과
df<-read.csv("c:\\data\\adsp_name.csv") # 동일한 결과
```

- 설정을 하지 않으면 기본값은 TRUE 설정

```
# header=T  (첫 번째행을 변수이름 설정)
# stringsAsFactors = TRUE (문자열을 범주형으로 인식)
# na.strings="a" ( "a" 저장된 문자열들은 NA로 저장)
```

4. 외부데이터 불러오기와 저장하기

- 첫 번째 행에 변수 이름 v1~v9 변수이름 부여하기

> df1<-read.csv("c:/data/adsp_unname.csv")

> head(df1)

```
  v1     v2     v3   v4 v5 v6 v7    v8    v9
1  1 김석현 남자 23   3 유   O 165.3 68.2
2  2 김도현 여자 22   2 무 AB 170.1 53.0
3  3 홍길동 남자 24   4 무   B 175.0 80.1
```

- names() 전체 변수 목록 확인

names(df1)<
-c("id","name","gender","grade","absense","age","bloodtype","height","weight")

4. 외부데이터 불러오기와 저장하기

- 특정 변수이름만 변경

```
names(df1)[7]<-"blood“
```

- id열만 삭제 #subset(x,select=열) x(data),열(데이터프레임 선택 열)

```
subset(df1,select = -id)
```

-ls() #자신이 만든 모든 변수목록 학인

-library(),installed.packages() #설치된 패키지 목록 확인

2)저장하기

```
-write.csv(df,"adsp_name.csv") # " " 큰따옴표 사이에 저장할 파일명
                               # row.names=FALSE, 행이름 제외 저장
```

4. 외부데이터 불러오기와 저장하기

3)txt 파일 불어오기

- txt 파일은 read.table() 함수 이용
- header=FALSE 기본설정

예) c 드라이브 adsp_name.txt 파일이 있고, df 객체에 저장

```
df<-read.table("c:/data/adsp_name.txt",header=T,sep=";")
#read.table() header 옵션이 기본 F고정, sep은 separate의 준말
 변수간 구분이 세미콜론,빈칸,등이 있을 때 사용)
```

adsp_name.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

이름	성별	나이	학년	휴학여부	혈액형	키
김석현	남자	23	3	유	O	165.3
김도현	여자	22	2	무	AB	170.1
홍길동	남자	24	4	무	B	175
김철수	남자	23	3	무	AB	182.1

4. 외부데이터 불러오기와 저장하기

4)R데이터 입출력

기본적으로 R에서 생성하였거나 수정한 데이터(또는 객체)를
저장하고, 불러오는 방법이 있다.

```
#일부의 데이터(또는 객체)를 저장하는 방법.
  save(x,file="  ") # x는 저장할 객체의 이름 " " 큰따옴표 안에 파일명

#iris라는 R데이터를 c드라이브의 data폴더에 iris1.RData라는 파일로 저장방법.
  data(iris)
  iris1 = iris
  save(iris, file="c:/data/iris1.RData")

#기본적으로 R프로그램이 종료되면, 저장되지 않은 데이터나 객체들은
자동적으로 사라진다. 하지만 위의 방법처럼 저장해 놓으면, R프로그램이
종료되었더라도, 지정된 파일에 저장되어 있다.

#저장된 데이터나 객체를 불러 오는방법.
load(file="c:/data/iris1.RData")
iris1
```

7.R의 기본적인 수치 계산

예) a<-1:10, b<-log(a)

구분	적용	결과
평균	mean(a)	5.5
분산	var(a)	9.166667
표준편차	sd(a)	3.02765
합	sum(a)	55
중앙값	median(a)	5.5
자연로그값	log(a)	생략
공분산	cov(a,b)	2.112062
상관계수	cor(a,b)	0.9516624
summary()	summary(a)	Min. 1st Qu. Median Mean 3rd Qu. Max. 1.00 3.25 5.50 5.50 7.75 10.00

5.R의 기초 통계 계산

예) a<-1:10, b<-log(a)

기초 통계		함수	결과	상세
평균	중심값	mean(a)	5.5	
분산		var(a)	9.166667	
중앙값		median(a)	5.5	평균보다 이상치에 덜 감하다
합		sum(a)	55	

5.R의 기초 통계 계산

기초 통계			함수	결과	상세
변동계수		흩어짐	표준편차/평균	측정단위가 서로 다른 데이터를 비교할때	변동계수 크다 ->편차가 크다
IQR (Interquartile Range) (사분위수범위)			IQR(a)	[1] 4.5 (3사분위수-1사분위수)	
범위			range(a)	[1] 1 10	최소값과 최대값
왜도	describe()		skewness(a)	'0' 정규분포 0보다크면 왼쪽으로 치운친분포	분포 모양 비대칭
첨도			kurtosis(a)	3보다 크면 정규분포보다 뽀족한 모량	뽀족한 정도

5.R의 기초 통계 계산

기초 통계		함수	결과	상세
사분위수	중심위치	quantile(a)	0% 25% 50% 75% 100% 1.00 3.25 5.50 7.75 10.00	50% 중위수 의미
자연로그값		log(a)	생략	
공분산		cov(a,b)	2.112062	
상관계수		cor(a,b)	0.9516624	
summary()		summary(a)	Min. 1st Qu. Median Mean 3rd Qu. Max. 1.00 3.25 5.50 5.50 7.75 10.00	50% 중위수 의미

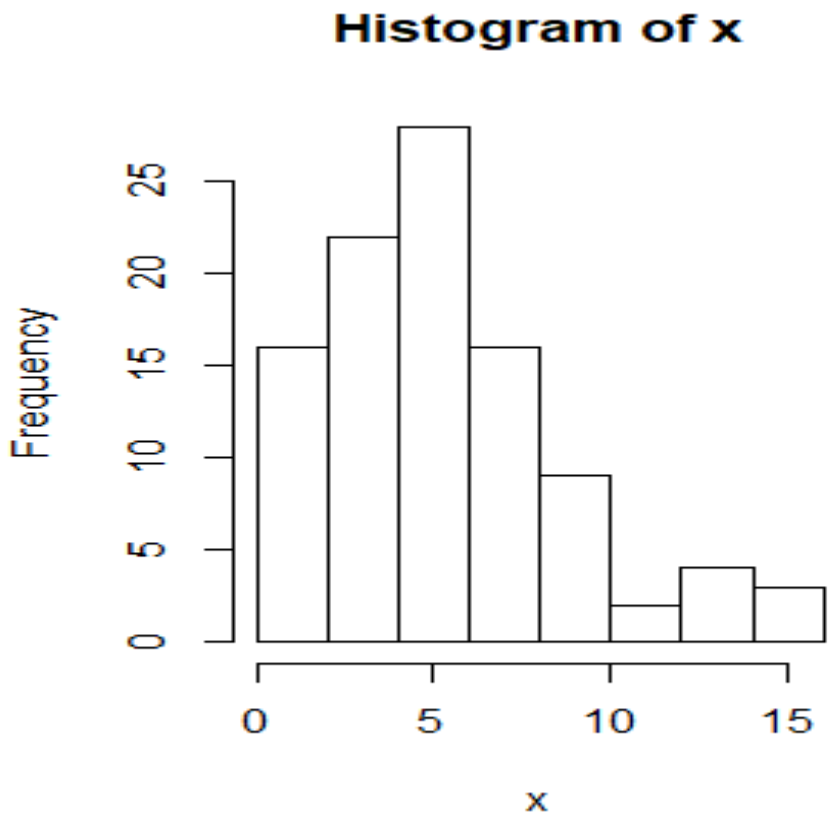
#참고로 describe()는 install.packages("psych")에 있음.
#na.rm=T 옵션은 mean,range,sd 등 수치형 변수에 NA값이 있으면 꼭 사용해야 합니다.
주의 summary() 사용할 때 범주형 열은 최소값, 최대값, 사분위수 등이 계산되지 않는다.

5.R의 기초 통계 계산

x<-rchisq(100,5)
#자유도가 5인 카이제곱분포에서 난수 100개를 만들면 오른쪽으로 기울어진 분포
> hist(x)
> library(psych)
> describe(x)

vars	n	mean	sd	median	trimmed	mad	min	max	range	**skew**	**kurtosis**	se
1	100	5.34	3.39	4.77	4.91	3.25	0.43	15.47	15.04	**1.02**	**0.73**	0.34

5.R의 기초 통계 계산



데이터가 왼쪽에 치우친 분포

시험에 출제되는 인덱싱 정리하기

R에서 특정 조건을 만족하는 데이터 선택 및 출력하는 작업
(슬라이싱=인덱싱=추출하기)

1.벡터

```
x<-c(1,2,3,4,5,6,7)
> x[1]
[1] 1
> x[c(5:7)]
[1] 5 6 7
> x[c(TRUE,TRUE,FALSE,FALSE,FALSE,TRUE,TRUE)]
[1] 1 2 6 7  # TRUE만 추출, R 패키지는 [0] 자리가 없음
```

시험에 출제되는 인덱싱 정리하기

2.행렬

행렬 인덱싱은 대괄호[i,j] # I번째 행, j번째 열 의미

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

시험에 출제되는 인덱싱 정리하기

3.데이터프레임 ★★

데이터프레임 인덱싱은 대괄호[i,j] # I번째 행, j번째 열 의미

대괄호 갯수에 따라 데이터 구조가 바뀜

예) data(iris)

데이터프레임 출력	벡터 값으로 출력
iris[1,]	iris[[1]]
iris[,c(1,2)]	iris$Sepal.Length
iris[1]	iris[["Sepal.Length"]]
iris["Sepal.Length"]	iris[,1]

8.R데이터 핸들링

(1)데이터 보기

- 데이터 값들이 어떻게 생겼는지를 확인
- 외부 데이터를 불러왔을 때 제대로 가져왔는지
- head(데이터명): 시작 데이터의 6행을 보여줌
- tail(데이터명): 끝 부분 6행을 보여줌

8.R데이터 핸들링

(2)데이터의 일부분 추출하기

gender	bloodtype	height	weight
F	AB	170	70
M	O	175	65
F	B	165	55

```
gender<-c("F","M","F")
bloodtype<-c("AB","O","B")
height<-c(170,175,165)
weight<-c(70,65,55)
df<-data.frame(gender,bloodtype,height,weight)
df
  gender bloodtype height weight
1 F          AB        170      70
2 M          O         175      65
3 F          B         165      55
```

8.R데이터 핸들링

-행렬 방식으로 열추출하기

데이터명[,인덱스]

예) df[,2]=df[,"bloodtype"]

[1] AB O B

Levels: AB B O

예)df[,c(2,3)]=df[,2:3]

	bloodtype	height
1	AB	170
2	O	175
3	B	165

6.R데이터 핸들링

(3)새로운 변수 만들기

- 데이터를 분석하는 과정에서 원자료를 이용하여 새로운 변수를 만듬
- 데이터명$새로운변수명<-함수 또는 수식

예) height와 weight의 합계를 나타내는 avg라는 새로운 변수를 생성

```
df$avg<-(height+weight)/3
df
  gender bloodtype height weight   avg
1 F      AB        170    70     80.00000
2 M      O         175    65     80.00000
3 F      B         165    55     73.33333
```

6.R데이터 핸들링

(4)데이터 조건으로 변수 선택

- 데이터 중에서 조건에 맞는 데이터 값을 추출하기
- subset()함수 이용

예) 키가 170이상인 데이터만 추출해서 df1에 저장하기

```
df1<-subset(df,subset=(height>=170))
df1
  gender bloodtype height weight
1 F          AB          170       70
2 M          O           175       65
```

6.R데이터 핸들링

(5)데이터 열 삭제하기

- 데이터 중에서 특정한 열을 삭제
- select() 함수 이용

예) gender 변수만 삭제하고, df2에 저장하기

```
df2<-subset(df,select=-gender)
df2
 bloodtype height weight
1   AB         170      70
2   O          175      65
3   B          165      55
```

8.R데이터 핸들링

(6)데이터 변수명 바꾸기

-colnames()함수는 데이터 프레임으로부터 이름만 가져와 벡터로 보여줌

예) bloodtype을 blood로 수정하기

```
    colnames(df)[2]<-“blood”
 gender blood height weight
1 F      AB       170     70
2 M      O        175     65
3 F      B        165     55
```

6.R데이터 핸들링

(7)병합하기

-merge(x,y,by)

#x(병합할 데이터프레임),y(병합할데이터프레임),by(병할기준 열)

(8)시뮬레이션에서 동일한 난수 발생하도록 초기화 함수

-set.seed()

7.1반복문

반복문

	for 반복 구문	while 반복 구문
형식	for(i in data) {i를 사용한 문장}	while(cond){조건이 참일 때 수행할문장}
의미	data에 들어있는 각각의 값을 변수 i에 할당하면서 각각에 대한 블록안의 문장 수행 몇회 반복인지 알수가 있다.	조건 cond가 참일 때 중괄호안의 문장을 수행한다. for 구문과 다른점은 중괄호 구문이 만족할 때까지 반복되므로 몇 회 반복될지 미래 정해지지 않는다.

적용	a<-c()
	for(i in 1:9){
	a[i]=i*i
	}
	a
		x=1
		while(x<5){
		x=x+1
		print(x)
		}

결과	1 4 9 16 25 36 49 64 81	2 3 4 5

7.2조건문

조건문

	if	if~else
형식	if 조건	if 조건1 else 조건2
의미		if~else의 경우 조건이 만족하지 않은 경우 else이하의 조건을 이용함

적용	x<-c(1,3,5,7,9)
	y<-c(2,4,6,8,10)
	if(sum(x)<sum(y)){print(y);print(sum(y));}
	# 1개 이상의 결과일 때 중괄호
		if (sum(x)>sum(y))
		{
		print(x)
		}else{
		print(y)
		}

결과	[1] 2 4 6 8 10 [1] 30	[1] 2 4 6 8 10

7.3 조건문

ifelse (더미변수 만들 때 사용)

-ifelse 데이터분석(회귀분석) 할 때 더미변수에 이용★

-ifelse(데이터,비교구문,TURE일때 실행구문,FALSE일대 실행구문)

  예) gender<-c("남자","남자","남자","여자","여자")

문자열 데이터 남자를 0 또는 여자 1 변환 할때

```
gender<-ifelse(gender=="남자",0,1)
gender
[1] 0 0 0 1 1
```

7.4 사용자 정의함수

해설 위의 함수는 r에 내장된 함수이고, 사용자 정의 함수는
분석자가 정의한 함수입니다.
Function 명령을 이용해서 함수를 정의할 수 있습니다.

```
func1<-function(aa)
{
  isum<-0
for(i in 1:aa){
isum=isum+i
}
print(isum)
}
# 함수이릉으로 호출
func1(3)  #    [1] 6
```

7.반복문

Q10 다음은 반복구문에 대한 설명 중 옳은 것은?

① for 구문은 괄호 안의 조건이 만족되어 있는 동안 이후의 구문을 반복한다.

② while 구문은 반복되는 구문 내에서 반복변수 i를 변화시켜 주어야 한다.

③ for 구문이 반복되는 횟수는 실행시키기 전까지 알 수 없다.

④ while 구문은 for 구문보다 빠르게 실행된다.

10.자료의 데이터 타입 변환

- R에서의 객체는 다양한 형태를 가짐.

  예) 문자형,논리연산자, factor,데이터 프레임

- 데이터 타입 변환 함수

함수	의미
as.factor(x)	주어진 객체x를 팩터(factor)로 변환
as.numeric(x)	주어진 객체x를 숫자로 저장한 벡터로변환
as.character(x)	주어진 객체x를 문자열로 저장한 벡터로 변환
as.data.frame(x)	주어진 객체x를 데이터 프레임으로 변환
적용	**결과값**
as.integer(3.14)	3
as.numeric(FALSE)	0
as.logical(0.45)	TRUE

Tip 논리값인 TRUE와FALSE를 수치형으로 변환할 때는 FALSE->0,TRUE->1
반대로 논리값으로 변경할 때에는 '0'인 경우에만 FALSE, '0'이 아니면 TRUE가 된다.

10.자료의 데이터 타입 변환

Q11 다음 자료형 데이터 구조 변환의 결과로 틀린 것은?

① as.numeric(TRUE): 1

② as.logical(1.2): FALSE

③ as.character(TRUE):"TRUE"

④ as.integer(FALSE):0

10.자료의 데이터 타입 변환

Q12 **아래는 쥐(Rat)의 먹이 종류(Diet)에 따른 무게(Weight)의 시간(Time) 변화를 측정한 자료의 요약이다. 다음 중 이에 대한 설명으로 부적절한 것은?**

➢ summary(BodyWeight)

```
     weight           Time           Rat          Diet
 Min.   :225.0   Min.   : 1.00   2      : 11   1:88
 1st Qu.:267.0   1st Qu.:15.00   3      : 11   2:44
 Median :344.5   Median :36.00   4      : 11   3:44
 Mean   :384.5   Mean   :33.55   1      : 11
 3rd Qu.:511.2   3rd Qu.:50.00   8      : 11
 Max.   :628.0   Max.   :64.00   5      : 11
                                 (Other):110
```

① Diet의 유형은 숫자형(Numeric)으로 인식한다.
② 데이터는 총 3가지 종류의 먹이를 포함한다
③ 명령어 'mean(BodyWeight$Diet)'는 오류를 발생시킨다.
④ Weight의 중위수는 344.5이다.

9.특수한 기능들

(1)paste

paste 명령어는 입력 받은 문자열들을 하나로 붙여준다.

number<-1:5

alphabet<-c("a","b","c")

paste(number,alphabet)

[1] "1 a" "2 b" "3 c" "4 a" "5 b"

paste(number,alphabet,sep="to the") # 'sep=' 옵션을 통해 붙이고자 하는 문자열들 사이에 구분자(separator) 삽입

[1] "1 to the a" "2 to the b" "3 to the c" "4 to the a" "5 to the b"

9.특수한 기능들

내장 데이터 airquality에 paste 함수를 이용해서 Month변수와

Day변수 구분자 "-" 결합된 Month-Day 새로운 변수 만들기★

```
> head(airquality)
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
attach(airquality)
airquality$Month.Day<-paste(Month,Day,sep = "-") ★
head(airquality)
    Ozone Solar.R Wind Temp Month Day Month.Day
1    41     190     7.4   67     5     1       5-1
2    36     118     8.0   72     5     2       5-2
3    12     149    12.6   74     5     3       5-3
```

9.특수한 기능들

(2)substr

paste와 반대로 주어진 문자열에서 특정 문자열 추출

country<-c("korea","Japan")

substr(country,1,2) #country 객체명에 시작위치 끝위치만 추출한 결과이다.

[1] "ko" "Ja"

9.특수한 기능들

(3)strsplit 함수

strsplit(x,split=“,”) 문자형 벡터 x를 split 기준으로 분리

```
nation<-c("korea,seoul","japan,tokyo")
nation_split<-strsplit(nation,split = ",")
nation_split
[[1]]
[1] "korea" "seoul"
[[2]]
[1] "japan" "tokyo"
```

11.R그래픽 이해

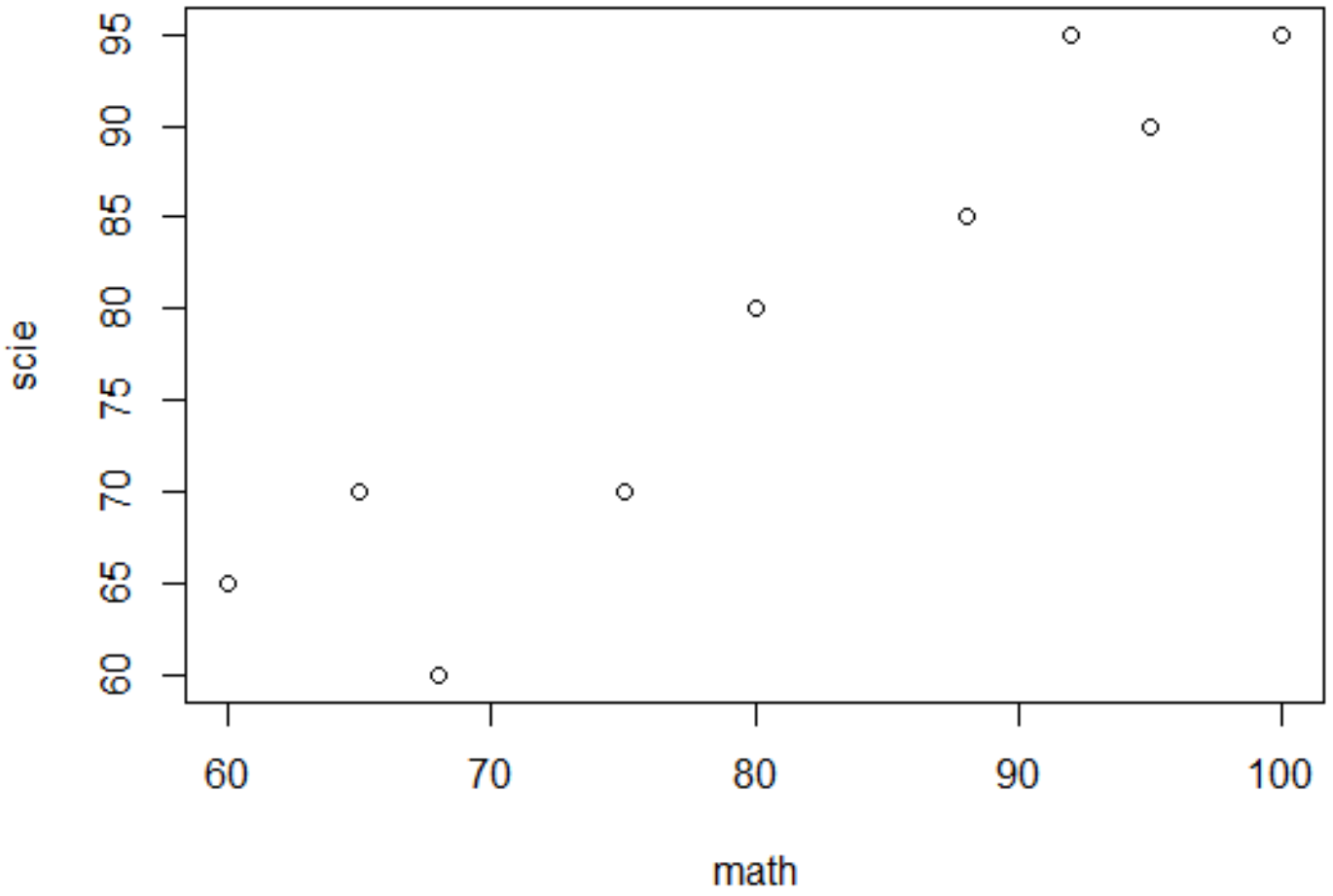
(1)산점도(Scatter Plot) 그래프

- 두 양적 자료 간의 관계(선형)를 살펴보기 위해서 작성하는 그래프
- plot(x,y)

예) 이 자료는 중학교 중간고사 수학점수와 과학점수를 이용하여
산점도를 그려보자

```
math<-c(95,65,80,92,60,75,88,100,75,68)
scie<-c(90,70,80,95,65,70,85,95,70,60)
plot(math,scie)
```

11.R그래픽 이해

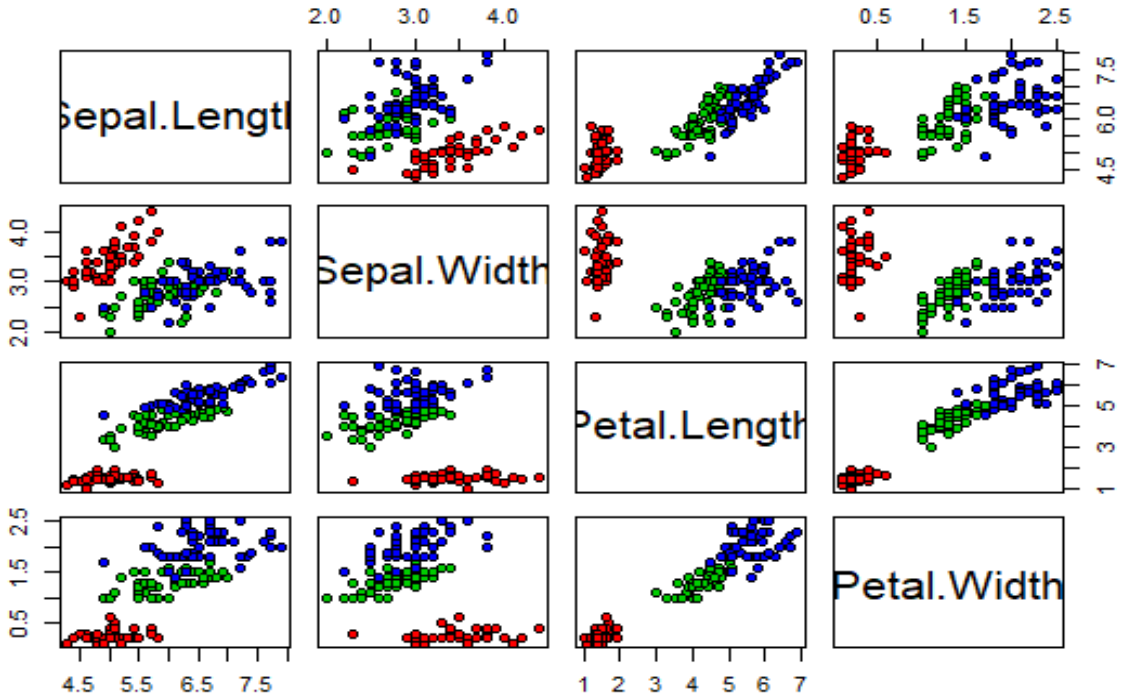


예) 산점도

11.R그래픽 이해

(2)산점도 행렬

여러가지 변수들에 대해서 각각의 산점도를 한눈에 살펴볼 수 있도록 확장된 산점도 행렬이다.



예) iris data 산점도 행렬

11.R그래픽 이해

Q13 **분포 패턴이 다양한 자료에서 같은 상관계수가 도출될 수 있다. 그 패턴을 확인하기 위한 분석으로 적절한 것은?**

① 상자그림
② 산점도
③ 빈도표
④ 히스토그램

11.R그래픽 이해

(3)교차표(분할표,contingency table)

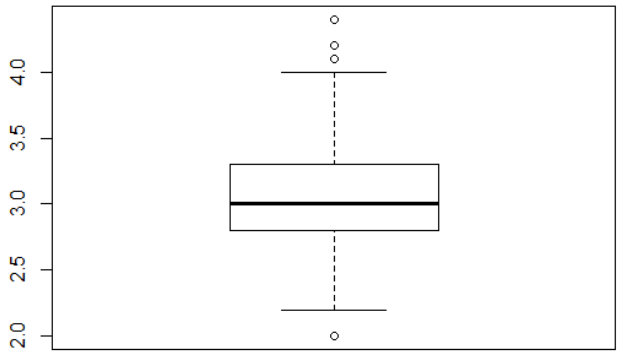
- 두 변수를 만족하는 관측치가 각각 몇 개씩인지 파악

- table()

```
gender<-c("F","M","F")
bloodtype<-c("AB","O","B")
height<-c(170,175,165)
weight<-c(70,65,55)
df<-data.frame(gender,bloodtype,height,weight)
df1<-table(df$gender,df$bloodtype)
      AB B O
  F   1  1  0
  M 0  0  1
```

11.R그래픽 이해

(4)상자 그림(boxplot)

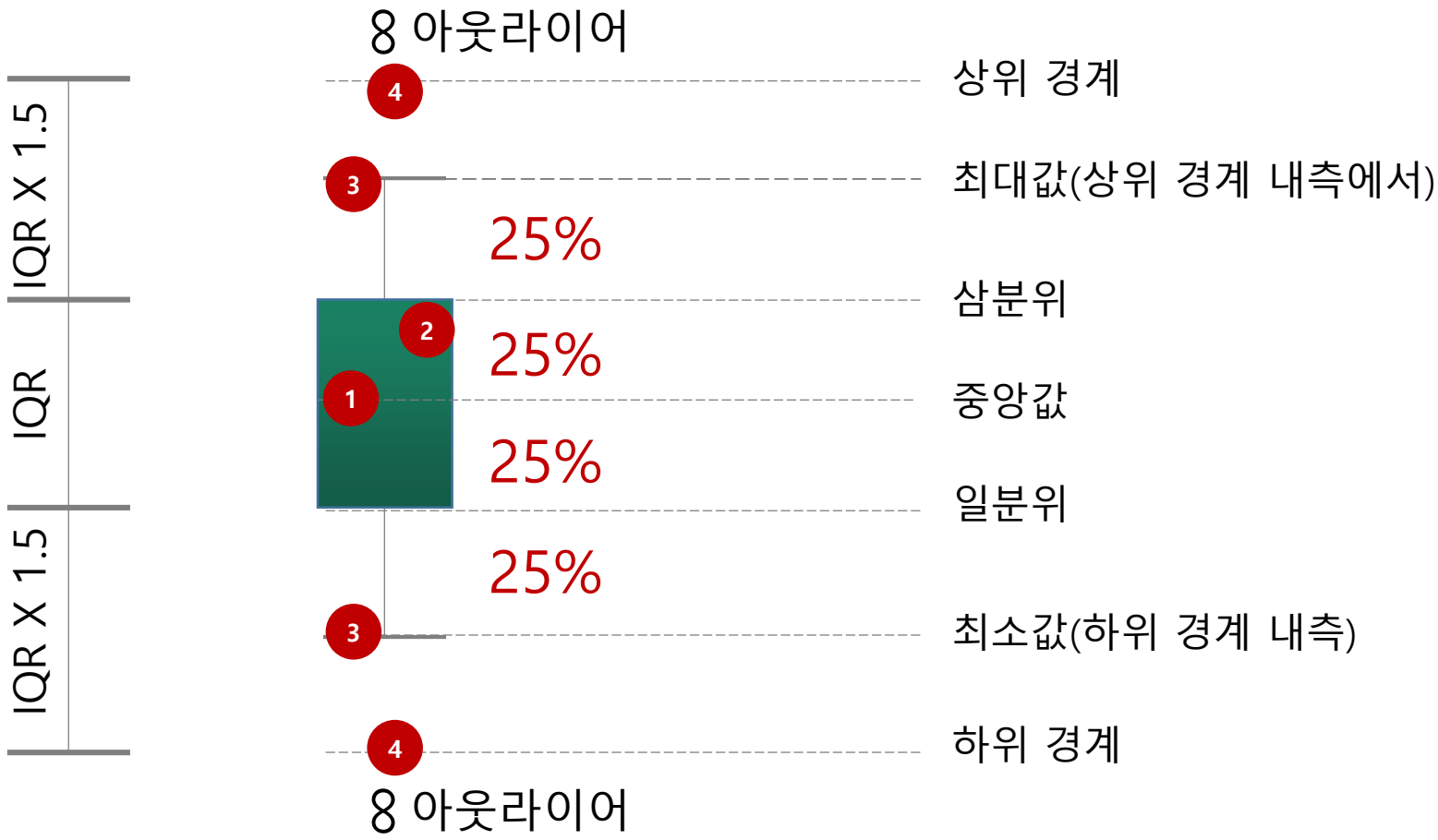


- 상자그림은 데이터의 분포를 보여주는 그림
- 가운데 상자 1사분위수,중앙값, 3사분위수 의미
- 상자의 상하로 뻗어나간 선(whisker)은
  '중앙값-1.5*IQR,중앙값+1.5IQR
- IQR은 제3사분위수-1사분위수 의미

- 그림에서 보이는 점들은 이상치(outlier),lower whisker 보다
  작은 데이터 1개 upper whisker 보다 큰 3개가 보인다.
- boxplot()
- 위 상자그림은 iris$Sepal.Width이며, 중앙값은 약3.0,1사분위수
  약2.8,3사분위수약 3.3 lowerwhisker 약 2.2,upper whisker 약 4.0 이다.

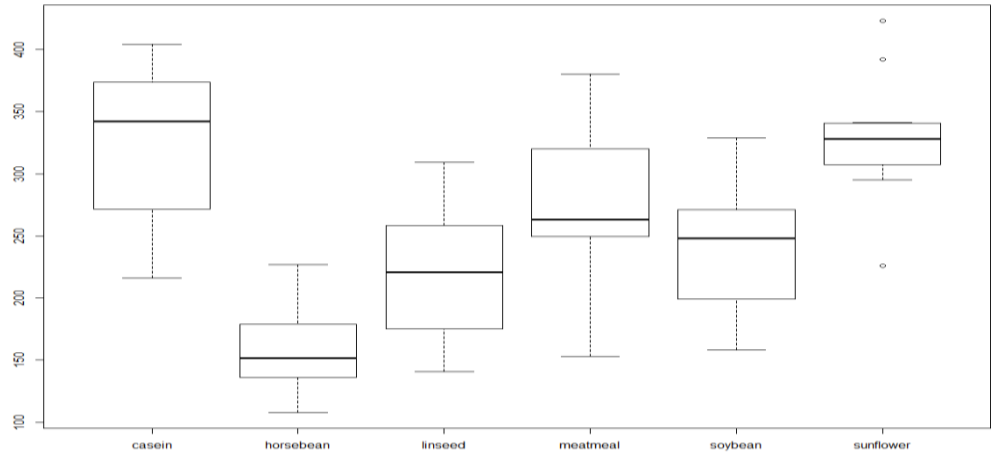
10.R그래픽 기능

(4)상자 그림(boxplot)



11.R그래픽 이해

Q14 **아래 그림은 닭 사료의 종료(feed)와 닭의 성장(weight)의 관측치를 포함하고 있다. 이를 통해 추론 가능한 사실로 올바르지 않은 것은?**



① casein이 포함된 사료를 먹은 닭의 몸무게 중위수가 가장 크다.
② sunflower가 포함된 사료를 먹은 닭 중 이상치의 몸무게를 가진 닭이 3마리 있다.
③ horsebean이 닭의 성장을 촉진하는데 가장 효율성이 떨어진다.
④ horsebean에 비해 meatmeal을 먹는 닭들의 몸무게의 분산이 더 작을 것이다.

11.R그래픽 이해

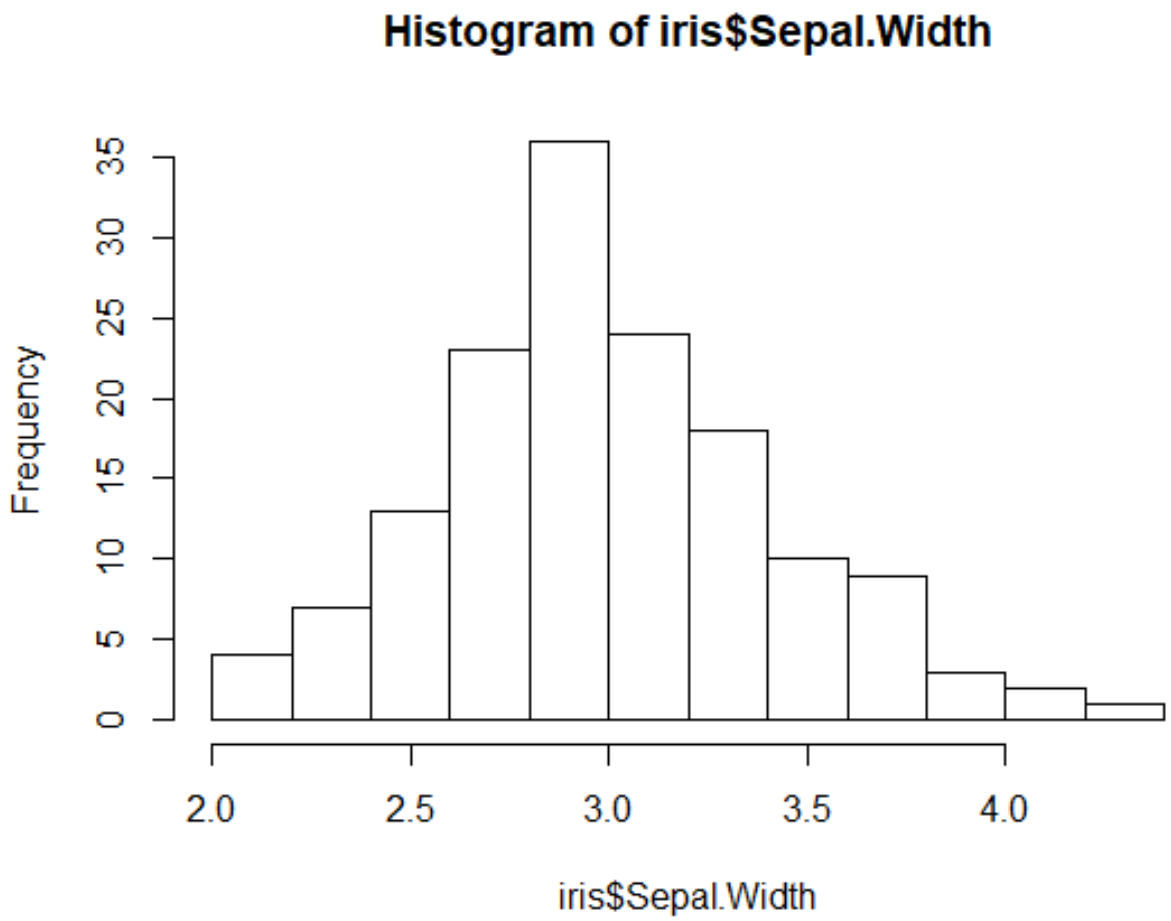
(5)히스토그램

- 데이터의 분포를 알 수 있고, 히스토그램은 값의 범위마다 빈도를 표시한 그래프
- hist()
- 히스토그램의 주요 파라미터 중 하나는 frq고, 각 구간의 확률밀도가 그려진다.
- 확률밀도므로 막대 너비의 합은 1이 된다. 도수밀도(확률밀도) =계급의도수/계급의간격

Tip 확률밀도 각 구간에 얼마만큼의 데이터가 속해 있는지를 확률로 나타내 값

예) hist(iris$Sepal.Width)

11.R그래픽 이해


Histogram of iris$Sepal.Width
Frequency
0
5
10
15
20
25
30
35
2.0
2.5
3.0
3.5
4.0
iris$Sepal.Width

10.R그래픽 기능

한 화면에 여러그림 그리기

```
Par(mfrow=c(2,2))
Hist(iris$Sepal.Length,breaks = 20)
Hist(iris$Sepal.Length,breaks = 20,probability = TRUE)
Boxplot(iris$Sepal.Length~iris$Species,data=iris)
Boxplot(iris$Sepal.Length~iris$Species,data=iris)
```

원래대로 돌아가기

```
par(mfrow=c(1,1))
```

11.R그래픽 이해

Q15 히스토그램에서 각 계급에 대한 사각형의 면적을 무엇이라고 하는가?

① 도수
② 누적도수
③ 중앙값
④ 계급의 크기

11.R그래픽 이해

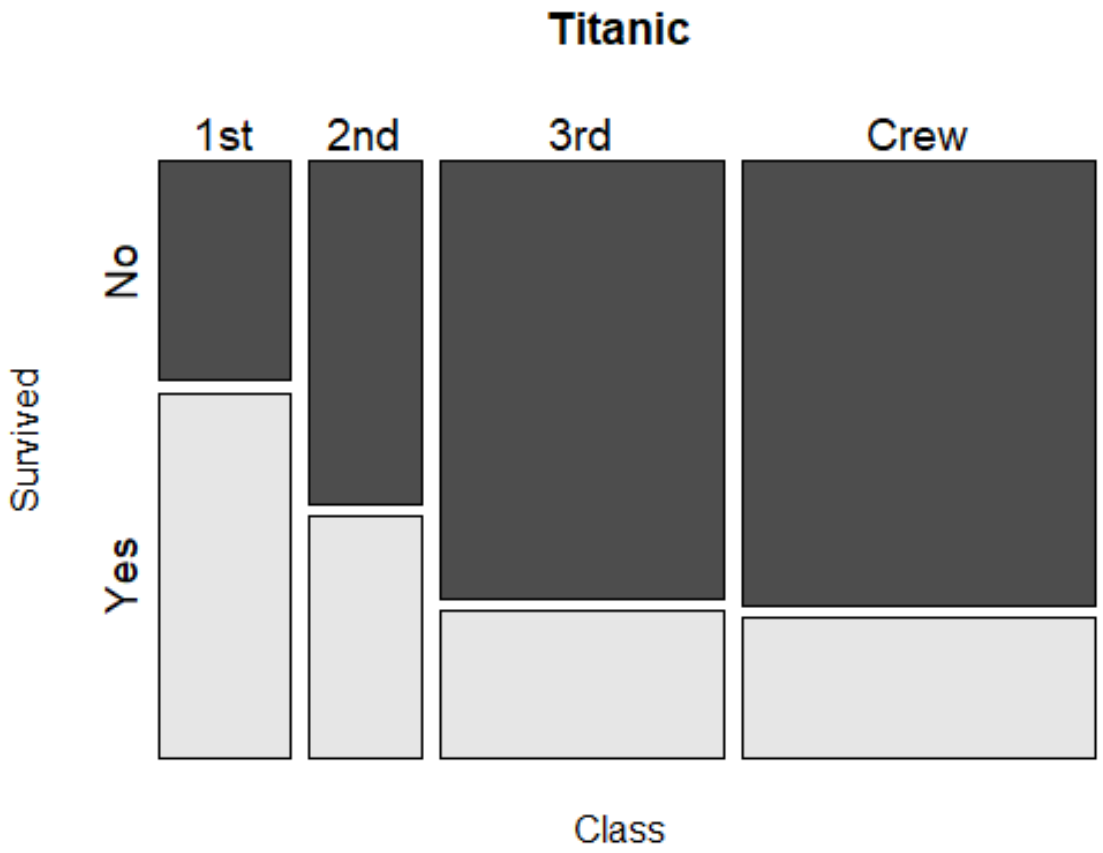
(6)모자이크 플롯(mosaicplot)

- 모자이크 플롯은 범주형 다변량 데이터를 표현하는 데 적합한 그래프이다.
- 모자이크 플롯에는 사각형들이 그래프에서 나열되며, 각 사각형의 넓이가 각 범주에 속한 데이터의 수에 해당된다.
- mosaicplot()

예) 타이타닉호 탑승실 등급과 생존 여부의 모자이크 플롯

```
mosaicplot(~Class+Survived,data=Titanic,color=TRUE,cex=1.2)
```

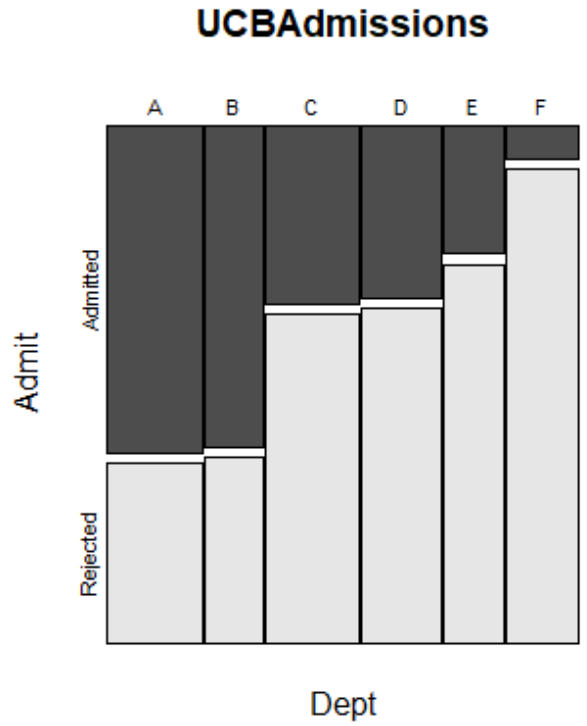
10.R그래픽 기능



11.R그래픽 이해

Q16

아래의 그림은 한 대학의 합격자 현황에 대해 학과(Dept)와 합격여부(Admit) 변수를 사용해 그린 모자이크 플롯이다. 학과는 A~F 6개 학과가 있고, 합격여부는 Admitted(합격)과 Rejected(불합격)로 구분된다.
아래 그림에 대한 설명하는 보기 중 부적절한 것은?



① E학과의 지원자 수가 가장 많다.

② B학과의 지원자 수가 가장 적다.

③ F학과의 합격률이 가장 낮다.

④ A학과의 합격률이 가장 높다.

11.R그래픽 이해

Q17

아래는 호흡기 환자의 상태(status:poor,good)와 새로운 치료방법의 적용 여부(treatment,placebo), 각 환자의 상태를 관찰한 시점 (month:0,1,2,3,4)변수를 사용한 모자이크 플롯이다. 올바른지 않은 설명은?

① month가 1인 시점에서는 placebo와 treatment 모두 상태가 좋은(good)환자가 많다.
② month가 0~3까지는 treatment 그룹은 상태가 좋은(good)환자가 증가하는 경향이 있다.
③ treatment의 month=4인 경우 치료방법의 효과가 month=3보다 감소한다.
④ 시간의 흐름에 따라 placebo 그룹에 속한 환자 수의 비율이 treatment 그룹에 배해 증가

10.R그래픽 기능

그래픽 기능 정리하기★★

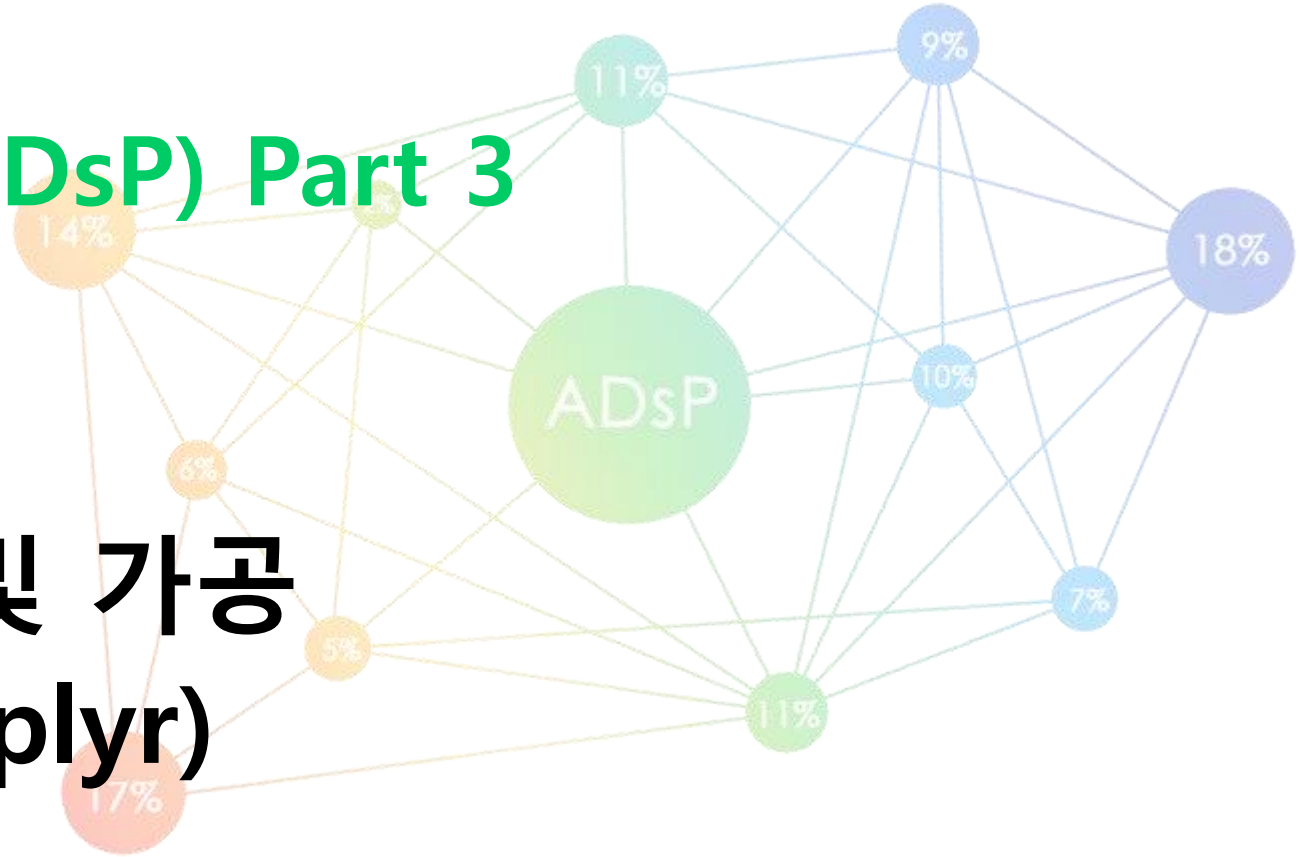
	함수	중요
삼전도	plot(x,y) plot(y~x)	2개 수치형변수의 직선=선형=상관 관계를 알아보기
산전도행렬	pairs()	여려개의 변수관계를 알아보기
상자그림	boxplot()	이상치존재 IQR , 최소, 최대 1사분위 3사분위 중위값 확인 NA 기본이 제거하고 그려짐
히스토그램	hist()	연속형수치에 적합 히스토의 사각형 상대도수의 의미
모자이크 플롯	mosaicplot()	사각형의 크기는 데이터의 수를 의미
막대 그래프	barplot()	명목형 변수의 빈도에 활용(교차표) 막대사이가 끊겨져 있는 모양

데이터분석준전문가(ADsP) Part 3

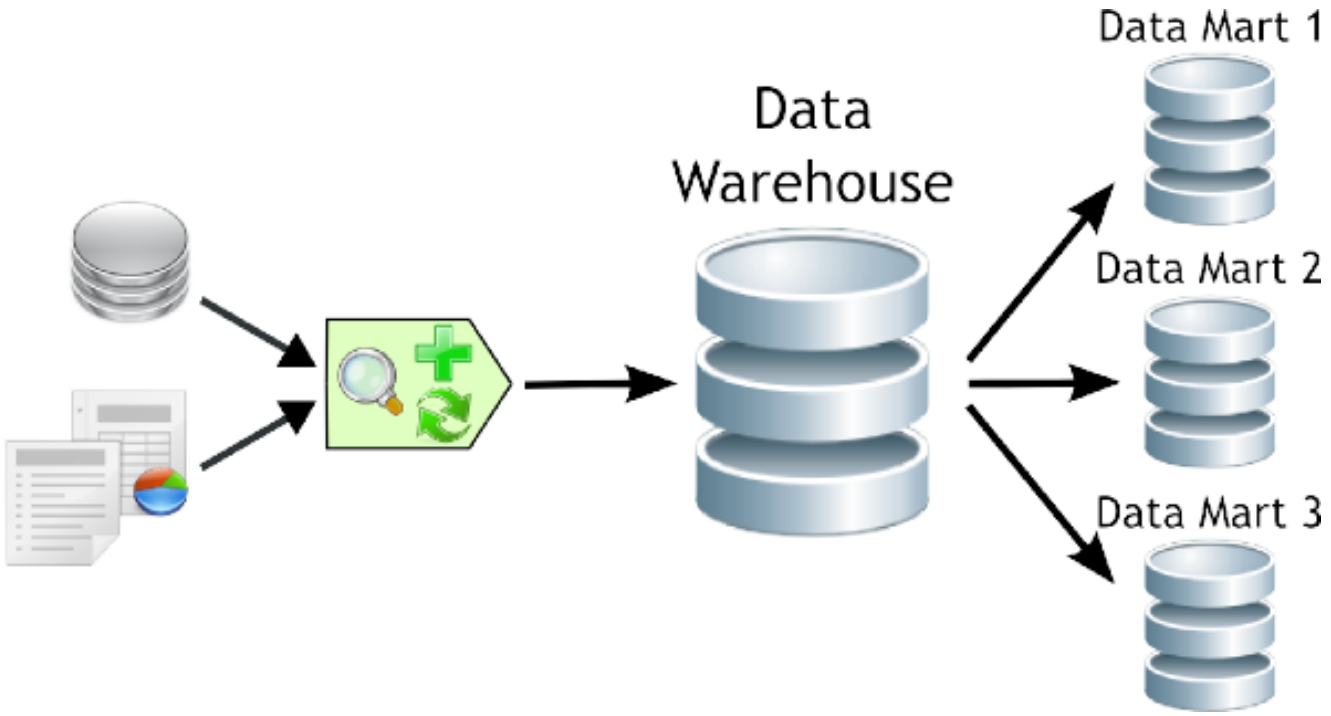
데이터분석

05

12. 데이터 처리 및 가공 (reshape, apply, plyr)



12.데이터 가공



데이터웨어하우스(DW) 데이터 마트(DM)

데이터 웨어하우스는 최적화되어 있지 않고 비효율적으로

배치된 상태라면 데이터마트는 사용하기 쉽게 시스템에 최적화된 곳

12.데이터 가공

#최적화된 데이터 마트를 위해 사용하는 패키지가

reshape패키지, apply 함수들, plyr패키지, dplyr패키지,

data.table()등이 사용

12.데이터 가공

(1)iris data

데이터 가공 및 데이터마이닝 예제로 사용할 iris dataset은 다음과 같다.

- Species: 붓꽃의 종. setosa, versicolor, virginica의 세가지 값 중 하나를 저장한 범주형 변수.
- Sepal.Width: 꽃받침의 너비. Number 변수.
- Sepal.Length: 꽃받침의 길이. Number 변수.
- Petal.Width: 꽃잎의 너비. Number 변수.
- Petal.Length: 꽃잎의 길이. Number 변수

```
> head(iris)
```

12.데이터 가공

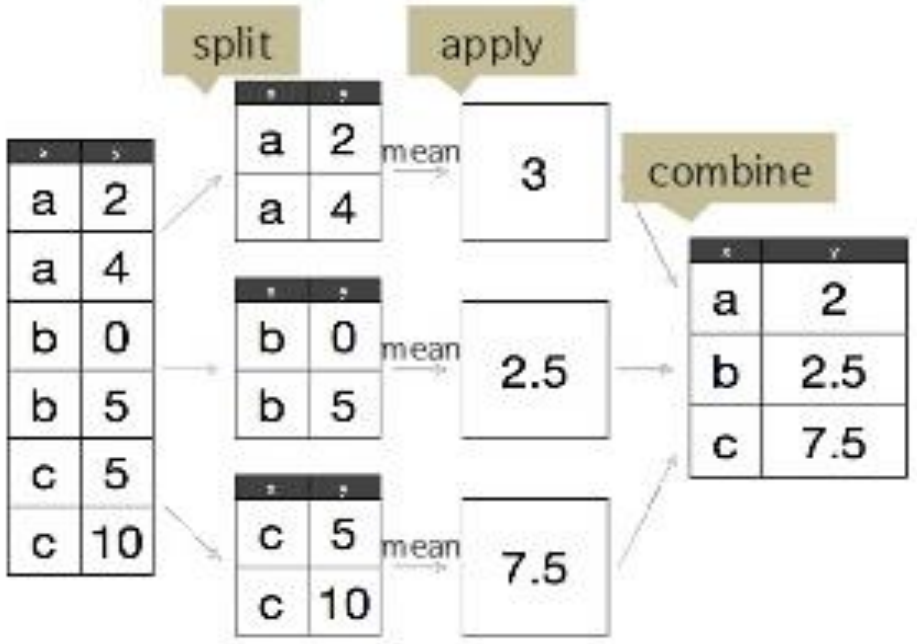
	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa

```
str(iris)
'data.frame':	150 obs. of  5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

12.plyr패키지

apply 함수에 기반해 데이터와 변수를 동시에 배열로 치환

split->apply->combine 기능 제공 즉 데이터분할->함수를 적용->재결합



12.plyr패키지

(1)apply 함수들 (lapply, sapply, tapply)★★

데이터프레임이나 스프레드시트의 형태의 행*열 구조는 apply의 간단한 명령으로 필요한 계산 및 자료의 요약이 가능하다.

함수	상세
apply	apply(x,margin,fun)★ # x에 fun을 margin 방향으로 적용 결과를 벡터,배열,리스트 반환 # margin 1이면='행',margin 2이면='열',c(1,2) 행,열방향 의미 > a<-matrix(1:6,ncol=2) > a [,1] [,2] [1,] 1 4 [2,] 2 5 [3,] 3 6

12.plyr패키지

함수	상세

apply	> apply(a,1,sum)
	#행렬의 각 행의 합(1+4,2+5,3+6)
	[1] 5 7 9
	
	#iris dataset 중 factor열인 Species 제외한 데이터프레임만 가져와 열의 합을 구함
	apply(iris[,-5],2,sum)
	
	Sepal.Length Sepal.Width Petal.Length Petal.Width
	876.5 458.6 563.7 179.9
	#colSums() 통해서도 같은 값을 구할수 있다.
	> colSums(iris[,-5])
	Sepal.Length Sepal.Width Petal.Length Petal.Width
	876.5 458.6 563.7 179.9
	
	#각행과 열의 합과 평균의 함수는
	colSums(),colMeans(),rowSums(),rowMeans()

12.plyr패키지

함수	상세
lapply	-lapply(x,함수) # x는 벡터 또는 리스트,데이터프레임 적용해서 리스트로 반환 > x<-list(c(1:10),c(1,3,5,7,9),c(4,5,6,7,8,9,10)) > x [[1]] [1] 1 2 3 4 5 6 7 8 9 10 [[2]] [1] 1 3 5 7 9 [[3]] [1] 4 5 6 7 8 9 10

12.plyr패키지

함수	상세
lapply	> lapply(x,mean) [[1]] [1] 5.5 [[2]] [1] 5 [[3]] [1] 7

12.plyr패키지

함수	상세
sapply	sapply()는 lapply()와 유사하지만 리스트대신 행렬, 벡터 등으로 결과를 반환하는 함수이다.입력으로는 벡터, 리스트, 데이터 프레임등이 쓰일 수 있다. > sapply(x,mean) [1] 5.5 5.0 7.0 # 벡터로 반환 > sapply(x,quantile,probs=(1:3)/4) # quantile(사분위수)중 probs(확률)=(1:3/4)=1/4(25%),2/4(50%),3/4(75%) [,1] [,2] [,3] 25% 3.25 3 5.5 50% 5.50 5 7.0 75% 7.75 7 8.5 # 행렬로 출력

12.plyr패키지

함수	상세
taaply()	tapply함수는 그룹별 처리를 위한 apply 함수로서 tapply(데이터,색인,함수) 색인은 factor 의미 > tapply(iris$Sepal.Length,iris$Species,mean) setosa versicolor virginica 5.006 5.936 6.588 # iris Species(범주형)3종류의 Sepal.Length의 평균을 의미 > tapply (1:5 , 1:5 %% 2 == 1, sum) # %% 나머지를 구하는 연산자 FALSE TRUE #1:5 %% 2 == 1는 FALSE/TRUE 구분 6 9 #FALSE(2,4), TRUE(1,3,5)

2.plyr패키지1

(2)plyr패키지 (adply, ddply)★★

데이터처리함수형식

(입력데이터)(출력데이터)ply 5글자의 함수명으로 이름이 지어진다.

	array	Data frame	list	nothing
array	apply	adply	alply	a_ply
Data frame	daply	ddply	dlply	d_ply
list	laply	ldply	llply	l_ply
n replicates	raply	rdply	rlply	r_ply
Function arguments	maply	mdply	mlply	m_ply

12.plyr패키지

1)adply()

#배열,행렬,데이터프레임을(a)을 받아 데이터 프레임(d)을 출력하는 함수.

#adply(data,margins,fun)

```
install.packages("plyr")
library(plyr)

apply(iris,1,function(row){print(row)}) # apply 함수내 사용자 정의함수로 iris 출력

epal.Length   Sepal.Width Petal.Length   Petal.Width       Species
       "5.1"         "3.5"         "1.4"         "0.2"      "setosa"
```

2.plyr패키지


```
Sepal.Length  Sepal.Width Petal.Length  Petal.Width       Species
       "4.9"        "3.0"        "1.4"        "0.2"     "setosa"

#apply 함수는 벡터,행렬,리스트로 반환하는 함수, 결과가 한행이면
벡터,여러행이면 행렬, 각행마다 컬럼수가 다르면 리스트 결과 반환.
iris는 5개변수가 열이 동일하여 행렬로 반환.
행렬=동일한 데이터 타입만 가능.
iris$Species 문자형 데이터로 전체가 문자로 출력됨

adply(iris,1,function(row){row$Sepal.Length>=5.0&row$Species=="setosa"})
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species    V1
1            5.1         3.5          1.4         0.2     setosa  TRUE
2            4.9         3.0          1.4         0.2     setosa FALSE
3            4.7         3.2          1.3         0.2     setosa FALSE
# adply함수 데이터프레임으로 결과 출력하고, species=setosa 확인하고 그 결과를
 v1나타냄.  즉 데이터프레임 다른 데이터 타입이 가능하여 문자형으로 표시가 안됨
```

12.plyr패키지

2)ddply

#데이터 프레임을 분할하고 함수에 적용 후에 결과를 데이터프레임으로 반환

#ddlpy(data,variables,fun) variables= 그룹핑 변수로 .() 형식임

adply와ddply 차이는 adply() 행 또는 컬럼 단위로 함수를 적용, dply()는 variables에 나열한 컬럼에 따라 그룹으로 나눈뒤에 함수 적용

```
ddply(iris,.(Species),function(sub){data.frame
(Sepal.width.mean<-mean(sub$Sepal.Width))})
     Species Sepal.width.mean....mean.sub.Sepal.Width.
1     setosa                                      3.428
2     versicolor                                  2.770
3     virginica                                   2.974
```

12.유용한 함수들

(1)split()

함수	상세
split()	split() 데이터를 분리할 때 사용 split(데이터,분리조건) > split (iris , iris $ Species) $`setosa` Sepal.Length Sepal.Width Petal.Length Petal.Width Species 1 5.1 3.5 1.4 0.2 setosa 2 4.9 3.0 1.4 0.2 setosa 이하생략 # split() 반환값은 리스트

12.유용한 함수들

(2)subset(), select()

함수	상세
subset()	특정 부부만 추출하는 용도로 사용 # Species의 setosa 중 Sepal.Length > 5.0 인 것만 추출 > subset (iris ,Species=="setosa"& Sepal.Length > 5.0) # &=and 의미 Sepal.Length Sepal.Width Petal.Length Petal.Width Species 1 5.1 3.5 1.4 0.2 setosa 6 5.4 3.9 1.7 0.4 setosa
select()	subset에 select 인자를 지정하면 특정 열을 선택하거나 제외용도로사용 # Sepal.Length와 Species열을 iris에서 선택 > subset (iris ,select =c(Sepal.Length , Species)) Sepal.Length Species 1 5.1 setosa 2 4.9 setosa

12.유용한 함수들

(3)attach(), detach()

함수	상세
attach()	·attach는 데이터프레임에 곧바로 접근할 수 있게 한다. # $없이 데이터프레임에 접근할 수 있다. attach(iris) > head(Sepal.Length) # $없이 접근할 수 있다. [1] 5.1 4.9 4.7 4.6 5.0 5.4
detach()	· attach () 해제하려면 detach() 사용한다. # 단 attach()한 변수값은 detach()시 원래의 데이터 프레임에는 반영되지 않는다. 즉 별개의 공간이다.

12.유용한 함수들

(4)which(), which.max(), which.min()

함수	상세
which()	· 벡터 또는 배열에서 주어진 조건을 만족하는 값,색인(위치)찾기 > which(iris$Sepal.Length==5.1) # 위치 찾기 [1] 1 18 20 22 24 40 45 47 99 > iris$Sepal.Length[which(iris$Sepal.Length==5.1)] #값 찾기 [1] 5.1 5.1 5.1 5.1 5.1 5.1 5.1 5.1 5.1
which,max() which.min()	· 주어진 벡터에서 최소,최대값이 저장된 위치 찾기 > which.max(iris$Sepal.Length) [1] 132 > which.min(iris$Sepal.Length) [1] 14

12.유용한 함수들

(5)aggregate()

함수	상세
aggregate()	· 그룹별(범주형) 연산을 위한 함수이다. > names(iris)<-tolower(names(iris)) # 변수명 소문자 > attach(iris) # 데이터프레임 바로 접근 # 형태는 aggregate(formula,데이터,함수) > aggregate(sepal.length~species,iris,mean) species sepal.length 1 setosa 5.006 2 versicolor 5.936 3 virginica 6.588

12.유용한 함수들

(6)sort(), order()

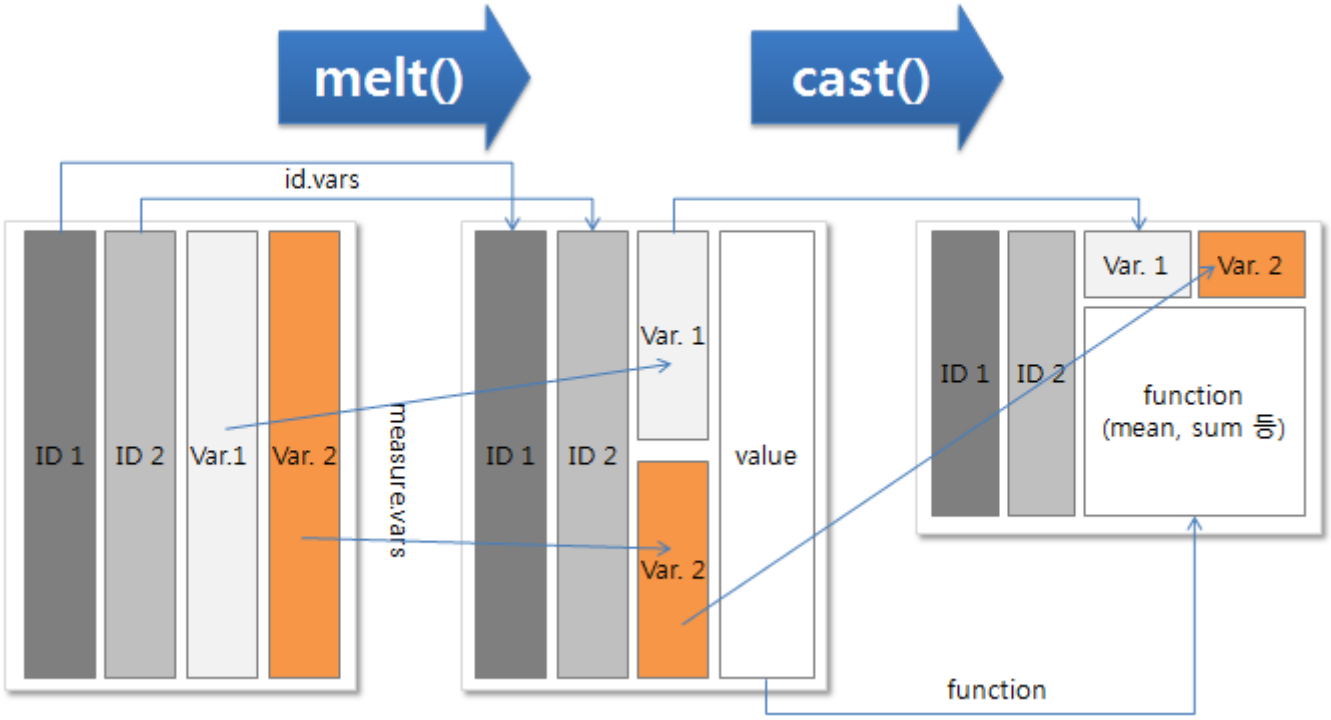
함수	상세
sort()	· 정렬을 위한 함수 sort(iris$sepal.length) [1] 4.3 4.4 4.4 4.4 4.5 4.6 4.6 4.6 4.6 4.7 4.7 4.8 4.8 4.8 4.8 4.8 생략 > iris$sepal.length [1] 5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 4.4 4.9 5.4 4.8 4.8 4.3 5.8 5.7 생략 #sort()는 값을 정렬한 그 결과를 반환하고 원래 벡터자체를 변경하지 않는다.

13.reshape2 패키지

reshape 패키지는 melt(),cast()만을 사용하여 데이터를 재구성하거나, 밀집화된 데이터를 유연하게 생성해준다. 데이터의 모양을 바꾸거나 그룹별 요약 값을 계산하는 함수들을 담고 있는 패키지.

함수	정의
melt() melt(data,id.vars,na.rm=FALSE)	여러 변수로 구성된 데이터를 데이터 id, variable, value 형태로 재구성 한다. # id.vars : 식별컬럼 # variable:측정변수 # value : 측정값 # na.rm=FALSE : NA인 행을 결과에 포함시킬지 여부.
cast() (data,id변수~variable변수, formula)	melt()된 데이터를 다시 여러 칼럼으로 변환한다

13.reshape2 패키지



13.reshape2 패키지


```
library(reshape2)
> data("airquality")
> names(airquality) <- tolower(names(airquality))
> head(airquality)
  ozone solar.r wind temp month day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
```

13.reshape2 패키지


```
> aql <- melt(airquality, id.vars = c("month", "day"))
> head(aql)
  month day variable value
1     5   1    ozone    41
2     5   2    ozone    36
3     5   3    ozone    12
4     5   4    ozone    18
5     5   5    ozone    NA
6     5   6    ozone    28
```

13.reshape2 패키지


```
> dcast(aql, month ~ variable)
 # 데이터를 요약하려면 melt()에서 사용한 것보다 적은 개수의 식별자를 지정한다.

 month ozone solar.r wind temp

1     5    31      31   31   31  # 식별자가 하나일 때 자동으로 length()를 적용해
2     6    30      30   30   30     같은셀에 모인 행의 개수를 센다.
3     7    31      31   31   31
4     8    31      31   31   31
5     9    30      30   30   30
> dcast(aql, month ~ variable, mean,  # sum,mean,range 구할수 있다.
+         na.rm = TRUE)
    month     ozone  solar.r       wind      temp
1     5 23. 61538 181.2963 11.622581 65.54839
2     6 29. 44444 190.1667 10.266667 79.10000
3     7 59. 11538 216.4839  8.941935 83.90323
4     8 59. 96154 171.8571  8.793548 83.96774
5     9 31. 44828 167.4333 10.180000 76.90000
```

13.reshape2 패키지

(1)sqldf를 이용한 데이터 분석

#SQL문을 이용한 데이터 분석

#데이터를 불러올 때 select을 이용

sqldf("select*from iris") # select문은 데이터를 추출 사용

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa

13.reshape2 패키지


```
sqldf("select*from iris limit 6")
# 특정행만 조회할 때 limit,head() 같은 기능
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          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          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
sqldf("select count(*) from iris where Species like 'se%'")
# like(문자열에서 원하는 값을 찾아준다), ' '(작은따옴표)
# species 중 se 시작되는 species 몇 개인가
  count(*)
1       50
```

13.데이터 테이블

r의 기본 데이터구조인 데이터 프레임을 대신하여 사용할 수 있다.

기존 데이터프레임보다 빠른 속도 특정 컬럼을 키값으로 색인을

지정한 후 데이터 처리하기 때문에 빠른 그룹화, 정렬, 짧은 문장

지원 측면에서 데이터프레임보다 유용

```
library(data.table)
data(iris)
iris_table<-as.data.table(iris)
iris_table
```

13.데이터 테이블


```
  Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
1:          5.1         3.5          1.4         0.2    setosa
2:          4.9         3.0          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:          5.0         3.6          1.4         0.2    setosa
# 행 번호가 콜론 형태로  구분
 tables()
# 만들어진 데이터테이블 목록 확인
```

키값을 지정해 해당 값으로 접근

```
# 함수은 setkey(데이터데이블,정렬할 컬럼) 형식
```

13.데이터 테이블

setkey(iris_table,Species) # Species 키값 부여
tables()

NAME NROW NCOL MB COLS KEY
1: iris_table 150 5 0 Sepal.Length,Sepal.Width,Petal.Length,Petal.Width,Species

iris_table["setosa"] 키값을 이용해 빠른 검색이 가능

14.결측값 처리와 이상값 검색

(1)결측값 확인 및 처리★★

1)is,na() # NA값을 조사해 논리값으로 반환 (NA=TRUE)

```
y<-c(1,2,3,NA)
is.na(y)

[1] FALSE FALSE FALSE TRUE

sum(y)   # 평균,분산적용하면   na.rm=TRUE 없으면 NA
[1] NA
> mean(y)
[1] NA

> mean(y,na.rm = TRUE)
[1] 2
```

14.결측값 처리와 이상값 검색

2)complete.cases() # NA값을 조사해 논리값으로 반환(NA=FALSE)

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

3)특정값을 결측 처리

```
iris[iris$sepal.length==4.0]<-NA  # 특정값 4.0을 NA처리하기
```

4) 데이터프레임에서 결측값만 선택 또는 삭제하기★★

```
iris_na<-iris
iris_na[c(10,20,30),3]<-NA
# iris 3번째 변수 Petal.Length의 10,20,30번째 행을 NA(결측처리)

iris_na[!complete.cases(iris_na),] # 결측값이 행만 추출
```

14.결측값 처리와 이상값 검색

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
10	4.9	3.1	NA	0.1	setosa
20	5.1	3.8	NA	0.3	setosa
30	4.7	3.2	NA	0.2	setosa

```
iris_na[complete.cases(iris_na),] # 결측값이 없는 행만 추출
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa

생략

14.결측값 처리와 이상값 검색

5)na.omit() #NA가 있는 행 전체 삭제

```
> a<-na.omit(iris_na) # 10,20,30행 삭제
> a
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species

9            4.4         2.9          1.4         0.2     setosa
11           5.4         3.7          1.5         0.2     setosa
```

14.결측값 처리와 이상값 검색

(2)결측값의 대치법(Imputation of Missing Values)

결측/무응답을 가진 자료를 무시하지 않고 분석할 수 있는 통계 방법론의 하나인 대치법(imputation)

①완전히 응답한 개체분석(Completes Analysis):
불완전 자료는 모두 무시하고 완전하게 관측된 자료만으로 표준적 통계기법에 의해 분석하는 방법

②평균대치법(Mean Imputation): 관측 또는 실험되어 얻어진 자료의 적절한 평균값으로 결측값을 대치

③단순확률 대치법(Single Stochastic Imputation): 평균대치법에서 추정량 표준오차의 과소 추정n문제를 보완하고자 고안된 방법. 변수들이 비슷한 집단에서 임의의 한 개체를 선택해서 대체

14.결측값 처리와 이상값 검색

④다중대치법(Multiple Imputation): 대치(가상의 완전데이터 생성)
->분석(가상자료의 추정량과 분산을 계산)-> 결합 (생성데이터와 추정량결합하여 통계적 추론)

(3)이상값 검색 ★★

①이상값은 의도하지 않게 잘못 입력된 경우(Bad data)
②의도하지 않게 입력됐으나 분석 목적에 부합되지 않아 제거해야 하는 경우(Bad data)
③의도되지 않은 현상이지만 분석에 포함해야 하는 경우(이상값)
④의도된 이상값 (아웃라이어) ----> 사기

관련 알고니즘 ESD(평균으로부터 3*표준편차 밖의 값을 이상치)
상자그림(Boxplot) IQR*1.5 밖의 값을 이상치
일반적으로 summary() 평균과 중앙값과 IQR보고 판단

→결과적으로 이상값과 분석 대상이 될 수 있어 무조건 삭제는 안됨

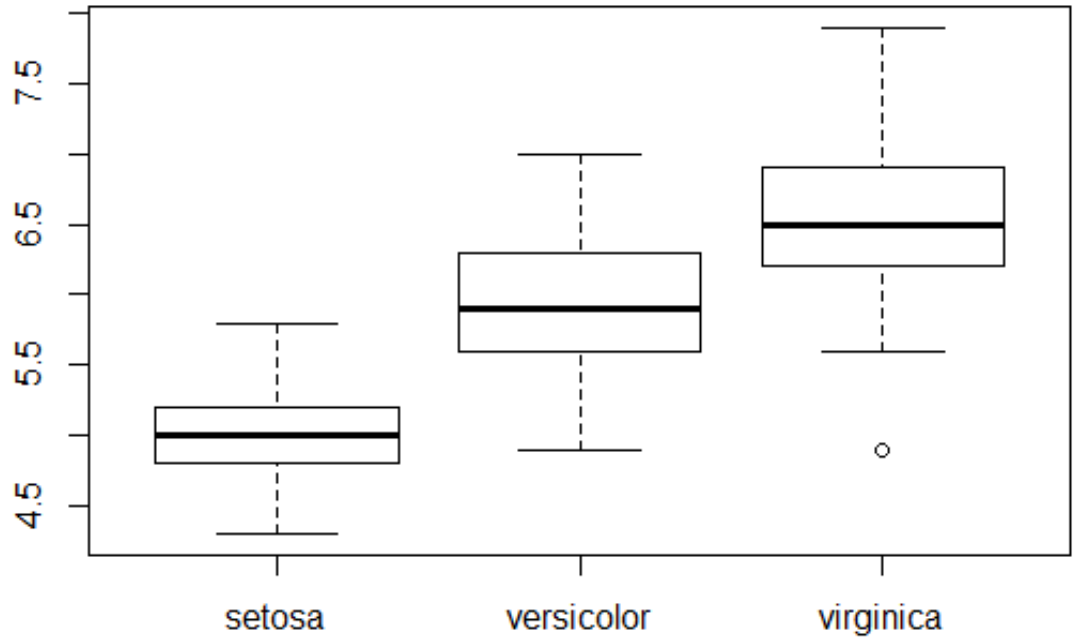
14.결측값 처리와 이상값 검색

outlier 패키지 활용하기

```
install.packages("outliers")
library(outliers)
data(iris)
attach(iris)
iris2<-subset(iris,select = c(1,5)) # Sepal.Length와 Species 추출

iris1<-subset(iris,select="Sepal.Length") # Sepal.Length 추출

outlier(iris1) # 평균과 가장 차이가 많이 나는 값 출력

Sepal.Length
     7.9
outlier(iris1,opposite = TRUE) # 반대방향으로 가장 차이가 많이 나는 값 출력

Sepal.Length
     4.3
```

14.결측값 처리와 이상값 검색


```
data(iris)
# iris 불러오기
plot(Species,Sepal.Length)
# boxplot(Sepal.Length~Species,data=iris)
# plot(Sepal.Length~Species)
# plot 함수 종속변수(Species)~독립변수(Sepal.Length)로 상자그림 그리기
```



14.결측값 처리와 이상값 검색


```
w<-which(iris$Sepal.Length==7.9) # which() 7.9 위치찾기
w
[1] 132  $iris data 132행 위치
data(iris)
iris[132,] # 확인하기
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
132	7.9	3.8	6.4	2	virginica

15. 기출유형

Q01 다음 중 성격이 다른 통계량은?

① mean
② median
③ mode
④ IQR

15. 기출유형

Q02 **scatter plot(산점도)를 통해서 알 수 없는 것은?**

① outlier 존재여부

② 선형관계 여부

③ 상관관계 여부

④ 독립변수와 종속변수간에 인과관계

15. 기출유형

Q03 다음 중 그룹별로 mean,sum 요약이 불가능한 명령어?

① data.table

② sqldf

③ aggregate

④ melt

15. 기출유형

Q04 **R 내장 data trees라는 데이터프레임에 관한 내용이다. 각 변수의 평균을 요약할 때 올바른지 않은 것은?**

```
> head(trees)
  Girth Height Volume
1   8.3     70   10.3
2   8.6     65   10.3
3   8.8     63   10.2
4  10.5     72   16.4
5  10.7     81   18.8
6  10.8     83   19.7
> str(trees)
'data.frame':	31 obs. of  3 variables:
 $ Girth : num  8.3 8.6 8.8 10.5 10.7 10.8 11 11 11.1 11.2 ...
 $ Height: num  70 65 63 72 81 83 66 75 80 75 ...
 $ Volume: num  10.3 10.3 10.2 16.4 18.8 19.7 15.6 18.2 22.6 19.9 ...
```

① apply(trees,1,mean)　② sapply(trees,mean)
③ colMeans(trees)　④ lapply(trees,mean)

15. 기출유형

Q05 다음은 R 명령어에 대한 설명으로 가정 적절한 것은?

```
library(plyr)
ddply(iris,.(Species),function(sub){data.frame(Sepal.width.mean<-mean(sub$Sepal.Width))})
```

① iris라는 데이터프레임에 Species 변수별로 사용자 정의함수을 이용하여 Sepal.width의 평균을 구해 Sepal.width.mean 변수명으로 구성된 데이터프레임을 생성하라
② iris라는 배열(array)에 Species 변수별로 사용자 정의함수을 이용하여 Sepal.width의 평균을 구해 Sepal.width.mean 변수명으로 구성된 데이터프레임을 생성하라
③ iris라는 데이터프레임에 Species 변수별로 사용자 정의함수을 이용하여 Sepal.width의 평균을 구해 Sepal.width.mean 변수명으로 구성된 백터를 생성하라
④ iris라는 데이터프레임에 Species 변수별로 사용자 정의함수을 이용하여 Sepal.width의 평균을 구해 Sepal.width.mean 변수명으로 구성된 배열을 생성하라

15. 기출유형

Q06 iris dataset을 이용하여 Species별 Sepal.Length 차이를 boxplot() 이용하여 스트립트를 작성하시오.

15. 기출유형

Q07 아래 R코드를 실행했을 때 출력되는 결과는?

```
x<-matrix(c(1:9),3,3)
x
min(apply(x,1,mean)) * max(apply(x,2,mean))
```