R에서 데이터 다루기

벡터 다루기

- 벡터에 데이터 추가

```
> x
[1] 1 2 3
> x=c(x,4,5)
> x
[1] 1 2 3 4 5
> y=c(6,7,8)
> x=c(x,y)
> x
[1] 1 2 3 4 5 6 7 8
```

- 반복구조의 벡터
  - 콜론(:)

```
> 1:5
[1] 1 2 3 4 5
> 1.1:2.8
[1] 1.1 2.1
> 1:10
 [1]  1  2  3  4  5  6  7  8  9 10
> 1.5:9.5
[1] 1.5 2.5 3.5 4.5 5.5 6.5 7.5 8.5 9.5
```

  - seq( )

```
> seq(0,5)
[1] 0 1 2 3 4 5
> seq(0,10,2)
[1]  0  2  4  6  8 10
> seq(0,1,length=5)
[1] 0.00 0.25 0.50 0.75 1.00
```

  - rep( )

```
> rep(1,5)
[1] 1 1 1 1 1
> rep(c(1,2),3)
[1] 1 2 1 2 1 2
> rep(c(10,20),c(2,3))
[1] 10 10 20 20 20
```

벡터 연산

- 벡터와 벡터의 연산
  - 길이가 같을 때: 대응되는 구성요소 끼리 연산
  - 길이가 다를 때: 짧은 벡터를 순환 연산
- 벡터와 스칼라의 연산: 모든 요소와 스칼라 연산

```
> x=1:5
> y=seq(2,10,2)
> x
[1] 1 2 3 4 5
> y
[1]  2  4  6  8 10
> x+y
[1]  3  6  9 12 15
> x+10
[1] 11 12 13 14 15
```

```
> x
[1] 1 2 3 4 5
> x*c(10,100)
[1]  10 200  30 400  50
Warning message:
In x * c(10, 100) :
  longer object length is not a multiple of shorter object length
```

〈표 3.1〉 수학관련 함수

함수	설명	사용 예
abs(x)	절댓값 계산	〉 abs(−2) [1] 2
sqrt(x)	제곱근 계산	〉 sqrt(25) [1] 5
ceiling(x)	x보다 작지 않은 가장 작은 정수	〉 ceiling(3.475) [1] 4
floor(x)	x보다 크지 않은 가장 큰 정수	〉 floor(3.475) [1] 3
trunc(x)	x의 소수점 이하 버림	〉 trunc(5.99) [1] 5
round(x, digits=n)	x를 소수 n자리로 반올림	〉 round(3.475, 2) [1] 3.48
signif(x, digits=n)	x를 유효수 n자리로 반올림	〉 signif(0.00347, 2) [1] 0.0035
sin(x), cos(x), tan(x)	삼각함수	〉 sin(1) [1] 0.841471
asin(x), acos(x), atan(x)	역삼각함수	〉 asin(.841471) [1] 1
log(x, base=n)	밑이 n인 x의 로그 값	〉 log(2,base=2) [1] 1
log(x)	x의 자연로그 값	〉 log(10) [1] 2.302585
log10(x)	x의 상용로그 값	〉 log10(10) [1] 1
exp(x)	지수 함수 e^x 값	〉 exp(2.3025855) [1] 10

〈표 3.2〉 통계관련 함수

함수	설명	사용 예
mean(x)	벡터 x의 산술평균값	〉 mean(c(1, 2, 3, 4, 50)) [1] 12
median(x)	벡터 x의 중앙값	〉 median(c(1, 2, 3, 4, 50)) [1] 3
range(x)	벡터 x의 범위(최솟값, 최댓값) 범위의 계산은 diff(range(x))	〉 range(c(1, 2, 3, 4, 50)) [1] 1 50
IQR(x)	벡터 x의 사분위편차	〉 IQR(c(1, 2, 3, 4, 50)) [1] 2
sd(x)	벡터 x의 표준편차	〉 sd(c(1, 2, 3, 4, 50)) [1] 21.27205
var(x)	벡터 x의 분산	〉 var(c(1, 2, 3, 4, 50)) [1] 452.5
sum(x)	벡터 x의 합	〉 sum(c(1, 2, 3, 4, 50)) [1] 60
diff(x, lag=n)	벡터 x의 차분, $x_{n+i} - x_i$ 시차 n의 디폴트 값은 1	〉 diff(c(1, 2, 4, 7, 11)) [1] 1 2 3 4
min(x)	벡터 x의 최솟값	〉 min(c(1, 2, 3, 4, 50)) [1] 1
max(x)	벡터 x의 최댓값	〉 max(c(1, 2, 3, 4, 50)) [1] 50

문자열의 결합

- paste(결합할 문자열들, sep=구분기호)
- 변수나 벡터의 요소와 결합 가능

```
> paste("Big","Data",sep="")
[1] "BigData"
> paste("Big","Data",sep=" ")
[1] "Big Data"
> i=2014
> paste("Big","Data",i,sep=" ")
[1] "Big Data 2014"
> paste("Big","Data",2014:2020,sep=" ")
[1] "Big Data 2014" "Big Data 2015" "Big Data 2016" "Big Data 2017"
[5] "Big Data 2018" "Big Data 2019" "Big Data 2020"
```

문자열 분리

- strsplit(문자열, split=구분기호)
- 벡터의 각 요소 분리

```
> cities=c("Kookmin University, Korea","Penn State University, USA", "Tokyo University, Japan")
> cities
[1] "Kookmin University, Korea"  "Penn State University, USA" "Tokyo University, Japan"
> cities=strsplit(cities,split=",")
> cities
[[1]]
[1] "Kookmin University" " Korea"

[[2]]
[1] "Penn State University" " USA"

[[3]]
[1] "Tokyo University" " Japan"

> univ=c(cities[[1]][1],cities[[2]][1],cities[[3]][1])
> univ
[1] "Kookmin University"    "Penn State University" "Tokyo University"
```

문자열 치환

- “문자열”의 “old”를 “new”로 치환
- gsub(old,new,문자열): 모든 “old” 치환
- sub(old,new,문자열): 첫 “old”만 치환

```
> x="Kookmin University is close to Korea University."
> gsub("University","Univ",x)
[1] "Kookmin Univ is close to Korea Univ."
> sub("University","Univ",x)
[1] "Kookmin Univ is close to Korea University."
```

문자 함수

〈표 3.3〉 문자 함수

함수	기능
nchar(x)	문자열 x를 구성하는 문자의 개수
paste(. . . , sep=" ")	문자열들의 결합
substr(x, start, stop)	문자열의 일부분 선택
toupper(x)	영문자 대문자로 변환
tolower(x)	영문자 소문자로 변환
strsplit(x, split)	문자열의 분리
sub(old, new, x)	문자열의 치환
gsub(old, new, x)	문자열의 치환

벡터의 비교

- 벡터의 연산과 유사
  - 벡터와 벡터의 비교: 각 요소끼리 비교
  - 벡터와 스칼라의 비교: 모든 요소와 스칼라의 비교
  - 결과는 논리형 벡터 TRUE/FALSE (산술연산 가능, TRUE=1, FALSE=0)

```
> x=c(3,8,2)
> y=c(5,4,2)
> x>y
[1] FALSE  TRUE FALSE
> x==y
[1] FALSE FALSE  TRUE
> x!=y
[1]  TRUE  TRUE FALSE
> x>2
[1]  TRUE  TRUE FALSE
> x<=2 | x>4
[1] FALSE  TRUE  TRUE
> any(x==5)
[1] FALSE
> all(x==3)
[1] FALSE
```

```
> x=1:100
> sum(x>50)
[1] 50
> mean(x>=30 & x<=60)
[1] 0.31
```

벡터의 비교

〈표 3.4〉 비교/논리 연산자

연산자	기능
〈	작다
〈=	작거나 같다
〉	크다
〉=	크거나 같다
==	같다
!=	같지 않다
! x	x가 아니다 (NOT)
x | y	x 또는 y (OR)
x & y	x 그리고 y (AND)

결측값

- NA (not available)로 표시
- is.na( ): 각 요소가 결측치인지 TRUE/FALSE 표시
- NA가 포함된 경우 많은 함수들이 NA로 결과출력
  - na.rm=TRUE 옵션을 통해 결측치를 제외한 결과 출력 가능

```
> x=c(1,2,3,4,NA)
> x
[1]  1  2  3  4 NA
> is.na(x)
[1] FALSE FALSE FALSE FALSE  TRUE
> mean(x)
[1] NA
> mean(x,na.rm=TRUE)
[1] 2.5
```

객체 유형 전환

〈표 3.5〉 데이터 객체의 유형 전환 함수

유형 확인	유형 전환
is.numeric( )	as.numeric( )
is.character( )	as.character( )
is.vector( )	as.vector( )
is.factor( )	as.factor( )
is.matrix( )	as.matrix( )
is.data.frame( )	as.data.frame( )

```
> x=1:10
> x
 [1]  1  2  3  4  5  6  7  8  9 10
> is.numeric(x)
[1] TRUE
> y=as.character(x)
> y
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10"
> is.numeric(y)
[1] FALSE
```

행렬 다루기

- cbind( ): 벡터를 열단위로 묶기
- rbind( ): 벡터를 행단위로 묶기
- colnames( ): 행렬의 열이름 지정 혹은 추출
- rownames( ): 행렬의 행이름 지정 혹은 추출

```
> x=1:3
> y=5:7
> cbind(x,y)
     x y
[1,] 1 5
[2,] 2 6
[3,] 3 7
> rbind(x,y)
  [,1] [,2] [,3]
x    1    2    3
y    5    6    7
> x=1:3
> y=5:7
> A=cbind(x,y)
> A
     x y
[1,] 1 5
[2,] 2 6
[3,] 3 7
```

```
> B=rbind(x,y)
> B
  [,1] [,2] [,3]
x    1    2    3
y    5    6    7
> rownames(A)=c("r1","r2","r3")
> A
   x y
r1 1 5
r2 2 6
r3 3 7
> colnames(B)=c("c1","c2","c3")
> B
  c1 c2 c3
x  1  2  3
y  5  6  7
```

행렬의 연산

- 기본연산자(+-*/^): 각 요소에 적용
- %*%: 행렬의 곱

```
> A=matrix(1:4,2,2)
> B=matrix(5:8,2,2)
> A
     [,1] [,2]
[1,]    1    3
[2,]    2    4
> B
     [,1] [,2]
[1,]    5    7
[2,]    6    8
> A*B
     [,1] [,2]
[1,]    5   21
[2,]   12   32
> A%*%B
     [,1] [,2]
[1,]   23   31
[2,]   34   46
```

- rowMeans( ), colMeans( )
  - 각 행 또는 열 방향으로 평균

```
> rowMeans(A)
[1] 2 3
> colMeans(A)
[1] 1.5 3.5
> A
     [,1] [,2]
[1,]    1    3
[2,]    2    4
```

〈표 3.7〉 행렬의 연산자에 유용하게 사용되는 함수 및 연산자

연산자 및 함수	기능
+ − * / ^	행렬을 구성하는 숫자 각각에 적용
A %*% B	행렬 A와 B의 곱하기
cbind(A, B, . . .)	행렬이나 벡터를 열 단위로 결합
colMeans(A)	행렬 A 각 열의 평균값으로 구성된 벡터
crossprod(A)	t(A) %*% A (A^TA)
crossprod(A, B)	t(A) %*% B (A^TB)
colSums(A)	행렬 A 각 열의 합으로 구성된 벡터
diag(A)	행렬 A의 주대각선 원소로 구성된 벡터
diag(x)	벡터 x를 주대각선 원소로 하는 대각행렬
diag(k)	k * k 단위행렬
eigen(A)	행렬 A의 고유값과 고유벡터로 구성된 리스트
rbind(A, B, . . .)	행렬이나 벡터를 행 단위로 결합
rowMeans(A)	행렬 A 각 행의 평균값으로 구성된 벡터
rowSums(A)	행렬 A 각 행의 합으로 구성된 벡터
solve(A)	행렬 A의 역행렬
solve(A, b)	연립방정식 Ax=b의 해
t(A)	행렬 A의 전치 (A^T)
tcrossprod(A) tcrossprod(A, B)	A %*% t(A) A %*% t(B)

데이터프레임 변수추가

- dataframe$변수명 활용

```
> data=read.csv("anorexia.csv")
> head(data)
  Treat Prewt Postwt
1  Cont  80.7   80.2
2  Cont  89.4   80.1
3  Cont  91.8   86.4
4  Cont  74.0   86.3
5  Cont  78.1   76.1
6  Cont  88.3   78.1
> data$diff=data$Postwt-data$Prewt
> head(data)
  Treat Prewt Postwt  diff
1  Cont  80.7   80.2  -0.5
2  Cont  89.4   80.1  -9.3
3  Cont  91.8   86.4  -5.4
4  Cont  74.0   86.3  12.3
5  Cont  78.1   76.1  -2.0
6  Cont  88.3   78.1 -10.2
```

- transform( ) 함수사용

```
> data=transform(data,diff=Postwt-Prewt, mean=(Postwt+Prewt)/2)
> head(data)
  Treat Prewt Postwt  diff  mean
1  Cont  80.7   80.2  -0.5 80.45
2  Cont  89.4   80.1  -9.3 84.75
3  Cont  91.8   86.4  -5.4 89.10
4  Cont  74.0   86.3  12.3 80.15
5  Cont  78.1   76.1  -2.0 77.10
6  Cont  88.3   78.1 -10.2 83.20
```

데이터프레임 변수추가

- 조건에 따라 변수추가
  - 논리연산자 TRUE/FASLE 값을 인덱스로 활용
  - TRUE인 행에만 변환적용

```
> data$group[data$diff>0]="Increase"
> data$group[data$diff<0]="Decrease"
> data$group[data$diff==0]="No Change"
> head(data,15)
   Treat Prewt Postwt  diff  mean    group
1   Cont  80.7   80.2  -0.5 80.45 Decrease
2   Cont  89.4   80.1  -9.3 84.75 Decrease
3   Cont  91.8   86.4  -5.4 89.10 Decrease
4   Cont  74.0   86.3  12.3 80.15 Increase
5   Cont  78.1   76.1  -2.0 77.10 Decrease
6   Cont  88.3   78.1 -10.2 83.20 Decrease
7   Cont  87.3   75.1 -12.2 81.20 Decrease
8   Cont  75.1   86.7  11.6 80.90 Increase
9   Cont  80.6   73.5  -7.1 77.05 Decrease
10  Cont  78.4   84.6   6.2 81.50 Increase
11  Cont  77.6   77.4  -0.2 77.50 Decrease
12  Cont  88.7   79.5  -9.2 84.10 Decrease
13  Cont  81.3   89.6   8.3 85.45 Increase
14  Cont  78.1   81.4   3.3 79.75 Increase
15  Cont  70.5   81.8  11.3 76.15 Increase
```

데이터프레임 정렬

- sort(): 정렬된 데이터 출력
- order(): 데이터를 정렬시키는 index 출력
- decreasing=TRUE 옵션으로 내림차순 정렬

```
> sort(data$Prewt)
 [1] 70.0 70.5 72.3 73.4 74.0 75.1 76.3 76.5 76.5 76.9 77.3 77.5 77.6 77.6 78.1 78.1 78.4 79.6 79.6 79.
7 79.7 79.9 80.2 80.4
[25] 80.5 80.5 80.5 80.6 80.7 80.8 81.0 81.3 81.3 81.5 81.6 82.1 82.5 82.6 83.0 83.3 83.3 83.3 83.5 83.
8 84.1 84.2 84.4 84.5
[49] 84.9 85.0 85.2 85.5 86.0 86.0 86.0 86.4 86.7 87.3 87.3 87.4 87.7 87.8 88.3 88.7 88.7 89.0 89.2 89.
4 89.9 91.8 94.2 94.9
> order(data$Prewt)
 [1] 41 15 25 64  4  8 34 40 48 62 16 24 11 68  5 14 10 23 61 20 52 31 49 42 27 36 65  9  1 54 35 13 3
 29 66 67 59 30 44 43
[41] 51 57 69 56 19 46 22 53 28 37 17 21 18 58 71 47 60  7 72 55 45 50  6 12 32 26 38  2 70  3 63 33
> data[order(data$Prewt),]
   Treat Prewt Postwt
41   CBT  70.0   90.9
15  Cont  70.5   81.8
25  Cont  72.3   88.2
64    FT  73.4   94.9
4   Cont  74.0   86.3
8   Cont  75.1   86.7
```

데이터의 취사선택

- 특정 조건에 맞는 데이터의 부분만을 선택
  - 기본적인 벡터의 인덱싱 이용
  - subset( ) 함수 사용
- na.omit( ): 결측치가 있는 행 제거

✓ Treat 변수가 “Cont”인 관찰치만 선택

```
> data[data$Treat=="Cont",]
   Treat Prewt Postwt
1   Cont  80.7   80.2
2   Cont  89.4   80.1
3   Cont  91.8   86.4
4   Cont  74.0   86.3
5   Cont  78.1   76.1
6   Cont  88.3   78.1
7   Cont  87.3   75.1
> subset(data,subset=(Treat=="Cont"))
   Treat Prewt Postwt
1   Cont  80.7   80.2
2   Cont  89.4   80.1
3   Cont  91.8   86.4
4   Cont  74.0   86.3
5   Cont  78.1   76.1
```

✓ Treat 변수가 “Cont”인 관찰치와 Postwt, Prewt변수만 남김

```
> data[data$Treat=="Cont",c("Postwt","Prewt")]
   Postwt Prewt
1    80.2  80.7
2    80.1  89.4
3    86.4  91.8
4    86.3  74.0
5    76.1  78.1
6    78.1  88.3
7    75.1  87.3
> subset(data,select=c(Prewt,Postwt),subset=(Treat=="Cont"))
   Prewt Postwt
1   80.7   80.2
2   89.4   80.1
3   91.8   86.4
4   74.0   86.3
5   78.1   76.1
```

데이터 프레임 결합

- 단순한 수평 수직 결합은 cbind(), rbind() 사용
- merge( ): 공통변수를 기준으로 수평결합

```
> authors
   surname nationality deceased
1    Tukey          US      yes
2 Venables   Australia       no
3  Tierney          US       no
4   Ripley          UK       no
5   McNeil   Australia       no
> books
      name                          title     other.author
1    Tukey      Exploratory Data Analysis             <NA>
2 Venables Modern Applied Statistics ...           Ripley
3  Tierney                      LISP-STAT             <NA>
4   Ripley             Spatial Statistics             <NA>
5   Ripley          Stochastic Simulation             <NA>
6   McNeil      Interactive Data Analysis             <NA>
7   R Core           An Introduction to R Venables & Smith
> merge(authors, books, by.x = "surname", by.y = "name")
   surname nationality deceased                          title other.author
1   McNeil   Australia       no      Interactive Data Analysis         <NA>
2   Ripley          UK       no             Spatial Statistics         <NA>
3   Ripley          UK       no          Stochastic Simulation         <NA>
4  Tierney          US       no                       LISP-STAT         <NA>
5    Tukey          US      yes      Exploratory Data Analysis         <NA>
6 Venables   Australia       no Modern Applied Statistics ...       Ripley
```

데이터 집계

- 특정 변수를 기준으로 형성된 그룹에 함수를 적용하여 집계
- aggregate(object, by=그룹변수,FUN=함수)
  - 그룹변수는 list 형태로 입력

```
> aggregate(data[,2:3],by=list(Treat=data$Treat),mean)
  Treat    Prewt   Postwt
1   CBT 82.68966 85.69655
2  Cont 81.55769 81.10769
3    FT 83.22941 90.49412
> aggregate(data[,2:3],by=list(Treat=data$Treat,Pre.over80=data$Prewt>80),mean)
  Treat Pre.over80    Prewt   Postwt
1   CBT      FALSE 76.48333 82.08333
2  Cont      FALSE 76.51667 81.28333
3    FT      FALSE 76.87500 84.77500
4   CBT       TRUE 84.30870 86.63913
5  Cont       TRUE 85.87857 80.95714
6    FT       TRUE 85.18462 92.25385
```

데이터의 구조변경

- reshape package
  - melt(object, id.var)
    - id.var을 기준으로 데이터를 아래로 펼침

```
> x=data.frame(age=c(22,28,34,24),gender=c("F","M","M","F"), income=c(2000,3100,3800,2800), region=c("S","S","G","G"))
> x
  age gender income region
1  22      F   2000      S
2  28      M   3100      S
3  34      M   3800      G
4  24      F   2800      G
> x.melt=melt(x,id.var=c("gender","region"))
> x.melt
  gender region variable value
1      F      S      age    22
2      M      S      age    28
3      M      G      age    34
4      F      G      age    24
5      F      S   income  2000
6      M      S   income  3100
7      M      G   income  3800
8      F      G   income  2800
```

데이터의 구조변경

- cast(mmelted object, formula, function)
  - melt( )를 통해 펼쳐진 데이터를 대상으로 집계
  - formular=var1~var2: var1의 level을 행으로 var2의 level을 열 방향으로 설정해 value의 값을 function으로 집계

```
> x.melt
  gender region variable value
1      F      S      age    22
2      M      S      age    28
3      M      G      age    34
4      F      G      age    24
5      F      S   income  2000
6      M      S   income  3100
7      M      G   income  3800
8      F      G   income  2800
> cast(x.melt,gender~variable,mean)
  gender age income
1      F  23   2400
2      M  31   3450
> cast(x.melt,gender~region,mean)
  gender    G    S
1      F 1412 1011
2      M 1917 1564
```

데이터프레임에 함수적용: apply류

- apply(행렬,1 or 2, 함수): 데이터의 행방향(1) 혹은 열방향(2)로 함수 적용

```
> x=matrix(c(1:12),3,4)
> x
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
> apply(x,1,mean)
[1] 5.5 6.5 7.5
> apply(x,2,mean)
[1]  2  5  8 11
```

데이터프레임에 함수적용: apply류

- lapply(list, 함수)
  - 리스트 (혹은 data frame)의 각 요소에 함수적용. list로 결과출력
- sapply(list, 함수)
  - 리스트 (혹은 data frame)의 각 요소에 함수적용. vector로 결과출력

```
> air=na.omit(airquality)
> head(air)
  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
7    23     299  8.6   65     5   7
8    19      99 13.8   59     5   8
> lapply(air,mean)
$Ozone
[1] 42.0991

$Solar.R
[1] 184.8018

$Wind
[1] 9.93964

$Temp
[1] 77.79279

$Month
[1] 7.216216

$Day
[1] 15.94595

> sapply(air,mean)
     Ozone    Solar.R       Wind       Temp      Month        Day
 42.099099 184.801802   9.939640  77.792793   7.216216  15.945946
```