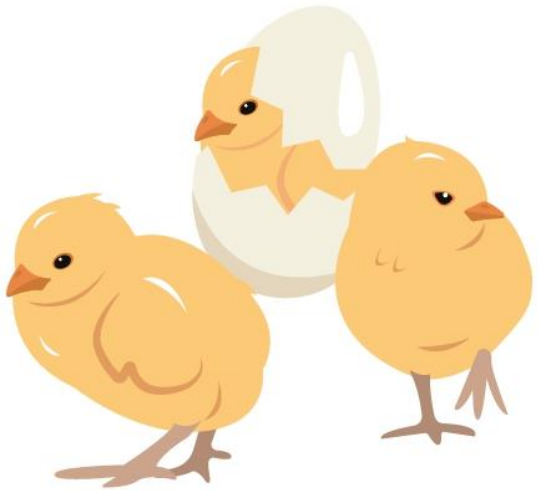
모두를 위한

R 데이터 분석 입문

2판



Chapter 07
데이터 전처리

목차

1. 결측값
2. 특이값
3. 데이터 정렬
4. 데이터 분리와 선택
5. 데이터 샘플링과 조합
6. 데이터 집계와 병합

Section 01

결측값

1. 결측값의 개념

- 결측값(missing value)은 데이터를 수집하고 저장하는 과정에서 저장할 값을 얻지 못하는 경우 발생
- 통계조사 응답자가 어떤 문항에 대해 응답을 안했다고 하면, 그 문항의 데이터값은 결측값이 됨
- 데이터셋에 결측값이 섞여 있으면, 데이터 분석 시 여러 가지 문제를 야기
- 성적자료에 결측값이 포함되어 있다면, 성적자료에 대한 합계 계산이나 평균 계산 등의 작업이 불가능
- 결측값의 처리 1: 결측값을 제거하거나 제외하고, 데이터를 분석
- 결측값의 처리 2: 결측값을 추정하여 적당한 값으로 치환한 후, 데이터를 분석

2. 벡터의 결측값 처리

2.1 결측값의 특성과 존재 여부 확인

코드 7-1

```
z <- c(1,2,3,NA,5,NA,8)          # 결측값이 포함된 벡터 z
sum(z)                           # 정상 계산이 안 됨
is.na(z)                         # NA 여부 확인
sum(is.na(z))                    # NA의 개수 확인
sum(z, na.rm=TRUE)               # NA를 제외하고 합계를 계산
```

```
> z <- c(1,2,3,NA,5,NA,8)        # 결측값이 포함된 벡터 z
> sum(z)                         # 정상 계산이 안 됨
[1] NA
> is.na(z)                       # NA 여부 확인
[1] FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE
> sum(is.na(z))                  # NA의 개수 확인
[1] 2
> sum(z, na.rm=TRUE)             # NA를 제외하고 합계를 계산
[1] 19
```

2.2 결측값 대체 및 제거

코드 7-2

```
z1 <- c(1,2,3,NA,5,NA,8)                    # 결측값이 포함된 벡터 z1
z2 <- c(5,8,1,NA,3,NA,7)                    # 결측값이 포함된 벡터 z2
z1[is.na(z1)] <- 0                          # NA를 0으로 치환
z1
z3 <- as.vector(na.omit(z2))                # NA를 제거하고 새로운 벡터 생성
z3
```

```
> z1 <- c(1,2,3,NA,5,NA,8)        # 결측값이 포함된 벡터 z1
> z2 <- c(5,8,1,NA,3,NA,7)        # 결측값이 포함된 벡터 z2
> z1[is.na(z1)] <- 0              # NA를 0으로 치환
> z1
[1] 1 2 3 0 5 0 8
> z3 <- as.vector(na.omit(z2))    # NA를 제거하고 새로운 벡터 생성
> z3
[1] 5 8 1 3 7
```

3. 매트릭스와 데이터프레임의 결측값 처리

3.1 결측값이 포함된 데이터프레임 생성

코드 7-3

```
# NA를 포함하는 test 데이터 생성
x <- iris
x[1,2]<- NA; x[1,3]<- NA
x[2,3]<- NA; x[3,4]<- NA
head(x)
```

```
> head(x)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1          NA           NA         0.2  setosa
2          4.9         3.0           NA         0.2  setosa
3          4.7         3.2          1.3          NA  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
```

3.2 데이터프레임의 열별 결측값 확인

코드 7-4

```
# for문을 이용한 방법
for (i in 1:ncol(x)) {
    this.na <- is.na(x[,i])
    cat(colnames(x)[i], "₩t", sum(this.na), "₩n")
}

# apply를 이용한 방법
col_na <- function(y) {
    return(sum(is.na(y)))
}

na_count <-apply(x, 2, FUN=col_na)
na_count
```

1. 결측값

```
> # for문을 이용한 방법
> for (i in 1:ncol(x)) {
+   this.na <- is.na(x[,i])
+   cat(colnames(x)[i], "\t", sum(this.na), "\n")
+ }
Sepal.Length  0
Sepal.Width   1
Petal.Length  2
Petal.Width   1
Species       0
>
> # apply를 이용한 방법
> col_na <- function(y) {
+   return(sum(is.na(y)))
+ }
>
> na_count <-apply(x, 2, FUN=col_na)
> na_count
Sepal.Length  Sepal.Width Petal.Length  Petal.Width      Species
           0           1            2            1            0
```

3.3 데이터프레임의 행별 결측값 확인

코드 7-5

```
rowSums(is.na(x))                    # 행별 NA의 개수
sum(rowSums(is.na(x))>0)             # NA가 포함된 행의 개수

sum(is.na(x))                        # 데이터셋 전체에서 NA 개수
```

```
> rowSums(is.na(x))                  # 행별 NA 개수
  [1] 2 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [43] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [85] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[127] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
> sum(rowSums(is.na(x))>0)           # NA 가 포함된 행의 개수
[1] 3
>
> sum(is.na(x))                      # 데이터셋 전체에서 NA 개수
[1] 4
```

3.4 결측값을 제외하고 새로운 데이터셋 만들기

코드 7-6

```
head(x)
x[!complete.cases(x),]                    # NA가 포함된 행들 출력
y <- x[complete.cases(x),]                # NA가 포함된 행들 제거
head(y)                                   # 새로운 데이터셋 y의 내용 확인
```

```
> head(x)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1          NA           NA         0.2  setosa
2          4.9         3.0           NA         0.2  setosa
3          4.7         3.2          1.3          NA  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
```

1. 결측값

```
> x[!complete.cases(x),]              # NA가 포함된 행들 출력
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1          NA           NA         0.2  setosa
2          4.9         3.0           NA         0.2  setosa
3          4.7         3.2          1.3          NA  setosa
> y <- x[complete.cases(x),]          # NA가 포함된 행들 제거
> head(y)                             # 새로운 데이터셋 y의 내용 확인
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
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
7          4.6         3.4          1.4         0.3  setosa
8          5.0         3.4          1.5         0.2  setosa
9          4.4         2.9          1.4         0.2  setosa
```

Section 02

특이값

1. 특이값의 개념

- **특이값(outlier) :** 정상적이라고 생각되는 데이터의 분포 범위 밖에 위치하는 값들을 말하며, '이상치'라고도 부름
- 특이값은 입력 오류에 의해 발생하기도 하고, 일반인의 몸무게 자료에 씨름선수의 몸무게가 합쳐진 경우처럼 실제로 특이한 값일 수도 있음
- 제조 공정에서 불량인 제품을 선별하거나 은행거래 시스템에서 사기거래를 탐지할 때 사용하기도 함
- 데이터 분석에서는 특이값을 포함한 채 평균 등을 계산하면 전체 데이터의 양상을 파악하는 데 왜곡을 가져올 수 있으므로 분석할 때 특이값을 제외하는 경우가 많음

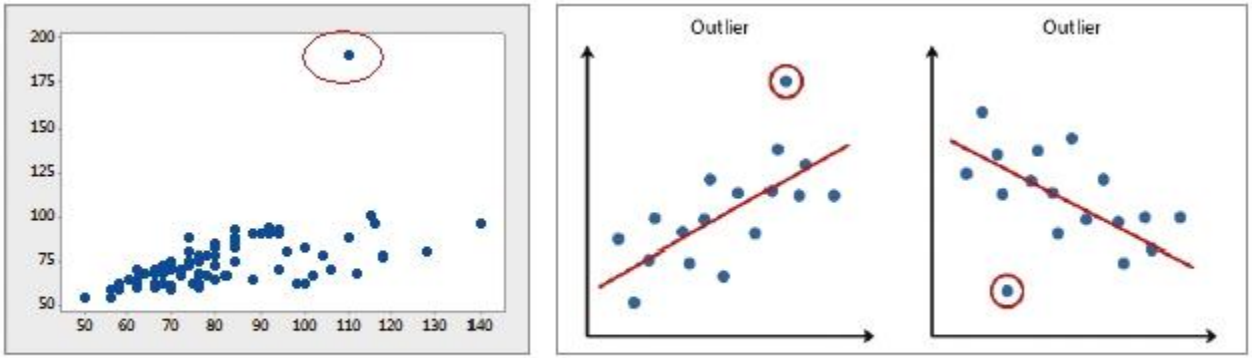


그림 7-1 특이값의 사례 © Minitab와 Richanchor blog

2. 특이값

- 특이값이 포함되어 있는지 여부 확인

  ① 논리적으로 있을 수 없는 값이 있는지 찾아봄

  ex) 좋아하는 색깔을 1~5로 표시하기로 했는데 7이 존재함

  ex) 몸무게에 마이너스 값이 있음

  ② 상식을 벗어난 값이 있는지 찾아봄

  ex) 나이가 120살 이상인 사람

  ③ 상자그림(boxplot)을 통해 찾아봄

  ex) 정상 범위 밖에 동그라미 표시가 있으면 특이값을 의미

데이터셋 state.x77

- 미국 50개 주(state)에 대한 1970년 인구 조사 및 기타 통계 정보
  - 이 데이터셋은 다변량 데이터 분석 연습에 유용하며, 각 주에 대한 여러 통계 지표를 제공
  - 데이터셋 구조이 데이터셋은 8개의 변수와 50개의 행으로 구성

① Population: 1975년의 인구 (천 명 단위)

② Income: 1974년의 1인당 소득 (달러).

③ Illiteracy: 1970년의 문맹률 (백분율).

④ Life Exp: 1969-1971년의 평균 기대수명 (년)

⑤ Murder: 1976년의 인구 10만 명당 살인율

⑥ HS Grad: 1970년의 고등학교 졸업률 (백분율)

⑦ Frost: 연간 서리 없는 날의 평균 일수

⑧ Area: 주의 총 면적 (천 평방마일)

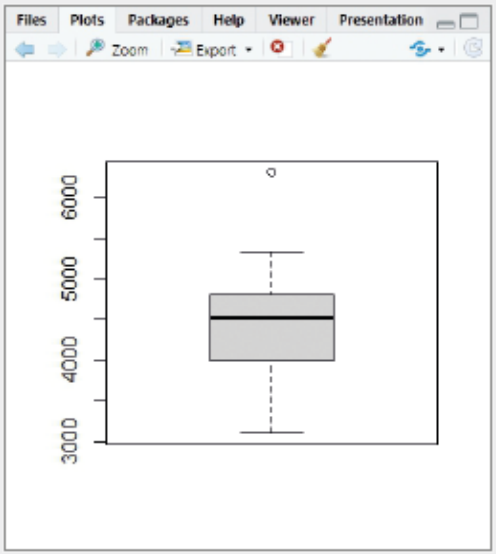
2. 특이값 추출 및 제거

2.1 상자그림을 통한 특이값 확인

코드 7-7

```
st <- data.frame(state.x77)
boxplot(st$Income)
boxplot.stats(st$Income)$out
```

```
> st <- data.frame(state.x77)
> boxplot(st$Income)
```




```
> boxplot.stats(st$Income)$out
[1] 6315
```

2.2 특이값을 포함한 행 제거

코드 7-8

```
out.val <- boxplot.stats(st$Income)$out              # 특이값 추출
st$Income[st$Income %in% out.val] <- NA              # 특이값을 NA로 대체
head(st)
newdata <- st[complete.cases(st),]                   # NA가 포함된 행 제거
head(newdata)
```

```
> out.val <- boxplot.stats(st$Income)$out      # 특이값 추출

> st$Income[st$Income %in% out.val] <- NA      # 특이값을 NA로 대체
> head(st)
           Population Income Illiteracy Life.Exp Murder HS.Grad Frost   Area
Alabama          3615   3624        2.1    69.05   15.1    41.3    20  50708
Alaska            365     NA        1.5    69.31   11.3    66.7   152 566432
Arizona          2212   4530        1.8    70.55    7.8    58.1    15 113417
Arkansas         2110   3378        1.9    70.66   10.1    39.9    65  51945
California      21198   5114        1.1    71.71   10.3    62.6    20 156361
Colorado         2541   4884        0.7    72.06    6.8    63.9   166 103766
```

2. 특이값

```
> newdata <- st[complete.cases(st),]          # NA가 포함된 행 제거
> head(newdata)
           Population Income Illiteracy Life.Exp Murder HS.Grad Frost   Area
Alabama          3615   3624        2.1    69.05   15.1    41.3    20  50708
Arizona          2212   4530        1.8    70.55    7.8    58.1    15 113417
Arkansas         2110   3378        1.9    70.66   10.1    39.9    65  51945
California      21198   5114        1.1    71.71   10.3    62.6    20 156361
Colorado         2541   4884        0.7    72.06    6.8    63.9   166 103766
Connecticut      3100   5348        1.1    72.48    3.1    56.0   139   4862
```

Section 03

데이터 정렬

1. 벡터의 정렬

- 정렬(sort)은 데이터를 주어진 기준에 따라 크기순으로 재배열하는 과정

코드 7-9

```
v1 <- c(1,7,6,8,4,2,3)
order(v1)
v1 <- sort(v1)                    # 오름차순
v1
v2 <- sort(v1, decreasing=T)      # 내림차순
v2
```

```
> v1 <- c(1,7,6,8,4,2,3)
> order(v1)
[1] 1 6 7 5 3 2 4
> v1 <- sort(v1)                  # 오름차순
> v1
[1] 1 2 3 4 6 7 8
> v2 <- sort(v1, decreasing=T)    # 내림차순
> v2
[1] 8 7 6 4 3 2 1
```

2. 매트릭스와 데이터프레임의 정렬

코드 7-10

```
head(iris)
order(iris$Sepal.Length)
iris[order(iris$Sepal.Length),]                          # 오름차순으로 정렬
iris[order(iris$Sepal.Length, decreasing=T),]            # 내림차순으로 정렬
iris.new <- iris[order(iris$Sepal.Length),]              # 정렬된 데이터를 저장
head(iris.new)
iris[order(iris$Species, -iris$Petal.Length, decreasing=T),] # 정렬 기준이 2개
```

```
> head(iris)
  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
```

3. 데이터 정렬

```
> order(iris$Sepal.Length)
  [1] 14  9 39 43 42  4  7 23 48  3 30 12 13 25 31 46  2 10 35 38 58
 [22] 107  5  8 26 27 36 41 44 50 61 94  1 18 20 22 24 40 45 47 99 28
 [43] 29 33 60 49  6 11 17 21 32 85 34 37 54 81 82 90 91 65 67 70 89
 [64] 95 122 16 19 56 80 96 97 100 114 15 68 83 93 102 115 143 62 71 150 63
 [85] 79 84 86 120 139 64 72 74 92 128 135 69 98 127 149 57 73 88 101 104 124
[106] 134 137 147 52 75 112 116 129 133 138 55 105 111 117 148 59 76 66 78 87 109
[127] 125 141 145 146 77 113 144 53 121 140 142 51 103 110 126 130 108 131 106 118 119
[148] 123 136132

> iris[order(iris$Sepal.Length),]                    # 오름차순으로 정렬
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
14          4.3         3.0          1.1         0.1     setosa
9           4.4         2.9          1.4         0.2     setosa
39          4.4         3.0          1.3         0.2     setosa
43          4.4         3.2          1.3         0.2     setosa
42          4.5         2.3          1.3         0.3     setosa
4           4.6         3.1          1.5         0.2     setosa
...(중간 생략)
```

3. 데이터 정렬

```
123          7.7          2.8          6.7          2.0  virginica
136          7.7          3.0          6.1          2.3  virginica
132          7.9          3.8          6.4          2.0  virginica
> iris[order(iris$Sepal.Length, decreasing=T),]    # 내림차순으로 정렬
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
132          7.9          3.8          6.4          2.0  virginica
118          7.7          3.8          6.7          2.2  virginica
119          7.7          2.6          6.9          2.3  virginica
123          7.7          2.8          6.7          2.0  virginica
136          7.7          3.0          6.1          2.3  virginica
106          7.6          3.0          6.6          2.1  virginica
131          7.4          2.8          6.1          1.9  virginica
...(중간 생략)
39           4.4          3.0          1.3          0.2     setosa
43           4.4          3.2          1.3          0.2     setosa
14           4.3          3.0          1.1          0.1     setosa
```

3. 데이터 정렬

```
> iris.new <- iris[order(iris$Sepal.Length),]        # 정렬된 데이터를 저장
> head(iris.new)
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
14          4.3         3.0          1.1         0.1  setosa
9           4.4         2.9          1.4         0.2  setosa
39          4.4         3.0          1.3         0.2  setosa
43          4.4         3.2          1.3         0.2  setosa
42          4.5         2.3          1.3         0.3  setosa
4           4.6         3.1          1.5         0.2  setosa
```

```
> iris[order(iris$Species, -iris$Petal.Length, decreasing=T),] # 정렬 기준이 2개
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
107          4.9         2.5          4.5         1.7  virginica
127          6.2         2.8          4.8         1.8  virginica
139          6.0         3.0          4.8         1.8  virginica
122          5.6         2.8          4.9         2.0  virginica
124          6.3         2.7          4.9         1.8  virginica
128          6.1         3.0          4.9         1.8  virginica
...(중간 생략)

21           5.4         3.4          1.7         0.2     setosa
24           5.1         3.3          1.7         0.5     setosa
25           4.8         3.4          1.9         0.2     setosa
45           5.1         3.8          1.9         0.4     setosa
```

Section 04

데이터 분리와 선택

1. 데이터 분리

코드 7-11

```
sp <- split(iris, iris$Species)          # 품종별로 데이터 분리
sp                                       # 분리 결과 확인
summary(sp)                              # 분리 결과 요약
sp$setosa                                # setosa 품종의 데이터 확인
```

```
> sp <- split(iris, iris$Species)          # 품종별로 데이터 분리
> sp                                       # 분리 결과 확인
$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
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
...(중간 생략)
```

4. 데이터 분리와 선택

```
$versicolor
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
51           7.0         3.2          4.7         1.4 versicolor
52           6.4         3.2          4.5         1.5 versicolor
53           6.9         3.1          4.9         1.5 versicolor
54           5.5         2.3          4.0         1.3 versicolor
55           6.5         2.8          4.6         1.5 versicolor
...(중간 생략)
$virginica
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
101          6.3         3.3          6.0         2.5 virginica
102          5.8         2.7          5.1         1.9 virginica
103          7.1         3.0          5.9         2.1 virginica
104          6.3         2.9          5.6         1.8 virginica
105          6.5         3.0          5.8         2.2 virginica
...(이하 생략)
```

4. 데이터 분리와 선택

```
> summary(sp)                              # 분리 결과 요약
           Length Class      Mode
setosa     5      data.frame list
versicolor 5      data.frame list
virginica  5      data.frame list
> sp$setosa                                # setosa 품종의 데이터 확인
   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
...(중간 생략)
48          4.6         3.2          1.4         0.2  setosa
49          5.3         3.7          1.5         0.2  setosa
50          5.0         3.3          1.4         0.2  setosa
```

2. 데이터 선택

코드 7-12

```
subset(iris, Species == "setosa")
subset(iris, Sepal.Length > 7.5)
subset(iris, Sepal.Length > 5.1 &
             Sepal.Width > 3.9)
subset(iris, Sepal.Length > 7.6,
             select=c(Petal.Length,Petal.Width))
```

```
> subset(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
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
...(중간 생략)
48          4.6         3.2          1.4         0.2  setosa
49          5.3         3.7          1.5         0.2  setosa
50          5.0         3.3          1.4         0.2  setosa
```

4. 데이터 분리와 선택

```
> subset(iris, Sepal.Length > 7.5)
    Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
106          7.6         3.0          6.6         2.1 virginica
118          7.7         3.8          6.7         2.2 virginica
119          7.7         2.6          6.9         2.3 virginica
123          7.7         2.8          6.7         2.0 virginica
132          7.9         3.8          6.4         2.0 virginica
136          7.7         3.0          6.1         2.3 virginica
> subset(iris, Sepal.Length > 5.1 &
+         Sepal.Width > 3.9)
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
15          5.8         4.0          1.2         0.2  setosa
16          5.7         4.4          1.5         0.4  setosa
33          5.2         4.1          1.5         0.1  setosa
34          5.5         4.2          1.4         0.2  setosa
> subset(iris, Sepal.Length > 7.6,
+       select=c(Petal.Length,Petal.Width))
    Petal.Length Petal.Width
118          6.7         2.2
119          6.9         2.3
123          6.7         2.0
132          6.4         2.0
136          6.1         2.3
```

Section 05

데이터 샘플링과 조합

1. 데이터 샘플링

- **샘플링(sampling):** 통계용어로, 주어진 값들이 있을 때 그중에서 임의의 개수의 값들을 추출하는 작업
- 샘플링이 필요한 경우의 예: 데이터셋의 크기가 너무 커서 데이터 분석에 시간이 많이 걸리는 경우, 일부 데이터만 샘플링하여 대략의 결과를 미리 확인하고자 할 때
- **복원추출:** 한번 뽑은 것을 다시 뽑을 수 있는 추출

  ex) 주머니에서 꺼낸 구슬을 도로 넣어 원상복구한 다음에 다시 구슬을 뽑음
- **비복원추출:** 한번 뽑은 것을 다시 뽑을 수 없는 추출

  ex) 한번 주머니에서 꺼낸 구슬은 다시 넣지 않음

1.1 숫자를 임의로 추출하기

코드 7-13

```
x <- 1:100
y <- sample(x, size=10, replace = FALSE)          # 비복원추출
y
```

```
> x <- 1:100
> y <- sample(x, size=10, replace = FALSE)    # 비복원추출
> y
 [1] 57 90 20 86 50 85 49  1 60 38
```

1.2 행을 임의로 추출하기

코드 7-14

```
idx <- sample(1:nrow(iris), size=50,
              replace = FALSE)
iris.50 <- iris[idx,]                    # 50개의 행 추출
dim(iris.50)                             # 행과 열의 개수 확인
head(iris.50)
```

```
> idx <- sample(1:nrow(iris), size=50,
+             replace = FALSE)
> iris.50 <- iris[idx,]                  # 50개의 행 추출
> dim(iris.50)                           # 행과 열의 개수 확인
[1] 50  5
> head(iris.50)
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
74           6.1         2.8          4.7         1.2 versicolor
37           5.5         3.5          1.3         0.2     setosa
62           5.9         3.0          4.2         1.5 versicolor
8            5.0         3.4          1.5         0.2     setosa
54           5.5         2.3          4.0         1.3 versicolor
131          7.4         2.8          6.1         1.9  virginica
```

1.3 set.seed( ) 함수 이해하기

코드 7-15

```
sample(1:20, size=5)
sample(1:20, size=5)
sample(1:20, size=5)

set.seed(100)
sample(1:20, size=5)
set.seed(100)
sample(1:20, size=5)
set.seed(100)
sample(1:20, size=5)
```

5. 데이터 샘플링과 조합

```
> sample(1:20, size=5)
[1]  3  9  8  1 17
> sample(1:20, size=5)
[1] 11 14  7 12 13
> sample(1:20, size=5)
[1]  2 14 16 20 13
>
> set.seed(100)
> sample(1:20, size=5)
[1]  7  5 10  1  8
> set.seed(100)
> sample(1:20, size=5)
[1]  7  5 10  1  8
> set.seed(100)
> sample(1:20, size=5)
[1]  7  5 10  1  8
```

2. 데이터 조합

- **조합(combination):** 글자 그대로 주어진 데이터값들 중에서 몇 개씩 짝을 지어 추출하는 작업

코드 7-16

```
combn(1:5,3)                                    # 1~5에서 3개를 뽑는 조합

x = c("red","green","blue","black","white")
com <- combn(x,2)                               # x의 원소를 2개씩 뽑는 조합
com

for(i in 1:ncol(com)) {                         # 조합을 출력
   cat(com[,i], "₩n")
}
```

```
> combn(1:5,3)                          # 1~5에서 3개를 뽑는 조합
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    1    1    1    1    1    1    2    2    2     3
[2,]    2    2    2    3    3    4    3    3    4     4
[3,]    3    4    5    4    5    5    4    5    5     5
>
```

5. 데이터 샘플링과 조합

```
> x = c("red","green","blue","black","white")
> com <- combn(x,2)                    # x의 원소를 2개씩 뽑는 조합
> com
     [,1]    [,2]    [,3]    [,4]    [,5]    [,6]    [,7]    [,8]    [,9]    [,10]
[1,] "red"   "red"   "red"   "red"   "green" "green" "green" "blue"  "blue"  "black"
[2,] "green" "blue"  "black" "white" "blue"  "black" "white" "black" "white" "white"
>
> for(i in 1:ncol(com)) {              # 조합을 출력
+   cat(com[,i], "\n")
+ }
red green
red blue
red black
red white
green blue
green black
green white
blue black
blue white
black white
```

Section 06

데이터 집계와 병합

1. 데이터 집계

1.1 iris 데이터셋에서 각 변수의 품종별 평균 출력

- 2차원 데이터는 데이터 그룹에 대해서 합계나 평균을 계산해야 하는 일이 많음
- 이와 같은 작업을 **집계(aggregation)**라고 함
- R에서는 aggregate() 함수를 통해서 사용 가능

코드 7-17

```
agg <- aggregate(iris[,-5], by=list(iris$Species),
                 FUN=mean)
agg
```

```
> agg <- aggregate(iris[,-5], by=list(iris$Species),
+                  FUN=mean)
> agg
     Group.1 Sepal.Length Sepal.Width Petal.Length Petal.Width
1     setosa        5.006       3.428        1.462       0.246
2 versicolor        5.936       2.770        4.260       1.326
3  virginica        6.588       2.974        5.552       2.026
```

6. 데이터 집계와 병합

- `iris[,-5]`
  집계 작업을 수행할 대상이 되는 데이터셋을 의미한다.
- `by=list(iris$Species)`
  집계 작업의 기준이 품종(Species) 열의 값을 의미한다.
- `FUN=mean`
  집계 작업의 내용이 평균(mean) 계산임을 의미한다.

```
> agg <- aggregate(iris[,-5], by=list(품종=iris$Species),
+                  FUN=mean)
> agg
        품종  Sepal.Length  Sepal.Width  Petal.Length  Petal.Width
1     setosa         5.006        3.428         1.462        0.246
2 versicolor         5.936        2.770         4.260        1.326
3  virginica         6.588        2.974         5.552        2.026
```

1.1 iris 데이터셋에서 각 변수의 품종별 표준편차 출력

코드 7-18

```
agg <- aggregate(iris[,-5], by=list(표준편차=iris$Species),
                 FUN=sd)
agg
```

```
> agg <- aggregate(iris[,-5], by=list(표준편차=iris$Species),
+                  FUN=sd)
> agg
    표준편차 Sepal.Length Sepal.Width Petal.Length Petal.Width
1     setosa    0.3524897   0.3790644    0.1736640   0.1053856
2 versicolor    0.5161711   0.3137983    0.4699110   0.1977527
3  virginica    0.6358796   0.3224966    0.5518947   0.2746501
```

1.2 mtcars 데이터셋에서 각 변수의 최댓값 출력

코드 7-19

```
head(mtcars)
agg <- aggregate(mtcars, by=list(cyl=mtcars$cyl,
                                 vs=mtcars$vs),FUN=max)
agg
```

```
> head(mtcars)
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
> agg <- aggregate(mtcars, by=list(cyl=mtcars$cyl,
+                                  vs=mtcars$vs),FUN=max)
> agg
```

6. 데이터 집계와 병합

	cyl	vs	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
1	4	0	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2
2	6	0	21.0	6	160.0	175	3.90	2.875	17.02	0	1	5	6
3	8	0	19.2	8	472.0	335	4.22	5.424	18.00	0	1	5	8
4	4	1	33.9	4	146.7	113	4.93	3.190	22.90	1	1	5	2
5	6	1	21.4	6	258.0	123	3.92	3.460	20.22	1	0	4	4

2. 데이터 병합

- **병합(merge) :** 분리된 데이터 파일을 공통 열을 기준으로 하나로 합치는 작업

name	math
a	90
b	80
c	40

(a) 파일 x

name	korean
a	75
b	60
d	90

(b) 파일 y

그림 7-2 병합이 필요한 x, y 파일

코드 7-20

```
x <- data.frame(name=c("a","b","c"), math=c(90,80,40))
y <- data.frame(name=c("a","b","d"), korean=c(75,60,90))
x
y
```

```
> x <- data.frame(name=c("a","b","c"), math=c(90,80,40))
> y <- data.frame(name=c("a","b","d"), korean=c(75,60,90))
> x
  name math
1    a   90
2    b   80
3    c   40
> y
  name korean
1    a     75
2    b     60
3    d     90
```

코드 7-21

```
z <- merge(x,y, by=c("name"))
z
```

```
> z <- merge(x,y, by=c("name"))
> z
  name math korean
1    a   90     75
2    b   80     60
```

- **x, y**
  병합할 대상 데이터셋이다.
- **by=c("name")**
  병합의 기준이 되는 열이 name임을 의미한다.

코드 7-22

```
merge(x,y, all.x=T)    # 첫 번째 데이터셋의 행들은 모두 표시되도록
merge(x,y, all.y=T)    # 두 번째 데이터셋의 행들은 모두 표시되도록
merge(x,y, all=T)      # 두 데이터셋의 모든 행들이 표시되도록
```

```
> merge(x,y, all.x=T)          # 첫 번째 데이터셋의 행들은 모두 표시되도록
  name math korean
1    a   90     75
2    b   80     60
3    c   40     NA
> merge(x,y, all.y=T)          # 두 번째 데이터셋의 행들은 모두 표시되도록
  name math korean
1    a   90     75
2    b   80     60
3    d   NA     90
```

```
> merge(x,y, all=T)          # 두 데이터셋의 모든 행들이 표시되도록
  name math korean
1    a   90     75
2    b   80     60
3    c   40     NA
4    d   NA     90
```

코드 7-23 공통 열의 이름이 다를 때

```
x <- data.frame(name=c("a","b","c"), math=c(90,80,40))
y <- data.frame(sname=c("a","b","d"), korean=c(75,60,90))
x                                  # 병합 기준 열의 이름이 name
y                                  # 병합 기준 열의 이름이 sname
merge(x,y, by.x=c("name"), by.y=c("sname"))
```

6. 데이터 집계와 병합

```
> x <- data.frame(name=c("a","b","c"), math=c(90,80,40))
> y <- data.frame(sname=c("a","b","d"), korean=c(75,60,90))
> x                              # 병합 기준 열의 이름이 name
  name math
1    a   90
2    b   80
3    c   40
> y                              # 병합 기준 열의 이름이 sname
  sname korean
1     a     75
2     b     60
3     d     90
> merge(x,y, by.x=c("name"), by.y=c("sname"))
  name math korean
1    a   90     75
2    b   80     60
```

Thank you!

한빛아카데미 Hanbit Academy, Inc.

Copyright© 2023 Hanbit Academy, Inc.
All rights reserved.