R을 이용한 자료분석. 마지막 수행평가 모범 답안

YL

2019-5-29

문항 1 (40점)

방정식 $ax^2 + bx + c = 0$의 근은 다음의 "근의 공식"으로 구할 수 있다.

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

방정식의 계수 a, b, c의 값을 전달할 때 근의 공식에 따라 두 개의 해(upper, lower)를 리스트 형식으로 반환하는 R 함수를 만들되 비파이프 버전과(i) 파이프 버전 두 개를(ii) 나누어 작성한뒤에 a = 1, b = 2, c = 1의 경우를 시험 출력한다(iii). 각각을 `getSolve1`과 `getSolve2`로 한다. 즉, `getSolve2`에는 magrittr 패키지가 필요하다.

비파이프 버전

근의 공식에서 중요한 점은 제곱근 안의 계산값이 음수가 되면 실수 해가 나오지 않아서 R에서는 에러로 발현된다는 점이다. 따라서 a, b, c를 넣어서 계산하기 전에 `b^2 - 4 * a * c < 0` 여부를 파악해야 한다. 아래에서는 그 값을 s에 넣었다.

```
getSolve <- function(a, b, c) {
    # Non-pipe-version

        s <- b^2 - 4 * a * c
    if (s < 0) "no real solution" else {
            upper <- (-b + sqrt(s)) / (2 * a)
                lower <- (-b - sqrt(s)) / (2 * a)

                return(list(upper = upper, lower = lower))
                }
}
```

파이프 버전

파이프 연산에서 특이할 만한 점은 없다.

```r
getSolve2 <- function(a, b, c) {
    # Pipe-version

        require(magrittr)
        s <- b %>% "^"(2) %>% "-"(4 %>% "*"(a) %>% "*"(c))
        if (s %>% "<"(0)) "no real solution" else {
                upper <- (-b %>% "+"(s %>% sqrt)) %>% "/"(2 %>% "*"(a))
                lower <- (-b %>% "-"(s %>% sqrt)) %>% "/"(2 %>% "*"(a))

                return(list(upper = upper, lower = lower))
        }
}
```

시험 출력

a, b, c를 명시적으로 지명할 수도 있고 단순히 순서에 따라 입력하여도 똑같이 작동한다.

```r
getSolve(a = 1, b = 2, c = 1)

## $upper
## [1] -1
##
## $lower
## [1] -1

getSolve2(1, 2, 1)

## $upper
## [1] -1
##
## $lower
## [1] -1
```

문항 2A (선택 60점)

실습 중에 다루었던 Times Higher Education의 대학랭킹 자료를 다시 받아서(i: 로딩)

```
> theKorea <- read.csv("https://raw.githubusercontent.com/ylee03/
r_for_students/master/theKorea.csv", header = TRUE)
```

국내대학이 아닌 대학 자료와 2011년 자료를 제거한 뒤(ii), 연구실적(research)과 산학재원(industry.income)과의 관계를 산포도로 묘사하되(iii), 산포도 내 점의 모양을 대학에 따라 다르게 표현하라(iv). 점의 모양은 숫자여도 좋다.

자료 로딩

```
theKorea <- read.csv("https://raw.githubusercontent.com/ylee03/
r_for_students/master/theKorea.csv", header = TRUE) #
```

자료 정돈

대한민국 대학이 든 행은 아래의 코드로 찾는다.

```
grep("South Korea", theKorea$name)

##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
## [16] 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
## [31] 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45
## [46] 47 48 49 51 52 53 54 55 56 57 58 59 60 61
```

지난 시간에 자료철을 분철하는 함수인 subset을 써서 2011년 자료를 빼내는 과정을 다음의 코드로 행할 수 있다. (다음 코드의 출력은 길기 때문에 포함하지 않았다)

```
subset(theKorea, year != 2011)
```

문자열을 찾을 때 썼던 grep 함수도 적용할 수 있다.

```
grep(2011, theKorea$year)

## [1] 58 59 60 61
```

한 번에 처리할 수도 있지만 대한민국 자료만 골라낸 뒤 2011년 자료를 빼내는 것이 이해하기 쉬울 것이다. 빼낸 뒤 2011년 자료가 확실히 빠졌는지 집계를 냈다. 작업하기 전에 보관본(buKorea)을 만들었다.

```
buKorea <- theKorea
newKorea <- theKorea[grep("South Korea", theKorea$name), ]
newKorea %<>% subset(year != 2011)
newKorea$year %>% table

## .
## 2012 2013 2014 2015 2016
##    7    6    9    9   24
```

또는 다음처럼 한다. (먼저 자료철을 원상복귀해야 한다.)

```
theKorea <- buKorea
newKorea <- theKorea[grep("South Korea", theKorea$name), ]
newKorea <- newKorea[-grep(2011, newKorea$year), ]
newKorea$year %>% table

## .
## 2012 2013 2014 2015 2016
##    7    6    9    9   24
```

산포도

산포도를 그리는 가장 간단한 방식은 `plot(y ~ x, dataset)` 꼴을 유지하면 되는데 산점의 모양을 대학에 따라 바꾸어야 하므로 name 변수를 숫자로 변환하여야 한다. 대학명의 총 개수를 알려면 다음을 행한다. 모두 26개임을 알 수 있다.

```
newKorea$name %>% levels %>% length
```

```
## [1] 26
```

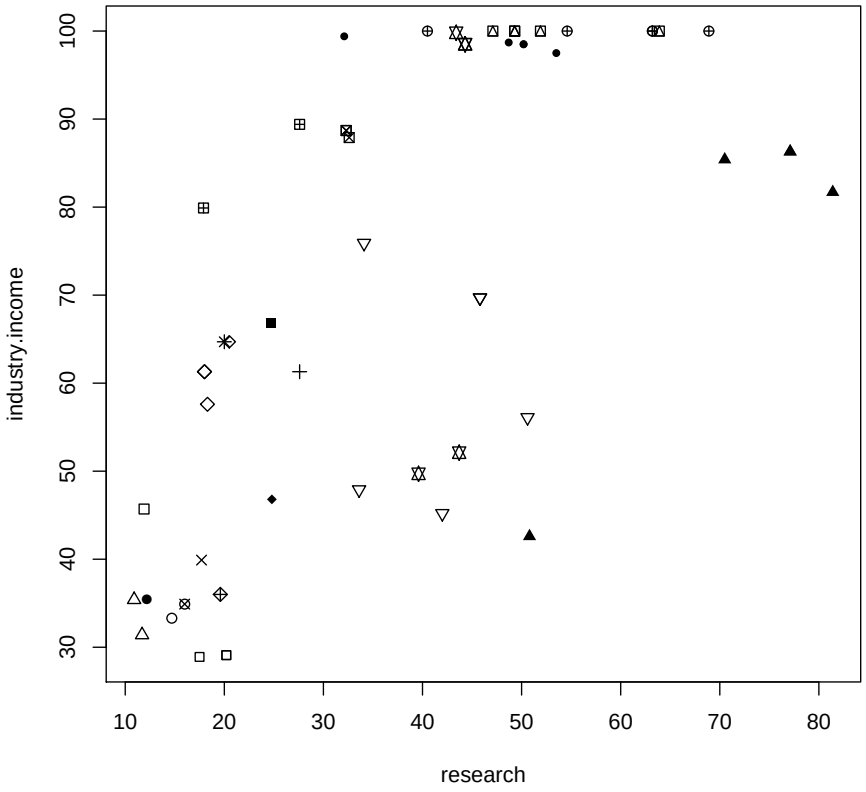
이를 `pch = ??`의 물음표 자리에 넣으려면 요인을 숫자로 변환하여야 한다.

```
newKorea$name %>% as.numeric %>% table
```

```
## .
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
##  1  1  1  1  1  3  1  3  1  1  5  5  2  1  5  1
## 17 18 19 21 23 24 25 26
##  1  5  1  5  3  1  1  5
```

그림을 그리기 전에 잠깐. plot 함수의 pch는 하필이면 0–25의 정수를 받을 수 있다. 즉, 위로 변환한 숫자를 바로 넣으면 26에서 에러가 발생한다는 점이다. 여기서 1을 빼야 한다. 0은 전달할 수 있다.

```
plot(industry.income ~ research, pch = as.numeric(name) - 1 ,
newKorea) #
```



문항 2B (선택 60점)

17명의 소아에서 두 개의 기기(m1, m2)로 각 두 번씩(o1, o2) 측정한 폐활량(spirometry) 측정값을 자료철 spiro로 불러들여서(i),

```
> spiro <- read.csv("https://raw.githubusercontent.com/ylee03/
r_for_students/master/spiro.csv", header = TRUE)
```

횡형(wide form)의 자료를 종형(long form)으로 변환하라(ii). 변환의 결과는 같은 저장소에 있는 pefr.csv와 동일하면 되지만 변수명만은 학생 각자의 임의에 맡긴다. 기기 종류에 따라, 반복 회수에 따라 폐활량 측정값을 boxplot으로 묘사하라(iii). 구분 가능한 네 개의 boxplot이 그려지고 지시되어야 한다(iv).

로딩

다음으로 로딩한 뒤 변수명을 확인한다.

```
## 'data.frame': 17 obs. of  5 variables:
##  $ id  : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ m1o1: int  494 395 516 434 476 557 413 442 650 433 ...
```

```
##  $ m1o2: int  490 397 512 401 470 611 415 431 638 429 ...
##  $ m2o1: int  512 430 520 428 500 600 364 380 658 445 ...
##  $ m2o2: int  525 415 508 444 500 625 460 390 642 432 ...
```

```
spiro <- read.csv("https://raw.githubusercontent.com/ylee03/
r_for_students/master/spiro.csv", header = TRUE) #
```

모으기 (Gather)

먼저 모으려는 대상 숫자들만 따로 조망한다. 17행 4열로 펼쳐 있으며 펼침에 사용된 변수는 기기 우선(m1, m2), 측정 순서(o1, o2) 나중이다.

```
spiro[, 2:5]
```

```
##    m1o1 m1o2 m2o1 m2o2
## 1   494  490  512  525
## 2   395  397  430  415
## 3   516  512  520  508
## 4   434  401  428  444
## 5   476  470  500  500
## 6   557  611  600  625
## 7   413  415  364  460
## 8   442  431  380  390
## 9   650  638  658  642
## 10  433  429  445  432
## 11  417  420  432  420
## 12  656  633  626  605
## 13  267  275  260  227
## 14  478  492  477  467
## 15  178  165  259  268
## 16  423  372  350  370
## 17  427  421  451  443
```

즉, 이를 매트릭스로 바꾼 뒤에 행 단위로 쌓는다면 기기에는 m1이 34개, m2가 34개 필요하고 o1, o2가 각각 17개씩 두 번 필요하다. 자료를 모으기 전에 자료 앞에 붙일 열쇠변수가 어떻게 붙을 것인지 미리 확인하면 다음과 같다.

```
1:17 %>% rep(4)                    # id
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
## [16] 16 17  1  2  3  4  5  6  7  8  9 10 11 12 13
## [31] 14 15 16 17  1  2  3  4  5  6  7  8  9 10 11
## [46] 12 13 14 15 16 17  1  2  3  4  5  6  7  8  9
## [61] 10 11 12 13 14 15 16 17
```

```
1:2 %>% rep(each = 34)             # method
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [24] 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2
## [47] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
1:2 %>% rep(each = 17) %>% rep(2) # occasion
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2
## [24] 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1
## [47] 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

관측값은 다음 명령으로 한 줄로 만들 수 있다. (`data.frame`의 열을 추출해도 여전히 `data.frame`이므로 바로 `as.vector` 함수로 보낼 수 없다. 반드시 `as.matrix`를 거쳐야 한다.)

```
spiro[, 2:5] %>% as.matrix %>% as.vector
```

```
##  [1] 494 395 516 434 476 557 413 442 650 433 417
## [12] 656 267 478 178 423 427 490 397 512 401 470
## [23] 611 415 431 638 429 420 633 275 492 165 372
## [34] 421 512 430 520 428 500 600 364 380 658 445
## [45] 432 626 260 477 259 350 451 525 415 508 444
## [56] 500 625 460 390 642 432 420 605 227 467 268
## [67] 370 443
```

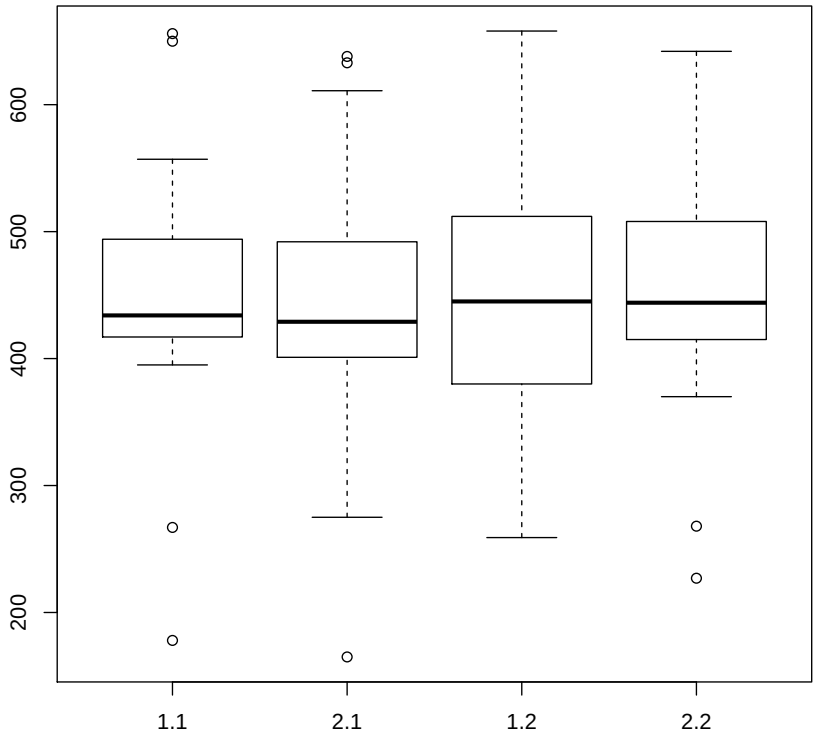
여기까지가 머릿속에서 정리되었다면 다음 R 구문으로 spiro 자료철을 모은다.

```
longSpiro <- data.frame(
    id = 1:17 %>% rep(4),
        method = 1:2 %>% rep(each = 34),
        occasion = 1:2 %>% rep(each = 17) %>% rep(2),
        spirometry = spiro[, 2:5] %>% as.matrix %>% as.vector)
```

그래프

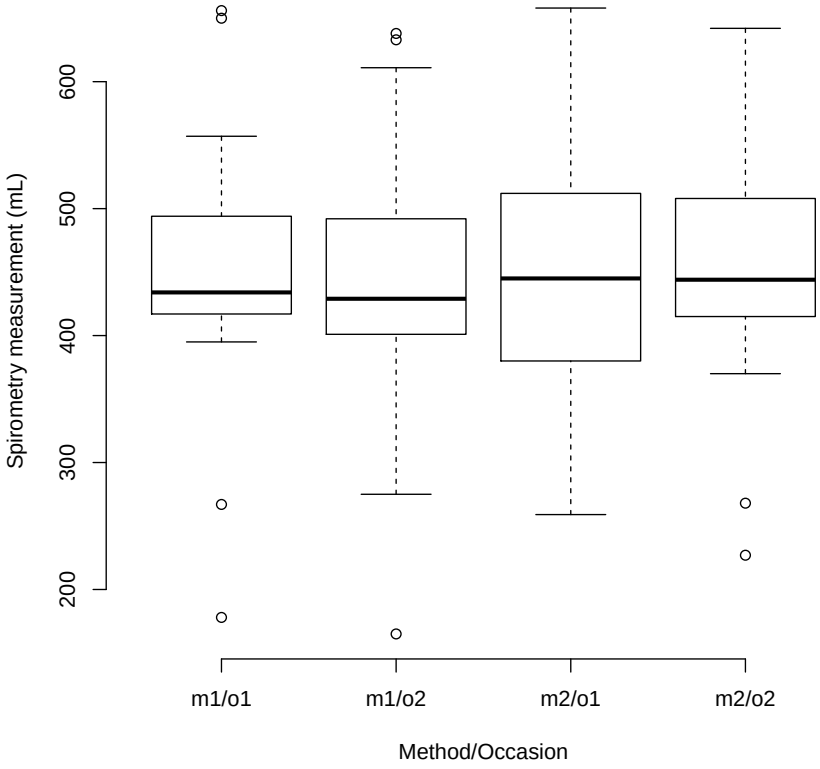
특별한 주문 없이 기기별 반복별 측정값의 boxplot을 그리는 것이므로 다음처럼 그리기만 하면 점수를 받을 수 있다.

```
boxplot(spirometry ~ occasion:method, longSpiro)
```



가로축을 식별하기 힘들기 때문에 다음처럼 바꾸는 것이 좋다. occasion을 위의 코드에서 앞에 두었으므로 method가 먼저 고정되고 occasion은 순환한다.

```
boxplot(spirometry ~ occasion:method, longSpiro,
    axes = FALSE,
        xlab = "Method/Occasion",
        ylab = "Spirometry measurement (mL)") #
axis(2)
axis(1,
    paste(c("m1", "m2") %>% rep(each = 2),
            c("o1", "o2") %>% rep(2),
                sep = "/"),
                at = 1:4) #
```



여기까지. 질문은 dryl@icloud.com로 할 것.