Ch4. Lists

Jongrak Jeong

'2023-01-10

1. lists

Overview

```
# seven members - leaders, assistants, and workers

members <- list(leaders = c("Park", "Choi"),
                assistants = "Kang",
                workers = c("Lee", "Kim", "Hong", "Song")
                )

members
```

```
## $leaders
## [1] "Park" "Choi"
##
## $assistants
## [1] "Kang"
##
## $workers
## [1] "Lee"  "Kim"  "Hong" "Song"
```

```
class(members)
```

```
## [1] "list"
```

```
names(members)
```

```
## [1] "leaders"    "assistants" "workers"
```

Indexing

```
## [[]] > vector
members[[1]]
```

```
## [1] "Park" "Choi"
```

```
members[[2]]
```

```
## [1] "Kang"
```

```
## sub-list: [] > list
members[1]
```

```
## $leaders
## [1] "Park" "Choi"
```

```
members[2]
```

```
## $assistants
## [1] "Kang"
```

```
## indexing by the names as vector
members[[1]]
```

```
## [1] "Park" "Choi"
```

```
members[["leaders"]]
```

```
## [1] "Park" "Choi"
```

```
members$leaders
```

```
## [1] "Park" "Choi"
```

```
## indexing by the names as sub-list
members[1]
```

```
## $leaders
## [1] "Park" "Choi"
```

```
members["leaders"]
```

```
## $leaders
## [1] "Park" "Choi"
```

Making another list with some lists

```
### team <- list(members + salaries)
salaries <- list(leaders = c(250, 200),
                 assistant = 100,
                 members = c(300, 200, 180, 120, 100))
team <- list(m = members, s = salaries)

team
```

```
## $m
## $m$leaders
## [1] "Park" "Choi"
##
## $m$assistants
## [1] "Kang"
##
## $m$workers
## [1] "Lee"  "Kim"  "Hong" "Song"
##
##
## $s
## $s$leaders
## [1] 250 200
##
## $s$assistant
## [1] 100
##
## $s$members
```

```
## [1] 300 200 180 120 100
```

```
### Making a list with c() function
### It is different with making it with list() > It is just a structure
team.1 <- c(m = members, s = salaries)

team.1
```

```
## $m.leaders
## [1] "Park" "Choi"
##
## $m.assistants
## [1] "Kang"
##
## $m.workers
## [1] "Lee"  "Kim"  "Hong" "Song"
##
## $s.leaders
## [1] 250 200
##
## $s.assistant
## [1] 100
##
## $s.members
## [1] 300 200 180 120 100
```

```
class(team)
```

```
## [1] "list"
```

```
class(team.1)
```

```
## [1] "list"
```

```
length(team) # list with two sub-list(?)
```

```
## [1] 2
```

```
length(team.1) # list with six sub-list
```

```
## [1] 6
```

2. lapply() and sapply()

lapply(): lapply(list, f). apply a function to the list

```
lapply(salaries, median)
```

```
## $leaders
## [1] 225
##
## $assistant
## [1] 100
##
## $members
## [1] 180
```

```r
# the output is also list
class(lapply(salaries, median))
```

```
## [1] "list"
```

sapply(): sapply(list, f). apply a function to the list and simplify it as vector (or matrix and array)

```r
sapply(salaries, median)
```

```
##   leaders assistant   members
##       225       100       180
```

```r
# the output is not a list
class(sapply(salaries, median))
```

```
## [1] "numeric"
```

vapply(): vapply(list, f). apply a function to the list and is able to denote the type of output

```r
sapply(salaries, range)
```

```
##      leaders assistant members
## [1,]     200       100     100
## [2,]     250       100     300
```

```r
vapply(salaries, range, c(min = 0, max = 0))
```

```
##     leaders assistant members
## min     200       100     100
## max     250       100     300
```

3. unlist(): list > vectors

unlist() and apply a function

```r
# median(salaries) > it returns error since it is not a vecotr but a list.
unlist(salaries)
```

```
##  leaders1  leaders2 assistant  members1  members2  members3  members4  members5
##       250       200       100       300       200       180       120       100
```

```r
class(unlist(salaries))
```

```
## [1] "numeric"
```

```r
median(unlist(salaries))
```

```
## [1] 190
```

4. application

ex1. user-defined function summarize()

Target - (average, standard deviation) - five-number summary (min, 25%, 50%, 75%, max) - out-liers > we need a list with three of elements. length: 2, 5, and non-defined

Define the function

```
summarize <- function(x) {
  mean.sd <- c(m = mean(x), s = sd(x))
  fivenum <- quantile(x, prob = c(0, 0.25, 0.5, 0.75, 1))
  lower <- fivenum[2] - 1.5 * (fivenum[4] - fivenum[2])
  upper <- fivenum[2] + 1.5 * (fivenum[4] - fivenum[2])
  outliers <- c(x[x < lower], x[x > upper])
  list(mean.sd = mean.sd, fivenum = fivenum, outliers = outliers)
}
```

Apply the function to a sample

```
set.seed(1); x <- rnorm(1000)

summarize(x)
```

```
## $mean.sd
##           m           s
## -0.01164814  1.03491584
##
## $fivenum
##          0%         25%         50%         75%        100%
## -3.00804860 -0.69737322 -0.03532423  0.68842795  3.81027668
##
## $outliers
##  [1] -2.888921 -3.008049 -2.939774 -2.996949  1.595281  1.511781  1.433024
##  [8]  1.980400  2.401618  1.465555  2.172612  1.586833  1.767287  1.682176
## [15]  1.432282  2.087167  1.869291  2.206102  2.307978  2.075245  1.464587
## [22]  1.441158  1.688873  1.586588  2.497662  1.519745  1.803142  1.719627
## [29]  2.649167  1.778429  1.971337  1.654145  1.512213  1.497041  1.887474
## [36]  1.473881  1.577892  1.763552  1.592914  1.895655  1.441820  1.709121
## [43]  1.435070  1.612347  1.772611  2.165369  1.480214  2.350554  2.446531
## [50]  2.284659  3.810277  1.593967  1.642028  1.549830  2.024842  1.584629
## [57]  1.677889  1.393846  1.502425  2.001719  2.675741  1.784663  1.763586
## [64]  1.829730  2.189752  1.556053  1.971572  1.753795  1.568365  1.659879
## [71]  1.743559  1.801725  1.771542  2.021347  1.692774  1.416827  1.382284
## [78]  1.979633  2.349493  1.800112  2.236323  2.005719  3.055742  1.398791
## [85]  1.772493  1.752036  1.644080  1.394253  1.773763  1.943536  2.321334
## [92]  2.169116  2.251883  2.210952  1.457738  2.027056
```

```
class(summarize(x))
```

```
## [1] "list"
```

```
# rounded output
lapply(summarize(x), round, 2)
```

```
## $mean.sd
```

```
##     m     s
## -0.01  1.03
##
## $fivenum
##    0%   25%   50%   75%  100%
## -3.01 -0.70 -0.04  0.69  3.81
##
## $outliers
##  [1] -2.89 -3.01 -2.94 -3.00  1.60  1.51  1.43  1.98  2.40  1.47  2.17  1.59
## [13]  1.77  1.68  1.43  2.09  1.87  2.21  2.31  2.08  1.46  1.44  1.69  1.59
## [25]  2.50  1.52  1.80  1.72  2.65  1.78  1.97  1.65  1.51  1.50  1.89  1.47
## [37]  1.58  1.76  1.59  1.90  1.44  1.71  1.44  1.61  1.77  2.17  1.48  2.35
## [49]  2.45  2.28  3.81  1.59  1.64  1.55  2.02  1.58  1.68  1.39  1.50  2.00
## [61]  2.68  1.78  1.76  1.83  2.19  1.56  1.97  1.75  1.57  1.66  1.74  1.80
## [73]  1.77  2.02  1.69  1.42  1.38  1.98  2.35  1.80  2.24  2.01  3.06  1.40
## [85]  1.77  1.75  1.64  1.39  1.77  1.94  2.32  2.17  2.25  2.21  1.46  2.03
```

ex2. classifying game cards

Target - Game cards: 4 kinds(Spade, Heart, Diamond, Club). 13 per kind, 52 per set. 4 sets. total 208 card - 52 cards sampling without replacement - counting the order of each kind

ex - S, S, D, C, S, H ... > S: 1, 2, 5 ... > D: 3, ...

Shuffle

```
kinds <- c("Spade", "Heart", "Diamond", "Club")
set.seed(1)
cards <- sample(rep(kinds, 13 * 4), 52, replace = F)

classified <- list()
cards
```

```
##  [1] "Club"    "Diamond" "Spade"   "Heart"   "Diamond" "Heart"   "Diamond"
##  [8] "Diamond" "Spade"   "Spade"   "Heart"   "Heart"   "Heart"   "Diamond"
## [15] "Spade"   "Diamond" "Spade"   "Spade"   "Heart"   "Spade"   "Heart"
## [22] "Heart"   "Heart"   "Spade"   "Club"    "Spade"   "Club"    "Diamond"
## [29] "Heart"   "Club"    "Heart"   "Heart"   "Diamond" "Club"    "Club"
## [36] "Club"    "Club"    "Spade"   "Diamond" "Club"    "Diamond" "Club"
## [43] "Spade"   "Diamond" "Heart"   "Diamond" "Heart"   "Heart"   "Club"
## [50] "Diamond" "Heart"   "Spade"
```

Counting

```
for (i in 1:52) {
  r <- cards[i]
  classified[[r]] <- c(classified[[r]], i)
}

classified
```

```
## $Club
##  [1]  1 25 27 30 34 35 36 37 40 42 49
##
## $Diamond
```

```
##  [1]  2  5  7  8 14 16 28 33 39 41 44 46 50
##
## $Spade
##  [1]  3  9 10 15 17 18 20 24 26 38 43 52
##
## $Heart
##  [1]  4  6 11 12 13 19 21 22 23 29 31 32 45 47 48 51
```

```
sapply(classified, length)
```

```
##    Club Diamond   Spade   Heart
##      11      13      12      16
```

ex3. application with linear regression function lm()

value(output) of lm() is 'lm'. But it is actually list

default setting

```
library(gclus)
```

```
## Loading required package: cluster
```

```
data(ozone)
str(ozone)
```

```
## 'data.frame':    330 obs. of  9 variables:
##  $ Ozone : int  3 5 5 6 4 4 6 7 4 6 ...
##  $ Temp  : int  40 45 54 35 45 55 41 44 54 51 ...
##  $ InvHt : int  2693 590 1450 1568 2631 554 2083 2654 5000 111 ...
##  $ Pres  : int  -25 -24 25 15 -33 -28 23 -2 -19 9 ...
##  $ Vis   : int  250 100 60 60 100 250 120 120 120 150 ...
##  $ Hgt   : int  5710 5700 5760 5720 5790 5790 5700 5700 5770 5720 ...
##  $ Hum   : int  28 37 51 69 19 25 73 59 27 44 ...
##  $ InvTmp: num  47.7 55 57 53.8 54.1 ...
##  $ Wind  : int  4 3 3 4 6 3 3 3 8 3 ...
```

linear regression model

```
reg.2 <- lm(Ozone ~ Temp + Pres, data = ozone)

summary(reg.2)
```

```
##
## Call:
## lm(formula = Ozone ~ Temp + Pres, data = ozone)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.6612  -3.5151  -0.3274   3.1176  15.2777
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -14.760083   1.210578 -12.193   <2e-16 ***
## Temp          0.425360   0.019386  21.942   <2e-16 ***
## Pres          0.015424   0.007848   1.966   0.0502 .
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.992 on 327 degrees of freedom
## Multiple R-squared:  0.6141, Adjusted R-squared:  0.6117
## F-statistic: 260.1 on 2 and 327 DF,  p-value: < 2.2e-16
```

```
class(reg.2)
```

```
## [1] "lm"
```

```
mode(reg.2) # type of (reg.2)
```

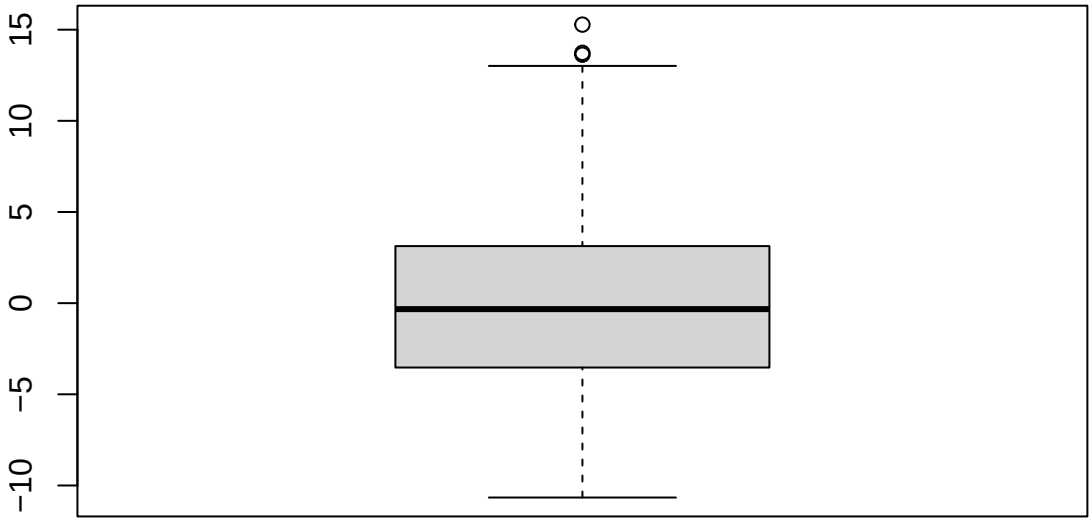
```
## [1] "list"
```

```
names(reg.2) # better than str(reg.2) in that it is too long
```

```
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
```

Residuals analysis and box plot > detecting outlier

```
resid.2 <- boxplot(reg.2$residuals)
```




```
class(resid.2)
```

```
## [1] "list"
```

```
names(resid.2)
```

```
## [1] "stats" "n"     "conf"  "out"   "group" "names"
```

```
# outliers
resid.2$out
```

```
##       53      124      220
## 15.27769 13.73127 13.62695
```

```
names(resid.2$out) # the case number of outliers
```

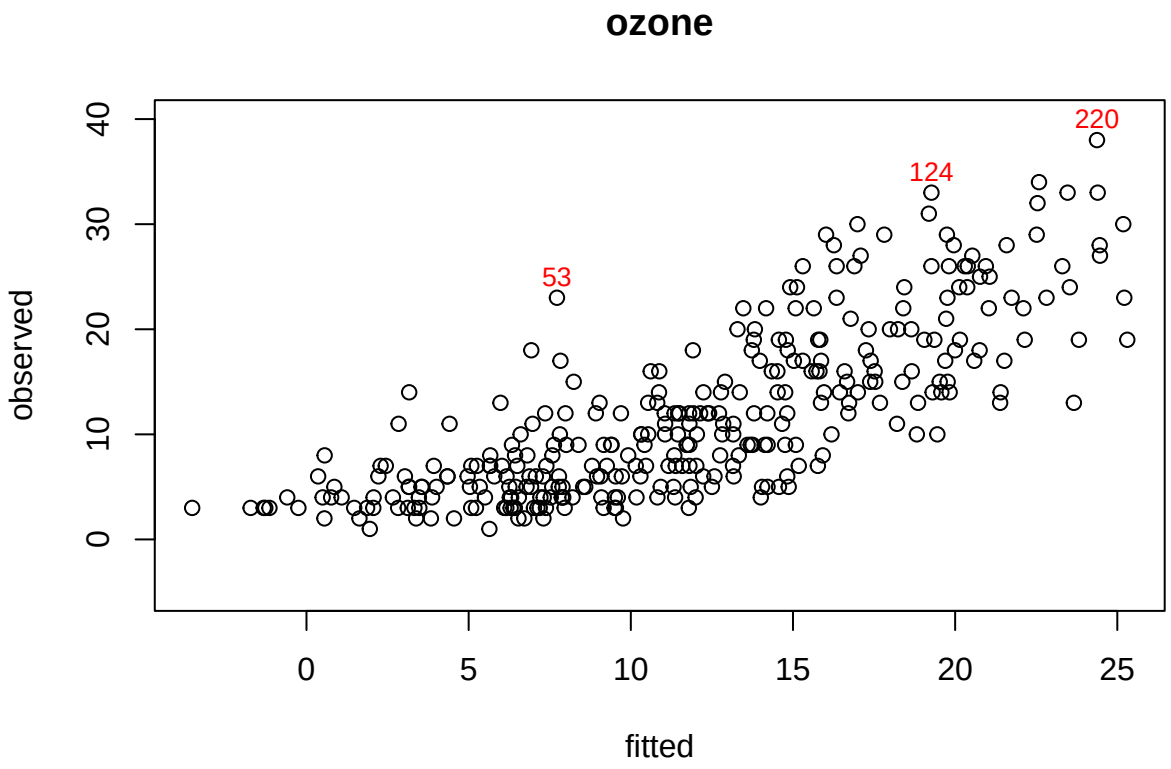
```
## [1] "53"  "124" "220"
```

```
ids.out <- as.numeric(names(resid.2$out))
ids.out
```

```
## [1]  53 124 220
```

Outlier marking on the scatterplot(observed : regression fitted value)

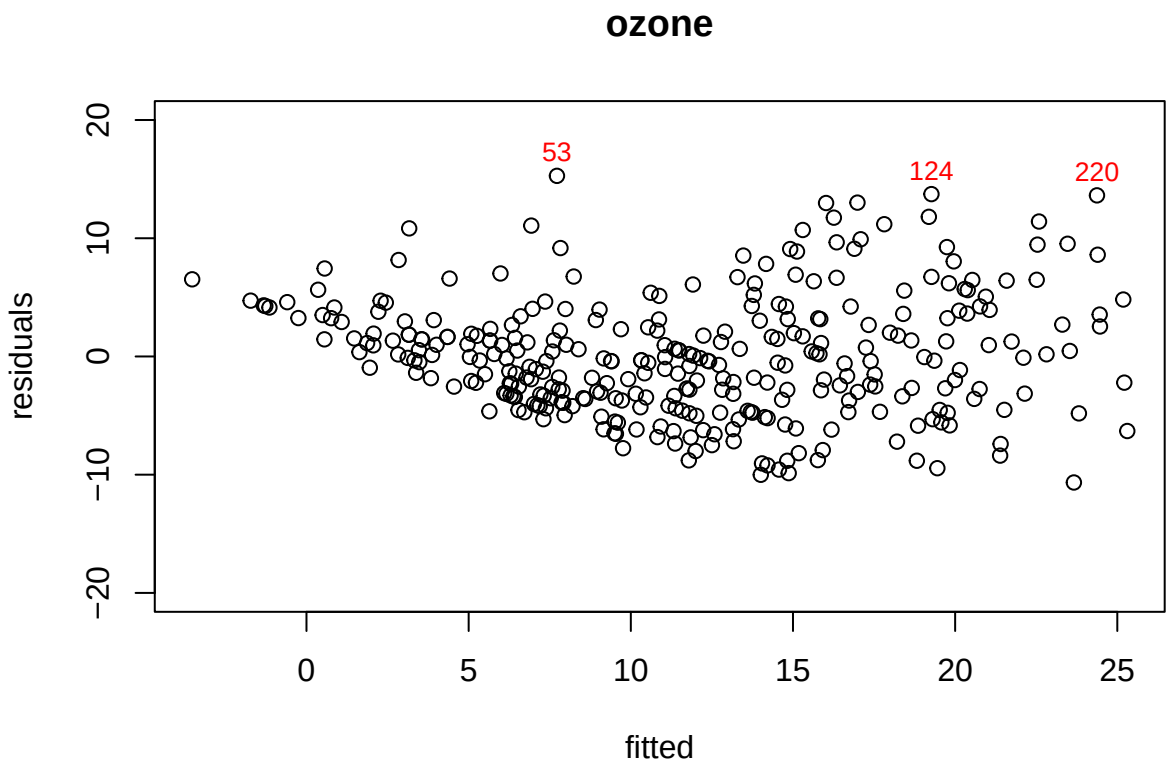
```
plot(ozone$Ozone ~ reg.2$fitted.values,
     main = "ozone", xlab = "fitted", ylab = "observed",
     ylim = c(-5, 40)) # scatterplot
text(reg.2$fitted.values[ids.out], ozone$Ozone[ids.out] + 2,
     ids.out, col = "red", cex = 0.8)
```



###

Outlier marking on the scatterplot(regression residuals : regression fitted value)

```
plot(reg.2$residuals ~ reg.2$fitted.values,
     main = "ozone", xlab = "fitted", ylab = "residuals",
     ylim = c(-20, 20)) # scatterplot.
     # x should not be 'fitted.values' and y should not be 'residuals' since reg.2 is 'lm' class.
text(reg.2$fitted.values[ids.out], reg.2$residuals[ids.out] + 2,
     ids.out, col = "red", cex = 0.8)
```


ozone
residuals
fitted
53
124
220
20
10
0
−10
−20
0
5
10
15
20
25