Advanced R: (Reference) method, name, value, and copy

川田恵介

# 関数とObject

## 基本文法

* データ(Object)をFunctionに入力し、 Objectを出力する

A <- c(1,2,3)  
  
B <- sum(A)  
  
sqrt(B)

[1] 2.44949

## Class

* あるObjectに関数を適用できるかどうかは、ObjectのClassで判別される
  + Class: Objectの”型” (消費財、資本、中間投入物などなど)
* class関数で確認できる

class(B)

[1] "numeric"

## list

* 最も多く採用されているclass
* -　多様な情報を入れ子構造で含めることができる
  + $で要素を抜き出せる
  + “Data”もリストの一種

C <- list(a = c("A","B"), b = 1)  
  
class(C)

[1] "list"

C$a

[1] "A" "B"

## list

* 無限に入れ子にできる

C <- list(a = c("A","B"), b = 1)  
  
D <- list(C = C,  
 D = 1)  
  
D$C$a

[1] "A" "B"

## おすすめ参考書

* [Advanced R](https://adv-r.hadley.nz/index.html)
  + [R for Data Science](https://r4ds.had.co.nz/)

# R6

* [Advanced R](https://adv-r.hadley.nz/index.html)
  + [R for Data Science](https://r4ds.had.co.nz/)

## 流れ作業の書き方: Pipe

* 現代的な文法 (関数ベース)
  + |> ないし %>%

c(1,2,3) |>   
 sum() |>   
 sqrt()

[1] 2.44949

## R6 object class

* Object自体の構造にも種類が存在: S4,S3,R6…
* R6: Pythonライク, Reference semantics, Method (Chane)
  + DoubleML や sl3 などが採用

library(R6)  
  
ExampleR6 <- R6Class("Example",   
 public = list(  
 value = 0,  
 define = function(x) {  
 self$value <- self$value + x   
 invisible(self)  
 },  
 sum = function() {  
 self$value <- sum(self$value)  
 invisible(self)  
 },  
 sqrt = function() {  
 self$value <- sqrt(self$value)  
 invisible(self)  
 }  
 )  
)

## Method

* Objectから関数を呼び出し、同じObject内のvalueを加工

A <- ExampleR6$new()  
  
A$define(c(1,2,3))  
A$value

[1] 1 2 3

A$sum()  
A$value

[1] 6

## Method chain

* Objectから関数を呼び出し、同じObject内のvalueを加工

A <- ExampleR6$new()  
  
A$define(c(1,2,3))$sum()$sqrt()$value

[1] 2.44949

# Name and Value

* name と values を区別することは有益
  + R6 class や多くのPython Libraryのような Reference Semantics が採用されている場合必須
  + 致命的なミスを犯す恐れ

## Name

* “データ”は、PCチップ上のどこかに保存
  + 真の住所が存在
  + ただの記号の羅列で覚えられない
* Name(参照名）をつけて活用

library(lobstr) # 真の住所を取得  
  
ReferenceName <- 1 # データを作り、Nameを作る  
  
obj\_addr(ReferenceName) # 真の住所

[1] "0x106341238"

* 日常例: 住所(GPS座標) 35.71037511368804, 139.76139216569044
  + 参照名: 東大経済学研究棟

## list

* 参照名の集合体
  + 巨大なリストを作ったとしても、それ自体はメモリを圧迫しない

## コピー

a <- 1  
  
b <- 1  
  
c <- a

* bとcの違いは？
  + cはaのソフトコピー: 住所は同じ

## 確認

obj\_addr(a)

[1] "0x106706e80"

obj\_addr(b)

[1] "0x106706e10"

obj\_addr(c)

[1] "0x106706e80"

## Copy-on-modify

* 以下を行うと何が起きる？

c <- c + 1

* Objectを加工する際に、実際のコピーが行われる

obj\_addr(a)

[1] "0x106706e80"

obj\_addr(c)

[1] "0x12965c568"

* “データ分析者”が意図せぬミスを避ける保守的な設定
  + しばしば不要なコピーを誘発し、計算速度が遅くなる

## R6の注意点

* Copy-on-modifyを採用していない
  + 勝手に設定が変更されてしまう可能性！！！
* コピー元を変更したくなければ、意図的に設定する必要がある
  + clone() 関数の利用
  + 安全のためにDeep = TRUEを設定

## No Copy-on-modify

library(mlr3verse)  
  
LASSO <- lrn("regr.glmnet")  
  
Ridge <- LASSO  
  
Ridge$param\_set$values$alpha = 0  
  
LASSO$param\_set

<ParamSet>  
 id class lower upper nlevels default parents  
 1: alignment ParamFct NA NA 2 lambda   
 2: alpha ParamDbl 0 1 Inf 1   
 3: big ParamDbl -Inf Inf Inf 9.9e+35   
 4: devmax ParamDbl 0 1 Inf 0.999   
 5: dfmax ParamInt 0 Inf Inf <NoDefault[3]>   
 6: eps ParamDbl 0 1 Inf 1e-06   
 7: epsnr ParamDbl 0 1 Inf 1e-08   
 8: exact ParamLgl NA NA 2 FALSE   
 9: exclude ParamInt 1 Inf Inf <NoDefault[3]>   
10: exmx ParamDbl -Inf Inf Inf 250   
11: family ParamFct NA NA 2 gaussian   
12: fdev ParamDbl 0 1 Inf 1e-05   
13: gamma ParamDbl -Inf Inf Inf 1 relax  
14: grouped ParamLgl NA NA 2 TRUE   
15: intercept ParamLgl NA NA 2 TRUE   
16: keep ParamLgl NA NA 2 FALSE   
17: lambda ParamUty NA NA Inf <NoDefault[3]>   
18: lambda.min.ratio ParamDbl 0 1 Inf <NoDefault[3]>   
19: lower.limits ParamUty NA NA Inf <NoDefault[3]>   
20: maxit ParamInt 1 Inf Inf 100000   
21: mnlam ParamInt 1 Inf Inf 5   
22: mxit ParamInt 1 Inf Inf 100   
23: mxitnr ParamInt 1 Inf Inf 25   
24: newoffset ParamUty NA NA Inf <NoDefault[3]>   
25: nlambda ParamInt 1 Inf Inf 100   
26: offset ParamUty NA NA Inf   
27: parallel ParamLgl NA NA 2 FALSE   
28: penalty.factor ParamUty NA NA Inf <NoDefault[3]>   
29: pmax ParamInt 0 Inf Inf <NoDefault[3]>   
30: pmin ParamDbl 0 1 Inf 1e-09   
31: prec ParamDbl -Inf Inf Inf 1e-10   
32: relax ParamLgl NA NA 2 FALSE   
33: s ParamDbl 0 Inf Inf 0.01   
34: standardize ParamLgl NA NA 2 TRUE   
35: standardize.response ParamLgl NA NA 2 FALSE   
36: thresh ParamDbl 0 Inf Inf 1e-07   
37: trace.it ParamInt 0 1 2 0   
38: type.gaussian ParamFct NA NA 2 <NoDefault[3]> family  
39: type.logistic ParamFct NA NA 2 <NoDefault[3]>   
40: type.multinomial ParamFct NA NA 2 <NoDefault[3]>   
41: upper.limits ParamUty NA NA Inf <NoDefault[3]>   
 id class lower upper nlevels default parents  
 value  
 1:   
 2: 0  
 3:   
 4:   
 5:   
 6:   
 7:   
 8:   
 9:   
10:   
11: gaussian  
12:   
13:   
14:   
15:   
16:   
17:   
18:   
19:   
20:   
21:   
22:   
23:   
24:   
25:   
26:   
27:   
28:   
29:   
30:   
31:   
32:   
33:   
34:   
35:   
36:   
37:   
38:   
39:   
40:   
41:   
 value

## Copy-on-modify

library(mlr3verse)  
  
LASSO <- lrn("regr.glmnet")  
  
Ridge <- LASSO$clone(deep = TRUE)  
  
Ridge$param\_set$values$alpha = 0  
  
LASSO$param\_set

<ParamSet>  
 id class lower upper nlevels default parents  
 1: alignment ParamFct NA NA 2 lambda   
 2: alpha ParamDbl 0 1 Inf 1   
 3: big ParamDbl -Inf Inf Inf 9.9e+35   
 4: devmax ParamDbl 0 1 Inf 0.999   
 5: dfmax ParamInt 0 Inf Inf <NoDefault[3]>   
 6: eps ParamDbl 0 1 Inf 1e-06   
 7: epsnr ParamDbl 0 1 Inf 1e-08   
 8: exact ParamLgl NA NA 2 FALSE   
 9: exclude ParamInt 1 Inf Inf <NoDefault[3]>   
10: exmx ParamDbl -Inf Inf Inf 250   
11: family ParamFct NA NA 2 gaussian   
12: fdev ParamDbl 0 1 Inf 1e-05   
13: gamma ParamDbl -Inf Inf Inf 1 relax  
14: grouped ParamLgl NA NA 2 TRUE   
15: intercept ParamLgl NA NA 2 TRUE   
16: keep ParamLgl NA NA 2 FALSE   
17: lambda ParamUty NA NA Inf <NoDefault[3]>   
18: lambda.min.ratio ParamDbl 0 1 Inf <NoDefault[3]>   
19: lower.limits ParamUty NA NA Inf <NoDefault[3]>   
20: maxit ParamInt 1 Inf Inf 100000   
21: mnlam ParamInt 1 Inf Inf 5   
22: mxit ParamInt 1 Inf Inf 100   
23: mxitnr ParamInt 1 Inf Inf 25   
24: newoffset ParamUty NA NA Inf <NoDefault[3]>   
25: nlambda ParamInt 1 Inf Inf 100   
26: offset ParamUty NA NA Inf   
27: parallel ParamLgl NA NA 2 FALSE   
28: penalty.factor ParamUty NA NA Inf <NoDefault[3]>   
29: pmax ParamInt 0 Inf Inf <NoDefault[3]>   
30: pmin ParamDbl 0 1 Inf 1e-09   
31: prec ParamDbl -Inf Inf Inf 1e-10   
32: relax ParamLgl NA NA 2 FALSE   
33: s ParamDbl 0 Inf Inf 0.01   
34: standardize ParamLgl NA NA 2 TRUE   
35: standardize.response ParamLgl NA NA 2 FALSE   
36: thresh ParamDbl 0 Inf Inf 1e-07   
37: trace.it ParamInt 0 1 2 0   
38: type.gaussian ParamFct NA NA 2 <NoDefault[3]> family  
39: type.logistic ParamFct NA NA 2 <NoDefault[3]>   
40: type.multinomial ParamFct NA NA 2 <NoDefault[3]>   
41: upper.limits ParamUty NA NA Inf <NoDefault[3]>   
 id class lower upper nlevels default parents  
 value  
 1:   
 2:   
 3:   
 4:   
 5:   
 6:   
 7:   
 8:   
 9:   
10:   
11: gaussian  
12:   
13:   
14:   
15:   
16:   
17:   
18:   
19:   
20:   
21:   
22:   
23:   
24:   
25:   
26:   
27:   
28:   
29:   
30:   
31:   
32:   
33:   
34:   
35:   
36:   
37:   
38:   
39:   
40:   
41:   
 value