APLM hw03and04

san teng

2018/12/26

```r
library(stringr)
setwd("D:/github_oicjacky/Applied Statistical Linear Model/APLM hw03")
rm(list = ls())
data <- read.table("CH09TA01.txt")
colnames(data) <- c("X1","X2","X3","X4","X5","Y","lnY")

n <- dim(data)[1]
P <- 5            # 5 predict variable
```

power set

```r
powerset = function(s){
  len = length(s)
  l = vector(mode="list",length=2^len)
  l[[1]]=numeric()
  counter = 1
  for(x in 1:length(s)){
    for(subset in 1:counter){
      counter=counter+1
      l[[counter]] = c(l[[subset]],s[x])
    }
  }
  return(l)
}
#powerset(1:P)
```

R^2 & adjusted R^2

```r
# R squares
R.square <- function(data , mod){

  SSTO <- sum( ( data$Y - mean(data$Y) )^2 )
  SSE <- sum(mod$residuals^2)

  return( 1 - (SSE / SSTO) )
}
# adjusted R squares
R.adjust <- function(data , mod){
  n <- dim(data)[1]
  p <- mod$rank

  SSTO <- sum( ( data$Y - mean(data$Y) )^2 )
```

```
  SSE <- sum(mod$residuals^2)

  return( 1 - ( (n-1) / (n-p) ) * (SSE / SSTO) )
}
```

```
#length(powerset(1:P))
all_possible <- powerset(1:P)
R_adj.square <- R_square <- c()
variable <- c()
for(i in 2: length(all_possible) ) {

  if( length(all_possible[[i]]) == 1 ){
    A <- data.frame( data[, all_possible[[i]] ] ,
                     Y = data$Y                          )
    colnames(A)[-2] <- paste0("X",all_possible[[i]])
    a <- R.adjust(A , lm( Y ~ A[,1] ,A) )
    b <- R.square(A , lm( Y ~ A[,1] ,A) )
    #print(colnames(A))
  }else if( length(all_possible[[i]]) == 2 ){
    A <- data.frame( data[, all_possible[[i]] ] [ ,1] ,
                     data[, all_possible[[i]] ] [ ,2] ,
                     Y = data$Y                          )
    colnames(A)[-3] <- colnames(data[, all_possible[[i]] ])
    a <- R.adjust(A , lm( Y ~ A[,1] + A[,2] ,A) )
    b <- R.square(A , lm( Y ~ A[,1] + A[,2] ,A) )
    #print(colnames(A))

  }else if( length(all_possible[[i]]) == 3 ){
    A <- data.frame( data[, all_possible[[i]] ] [ ,1] ,
                     data[, all_possible[[i]] ] [ ,2] ,
                     data[, all_possible[[i]] ] [ ,3] ,
                     Y = data$Y                          )
    colnames(A)[-4] <- colnames(data[, all_possible[[i]] ])
    a <- R.adjust(A , lm( Y ~ A[,1] + A[,2] + A[,3] ,A) )
    b <- R.square(A , lm( Y ~ A[,1] + A[,2] + A[,3] ,A) )
    #print(colnames(A))

  }else if( length(all_possible[[i]]) == 4 ){
    A <- data.frame( data[, all_possible[[i]] ] [ ,1] ,
                     data[, all_possible[[i]] ] [ ,2] ,
                     data[, all_possible[[i]] ] [ ,3] ,
                     data[, all_possible[[i]] ] [ ,4] ,
                     Y = data$Y                          )
    colnames(A)[-5] <- colnames(data[, all_possible[[i]] ])
    a <- R.adjust(A , lm( Y ~ A[,1] + A[,2] + A[,3] + A[,4] ,A) )
    b <- R.square(A , lm( Y ~ A[,1] + A[,2] + A[,3] + A[,4] ,A) )
    #print(colnames(A))

  }else if( length(all_possible[[i]]) == 5 ){
    A <- data.frame( data[, all_possible[[i]] ] [ ,1] ,
                     data[, all_possible[[i]] ] [ ,2] ,
                     data[, all_possible[[i]] ] [ ,3] ,
                     data[, all_possible[[i]] ] [ ,4] ,
                     data[, all_possible[[i]] ] [ ,5] ,
```

```
                    Y = data$Y                           )
    colnames(A)[-6] <- colnames(data[, all_possible[[i]] ])
    a <- R.adjust(A , lm( Y ~ A[,1] + A[,2] + A[,3] + A[,4] + A[,5] ,A) )
    b <- R.square(A , lm( Y ~ A[,1] + A[,2] + A[,3] + A[,4] + A[,5] ,A) )
    #print(colnames(A))
  }

  R_adj.square <- rbind(R_adj.square , a )
  R_square <- rbind(R_square , b )
  variable <- c(variable ,str_c(colnames(A)[-dim(A)[2]] ,collapse = ","))
}
A <- data.frame(R_square = R_square ,
                R_adj.square = R_adj.square ,
                variable = variable )
```

the model with highest R^2

```
##       R_square R_adj.square       variable
## b.30 0.6949877    0.6632155 X1,X2,X3,X4,X5
```

the model with highest adjusted R^2

```
##       R_square R_adj.square    variable
## b.22 0.6910939    0.6658771 X1,X2,X3,X5
```

CV(1) or leave-one-out cross validation

```
candidate <- powerset(1:P)[-1]

CV_value <- rep(0 ,length(candidate) )
for (j in 1:length(candidate) ) {

  pred.value <- rep(0 ,n)
  for(i in 1:n){
    d <- combn(1:n ,1)[,i] # CV(1) or leave-one-out cross validation
    data_train <- data[-d ,]
    data_test <- data[ d ,]
    # training
    xnam <- paste0("X", candidate [[j]] )
    fmla <- as.formula(paste("Y ~ ", paste(xnam, collapse= "+")))
    model <- lm(fmla ,data_train)
    # testing
    pred.value[i] <- as.numeric(
      (data_test$Y - as.matrix(cbind(1 ,data_test[,xnam])) %*% model$coefficients)^2 )

  }
  CV_value[j] <- sum(pred.value) / dim(combn(1:n ,1))[2]
  #print(xnam)
}
```

the model with smallest CV(1) value

```
## [1] "X1,X2,X3"
```