Notes 3/3/19

Functions and Conditionals (if, then, and else)

* Ex) we want to calculate the variance of X
  + X (height): x1,x2,x3,x4,…,xn
  + Use mean(), sd(), var(0
  + Mean of x: (X1,x2,…,xn)/n
  + Var of x: ((x1-mean of x)2+(x2-mean of x)2…)/n
    - Use n-1 for sample denominator, n for population variance denom

Ex: Pretending the mean function doesn’t exist in R syntax

* X <- 1:5
* N <- length(x)
* Xmean <- sum(x)/n
* Xsvar <- Sum((x -xmean)^2) / (n-1)
* With conditional statements:
  + Sample\_var <- function(x) {

N <- length(x)

Xmean <- sum(x)/n

Xsvar <- Sum((x -xmean)^2) / (n-1)

Print(Xsvar)

}

* + population\_var <- function(x) {

N <- length(x)

Xmean <- sum(x)/n

Xpvar <- Sum((x -xmean)^2) / n

Print(Xpvar)

}

* Can test the functions:
  + Population\_var and sample\_var will return NA for results if you include an NA

IF-THEN-ELSE

* If(condition) {

# do something

} else {

# do something else

}

* + Condition interpolated as a logical vector of true or false
    - If true, do what is in 1st brackets
    - If false, do what is in 2nd brackets (after else)
* Using this on our variance example
  + Make a trait:
    - type <- “sample” or “population”
  + if (type ==”sample”){

#compute sample variance

} else {

# compute population variance

}

* Ex:
  + if (type ==”sample”){

xvar <- Sum((x -xmean)^2) / (n-1)

} else {

Xvar <- Sum((x -xmean)^2) / n

}

* can also add another if after the else
  + if () { } else if() {}
* can do another else clause to cover for all other kinds of inputs
  + Start to think about other potential users
  + How to use all the decisions of those using your function
  + Could end the chain with else {

Print(“invalid input type”)

}

* + - * Better to use stop to make error messages
* Can use the stop() to terminate further execution and warn users with an error message
  + Could end the chain instead with else {

Stop(“invalid input type”)

}

* Good error messages are both and art and a science
  + Want to provide an error message that helps the user fix the problem 🡪 “invalid input type: must be ‘sample’ or ‘population’”
* Ex: Combining it all
  + variance <- function(x,type) {

N <- length(x)

Xmean <- sum(x)/n

If (type ==”sample”){

Xvar <- Sum((x -xmean)^2) / (n-1)

} else if (type===”population”) {

Xvar <- Sum((x -xmean)^2) / n

} else {

Stop(“error message”)

}

Return(xvar)

}

* + If the type is typed incorrectly could use:
    - Type\_match <- match.arg(type, c(“sample”,”population”))
      * Can now partially type “pop” or “samp” to get results
  + To switch between conditions could use:
    - Switch(type\_match,

Sample = Sum((x -xmean)^2) / (n-1),

Population = Sum((x -xmean)^2) / (n)

)

* Combining with match.arg and switch:
  + function(x,type) {

N <- length(x)

Xmean <- sum(x)/n

Type\_match <- match.arg(type, c(“sample”,”population”))

Xvar <- Switch(type\_match,

Sample = Sum((x -xmean)^2) / (n-1),

Population = Sum((x -xmean)^2) / (n)

)

}

Return(xvar)

}