Group 29: Permutation Test

Nikhit Rao (nmrao2)

12/10/2019

Below is code from lecture/homework for permutation test.

```
data("chickwts")
attach(chickwts)
x <- sort(as.vector(weight[feed == "soybean"]))
y <- sort(as.vector(weight[feed == "linseed"]))
n1 <- length(x)
n2 <- length(y)
detach(chickwts)
# Defining the two-sample Cramer Von Mises test statistic as a function
cvmts.test <- function(x,y){
  nx <- length(x)
  ny <- length(y)
  sx <- ecdf(x)(x) - ecdf(y)(x)
  sy <- ecdf(x)(y) - ecdf(y)(y)
  return(nx*ny*(sum(sx^2)+sum(ny^2))/(nx+ny)^2)
}
# Performing the permutation test
R <- 999              #number of replicates
z <- c(x, y)          #pooled sample
K <- 1:(n1+n2)
reps <- numeric(R)    #storage for replicates
t0 <- cvmts.test(x, y)
for (i in 1:R) {
  # generate indices k for the first sample
  k <- sample(K, size = 14, replace = FALSE)
  x1 <- z[k]
  y1 <- z[-k]           #complement of x1
  reps[i] <- cvmts.test(x1, y1)
}
p <- mean(c(t0, reps) >= t0)
p
```

```
## [1] 0.659
```

Below is code using SimDesign for permutation test.

```
library(SimDesign)

data("chickwts")
attach(chickwts)
x <- sort(as.vector(weight[feed == "soybean"]))
y <- sort(as.vector(weight[feed == "linseed"]))
n1 <- length(x)
n2 <- length(y)
detach(chickwts)
# Defining the two-sample Cramer Von Mises test statistic as a function
cvmts.test <- function(x,y){
  nx <- length(x)
```

```
  ny <- length(y)
  sx <- ecdf(x)(x) - ecdf(y)(x)
  sy <- ecdf(x)(y) - ecdf(y)(y)
  return(nx*ny*(sum(sx^2)+sum(ny^2))/(nx+ny)^2)
}

z <- c(x, y)
K <- 1:(n1+n2)
t0 <- cvmts.test(x, y)

Design <- data.frame(R = c(1)) #Don't need design since we don't have any conditions

#Generates data and returns result of cvmts.test
Generate <- function(condition, fixed_objects=FALSE) {
  k <- sample(K, size = 14, replace = FALSE)
  x1 <- z[k]
  y1 <- z[-k]          #complement of x1
  dat <- cvmts.test(x1, y1)
  dat
}

#checks if returned data is >= 0
Analyse <- function(condition, dat, fixed_objects=NULL) {
  ret <- mean(c(dat) >= t0)
  ret
}

#Summarizes the results of the data to get p-value for permutation test
Summarise <- function(condition, results, fixed_objects=NULL) {
  ret <- c(p=mean(results))
  ret
}

final <- runSimulation(design=Design, replications=999, generate=Generate, analyse=Analyse, summarise=Su
```

```
##
##
Design row: 1/1;   Started: Fri Dec 13 06:24:24 2019;   Total elapsed time: 0.00s
```

```
##
## Simulation complete. Total execution time: 5.06s
```

```
final
```

```
##   R         p REPLICATIONS SIM_TIME                COMPLETED      SEED
## 1 1 0.6546547          999    5.06s Fri Dec 13 06:24:29 2019 645518590
```