T test

Peter Nandori

3/14/2022

T test in R

For the Z-test, we can compute the type II error probability analytically. However for the t-test, the type II error probability cannot be given in a closed form because the test statistic contains the sample standard deviation S. What we can do, is to approximate the type II error probability by simulations.

General principle: if we do not know the probability p of a random event E, then we can generate a lot of independent experiments and count the relative frequency of E. This relative frequency will be our estimate of p. This strategy is called *Monte Carlo simulation* and is very common in computational probability.

```
##################################################################################
## tTestSim
##   Monte Carlo simulation of type I and type II error probabilities of T test
##  Input:
##   N: number of iterations
##   n: sample size
##   mean: population mean
##   mean0: mu_0 in the null hypothesis
##   sigma: population standard deviation
##   test: type of alternate hypothesis
##  Output:
##   y: Monte Carlo simulation of type I / type II error probability
##################################################################################
tTestSim<-function(N=1000,n=20,mean=0,
        mean0=1,sigma=1,alpha=.05,seedNum=1,test="lowerTail"){
    ## determine seed for reproducing a result for simulation
    set.seed(seedNum)
    ## result vector
    rejections <- numeric(N)

    for(i in 1:N){
        ## generate data from a normal distribution
        dataSet <- rnorm(n=n,mean=mean,sd=sigma)
        ## compute test statistic
        xbar <- mean(dataSet)
        s <- sd(dataSet)
        testStat <- (xbar-mean0)/(s/sqrt(n))

        if(test=="lowerTail"){
            ## H0: mu=mu0 vs Ha: mu < mu0
            ## RR testStat < -t_{alpha,n-1}
            if(testStat < -qt(p=alpha,df=n-1,lower.tail=FALSE)){
```

```r
                rejections[i] <- 1
            }
        }
    }

    ## Reporting Results
    if(test=="lowerTail"){
        cat("Hypothesis Testing: H0: mu=",mean0," vs. Ha: mu <",mean0,"\n")
        if(mean>=mean0){
            cat("Simulation: Type I Error Probability = ",mean(rejections),"\n")
        }else{
            cat("Simulation: Type II Error Probability",1-mean(rejections)," at mu = ",mean,"\n")
        }
    }
}
```

Let us now test this function:

```r
tTestSim()
```

```
## Hypothesis Testing: H0: mu= 1  vs. Ha: mu < 1
## Simulation: Type II Error Probability 0.004  at mu =  0
```

```r
tTestSim(mean = 0.9)
```

```
## Hypothesis Testing: H0: mu= 1  vs. Ha: mu < 1
## Simulation: Type II Error Probability 0.898  at mu =  0.9
```

As you can see, the function is incomplete as only the lower tail part is implemented. Please implement the other 2 cases as a homework.

```r
##################################################################################
## tTestSim
##   Monte Carlo simulation of type I and type II error probabilities of T test
##  Input:
##   N: number of iterations
##   n: sample size
##   mean: population mean
##   mean0: mu_0 in the null hypothesis
##   sigma: population standard deviation
##   test: type of alternate hypothesis
##  Output:
##   y: Monte Carlo simulation of type I / type II error probability
##################################################################################
tTestSim<-function(N=1000,n=20,mean=0,
        mean0=1,sigma=1,alpha=.05,seedNum=1,test="lowerTail"){
    ## determine seed for reproducing a result for simulation
    set.seed(seedNum)
    ## result vector
    rejections <- numeric(N)

    for(i in 1:N){
```

```r
        ## generate data from a normal distribution
        dataSet <- rnorm(n=n,mean=mean,sd=sigma)
        ## compute test statistic
        xbar <- mean(dataSet)
        s <- sd(dataSet)
        testStat <- (xbar-mean0)/(s/sqrt(n))

        if(test=="lowerTail"){
            ## H0: mu=mu0 vs Ha: mu < mu0
            ## RR testStat < -t_{alpha,n-1}
            if(testStat < -qt(p=alpha,df=n-1,lower.tail=FALSE)){
                rejections[i] <- 1
            }
        }else if(test=="upperTail"){
          ## H0: mu=mu0 vs Ha: mu > mu0
            ## RR testStat > t_{alpha,n-1}
          if(testStat > qt(p=alpha,df=n-1,lower.tail=FALSE)){
                rejections[i] <- 1
            }
        }else if(test=="twoSided"){
          ## H0: mu=mu0 vs Ha: mu =/= mu0
            ## RR |testStat| >= t_{alpha/2,n-1}
          if(abs(testStat) >= qt(p=alpha/2,df=n-1,lower.tail=FALSE)){
                rejections[i] <- 1
            }
        }
    }

    ## Reporting Results
    if(test=="lowerTail"){
        cat("Hypothesis Testing: H0: mu=",mean0," vs. Ha: mu <",mean0,"\n")
        if(mean>=mean0){
            cat("Simulation: Type I Error Probability = ",mean(rejections),"\n")
        }else{
            cat("Simulation: Type II Error Probability",1-mean(rejections)," at mu = ",mean,"\n")
        }
    }else if(test=="upperTail"){
      cat("Hypothesis Testing: H0: mu=",mean0," vs. Ha: mu >",mean0,"\n")
      if(mean<=mean0){
            cat("Simulation: Type I Error Probability = ",mean(rejections),"\n")
        }else{
            cat("Simulation: Type II Error Probability",mean(rejections)," at mu = ",mean,"\n")
        }
    }else if(test=="twoSided"){
      cat("Hypothesis Testing: H0: mu=",mean0," vs. Ha: mu =/=",mean0,"\n")
      if(mean==mean0){
            cat("Simulation: Type I Error Probability = ",mean(rejections),"\n")
        }else{
            cat("Simulation: Type II Error Probability",mean(rejections)," at mu = ",mean,"\n")
        }
    }
}
```

```r
tTestSim(mean = 0.9,test="lowerTail")
```

```
## Hypothesis Testing: H0: mu= 1  vs. Ha: mu < 1
## Simulation: Type II Error Probability 0.898  at mu =  0.9
```

```r
tTestSim(mean = 0.9,test="upperTail")
```

```
## Hypothesis Testing: H0: mu= 1  vs. Ha: mu > 1
## Simulation: Type I Error Probability =  0.014
```

```r
tTestSim(mean = 0.9,test="twoSided")
```

```
## Hypothesis Testing: H0: mu= 1  vs. Ha: mu =/= 1
## Simulation: Type II Error Probability 0.064  at mu =  0.9
```

```r
tTestSim(mean = 1.1,test="lowerTail")
```

```
## Hypothesis Testing: H0: mu= 1  vs. Ha: mu < 1
## Simulation: Type I Error Probability =  0.017
```

```r
tTestSim(mean = 1.1,test="upperTail")
```

```
## Hypothesis Testing: H0: mu= 1  vs. Ha: mu > 1
## Simulation: Type II Error Probability 0.095  at mu =  1.1
```

```r
tTestSim(mean = 1.1,test="twoSided")
```

```
## Hypothesis Testing: H0: mu= 1  vs. Ha: mu =/= 1
## Simulation: Type II Error Probability 0.061  at mu =  1.1
```