Quiz 3 Sequential Probability Ratio Test

Naveen Narayanan Meyyappam

29/10/2019

## Sequential Probability Ratio Test

The Sequential Probability Ratio Test is a type of hypothesis test. In the given problem the null hypothesis (H0) is p<=0.45 and the alternate hypothesis (H1) is p>=0.55.

### Question

In groups, write R code to calculate to test for a Bernoulli random variable p<=.45 vs P>=.55 using an SPRT . At a1=a0=.01. have the alphas as input variables, have n , the input sequence and the accepted hypothesis as output variables. Embed it in a simulation that produces Bernoulli random variables one at a time and tests to rejection (rbinom(1,1,p)) Run it on a sequence of x’s distributed Bernoulli .3, and .56

### Answer

We will write a function based on sprt to test for a bernoulli random variable.

mysprt<-function(val)  
{  
 alpha0=0.01 # Given alpha0 levels  
 alpha1=0.01 #Given alpha1 level  
 A=log(alpha1/(1-alpha0)) #Calculate the threshold levels  
 B=log((1-alpha1)/alpha0)  
 print(A)  
 print(B)  
 lambdan=0 #Initially the log likelihood ratio is set to 0  
 p=0.45 #The given p value  
 n=0 # n is the counter   
 y=NULL #y is the vector that stores the randonmly generated bernoulli trials  
 while((lambdan<=B)||(lambdan>=A)) #Continue the simulation as long as one of the threshold is reached  
 {  
 x=rbinom(1,1,val) #Generating a bernoulli trial  
 y=c(y,x) #Appending to the vector y  
 n=n+1 #incrementing the counter  
 if(x==0) #Calculating the log likelihod ratio based on the outcome  
 {  
 lambdan=lambdan+log(1-p)-log(p)  
 }  
 if(x==1)  
 {  
 lambdan=lambdan+log(p)-log(1-p)  
 }  
 if(lambdan>=B) # Determine if the ratio exeded the threshold levels  
 {  
 print("Accept H0 (p>=0.45)")  
 break # Print the result and exit the loop  
 }  
 if(lambdan<=A)  
 {  
 print("Accept H1 (p<=0.55)")  
 break  
 }  
 }  
 print(n) # Print the counter  
 print(y) # Print the vector y which stored the bernoulli trials  
}

In the above function alpha0 and alpha1 are set to levels of 0.01. Based on these levels we calculate the thresold levels A and B. Next we generate a bernoulli trial (0 or 1). Depending on the outcome we calculate the log likelihood ratio. The cummulative sum of the ratio is maintained in variable ‘lambdan’. Each time a new bernoulli trial is generated, the value of lambdan is updated and it is compared to the threshold levels. If the value exceeds the threshold levels we print the decision and exit the loop. The variable ‘val’ is the probability with which the bernoulli numbers are generated.

Let us call the mysprt() function for the val = 0.3 and val = 0.56 The expected result is that for 0.3, we should accept H0 and for 0.56 we should accept H1. Another expectation is that for 0.56, the function should take more iterations to converge than 0.3

mysprt(0.3)

## [1] -4.59512  
## [1] 4.59512  
## [1] "Accept H0 (p>=0.45)"  
## [1] 59  
## [1] 0 0 1 0 0 1 0 0 0 1 1 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 0 1 0 1 1 0 0  
## [36] 0 1 0 0 0 1 1 0 0 0 1 1 0 0 1 1 0 0 0 0 1 0 0 0

mysprt(0.56)

## [1] -4.59512  
## [1] 4.59512  
## [1] "Accept H1 (p<=0.55)"  
## [1] 275  
## [1] 0 1 1 1 0 0 1 0 1 1 1 0 1 1 0 0 1 1 0 1 1 1 0 1 0 1 0 1 1 0 0 0 0 0 1  
## [36] 0 1 1 1 0 1 0 0 0 0 1 0 1 0 1 1 0 0 1 0 1 1 1 1 0 0 0 0 0 0 1 0 0 0 1  
## [71] 1 0 0 0 1 1 0 1 1 1 0 0 1 0 1 1 1 0 1 1 0 0 1 1 1 0 1 1 1 0 0 1 1 1 1  
## [106] 1 1 1 0 0 1 0 1 1 1 0 0 0 1 1 1 1 0 0 0 0 0 1 1 1 0 0 0 1 0 0 1 1 0 1  
## [141] 1 1 1 1 1 1 0 1 1 1 0 0 1 1 0 1 0 1 1 0 0 1 1 0 1 0 1 0 0 1 0 1 1 0 1  
## [176] 0 0 0 1 1 1 0 1 0 1 0 0 1 1 1 1 1 1 0 1 0 0 1 0 1 0 1 1 0 0 1 1 0 0 1  
## [211] 1 0 1 0 1 0 0 0 1 1 0 0 0 0 0 0 1 1 0 1 0 0 1 0 0 1 1 0 1 1 0 1 0 0 0  
## [246] 0 1 1 1 0 0 0 1 1 0 1 0 1 1 1 1 1 0 1 0 1 0 1 1 1 1 0 1 1 1

### Question

What do you think it would do for .54 ? Try it Why does it give the result you got?

### Answer

For the value of 0.54, ideally we should not accept H0 or H1 as 0.54<0.55 and 0.54>0.45. The above function would keep iterating and try to converge to either of the result. Since 0.54 is close to the region of 0.55. The function would give us a result of accepting H1. It would take a lot of iterations to converge as it is close to the threshold.

The function takes one bernoulli trial at a time and the log likelihood ratio is updated each time. As more and more 0s are get appended to the vector, the value of lambdan keeps decreasing. In the same way as more and more 1s are added to the vector y, the value of lambdan keeps increasing. For the value of lambdan to cross the threshold we need a continous long string of 0s or 1s. In the case of 0.54, the option of getting a 1 or 0 is equally likely. As a result it takes lot of iteration to get a continuous string of 1 or 0. Since the val of 0.54>0.5, the probability of getting a long continuous string of 1 is more than getting a long continous string of zero. Hence the function will converge to accept H1.

mysprt(0.54)

## [1] -4.59512  
## [1] 4.59512  
## [1] "Accept H1 (p<=0.55)"  
## [1] 1019  
## [1] 1 0 0 1 0 1 1 0 1 0 0 0 0 1 1 1 1 0 1 0 1 1 0 1 1 0 0 0 1 0 0 0 1 1  
## [35] 1 1 1 1 1 1 0 0 1 1 0 1 0 0 1 1 0 1 0 1 1 1 0 1 1 0 0 1 0 0 0 0 0 0  
## [69] 1 1 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 1 0 1 0 0 0 0 1 1 0 1 1 0 1 1 1  
## [103] 0 0 0 1 0 0 1 0 1 1 1 1 0 0 1 1 0 1 1 1 0 1 1 0 1 0 1 0 1 0 0 1 1 0  
## [137] 1 0 1 1 1 1 0 0 0 0 0 1 0 1 1 0 1 0 0 0 0 1 0 0 1 0 0 1 0 1 0 1 1 0  
## [171] 0 1 1 0 0 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 0 0 0 0 0 1 0 1 0 1 1 0 0 0  
## [205] 1 1 0 1 1 0 1 1 1 1 1 0 0 1 0 0 0 1 1 1 0 1 0 1 1 1 1 0 1 0 0 0 0 1  
## [239] 1 1 0 0 1 0 1 1 1 0 0 1 1 0 0 1 0 0 0 0 0 1 1 0 1 1 0 1 1 1 1 1 1 0  
## [273] 0 1 1 1 1 1 1 0 0 0 0 1 0 1 1 1 1 1 0 1 0 0 0 1 0 1 0 1 1 0 1 0 0 1  
## [307] 1 0 1 1 1 0 0 1 1 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 1 1 1 0 0 0 1 1 0  
## [341] 1 0 1 0 0 1 1 0 0 0 1 0 0 0 0 1 0 0 1 1 0 1 1 1 1 1 0 0 0 0 0 0 0 1  
## [375] 0 0 1 0 1 0 1 0 1 0 0 1 1 1 0 1 0 0 0 1 0 0 1 1 0 0 1 0 0 0 0 0 1 1  
## [409] 1 1 0 0 1 1 0 1 1 1 1 0 1 0 0 1 1 1 0 1 1 0 0 0 0 1 1 1 0 1 0 0 0 1  
## [443] 1 0 1 0 1 0 1 0 1 1 0 0 1 0 1 1 0 0 0 0 1 1 0 1 0 0 0 0 1 0 0 1 0 0  
## [477] 0 1 1 0 1 1 0 0 0 0 0 0 0 1 0 0 1 1 0 1 0 0 1 1 1 1 0 0 1 0 0 1 0 0  
## [511] 0 1 0 0 1 1 0 1 0 0 0 0 0 0 1 0 0 0 1 1 0 0 1 0 0 1 0 0 0 1 0 0 1 1  
## [545] 0 0 0 0 1 1 1 0 0 0 1 0 0 1 0 1 1 1 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 1  
## [579] 0 0 1 1 0 0 1 1 1 1 0 0 0 0 0 0 1 0 1 1 0 1 1 0 1 1 0 0 1 1 1 0 1 1  
## [613] 0 1 0 0 0 0 0 0 0 1 1 1 0 0 0 1 1 0 1 1 0 0 1 1 1 0 1 1 0 1 1 0 0 0  
## [647] 0 0 0 0 1 1 0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 1 1 0 0 1 0 0 1 0 1 0 1 1  
## [681] 0 1 1 1 0 1 1 1 1 0 0 1 1 1 1 0 1 1 0 1 0 1 1 1 0 0 1 0 1 0 0 1 1 1  
## [715] 0 0 1 1 1 1 1 1 0 1 0 0 1 0 1 0 1 0 0 1 1 1 1 0 0 0 0 0 0 1 1 0 1 0  
## [749] 1 0 0 0 1 1 1 1 0 0 1 1 0 0 1 0 0 1 1 1 0 0 1 1 0 0 0 1 1 1 1 0 0 1  
## [783] 0 0 0 1 1 1 0 1 1 0 0 0 1 0 1 0 1 1 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 1  
## [817] 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 0 0 1 0 1 1 1 1 1 1 0 1 1 0  
## [851] 1 0 0 0 1 0 1 1 0 1 1 0 1 1 0 1 0 1 1 0 0 1 1 1 0 0 0 0 0 0 1 0 0 1  
## [885] 1 1 1 1 0 1 0 0 1 1 0 0 1 1 1 0 0 1 0 1 1 0 0 1 1 0 0 1 1 1 1 0 1 1  
## [919] 1 1 0 1 0 0 1 0 1 0 1 1 1 1 1 0 0 1 0 1 0 1 1 1 0 1 1 0 0 0 1 0 1 0  
## [953] 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 0 0 1 0 1 1 1 0 0 1 1 1 0 0 1 0 1  
## [987] 0 0 0 1 1 0 1 1 0 1 1 0 0 1 0 0 0 1 1 1 1 0 1 1 1 0 1 0 0 1 1 1 1