# Investigate the impact of subgroup mining, a type of p-hacking, on type I error control

Eliza Chai

01/27/2021

## Questions

Generate 50,000 random datasets, each containing data for n=200 patients. To create a single random dataset, for each patient i=1,2,...,200, generate independently:

(a) an outcome Yi drawn from the standard normal distribution;
(b) independent covariates Xi1, Xi2, ..., Xi10 each drawn from a Bernoulli distribution with success probability 0.5.

Observe that, by design, in the setting of this simulation, none of the covariates are truly (i.e., at the population level) associated with the outcome.

For any single random dataset generated, perform the following analysis:

(a) for each of the 10 covariates, record the p-value from a t-test comparing the mean outcome in those with covariate value 1 versus 0;
(b) for each k=1,2,...,10, compute the smallest among the first k p-values obtained and denote it by $s_k$.

Observe that, for each k=1,2,...,10, step (b) is meant to mimic a situation in which the analyst is looking at the marginal association between the outcome and each of k covariates and then reporting the 'most significant' p-value found among the k hypothesis tests conducted.

For each k=1,2,...,10, compute the proportion $q_k$ of times that $s_k < 0.05$ (i.e., a significant association is found for any of the first k covariates) across 50,000 randomly generated datasets.

Produce a plot of q_k versus k and interpret the trend you see in a single sentence.

## Responses

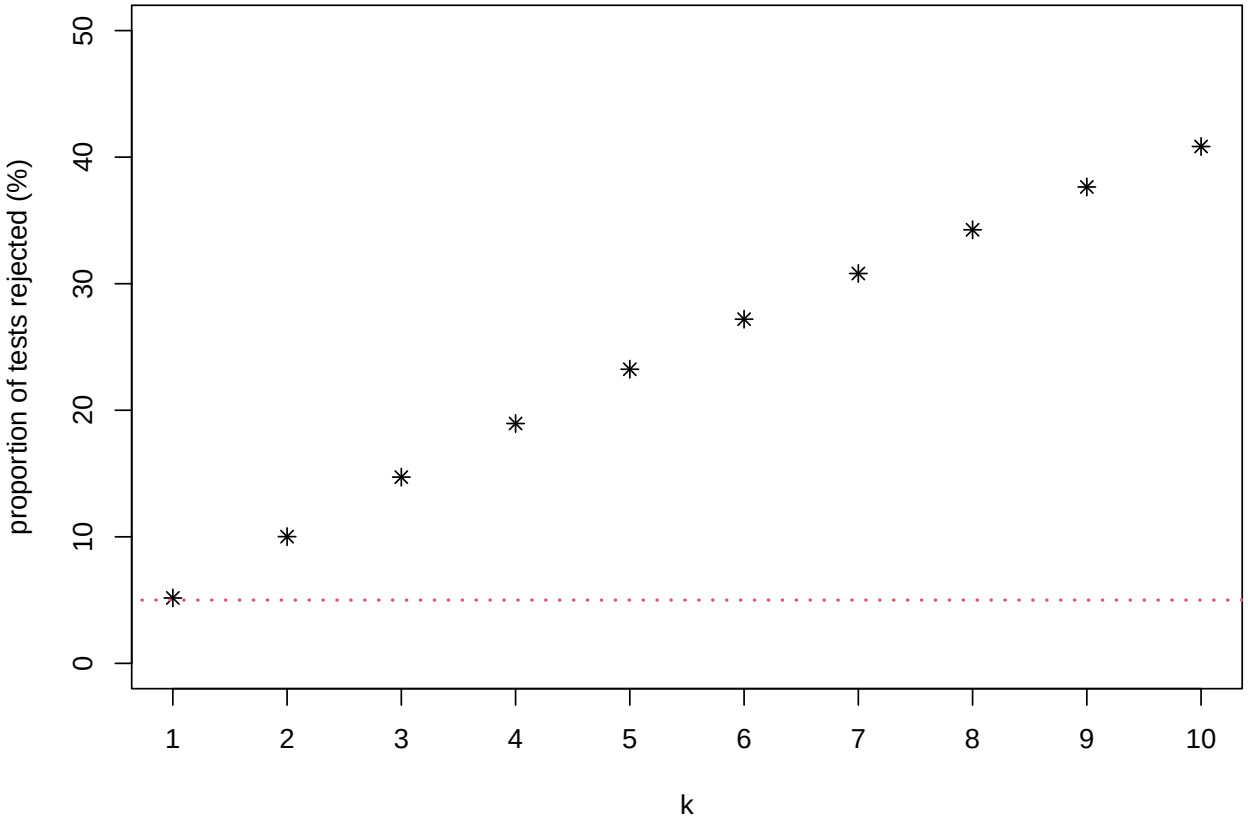


Figure 1 shows an inflated type I error, if we run a two-sample t-test for many k values and report only the most significant p-value.

## Code Appendix

```
### Setting up the packages, options we'll need:
library(knitr)
knitr::opts_chunk$set(echo = FALSE)
### ------------------------------------------------------------
## The function below generates a single random dataset with data for each
## patient i=1,2,...,200, (n = 200 independent patients). The data for each
## patient consist of an outcome Yi(drawn from a standard normal distribution)
## and a independent covariate Xi1, Xi2, ..., Xi10 each drawn from a Bernoulli
## distribution with success probability 0.5.

## All patients are split into two subgroups based on whether their X value is
## <= or > than this cutpoint, and the p-value from a t-test comparing the mean
## outcome in these two subgroups is computed. The smallest p-value found across
## all cutpoints considered is then returned.

onerun = function(){

  # generate data: x~Ber(0,1), y=norm(0,1), x and y independent
  x = replicate(10, rbinom(200, 1, 0.5))
  y = rnorm(200)

  # set cut points: x=0, x=1
  outpvals = NULL
  c = 0.5

  # perform t-test and calculate p-values for each cutpoint considered
  for(i in 1:10){
    y_x0 = y[x[,i]<=c]
    y_x1 = y[x[,i]>c]

    teststat = abs((mean(y_x0)-mean(y_x1))/
                        sqrt(var(y_x0)/length(y_x0)+var(y_x1)/length(y_x1)))
    outpvals[i] = 2*pnorm(teststat,lower.tail=FALSE)

  }

  # output all sk < 0.05
  sk = NULL   #result TRUE or FALSE
  for(k in 1:10){
           sk[k] = any(outpvals[1:k] <0.05)
  }
  return(sk)

}

## run simulation with m=50000, n=200
## compuate the proportion qk of times that sk < 0.05
## set seed for reproducibility

set.seed(505)
qk = rowSums(replicate(50000,onerun()))/50000
```

```
## The code below produces a plot of estimated rejection proportion versus
## the number of cutpoints considered from simulation output.

plot(qk~seq(1:10),ylim=c(0,.5), ylab="proportion of tests rejected (%)",
     xlab="k",xaxt='n',yaxt='n',pch=8)
axis(1,at=seq(1,10,by=1),labels=seq(1,10,by=1))
axis(2,at=seq(0,0.5,by=0.1),labels=seq(0,50,by=10))
lines(c(0,20),rep(0.05,2),col=2,lty=3,lwd=2)
### ------------------------------------------------------------
```