Calculating Power

eCornell

7/1/2021

Scenario: In this example, you will assume that you've run a clinical trial to compare an Old treatment and a New treatment. Both treatments have two possible outcomes — either the treatment works (is successful) or does not work (is not successful).

You want to specify an appropriate cut-off value to help you determine when your results are significant, but you also want to make sure that your chances of a false positive and of a false negative are not too high. In this case, we will assume that our cut-off value is a sample (observed) statistic of $> 16.84\%$, or a p-value of $< 0.05\%$. The null hypothesis is that both the New and Old treatments result in a success rate of 47.6%.

##Step 1: Specify a function to calculate power. The following code creates a user-generated function to calculate power for a particular signal strength, which is the actual difference between the Old and New treatments. You do not need to know how to write a function for this course.

```
# R function to calculate power for a desired signal level
calc_power <- function(delta){

set.seed(1)
outcome = c("Worked", "Did not Work")
nsim = 100000
store_p_diff = rep(0, nsim)

p_new = 27/52
p_old = 22/51
p_all = (27+22)/(52+51)

for (i in 1:nsim){
 result_new = sample(outcome, 52, replace = TRUE, prob = c(p_all+delta, 1-p_all-delta))
 p_new_sim = mean(result_new == "Worked")

 result_old = sample(outcome, 51, replace = TRUE, prob = c(p_all, 1-p_all))
 p_old_sim = mean(result_old == "Worked")

 p_diff = p_new_sim - p_old_sim
 store_p_diff[i] = p_diff
}

return(mean(store_p_diff > 0.1648))
}
```

##Step 2: Use the function to calculate power. Use this function to calculate power for signal levels between 0% and 50% at intervals of 5%.

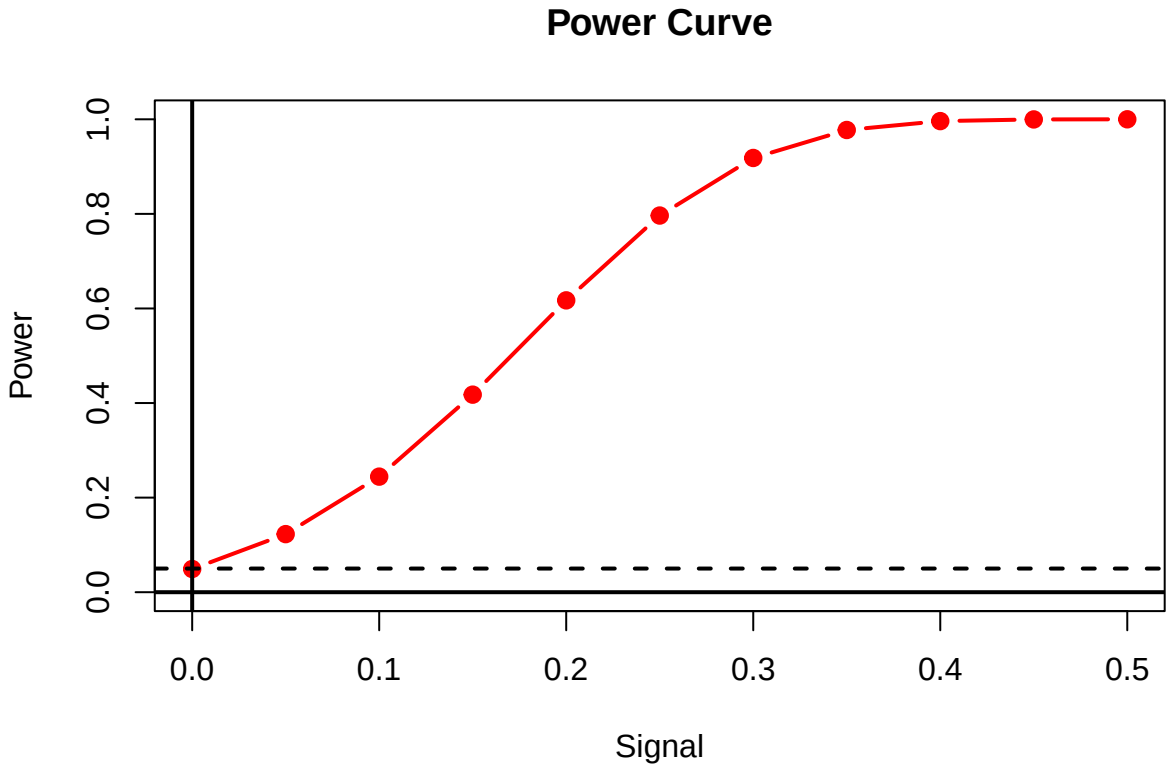
```
# Create a sequence from 0 to .5 by intervals of 0.05
delta_seq = seq(0, 0.5, by=0.05)
# Create a vector of the same length as delta_seq to store your power calculations for different signal
```

```r
power_seq = rep(0, length(delta_seq))

# Use a for loop with the function you created in Step 1 to calculate power for each value in delta_seq

for (i in 1:length(delta_seq)){
  delta = delta_seq[i]
  power_seq[i] = calc_power(delta = delta)
  print(paste('signal = ', delta, '; power = ', round(power_seq[i], 4)))
}
```

```
## [1] "signal =  0 ; power =  0.049"
## [1] "signal =  0.05 ; power =  0.1228"
## [1] "signal =  0.1 ; power =  0.2444"
## [1] "signal =  0.15 ; power =  0.4177"
## [1] "signal =  0.2 ; power =  0.6171"
## [1] "signal =  0.25 ; power =  0.7965"
## [1] "signal =  0.3 ; power =  0.9182"
## [1] "signal =  0.35 ; power =  0.9774"
## [1] "signal =  0.4 ; power =  0.9962"
## [1] "signal =  0.45 ; power =  0.9997"
## [1] "signal =  0.5 ; power =  1"
```

##Step 4: Plot the power curve.

```r
plot(delta_seq, power_seq, type = 'b', pch = 19, lwd = 2, col = "red",
     xlab = 'Signal', ylab = 'Power', main = 'Power Curve',
     ylim = c(0,1), xlim = c(0, 0.5))
abline(h = 0.05, col = "black", lwd = 2, lty = 2)
abline(h = 0, col = "black", lwd = 2, lty = 1)
abline(v = 0, col = "black", lwd = 2, lty = 1)
```



Step 5: Interpret the curve.

Suppose the new treatment actually worked 10% better than the old one (in this case, the signal strength is 10%).

If we use a decision rule that controls chance of FP at 5%, we would have only a 24% chance of picking up that from our small data set of 103 people.