STAT 100B Lab 8

Wesley Chang

8/26/2020

Introducing Simulations

First Approach

1. Simulate probability of getting "heads" exactly 4 times in 10 flips of a fair coin
2. Generate a flip of the coin
3. Create a vector in R with all of the possible outcomes and then randomly select one of those outcomes
4. Use sample function to take a vector of elements (heads or tails) and choose a random sample of size elements

```
coin <- c("heads","tails")

# 1 random sample from the vector coin, with replacement
sample(coin, size = 1, replace = TRUE)
```

```
## [1] "heads"
```

```
# 10 random samples from the vector coin, with replacement
sample(coin, size = 10, replace = TRUE)
```

```
##  [1] "tails" "tails" "heads" "heads" "heads" "heads" "heads" "tails" "tails"
## [10] "heads"
```

Second Approach

1. Describe all possible outcomes: *Two possible outcomes: heads or tails*
2. Connect these outcomes to a random variable: *Let 1 represents "heads" and 0 represent "tails".*
3. Choose a source of random variables: *Use R to generate draws from the appropriate distribution. We use the bernoulli distribution, as there are exactly two outcomes, with each having 50% probability. For repeated flips of the coin, this is a binomial distribution.*
4. Generate a number and note the outcome: *Generate a single flip of the coin using* `rbinom()`.
5. Repeat (4) until the generated numbers show a stable pattern
6. Analyze the simulated outcomes

```r
# generate a single flip of the coin
rbinom(n=1, size=1, prob=0.5)
```

```
## [1] 0
```

```r
# generate the number of 1s (heads) in 10 flips of the coin (1 experimental repetition)
rbinom(n=1, size=10, prob=0.5)
```

```
## [1] 2
```

```r
# generate 10 flips and see each output
rbinom(n=10, size=1, prob=0.5)
```

```
##  [1] 0 1 0 0 0 1 1 0 0 1
```

Exercise 1

Use your simulated data, report the estimate of the probability that exactly 4 in 100 coin flips result in `heads`.

```r
# generate the number of heads in 10 flips of the coin for 1000 experimental repetitions
# store as nheads
nheads <- vector() # empty vector
for(i in 1:1000){
  num <- sum(rbinom(10, size=1, prob=0.5)) # prob of getting a head
  nheads <- c(nheads, num) # add a new observation to the vector
}

# since we want to simulate the probability of getting 4 "heads"
# in 10 flips of a fair coin, we set nheads=4
# true/false values are stored as 1 and 0, so we can simply just
# tally the number of times that something happens by summing
# over all the true/false values
e1 <- sum(nheads == 4)/1000
e1
```

```
## [1] 0.201
```

Answer

The estimate of the probability that exactly 4 in 10 coin flips result in `heads` is 0.201.

Exercise 2

Now, using the Binomial probability formula, calculate the probability of observing 4 heads in 10 coin flips. Is the estimated probability from exercise 1 close to the true probability?

```r
e2 <- choose(10,4)*(0.5**4)*(0.5**(10-4))
e2
```

```
## [1] 0.2050781
```

Answer

Using the Binomial probability formula, I calculated the probability of observing 4 heads in 10 coin flips to be 0.2050781. The estimated probability is relatively close to the calculated probability at 0.201.

Calculating Power

We want to calculate power and sample size for t-tests of two means when the group sizes are different. For example, if we want to test a promising new medical treatment, with only 15 people in our treatment group. The control group consists of people taking some existing drug, a population which is readily available to us. Since this population is readily available, we get a sample size of 150 in our control group. We want to be able to detect a difference of at least 2 units (minimum effect size). For a 0.05 level of confidence, what is the power of this study?

First, we need to simulate the data for the control group. We need to generate data from a normal distribution, so we use `rnorm`. For the standard deviation, we need historical data, and since this control is based on the current treatment standard, we could get this dta from previous research. Let's say this data is 5.

```
# data for control, with size 150 and 0 for difference of mean
control <- rnorm(n=150, mean=0, sd=5)
control
```

```
##   [1]  -4.35126482  -3.06867312   5.04293948   0.88447757   2.33896467
##   [6]  -4.08289364  -7.02711761   7.30838921   4.33749475   6.90631142
##  [11]  -1.32742002   0.97528327  -5.57637449  -0.56392346   4.86014458
##  [16]  -2.82459110  -3.82842707   4.64478062  11.62145679   4.53103564
##  [21]  -1.58590335 -13.31063855   3.35318090   0.43282243  -2.82159598
##  [26]   7.28035338  -3.92111147  -0.36641756   4.59660137  -0.28261699
##  [31]  -3.05445216   1.66523567   0.66405530   3.09786498   2.43287571
##  [36]   2.05678307  -7.71259265   1.74852563   1.10968172  13.41791696
##  [41] -14.15519518   4.76723986   6.76896806   7.04893533  -6.77478179
##  [46]  -3.09978347   4.46896880   3.20445223  -5.15419706  -6.69834006
##  [51]   0.55190032 -11.55405867   3.74798133   2.57083465  -0.34101169
##  [56]  -2.84239198  -4.66205760  -4.29799783   2.84731443  -2.77450243
##  [61]   4.20480551  -0.99173157   4.01762164  -0.49434821   2.31711522
##  [66]  -3.64421088  -8.72724434  -2.02517812   6.98610437  -0.78279515
##  [71]   3.15452621   5.36307941   6.12895169   1.50975080   1.46131246
##  [76]   2.65840850 -10.74852190   3.93712637  -2.17724802  -4.17520998
##  [81]   3.34278747   3.26773498  -3.47641298  -1.38229644  -6.41893039
##  [86]   3.55888858   5.13439890   2.27416016   1.44249212   4.63490647
##  [91]   5.73980140   7.02037695  -0.23168064   3.90305072  -5.15637400
##  [96]  -3.83614876  -3.81180399  -2.55590216   6.96759967  -2.67455093
## [101]   6.80702130  -1.48554176  -1.43094716   0.37552536  -0.60253810
## [106]   9.69426846   3.75995303   1.68624817  -0.63408963   6.37407313
## [111]  -9.98044301  -0.03507678   2.69408896  -2.55659940   4.84062243
## [116]  -0.74623259   7.67891870   2.40582952   2.64151245   2.41811731
## [121]  -7.54864329   2.95665054   6.88660865  -1.13329806   7.07701125
## [126]  -2.04247924   3.78066748  -3.21638494   4.35626009  -1.52632442
## [131]  -2.86238230   5.49324416   6.49800553   0.07900852  -1.13073040
## [136]  -2.74630949  -3.89525978   0.89027416  -6.46967351  -0.60808949
## [141]  -0.27286235  -1.39904924  -3.61633228  -4.95131143  -4.46298224
## [146]  -6.24628616   4.74024933 -14.04082480 -14.34486743   2.89271762
```

```r
# data for treatment, with size 15 and true mean 2
treatment <- rnorm(n=15, mean=2, sd=5)
treatment
```

```
##  [1]  7.27689717 -0.05571067  2.62919177  5.49922909 -0.99050692  1.30285234
##  [7] 10.95599308  0.83960261  0.64093904  0.84962675 -0.62466407  7.95835110
## [13] -4.42670032  0.82950930  6.24601823
```

Exercise 3

Run the t-test, were you able to find a significant difference between the treatment and control groups?

```r
e3 <- t.test(treatment, control)
e3$p.value
```

```
## [1] 0.04725221
```

Answer

With a p-value of 0.0472522, we do not reject the null hypothesis that there is a significant difference in means, and conclude that there is not enough evidence to show that the true difference in means is not equal to 0.

Exercise 4

Now we need to repeat this many times and see how often we are able to accurately reject the null hypothesis with a high probability. This will allow us to quickly calculate the power of the test. We randomly genereate data for the treatment and control groups, run the t-test and save the results, compare the saved p-value to an α=0.05 level of significant, and add `rejects` to `numrejects`. This value starts at 0 and counts the number of times that reject the null hypothesis (If p-value < alpha, we reject and `reject` will be `TRUE`, which is stored as the value 1).

Using the value of `numrejects`*, report the power of this test.*

```r
alpha = 0.05
numrejects <- 0

for(i in 1:1000){
  # generate observations for control and treatment group with mean difference of 2
  control <- rnorm(n=150, mean=0, sd=5)
  treatment <- rnorm(n= 15, mean=2 , sd=5)

  # use t test to compare the two sample means
  ttest <- t.test(treatment, control)

  # reject the equal sample mean hypothesis if p-value is small
  reject <- ttest$p.value < alpha

  # count for rejections
  numrejects <- numrejects + reject
}
```

```
# power is the proportion of times that we reject out of the total number
# of repetitions done by the loop
# 1000 total repetitions
e4 <- (numrejects/1000)*100
e4
```

```
## [1] 28.8
```

Answer

The power of the test, based on `numrejects` is 28.8%.

Chocolate Frogs

Suppose we want to determine the average number of chocolate frogs we would need to buy in order to collect all of the cards we want.

```
# download code to run a custom function to run this simulation
source('https://lgpcappiello.github.io/teaching/stat100b/harrypotter.R')
```

We downloaded a function that runs `n` iterations. It simulates buying chocolate covered frogs with card possibilities `x` with associated probabilities `p`. For each iteration, cards are randomly drawn based on the provided probabilities until all cards have successfully been collected. This is repeated `n` times. The function returns the mean and standard deviation of the number of chocolate frogs purchased before all cards were collected.

Exercise 5

Based on your simulation, on average, how many chocolate frogs would you need to buy in order to collect all of the cards of interest?

```
## tell R which cards we are interested in getting and list a probability for each

# define the set of cards that we are interested in
cards <- c("Dumbledore", "McGonagall", "Grindewald", "Lestrange", "Snape", "Scamander", "Moody", "Flitw
# assign probabilities to each card
# We calculate the probability of "Others" by taking:
#   1 - sum(probabilities of cards of interest)
length(cards)
```

```
## [1] 11
```

```
p1 <- .005
p2 <- .01
p3 <- .04
p4 <- .05
p5 <- .025
p6 <- .02
```

```
p7 <- .03
p8 <- .025
p9 <- .02
p10 <- .02

# create a vector with all probabilities matched with each card
# add in a probability for "Others"
probs.pre <- c(p1,p2,p3,p4,p5,p6,p7,p8,p9,p10)
probs <- c(probs.pre,1-sum(probs.pre))

# now we are ready to run the simulation, we will use MC_geom() to run 10,000 iterations
e5 <- MC_geom(10000, cards, probs)
e5
```

```
## $mean
## [1] 253.6461
##
## $sd
## [1] 184.0359
```

Answer

The simulation returned a mean of 253.6461 and a standard deviation of 184.0358997. This means that you would have buy around 254 frogs to collect all cards of interest.

On Your Own

Question 1

Use simulations to estimate the probability of rolling three 1s in a row on a fair, six-sided die. Hint: use the `rbinom()` *function. Report your estimated probability and compare with the probability you calculated using the Binomial probability formula.*

```
## simulated probability with 1000 flips
n1s <- vector()
for(i in 1:1000){
  num <- sum(rbinom(n=1, size=3, p=1/6))
  n1s <- c(n1s, num)
}

# probability
q1p1 <- (sum(n1s == 3))/1000
q1p1
```

```
## [1] 0.006
```

```
## calculated Binomial probability
p <- 1/6
n = 3
x = 3
q1p2 <- choose(x,n)*((p)**x)*((1-p)**(x-n))
q1p2
```

```
## [1] 0.00462963
```

Answer

The result from the simulated probability of rolling three 1s is 0.006, which is very similar to the calculated Binomial probability of 0.0046296.

Question 2

Suppose some scientists are working on a dietary supplement for pet raccoon. They get 75 raccoons in the treatement group and 100 in the control group. They would like to be able to detect a difference in at least 1 unit. Suppose existing research on raccoon dietary supplements suggests that a reasonable standard deviation estimate is 2.5. For a 0.05 level of confidence, use n=1000 simulations, estimate the power of this study?

```
# treatment n=75, control n = 100
# difference in at least 1 unit
# reasonable sd est is 2.5
# use n=1000 simulations to estimate the power at 0.05 level

alpha1 <- 0.05
numrejects1 <- 0
for(i in 1:1000){
  # generate samples for control and treatment
  control1 <- rnorm(n=100, mean = 0, sd = 2.5)
  treatment1 <- rnorm(n=75, mean= 1, sd = 2.5)

  # run ttest of treatment and control
  ttest <- t.test(treatment1, control1)

  # tally number of results reject at 0.05 level
  reject1 <- ttest$p.value < alpha1
  numrejects1 <- numrejects1 + reject1
}

# calculate power
power <- (numrejects1/1000)*100
power
```

```
## [1] 76.1
```

Answer

The power for this study at a 0.05 level of confidence, based on `numrejects` is 76.1%.

Question 3

Suppose we are buying chocolate frogs and want to collect cards of Luna, McGonagall, Neville, and Harry, with probabilities of 0.024, 0.01, 0.026, and 0.02, respectively. Run a simulation of 100 to determine how many chocolate frogs you would need to buy, on average in order to collect each of these cards. Show your R code below:

```r
# define set of cards
cards <- c("Luna","McGonagall","Neville","Harry")

# define probabilities
p1 <- 0.024
p2 <- 0.01
p3 <- 0.026
p4 <- 0.02
probs <- c(p1,p2,p3,p4)

# run 100 simulations, using MC_geom()
q3 <- MC_geom(100, cards, probs)
q3
```

```
## $mean
## [1] 9.18
##
## $sd
## [1] 4.427599
```

Answer

Based on the above simulation, you would need to buy about 10 cards on average to collect each of the above cards.

Question 4

Use the output from the previous problem to calculate a 95% confidence interval for the average number of chocolate frogs bought in the previous problem. Hint: if you ran `n=100` *iterations, you are working with a sample of 100 observations.*

```r
pe = q3$mean
t = qt(0.975, 99)
sd = q3$sd
n = 100

ci_lower <- pe - t*(sd/sqrt(n))
ci_upper <- pe + t*(sd/sqrt(n))

ci <- c(ci_lower,ci_upper)
ci
```

```
## [1]  8.301468 10.058532
```

Answer

Based on the simulation in Question 3, we calculated a confidence interval of (8.3014682, 10.0585318) with 95% confidence.