Bios 6301: Assignment 3

Jiaheng Yu

Due Tuesday, 26 September, 1:00 PM

50 points total.

Add your name as `author` to the file's metadata section.

Submit a single knitr file (named `homework3.rmd`) by email to marisa.h.blackman@vanderbilt.edu. Place your R code in between the appropriate chunks for each question. Check your output by using the `Knit HTML` button in RStudio.

$5^{n=day}$ points taken off for each day late.

Question 1

15 points

Write a simulation to calculate the power for the following study design. The study has two variables, treatment group and outcome. There are two treatment groups (0, 1) and they should be assigned randomly with equal probability. The outcome should be a random normal variable with a mean of 60 and standard deviation of 20. If a patient is in the treatment group, add 5 to the outcome. 5 is the true treatment effect. Create a linear model for the outcome by the treatment group, and extract the p-value (hint: see assigment1). Test if the p-value is less than or equal to the alpha level, which should be set to 0.05.

Repeat this procedure 1000 times. The power is calculated by finding the percentage of times the p-value is less than or equal to the alpha level. Use the `set.seed` command so that the professor can reproduce your results.

n <- 100 newoutcome <- c() set.seed(n) treatment <- rbinom(n, 1, 0.5) outcome <- rnorm(n, mean=60, sd=20) for (i in 1:n) { + if (treatment[i]==1){ + newoutcome[i] <- outcome[i] + 5} + else + newoutcome[i] <- outcome[i] + } t.test(newoutcome ~ treatment, alternative='two.sided', mu=0)$p.value [1] 0.2832892 # p-value = 0.2832892

check that the treatment effect works

my_data <- data.frame(treatment, outcome, newoutcome)

linear model

model <- lm(newoutcome ~ treatment) get_p <- summary(model)$coefficients[2, 4] # extract the p-value from the linear model

1. Find the power when the sample size is 100 patients. (10 points) > # power = 23.3% > set.seed(100) > > mean(replicate(1000, {

- `treatment <- rbinom(100, 1, 0.5)`
- `outcome <- rnorm(100, mean=60, sd=20)`
- `for (i in 1:100) {`
- `    if (treatment[i]==1){`
- `        outcome[i] <- outcome[i] + 5}`
- `}`
- `t.test(outcome ~ treatment, alternative='two.sided', mu=0)$p.value`
- }) < 0.05) [1] 0.233

1. Find the power when the sample size is 1000 patients. (5 points) > set.seed(1000) > > # how do I get the t.test working? > mean(replicate(1000, {

- `treatment <- rbinom(1000, 1, 0.5)`
- `outcome <- rnorm(1000, mean=60, sd=20)`
- `for (i in 1:1000) {`
- `    if (treatment[i]==1){`
- `        outcome[i] <- outcome[i] + 5}`
- `}`
- `t.test(outcome ~ treatment, alternative='two.sided', mu=0)$p.value`
- }) < 0.05) [1] 0.968

Question 2

14 points

Obtain a copy of the football-values lecture. Save the `2023/proj_wr23.csv` file in your working directory. Read in the data set and remove the first two columns. > hw3_football <- read.csv("~/Desktop/2023:proj_wr23.csv") > hw3_football[, 'PlayerName'] <- NULL > hw3_football[, 'Team'] <- NULL > #summary(hw3_football)

1. Show the correlation matrix of this data set. (4 points) # grab some key aspects of the data set means.football <- colMeans(hw3_football) var.football <- var(hw3_football) # correlation matrix cor(hw3_football)
2. Generate a data set with 30 rows that has a similar correlation structure. Repeat the procedure 1,000 times and return the mean correlation matrix. (10 points) > # install.packages("MASS") > library(MASS) > > # make the simulated data set > football.sim <- mvrnorm(30, mu = means.football, Sigma = var.football) # simulate from a multivariate normal distribution > football.sim <- as.data.frame(football.sim) # turn it into a data.frame > football.sim # show data.frame rec_att rec_yds rec_tds rush_att rush_yds 1 47.019689 562.70166 2.654476660 9.74505948 60.129683 2 13.432289 214.07489 1.160982280 4.09327675 39.160271 3 59.402476 768.70682 4.595392801 2.68604474 7.747860 4 -8.363708 -105.16535 -1.481395729 5.06452970 30.665872 5 2.300904 2.42963 -0.000297193 -4.32015613 -19.681387 6 31.821613 403.91954 1.884018246 -0.61740526 3.694221 7

-54.841137 -555.26678 -3.980473044 -4.88231530 -27.812822 8 26.682430 478.81590 2.332304748 -0.88504133 -4.685517 9 37.361398 561.78742 2.847630796 0.75958757 12.173640 10 15.569569 385.55075 1.494051926 0.89926378 10.257994 11 13.832773 225.85234 1.101648991 5.18537198 35.732497 12 10.593569 108.36496 0.255818717 1.90703880 7.431945 13 -9.041572 17.33781 0.315000803 0.03805022 10.101652 14 -5.344470 -11.01604 -0.987816986 -0.51902651 6.722805 15 20.727094 253.29171 0.499330344 5.57454247 41.882754 16 23.684788 215.37767 0.696849794 7.76612310 54.089721 17 49.106033 646.89432 3.321661558 0.16005008 -4.042323 18 31.218161 267.53498 1.522253151 -1.32230640 -2.274115 19 63.009014 822.49616 4.338747132 2.84413754 19.146941 20 78.953214 1034.15575 6.415041743 0.28602674 -10.859646 21 45.723903 649.96820 3.862107639 -0.52667919 3.027759 22 65.841575 591.78798 3.989316662 0.83538471 13.408075 23 52.425930 707.70196 3.785377406 -2.57067085 -19.998406 24 -30.230655 -289.27285 -2.062605979 1.87017389 16.629199 25 -10.331144 23.59548 -0.981961110 -1.88248863 -14.059510 26 36.843726 456.91987 2.596493738 1.77048357 16.904772 27 77.620673 889.12618 5.164101474 4.10886226 20.426314 28 1.223363 -16.46402 -0.645419465 1.47379933 -6.651670 29 35.209703 555.71929 2.184707669 -6.56740832 -33.536165 30 65.325366 829.67537 4.362153537 5.58386990 25.340368 rush_tds fumbles fpts 1 0.579338580 0.1903983 81.543435 2 0.345375589 0.3356008 33.652570 3 0.144885846 0.6495374 104.754333 4 0.356149024 0.1295287 -14.290634 5 -0.179668942 -0.1572641 -2.711973 6 -0.111168710 0.6930551 50.285430 7 -0.141090692 -0.5013632 -82.152532 8 -0.256710563 0.3268014 59.269364 9 -0.007662688 0.2331796 73.911369 10 0.100393662 -0.0680226 49.478812 11 0.213209414 0.3211882 33.615212 12 -0.008988448 0.4004545 12.144899 13 -0.077955403 -0.2266228 4.399096 14 -0.023869628 0.3582638 -6.857409 15 0.302834569 0.5545972 33.001298 16 0.630327456 0.3044209 34.200871 17 -0.092238907 0.9913071 81.841979 18 -0.109382616 -0.2512532 35.383666 19 0.303388692 0.8042011 110.391317 20 -0.083360890 0.3130013 139.214452 21 0.077853562 0.7007574 87.394561 22 -0.005662305 0.0924185 83.923003 23 -0.080831158 0.8641679 88.899425 24 0.066957293 -0.2517512 -38.694071 25 -0.208596531 0.2203736 -6.191311 26 0.067489968 0.0434103 63.544344 27 0.055035165 0.8876423 120.561321 28 -0.261626476 0.2933990 -8.038782 29 -0.322120162 0.2326191 62.890774 30 0.164797451 0.9024604 110.315801 > > # then loop it to figure out the average matrix > final_football <- 0 # start at 0 > for (i in 1:1000){

- `football.sim <- mvrnorm(30, mu = means.football, Sigma = var.football) # same as above`
- `final_football <- final_football + cor(football.sim)/1000`
- } > final_football # mean correlation matrix rec_att rec_yds rec_tds rush_att rush_yds rec_att 1.0000000 0.9739995 0.9615175 0.3437063 0.3241331 rec_yds 0.9739995 1.0000000 0.9682236 0.3245472 0.3046831 rec_tds 0.9615175 0.9682236 1.0000000 0.2632733 0.2523576 rush_att 0.3437063 0.3245472 0.2632733 1.0000000 0.9515617 rush_yds 0.3241331 0.3046831 0.2523576 0.9515617 1.0000000 rush_tds 0.2661009 0.2590047 0.2215764 0.8305639 0.8956625 fumbles 0.6417622 0.6401381 0.6050654 0.3934740 0.3679475 fpts 0.9766487 0.9960438 0.9784909 0.3670954 0.3541480 rush_tds fumbles fpts rec_att 0.2661009 0.6417622 0.9766487 rec_yds 0.2590047 0.6401381 0.9960438 rec_tds 0.2215764 0.6050654 0.9784909 rush_att 0.8305639 0.3934740 0.3670954 rush_yds 0.8956625 0.3679475 0.3541480 rush_tds 1.0000000 0.3083473 0.3114634 fumbles 0.3083473 1.0000000 0.6379099 fpts 0.3114634 0.6379099 1.0000000

Question 3

21 points

Here's some code:

```
nDist <- function(n = 100) {
    df <- 10
    prob <- 1/3
    shape <- 1
    size <- 16
```

```
    list(
        beta = rbeta(n, shape1 = 5, shape2 = 45),
        binomial = rbinom(n, size, prob),
        chisquared = rchisq(n, df),
        exponential = rexp(n),
        f = rf(n, df1 = 11, df2 = 17),
        gamma = rgamma(n, shape),
        geometric = rgeom(n, prob),
        hypergeometric = rhyper(n, m = 50, n = 100, k = 8),
        lognormal = rlnorm(n),
        negbinomial = rnbinom(n, size, prob),
        normal = rnorm(n),
        poisson = rpois(n, lambda = 25),
        t = rt(n, df),
        uniform = runif(n),
        weibull = rweibull(n, shape)
    )
}
```

1. What does this do? (3 points)

```
round(sapply(nDist(500), mean), 2)
```

```
##           beta       binomial     chisquared    exponential              f
##           0.10           5.33          10.00           0.97           1.17
##          gamma      geometric hypergeometric      lognormal     negbinomial
##           0.99           1.86           2.58           1.69          32.04
##         normal        poisson              t        uniform        weibull
##          -0.04          24.69          -0.03           0.47           0.94
```

```
The expression above randomly samples 500 values (instead of the default 100 listed) from each of t
```

2. What about this? (3 points)

```
sort(apply(replicate(20, round(sapply(nDist(10000), mean), 2)), 1, sd))
```

```
##           beta        uniform              f          gamma        weibull
##    0.000000000    0.002236068    0.006386664    0.008335088    0.008870412
##    exponential              t         normal hypergeometric      lognormal
##    0.009119095    0.009665457    0.010500627    0.014608937    0.017137217
##       binomial      geometric     chisquared        poisson    negbinomial
##    0.020844032    0.023004576    0.041836147    0.051018572    0.096478795
```

```
The expression above randomly samples 10000 values from each of the given distributions, then calcu
```

In the output above, a small value would indicate that `N=10,000` would provide a sufficent sample size as to estimate the mean of the distribution. Let's say that a value *less than 0.02* is "close enough".

3. For each distribution, estimate the sample size required to simulate the distribution's mean. (15 points)

Don't worry about being exact. It should already be clear that N < 10,000 for many of the distributions. You don't have to show your work. Put your answer to the right of the vertical bars (|) below.

distribution	N
beta	10
binomial	10000
chisquared	60000
exponential	3000
f	2000
gamma	4000
geometric	11000
hypergeometric	5000
lognormal	10000
negbinomial	250000
normal	3000
poisson	70000
t	6000
uniform	300
weibull	3000