Bios 6301: Assignment 3

Catherine Greene

Due Tuesday, 27 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 tianyi.sun@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 assignment1). 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.

```r
# scratchwork, I'm testing the code here

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
```

```r
# p-value = 0.2832892

# check that the treatment effect works
my_data <- data.frame(treatment, outcome, newoutcome)
```

```r
# 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)

```r
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
```

```r
# power = 23.3%
```

1. Find the power when the sample size is 1000 patients. (5 points)

```r
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
```

```r
# power = 96.8%
```

Question 2

14 points

Obtain a copy of the football-values lecture. Save the `2021/proj_wr21.csv` file in your working directory. Read in the data set and remove the first two columns.

```r
hw3_football <- read.csv("/Users/KatietheWise/Desktop/2022_Fall/StatComp/Homework/proj_wr21.csv")
hw3_football[, 'PlayerName'] <- NULL
hw3_football[, 'Team'] <- NULL

#summary(hw3_football)
```

1. Show the correlation matrix of this data set. (4 points)

```r
# grab some key aspects of the data set
means.football <- colMeans(hw3_football)
```

```r
var.football <- var(hw3_football)
# correlation matrix
cor(hw3_football)
```

```
##            rec_att   rec_yds   rec_tds  rush_att  rush_yds  rush_tds   fumbles
## rec_att  1.0000000 0.9889836 0.9620513 0.2242480 0.2810831 0.2312038 0.6423627
## rec_yds  0.9889836 1.0000000 0.9720400 0.2062038 0.2614786 0.2115013 0.6487247
## rec_tds  0.9620513 0.9720400 1.0000000 0.2004448 0.2540571 0.2151580 0.6021914
## rush_att 0.2242480 0.2062038 0.2004448 1.0000000 0.9779751 0.9308512 0.1446322
## rush_yds 0.2810831 0.2614786 0.2540571 0.9779751 1.0000000 0.9298581 0.1761579
## rush_tds 0.2312038 0.2115013 0.2151580 0.9308512 0.9298581 1.0000000 0.1809564
## fumbles  0.6423627 0.6487247 0.6021914 0.1446322 0.1761579 0.1809564 1.0000000
## fpts     0.9863078 0.9957911 0.9842850 0.2610623 0.3162080 0.2677821 0.6288445
##               fpts
## rec_att  0.9863078
## rec_yds  0.9957911
## rec_tds  0.9842850
## rush_att 0.2610623
## rush_yds 0.3162080
## rush_tds 0.2677821
## fumbles  0.6288445
## fpts     1.0000000
```

1. 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)

```r
# install.packages("MASS")
library(MASS)

# make the simulated data set
football.sim <- mvrnorm(30, mu = means.football, Sigma = var.football) # simulate from a multivariate n
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    rush_tds
## 1   50.8611018  616.120828  3.91338990 10.6260210  57.7096265  0.39986047
## 2   17.1498659  230.942725  1.73039868  8.4483979  36.7295818  0.31742715
## 3   65.8386871  842.931127  5.51290283 -0.7692712   5.0297429 -0.08793029
## 4   -8.4154621 -120.909634 -1.18067427  5.7045601  30.3638371  0.13567376
## 5    2.6042549   -2.333362 -0.07229731 -3.2165831 -20.5624214 -0.15757974
## 6   34.8273806  440.630478  1.85990253  1.1977151   1.5195940  0.04669255
## 7  -54.6006594 -618.415977 -3.92693120 -4.7833807 -27.3083115 -0.27707120
## 8   33.5206888  522.500392  2.83189520 -1.4660311  -7.3325901  0.04264181
## 9   43.7466387  614.321261  3.65742804  2.4408515   9.0437139  0.14071042
## 10  23.1764271  419.375722  2.85866308  1.8062710   7.4025684  0.07454867
## 11  17.5641299  243.944952  1.58274629  6.6160801  33.8275589  0.32149075
## 12  11.2448129  114.720067  0.02367527  0.6392826   6.8356391  0.01271085
## 13  -4.4202241   13.315558  0.72201370  2.4327181   8.1474403  0.23257757
## 14  -3.8578749  -17.391152 -1.01857083  2.4953780   5.5060134  0.06561255
## 15  22.4070069  274.675298  0.61899131  8.7019151  40.1607554  0.29189368
## 16  24.0761590  233.177779  1.28635886 10.6073364  52.6529863  0.28621559
## 17  54.1510789  708.574660  3.54186615 -2.0357274  -6.3746684 -0.13585574
## 18  30.8397824  290.732735  1.60679038 -0.1851869  -3.5281622  0.02604521
## 19  69.5774652  902.318398  5.40753197  2.8630256  15.7773965 -0.02970979
## 20  87.5870722 1135.608437  7.61147620 -5.1021976 -14.2536185 -0.18864388
```

```
## 21  52.6311105  711.606703  4.60859847  0.2992625  -0.3126875 -0.06275780
## 22  65.2682062  649.038185  3.94693781  2.6997500  11.3647816  0.15525067
## 23  58.3834380  775.485275  4.22862932 -5.0822818 -22.6245698 -0.35465312
## 24 -28.8903096 -324.639407 -1.89110897  3.6102106  16.1280922  0.20052655
## 25  -6.5753645   20.272803 -0.72191461 -3.0058601 -15.0383299 -0.12613240
## 26  40.6740856  499.094281  3.33665256  3.1172151  14.4643204  0.18096278
## 27  81.8828246  976.447846  5.50487051  2.2945483  17.6608614  0.11249754
## 28   0.5931707  -22.938245 -1.19921940 -3.1923076  -6.1276053 -0.04949808
## 29  41.1709516  607.484408  2.83036688 -5.9179237 -36.3602634 -0.31398063
## 30  71.1220060  910.443752  4.84646370  2.8629541  22.6512319  0.07646358
##         fumbles       fpts
## 1   1.076955363  90.875335
## 2  -0.127427190  39.368664
## 3   0.416483498 116.455177
## 4  -0.037569348 -15.418661
## 5   0.189274739  -3.846985
## 6  -0.090252029  55.638761
## 7  -0.330045450 -89.008437
## 8   0.328707882  68.152605
## 9   0.682085767  83.880421
## 10  0.740482004  58.679001
## 11 -0.032220997  39.125154
## 12 -0.183060277  12.840250
## 13  0.009003161   8.155585
## 14 -0.298401644  -6.558432
## 15  0.094007514  36.998461
## 16  0.334247703  37.379575
## 17 -0.087831493  90.663155
## 18  0.986766920  36.666325
## 19  0.500422692 123.020138
## 20  1.213781611 154.599065
## 21  0.027320822  98.445530
## 22  1.078206028  88.755746
## 23  0.106999092  98.568583
## 24 -0.209191582 -40.538468
## 25 -0.127212044  -4.632452
## 26  0.769653854  70.715629
## 27  0.544155377 131.936824
## 28 -0.183137227 -10.275962
## 29  0.821501813  70.649578
## 30  0.466204897 122.365937
```

```r
# 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  rush_tds   fumbles
## rec_att   1.0000000 0.9885136 0.9601943 0.2237116 0.2794244 0.2341890 0.6379920
## rec_yds   0.9885136 1.0000000 0.9707968 0.2066551 0.2606589 0.2156990 0.6421565
## rec_tds   0.9601943 0.9707968 1.0000000 0.1994294 0.2522234 0.2180419 0.5958923
## rush_att  0.2237116 0.2066551 0.1994294 1.0000000 0.9771547 0.9276353 0.1390555
```

```
## rush_yds 0.2794244 0.2606589 0.2522234 0.9771547 1.0000000 0.9260400 0.1692499
## rush_tds 0.2341890 0.2156990 0.2180419 0.9276353 0.9260400 1.0000000 0.1771592
## fumbles  0.6379920 0.6421565 0.5958923 0.1390555 0.1692499 0.1771592 1.0000000
## fpts     0.9855808 0.9956094 0.9835312 0.2604771 0.3145548 0.2709661 0.6219571
##               fpts
## rec_att  0.9855808
## rec_yds  0.9956094
## rec_tds  0.9835312
## rush_att 0.2604771
## rush_yds 0.3145548
## rush_tds 0.2709661
## fumbles  0.6219571
## fpts     1.0000000
```

Question 3

21 points

Here's some code:

```r
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)

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

```
##           beta       binomial     chisquared    exponential              f
##           0.10           5.34           9.83           1.02           1.14
##          gamma      geometric hypergeometric      lognormal    negbinomial
##           0.99           1.90           2.62           1.74          32.04
##         normal        poisson              t        uniform        weibull
##           0.03          25.40           0.09           0.50           1.01
```

```
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         normal    exponential
##    0.000000000    0.000000000    0.006708204    0.008645047    0.009119095
##              t          gamma        weibull hypergeometric       binomial
##    0.009679060    0.010052494    0.010990426    0.012139540    0.020589982
##      lognormal      geometric        poisson     chisquared     negbinomial
##    0.020641042    0.024899799    0.054335701    0.061567250    0.109037656
```

```
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.

```
I just used approximate round numbers.
```

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