Resampling Methods

Lecture 5 Handout

Statistics 139

Topics

- Permutation Testing
- Bootstrap Estimation

The material in this lab corresponds to the Lecture 5 Notes.

In this lab we will continue to analyze the Trump tweets data set (all of President Trump's original tweets in 2020-21 from Nov 1, 2020 until he was banned from Twitter), in the data file 'trumptweets.csv', to answer the question:

1. Are the number of retweets associated with the inclusion of the word "democrat" in tweets?

The following variables are measured:

- `date`: date of the tweet, in month/day/year format
- `time`: date of the tweet, in 24 hour time (aka, military time)
- `retweets`: the number of retweets of Trump's original tweet.
- `favorites`: the number of times the tweet was 'favorited'.
- `isRetweet`: a logical variable (with only `FALSE` in this data set).
- `id`: a unique ID from Twitter for each tweet.
- `text`: the actual text of the tweet.

```
# install.packages( c("coin","perm","boot") )
# library(coin)
library(perm)
library(boot)
```

Concept Checks

a) When bootstrapping, why do we sample with replacement? Why do we sample the original sample size, n, each time?

b) How could we determine if bootstrapped CIs *improve* things in comparison to the standard methods? What was wrong in the first place?

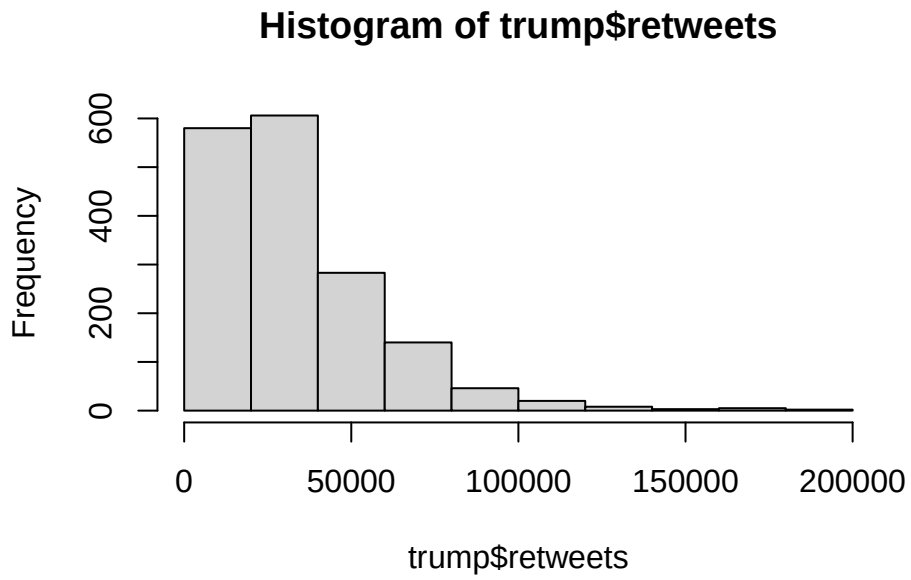
Question 1: The Democrats: A Trump Tweet Favorite

a) Look at the histogram of `retweets` and comment on the appropriateness of t based methods. Explore transformations to use to improve the situation (do not get too exotic), and comment on what you find.

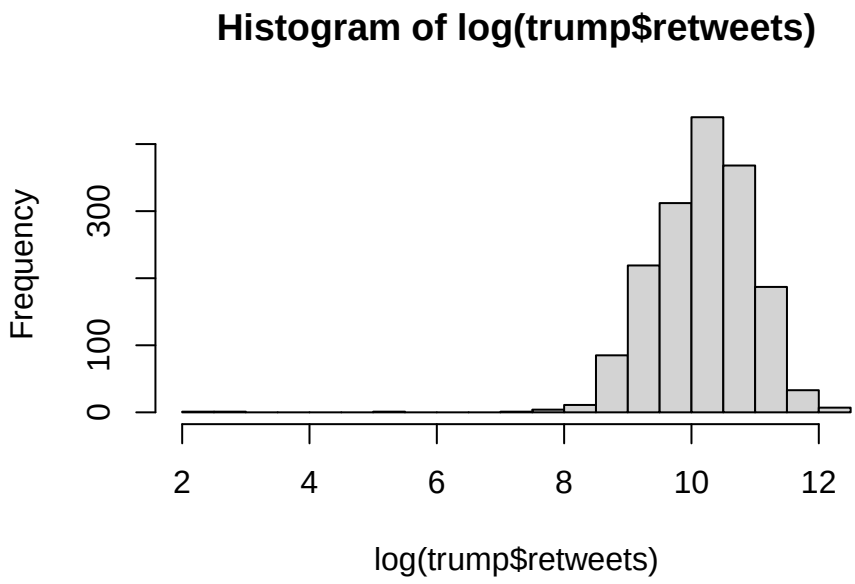
Very right skewed

```
# look at some histograms
trump = read.csv('data/trumptweets.csv')

hist(trump$retweets)
```




```
hist(log(trump$retweets), breaks=30)
```



b) Use the following code (ignore any warnings for now) to add a variable `dem` to the data frame which indicates whether Trump mentioned word "democrat" in the tweet.

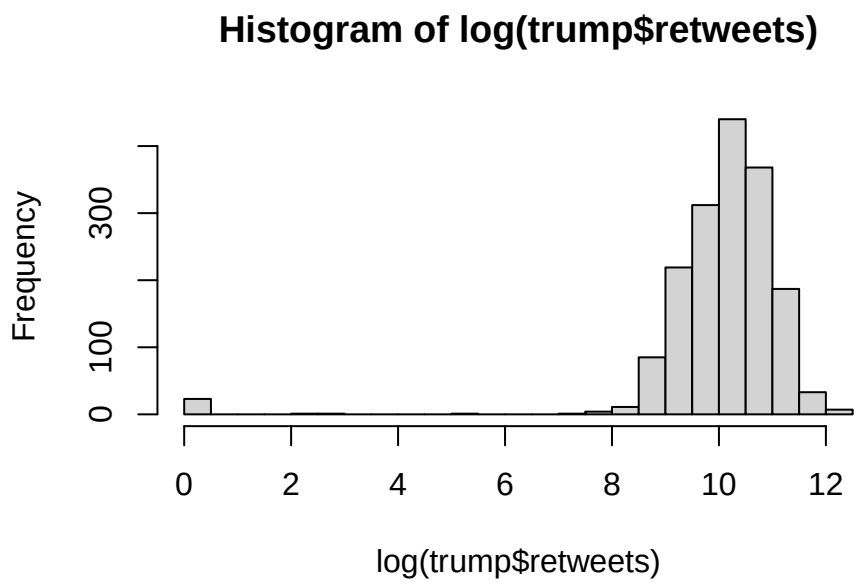
```
dem.indices = grep("democrat",trump$text,ignore.case=T,useBytes = TRUE)
trump$dem = rep(0,nrow(trump))
trump$dem[dem.indices] = 1
table(trump$dem)
```

```
##
##    0    1
## 1620   73
```

c) Perform 3 tests to see if Trump tweets including the word 'democrat' changes the popularity of the tweets (based on retweets): (i) a t based method with the log-transformed response, (ii) a method based on ranks, and (iii) a permutation test (performed 'manually').

```
# t, ranksum, and permutation test
# don't forget to set.seed
set.seed(139)

# recode entries with 0 retweet
trump[trump$retweets <= 0, ] <- 1
hist(log(trump$retweets), breaks=30)
```




```
# t
t.test(log(trump$retweets) ~ trump$dem)
```

```
##
##  Welch Two Sample t-test
##
## data:  log(trump$retweets) by trump$dem
## t = 5.201, df = 94.349, p-value = 1.149e-06
## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to
## 95 percent confidence interval:
##  1.476350 3.299476
## sample estimates:
## mean in group 0 mean in group 1
##       10.169751        7.781838
```

```
# ranksum
wilcox.test(trump$retweets ~ trump$dem)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  trump$retweets by trump$dem
## W = 89039, p-value = 0.004553
## alternative hypothesis: true location shift is not equal to 0
```

```r
# permutation
library(perm)
permTS(retweets~dem, data=trump)
```

```
##
##  Permutation Test using Asymptotic Approximation
##
## data:  retweets by dem
## Z = 2.2534, p-value = 0.02423
## alternative hypothesis: true mean dem=0 - mean dem=1 is not equal to 0
## sample estimates:
## mean dem=0 - mean dem=1
##                 5973.47
```

d) Provide a bootstrapped confidence interval to estimate the difference in mean number of 'retweets' when mentioning democrats vs. not. Interpret this interval.

```r
# bootstrap interval
set.seed(139)
nsims = 10000
boot.diff = c(NA, nsims)

for(i in 1:nsims){

  boot1 = sample(trump$retweets[trump$dem == 1], replace=T)
  boot2 = sample(trump$retweets[trump$dem == 0], replace=T)

  boot.diff[i] = mean(boot1)-mean(boot2)
}

quantile(boot.diff, c(0.025, 0.975))
```

```
##        2.5%       97.5%
## -11209.0595   -620.2043
```

e) Confirm your results with the packaged `boot` and `perm` packages in the R chunk below:

```r
 library(boot)
 library(perm)

# function to perform the bootstrap

mean.diff <- function(data,indices){
  d <- data[indices,] # allows boot to select sample
  return(diff(by(d$retweets,d$dem,mean)))
}

set.seed(12345)
results <- boot(data=trump, statistic = mean.diff, R=1000)
boot.ci(results,type=c("norm","basic"))

permTS(retweets~dem, data=trump, exact = T)
```

Question 2: coverage probability simulations The code below uses a `for` loop to repeatedly (`nsims = 100`) create samples of size $n_1 = n_2 = 10$ for 2 different groups of Y_1, Y_2 where $Y_1 \sim Expo(\lambda_1 = 1)$ and $Y_2 \sim Expo(\lambda_2 = 2)$. It then calculates 1000 confidence intervals from two approaches: t-based and bootstrap based.

```
# in case you forgot them earlier
# library(boot) # library(perm)
starttime = Sys.time()
set.seed(139)
nsims = 100
nboots = 500
mean.diff.sim <- function(data,indices){
  d <- data[indices,] # allows boot to select sample
  return(-diff(by(d$y,d$x,mean)))
}

# set up the parameters for the data generating process
lambda1 = 1
lambda2 = 2
n1 = 10
n2 = 10

# create blank storage matrices for results (ci bounds)
t.cis = matrix(NA,ncol=2,nrow=nsims)
boot.cis = matrix(NA,ncol=2,nrow=nsims)

#the for loop does the bulk of the work
for(i in 1:nsims){
  # generate the data
  y1 = rexp(n1,lambda1)
  y2 = rexp(n2,lambda2)
  y = c(y1,y2)
  x = c(rep(1,n1),rep(2,n2))
  data = data.frame(y=y,x=x)

  # calculate cis
  ttest = t.test(y1,y2)
  t.cis[i,] = ttest$conf.int
  boots <- boot(data=data, statistic=mean.diff.sim,R=500)
  boot.cis[i,] = boot.ci(boots, type = c("basic"))$basic[4:5]
}
endtime = Sys.time()
```

a) Determine $E(Y_1)$ and $E(Y_2)$.

- $E(Y_1) = 1$
- $E(Y_2) = \frac{1}{2}$

b) Determine the empirical coverage probability of each of the 3 methods, and compare their average widths.

```r
# Determine the coverage probabilities of each of the 3 methods
mu.diff = 1/lambda1-1/lambda2
```

c) How long did the simulation take (running time, in seconds)?

```r
# run time
```

d) Rerun the simulation so that the sample sizes are now $n_1 = n_2 = 30$ and $n_1 = n_2 = 100$? How have things changed?