Resampling and Polling

Prof. Josh Clinton

9/29/2021

Motivating questions:

- What is *random sampling* and why is random sampling so powerful as a method of collecting data?
- Why can the opinions of 1000 people be used to measure the public opinion of 350 million people?
- If my data is a random sample, how many observations do I need to calculate an accurate average?
- How much better is a random sample of size 1000 than a sample of size 100? In what ways?

Quantifying uncertainty is critical for science

- Without knowing how much your results may change it is hard to describe, predict, or understand relationships.
- Lots of ways that uncertainty/error can arise! Data, Modelling, User Error, ...
- Focus on on "best case" - what if our data is a random sample from the population of interest, how much would the results change if we did everything we just did again? Variation due to *random sampling*!
- NOTE: Very rarely are we in this condition (hence the need for statistical modelling).

Some Simple Sampling

The function `sample(X, Y, replace = TRUE, prob = P)` will sample `Y` units from a vector `X` with or without replacement (`replace = TRUE`) or {`replace = FALSE`) using a vector of probability {`P`} (the default is equal probability).

```
Z <- seq(1,8)

## Randomly draw 8 samples from Z, with replacement
sample(Z, 8, replace = TRUE)
## [1] 1 5 1 1 2 4 2 2
## Randomly draw 8 samples from Z, without replacement
sample(Z, 8, replace = FALSE)
## [1] 1 7 4 8 5 2 3 6
## Randomly draw 8 samples from Z, with replacement
sample(Z, 8, replace = TRUE, prob = Z/sum(Z))
## [1] 6 3 8 1 2 8 6 7
```

Calculating Probability through Simulation

- Probability can be thought of as the "limit'' of repeated identical experiments.
- Use loops to repeat an experiment and calculate the probability of certain events.

e.g., Birthday Problem

- How many people do you need for the probability that at least two people have the same birthday exceeds 0.5?
- `unique`: the number of unique values in a vector

```
values <- c(1,2,3,3)
length(values)
```

```
## [1] 4
```

- `length`: the length of a vector

```
unique(values)
```

```
## [1] 1 2 3
```

Solving via simulation

```
sims <- 10000 ## number of simulations
bday <- 1:365 ## possible birthdays
answer <- NULL ## holder for our answers

for (k in 1:25) {
  count <- 0  ## counter
  for (i in 1:sims) {
    class <- sample(bday, k, replace = TRUE) # sampling with replacement
    if (length(unique(class)) < length(class)) {
      count <- count + 1
    }
  }
  ## printing the estimate
  cat("The estimated probability for", k,"people is:", count/sims, "\n")
  answer[k] <- count/sims # store the answers
}
```

```
## The estimated probability for 1 people is: 0
## The estimated probability for 2 people is: 0.0023
## The estimated probability for 3 people is: 0.0078
## The estimated probability for 4 people is: 0.0144
## The estimated probability for 5 people is: 0.0271
## The estimated probability for 6 people is: 0.0387
## The estimated probability for 7 people is: 0.0575
## The estimated probability for 8 people is: 0.0766
## The estimated probability for 9 people is: 0.0947
## The estimated probability for 10 people is: 0.1206
## The estimated probability for 11 people is: 0.1381
## The estimated probability for 12 people is: 0.1761
## The estimated probability for 13 people is: 0.1945
## The estimated probability for 14 people is: 0.2171
## The estimated probability for 15 people is: 0.26
```

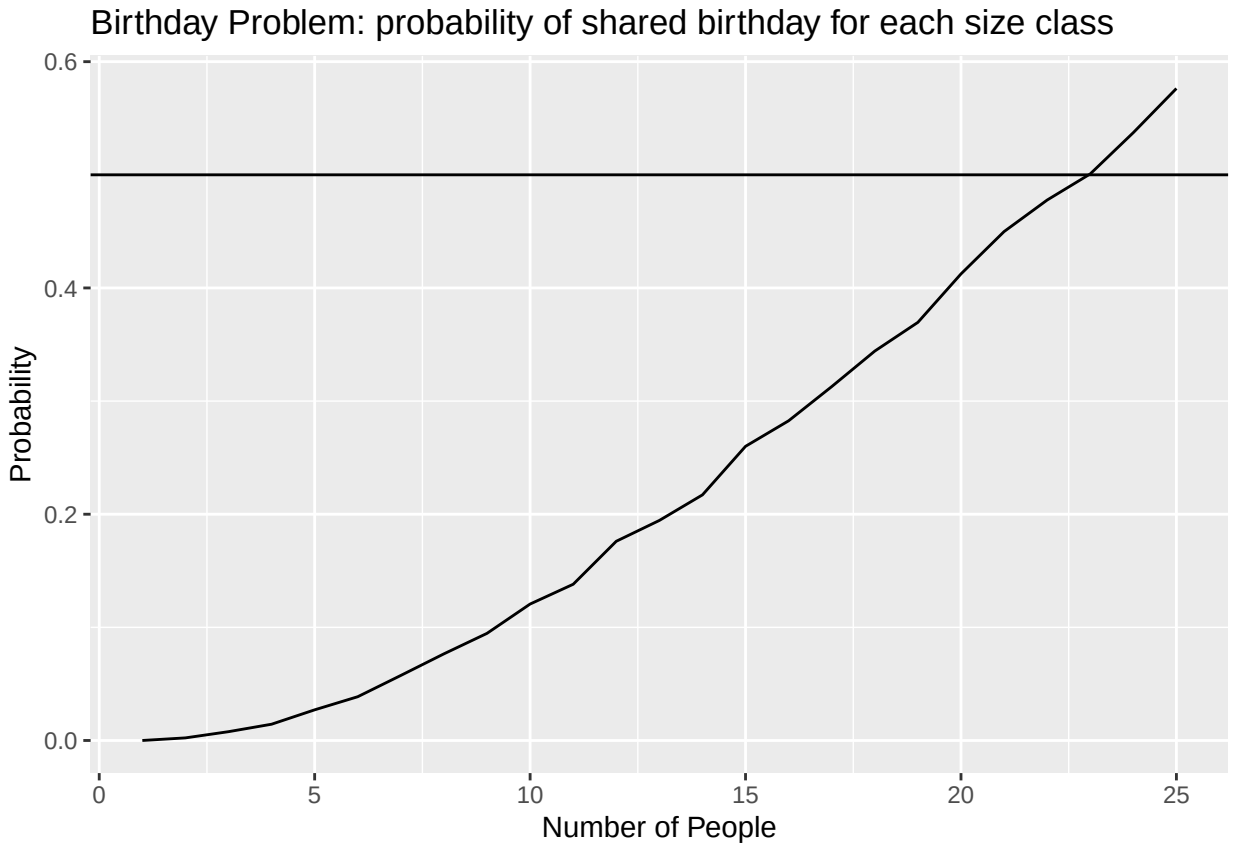
```
## The estimated probability for 16 people is: 0.2826
## The estimated probability for 17 people is: 0.3128
## The estimated probability for 18 people is: 0.3442
## The estimated probability for 19 people is: 0.3696
## The estimated probability for 20 people is: 0.4124
## The estimated probability for 21 people is: 0.4498
## The estimated probability for 22 people is: 0.4777
## The estimated probability for 23 people is: 0.5008
## The estimated probability for 24 people is: 0.5373
## The estimated probability for 25 people is: 0.5763
```

In graphical terms:

```
dat <- bind_cols(npeople=seq(1,25),answer)
```

```
## New names:
## * `` -> ...2
```

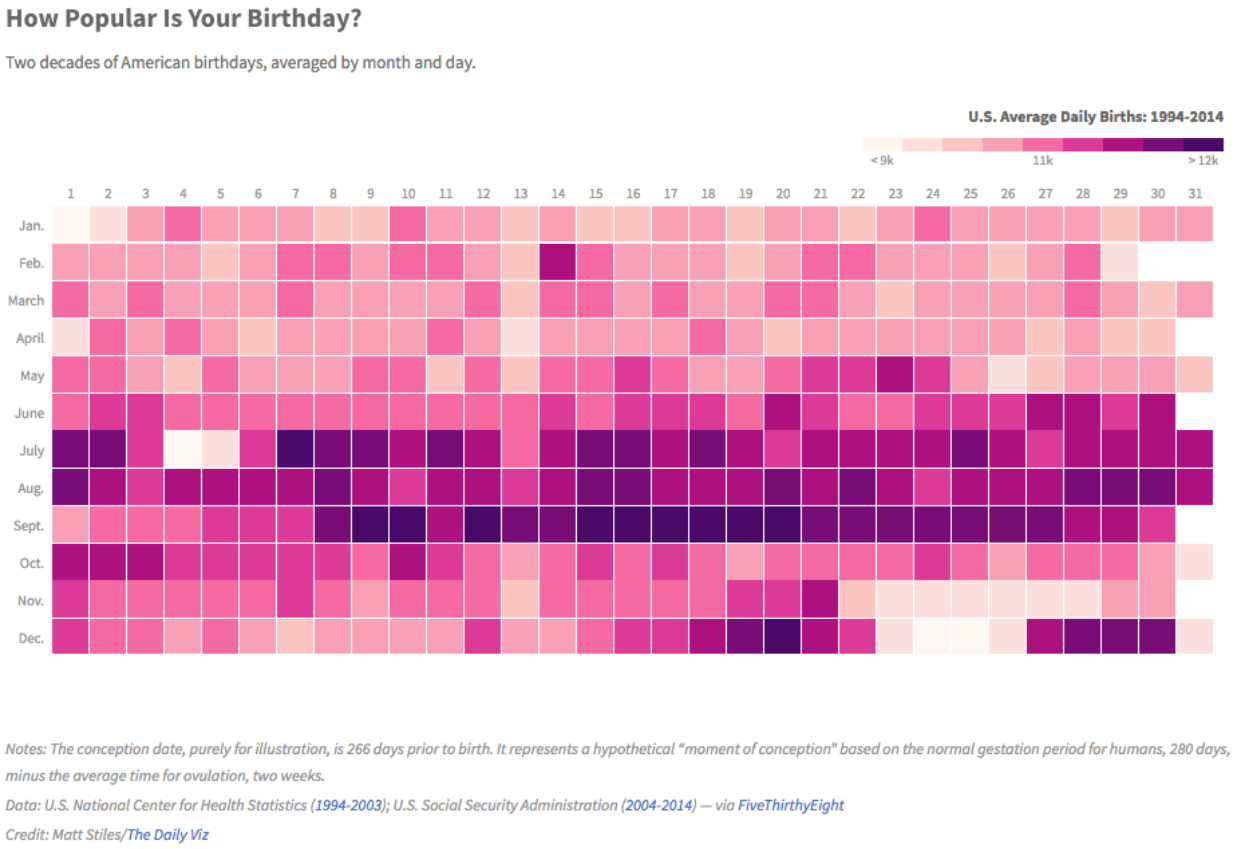
```
ggplot(dat,aes(x=npeople,y=answer)) +
  geom_line() +
  labs(x = "Number of People") +
  labs(y="Probability") +
  labs(title = "Birthday Problem: probability of shared birthday for each size class") +
  geom_hline(yintercept=.5)
```



IN-CLASS: How many students needed to have the probability of a shared birthday exceed .75?

- copy the code from above
- what do you need to change?

STRETCH: but uniform (equal) probability of birth?



- Can you change the code `sample` to recalculate? - Will require you to wrangle the data to change from number of births per day to the relative probability. How?

Application: Voter Files & Polling

Many political parties, interest groups, and media organizations rely on voter files to predict elections.

Suppose you want to determine which counties to target with political activity based on the support for each party (e.g., get-out-the vote efforts, advertising, resource allocation, etc.).

```
load("data/pa.sample.select.Rdata")
glimpse(pa.sample)
```

```
## Rows: 98,548
## Columns: 16
## $ city              <chr> "Aliquippa", "Aliquippa", "Aliquippa", "Aliquippa", ~
## $ likely.party      <fct> D, D, D, R, D, R, R, D, R, D, D, D, D, D, D, R, R, R~
```

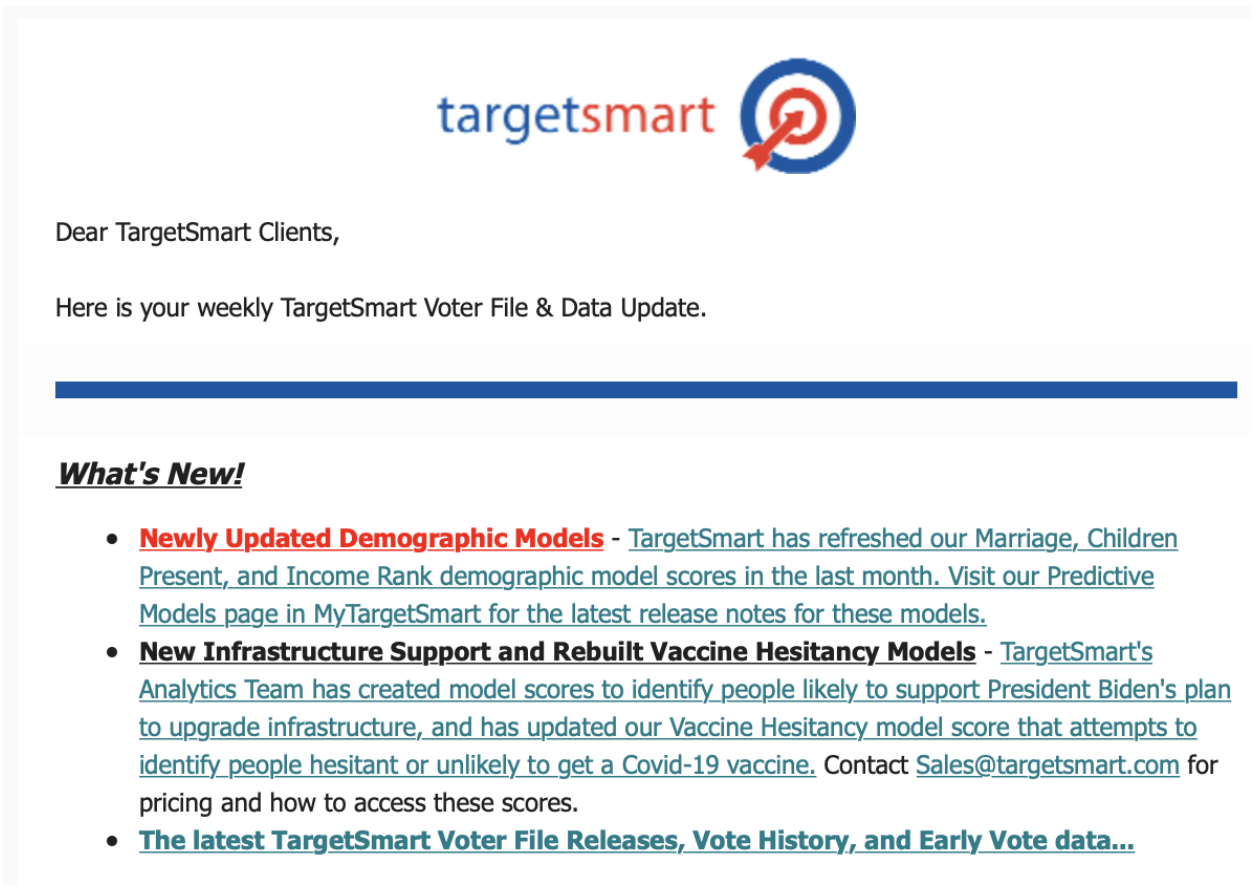


Figure 1: Email from yesterday (9/29/21)

```
## $ female            <dbl> 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1~
## $ AgeUnder30        <dbl> 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0~
## $ Age3039           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0~
## $ Age4049           <dbl> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0~
## $ Age5059           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0~
## $ Age6074           <dbl> 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1~
## $ Age75p            <dbl> 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ imputed.white     <dbl> 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1~
## $ imputed.black     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ imputed.hispanic  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ fips.county       <chr> "42007", "42007", "42007", "42007", "42007", "42007"~
## $ PctDemCtyVote2020 <dbl> 41.04, 41.04, 41.04, 41.04, 41.04, 41.04, 41.04, 41.~
## $ PctDemCtyVote2016 <dbl> 40.31, 40.31, 40.31, 40.31, 40.31, 40.31, 40.31, 40.~
## $ CooperateSurvey   <dbl> NA, 0, NA, NA, NA, NA, NA, NA, NA, 1, NA, NA, NA, NA~
```

One critical variable is partisanship!

```
pa.sample %>%
  count(likely.party)
```

```
##   likely.party     n
## 1            D 46937
## 2            I 12830
## 3            R 38781
```

Let's clean it up a bit. Why do we like indicator variables?

```
pa.sample <- pa.sample %>%
    mutate(likely.dem = ifelse(likely.party == "D",1,0),
          likely.rep = ifelse(likely.party == "R",1,0),
          likely.ind = ifelse(likely.party == "I",1,0))
```

```
pa.sample %>%
  count(likely.dem)
```

```
##   likely.dem     n
## 1          0 51611
## 2          1 46937
```

How does partisanship at the county-level vary?

```
pa.sample %>%
  group_by(fips.county) %>%
  summarize(DemPct = mean(likely.dem)) %>%
  arrange(-DemPct)
```

```
## # A tibble: 68 x 2
##    fips.county DemPct
##    <chr>        <dbl>
##  1 42101        0.775
##  2 42003        0.587
##  3 42069        0.578
##  4 42091        0.519
##  5 42077        0.514
##  6 42049        0.511
##  7 42045        0.496
##  8 42079        0.492
##  9 42007        0.480
```

```
## 10 <NA>         0.467
## # ... with 58 more rows
```

What fraction of counties in PA have 50% Democrats?

```
pa.sample %>%
  group_by(fips.county) %>%
  summarize(DemMaj = mean(likely.dem) > .5) %>%
  ungroup %>%
  summarize(PctDemCounties = mean(DemMaj))
```

```
## # A tibble: 1 x 1
##   PctDemCounties
##            <dbl>
## 1         0.0882
```

What does that imply about the efficiency of targetting particular counties for activity?

IN CLASS: how many counties have more Democrats than Republicans?

- What does that mean?
- How might the Democrats and Republicans approach elections differently given this?

Statewide breakdown?

```
PA.pty.breakdown <- pa.sample %>%
          summarize(pct.dem = mean(likely.dem),
                    pct.rep = mean(likely.rep),
                    pct.ind = mean(likely.ind))

PA.pty.breakdown
```

```
##     pct.dem  pct.rep   pct.ind
## 1 0.4762857 0.393524 0.1301904
```

- Does this seem right?

Motivation

- Suppose you want to conduct a study of voters to help your candidate. How many voters do you need to get an accurate result?
- How much accuracy do you have for a study of a given size?
- Much much does size matter? What matters more?
- When pollsters do polls they often sample directly from the voter file! If the voter file is "true" (is it?) how accurate are samples from the voter file?

Sampling (at least) Two Ways:

- `sample` is a "base R" function that samples from a vector
- `sample_n` is a `tidyverse` version that samples rows from a tibble. (need `dplyr` for `sample_n`)
- `replace = FALSE` is the default for each!

Using sample_n

```
#library(dplyr)
sample_n(pa.sample,5,replace= TRUE)
```

```
##            city likely.party female AgeUnder30 Age3039 Age4049 Age5059 Age6074
## 1          Erie             D      1          0       1       0       0       0
## 2    Pittsburgh             D      0          0       1       0       0       0
## 3 Connellsville             D      0          0       0       0       0       1
## 4 Elizabethtown             D      0          0       0       0       1       0
## 5  Philadelphia             D      1          0       0       1       0       0
##   Age75p imputed.white imputed.black imputed.hispanic fips.county
## 1      0             1             0                0       42049
## 2      0             0             1                0       42003
## 3      0             0             0                0       42051
## 4      0             1             0                0       42071
## 5      0             1             0                0       42101
##   PctDemCtyVote2020 PctDemCtyVote2016 CooperateSurvey likely.dem likely.rep
## 1             50.52             49.17              NA          1          0
## 2             60.36             58.62              NA          1          0
## 3             33.15             34.16              NA          1          0
## 4             41.97             39.78               0          1          0
## 5             81.98             84.30              NA          1          0
##   likely.ind
## 1          0
## 2          0
## 3          0
## 4          0
## 5          0
```

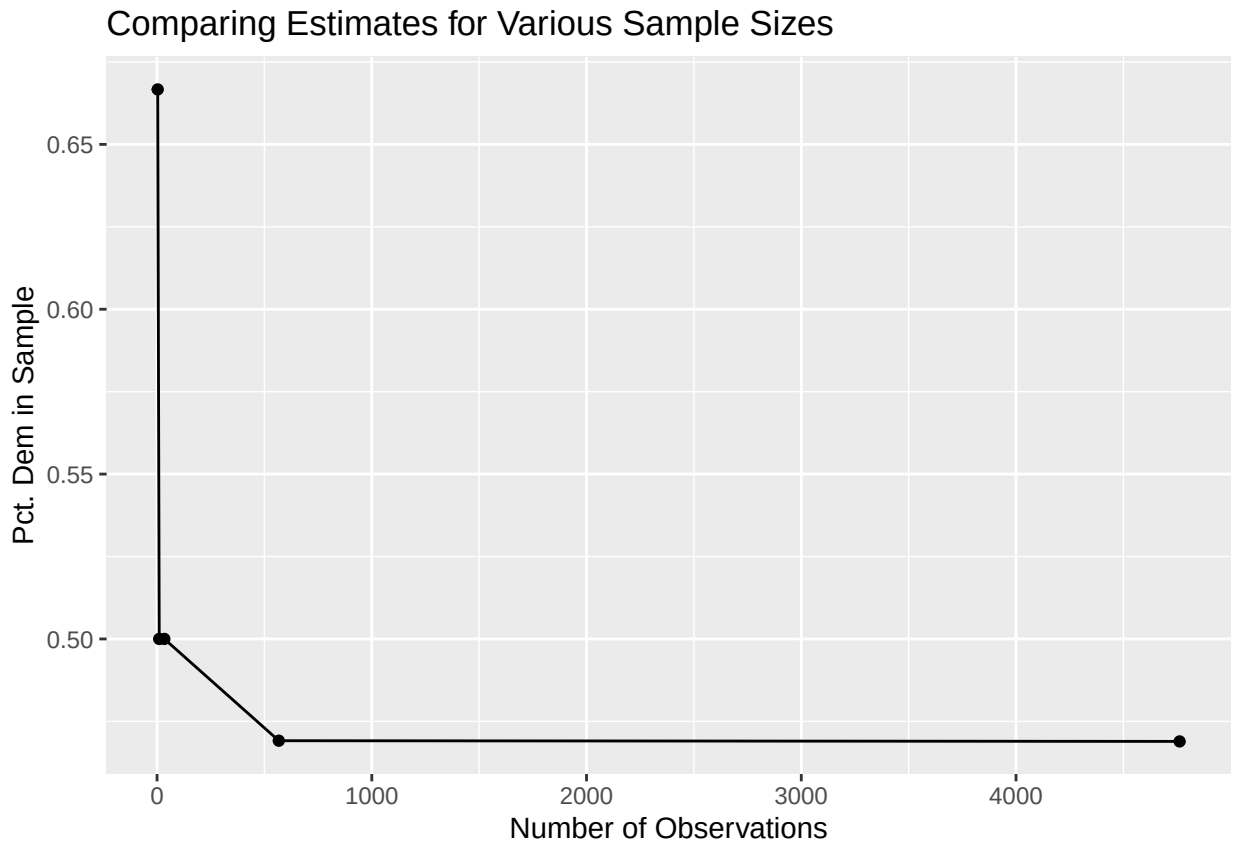
Now let's use `sample_n` in a loop to illustrate the importance of sample size.

```
pty.est <- NULL
samplesize <- c(3,10,34,567,4762)

for(i in seq_along(samplesize)){
  pty.est <- pa.sample %>%
    sample_n(samplesize[i], replace= TRUE) %>%
    summarize(MeanDem = mean(likely.dem),
              MeanRep = mean(likely.rep),
              MeanInd = mean(likely.ind)) %>%
    mutate(SampleSize = samplesize[i]) %>%
    bind_rows(pty.est)
}


pty.est
```

```
##     MeanDem   MeanRep   MeanInd SampleSize
## 1 0.4689206 0.3998320 0.1312474       4762
## 2 0.4691358 0.4056437 0.1252205        567
## 3 0.5000000 0.3823529 0.1176471         34
## 4 0.5000000 0.3000000 0.2000000         10
## 5 0.6666667 0.3333333 0.0000000          3
```

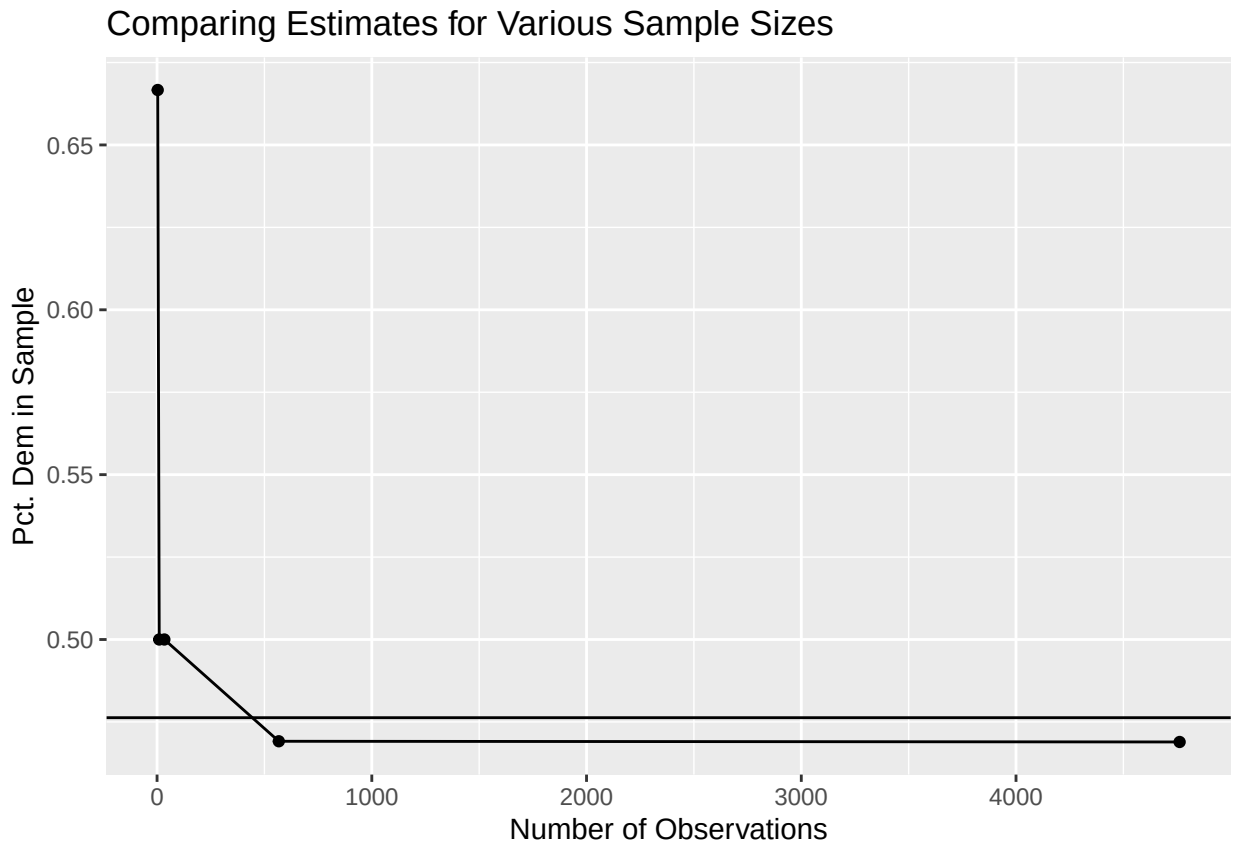
Can we visualize the relationship Between sample size and the estimate we get?

```
pasample.plot <- pty.est %>%
  ggplot(aes(x=SampleSize,y=MeanDem)) +
  geom_line() +
  geom_point() +
  labs(x = "Number of Observations") +
  labs(y = "Pct. Dem in Sample") +
  labs(title = "Comparing Estimates for Various Sample Sizes")

pasample.plot
```



Now add the "truth" to compare!

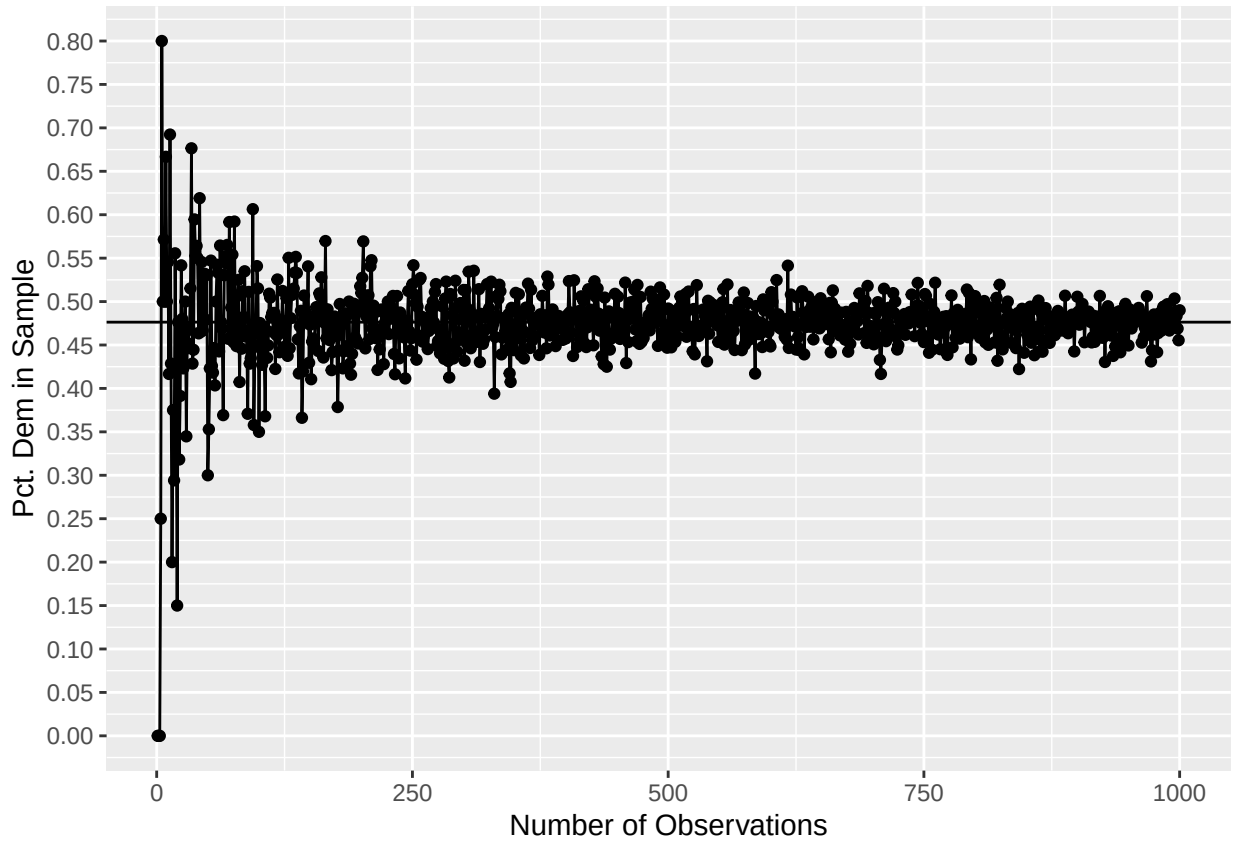
```
pasample.plot + geom_hline(yintercept=PA.pty.breakdown[[1]])
```



IN-CLASS: Can you add `MeanRep` to this plot? What do you have to change?

Now let's get more granular - look at the relationship for samples of size 1 to 1000!

```
samplesize <- seq(1,1000)

pty.est <- NULL

for(i in seq_along(samplesize)){
  pty.est <- pa.sample %>%
    sample_n(samplesize[i], replace= TRUE) %>%
    summarize(MeanDem = mean(likely.dem),
              MeanRep = mean(likely.rep),
              MeanInd = mean(likely.ind)) %>%
    mutate(SampleSize = samplesize[i]) %>%
    bind_rows(pty.est)
}

pty.est %>%
  ggplot() +
  geom_line(aes(x=SampleSize,y=MeanDem)) +
  geom_point(aes(x=SampleSize,y=MeanDem)) +
  labs(x = "Number of Observations") +
  labs(y = "Pct. Dem in Sample") +
  scale_y_continuous(breaks=seq(0,1,by=.05)) +
  geom_hline(yintercept=PA.pty.breakdown[[1]])
```



IN CLASS: How much better do we do if have up to 10,000 respondents/data points?

So how big of a (random) sample do we need?

- How does the precision of our estimate change as the sample size increases?
- How many data points do we need to give "accurate" estimates? (Assuming everything else is OK!)
- ASIDE: `cut`: takes a vector and creates a factor that labels groups based on dividing vector into equal sizes based on `breaks`

```
x <- c(1,2,3,4,5,6)
cut(x,breaks=2)
```

```
## [1] (0.995,3.5] (0.995,3.5] (0.995,3.5] (3.5,6]     (3.5,6]     (3.5,6]
## Levels: (0.995,3.5] (3.5,6]
```

What do we mean by "accurate"? Two measures:

```
pty.est <-  pty.est %>%
 mutate(Absolute.Error = abs(MeanDem - PA.pty.breakdown[[1]]),
        Squared.Error = (MeanDem - PA.pty.breakdown[[1]])^2,
        cut = cut(SampleSize, breaks = 10))
```

How does absolute error change by sample size?

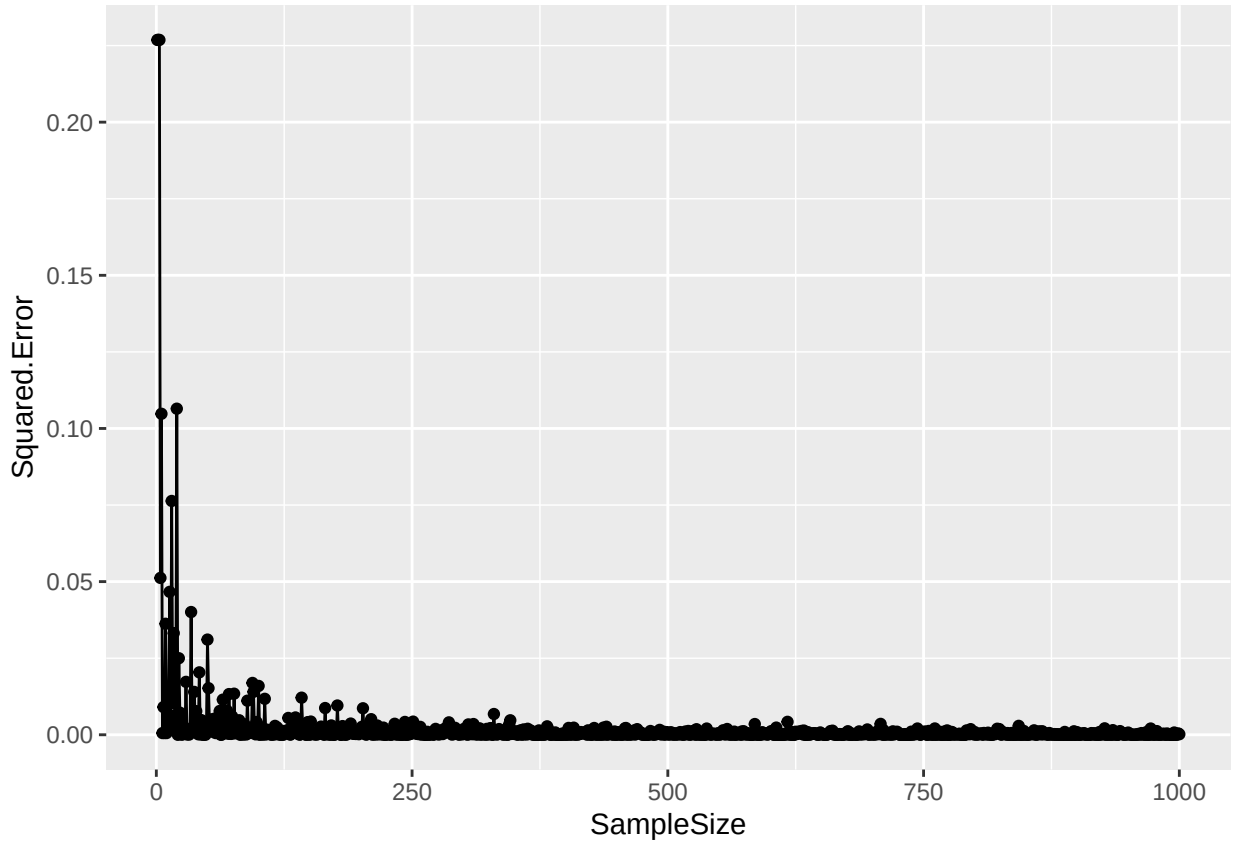
```
pty.est %>%
 group_by(cut) %>%
```

```
  summarize(avgae = mean(Absolute.Error),
            sdae = sd(Absolute.Error))
```

```
## # A tibble: 10 x 3
##    cut          avgae   sdae
##    <fct>        <dbl>  <dbl>
##  1 (0.001,101] 0.0820 0.0958
##  2 (101,201]   0.0290 0.0237
##  3 (201,301]   0.0246 0.0188
##  4 (301,401]   0.0210 0.0171
##  5 (401,500]   0.0189 0.0136
##  6 (500,600]   0.0164 0.0113
##  7 (600,700]   0.0153 0.0123
##  8 (700,800]   0.0160 0.0128
##  9 (800,900]   0.0143 0.0110
## 10 (900,1e+03] 0.0122 0.0101
```

Visualize!

```
pty.est %>% ggplot() +
  geom_point(aes(x=SampleSize,y=Squared.Error)) +
  geom_line(aes(x=SampleSize,y=Squared.Error))
```



RECAP: Law of Large Numbers

- As the number of data points being analyzed get larger, the mean of a random sample of that data will get closer and closer to the true mean in the data generating process.

- Importance of random sampling! If every observation has an equal chance of being observed/measured/studied, more data means more accurate results!
- BUT, is the data really a random sample? Is random sampling the largest source of error?

Amazing Result 2: Central Limit Theorem

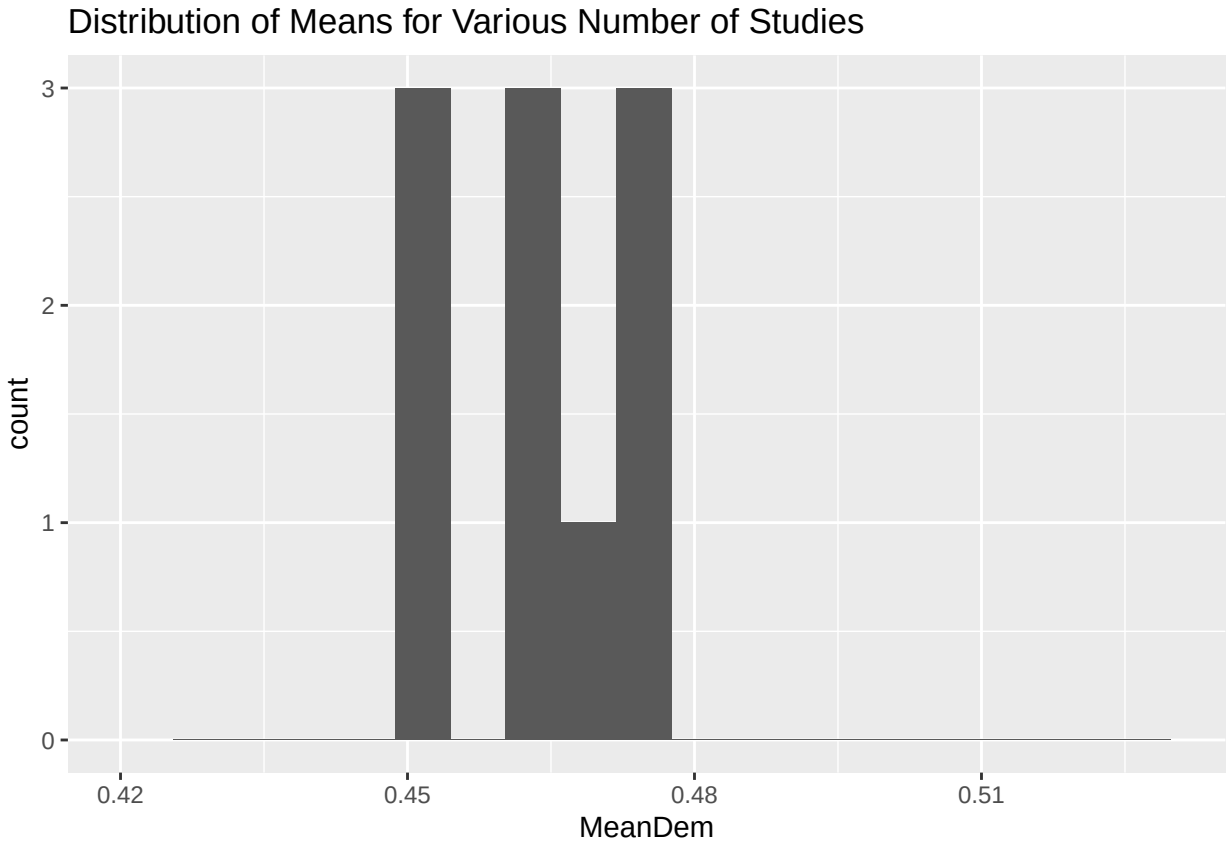
So we know that a large random sample of data is expected to give a sample average close to the true average (Law of Large Numbers).

- But can we say anything about the distribution of the sample mean?
- i.e., What if we take repeated observations from the same data generating process, calculate the mean, and then look at the shape of the result histogram? What will the histogram of means look like?
- How does the answer depend on if our data is binary (0,1), categorical (e.g., 1,2,3,4), or continuous?
- Hard case? Distribution of the mean of a binary variable: "likely Dem" (1), or not (0)

Let's consider resampling means:

```
# Fix Sample Size, Vary number of Samples
n.samplesize <- 1000
n.samples <- 10

pty.mean <- NULL

for(i in 1:n.samples){
  pty.mean <- pa.sample %>%
    sample_n(n.samplesize, replace= TRUE) %>%
    summarize(MeanDem = mean(likely.dem),
              MeanRep = mean(likely.rep),
              MeanInd = mean(likely.ind)) %>%
    bind_rows(pty.mean)
}

clt.plot <- pty.mean %>% ggplot() +
  geom_histogram(aes(x=MeanDem), bins=20) +
  labs(title="Distribution of Means for Various Number of Studies") +
  xlim(.42,.53)

clt.plot
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



IN-CLASS: Plot distribution of means for 50 studies, 100 studies, 500 studies, 1000 studies

- Think of this as the number of polls being done each week.
- Or number of quality assurance tests being done on a product to test for manufacturing defects.
- What do you observe?

Central Limit Theorem

- The distribution of means from random samples from a population will approach a normal distribution as the number of random samples being drawn (and analyzed) increases.
- This is why the normal distribution is so special!
- Basis of a lot of statistics: e.g., difference of means tests (Z-test, T-test)

One application is the margin of error in polls.

We start with a simple sample `my.sample` and then use the results of that study to make inferences about the larger population (`pa.sample`).

```
n.samplesize <- 1000
my.sample <- sample_n(pa.sample,n.samplesize,replace= TRUE)

mean(my.sample$likely.dem)
```

```
## [1] 0.476
```

```
mean(pa.sample$likely.dem) # Truth
```

```
## [1] 0.4762857
```

```
mean(my.sample$likely.dem) - mean(pa.sample$likely.dem) # Error
```

```
## [1] -0.0002856679
```

Let's see what the distribution of 5000 means looks like!

First lets create a matrix of 5000 poll results. How many responses are in each poll? So how many total observations?

```
B <- 5000

resample5000 <- NULL
for(i in 1:n.samples){
  resample5000 <- my.sample %>%
    sample_n(nrow(my.sample), replace= TRUE) %>%
    summarize(MeanDem = mean(likely.dem),
              MeanRep = mean(likely.rep),
              MeanInd = mean(likely.ind)) %>%
    bind_rows(resample5000)
}
```

Now use this tibble to summarize the variation/dispersion in the means!

```
quantile5000 <- resample5000 %>%
  summarize(pct025.DemMean = quantile(MeanDem,.025),
            pct05.DemMean = quantile(MeanDem,.05),
            pct25.DemMean = quantile(MeanDem,.25),
            pct75.DemMean = quantile(MeanDem,.75),
            pct95.DemMean = quantile(MeanDem,.95),
            pct975.DemMean = quantile(MeanDem,.975))

quantile5000
```

```
##   pct025.DemMean pct05.DemMean pct25.DemMean pct75.DemMean pct95.DemMean
## 1       0.465225       0.46545       0.47625       0.49525         0.513
##   pct975.DemMean
## 1          0.513
```

If we recenter by subtracting off the mean we get the margin of error. (May be useful to multiple by 100 to put in percentage points?)

```
quantile5000 - mean(my.sample$likely.dem)
```

```
##   pct025.DemMean pct05.DemMean pct25.DemMean pct75.DemMean pct95.DemMean
## 1      -0.010775      -0.01055       0.00025       0.01925         0.037
##   pct975.DemMean
## 1          0.037
```

```
100*(quantile5000 - mean(my.sample$likely.dem))
```

```
##   pct025.DemMean pct05.DemMean pct25.DemMean pct75.DemMean pct95.DemMean
## 1        -1.0775        -1.055         0.025         1.925           3.7
##   pct975.DemMean
## 1            3.7
```

- 50% of sample means fall between what 2 values?
- Why does it get bigger for bigger percentiles?

How does the size of the margin of error change by sample size? Double Looping

- What does `i` do?
- What does `j` do?

Why is `resample <- NULL` within the first loop but `MoE <- Null` is not?

```
samplesizes <- c(10,100,200,300,400,500,600,700,800,900,1000,5000,10000)
MoE <- NULL

for(i in seq_along(samplesizes)){

  resample <- NULL

  for(j in 1:1000){
  resample <- pa.sample %>%
    sample_n(samplesizes[i], replace= TRUE) %>%
    summarize(MeanDem = mean(likely.dem),
              MeanRep = mean(likely.rep),
              MeanInd = mean(likely.ind)) %>%
    mutate(SampleSize = samplesizes[i]) %>%
    bind_rows(resample)
  }

  MoE <- resample %>%
    summarize(moe = abs(quantile(MeanDem,.025) - mean(MeanDem))) %>%
    mutate(SampleSize = samplesizes[i]) %>%
    bind_rows(MoE)
}
```

What does that look like?

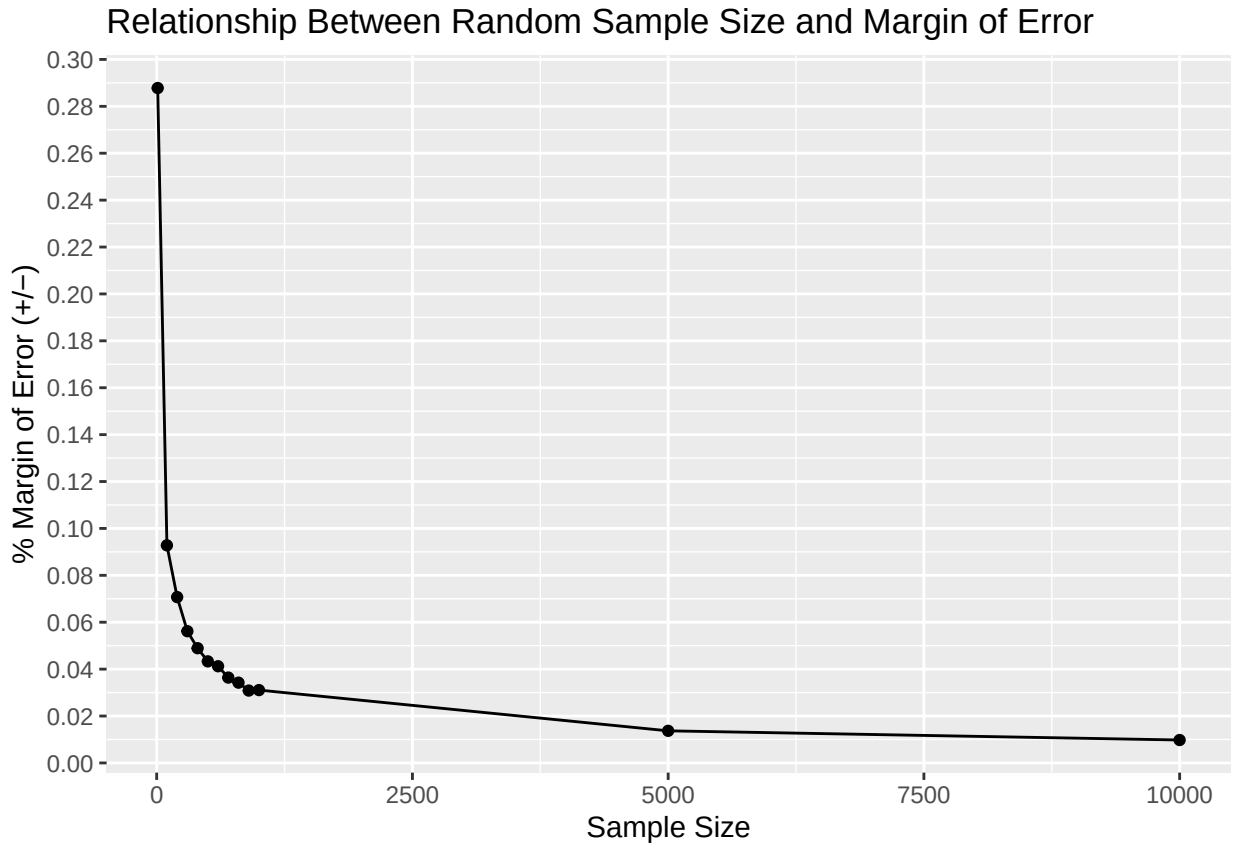
```
MoE
```

```
##           moe SampleSize
## 1  0.00976880      10000
## 2  0.01368980       5000
## 3  0.03106200       1000
## 4  0.03086222        900
## 5  0.03423125        800
## 6  0.03639571        700
## 7  0.04123500        600
## 8  0.04331400        500
## 9  0.04891750        400
## 10 0.05617333        300
## 11 0.07072000        200
## 12 0.09278000        100
## 13 0.28780000         10
```

Visualize for clarity!

```
ggplot(MoE) +
  geom_point(aes(x=SampleSize,y=moe)) +
```

```
geom_line(aes(x=SampleSize,y=moe)) +
labs(x="Sample Size") +
labs(title = "Relationship Between Random Sample Size and Margin of Error") +
scale_y_continuous(breaks=seq(0,.3,by=.02)) +
labs(y="% Margin of Error (+/-)")
```



IN-CLASS: Now do the same for likely independents. How does the width change? Which is bigger? Why?

Bootstrap: General Idea

- General idea is to use our sample as the population – draw repeated samples to learn about the variation in our estimates.
- Completely generic – can be applied to *any* function/statistic of the data!
- Completely easy – all we need to do is to be able to sample from our data with replacement!
- The basic idea – variation in the samples we draw reflect how any single sample may vary from the true population. This between-sample variation will result in variation in the statistic/function we are interested in that we can then use.
- Limitation? – computing time/memory.

NOTE: Assumes that our samples are identically and independently distributed.

Implementation:

1. Sample from our data `B` times with replacement.
2. For each sample `b` in `B`, calculate the statistic of interest.
3. Use the distribution of those `b` statistics to evaluate precision!

OK, let's work with this – bootstrap the sample mean.

We get sample of individual level data from the voter file. How accurate are various samples?

- Assume voter file is truth and our sample is an estimate!
- Probability the % of likely Democrats is greater than .4?
- Probability Dem % is greater than 3 times the Ind %?
- Probability 10% more Dems than Reps in the state of PA?

```
B <- 5000

bootstrap5000 <- NULL
for(i in 1:n.samples){
  bootstrap5000<- my.sample %>%
    sample_n(nrow(my.sample), replace= TRUE) %>%
    summarize(MeanDem = mean(likely.dem),
              MeanRep = mean(likely.rep),
              MeanInd = mean(likely.ind),
              MeanDemgt = ifelse(MeanDem > .4,1,0),
              MeanDemRep = ifelse(MeanDem > MeanRep,1,0),
              Dem3Ind = ifelse(MeanDem > 3*MeanInd,1,0),
              DR10diff = ifelse(MeanDem-MeanRep > .1,1,0)) %>%
    bind_rows(bootstrap5000)
}
```

Now lets analyze the bootstrap tibble to get the results! (Note we could look at the quantiles as well to estimate the precision of these point estimates!)

```
bootstrap5000 %>%
  summarize(MeanDemGT = mean(MeanDemgt),
            MeanDemRep = mean(MeanDemRep),
            Dem3Ind  = mean(Dem3Ind),
            DR10diff  = mean(DR10diff))
```

```
##   MeanDemGT MeanDemRep Dem3Ind DR10diff
## 1         1          1       1      0.4
```

What if we lack the underlying individual level data?

- In the 2020 Democratic Primary, a candidate would only receive delegates to the Democratic Convention if they get at least 15%.
- What is the probability that a candidate will get a delegate?
- `rbinom` - how many 1's if we draw `n` samples of `size` observations with the probability of seeing a 1 is `prob` and the probability of seeing a 0 is `1-prob`.
- 1000 Samples of a poll of 579 respondents where the probability of Sanders Support is .14

What does this do?

***The Economist*/YouGov Poll**
November 3 - 5, 2019 - 1500 US Adult citizens

YouGov

51. Democratic candidate - first choice
If the Democratic presidential primary or caucus in your state were held today, who would you vote for?
Asked of registered voters who say they will vote in the Democratic Presidential primary or caucus in 2020

		Gender		Age (4 category)				Race (4 category)			
	Total	Male	Female	18-29	30-44	45-64	65+	White	Black	Hispanic	Other
Joe Biden	26%	27%	24%	10%	22%	30%	35%	20%	42%	35%	*
Elizabeth Warren	25%	25%	25%	26%	22%	26%	25%	30%	17%	9%	*
Bernie Sanders	14%	14%	15%	27%	24%	9%	3%	13%	14%	21%	*
Pete Buttigieg	8%	7%	8%	4%	3%	6%	16%	10%	0%	6%	*
Kamala Harris	6%	4%	8%	5%	8%	8%	3%	6%	7%	6%	*
Julian Castro	3%	3%	3%	2%	6%	2%	1%	1%	5%	7%	*
Tulsi Gabbard	3%	4%	1%	2%	1%	3%	3%	4%	1%	0%	*
Cory Booker	2%	1%	2%	1%	4%	2%	1%	1%	3%	3%	*
Amy Klobuchar	2%	2%	2%	1%	1%	2%	3%	2%	0%	2%	*
Marianne Williamson	1%	2%	0%	2%	1%	1%	0%	1%	1%	0%	*
Steve Bullock	1%	1%	0%	3%	1%	0%	0%	1%	0%	2%	*
Tom Steyer	1%	1%	0%	2%	0%	1%	1%	1%	0%	0%	*
Andrew Yang	1%	1%	1%	1%	2%	0%	0%	1%	0%	0%	*
John Delaney	1%	1%	1%	1%	1%	1%	0%	0%	0%	6%	*
Michael Bennet	0%	1%	0%	2%	0%	0%	0%	0%	2%	0%	*
Wayne Messam	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	*
Joe Sestak	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	*
Not sure	6%	5%	7%	7%	3%	5%	9%	6%	5%	0%	*
I would not vote	2%	1%	3%	3%	1%	3%	1%	2%	2%	3%	*
Totals	102%	100%	100%	99%	100%	99%	101%	99%	99%	100%	*
Unweighted N	(579)	(251)	(328)	(110)	(102)	(243)	(124)	(371)	(106)	(74)	(28)

Figure 2: YouGov Polling Results

```
SandersSupport <- rbinom(n=1000, size=579, prob=.14)
summary(SandersSupport)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   49.00   75.00   81.00   80.64   86.00  108.00
```

What does this do?

```
SandersSupport <- SandersSupport/579
summary(SandersSupport)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.08463 0.12953 0.13990 0.13928 0.14853 0.18653
```

IN CLASS: How can we calculate the probability that Sanders is > .15?

Can we do this? Why or why not? How does this differ from what we did above?

```
MeanDem <- rbinom(n=1000, size=100, prob=PA.pty.breakdown$pct.dem)
MeanRep <- rbinom(n=1000, size=100, prob=PA.pty.breakdown$pct.rep)

mean(MeanDem > MeanRep)
```

```
## [1] 0.86
```

But we can use a similar process for the presidential polls!

```
load(file="data/Pres2020.PV.Rdata")
election.day <- as.Date("11/3/2020", "%m/%d/%Y")
Pres2020.PV <- Pres2020.PV %>%
                mutate(EndDate = as.Date(Pres2020.PV$EndDate, "%m/%d/%Y"),
                       StartDate = as.Date(Pres2020.PV$StartDate, "%m/%d/%Y"),
                       DaysToED = as.numeric(election.day - EndDate),
                       margin = Biden - Trump,
                       pollnum = as.numeric(rownames(Pres2020.PV)),
                       BidenError = Biden - DemCertVote,
                       TrumpError = Trump - RepCertVote,
                       SignedError = (Biden-Trump) - (DemCertVote - RepCertVote))
```

```
dat <- NULL
r_sampleBiden <- matrix(NA,nrow=1000,ncol=nrow(Pres2020.PV))
r_sampleTrump <- matrix(NA,nrow=1000,ncol=nrow(Pres2020.PV))

for(i in 1:nrow(Pres2020.PV)){
  dat <- Pres2020.PV[i,]
  r_sampleBiden[,i] <- rbinom(n = 1000, size = dat$SampleSize, prob = dat$Biden/100)/dat$SampleSize
  r_sampleTrump[,i] <- rbinom(n = 1000, size = dat$SampleSize, prob = dat$Trump/100)/dat$SampleSize

  }
```

How does this compare to the reported "Margin of Error"

```
Pres2020.PV$MoE[2]
```

```
## [1] 3.1
```

```
abs(mean(r_sampleBiden[,2]) - quantile(r_sampleBiden[,2], .025))
```

```
##      2.5%
## 0.0309271
```

How much overall average error by candidate?

```
mean(r_sampleBiden - Pres2020.PV$DemCertVote[1]/100)
```

```
## [1] -0.01749731
```

```
mean(Pres2020.PV$BidenError/100)
```

```
## [1] -0.01748106
```

IN-CLASS: How much error for Trump?

- Does this suggest anything about random sampling?

SUPER STRETCH: which polls are more "accurate" than others? How does accuracy change over time?