# Gov 50: 21. More Hypothesis testing

Matthew Blackwell

Harvard University

# Roadmap

1. Hypothesis testing using infer

2. Two-sample tests

3. Two-sample permutation tests with infer

# **1/** Hypothesis testing using infer

# Statistical hypothesis testing

- Statistical hypothesis testing is a **thought experiment**.
- What would the world look like **if we knew the truth**?
- Conducted with several steps:
    1. Specify your **null** and **alternative hypotheses**
    2. Choose an appropriate **test statistic** and level of test $\alpha$
    3. Derive the **reference distribution** of the test statistic under the null.
    4. Use this distribution to calculate the **p-value**.
    5. Use p-value to decide whether to reject the null hypothesis or not

# GSS data from `infer`

```r
library(infer)
gss
```

```
## # A tibble: 500 x 11
##     year   age sex    college   partyid hompop hours income
##    <dbl> <dbl> <fct>  <fct>     <fct>    <dbl> <dbl> <ord>
##  1  2014    36 male   degree    ind          3    50 $25000~
##  2  1994    34 female no degree rep          4    31 $20000~
##  3  1998    24 male   degree    ind          1    40 $25000~
##  4  1996    42 male   no degree ind          4    40 $25000~
##  5  1994    31 male   degree    rep          2    40 $25000~
##  6  1996    32 female no degree rep          4    53 $25000~
##  7  1990    48 female no degree dem          2    32 $25000~
##  8  2016    36 female degree    ind          1    20 $25000~
##  9  2000    30 female degree    rep          5    40 $25000~
## 10  1998    33 female no degree dem          2    40 $15000~
## # ... with 490 more rows, and 3 more variables:
## #   class <fct>, finrela <fct>, weight <dbl>
```

# What is the average hours worked?

`dplyr` way:

```
gss |>
  summarize(mean(hours))
```

```
## # A tibble: 1 x 1
##   `mean(hours)`
##           <dbl>
## 1          41.4
```

`infer` way:

```
observed_mean <- gss |>
  specify(response = hours) |>
  calculate(stat = "mean")
observed_mean
```

```
## Response: hours (numeric)
## # A tibble: 1 x 1
##    stat
##   <dbl>
## 1  41.4
```

# Hypothesis test

Could we get a mean this different from 40 hours if that was the true population average of hours worked?

Null and alternative:

$$H_0 : \mu_{\texttt{hours}} = 40$$
$$H_1 : \mu_{\texttt{hours}} \neq 40$$

How do we perform this test using infer? The **bootstrap!**

# Specifying the hypotheses

```r
gss |>
  specify(response = hours) |>
  hypothesize(null = "point", mu = 40)
```

```
## Response: hours (numeric)
## Null Hypothesis: point
## # A tibble: 500 x 1
##    hours
##    <dbl>
##  1    50
##  2    31
##  3    40
##  4    40
##  5    40
##  6    53
##  7    32
##  8    20
##  9    40
## 10    40
## # ... with 490 more rows
```

# Generating the null distribution

We can use the bootstrap to determine how much variation there will be around 40 in the null distribution.

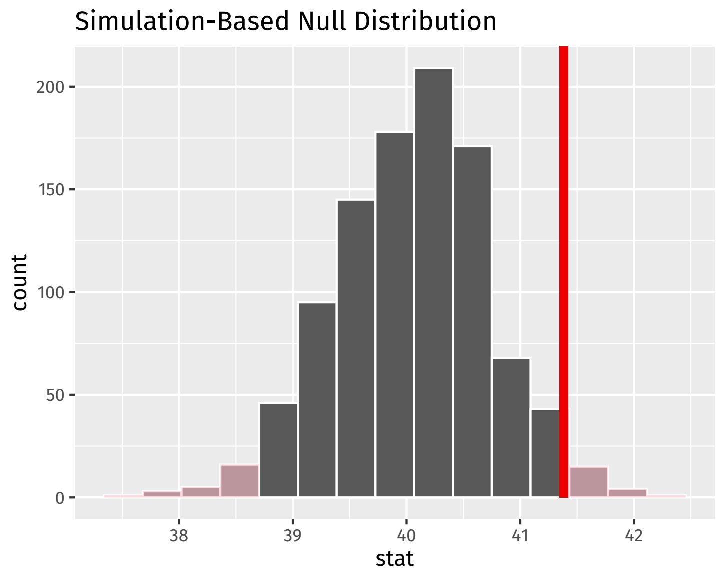
```
null_dist <- gss |>
  specify(response = hours) |>
  hypothesize(null = "point", mu = 40) |>
  generate(reps = 1000, type = "bootstrap") |>
  calculate(stat = "mean")
null_dist
```

```
## Response: hours (numeric)
## Null Hypothesis: point
## # A tibble: 1,000 x 2
##    replicate  stat
##        <int> <dbl>
##  1         1  40.3
##  2         2  39.6
##  3         3  40.8
##  4         4  39.6
##  5         5  39.8
##  6         6  39.8
##  7         7  40.6
```

# Visualizing the p-value

We can visualize our bootstrapped null distribution and the p-value as a shaded region:

```r
null_dist |>
  visualize() +
  shade_p_value(observed_mean,
                direction = "two-sided")
```


Simulation-Based Null Distribution
count
200
150
100
50
0
38
39
40
41
42
stat

# 2/ Two-sample tests

# Social pressure experiment

- Experimental study where each household for 2006 MI primary was randomly assigned to one of 4 conditions:
    - Control: no mailer
    - Civic Duty: mailer saying voting is your civic duty.
    - Hawthorne: a “we’re watching you” message.
    - Neighbors: naming-and-shaming social pressure mailer.
- Outcome: whether household members voted or not.
- We’ll focus on Neighbors vs Control
- Randomized implies samples are **independent**

# Neighbors mailer

Dear Registered Voter:

WHAT IF YOUR NEIGHBORS KNEW WHETHER YOU VOTED?

Why do so many people fail to vote? We've been talking about the problem for years, but it only seems to get worse. This year, we're taking a new approach. We're sending this mailing to you and your neighbors to publicize who does and does not vote.

The chart shows the names of some of your neighbors, showing which have voted in the past. After the August 8 election, we intend to mail an updated chart. You and your neighbors will all know who voted and who did not.

DO YOUR CIVIC DUTY — VOTE!

---

| MAPLE DR | Aug 04 | Nov 04 | Aug 06 |
|---|---|---|---|
| 9995 JOSEPH JAMES SMITH | Voted | Voted | ______ |
| 9995 JENNIFER KAY SMITH | | Voted | ______ |
| 9997 RICHARD B JACKSON | | Voted | ______ |
| 9999 KATHY MARIE JACKSON | | Voted | ______ |

# Social pressure data

```r
data(social, package = "qss")
social <- as_tibble(social)
social
```

```
## # A tibble: 305,866 x 6
##    sex    yearofbirth primary2004 messages   primar~1 hhsize
##    <chr>        <int>       <int> <chr>         <int>  <int>
##  1 male          1941           0 Civic Duty        0      2
##  2 female        1947           0 Civic Duty        0      2
##  3 male          1951           0 Hawthorne         1      3
##  4 female        1950           0 Hawthorne         1      3
##  5 female        1982           0 Hawthorne         1      3
##  6 male          1981           0 Control           0      3
##  7 female        1959           0 Control           1      3
##  8 male          1956           0 Control           1      3
##  9 female        1968           0 Control           0      2
## 10 male          1967           0 Control           0      2
## # ... with 305,856 more rows, and abbreviated variable name
## #   1: primary2006
```

# Two-sample hypotheses

- Parameter: **population ATE** $\mu_T - \mu_C$
  - $\mu_T$: Turnout rate in the population if everyone received treatment.
  - $\mu_C$: Turnout rate in the population if everyone received control.
- Goal: learn about the population difference in means
- Usual null hypothesis: no difference in population means (ATE = 0)
  - Null: $H_0 : \mu_T - \mu_C = 0$
  - Two-sided alternative: $H_1 : \mu_T - \mu_C \neq 0$
- In words: are the differences in sample means just due to chance?

# Permutation test

How do we generate draws of the difference in means under the null?
$H_0 : \mu_T - \mu_C = 0$

If the voting distribution is the same in the treatment and control groups, we could randomly swap who is labelled as treated and who is labelled as control and it shouldn't matter.

**Permutation test**: generate the null distribution by permuting the group labels and see the resulting distribution of differences in proportions

# Permuting the labels

```r
social <- social |>
  filter(messages %in% c("Neighbors", "Control"))

social |>
  mutate(messages_permute = sample(messages)) |>
  select(primary2006, messages, messages_permute)
```

```
## # A tibble: 229,444 x 3
##    primary2006 messages messages_permute
##          <int> <chr>    <chr>
##  1           0 Control  Control
##  2           1 Control  Control
##  3           1 Control  Neighbors
##  4           0 Control  Control
##  5           0 Control  Control
##  6           1 Control  Neighbors
##  7           0 Control  Control
##  8           1 Control  Control
##  9           1 Control  Control
## 10           1 Control  Control
## # ... with 229,434 more rows
```

# 3/ Two-sample permutation tests with infer

# Calculating the difference in proportion

`infer` functions with binary outcomes work best with factor variables:

```
social <- social |>
  mutate(turnout = if_else(primary2006 == 1, "Voted", "Didn't Vote"))

est_ate <- social |>
  specify(turnout ~ messages, success = "Voted") |>
  calculate(stat = "diff in props", order = c("Neighbors", "Control"))
est_ate
```

```
## Response: turnout (factor)
## Explanatory: messages (factor)
## # A tibble: 1 x 1
##     stat
##    <dbl>
## 1 0.0813
```

# Specifying the relationship of interest

`infer` functions with binary outcomes work best with factor variables:

```r
social |>
  specify(turnout ~ messages, success = "Voted")
```

```
## Response: turnout (factor)
## Explanatory: messages (factor)
## # A tibble: 229,444 x 2
##    turnout    messages
##    <fct>      <fct>
##  1 Didn't Vote Control
##  2 Voted      Control
##  3 Voted      Control
##  4 Didn't Vote Control
##  5 Didn't Vote Control
##  6 Voted      Control
##  7 Didn't Vote Control
##  8 Voted      Control
##  9 Voted      Control
## 10 Voted      Control
## # ... with 229,434 more rows
```

# Setting the hypotheses

The null for these two-sample tests is called "independence" for the infer package because the assumption is that the two variables are statistically independent.

```
social |>
  specify(turnout ~ messages, success = "Voted") |>
  hypothesize(null = "independence")
```

```
## Response: turnout (factor)
## Explanatory: messages (factor)
## Null Hypothesis: independence
## # A tibble: 229,444 x 2
##    turnout    messages
##    <fct>      <fct>
##  1 Didn't Vote Control
##  2 Voted      Control
##  3 Voted      Control
##  4 Didn't Vote Control
##  5 Didn't Vote Control
##  6 Voted      Control
##  7 Didn't Vote Control
##  8 Voted      Control
```

# Generating the permutations

We can tell `infer` to do our permutation test by using the argument `type = "permute"` to `generate()`:

```
social |>
  specify(turnout ~ messages, success = "Voted") |>
  hypothesize(null = "independence") |>
  generate(reps = 1000, type = "permute")
```

```
## Response: turnout (factor)
## Explanatory: messages (factor)
## Null Hypothesis: independence
## # A tibble: 229,444,000 x 3
## # Groups:   replicate [1,000]
##    turnout     messages replicate
##    <fct>       <fct>        <int>
##  1 Voted       Control          1
##  2 Didn't Vote Control          1
##  3 Voted       Control          1
##  4 Didn't Vote Control          1
##  5 Didn't Vote Control          1
##  6 Voted       Control          1
##  7 Voted       Control          1
```

# Calculating the diff in proportions in each sample

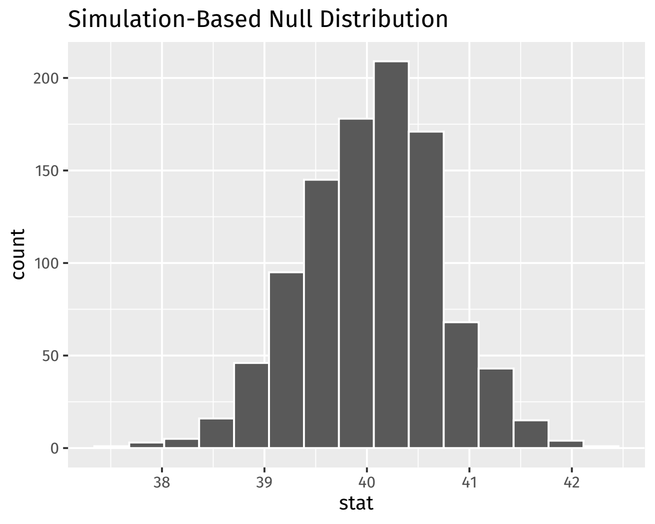
```r
null_dist <- social |>
  specify(turnout ~ messages, success = "Voted") |>
  hypothesize(null = "independence") |>
  generate(reps = 1000, type = "permute") |>
  calculate(stat = "diff in props", order = c("Neighbors", "Control"))
```

```r
null_dist
```

```
## Response: hours (numeric)
## Null Hypothesis: point
## # A tibble: 1,000 x 2
##    replicate  stat
##        <int> <dbl>
##  1         1  40.3
##  2         2  39.6
##  3         3  40.8
##  4         4  39.6
##  5         5  39.8
##  6         6  39.8
##  7         7  40.6
##  8         8  40.5
##  9         9  38.6
## 10        10  41.2
## # ... with 990 more rows
```

# Visualizing

```r
null_dist |>
  visualize()
```



# Calculating p-values

```r
ate_pval <- null_dist |>
  get_p_value(obs_stat = est_ate, direction = "both")
ate_pval
```

```
## # A tibble: 1 x 1
##   p_value
##     <dbl>
## 1       0
```

# Visualizing p-values

```
null_dist |>
  visualize() +
  shade_p_value(obs_stat = est_ate, direction = "both")
```