

Optimal parameters

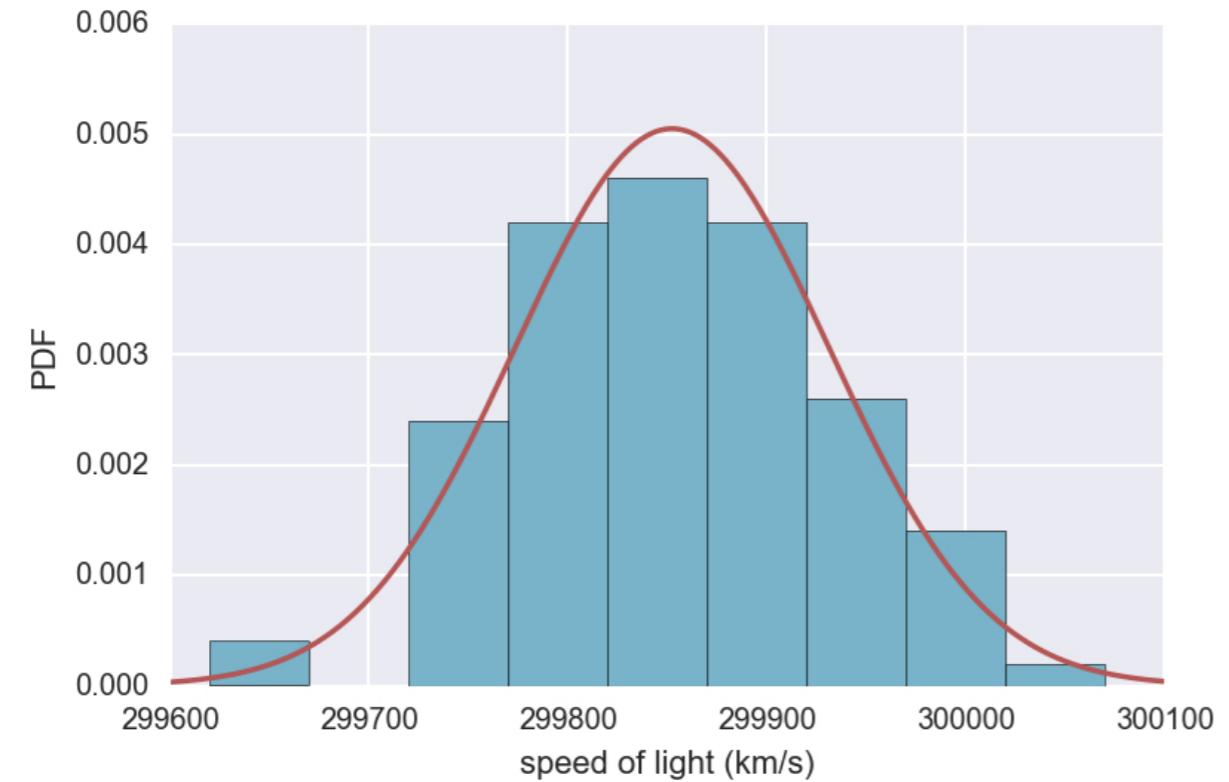
STATISTICAL THINKING IN PYTHON (PART 2)



Justin Bois

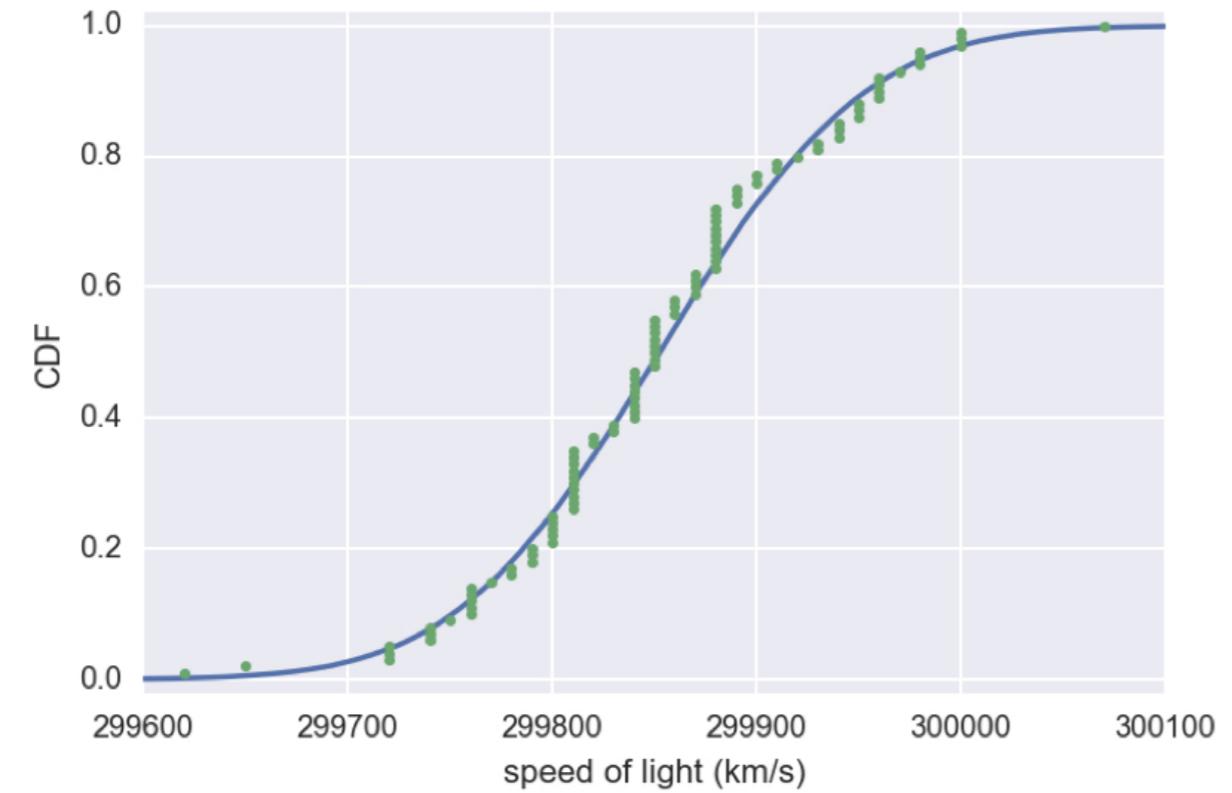
Lecturer at the California Institute of
Technology

Histogram of Michelson's measurements



¹ Data: Michelson, 1880

CDF of Michelson's measurements

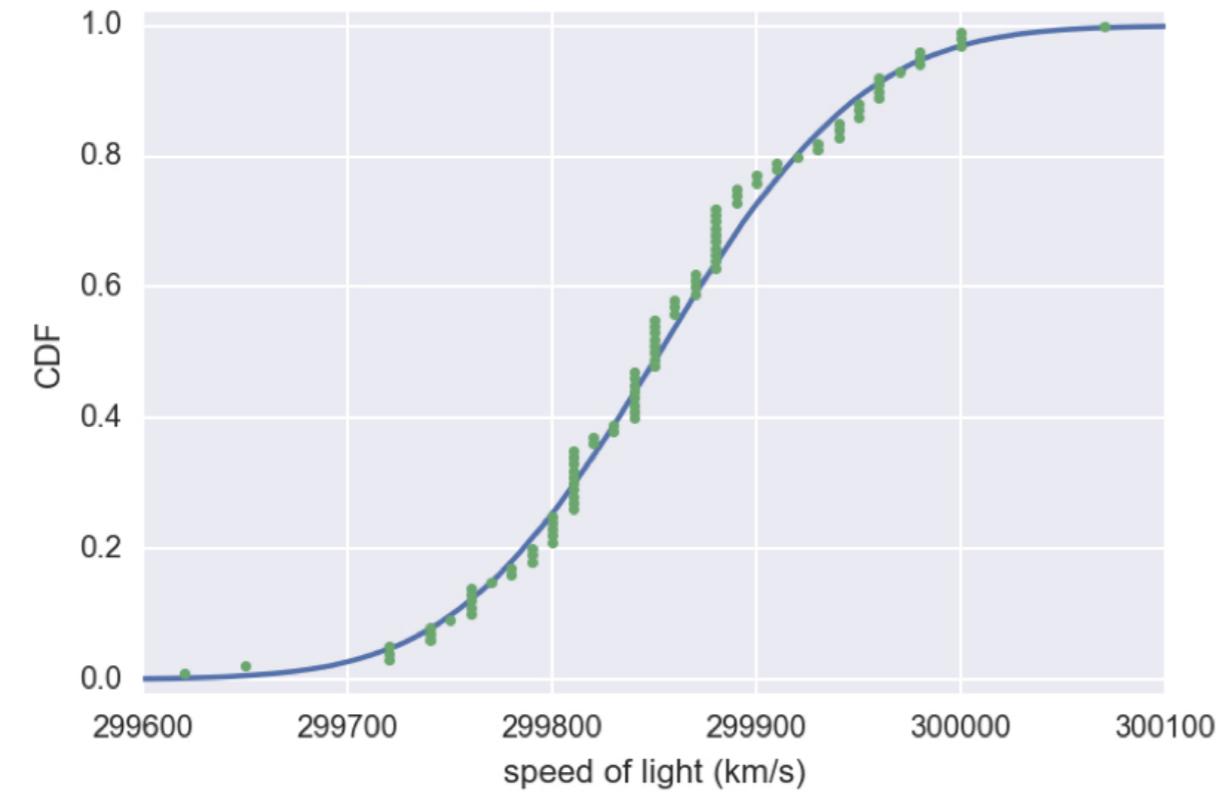


¹ Data: Michelson, 1880

Checking Normality of Michelson data

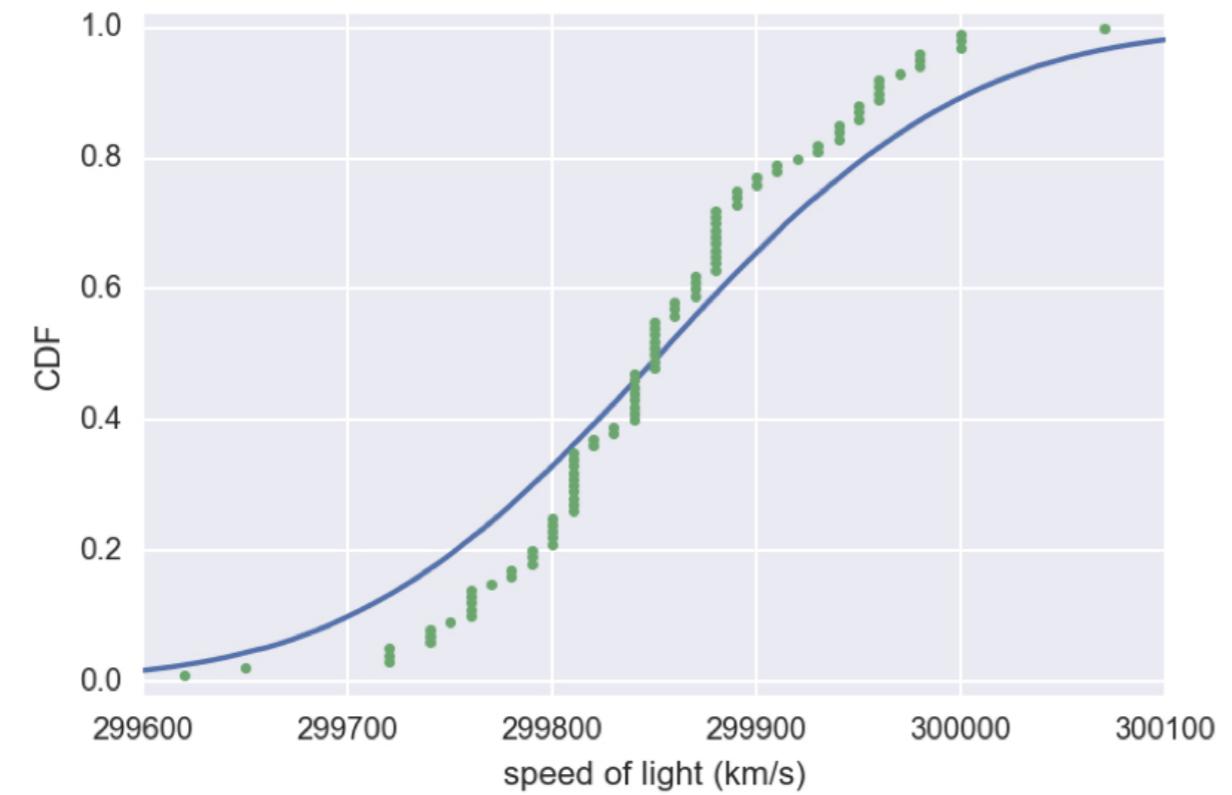
```
import numpy as np  
import matplotlib.pyplot as plt  
  
mean = np.mean(michelson_speed_of_light)  
std = np.std(michelson_speed_of_light)  
samples = np.random.normal(mean, std, size=10000)
```

CDF of Michelson's measurements



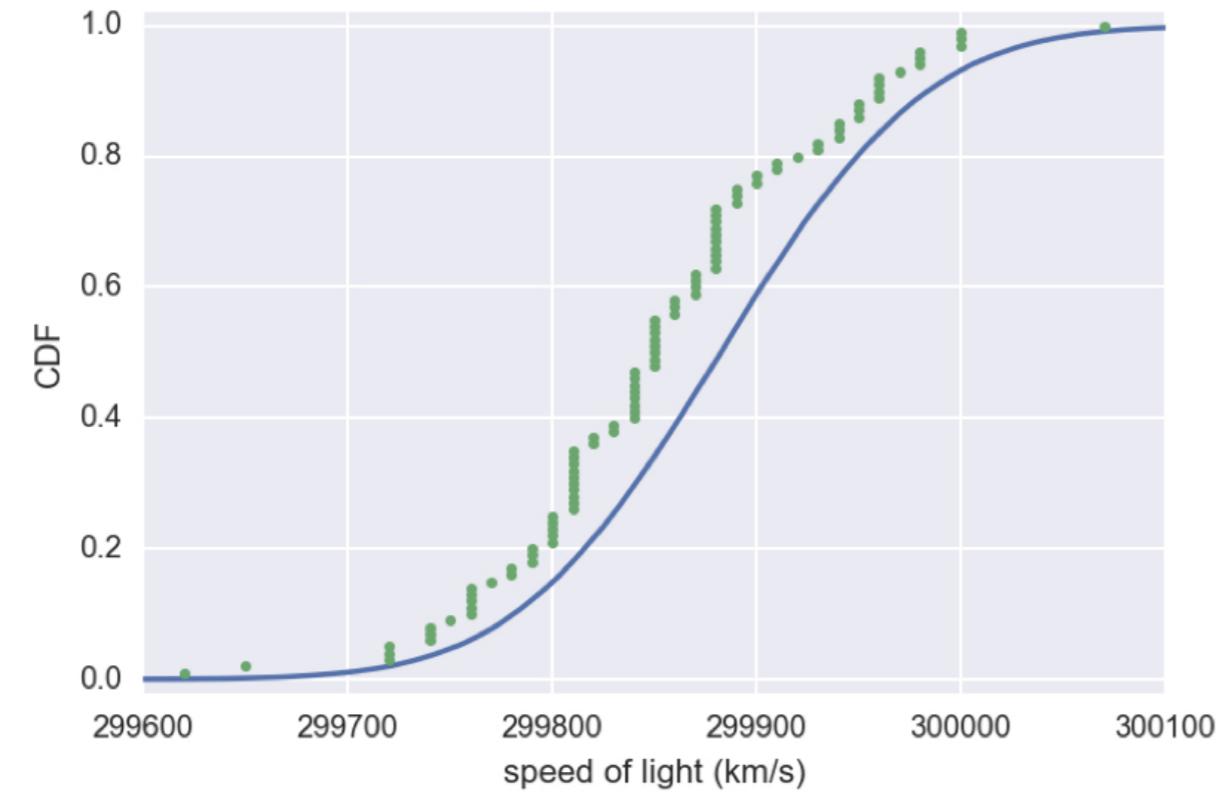
¹ Data: Michelson, 1880

CDF with bad estimate of st. dev.



¹ Data: Michelson, 1880

CDF with bad estimate of mean

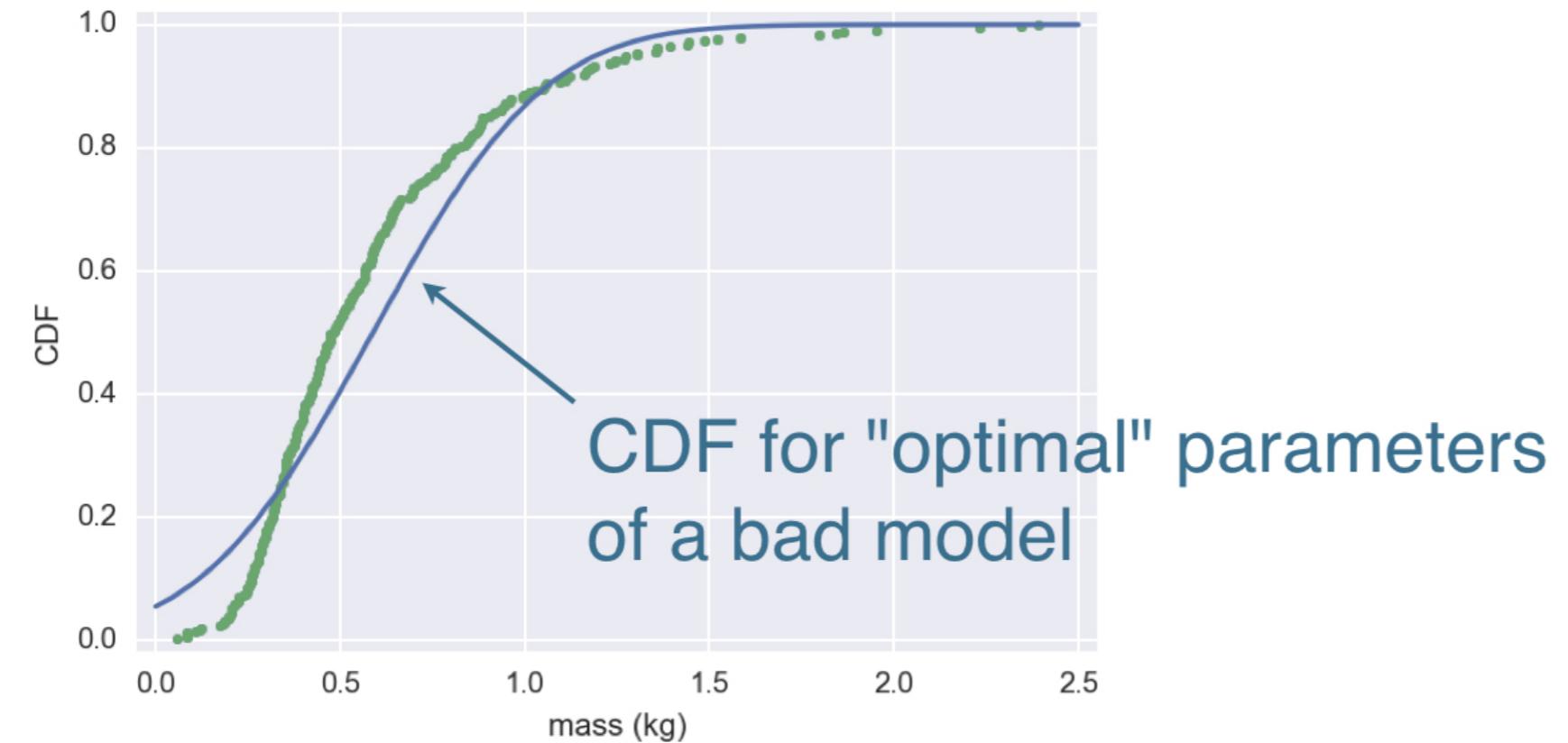


¹ Data: Michelson, 1880

Optimal parameters

- Parameter values that bring the model in closest agreement with the data

Mass of MA large mouth bass



¹ Source: Mass. Dept. of Environmental Protection

Packages to do statistical inference



scipy.stats

Packages to do statistical inference



scipy.stats



statsmodels

Packages to do statistical inference



scipy.stats



statsmodels



hacker stats
with numpy

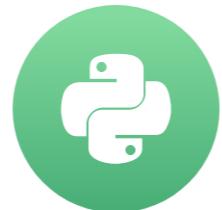
¹ Knife image: D²M Commons, CC BY³ SA 3.0

Let's practice!

STATISTICAL THINKING IN PYTHON (PART 2)

Linear regression by least squares

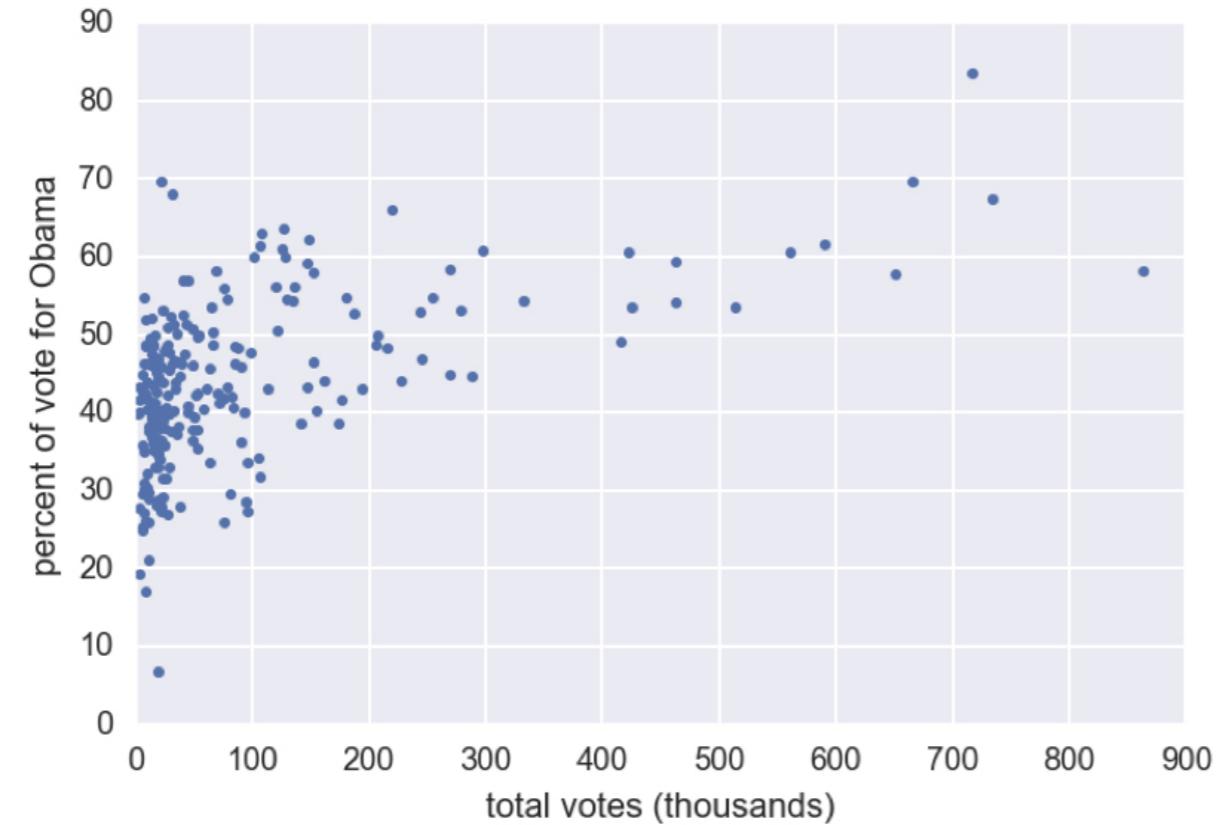
STATISTICAL THINKING IN PYTHON (PART 2)



Justin Bois

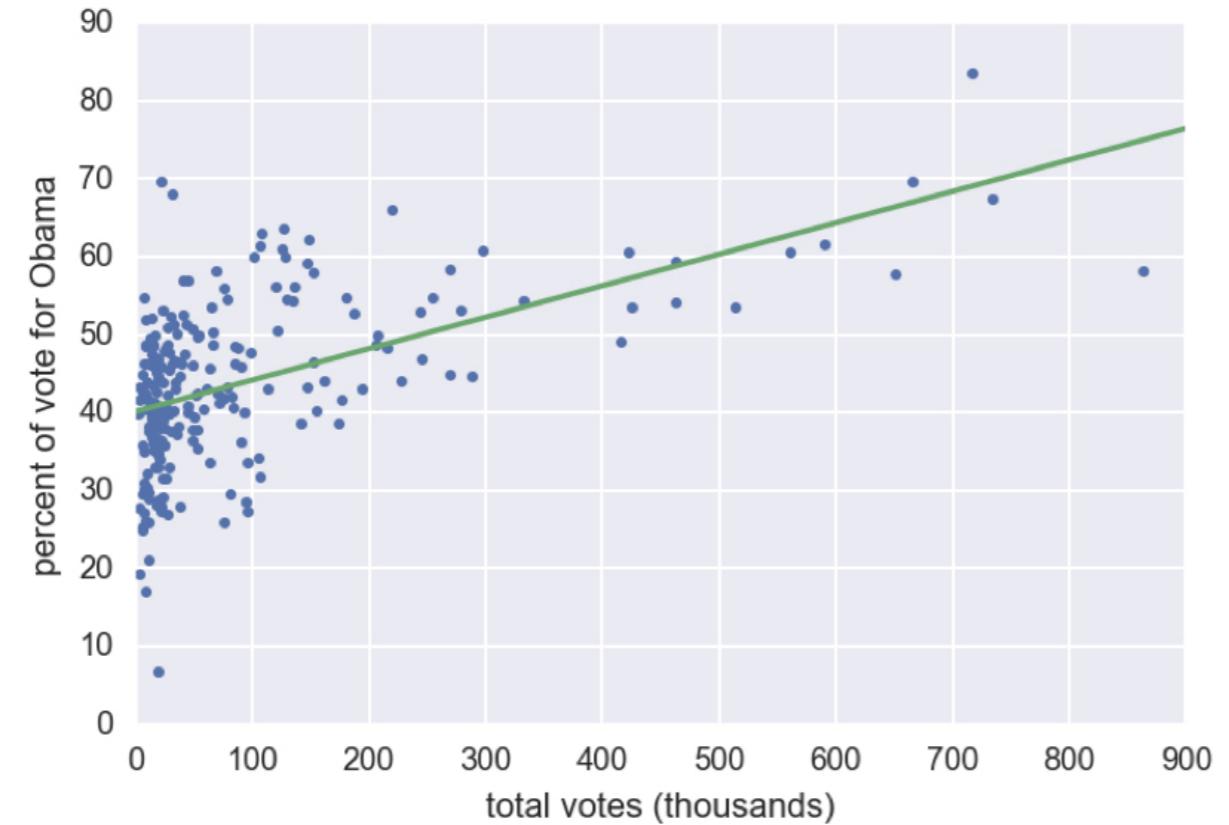
Lecturer at the California Institute of
Technology

2008 US swing state election results



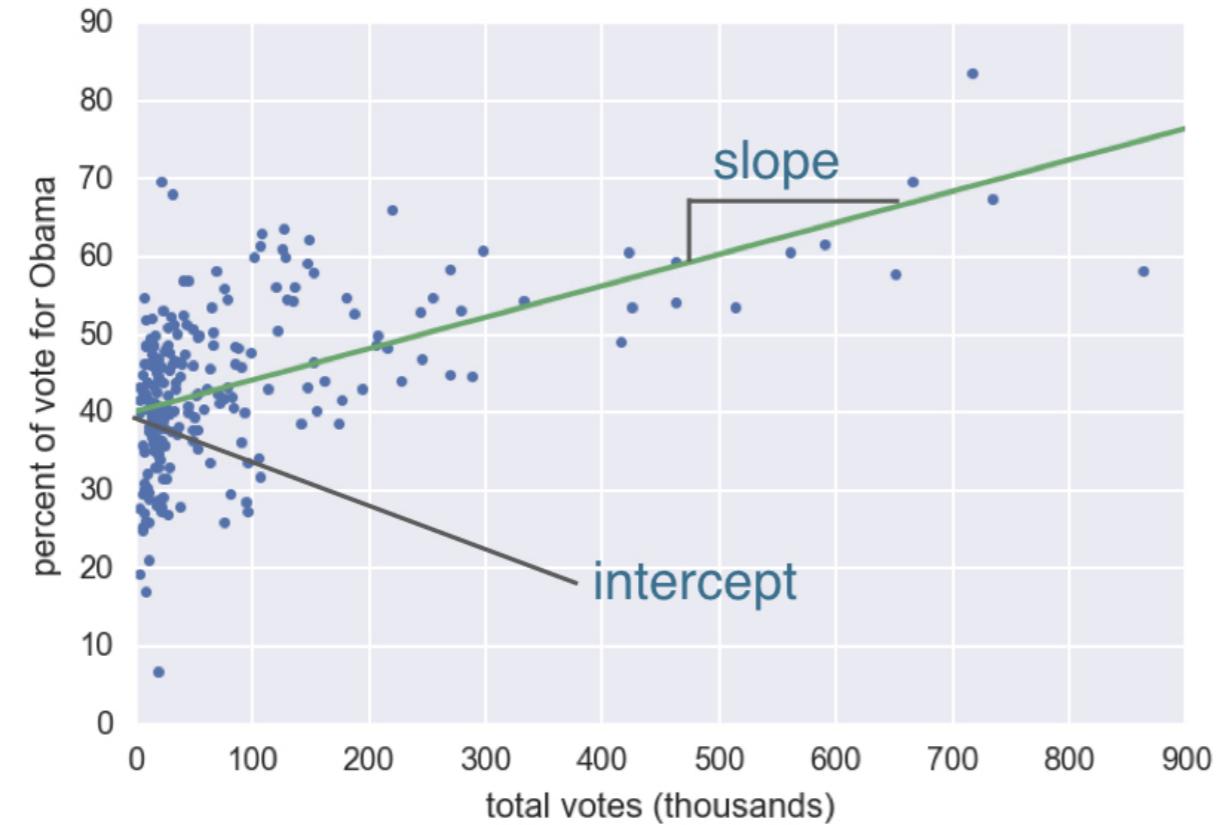
¹ Data retrieved from Data.gov (<https://www.data.gov/>)

2008 US swing state election results



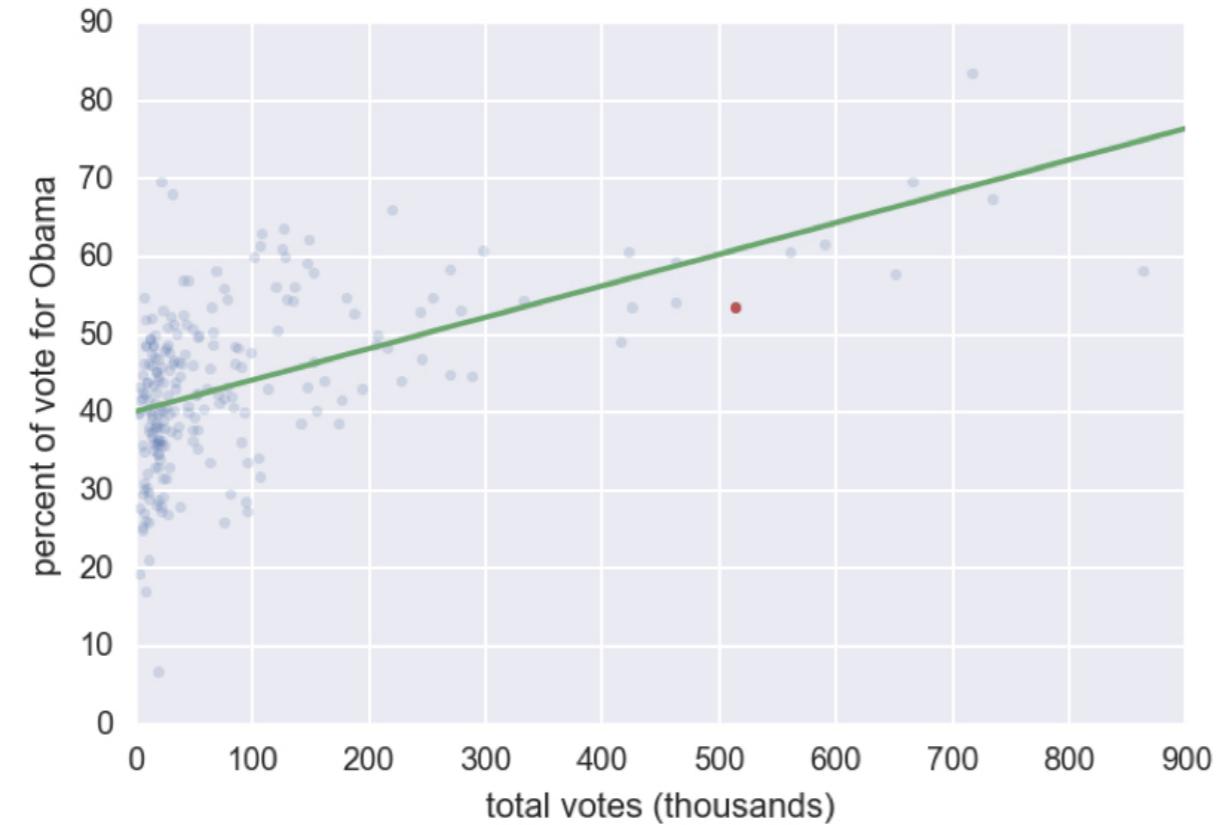
¹ Data retrieved from Data.gov (<https://www.data.gov/>)

2008 US swing state election results



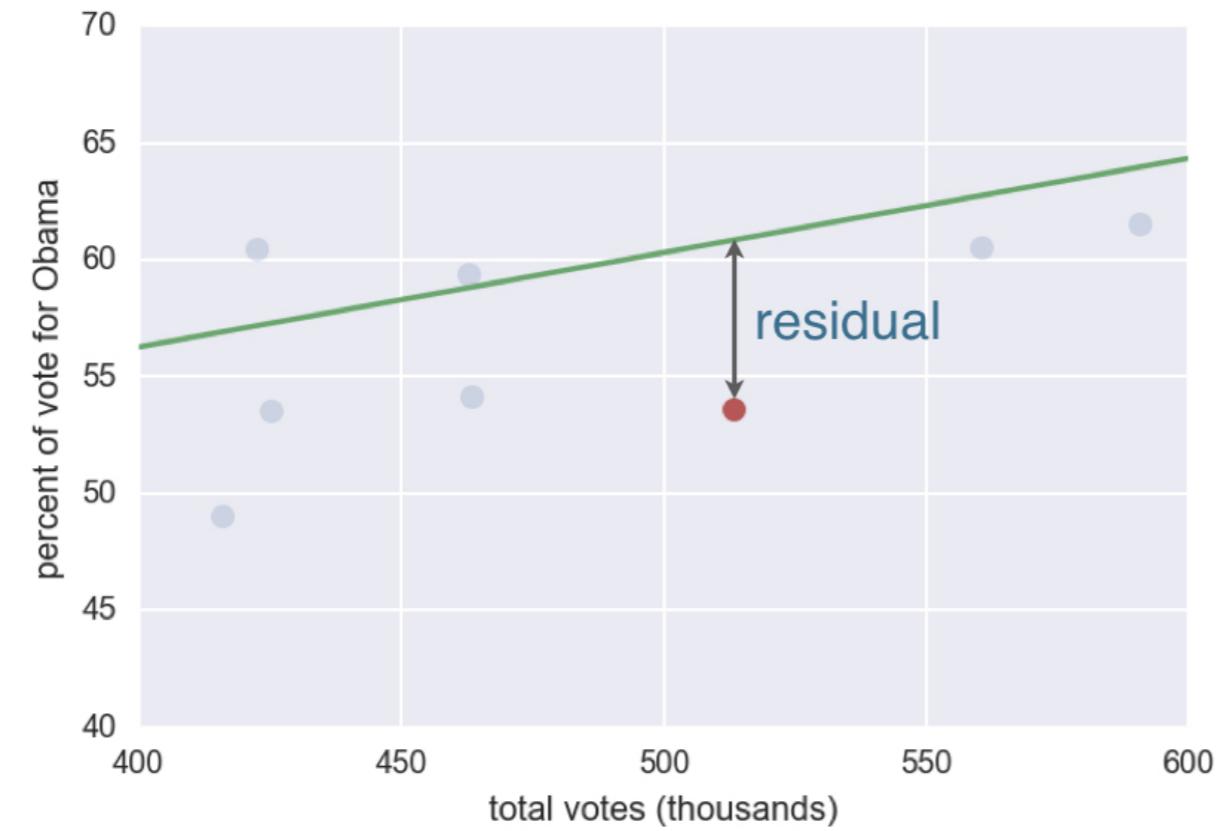
¹ Data retrieved from Data.gov (<https://www.data.gov/>)

2008 US swing state election results



¹ Data retrieved from Data.gov (<https://www.data.gov/>)

Residuals



¹ Data retrieved from Data.gov (<https://www.data.gov/>)

Least squares

- The process of finding the parameters for which the sum of the squares of the residuals is minimal

Least squares with np.polyfit()

```
slope, intercept = np.polyfit(total_votes,  
                             dem_share, 1)
```

slope

```
4.037071700946555e-05
```

intercept

```
40.113911968641744
```

Let's practice!

STATISTICAL THINKING IN PYTHON (PART 2)

The importance of EDA: Anscombe's quartet

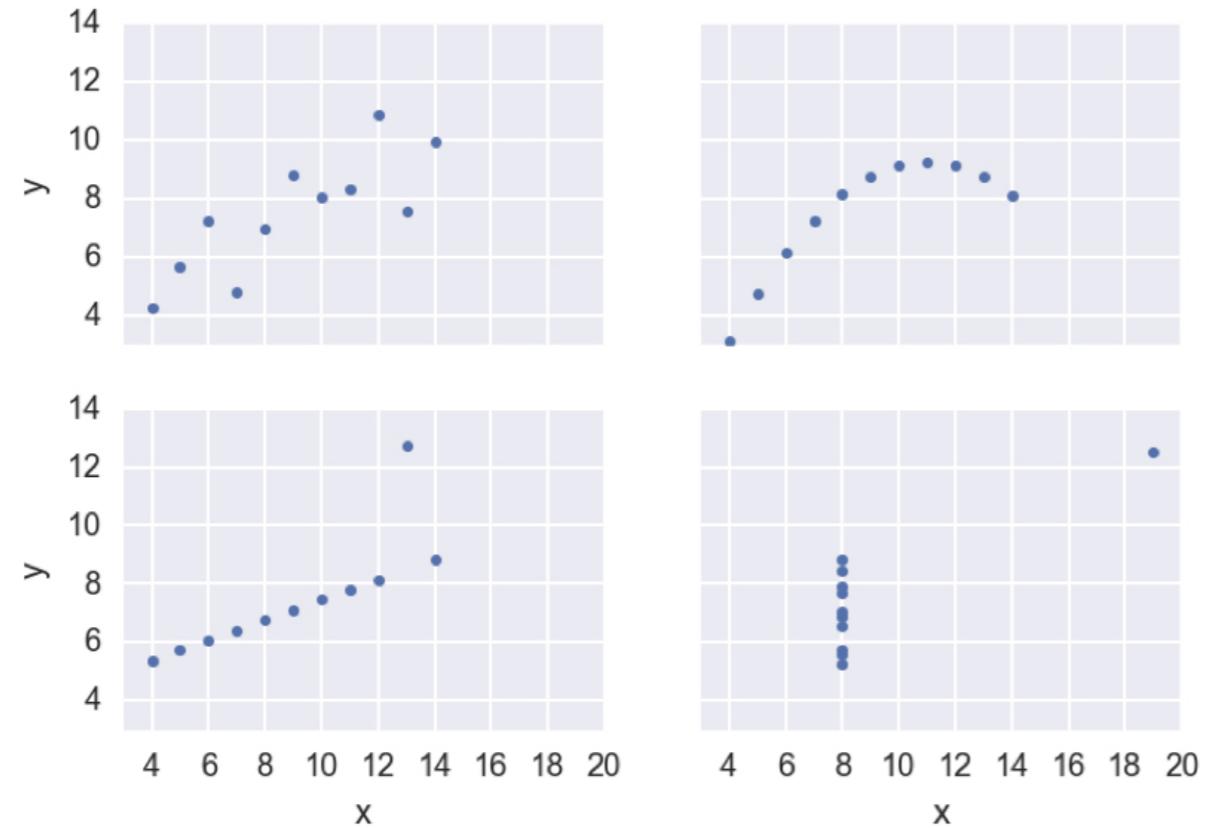
STATISTICAL THINKING IN PYTHON (PART 2)

Justin Bois

Lecturer at the California Institute of
Technology

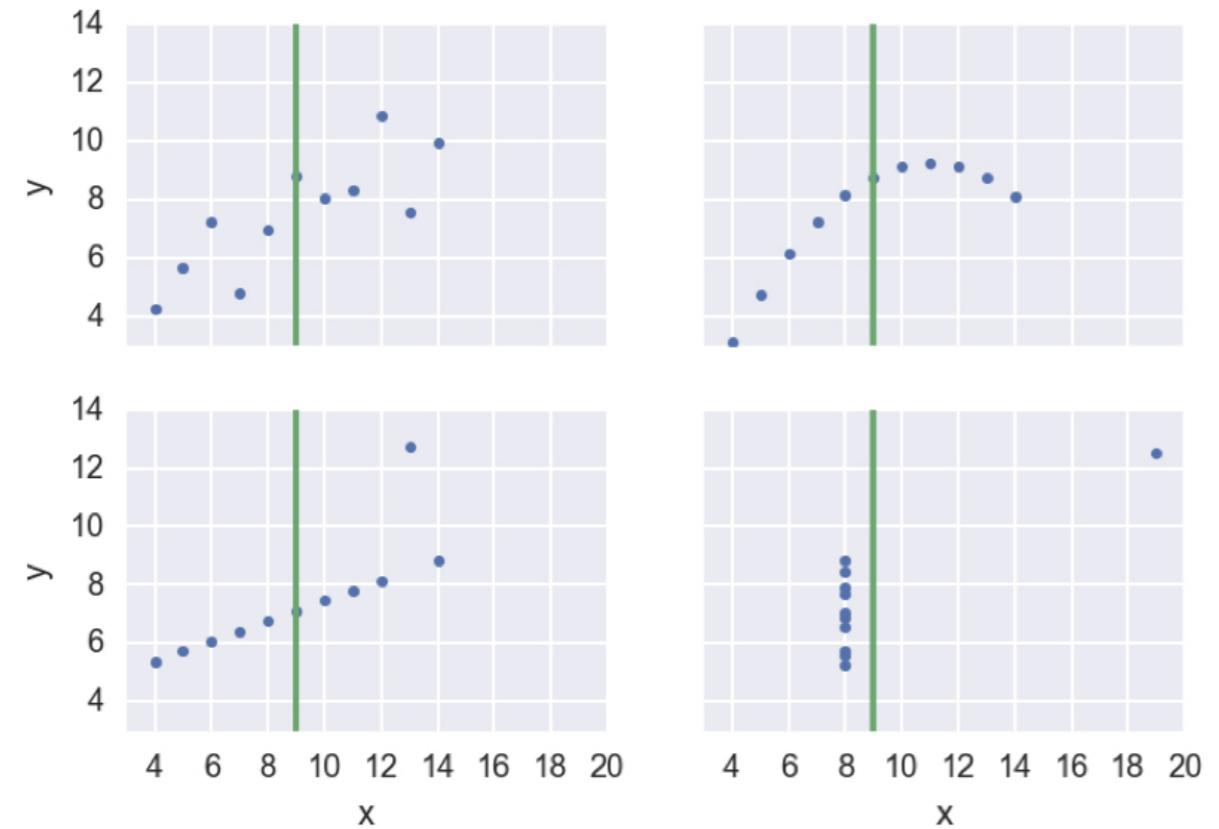


Anscombe's quartet



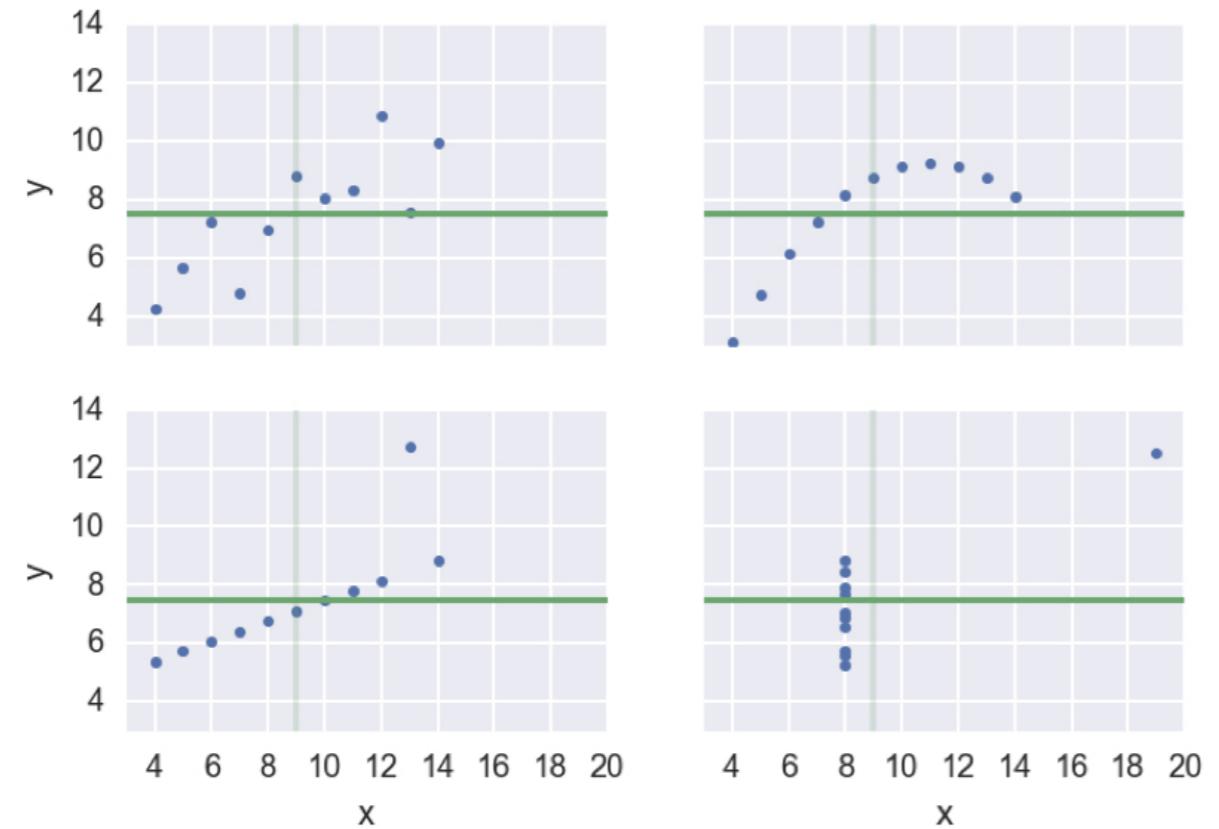
¹ Data: Anscombe, *The American Statistician*, 1973

Anscombe's quartet



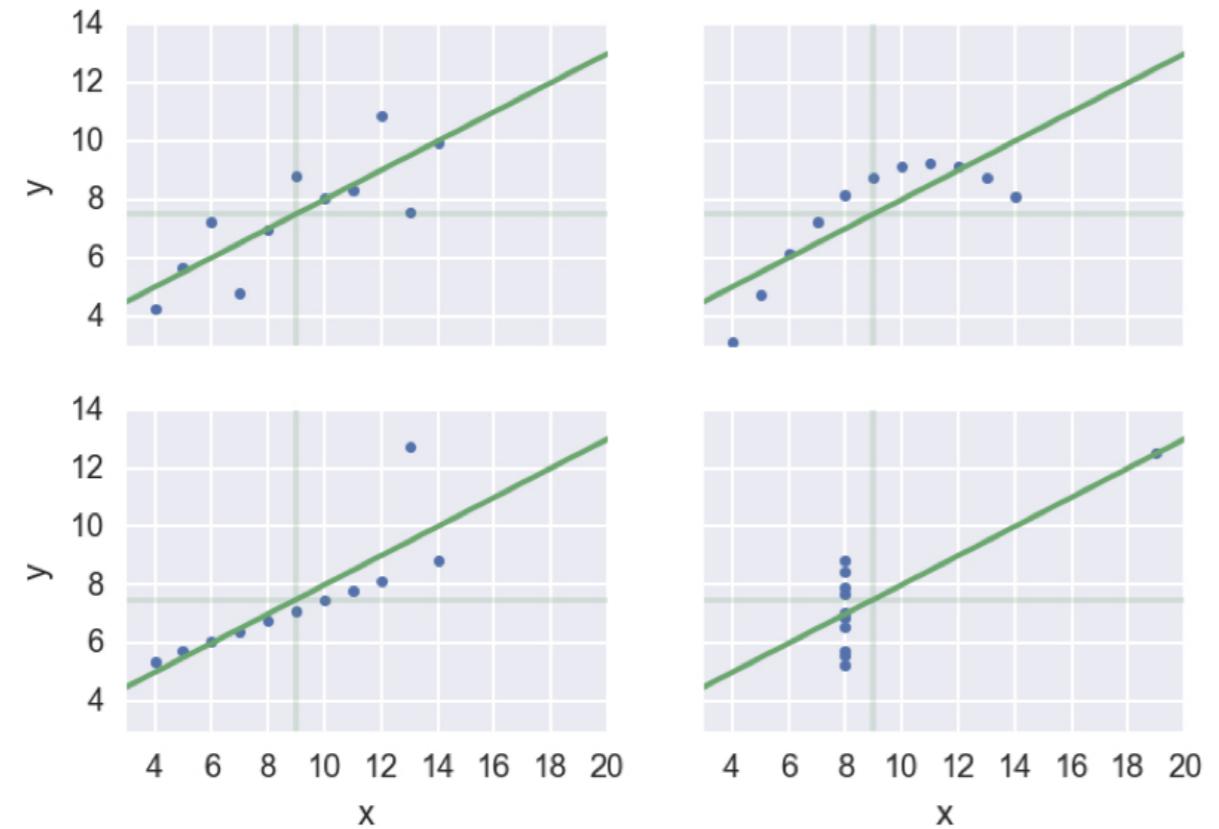
¹ Data: Anscombe, *The American Statistician*, 1973

Anscombe's quartet



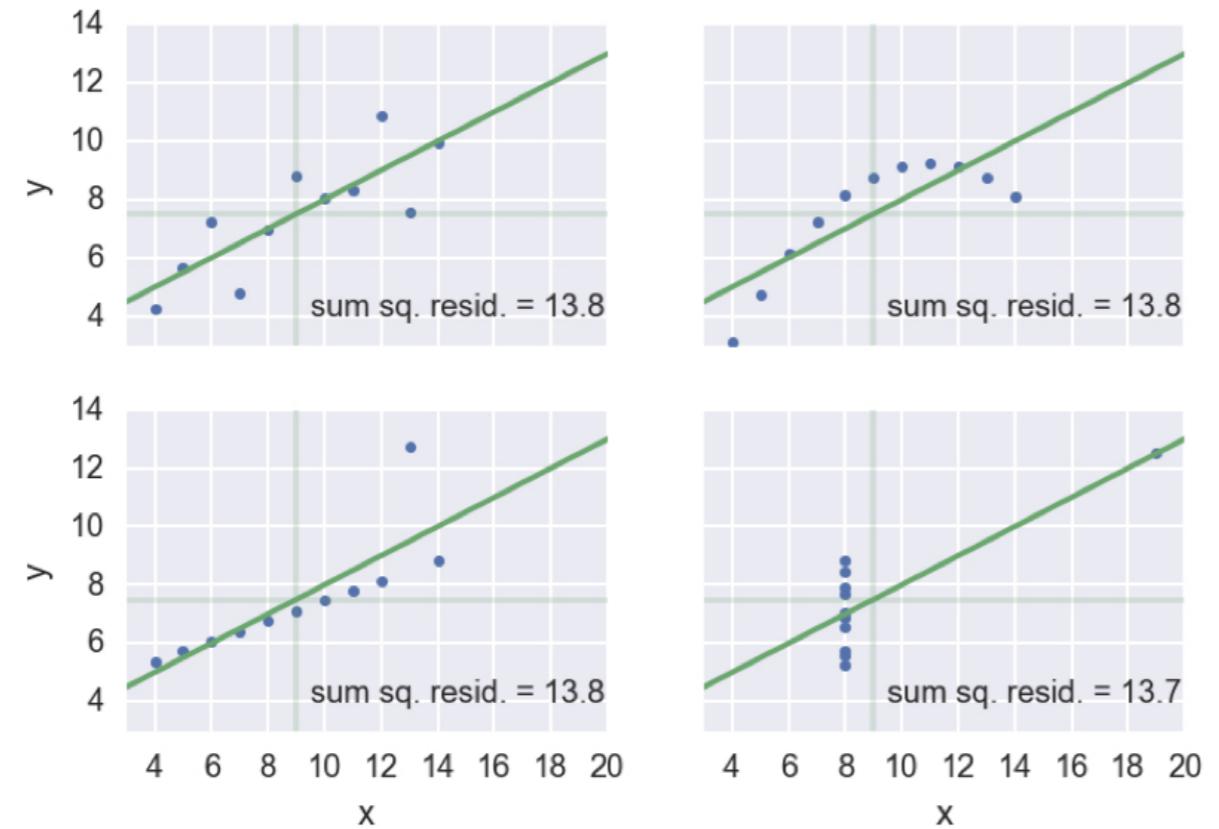
¹ Data: Anscombe, *The American Statistician*, 1973

Anscombe's quartet



¹ Data: Anscombe, *The American Statistician*, 1973

Anscombe's quartet

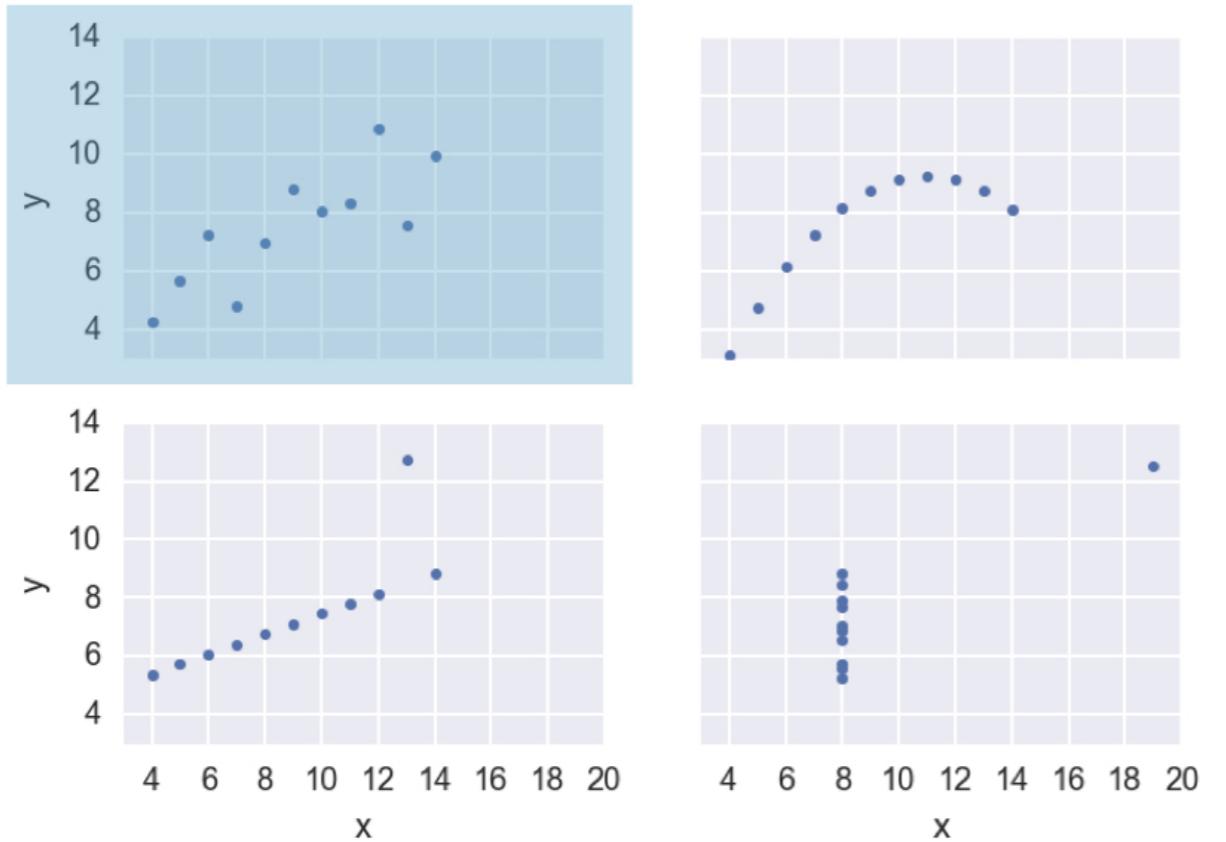


¹ Data: Anscombe, *The American Statistician*, 1973

Look before you leap!

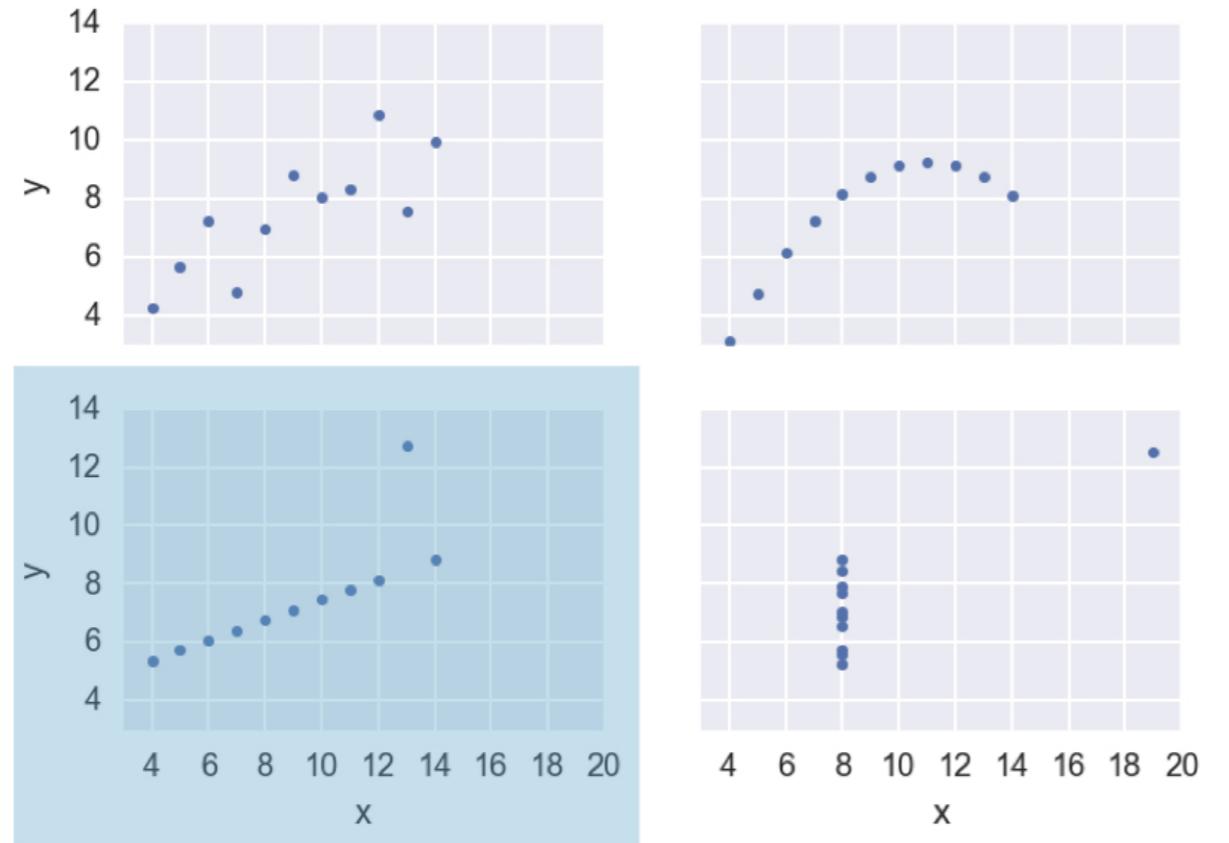
- Do graphical EDA first

Anscombe's quartet



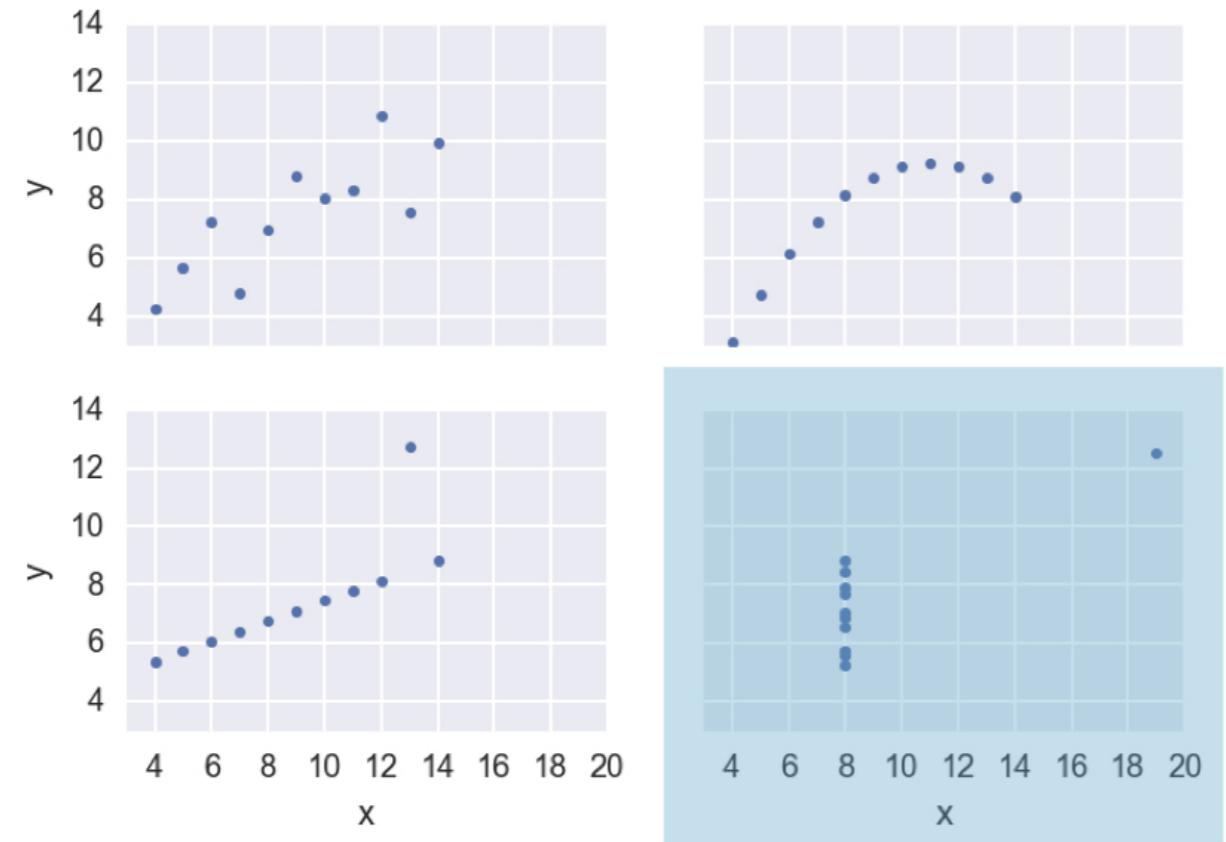
¹ Data: Anscombe, *The American Statistician*, 1973

Anscombe's quartet



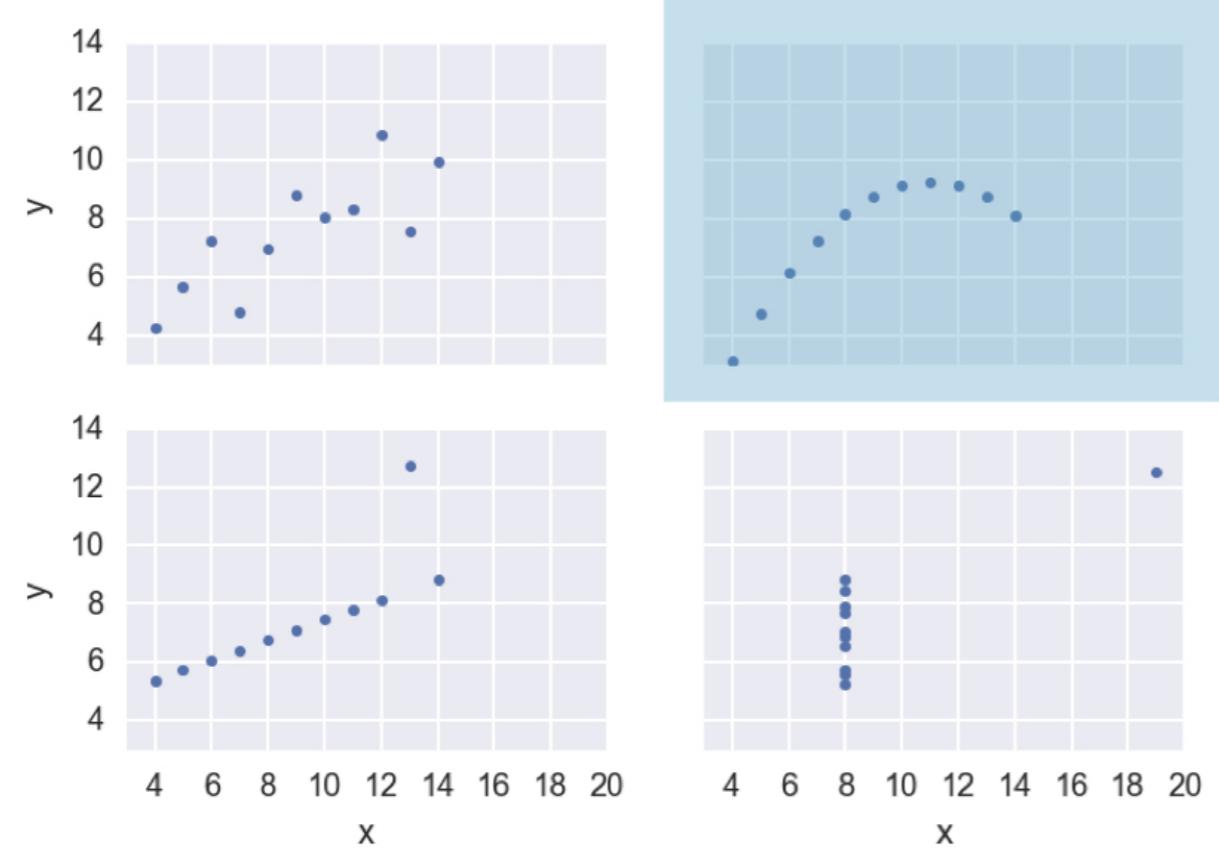
¹ Data: Anscombe, *The American Statistician*, 1973

Anscombe's quartet



¹ Data: Anscombe, *The American Statistician*, 1973

Anscombe's quartet



¹ Data: Anscombe, *The American Statistician*, 1973

Let's practice!

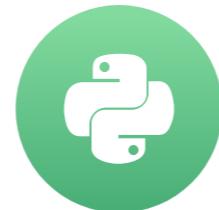
STATISTICAL THINKING IN PYTHON (PART 2)

Generating bootstrap replicates

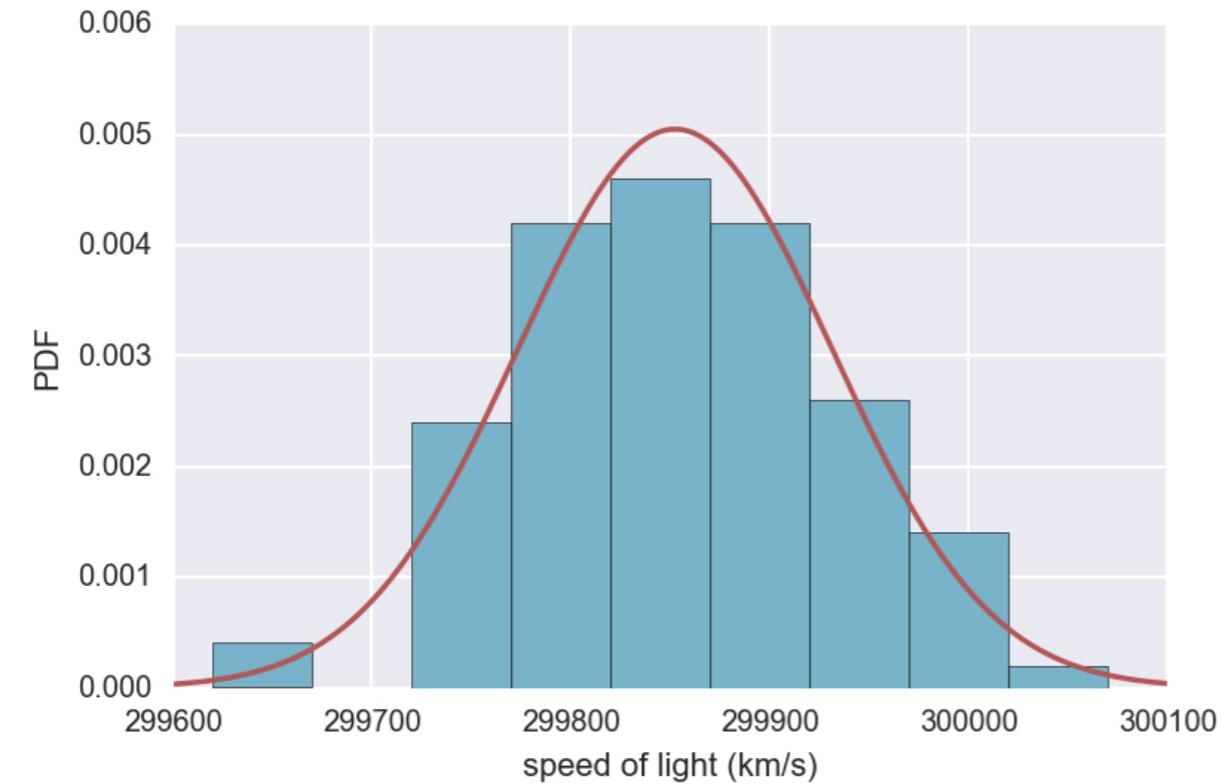
STATISTICAL THINKING IN PYTHON (PART 2)

Justin Bois

Lecturer at the California Institute of
Technology



Michelson's speed of light measurements



¹ Data: Michelson, 1880

Resampling an array

Data:

[23.3, 27.1, 24.3, 25.3, 26.0]

Mean = 25.2

Resampled data:

[, , , ,]

Resampling an array

Data:

[23.3, 27.1, 24.3, 25.3, 26.0]

Mean = 25.2

Resampled data:

[, , , ,]

Resampling an array

Data:

```
[ 23.3, , 24.3, 25.3, 26.0 ]
```

Mean = 25.2

Resampled data:

```
[ 27.1, , , , ]
```

Resampling an array

Data:

[23.3, 27.1, 24.3, 25.3, 26.0]

Mean = 25.2

Resampled data:

[27.1, , , ,]

Resampling an array

Data:

[23.3, 27.1, 24.3, 25.3, 26.0]

Mean = 25.2

Resampled data:

[27.1, 26.0, , ,]

Resampling an array

Data:

```
[23.3, 27.1, 24.3, 25.3, 26.0]
```

Mean = 25.2

Resampled data:

```
[27.1, 26.0, , , ]
```

Resampling an array

Data:

[23.3, 27.1, 24.3, 25.7, 26.0]

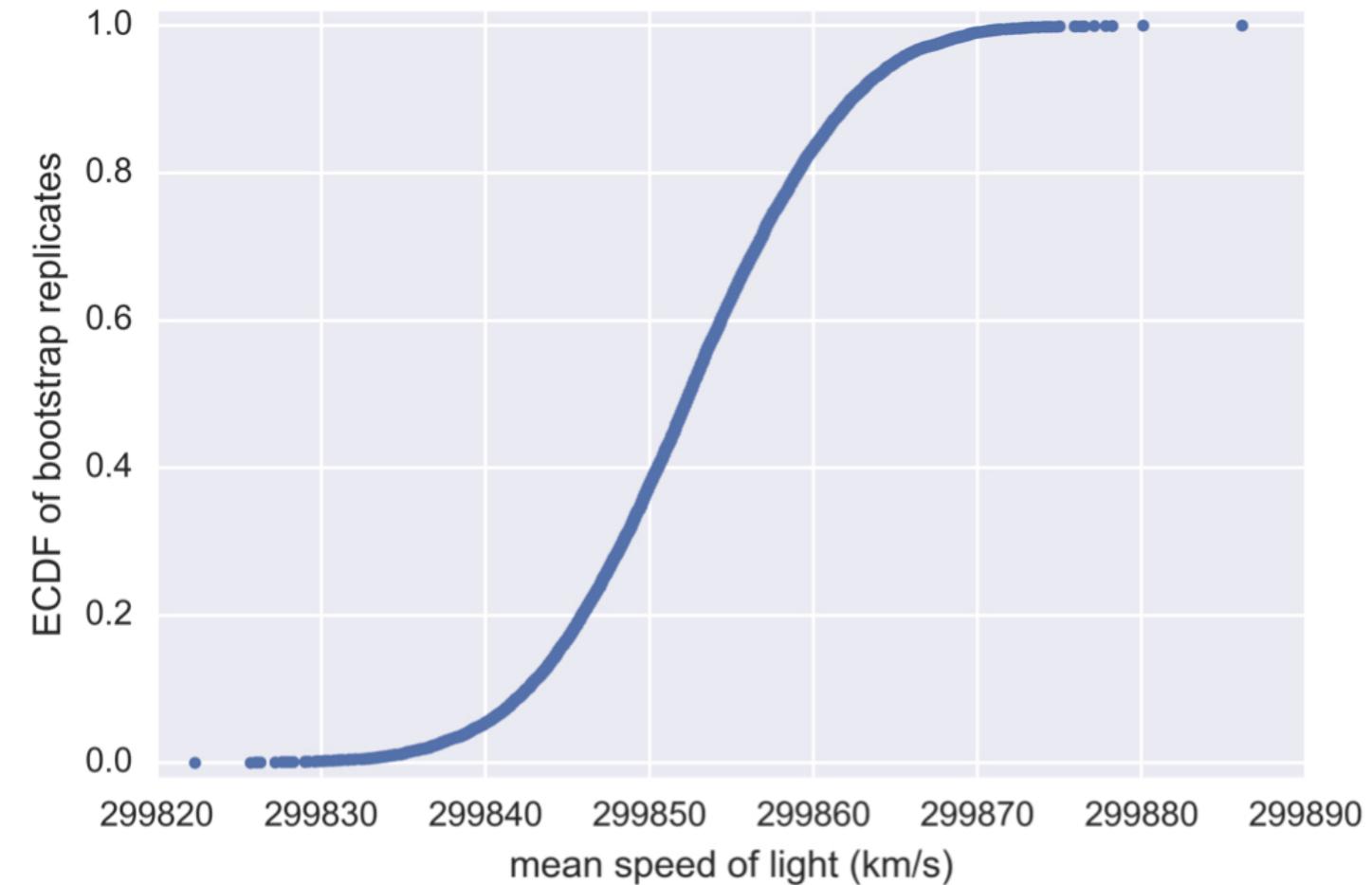
Mean = 25.2

Resampled data:

[27.1, 26.0, 23.3, 25.7, 23.3]

Mean = 25.08

Mean of resampled Michelson measurements



Bootstrapping

- The use of resampled data to perform statistical inference

Bootstrap sample

- A resampled array of the data

Bootstrap replicate

- A statistic computed from a resampled array

Resampling engine: np.random.choice()

```
import numpy as np  
np.random.choice([1,2,3,4,5], size=5)
```

```
array([5, 3, 5, 5, 2])
```

Computing a bootstrap replicate

```
bs_sample = np.random.choice(michelson_speed_of_light,  
                             size=100)  
  
np.mean(bs_sample)
```

```
299847.79999999999
```

```
np.median(bs_sample)
```

```
299845.0
```

```
np.std(bs_sample)
```

```
83.564286025729331
```

Let's practice!

STATISTICAL THINKING IN PYTHON (PART 2)

Bootstrap confidence intervals

STATISTICAL THINKING IN PYTHON (PART 2)



Justin Bois

Lecturer at the California Institute of
Technology

Bootstrap replicate function

```
def bootstrap_replicate_1d(data, func):  
    """Generate bootstrap replicate of 1D data."""  
    bs_sample = np.random.choice(data, len(data))  
    return func(bs_sample)  
  
bootstrap_replicate_1d(michelson_speed_of_light, np.mean)
```

```
299859.2000000001
```

```
bootstrap_replicate_1d(michelson_speed_of_light, np.mean)
```

```
299855.7000000001
```

```
bootstrap_replicate_1d(michelson_speed_of_light, np.mean)
```

```
299850.2999999999
```

Many bootstrap replicates

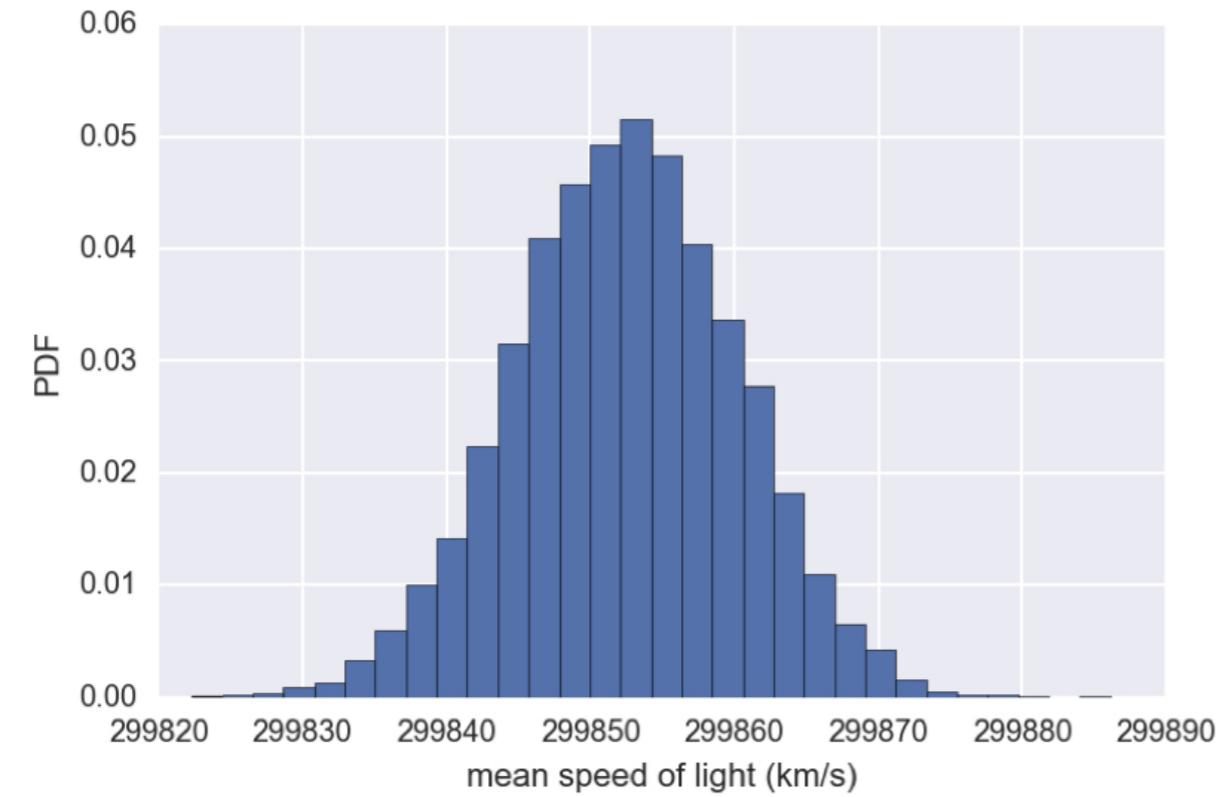
```
bs_replicates = np.empty(10000)

for i in range(10000):
    bs_replicates[i] = bootstrap_replicate_1d(
        michelson_speed_of_light, np.mean)
```

Plotting a histogram of bootstrap replicates

```
_ = plt.hist(bs_replicates, bins=30, normed=True)
_= plt.xlabel('mean speed of light (km/s)')
_= plt.ylabel('PDF')
plt.show()
```

Bootstrap estimate of the mean



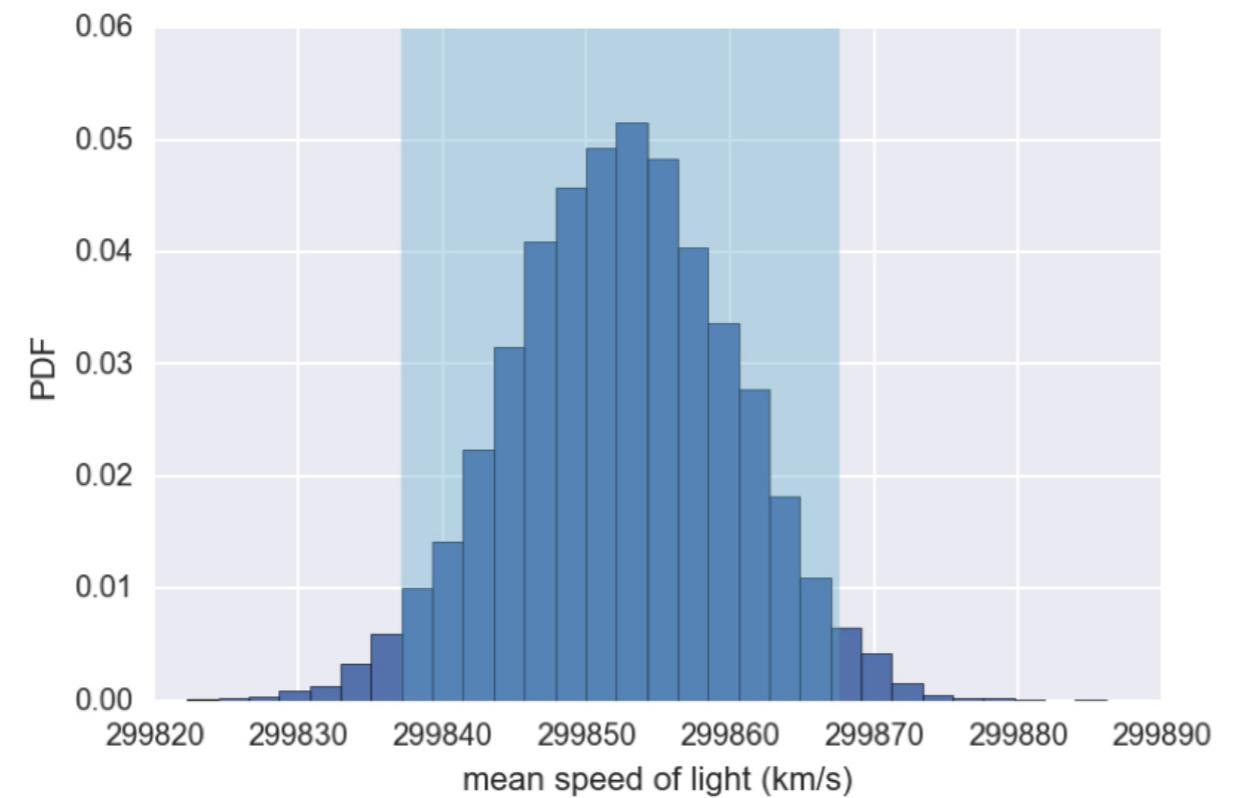
Confidence interval of a statistic

- If we repeated measurements over and over again, $p\%$ of the observed values would lie within the $p\%$ confidence interval.

Bootstrap confidence interval

```
conf_int = np.percentile(bs_replicates, [2.5, 97.5])
```

```
array([ 299837.,  299868.])
```

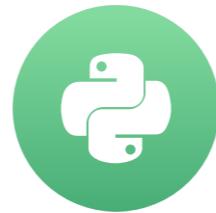


Let's practice!

STATISTICAL THINKING IN PYTHON (PART 2)

Pairs bootstrap

STATISTICAL THINKING IN PYTHON (PART 2)



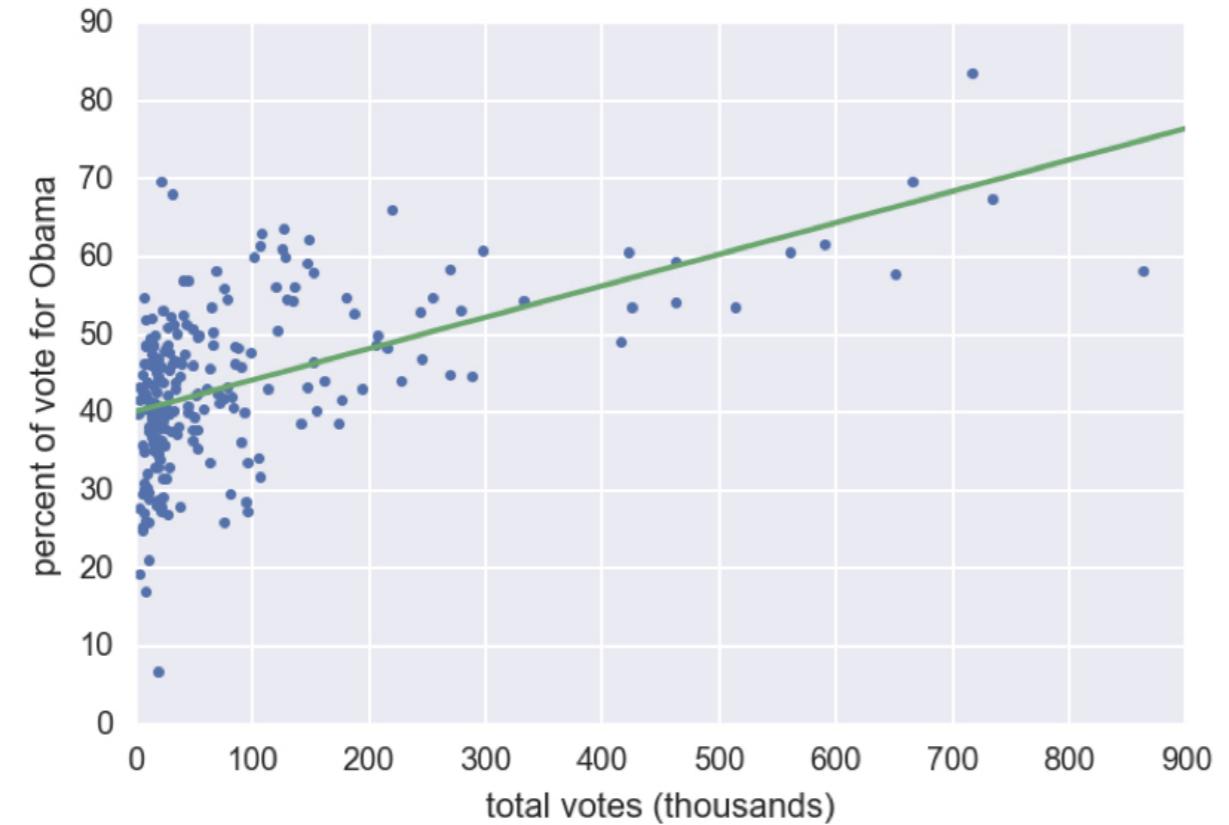
Justin Bois

Lecturer at the California Institute of
Technology

Nonparametric inference

- Make no assumptions about the model or probability distribution underlying the data

2008 US swing state election results



¹ Data retrieved from Data.gov (<https://www.data.gov/>)

Pairs bootstrap for linear regression

- Resample data in pairs
- Compute slope and intercept from resampled data
- Each slope and intercept is a bootstrap replicate
- Compute confidence intervals from percentiles of bootstrap replicates

Generating a pairs bootstrap sample

```
np.arange(7)
```

```
array([0, 1, 2, 3, 4, 5, 6])
```

```
inds = np.arange(len(total_votes))

bs_inds = np.random.choice(inds, len(inds))

bs_total_votes = total_votes[bs_inds]
bs_dem_share = dem_share[bs_inds]
```

Computing a pairs bootstrap replicate

```
bs_slope, bs_intercept = np.polyfit(bs_total_votes,  
                                    bs_dem_share, 1)
```

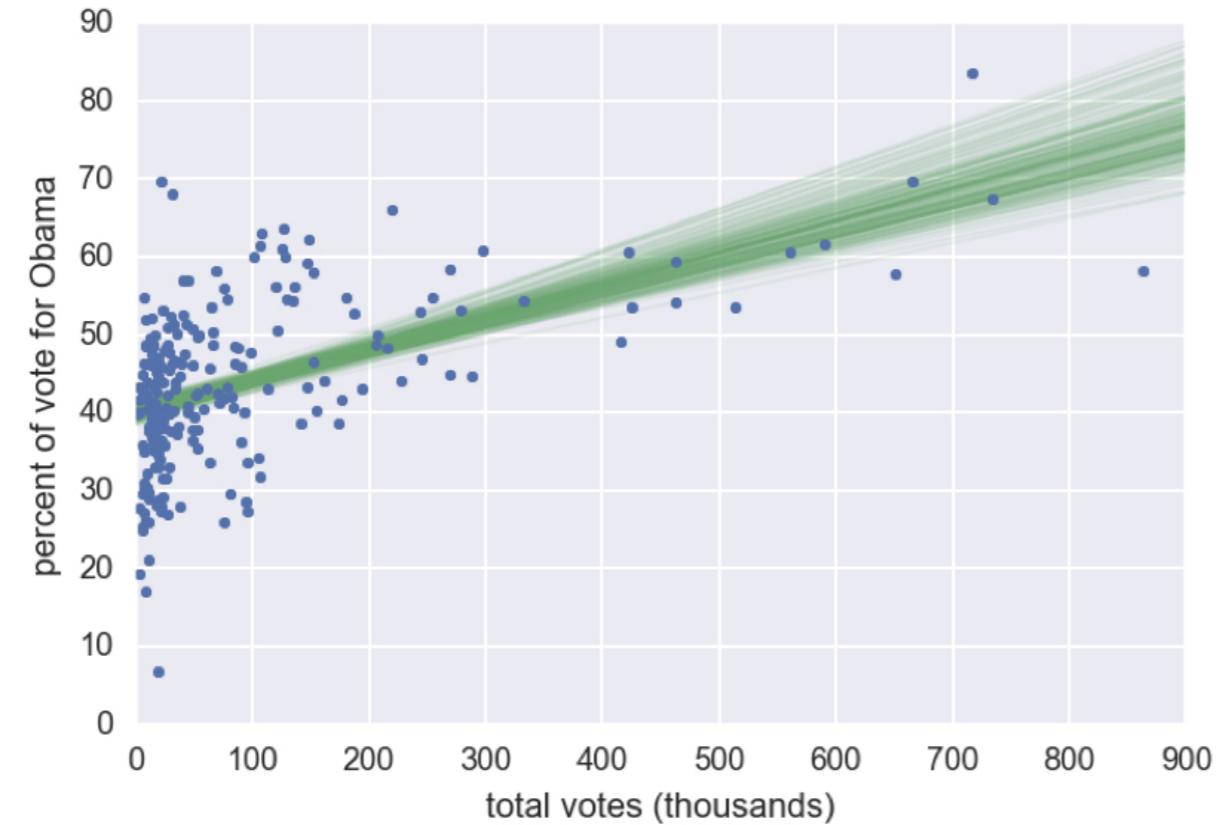
```
bs_slope, bs_intercept
```

```
(3.9053605692223672e-05, 40.387910131803025)
```

```
np.polyfit(total_votes, dem_share, 1) # fit of original
```

```
array([ 4.03707170e-05, 4.01139120e+01])
```

2008 US swing state election results



¹ Data retrieved from Data.gov (<https://www.data.gov/>)

Let's practice!

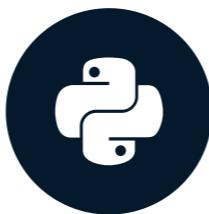
STATISTICAL THINKING IN PYTHON (PART 2)

Formulating and simulating a hypothesis

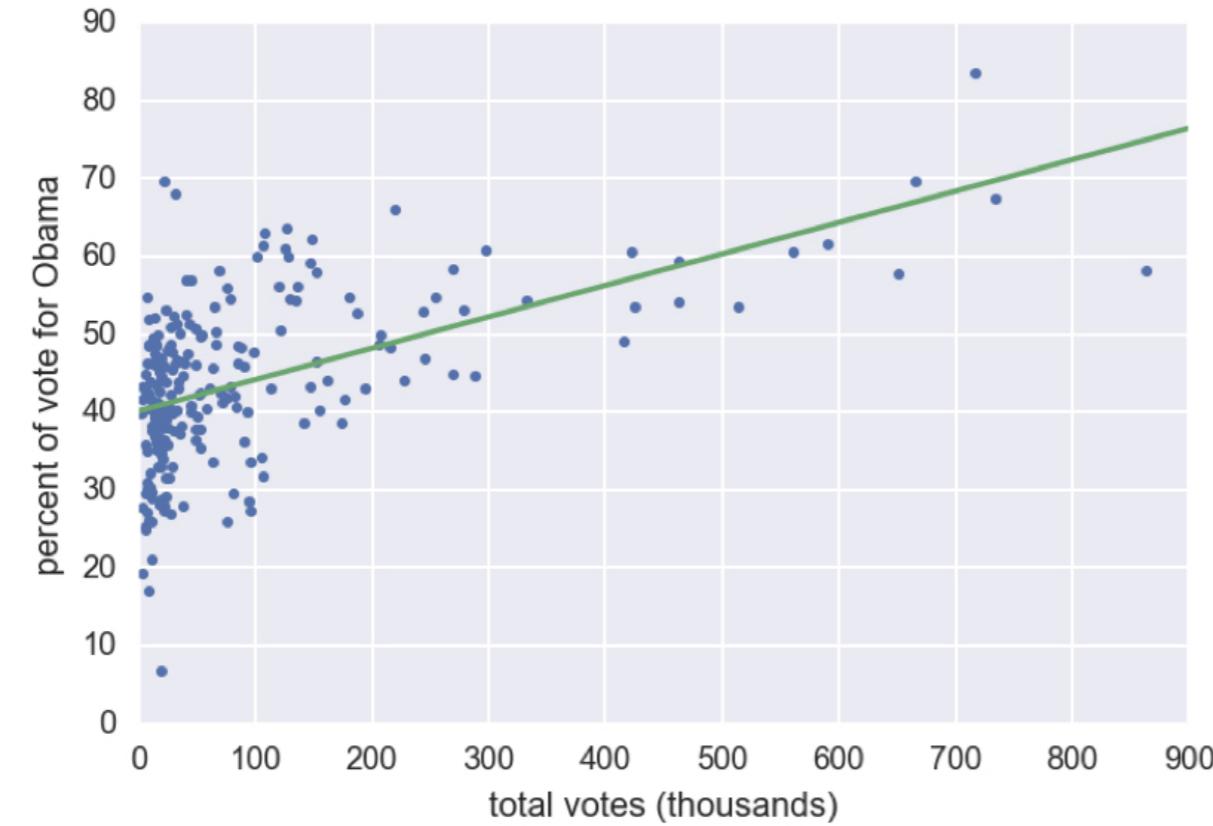
STATISTICAL THINKING IN PYTHON (PART 2)

Justin Bois

Lecturer at the California Institute of
Technology



2008 US swing state election results



¹ Data retrieved from Data.gov (<https://www.data.gov/>)



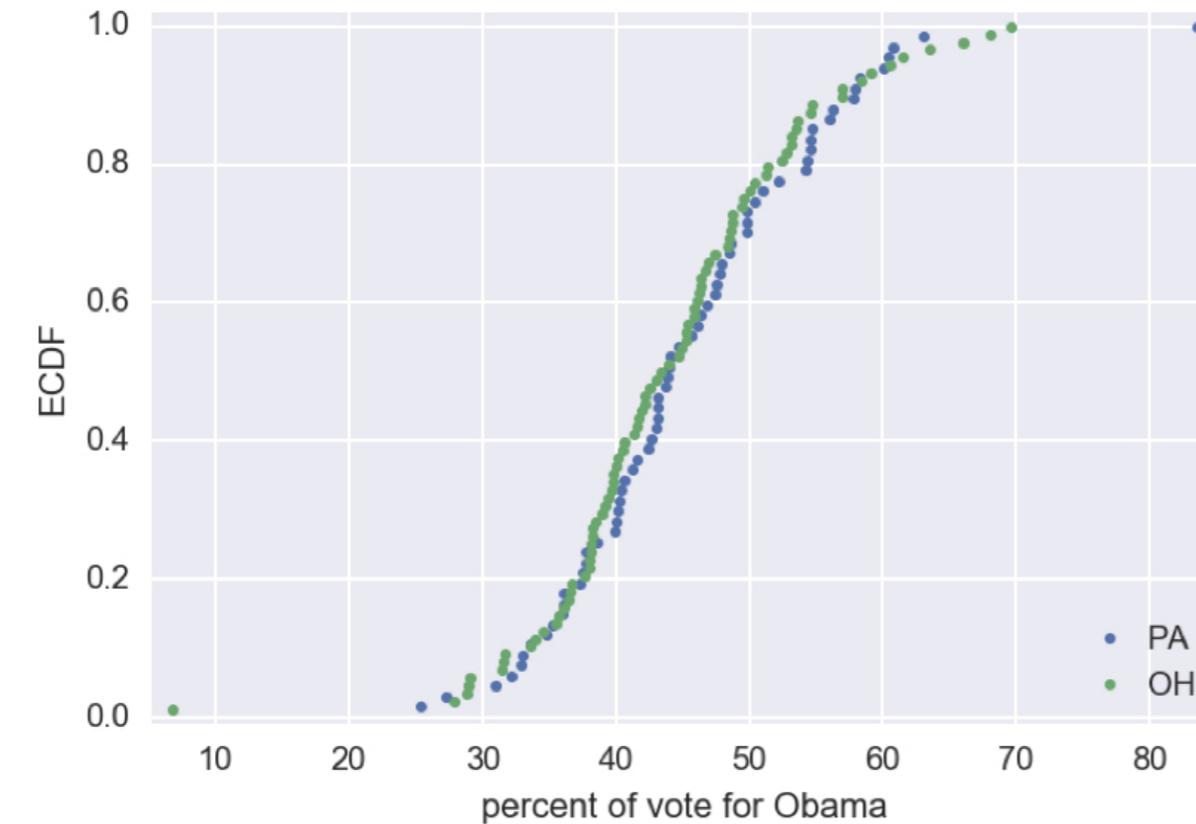
Hypothesis testing

- Assessment of how reasonable the observed data are assuming a hypothesis is true

Null hypothesis

- Another name for the hypothesis you are testing

ECDFs of swing state election results



¹ Data retrieved from Data.gov (<https://www.data.gov/>)

Percent vote for Obama

| | PA | OH | PA – OH difference |
|--------------------|-------|-------|-----------------------|
| mean | 45.5% | 44.3% | 1.2% |
| median | 44.0% | 43.7% | 0.4% |
| standard deviation | 9.8% | 9.9% | -0.1% |

¹ Data retrieved from Data.gov (<https://www.data.gov/>)

Simulating the hypothesis

| | | | | | | | |
|--------|--------|--------|--------|--------|--------|--------|--------|
| 60.08, | 40.64, | 36.07, | 41.21, | 31.04, | 43.78, | 44.08, | 46.85, |
| 44.71, | 46.15, | 63.10, | 52.20, | 43.18, | 40.24, | 39.92, | 47.87, |
| 37.77, | 40.11, | 49.85, | 48.61, | 38.62, | 54.25, | 34.84, | 47.75, |
| 43.82, | 55.97, | 58.23, | 42.97, | 42.38, | 36.11, | 37.53, | 42.65, |
| 50.96, | 47.43, | 56.24, | 45.60, | 46.39, | 35.22, | 48.56, | 32.97, |
| 57.88, | 36.05, | 37.72, | 50.36, | 32.12, | 41.55, | 54.66, | 57.81, |
| 54.58, | 32.88, | 54.37, | 40.45, | 47.61, | 60.49, | 43.11, | 27.32, |
| 44.03, | 33.56, | 37.26, | 54.64, | 43.12, | 25.34, | 49.79, | 83.56, |
| 40.09, | 60.81, | 49.81, | 56.94, | 50.46, | 65.99, | 45.88, | 42.23, |
| 45.26, | 57.01, | 53.61, | 59.10, | 61.48, | 43.43, | 44.69, | 54.59, |
| 48.36, | 45.89, | 48.62, | 43.92, | 38.23, | 28.79, | 63.57, | 38.07, |
| 40.18, | 43.05, | 41.56, | 42.49, | 36.06, | 52.76, | 46.07, | 39.43, |
| 39.26, | 47.47, | 27.92, | 38.01, | 45.45, | 29.07, | 28.94, | 51.28, |
| 50.10, | 39.84, | 36.43, | 35.71, | 31.47, | 47.01, | 40.10, | 48.76, |
| 31.56, | 39.86, | 45.31, | 35.47, | 51.38, | 46.33, | 48.73, | 41.77, |
| 41.32, | 48.46, | 53.14, | 34.01, | 54.74, | 40.67, | 38.96, | 46.29, |
| 38.25, | 6.80, | 31.75, | 46.33, | 44.90, | 33.57, | 38.10, | 39.67, |
| 40.47, | 49.44, | 37.62, | 36.71, | 46.73, | 42.20, | 53.16, | 52.40, |
| 58.36, | 68.02, | 38.53, | 34.58, | 69.64, | 60.50, | 53.53, | 36.54, |
| 49.58, | 41.97, | 38.11 | | | | | |

Pennsylvania

Ohio

¹ Data retrieved from Data.gov (<https://www.data.gov/>)

Simulating the hypothesis

```
60.08, 40.64, 36.07, 41.21, 31.04, 43.78, 44.08, 46.85,  
44.71, 46.15, 63.10, 52.20, 43.18, 40.24, 39.92, 47.87,  
37.77, 40.11, 49.85, 48.61, 38.62, 54.25, 34.84, 47.75,  
43.82, 55.97, 58.23, 42.97, 42.38, 36.11, 37.53, 42.65,  
50.96, 47.43, 56.24, 45.60, 46.39, 35.22, 48.56, 32.97,  
57.88, 36.05, 37.72, 50.36, 32.12, 41.55, 54.66, 57.81,  
54.58, 32.88, 54.37, 40.45, 47.61, 60.49, 43.11, 27.32,  
44.03, 33.56, 37.26, 54.64, 43.12, 25.34, 49.79, 83.56,  
40.09, 60.81, 49.81, 56.94, 50.46, 65.99, 45.88, 42.23,  
45.26, 57.01, 53.61, 59.10, 61.48, 43.43, 44.69, 54.59,  
48.36, 45.89, 48.62, 43.92, 38.23, 28.79, 63.57, 38.07,  
40.18, 43.05, 41.56, 42.49, 36.06, 52.76, 46.07, 39.43,  
39.26, 47.47, 27.92, 38.01, 45.45, 29.07, 28.94, 51.28,  
50.10, 39.84, 36.43, 35.71, 31.47, 47.01, 40.10, 48.76,  
31.56, 39.86, 45.31, 35.47, 51.38, 46.33, 48.73, 41.77,  
41.32, 48.46, 53.14, 34.01, 54.74, 40.67, 38.96, 46.29,  
38.25, 6.80, 31.75, 46.33, 44.90, 33.57, 38.10, 39.67,  
40.47, 49.44, 37.62, 36.71, 46.73, 42.20, 53.16, 52.40,  
58.36, 68.02, 38.53, 34.58, 69.64, 60.50, 53.53, 36.54,  
49.58, 41.97, 38.11
```

¹ Data retrieved from Data.gov (<https://www.data.gov/>)

Simulating the hypothesis

```
59.10, 38.62, 51.38, 60.49, 6.80, 41.97, 48.56, 37.77,  
48.36, 54.59, 40.11, 57.81, 45.89, 83.56, 40.64, 46.07,  
28.79, 55.97, 33.57, 42.23, 48.61, 44.69, 39.67, 57.88,  
48.62, 54.66, 54.74, 48.46, 36.07, 43.92, 49.85, 53.53,  
48.76, 41.77, 36.54, 47.01, 52.76, 49.44, 34.58, 40.24,  
44.08, 46.29, 49.81, 69.64, 60.50, 27.32, 45.60, 63.10,  
35.71, 39.86, 40.67, 65.99, 50.46, 37.72, 50.96, 42.49,  
31.56, 38.23, 37.26, 41.21, 37.53, 46.85, 44.03, 41.32,  
45.88, 40.45, 32.12, 35.22, 49.79, 43.12, 43.18, 45.45,  
25.34, 46.73, 44.90, 56.94, 58.23, 39.84, 36.05, 43.05,  
38.25, 40.47, 31.04, 54.25, 46.15, 57.01, 52.20, 47.75,  
36.06, 47.61, 51.28, 43.43, 42.97, 38.01, 54.64, 45.26,  
47.47, 34.84, 49.58, 48.73, 29.07, 54.58, 27.92, 34.01,  
38.07, 31.47, 36.11, 39.26, 41.56, 52.40, 40.18, 47.87,  
46.33, 46.39, 43.11, 38.53, 33.56, 42.65, 68.02, 35.47,  
40.09, 36.43, 36.71, 60.08, 50.36, 39.43, 28.94, 58.36,  
42.20, 47.43, 44.71, 43.78, 39.92, 37.62, 63.57, 53.61,  
40.10, 46.33, 53.16, 32.88, 38.96, 41.55, 56.24, 38.11,  
42.38, 38.10, 43.82, 45.31, 60.81, 54.37, 53.14, 32.97,  
61.48, 50.10, 31.75
```

¹ Data retrieved from Data.gov (<https://www.data.gov/>)

Simulating the hypothesis

| | | | | | | | |
|--------|--------|--------|--------|--------|--------|--------|--------|
| 59.10, | 38.62, | 51.38, | 60.49, | 6.80, | 41.97, | 48.56, | 37.77, |
| 48.36, | 54.59, | 40.11, | 57.81, | 45.89, | 83.56, | 40.64, | 46.07, |
| 28.79, | 55.97, | 33.57, | 42.23, | 48.61, | 44.69, | 39.67, | 57.88, |
| 48.62, | 54.66, | 54.74, | 48.46, | 36.07, | 43.92, | 49.85, | 53.53, |
| 48.76, | 41.77, | 36.54, | 47.01, | 52.76, | 49.44, | 34.58, | 40.24, |
| 44.08, | 46.29, | 49.81, | 69.64, | 60.50, | 27.32, | 45.60, | 63.10, |
| 35.71, | 39.86, | 40.67, | 65.99, | 50.46, | 37.72, | 50.96, | 42.49, |
| 31.56, | 38.23, | 37.26, | 41.21, | 37.53, | 46.85, | 44.03, | 41.32, |
| 45.88, | 40.45, | 32.12, | 35.22, | 49.79, | 43.12, | 43.18, | 45.45, |
| 25.34, | 46.73, | 44.90, | 56.94, | 58.23, | 39.84, | 36.05, | 43.05, |
| 38.25, | 40.47, | 31.04, | 54.25, | 46.15, | 57.01, | 52.20, | 47.75, |
| 36.06, | 47.61, | 51.28, | 43.43, | 42.97, | 38.01, | 54.64, | 45.26, |
| 47.47, | 34.84, | 49.58, | 48.73, | 29.07, | 54.58, | 27.92, | 34.01, |
| 38.07, | 31.47, | 36.11, | 39.26, | 41.56, | 52.40, | 40.18, | 47.87, |
| 46.33, | 46.39, | 43.11, | 38.53, | 33.56, | 42.65, | 68.02, | 35.47, |
| 40.09, | 36.43, | 36.71, | 60.08, | 50.36, | 39.43, | 28.94, | 58.36, |
| 42.20, | 47.43, | 44.71, | 43.78, | 39.92, | 37.62, | 63.57, | 53.61, |
| 40.10, | 46.33, | 53.16, | 32.88, | 38.96, | 41.55, | 56.24, | 38.11, |
| 42.38, | 38.10, | 43.82, | 45.31, | 60.81, | 54.37, | 53.14, | 32.97, |
| 61.48, | 50.10, | 31.75 | | | | | |

"Pennsylvania"

"Ohio"

Permutation

- Random reordering of entries in an array

Generating a permutation sample

```
import numpy as np  
dem_share_both = np.concatenate(  
    (dem_share_PA, dem_share_OH))  
dem_share_perm = np.random.permutation(dem_share_both)  
perm_sample_PA = dem_share_perm[:len(dem_share_PA)]  
perm_sample_OH = dem_share_perm[len(dem_share_PA):]
```

Let's practice!

STATISTICAL THINKING IN PYTHON (PART 2)

Test statistics and p-values

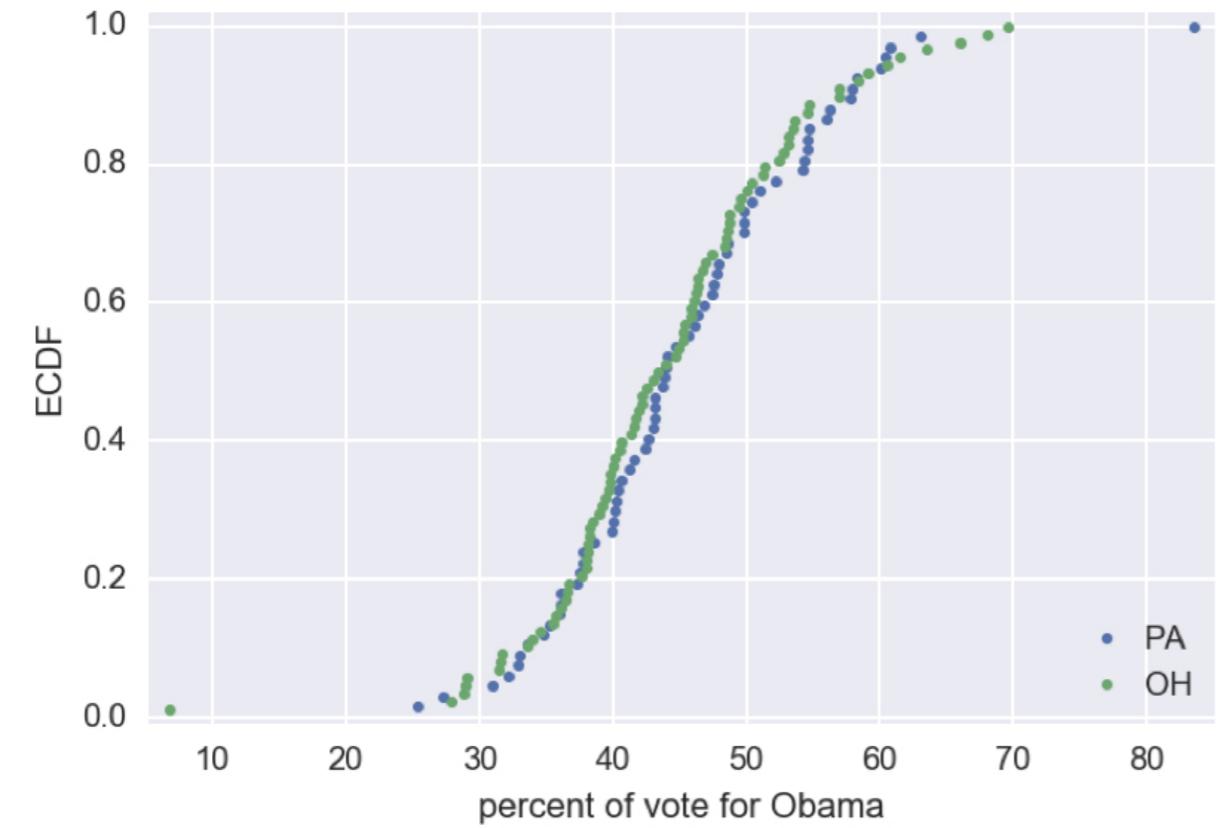
STATISTICAL THINKING IN PYTHON (PART 2)



Justin Bois

Lecturer at the California Institute of
Technology

Are OH and PA different?



¹ Data retrieved from Data.gov (<https://www.data.gov/>)

Hypothesis testing

- Assessment of how reasonable the observed data are assuming a hypothesis is true

Test statistic

- A single number that can be computed from observed data and from data you simulate under the null hypothesis
- It serves as a basis of comparison between the two

Permutation replicate

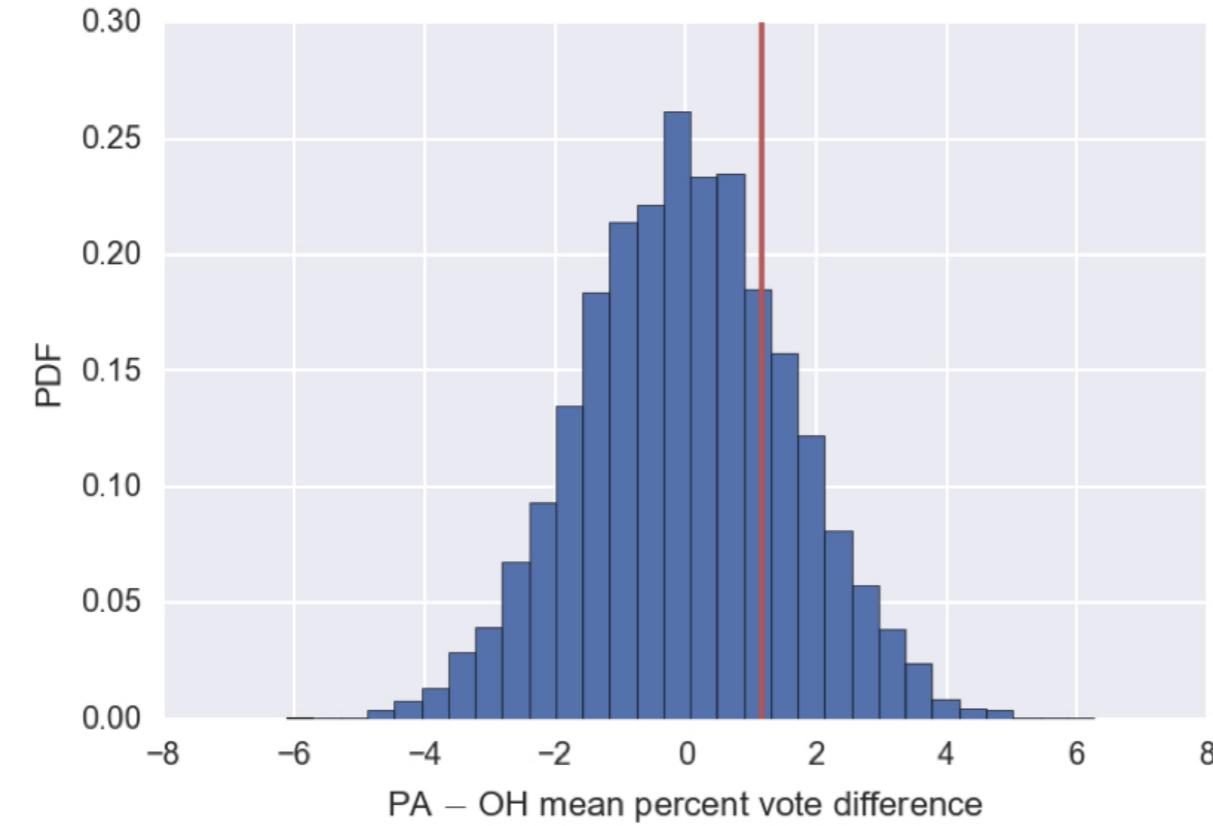
```
np.mean(perm_sample_PA) - np.mean(perm_sample_OH)
```

1.122220149253728

```
np.mean(dem_share_PA) - np.mean(dem_share_OH) # orig. data
```

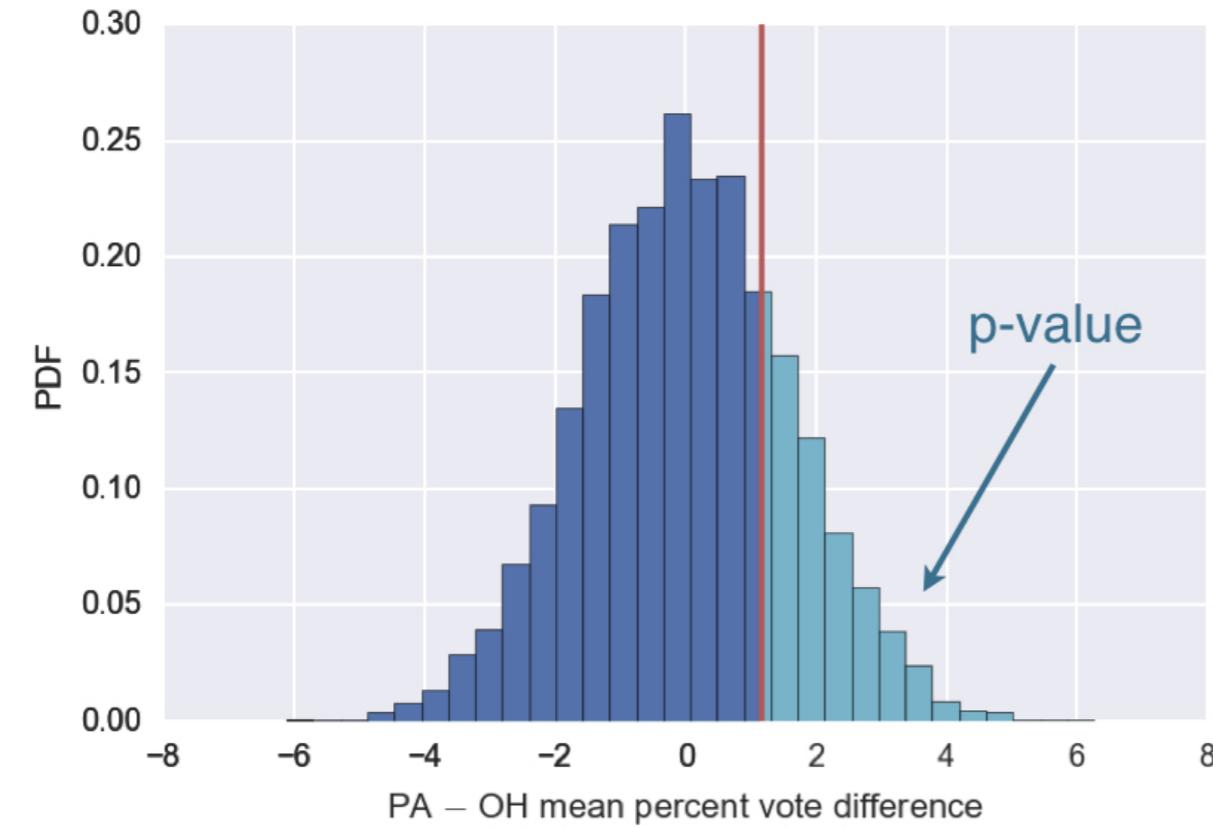
1.1582360922659518

Mean vote difference under null hypothesis



¹ Data retrieved from Data.gov (<https://www.data.gov/>)

Mean vote difference under null hypothesis



¹ Data retrieved from Data.gov (<https://www.data.gov/>)

p-value

- The probability of obtaining a value of your test statistic that is at least as extreme as what was observed, under the assumption the null hypothesis is true
- NOT the probability that the null hypothesis is true

Statistical significance

- Determined by the smallness of a p-value

Null hypothesis significance testing (NHST)

- Another name for what we are doing in this chapter

statistical significance ? practical significance

Let's practice!

STATISTICAL THINKING IN PYTHON (PART 2)

Bootstrap hypothesis tests

STATISTICAL THINKING IN PYTHON (PART 2)



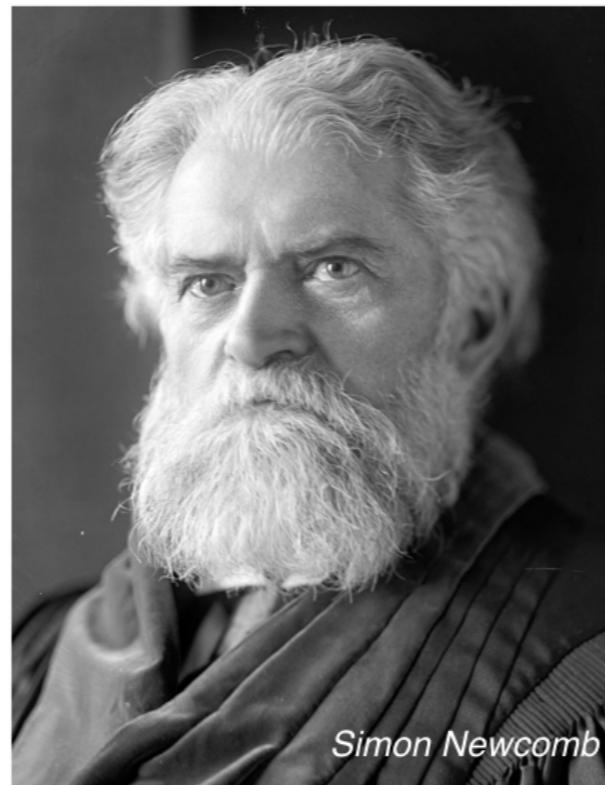
Justin Bois

Lecturer at the California Institute of
Technology

Pipeline for hypothesis testing

- Clearly state the null hypothesis
- Define your test statistic
- Generate many sets of simulated data assuming the null hypothesis is true
- Compute the test statistic for each simulated data set
- The p-value is the fraction of your simulated data sets for which the test statistic is at least as extreme as for the real data

Michelson and Newcomb: speed of light pioneers

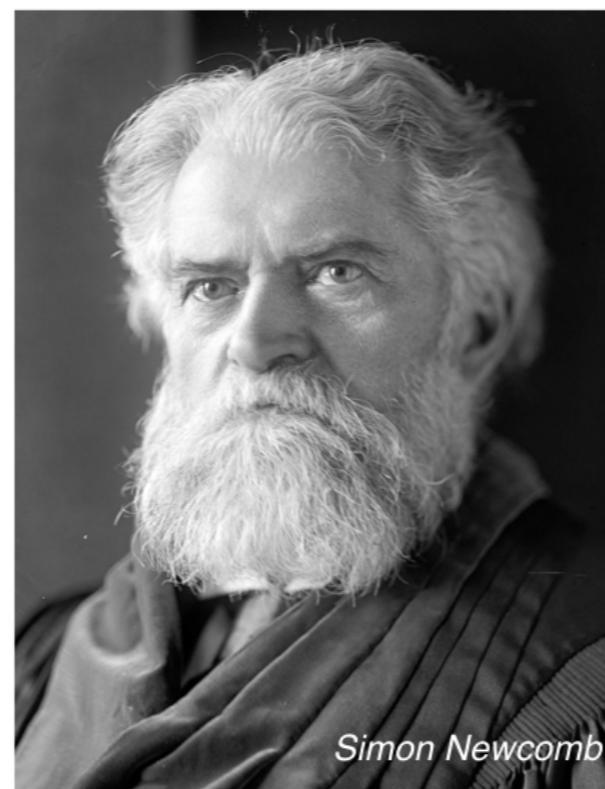


¹ Michelson image: public domain, Smithsonian ² Newcomb image: US Library of Congress

Michelson and Newcomb: speed of light pioneers



299,852 km/s

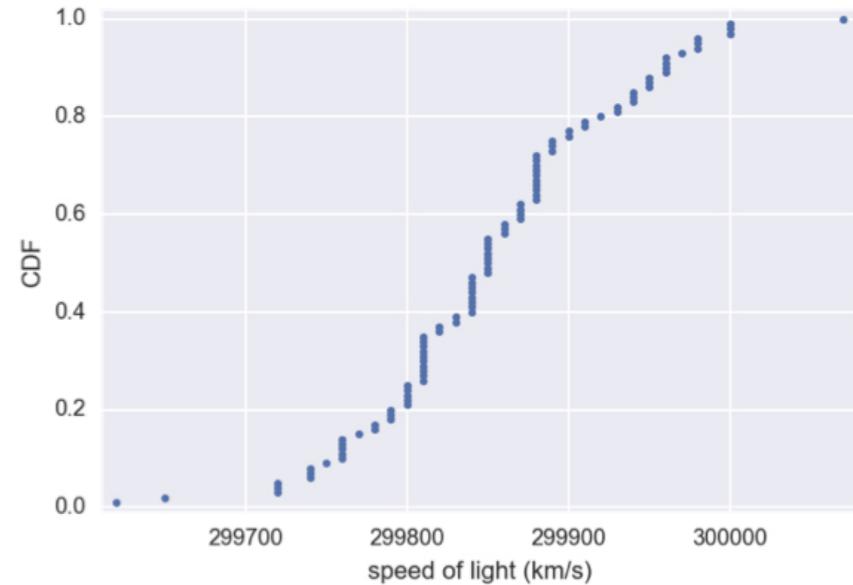


299,860 km/s

¹ Michelson image: public domain, Smithsonian ² Newcomb image: US Library of Congress

The data we have

Michelson:



Newcomb:

mean = 299,860 km/s

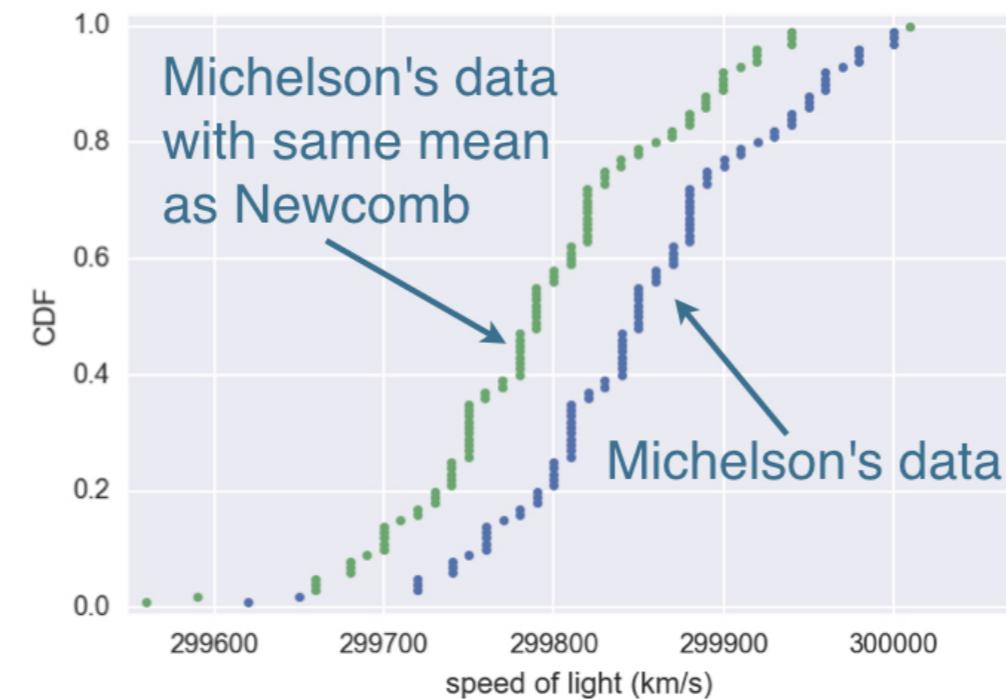
¹ Data: Michelson, 1880

Null hypothesis

- The true mean speed of light in Michelson's experiments was actually Newcomb's reported value

Shifting the Michelson data

```
newcomb_value = 299860 # km/s  
michelson_shifted = michelson_speed_of_light \\  
    - np.mean(michelson_speed_of_light) + newcomb_value
```



Calculating the test statistic

```
def diff_from_newcomb(data, newcomb_value=299860):  
    return np.mean(data) - newcomb_value
```

```
diff_obs = diff_from_newcomb(michelson_speed_of_light)  
diff_obs
```

```
-7.599999999767169
```

Computing the p-value

```
bs_replicates = draw_bs_reps(michelson_shifted,  
                           diff_from_newcomb, 1000)  
p_value = np.sum(bs_replicates <= diff_observed) / 1000  
p_value
```

0.1603999999999999

One sample test

- Compare one set of data to a single number

Two sample test

- Compare two sets of data

Let's practice!

STATISTICAL THINKING IN PYTHON (PART 2)

A/B testing

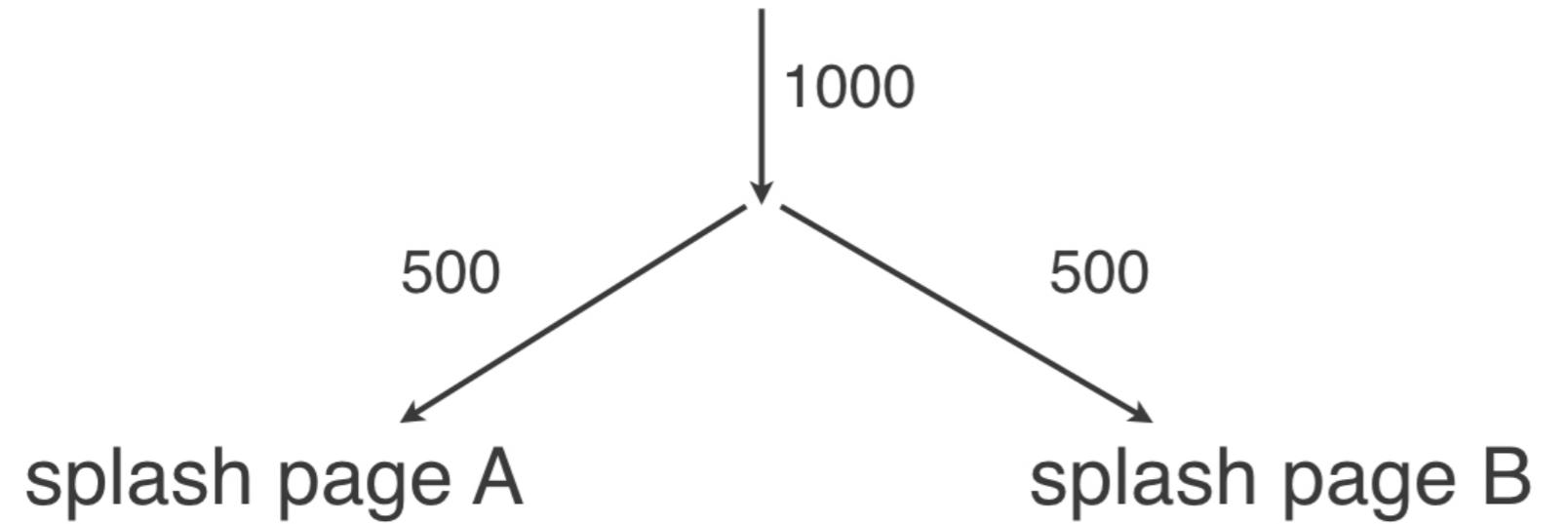
STATISTICAL THINKING IN PYTHON (PART 2)



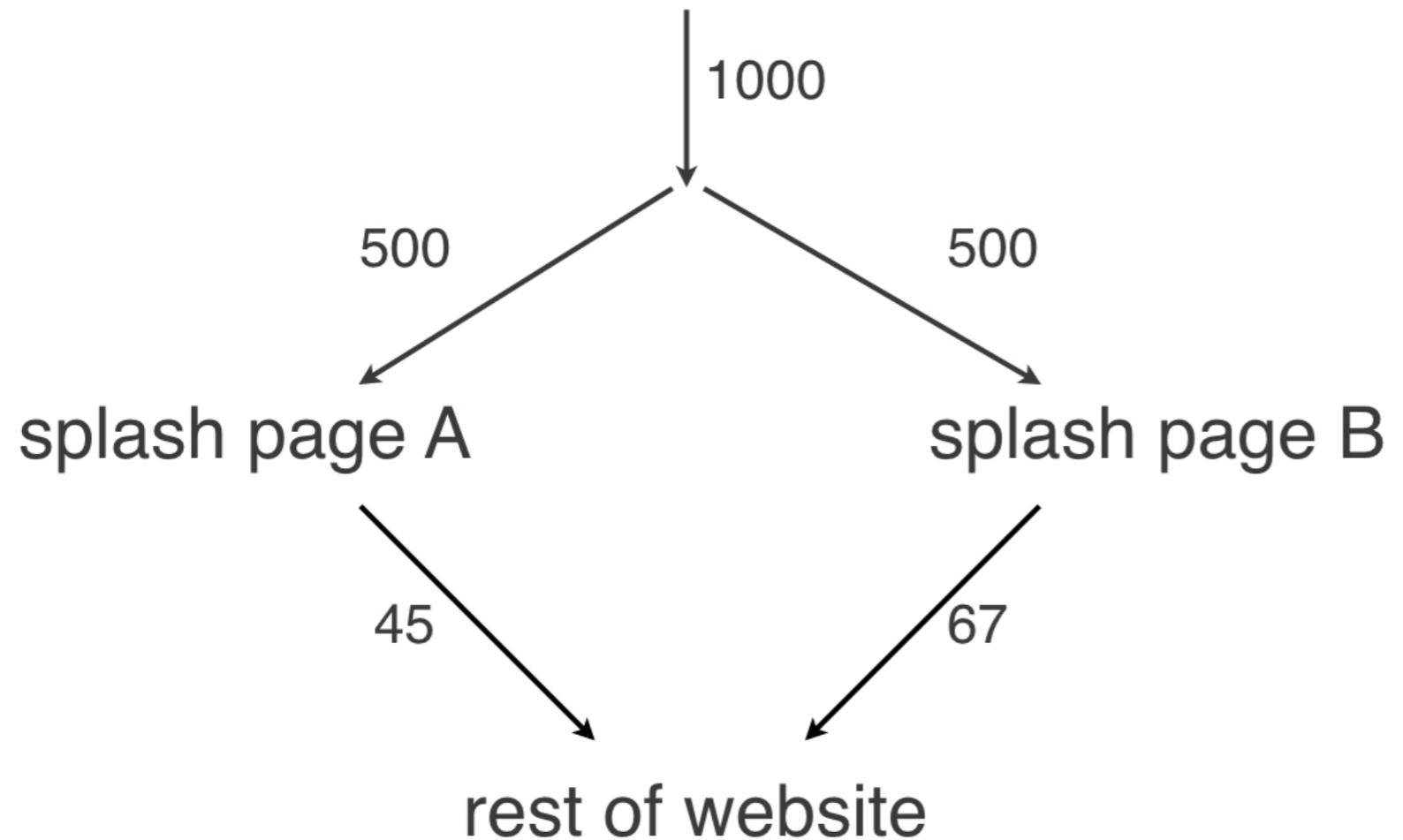
Justin Bois

Lecturer at the California Institute of
Technology

Is your redesign effective?



Is your redesign effective?



Null hypothesis

- The click-through rate is not affected by the redesign

Permutation test of clicks through

```
import numpy as np  
  
# clickthrough_A, clickthrough_B: arr. of 1s and 0s  
  
def diff_frac(data_A, data_B):  
    frac_A = np.sum(data_A) / len(data_A)  
    frac_B = np.sum(data_B) / len(data_B)  
    return frac_B - frac_A  
  
diff_frac_obs = diff_frac(clickthrough_A,  
                           clickthrough_B)
```

Permutation test of clicks through

```
perm_replicates = np.empty(10000)
for i in range(10000):
    perm_replicates[i] = permutation_replicate(
        clickthrough_A, clickthrough_B, diff_frac)
p_value = np.sum(perm_replicates >= diff_frac_obs) / 10000
p_value
```

0.016

A/B test

- Used by organizations to see if a strategy change gives a better result

Null hypothesis of an A/B test

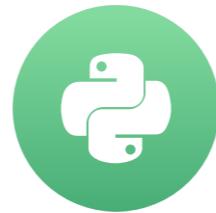
- The test statistic is impervious to the change

Let's practice!

STATISTICAL THINKING IN PYTHON (PART 2)

Test of correlation

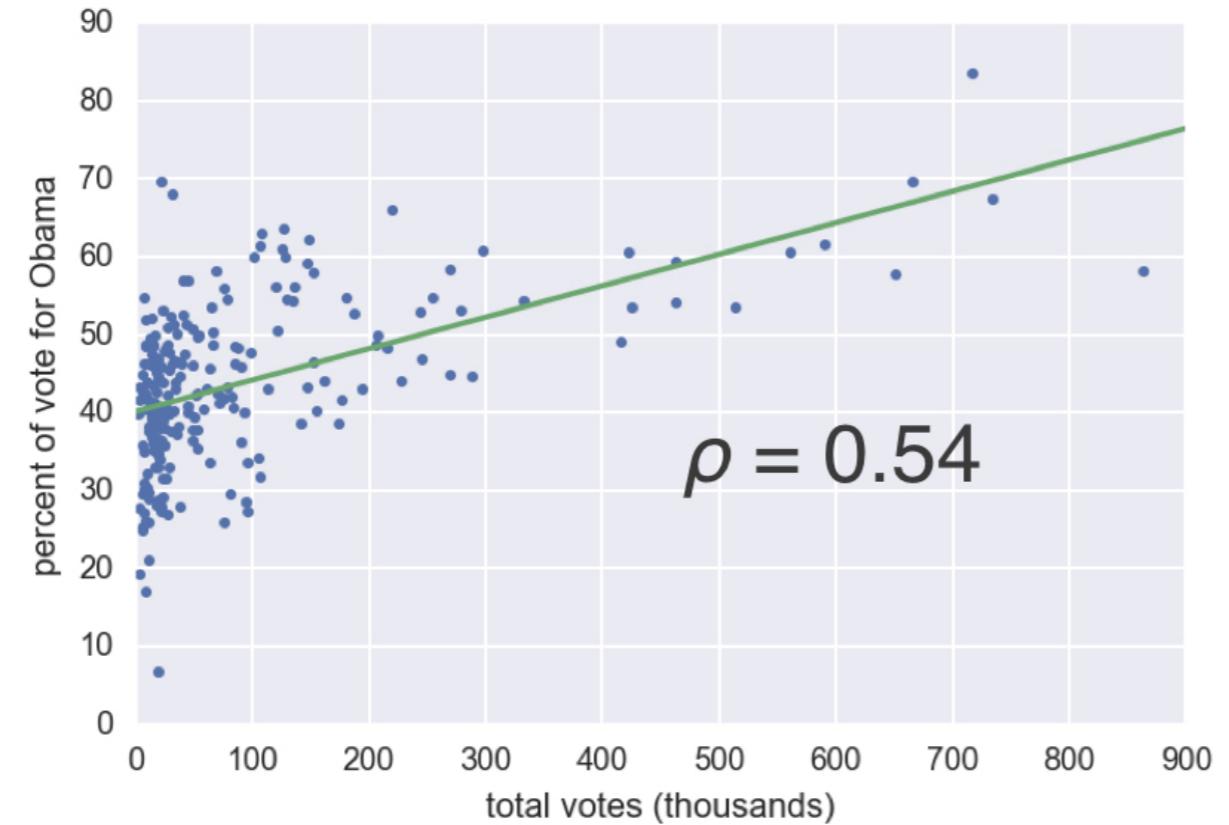
STATISTICAL THINKING IN PYTHON (PART 2)



Justin Bois

Lecturer at the California Institute of
Technology

2008 US swing state election results

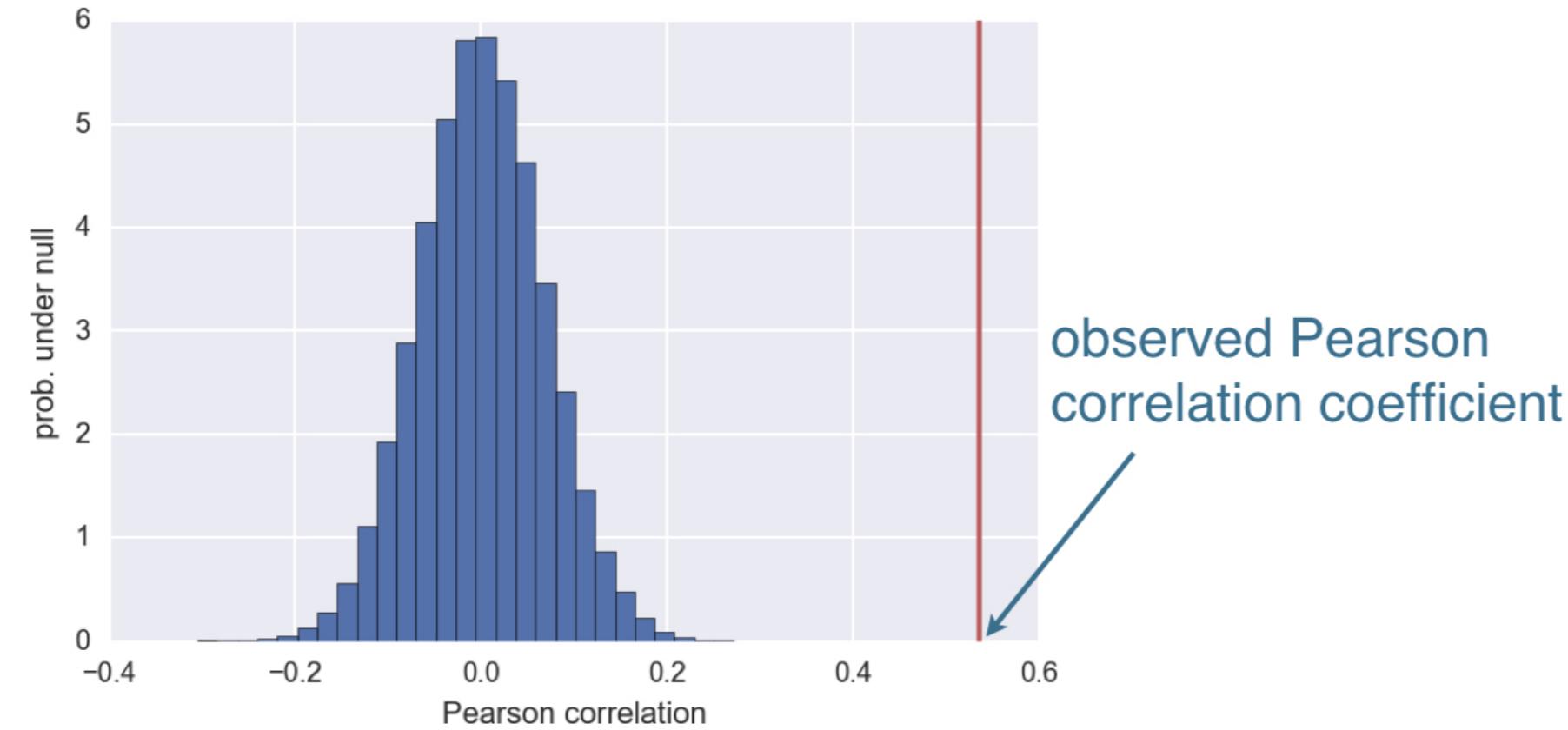


¹ Data retrieved from Data.gov (<https://www.data.gov/>)

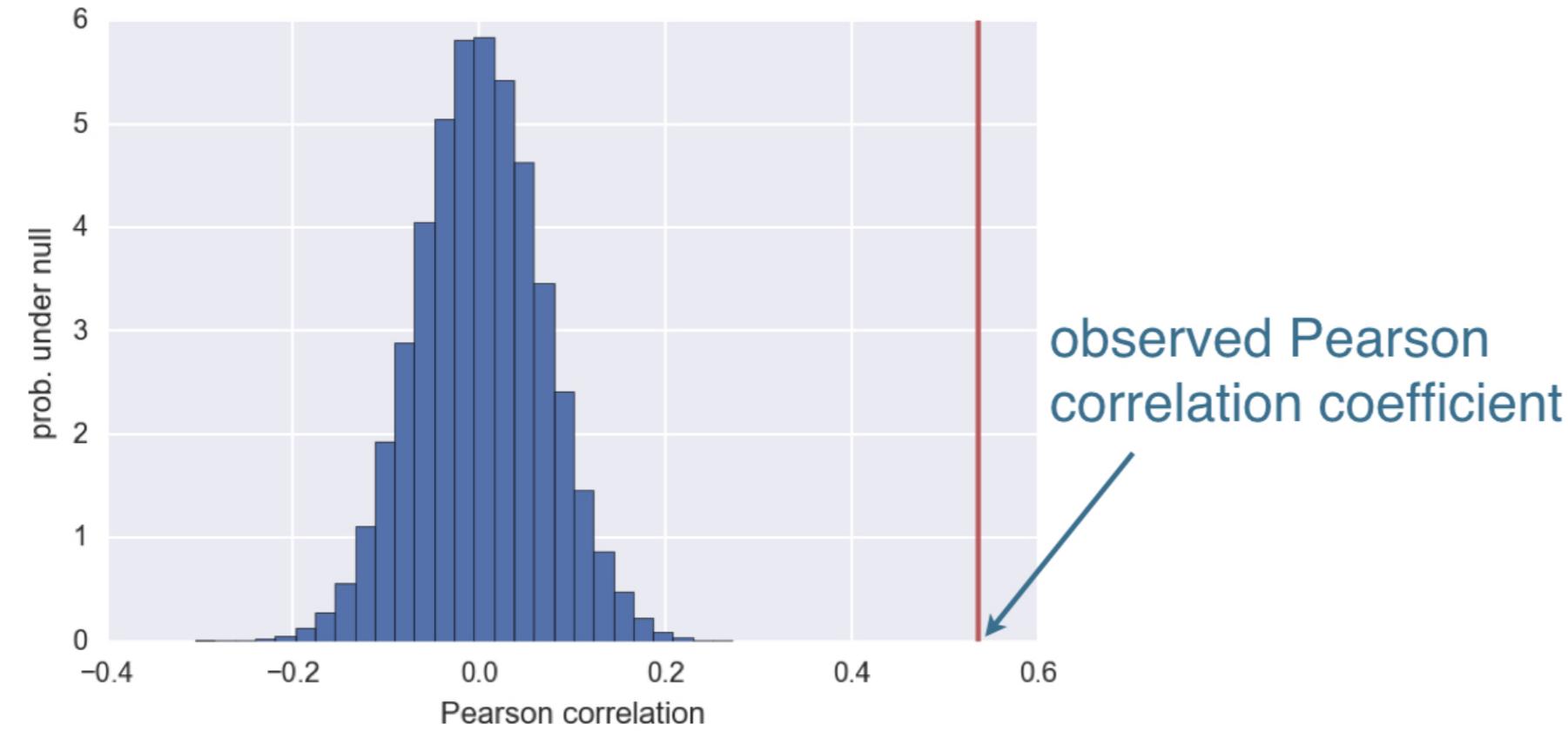
Hypothesis test of correlation

- Posit null hypothesis: the two variables are completely uncorrelated
- Simulate data assuming null hypothesis is true
- Use Pearson correlation, ρ , as test statistic
- Compute p-value as fraction of replicates that have ρ at least as large as observed.

More populous counties voted for Obama



More populous counties voted for Obama



p-value is very very small

Let's practice!

STATISTICAL THINKING IN PYTHON (PART 2)

Finch beaks and the need for statistics

STATISTICAL THINKING IN PYTHON (PART 2)

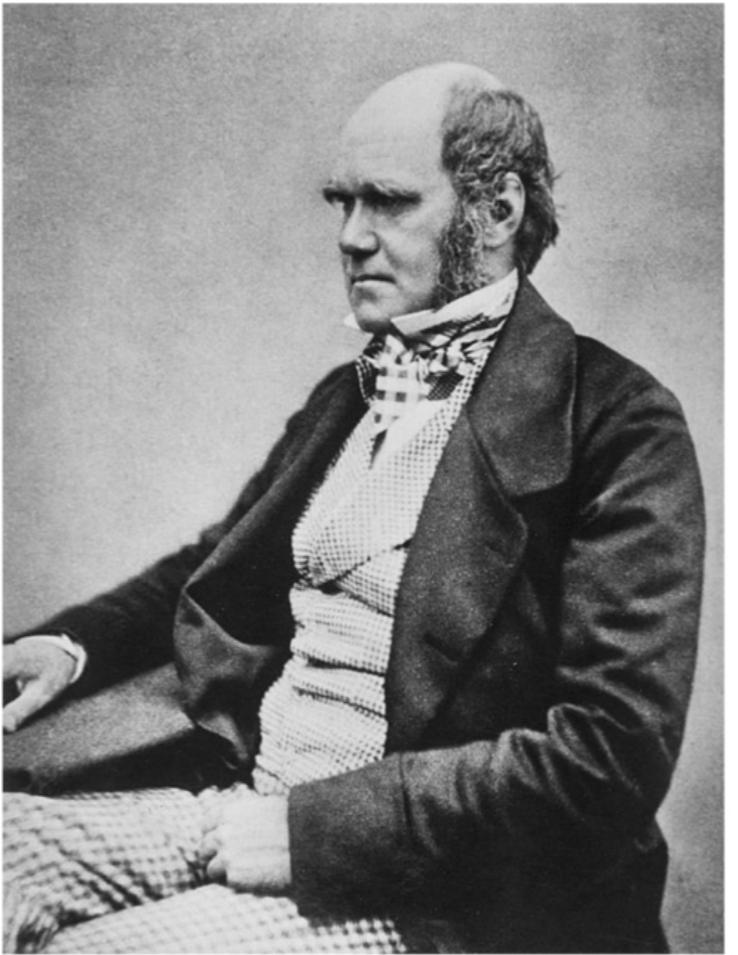
Justin Bois

Lecturer at the California Institute of
Technology



Your well-equipped toolbox

- Graphical and quantitative EDA
- Parameter estimation
- Confidence interval calculation
- Hypothesis testing



¹ Image: Public domain, US



¹ Image: NASA

The island of Daphne Major



¹ Image: Grant and Grant, 2014

The finches of Daphne Major



Geospiza fortis



Geospiza scandens

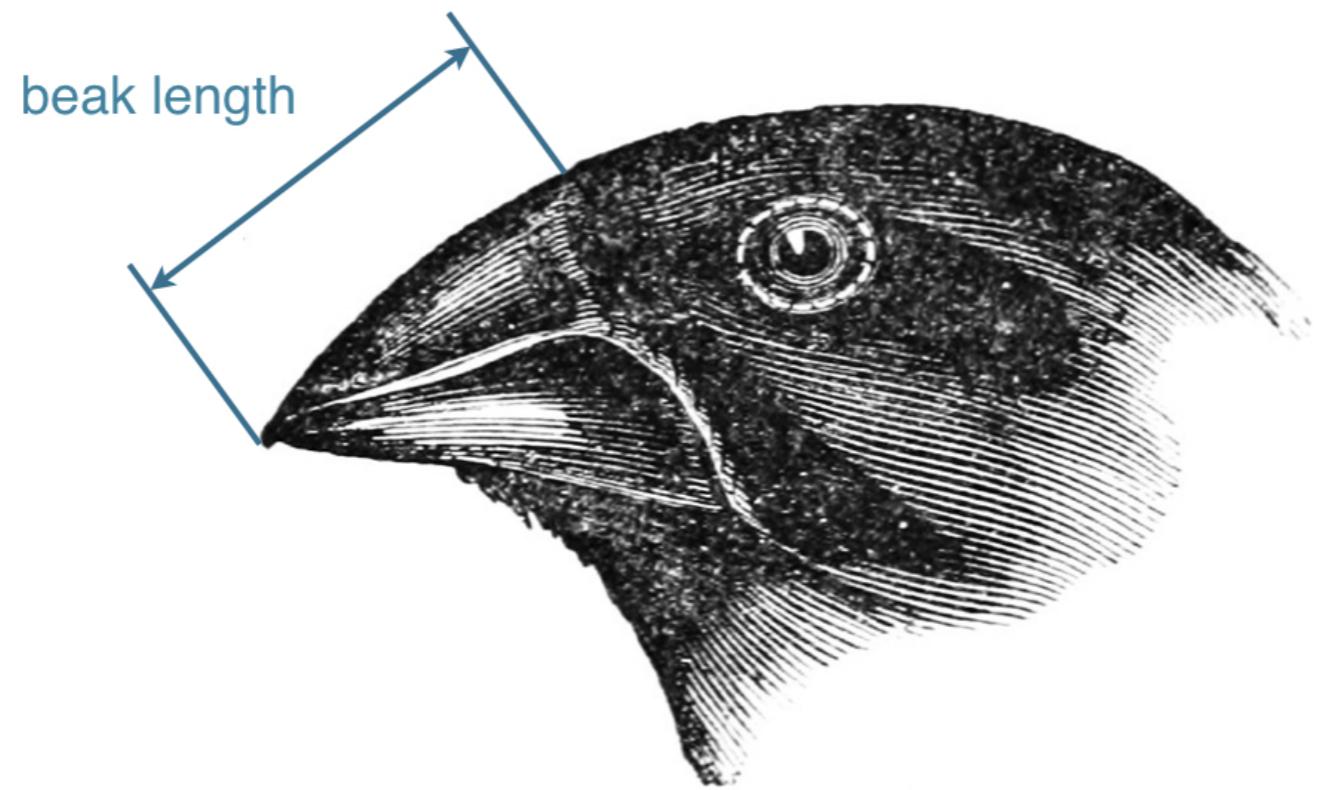
¹ Source: John Gould, public domain

Our data source

- Peter and Rosemary Grant
 - 40 Years of Evolution: Darwin's Finches on Daphne Major Island
 - Princeton University Press, 2014
- Data acquired from Dryad Digital Repository
 - <http://dx.doi.org/10.5061/dryad.g6g3h>

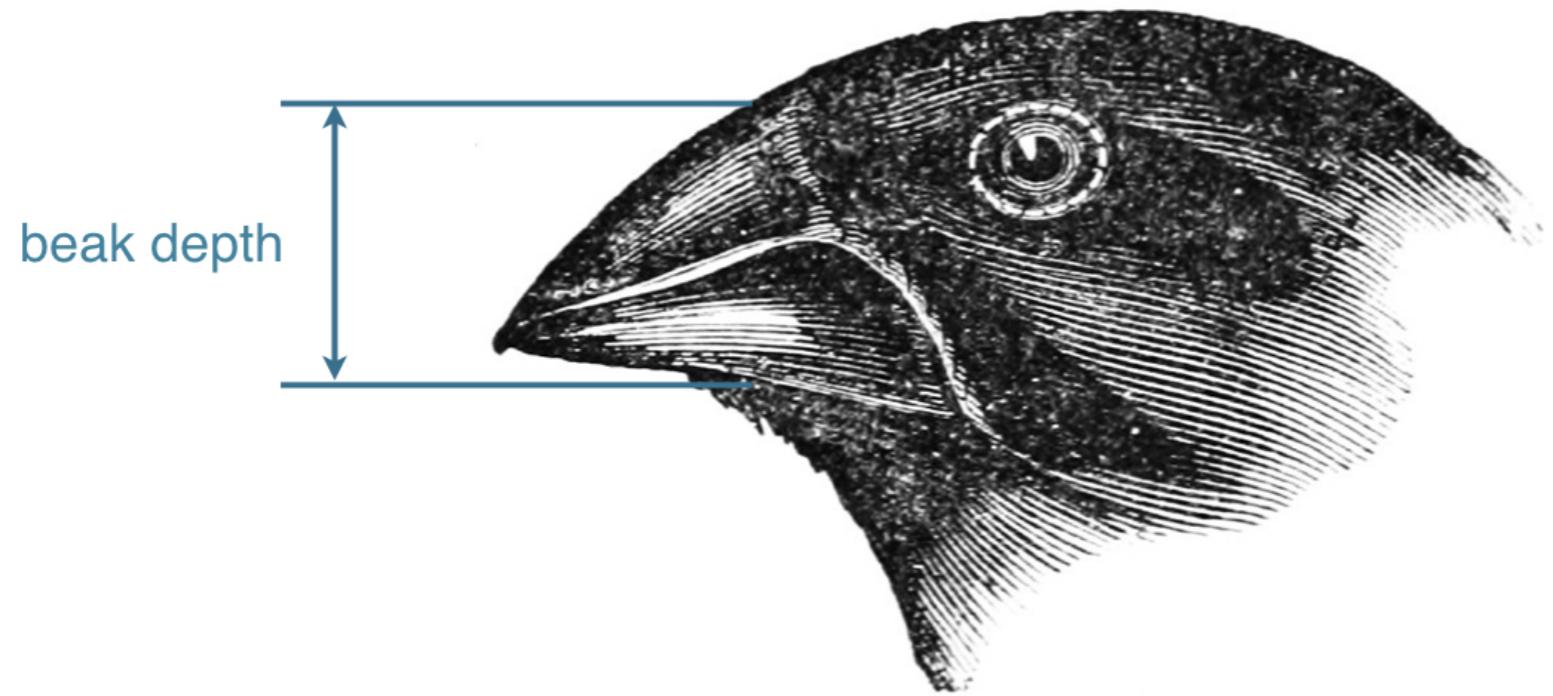


The dimensions of the finch beak



¹ Source: John Gould, public domain

The dimensions of the finch beak



¹ Source: John Gould, public domain

Investigation of *G. scandens* beak depth

- EDA of beak depths in 1975 and 2012
- Parameter estimates of mean beak depth
- Hypothesis test: did the beaks get deeper?

Let's practice!

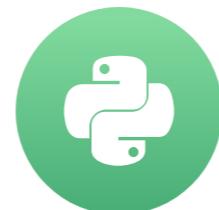
STATISTICAL THINKING IN PYTHON (PART 2)

Variation in beak shapes

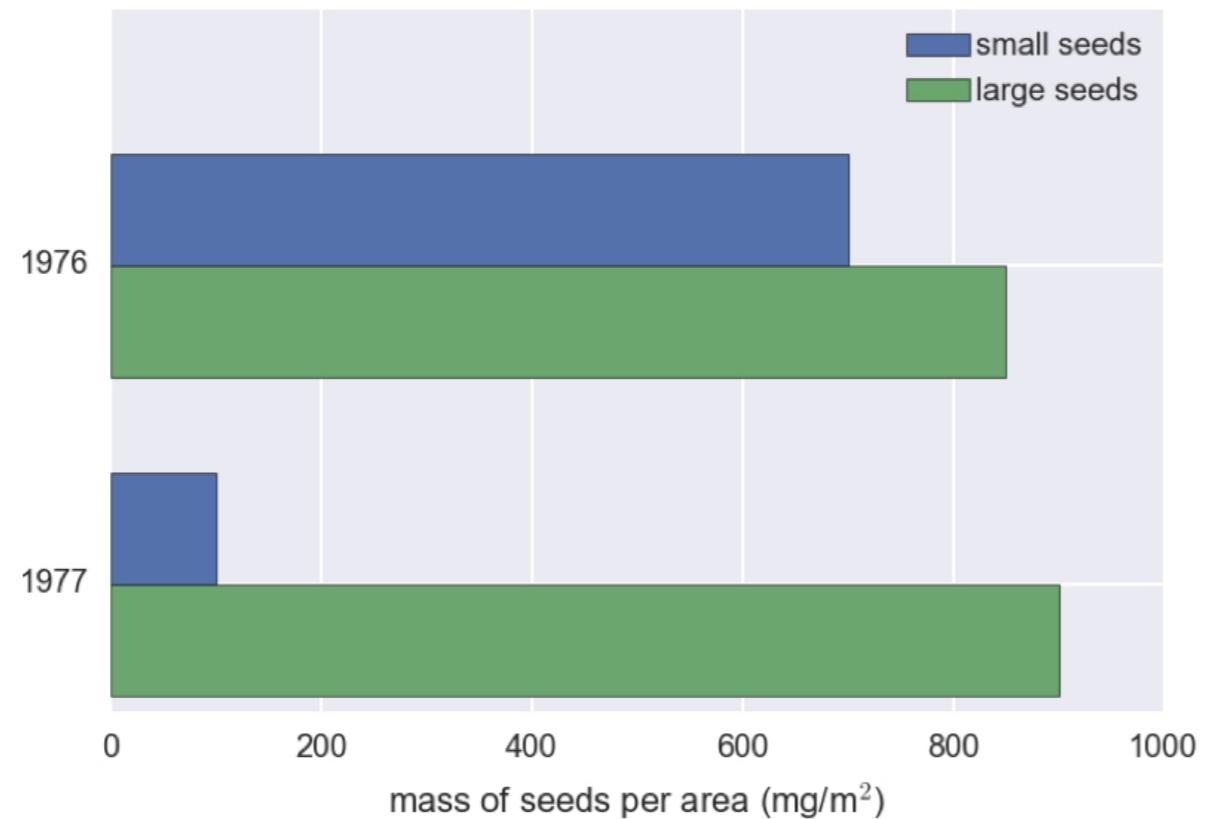
STATISTICAL THINKING IN PYTHON (PART 2)

Justin Bois

Lecturer at the California Institute of
Technology

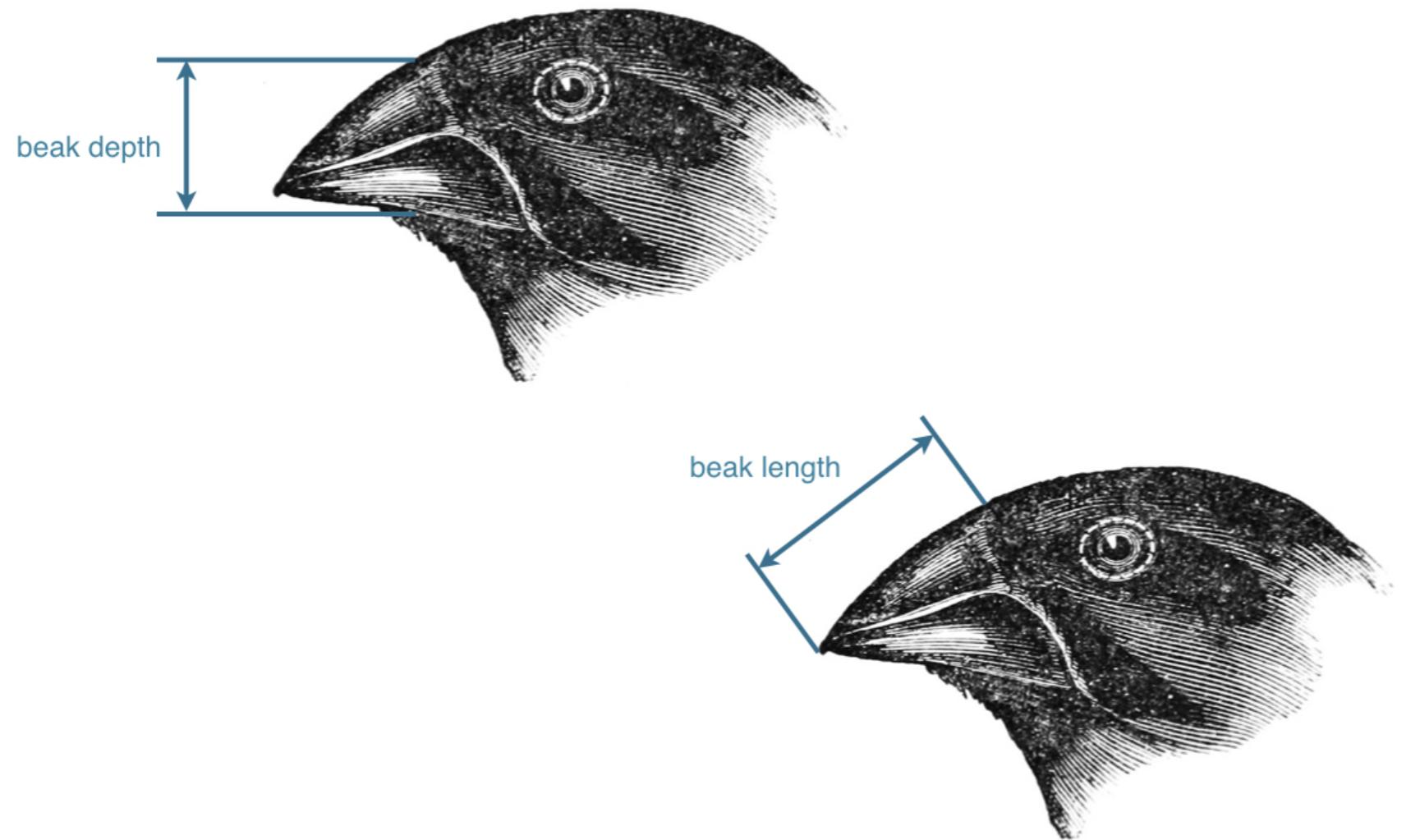


The drought of winter 1976/1977



¹ Source: Grant and Grant, 2014

Beak geometry



¹ Source: John Gould, public domain

Hint

- `draw_bs_pairs_linreg()` will come in handy

Let's practice!

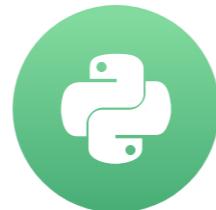
STATISTICAL THINKING IN PYTHON (PART 2)

Calculation of heritability

STATISTICAL THINKING IN PYTHON (PART 2)

Justin Bois

Lecturer at the California Institute of
Technology



The finches of Daphne Major



Geospiza fortis



Geospiza scandens

¹ Source: John Gould, public domain

Heredity

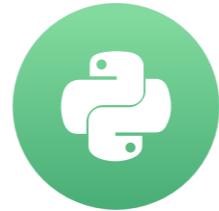
- The tendency for parental traits to be inherited by offspring

Let's practice!

STATISTICAL THINKING IN PYTHON (PART 2)

Final thoughts

STATISTICAL THINKING IN PYTHON (PART 2)



Justin Bois

Lecturer at the California Institute of
Technology

Your statistical thinking skills

- Perform EDA
 - Generate effective plots like ECDFs
 - Compute summary statistics
- Estimate parameters
 - By optimization, including linear regression
 - Determine confidence intervals
- Formulate and test hypotheses

Let's practice!

STATISTICAL THINKING IN PYTHON (PART 2)