Statistical inference and random sampling

FOUNDATIONS OF INFERENCE IN PYTHON

Paul Savala
Assistant Professor of Mathematics

Descriptive statistics

- Sample statistics meant to summarize the data
- Descriptive statistics summarize our sample

Date	SP500 Close	Daily Change
2017-08-07	2480.91	6.14
2017-08-08	2474.92	-5.99
2017-08-09	2474.02	-0.90
2017-08-10	2438.21	-35.81

Average daily change: -$9.14

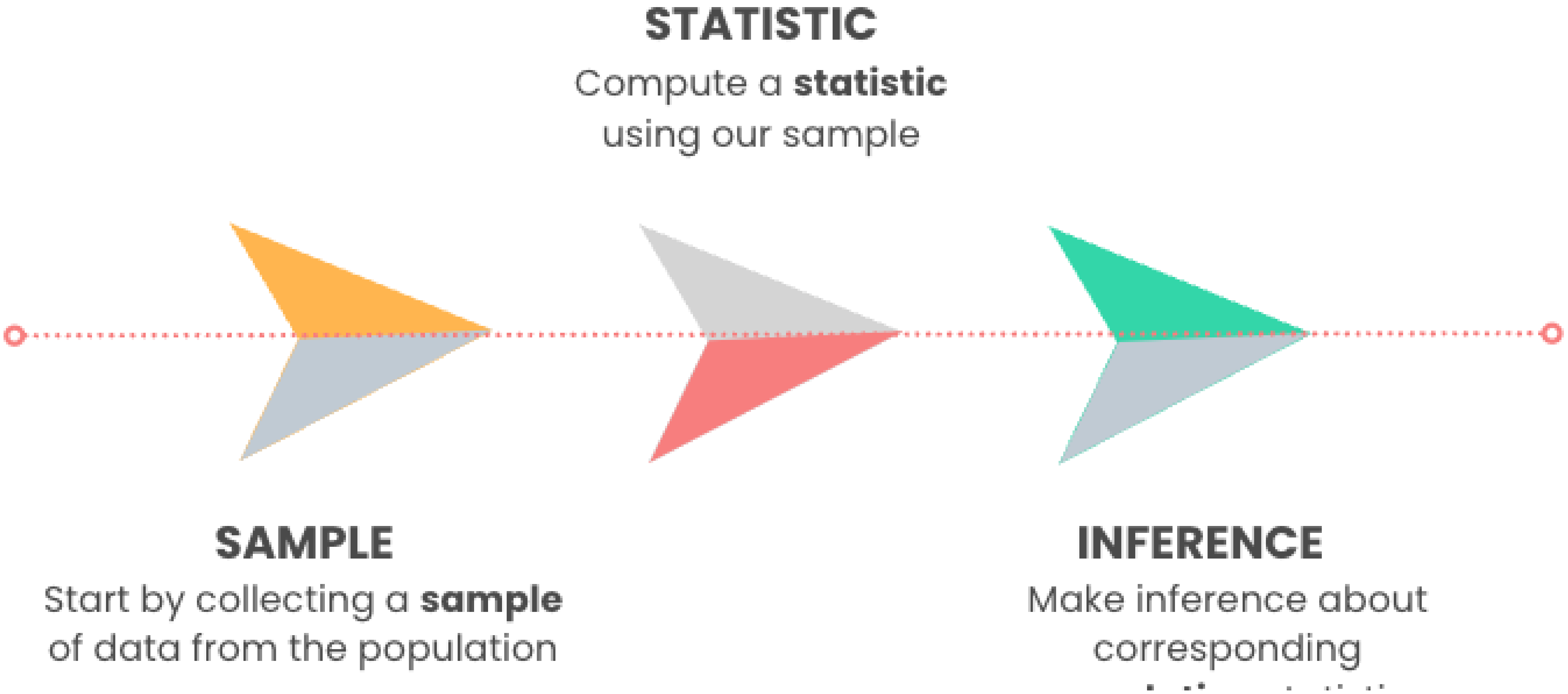
Inference

- Infer something about our population
- **Descriptive statistics:** Describe data
- **Inference:** Make conclusions and decisions

Date	SP500 Close	Daily Change
2017-08-07	2480.91	6.14
2017-08-08	2474.92	-5.99
2017-08-09	2474.02	-0.90
2017-08-10	2438.21	-35.81

Average daily swing for *any* days ~$9.14

Statistical inference process



STATISTIC

Compute a **statistic** using our sample

SAMPLE

Start by collecting a **sample** of data from the population

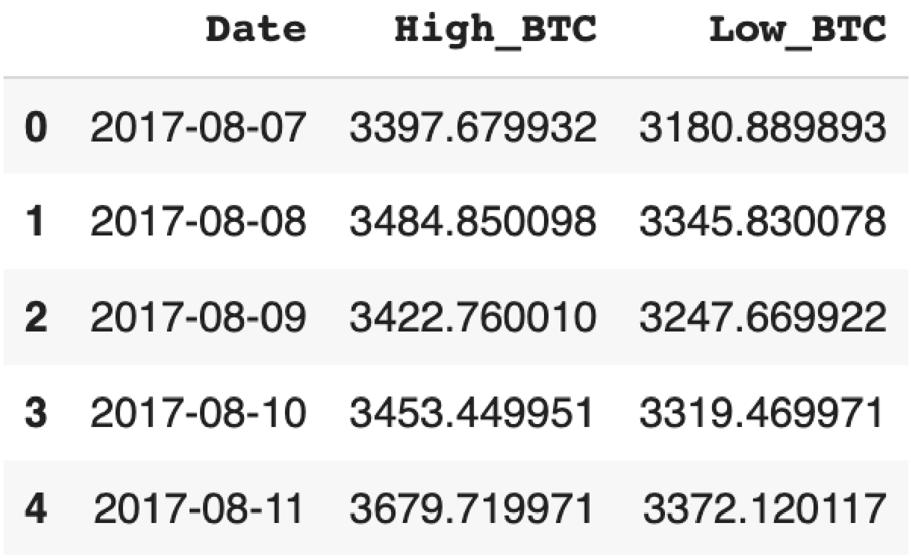
INFERENCE

Make inference about corresponding

Point estimates

- Given by a single value
- "Best guess" at an unknown population statistic

Point estimate: 1158.95 BTC daily swing



	Date	High_BTC	Low_BTC
0	2017-08-07	3397.679932	3180.889893
1	2017-08-08	3484.850098	3345.830078
2	2017-08-09	3422.760010	3247.669922
3	2017-08-10	3453.449951	3319.469971
4	2017-08-11	3679.719971	3372.120117

```python
btc_high = btc_sp_df['High_BTC']
btc_low = btc_sp_df['Low_BTC']

np.mean(btc_high - btc_low)
```

```
1158.95
```

Sampling

Point estimates depend on the sample

```
btc_sp_first100 = btc_sp_df.iloc[:100]
np.mean(btc_sp_first100['High_BTC'] - btc_sp_first100['Low_BTC'])
```

```
659.60
```

```
initial_row = np.random.choice(btc_sp_df.shape[0]-100)
btc_sp_random_100 = btc_sp_df.iloc[initial_row:initial_row+100]
np.mean(btc_sp_first100['High_BTC'] - btc_sp_first100['Low_BTC'])
```

```
943.83
```

Let's practice!

FOUNDATIONS OF INFERENCE IN PYTHON

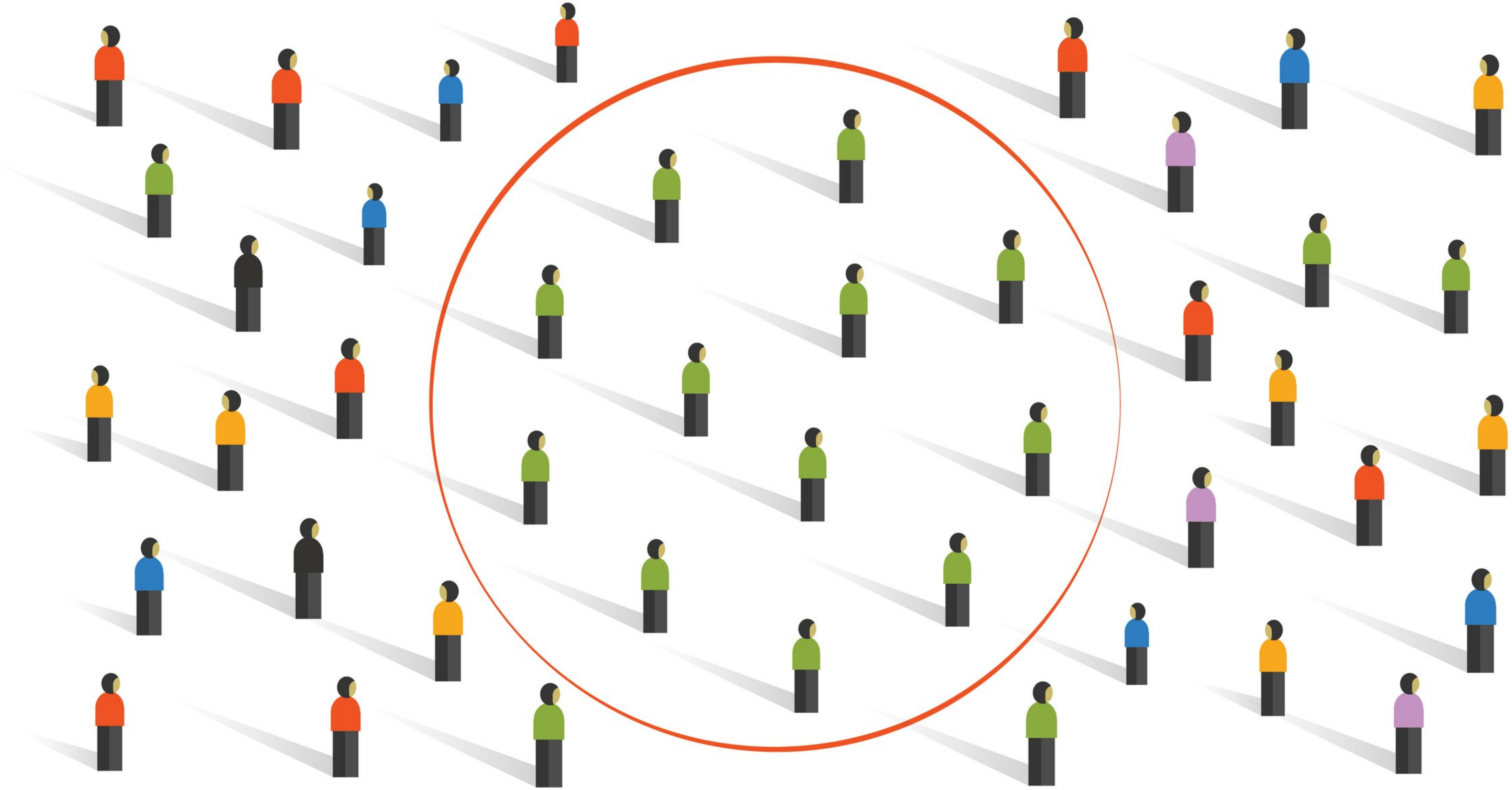
Sampling and bias

FOUNDATIONS OF INFERENCE IN PYTHON

Paul Savala
Assistant Professor of Mathematics

Bias

- **Biased sample:** A group occurs more/less often in sample than in population



Biased samples

```
all_salaries = [75000, 82000, ...]
friends_salaries = [93000, 87000, 103000, 101000]


np.mean(friends_salaries)
```

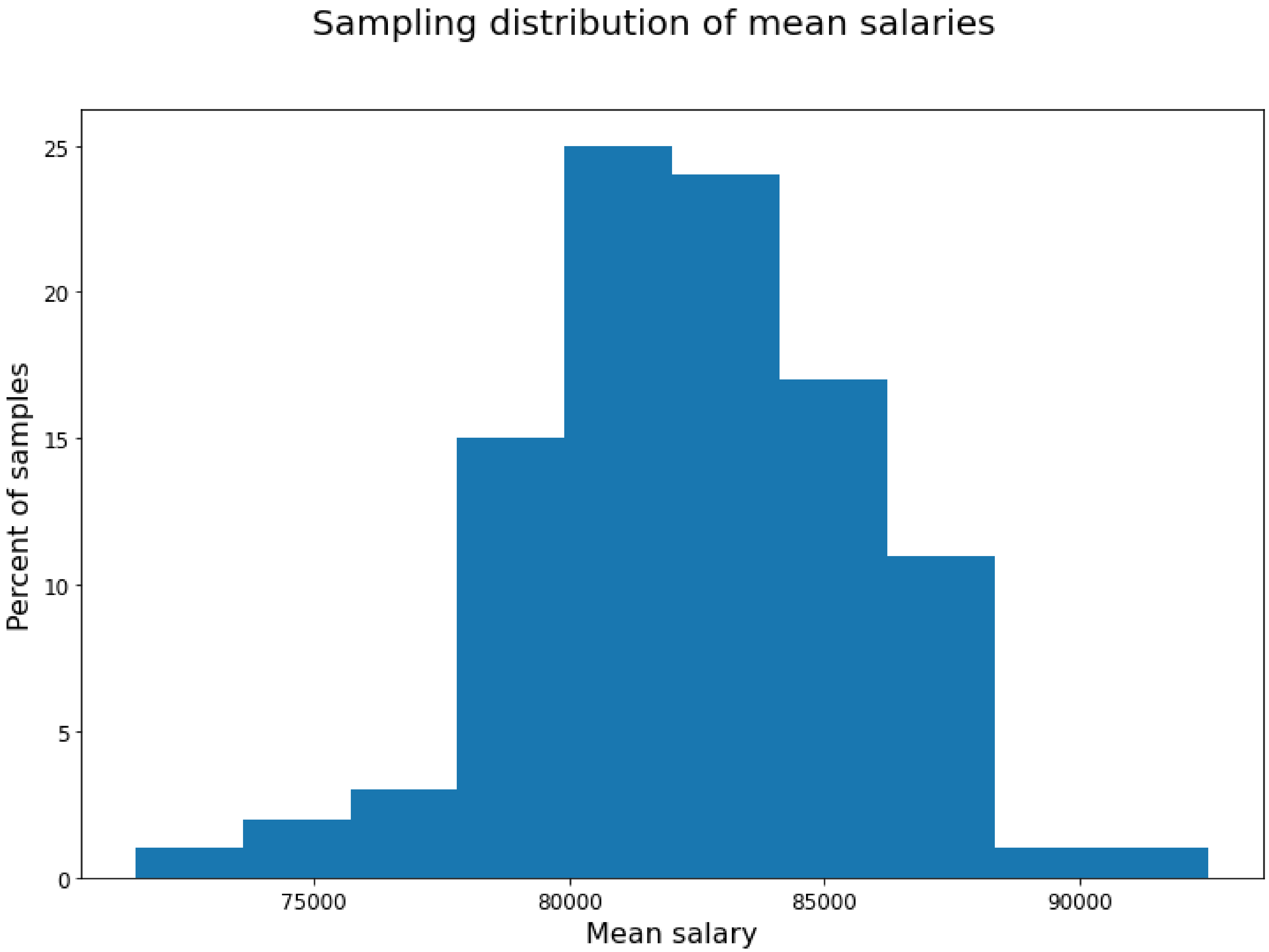
```
96000
```

Sampling distribution

```
sampling_distribution = []

for i in range(100):
    random_sample = np.random.choice(salaries, size=10)
    sample_mean = np.mean(random_sample)
    sampling_distribution.append(sample_mean)

plt.hist(sampling_distribution)
plt.xlabel('Mean salary')
plt.ylabel('Percent of samples')
plt.title('Sampling distribution of mean salaries')
plt.show()
```


Sampling distribution of mean salaries
Percent of samples
0
5
10
15
20
25
75000
80000
85000
90000
Mean salary

Depends on the sample

- Samples affect point estimates
- Point estimates affect inference
- Samples affect p-value calculations

Doesn't depend on the sample

- Population statistic
  - Is unaffected by sample chosen
- Conclusion from test
  - Given a p-value, conclusion is unaffected by sample chosen

Let's practice!

FOUNDATIONS OF INFERENCE IN PYTHON

Confidence intervals and sampling

FOUNDATIONS OF INFERENCE IN PYTHON

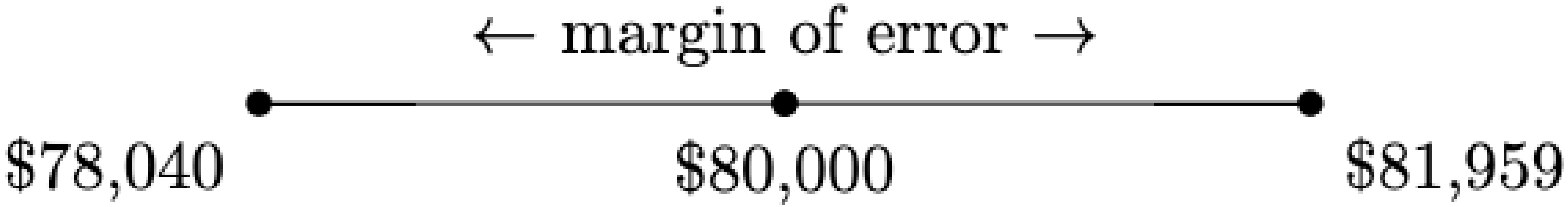
Paul Savala
Assistant Professor of Mathematics

What is a confidence interval?

- Uses samples to generate range of values
- Range of values estimate the population statistic

Example:

- Sample of 100 employees
- Mean salary of $80,000
- Standard deviation of $10,000



Calculating a confidence interval

```
from scipy import stats
import numpy as np

ci = stats.norm.interval(loc=80000,                    # Mean
                         scale=10000/np.sqrt(100),    # Standard error
                         alpha=0.95)                  # Confidence level
print(ci)
```

```
(78040.04, 81959.96)
```

Valid inference requires a normal sampling distribution

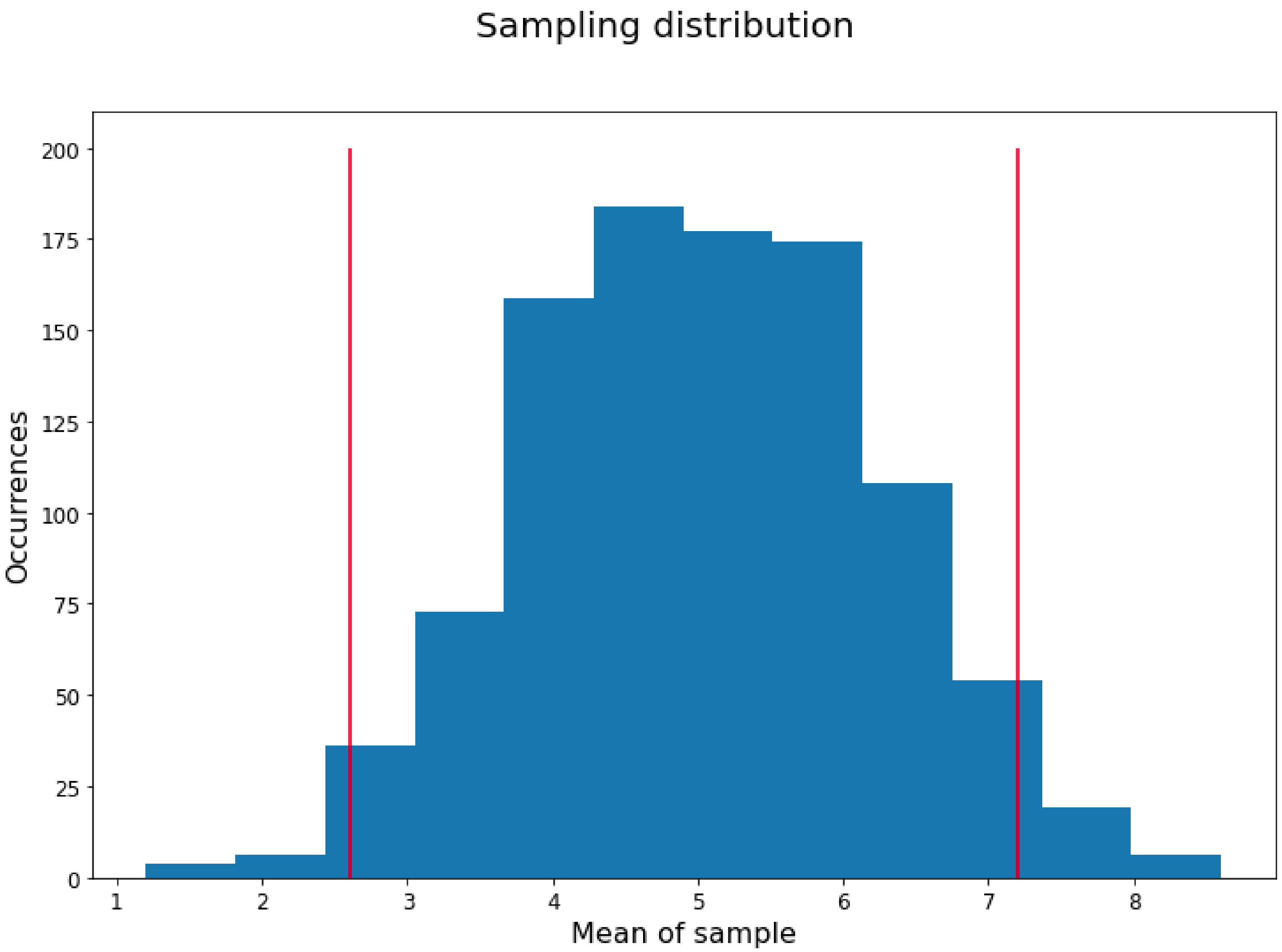
Central Limit Theorem

- Average many independent samples
- Sampling distribution is approximately normal

```
population = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
sample_means = []

for i in range(1000):
  sample_5 = np.random.choice(population, size=5)
  sample_means.append(sample_5.mean())
```

```
plt.hist(sample_means)
```



What a confidence interval tells us

(and what it doesn't tell us)

- Population statistic is or is not in confidence interval
- Repeated samples -> 95% of confidence intervals contain population statistic

Let's practice!

FOUNDATIONS OF INFERENCE IN PYTHON