Introduction to bootstrapping

SAMPLING IN PYTHON

James Chapman
Curriculum Manager, DataCamp

With or without

Sampling without replacement:

Sampling with replacement ("resampling"):

Simple random sampling without replacement

Population:

Sample:

Simple random sampling with replacement

Population:

Resample:

Why sample with replacement?

- `coffee_ratings` : a sample of a larger population of all coffees

- Each coffee in our sample represents many different hypothetical population coffees

- Sampling with replacement is a proxy

Coffee data preparation

```
coffee_focus = coffee_ratings[["variety", "country_of_origin", "flavor"]]
coffee_focus = coffee_focus.reset_index()
```

```
      index   variety country_of_origin   flavor
0         0      None          Ethiopia     8.83
1         1     Other          Ethiopia     8.67
2         2   Bourbon         Guatemala     8.50
3         3      None          Ethiopia     8.58
4         4     Other          Ethiopia     8.50
...     ...       ...               ...      ...
1333   1333      None           Ecuador     7.58
1334   1334      None           Ecuador     7.67
1335   1335      None     United States     7.33
1336   1336      None             India     6.83
1337   1337      None           Vietnam     6.67

[1338 rows x 4 columns]
```

Resampling with .sample()

```
coffee_resamp = coffee_focus.sample(frac=1, replace=True)
```

```
      index   variety country_of_origin   flavor
1140   1140   Bourbon          Guatemala     7.25
57       57   Bourbon          Guatemala     8.00
1152   1152   Bourbon             Mexico     7.08
621     621   Caturra           Thailand     7.50
44       44      SL28              Kenya     8.08
...     ...       ...                ...      ...
996     996    Typica             Mexico     7.33
1090   1090   Bourbon          Guatemala     7.33
918     918     Other          Guatemala     7.42
249     249   Caturra           Colombia     7.67
467     467   Caturra           Colombia     7.50

[1338 rows x 4 columns]
```

Repeated coffees

```
coffee_resamp["index"].value_counts()
```

```
658     5
167     4
363     4
357     4
1047    4
        ..
771     1
770     1
766     1
764     1
0       1
Name: index, Length: 868, dtype: int64
```

Missing coffees

```python
num_unique_coffees = len(coffee_resamp.drop_duplicates(subset="index"))
```

```
868
```

```python
len(coffee_ratings) - num_unique_coffees
```

```
470
```

Bootstrapping

The opposite of sampling from a population

Sampling: going from a population to a smaller sample

Bootstrapping: building up a theoretical population from the sample

Bootstrapping use case:

- Develop understanding of sampling variability using a single sample

Bootstraps

Bootstrapping process

1. Make a resample of the same size as the original sample

2. Calculate the statistic of interest for this bootstrap sample

3. Repeat steps 1 and 2 many times

The resulting statistics are *bootstrap statistics*, and they form a *bootstrap distribution*

Bootstrapping coffee mean flavor

```python
import numpy as np
mean_flavors_1000 = []
for i in range(1000):
    mean_flavors_1000.append(
        np.mean(coffee_sample.sample(frac=1, replace=True)['flavor'])
    )
```

Bootstrap distribution histogram

```python
import matplotlib.pyplot as plt
plt.hist(mean_flavors_1000)
plt.show()
```
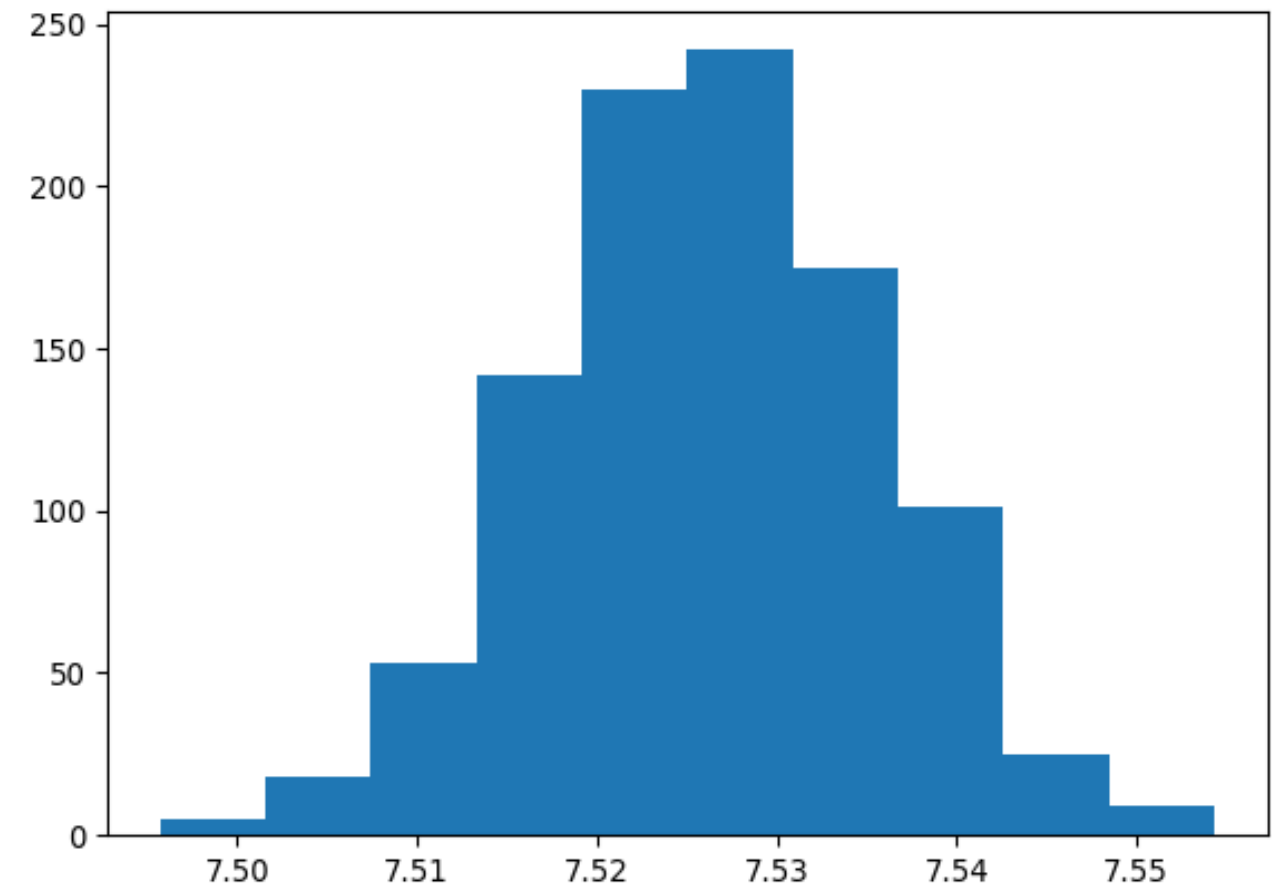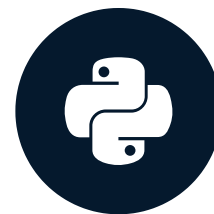
Let's practice!

SAMPLING IN PYTHON

Comparing sampling and bootstrap distributions

SAMPLING IN PYTHON

James Chapman
Curriculum Manager, DataCamp

Coffee focused subset

```
coffee_sample = coffee_ratings[["variety", "country_of_origin", "flavor"]]\
    .reset_index().sample(n=500)
```

```
     index        variety    country_of_origin  flavor
132    132          Other           Costa Rica    7.58
51      51           None  United States (Hawaii)  8.17
42      42  Yellow Bourbon               Brazil    7.92
569    569        Bourbon            Guatemala    7.67
..     ...            ...                  ...     ...
643    643         Catuai           Costa Rica    7.42
356    356        Caturra             Colombia    7.58
494    494           None            Indonesia    7.58
169    169           None               Brazil    7.81

[500 rows x 4 columns]
```
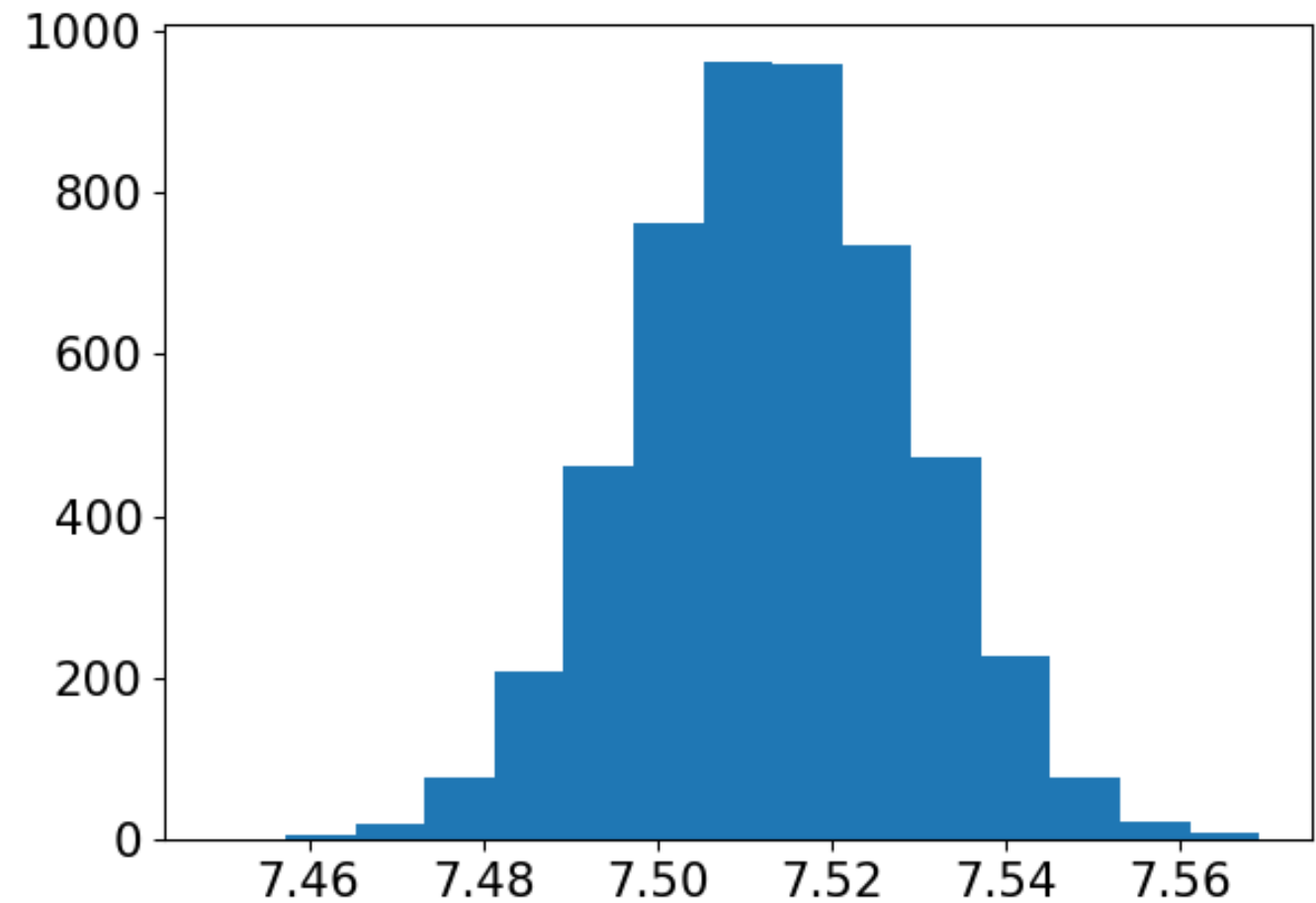
The bootstrap of mean coffee flavors

```python
import numpy as np
mean_flavors_5000 = []
for i in range(5000):
    mean_flavors_5000.append(
        np.mean(coffee_sample.sample(frac=1, replace=True)['flavor'])
    )
bootstrap_distn = mean_flavors_5000
```

Mean flavor bootstrap distribution

```python
import matplotlib.pyplot as plt
plt.hist(bootstrap_distn, bins=15)
plt.show()
```

Sample, bootstrap distribution, population means

Sample mean:

```
coffee_sample['flavor'].mean()
```

```
7.513220000000005
```

Estimated population mean:

```
np.mean(bootstrap_distn)
```

```
7.513357731999999
```

True population mean:

```
coffee_ratings['flavor'].mean()
```

```
7.526046337817639
```

Interpreting the means

Bootstrap distribution mean:

- Usually close to the sample mean

- May not be a good estimate of the population mean

Bootstrapping cannot correct biases from sampling

Sample sd vs. bootstrap distribution sd

Sample standard deviation:

```
coffee_sample['flavor'].std()
```

```
0.3540883911928703
```

Estimated population standard deviation?

```
np.std(bootstrap_distn, ddof=1)
```

```
0.01576847436795821
```

Sample, bootstrap dist'n, pop'n standard deviations

Sample standard deviation:

```
coffee_sample['flavor'].std()
```

```
0.3540883911928703
```

True standard deviation:

```
coffee_ratings['flavor'].std(ddof=0)
```

```
0.34125481224622645
```

Estimated population standard deviation:

```
standard_error = np.std(bootstrap_distn, ddof=1)
```

Standard error is the standard deviation of the statistic of interest

```
standard_error * np.sqrt(500)
```

```
0.3525938058821761
```

Standard error times square root of sample size estimates the population standard deviation

Interpreting the standard errors

- *Estimated standard error* → standard deviation of the bootstrap distribution for a sample statistic

- Population std. dev $\approx$ Std. Error $\times \sqrt{\text{Sample size}}$
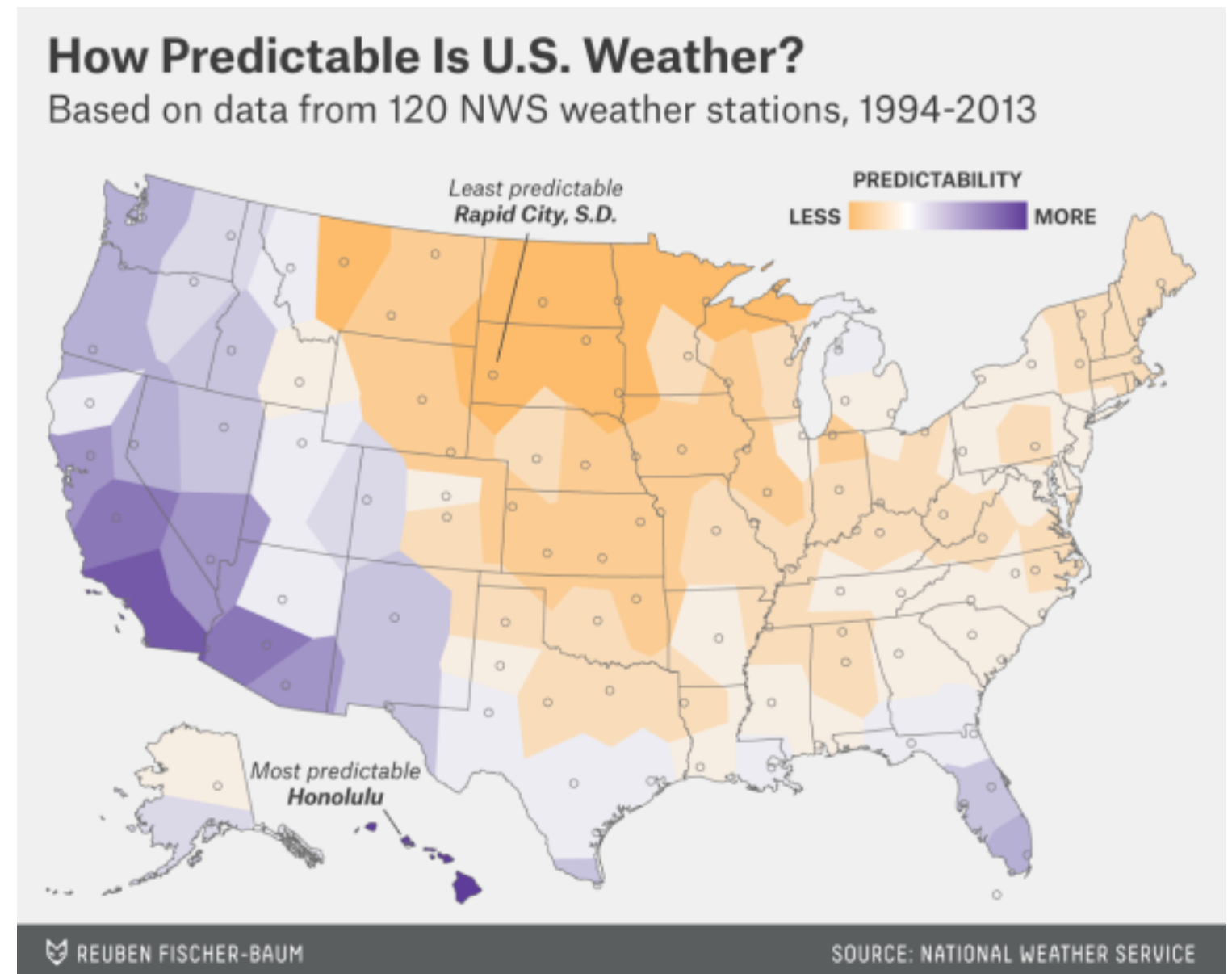
Let's practice!

SAMPLING IN PYTHON

Confidence intervals

- "Values within one standard deviation of the mean" includes a large number of values from each of these distributions

- We'll define a related concept called a *confidence interval*

Predicting the weather

- Rapid City, South Dakota in the United States has the least predictable weather

- Our job is to predict the high temperature there tomorrow

How Predictable Is U.S. Weather?
Based on data from 120 NWS weather stations, 1994-2013

Least predictable
Rapid City, S.D.

PREDICTABILITY
LESS   MORE

Most predictable
Honolulu

REUBEN FISCHER-BAUM   SOURCE: NATIONAL WEATHER SERVICE

Our weather prediction

- Point estimate = 47°F (8.3°C)

- Range of plausible high temperature values = 40 to 54°F (4.4 to 12.8°C)

We just reported a confidence interval!

- 40 to 54°F is a *confidence interval*

- Sometimes written as 47 °F (40°F, 54°F) or 47°F [40°F, 54°F]

- ... or, 47 ± 7°F

- 7°F is the *margin of error*

Bootstrap distribution of mean flavor

```python
import matplotlib.pyplot as plt
plt.hist(coffee_boot_distn, bins=15)
plt.show()
```
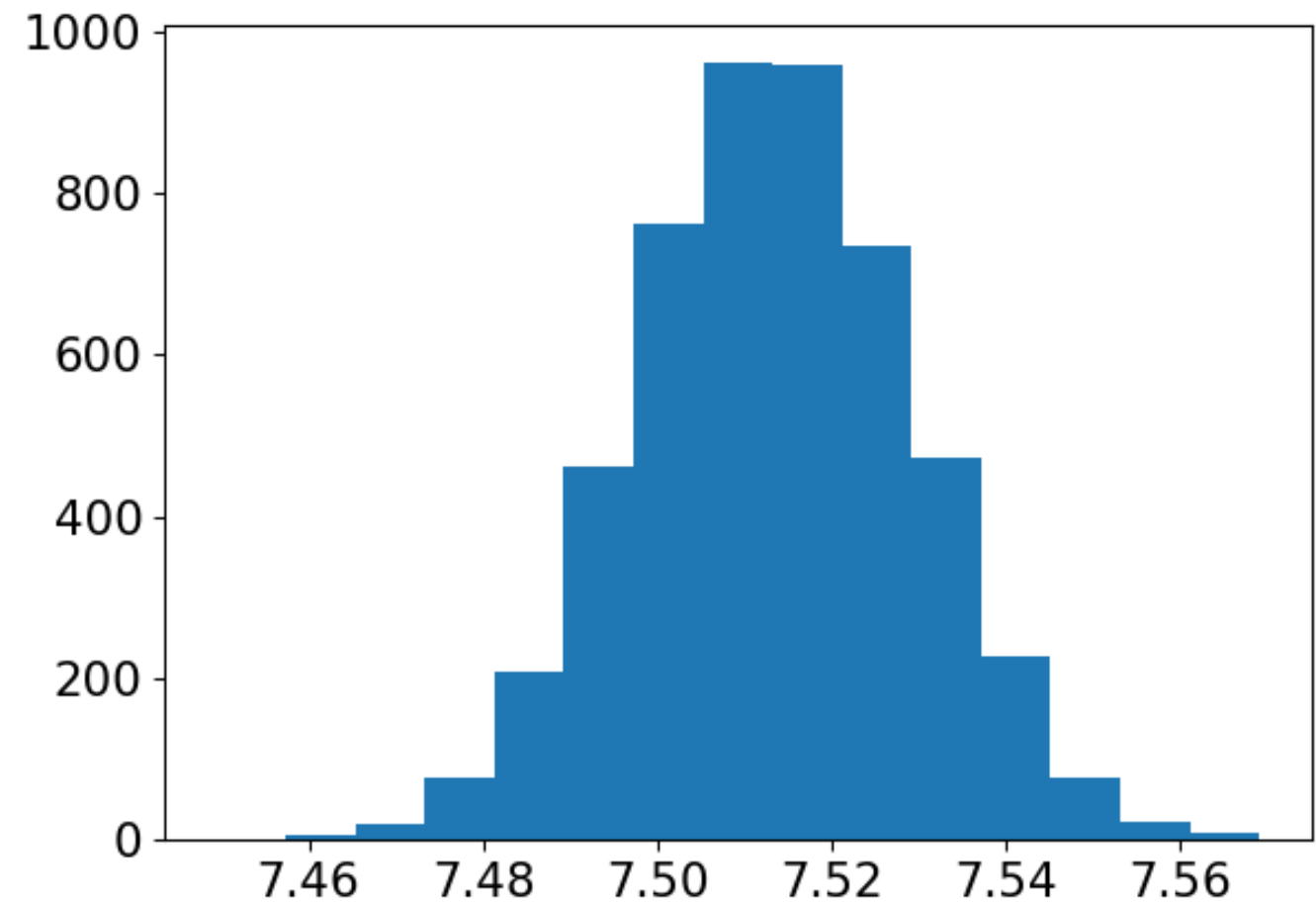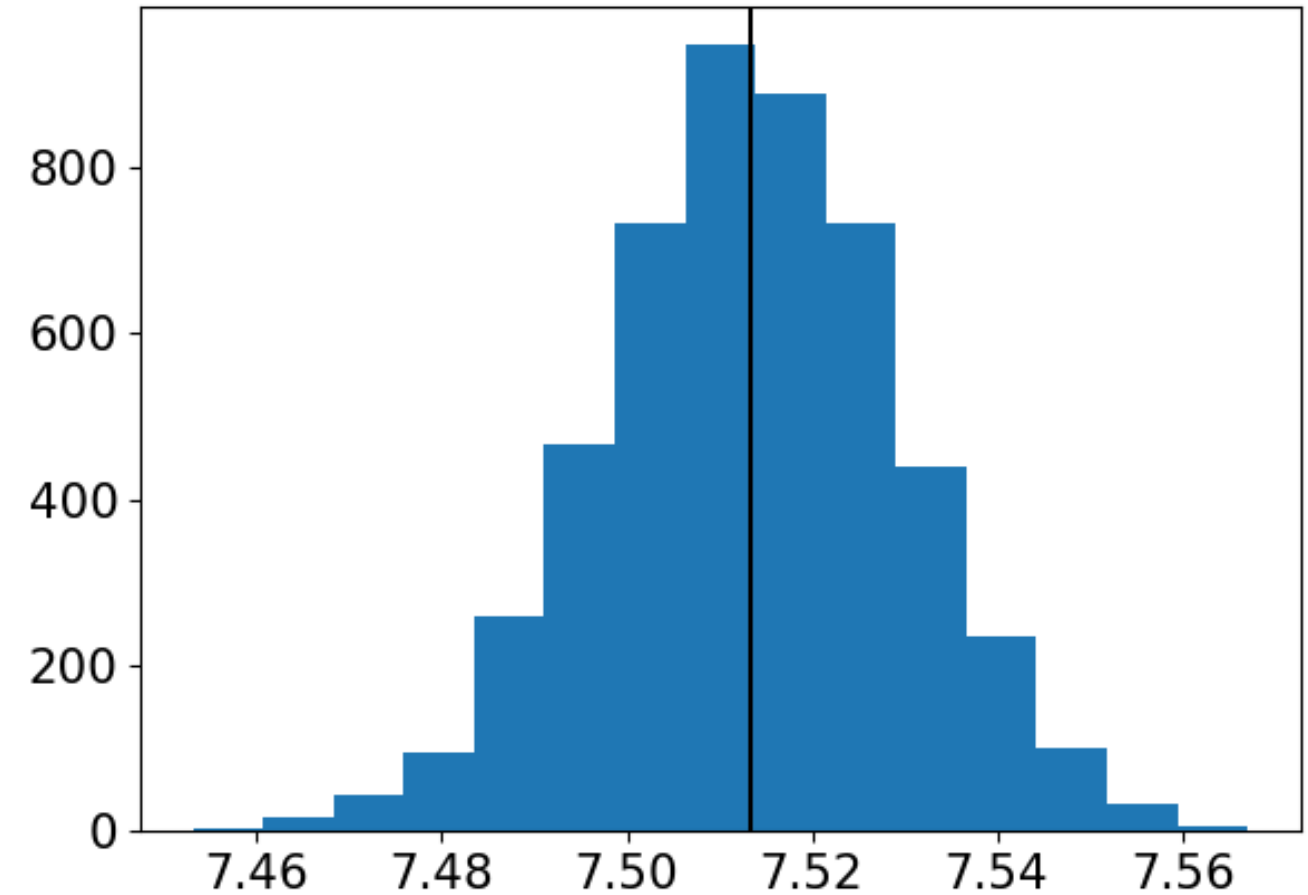
Mean of the resamples

```python
import numpy as np
np.mean(coffee_boot_distn)
```

```
7.513452892
```

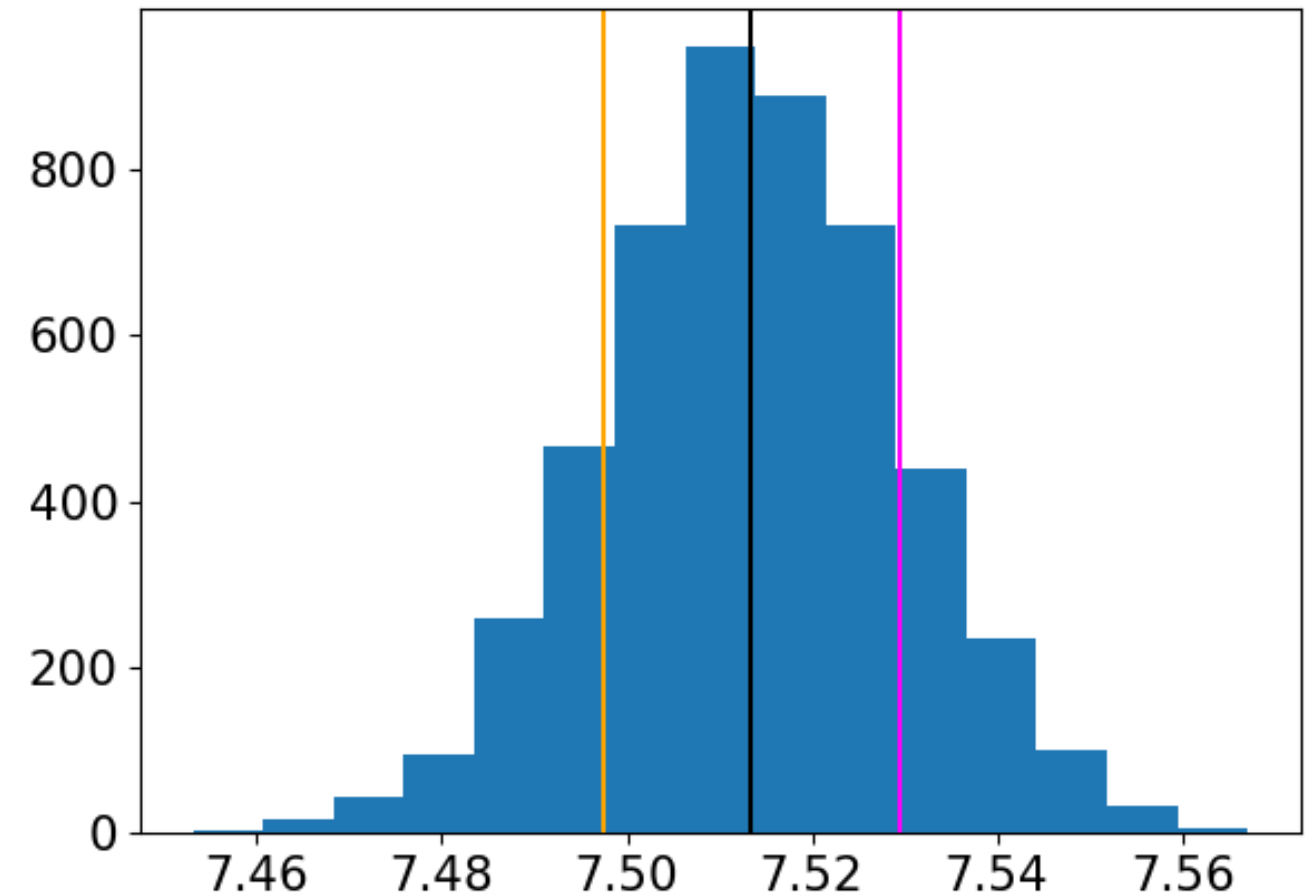# Mean plus or minus one standard deviation

```
np.mean(coffee_boot_distn)
```

`7.513452892`

```
np.mean(coffee_boot_distn) - np.std(coffee_boot_distn, ddof=1)
```

`7.497385709174466`

```
np.mean(coffee_boot_distn) + np.std(coffee_boot_distn, ddof=1)
```

`7.529520074825534`

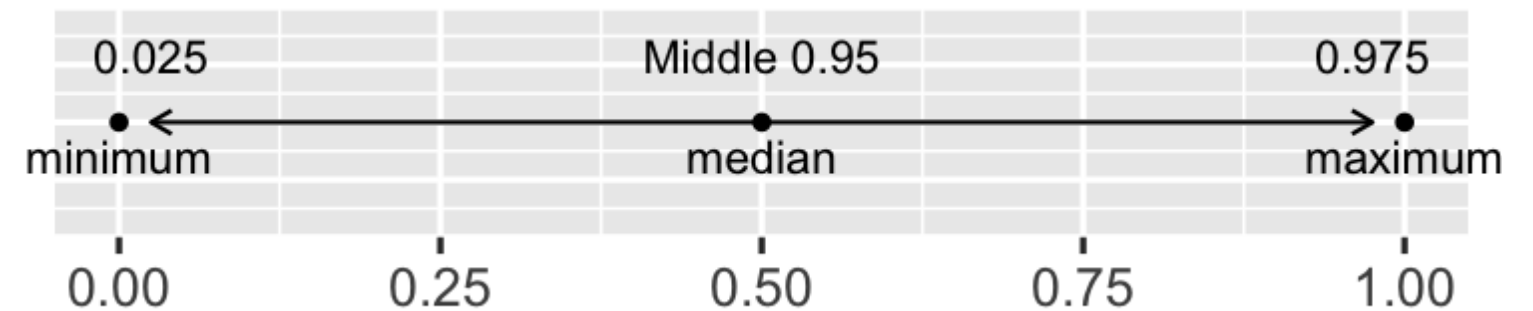# Quantile method for confidence intervals

```python
np.quantile(coffee_boot_distn, 0.025)
```

```
7.4817195
```

```python
np.quantile(coffee_boot_distn, 0.975)
```
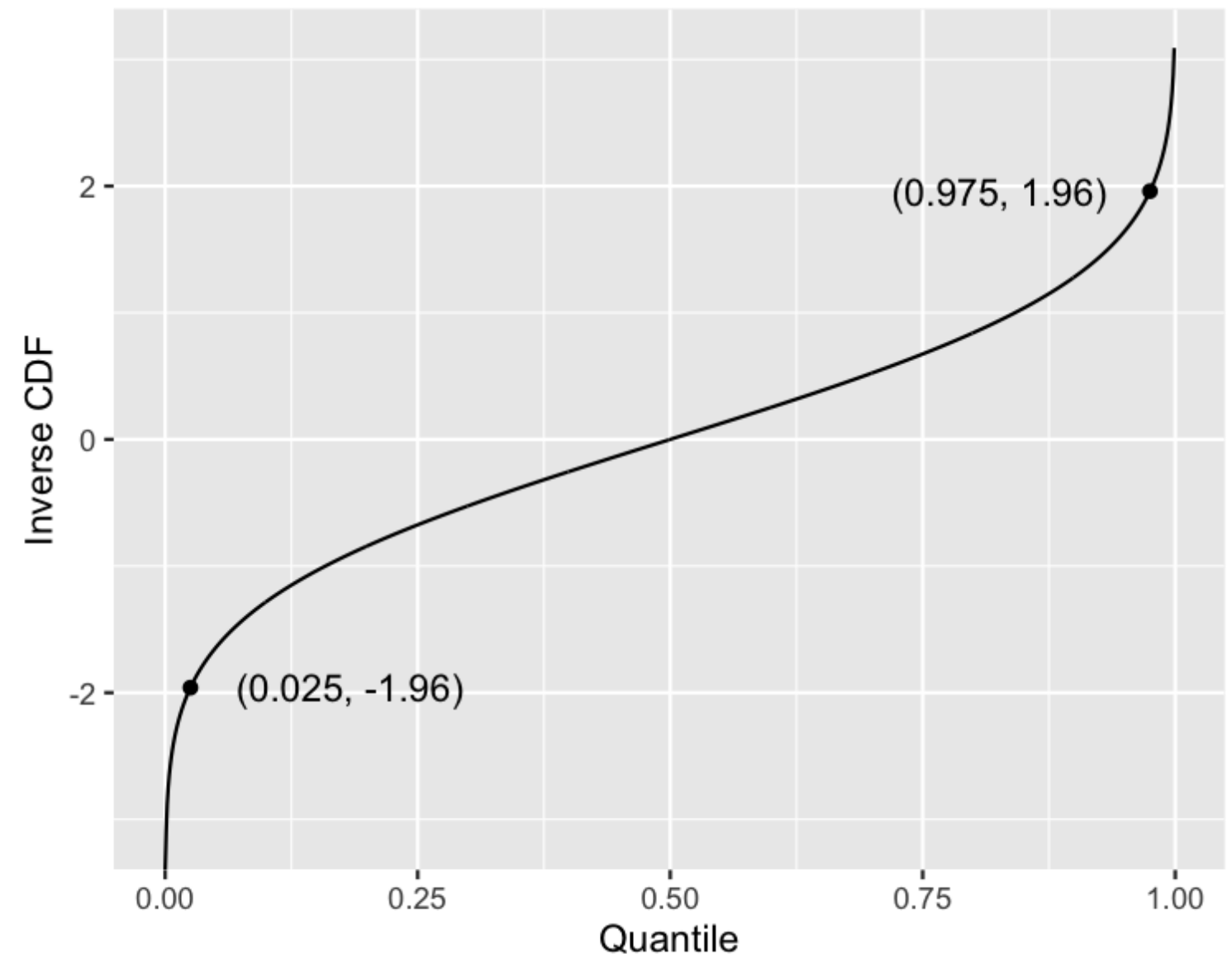
```
7.5448805
```

Inverse cumulative distribution function

- PDF: The bell curve

- CDF: integrate to get area under bell curve

- Inv. CDF: flip x and y axes

Implemented in Python with

```python
from scipy.stats import norm
norm.ppf(quantile, loc=0, scale=1)
```

Standard error method for confidence interval

```python
point_estimate = np.mean(coffee_boot_distn)
```

```
7.513452892
```

```python
std_error = np.std(coffee_boot_distn, ddof=1)
```

```
0.016067182825533724
```

```python
from scipy.stats import norm
lower = norm.ppf(0.025, loc=point_estimate, scale=std_error)
upper = norm.ppf(0.975, loc=point_estimate, scale=std_error)
print((lower, upper))
```
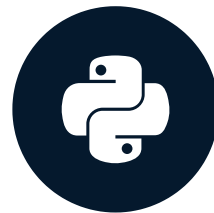
```
(7.481961792328933, 7.544943991671067)
```

Let's practice!

SAMPLING IN PYTHON

Congratulations!

SAMPLING IN PYTHON

James Chapman
Curriculum Manager, DataCamp

Recap

Chapter 1

- Sampling basics

- Selection bias

- Pseudo-random numbers

Chapter 2

- Simple random sampling

- Systematic sampling

- Stratified sampling

- Cluster sampling

Chapter 3

- Sample size and population parameters

- Creating sampling distributions

- Approximate vs. actual sampling dist'ns

- Central limit theorem

Chapter 4

- Bootstrapping from a single sample

- Standard error

- Confidence intervals

The most important things

- The std. deviation of a bootstrap statistic is a good approximation of the *standard error*

- Can assume bootstrap distributions are normally distributed for confidence intervals

What's next?

- **Experimental Design in Python** and **Customer Analytics and A/B Testing in Python**

- **Hypothesis Testing in Python**

- **Foundations of Probability in Python** and **Bayesian Data Analysis in Python**

Happy learning!

SAMPLING IN PYTHON