What is statistics?

INTRODUCTION TO STATISTICS IN PYTHON

Maggie Matsui
Content Developer, DataCamp

What is statistics?

- **The field of statistics** - the practice and study of collecting and analyzing data
- **A summary statistic** - a fact about or summary of some data

What can statistics do?

What is statistics?

- **The field of statistics** - the practice and study of collecting and analyzing data
- **A summary statistic** - a fact about or summary of some data

What can statistics do?

- How likely is someone to purchase a product? Are people more likely to purchase it if they can use a different payment system?
- How many occupants will your hotel have? How can you optimize occupancy?
- How many sizes of jeans need to be manufactured so they can fit 95% of the population? Should the same number of each size be produced?
- A/B tests: Which ad is more effective in getting people to purchase a product?

What can't statistics do?

- *Why* is *Game of Thrones* so popular?

Instead...

- Are series with more violent scenes viewed by more people?

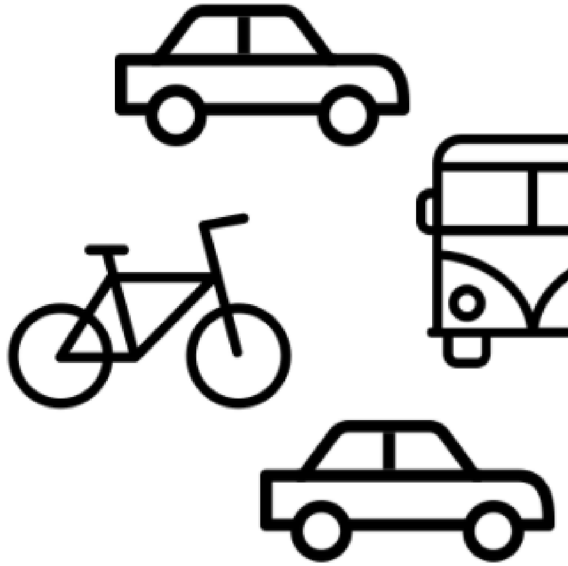
But...

- Even so, this can't tell us if more violent scenes lead to more views

Types of statistics

Descriptive statistics

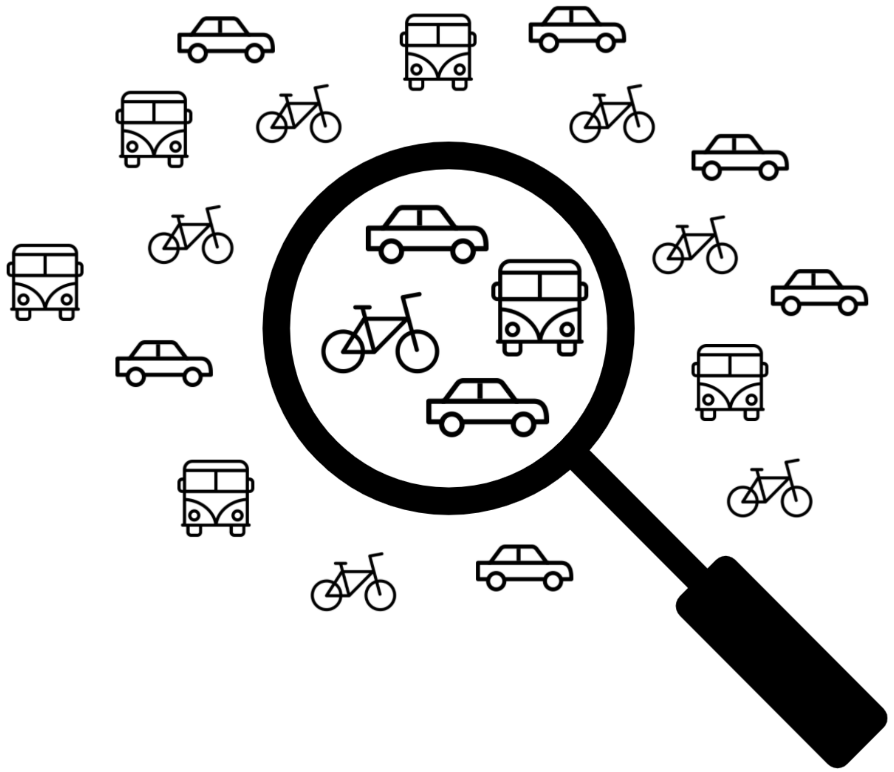
- *Describe* and summarize data



- 50% of friends drive to work
- 25% take the bus
- 25% bike

Inferential statistics

- Use a sample of data to make *inferences* about a larger population



What percent of people drive to work?

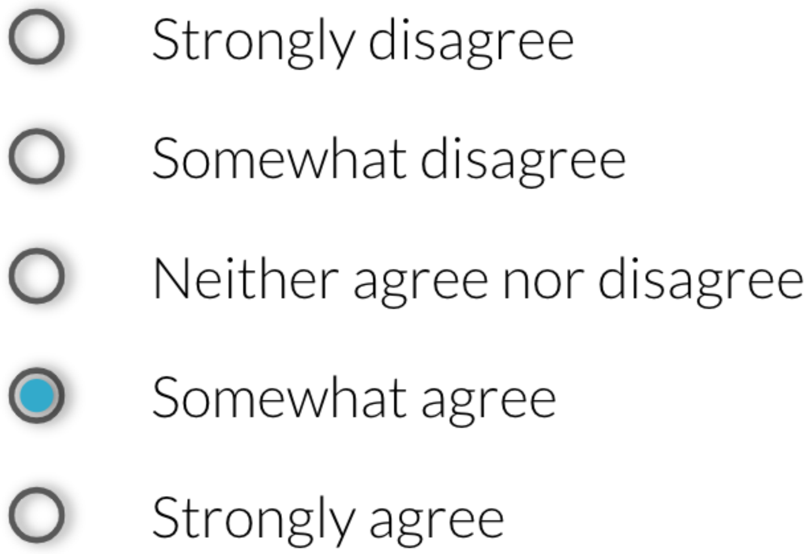
Types of data

Numeric (Quantitative)

- **Continuous (Measured)**
  - Airplane speed
  - Time spent waiting in line
- **Discrete (Counted)**
  - Number of pets
  - Number of packages shipped

Categorical (Qualitative)

- **Nominal (Unordered)**
  - Married/unmarried
  - Country of residence
- **Ordinal (Ordered)**



Categorical data can be represented as numbers

Nominal (Unordered)

- Married/unmarried (`1` / `0`)
- Country of residence (`1` , `2` , ...)

Ordinal (Ordered)

- Strongly disagree (`1`)
- Somewhat disagree (`2`)
- Neither agree nor disagree (`3`)
- Somewhat agree (`4`)
- Strongly agree (`5`)

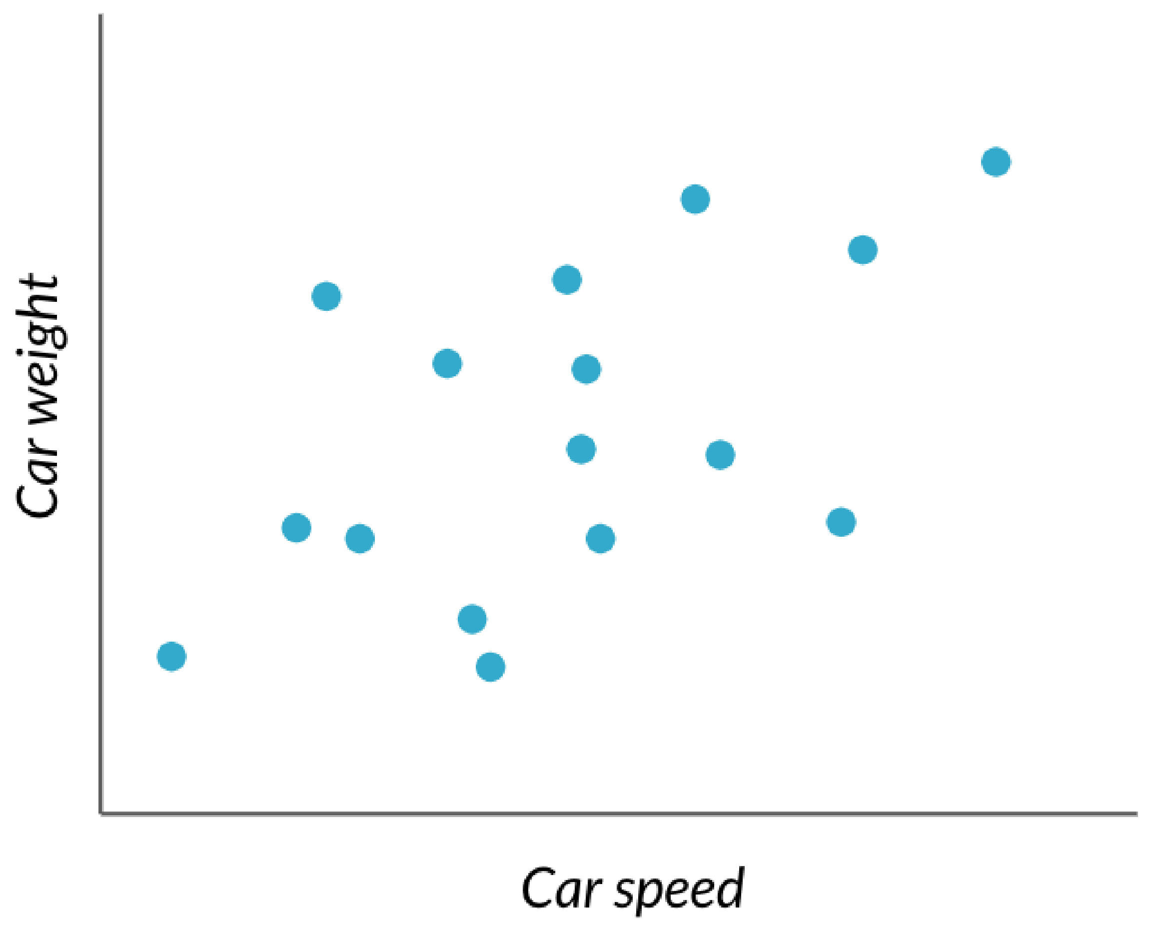
Why does data type matter?

Summary statistics

```
import numpy as np
np.mean(car_speeds['speed_mph'])
```

```
40.09062
```

Plots



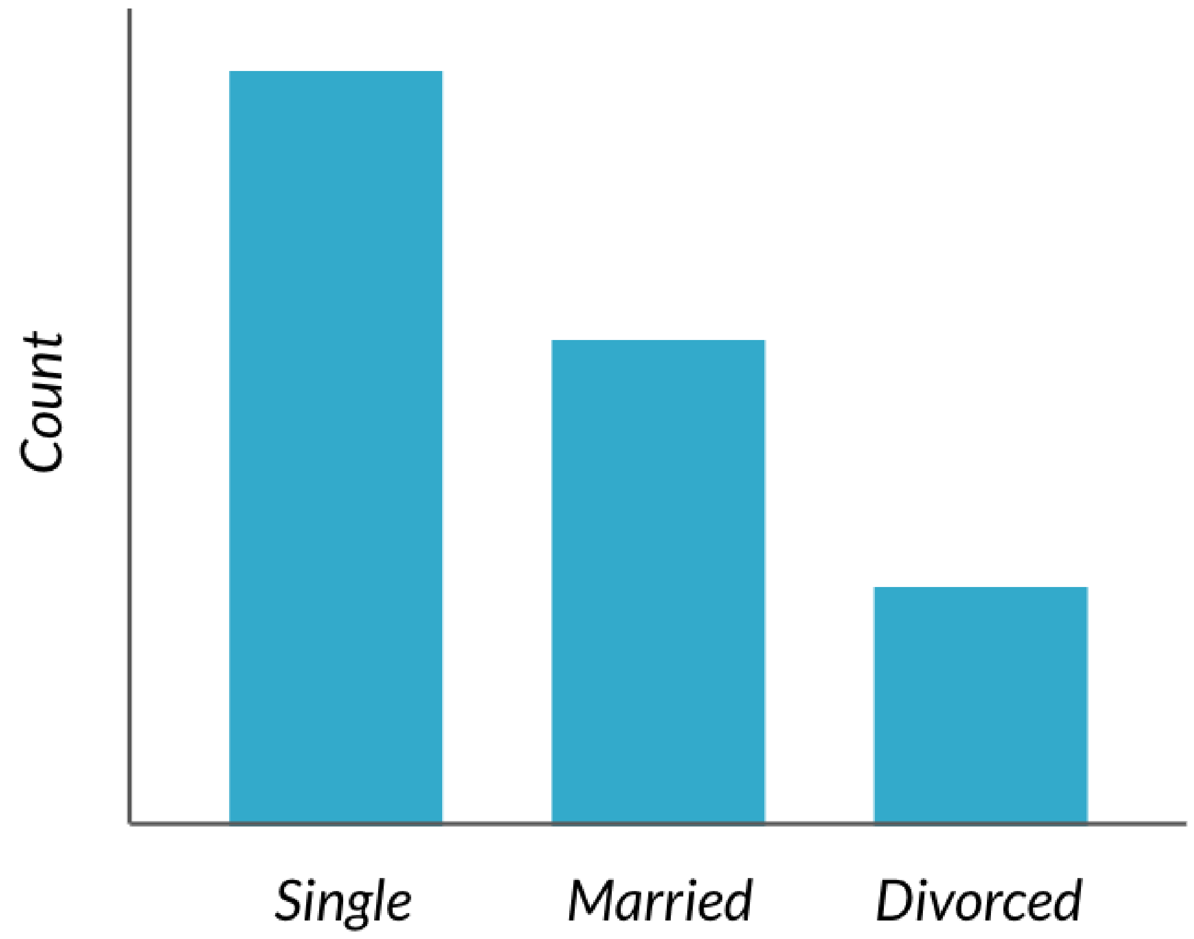
Why does data type matter?

Summary statistics

```
demographics['marriage_status'].value_counts()
```

```
single      188
married     143
divorced    124
dtype: int64
```

Plots



Let's practice!

Measures of center

INTRODUCTION TO STATISTICS IN PYTHON

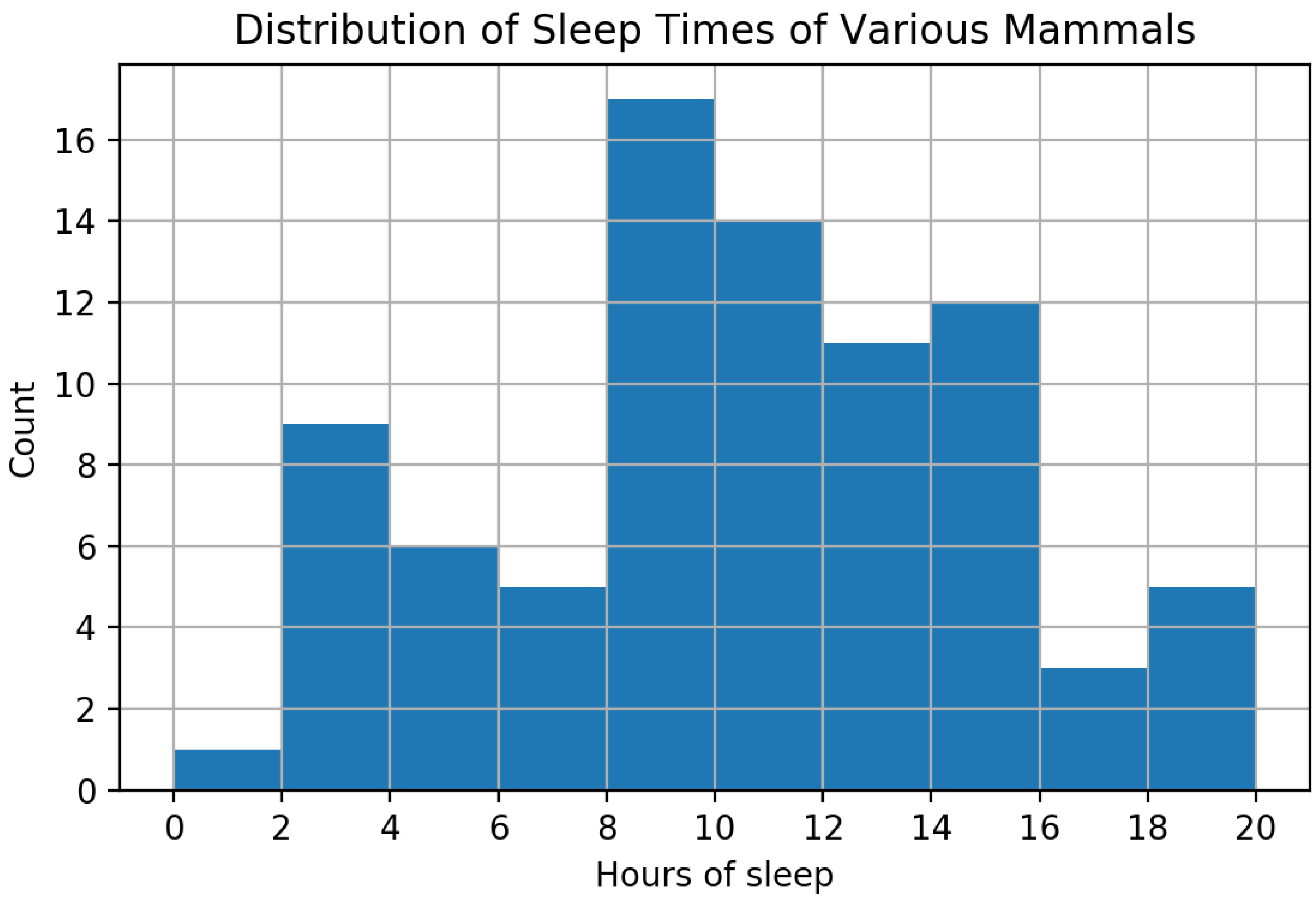
Maggie Matsui
Content Developer, DataCamp

Mammal sleep data

```python
print(msleep)
```

```
                  name       genus   vore         order  ... sleep_cycle  awake  brainwt   bodywt
1              Cheetah    Acinonyx  carni     Carnivora  ...         NaN   11.9      NaN   50.000
2           Owl monkey       Aotus   omni      Primates  ...         NaN    7.0  0.01550    0.480
3      Mountain beaver  Aplodontia  herbi      Rodentia  ...         NaN    9.6      NaN    1.350
4 Greater short-ta...     Blarina   omni  Soricomorpha  ...    0.133333    9.1  0.00029    0.019
5                  Cow         Bos  herbi  Artiodactyla  ...    0.666667   20.0  0.42300  600.000
..                 ...         ...    ...           ...  ...         ...    ...      ...      ...
79          Tree shrew      Tupaia   omni    Scandentia  ...    0.233333   15.1  0.00250    0.104
80 Bottle-nosed do...    Tursiops  carni       Cetacea  ...         NaN   18.8      NaN  173.330
81               Genet     Genetta  carni     Carnivora  ...         NaN   17.7  0.01750    2.000
82          Arctic fox      Vulpes  carni     Carnivora  ...         NaN   11.5  0.04450    3.380
83             Red fox      Vulpes  carni     Carnivora  ...    0.350000   14.2  0.05040    4.230
```

Histograms



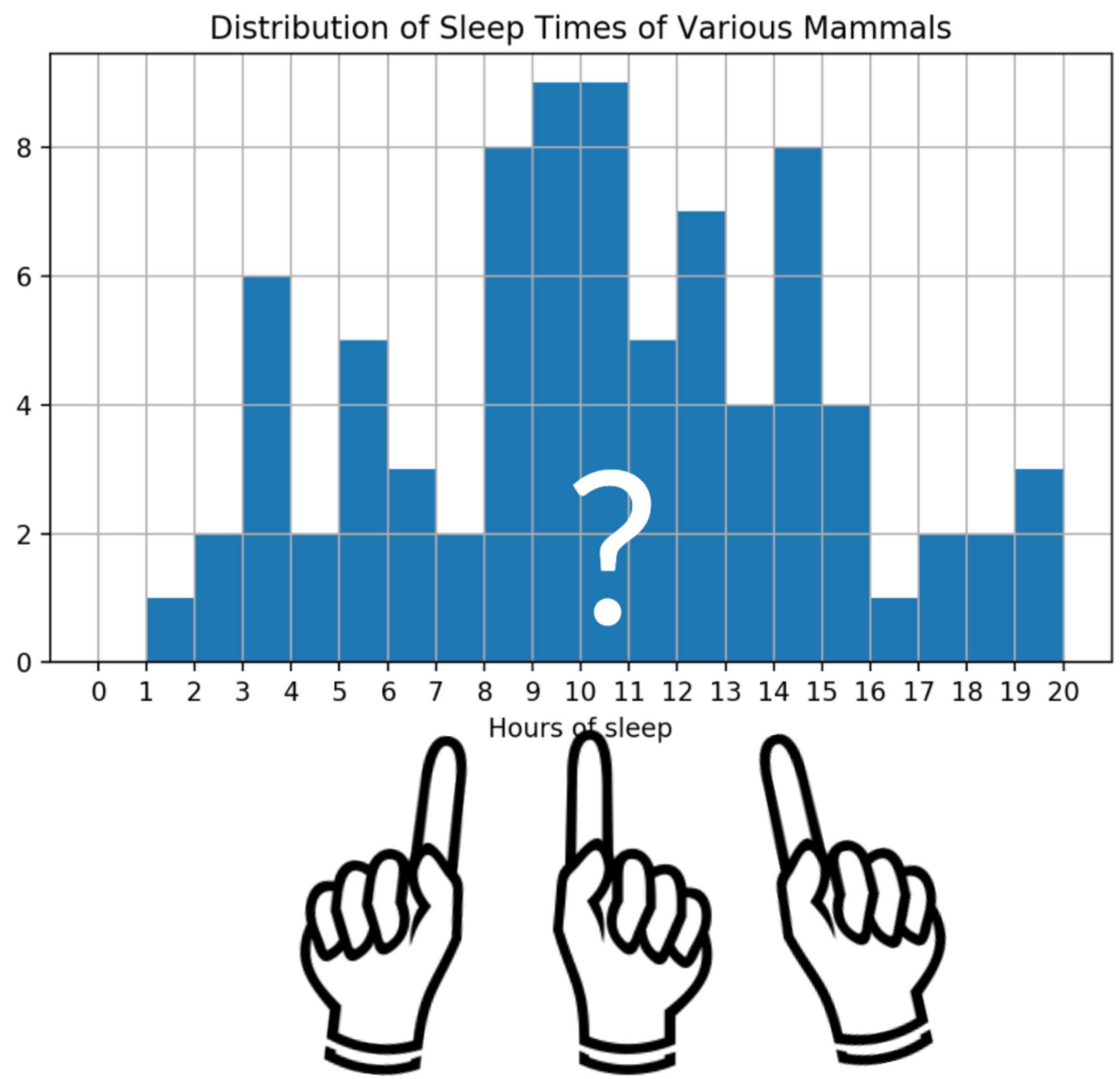
Distribution of Sleep Times of Various Mammals
Count
16
14
12
10
8
6
4
2
0
0
2
4
6
8
10
12
14
16
18
20
Hours of sleep

How long do mammals in this dataset typically sleep?

What's a typical value?

Where is the center of the data?

- Mean
- Median
- Mode



Measures of center: mean

```
                 name  sleep_total
1             Cheetah         12.1
2          Owl monkey         17.0
3     Mountain beaver         14.4
4  Greater short-t...         14.9
5                 Cow          4.0
..                ...          ...
```

```python
import numpy as np
np.mean(msleep['sleep_total'])
```

```
10.43373
```

$$\text{Mean sleep time} =$$

$$\frac{12.1 + 17.0 + 14.4 + 14.9 + ...}{83} = 10.43$$

Measures of center: median

```
msleep['sleep_total'].sort_values()
```

```
29     1.9
30     2.7
22     2.9
9      3.0
23     3.1
      ...
19    18.0
61    18.1
36    19.4
21    19.7
42    19.9
```

```
msleep['sleep_total'].sort_values().iloc[41]
```

```
10.1
```

```
np.median(msleep['sleep_total'])
```

```
10.1
```

Measures of center: mode

Most frequent value

```
msleep['sleep_total'].value_counts()
```

```
12.5    4
10.1    3
14.9    2
11.0    2
8.4     2
...
14.3    1
17.0    1
Name: sleep_total, Length: 65, dtype: int64
```

```
msleep['vore'].value_counts()
```

```
herbi      32
omni       20
carni      19
insecti     5
Name: vore, dtype: int64
```

```
import statistics
statistics.mode(msleep['vore'])
```

```
'herbi'
```

Adding an outlier

```
msleep[msleep['vore'] == 'insecti']
```

	name	genus	vore	order	sleep_total
22	Big brown bat	Eptesicus	insecti	Chiroptera	19.7
43	Little brown bat	Myotis	insecti	Chiroptera	19.9
62	Giant armadillo	Priodontes	insecti	Cingulata	18.1
67	Eastern american mole	Scalopus	insecti	Soricomorpha	8.4

Adding an outlier

```
msleep[msleep['vore'] == "insecti"]['sleep_total'].agg([np.mean, np.median])
```

```
mean      16.53
median    18.9
Name: sleep_total, dtype: float64
```

Adding an outlier

```
msleep[msleep['vore'] == 'insecti']
```

```
                    name       genus     vore        order  sleep_total
22         Big brown bat   Eptesicus  insecti   Chiroptera         19.7
43       Little brown bat      Myotis  insecti   Chiroptera         19.9
62        Giant armadillo  Priodontes  insecti    Cingulata         18.1
67  Eastern american mole    Scalopus  insecti  Soricomorpha        8.4
84    Mystery insectivore         ...  insecti          ...          0.0
```

Adding an outlier

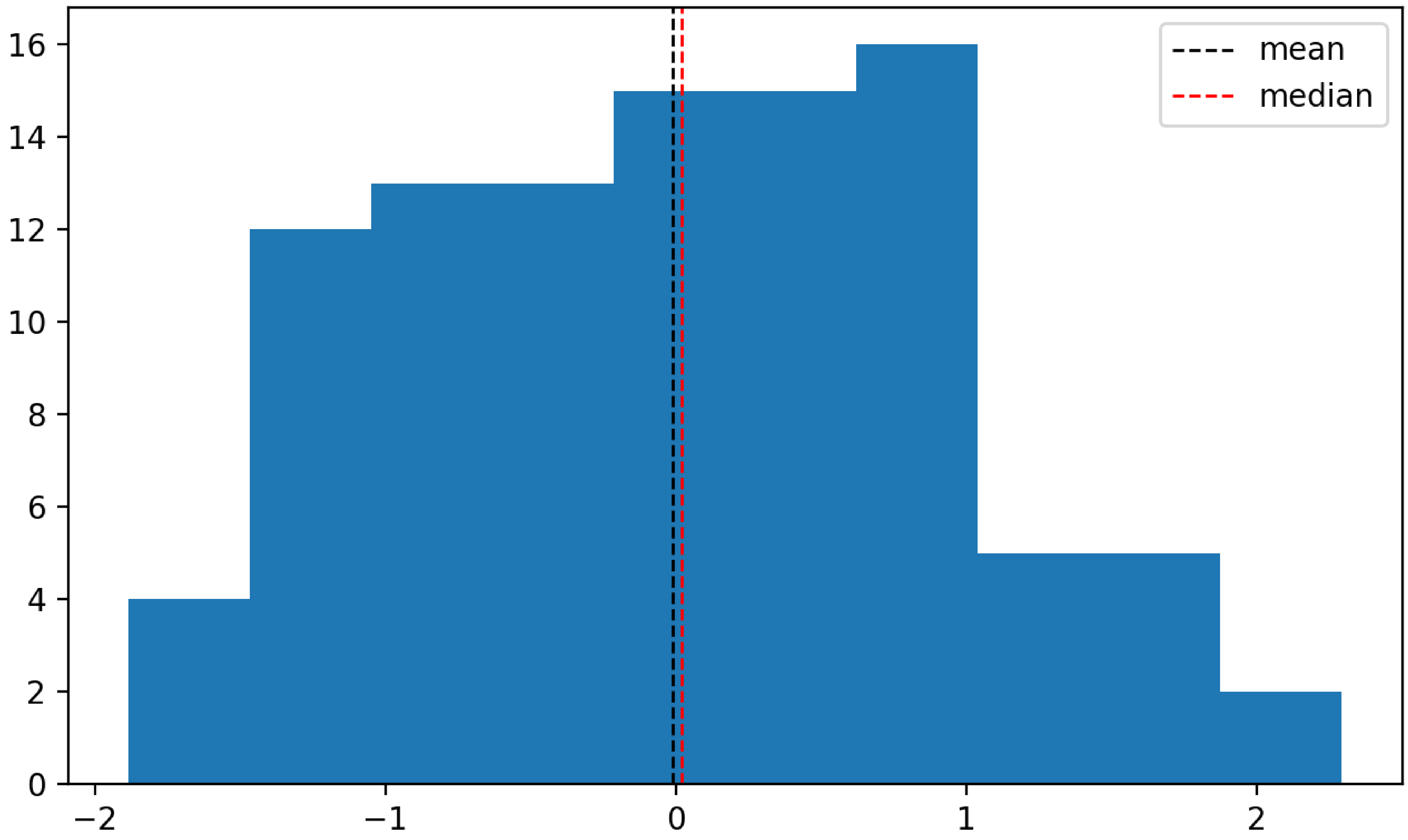
```
msleep[msleep['vore'] == "insecti"]['sleep_total'].agg([np.mean, np.median])
```

```
mean      13.22
median    18.1
Name: sleep_total, dtype: float64
```

Mean: 16.5 → 13.2

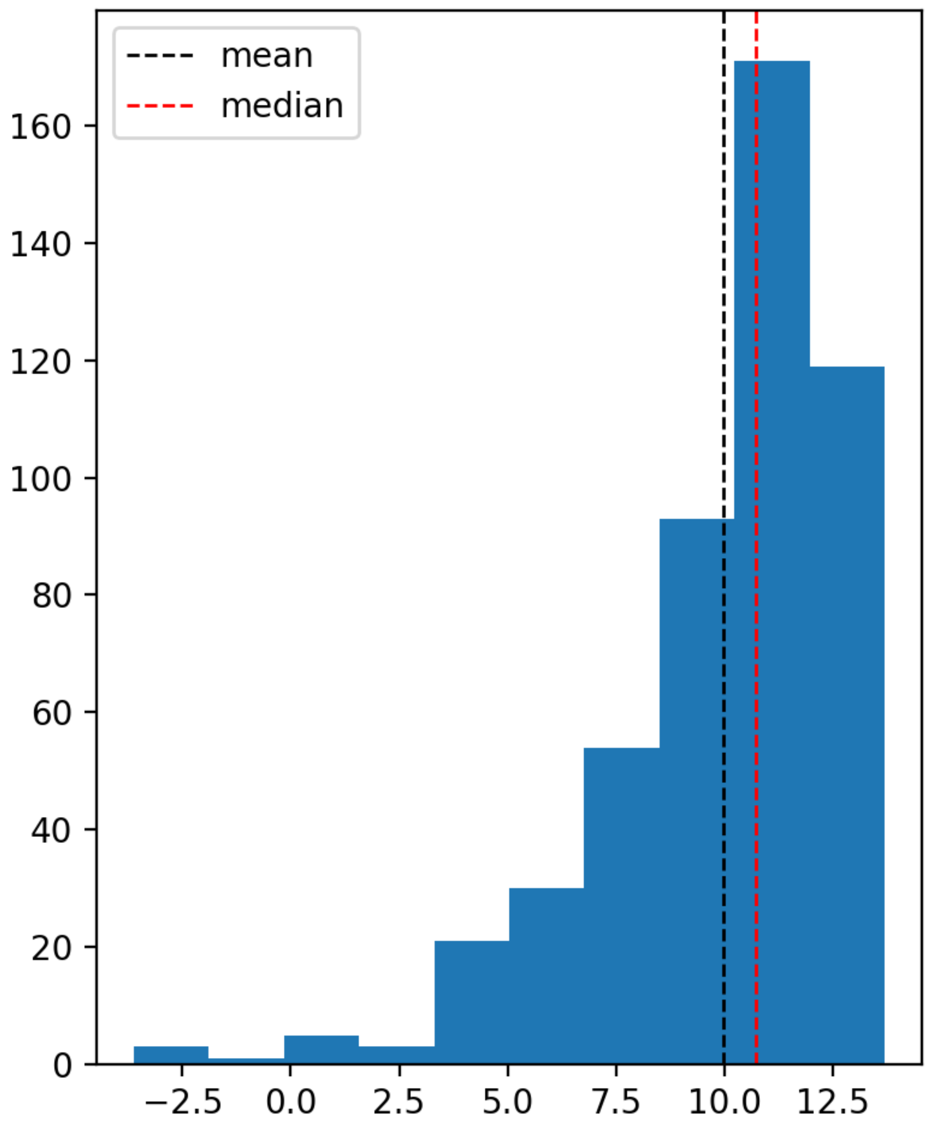
Median: 18.9 → 18.1

Which measure to use?

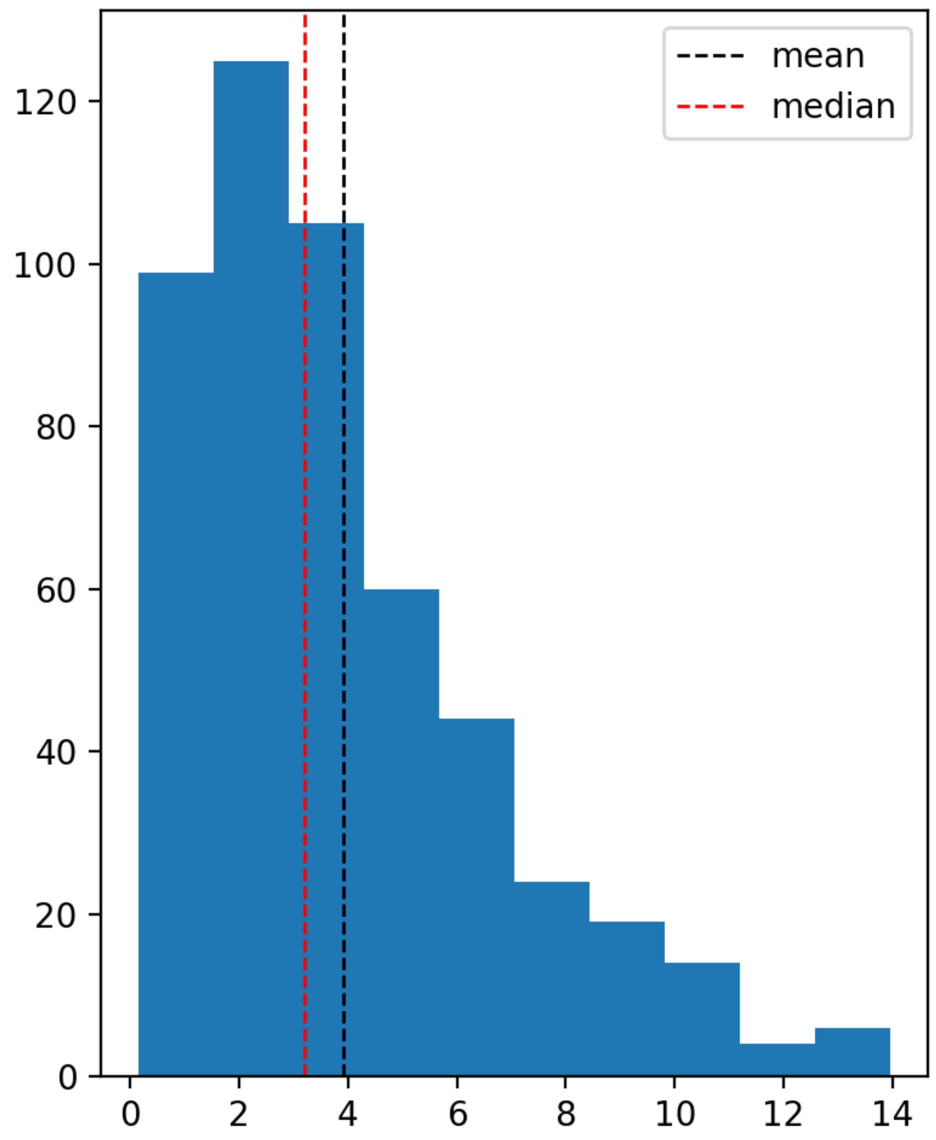


Skew

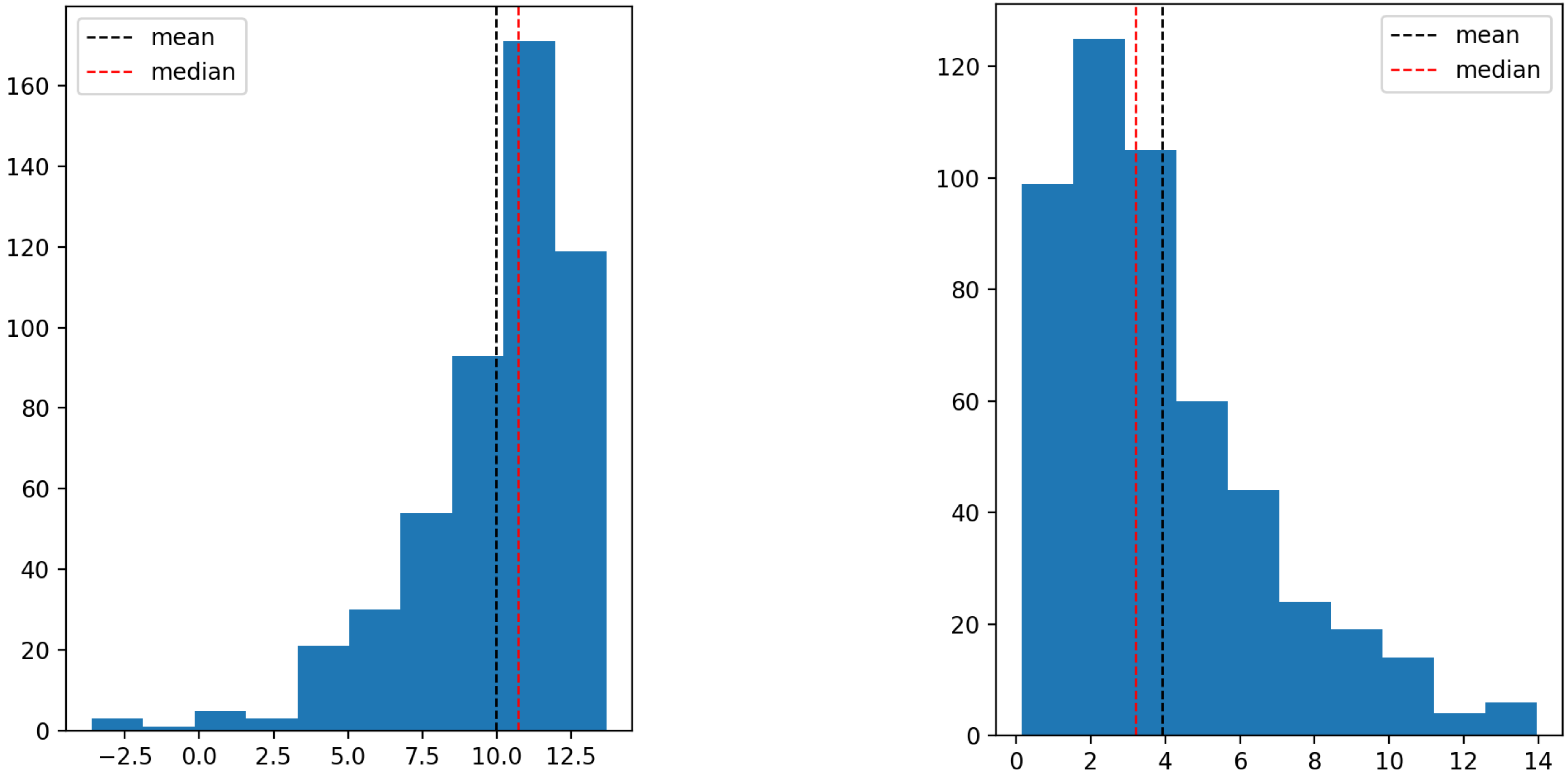
Left-skewed



Right-skewed



Which measure to use?



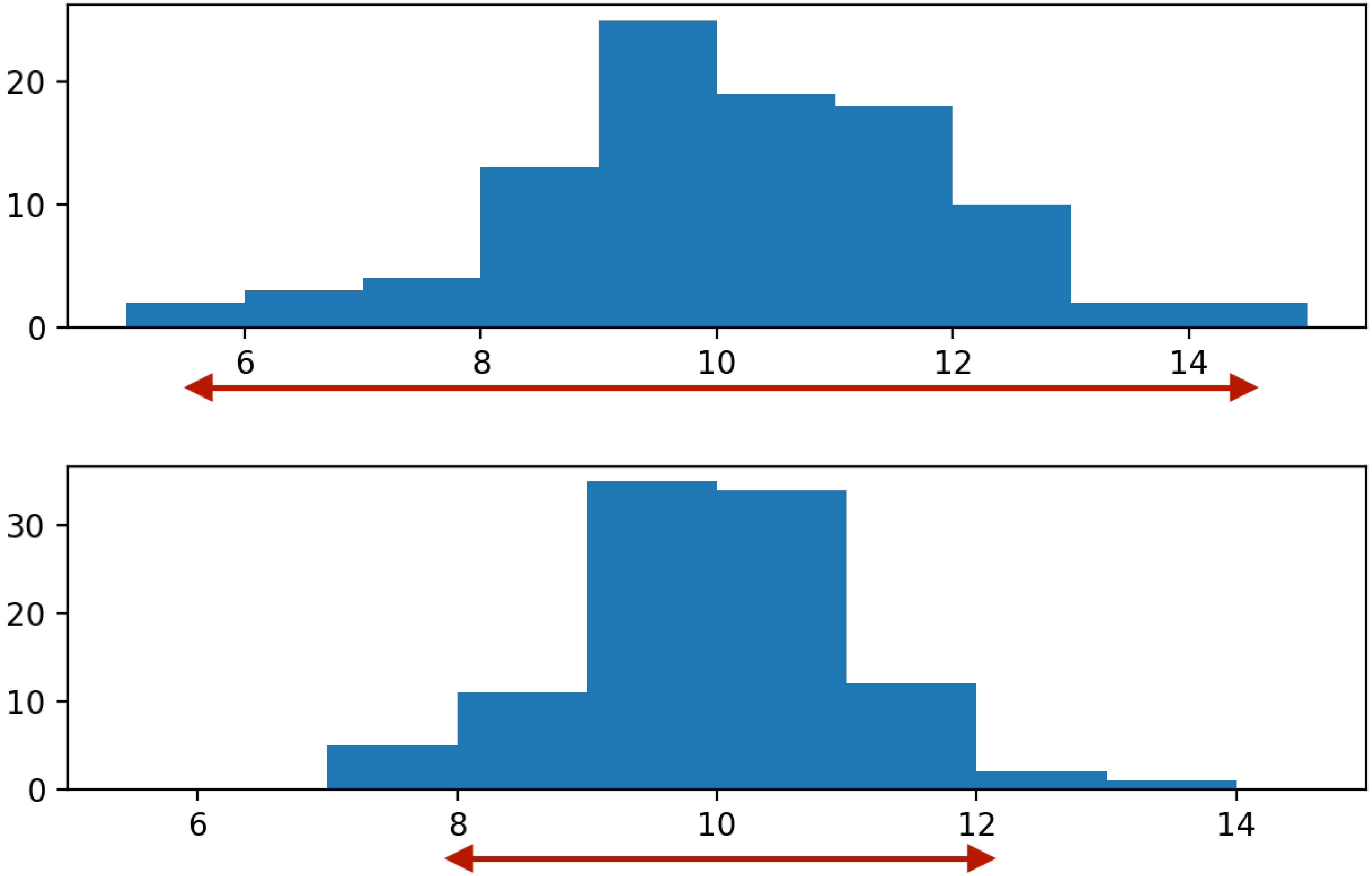
Let's practice!

Measures of spread

INTRODUCTION TO STATISTICS IN PYTHON

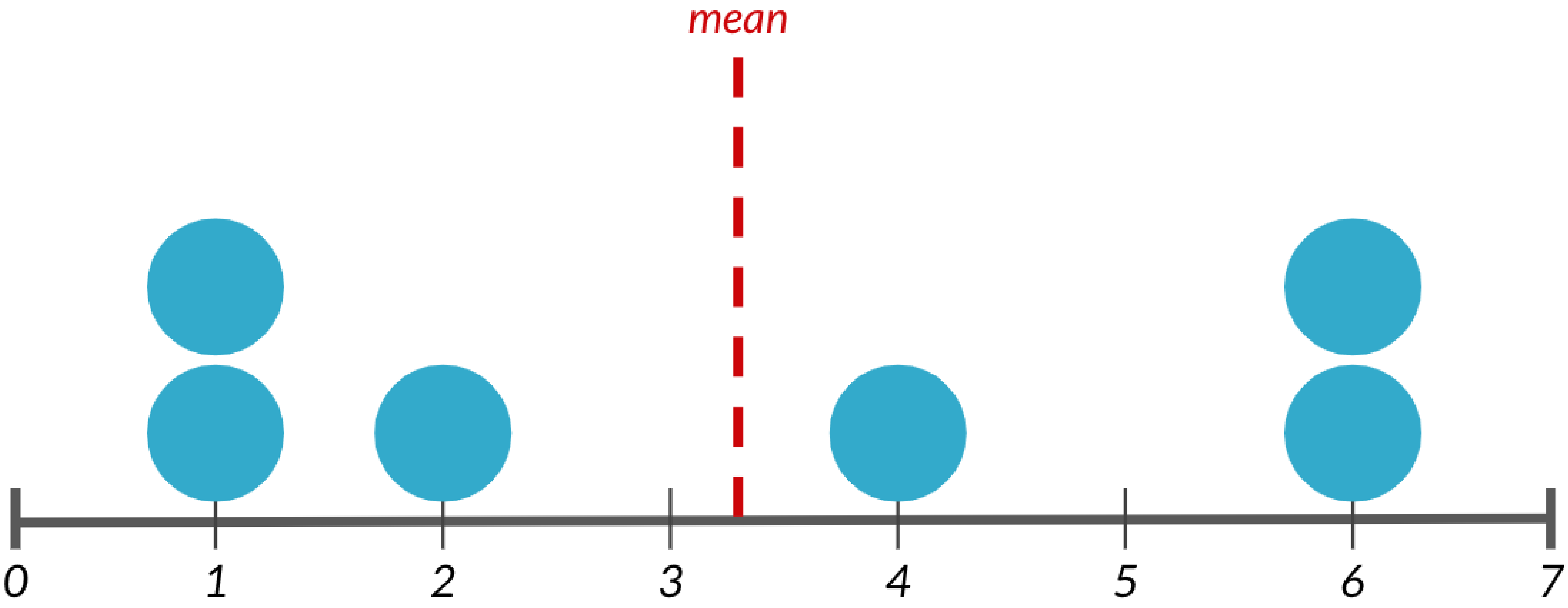
Maggie Matsui
Content Developer, DataCamp

What is spread?



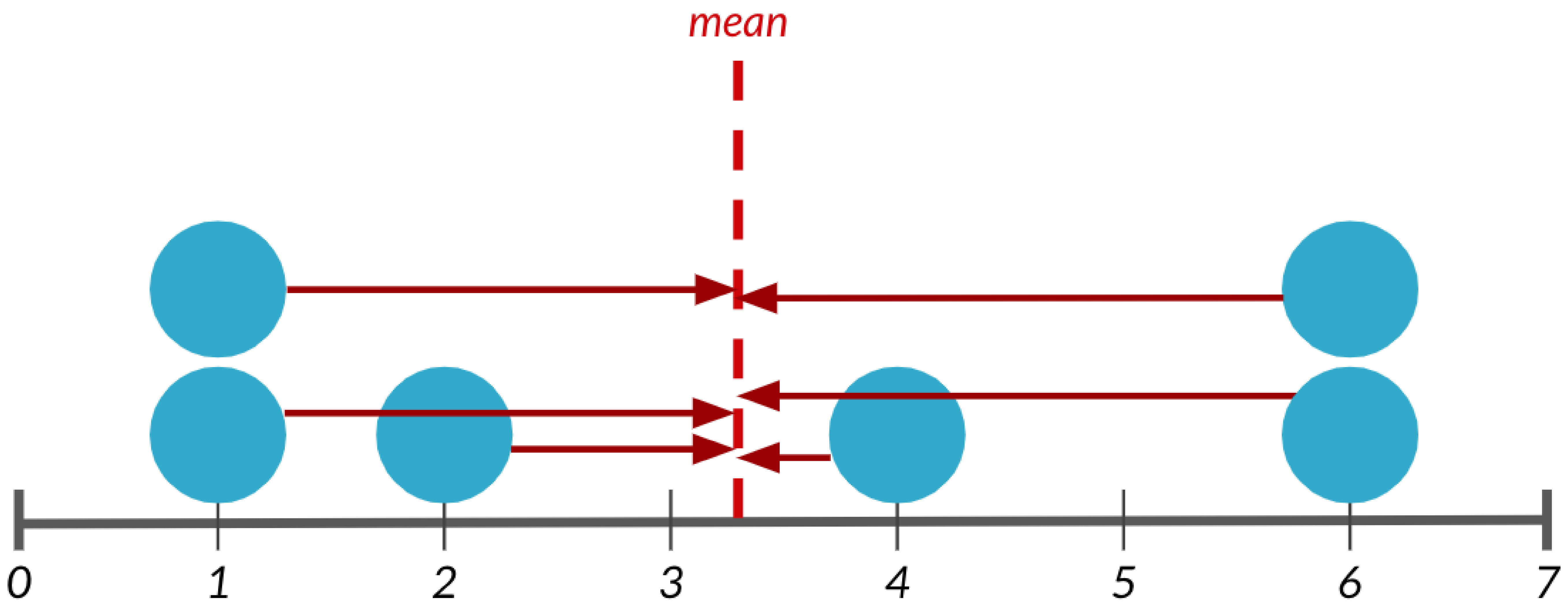
Variance

Average distance from each data point to the data's mean



Variance

Average distance from each data point to the data's mean



Calculating variance

1. Subtract mean from each data point

```
dists = msleep['sleep_total'] -
          np.mean(msleep['sleep_total'])
print(dists)
```

```
0      1.666265
1      6.566265
2      3.966265
3      4.466265
4     -6.433735
        ...
```

2. Square each distance

```
sq_dists = dists ** 2
print(sq_dists)
```

```
0       2.776439
1      43.115837
2      15.731259
3      19.947524
4      41.392945
       ...
```

Calculating variance

3. Sum squared distances

```
sum_sq_dists = np.sum(sq_dists)
print(sum_sq_dists)
```

```
1624.065542
```

4. Divide by number of data points - 1

```
variance = sum_sq_dists / (83 - 1)
print(variance)
```

```
19.805677
```

Use `np.var()`

```
np.var(msleep['sleep_total'], ddof=1)
```

```
19.805677
```

*Without `ddof=1` , **population variance** is calculated instead of **sample variance**:*

```
np.var(msleep['sleep_total'])
```

```
19.567055
```

Standard deviation

```
np.sqrt(np.var(msleep['sleep_total'], ddof=1))
```

```
4.450357
```

```
np.std(msleep['sleep_total'], ddof=1)
```

```
4.450357
```

Mean absolute deviation

```
dists = msleep['sleep_total'] - mean(msleep$sleep_total)
np.mean(np.abs(dists))
```

```
3.566701
```

Standard deviation vs. mean absolute deviation

- Standard deviation squares distances, penalizing longer distances more than shorter ones.
- Mean absolute deviation penalizes each distance equally.
- One isn't better than the other, but SD is more common than MAD.

Quantiles

```
np.quantile(msleep['sleep_total'], 0.5)
```

```
10.1
```

0.5 quantile = median

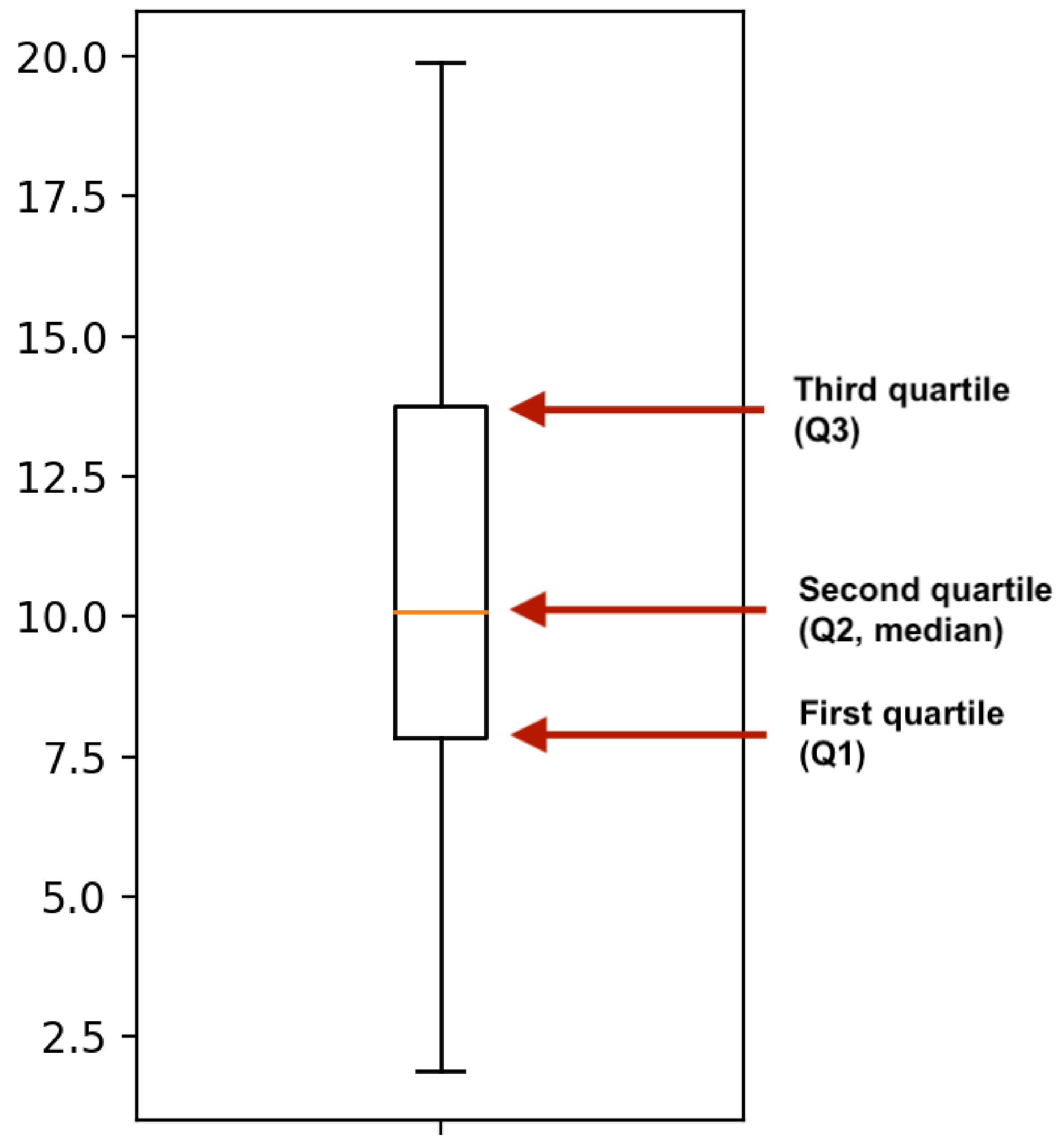
Quartiles:

```
np.quantile(msleep['sleep_total'], [0, 0.25, 0.5, 0.75, 1])
```

```
array([ 1.9 ,  7.85, 10.1 , 13.75, 19.9 ])
```

Boxplots use quartiles

```
import matplotlib.pyplot as plt
plt.boxplot(msleep['sleep_total'])
plt.show()
```



Quantiles using np.linspace()

```
np.quantile(msleep['sleep_total'], [0, 0.2, 0.4, 0.6, 0.8, 1])
```

```
array([ 1.9 ,  6.24,  9.48, 11.14, 14.4 , 19.9 ])
```

`np.linspace(start, stop, num)`

```
np.quantile(msleep['sleep_total'], np.linspace(0, 1, 5))
```

```
array([ 1.9 ,  7.85, 10.1 , 13.75, 19.9 ])
```

Interquartile range (IQR)

Height of the box in a boxplot

```
np.quantile(msleep['sleep_total'], 0.75) - np.quantile(msleep['sleep_total'], 0.25)
```

```
5.9
```

```
from scipy.stats import iqr
iqr(msleep['sleep_total'])
```

```
5.9
```

Outliers

Outlier: data point that is substantially different from the others

How do we know what a substantial difference is? A data point is an outlier if:

- $\text{data} < \text{Q1} - 1.5 \times \text{IQR}$ or
- $\text{data} > \text{Q3} + 1.5 \times \text{IQR}$

Finding outliers

```
from scipy.stats import iqr
iqr = iqr(msleep['bodywt'])
lower_threshold = np.quantile(msleep['bodywt'], 0.25) - 1.5 * iqr
upper_threshold = np.quantile(msleep['bodywt'], 0.75) + 1.5 * iqr
```

```
msleep[(msleep['bodywt'] < lower_threshold) | (msleep['bodywt'] > upper_threshold)]
```

```
                    name   vore  sleep_total    bodywt
4                    Cow  herbi          4.0   600.000
20        Asian elephant  herbi          3.9  2547.000
22                 Horse  herbi          2.9   521.000
...
```

All in one go

```
msleep['bodywt'].describe()
```

```
count      83.000000
mean      166.136349
std       786.839732
min         0.005000
25%         0.174000
50%         1.670000
75%        41.750000
max      6654.000000
Name: bodywt, dtype: float64
```

Let's practice!

INTRODUCTION TO STATISTICS IN PYTHON

What are the chances?

INTRODUCTION TO STATISTICS IN PYTHON

Maggie Matsui
Content Developer, DataCamp

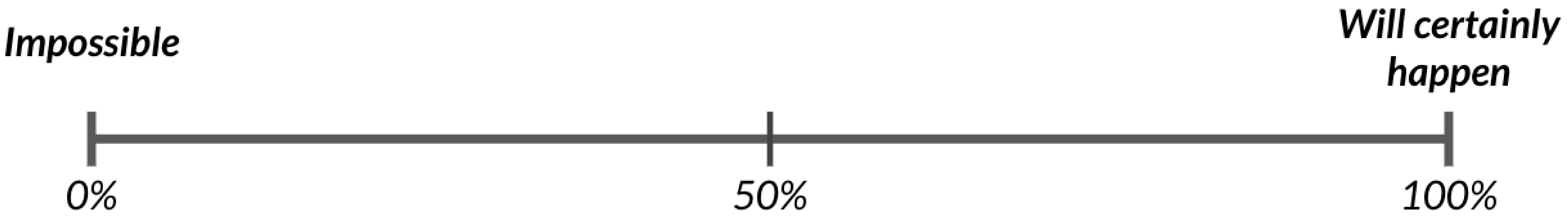
Measuring chance

What's the probability of an event?

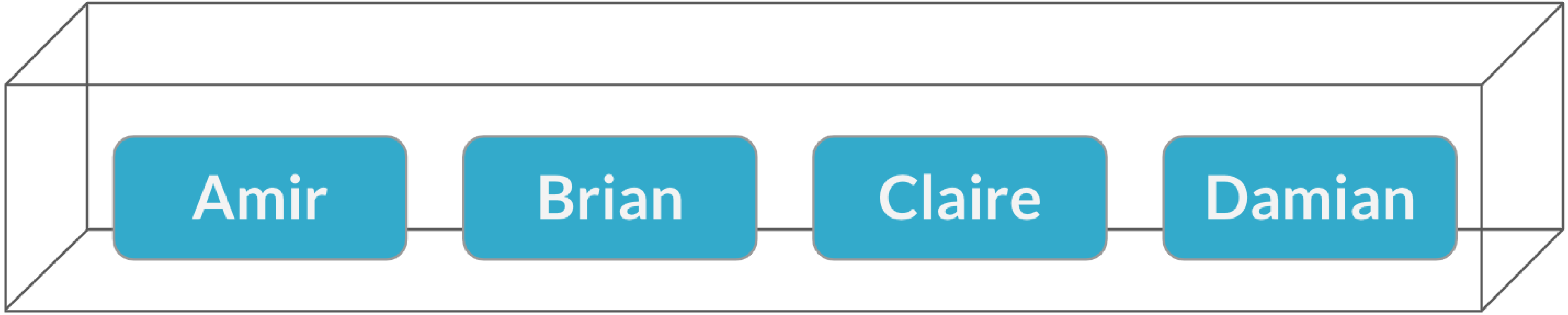
$$P(\text{event}) = \frac{\#\text{ ways event can happen}}{\text{total }\#\text{ of possible outcomes}}$$

Example: a coin flip

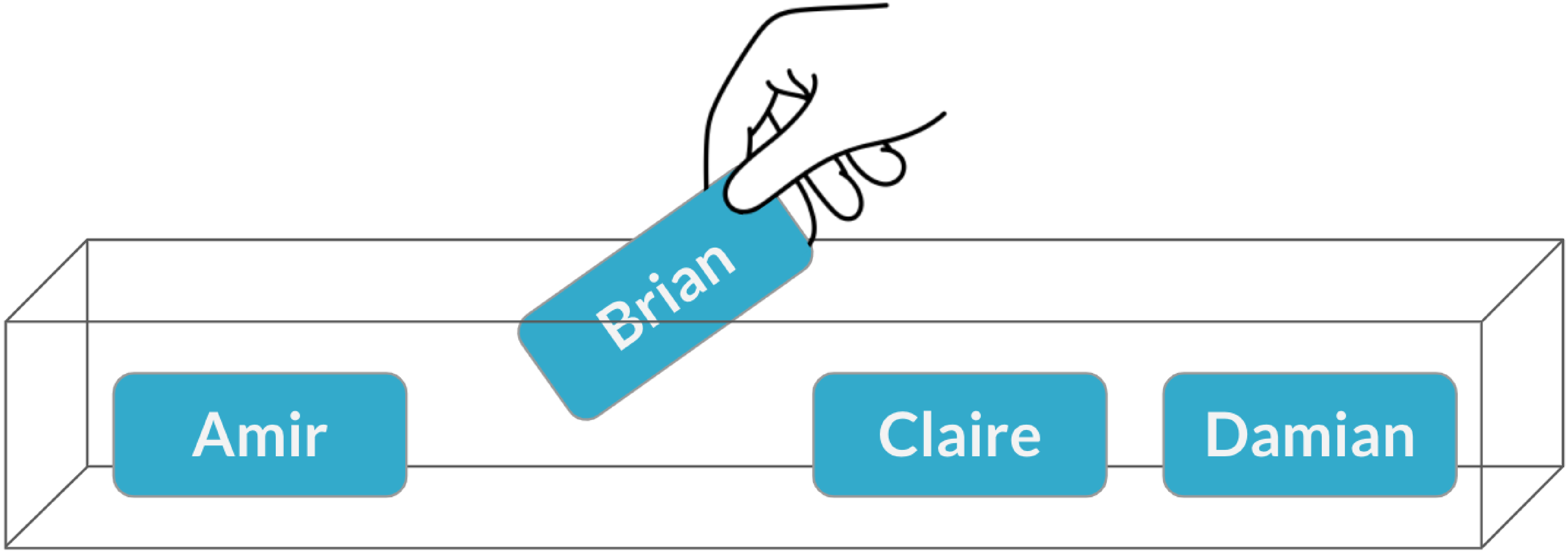
$$P(\text{heads}) = \frac{1 \text{ way to get heads}}{2 \text{ possible outcomes}} = \frac{1}{2} = 50\%$$



Assigning salespeople



Assigning salespeople



$$P(\text{Brian}) = \frac{1}{4} = 25\%$$

Sampling from a DataFrame

```
print(sales_counts)
```

```
     name  n_sales
0    Amir      178
1   Brian      128
2  Claire       75
3  Damian       69
```

```
sales_counts.sample()
```

```
    name  n_sales
1  Brian      128
```

```
sales_counts.sample()
```

```
     name  n_sales
2  Claire       75
```

Setting a random seed

```
np.random.seed(10)
sales_counts.sample()
```

```
    name  n_sales
1  Brian      128
```

```
np.random.seed(10)
sales_counts.sample()
```

```
    name  n_sales
1  Brian      128
```

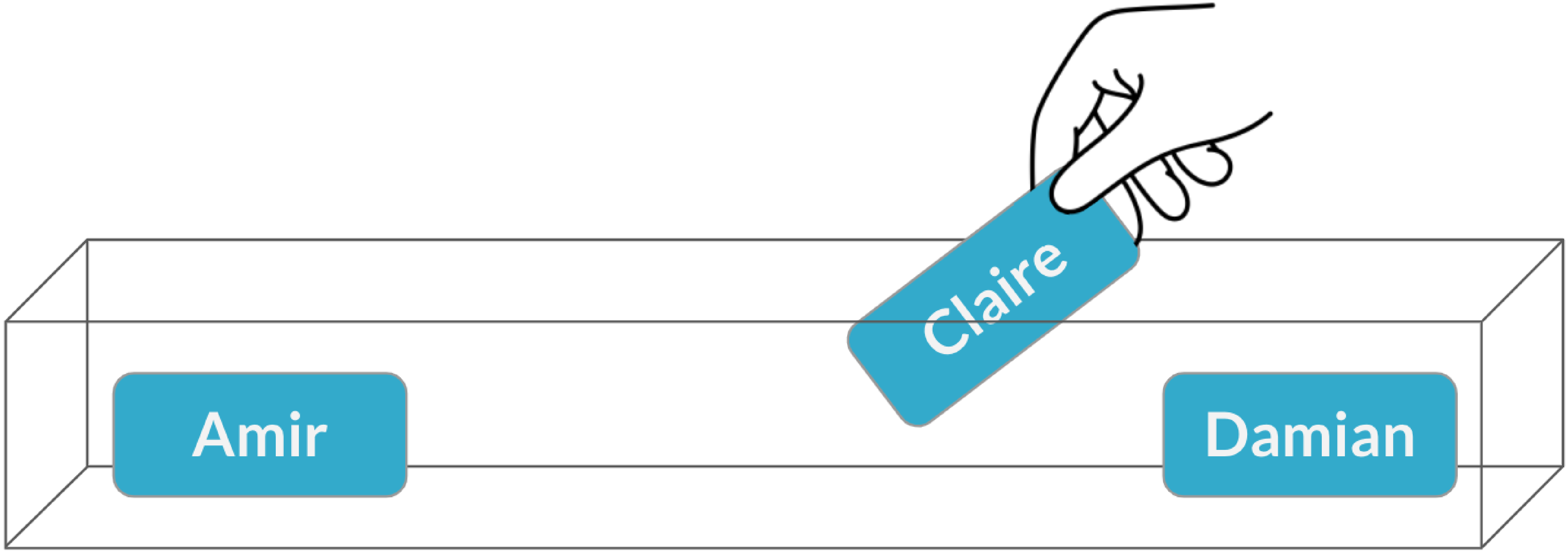
```
np.random.seed(10)
sales_counts.sample()
```

```
    name  n_sales
1  Brian      128
```

A second meeting

Sampling without replacement

A second meeting



$$P(\text{Claire}) = \frac{1}{3} = 33\%$$

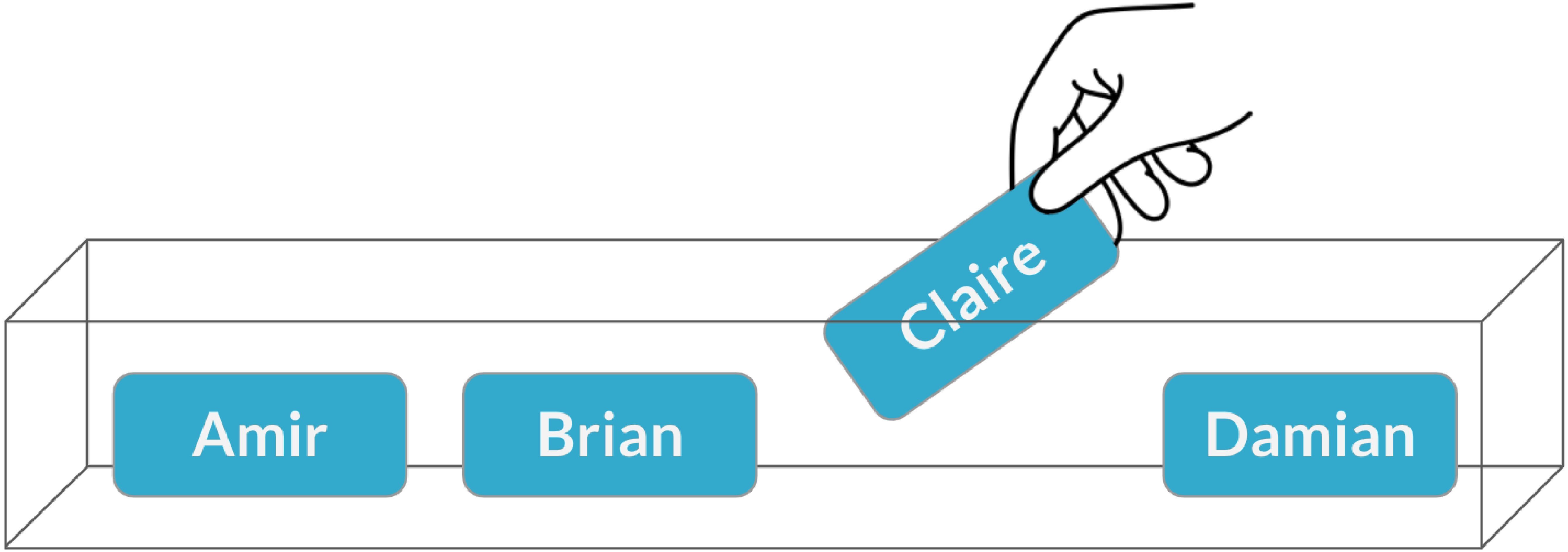
Sampling twice in Python

```
sales_counts.sample(2)
```

```
     name  n_sales
1   Brian      128
2  Claire       75
```

Sampling with replacement

Sampling with replacement



$$P(\text{Claire}) = \frac{1}{4} = 25\%$$

Sampling with/without replacement in Python

```
sales_counts.sample(5, replace = True)
```

```
     name  n_sales
1   Brian      128
2  Claire       75
1   Brian      128
3  Damian       69
0    Amir      178
```

Independent events

Two events are ***independent*** *if the probability of the second event* ***isn't*** *affected by the outcome of the first event.*

Sampling with Replacement

First pick *Second pick*

Amir

Brian

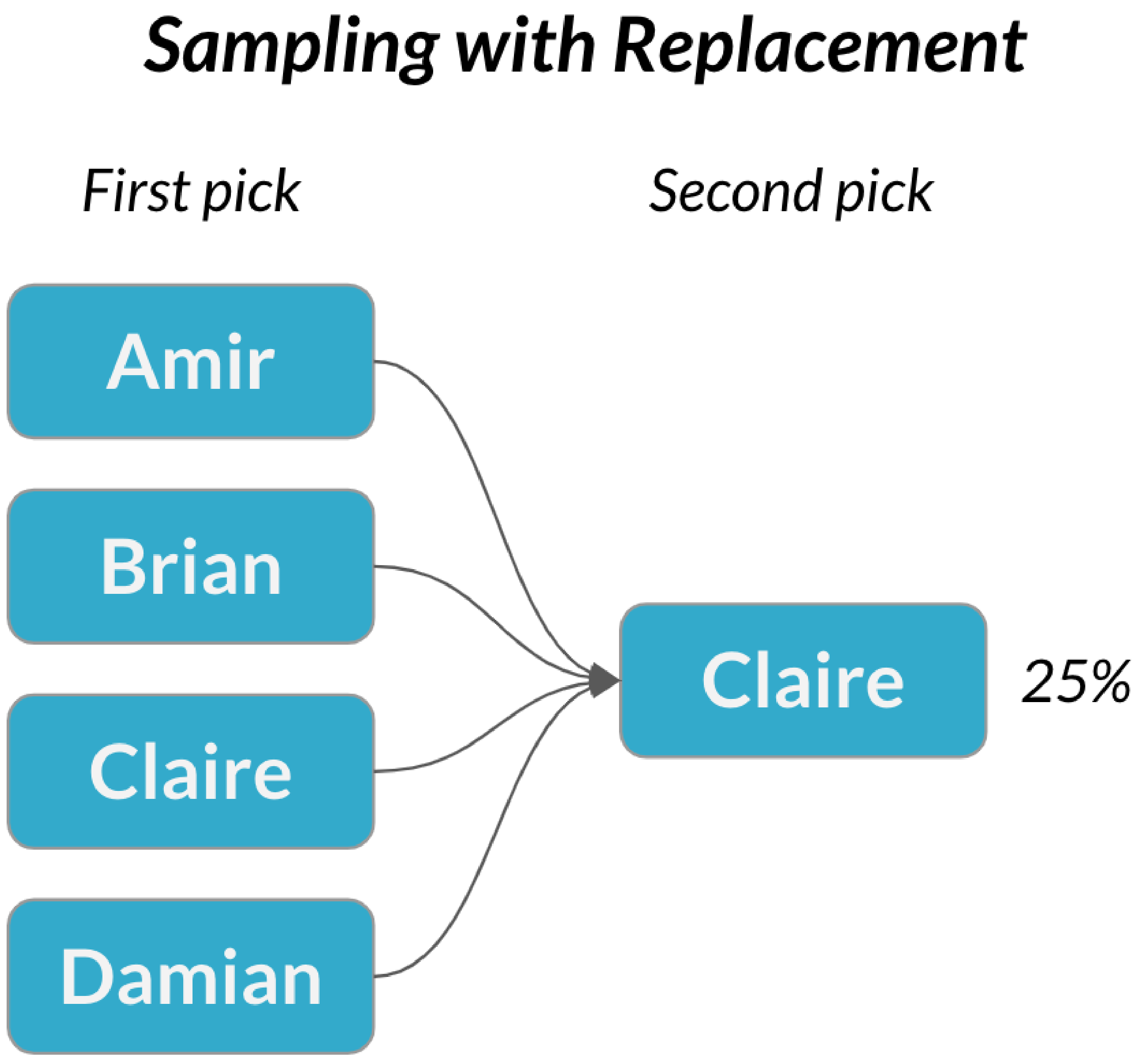
Claire

Damian

Independent events

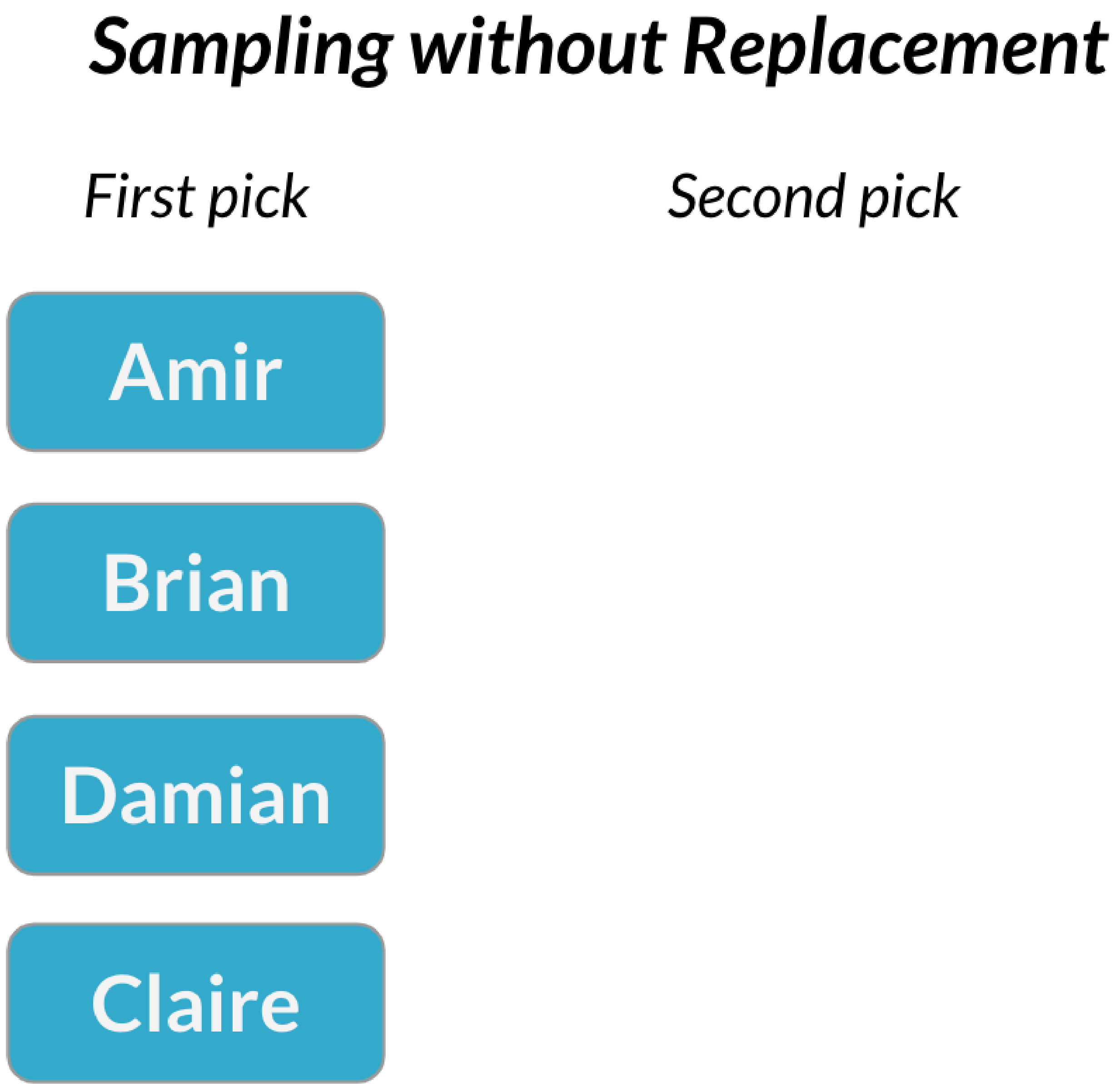
*Two events are **independent** if the probability of the second event **isn't** affected by the outcome of the first event.*

Sampling with replacement = each pick is independent



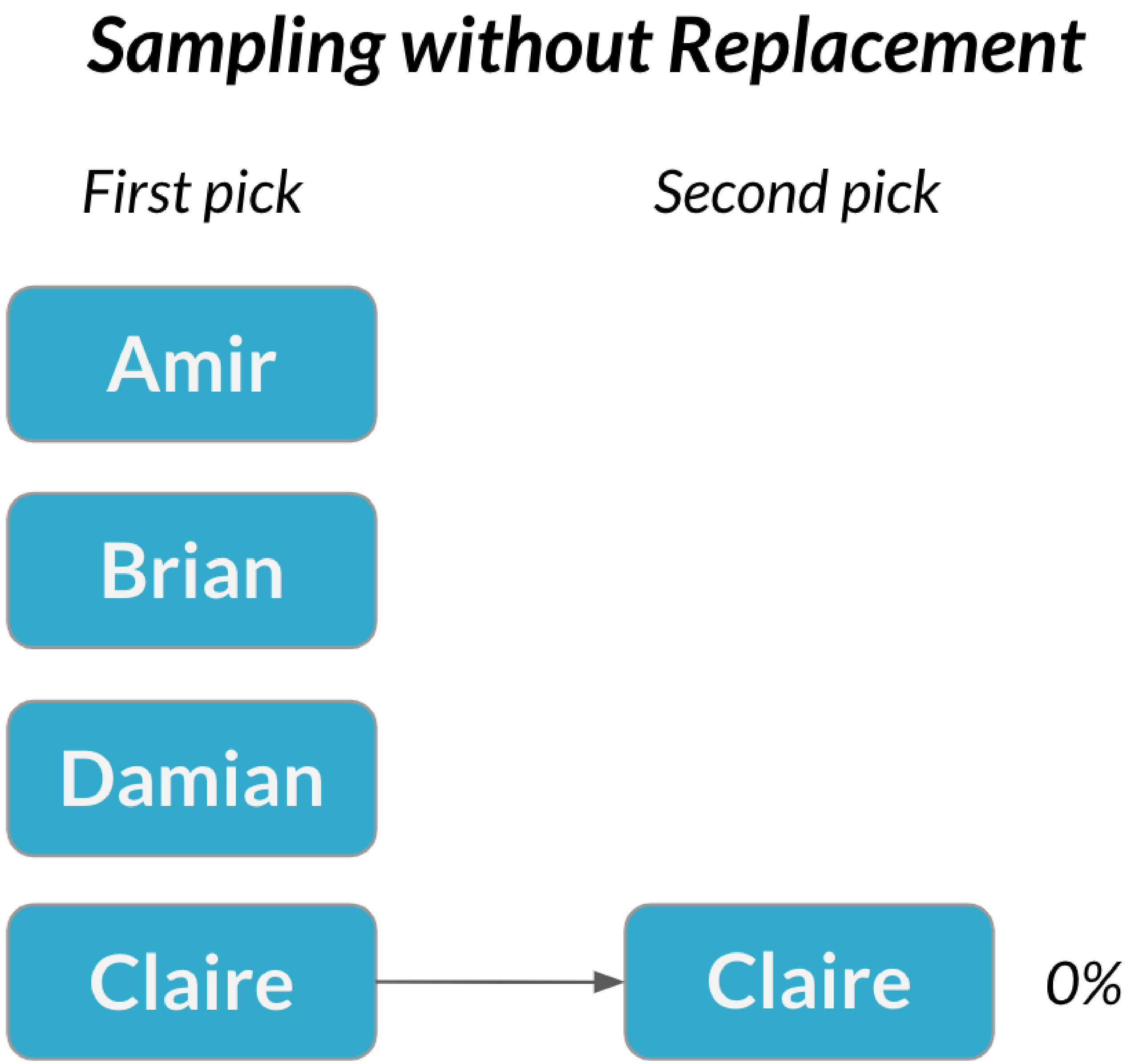
Dependent events

*Two events are **dependent** if the probability of the second event **is** affected by the the outcome of the first event.*



Dependent events

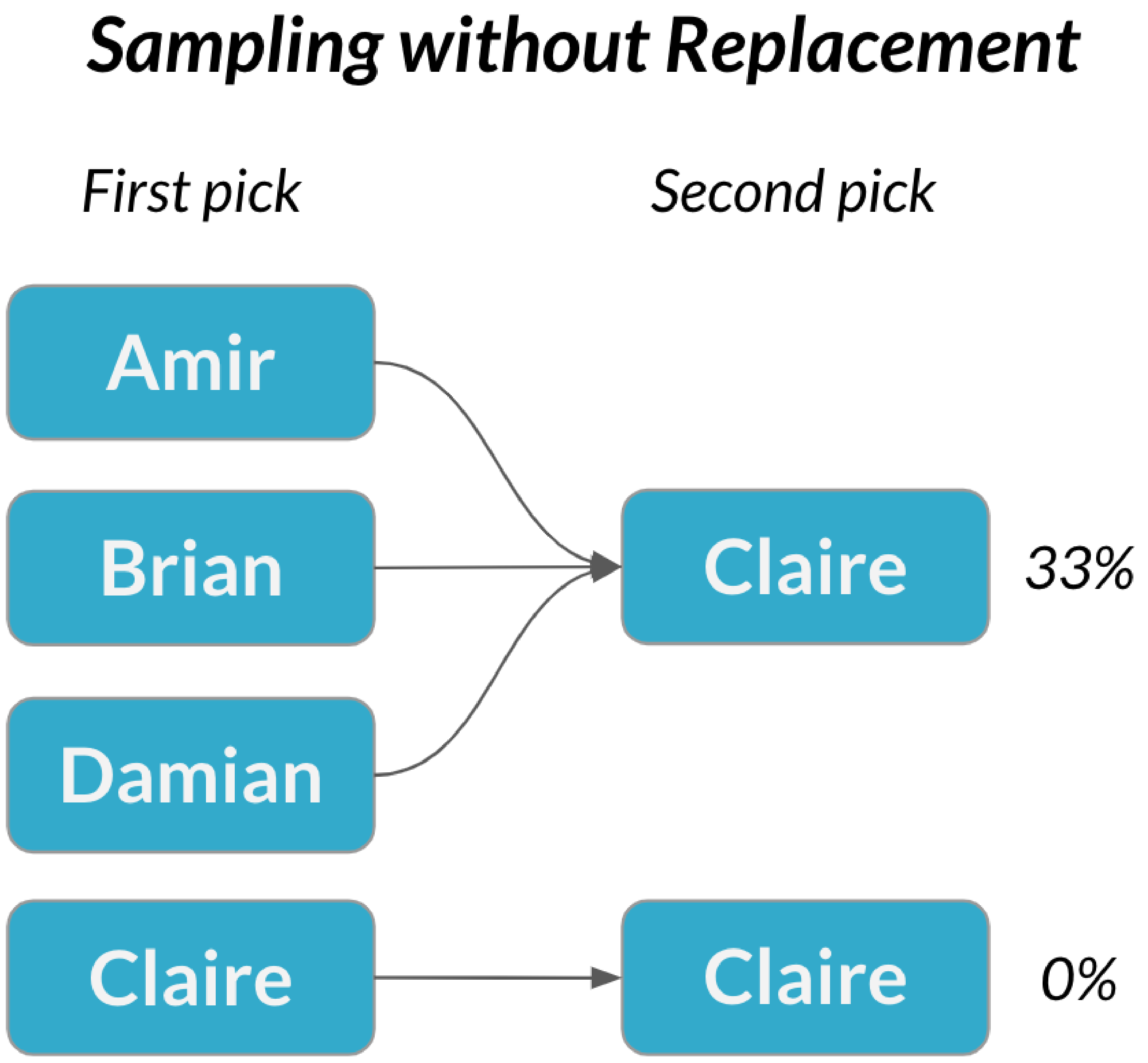
*Two events are **dependent** if the probability of the second event **is** affected by the outcome of the first event.*



Dependent events

*Two events are **dependent** if the probability of the second event **is** affected by the outcome of the first event.*

Sampling without replacement = each pick is dependent



Let's practice!

INTRODUCTION TO STATISTICS IN PYTHON

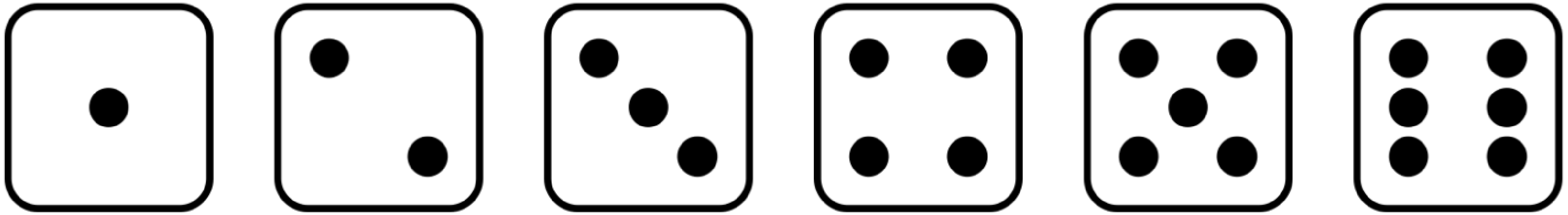
Discrete distributions

INTRODUCTION TO STATISTICS IN PYTHON

Maggie Matsui
Content Developer, DataCamp

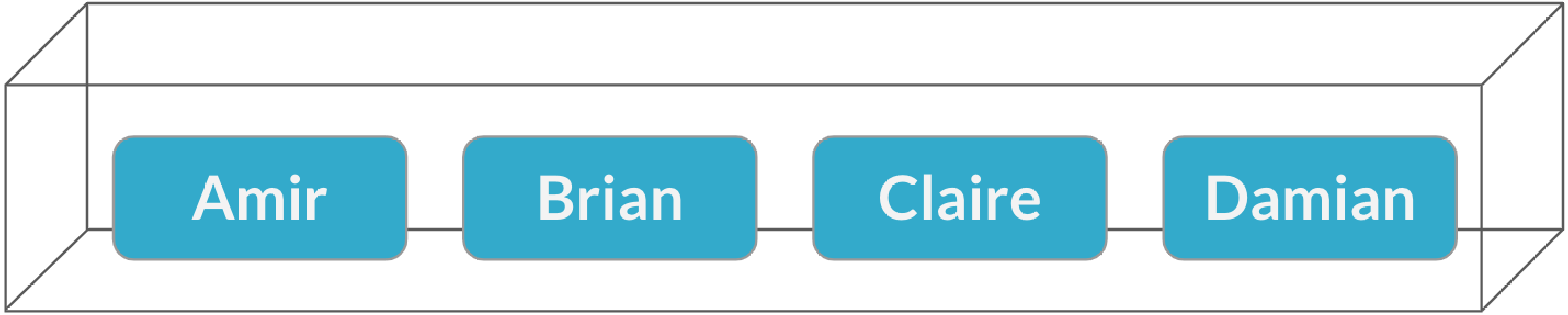
Rolling the dice

Rolling the dice



1/6 1/6 1/6 1/6 1/6 1/6

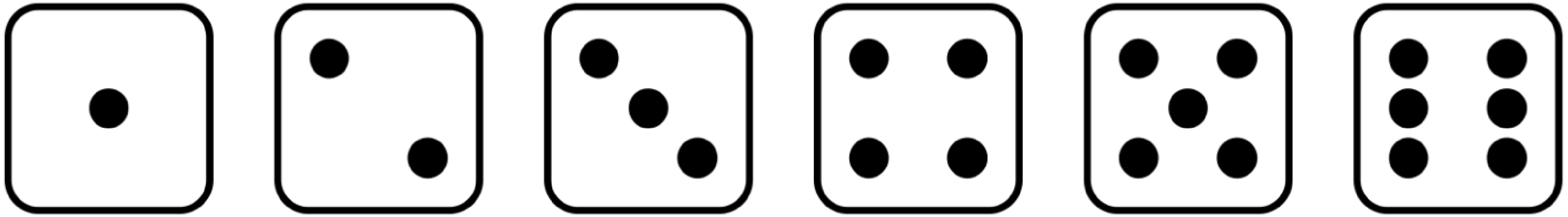
Choosing salespeople



25% 25% 25% 25%

Probability distribution

Describes the probability of each possible outcome in a scenario



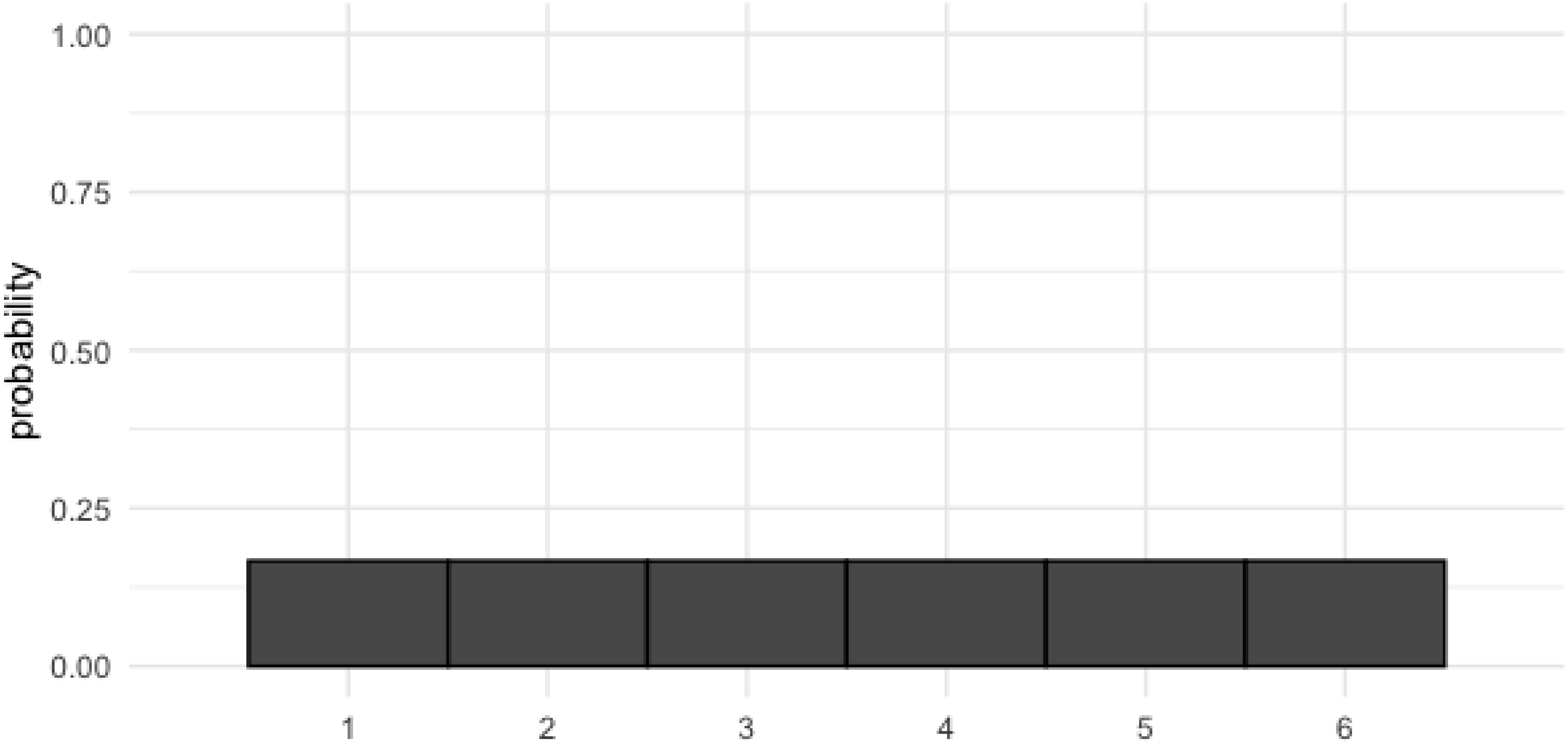
$\frac{1}{6}$ $\frac{1}{6}$ $\frac{1}{6}$ $\frac{1}{6}$ $\frac{1}{6}$ $\frac{1}{6}$

Expected value: mean of a probability distribution

Expected value of a fair die roll =

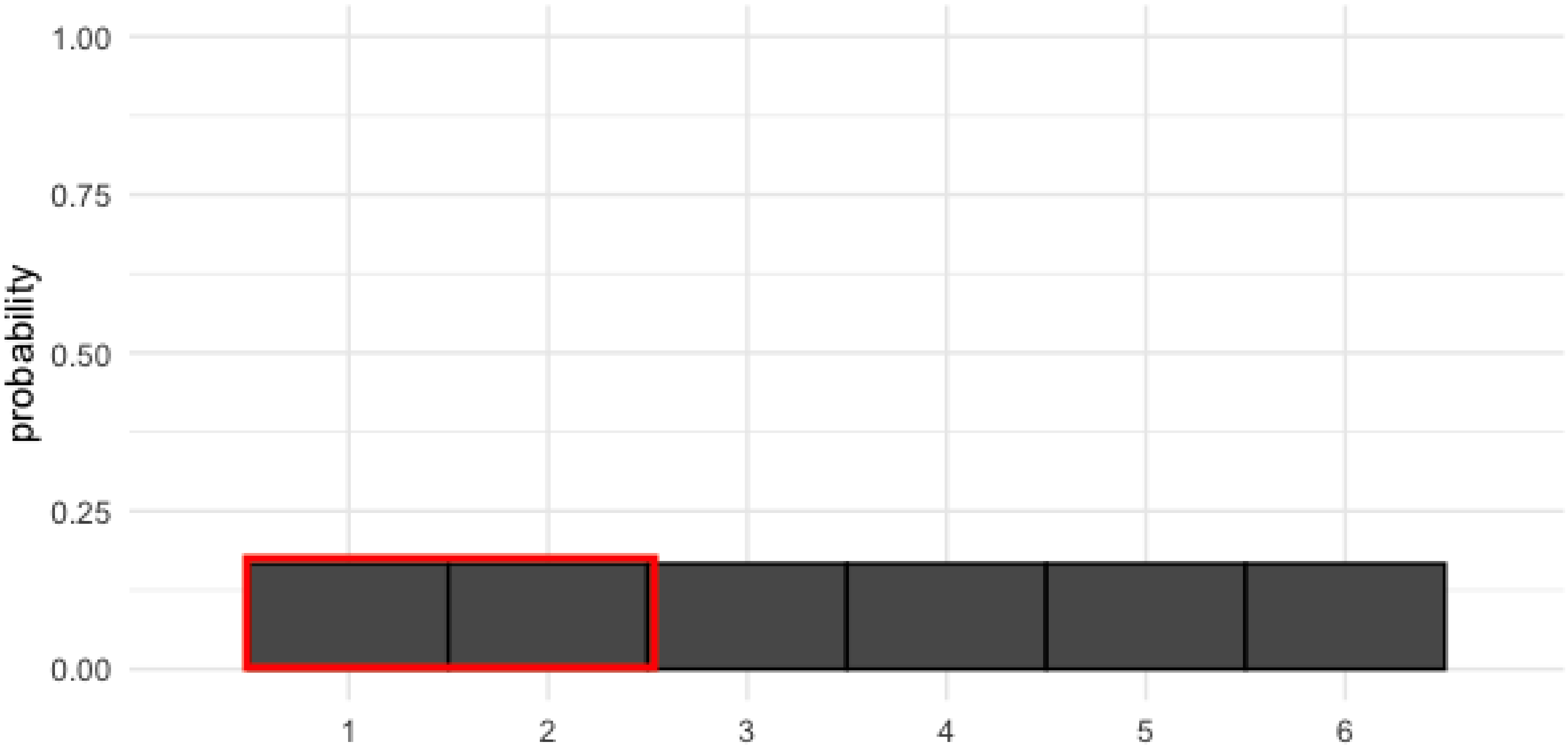
$$(1 \times \tfrac{1}{6}) + (2 \times \tfrac{1}{6}) + (3 \times \tfrac{1}{6}) + (4 \times \tfrac{1}{6}) + (5 \times \tfrac{1}{6}) + (6 \times \tfrac{1}{6}) = 3.5$$

Visualizing a probability distribution



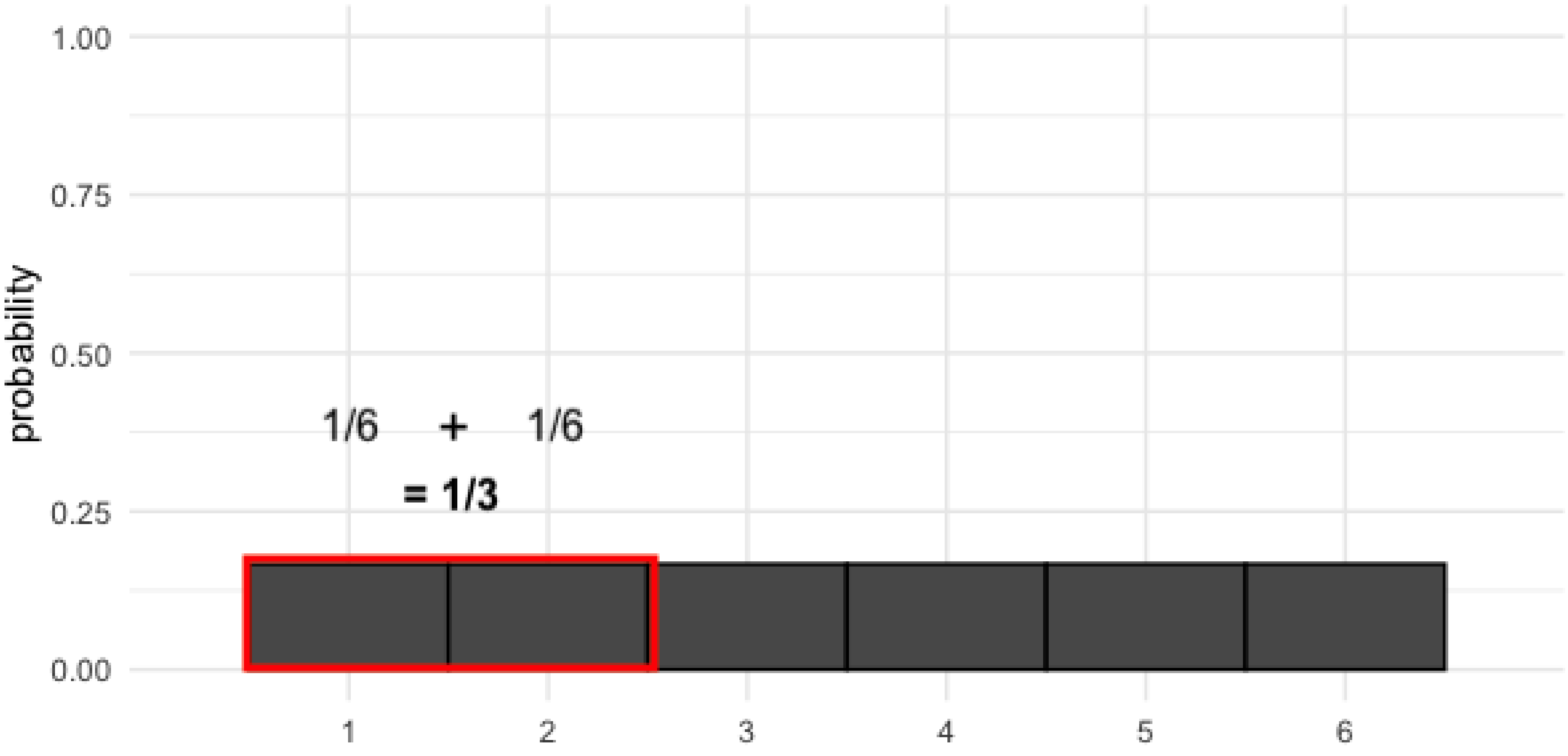
Probability = area

$$P(\text{die roll}) \leq 2 = ?$$

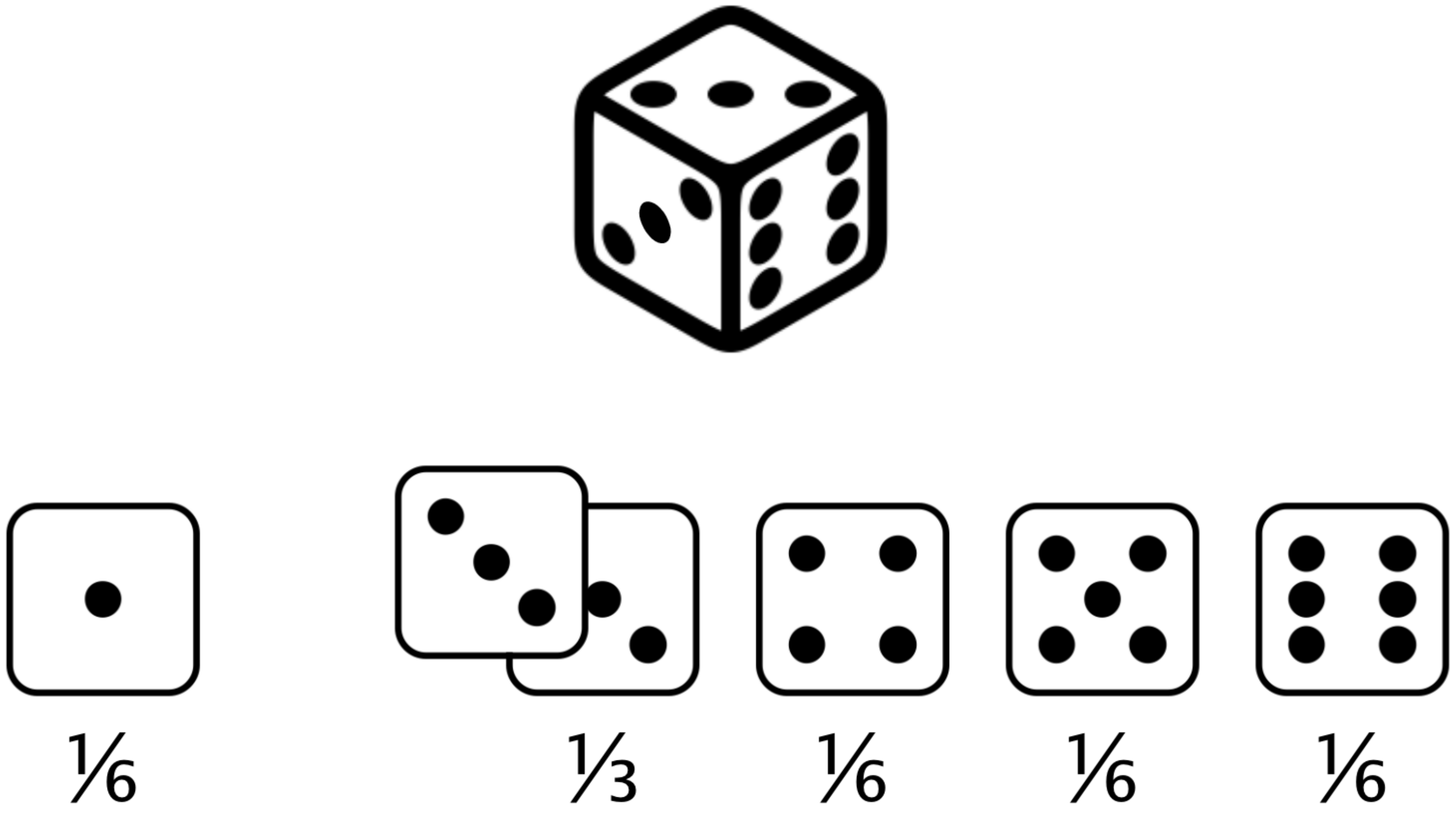


Probability = area

$$P(\text{die roll}) \leq 2 = 1/3$$



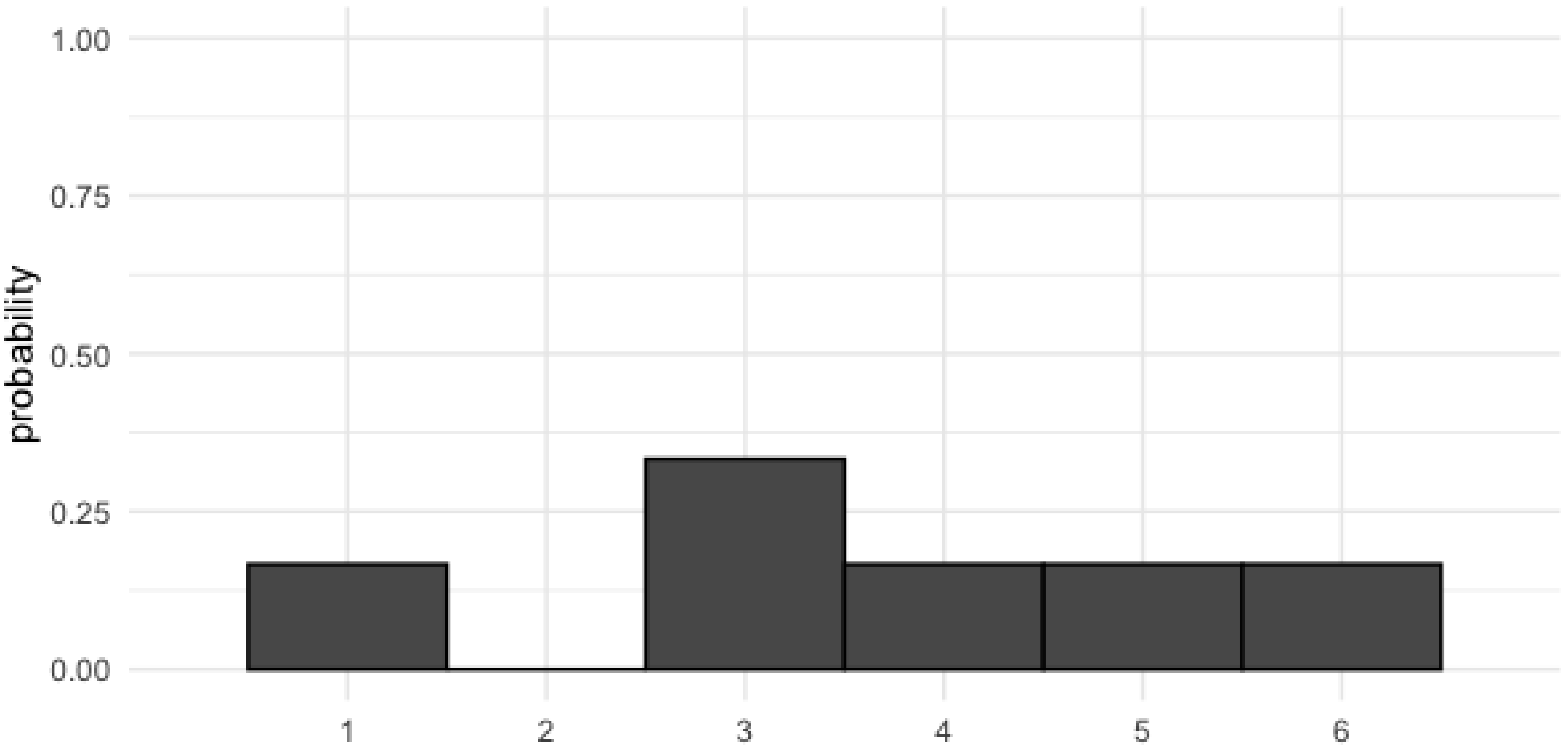
Uneven die



Expected value of uneven die roll =

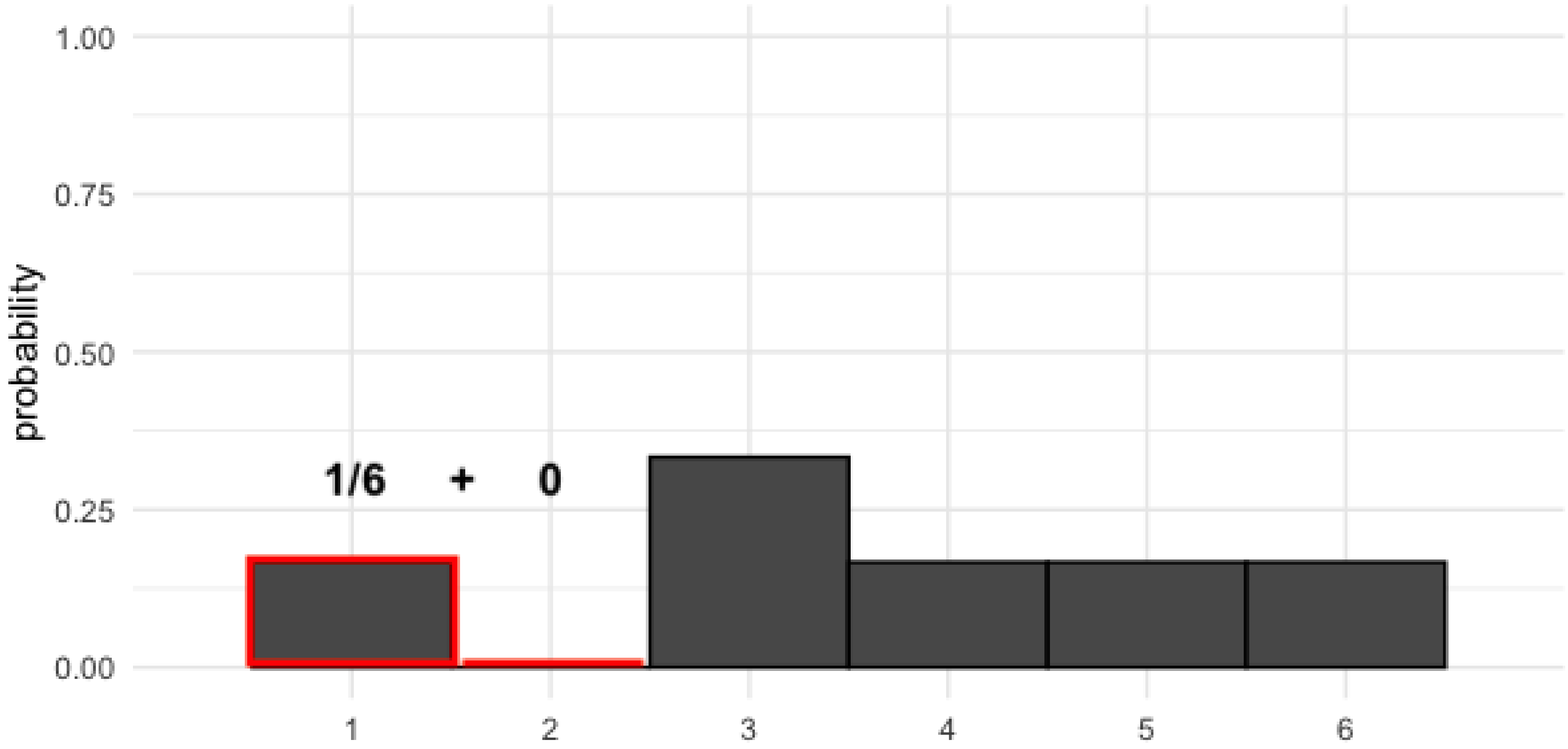
$$(1 \times \frac{1}{6}) + (2 \times 0) + (3 \times \frac{1}{3}) + (4 \times \frac{1}{6}) + (5 \times \frac{1}{6}) + (6 \times \frac{1}{6}) = 3.67$$

Visualizing uneven probabilities



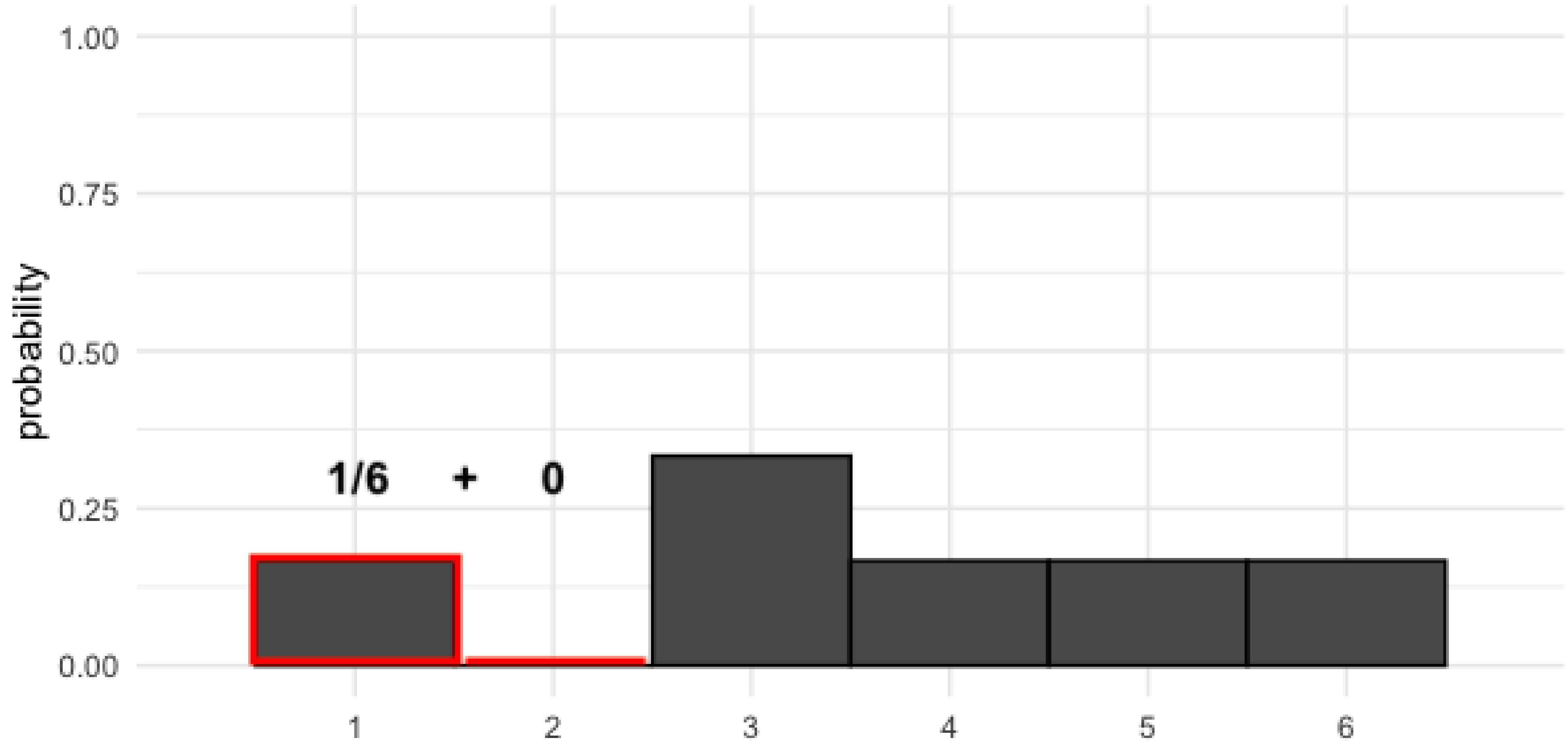
Adding areas

$$P(\text{uneven die roll}) \le 2 = ?$$



Adding areas

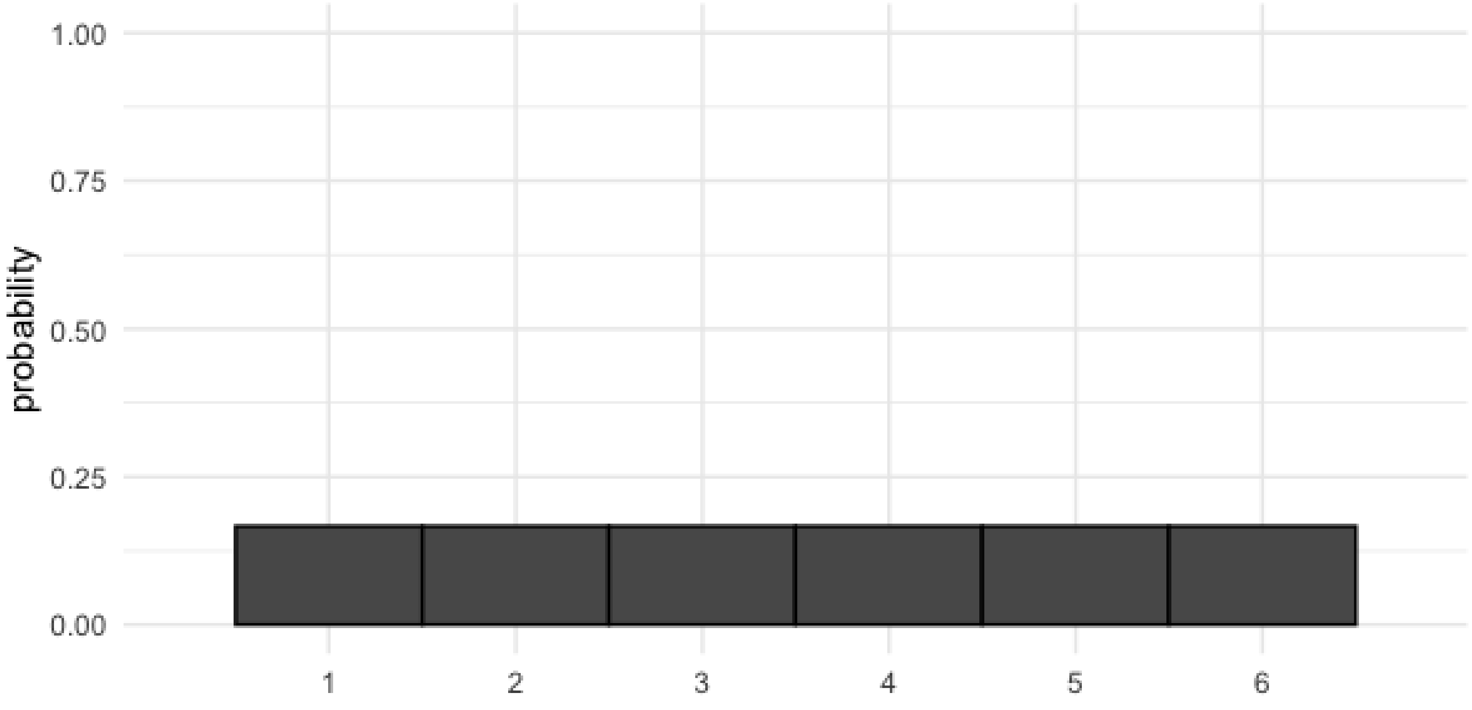
$$P(\text{uneven die roll}) \leq 2 = 1/6$$



Discrete probability distributions

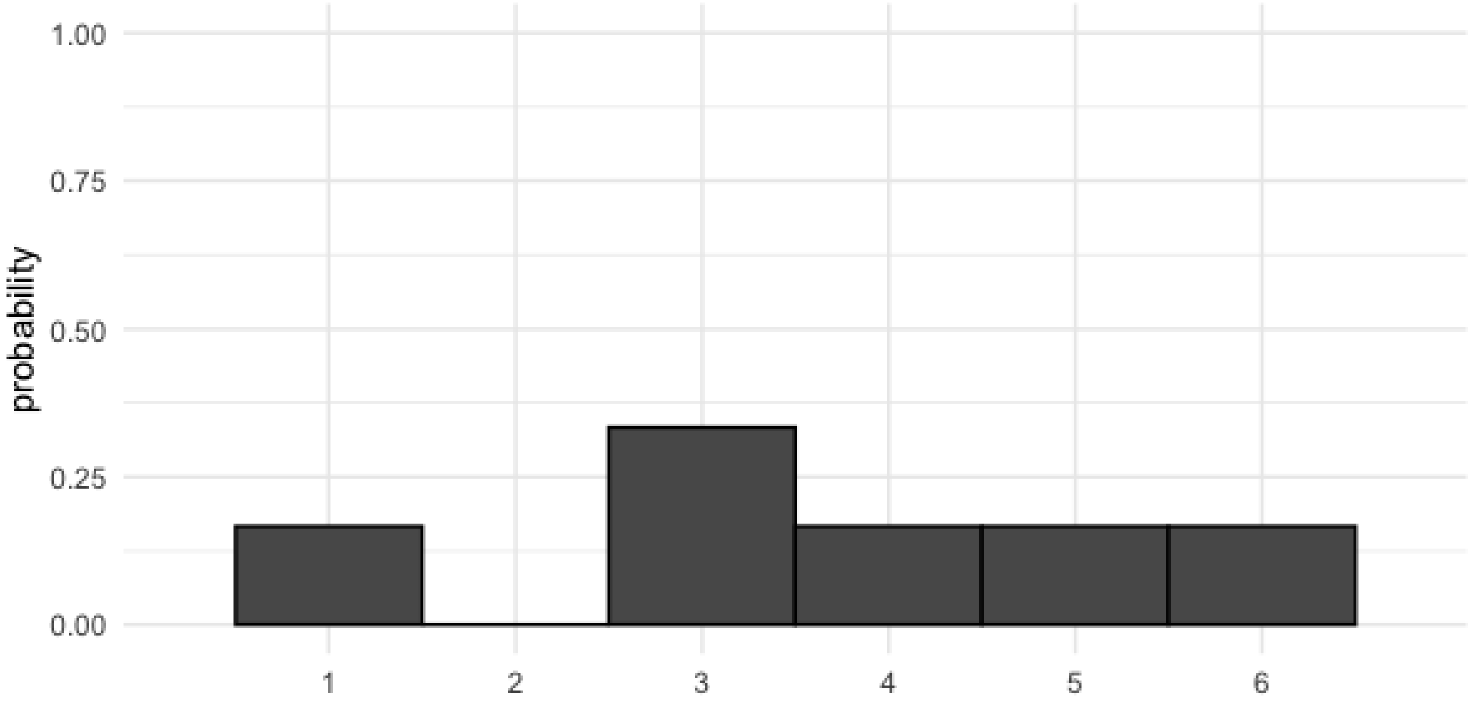
Describe probabilities for discrete outcomes

Fair die



Discrete uniform distribution

Uneven die



Sampling from discrete distributions

```
print(die)
```

```
   number      prob
0       1  0.166667
1       2  0.166667
2       3  0.166667
3       4  0.166667
4       5  0.166667
5       6  0.166667
```

```
np.mean(die['number'])
```

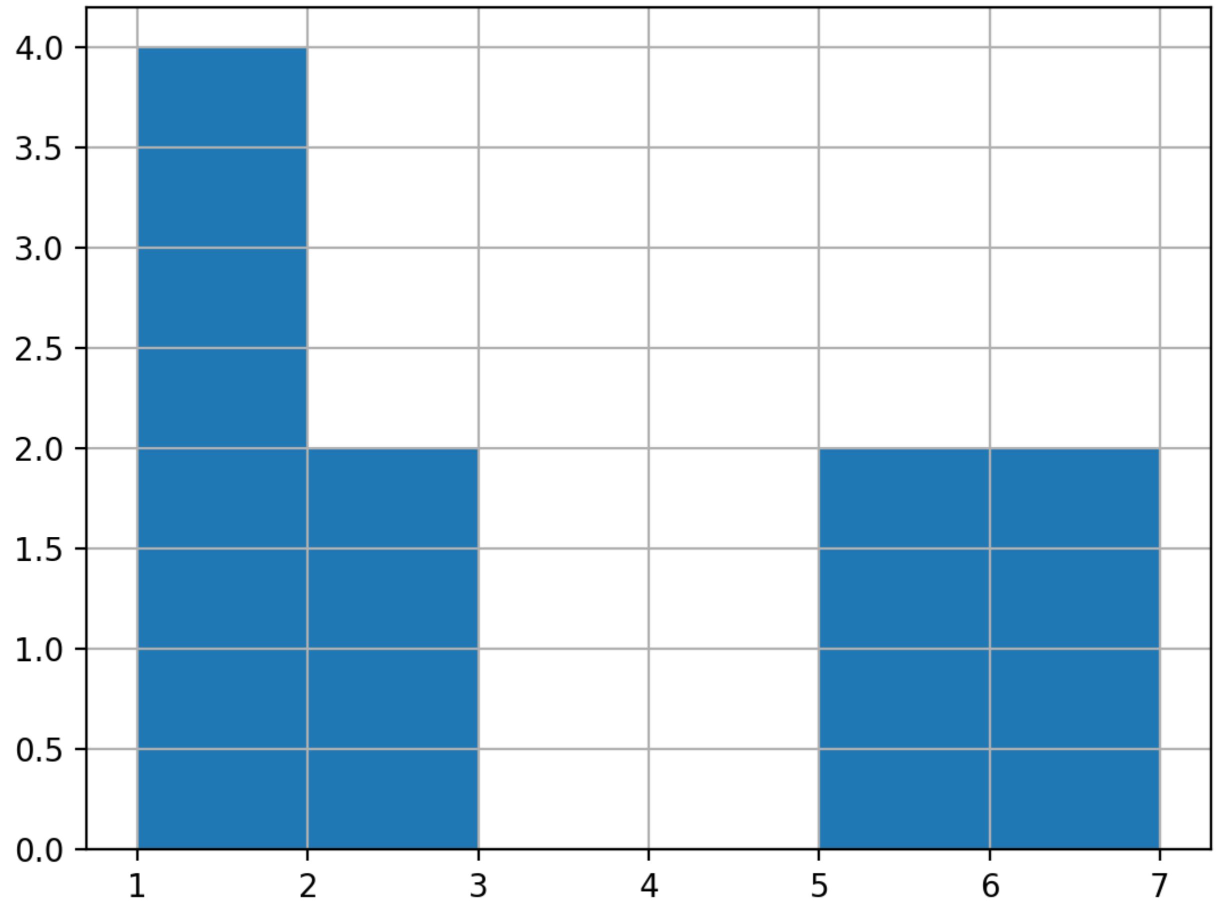
```
3.5
```

```
rolls_10 = die.sample(10, replace = True)
rolls_10
```

```
   number      prob
0       1  0.166667
0       1  0.166667
4       5  0.166667
1       2  0.166667
0       1  0.166667
0       1  0.166667
5       6  0.166667
5       6  0.166667
...
```

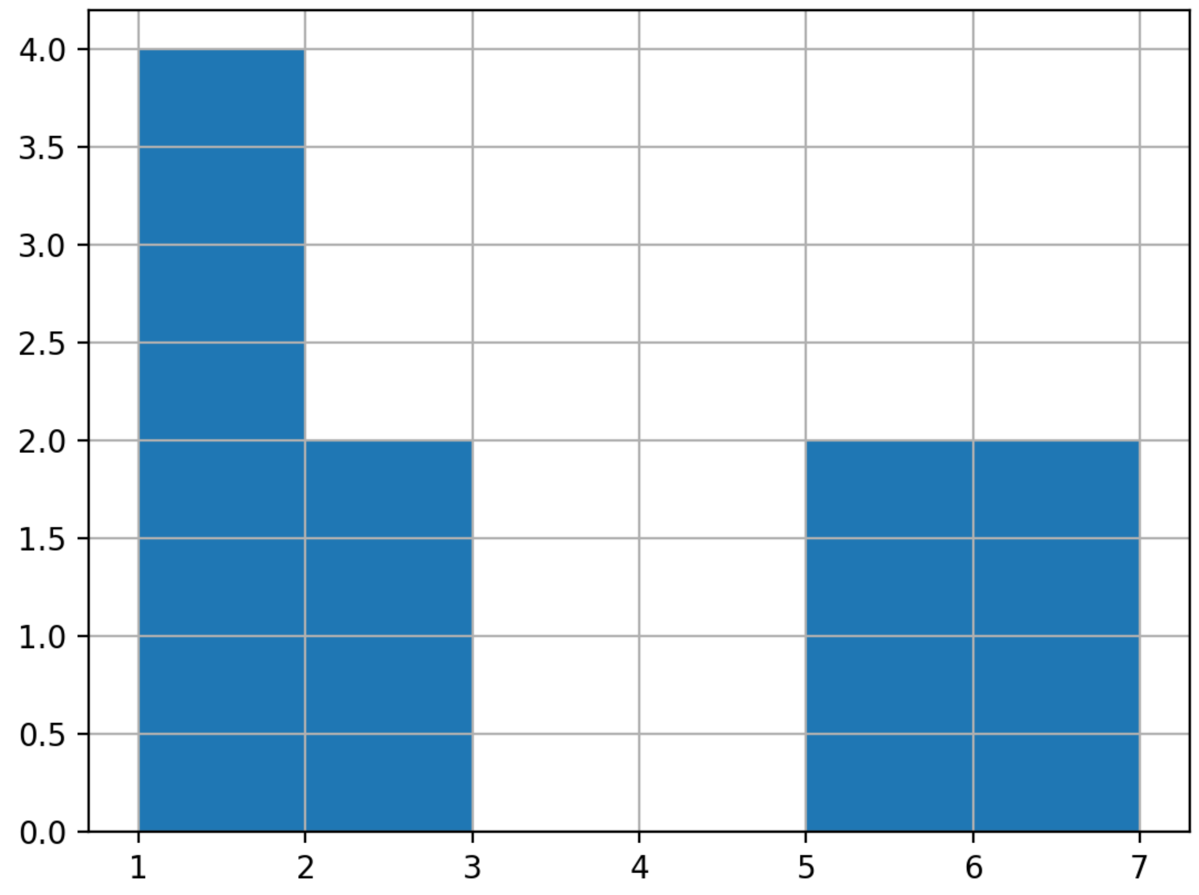
Visualizing a sample

```
rolls_10['number'].hist(bins=np.linspace(1,7,7))
plt.show()
```



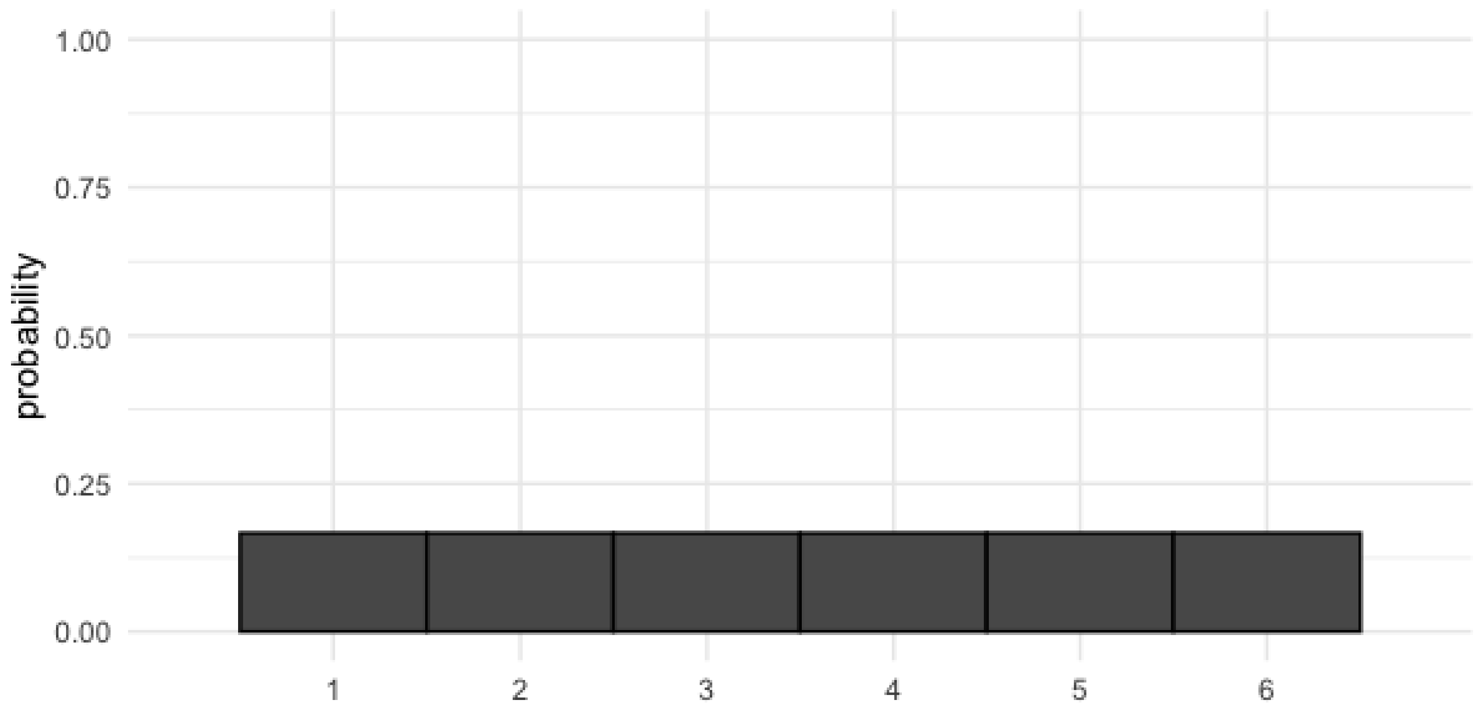
Sample distribution vs. theoretical distribution

Sample of 10 rolls



`np.mean(rolls_10['number'])` = `3.0`

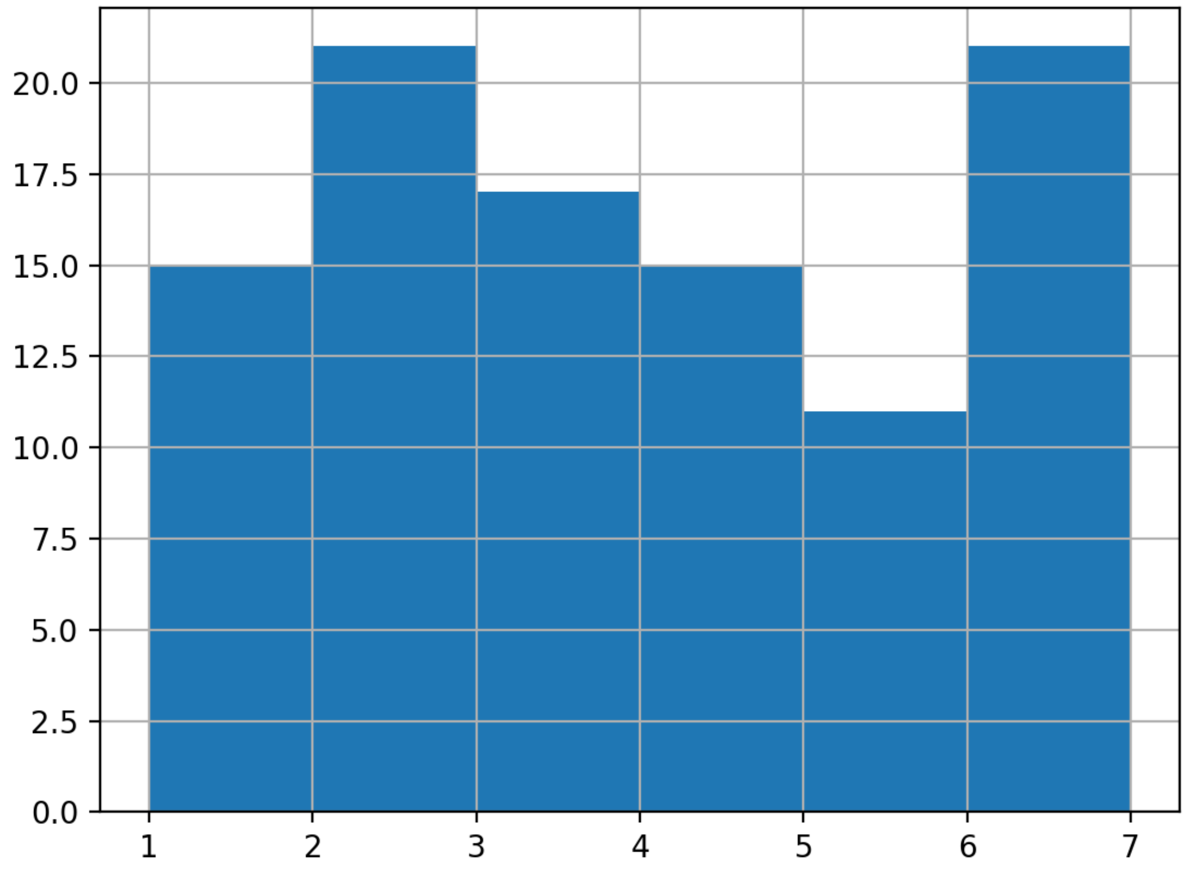
Theoretical probability distribution



`mean(die['number'])` = `3.5`

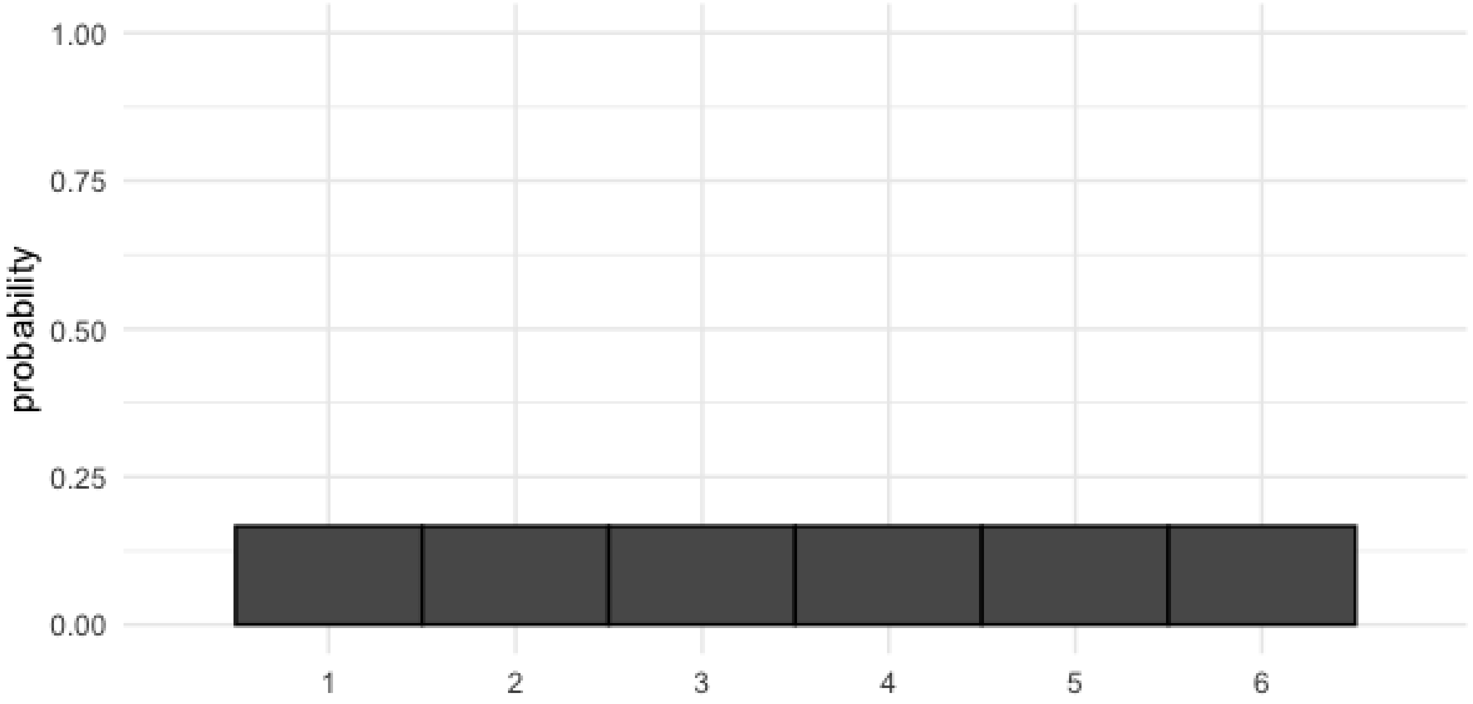
A bigger sample

Sample of 100 rolls



`np.mean(rolls_100['number'])` = `3.4`

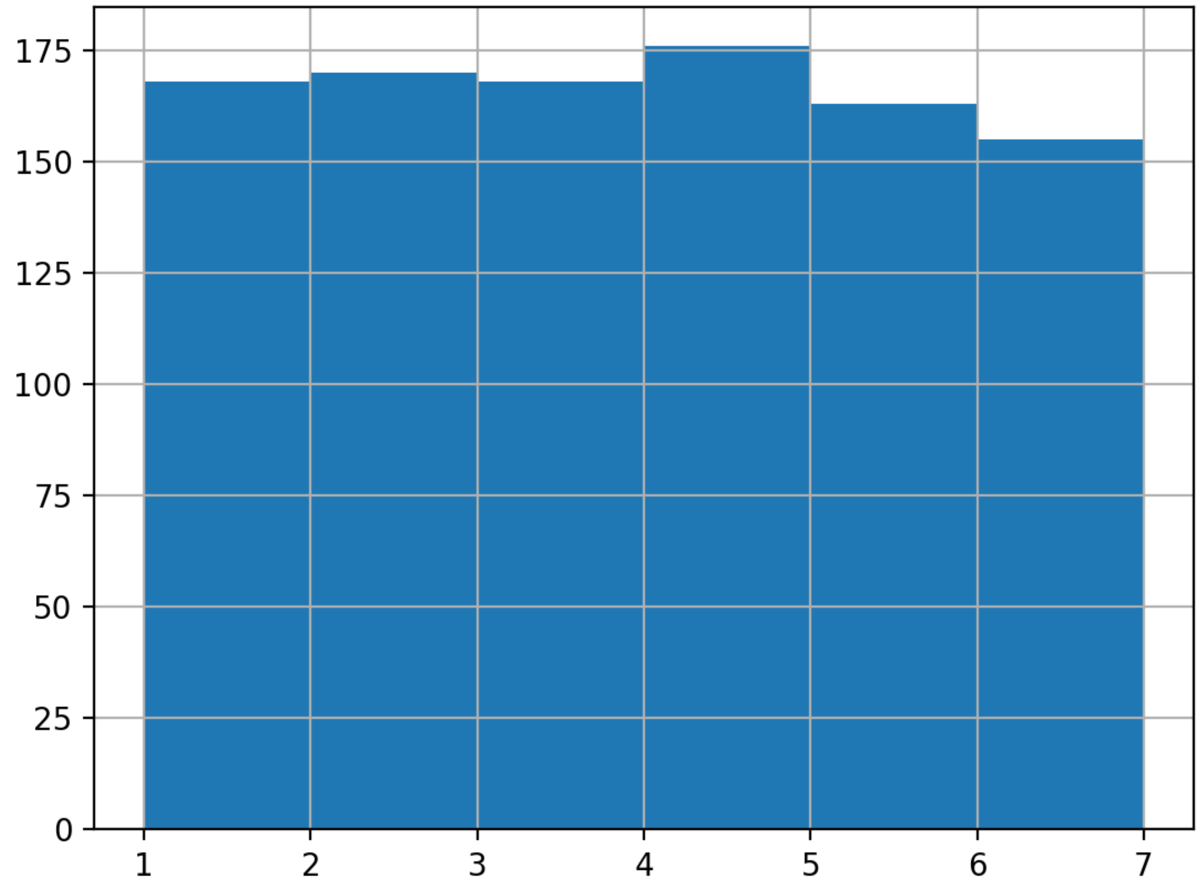
Theoretical probability distribution



`mean(die['number'])` = `3.5`

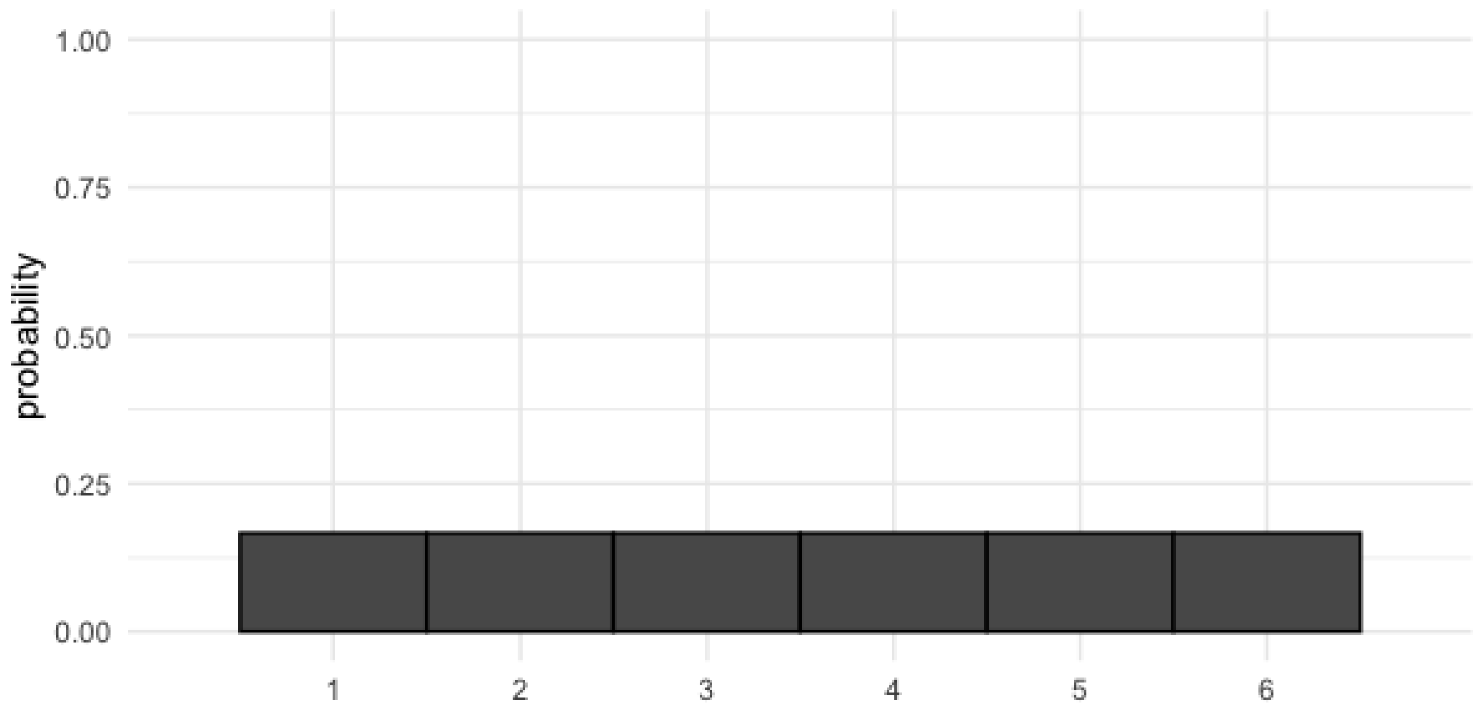
An even bigger sample

Sample of 1000 rolls



`np.mean(rolls_1000['number'])` = `3.48`

Theoretical probability distribution



`mean(die['number'])` = `3.5`

Law of large numbers

As the size of your sample increases, the sample mean will approach the expected value.

Sample size	Mean
10	3.00
100	3.40
1000	3.48

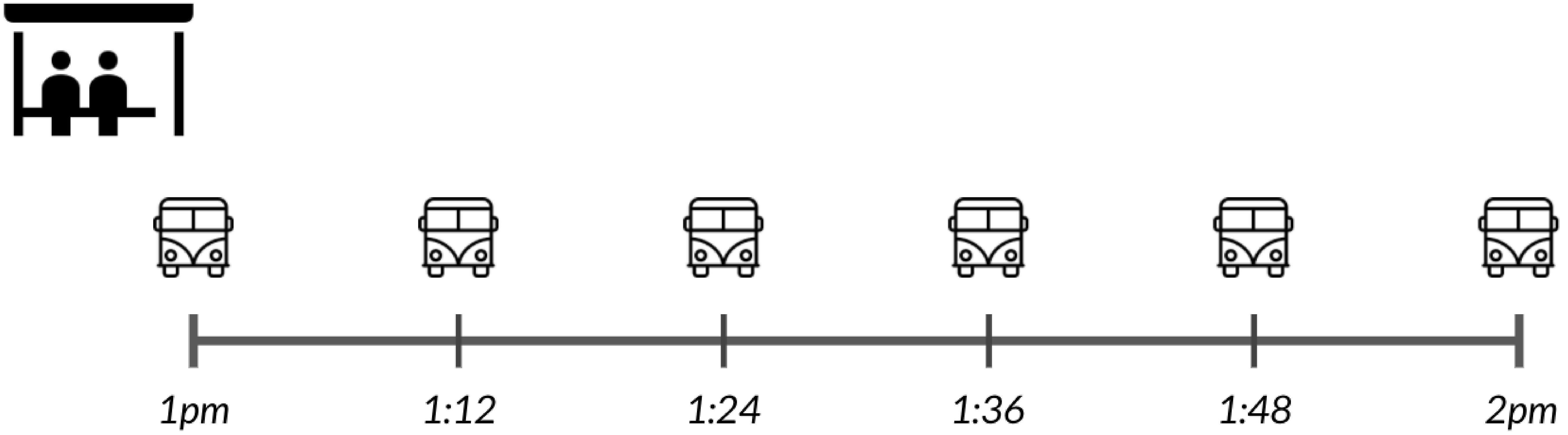
Let's practice!

Continuous distributions

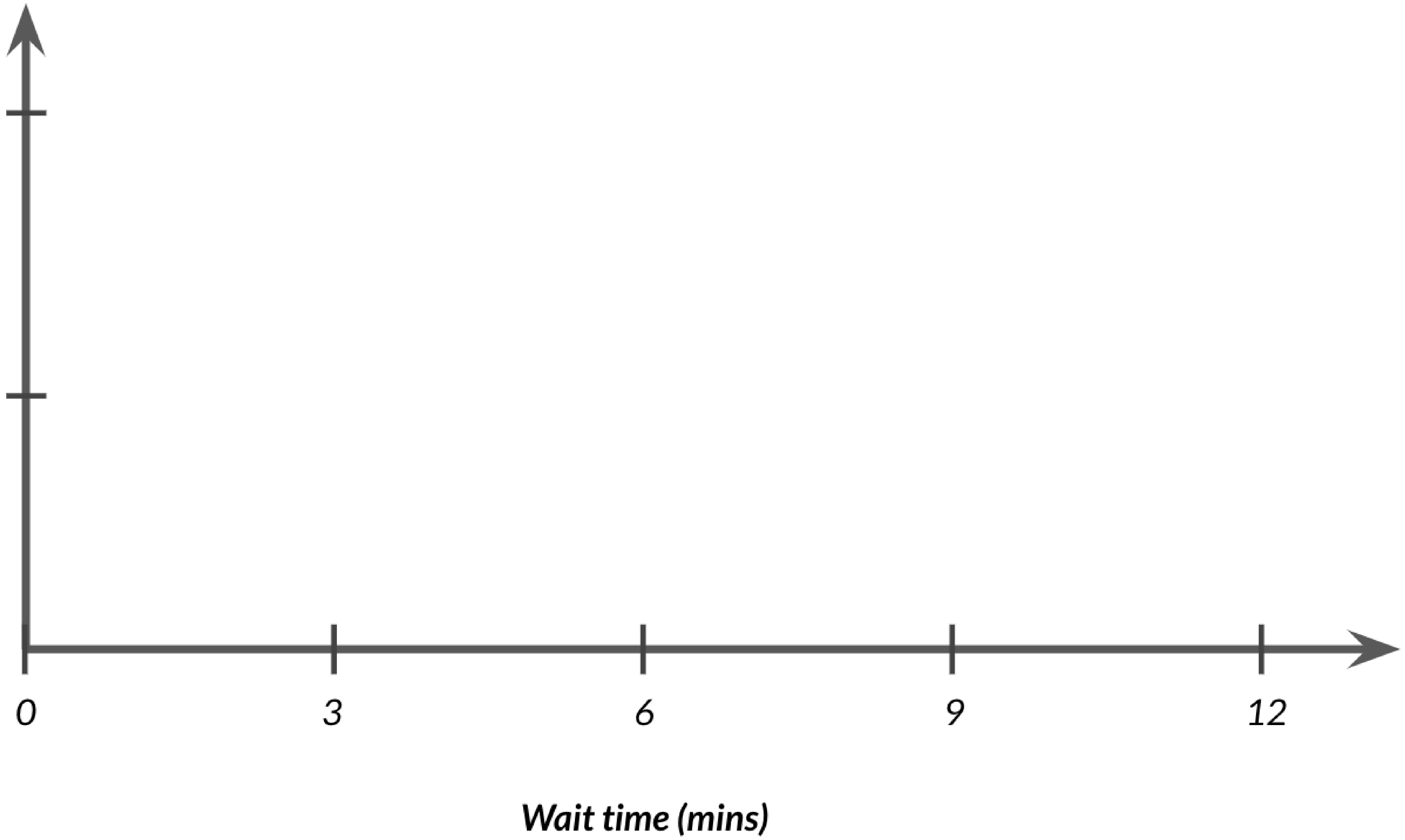
INTRODUCTION TO STATISTICS IN PYTHON

Maggie Matsui
Content Developer, DataCamp

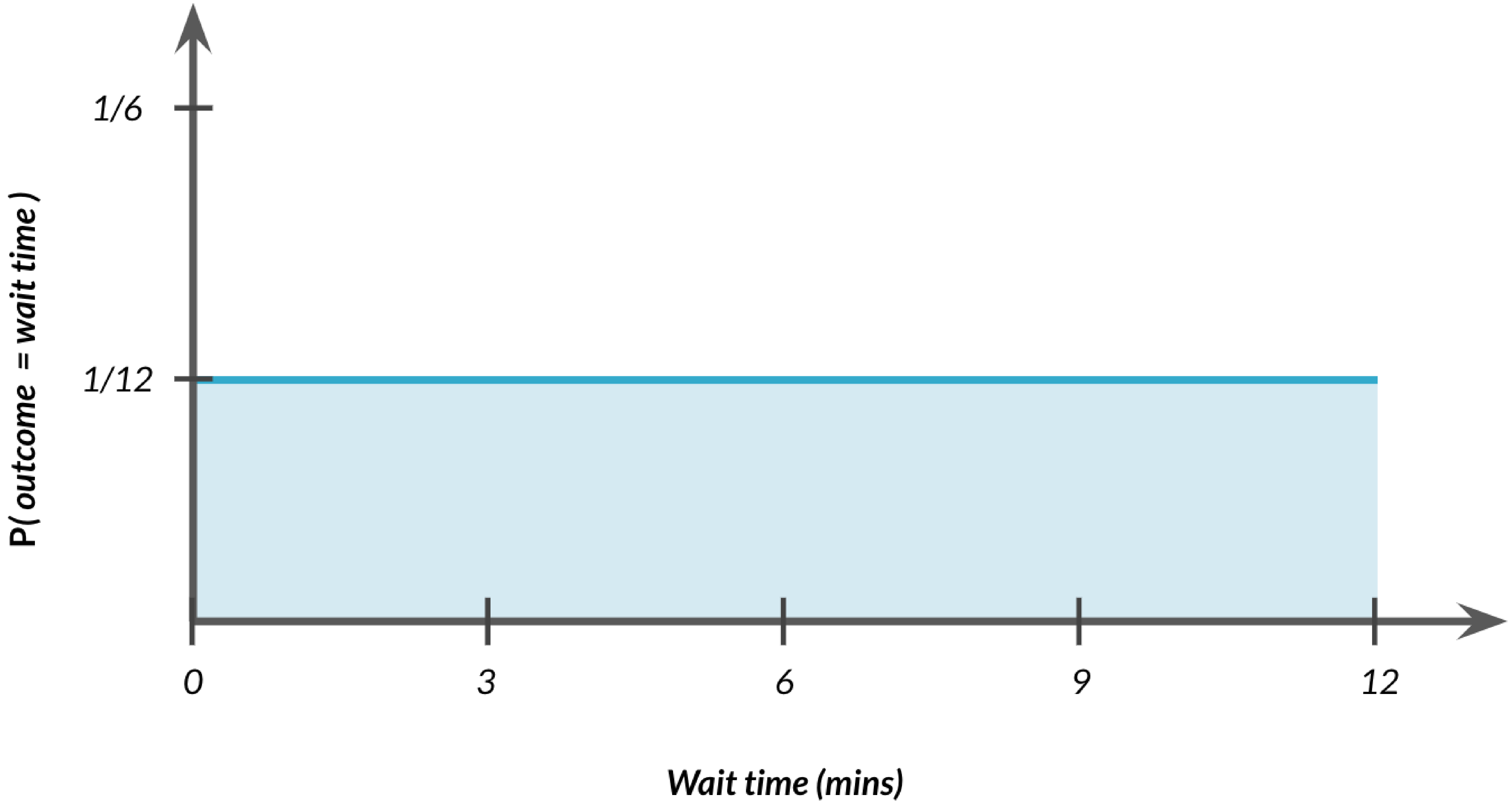
Waiting for the bus



Continuous uniform distribution

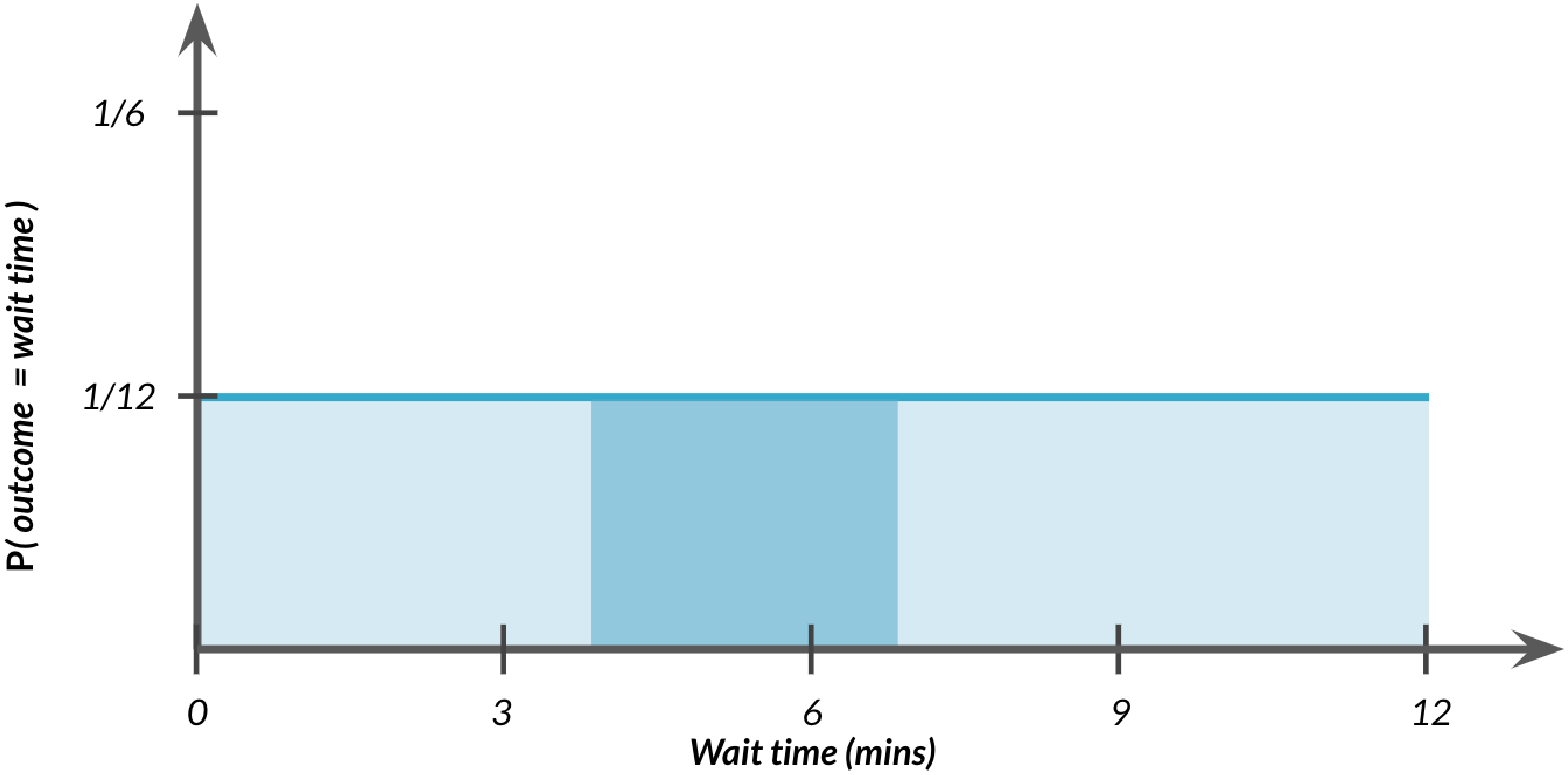


Continuous uniform distribution



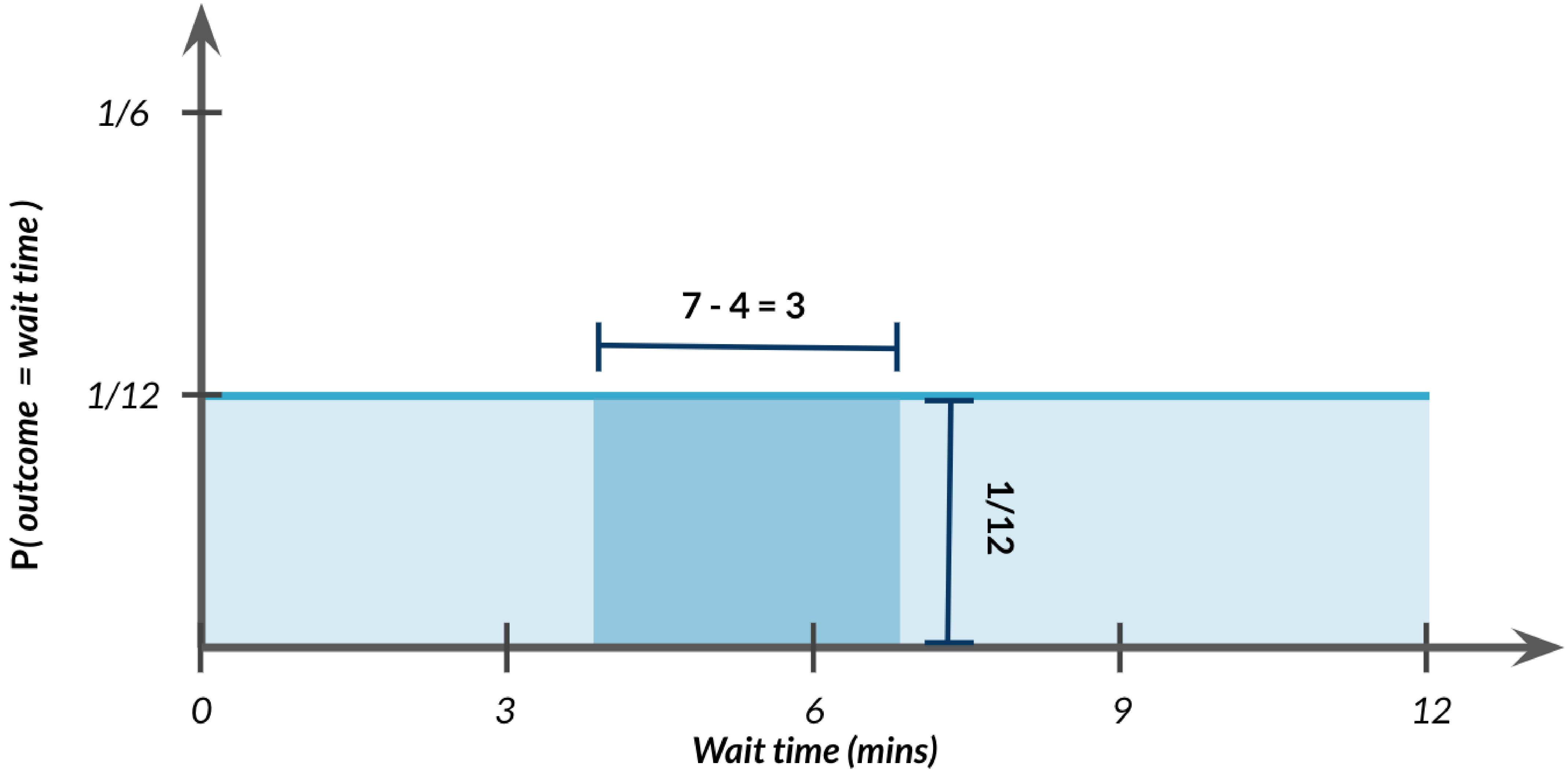
Probability still = area

$$P(4 \leq \text{wait time} \leq 7) = \ ?$$



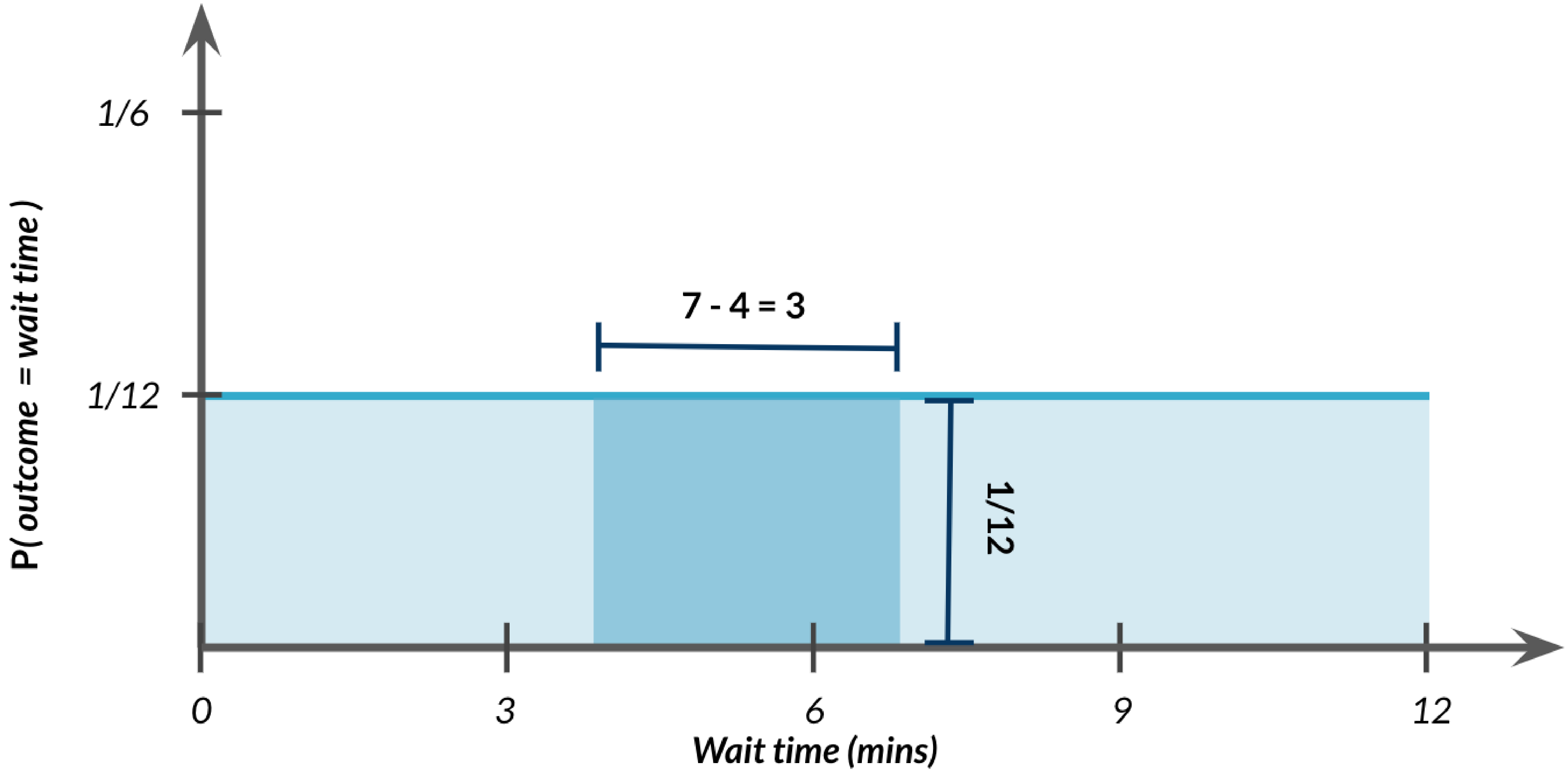
Probability still = area

$$P(4 \leq \text{wait time} \leq 7) = \ ?$$



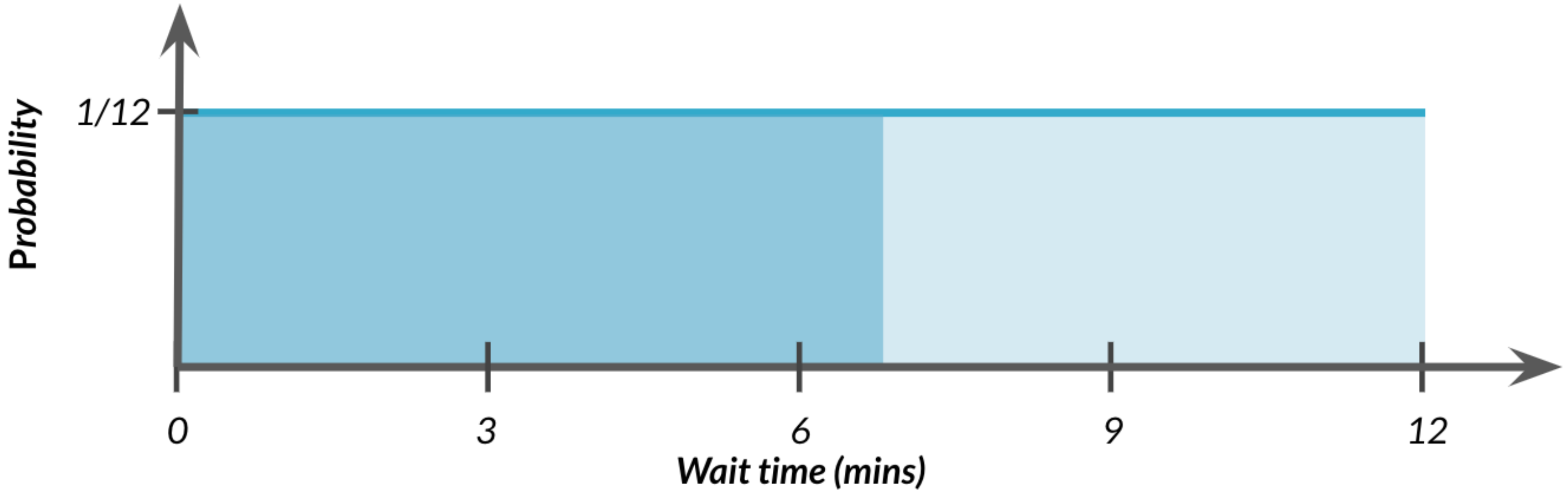
Probability still = area

$$P(4 \leq \text{wait time} \leq 7) = 3 \times 1/12 = 3/12$$



Uniform distribution in Python

$$P(\text{wait time} \leq 7)$$

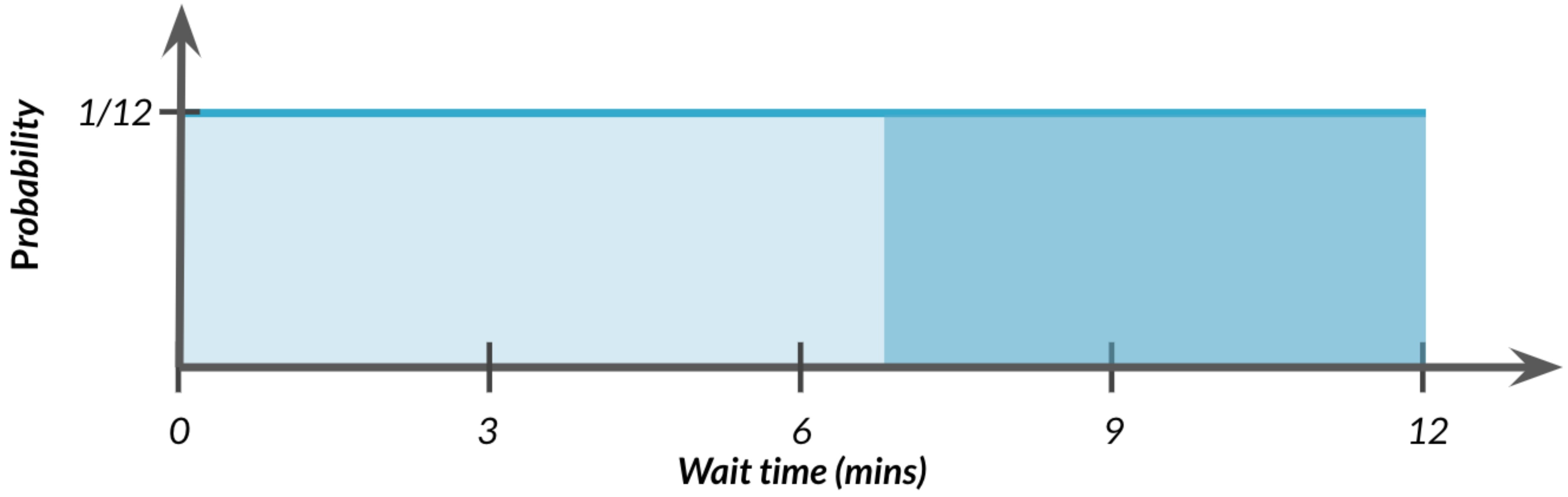



```python
from scipy.stats import uniform
uniform.cdf(7, 0, 12)
```

```
0.5833333
```

"Greater than" probabilities

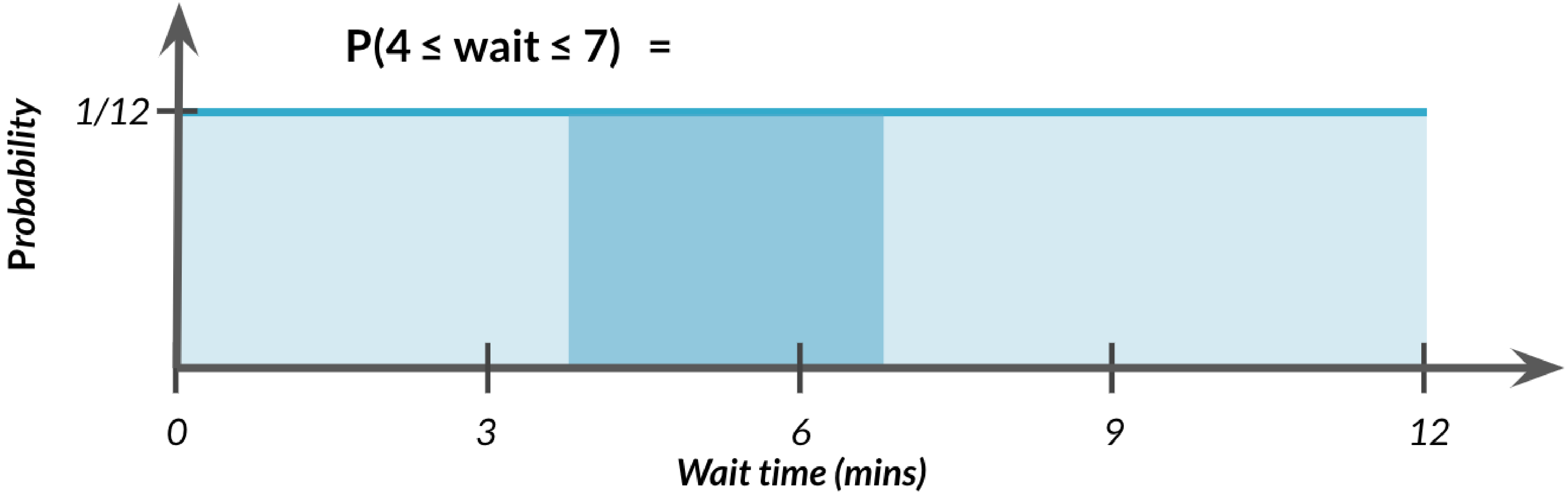
$$P(\text{wait time} \geq 7) = 1 - P(\text{wait time} \leq 7)$$



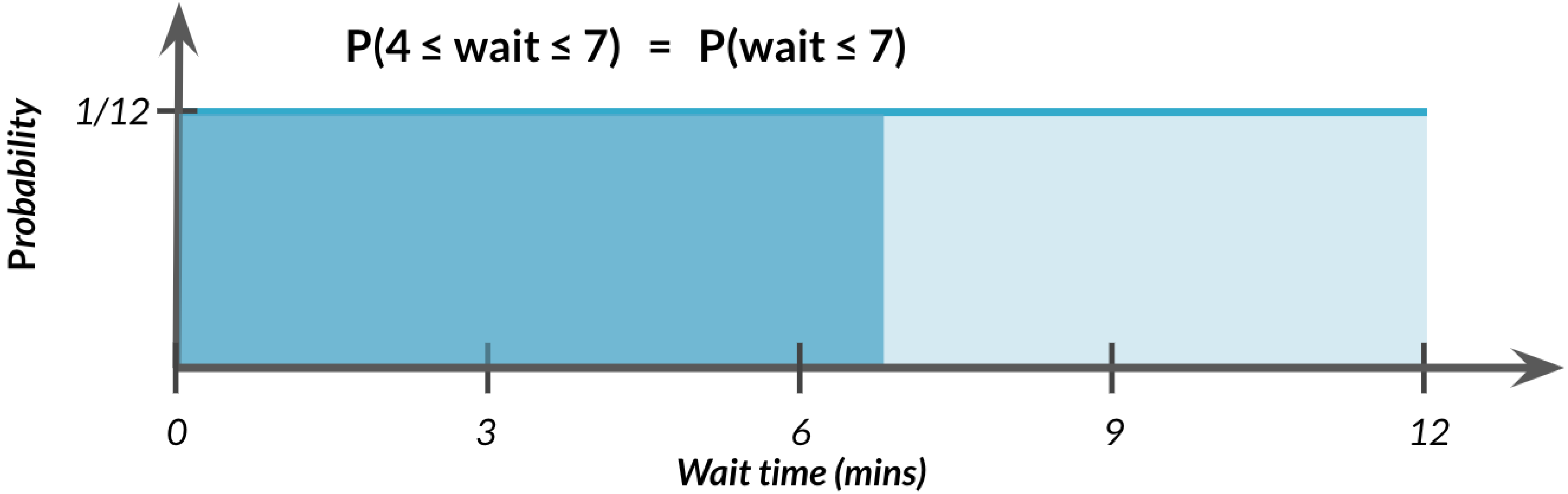

```
from scipy.stats import uniform
1 - uniform.cdf(7, 0, 12)
```

```
0.4166667
```

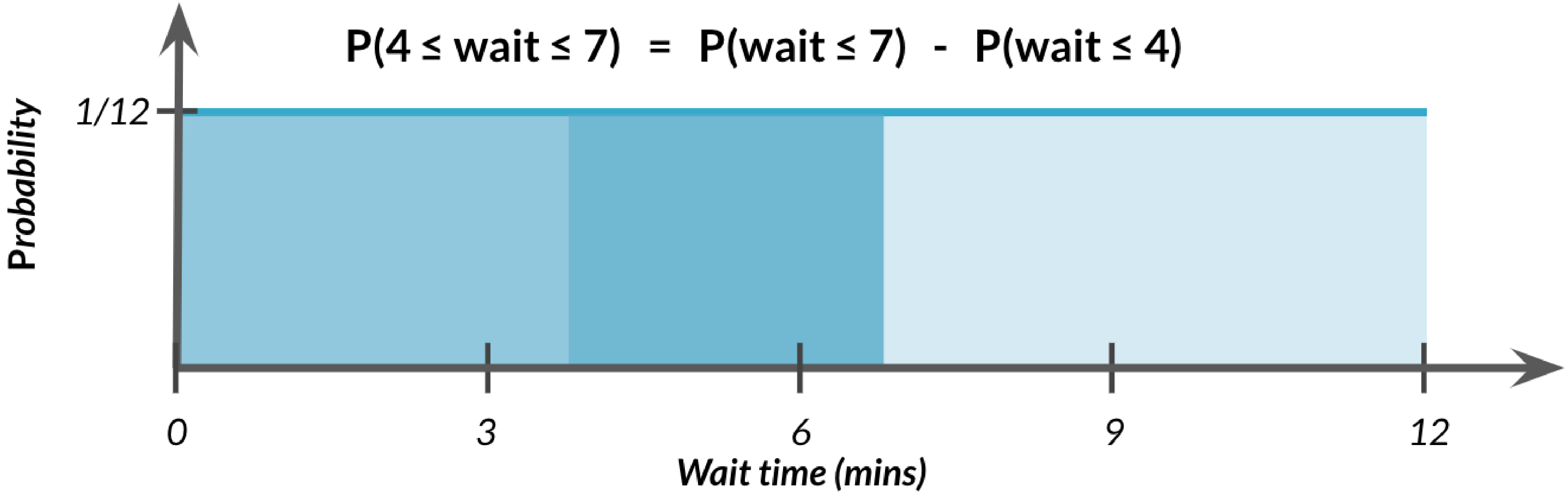
$$P(4 \leq \text{wait time} \leq 7)$$



$$P(4 \le \text{wait time} \le 7)$$



$$P(4 \leq \text{wait time} \leq 7)$$

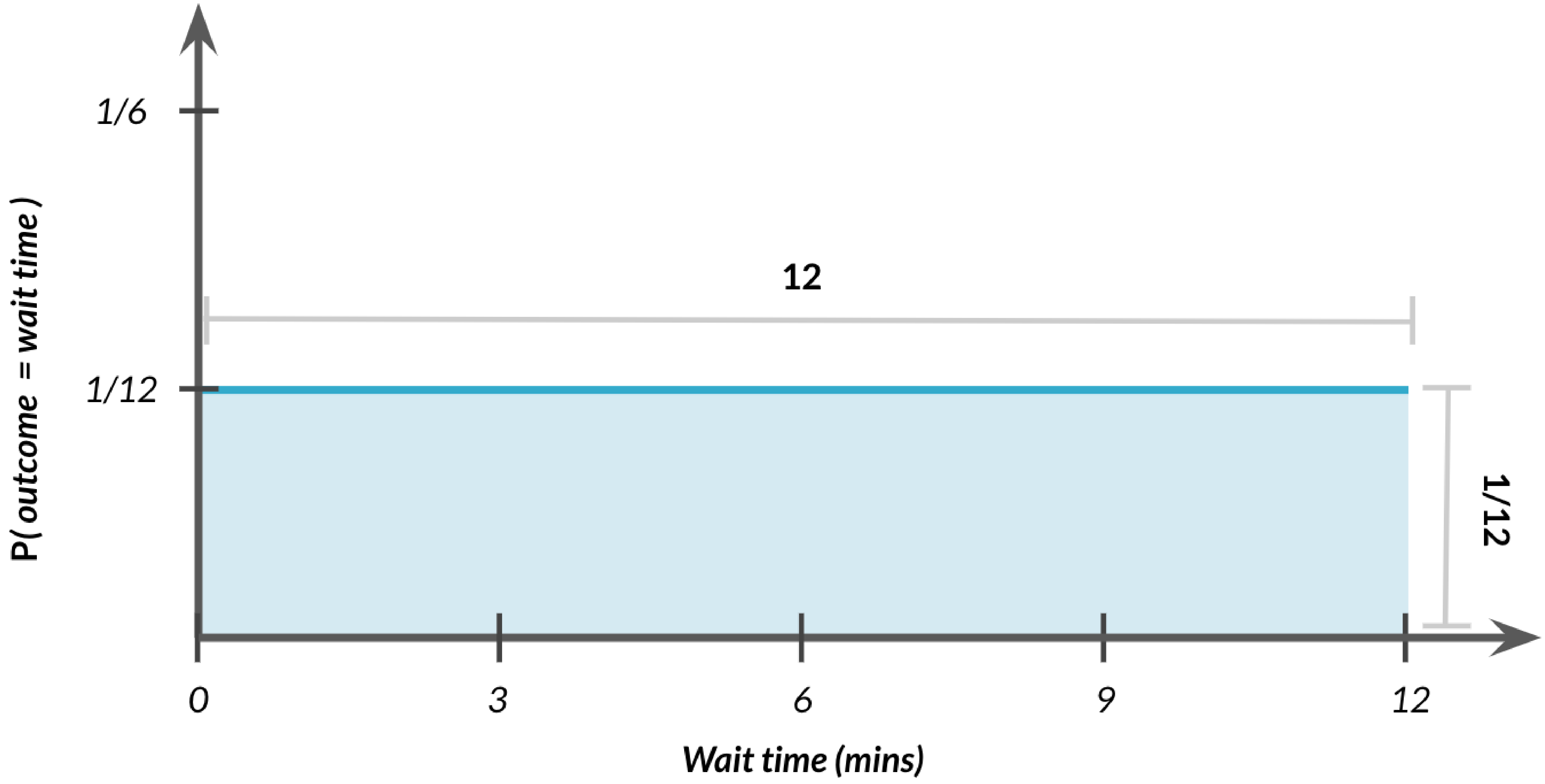



```python
from scipy.stats import uniform
uniform.cdf(7, 0, 12) - uniform.cdf(4, 0, 12)
```

```
0.25
```

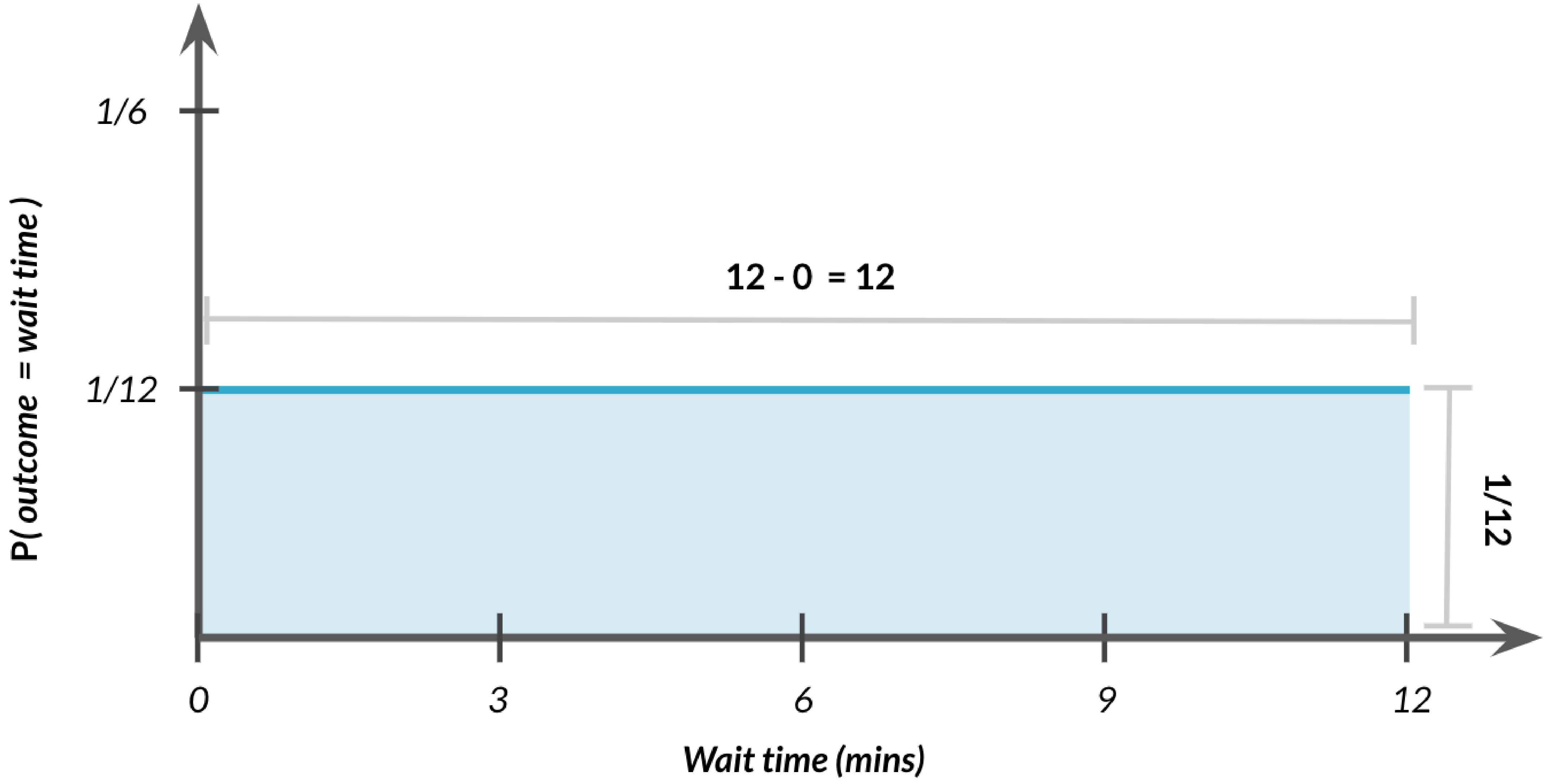
Total area = 1

$$P(0 \leq \text{wait time} \leq 12) = \ ?$$



Total area = 1

$$P(0 \leq \text{outcome} \leq 12) = 12 \times 1/12 = 1$$

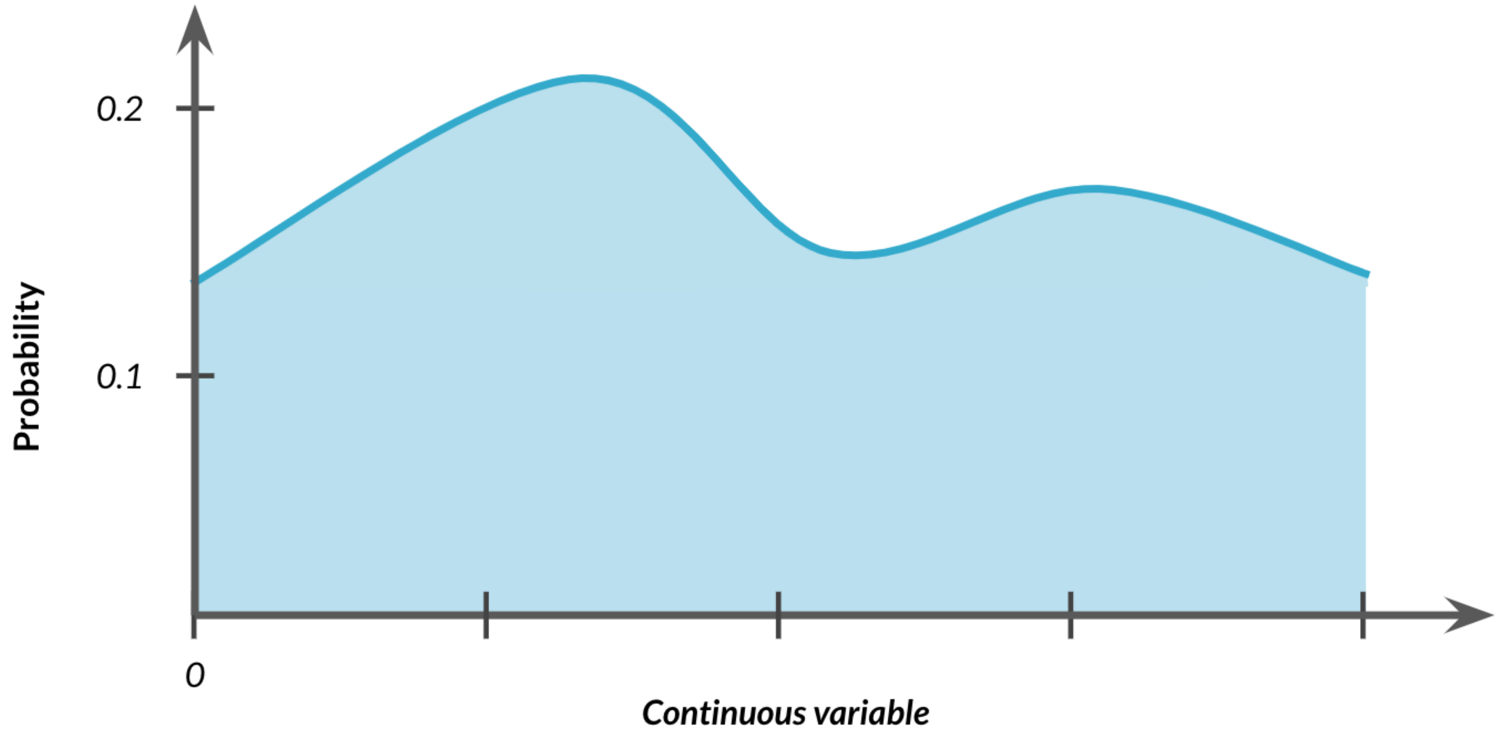


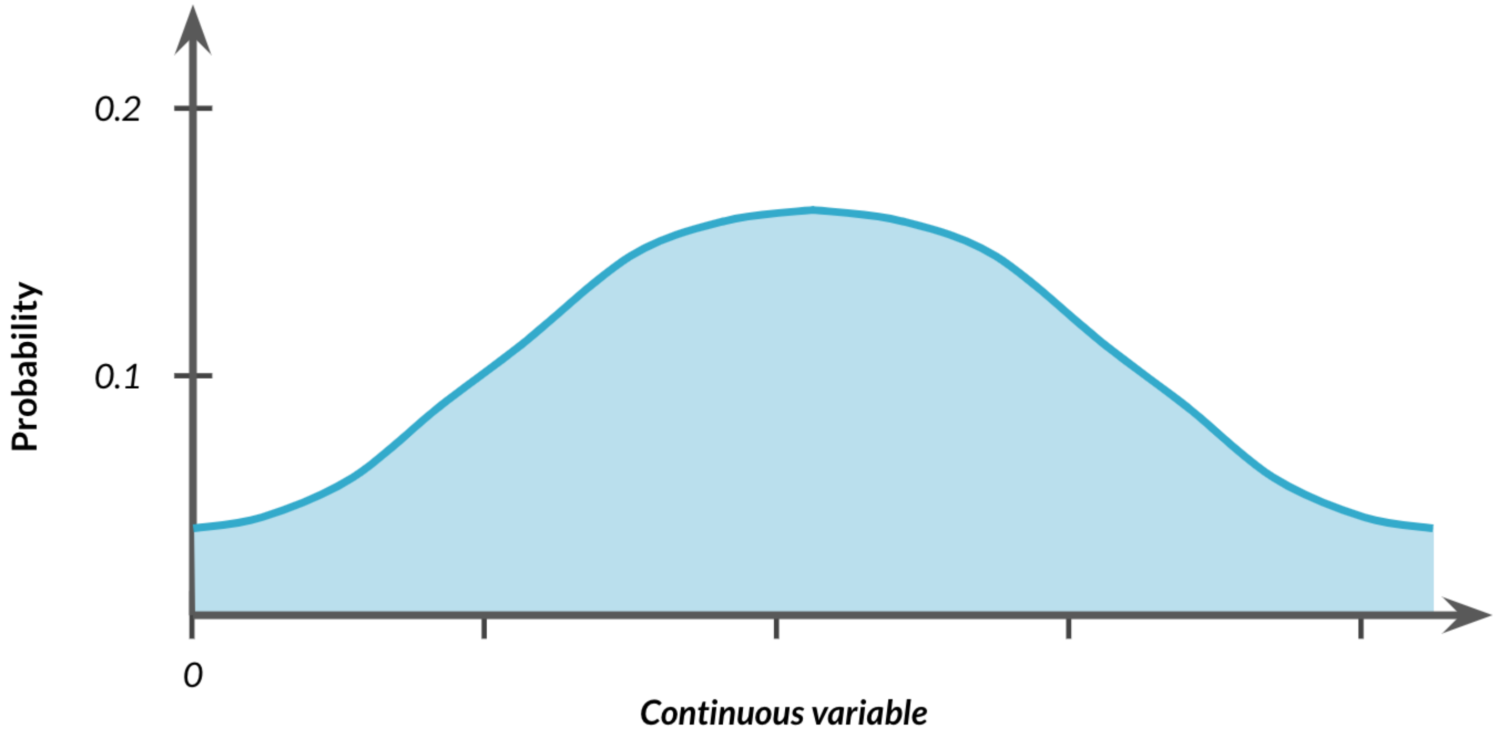
Generating random numbers according to uniform distribution

```python
from scipy.stats import uniform
uniform.rvs(0, 5, size=10)
```

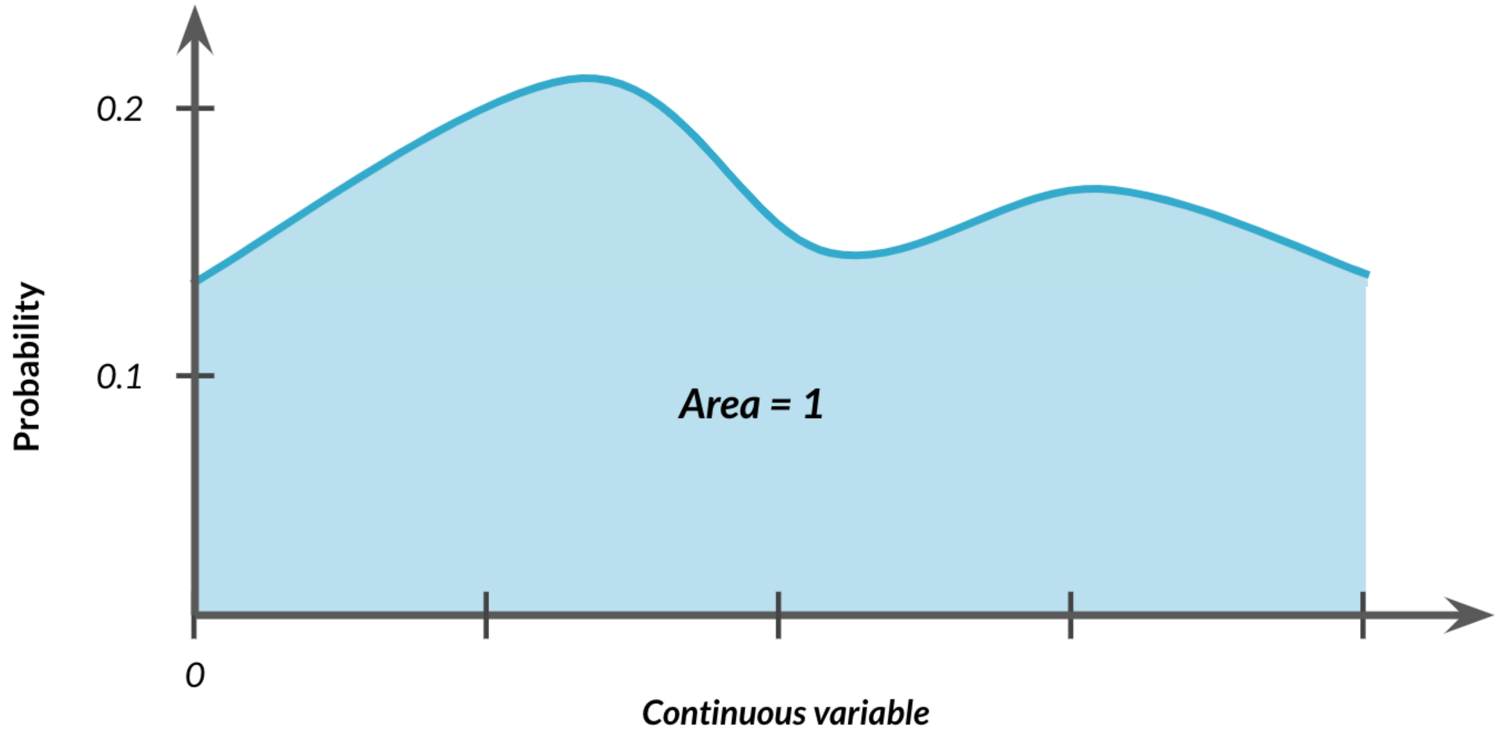
```
array([1.89740094, 4.70673196, 0.33224683, 1.0137103 , 2.31641255,
       3.49969897, 0.29688598, 0.92057234, 4.71086658, 1.56815855])
```

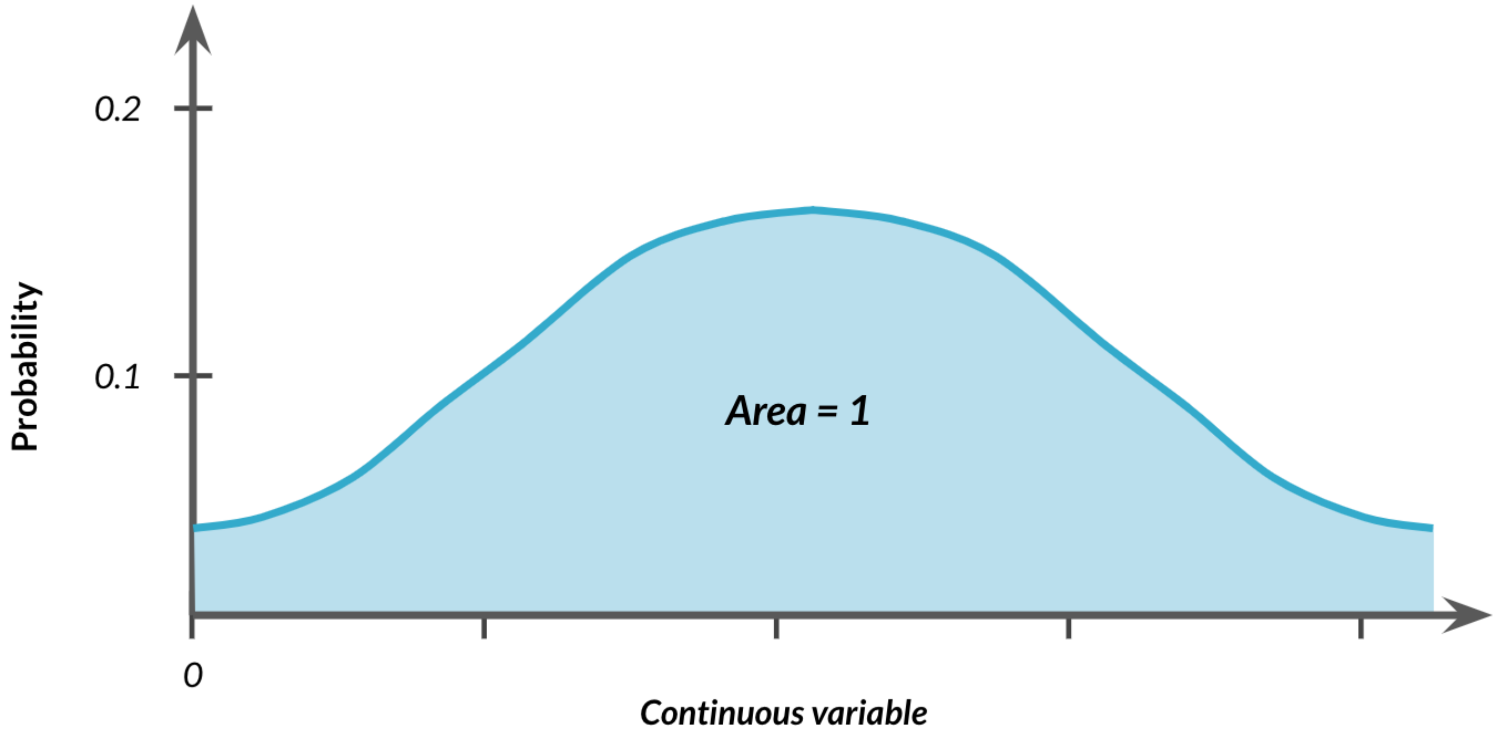
Other continuous distributions





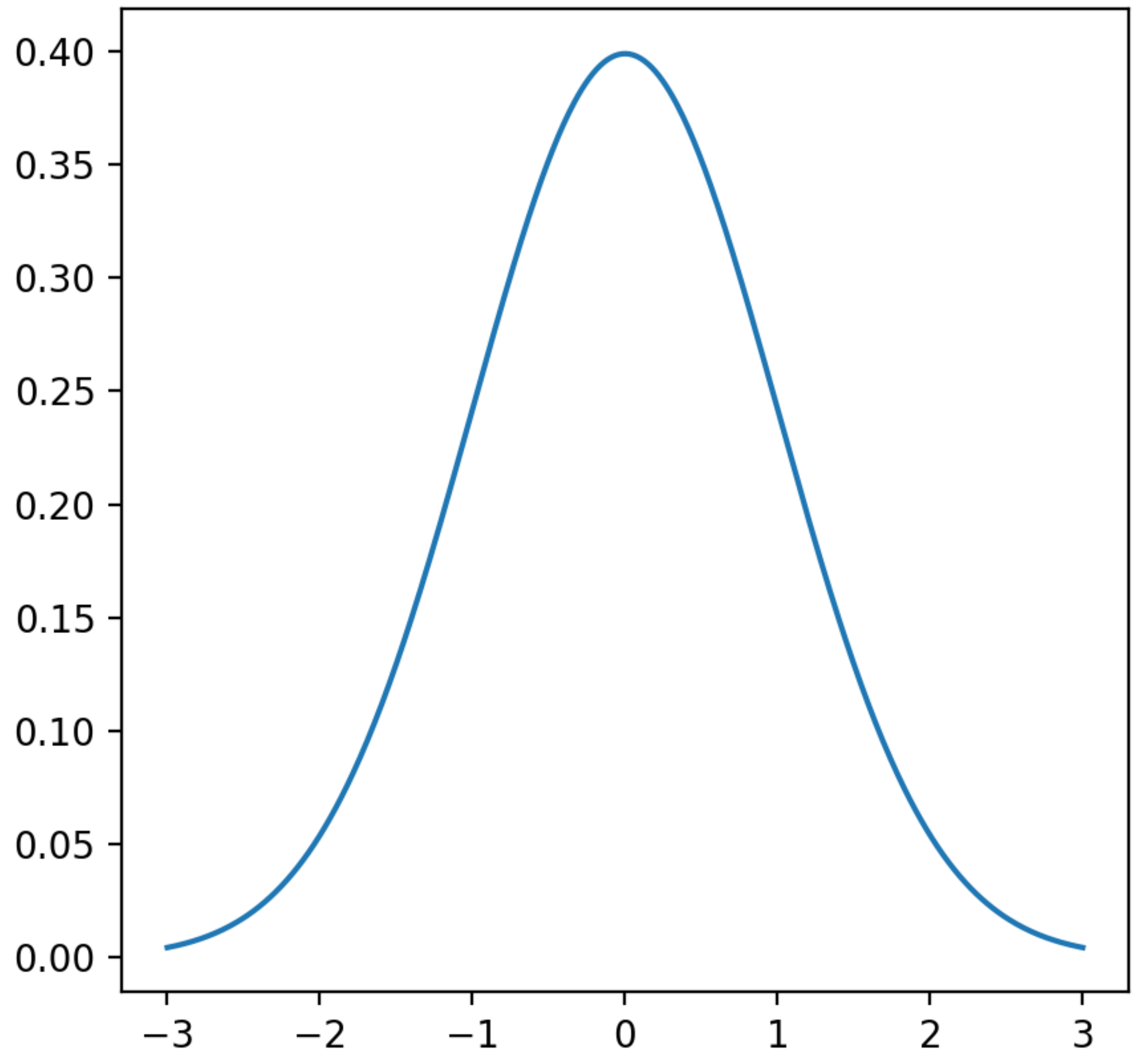
Other continuous distributions



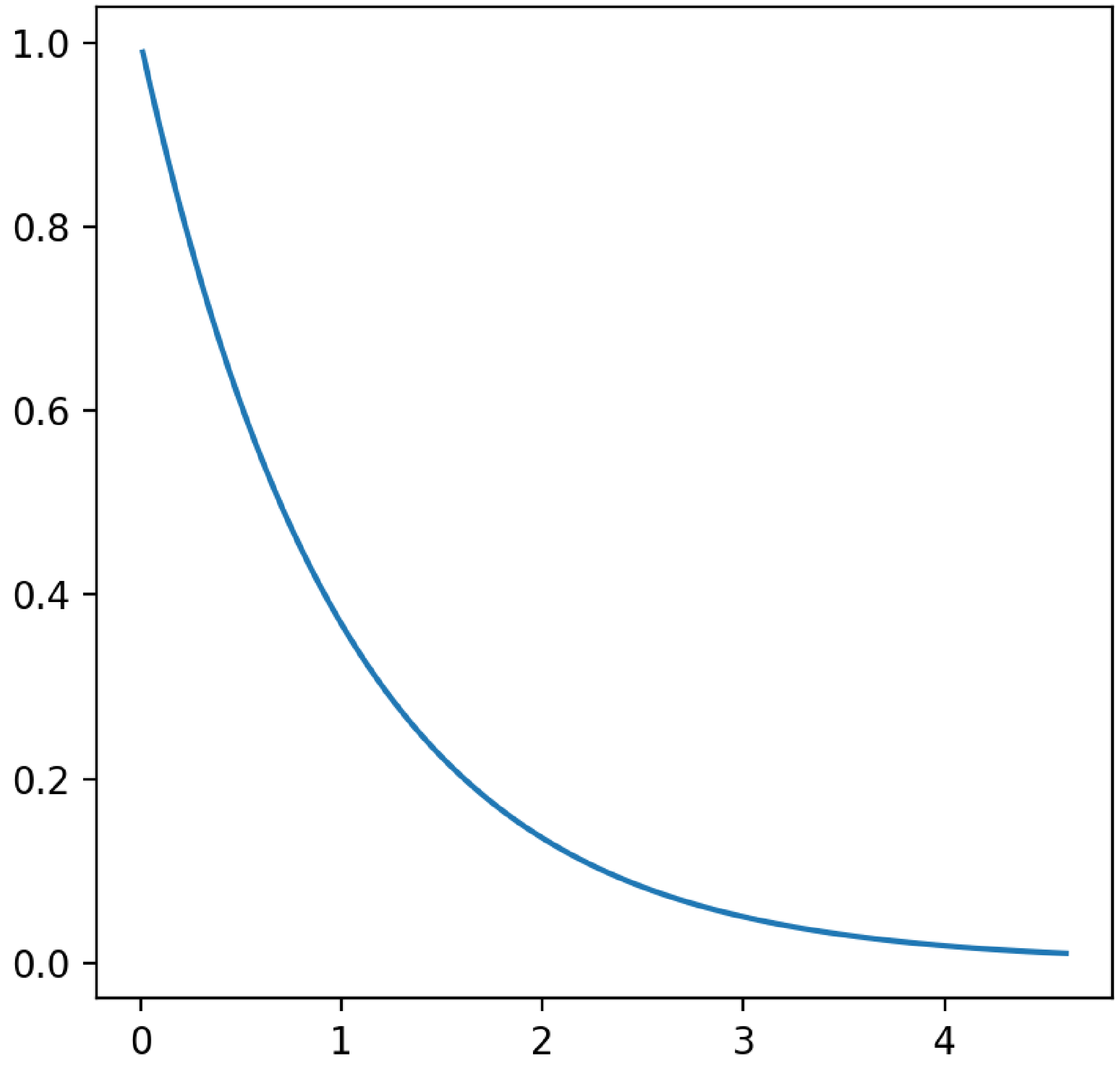


Other special types of distributions

Normal distribution



Exponential distribution



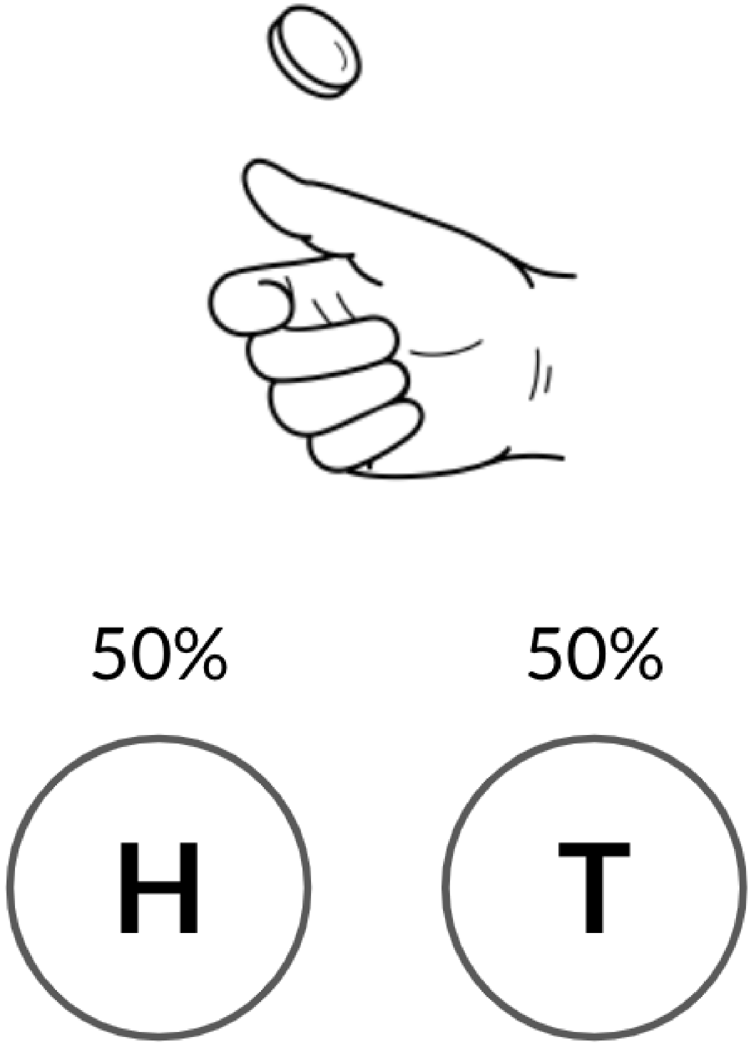
Let's practice!

The binomial distribution

INTRODUCTION TO STATISTICS IN PYTHON

Maggie Matsui
Content Developer, DataCamp

Coin flipping



Binary outcomes

H	T
1	0
Success	Failure
Win	Loss

A single flip

`binom.rvs(# of coins, probability of heads/success, size=# of trials)`

`1` = head, `0` = tails

```
from scipy.stats import binom
binom.rvs(1, 0.5, size=1)
```

```
array([1])
```

One flip many times

```
binom.rvs(1, 0.5, size=8)
```

```
array([0, 1, 1, 0, 1, 0, 1, 1])
```

```
binom.rvs(1, 0.5, size = 8)
```

Flip 1 coin with 50% chance of success 8 times

Many flips one time

```
binom.rvs(8, 0.5, size=1)
```

```
array([5])
```

```
binom.rvs(8, 0.5, size = 1)
```

Flip 8 coins with 50% chance of success 1 time

Many flips many times

```
binom.rvs(3, 0.5, size=10)
```

```
array([0, 3, 2, 1, 3, 0, 2, 2, 0, 0])
```

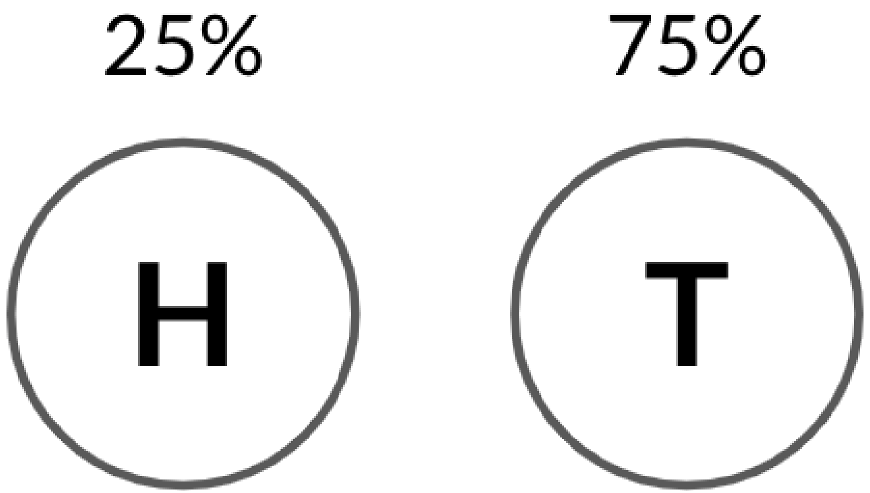
```
binom.rvs(3, 0.5, size = 10)
```

Flip 3 coins with 50% chance of success 10 times

Other probabilities

```
binom.rvs(3, 0.25, size=10)
```

```
array([1, 1, 1, 1, 0, 0, 2, 0, 1, 0])
```



Binomial distribution

Probability distribution of the number of successes in a sequence of independent trials

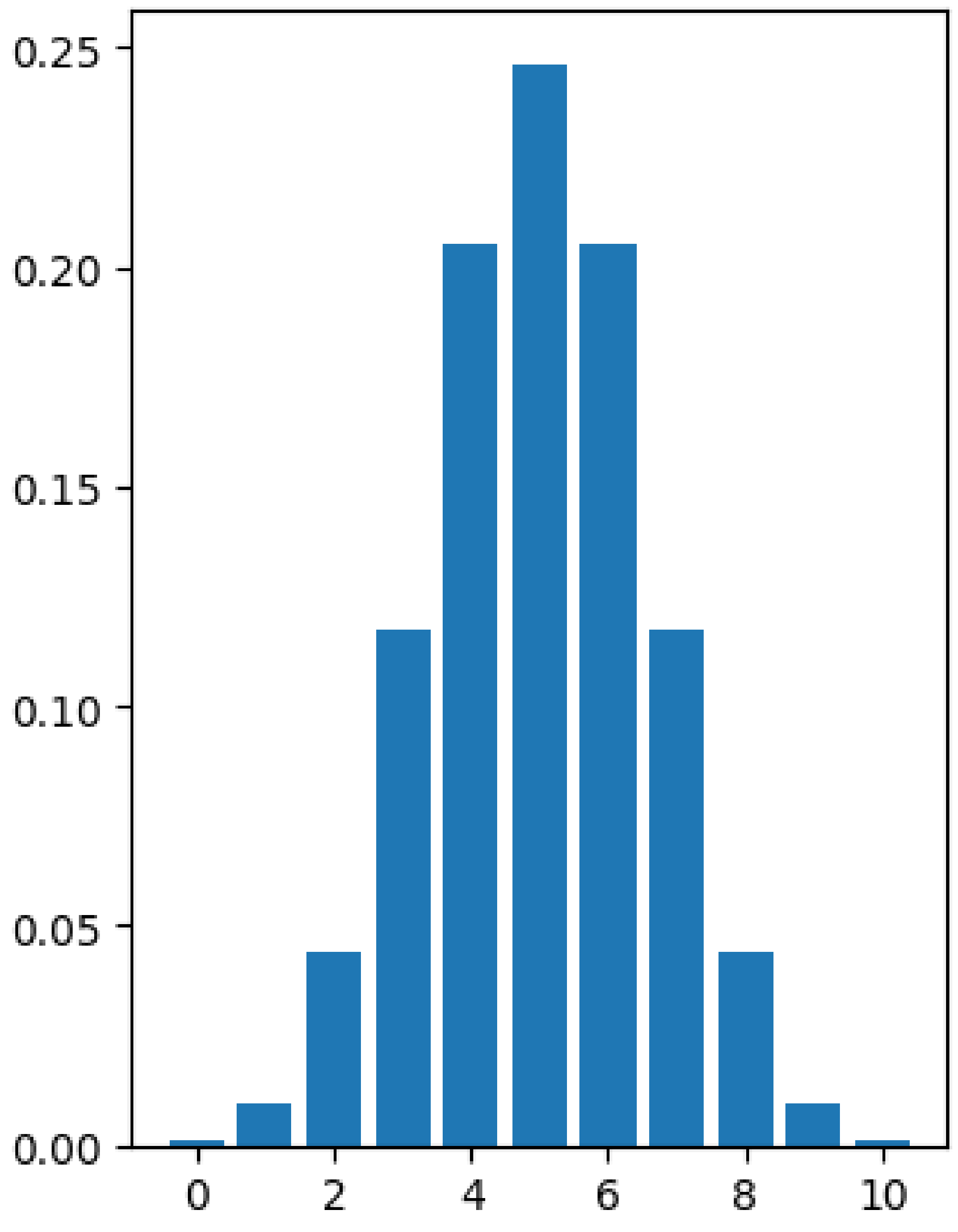
E.g. Number of heads in a sequence of coin flips

Described by n and p

- n: total number of trials
- p: probability of success

p ***n***

```
binom.rvs(3, 0.5, size = 10)
```



What's the probability of 7 heads?

$P(\text{heads} = 7)$

```
# binom.pmf(num heads, num trials, prob of heads)
binom.pmf(7, 10, 0.5)
```

```
0.1171875
```

What's the probability of 7 or fewer heads?

$P(\text{heads} \leq 7)$

```
binom.cdf(7, 10, 0.5)
```

```
0.9453125
```

What's the probability of more than 7 heads?

$P(\text{heads} > 7)$

```
1 - binom.cdf(7, 10, 0.5)
```

```
0.0546875
```

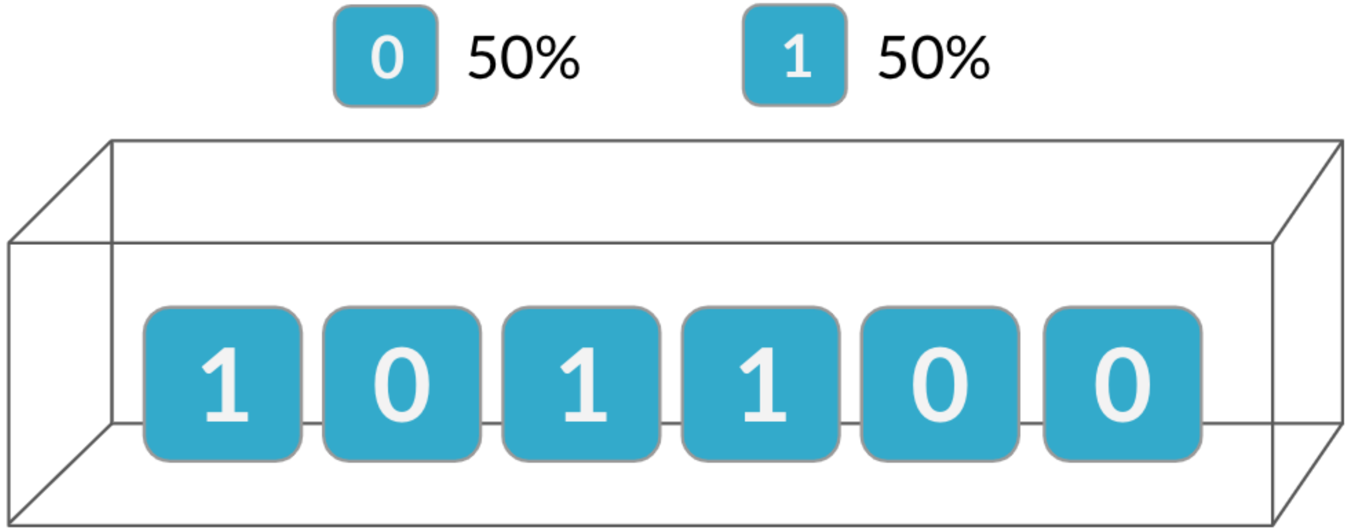
Expected value

$$\text{Expected value} = n \times p$$

Expected number of heads out of 10 flips $= 10 \times 0.5 = 5$

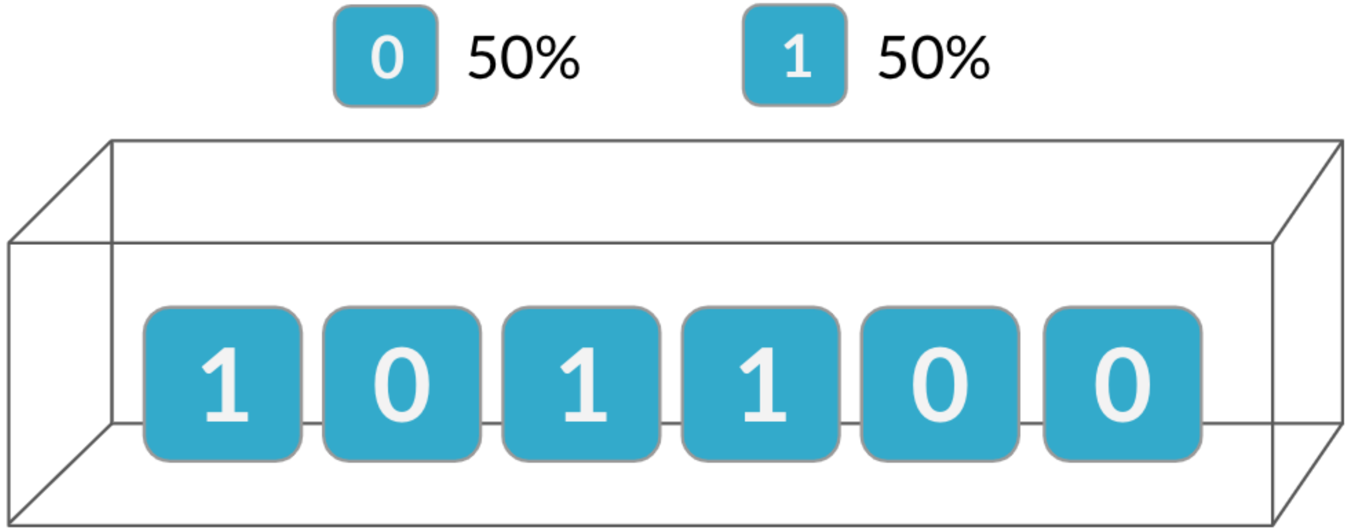
Independence

*The binomial distribution is a probability distribution of the number of successes in a sequence of **independent** trials*

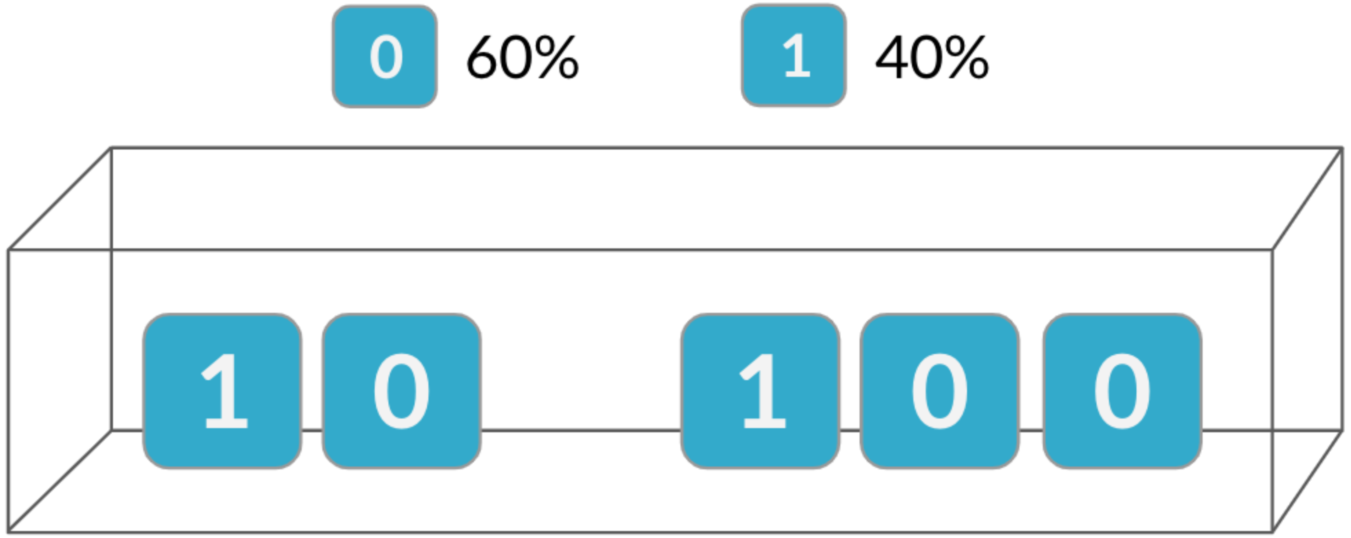


Independence

*The binomial distribution is a probability distribution of the number of successes in a sequence of **independent** trials*



Probabilities of second trial are altered due to outcome of the first



If trials are not independent, the binomial distribution does not apply!

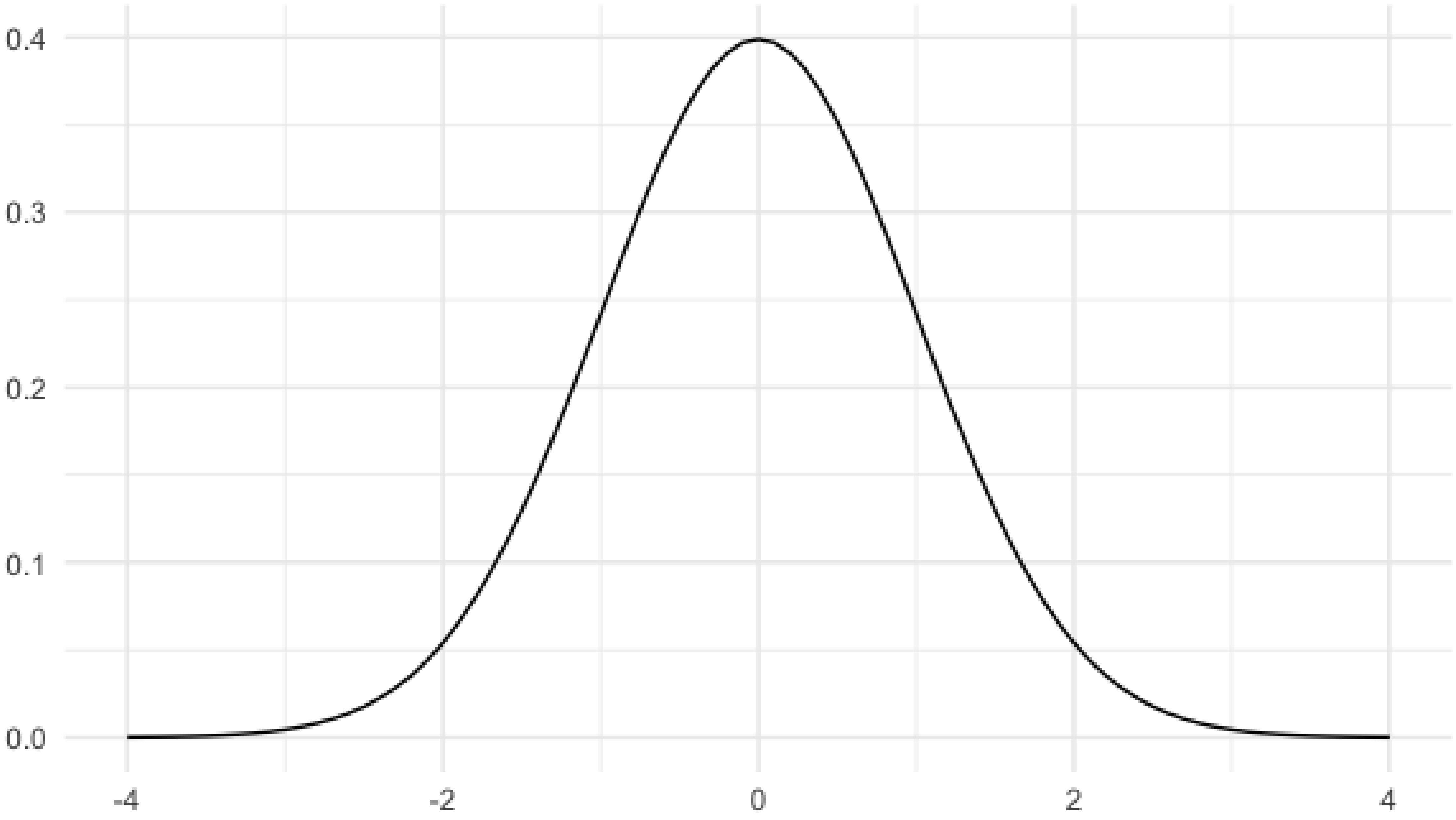
Let's practice!

The normal distribution

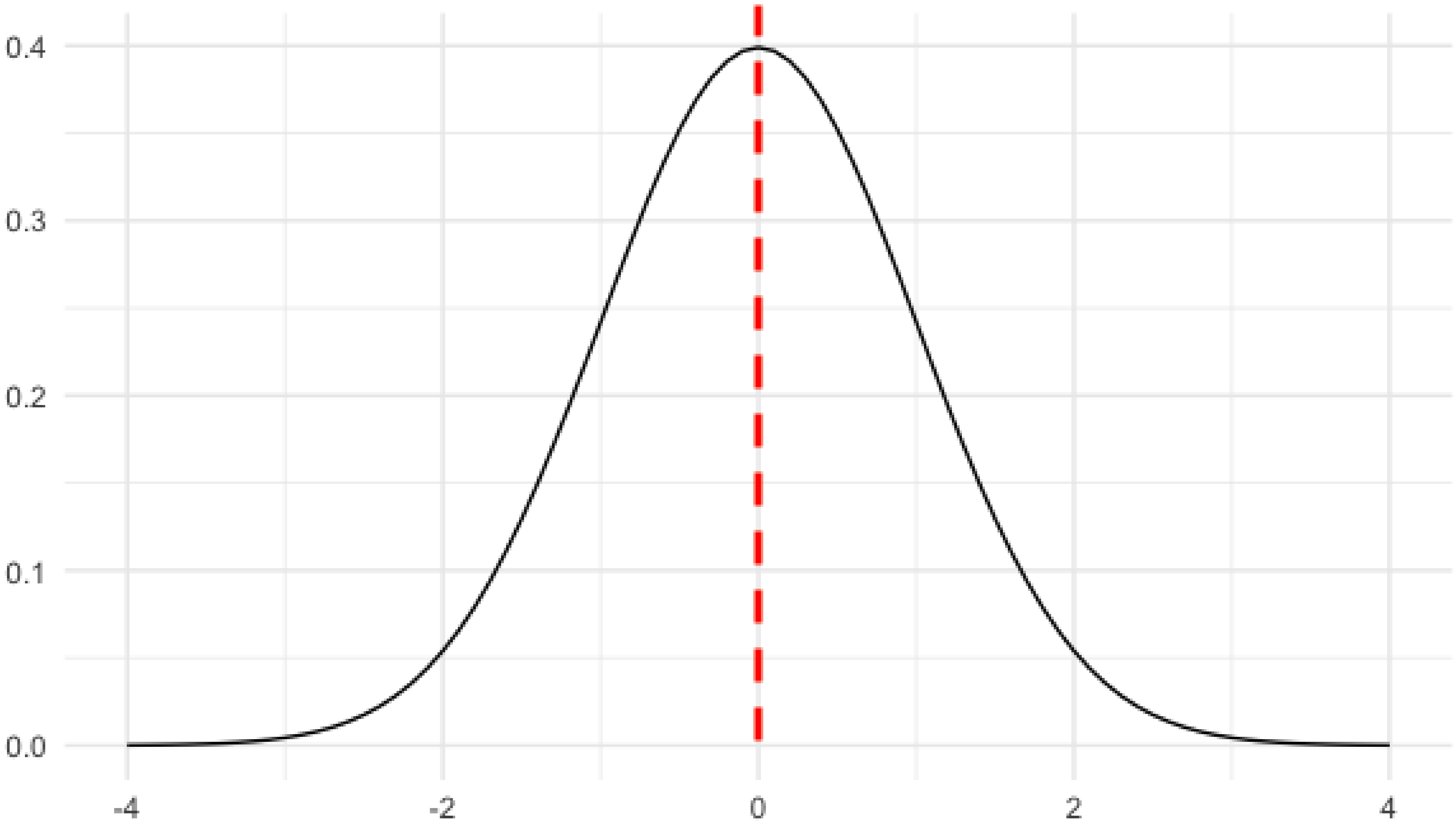
INTRODUCTION TO STATISTICS IN PYTHON

Maggie Matsui
Content Developer, DataCamp

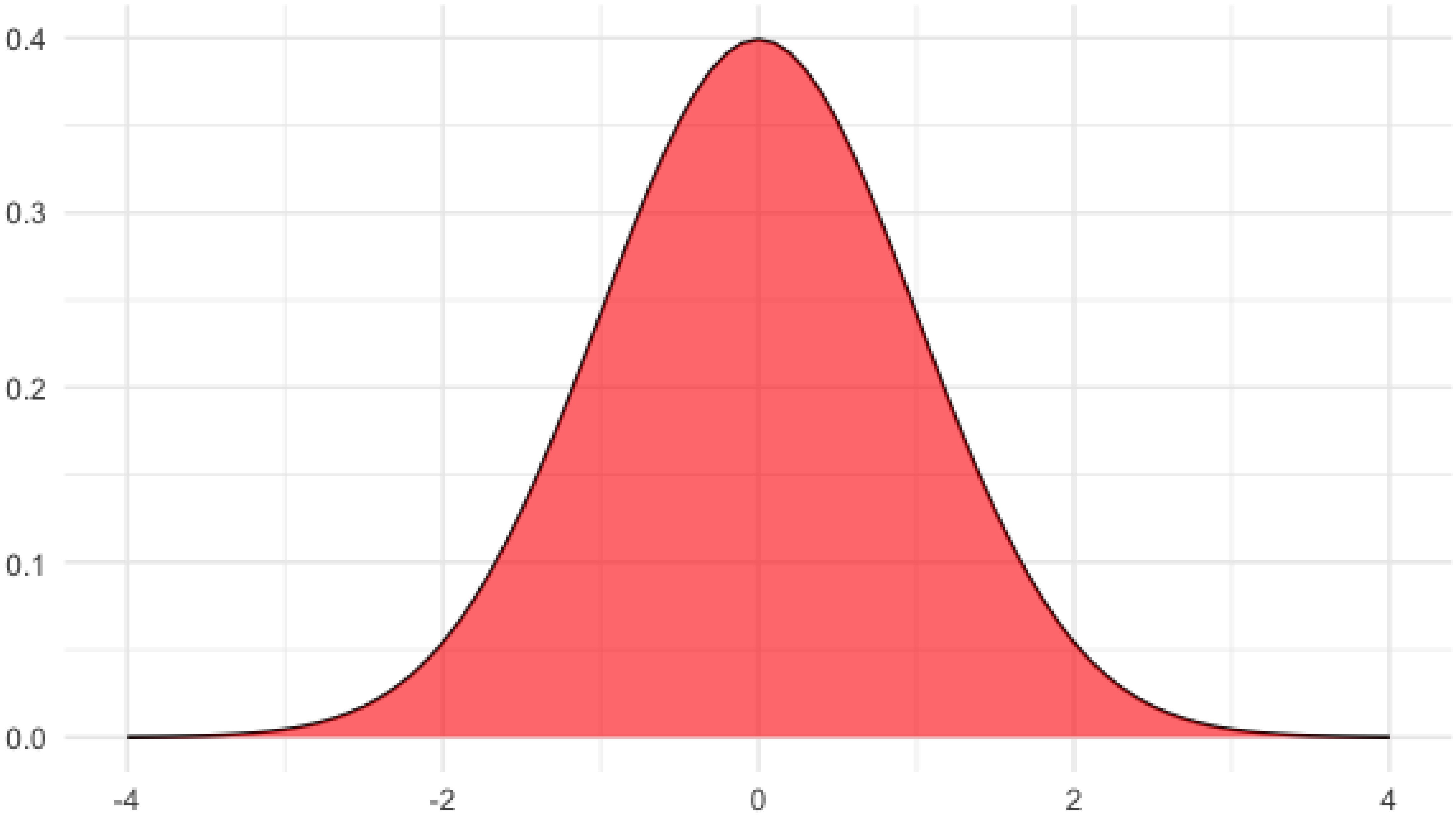
What is the normal distribution?



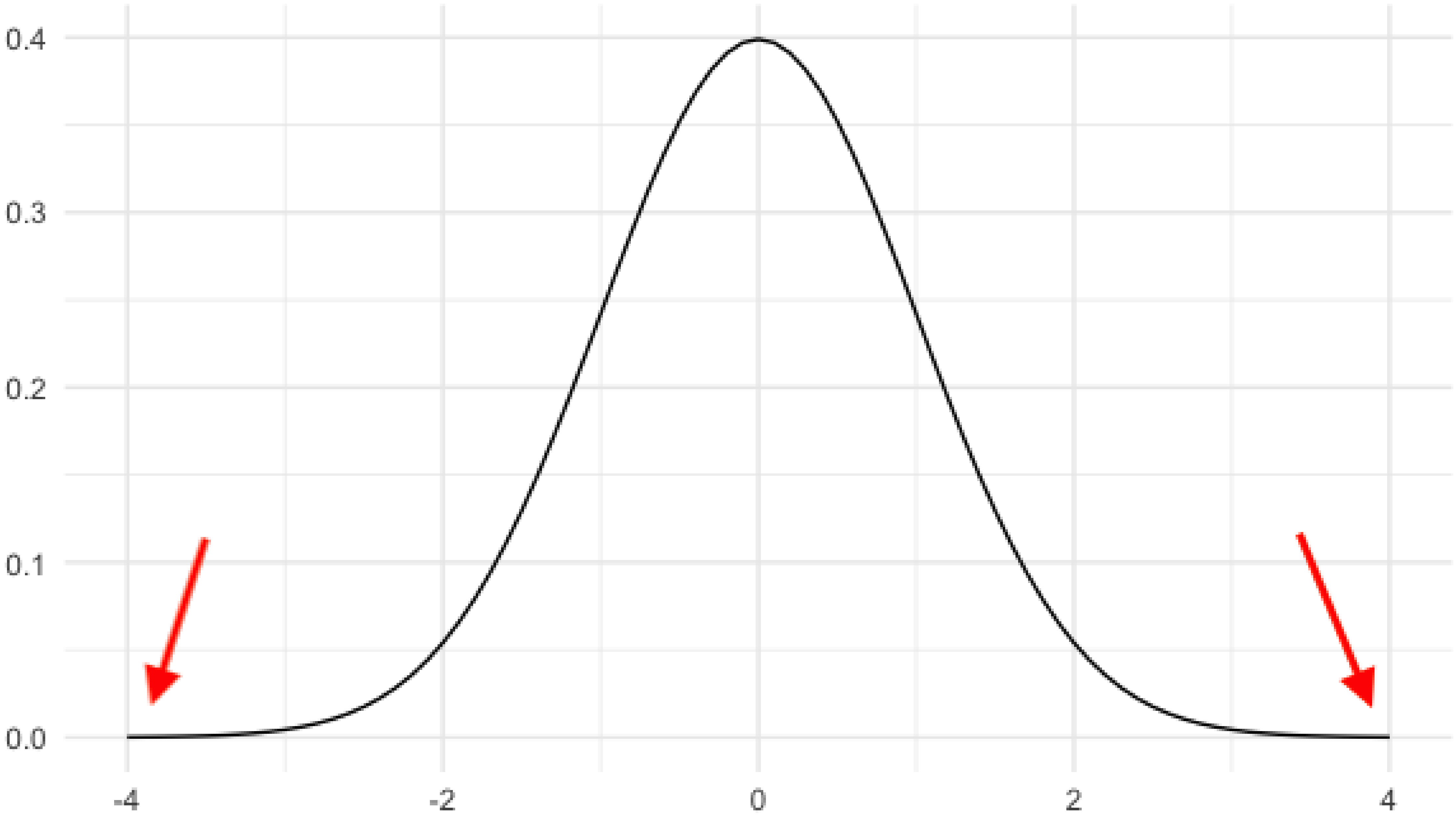
Symmetrical



Area = 1



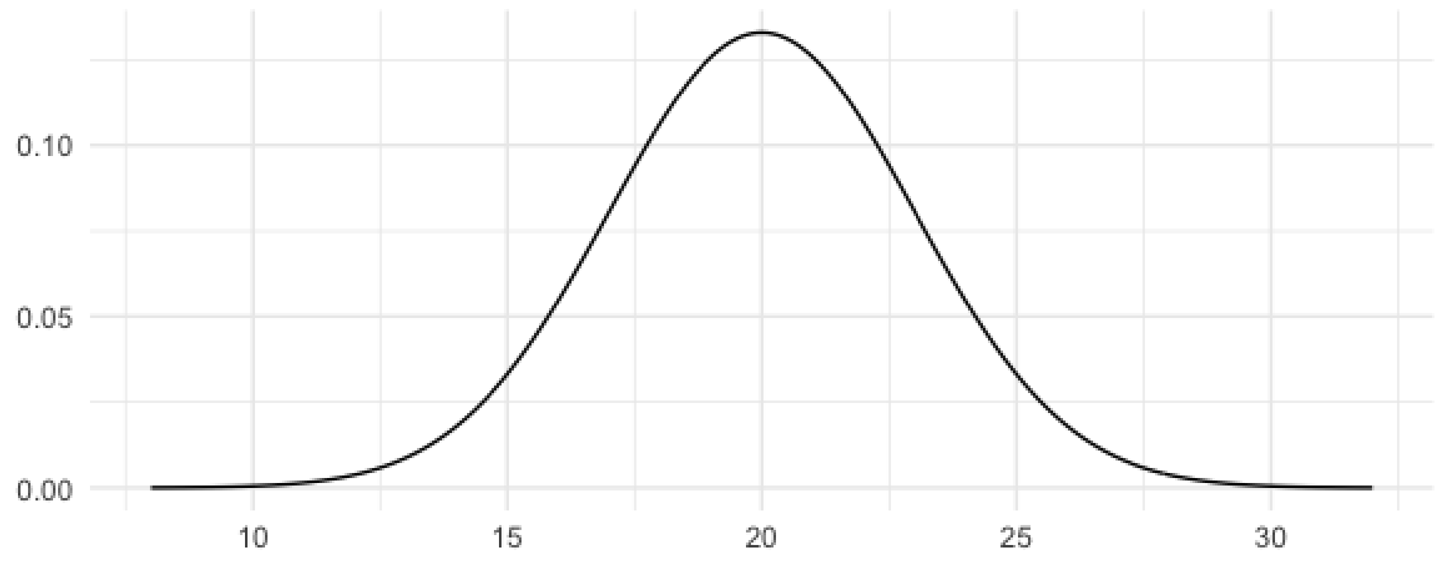
Curve never hits 0



Described by mean and standard deviation

Mean: 20

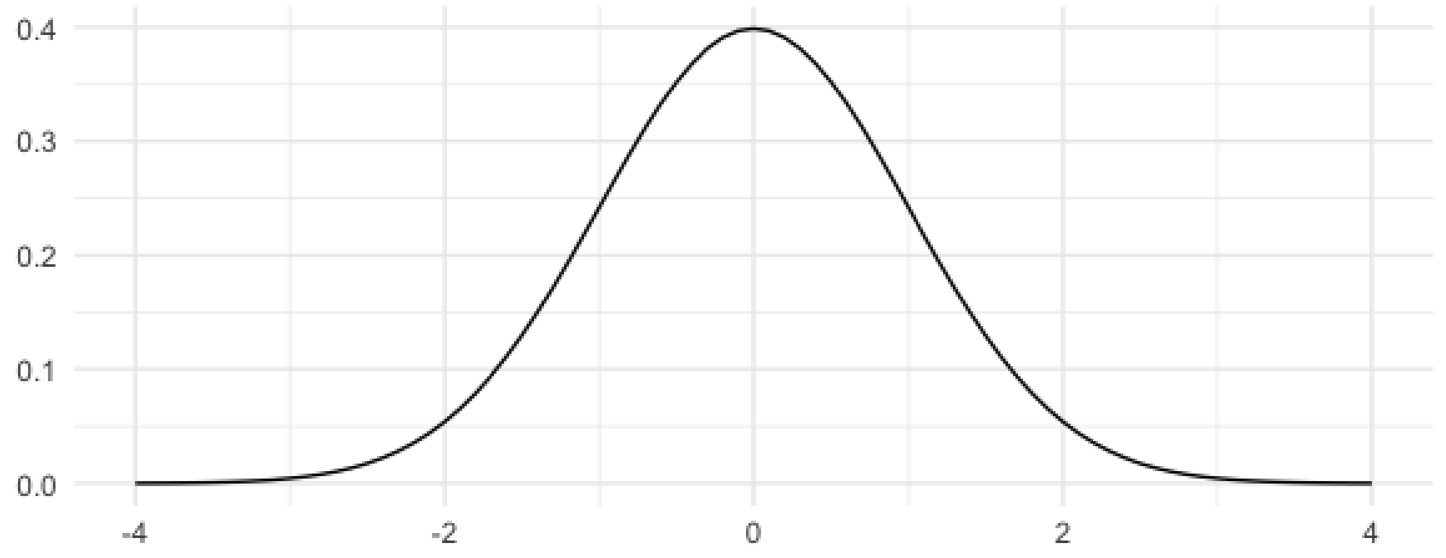
Standard deviation: 3



Standard normal distribution

Mean: 0

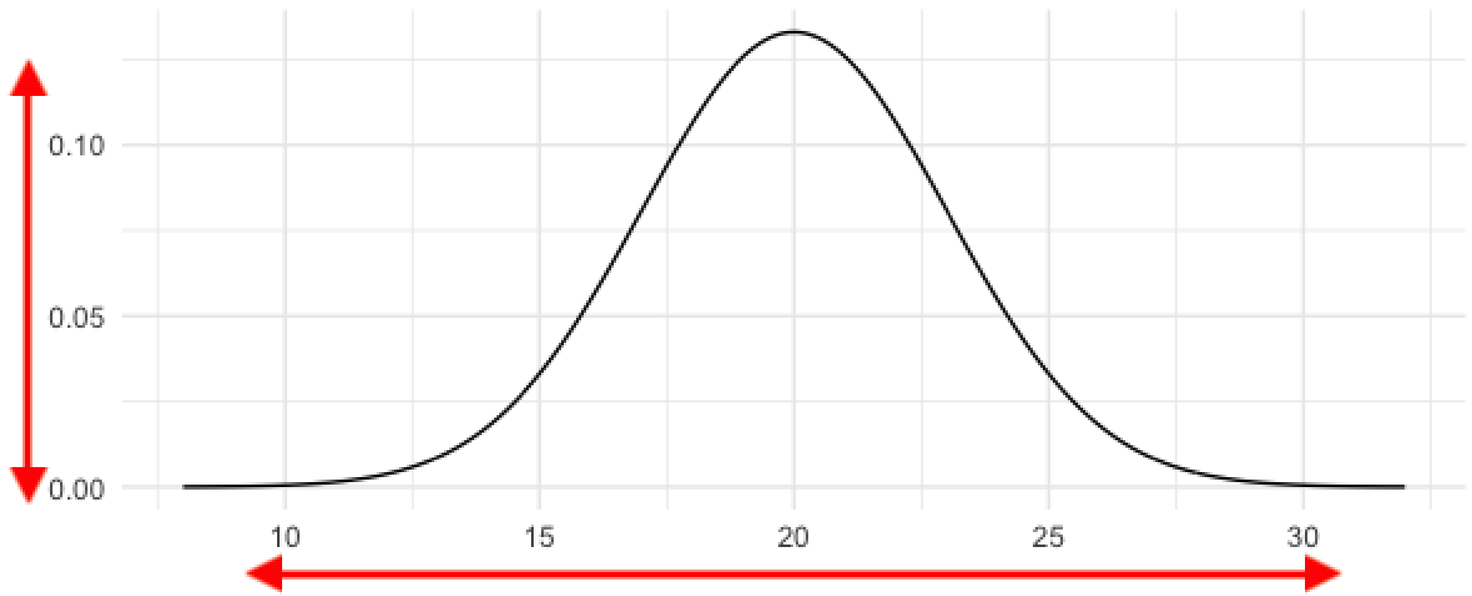
Standard deviation: 1



Described by mean and standard deviation

Mean: 20

Standard deviation: 3



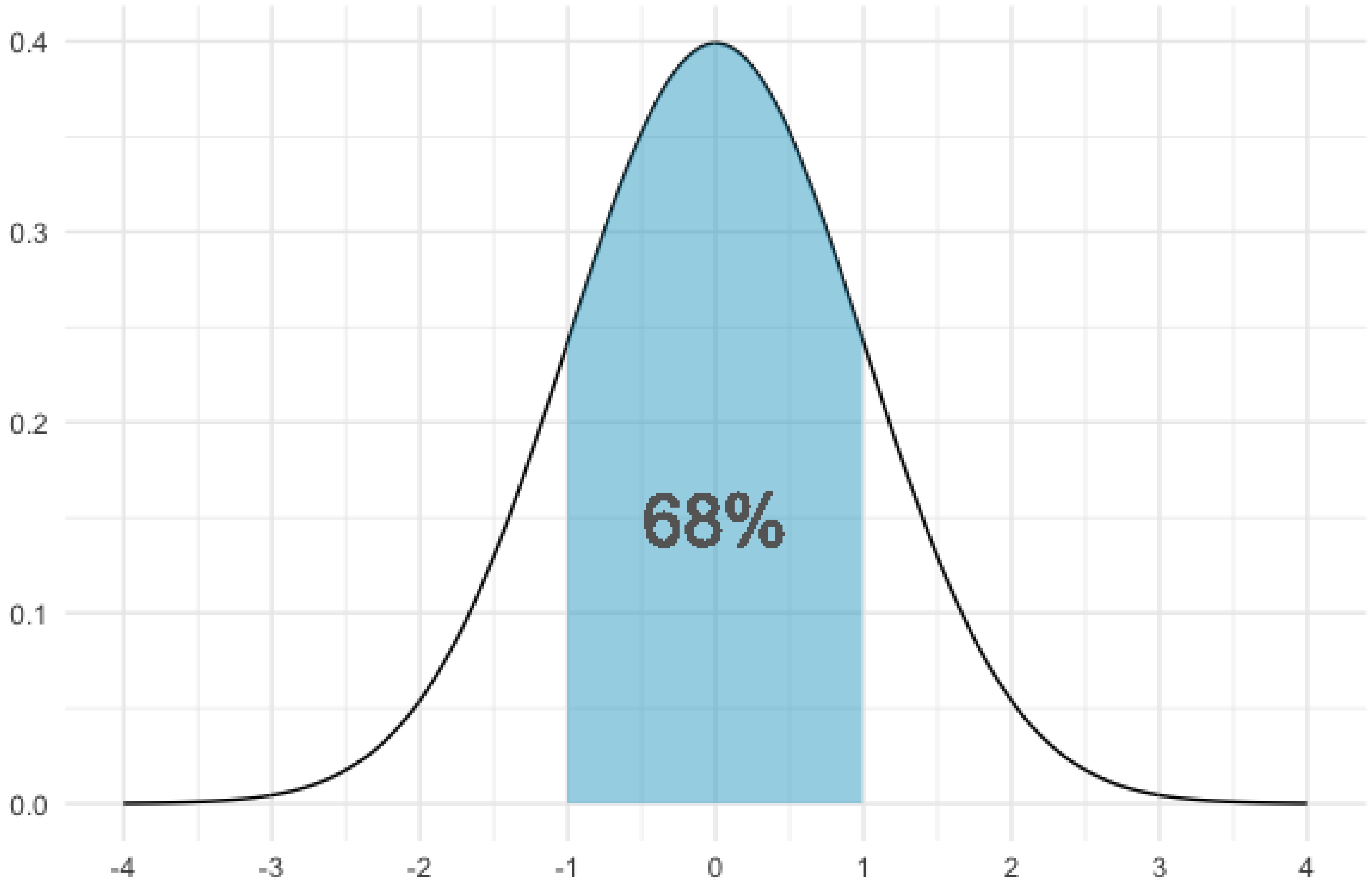
Standard normal distribution

Mean: 0

Standard deviation: 1

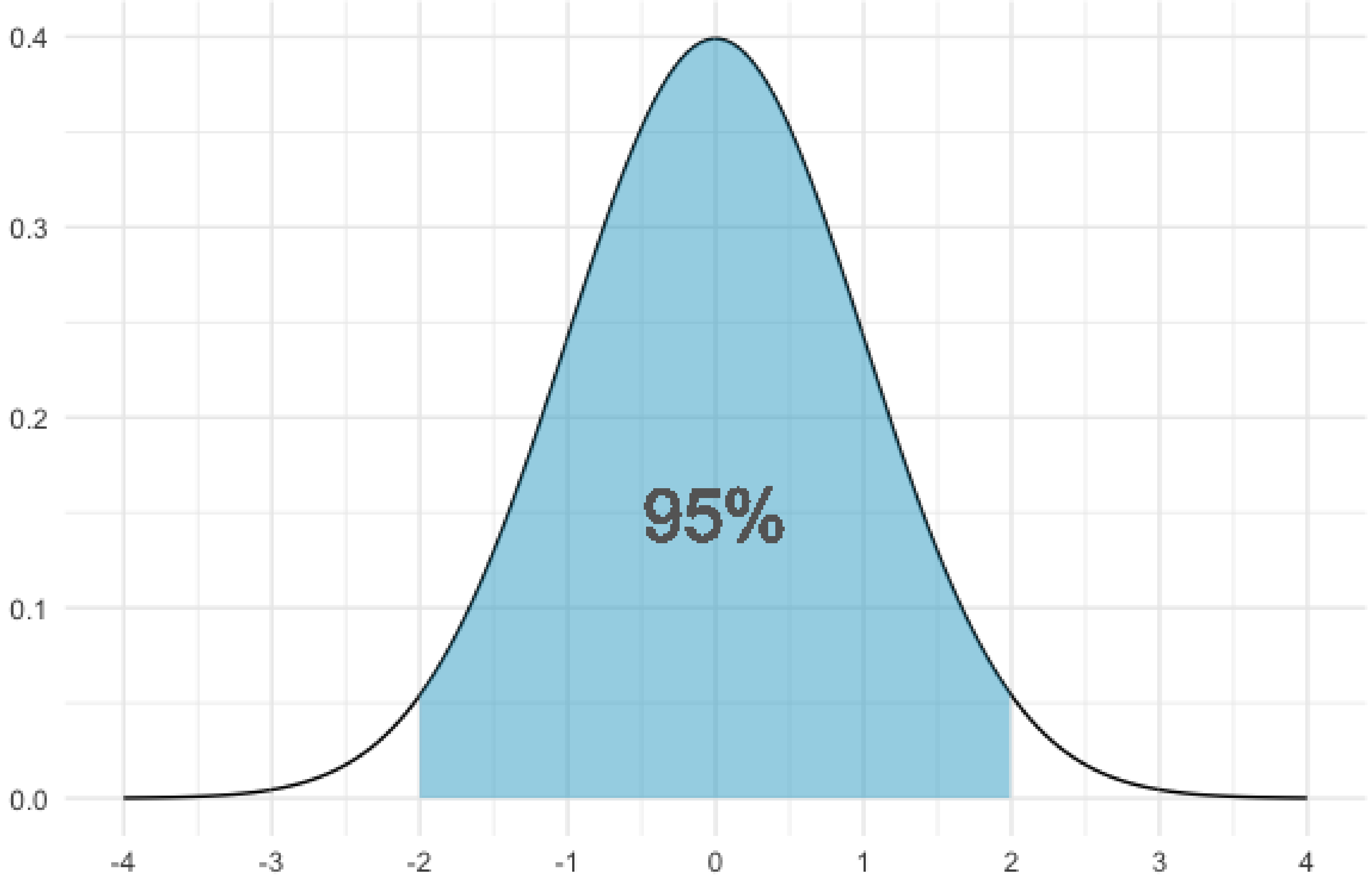
Areas under the normal distribution

68% falls within 1 standard deviation



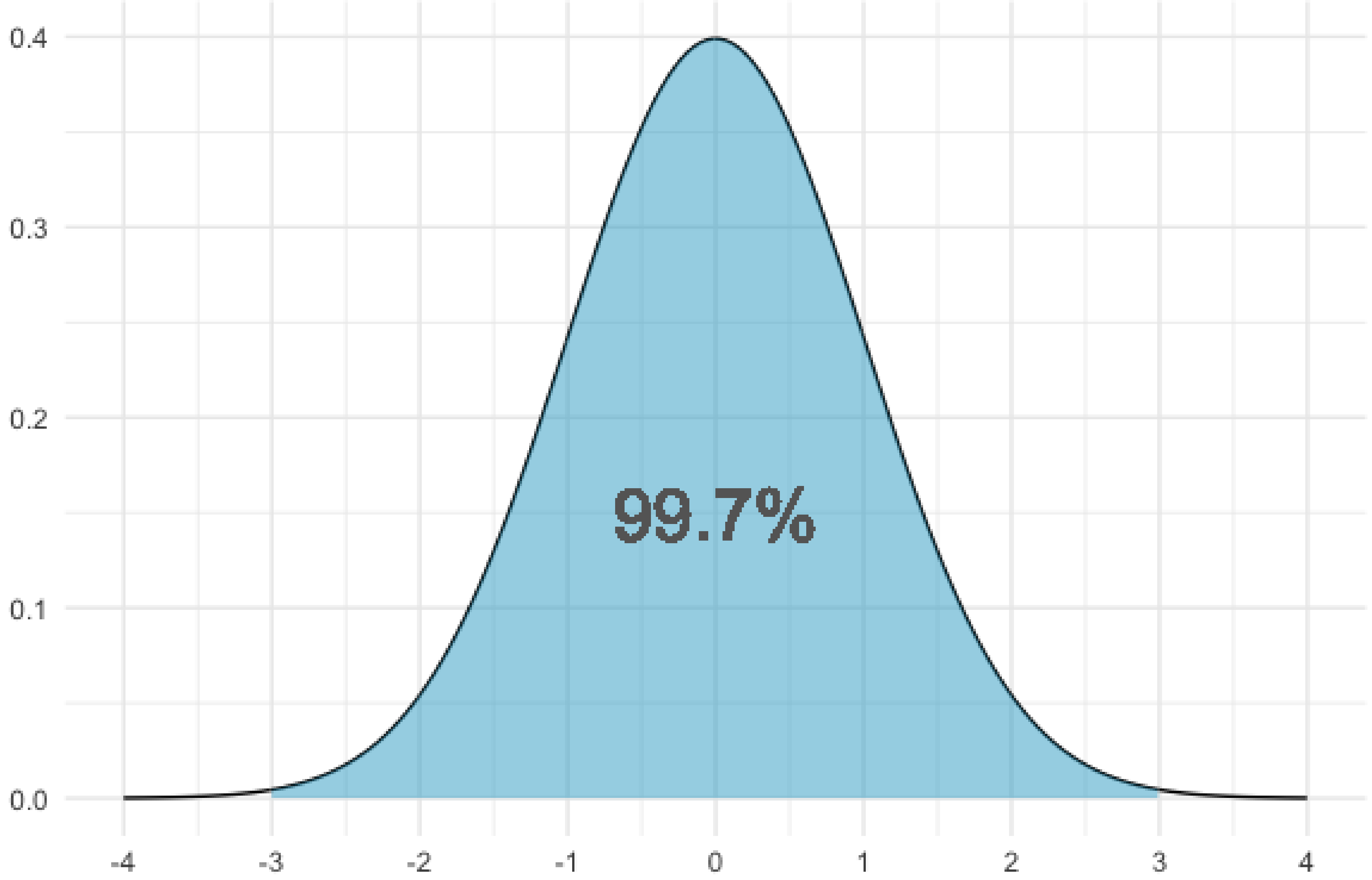
Areas under the normal distribution

95% falls within 2 standard deviations



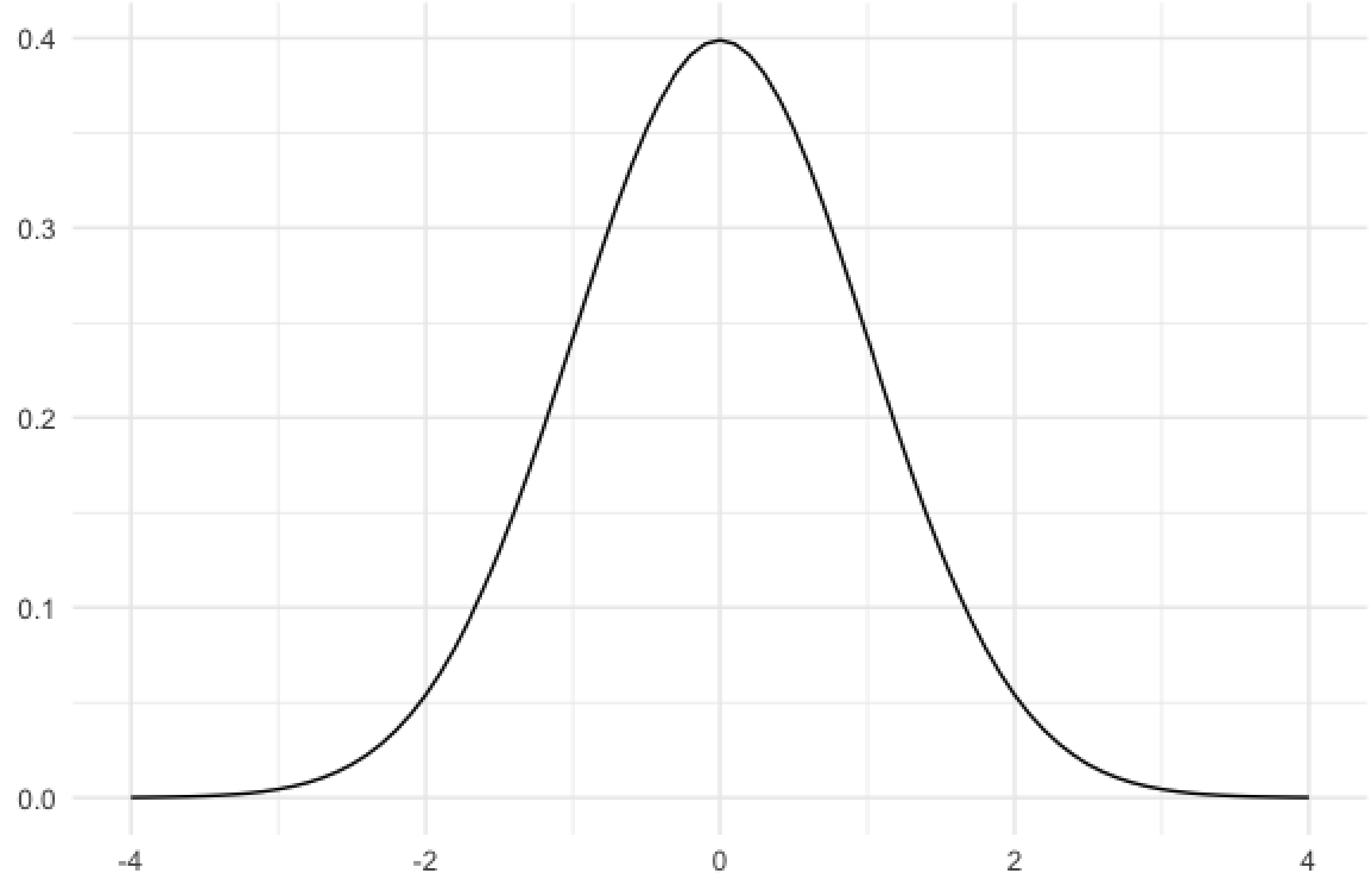
Areas under the normal distribution

99.7% falls within 3 standard deviations

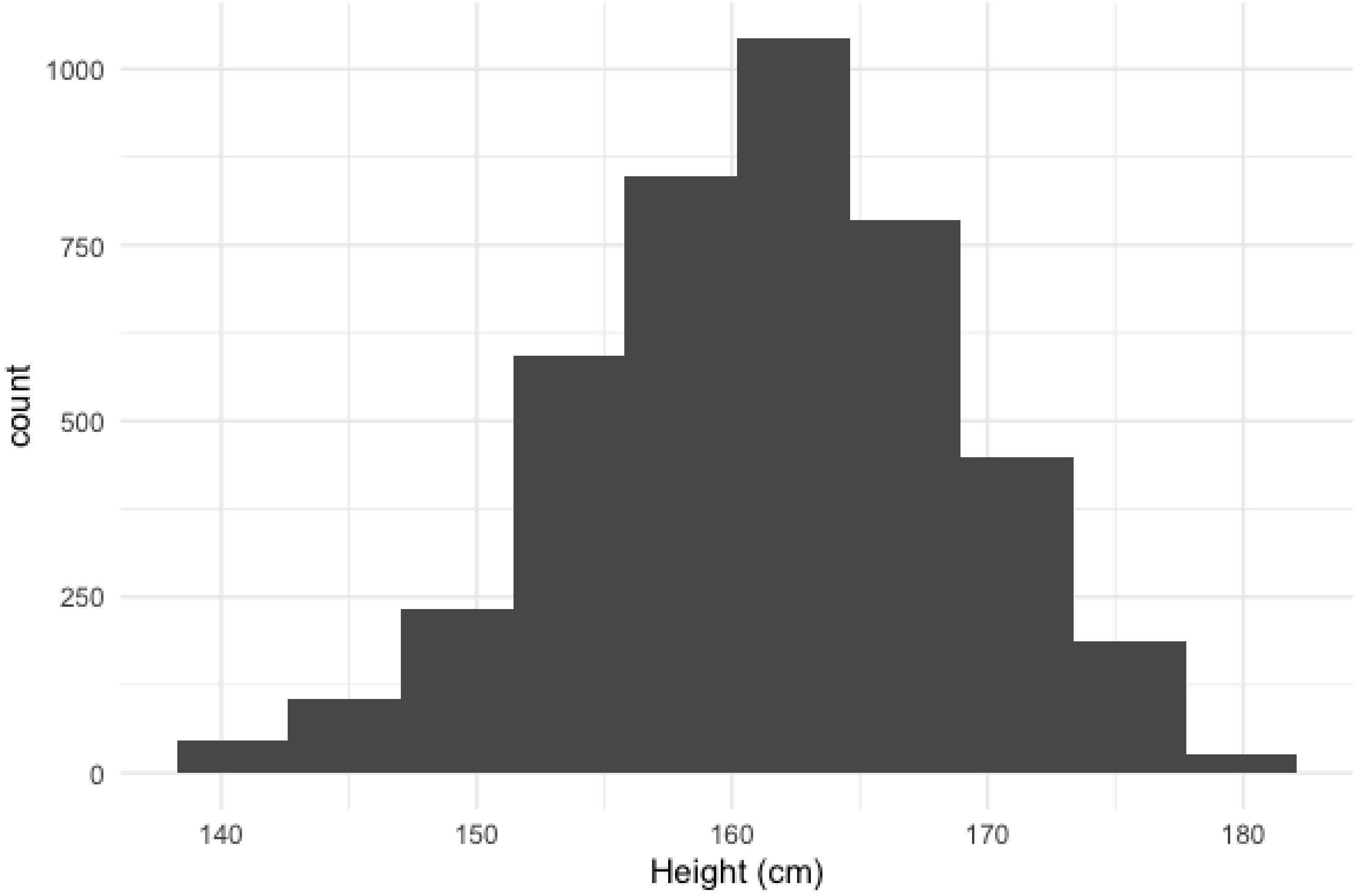


Lots of histograms look normal

Normal distribution

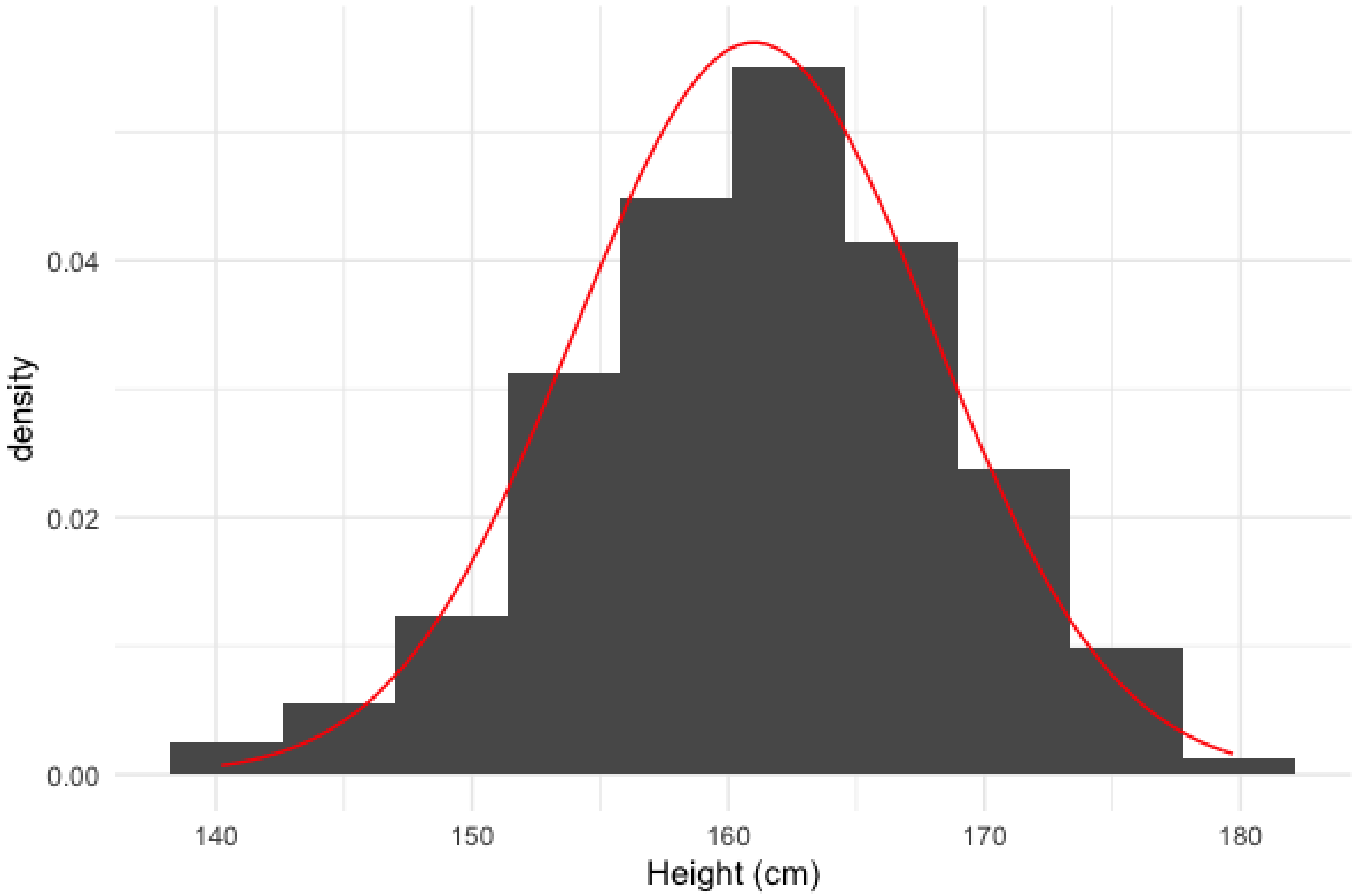


Women's heights from NHANES

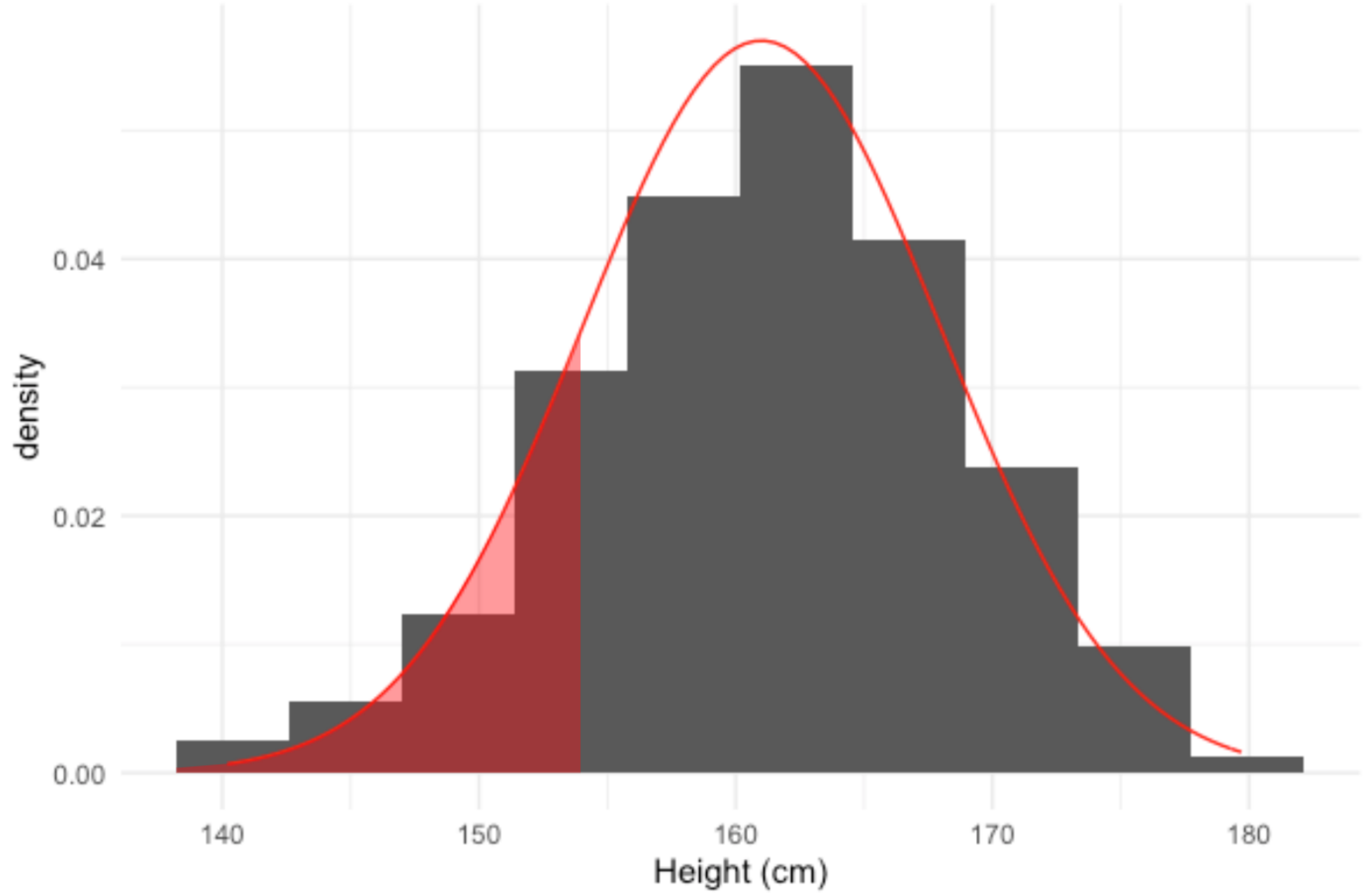


Mean: 161 cm *Standard deviation: 7 cm*

Approximating data with the normal distribution



What percent of women are shorter than 154 cm?

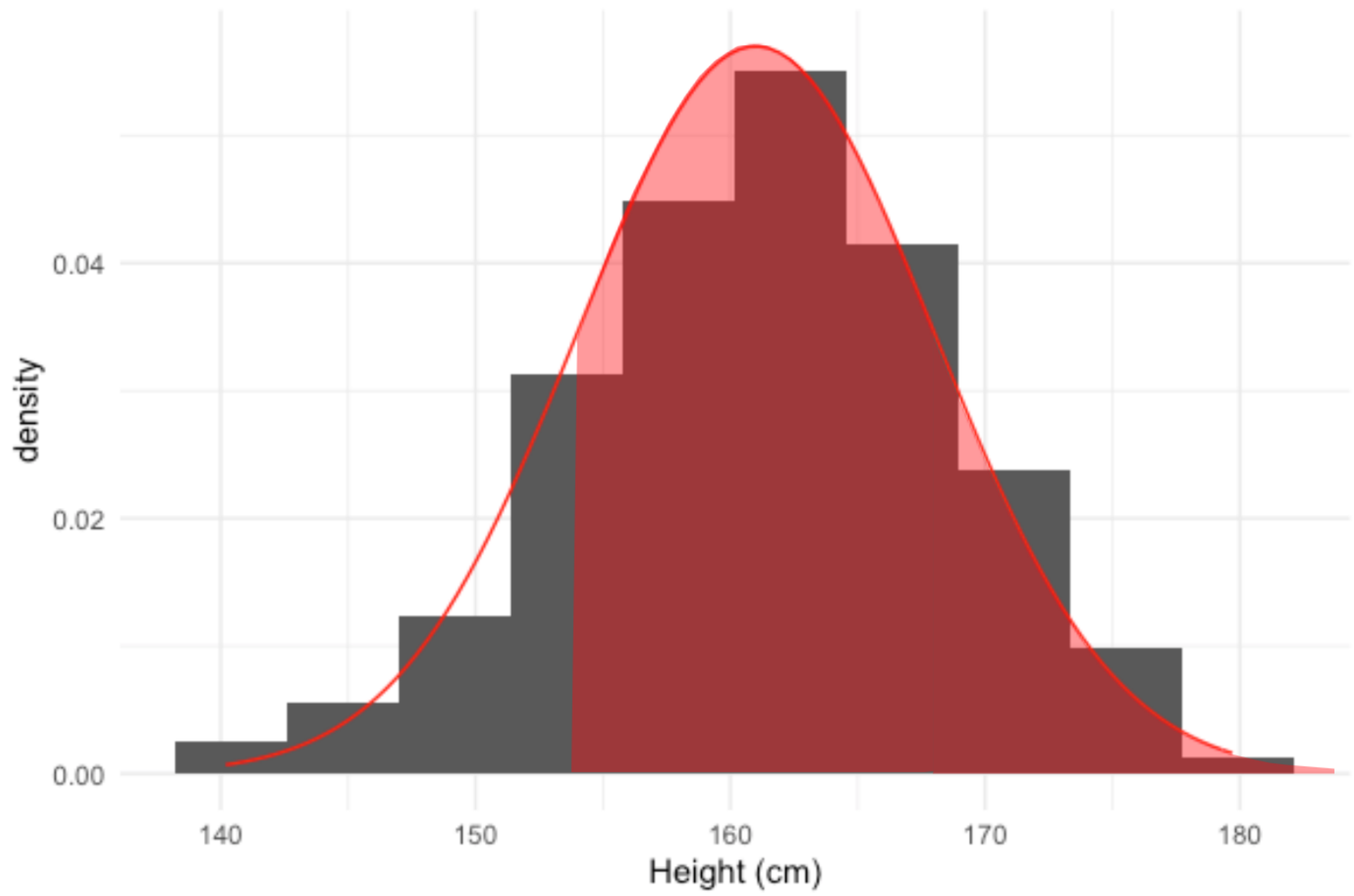



```python
from scipy.stats import norm
norm.cdf(154, 161, 7)
```

```
0.158655
```

16% of women in the survey are shorter than 154 cm

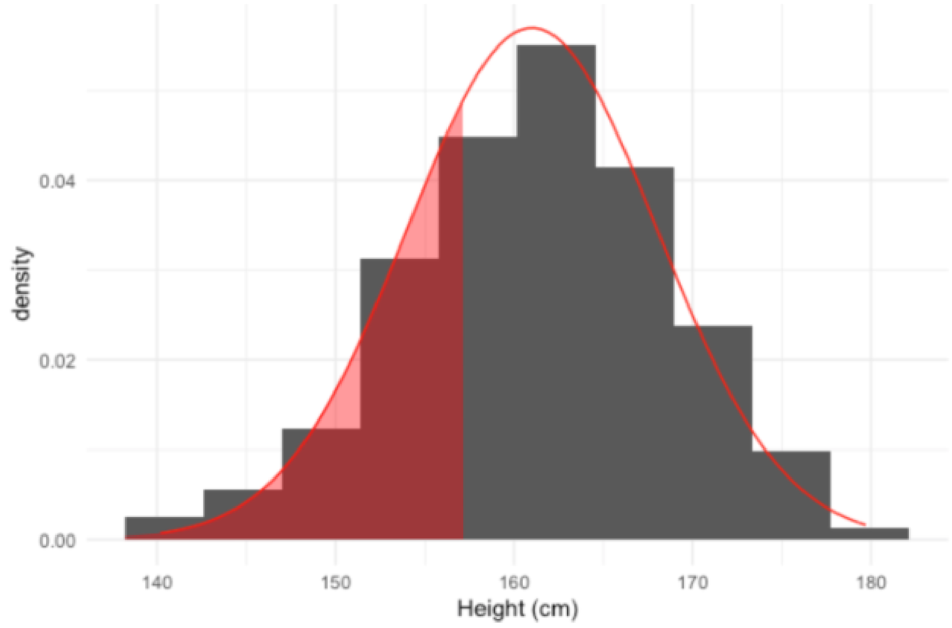
What percent of women are taller than 154 cm?



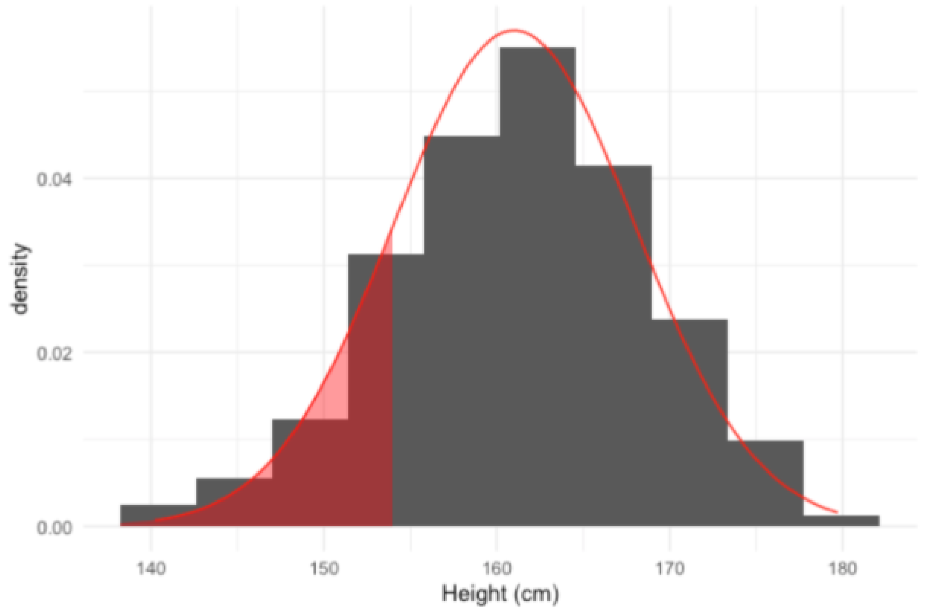

```python
from scipy.stats import norm
1 - norm.cdf(154, 161, 7)
```

```
0.841345
```

What percent of women are 154-157 cm?

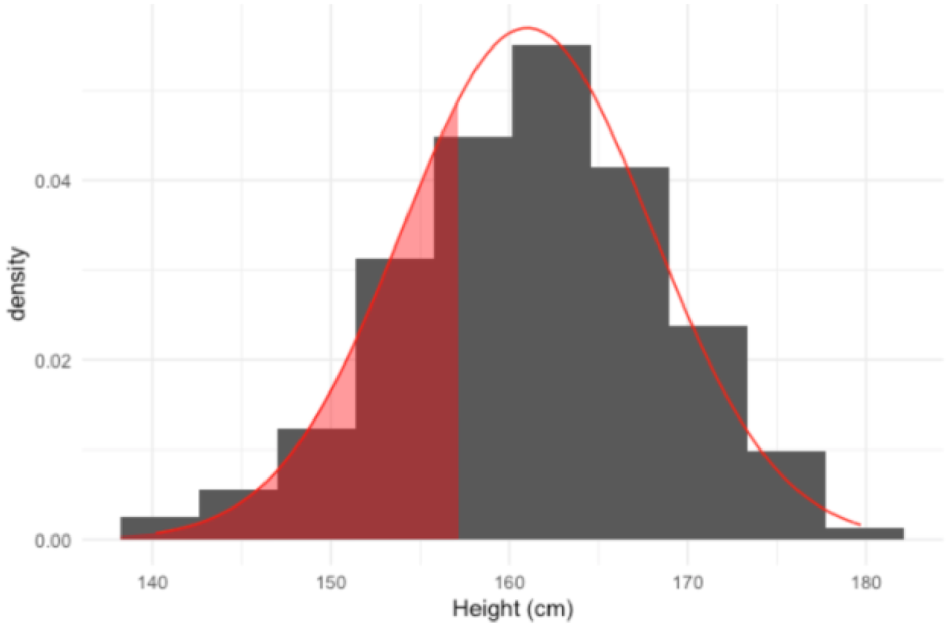


-

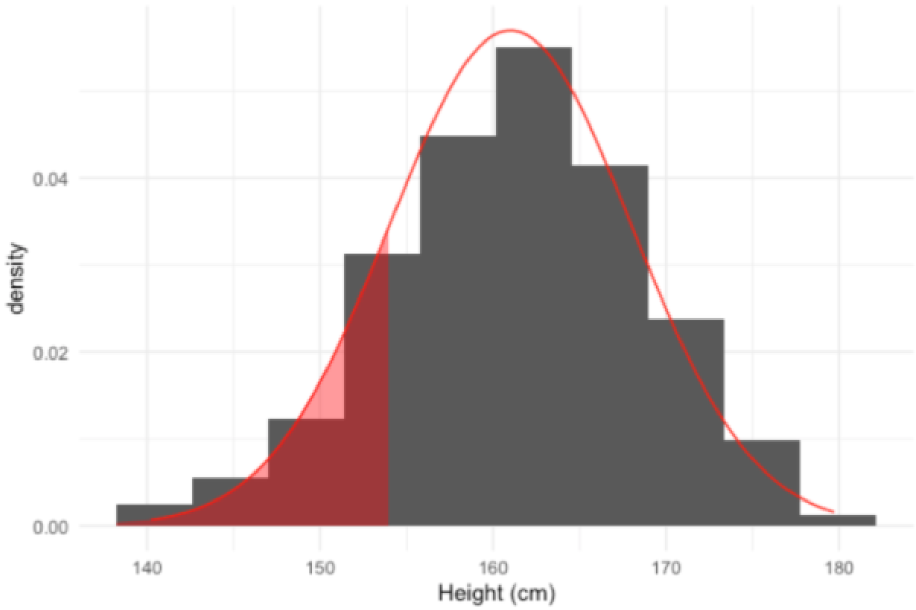



```
norm.cdf(157, 161, 7) - norm.cdf(154, 161, 7)
```

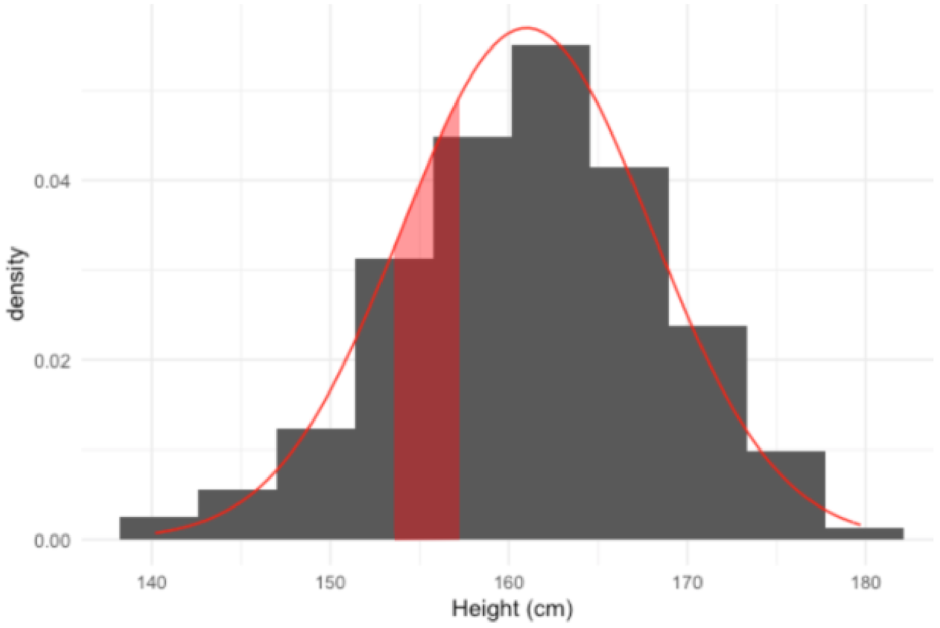
What percent of women are 154-157 cm?



-



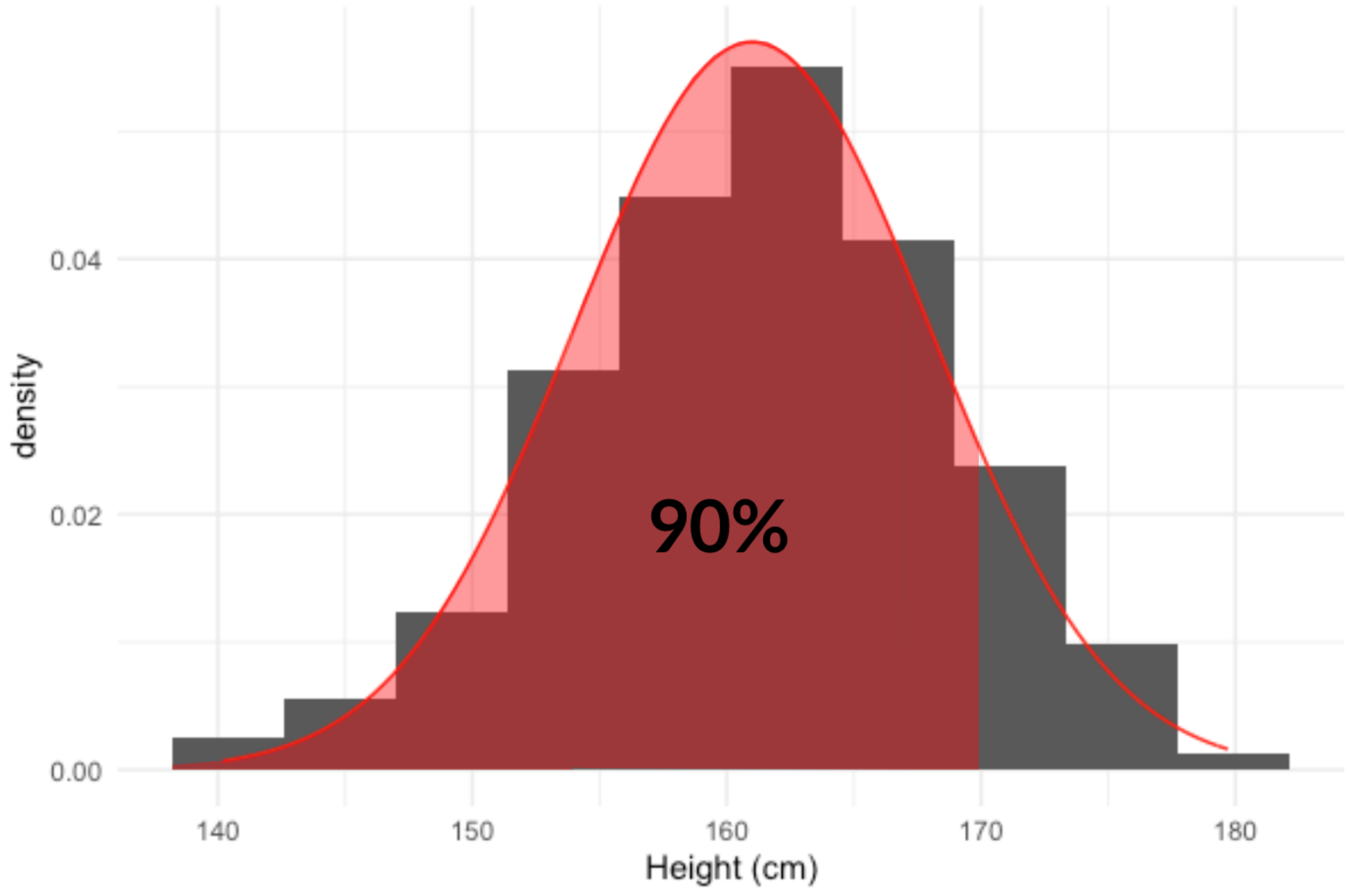
=




```
norm.cdf(157, 161, 7) - norm.cdf(154, 161, 7)
```

```
0.1252
```

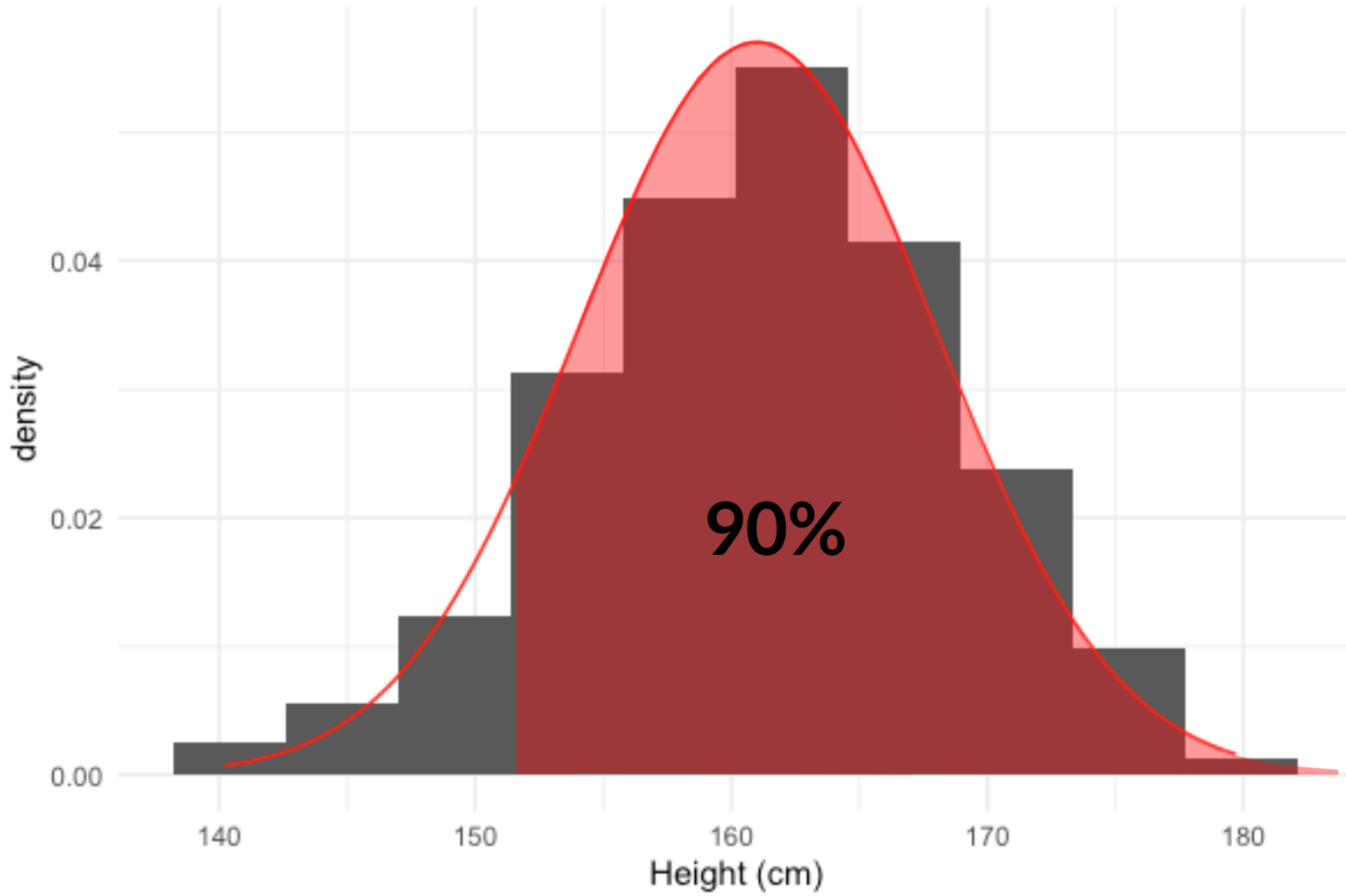
What height are 90% of women shorter than?




```
norm.ppf(0.9, 161, 7)
```

```
169.97086
```

What height are 90% of women taller than?




```
norm.ppf((1-0.9), 161, 7)
```

```
152.029
```

Generating random numbers

```
# Generate 10 random heights
norm.rvs(161, 7, size=10)
```

```
array([155.5758223 , 155.13133235, 160.06377097, 168.33345778,
       165.92273375, 163.32677057, 165.13280753, 146.36133538,
       149.07845021, 160.5790856 ])
```

Let's practice!

The central limit theorem

INTRODUCTION TO STATISTICS IN PYTHON

Maggie Matsui
Content Developer, DataCamp

Rolling the dice 5 times

```
die = pd.Series([1, 2, 3, 4, 5, 6])
# Roll 5 times
samp_5 = die.sample(5, replace=True)
print(samp_5)
```

```
array([3, 1, 4, 1, 1])
```

```
np.mean(samp_5)
```

```
2.0
```

Rolling the dice 5 times

```
# Roll 5 times and take mean
samp_5 = die.sample(5, replace=True)
np.mean(samp_5)
```

```
4.4
```

```
samp_5 = die.sample(5, replace=True)
np.mean(samp_5)
```

```
3.8
```

Rolling the dice 5 times 10 times

Repeat 10 times:

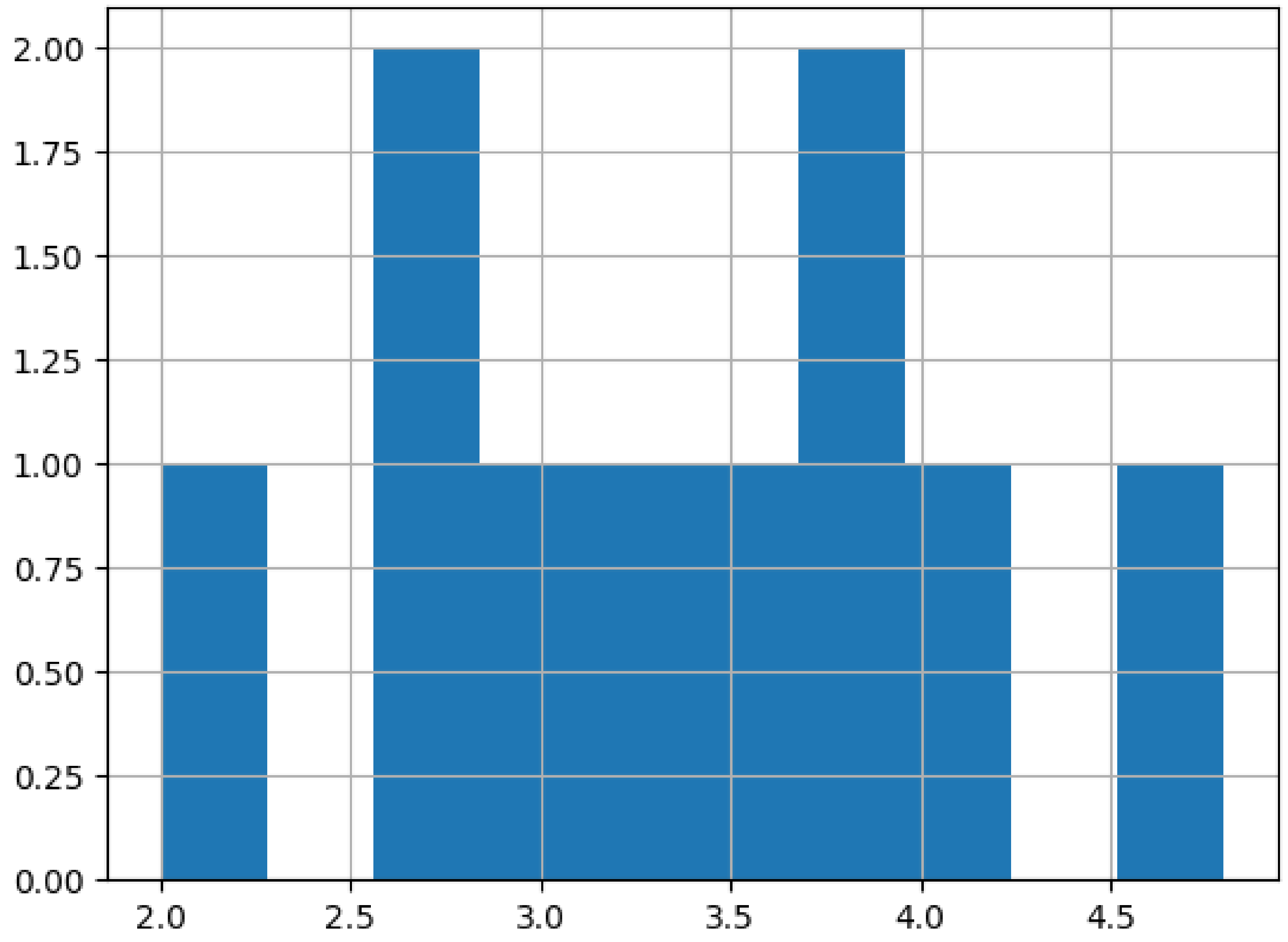
- Roll 5 times
- Take the mean

```
sample_means = []
for i in range(10):
  samp_5 = die.sample(5, replace=True)
  sample_means.append(np.mean(samp_5))
print(sample_means)
```

```
[3.8, 4.0, 3.8, 3.6, 3.2, 4.8, 2.6,
3.0, 2.6, 2.0]
```

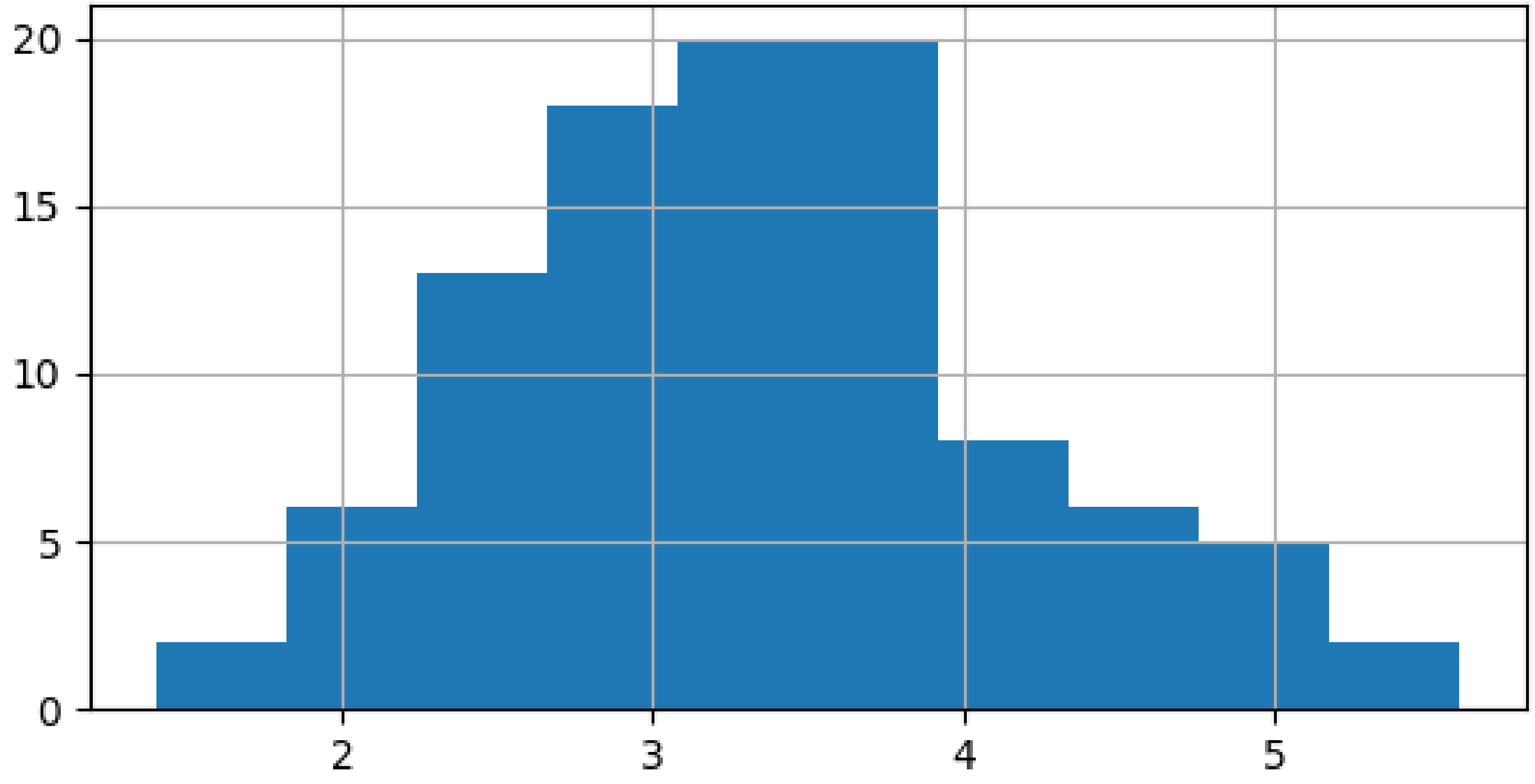
Sampling distributions

Sampling distribution of the sample mean



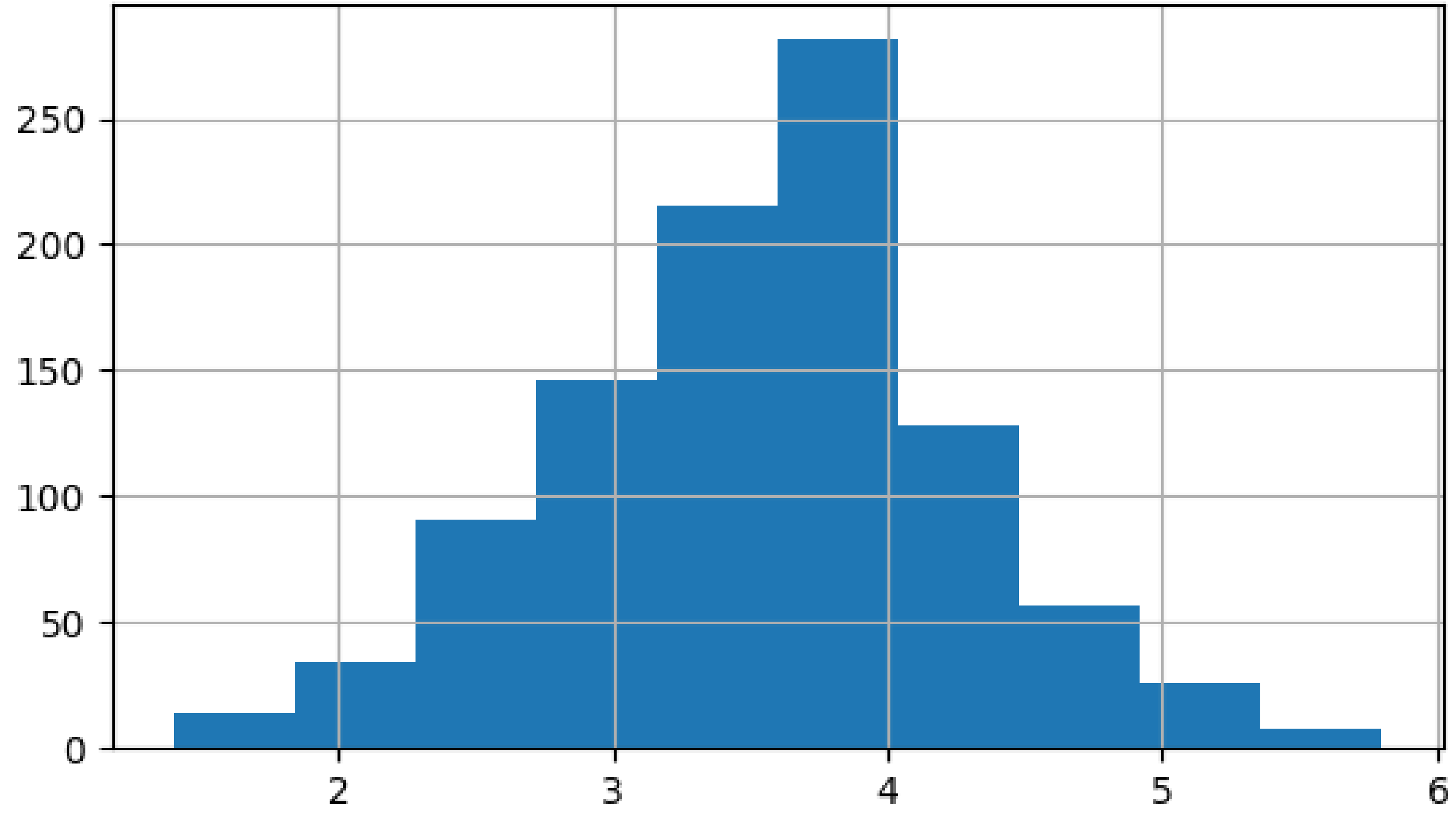
100 sample means

```
sample_means = []
for i in range(100):
    sample_means.append(np.mean(die.sample(5, replace=True)))
```



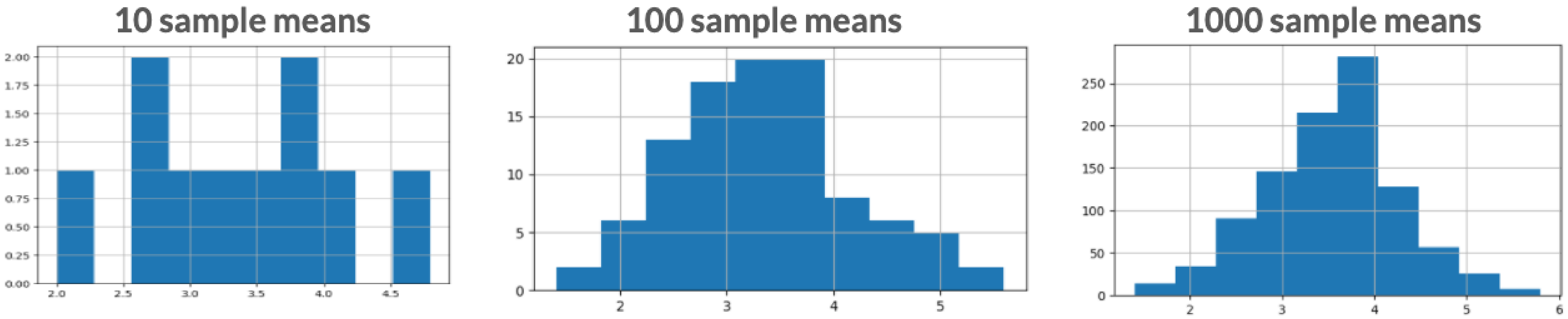
1000 sample means

```
sample_means = []
for i in range(1000):
    sample_means.append(np.mean(die.sample(5, replace=True)))
```



Central limit theorem

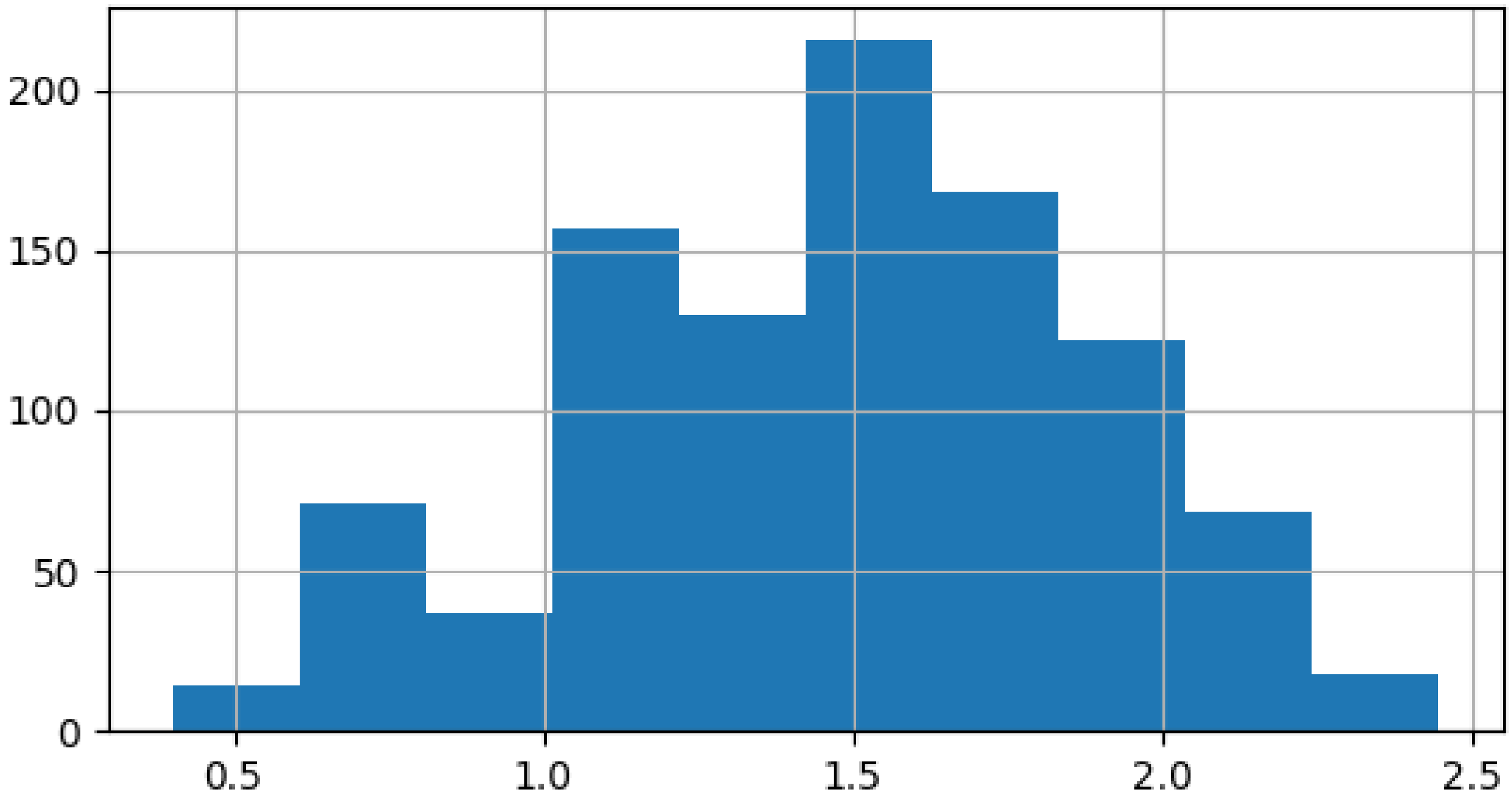
The sampling distribution of a statistic becomes closer to the normal distribution as the number of trials increases.



* *Samples should be random and independent*

Standard deviation and the CLT

```
sample_sds = []
for i in range(1000):
  sample_sds.append(np.std(die.sample(5, replace=True)))
```



Proportions and the CLT

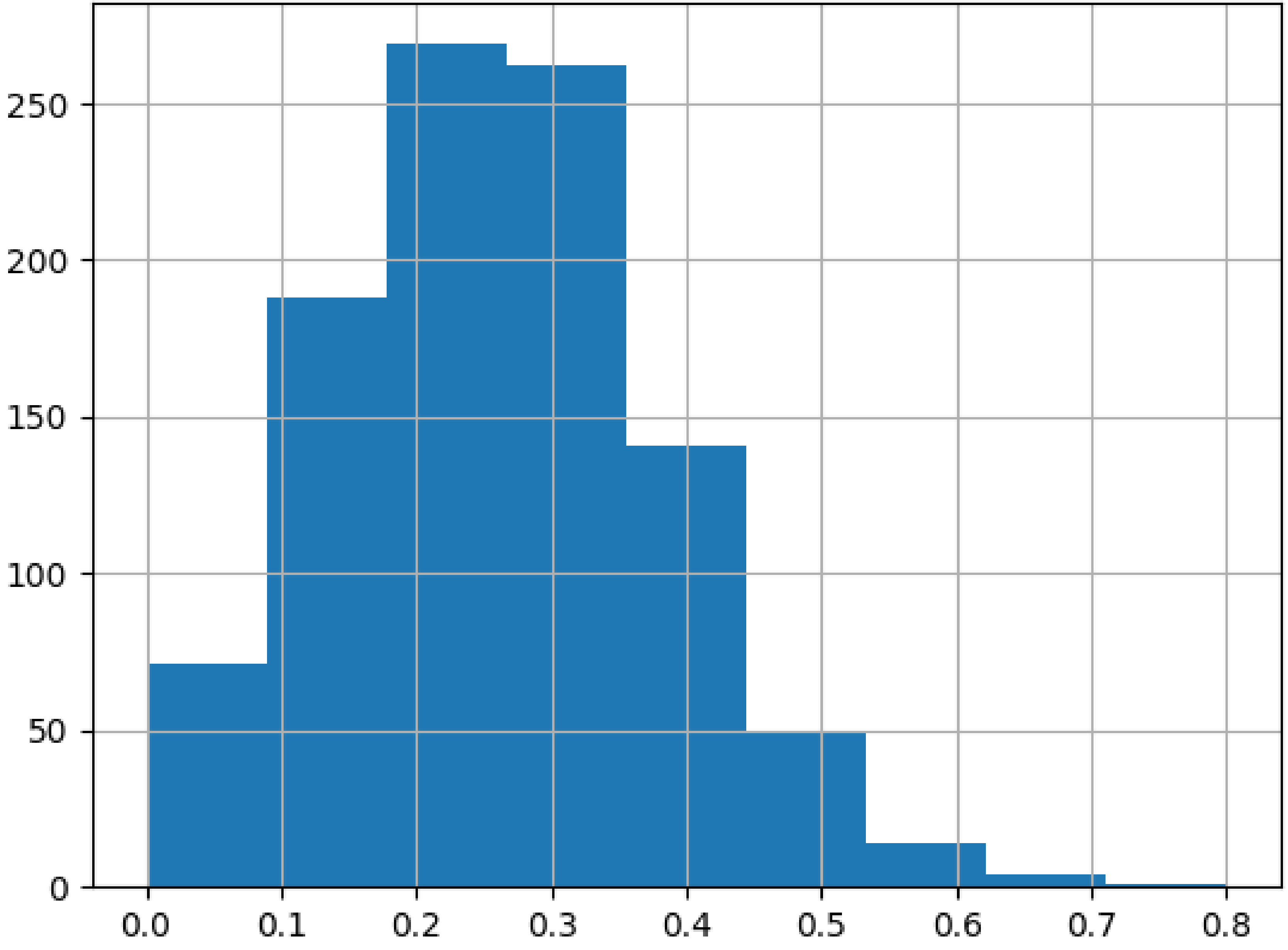
```
sales_team = pd.Series(["Amir", "Brian", "Claire", "Damian"])
sales_team.sample(10, replace=True)
```

```
array(['Claire', 'Damian', 'Brian', 'Damian', 'Damian', 'Amir', 'Amir', 'Amir',
       'Amir', 'Damian'], dtype=object)
```

```
sales_team.sample(10, replace=True)
```

```
array(['Brian', 'Amir', 'Brian', 'Claire', 'Brian', 'Damian', 'Claire', 'Brian',
       'Claire', 'Claire'], dtype=object)
```

Sampling distribution of proportion



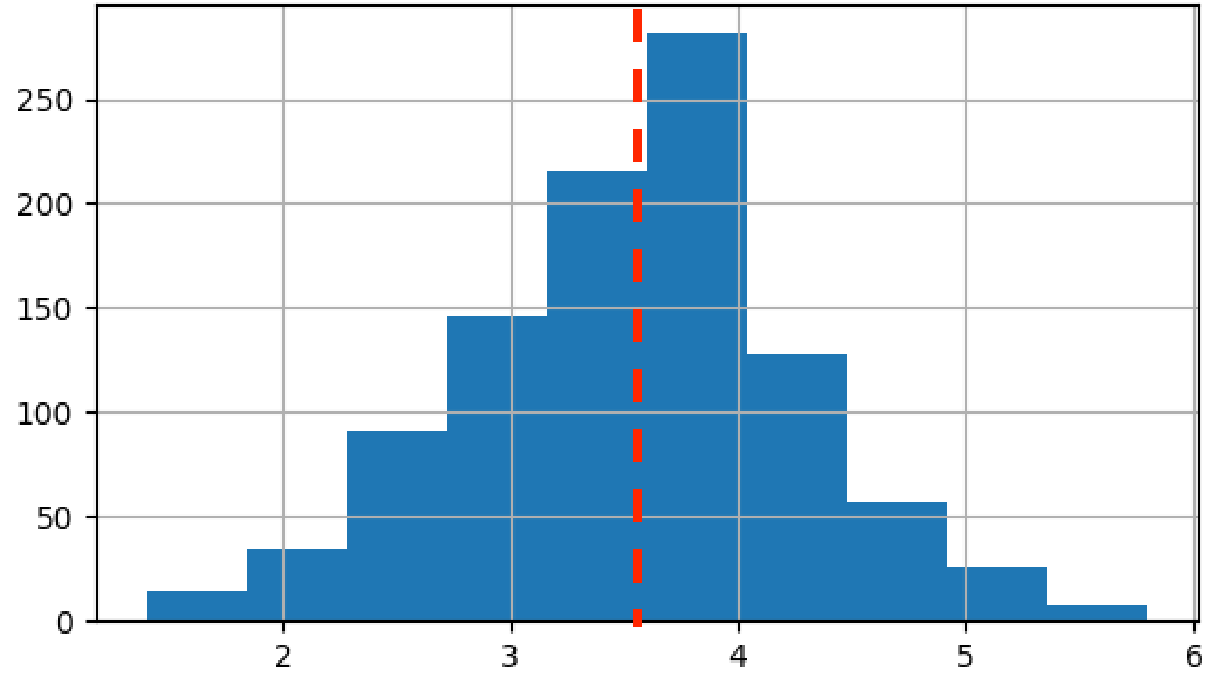
Mean of sampling distribution

```
# Estimate expected value of die
np.mean(sample_means)
```

```
3.48
```

```
# Estimate proportion of "Claire"s
np.mean(sample_props)
```

```
0.26
```



- Estimate characteristics of unknown underlying distribution
- More easily estimate characteristics of large populations

Let's practice!

The Poisson distribution

INTRODUCTION TO STATISTICS IN PYTHON

Maggie Matsui
Content Developer, DataCamp

Poisson processes

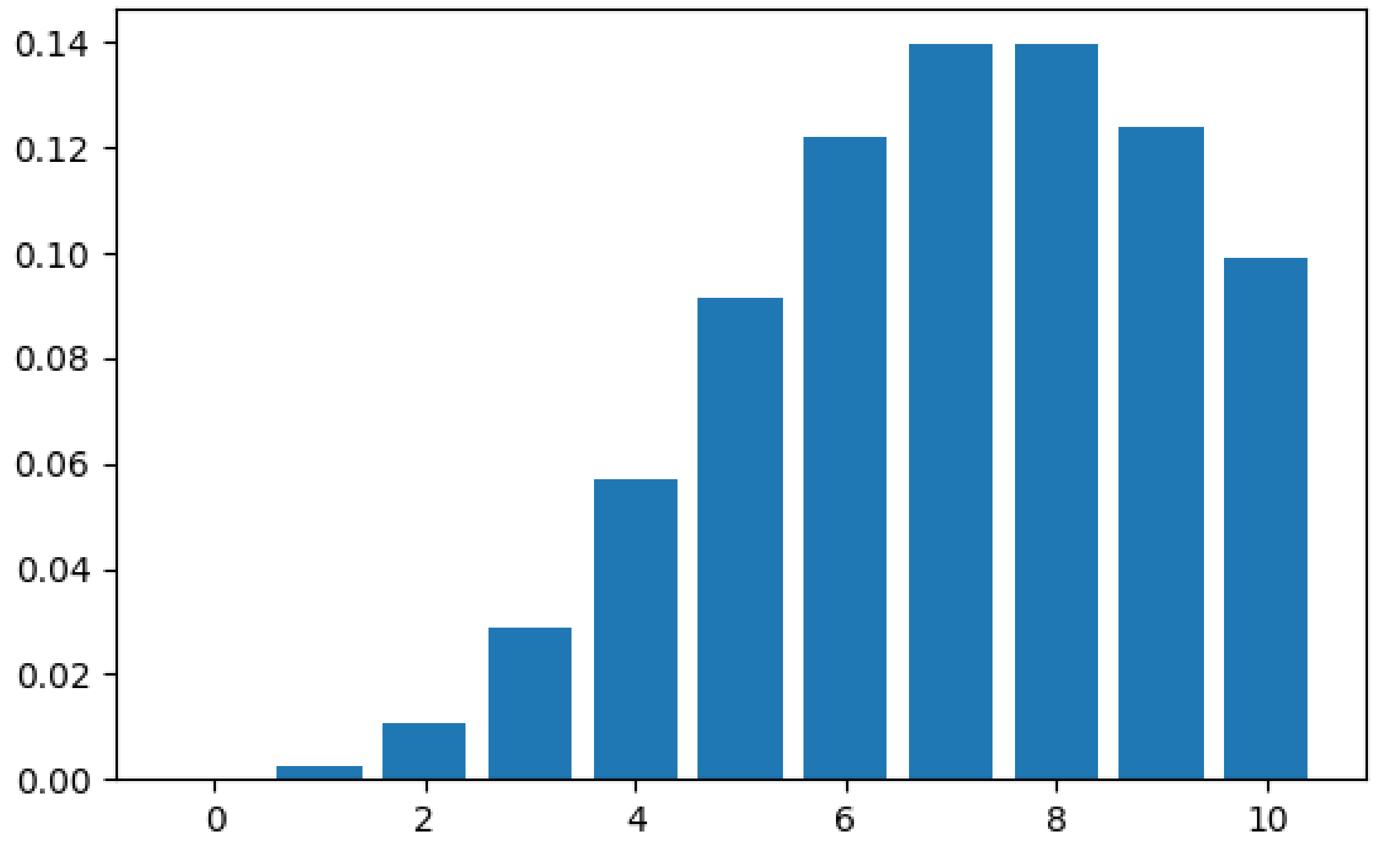
- Events appear to happen at a certain rate, but completely at random
- Examples
  - Number of animals adopted from an animal shelter per week
  - Number of people arriving at a restaurant per hour
  - Number of earthquakes in California per year
- Time unit is irrelevant, as long as you use the same unit when talking about the same situation

Poisson distribution

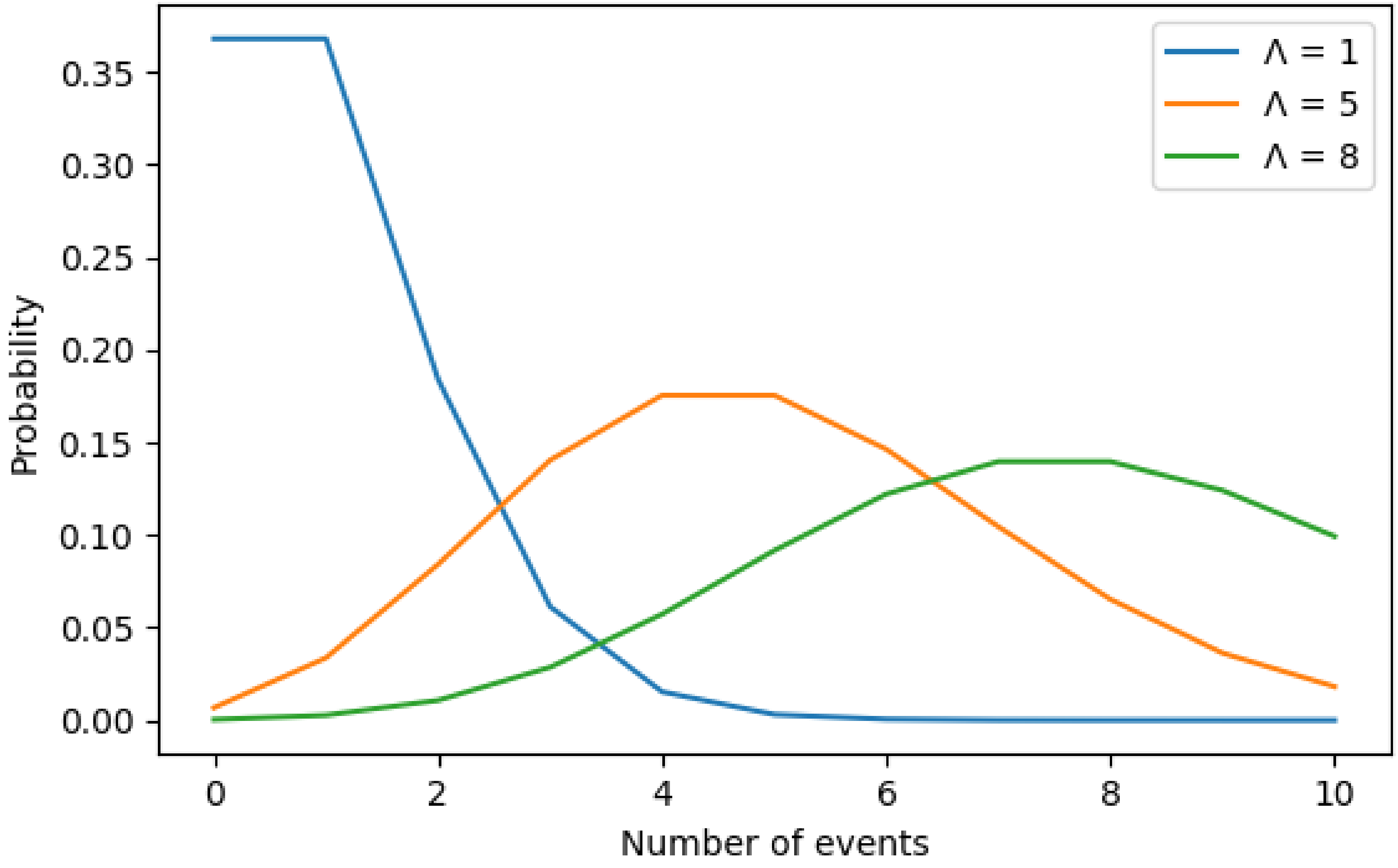
- Probability of some # of events occurring over a fixed period of time
- Examples
  - Probability of ≥ 5 animals adopted from an animal shelter per week
  - Probability of 12 people arriving at a restaurant per hour
  - Probability of < 20 earthquakes in California per year

Lambda (λ)

- λ = average number of events per time interval
  - Average number of adoptions per week = 8



Lambda is the distribution's peak



Probability of a single value

If the average number of adoptions per week is 8, what is $P(\# \text{ adoptions in a week} = 5)$?

```python
from scipy.stats import poisson
poisson.pmf(5, 8)
```

```
0.09160366
```

Probability of less than or equal to

If the average number of adoptions per week is 8, what is $P(\#\text{ adoptions in a week} \leq 5)$?

```
from scipy.stats import poisson
poisson.cdf(5, 8)
```

```
0.1912361
```

Probability of greater than

If the average number of adoptions per week is 8, what is $P(\# \text{ adoptions in a week} > 5)$?

```
1 - poisson.cdf(5, 8)
```

```
0.8087639
```

If the average number of adoptions per week is **10**, what is $P(\# \text{ adoptions in a week} > 5)$?

```
1 - poisson.cdf(5, 10)
```

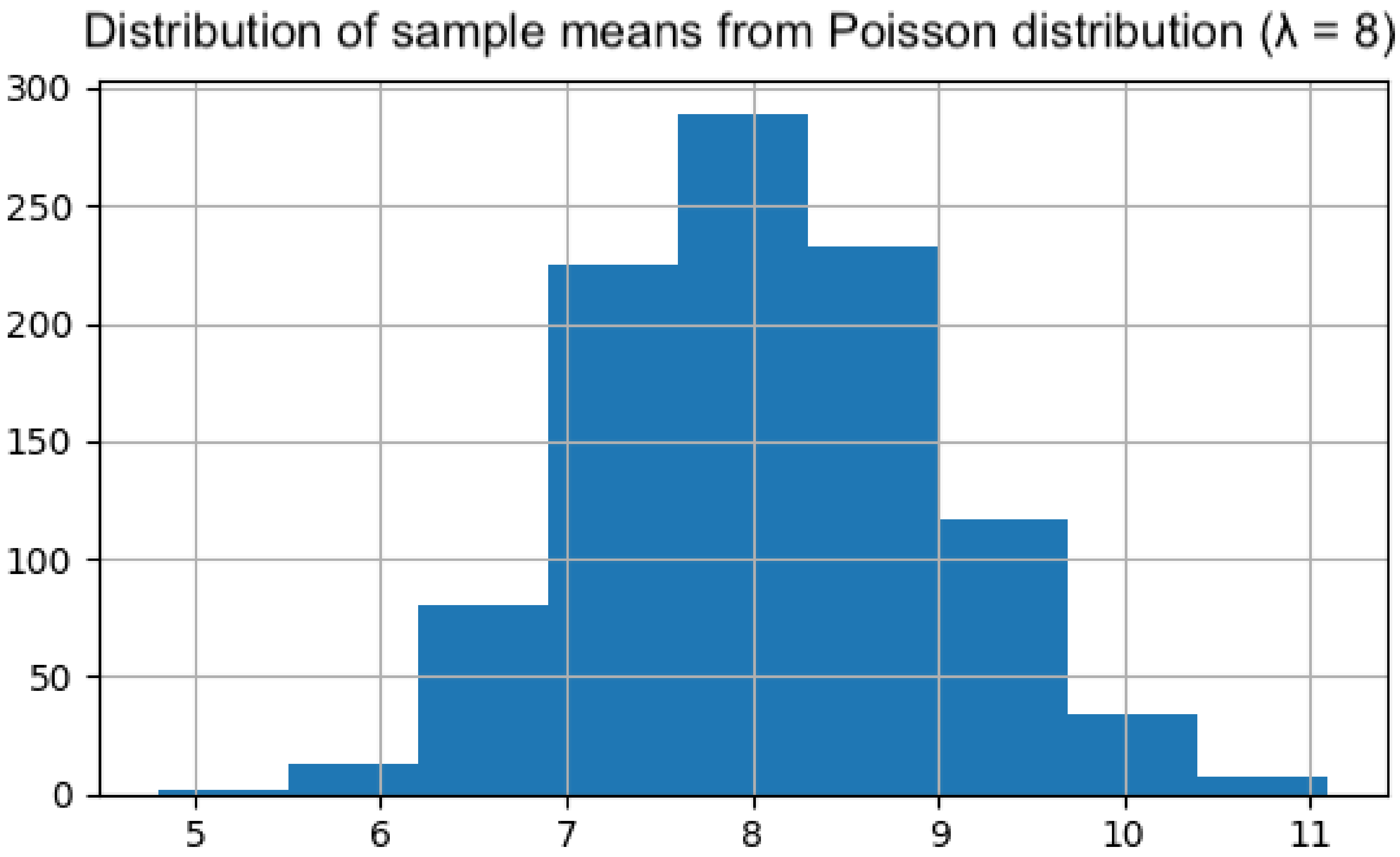
```
0.932914
```

Sampling from a Poisson distribution

```
from scipy.stats import poisson
poisson.rvs(8, size=10)
```

```
array([ 9,  9,  8,  7, 11,  3, 10,  6,  8, 14])
```

The CLT still applies!



Let's practice!

INTRODUCTION TO STATISTICS IN PYTHON

More probability distributions

INTRODUCTION TO STATISTICS IN PYTHON

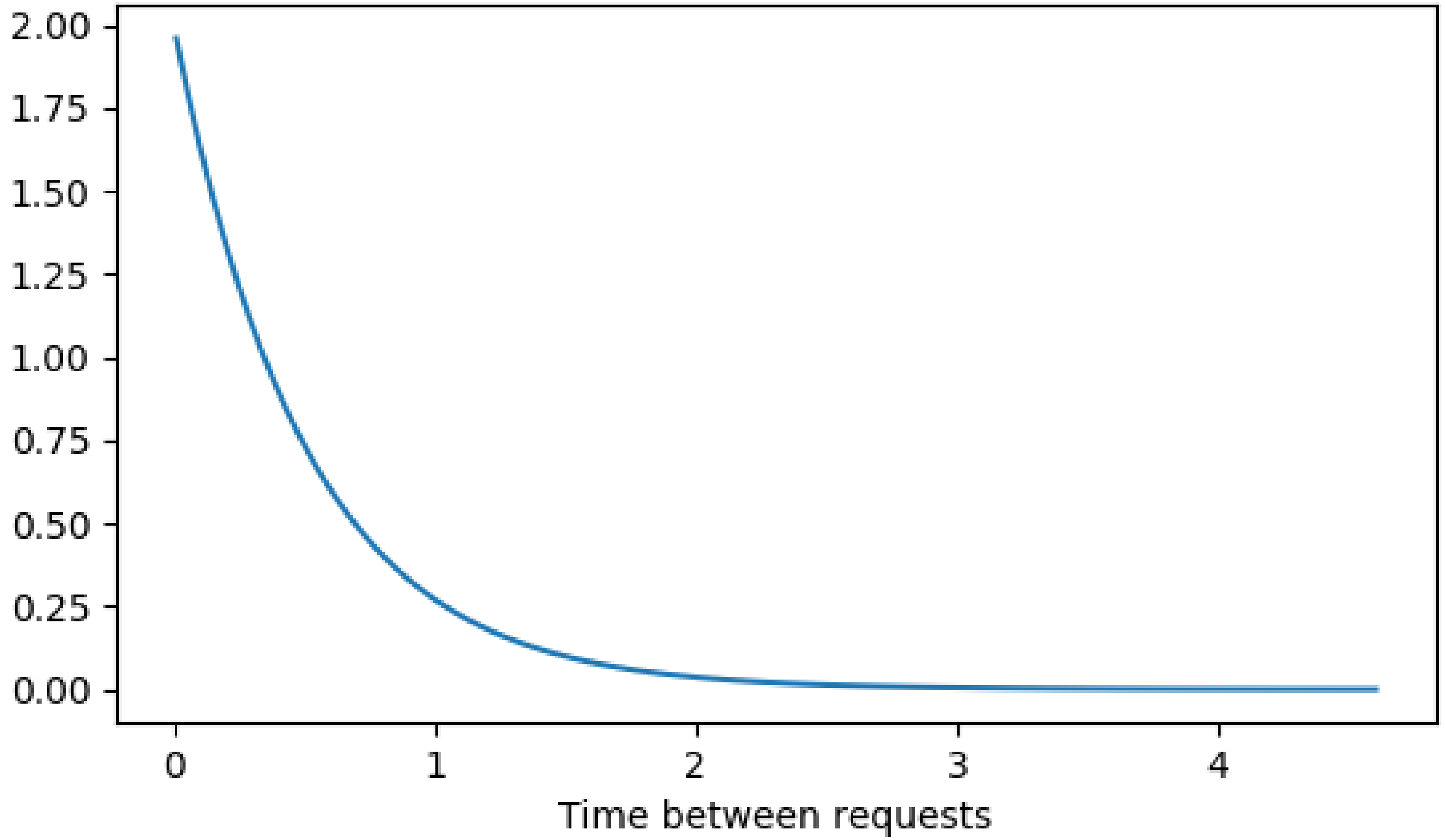
Maggie Matsui
Content Developer, DataCamp

Exponential distribution

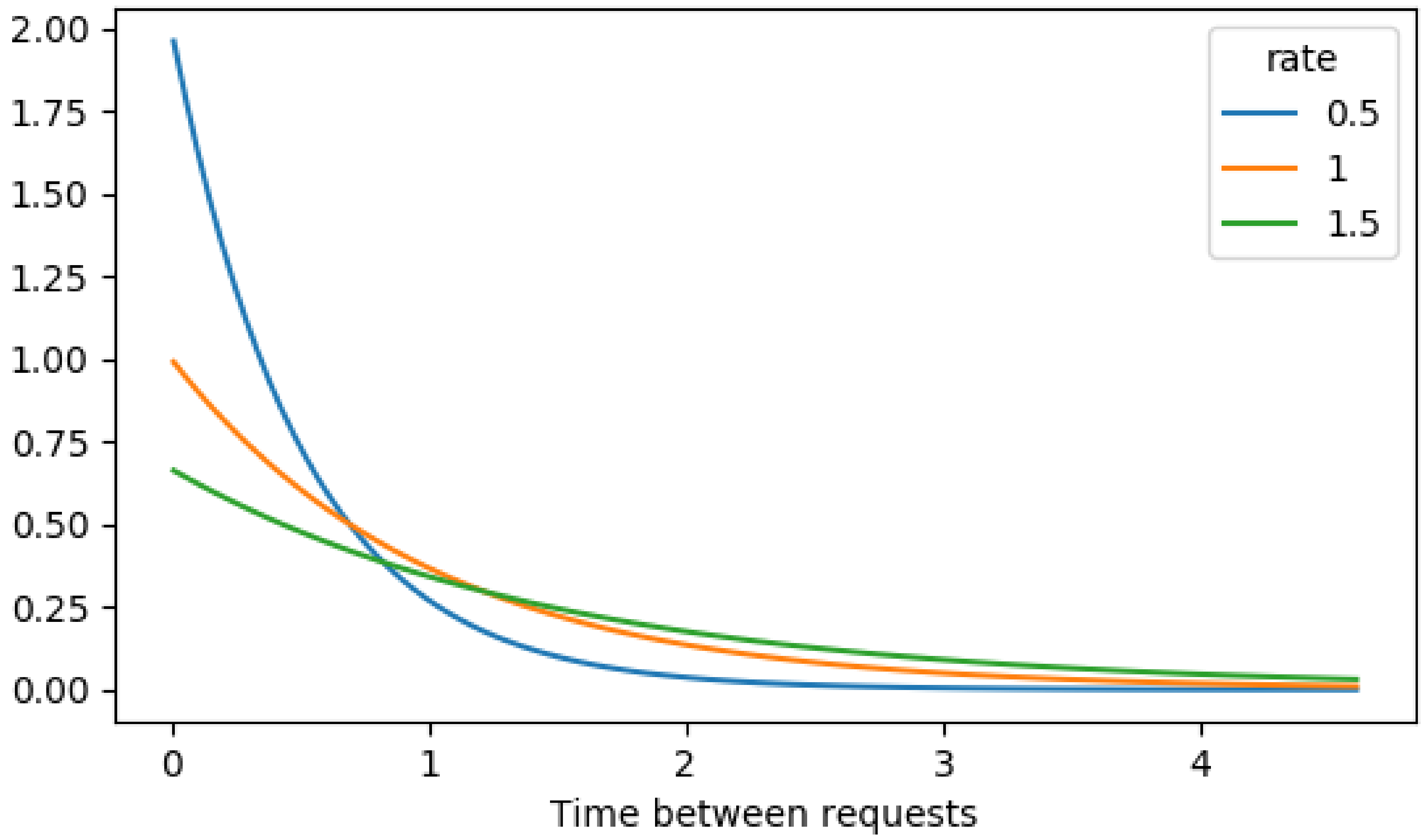
- Probability of time between Poisson events
- Examples
  - Probability of > 1 day between adoptions
  - Probability of < 10 minutes between restaurant arrivals
  - Probability of 6-8 months between earthquakes
- Also uses lambda (rate)
- Continuous (time)

Customer service requests

- On average, one customer service ticket is created every 2 minutes
  - λ = 0.5 customer service tickets created each minute



Lambda in exponential distribution



Expected value of exponential distribution

In terms of rate (Poisson):

- $\lambda = 0.5 \text{ requests}$ per minute

In terms of time between events (exponential):

- $1/\lambda$ = 1 request per 2 minutes
- $1/0.5 = 2$

How long until a new request is created?

```
from scipy.stats import expon
```

- `scale` $= 1/\lambda = 1/0.5 = 2$

$P(\text{wait} < 1 \text{ min}) =$

```
expon.cdf(1, scale=2)
```

```
0.3934693402873666
```

$P(\text{wait} > 4 \text{ min}) =$

```
1- expon.cdf(4, scale=2)
```

```
0.1353352832366127
```

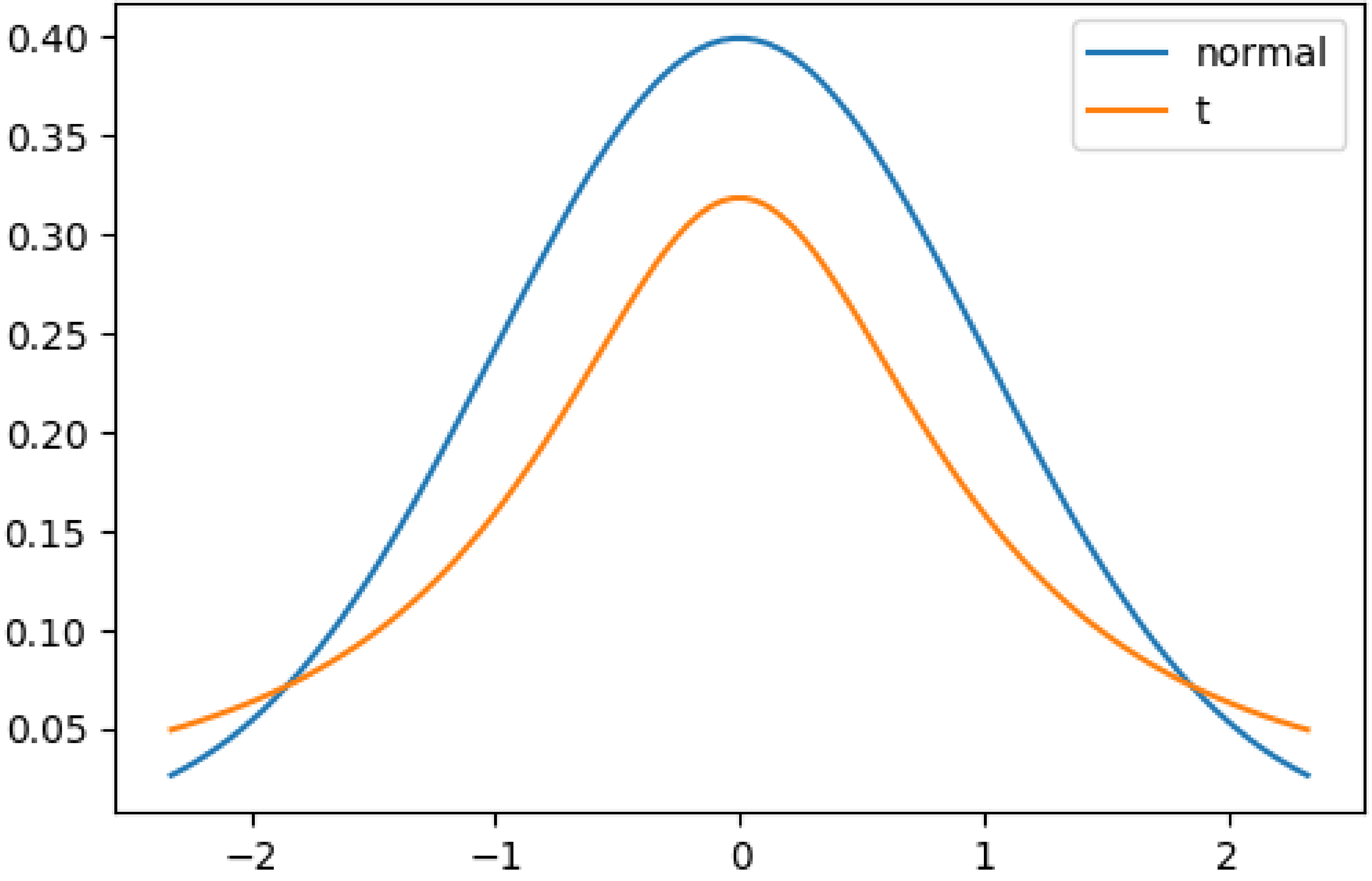
$P(1 \text{ min} < \text{wait} < 4 \text{ min}) =$

```
expon.cdf(4, scale=2) - expon.cdf(1, scale=2)
```

```
0.4711953764760207
```

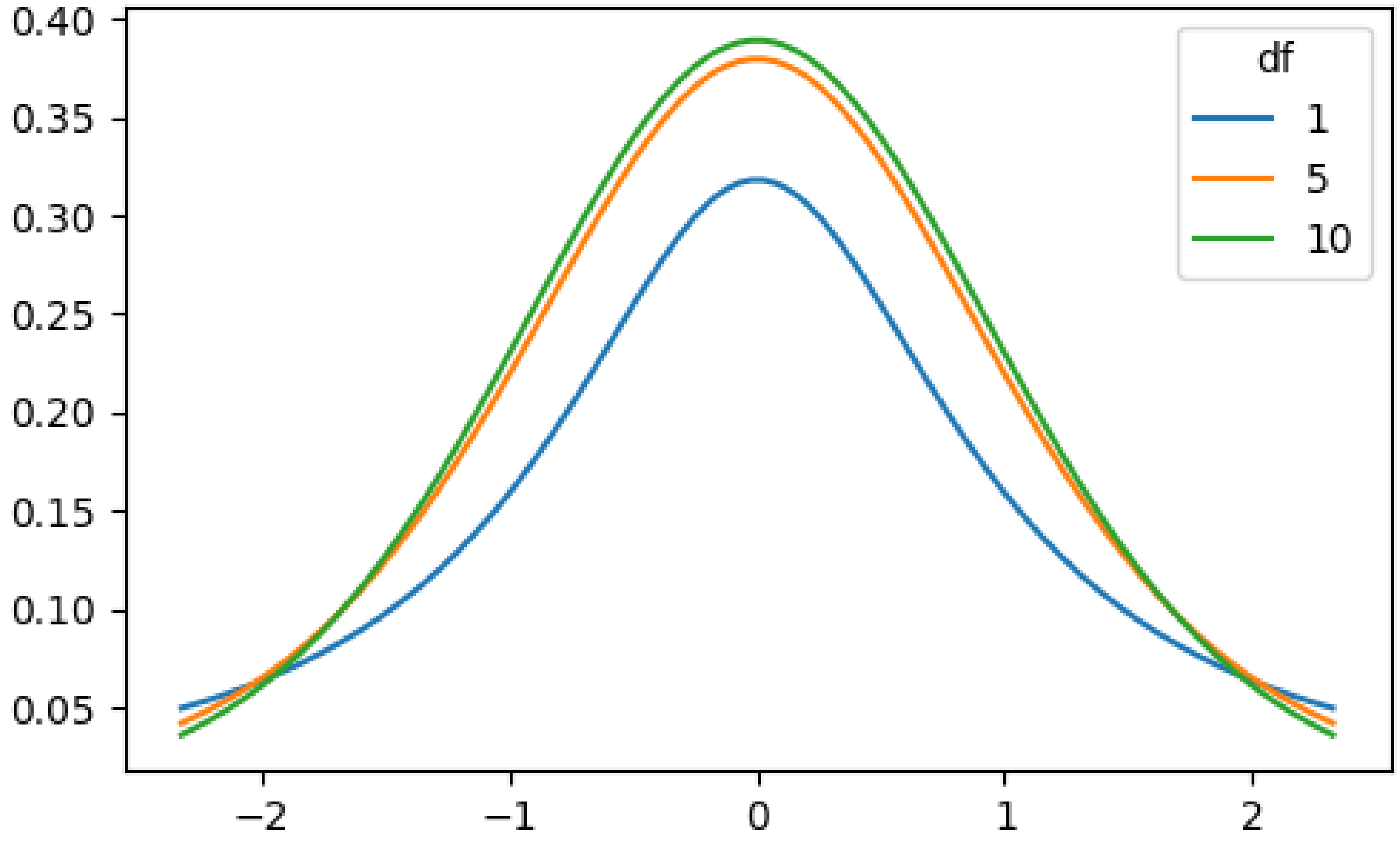
(Student's) t-distribution

- Similar shape as the normal distribution



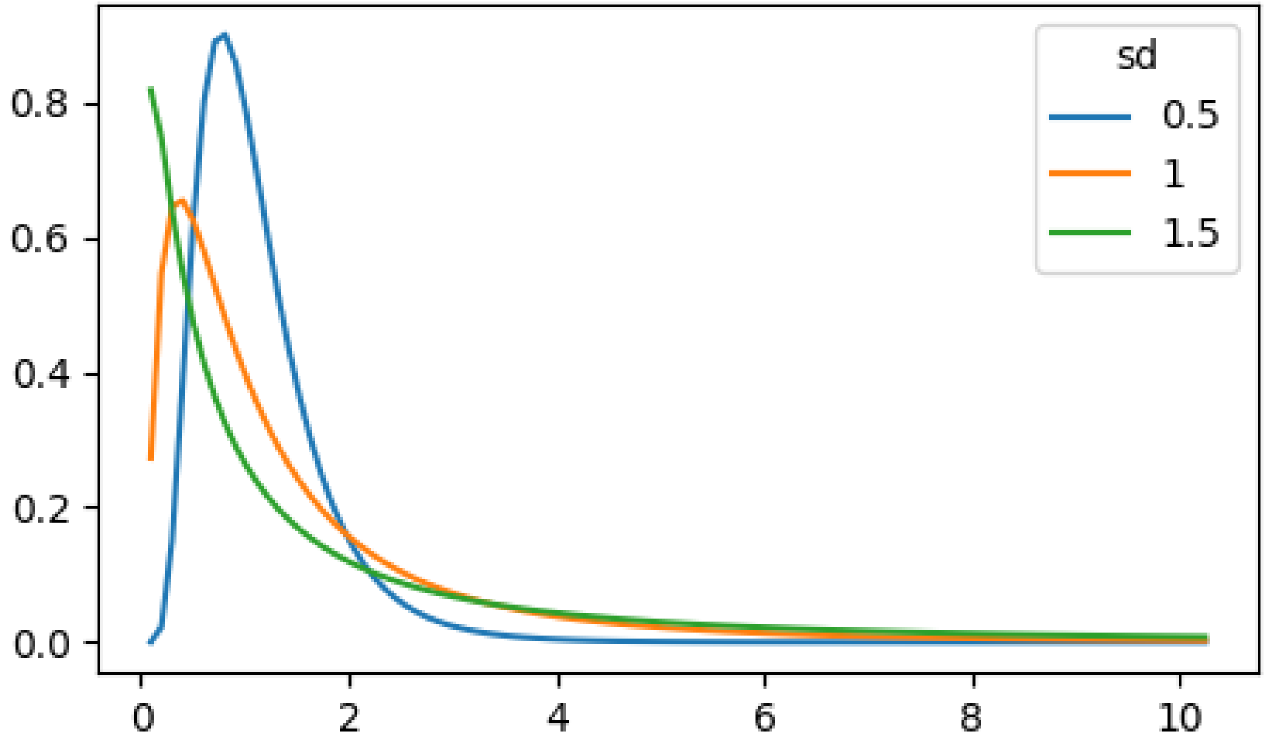
Degrees of freedom

- Has parameter degrees of freedom (df) which affects the thickness of the tails
  - Lower df = thicker tails, higher standard deviation
  - Higher df = closer to normal distribution



Log-normal distribution

- Variable whose logarithm is normally distributed
- Examples:
  - Length of chess games
  - Adult blood pressure
  - Number of hospitalizations in the 2003 SARS outbreak



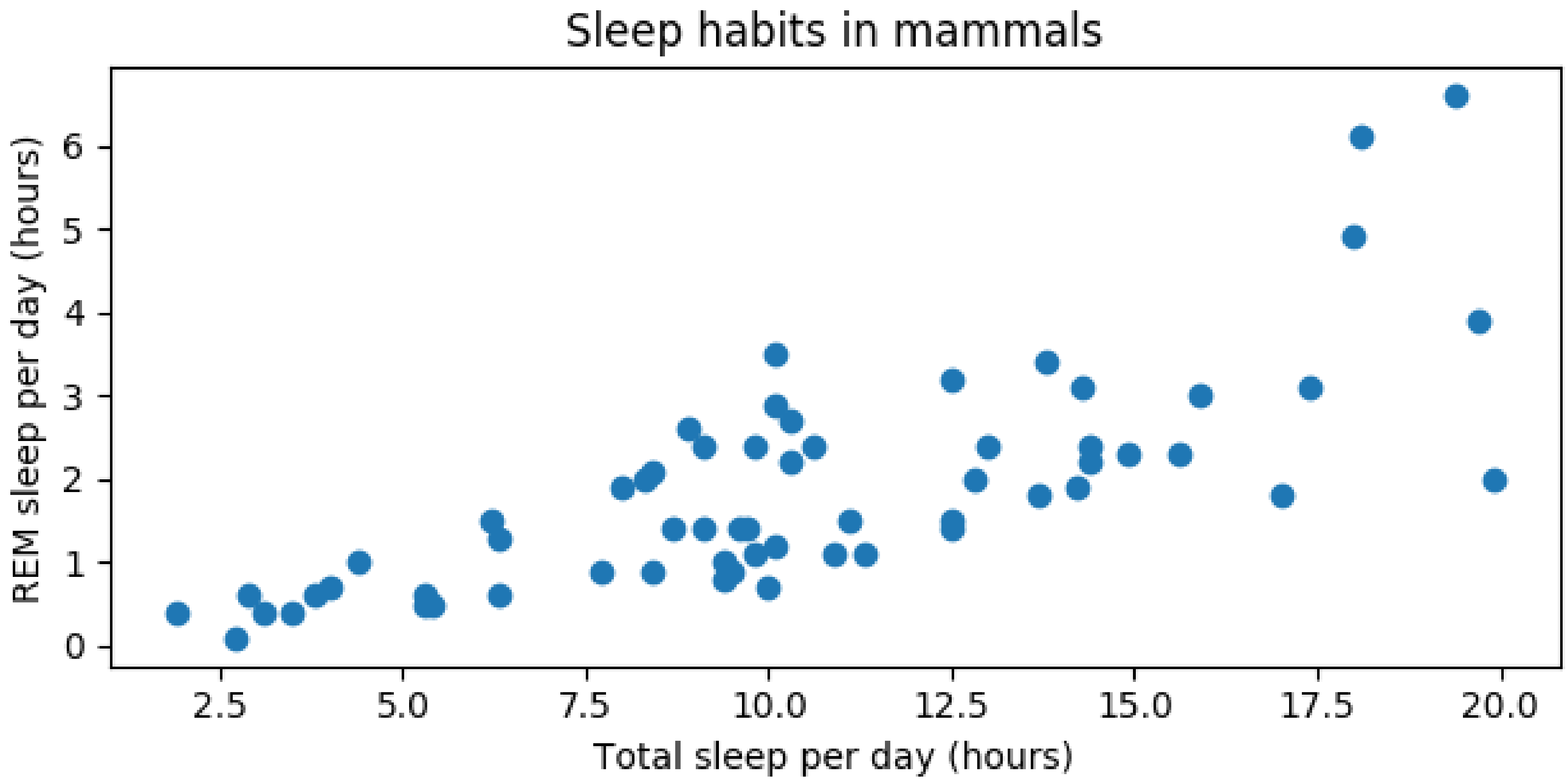
Let's practice!

Correlation

INTRODUCTION TO STATISTICS IN PYTHON

Maggie Matsui
Content Developer, DataCamp

Relationships between two variables



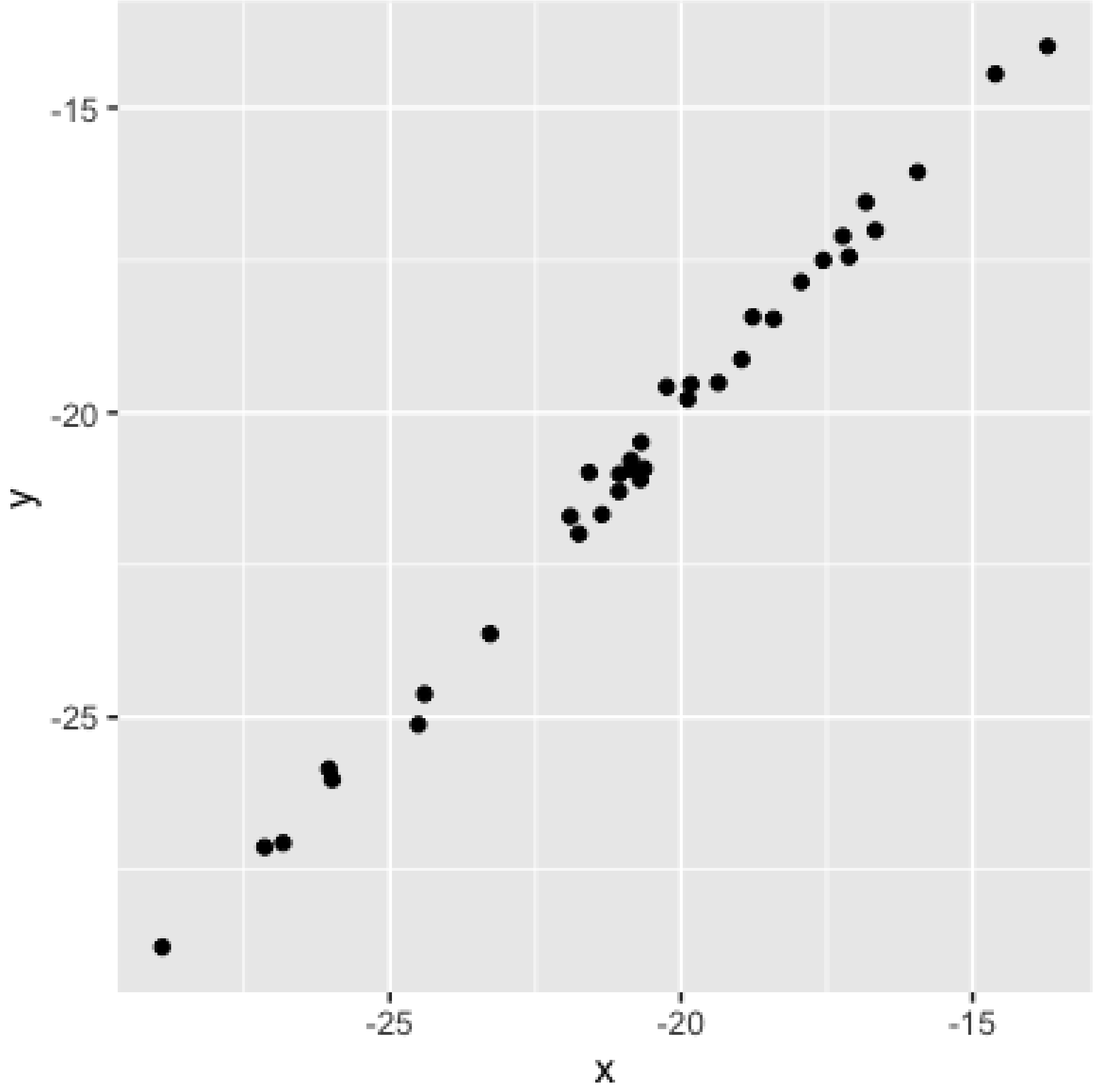
- x = explanatory/independent variable
- y = response/dependent variable

Correlation coefficient

- Quantifies the linear relationship between two variables
- Number between -1 and 1
- Magnitude corresponds to strength of relationship
- Sign (+ or -) corresponds to direction of relationship

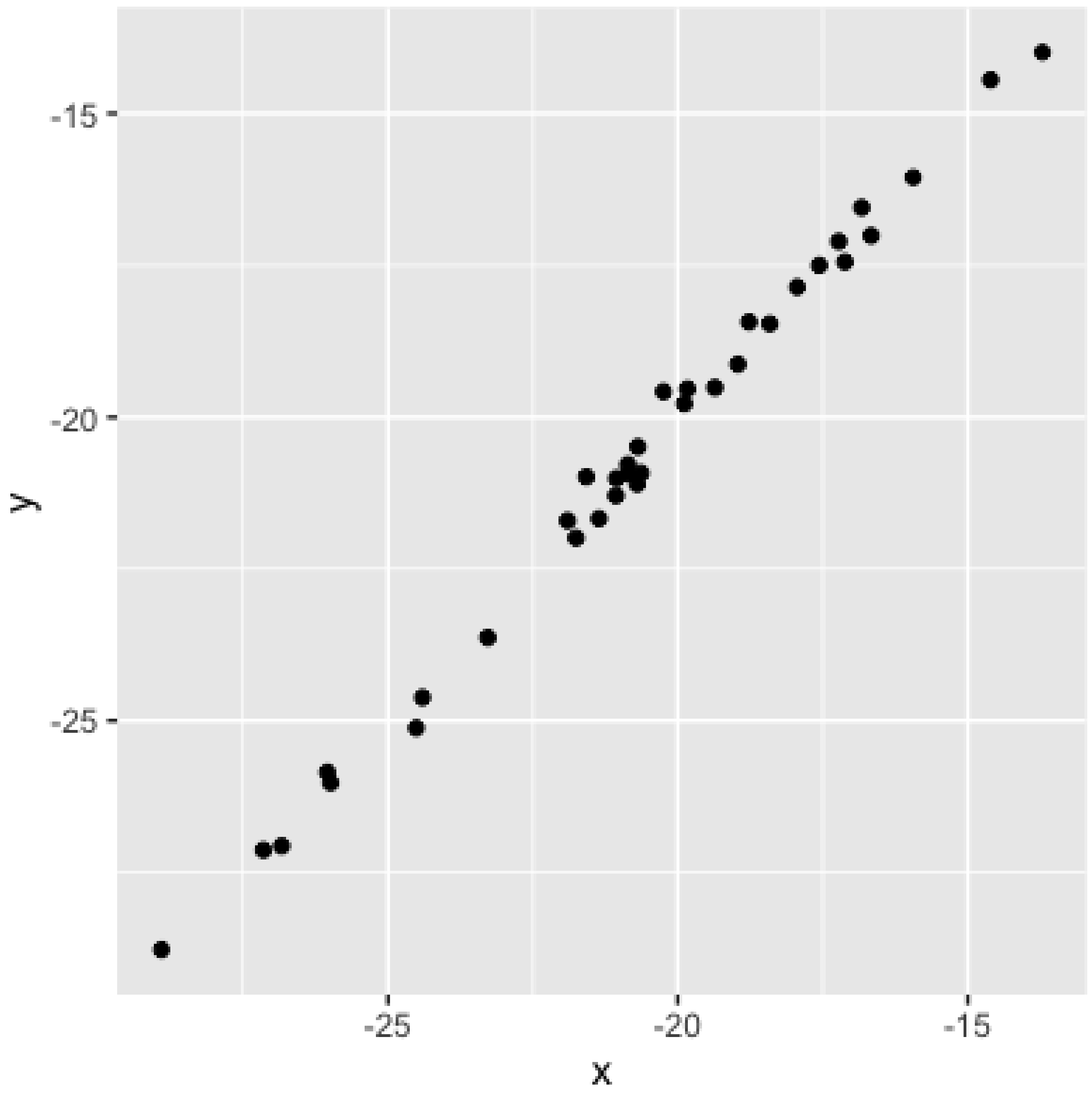
Magnitude = strength of relationship

0.99 (very strong relationship)

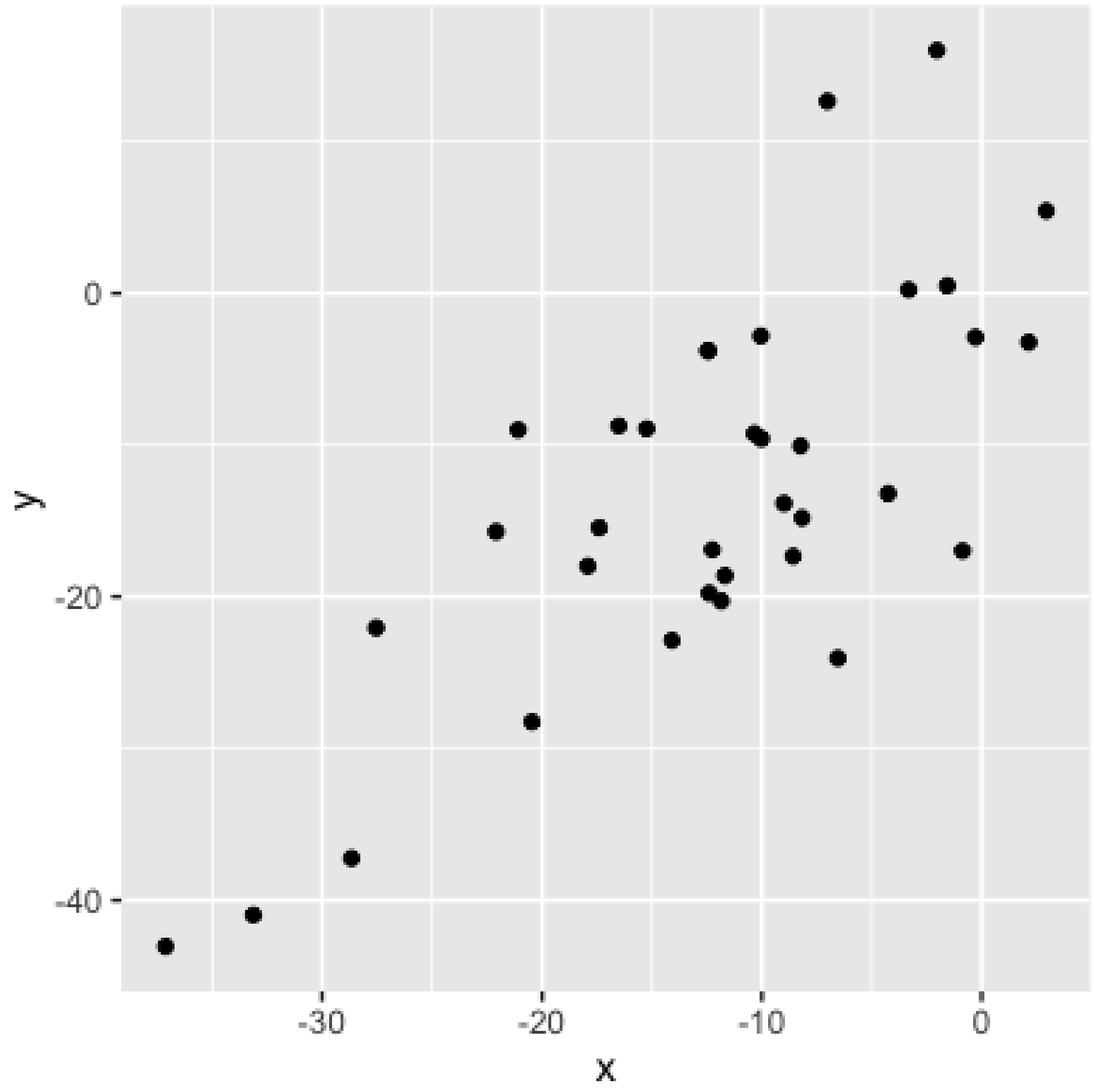


Magnitude = strength of relationship

0.99 (very strong relationship)

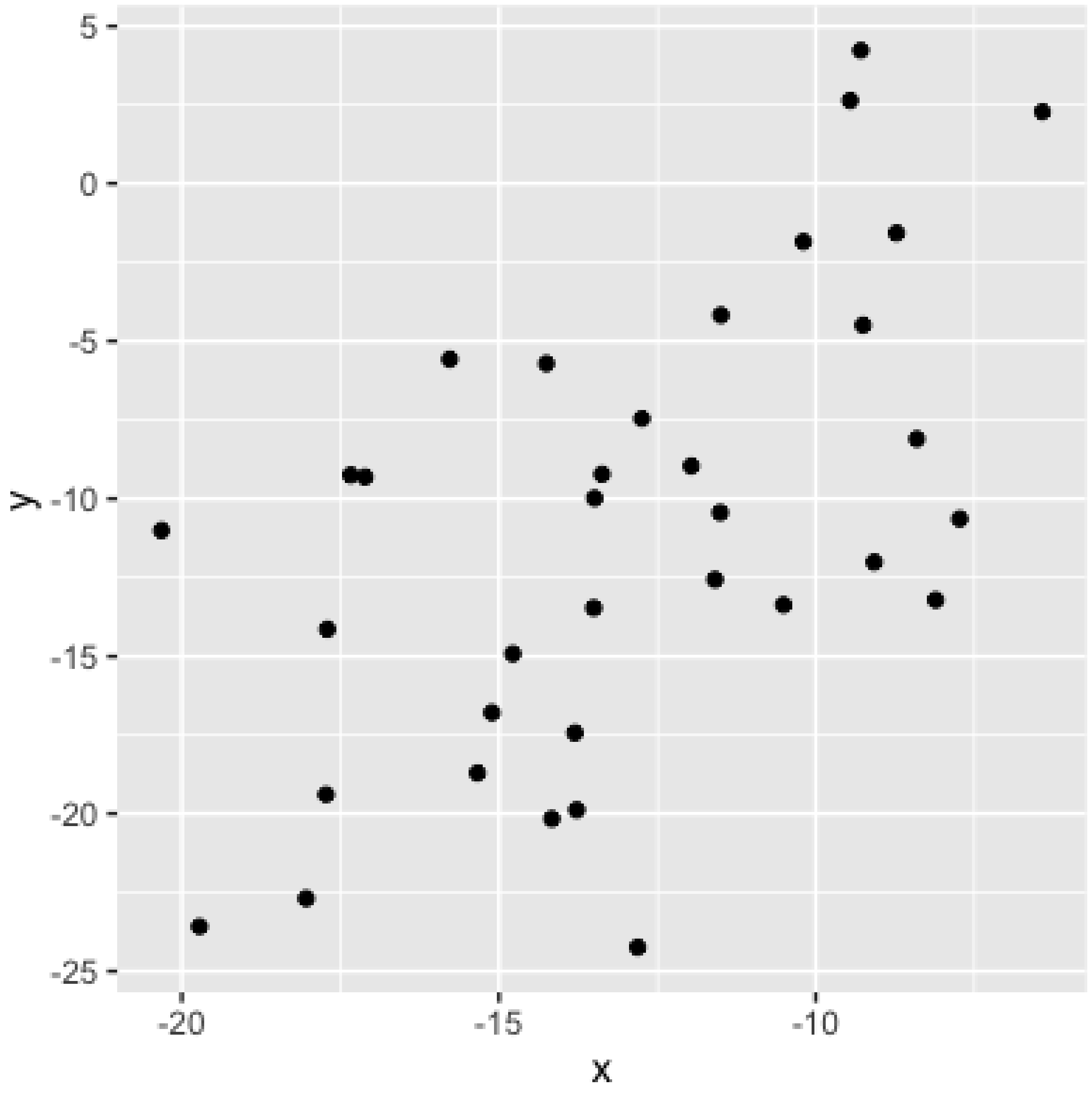


0.75 (strong relationship)



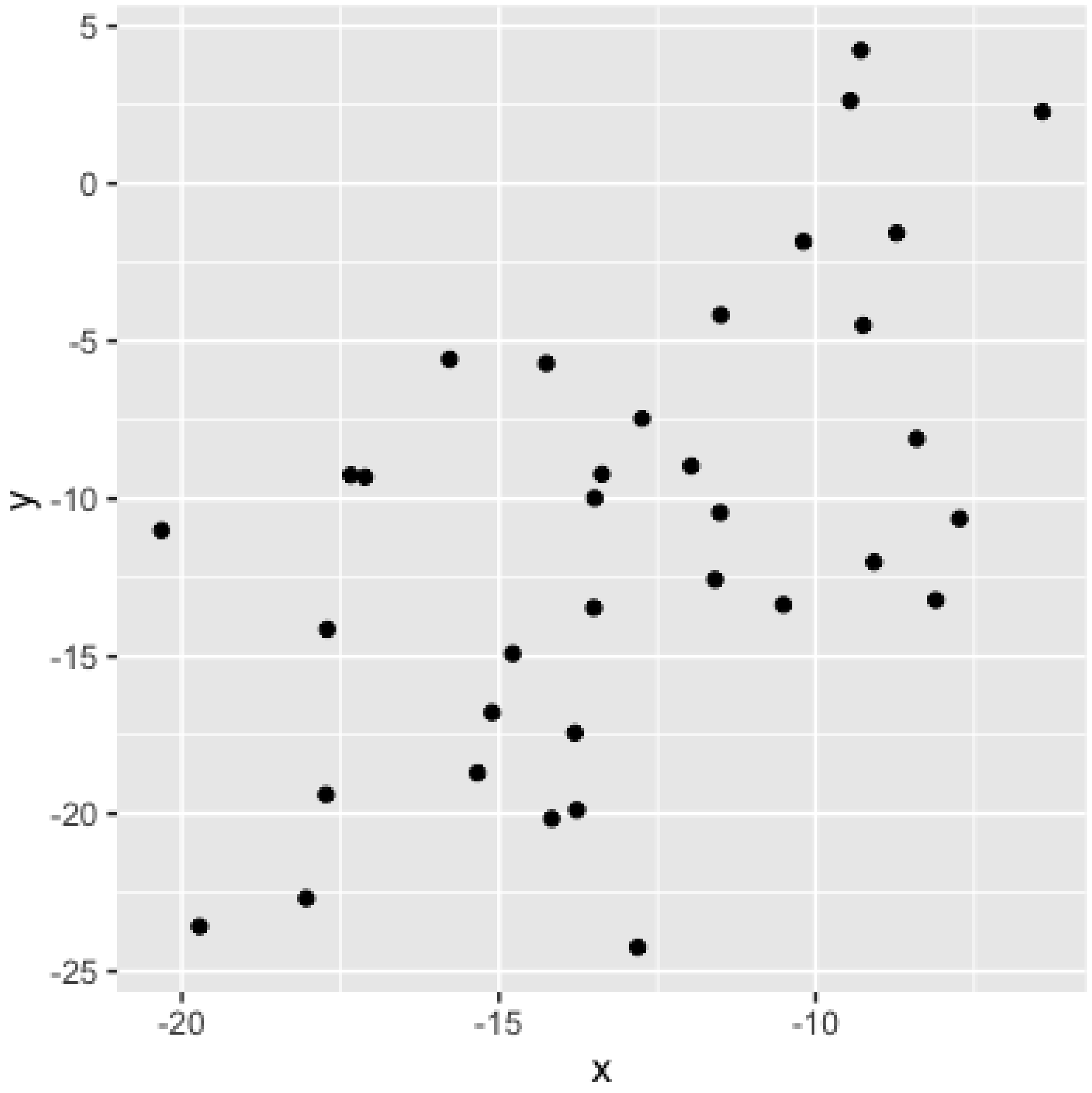
Magnitude = strength of relationship

0.56 (moderate relationship)

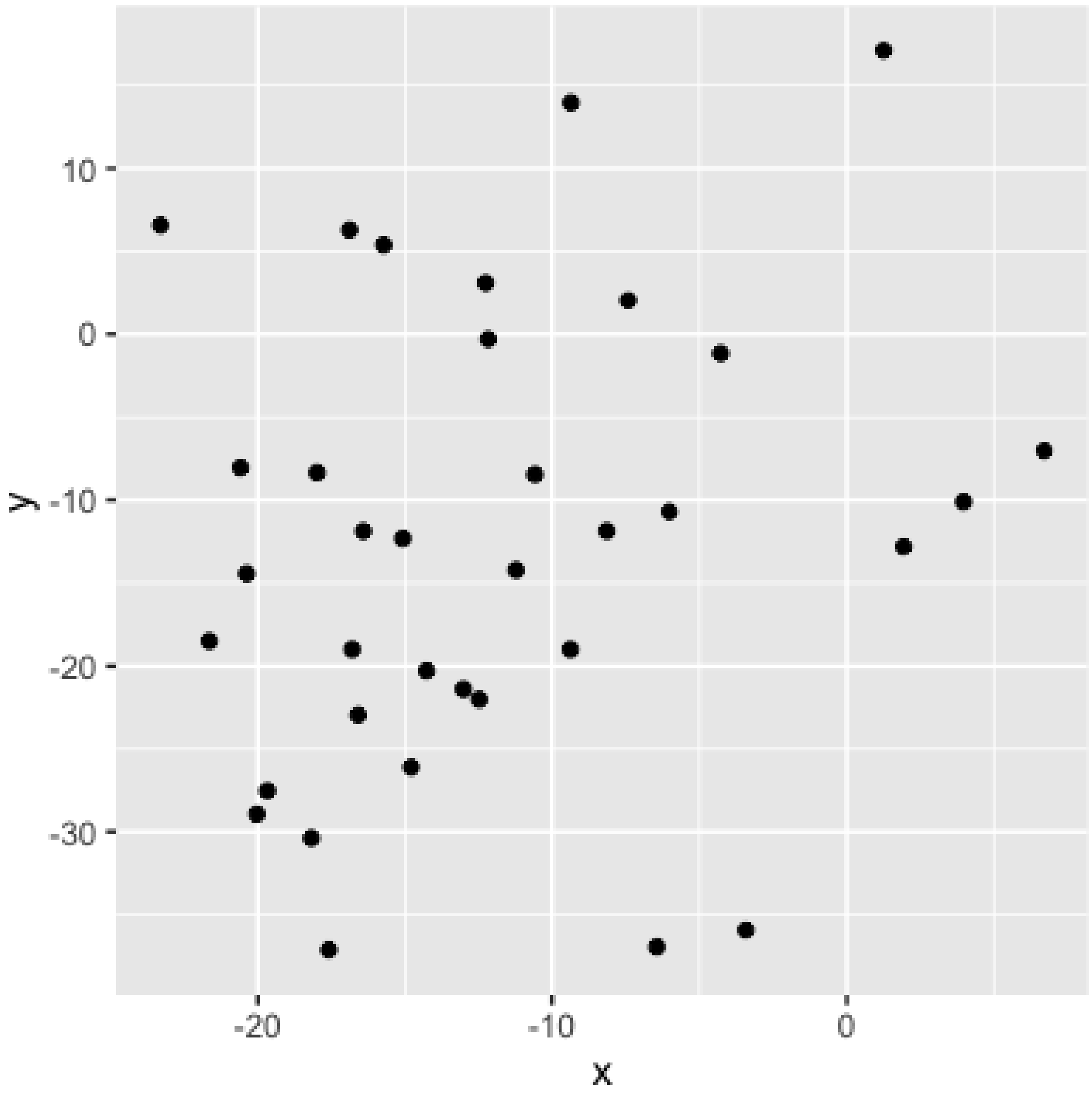


Magnitude = strength of relationship

0.56 (moderate relationship)

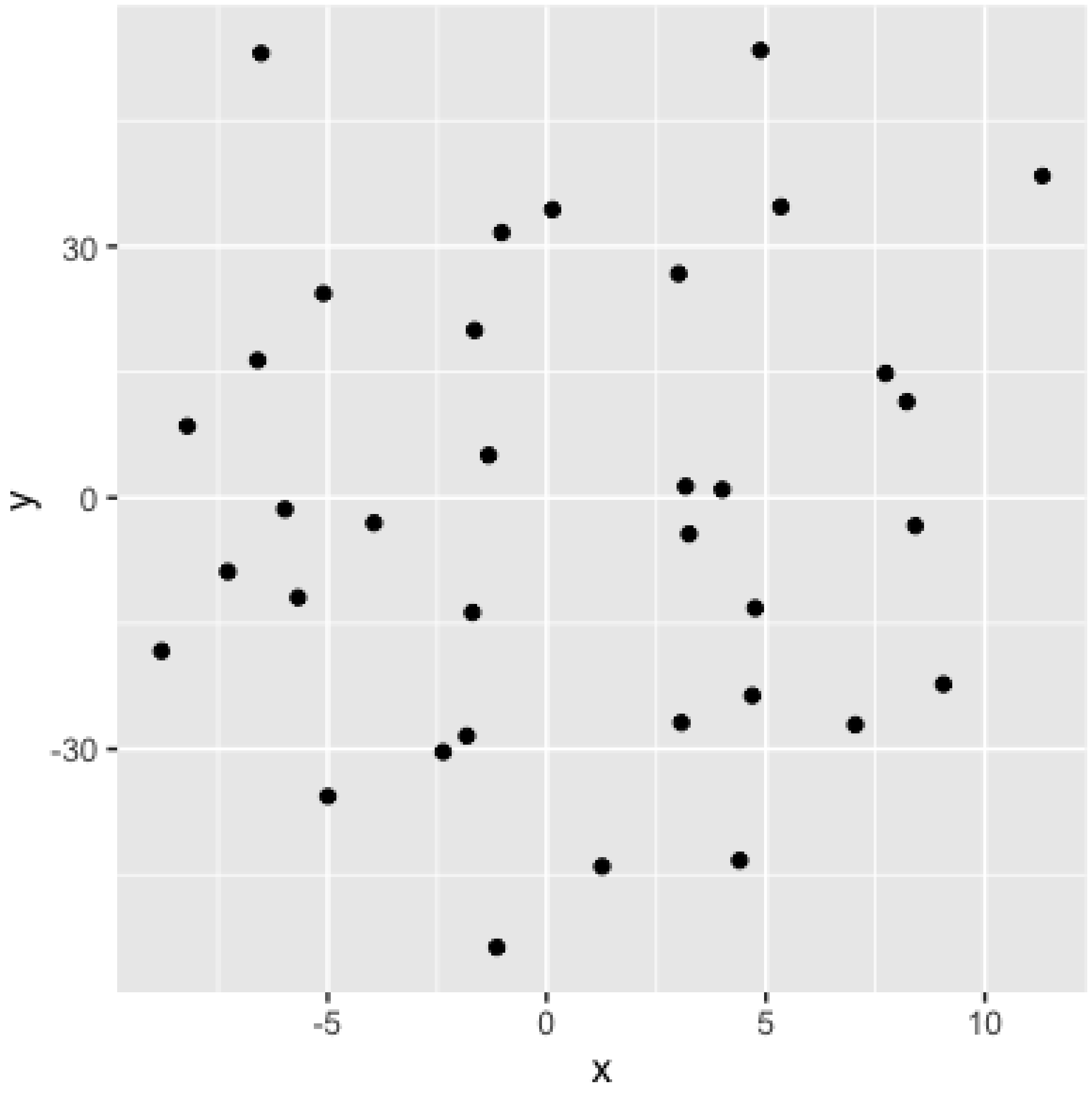


0.21 (weak relationship)



Magnitude = strength of relationship

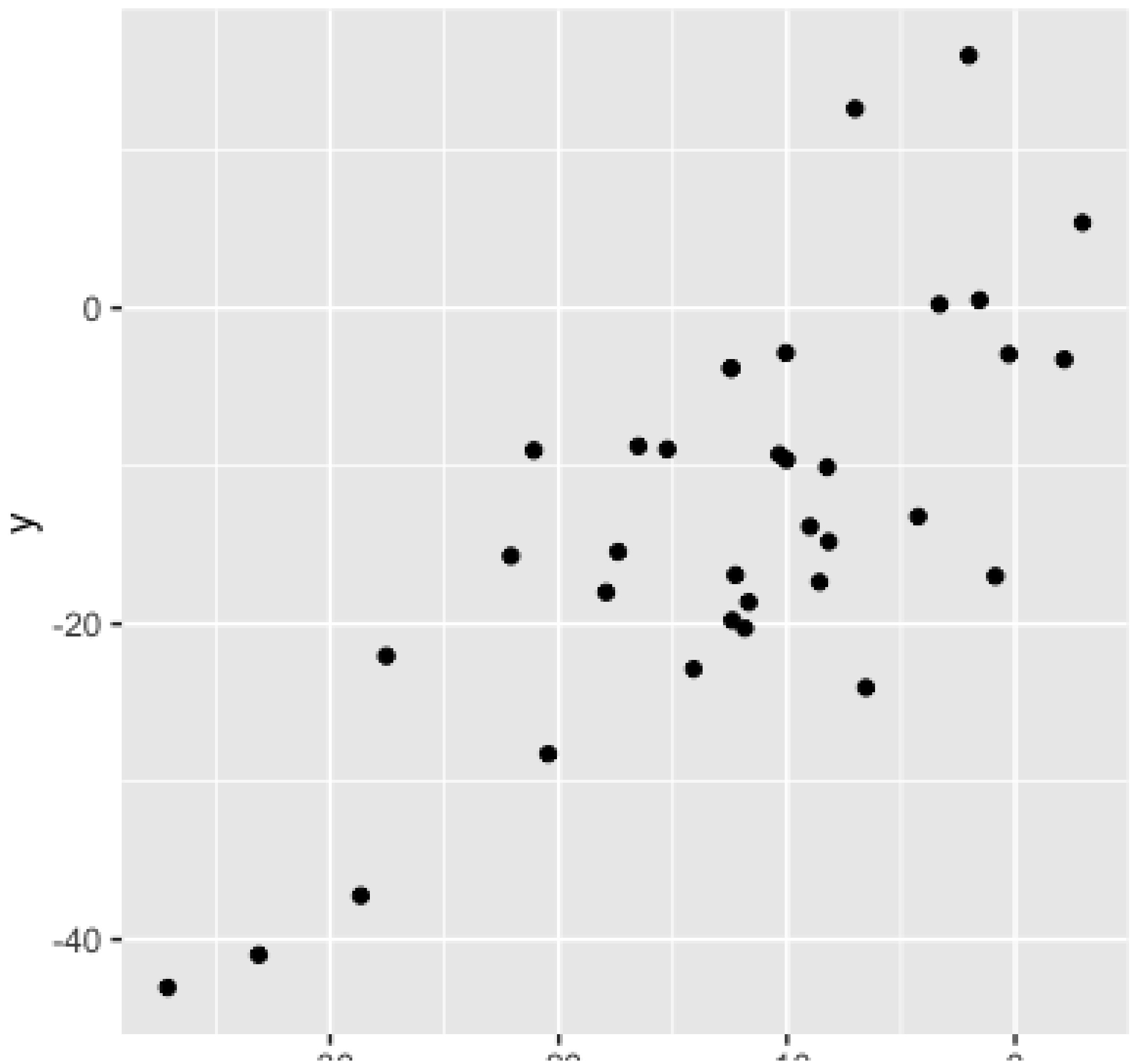
0.04 (no relationship)



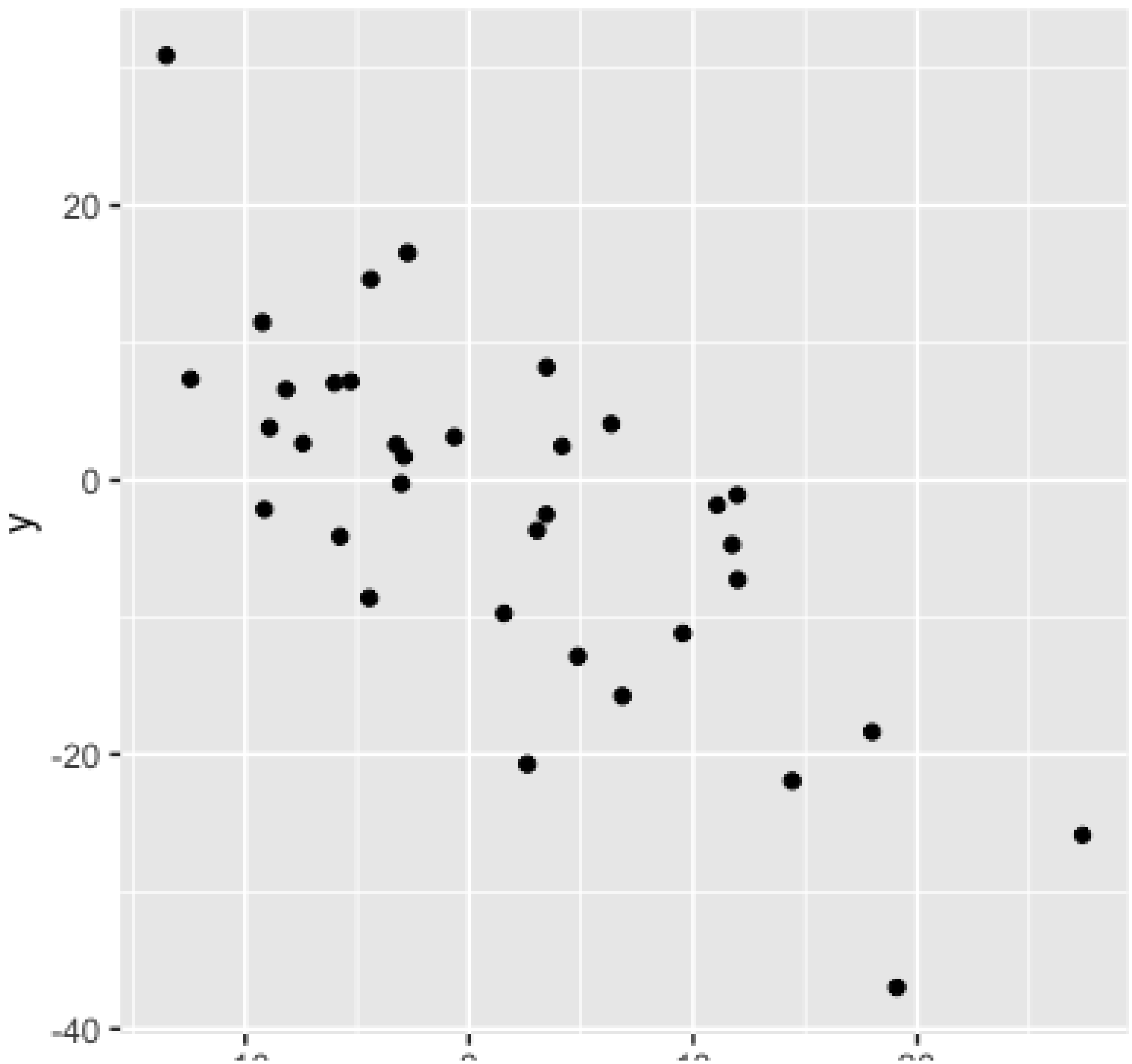
- Knowing the value of `x` doesn't tell us anything about `y`

Sign = direction

0.75: as `x` increases, `y` increases

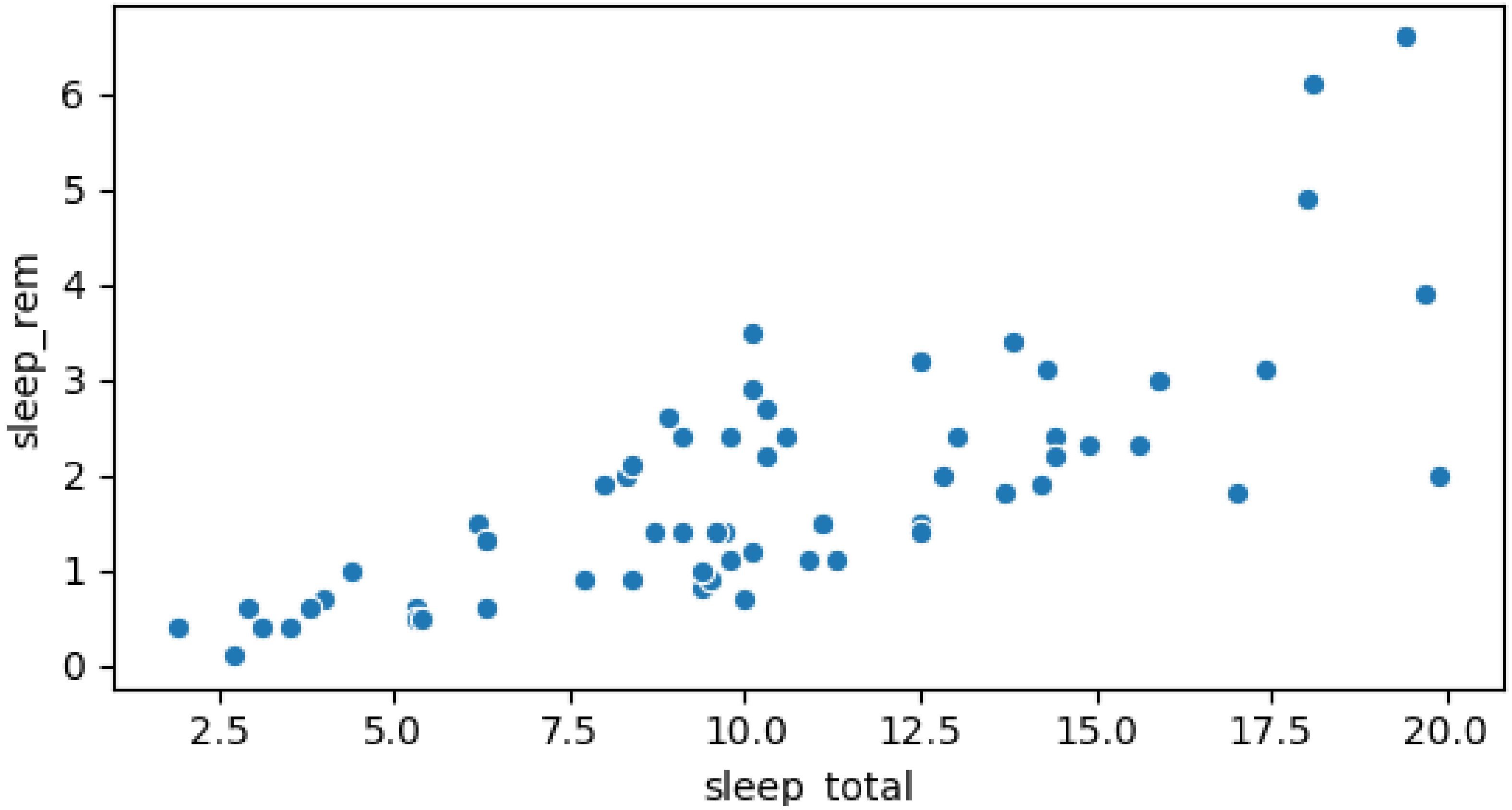


-0.75: as `x` increases, `y` decreases



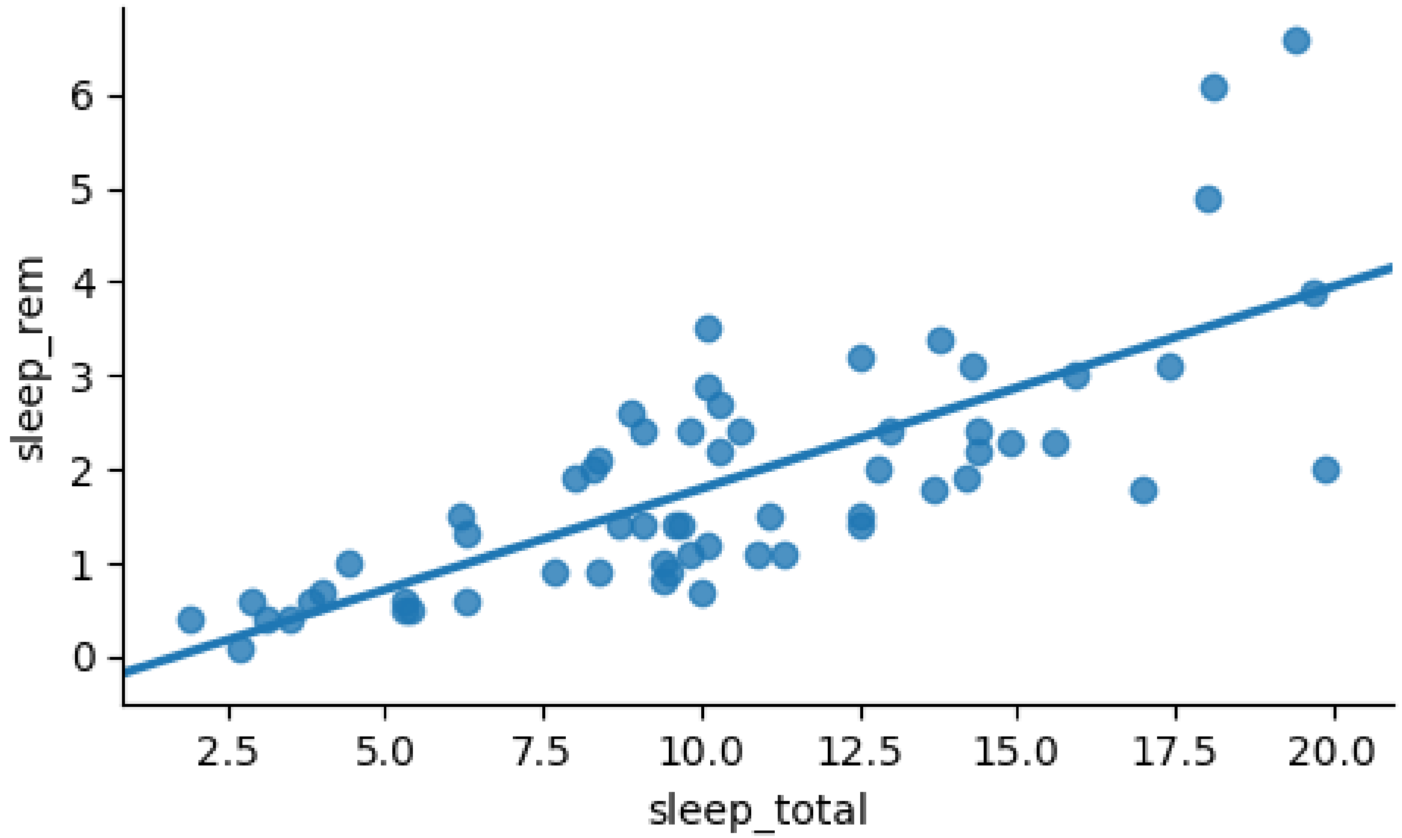
Visualizing relationships

```
import seaborn as sns
sns.scatterplot(x="sleep_total", y="sleep_rem", data=msleep)
plt.show()
```



Adding a trendline

```python
import seaborn as sns
sns.lmplot(x="sleep_total", y="sleep_rem", data=msleep, ci=None)
plt.show()
```



Computing correlation

```
msleep['sleep_total'].corr(msleep['sleep_rem'])
```

```
0.751755
```

```
msleep['sleep_rem'].corr(msleep['sleep_total'])
```

```
0.751755
```

Many ways to calculate correlation

- Used in this course: Pearson product-moment correlation (r)
  - Most common
  - $\bar{x} =$ mean of x
  - $\sigma_x =$ standard deviation of x

$$r = \sum_{i=1}^{n} \frac{(x_i - \bar{x})(y_i - \bar{y})}{\sigma_x \times \sigma_y}$$

- Variations on this formula:
  - Kendall's tau
  - Spearman's rho

Let's practice!

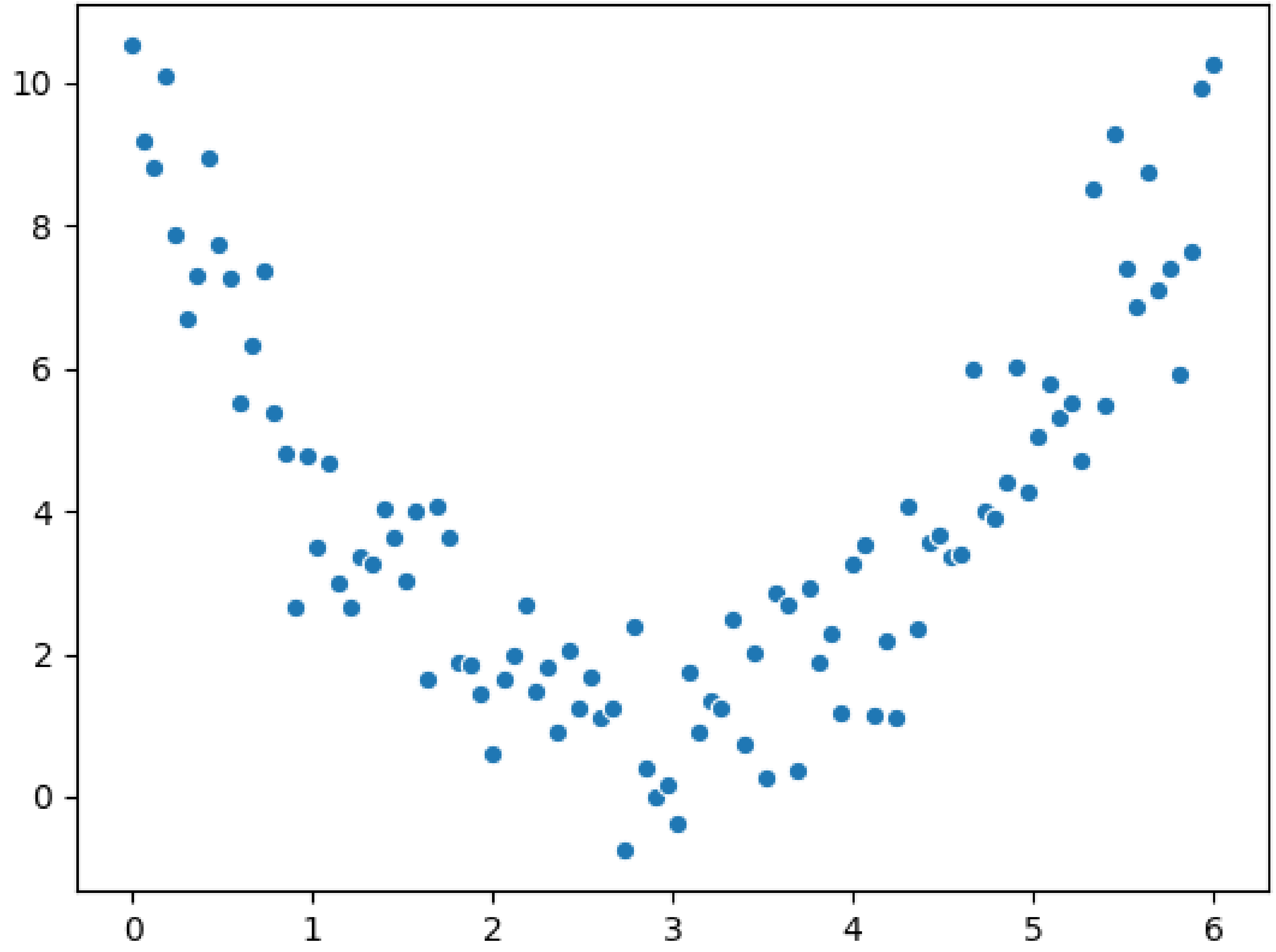
INTRODUCTION TO STATISTICS IN PYTHON

Correlation caveats

INTRODUCTION TO STATISTICS IN PYTHON

Maggie Matsui
Content Developer, DataCamp

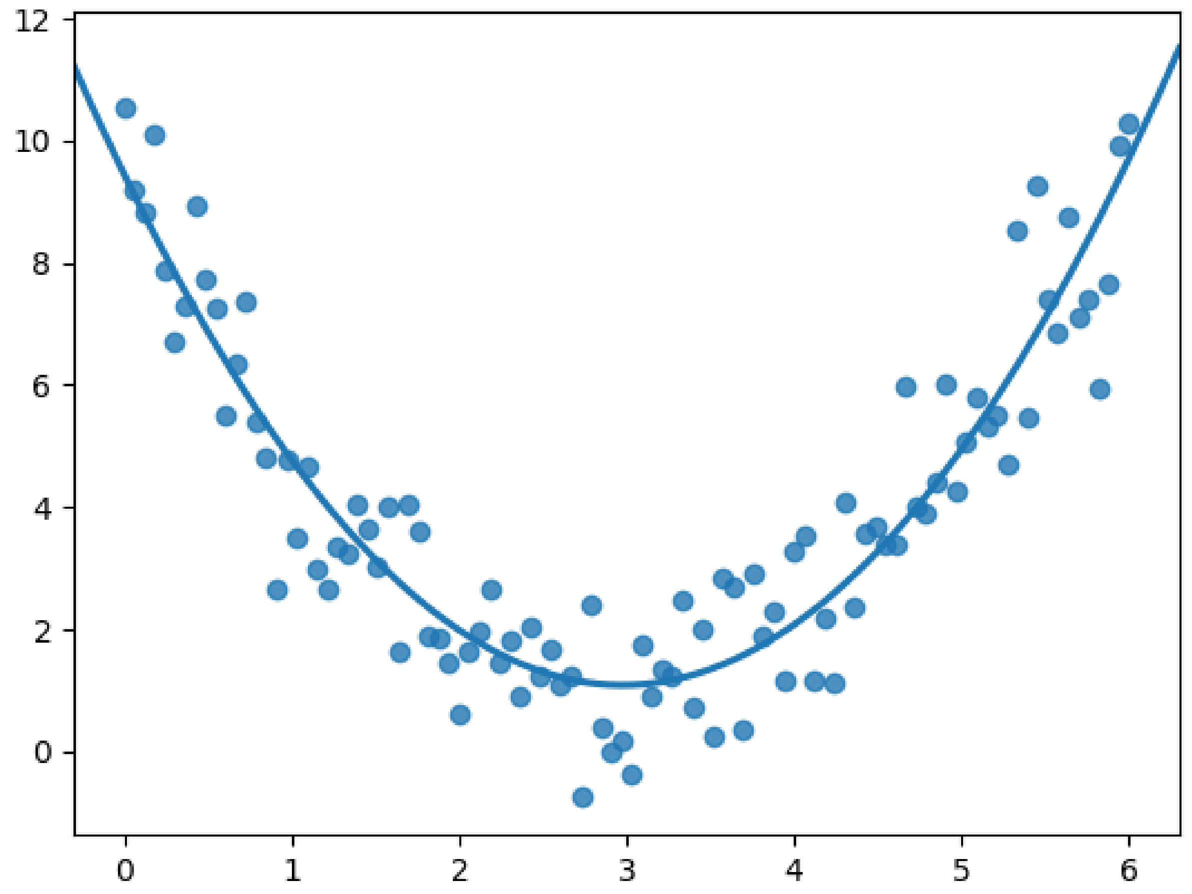
Non-linear relationships



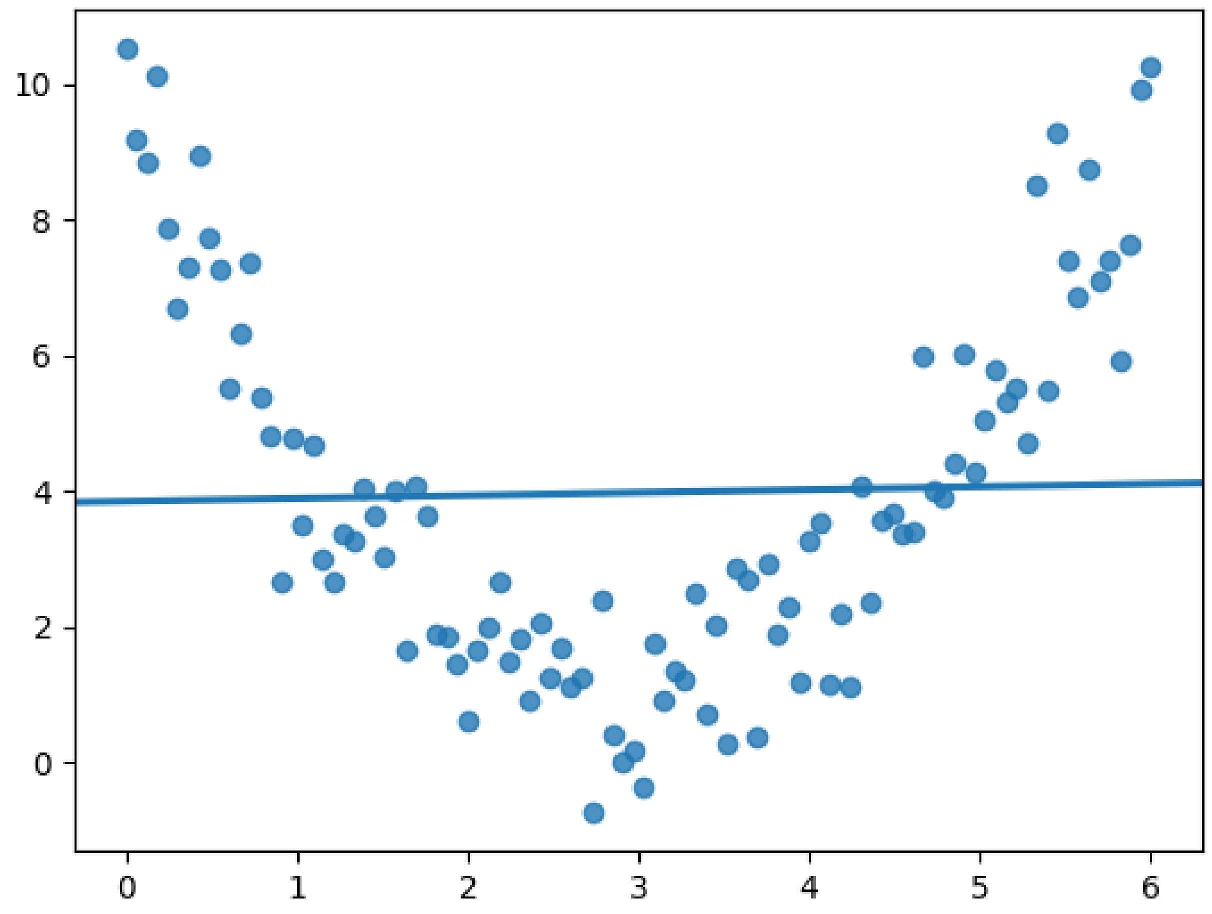
$$r = 0.18$$

Non-linear relationships

What we see:



What the correlation coefficient sees:



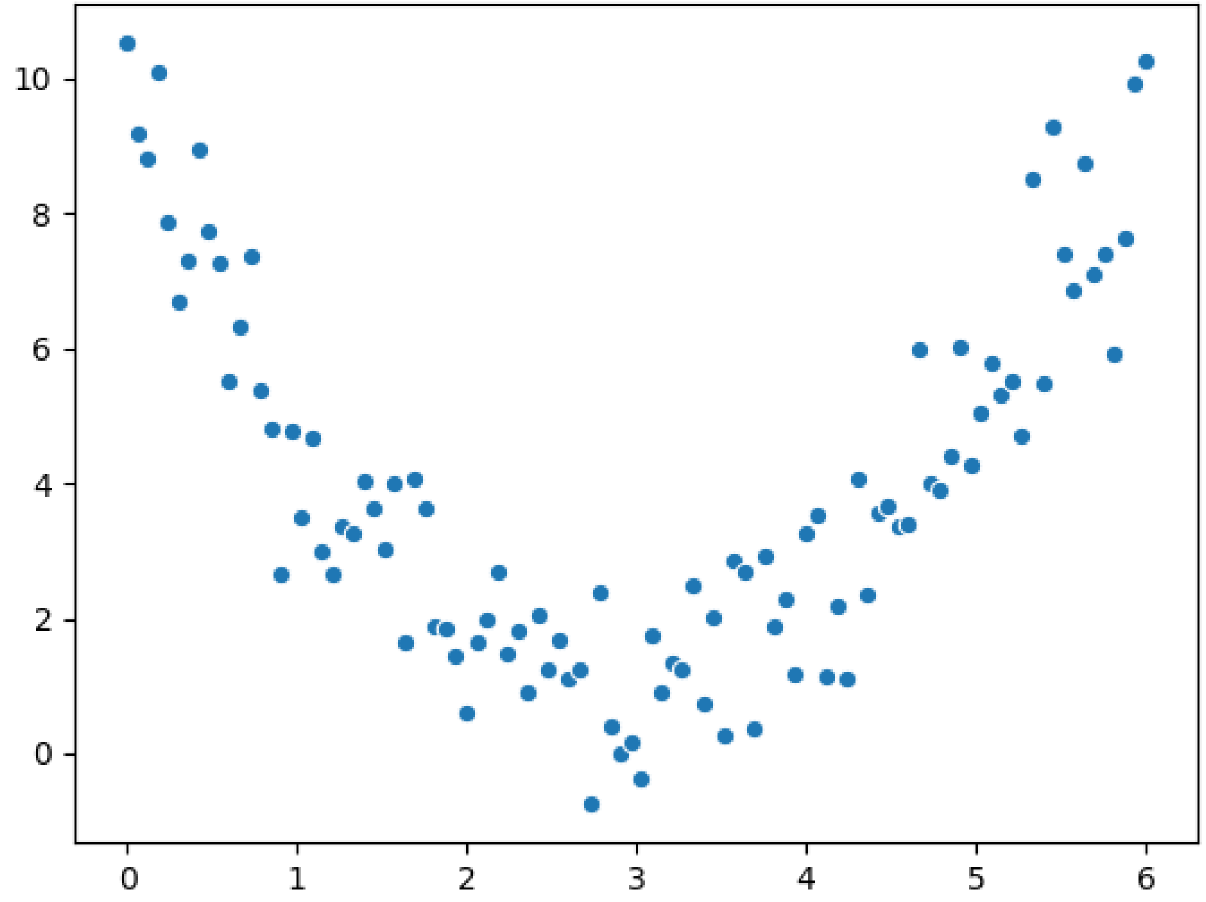
Correlation only accounts for linear relationships

Correlation shouldn't be used blindly

```
df['x'].corr(df['y'])
```

```
0.081094
```

Always visualize your data

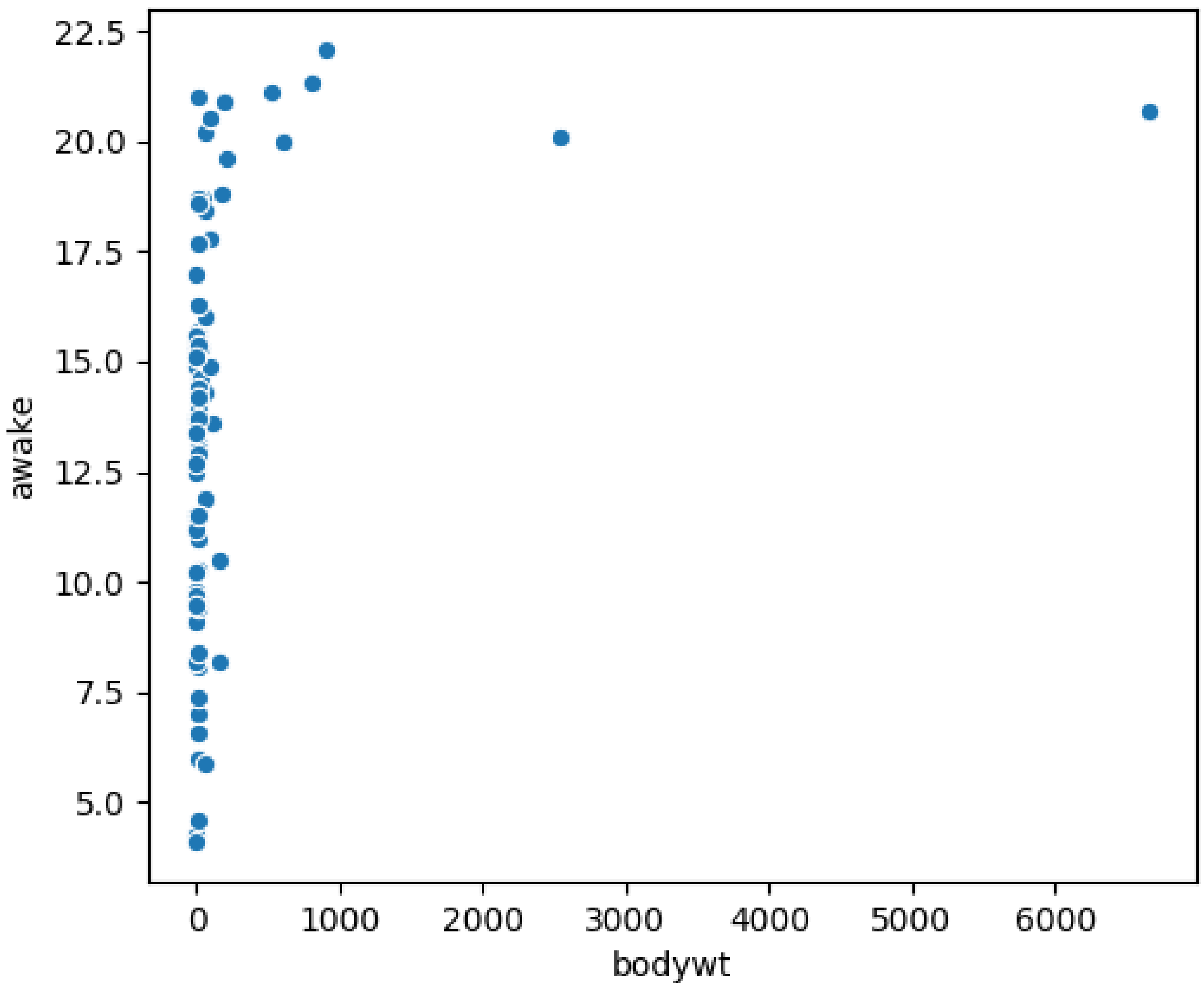


Mammal sleep data

```python
print(msleep)
```

```
                  name       genus   vore         order  ... sleep_cycle  awake  brainwt   bodywt
1              Cheetah    Acinonyx  carni     Carnivora  ...         NaN   11.9      NaN   50.000
2           Owl monkey       Aotus   omni      Primates  ...         NaN    7.0  0.01550    0.480
3      Mountain beaver  Aplodontia  herbi      Rodentia  ...         NaN    9.6      NaN    1.350
4 Greater short-ta...     Blarina   omni  Soricomorpha  ...    0.133333    9.1  0.00029    0.019
5                  Cow         Bos  herbi  Artiodactyla  ...    0.666667   20.0  0.42300  600.000
..                 ...         ...    ...           ...  ...         ...    ...      ...      ...
79          Tree shrew      Tupaia   omni    Scandentia  ...    0.233333   15.1  0.00250    0.104
80 Bottle-nosed do...    Tursiops  carni       Cetacea  ...         NaN   18.8      NaN  173.330
81               Genet     Genetta  carni     Carnivora  ...         NaN   17.7  0.01750    2.000
82          Arctic fox      Vulpes  carni     Carnivora  ...         NaN   11.5  0.04450    3.380
83             Red fox      Vulpes  carni     Carnivora  ...    0.350000   14.2  0.05040    4.230
```

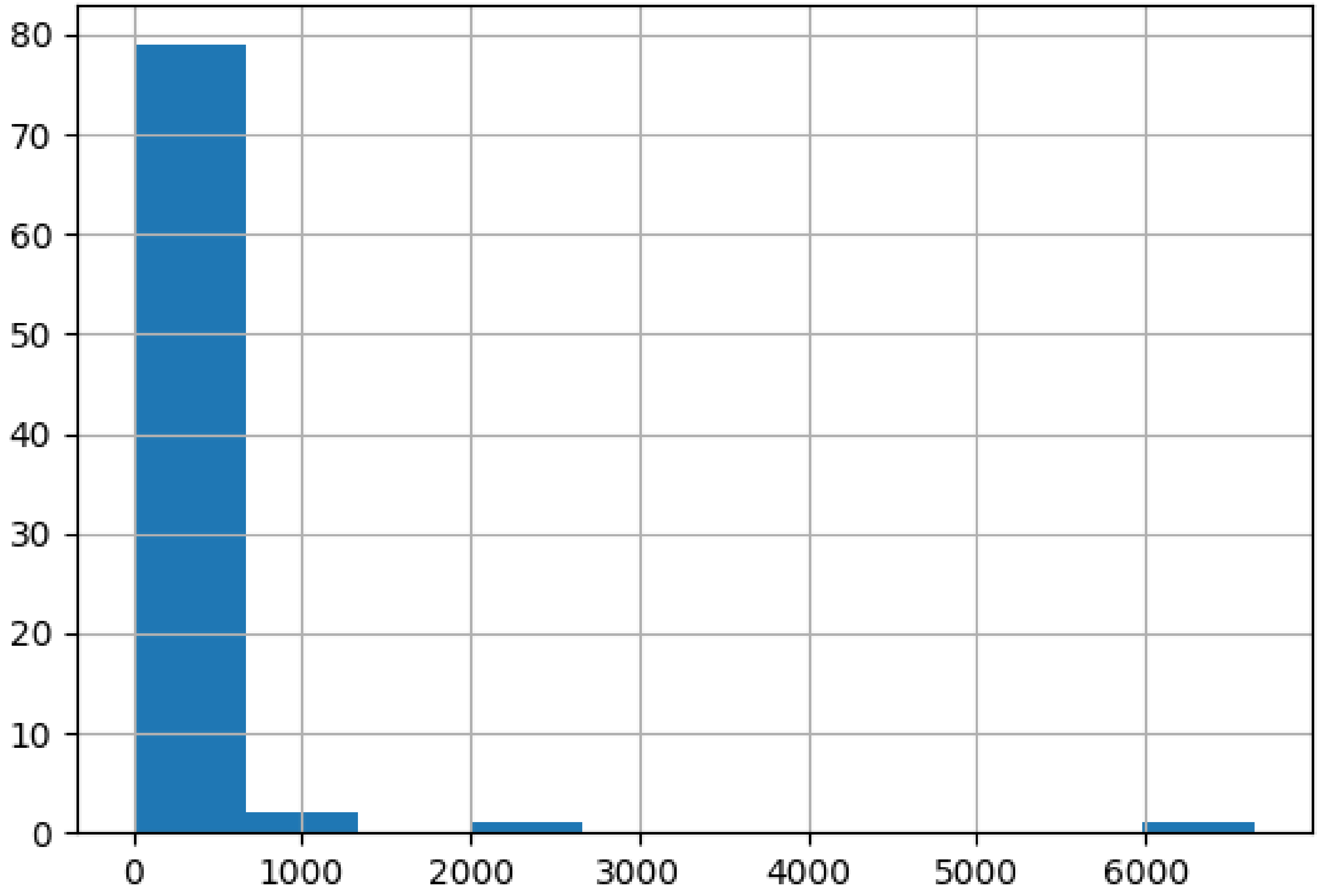
Body weight vs. awake time




```
msleep['bodywt'].corr(msleep['awake'])
```

```
0.3119801
```

Distribution of body weight



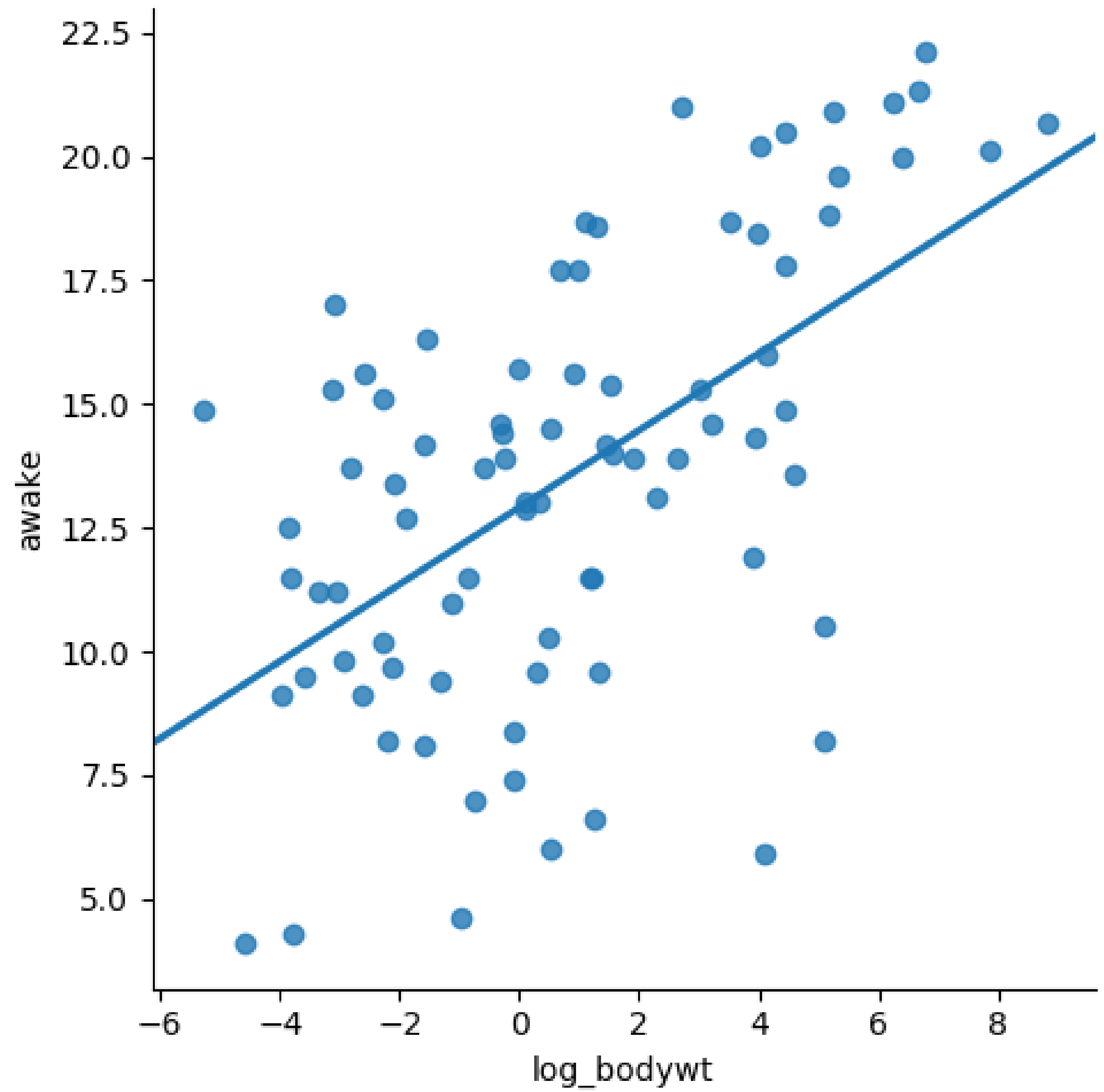
Log transformation

```
msleep['log_bodywt'] = np.log(msleep['bodywt'])

sns.lmplot(x='log_bodywt',
           y='awake',
           data=msleep,
           ci=None)
plt.show()
```

```
msleep['log_bodywt'].corr(msleep['awake'])
```

```
0.5687943
```



Other transformations

- Log transformation (`log(x)`)
- Square root transformation (`sqrt(x)`)
- Reciprocal transformation (`1 / x`)
- Combinations of these, e.g.:
  - `log(x)` and `log(y)`
  - `sqrt(x)` and `1 / y`

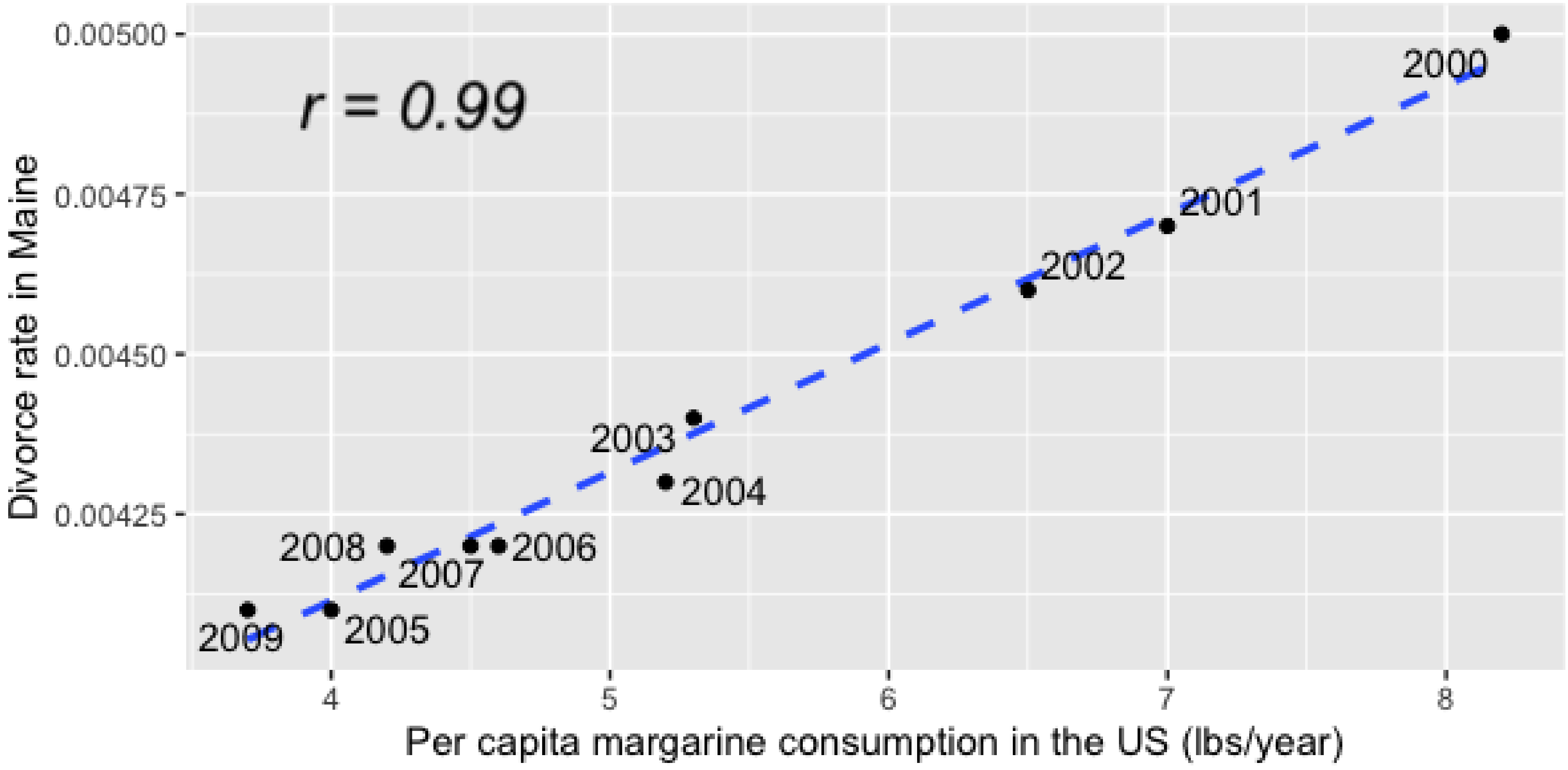
Why use a transformation?

- Certain statistical methods rely on variables having a linear relationship
  - Correlation coefficient
  - Linear regression

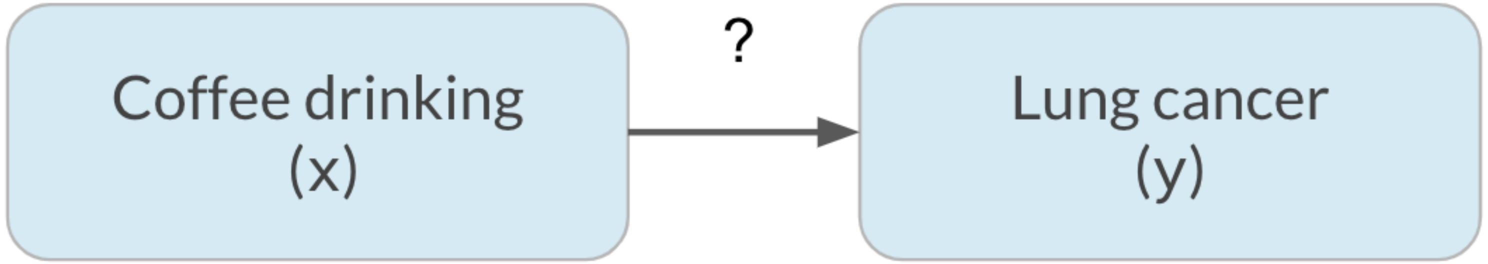
Introduction to Linear Modeling in Python

Correlation does not imply causation

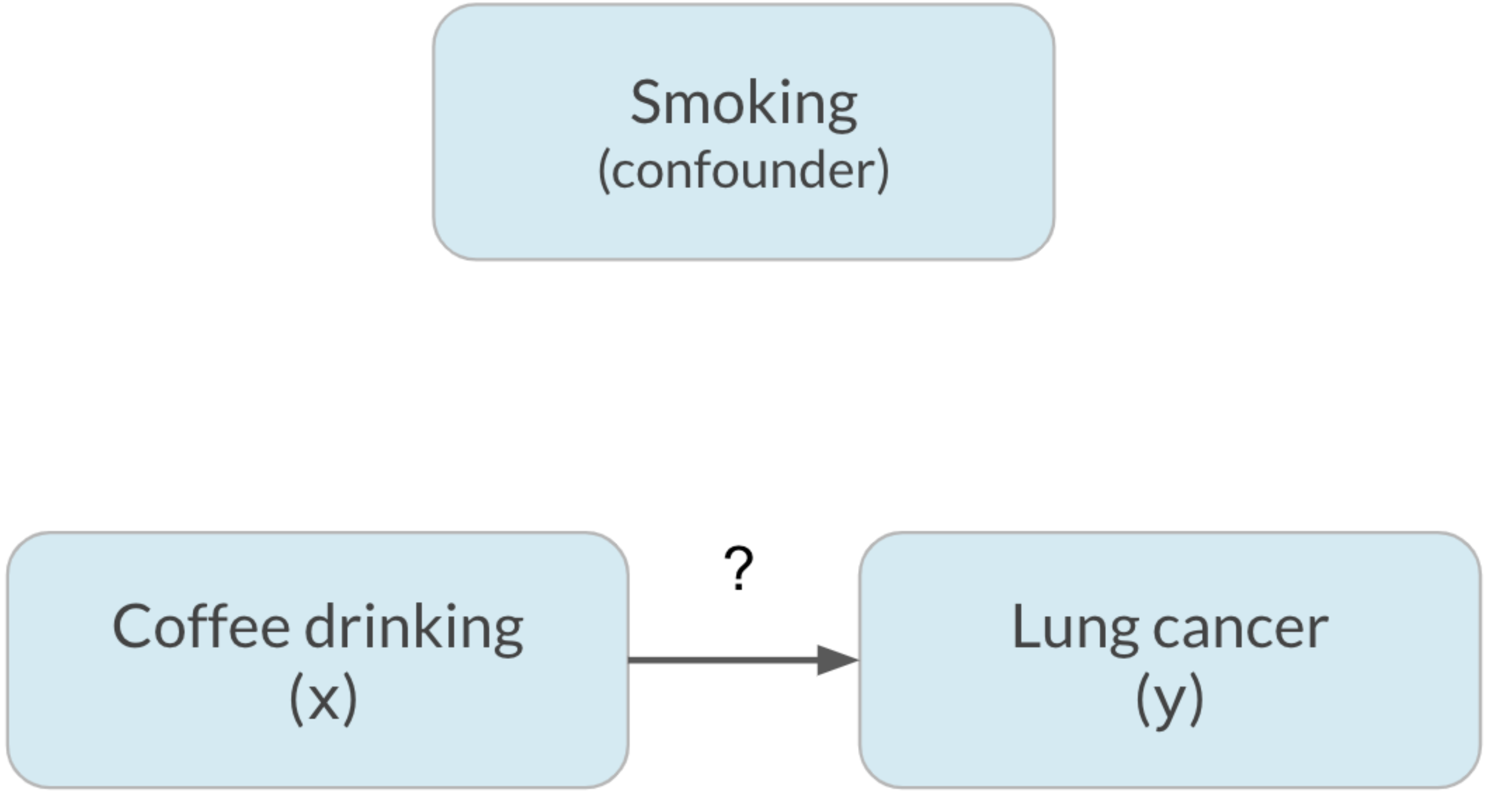
x *is correlated with* y **does not mean** x *causes* y



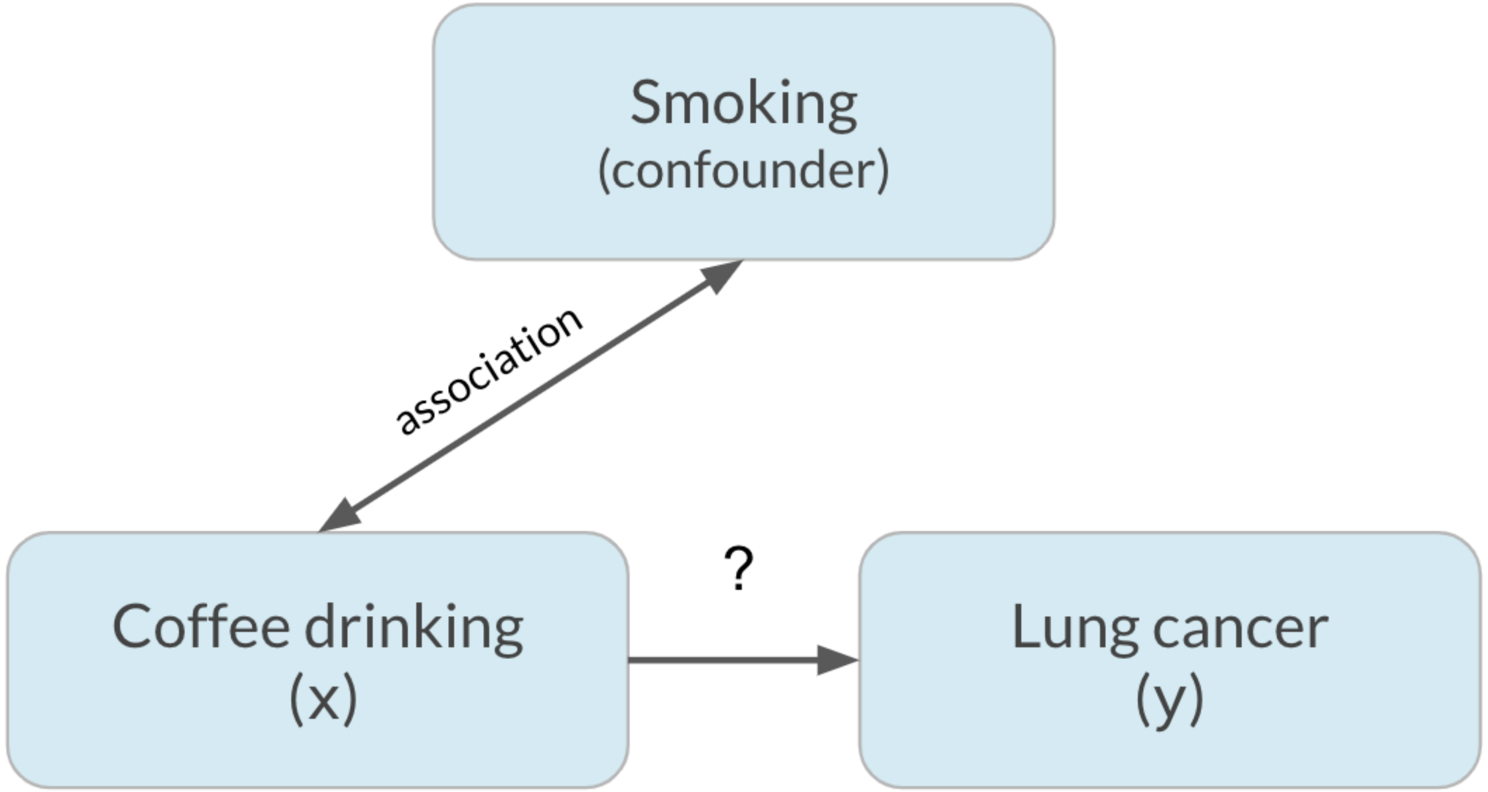
Confounding



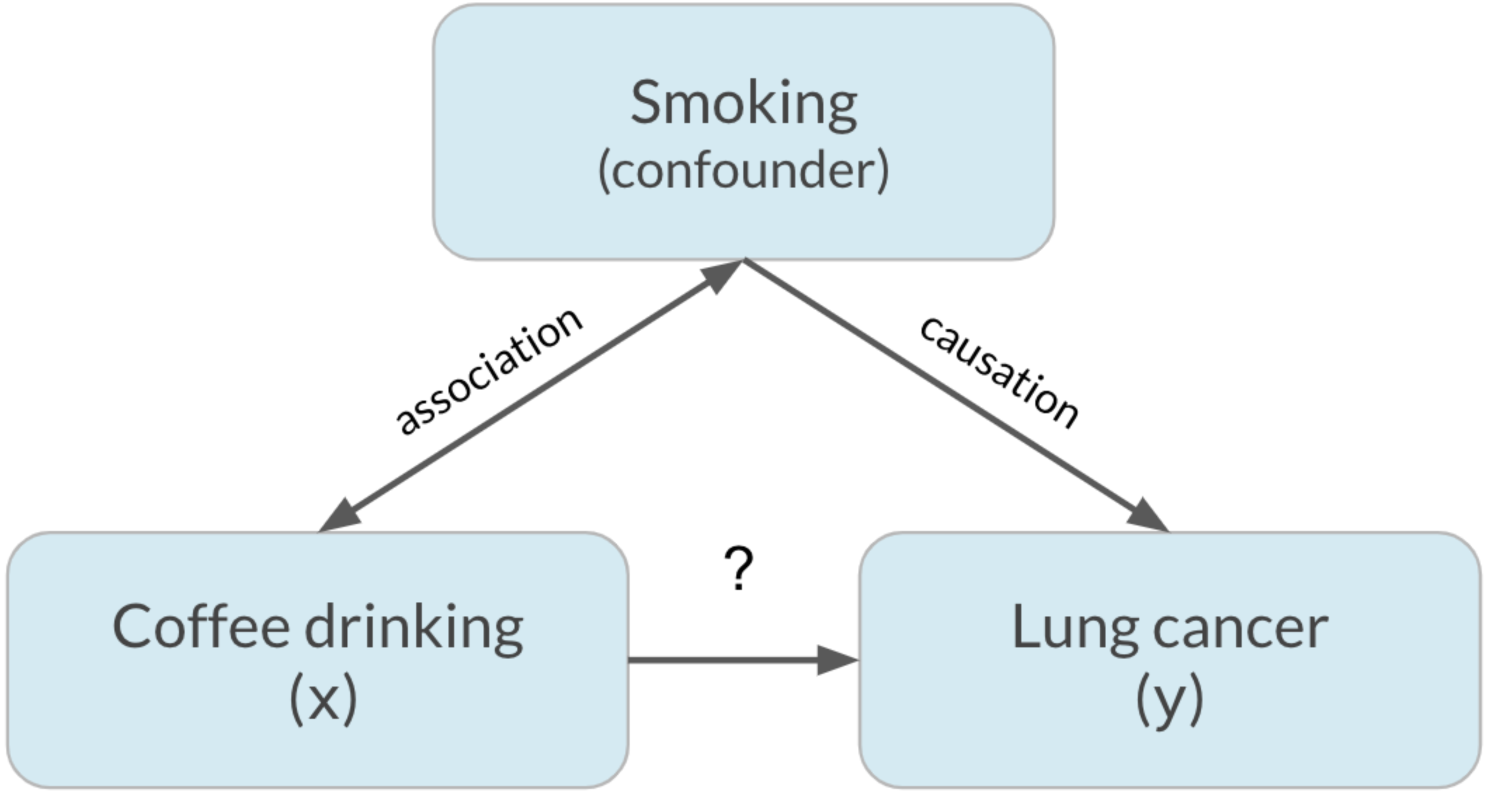
Confounding



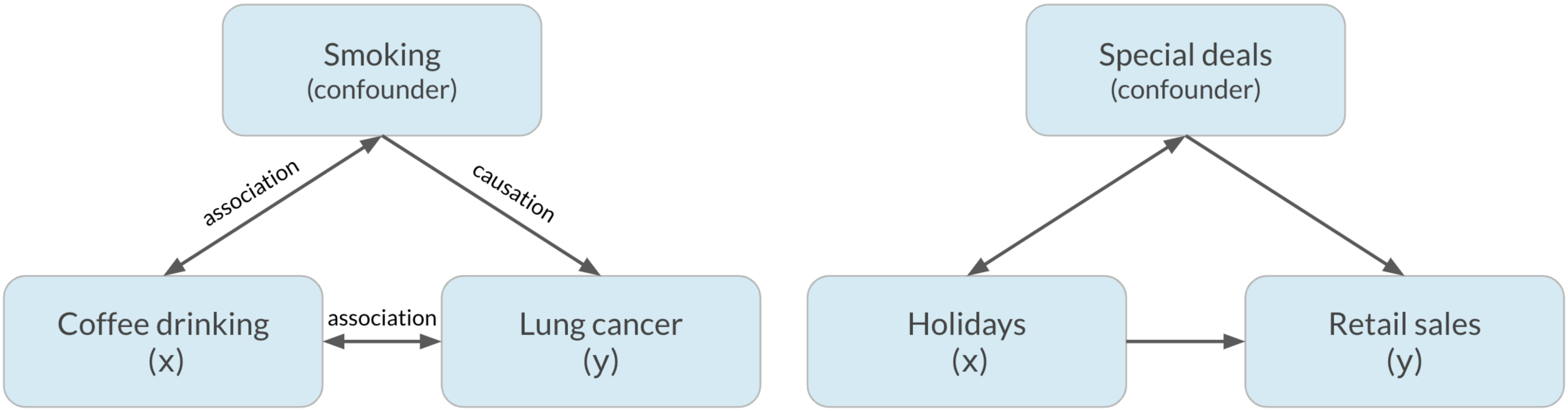
Confounding



Confounding



Confounding



Let's practice!

INTRODUCTION TO STATISTICS IN PYTHON

Design of experiments

INTRODUCTION TO STATISTICS IN PYTHON

Maggie Matsui
Content Developer, DataCamp

Vocabulary

Experiment aims to answer: *What is the effect of the treatment on the response?*

- Treatment: explanatory/independent variable
- Response: response/dependent variable

E.g.: *What is the effect of an advertisement on the number of products purchased?*

- Treatment: advertisement
- Response: number of products purchased

Controlled experiments

- Participants are assigned by researchers to either treatment group or control group
  - Treatment group sees advertisement
  - Control group does not
- Groups should be comparable so that causation can be inferred
- If groups are not comparable, this could lead to confounding (bias)
  - Treatment group average age: 25
  - Control group average age: 50
  - Age is a potential confounder

The gold standard of experiments will use...

- Randomized controlled trial
  - Participants are assigned to treatment/control *randomly*, not based on any other characteristics
  - Choosing randomly helps ensure that groups are comparable
- Placebo
  - Resembles treatment, but has no effect
  - Participants will not know which group they're in
  - In clinical trials, a sugar pill ensures that the effect of the drug is actually due to the drug itself and not the idea of receiving the drug

The gold standard of experiments will use...

- Double-blind trial
  - Person administering the treatment/running the study doesn't know whether the treatment is real or a placebo
  - Prevents bias in the response and/or analysis of results

Fewer opportunities for bias = more reliable conclusion about causation

Observational studies

- Participants are not assigned randomly to groups
  - Participants assign themselves, usually based on pre-existing characteristics
- Many research questions are not conducive to a controlled experiment
  - You can't force someone to smoke or have a disease
  - You can't make someone have certain past behavior
- Establish association, not causation
  - Effects can be confounded by factors that got certain people into the control or treatment group
  - There are ways to control for confounders to get more reliable conclusions about association

Longitudinal vs. cross-sectional studies

Longitudinal study

- Participants are followed over a period of time to examine effect of treatment on response
- Effect of age on height is not confounded by generation
- More expensive, results take longer

Cross-sectional study

- Data on participants is collected from a single snapshot in time
- Effect of age on height is confounded by generation
- Cheaper, faster, more convenient

Let's practice!

Congratulations!

INTRODUCTION TO STATISTICS IN PYTHON

Maggie Matsui
Content Developer, DataCamp

Overview

Chapter 1

- What is statistics?
- Measures of center
- Measures of spread

Chapter 2

- Measuring chance
- Probability distributions
- Binomial distribution

Chapter 3

- Normal distribution
- Central limit theorem
- Poisson distribution

Chapter 4

- Correlation
- Controlled experiments
- Observational studies

Build on your skills

- Introduction to Linear Modeling in Python

Congratulations!

INTRODUCTION TO STATISTICS IN PYTHON