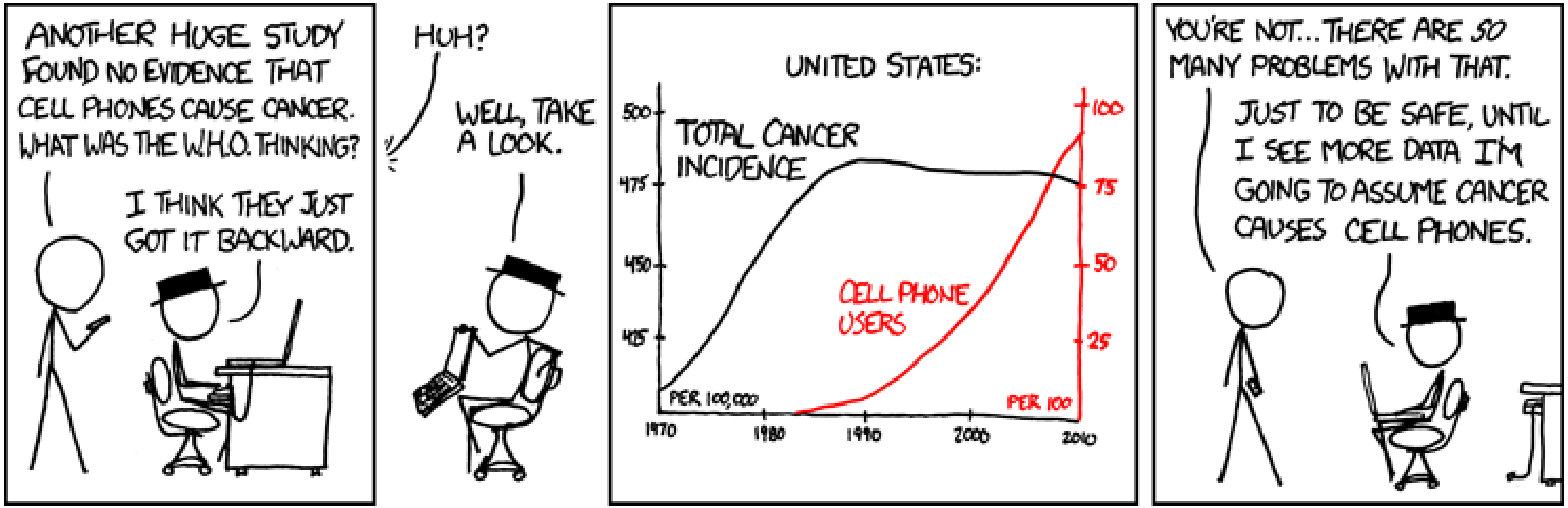
Statistics

Welcome back to CS 2100!

Prof. Rasika Bhalerao


ANOTHER HUGE STUDY FOUND NO EVIDENCE THAT CELL PHONES CAUSE CANCER. WHAT WAS THE W.H.O. THINKING?
I THINK THEY JUST GOT IT BACKWARD.
HUH?
WELL, TAKE A LOOK.
UNITED STATES:
TOTAL CANCER INCIDENCE
CELL PHONE USERS
500
475
450
425
PER 100,000
100
75
50
25
PER 100
1970
1980
1990
2000
2010
YOU'RE NOT... THERE ARE SO MANY PROBLEMS WITH THAT.
JUST TO BE SAFE, UNTIL I SEE MORE DATA I'M GOING TO ASSUME CANCER CAUSES CELL PHONES.

Source: https://xkcd.com/925

Correlation

Correlation measures the strength of a linear relationship between two variables.

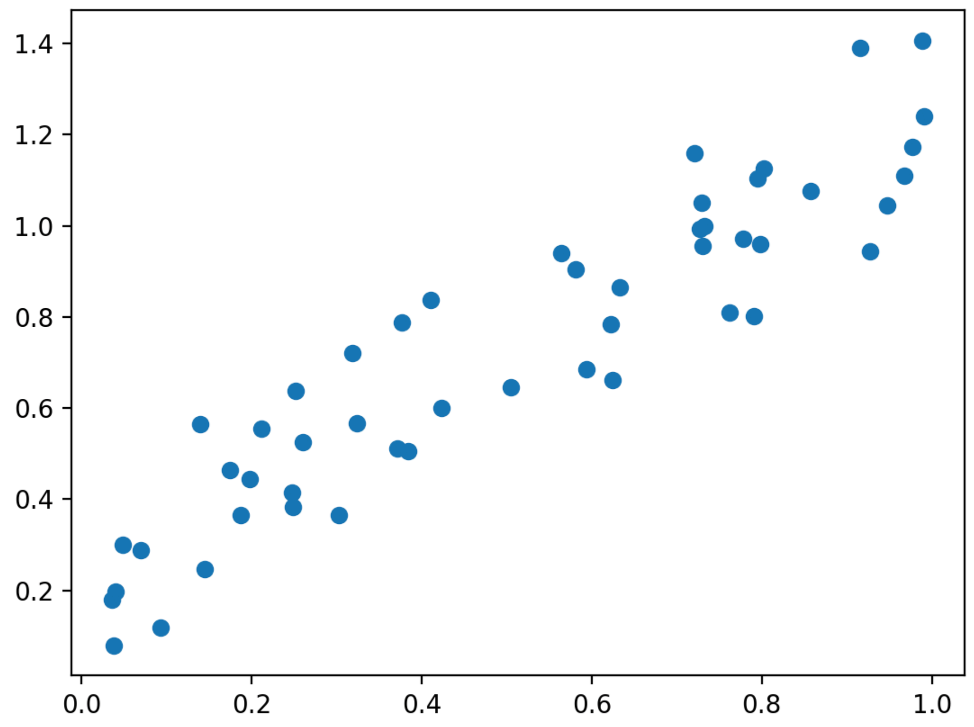
Variable: a characteristic or attribute that can take on different values

E.g., `age` can be a variable, for which we collect data from a group of people

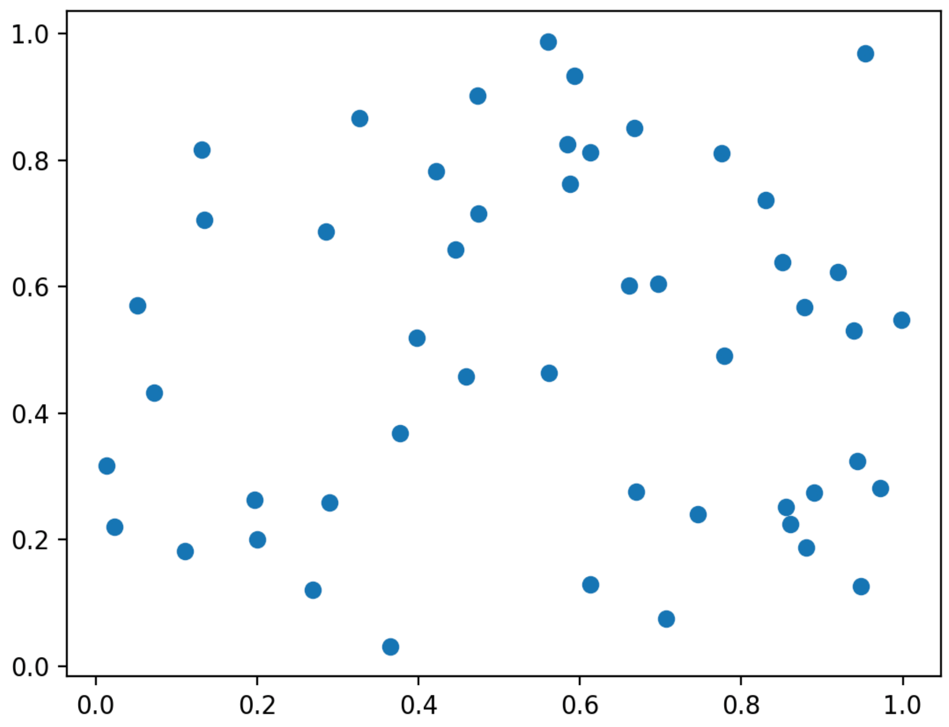
Correlation

Two variables: one on the x axis, and one on the y axis. (Imagine each dot is a person, with their age on the x axis and their height on the y axis.)

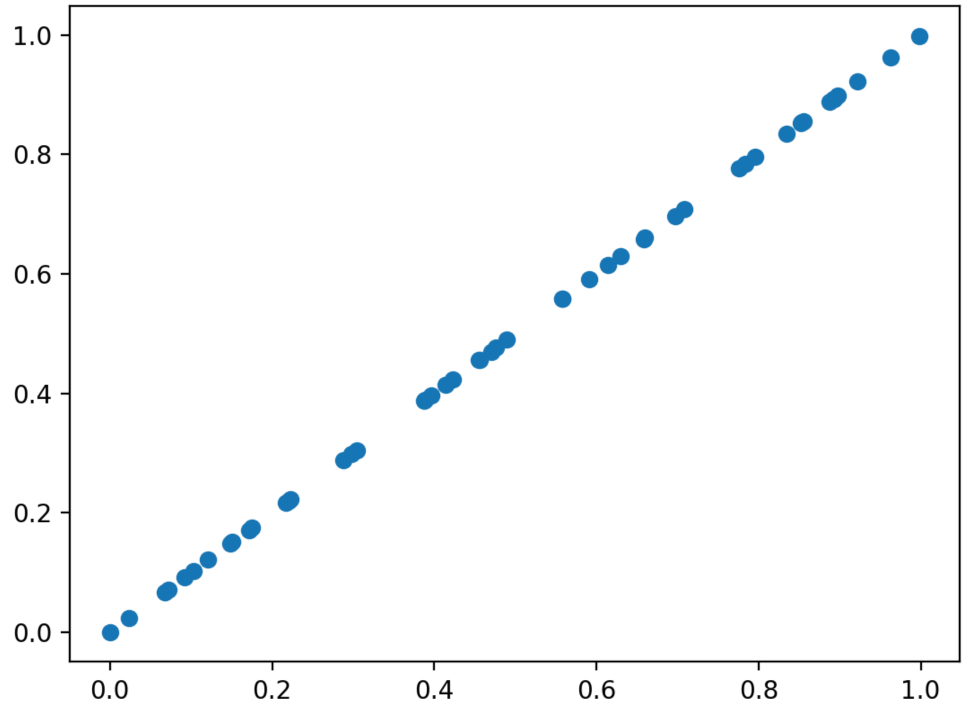
Two variables with a somewhat strong correlation:



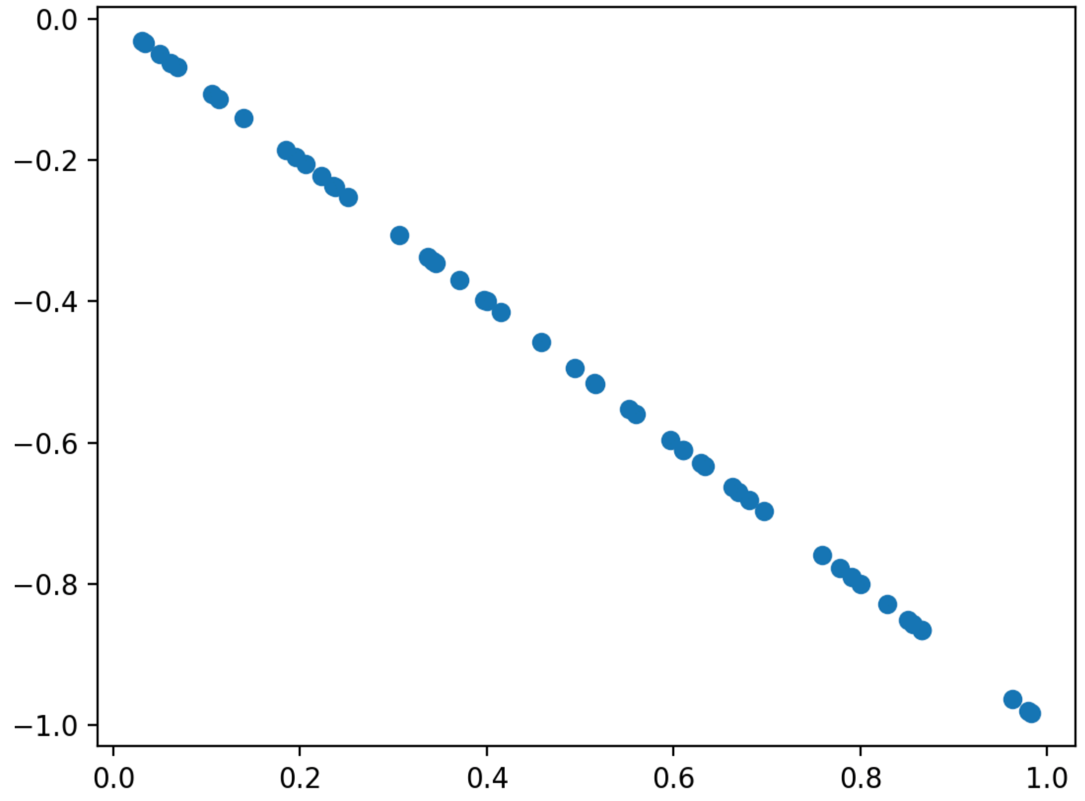
Two variables with very weak correlation, if at all (*correlation coefficient* close to 0):



Extremely strong correlation (correlation coefficient) close to 1:

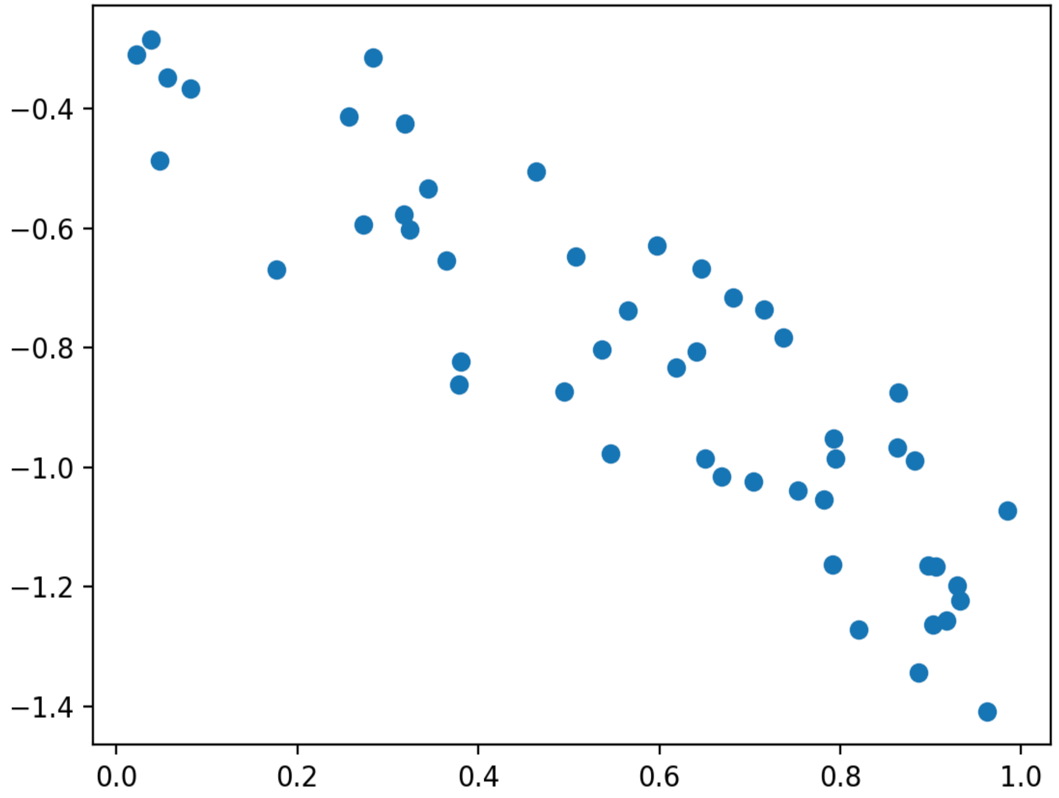


Extremely strong correlation, but in the negative direction (correlation coefficient close to -1):

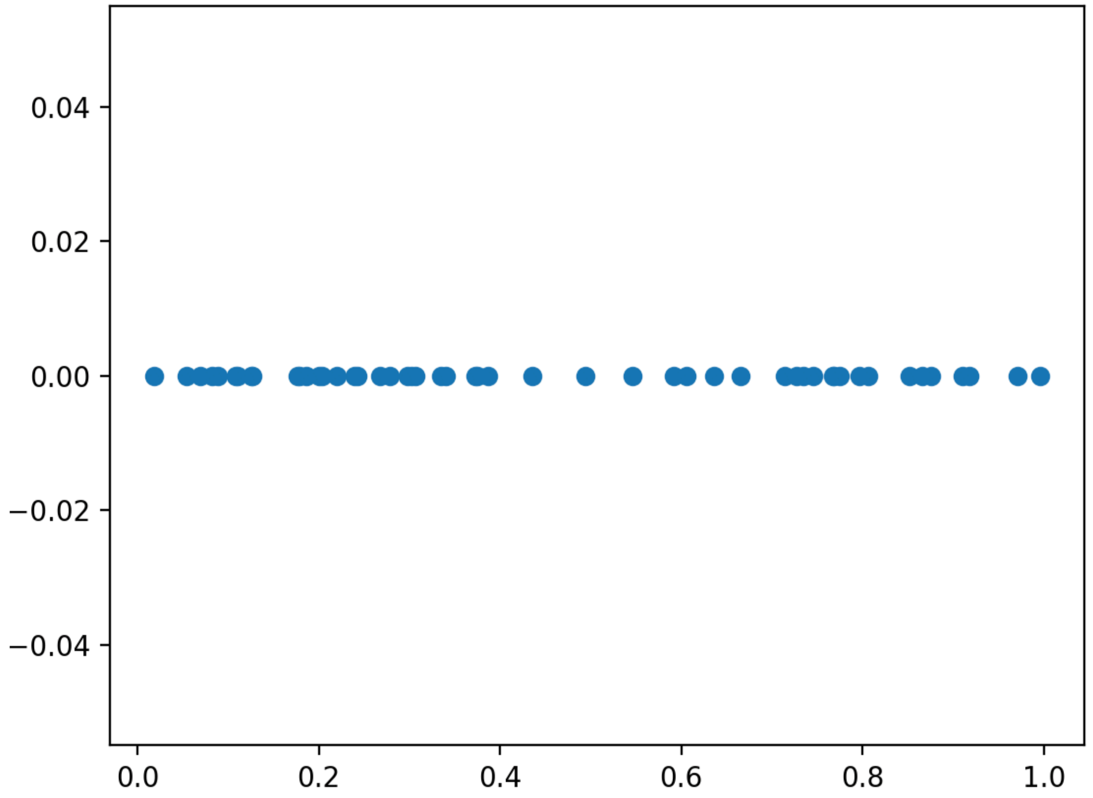


As `x` increases, `y` decreaes.

Slightly strong negative correlaion (correlation coefficient between -1 and 0):



Correlation coefficient of 0:



Pearson Correlation Coefficient: the most popular method of calculating correlation

The Pearson Correlation Coefficient between two variables `x` and `y` can be calculated using this formula:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

where `x̄` is the mean of all `x` values, and `ȳ` is the mean of all `y` values.

You do not need to memorize this formula -- Python will calculate it for you.

Calculating Correlation

```
df = pd.DataFrame(
    {'Person': ['Elephant', 'Cat', 'Dog'],
    'Age': [13, 10, 3],
    'Height': [5, 4, 1]})

print(np.corrcoef(df['Age'], df['Height']))
```

```
[[1.         0.99853837]
 [0.99853837 1.        ]]
```

This is saying that the correlation coefficient between age and height is 0.9985 -- that's very close to the maximum, which is 1. It makes sense for age and height to be correlated, at least early in life.

Calculating Correlation

`numpy.corrcoef(x, y)` returns a 2-dimensional array containing the correlation coefficients:

```
[[corr(x,x)      corr(x,y)]
 [corr(y,x)      corr(y,y)]]
```

We can calculate the same thing for every pair of columns in a Pandas dataframe:

```
print(df.corr(numeric_only=True))
```

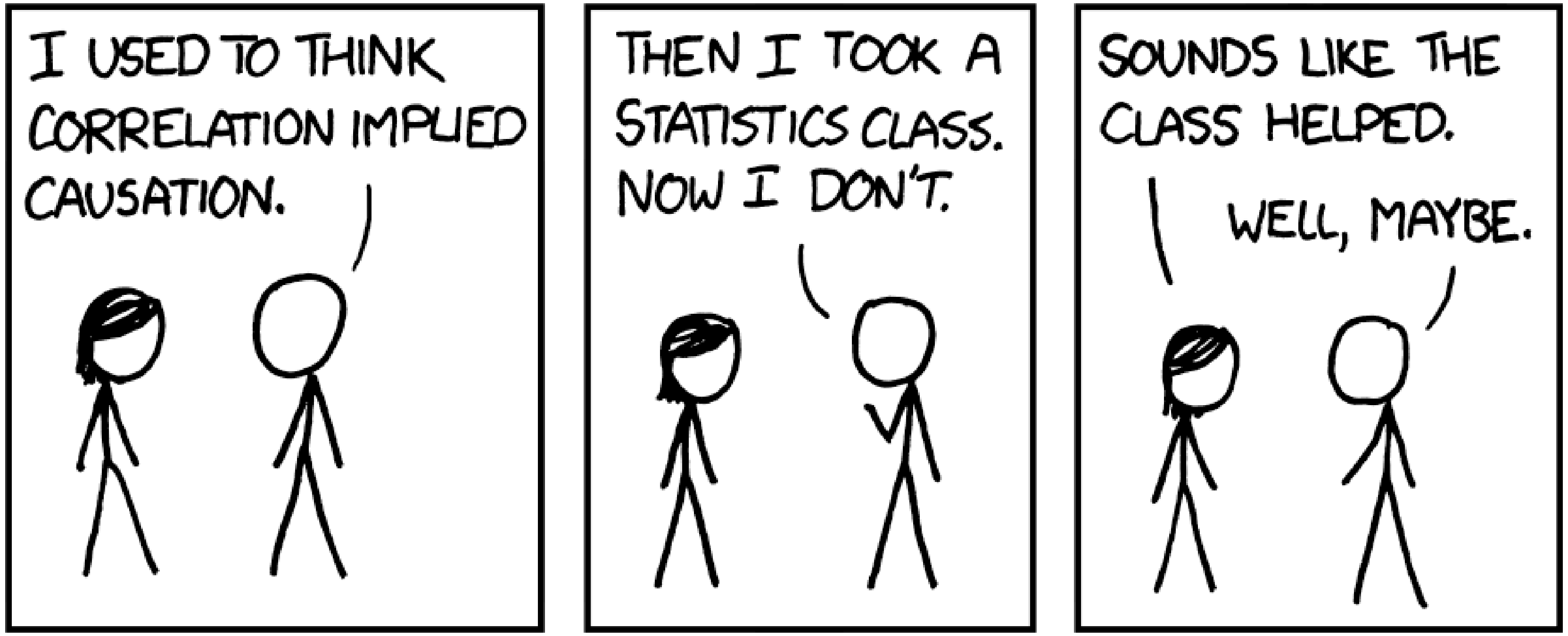
```
             Age    Height
Age     1.000000  0.998538
Height  0.998538  1.000000
```

`numeric_only=True` prevents it from trying to do calculations using text columns.

Common logical fallacy: Mistaking correlation for causation

"Correlation does not imply causation."

Correlation does not imply that an increase in one variable *causes* the other to increase (or decrease). We cannot know whether one of the variables is the cause.


I USED TO THINK CORRELATION IMPLIED CAUSATION.
THEN I TOOK A STATISTICS CLASS. NOW I DON'T.
SOUNDS LIKE THE CLASS HELPED.
WELL, MAYBE.

Source: https://xkcd.com/552

Poll: Suppose we have an array which is holding measurements for the variable `x`. Without knowing what `x` is, what is the correlation between `x` and `-x`?

1. 0
2. 1
3. -1
4. Between 0 and 1

Poll: It is sometimes the trend in some courses that students who perform poorly on exams early on end up performing at the top of their class at the end of the semester. (This might be because they feel motivated to study harder.) In a course where this is the case, what would be the correlation between the variables `Quiz_1_score` and `Quiz_4_score`?

1. 0
2. Between -1 and 0
3. Between 0 and 1
4. 1

Visualization with matplotlib

Those plots were created using a Python package called `matplotlib`.

Randomly generate `size` number of values for variables `x` and `y` and plot them:

```python
import matplotlib.pyplot as plt

size: int = 50
x = np.random.rand(size)
y = [-i + 0.1 * np.random.rand() for i in x]

plt.scatter(x, y)

plt.title('Relationship between x and y', fontsize=16, color='blue')
plt.xlabel('x')  # title of the x axis
plt.ylabel('y')  # title of the y axis

plt.show() # type: ignore
```

Bar Graph

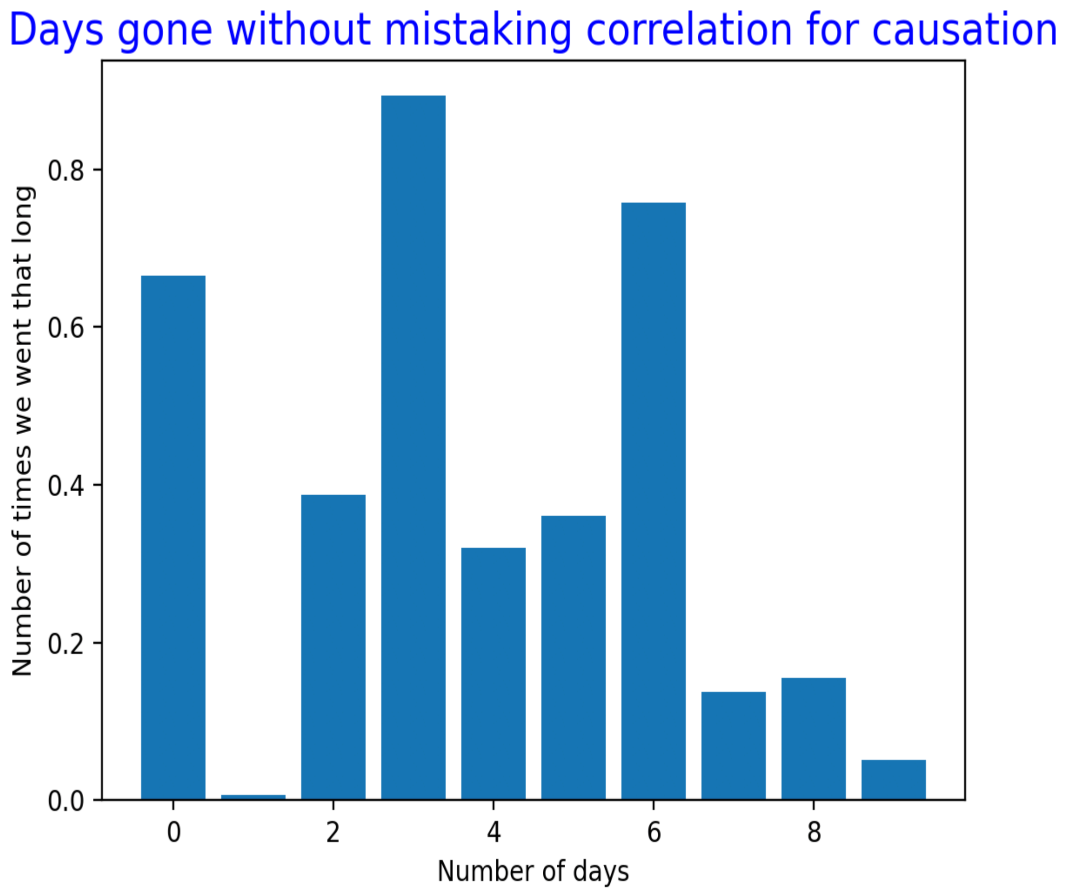
```
import matplotlib.pyplot as plt

num_categories: int = 10
categories: = [i for i in range(num_categories)]
values = np.random.rand(num_categories)

plt.bar(categories, values)

plt.title(
    'Days gone without mistaking correlation',
    fontsize=16, color='blue')
plt.xlabel('Number of days')
plt.ylabel('Number of times we went that long')

plt.show() # type: ignore
```



Histogram

A bar graph that displays the number of occurences of a single variable

Histogram of cat heights:

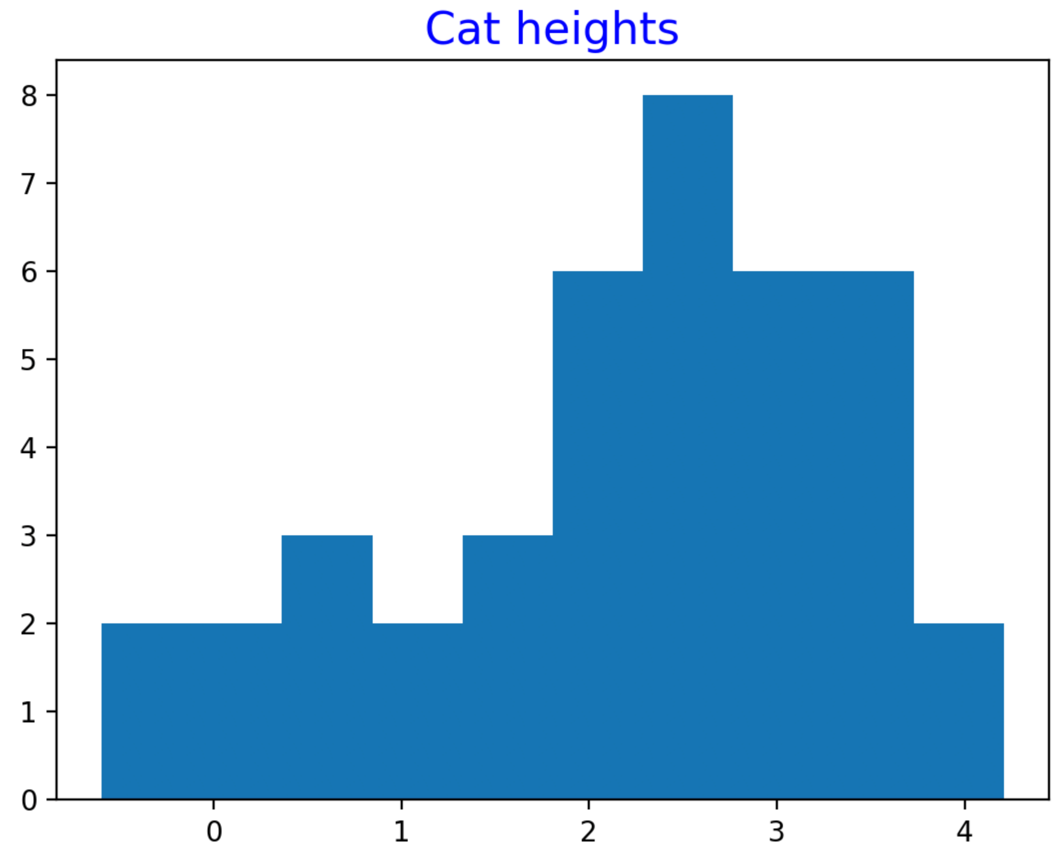
```
import matplotlib.pyplot as plt

heights = np.random.normal(
    loc=2, size=40)

plt.hist(heights)

plt.title(
    'Cat heights', fontsize=16,
    color='blue')

plt.show() # type: ignore
```



Poll: Which type of visualization should we use to display:

The number of roommates that students have (in this class)

1. Scatterplot
2. Bar graph
3. Histogram

Poll: Which type of visualization should we use to display:

The relationship between students' number of roommates and average score on an assignment

1. Scatterplot
2. Bar graph
3. Histogram

Poll: Which type of visualization should we use to display:

Students' favorite colors

1. Scatterplot
2. Bar graph
3. Histogram

Poll: Which type of visualization should we use to display:

The number of hours of sleep that students got last night

1. Scatterplot
2. Bar graph
3. Histogram

Poll: Which are true?

1. In this course, we should use `plt.show() # type: ignore` so that our data visualizations show up without triggering the MyPy error about an untyped function call.
2. In this course, we should not use `# type: ignore` anywhere in our code except on the same line as `plt.show()`.
3. `# type: ignore` makes VSCode ignore important errors that MyPy catches.
4. All of the above

Poll:

1. What is your main takeaway from today?
2. What would you like to revisit next time?