

Exploring relationships

EXPLORATORY DATA ANALYSIS IN PYTHON



Allen Downey

Professor, Olin College

Height and weight



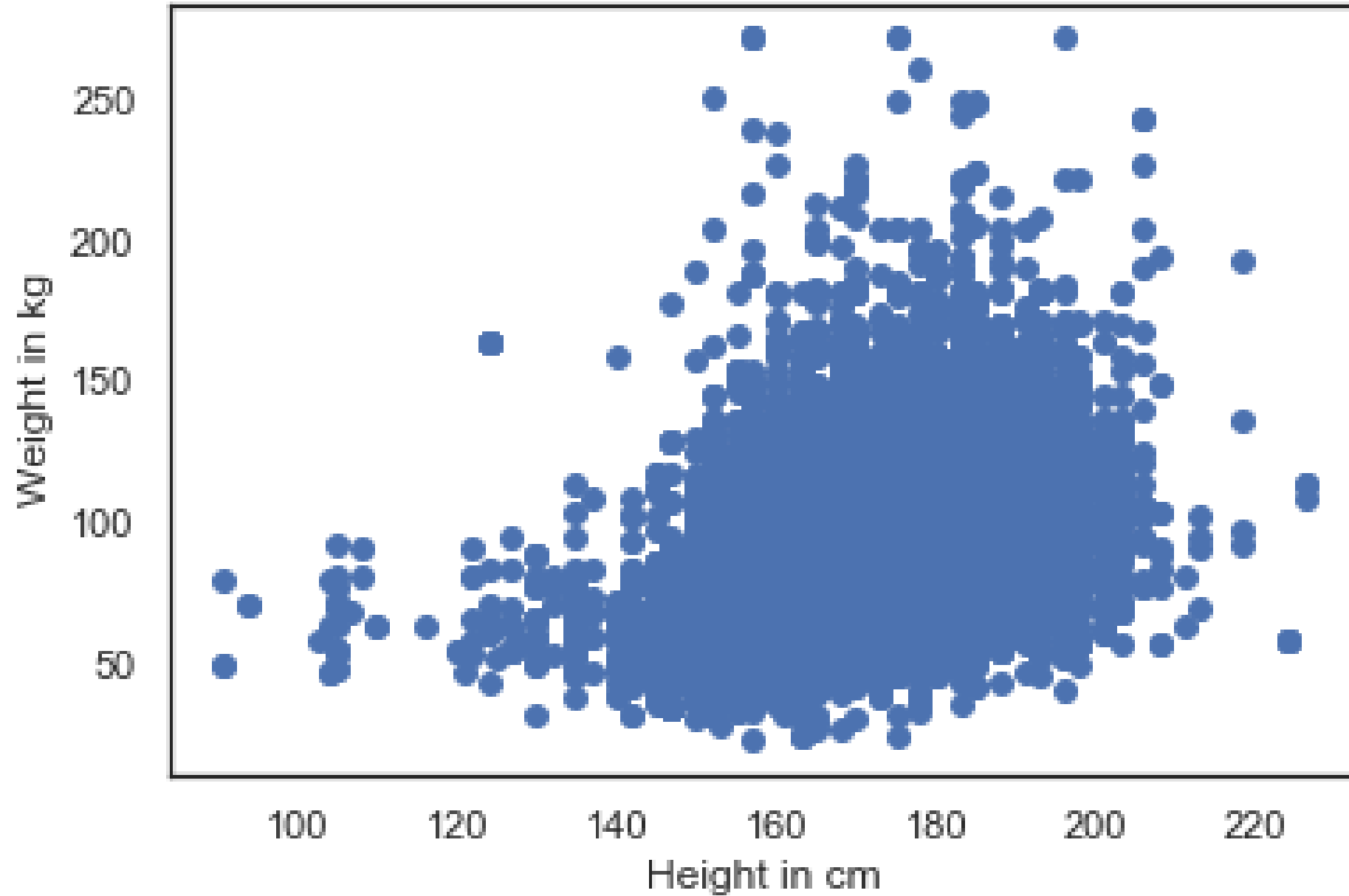
Behavioral Risk Factor Surveillance System



The Behavioral Risk Factor Surveillance System (BRFSS) is the nation's premier system of health-related telephone surveys that collect state data about U.S. residents regarding their health-related risk behaviors, chronic health conditions, and use of preventive services. Established in 1984 with 15 states, BRFSS now collects data in all 50 states as well as the District of Columbia and three U.S. territories. BRFSS completes more than 400,000 adult interviews each year, making it the largest continuously conducted health survey system in the world. [See More.](#)

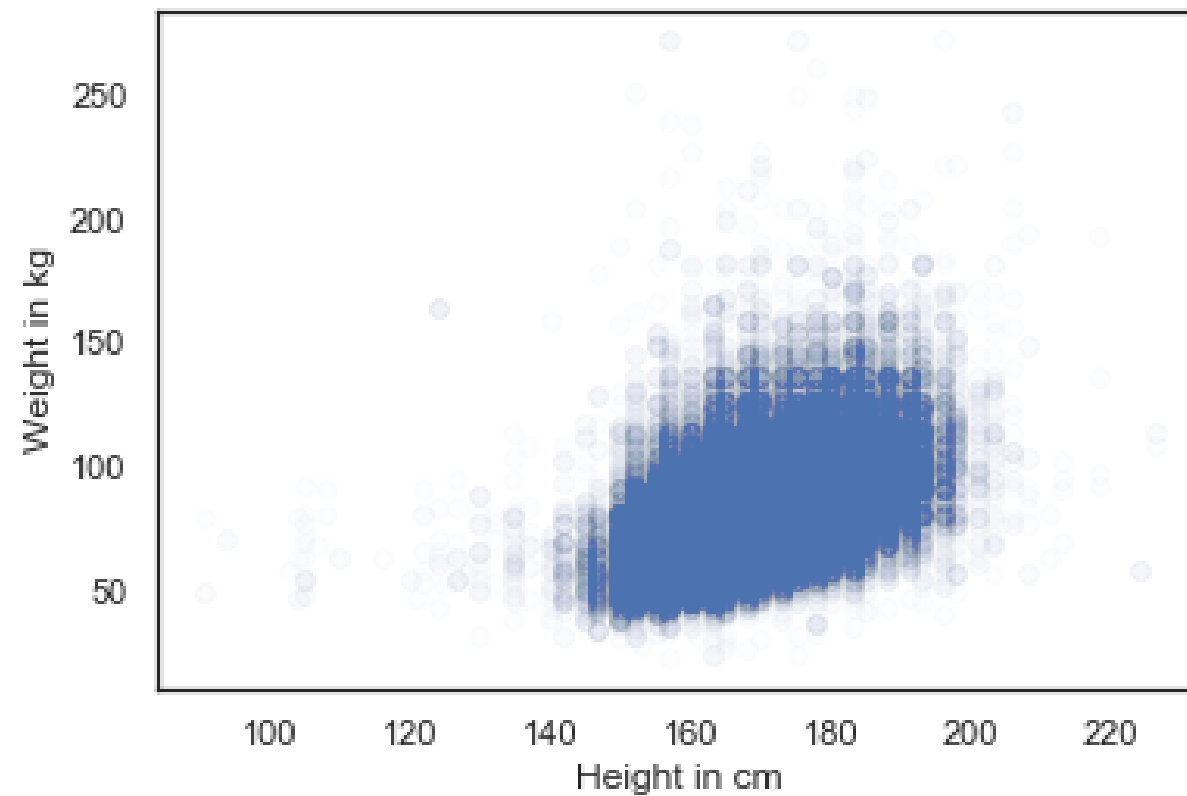
Scatter plot

```
brfss = pd.read_hdf('brfss.hdf5', 'brfss')  
height = brfss['HTM4']  
weight = brfss['WTKG3']  
plt.plot(height, weight, 'o')  
plt.xlabel('Height in cm')  
plt.ylabel('Weight in kg')  
plt.show()
```



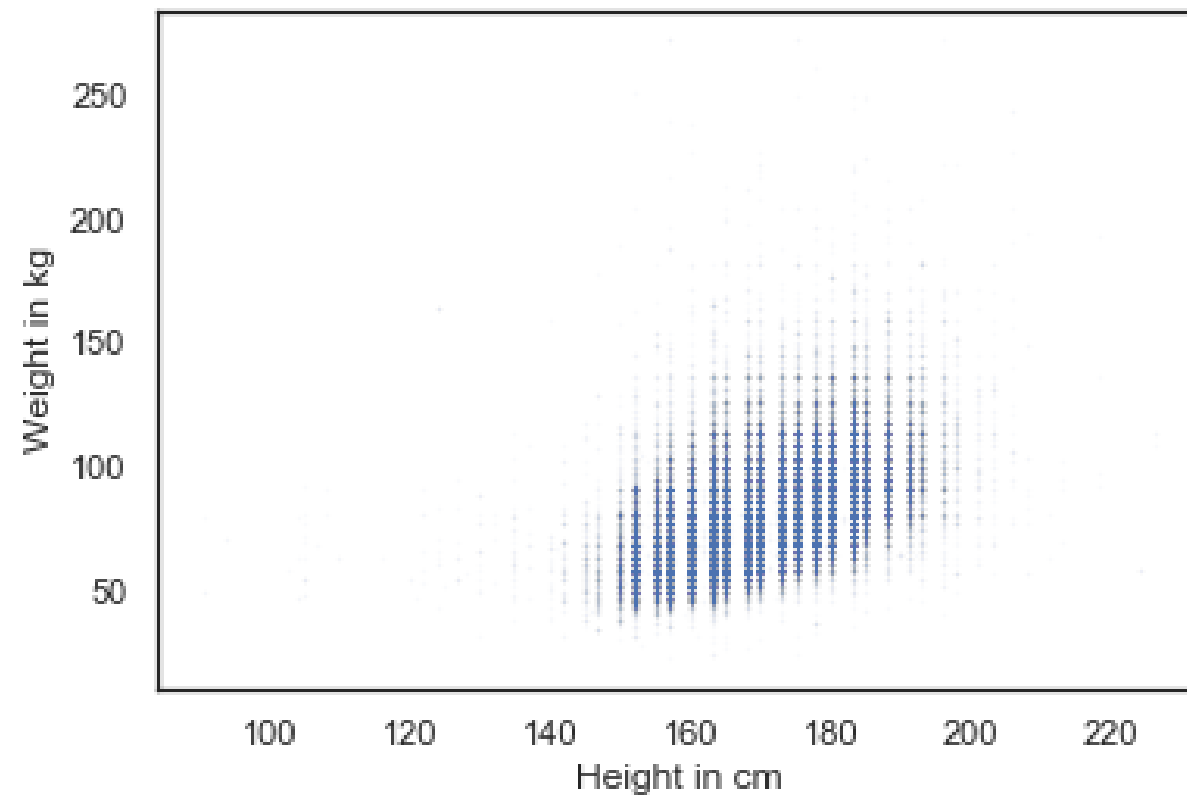
Transparency

```
plt.plot(height, weight, 'o', alpha=0.02)  
plt.show()
```



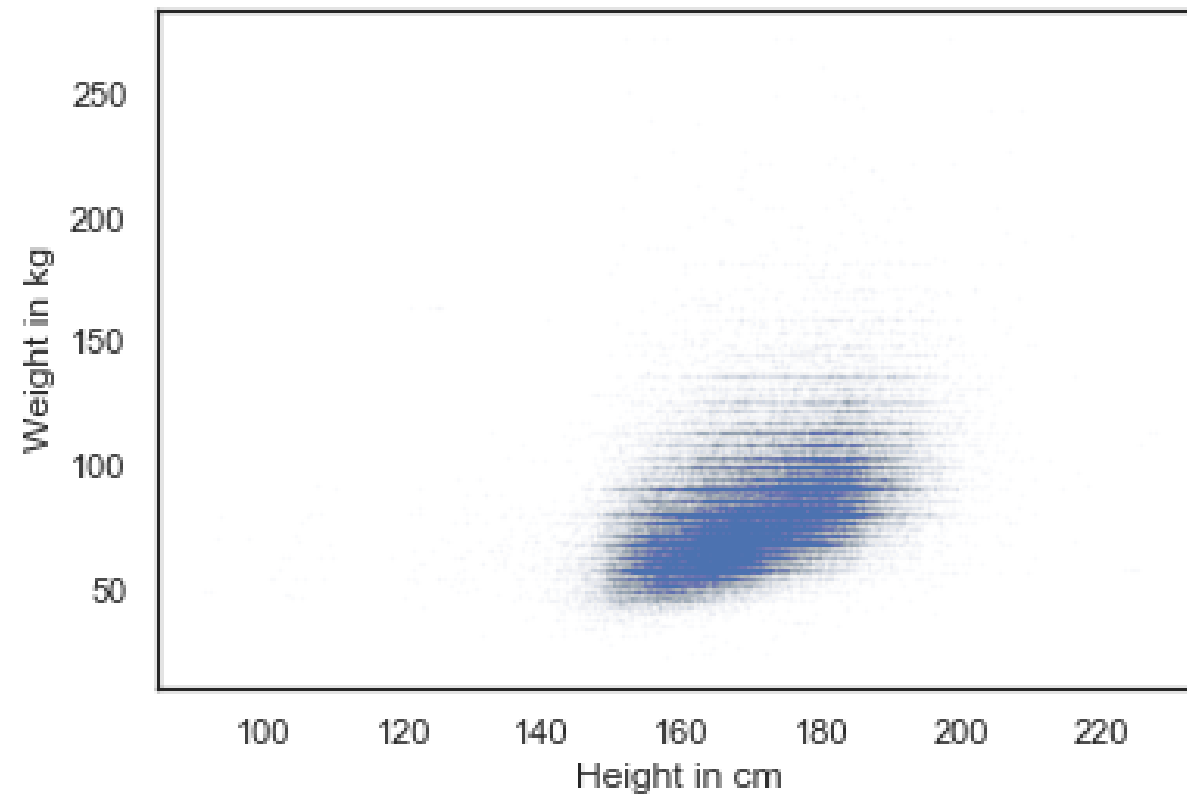
Marker size

```
plt.plot(height, weight, 'o', markersize=1, alpha=0.02)  
plt.show()
```



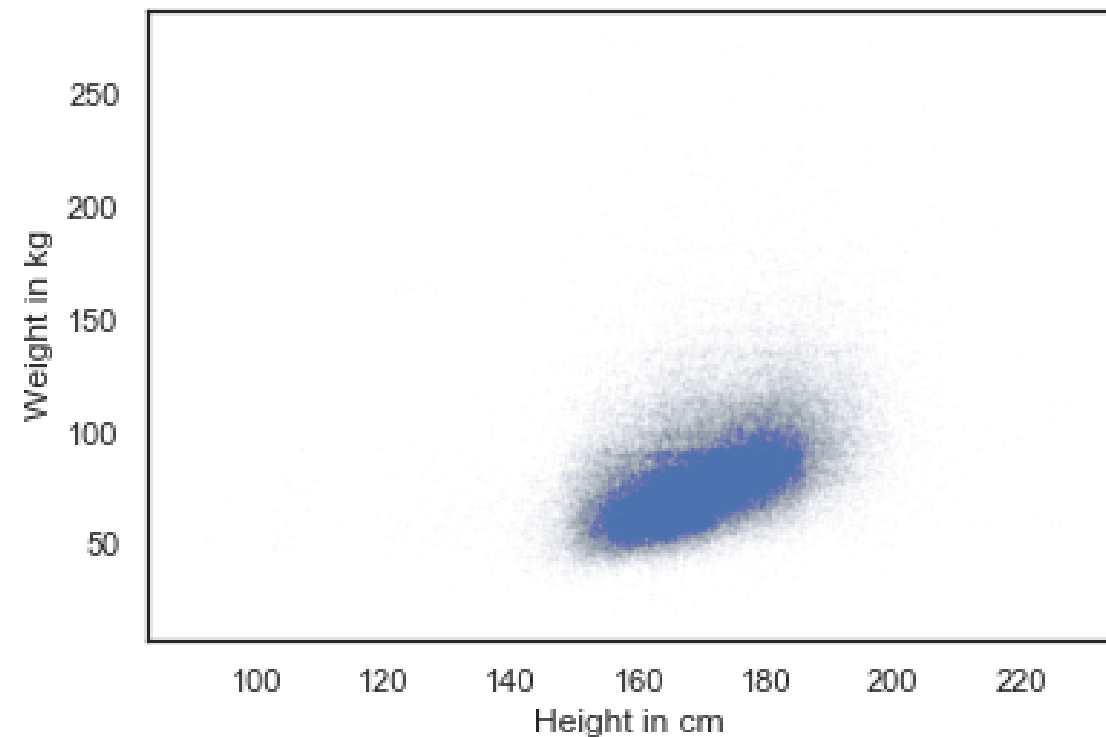
Jittering

```
height_jitter = height + np.random.normal(0, 2, size=len(brfss))  
plt.plot(height_jitter, weight, 'o', markersize=1, alpha=0.02)  
plt.show()
```



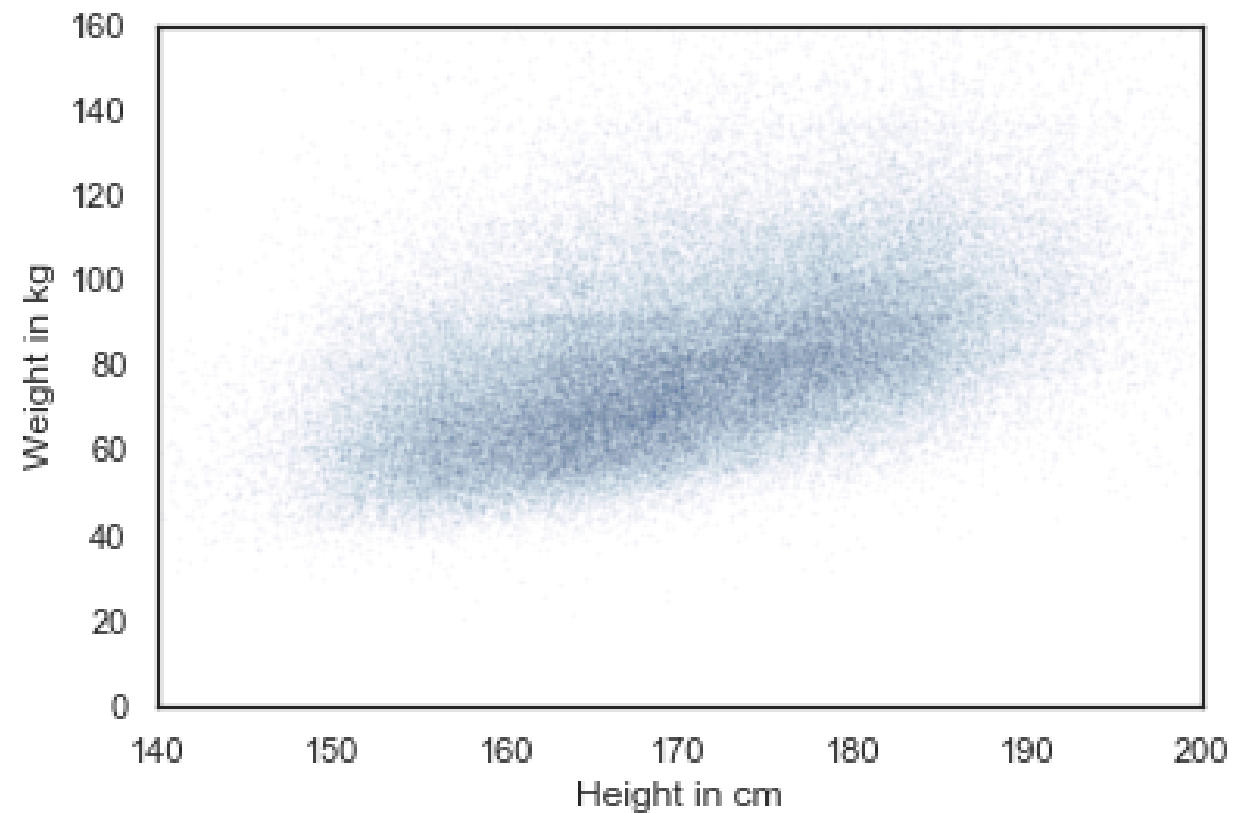
More jittering

```
height_jitter = height + np.random.normal(0, 2, size=len(brfss))  
weight_jitter = weight + np.random.normal(0, 2, size=len(brfss))  
plt.plot(height_jitter, weight_jitter, 'o', markersize=1, alpha=0.01)  
plt.show()
```

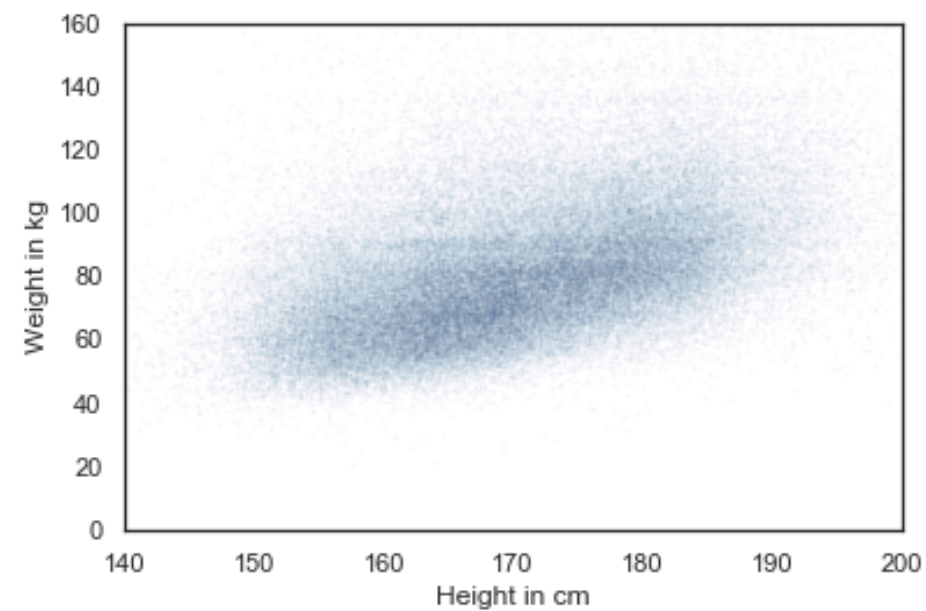
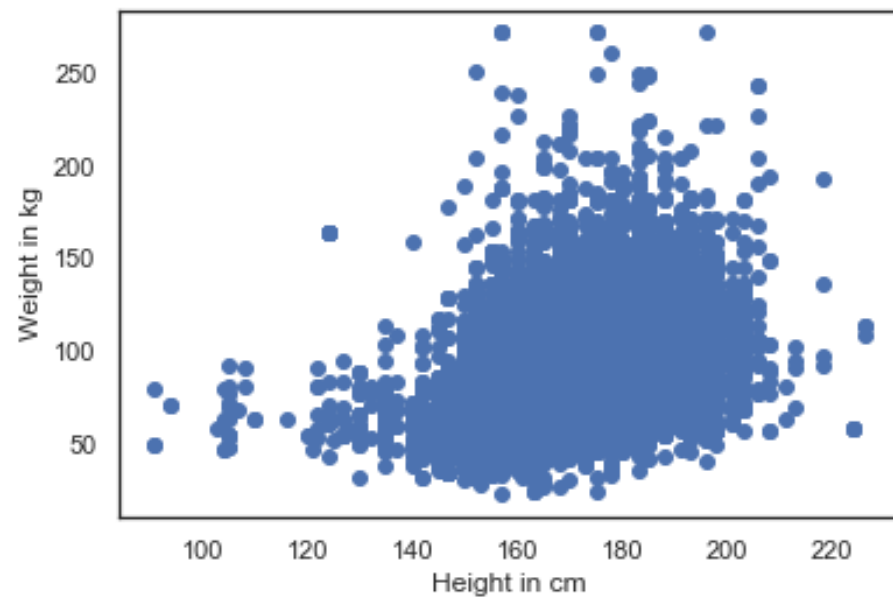


Zoom

```
plt.plot(height_jitter, weight_jitter, 'o', markersize=1, alpha=0.02)  
plt.axis([140, 200, 0, 160])  
plt.show()
```



Before and after



Let's explore!

EXPLORATORY DATA ANALYSIS IN PYTHON

Visualizing relationships

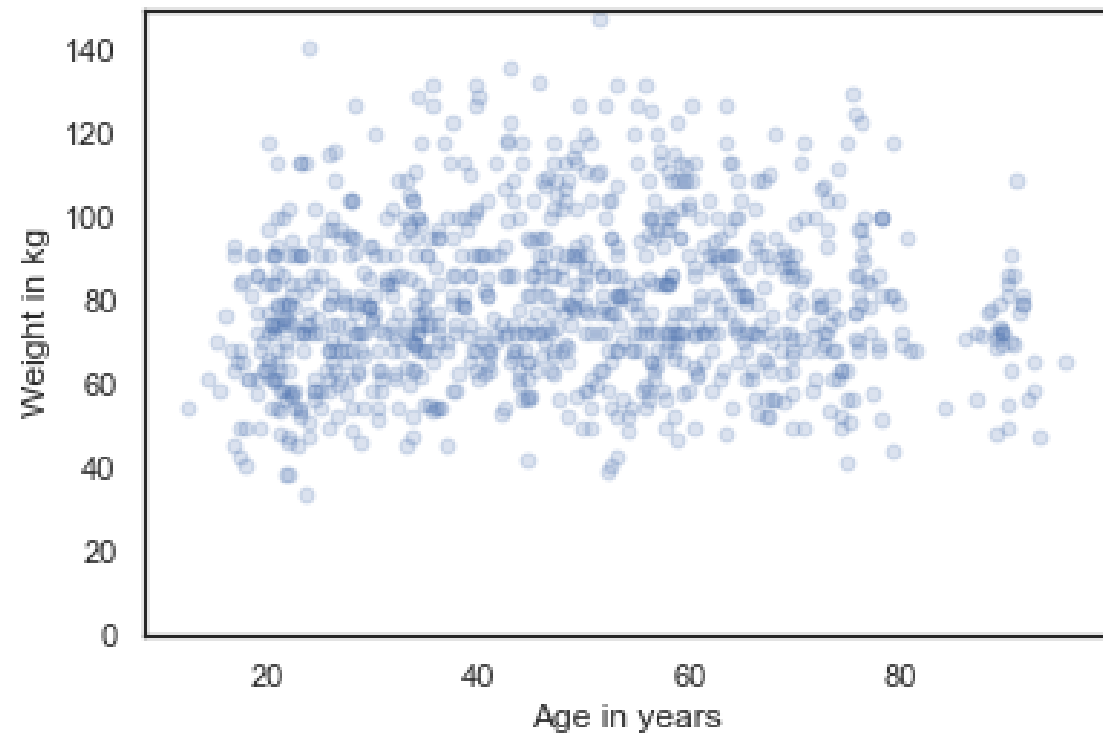
EXPLORATORY DATA ANALYSIS IN PYTHON



Allen Downey
Professor, Olin College

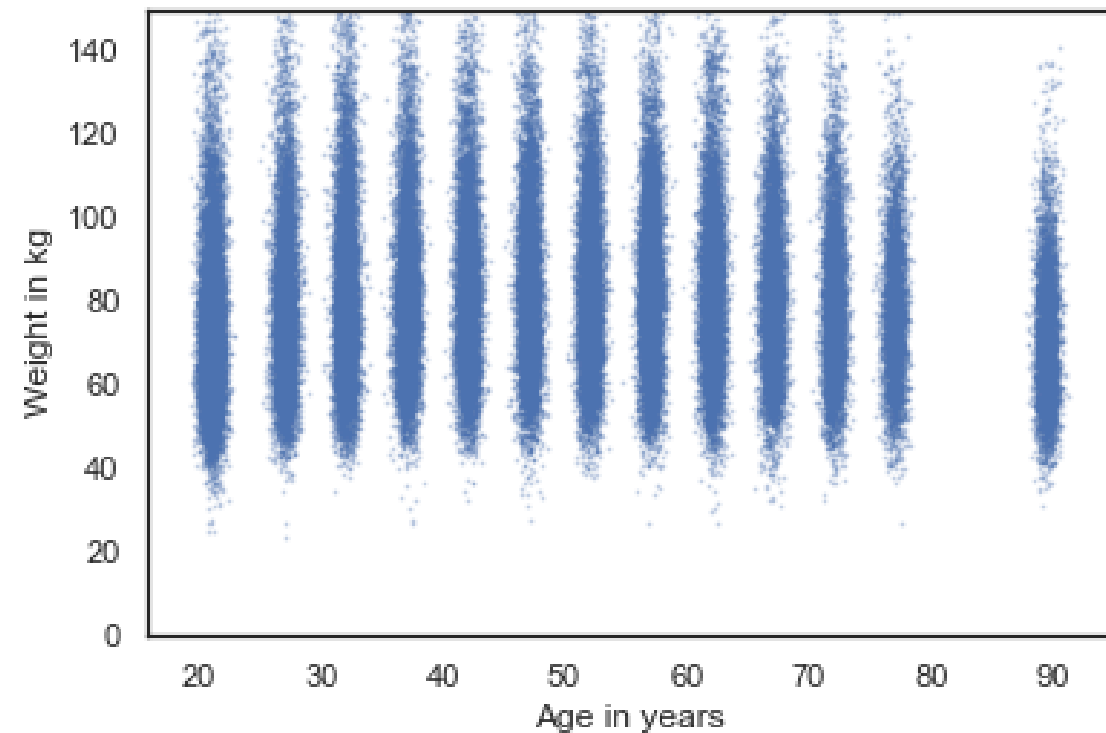
Weight and age

```
age = brfss['AGE'] + np.random.normal(0, 2.5, size=len(brfss))  
weight = brfss['WTKG3']  
plt.plot(age, weight, 'o', markersize=5, alpha=0.2)  
plt.show()
```



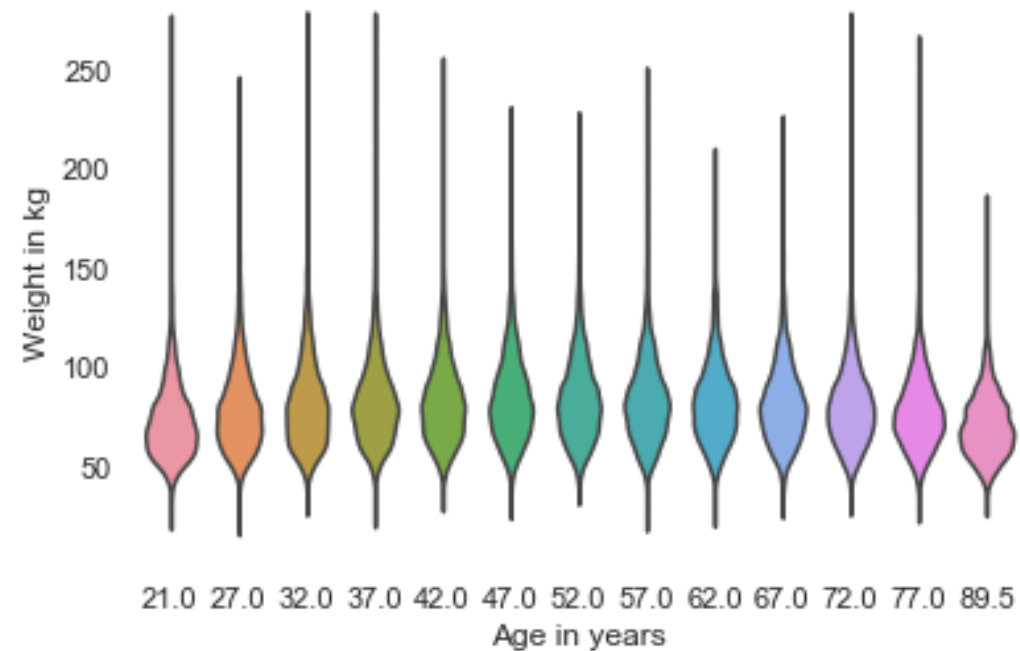
More data

```
age = brfss['AGE'] + np.random.normal(0, 0.5, size=len(brfss))  
weight = brfss['WTKG3'] + np.random.normal(0, 2, size=len(brfss))  
plt.plot(age, weight, 'o', markersize=1, alpha=0.2)  
plt.show()
```



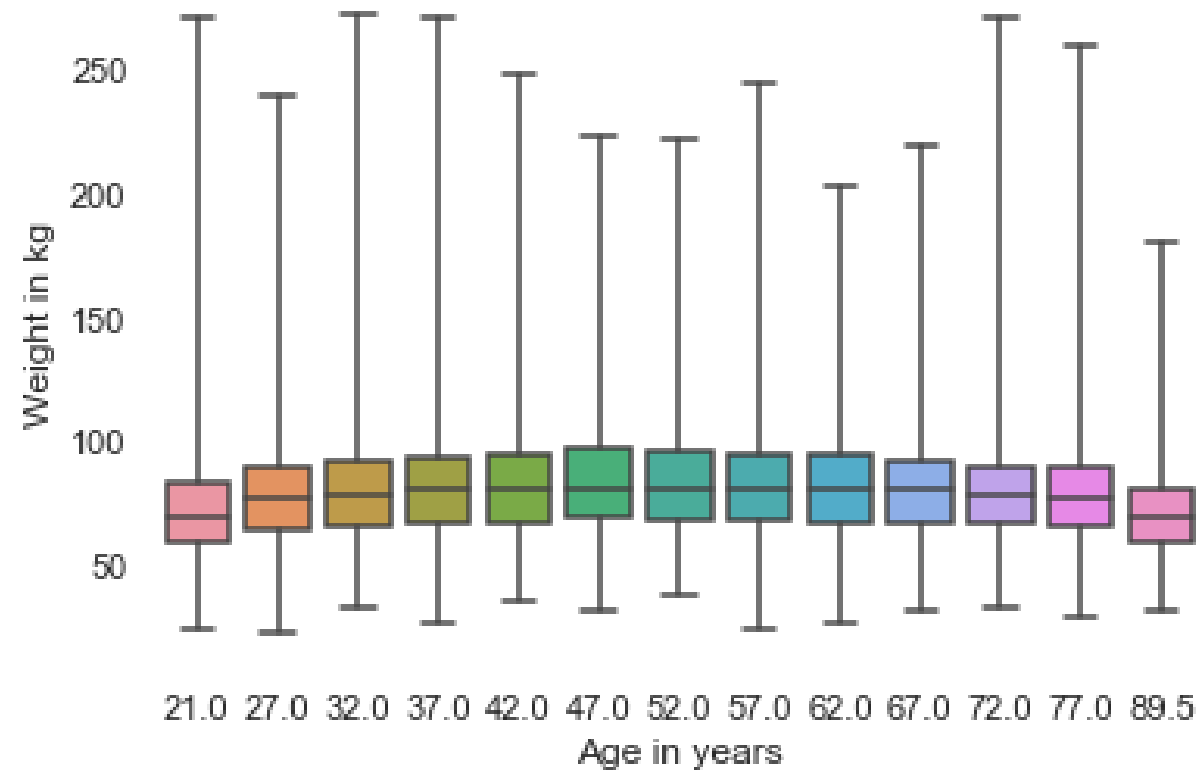
Violin plot

```
data = brfss.dropna(subset=['AGE', 'WTKG3'])  
sns.violinplot(x='AGE', y='WTKG3', data=data, inner=None)  
plt.show()
```



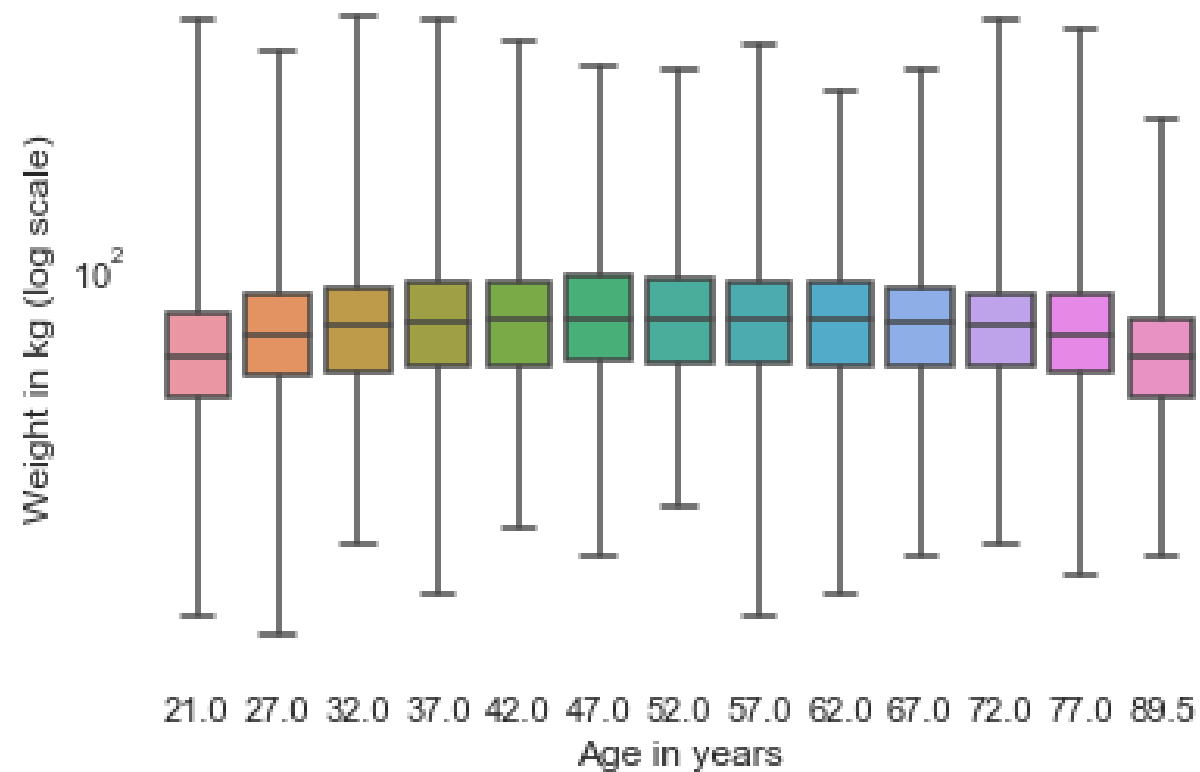
Box plot

```
sns.boxplot(x='AGE', y='WTKG3', data=data, whis=10)  
plt.show()
```



Log scale

```
sns.boxplot(x='AGE', y='WTKG3', data=data, whis=10)  
plt.yscale('log')  
plt.show()
```



Let's practice!

EXPLORATORY DATA ANALYSIS IN PYTHON

Correlation

EXPLORATORY DATA ANALYSIS IN PYTHON



Allen Downey

Professor, Olin College

Correlation coefficient

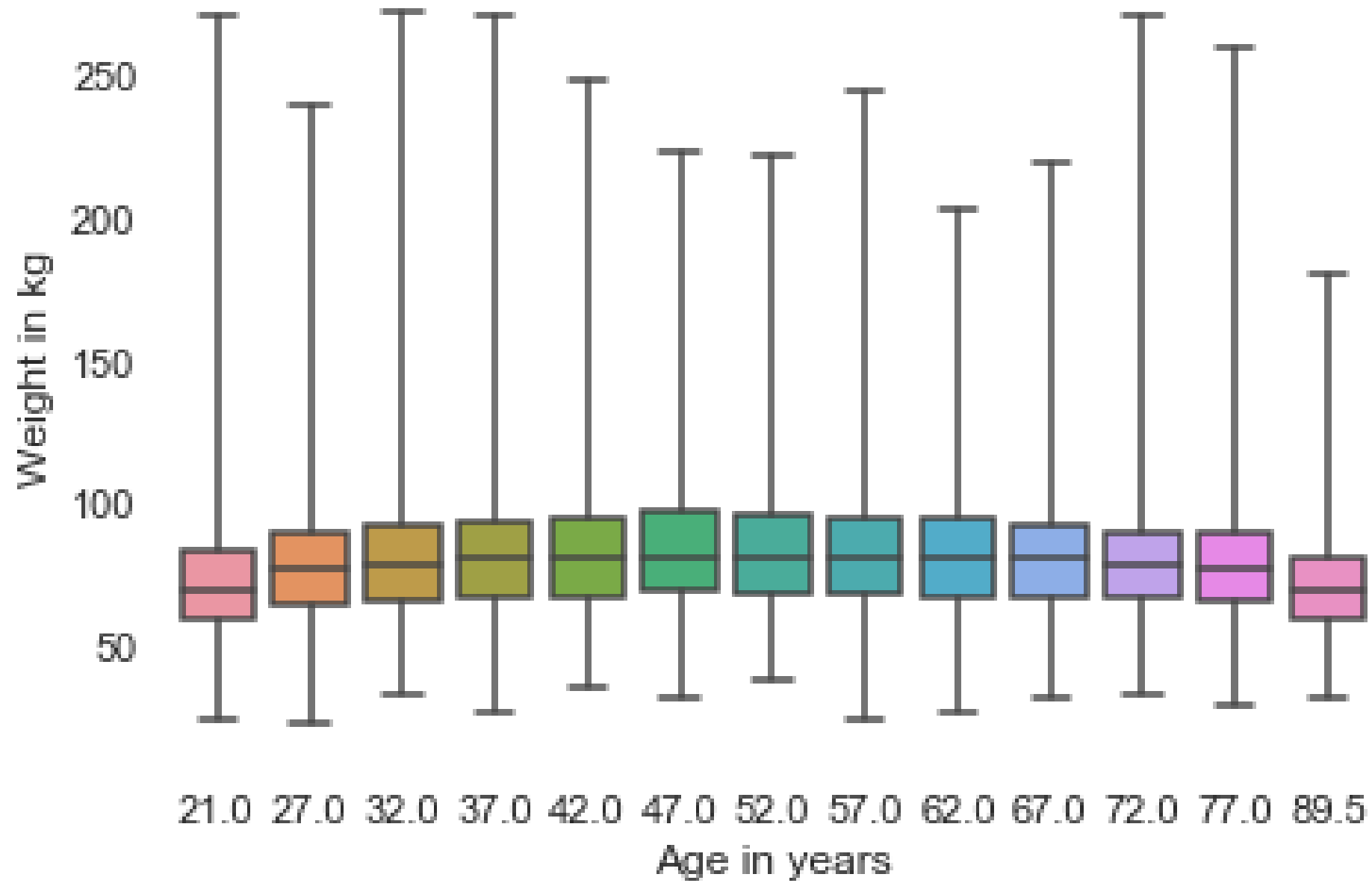
```
columns = ['HTM4', 'WTKG3', 'AGE']  
subset = brfss[columns]
```

```
subset.corr()
```

Correlation matrix

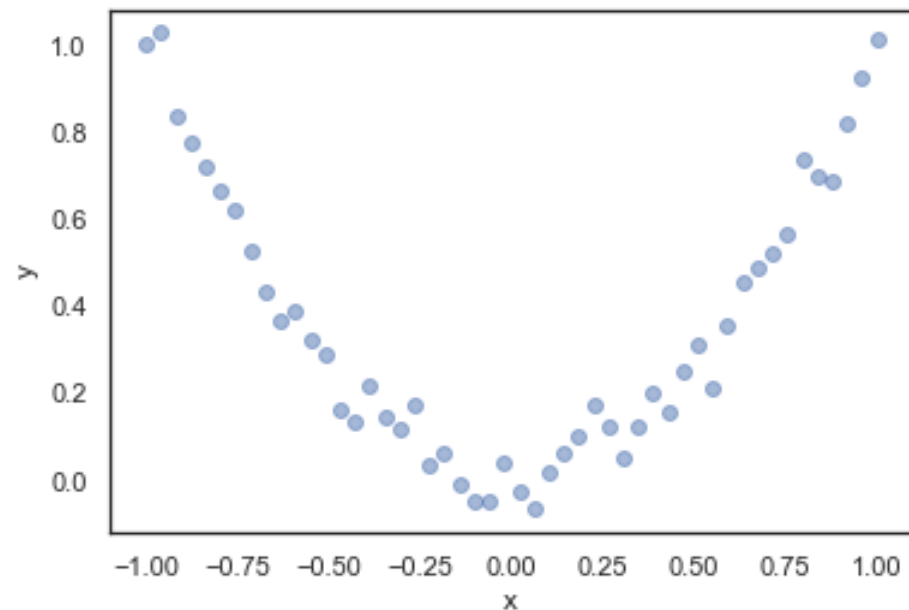
	HTM4	WTKG3	AGE
HTM4	1.000000	0.474203	-0.093684
WTKG3	0.474203	1.000000	0.021641
AGE	-0.093684	0.021641	1.000000

- Height with itself: 1
- Height and weight: 0.47
- Height and age: -0.09
- Weight and age: 0.02



```
xs = np.linspace(-1, 1)
ys = xs**2
ys += normal(0, 0.05, len(xs))
np.corrcoef(xs, ys)
```

```
array([[ 1.          , -0.01111647],
       [-0.01111647,  1.          ]])
```



To demonstrate, I'll generate some fake data: `xs` contains equally-spaced points between -1 and 1.

`ys` is `xs` squared plus some random noise. Here's the scatter plot of `xs` and `ys`.

It's clear that this is a strong relationship; if you are given `x`, you can make a much better guess about `y`.

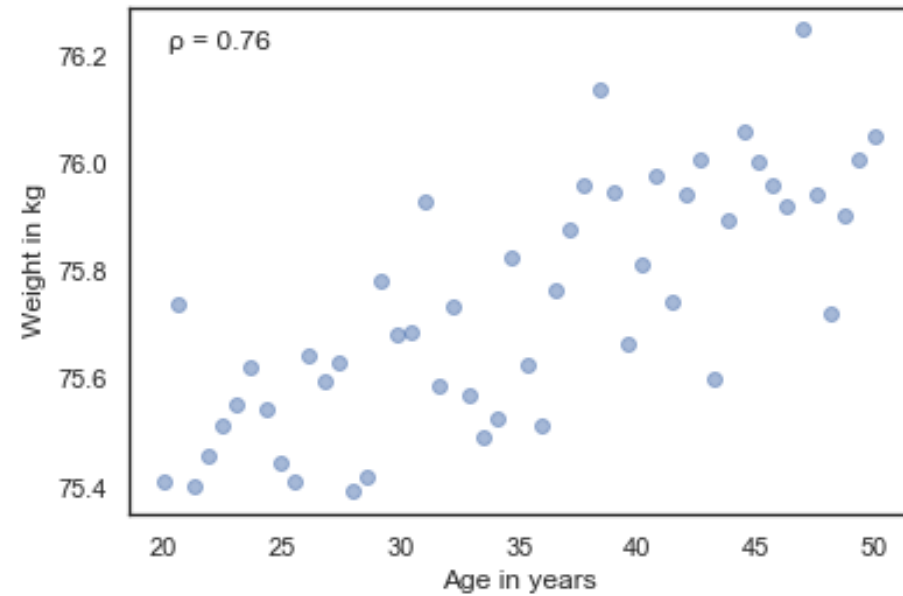
But here's the correlation matrix; the computed correlation is close to 0.

You keep using that word

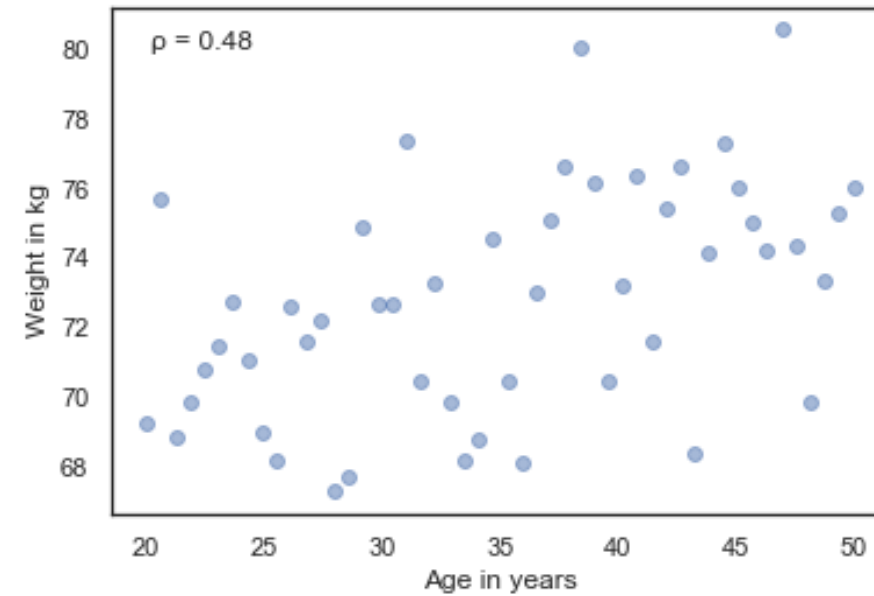
I do not think it means what you think it means

Strength of relationship

Hypothetical #1



Hypothetical #2



Let's practice!

EXPLORATORY DATA ANALYSIS IN PYTHON

Simple regression

EXPLORATORY DATA ANALYSIS IN PYTHON

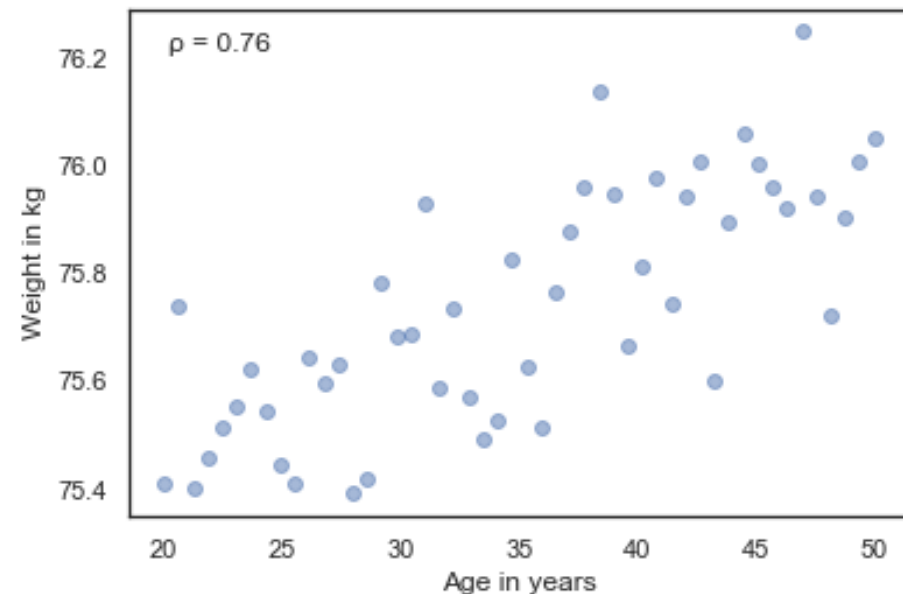


Allen Downey

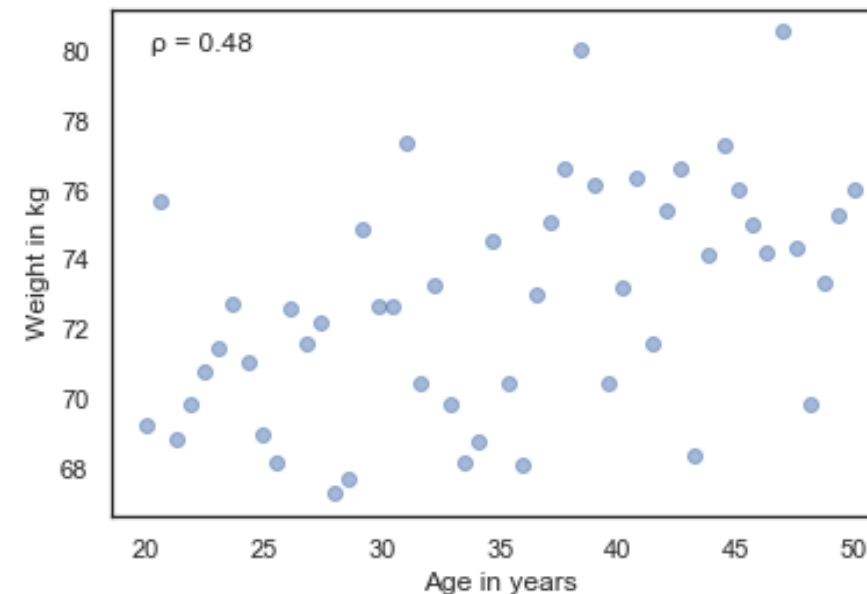
Professor, Olin College

Strength of relationship

Hypothetical #1



Hypothetical #2



Let's look again at an example from the previous lesson, a hypothetical relationship between weight and age.

we have generated two fake datasets to make a point: The one on the left has higher correlation, about 0 point 76 compared to 0 point 48.

But in the one on the left, the average weight gain over 30 years is less than 1 kg; on the right, it is almost 10 kilograms!

In this context, the statistic we probably care about is the slope of the line, not the correlation coefficient.

Strength of effect

```
from scipy.stats import linregress

# Hypothetical 1
res = linregress(xs, ys)
```

```
LinregressResult(slope=0.018821034903244386,
                  intercept=75.08049023710964,
                  rvalue=0.7579660563439402,
                  pvalue=1.8470158725246148e-10,
                  stderr=0.002337849260560818)
```

Strength of effect

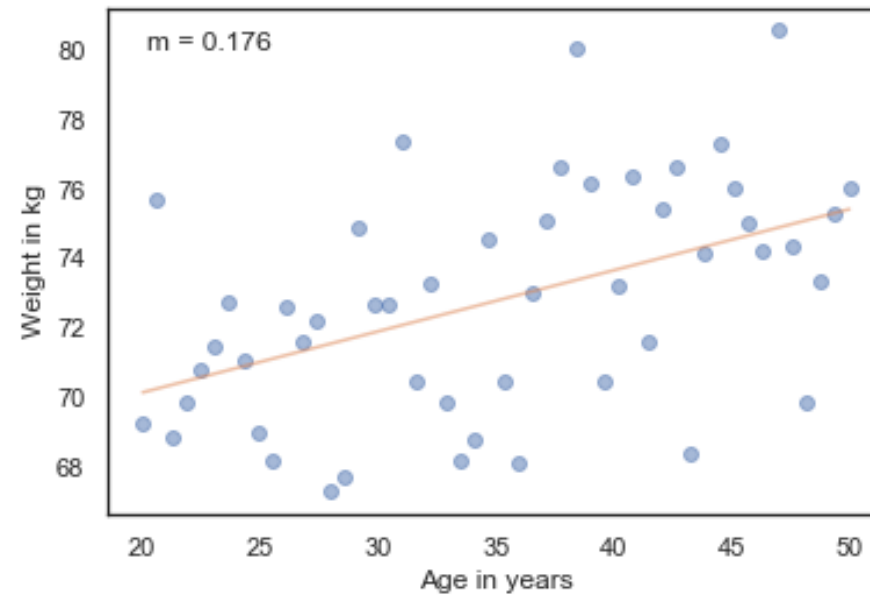
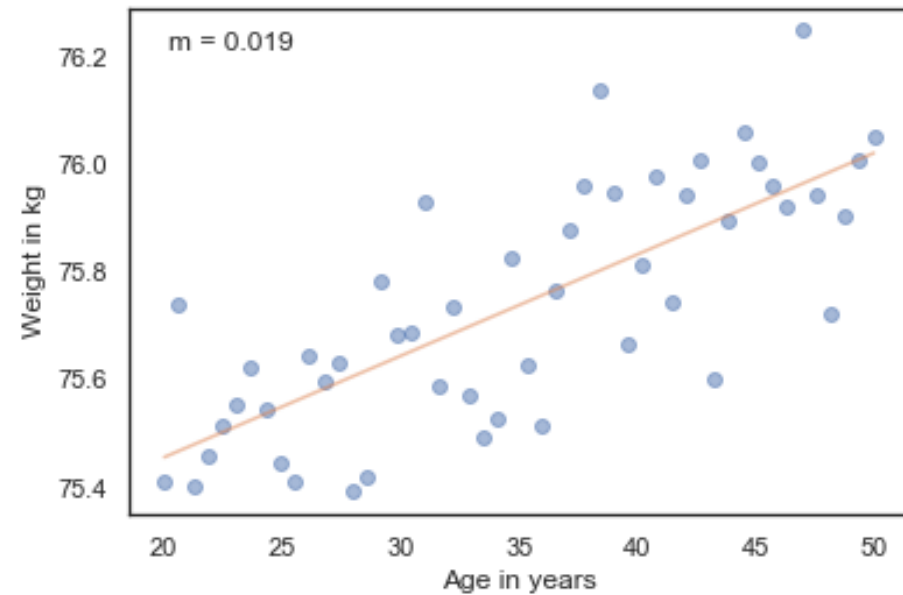
```
# Hypothetical 2  
res = linregress(xs, ys)
```

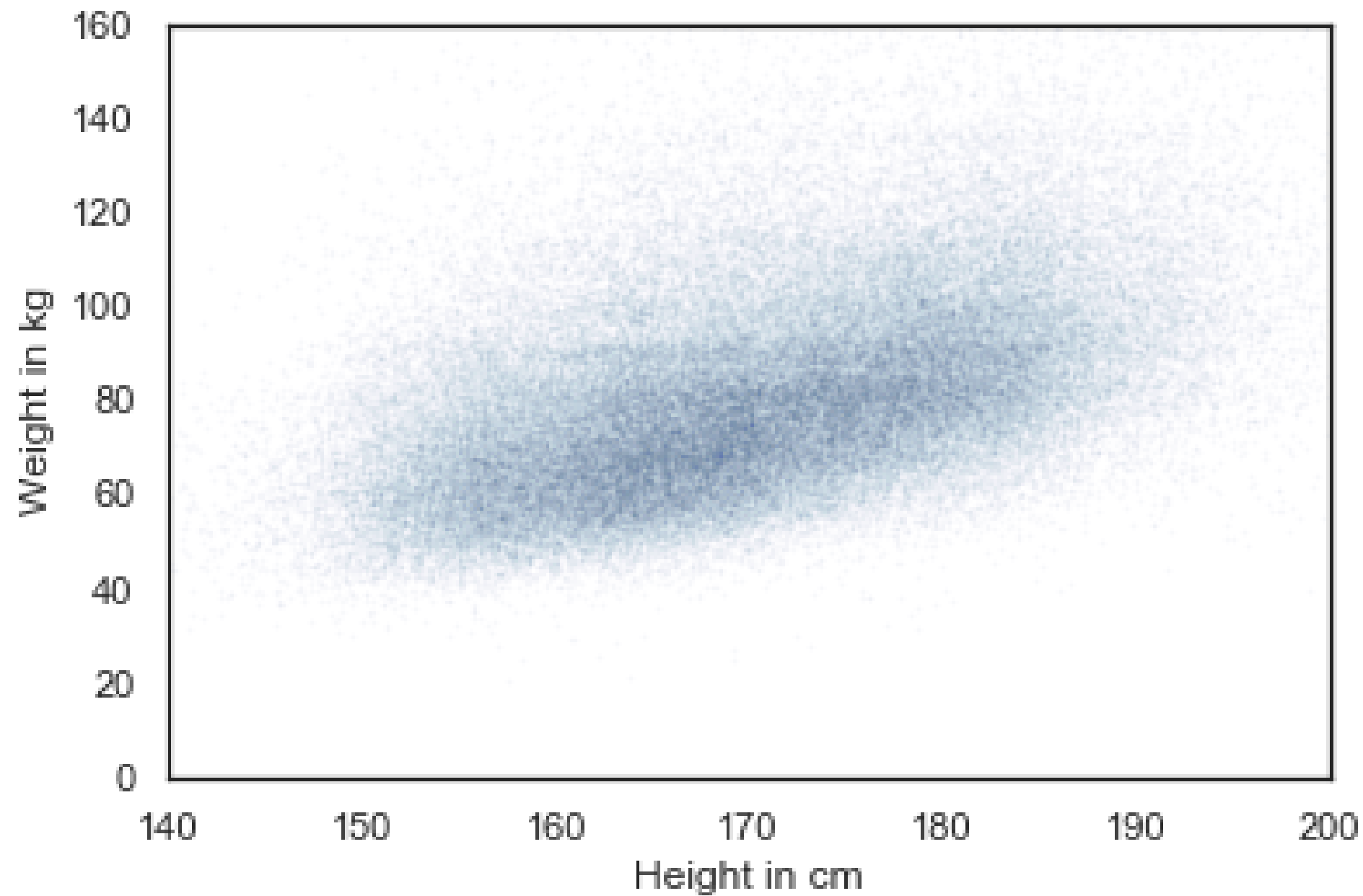
```
LinregressResult(slope=0.17642069806488855,  
                 intercept=66.60980474219305,  
                 rvalue=0.47827769765763173,  
                 pvalue=0.0004430600283776241,  
                 stderr=0.04675698521121631)
```

Regression lines

```
fx = np.array([xs.min(), xs.max()])  
fy = res.intercept + res.slope * fx  
plt.plot(fx, fy, '-')
```

```
fx = ...  
fy = ...  
plt.plot(fx, fy, '-')
```





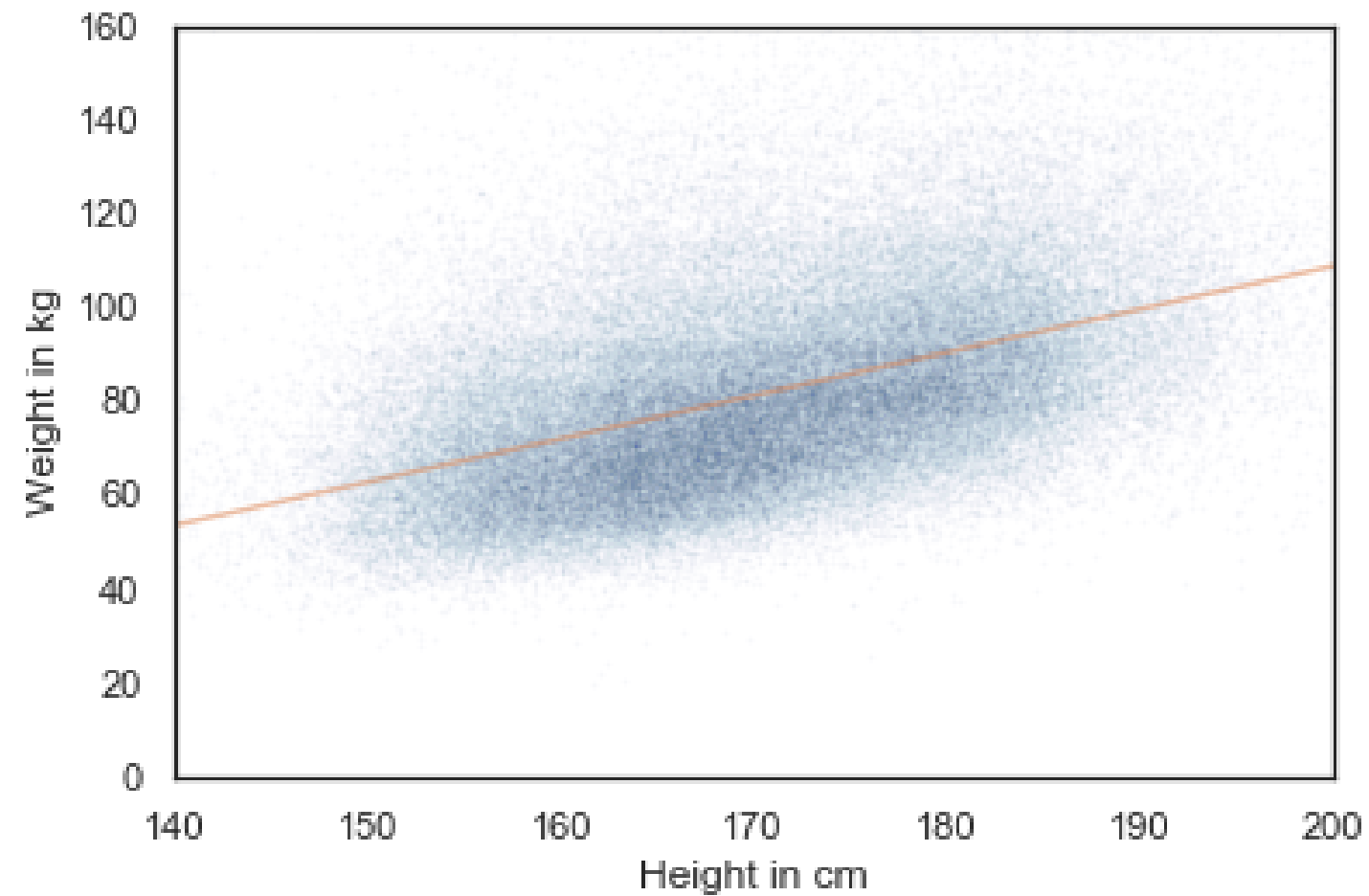
Regression line

```
subset = brfss.dropna(subset=['WTKG3', 'HTM4'])
```

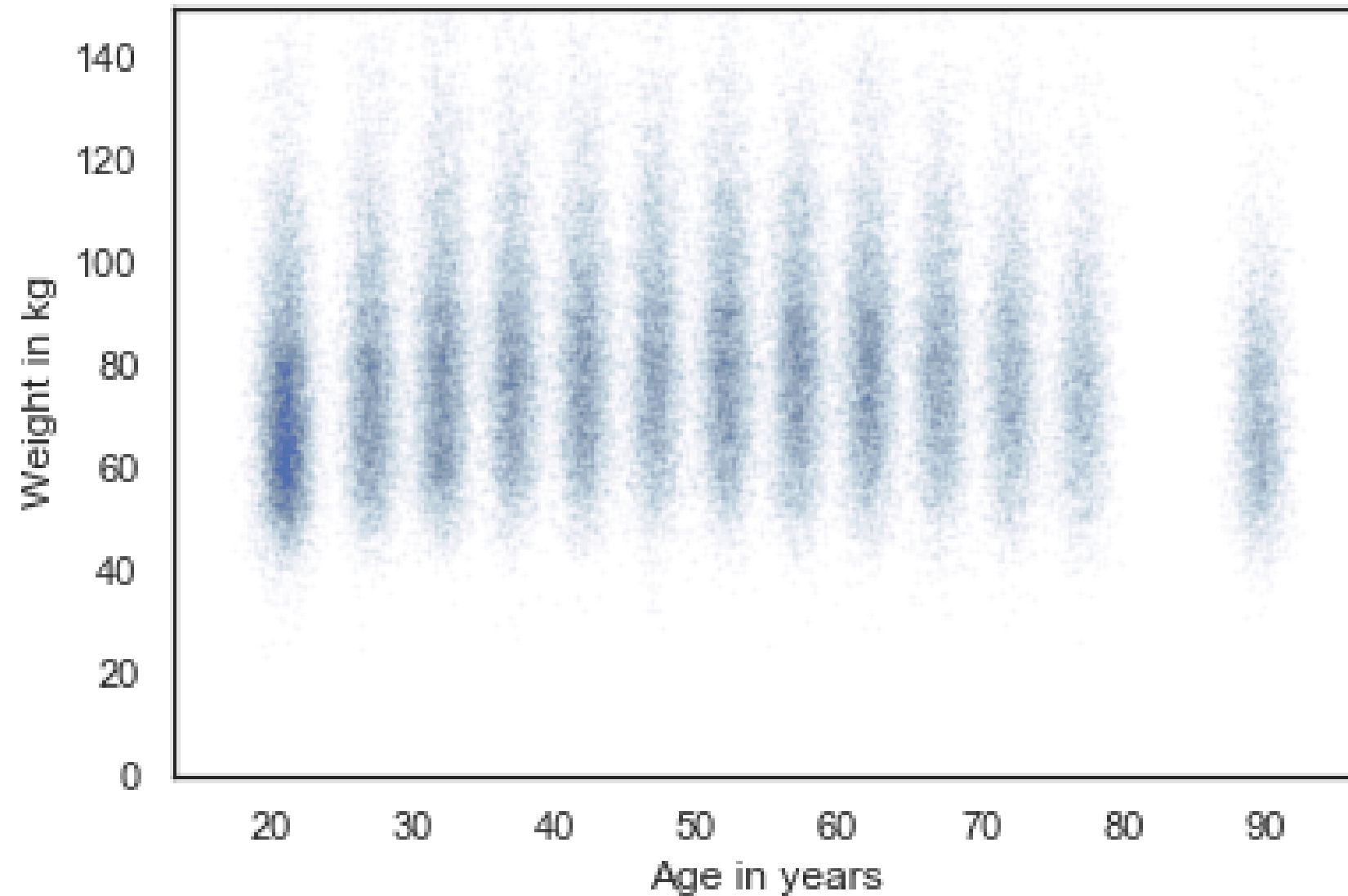
```
xs = subset['HTM4']  
ys = subset['WTKG3']  
res = linregress(xs, ys)
```

```
LinregressResult(slope=0.9192115381848297,  
                  intercept=-75.12704250330233,  
                  rvalue=0.47420308979024584,  
                  pvalue=0.0,  
                  stderr=0.005632863769802998)
```

```
fx = np.array([xs.min(), xs.max()])  
fy = res.intercept + res.slope * fx  
plt.plot(fx, fy, '-')
```



Linear relationships

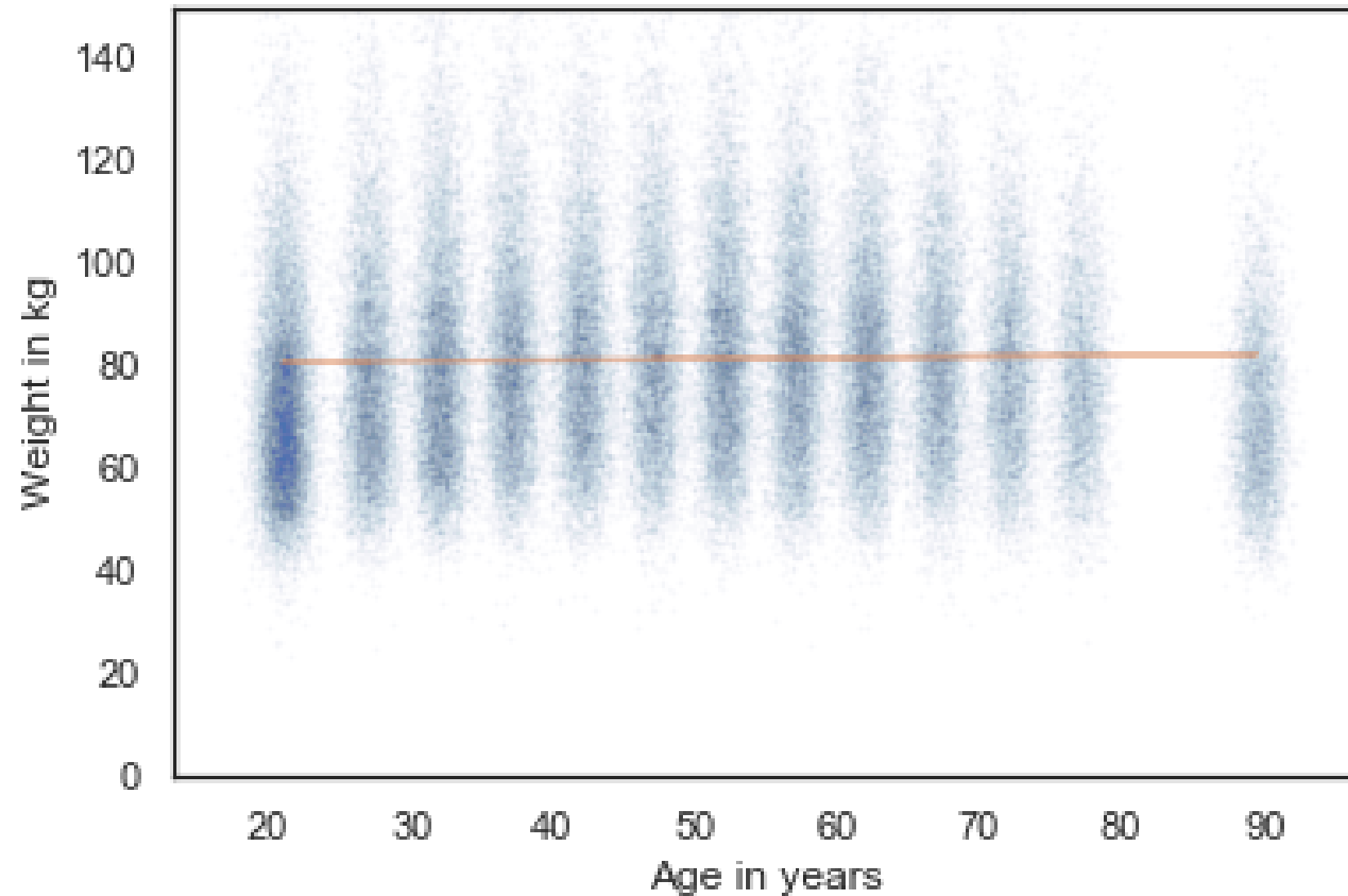


Nonlinear relationships

```
subset = brfss.dropna(subset=['WTKG3', 'AGE'])  
xs = subset['AGE']  
ys = subset['WTKG3']  
  
res = linregress(xs, ys)
```

```
LinregressResult(slope=0.023981159566968724,  
                 intercept=80.07977583683224,  
                 rvalue=0.021641432889064068,  
                 pvalue=4.374327493007566e-11,  
                 stderr=0.003638139410742186)
```

Not a good fit



Let's practice!

EXPLORATORY DATA ANALYSIS IN PYTHON