

# Welcome to the course!

EXPERIMENTAL DESIGN IN PYTHON



Luke Hayden  
Instructor

# Experimental design

## Data

- Allows us to answer questions

## How do we get answers?

- Need rigorous methods

## Approach

- Build hypotheses with exploratory data analysis
- Test hypotheses with statistical tests

# Mapping variables

## Variable types

- Discrete: Finite set of possible values (Ex: True or False)
- Continuous: Any value (Ex: Measurement)

## Mapping

- X or Y axes
- Change color with `fill` or `color` arguments

# Making plots with plotnine

1. Call `ggplot()` function and give it a DataFrame
2. Assign mapping of variables with `aes()`
3. Specify a geometry

```
import plotnine as p9

(p9.ggplot([pandas DataFrame]) +

p9.aes(
    x='variable to put on X-axis',
    y='variable to put on Y-axis',
    color='variable ')+

p9.geom_point()
)
```

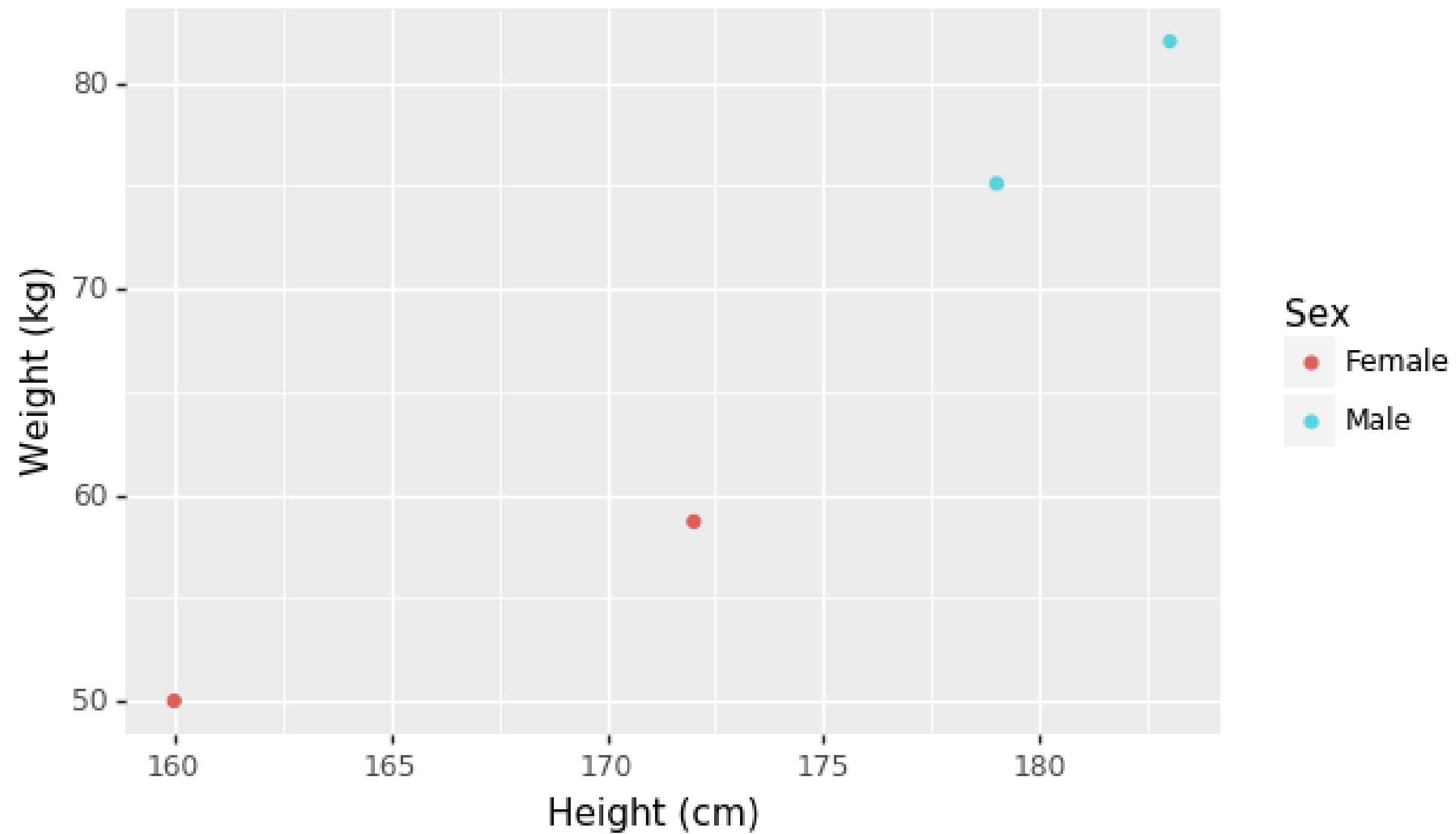
# Scatter plot

`geom_point()`

```
import plotnine as p9
import pandas as pd

df = pd.DataFrame(data= {'Sex': ["Male", "Male", "Female", "Female"] ,
                          "Height (cm)": [183, 179, 160, 172],
                          "Weight (kg)": [82, 75.1, 50, 58.7]})

print(p9.ggplot(df)+ p9.aes(x='Height (cm)', y='Weight (kg)', color='Sex')+ p9.geom_point())
```



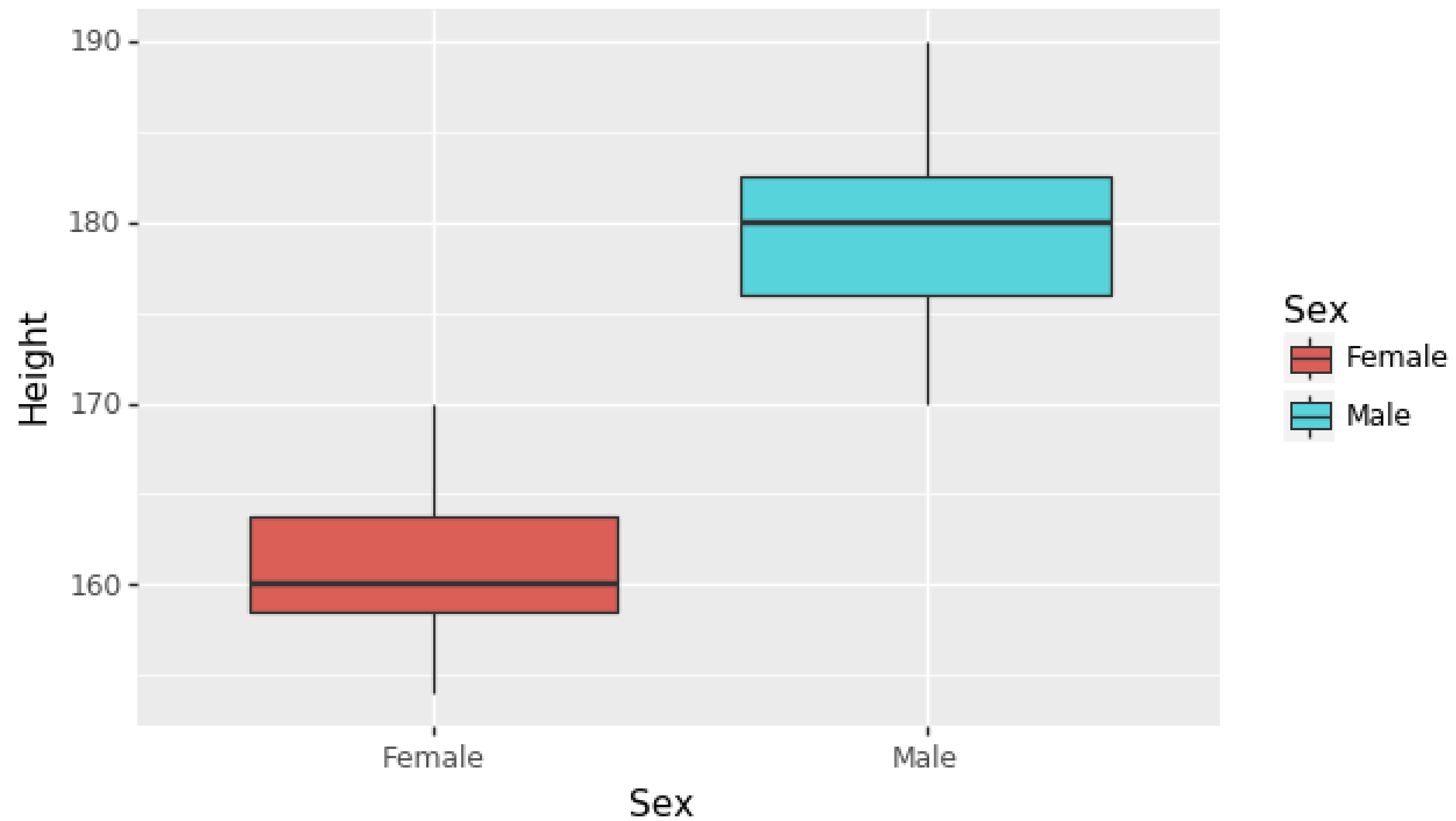
# Boxplot

`geom_boxplot()`

```
import plotnine as p9
import pandas as pd

df = pd.DataFrame(data= {'Sex': ["Male", "Male", "Male", "Male", "Male", "Male",
                                "Female", "Female", "Female", "Female", "Female", "Female"],
                        "Height": [183, 179, 190, 181, 170, 175,
                                   160, 165, 158, 154, 170, 160]})

(p9.ggplot(df)+ p9.aes(x='Sex', y='Height', fill='Sex')+ p9.geom_boxplot())
```

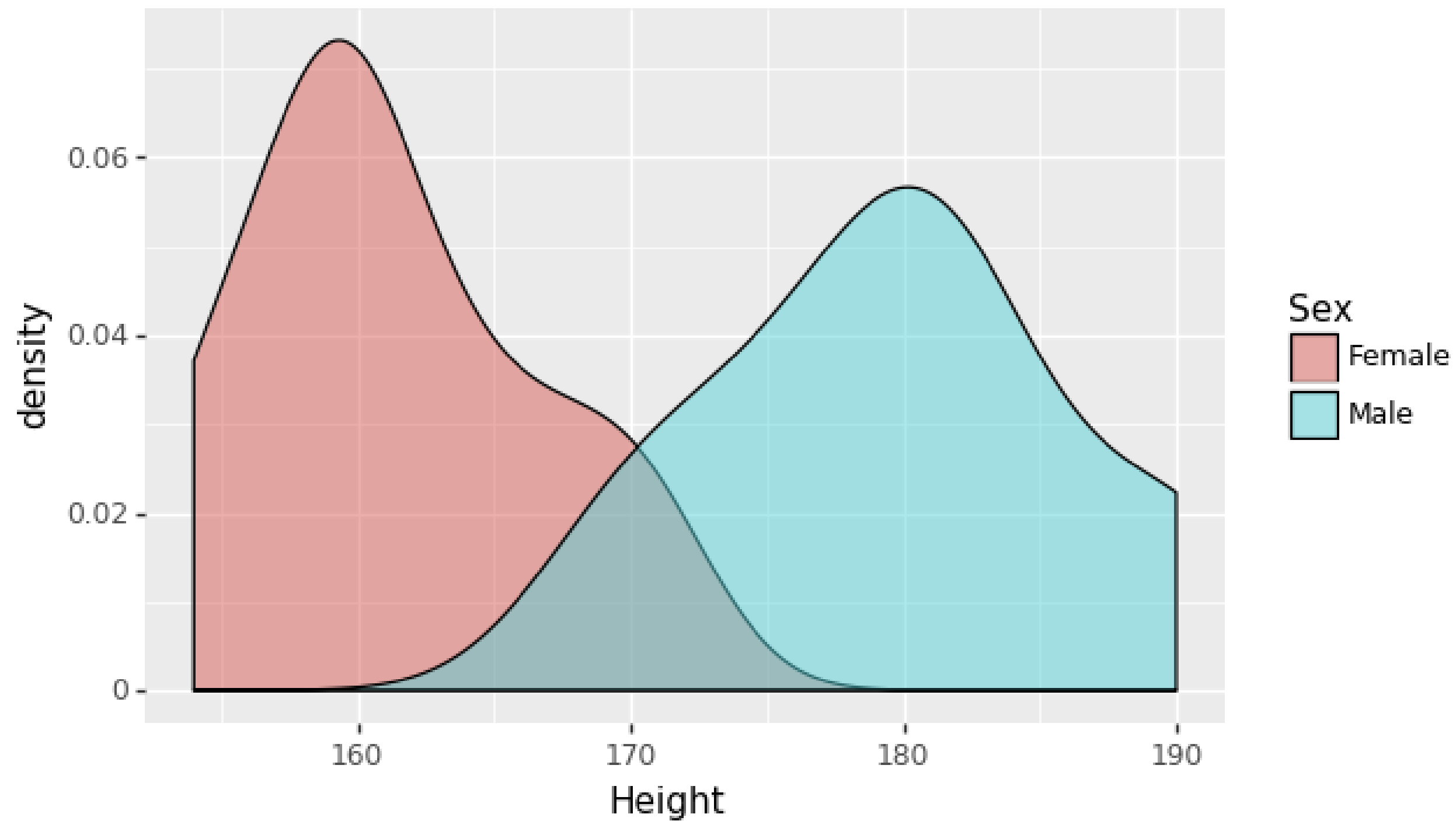




# Density plot

`geom_density()`

```
import plotnine as p9
import pandas as pd
df = pd.DataFrame(data= {'Sex': ["Male", "Male", "Male", "Male", "Male", "Male",
                                "Female", "Female", "Female", "Female", "Female", "Female"],
                        "Height": [183, 179, 190, 181, 170, 175,
                                  160, 165, 158, 154, 170, 160]})
(p9.ggplot(df)+ p9.aes(x='Height', fill='Sex') + p9.geom_density(alpha=0.5))
```



# Let's practice!

EXPERIMENTAL DESIGN IN PYTHON

# Our first hypothesis test - Student's t-test

EXPERIMENTAL DESIGN IN PYTHON



Luke Hayden  
Instructor

# From observed pattern to reliable result

## Data contains patterns

- Some expected
- Others surprising
- Random variation also

## Dealing with this

- How do we go from observation to result?

# Are these groups different?

Weights of two groups of adults

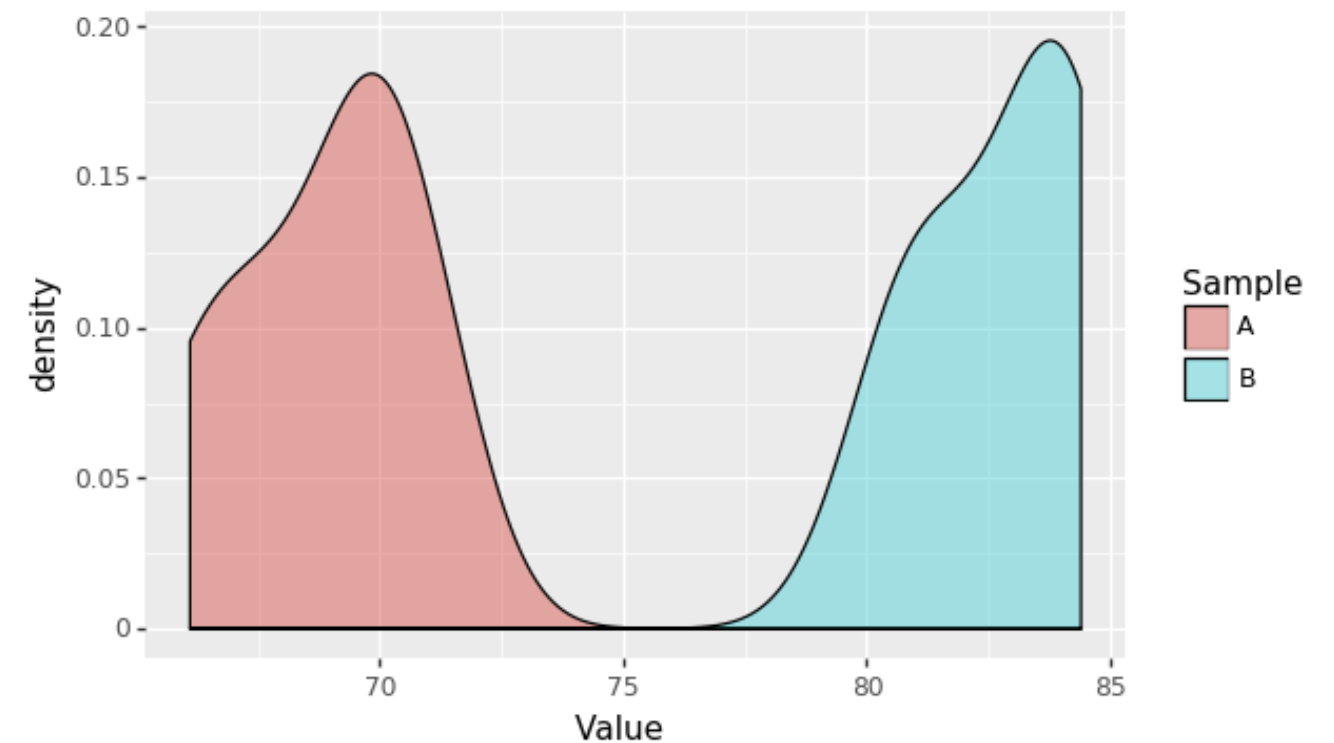
Sample A:

```
[66.1, 69.8, 67.7, 69.6, 71.1]
```

Sample B:

```
[83.7, 81.5, 80.6, 83.9, 84.4]
```

```
(p9.ggplot(df)+  
p9.aes('Value', fill='Sample')+  
p9.geom_density(alpha=0.5))
```



# Two hypotheses

## Null hypothesis

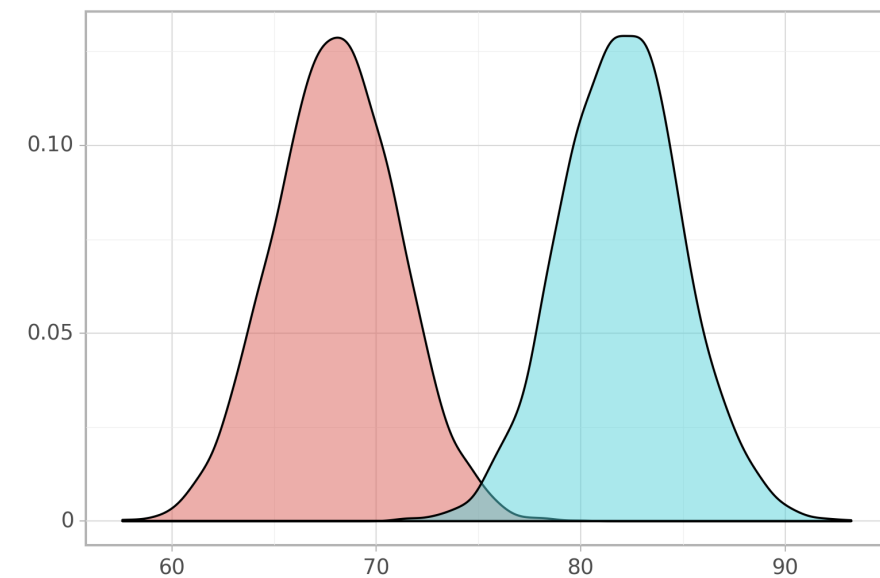
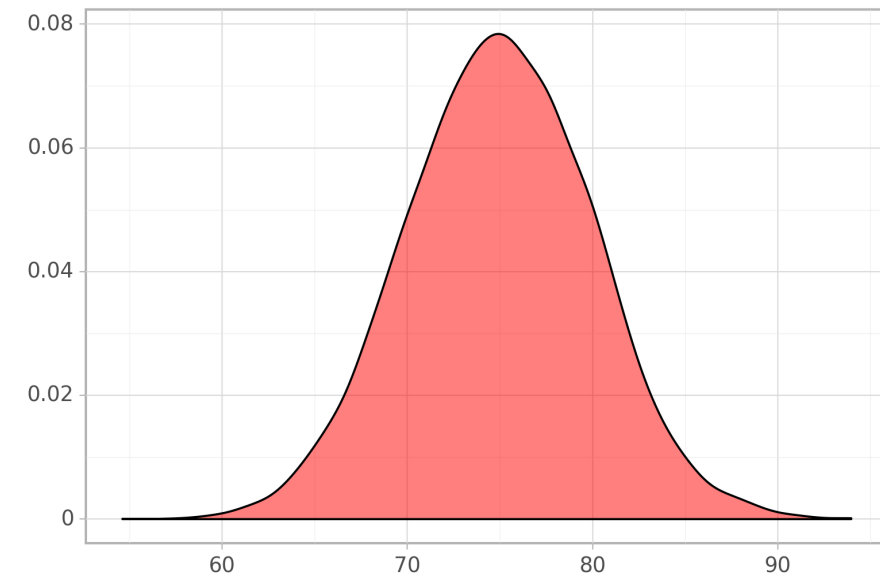
$$A = B$$

- Observed patterns are the product of random chance

## Alternative hypothesis

$$A \neq B$$

- Difference between samples represents a real difference between the populations



# Some statistical terms

## p-value

- Likelihood of pattern under **null hypothesis**

## alpha

- Crucial threshold of p-value
- Usually  *$\alpha < 0.05$ : reject null hypothesis*



# Student's t-test

- Invented by William Sealy Gosset
- Two basic types:

**One-sample:** Mean of population different from a given value?

**Two-sample:** Two means equal?

## Coding a t-test

```
from scipy import stats
```

```
stats.ttest_ind(Sample_A, Sample_B)
```

# Implementing a one-sample t-test

```
from scipy import stats

Sample_A = df[df.Sample == "A"]

t_result = stats.ttest_1sample(Sample_A, 65)

alpha = 0.05

if (t_result[1] < alpha):
    print("mean(A) != 65")
```

```
mean(A) != 65
```

# Implementing a two-sample t-test

```
from scipy import stats

Sample_A = df[df.Sample == "A"]
Sample_B = df[df.Sample == "B"]

t_result = stats.ttest_ind(Sample_A, Sample_B)

alpha = 0.05

if (t_result[1] < alpha):
    print("A and B are different!")
```

```
A and B are different!
```

# Now let's try it out!

EXPERIMENTAL DESIGN IN PYTHON

# Testing proportion and correlation

EXPERIMENTAL DESIGN IN PYTHON



Luke Hayden  
Instructor

# Hypothesis tests

## t-test:

- Compare means of continuous variables

## Chi-square:

- Examine proportions of discrete categories

## Fisher exact test:

- Examine proportions of discrete categories

## Pearson test:

- Examine if continuous variables are correlated

# Chi-square

Test distinguishes between:

**Null hypothesis:**

- Observed outcomes fit distribution
- *coin is not biased*

**Alternative hypothesis:**

- Observed outcomes doesn't fit distribution
- *coin is biased*

**Example**

Coin flipped 30 times

Expected: 15 heads, 15 tails

Observed: 24 heads, 6 tails

*Expected outcomes significantly different from expected?*

# Implementing a simple Chi-square test

```
from scipy import stats

coins = df['Flip'].value_counts()

chi = stats.chisquare(coins)

print(chi)
```

```
Power_divergenceResult(statistic=10.8, pvalue=0.0010150009471130682)
```



# Fisher exact test

Two-sample version of Chi-square test

Test distinguishes between:

**Null hypothesis:**

- Two samples have same distribution of outcomes

**Alternative hypothesis:**

- Two samples have different distribution of outcomes

**Example**

Two coins each flipped 30 times

*Expected outcomes significantly differ?*

*Are these two discrete variables related?*

# Implementing a Fisher exact test

```
from scipy import stats
import pandas as pd
table = pd.crosstab(df.Coin, df.Flip)
print(table)
```

```
Flip  heads  tails
Coin
1       22     8
2       17    13
```

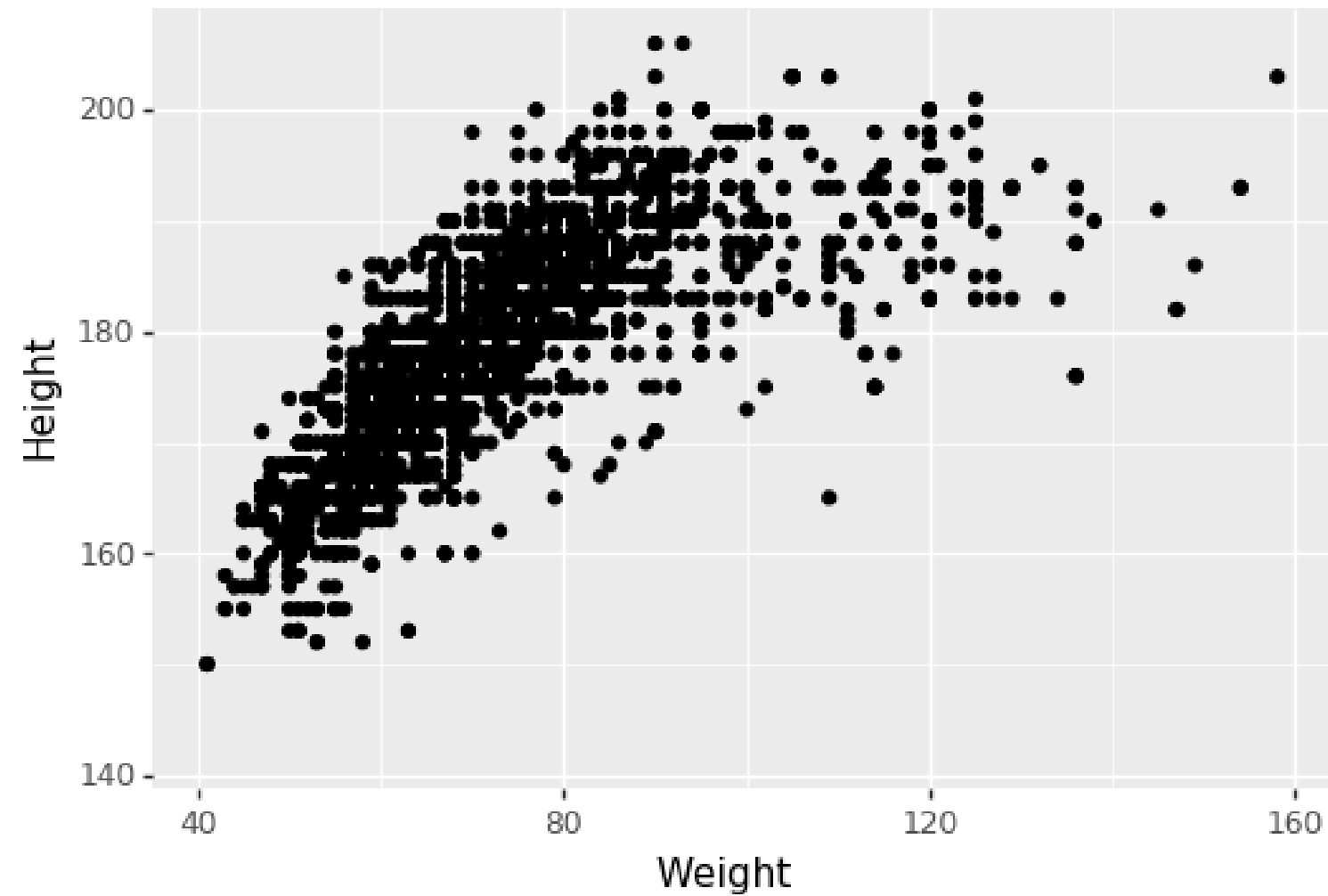
# Implementing a Fisher exact test

```
chi = stats.fisher_exact(table, alternative='two-sided')  
print(chi[1])
```

```
0.421975381019902
```

# Correlation

```
import plotnine as p9
(p9.ggplot(olyAmericans)+ p9.aes(x='Weight',y='Height')+ p9.geom_point())
```



# Pearson test for correlation

```
from scipy import stats
import pandas as pd

pearson = stats.pearsonr(df.Weight, df.Height)

print(pearson)
```

```
(0.7922545330545416, 0.0)
```

(Correlation coefficient, p-value)

# Let's practice!

EXPERIMENTAL DESIGN IN PYTHON