

# Creating line plots

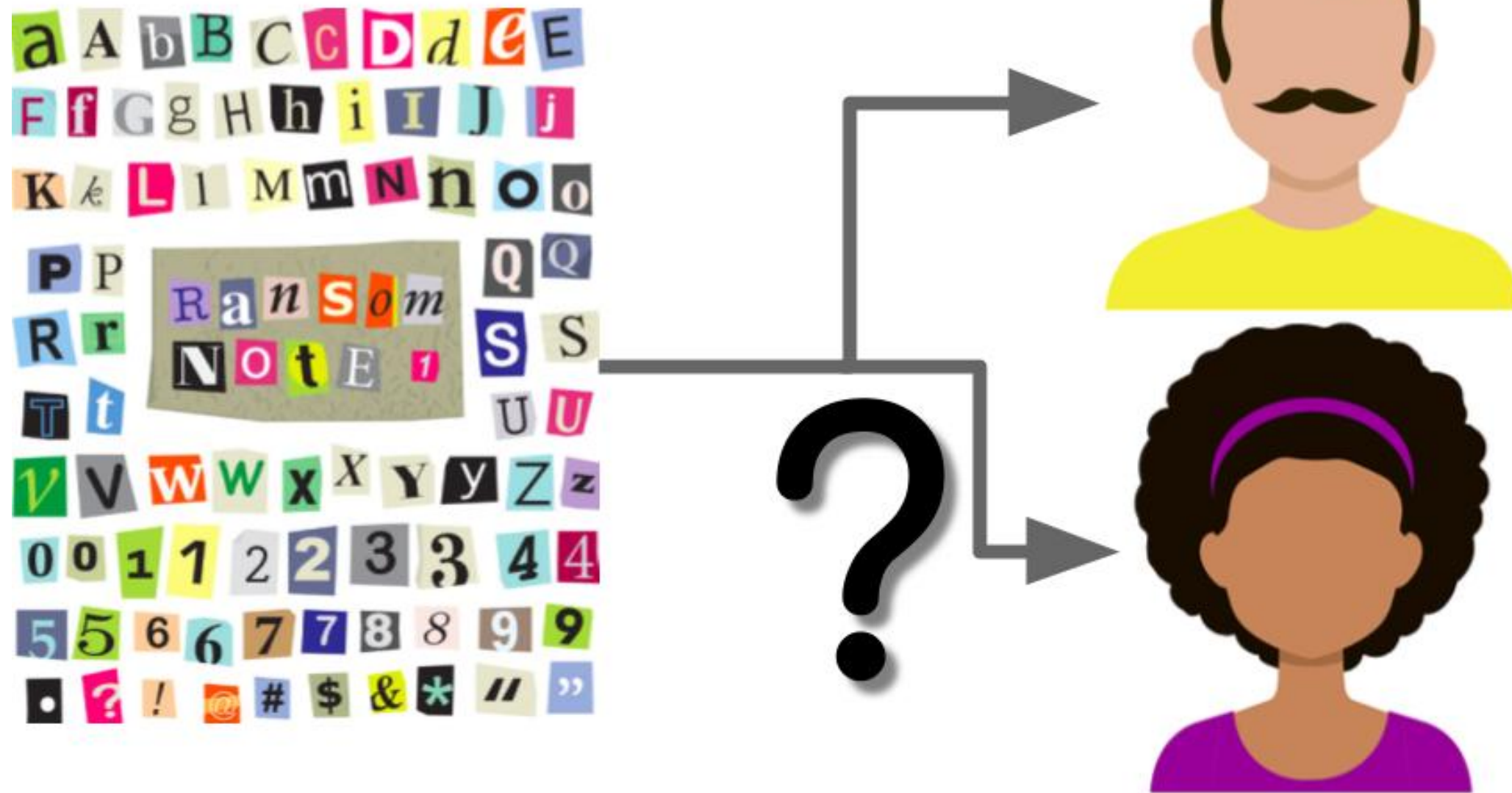
INTRODUCTION TO DATA SCIENCE IN PYTHON



**Hillary Green-Lerman**

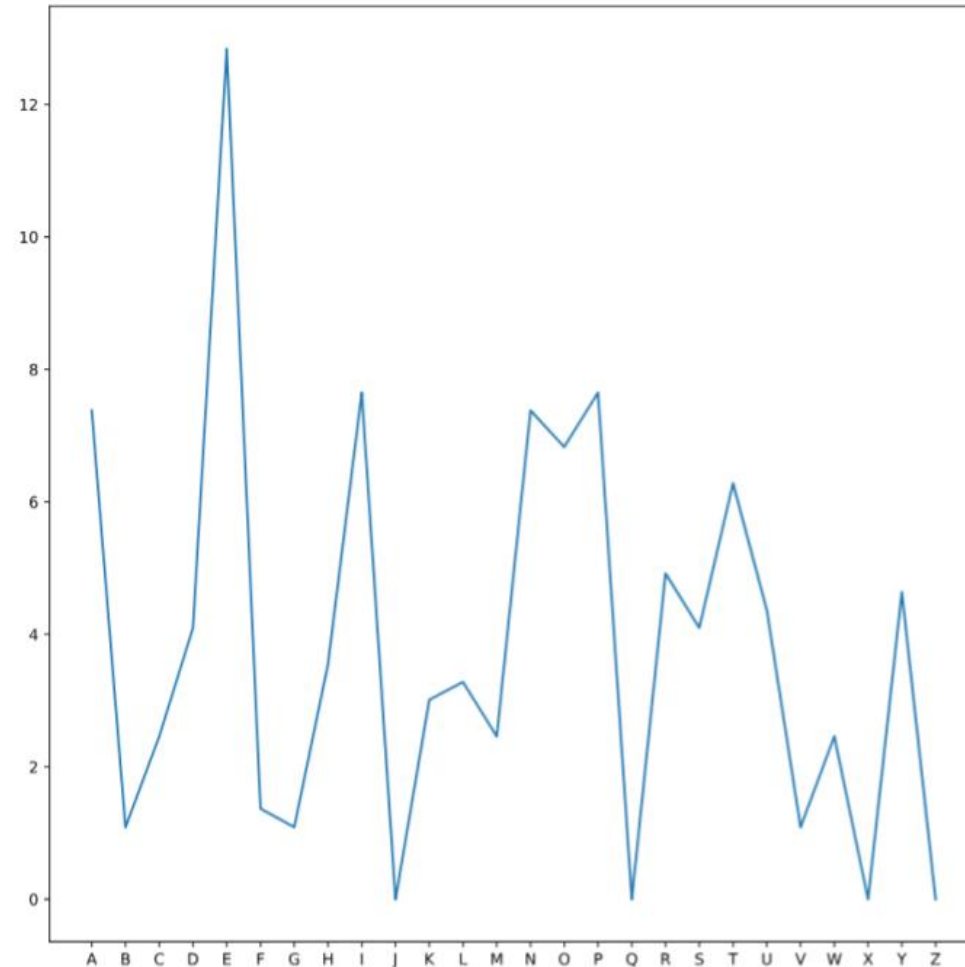
Lead Data Scientist, Looker

# The plot thickens



# From DataFrame to Visualization

letter_index	letter	frequency
1	A	7.38
2	B	1.09
3	C	2.46
4	D	4.10
...	...	...

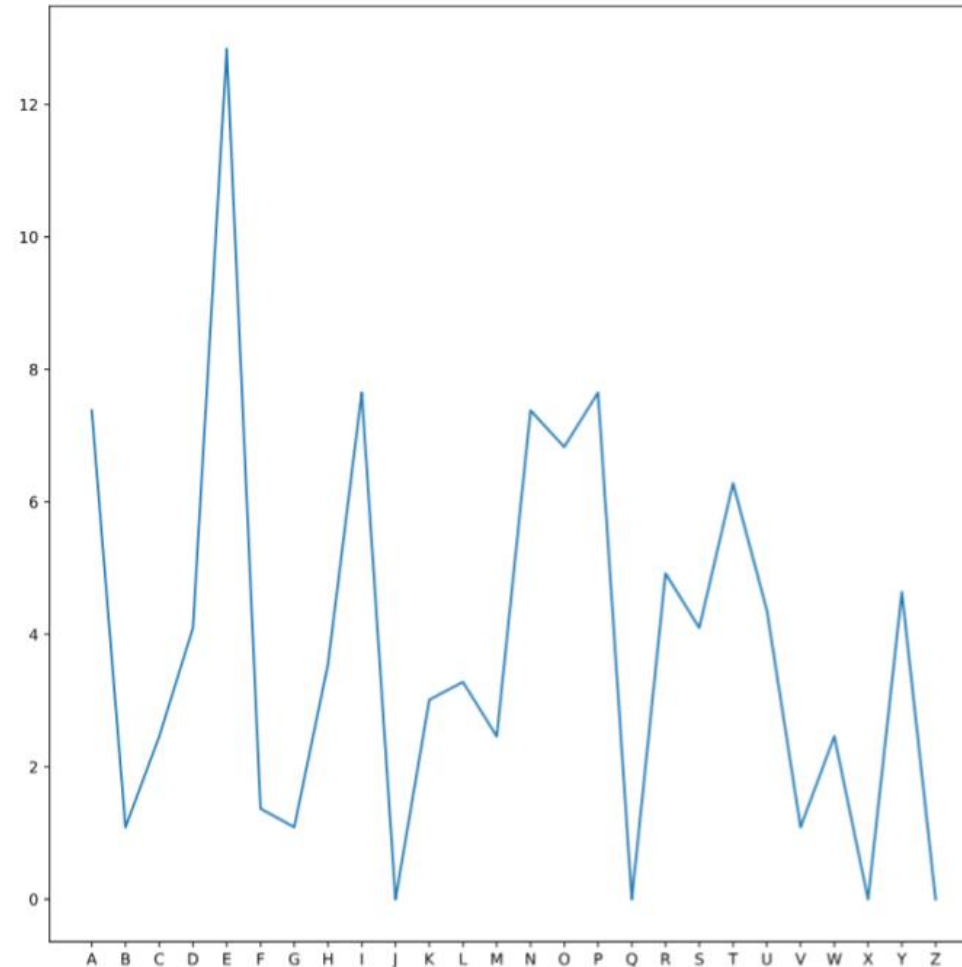


# Introducing Matplotlib

```
from matplotlib import pyplot as plt
```

```
plt.plot(x_values, y_values)
```

```
plt.show()
```



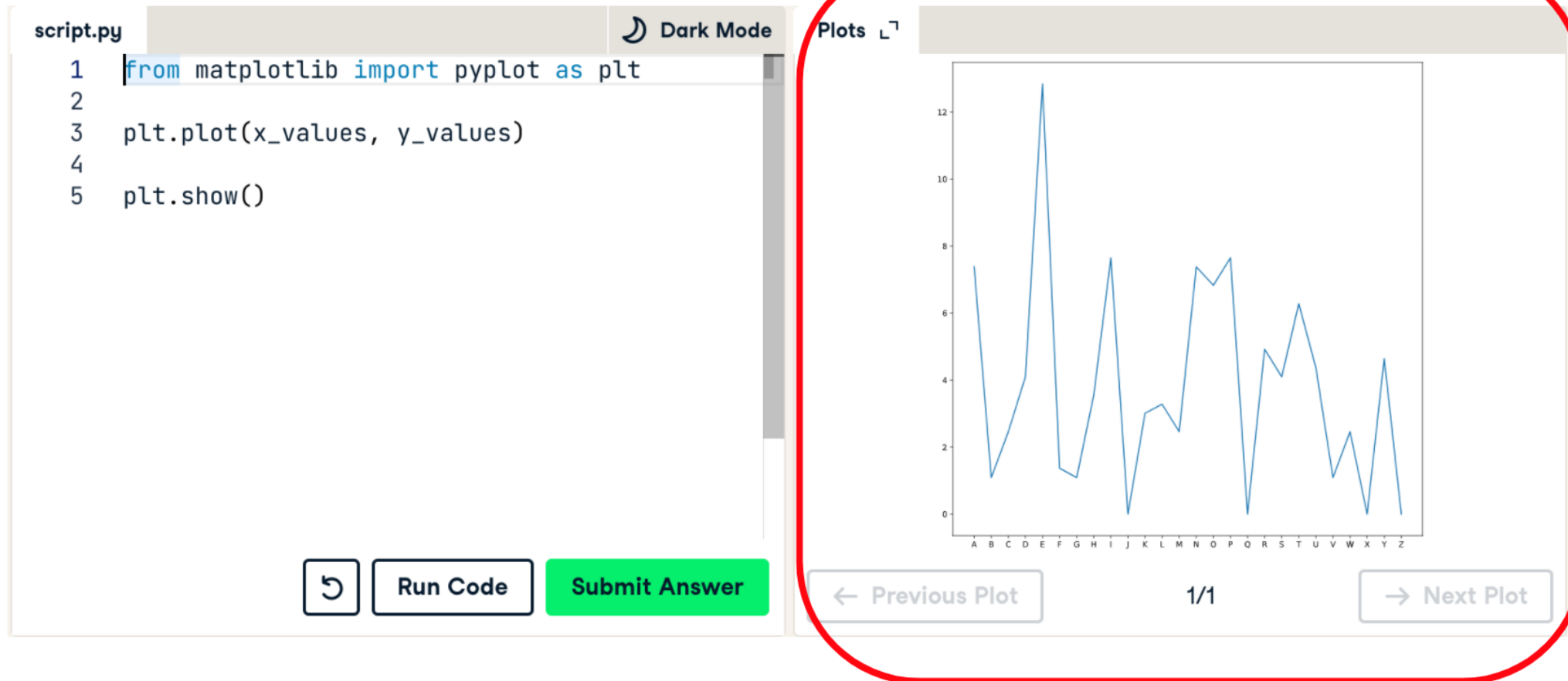
# Line Plot

```
plt.plot( ransom.letter, ransom.frequency )
```

The diagram illustrates the components of the function call `plt.plot( ransom.letter, ransom.frequency )`. It features four labels with arrows pointing to their respective parts in the code: **Function Name** (orange) points to `plt.plot`; **First Positional Argument** (blue) points to `ransom.letter`; **Second Positional Argument** (purple) points to `ransom.frequency`; and **Parenthesis** (green) has two arrows pointing to the opening `(` and closing `)` parentheses. The entire diagram is enclosed in a red rectangular border.

# Displaying the Results

```
plt.show()
```

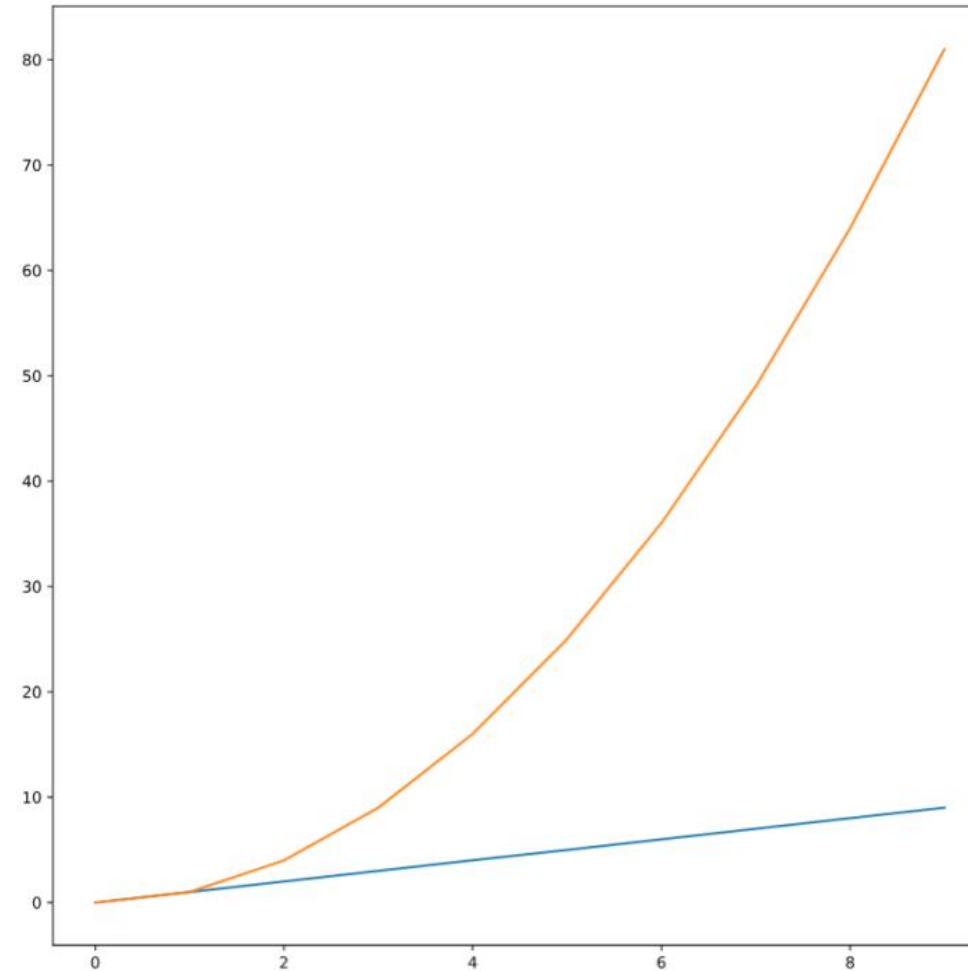


# Multiple Lines

```
plt.plot(data1.x_values,  
         data1.y_values)
```

```
plt.plot(data2.x_values,  
         data2.y_values)
```

```
plt.show()
```



# Let's practice!

INTRODUCTION TO DATA SCIENCE IN PYTHON



# Adding labels and legends

INTRODUCTION TO DATA SCIENCE IN PYTHON



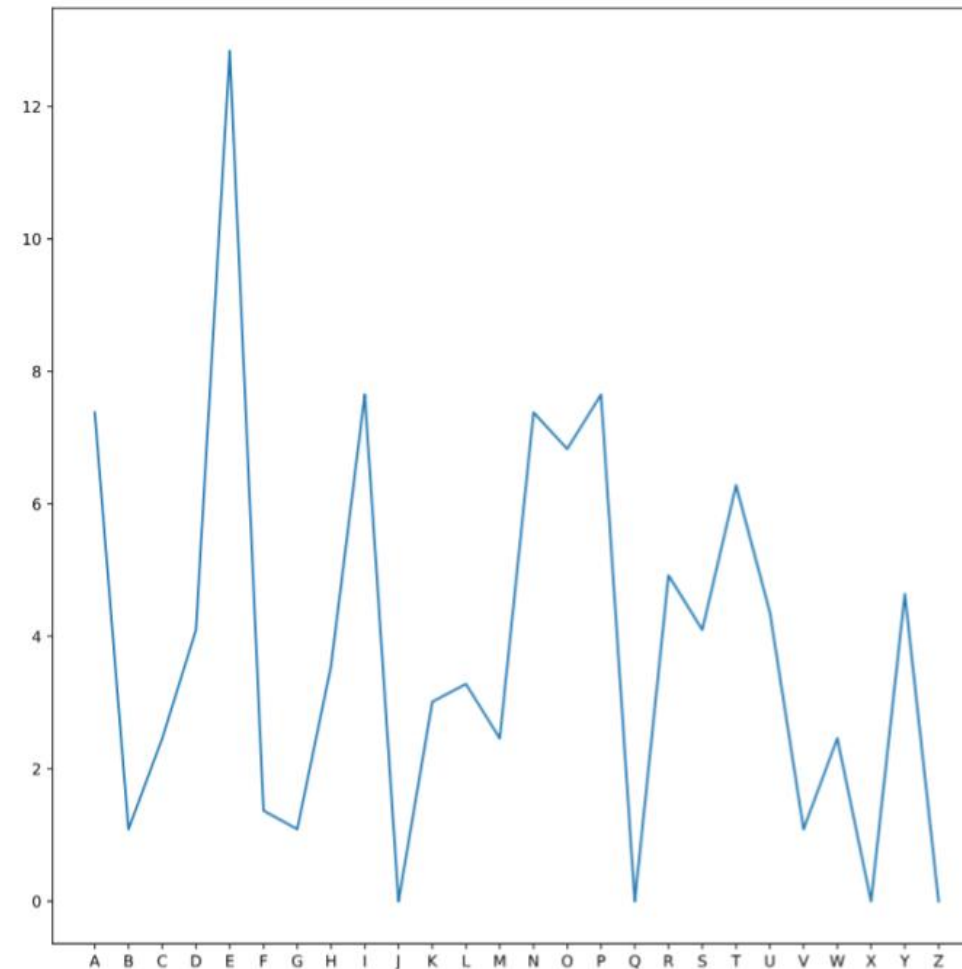
**Hillary Green-Lerman**  
Lead Data Scientist, Looker

# What did we just plot?

```
from matplotlib import pyplot as plt

plt.plot(ransom.letter,
         ransom.frequency)

plt.show()
```



# Axes and title labels

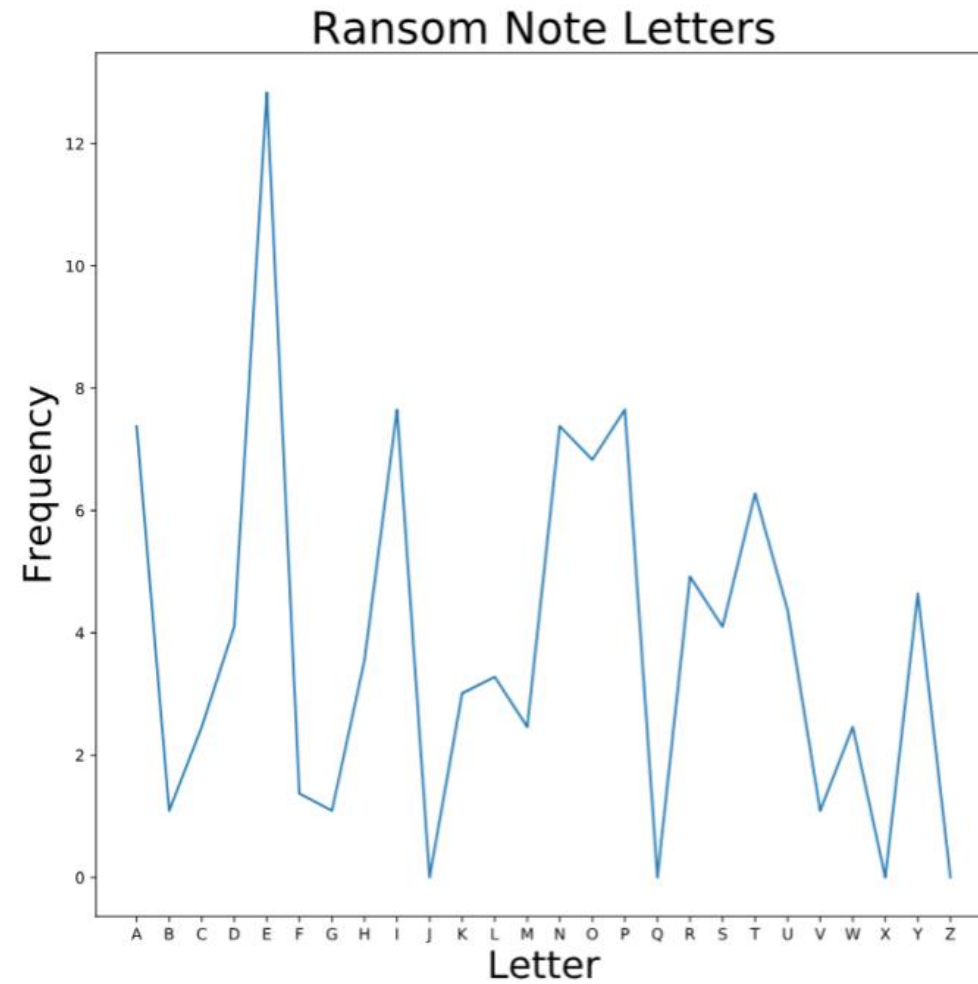
```
plt.xlabel("Letter")
```

```
plt.ylabel("Frequency")
```

```
plt.title("Ransom Note Letters")
```

Labels anywhere before

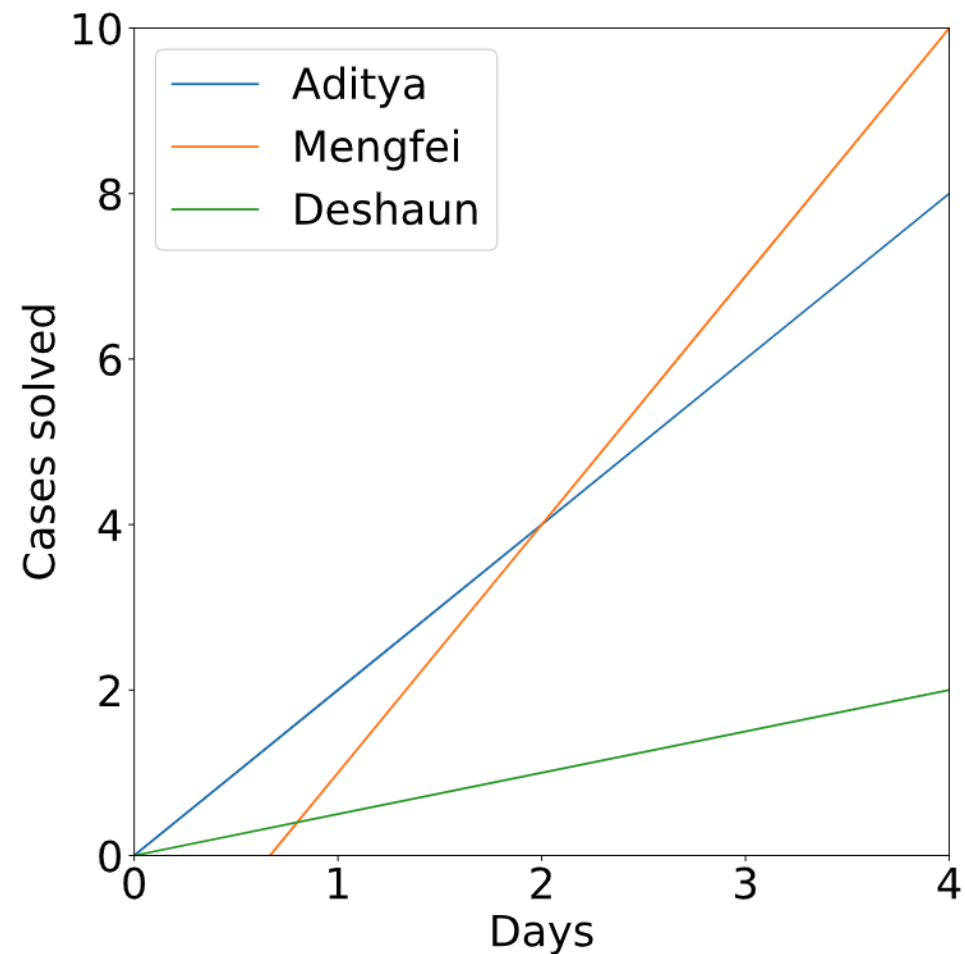
```
plt.show()
```



# Legends

```
plt.plot(aditya.days,
         aditya.cases,
         label="Aditya")
plt.plot(deshaun.days,
         deshaun.cases,
         label="Deshaun")
plt.plot(mengfei.days,
         mengfei.cases,
         label="Mengfei")

plt.legend()
```



## 4. Legends

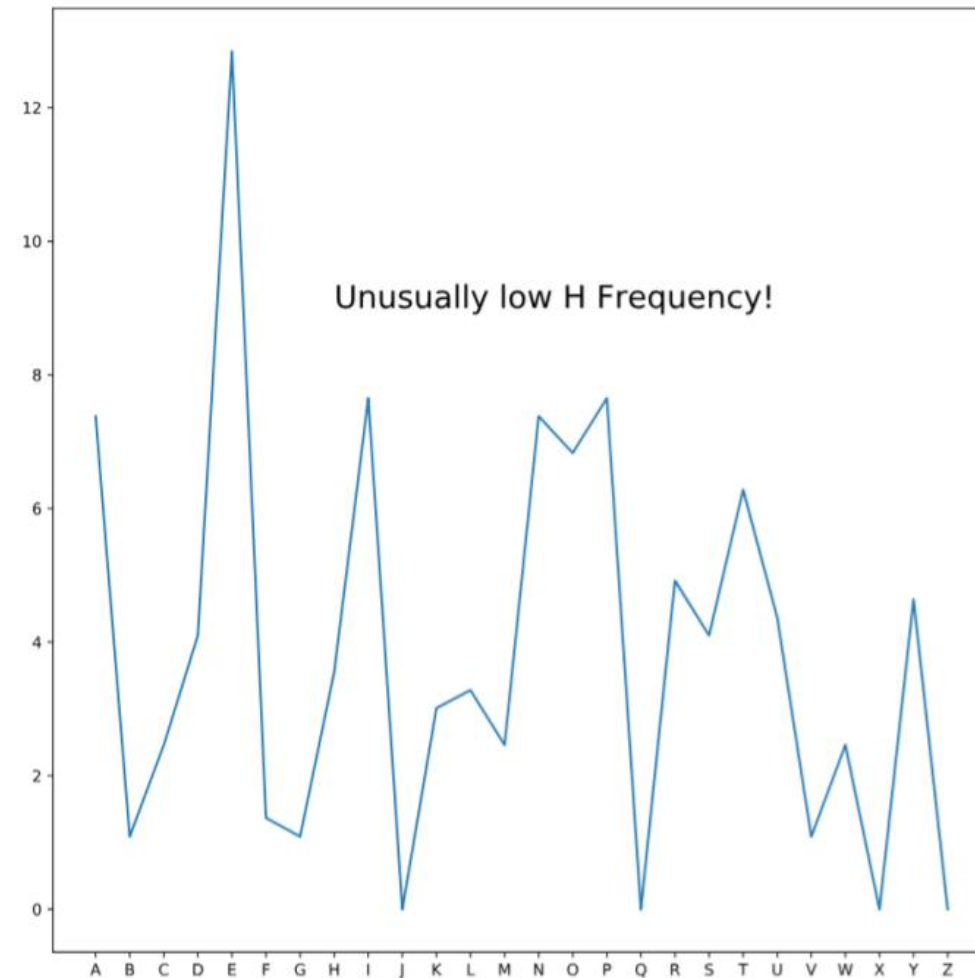
If we have multiple lines in the same plot, we'll want to add a legend. There are two steps for adding a legend: First, we must add the keyword argument `label` to each instance of `plt-dot-plot`. The label will be a string that we want in our legend. By adding a label to each `plt-dot-plot` function, we make our code easier to read. Now, if someone looks at this code, they know what each line is plotting before ever looking at the result. If we just type `plt-dot-show` now, we still won't see a legend. In order to add a legend, we must add a final function: `plt-dot-legend`. Like `plt-dot-show`, `plt-dot-legend` does not take any arguments. It just tells Matplotlib to use the labels from our `plt-dot-plot` functions to create a legend.

# Arbitrary text

## Annotation

```
plt.text(xcoord,  
         ycoord,  
         "Text Message")
```

```
plt.text(5,  
         9,  
         "Unusually low H frequency!")
```



# Modifying text

- Change font size

```
plt.title("Plot title", fontsize=20)
```

- Change font color

```
plt.legend(color="green")
```

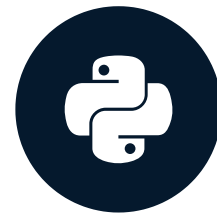
[https://en.wikipedia.org/wiki/Web\\_colors](https://en.wikipedia.org/wiki/Web_colors)

# Let's practice!

INTRODUCTION TO DATA SCIENCE IN PYTHON

# Adding some style

INTRODUCTION TO DATA SCIENCE IN PYTHON

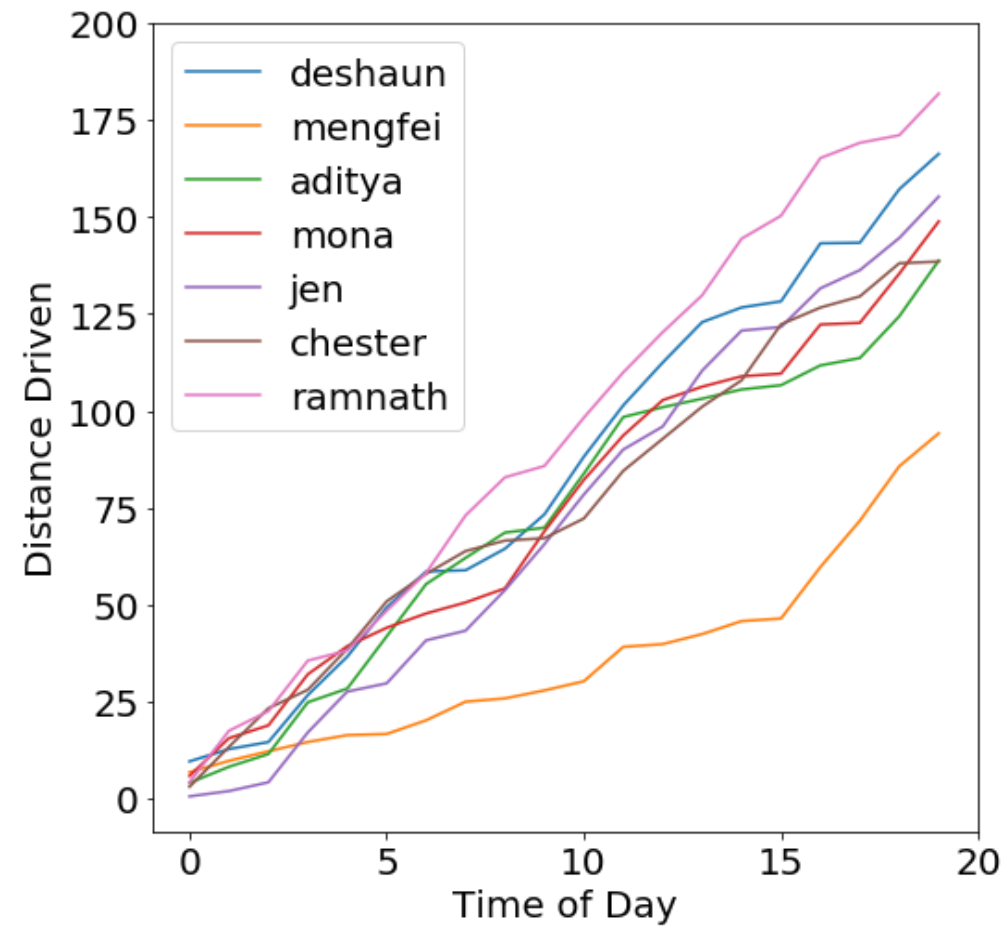


**Hillary Green-Lerman**

Senior Curriculum Lead, DataCamp

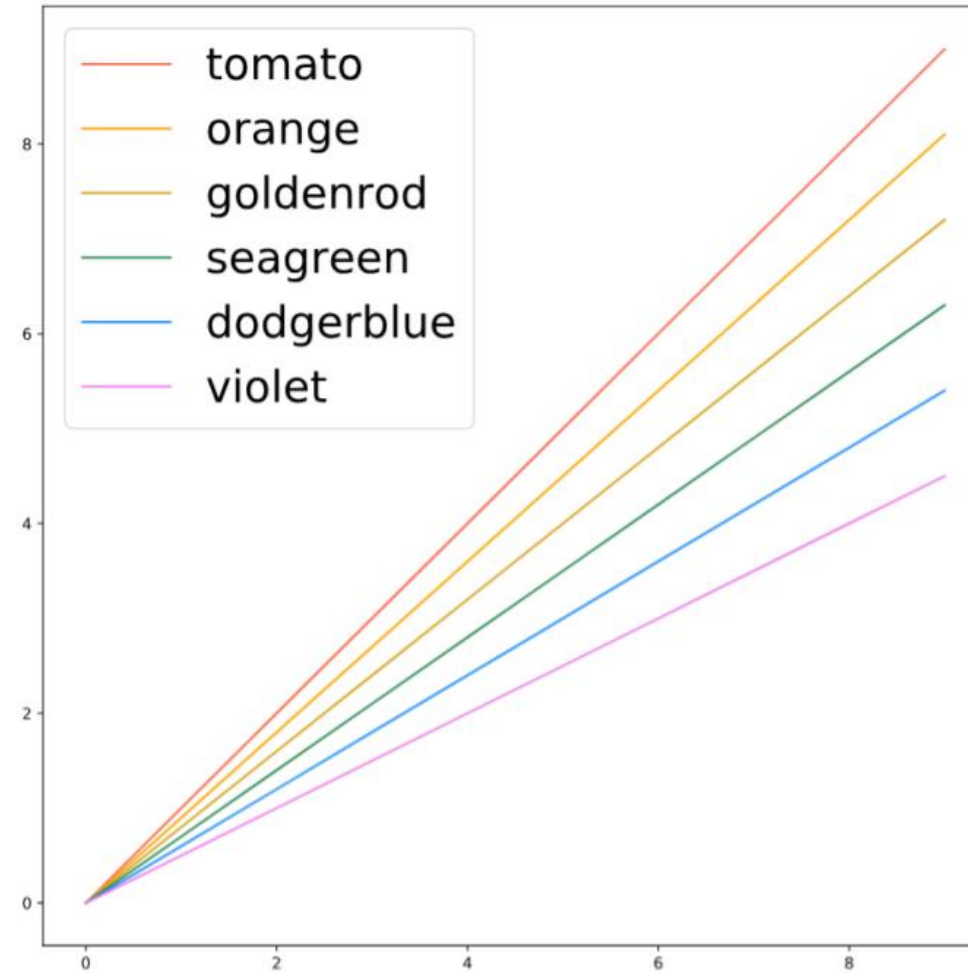


# And miles to go



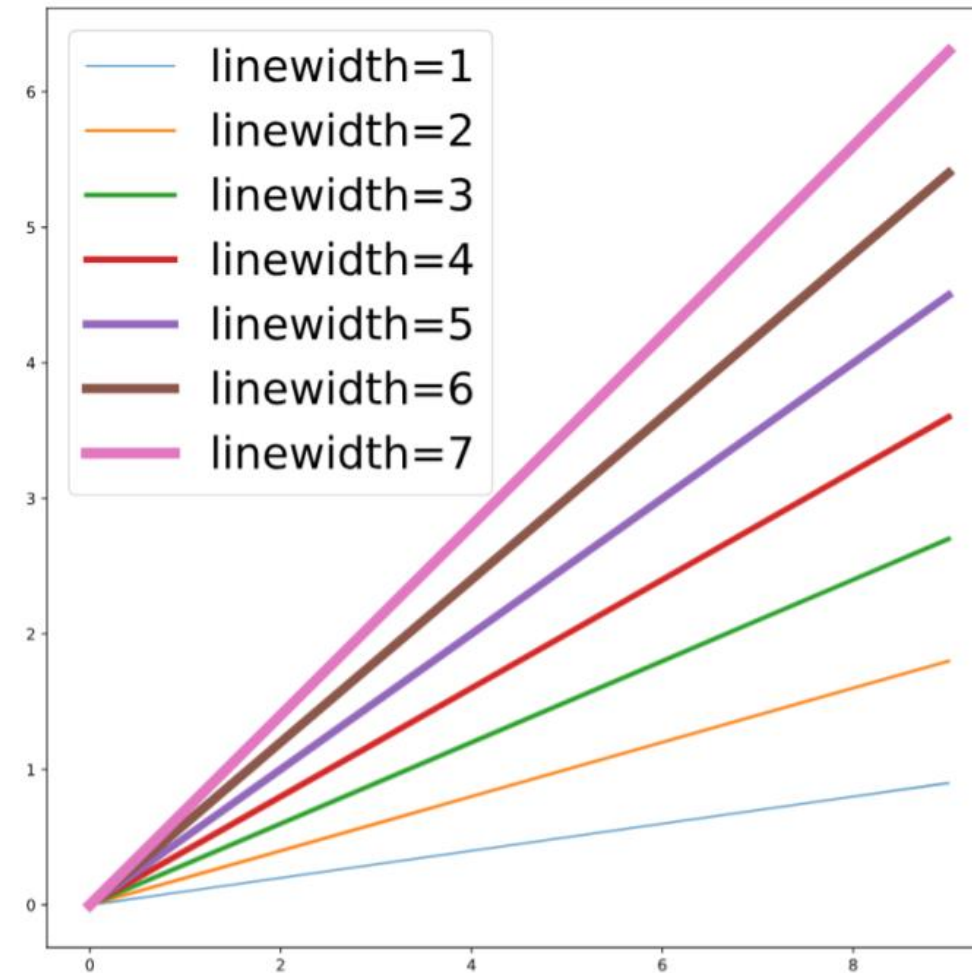
# Changing line color

```
plt.plot(x, y1, color="tomato")  
plt.plot(x, y2, color="orange")  
plt.plot(x, y3, color="goldenrod")  
plt.plot(x, y4, color="seagreen")  
plt.plot(x, y5, color="dodgerblue")  
plt.plot(x, y6, color="violet")
```



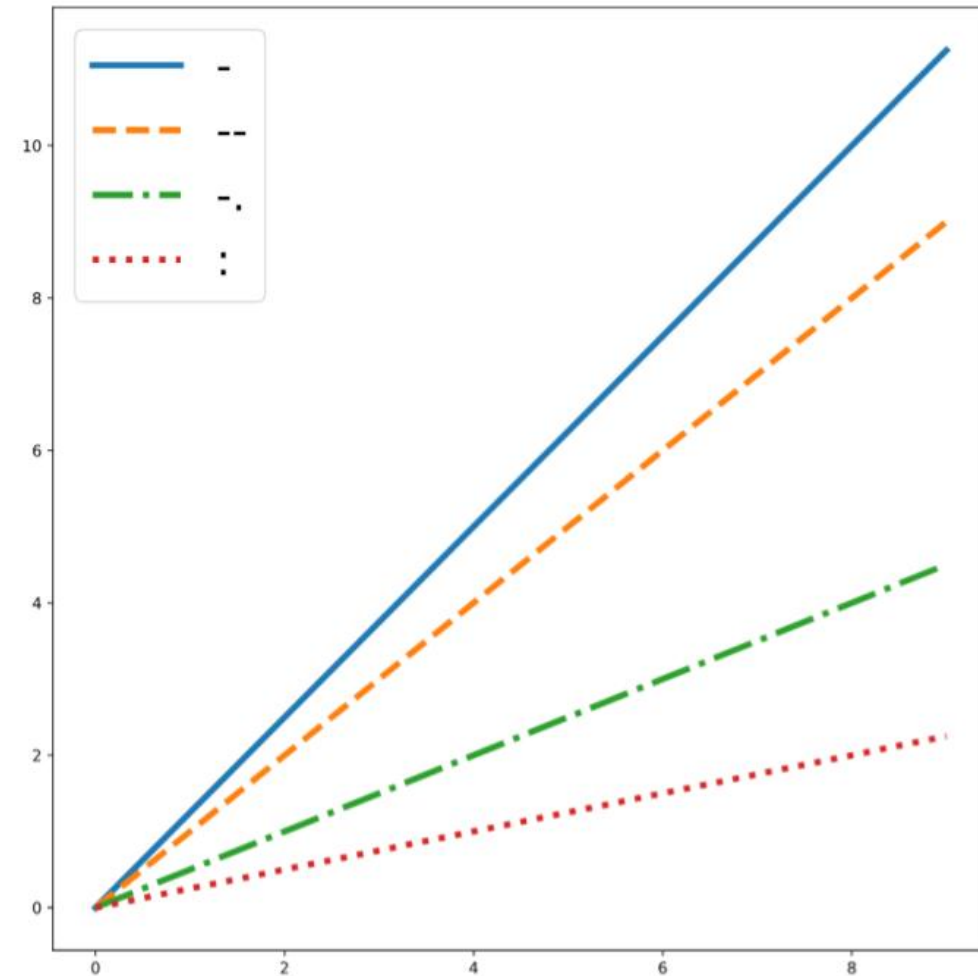
# Changing line width

```
plt.plot(x, y1, linewidth=1)  
plt.plot(x, y2, linewidth=2)  
plt.plot(x, y3, linewidth=3)  
plt.plot(x, y4, linewidth=4)  
plt.plot(x, y5, linewidth=5)  
plt.plot(x, y6, linewidth=6)  
plt.plot(x, y7, linewidth=7)
```



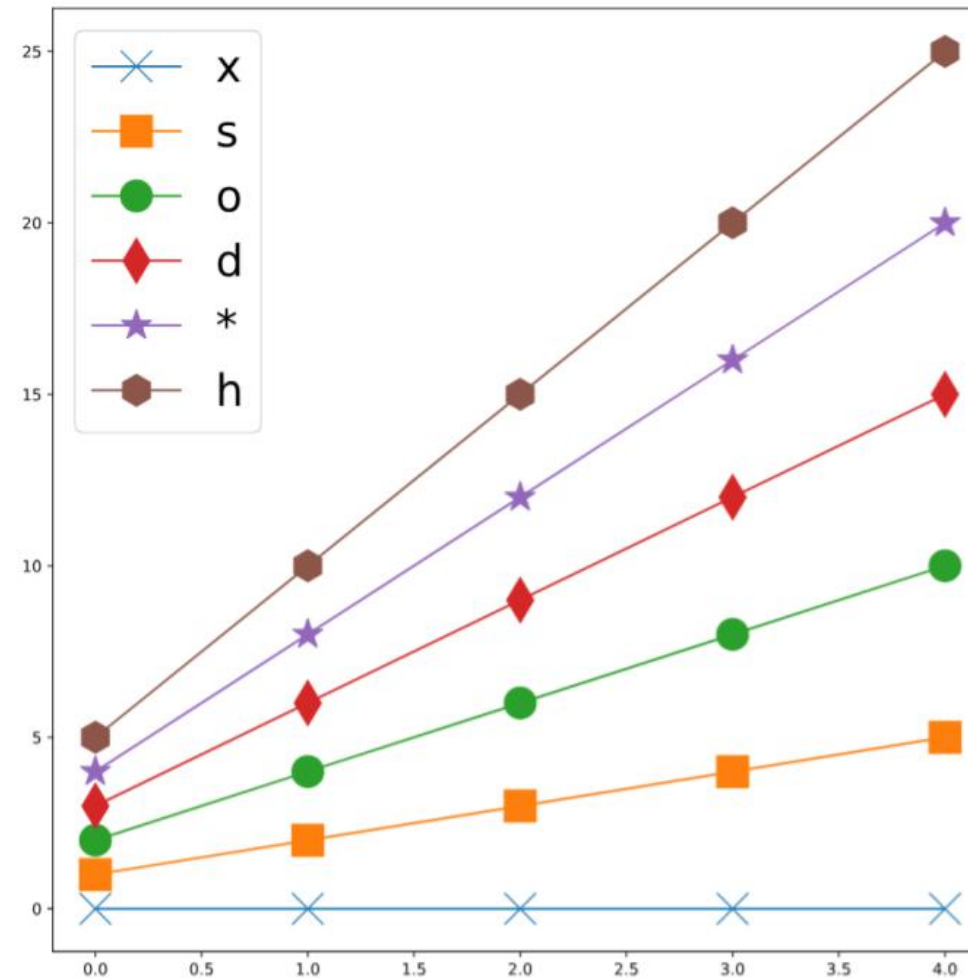
# Changing line style

```
plt.plot(x, y1, linestyle='-')  
plt.plot(x, y2, linestyle='--')  
plt.plot(x, y3, linestyle='-.')  
plt.plot(x, y4, linestyle=':')
```



# Adding markers

```
plt.plot(x, y1, marker='x')  
plt.plot(x, y2, marker='s')  
plt.plot(x, y3, marker='o')  
plt.plot(x, y4, marker='d')  
plt.plot(x, y5, marker='*')  
plt.plot(x, y6, marker='h')
```



# Setting a style

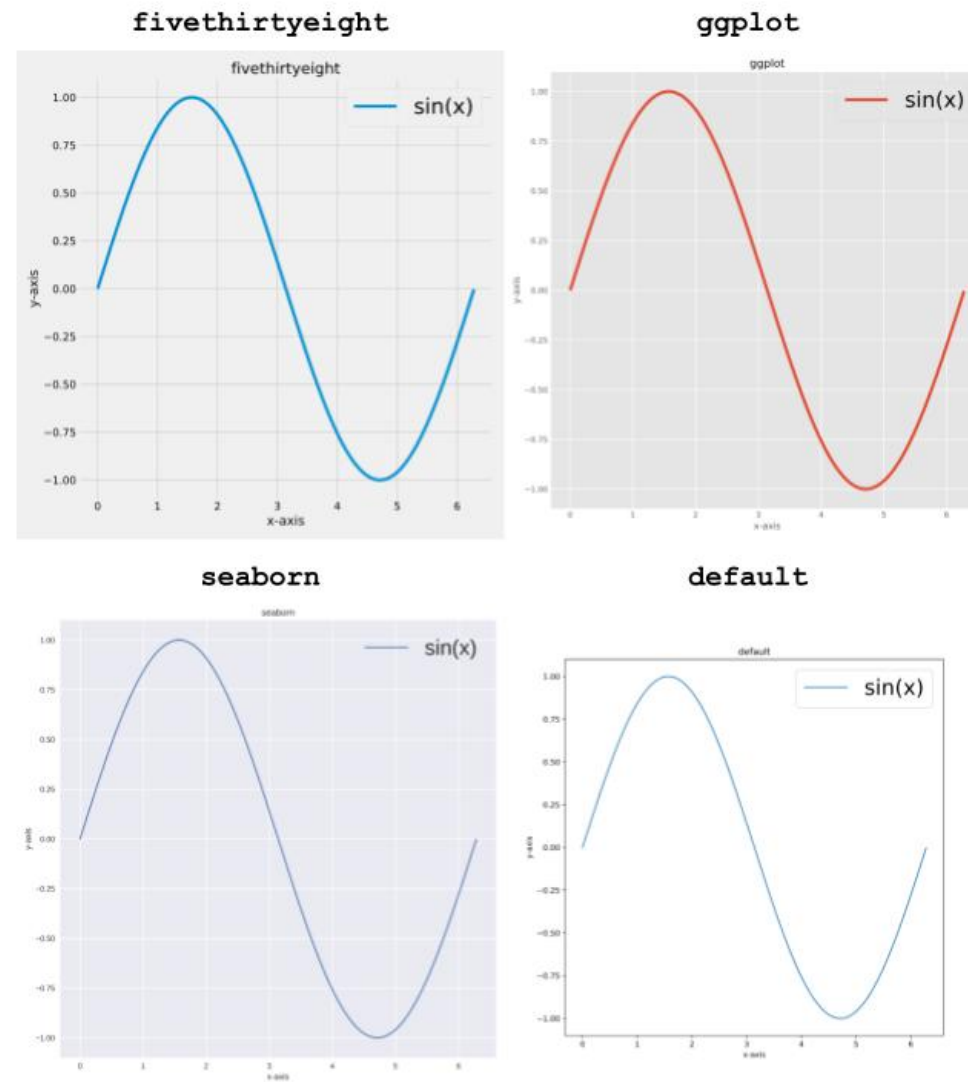
**Before** any other plotting code:

```
plt.style.use('fivethirtyeight')
```

```
plt.style.use('ggplot')
```

```
plt.style.use('seaborn')
```

```
plt.style.use('default')
```



# Let's practice!

INTRODUCTION TO DATA SCIENCE IN PYTHON