Day 7: Numpy and Matplotlib

July 18th 2023

Review From Yesterday’s Homework

common bugs to work through – a good quickstart guide!

Common Errors

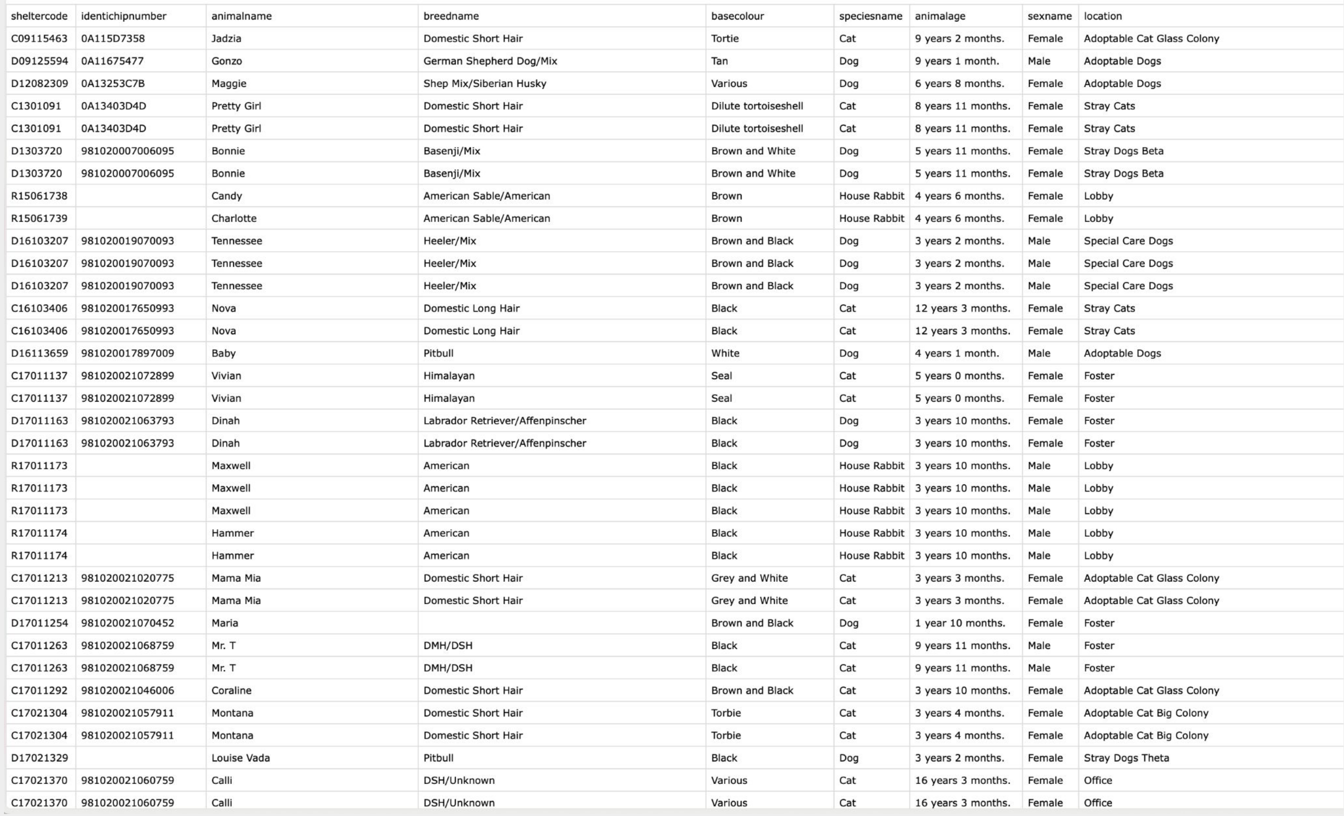
loop example: washing dishes

- when do you want to enter the kitchen?
- when do you want to leave the kitchen?
- if you are paid per dish, how would you count it?
- if you are paid per time worked, how would you count it?

Review From Yesterday's Homework

columns are typically organized along the top of the dataset, left to right

rows are entries for each measurement taken



sheltercode	identichipnumber	animalname	breedname	basecolour	speciesname	animalage	sexname	location
C09115463	0A115D7358	Jadzia	Domestic Short Hair	Tortie	Cat	9 years 2 months.	Female	Adoptable Cat Glass Colony
D09125594	0A11675477	Gonzo	German Shepherd Dog/Mix	Tan	Dog	9 years 1 month.	Male	Adoptable Dogs
D12082309	0A13253C7B	Maggie	Shep Mix/Siberian Husky	Various	Dog	6 years 8 months.	Female	Adoptable Dogs
C1301091	0A13403D4D	Pretty Girl	Domestic Short Hair	Dilute tortoiseshell	Cat	8 years 11 months.	Female	Stray Cats
C1301091	0A13403D4D	Pretty Girl	Domestic Short Hair	Dilute tortoiseshell	Cat	8 years 11 months.	Female	Stray Cats
D1303720	981020007006095	Bonnie	Basenji/Mix	Brown and White	Dog	5 years 11 months.	Female	Stray Dogs Beta
D1303720	981020007006095	Bonnie	Basenji/Mix	Brown and White	Dog	5 years 11 months.	Female	Stray Dogs Beta
R15061738		Candy	American Sable/American	Brown	House Rabbit	4 years 6 months.	Female	Lobby
R15061739		Charlotte	American Sable/American	Brown	House Rabbit	4 years 6 months.	Female	Lobby
D16103207	981020019070093	Tennessee	Heeler/Mix	Brown and Black	Dog	3 years 2 months.	Male	Special Care Dogs
D16103207	981020019070093	Tennessee	Heeler/Mix	Brown and Black	Dog	3 years 2 months.	Male	Special Care Dogs
D16103207	981020019070093	Tennessee	Heeler/Mix	Brown and Black	Dog	3 years 2 months.	Male	Special Care Dogs
C16103406	981020017650993	Nova	Domestic Long Hair	Black	Cat	12 years 3 months.	Female	Stray Cats
C16103406	981020017650993	Nova	Domestic Long Hair	Black	Cat	12 years 3 months.	Female	Stray Cats
D16113659	981020017897009	Baby	Pitbull	White	Dog	4 years 1 month.	Male	Adoptable Dogs
C17011137	981020021072899	Vivian	Himalayan	Seal	Cat	5 years 0 months.	Female	Foster
C17011137	981020021072899	Vivian	Himalayan	Seal	Cat	5 years 0 months.	Female	Foster
D17011163	981020021063793	Dinah	Labrador Retriever/Affenpinscher	Black	Dog	3 years 10 months.	Female	Foster
D17011163	981020021063793	Dinah	Labrador Retriever/Affenpinscher	Black	Dog	3 years 10 months.	Female	Foster
R17011173		Maxwell	American	Black	House Rabbit	3 years 10 months.	Male	Lobby
R17011173		Maxwell	American	Black	House Rabbit	3 years 10 months.	Male	Lobby
R17011173		Maxwell	American	Black	House Rabbit	3 years 10 months.	Male	Lobby
R17011174		Hammer	American	Black	House Rabbit	3 years 10 months.	Male	Lobby
R17011174		Hammer	American	Black	House Rabbit	3 years 10 months.	Male	Lobby
C17011213	981020021020775	Mama Mia	Domestic Short Hair	Grey and White	Cat	3 years 3 months.	Female	Adoptable Cat Glass Colony
C17011213	981020021020775	Mama Mia	Domestic Short Hair	Grey and White	Cat	3 years 3 months.	Female	Adoptable Cat Glass Colony
D17011254	981020021070452	Maria		Brown and Black	Dog	1 year 10 months.	Female	Foster
C17011263	981020021068759	Mr. T	DMH/DSH	Black	Cat	9 years 11 months.	Male	Foster
C17011263	981020021068759	Mr. T	DMH/DSH	Black	Cat	9 years 11 months.	Male	Foster
C17011292	981020021046006	Coraline	Domestic Short Hair	Brown and Black	Cat	3 years 10 months.	Female	Adoptable Cat Glass Colony
C17021304	981020021057911	Montana	Domestic Short Hair	Torbie	Cat	3 years 4 months.	Female	Adoptable Cat Big Colony
C17021304	981020021057911	Montana	Domestic Short Hair	Torbie	Cat	3 years 4 months.	Female	Adoptable Cat Big Colony
D17021329		Louise Vada	Pitbull	Black	Dog	3 years 2 months.	Female	Stray Dogs Theta
C17021370	981020021060759	Calli	DSH/Unknown	Various	Cat	16 years 3 months.	Female	Office
C17021370	981020021060759	Calli	DSH/Unknown	Various	Cat	16 years 3 months.	Female	Office

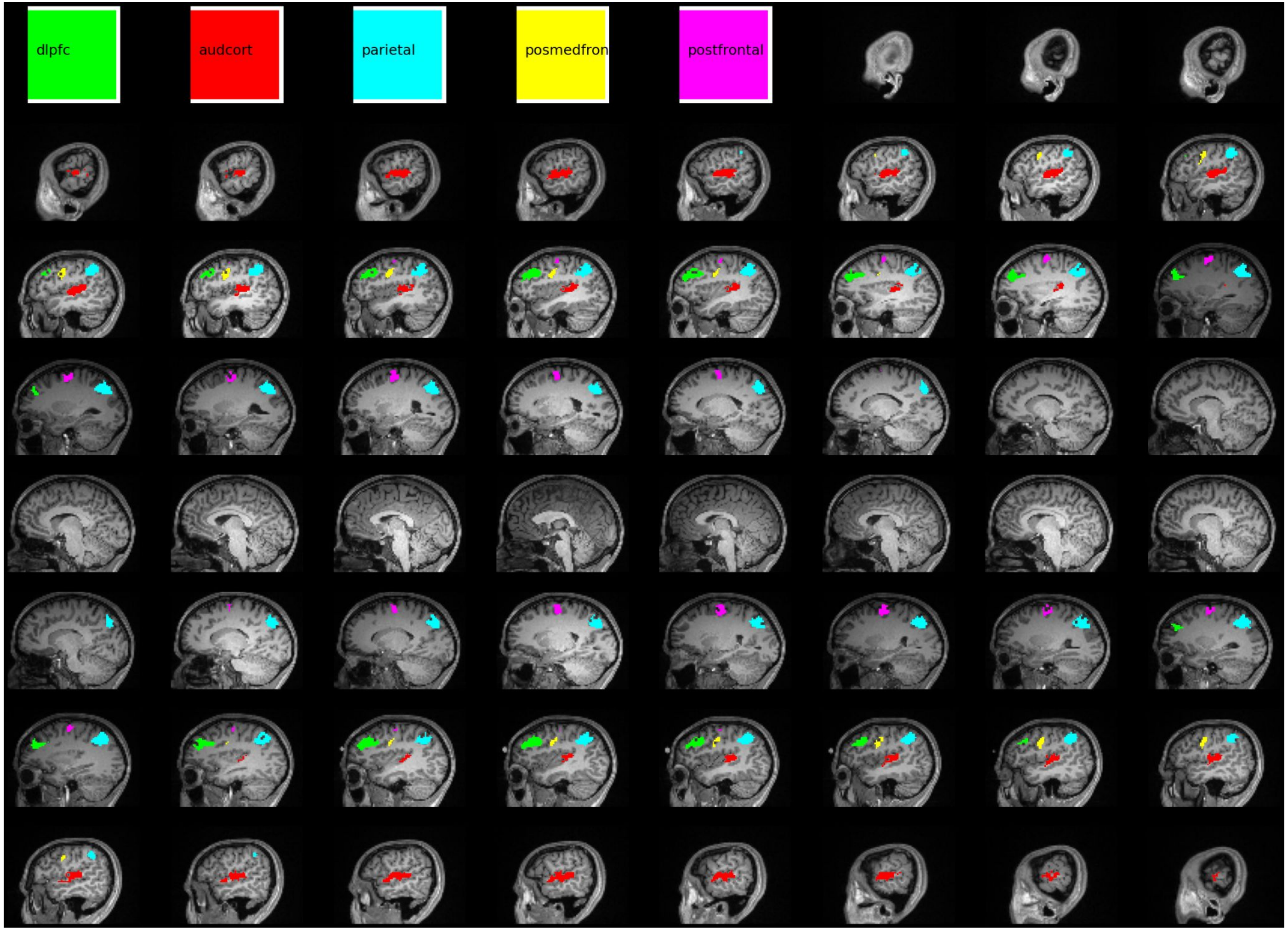
Review From Yesterday’s Homework

- functions returning vs printing
- see history on replit!

Today: Numpy (Numerical Python) and Matplotlib (matlab plotting library)

- numpy provides the basis for most more complicated packages, like tensorflow (machine learning) or image processing (including MRI data!)
- matplotlib is the basis for **data visualization** – making plots!
- we now move from just working with things we can type in the code shell into using big data files and producing more complicated code

- this class does not have time to cover multi-dimensional data, but numpy is the place to start if you are interested in learning more!


dlpfc
audcort
parietal
posmedfron
postfrontal

What is an array?

An array is a central data structure of the NumPy library. An array is a grid of values and it contains information about the raw data, how to locate an element, and how to interpret an element. It has a grid of elements that can be indexed in various ways. The elements are all of the same type, referred to as the array `dtype`.

An array can be indexed by a tuple of nonnegative integers, by booleans, by another array, or by integers. The `rank` of the array is the number of dimensions. The `shape` of the array is a tuple of integers giving the size of the array along each dimension.

One way we can initialize NumPy arrays is from Python lists, using nested lists for two- or higher-dimensional data.

For example:

```
>>> a = np.array([1, 2, 3, 4, 5, 6])
```

Numpy Basics

- typically imported as **np**
- the list indexing you learned still works!
    - arr = np.array([1,2,3])
    - print(arr[0]) # prints 1
- you can also select slices of an array, using the same logic as **range** (*includes first index, not last*)
    - arr = np.array([1,2,4,5,6,7])
    - print(arr[2:4]) # prints [4,5]
- your turn:
    - how would you get the first 3 elements of array **arr?**
    - how would you get the last 3 elements of array **arr**?

What’s the difference between a Python list and a NumPy array?

NumPy gives you an enormous range of fast and efficient ways of creating arrays and manipulating numerical data inside them. While a Python list can contain different data types within a single list, all of the elements in a NumPy array should be homogeneous. The mathematical operations that are meant to be performed on arrays would be extremely inefficient if the arrays weren’t homogeneous.

Why use NumPy?

NumPy arrays are faster and more compact than Python lists. An array consumes less memory and is convenient to use. NumPy uses much less memory to store data and it provides a mechanism of specifying the data types. This allows the code to be optimized even further.

Numpy is really good at math

- what if we want to take the square of a whole list of numbers?
- how about the absolute number?

- you can also use two arrays of the same size to subtract from each other!
    - what if you have the current money each player in a game has earned, and then their bonus payment as part of the final round of the game. we can use numpy to combine this data very easily!
- this array comparison can also work for doing logical operations

- finally, numpy has a way to generate lots of random data

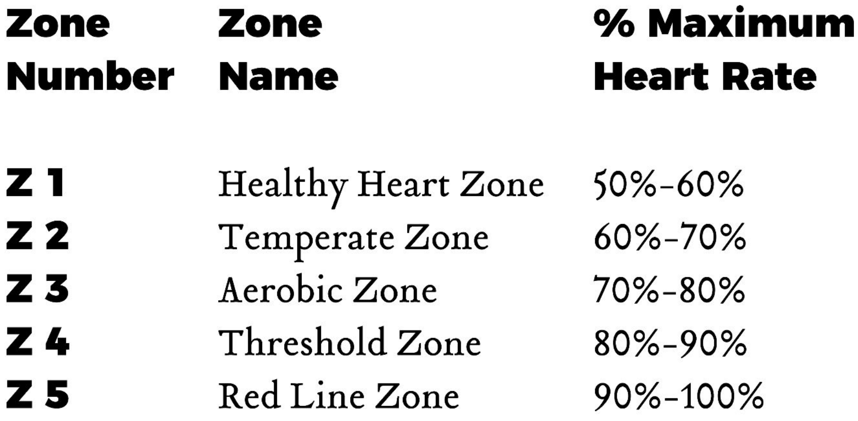
Class data collection exercise: heart rates

for each student, going to collect heart rate 3 times

MAXIMUM target heart rate is (220 - your age)

going to calculate what zone your heart rate falls in for each measure

eventually going to make
this into a plot!

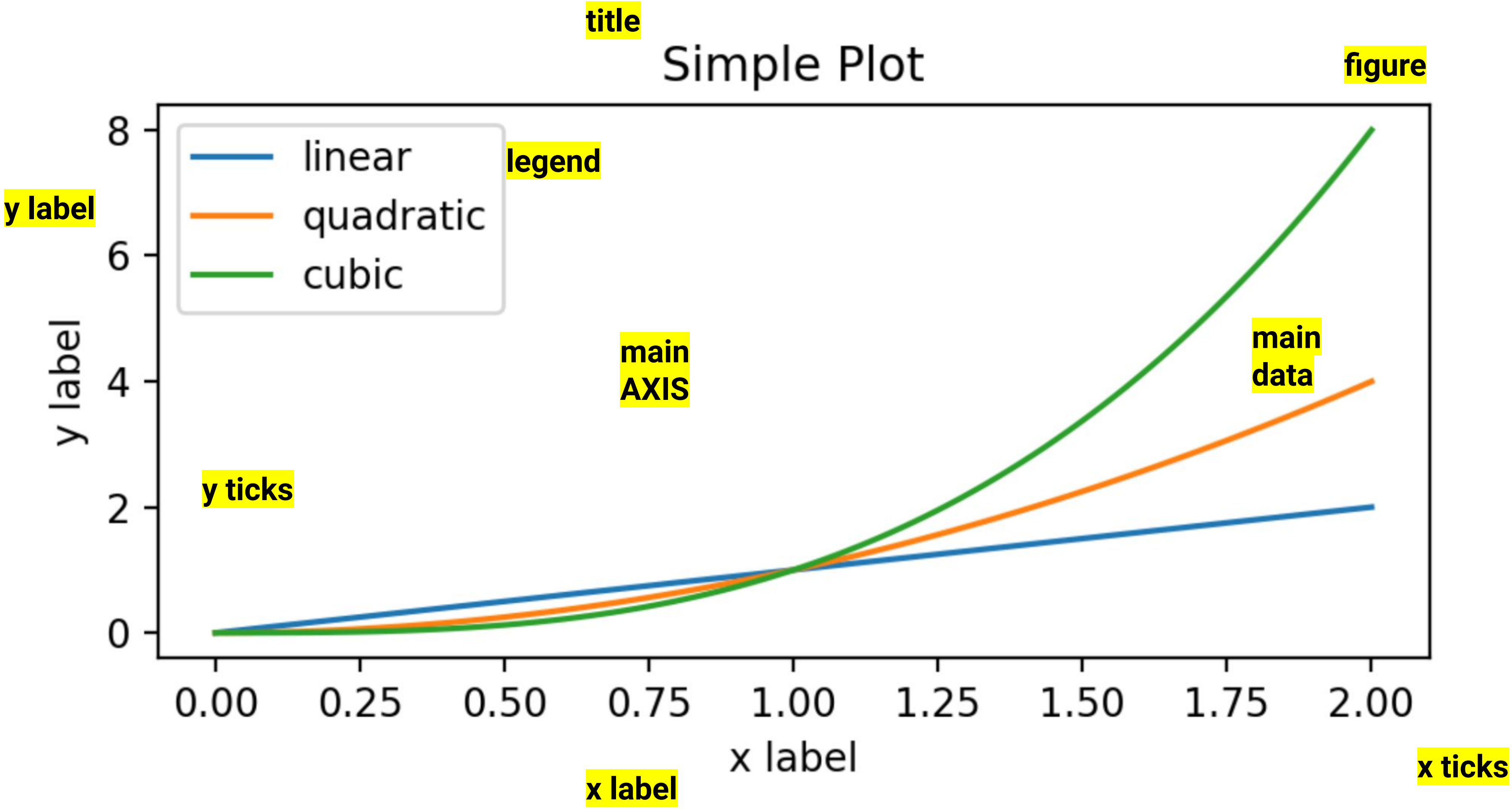


Zone Number	Zone Name	% Maximum Heart Rate
Z 1	Healthy Heart Zone	50%-60%
Z 2	Temperate Zone	60%-70%
Z 3	Aerobic Zone	70%-80%
Z 4	Threshold Zone	80%-90%
Z 5	Red Line Zone	90%-100%

Matplotlib – turning our numbers into plots

Going to plot three points in a line to start with – our three heart rate measurements

before starting on coding plots, let’s walk through basic elements of a graph / plot



title

figure

legend

y label

main AXIS

main data

y ticks

x label

x ticks

Matplotlib

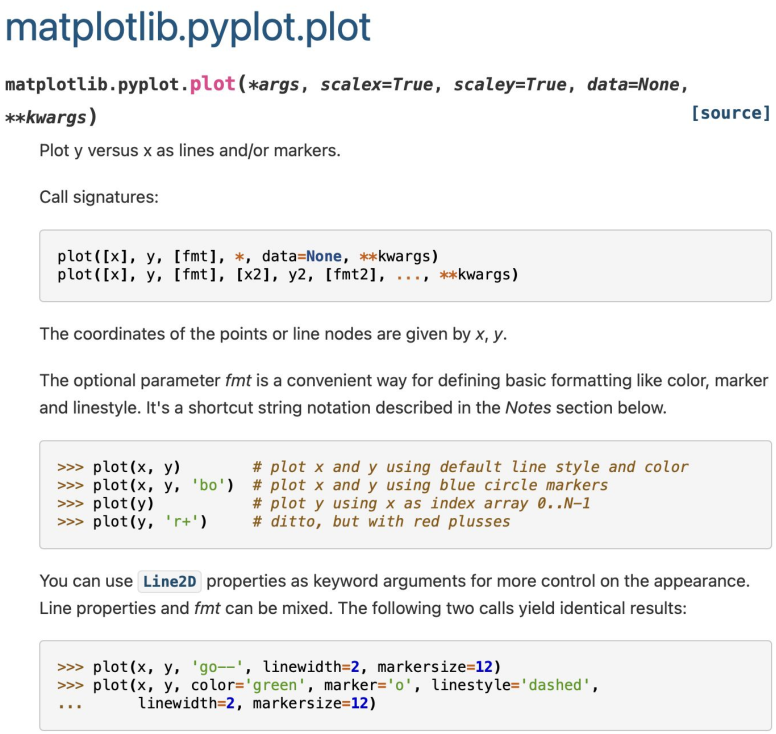
- to make a new place to plot, we will use **fig, ax = plt.subplots()** to make a new canvas to work on
  - breakdown this line: why do we have fig, ax before the equals sign? what is plt? what is subplots()
- to show what we have plotted, we will use **plt.show()** *at the end of our plotting!!*
  - you need to call plt.show() and now just show() because you may have that function defined yourself!
  - plt.show() is like taking a picture of your artwork – you don’t want to do that and then keep working on it!

First Plot – Random Data

Using the random data we generated from earlier, we will make our first plot!

The simplest type of call is **plt.plot()** which takes in *at least* y values you want to plot, or more information!

let’s use rand1 and rand2 from earlier and try to plot x locations are the values in rand1 and y locations are the values in rand2



matplotlib.pyplot.plot

`matplotlib.pyplot.plot(*args, scalex=True, scaley=True, data=None, **kwargs)` [source]

Plot y versus x as lines and/or markers.

Call signatures:

```
plot([x], y, [fmt], *, data=None, **kwargs)
plot([x], y, [fmt], [x2], y2, [fmt2], ..., **kwargs)
```

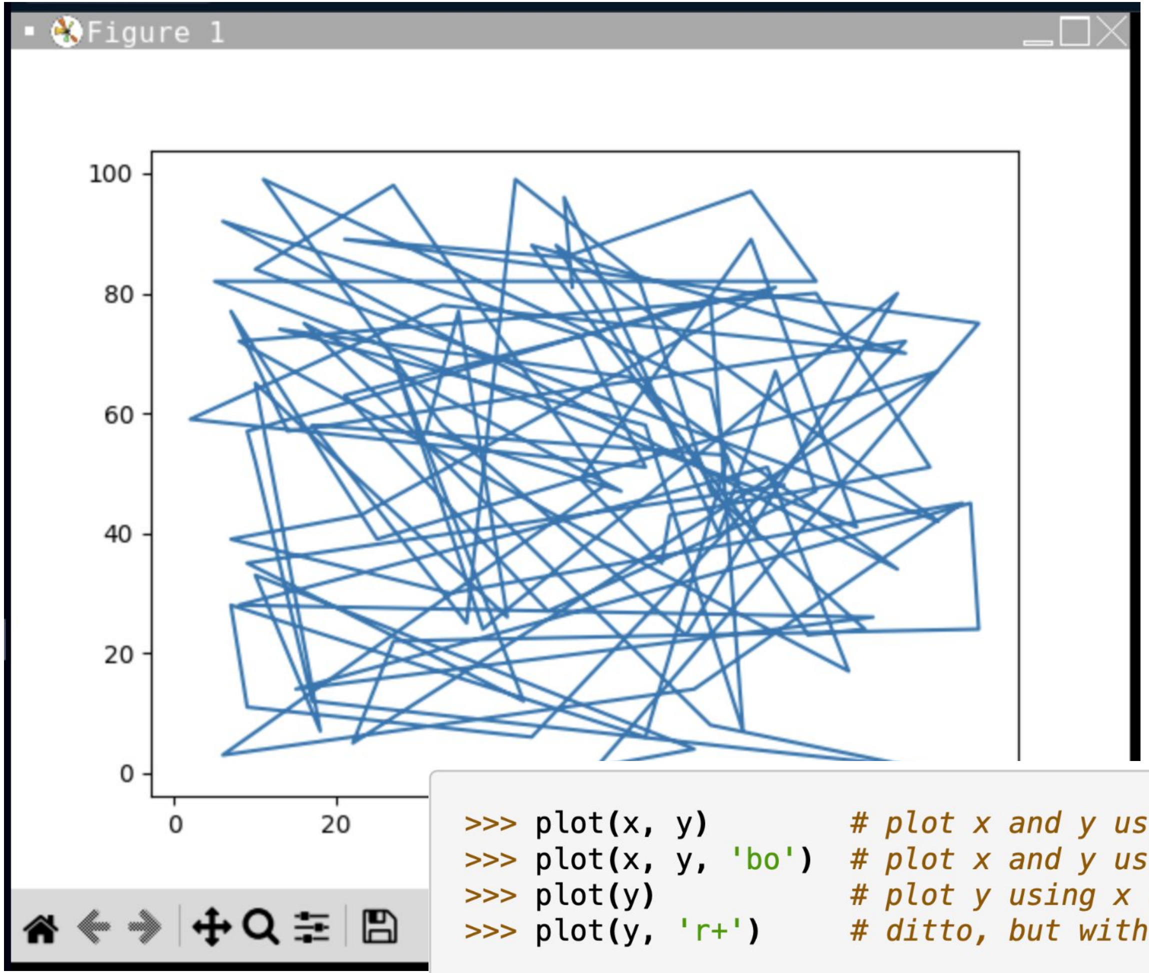
The coordinates of the points or line nodes are given by *x*, *y*.

The optional parameter *fmt* is a convenient way for defining basic formatting like color, marker and linestyle. It's a shortcut string notation described in the *Notes* section below.

```
>>> plot(x, y)        # plot x and y using default line style and color
>>> plot(x, y, 'bo')  # plot x and y using blue circle markers
>>> plot(y)           # plot y using x as index array 0..N-1
>>> plot(y, 'r+')     # ditto, but with red plusses
```

You can use `Line2D` properties as keyword arguments for more control on the appearance. Line properties and *fmt* can be mixed. The following two calls yield identical results:

```
>>> plot(x, y, 'go--', linewidth=2, markersize=12)
>>> plot(x, y, color='green', marker='o', linestyle='dashed',
...      linewidth=2, markersize=12)
```



why did we get this plot? matplotlib tried to be helpful and assumed you wanted to connect all your lines. in this case, we want to just have dots and not connect them

an easy fix – we tell matplotlib what kind of **marker** style we want to use!

```
>>> plot(x, y)        # plot x and y using default line style and color
>>> plot(x, y, 'bo')  # plot x and y using blue circle markers
>>> plot(y)           # plot y using x as index array 0..N-1
>>> plot(y, 'r+')     # ditto, but with red plusses
```

**kwargs : an expansive world of options

- get familiar with some of the colors that are available as shortcuts, but no need to memorize them – this is what documentation is for and why we practice finding what we need online!
- how would we make:
  - a cyan plus?
  - a yellow diamond?
  - a dash-dot line?
  - a green octagon?
- try adding these to your plot and see the changes!

Plot Labeling – Making things readable

- when we just look at our plot, we have no idea what it is showing us
- this is why labels are important – **strings** we use to say what data shows
- we can use plt.title(), plt.xlabel(), and plt.ylabel() to answer this problem!
- remember, you want to put these labels on **before** you take a picture and show off your artwork!

- add the title ‘my first plot’ and the x label ‘random 1’ and the y label ‘random 2’

Plotting Our Heart Rate

- using the data from earlier, make a line plot showing your heart rate over time
- you can choose the style of marker and color but you should use a dotted line between points
- add appropriate x, y, and title labels!

More Types of plots…

- as you could probably guess from the examples, this is not the only way to make a plot in matplotlib. lots of other options which will help you visualize your data
- your turn: how can you make your heart rate into a **bar** plot? time to turn to documentation!

Homework Problem 1: Practicing Numpy

given a vector **vec**, please write a script that prints the following values:

1. the sum of the numbers in the vector
2. the sum of the squares of the numbers in the vector
3. the first two elements of the vector
4. the fourth, fifth, and sixth elements of the vector
5. a boolean vector representing if each number is larger than 3
6. *optional challenge: every third element, starting from the second one!*

use documentation where something is new to you!

Homework Problem 2: Practicing Matplotlib

We will provide started code to extract one data **column** from your dataset which you should make a plot of for your homework tonight. This plot can be of any style of your choice, but should include the following elements:

- title and axis (x,y) labels
- a non blue-line style plot (any other is fine)

Look back to make sure you create a figure and then show what you made!

The data you will use will be in a variable **data**.

Write 1-3 sentences describing what you have shown in your plot (put in comments underneath)