Introduction to Python

Fondren Library
Research Data Services

Walk-Through 1: Iterating Over a List

```
#  Define a list of colors
colors = ['red', 'green', 'blue', 'yellow']
for color in colors:
    print(color)
```

```
#  Define a list of colors
colors = ['red', 'green', 'blue', 'yellow']
for color in colors:
    print(colors)
```

Walk-Through 1.5: Creating a List with a Loop

```
# Create an empty list to store squares of numbers
squares = []
# Use a for loop to calculate the squares of numbers from 1 to 5
for number in range(1,7):
    squares.append(number ** 2)
print(squares)
```

Walk-Through 2: Find the Maximum Value in a List

```
# Given a list of numbers, write a loop to find the maximum value within a list.
numbers = [7, 3, 14, 1, 8]
max_value = numbers[0]

for num in numbers:
    if num > max_value:
        max_value = num

print("The Maximum Value is:", max_value)
```

Walk-Through 3: Nested Loops With Lists

```
# Defining a lists of lists (matrix)
matrix = [
[1, 2, 3],
[4, 5, 6],
[7, 8, 9]
]

# Use nested loops to iterate over each element of the matrix for each row in matrix:
for row in matrix:
    for element in row:
        print(element, end='')
    print()
```

in your notebook

- installing and loading packages

```
!pip install pandas
!pip install matplotlib
```

```
import pandas as pd
import matplotlib.pyplot as plt
```

- Bullet point number four.
- **as** keyword acts as an alias (shortcut)

analyzing a dataset together

- first, import data into the notebook
-
```
df =
  pd.read_csv('https://raw.githubusercontent.com/CunyLaguardiaDataAnalytics/datasets/master/faithful.csv', index_col=0)
```

- create a variable (or object) called **df** to hold our data

- **pd.read_csv** is a pre-build or pre-established function inside of the pandas library we imported/loaded

- review: anatomy of a function

analyzing a dataset together

- next, inspect the data
- why?
- it’s important to know what we are dealing with (preview the data)

```
df.head() #this shows by default, the first 5 rows of data
df.tail() #this shows by default, the first 5 rows of data
df.head(10) #this shows the first 10 rows
```

try some inspection on your own

- how many non-null datapoints are in each column?

```
df.count()
```

- finding

```
df.max()
df.min()
```

e of values

```
df.mean()
```

```
df.median()
```

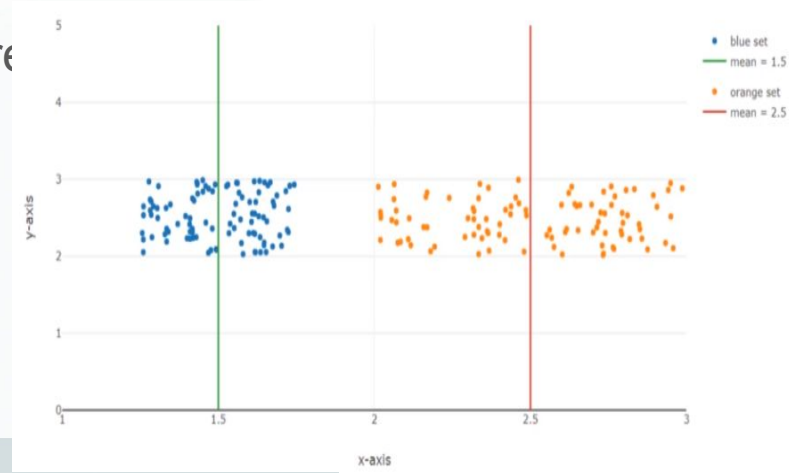
- mean (average) and median (middle)

```
df.describe()
```

- one-stop shop

side note: measures of central tendency

- mean, median, and mode
- mean is calculated average, median is middle value, and mode is most frequent value
- mean shows location, but not how spre



side note: measures of central tendency

- mean vs. median
- mean can be influenced by outliers while median is immune to outliers
- jeff bezos and michael jordan examples

columns all day all night

- columns are core to a dataframe as a defining element
- working with specific columns

```
df['column name']
```

```
df['eruptions'].mean()
```

- in this case
- what other functions can you get to work on different columns?

exploratory data analysis aka eda

- eda - summarizing main characteristics of dataset via basic visualization and statistical techniques
- helpful here to import in a visualization package like matplotlib

```
import matplotlib.pyplot as plot
%matplotlib inline
```

- how would you approach data you haven’t seen before or know nothing about?

basic data viz

- bare python has a few basic viz functions

`df.hist()` ır hand at a few

```
df.plot()
```

```
df.plot(kind="hist")
```

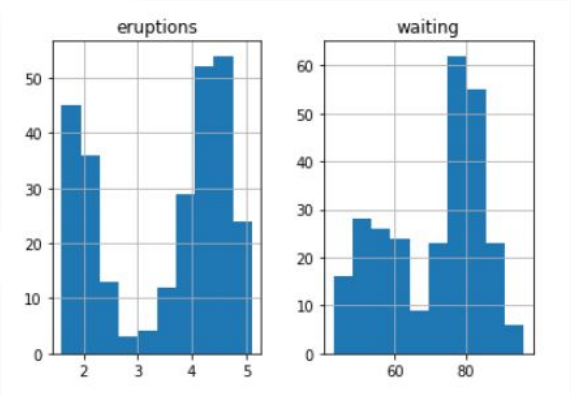
```
df.plot(kind="scatter", x='waiting', y='eruptions')
```

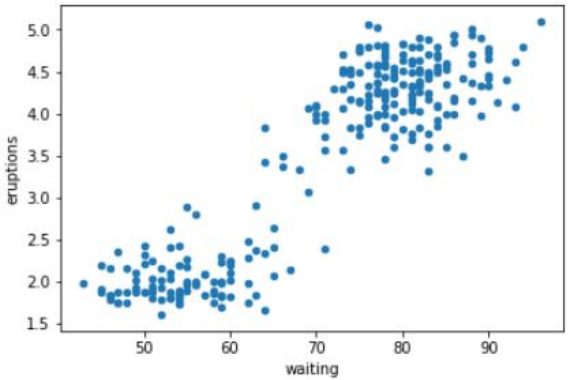
- what makes for a good data viz?

customizing a data viz

- quite a few ways to customize a data viz
- https://bit.ly/2WPTwtW
- let your inner artist and creative self shine

results of a basic eda





	eruptions	waiting
count	272.000000	272.000000
mean	3.487783	70.897059
std	1.141371	13.594974
min	1.600000	43.000000
25%	2.162750	58.000000
50%	4.000000	76.000000
75%	4.454250	82.000000
max	5.100000	96.000000