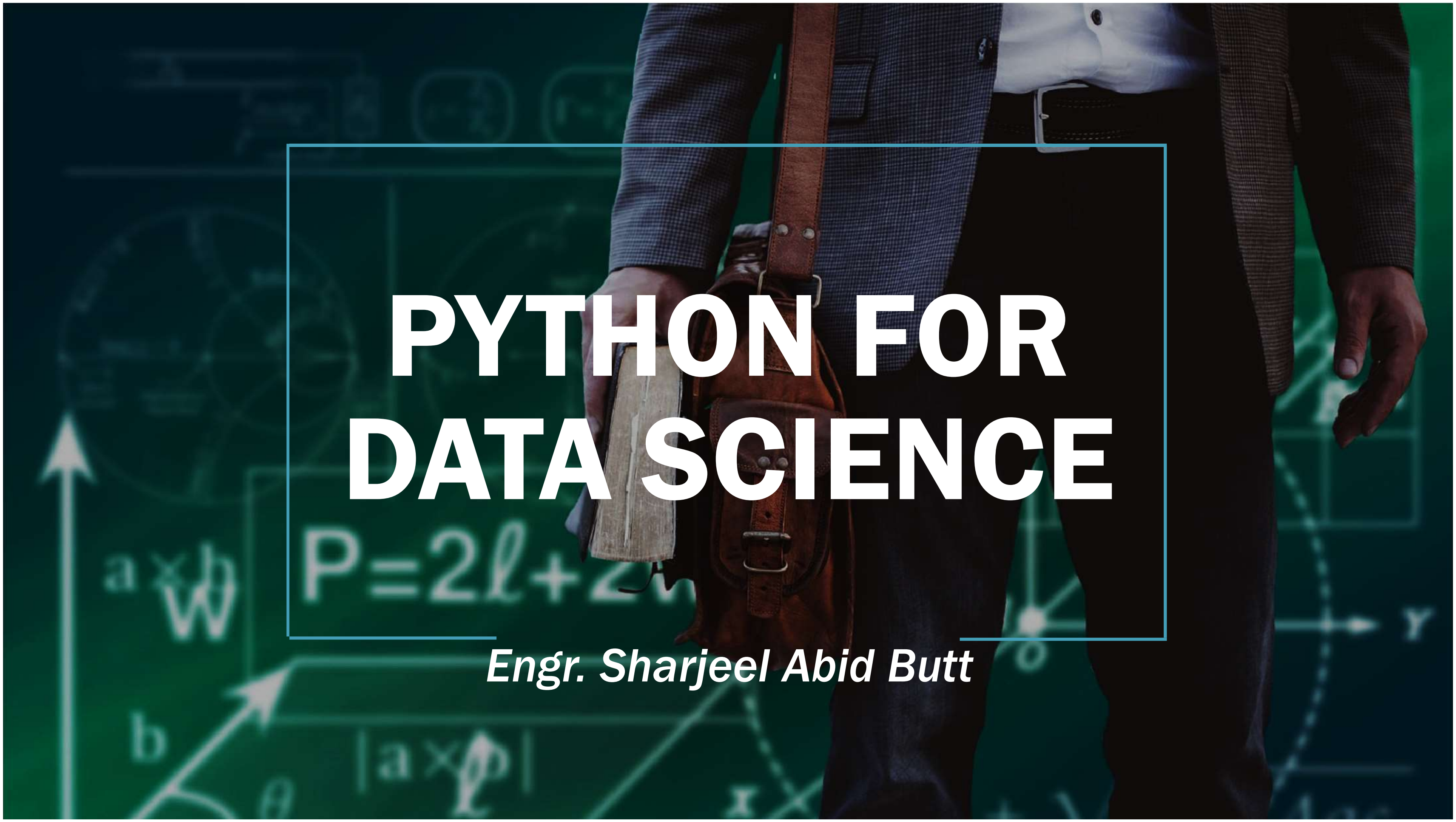


PYTHON FOR DATA SCIENCE

Engr. Sharjeel Abid Butt

About me

B.E. Electrical Engineering (2009)

Instructor @ Department of Electrical Engineering, IIUI since January 2010

MS Electronic Engineering (2014)

PhD Electronic Engineering (in progress)

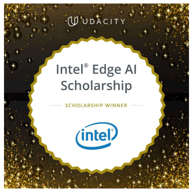
2019 Recipient **Facebook Secure & Private AI Challenge** Scholarship

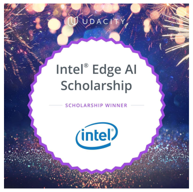
2019 Recipient **Udacity Deep Learning Nano Degree** Scholarship

2019 Recipient **Intel Edge AI Challenge** Scholarship

2020 Recipient **Udacity Intel Edge AI Nano Degree** Scholarship

UDACITY
Secure and Private AI Scholarship Challenge
2019 RECIPIENT
facebook Artificial Intelligence





Contents

- Introduction
- IPython / Jupyter: Beyond Normal Python
- Introduction to NumPy
- Data Manipulation with Pandas
- Visualization with Matplotlib

What is Data Science?

There are a few existing definitions

Obtain, **Scrub**, **Explore**, **Model**, and **iNterpret** (**OSEMN**)

Mason and Wiggins, 2010

The “ability to [create] **prototype-level** versions of … the steps needed to derive **new insights** or build **data products**”

Analyzing the Analyzers, 2013

Data Science exists to drive better outcomes

Using **multidisciplinary methods** to understand and have a positive **impact** on a **business process** or **product**

- **Route optimization** in a supply chain
- **Conjoint analysis** for product ideation
- **Attribution modeling** for connecting marketing spend to outcomes
- **Marketing spend optimization** for efficient outreach given a budget
- **Effectiveness testing** for creative or offers
- **Detecting fraud** in insurance claims
- Predicting and influencing **employee or customer retention**
- Understanding **who is likely to vote**

How do we *do* Data Science?

We **collaborate** across disciplines.

Not only do we need to speak the same **language of mathematics** we must **share similar processes and tools** to produce impactful data science.

Some of these processes and tools come from agile **product development** and **software engineering**.

Processes like **design sprints**, **project planning**, **planning poker**, and **daily standups**.

Tools like **version control**, **open source languages**, and linux **software containers**.

Why Python?

Python is one of these open source languages that you may **choose** to use.

It's a **full-featured** language with **many, many packages** for making data science tasks easier.

There are robust libraries and services for **testing** your code and methods

It makes it easy to write **defensive code**.

Readability counts and **style matters**.

Straightforward to go **from prototype to production**.

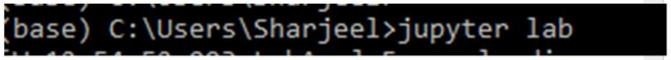
A **large community** of disciplined, helpful, and seasoned programmers.

IPython / Jupyter

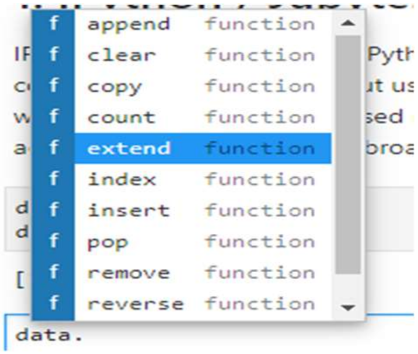
Environment:

Anaconda 3.7

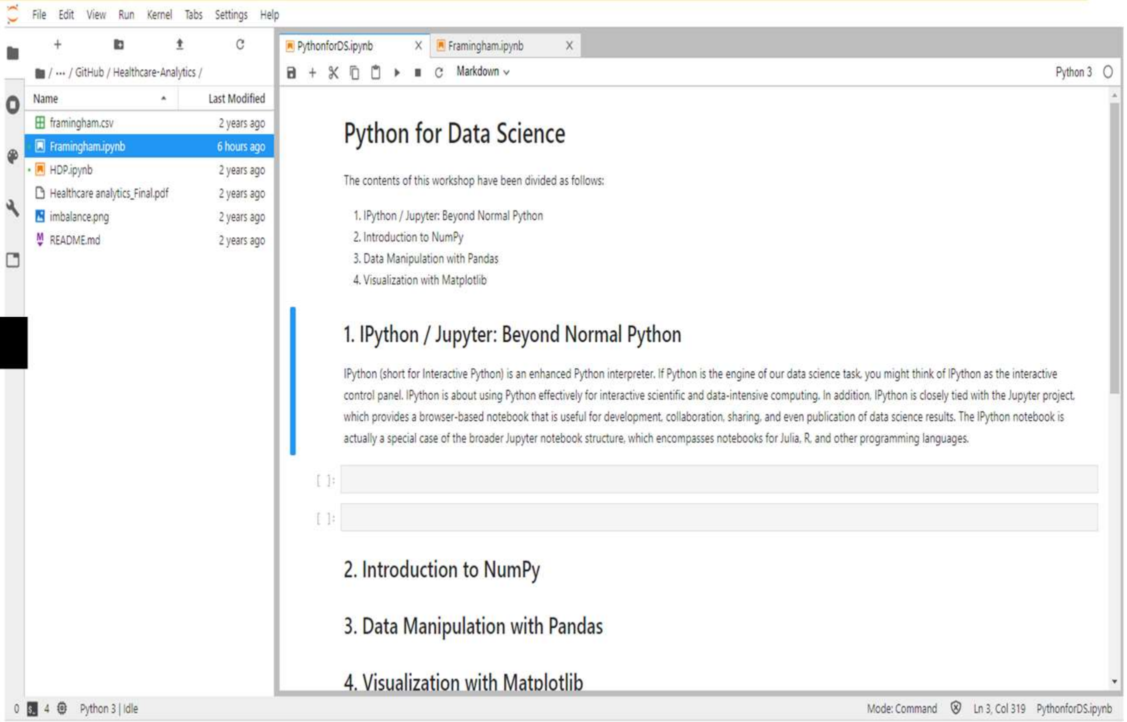
Command:



```
(base) C:\Users\Sharjeel>jupyter lab
```



```
append   function
clear    function
copy     function
count    function
extend   function
index    function
insert   function
pop      function
remove   function
reverse  function
data.
```



Name	Last Modified
framingham.csv	2 years ago
Framingham.ipynb	6 hours ago
HDP.ipynb	2 years ago
Healthcare analytics_Final.pdf	2 years ago
imbalance.png	2 years ago
README.md	2 years ago

Python for Data Science

The contents of this workshop have been divided as follows:

1. IPython / Jupyter: Beyond Normal Python
2. Introduction to NumPy
3. Data Manipulation with Pandas
4. Visualization with Matplotlib

1. IPython / Jupyter: Beyond Normal Python

IPython (short for Interactive Python) is an enhanced Python interpreter. If Python is the engine of our data science task, you might think of IPython as the interactive control panel. IPython is about using Python effectively for interactive scientific and data-intensive computing. In addition, IPython is closely tied with the Jupyter project, which provides a browser-based notebook that is useful for development, collaboration, sharing, and even publication of data science results. The IPython notebook is actually a special case of the broader Jupyter notebook structure, which encompasses notebooks for Julia, R, and other programming languages.

[ ]:

[ ]:

2. Introduction to NumPy

3. Data Manipulation with Pandas

4. Visualization with Matplotlib

For detail about Markdown cells:
https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Working%20With%20Markdown%20Cells.html

Python --> NumPy

Data in Python can be broadly classified as:

Basic data types

- Numbers: Integers and floats work as you would expect from other languages
- Booleans: Python implements all of the usual operators for Boolean logic
- Strings: Python has great support for strings

Containers

- Lists: A list is the Python equivalent of an array
- Dictionaries: stores (key, value) pairs
- Sets: A set is an unordered collection of distinct elements
- Tuples: A tuple is an (immutable) ordered list of values.

NumPy is a wrapper around a library implemented in C allowing mathematical operations not directly / easily possible in Python.

NumPy

- **Core library for scientific computing**

- main object is the homogeneous multidimensional array which is a table of elements (usually numbers), all of the same type

- NumPy functions, being compiled, execute much faster than their Python counterparts

```
import numpy as np
d1 = np.array([1,2])          # 1D array
d2 = np.array([[1,2],[10,20]]) # 2D array
```

Pandas

- Python package for providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python.

- **Pandas Series** is like generalized 1D NumPy array or a specialization of a Python dictionary.

- **Pandas DataFrame** is like generalized 2D NumPy array.

```
import pandas as pd
data = pd.Series([0.25, 0.5, 0.75, 1.0], index=['a', 'b', 'c', 'd’])
df = pd.DataFrame(np.random.rand(3, 2), columns=['foo', 'bar’], index=['a', 'b', 'c'])
```

Pandas --> Read a CSV File

CSV = Comma Separated Values
TSV = Tab Separated Values
JSON = JavaScript Object Notation

```
csv_file = open("IMDB-Movie-Data.csv")
reader = csv.reader(csv_file)
line = next(reader)
pprint(line)
```

VS

```
movies  = pd.read_csv("IMDB-Movie-Data.csv", delimiter=',')
```

Pandas --> Functions + Data Extraction

Functions:

.head()

.tail()

.info()

.describe()

.isnull()

.dropna()

Data Extraction:

Column:

data = movies[['Genre', 'Rating']]

Row:

- **.loc - loc**ates by name
- **.iloc- loc**ates by numerical **I**ndex

data = moviesT.loc['Prometheus']

data = moviesT.iloc[2]

Pandas --> SQL for Python

SQL:	Pandas:
SELECT TOP 5 * FROM movies	movies.head(5)
SELECT * FROM movies	movies
SELECT Title FROM movies	movies[[‘Title’]]
SELECT Title, Genre FROM movies	movies[[‘Title’, ‘Genre’]]
SELECT * FROM movies WHERE Year = 2014	movies[movies[‘Year’] == 2014]
SELECT * FROM movies where Year = 2014 AND Rating > 8	movies[(movies['Year'] == 2014) & (movies['Rating'] > 8)]
SELECT * FROM movies WHERE Year = 2014 OR Rating > 8	movies[(movies['Year'] == 2014) | (movies['Rating'] > 8)]

Matplotlib

- Most widely used Python visualization library/package.
- Cross-Platform
- Large Number of backends and Outputs
- Advanced usage is achieved by using Higher-level package like **Seaborn**
- Simplest plots are Line plot and Scatter plot

```
import matplotlib.pyplot as plt
plt.plot(x, y)
plt.scatter(x, y)
```

Final Words + Question/Answers