

# Week 03: Python Libraries

Data Science Bootcamp  
Summer, 2021

Instructor: Sagar Patel



# Where are we?




# Communities

- Join the **Slack** community to not miss out on any announcements and updates  
Link: [https://join.slack.com/t/nyu-dsbc-s21/shared\\_invite/zt-reskrjo0-eCwFfCmnVYnTAc82ITVNJw](https://join.slack.com/t/nyu-dsbc-s21/shared_invite/zt-reskrjo0-eCwFfCmnVYnTAc82ITVNJw)
- Share your **GitHub** Username on **#general** to be added to the NYU Data Science Bootcamp Organization where all the resources and tasks will be available after each session
  - If you do not have a GitHub account, create one!
- You can also email us at **[datasciencebootcamp@nyu.edu](mailto:datasciencebootcamp@nyu.edu)**

slido

# Name a few python libraries

 Start presenting to display the poll results on this slide.

# Agenda

- The most common libraries in Python
  - Numpy
  - Pandas
- Data Visualization
  - Matplotlib
  - Seaborn

# What are packages?

- In order to organize modules in Python, **packages** were introduced
- Kinda like **directories** on a file system, and modules as files within those directories
- Installing a package in Python
  - **pip install <package name>**
  - Example: **pip install pandas**

# What are packages?

- In order to organize modules in Python, **packages** were introduced
- Kinda like **directories** on a file system, and modules as files within those directories
- Installing a package in Python
  - **pip install <package name>**
  - Example: **pip install pandas**

**NOTE:** Depending on the version of pip installed, you might want to switch between **pip** and **pip3**

# Libraries in Python

- A reusable chunk of code that you may want to include your program/projects
- Compared to C++ or C, Python libraries do not pertain to any specific **context** in Python
- To use libraries in the program, we need to **import** them!
  - Preferably done at the start of the program or the notebook
  - **import** <module / library> as <variable>



# Libraries in Python

- A reusable chunk of code that you may want to include your program/projects
- Compared to C++ or C, Python libraries do not pertain to any specific **context** in Python
- To use libraries in the program, we need to **import** them!
  - Preferably done at the start of the program or the notebook
  - **import** <module / library> as <variable>

## JUST FOR FUN!

To list all packages installed in Python:

- `help("modules")`

# Numpy and Pandas

# Numpy

- Provides **fast** mathematical computation on **arrays** and **matrices**
- Numpy also provides the essential multi-dimensional array-oriented computing functionalities designed for high-level mathematical functions and scientific computation
- The main object of Numpy is the **homogenous multidimensional array**
  - Table(/list) with **same** type of elements, usually integers
  - Example: `a = np.array([1, 2, 3])`

# Attributes of Numpy

- **ndim** ⇐ displays the dimension of the array
- **shape** ⇐ returns a tuple indicating size of array
- **size** ⇐ total number of elements in the array
- **dtype** ⇐ returns the type of elements
- **itemsize** ⇐ size in bytes of each item
- **reshape** ⇐ reshapes the array

Numpy documentation: <https://numpy.org/>

# Understanding Pandas

- Unlike **Numpy**, **Pandas** provides provides in-memory 2-D table object called **“Dataframe”**
- Dataframes are like a spreadsheet with column names and row labels!
  - Pandas can be used to create pivot tables, computing column based operations, etc.
  - Data which can be used for plotting graphs

# Pandas “Series” object

- In a dataframe, each row is provided with an index
  - By default, they are assigned numerical values starting from 0
- A dataframe can be visualized as **dictionaries of Series**

- How do we declare it?
  - `pd.Series([val1, val2, ..., valn])`

```
>>> people_dict = { "weight": pd.Series([68, 83, 112], index=["alice", "bob",  
"charles"]), "birthyear": pd.Series([1984, 1985, 1992], index=["bob", "alice",  
"charles"], name="year"),  
"children": pd.Series([0, 3], index=["charles", "bob"]),  
"hobby": pd.Series(["Biking", "Dancing"], index=["alice", "bob"]), }
```

|         | birthyear | children | hobby   | weight |
|---------|-----------|----------|---------|--------|
| alice   | 1985      | NaN      | Biking  | 68     |
| bob     | 1984      | 3.0      | Dancing | 83     |
| charles | 1992      | 0.0      | NaN     | 112    |

# What to use when working with dataframes?

- **head()** ⇐ returns the top 5 rows in the object
- **tail()** ⇐ returns the bottom 5 rows in the dataframe
- **info()** ⇐ summary of the dataframe
- **describe()** ⇐ overview of the aggregated values

Documentation can be found here <https://pandas.pydata.org/>

# Data Visualization



# What is matplotlib?

- **matplotlib** is a 2D **plotting library**
- Can be used in Python scripts, Python shell, notebooks and web applications
- **matplotlib.pyplot** makes **matplotlib** work like **MATLAB**
  - If you are familiar with MATLAB, the commands are fairly intuitive
- Documentation can be found at <https://matplotlib.org/>

# Seaborn: statistical data visualization

- Based on **matplotlib**, seaborn is a Python data visualization library
  - Integrates closely with **Pandas** data structures
- Seaborn documentation: <https://seaborn.pydata.org/>

# matplotlib vs seaborn

- seaborn utilizes **fascinating themes**, while matplotlib is used for **basic graphs**
- matplotlib is profoundly **robust and customizable**, seaborn is not (entirely)!
- seaborn is more **agreeable** in taking care of dataframes in Pandas  
matplotlib, with both Pandas and Numpy goes about as a **graphics package** for Python

# Summary

- Numpy and Pandas make matrix multiplication easy
  - This makes them very useful for Machine Learning model development
- The utilization of seaborn or matplotlib exclusively relies on the motivation of plotting
- However, seaborn is more aesthetic!

# That's all Folks!

See you in the next session :)

Give us a feedback: <https://bit.ly/3q6ZDID>