



# PYTHON FOR DATA SCIENCE

NECESSARY SKILLS  
TO  
STEP INTO AI WORLD



# WORKING WITH DATABASES – P1

- Introduction to Databases
- Database Concepts
- Keys
- Introduction to SQLite
- CRUD on SQLite
- Introducing 'Cursor'

```
response = requests.get(url)

# Print the status code (if you want)
print(response.status_code)

# Print the status code (if you want)
print(f"Status: {response.status_code}")

# Use BeautifulSoup to parse the response
soup = BeautifulSoup(response.content, "html.parser")

# Find all images in the soup
images = soup.find_all("img", attrs={"alt": "image"})

# Print the number of images
print(len(images))
```

# WORKING WITH DATABASES – P2

- Tools of the Trade in Big Databases
- Configuring MySQL
- Connection Error Handling
- CRUD in MySQL

```
requests.get(url)

ing response.status_code (if you g
onse.status_code != 200:
nt(f"Status: {response.status_code

nt(f"Status: {response.status_code

BeautifulSoup to parse the respons
BeautifulSoup(response.content, "ht

ing Post images in the soup
= soup.find_all("img", attrs={"alt

loading images
= 0
images:
```



# INTRODUCING API CONCEPTS – P1

- Introduction to API
- Discussing Rest-API
- API Call
- Getting to Know JSON Format
- Working with Restful APIs
- Project: Getting Data From 'NASA' Mars Rover

```
requests.get(url)

ing response.status_code (if you g
onse.status_code != 200:
nt(f"Status: {response.status_code

nt(f"Status: {response.status_code

BeautifulSoup to parse the respons
BeautifulSoup(response.content, "ht

ing Post images in the soup
= soup.find_all("img", attrs={"alt

Loading images
= 0
images:
```

# INTRODUCING API CONCEPTS – P2

- Concepts of fastAPI
- Writing Your First API
- Path Parameters
- Enumerations
- Query Parameters

```
requests.get(url)

ing response.status_code (if you g
onse.status_code != 200:
nt(f"Status: {response.status_code

nt(f"Status: {response.status_code

BeautifulSoup to parse the respons
BeautifulSoup(response.content, "ht

ing Post images in the soup
= soup.find_all("img", attrs={"alt

Loading images
= 0
images:
```



# INTRODUCING NUMPY

- Jupyter Notebook
- Introducing Numpy
- Creating a Numpy Array
- Numpy Array vs List
- Calculating Norm and Inner Product
- Matrices in Numpy
- Solving Linear Systems in Numpy

```
...load from the websi  
e = requests.get(url)  
  
ing response.status_code (if you g  
onse.status_code != 200:  
nt(f"Status: {response.status_code  
  
nt(f"Status: {response.status_code  
  
BeautifulSoup to parse the respons  
BeautifulSoup(response.content, "ht  
  
ing Post images in the soup  
= soup.find_all("img", attrs={"alt"  
  
loading images  
= 0  
images:
```

# INTRODUCING MATPLOTLIB

- Getting to Know Charts
- Histograms
- Piechart
- Boxplot
- Errorbar

```
response = requests.get(url)

# Print the status code (if you want)
print(response.status_code)

# Print the status code (if you want)
print(f"Status: {response.status_code}")

# Use BeautifulSoup to parse the response
soup = BeautifulSoup(response.content, "html.parser")

# Find all images in the soup
images = soup.find_all("img", attrs={"alt": "image"})

# Print the number of images
print(len(images))
```



# INTRODUCING PANDAS

- Reading Files Into Pandas
- Matrix Manipulation in Pandas
- Introducing Data Frames
- Working With Rows and Columns

```
response = requests.get(url)

if response.status_code != 200:
    print(f"Status: {response.status_code}")

print(f"Status: {response.status_code}")

BeautifulSoup to parse the response
BeautifulSoup(response.content, "html")

Loading Post images in the soup
images = soup.find_all("img", attrs={"alt": "image"})

loading images
images = []
```



# INTRODUCING SCIKIT-LEARN

- Data Cleaning
- Data Encoding
- Re-scaling Data
- Introducing Outliers
- Outlier Detection
- Project: Avocado Price Estimation

```
response = requests.get(url)

if response.status_code != 200:
    print(f"Status: {response.status_code}")

print(f"Status: {response.status_code}")

BeautifulSoup to parse the response
soup = BeautifulSoup(response.content, "html.parser")

# Extracting Post images in the soup
images = soup.find_all("img", attrs={"alt": "Post image"})

# Loading images
images = []
```

# SCIPY

- Quick Intro to Scipy

```
response = requests.get(url)

# Print the status code (if you get a 404, it means the page is not found)
print(response.status_code)

# Print the status code (if you get a 404, it means the page is not found)
print(f"Status: {response.status_code}")

# Use BeautifulSoup to parse the response
soup = BeautifulSoup(response.content, "html.parser")

# Find all images in the soup
images = soup.find_all("img", attrs={"alt": "image"})

# Print the number of images found
print(len(images))
```



# CRAWLING

- What Are Crawlers?
- Selenium
- Quick Overview on CSS Styling
- Quick Overview on HTML Tags
- Extracting the Exact Information From Websites
- Project: Crawling Digikala

```
response = requests.get(url)

# Print response.status_code (if you get a 404, it means the page does not exist)
print(response.status_code)

# Print response.status_code (if you get a 404, it means the page does not exist)
print(f"Status: {response.status_code}")

# Parse the response with BeautifulSoup
soup = BeautifulSoup(response.content, "html.parser")

# Find all images in the soup
images = soup.find_all("img", attrs={"alt": "image"})

# Print the number of images found
print(len(images))
```

# FINAL PROJECT

- Dataset Introduction
- Data Pre-processing Modules
- KNN Estimator
- Confusion Matrix of KNN Results
- Creating CNN Model
- CNN Results

```
...load from the websi  
e = requests.get(url)  
  
ing response.status_code (if you g  
onse.status_code != 200:  
nt(f"Status: {response.status_code  
  
nt(f"Status: {response.status_code  
  
BeautifulSoup to parse the respons  
BeautifulSoup(response.content, "ht  
  
ing Post images in the soup  
= soup.find_all("img", attrs={"alt"  
  
Loading images  
= 0  
images:
```



# BONUS

- Getting to Know Streamlit
- Creating Web Based UI
- Backend Handling
- Model Serialisation
- Sending Requests to Server
- Updating UI Based on Server Response

```
response = requests.get(url)

if response.status_code != 200:
    print(f"Status: {response.status_code}")

print(f"Status: {response.status_code}")

BeautifulSoup to parse the response
BeautifulSoup(response.content, "html")

Find Post images in the soup
images = soup.find_all("img", attrs={"alt": "Post image"})

Loading images
images = []
```

# NOTES

## WEBSITES

- [www.python.org](http://www.python.org)
- <https://scikit-learn.org/stable>
- <https://numpy.org/>

## TEXTS

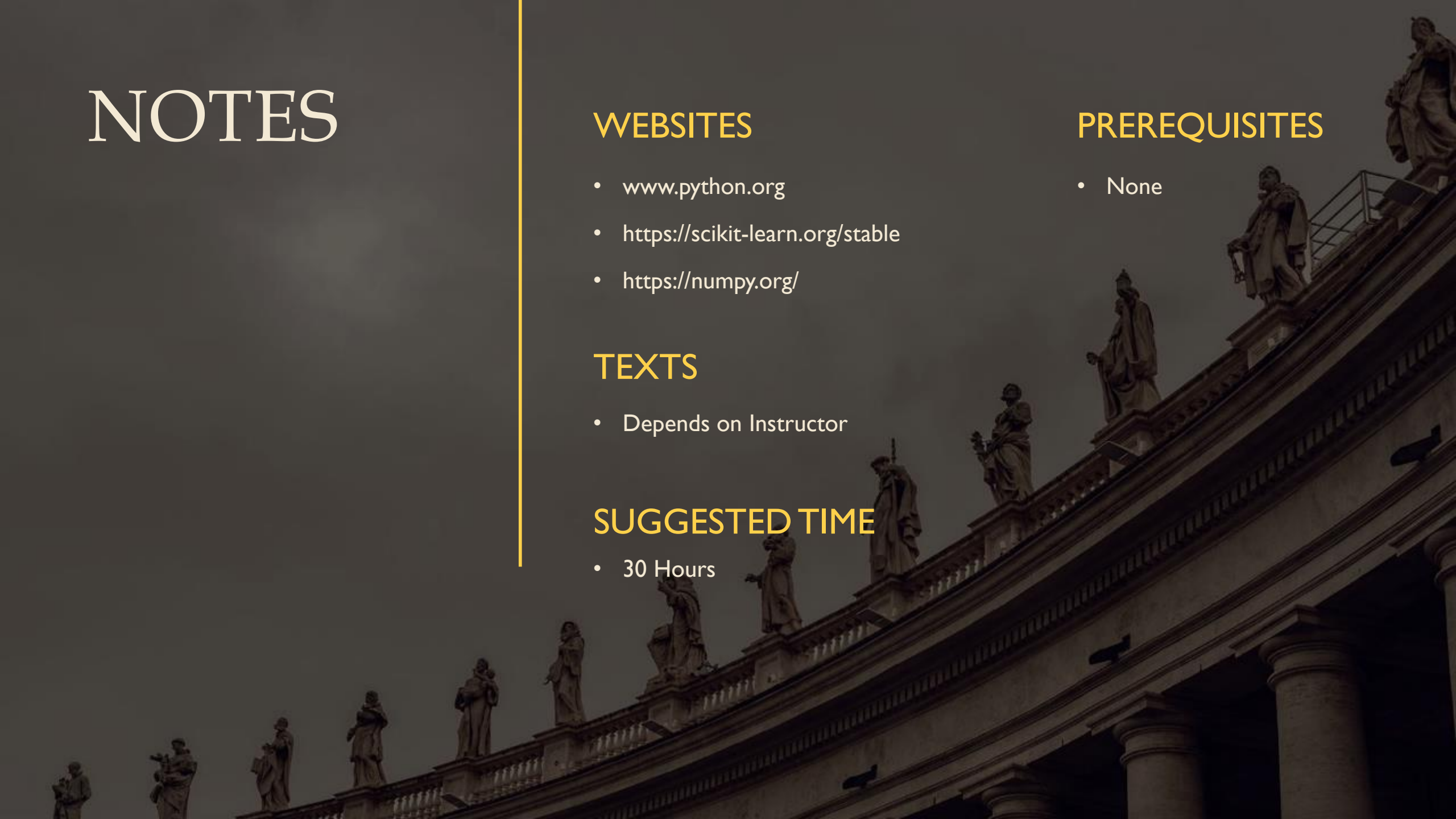
- Depends on Instructor

## SUGGESTED TIME

- 30 Hours

## PREREQUISITES

- None





# DOCUMENT HISTORY

| Author       | Version | Revision | Date / Time | Department | Validity |
|--------------|---------|----------|-------------|------------|----------|
| Mehdi Shokri | 1.0.0   |          | 14-05-2023  | AI         | 3 Months |
|              |         |          |             |            |          |
|              |         |          |             |            |          |
|              |         |          |             |            |          |
|              |         |          |             |            |          |
|              |         |          |             |            |          |

