DATA SCIENCE

DL-2001

Lab 01

Introduction to DataScience and Python Programming

National University of Computer & Emerging Sciences – NUCES – Karachi

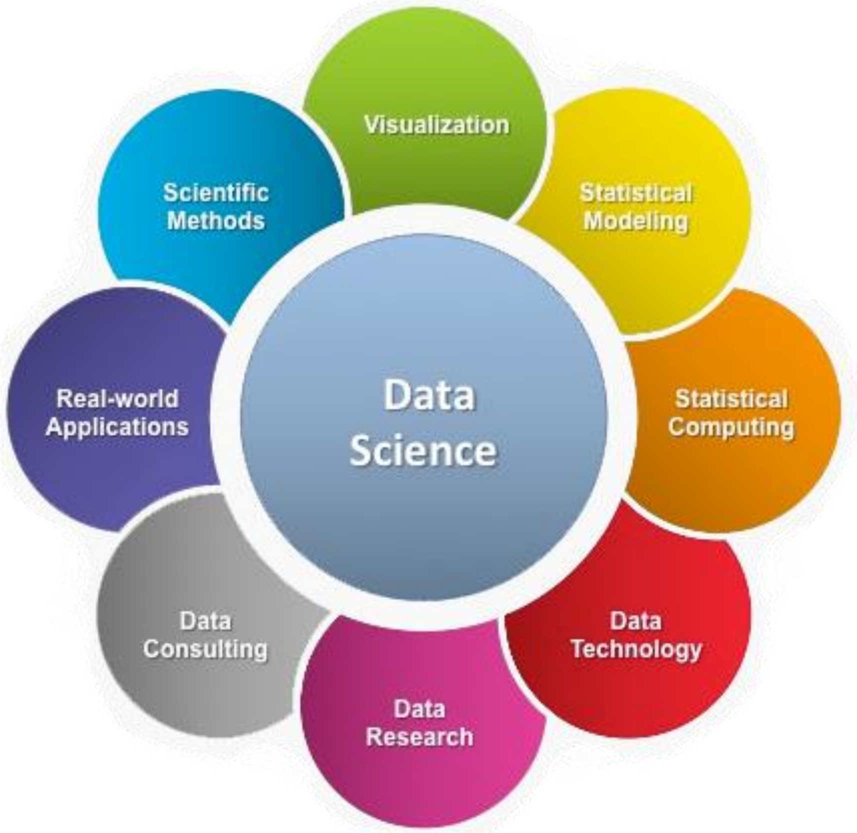
Course Code:	DataScience Lab

1. Objectives

- Introduction to Data science
- Setup python for DS
- Python basics

2. Introduction Data Science

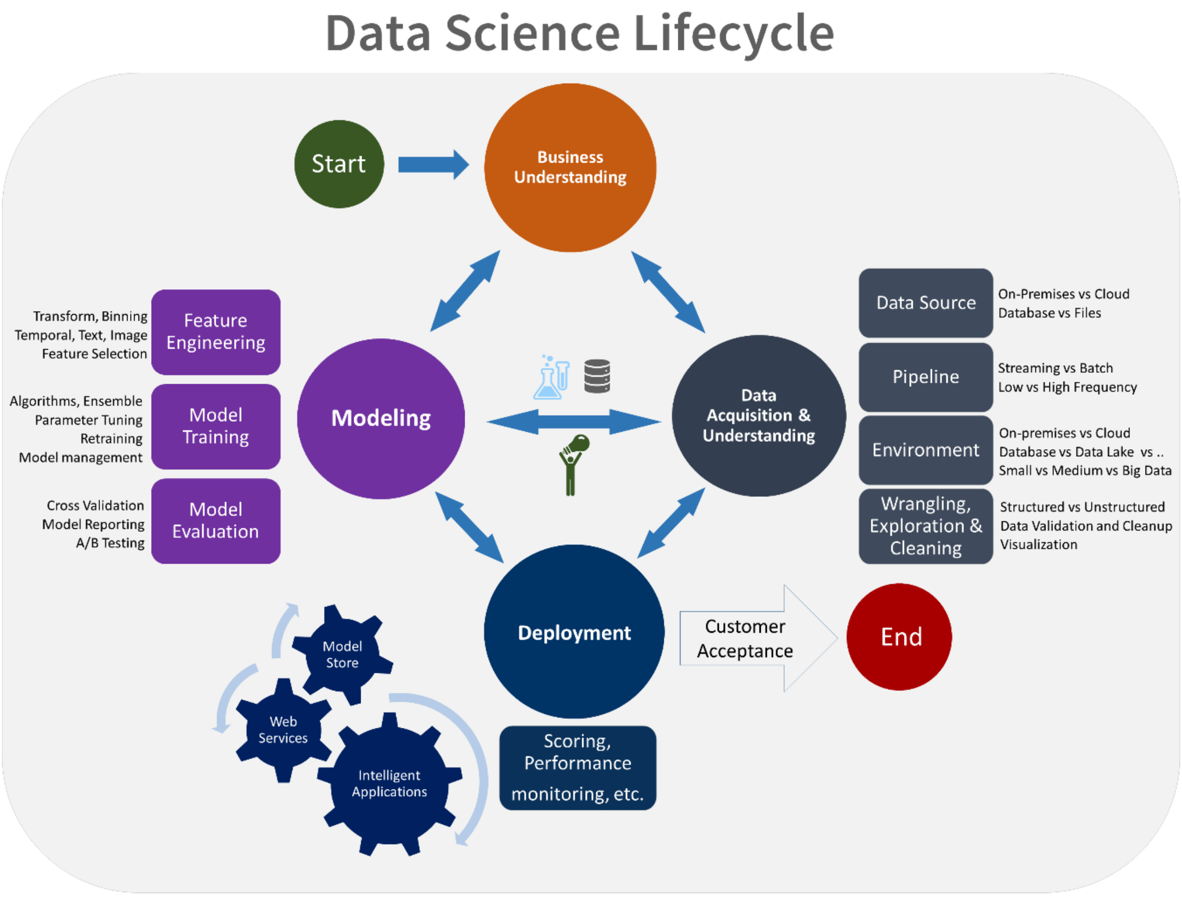
- Data science is the practice of mining large data sets of raw data, both structured and unstructured, to identify patterns and extract actionable insight from them. This is an interdisciplinary field, and the foundations of data science include statistics, inference, computer science, predictive analytics, machine learning algorithm development, and new technologies to gain insights from big data.



3. Data Science Lifecycle

The Data Science Life Cycle is the process that guides how data is collected, analyzed, and used for insights. It usually includes these main steps:

1. **Data Collection** – Gathering raw data from various sources.
2. **Data Preparation** – Cleaning, transforming, and organizing data.
3. **Data Analysis & Modeling** – Applying statistical and machine learning techniques.
4. **Interpretation & Visualization** – Presenting results in understandable form.
5. **Deployment & Monitoring** – Using the model in real-world applications and improving it over time.



4. Tools for Data Science

1. Data Storage & Collection Tools

- **Databases:** MySQL, PostgreSQL, MongoDB

- **Big Data Storage:** Hadoop, Spark, Google BigQuery
- **Cloud Storage:** AWS S3, Azure Blob, Google Cloud Storage

2. Data Preparation & Cleaning Tools

- **Python Libraries:** Pandas, NumPy
- **R Packages:** dplyr, tidyr
- **ETL Tools:** Talend, Apache Nifi

3. Coding / Programming Tools

- **Languages:** Python, R, SQL
- **IDEs/Notebooks:** Jupyter Notebook, Google Colab, RStudio, VS Code

4. Data Visualization Tools

- **Python Libraries:** Matplotlib, Seaborn, Plotly
- **BI Tools:** Tableau, Power BI, Google Data Studio

5. Modeling & Machine Learning Tools

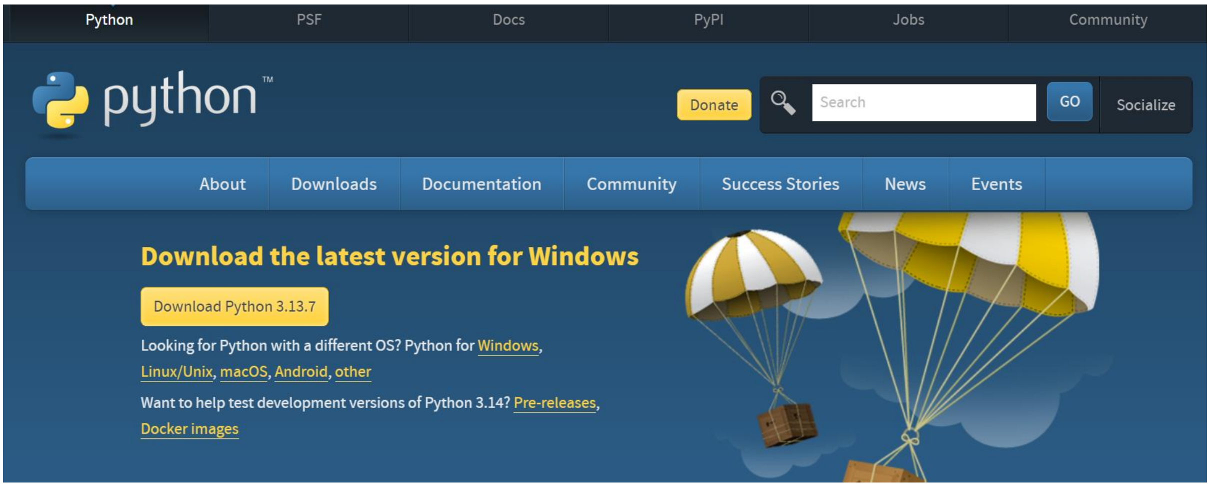
- **Python ML Libraries:** Scikit-learn, TensorFlow, PyTorch, Keras
- **R ML Libraries:** caret, randomForest, xgboost

6. Deployment & Monitoring Tools

- **Web Frameworks:** Flask, FastAPI, Django
- **Cloud Platforms:** AWS (SageMaker), Azure ML, Google AI Platform
- **Containerization:** Docker, Kubernetes

5. Installation guidelines

Step 1:
Download and Install Python 1. Download the latest python version from here: https://www.python.org/downloads/ (Note: At the time this tutorial was made, the latest was 3.7.2. You can download the latest version or use any version that works for you.)



Download the latest version for Windows

Looking for Python with a different OS? Python for Windows, Linux/Unix, macOS, Android, other

Want to help test development versions of Python 3.14? Pre-releases, Docker images

6. Install for all users and add to path - Make sure both boxes are checked
7. Follow the steps and install the setup.
8. Open terminal type python it will open python shell where python script can be run directly.

You can install Vscode IDE for code or jupter note book would be suitable for data science tasks.
Links for the installation given below
Vscode
https://www.youtube.com/watch?v=9o4gDQvVkLU
Anaconda jupter notebook
https://www.youtube.com/watch?v=4DQGBQMvwZo

Python

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

- web development (server-side),
- software development,
- mathematics,
- system scripting.
- Datascience

Variables

Variables do not need to be declared with any particular *type*, and can even change type after they have been set.

```python
# Different types of variables
name = "Ali"           # string
age = 20               # integer
gpa = 3.45             # float
is_student = True      # boolean
subjects = ["AI", "DS", "ML"]    # list
roll_tuple = (123, "CS")         # tuple
student_info = {"name": "Ali", "age": 20, "gpa": 3.45}  # dictionary

print("Global variable (University):", university)
print("String:", name)
print("Integer:", age)
print("Float:", gpa)
print("Boolean:", is_student)
print("List:", subjects)
print("Tuple:", roll_tuple)
print("Dictionary:", student_info)

# Function demonstrating local vs global variables
def student_report():
    # Local variable (exists only inside function)
    status = "Active"

    # Access global variable using 'global' keyword
    global university
    university = "FAST-NUCES"   # modifying global variable

    print("\nInside Function:")
    print("Local variable (status):", status)
    print("Accessed Global variable (university):", university)

# Call function
student_report()

print("\nOutside Function:")
print("Modified Global variable (university):", university)  # changed inside function

```

Input and output

- input() function is used to take user input.
- Input is always read as string.
- `print()` is used to display output.
- Multiple items can be printed with commas.

String Formatting

Python provides several ways to format strings:

- f-strings (Python 3.6+) → f"{variable}"
- .format() method → "{}".format(variable)
- % formatting → "%s" % variable

```python
name = "Ali"
age = 20
gpa = 3.45

# f-string
```

```python
print(f"My name is {name}, I am {age} years old, GPA = {gpa}")

# format method
print("My name is {}, I am {} years old, GPA = {}".format(name, age, gpa))

# % formatting
print("My name is %s, I am %d years old, GPA = %.2f" % (name, age, gpa))
```

Math Module

```python
import math

x = 25
y = 3

print("Square Root of 25:", math.sqrt(x))
print("Power 25^3:", math.pow(x, y))
print("Factorial of 5:", math.factorial(5))
print("Value of PI:", math.pi)
print("Value of e:", math.e)

# Trigonometry
angle = 45
print("sin(45 degrees):", math.sin(math.radians(angle)))
print("cos(45 degrees):", math.cos(math.radians(angle)))

# Logarithms
print("Natural log of 10:", math.log(10))
print("Log base 10 of 1000:", math.log10(1000))

# Rounding
print("Ceil of 3.2:", math.ceil(3.2))
print("Floor of 3.8:", math.floor(3.8))
```

String Operations

```python
s = "Data Science"
print("First char:", s[0])
print("Last char:", s[-1])
print("Slice [0:4]:", s[0:4])
print("Length:", len(s))
print("Uppercase:", s.upper())
print("Lowercase:", s.lower())
print("Replace:", s.replace("Data", "AI"))
```

IF-Else

```python
marks = int(input("Enter marks: "))
if marks >= 90:
    print("Grade A")
elif marks >= 80:
    print("Grade B")
elif marks >= 70:
    print("Grade C")
else:
    print("Fail")
```

LOOPS

```

print("\n1. For loop with range()")
for i in range(5):
    print("Iteration:", i)

print("\n2. For loop with start, end, step")
for i in range(1, 11, 2):
    print(i)

print("\n3. For loop over a list")
subjects = ["Math", "AI", "DS", "ML"]
for sub in subjects:
    print("Subject:", sub)

print("\n4. For loop with index (enumerate)")
for i, sub in enumerate(subjects, start=1):
    print(f"{i}. {sub}")

print("\n5. While loop (condition based)")
n = 5
while n > 0:
    print("Countdown:", n)
    n -= 1

print("\n6. Infinite loop with break (stops at 3 inputs)")
count = 0
while True:
    num = count + 1   # fake user input for demo
    print("You entered:", num)
    if num == 3:
        break
    count += 1

print("\n7. Using continue (skip iteration)")
for i in range(1, 6):
    if i == 3:
        continue
    print(i)

print("\n8. Nested loops")
for i in range(1, 4):
    for j in range(1, 4):
        print(f"i={i}, j={j}")

print("\n9. For loop with else")
for i in range(3):
    print("Value:", i)
else:
    print("For loop ended successfully")

print("\n10. While loop with else")
m = 3
while m > 0:
    print("m =", m)
    m -= 1
else:
    print("While loop finished")

print("\n11. Pattern printing with nested loops")
```

```
rows = 4
for i in range(1, rows+1):
    for j in range(i):
        print("*", end="")
    print()

```

Lab Tasks

1. Create three variables: a string representing a number (e.g., "123.45"), an integer (e.g., 10), and a boolean indicating if the number is positive. Perform the following: convert the string to a float, multiply it by the integer, and use the boolean to print whether the result is positive or negative. Handle potential errors in string conversion.

2. Prompt the user to input a value that could be an integer, float, or invalid input. Convert the input to an integer if possible; if it's a float, round it to the nearest integer; if it's invalid, set it to 0. Print the final integer and its original type.

3. Ask the user to input the names and prices of three items (as strings, e.g., "Apple, 2.99"). Parse the input to extract item names and prices, then print a formatted receipt with total cost, aligning item names and prices in columns (use string formatting for alignment).

4. Prompt the user to input coefficients (a), (b), and (c) for a quadratic equation ((ax^2 + bx + c = 0)). Calculate the discriminant ((b^2 - 4ac)) and print the real roots (if they exist) using the quadratic formula. Handle cases where the discriminant is negative.

5. Ask the user to input a string. Check if it's a palindrome (ignoring spaces, punctuation, and case) and count the number of vowels (a, e, i, o, u). Print both results.

6. Prompt the user to input their exam score (0–100) and attendance percentage (0–100). Assign a grade (A: ≥90, B: ≥80, C: ≥70, D: ≥60, F: <60) but penalize one grade lower if attendance is below 75%. If attendance is below 50%, assign F regardless of score. Print the final grade.

7. Use nested loops to print a right-angled triangle of numbers, where each row (i) contains numbers from 1 to (i). For example, for 4 rows:
   1
   1 2
   1 2 3
   1 2 3 4
   Prompt the user for the number of rows (1–10) and print the pattern.

8. Prompt the user for a positive integer (n). Calculate the sum of the series (1 + 1/2 + 1/3 + \ldots + 1/n) using a loop. Print each term and the final sum, rounded to 2 decimal places.