**ASSIGNMENT-4**

Virtual Environment Setup

**1. Explain the concept of a virtual environment in Python.**

A virtual environment in Python is an isolated environment that allows you to manage dependencies and packages for different projects separately. It creates a self-contained directory that includes the Python interpreter and libraries specific to your project, ensuring that dependencies do not interfere with each other.

**2. Why is it important, especially in data science projects?**

1. Dependency Management: Data science projects often require different versions of libraries. Virtual environments allow you to manage these dependencies separately for each project, avoiding conflicts.

2. Reproducibility: By isolating project-specific dependencies, you can ensure that your code runs consistently across different environments, making it easier to reproduce results.

3. Clean Environment: Virtual environments provide a clean slate for each project, preventing clutter from unnecessary libraries and ensuring only the required packages are installed.

4. Version Control: You can use virtual environments to maintain specific versions of libraries needed for your project, which is particularly useful for collaborating with others or deploying your project.

5. System Integrity: By isolating project dependencies, you prevent potential conflicts with system-wide packages, keeping your system’s Python installation clean and stable.

**3. Create a virtual environment named "ml\_env" using the `venv` module (for Python 3.3 and later).**

Step-by-Step Guide

1. Create a Virtual Environment

Open your terminal or command prompt and navigate to your project directory. Then, run the following command to create a virtual environment named "ml\_env":

```bash

python -m venv ml\_env

```

2. Activate the Virtual Environment

- On Windows:

```

ml\_env\Scripts\activate

```

- On macOS and Linux:

```

source ml\_env/bin/activate

```

3. Install a Python Library

Once the virtual environment is activated, you can install any Python library using pip. For this example, we will install NumPy:

```

pip install numpy

```

4. Write a Python Script

Create a Python script named `example.py` that imports NumPy and performs a basic operation, such as creating an array:

```python

import numpy as np

Create a 1D NumPy array

array = np.array([1, 2, 3, 4, 5])

print("NumPy Array:", array)

```

5. Run the Script

Run the script within the activated virtual environment:

```

python example.py

```

You should see the following output:

```

NumPy Array: [1 2 3 4 5]

```