Environment Setup and Best Practices

Introduction

It's important to set up a proper working environment. This ensures that everyone is using the same set of tools and libraries, making it easier to collaborate and share code. In this practical assignment , you will learn setting up a Python virtual environment and installing the necessary packages.

Table of Contents

Exercise 0: Setting up a Python Virtual Environment

Objective:

Set up a Python virtual environment to isolate project dependencies.

Option 1: Using `venv`

1. Open your terminal and navigate to the project directory.

2. Run the following command to create a virtual environment named `Tp1`:

```
python3 -m venv Tp1
```

3. Activate the virtual environment:

- In case you are using *Linux/MacOS:*

  ```
  source Tp1/bin/activate
  ```

- In case you are using *Windows*

```
.\Tp1\Scripts\activate
```

Option 2: Using `conda`

1. Open your terminal and navigate to the project directory.

2. Run the following command to create a new conda environment named `Tp1`:

```
conda create --name Tp1 python=3.9
```

3. Activate the conda environment:

```
conda activate Tp1
```

Exercise 0.1: Installing Required Packages

Objective:

Install the required Python packages.

1. While your virtual environment is activated, install the necessary packages using `pip` or conda in case you are using conda env :

```
pip install numpy pandas matplotlib seaborn scikit-learn tensorflow
```

or

```
conda  install numpy pandas matplotlib seaborn scikit-learn tensorflow
```

Exercise 0.2: Creating a `requirements.txt` File

Objective:

Create a `requirements.txt` file for easier dependency management and portability.

Option 1: Using `requirements.txt` for `tp1`venv

1. Generate a `requirements.txt` file by running:

```
pip freeze > requirements.txt
```

2. Whenever someone needs to run your project, they can install all the dependencies using:

```
pip install -r requirements.txt
```

Option 2: Using `environment.yml` for `conda`

1. Generate an `environment.yml` file by running:

activated a Conda environment by running `conda activateTp1` , then run:

```
conda env export > environment.yml
```

This will export the packages for Tp1 into environment.yml.

2. To recreate the environment on a different machine or by another team member, run:

```
conda env create -f environment.yml
```

By completing these steps, you configured a development environment that is isolated from other projects. This is considered a best practice in Python development, and is especially important in data science and MLOps projects.