TP 1: Introduction to Data Pipelines with Data Lakes

EFREI Course 2024-2025

Introduction

This lab introduces you to the basic principles of data pipelines in the context of Data Lakes. You will learn to:

- Create an isolated Python environment with Conda.
- Manage a Git project by forking and cloning a repository.
- Handle raw data to organize it in a Bronze layer.
- Preprocess this data to prepare it for analysis in a Silver layer.

Exercise 1: Setting Up the Environment

Objective

Create an isolated Python environment and clone the Git repository containing the necessary files for the lab.

Steps

1. **Install Miniconda**: Go to the official Miniconda website and download the installer for your operating system. Follow the installation instructions.
2. **Create the Environment**: After installing Miniconda, open a terminal and run the following commands:

```
# Update conda
conda update -n base -c defaults conda

# Create an environment named 'data-lakes'
conda create -n data-lakes python=3.9

# Activate the environment
conda activate data-lakes
```

3. **Fork and Clone the GitHub Repository (click here to be redirected to the Repository)** :
   - Go to the GitHub repository.
   - Click the Fork button to copy the project to your GitHub account.
   - Clone your fork locally using:

```
git clone https://github.com/YourUsername/Data-Lakes.git
```

4. **Install Dependencies**:

```
cd Data-Lakes
pip install -r build/requirements.txt
```

Exercise 2: Developing the `unpack_data` Function

Objective

Complete the `unpack_data` function in `build/unpack_data.py` to decompress and combine multiple CSV files.

Context

This function processes raw data and saves it to the Bronze layer of the Data Lake. CSV files must be combined into one single file to ease the following pipeline steps.

Instructions

1. Download the dataset from Kaggle and extract the contents (train, test, val, random split) into a new folder named `data/bronze`.
2. Open the file `build/unpack_data.py` and complete the empty function. You can follow the steps described in the function's docstring:

```
def unpack_data(input_dir, output_file):
    """
    Unpacks and combines multiple CSV files from a directory into a single CSV file.
    Parameters:
    input_dir (str): Path to the directory containing the CSV files.
    output_file (str): Path to the output combined CSV file.
    """
    # Step 1: Traverse all files in the directory
    # Step 2: Validate that the files are .csv
    # Step 3: Read each file using pandas and add to a list
    # Step 4: Combine all DataFrames
    # Step 5: Save the combined file
    pass
```

Exercise 3: Data Preprocessing

Objective

Complete the `preprocess_data` function in `src/preprocess.py` to transform raw data into data ready for analysis.

Context

This function processes data from the Bronze layer and prepares it for the Silver layer. It includes steps such as:

- Removing missing values.
- Encoding categories.
- Splitting the data into training, validation, and test sets.

Instructions

1. Open the file `src/preprocess.py` and complete the function `preprocess_data`. You may use the steps indicated in the function's docstring:

```
def preprocess_data(data_file, output_dir):
    """
    Preprocesses the data for model training and evaluation.
    Parameters:
    data_file (str): Path to the raw data file.
    output_dir (str): Directory to save the processed files.
    """
    # Step 1: Load the data with pandas
    # Step 2: Drop missing values
    # Step 3: Encode 'family_accession' column using LabelEncoder
    # Step 4: Split data into train/dev/test
    # Step 5: Save preprocessed files
    pass
```

2. Run and check `data_analysis.ipynb` to verify your preprocessing is correct.

Conclusion

By the end of this lab, you should have:

- Set up an isolated Python environment for managing your projects.
- Developed a function to integrate and structure raw data.
- Transformed this data to make it ready for analysis or modeling.

If you have any questions, feel free to reach out!