Week 1 Lab Session: Hands-On Machine Learning Environment Setup and Exploratory Data Profiling

This **Detailed Lab Worksheet** is designed for the practical session of **Machine Learning with Python for Managers**. It provides a step-by-step technical guide for establishing a robust local data science workstation and profiling standard practice datasets. The goal of this lab is to move from conceptual understanding to hands-on configuration, ensuring managers can navigate the tools and data pipelines used by technical teams.

1. Lab Session Overview & Objectives

By completing this 60-minute hands-on lab, you will be able to:

1. **Configure a Virtual Environment** using Anaconda and Conda CLI tools to isolate project dependencies.
2. **Deploy Jupyter Notebook** and configure a structured project directory for file management.
3. **Analyze and Map Practice Datasets** to specific strategic business challenges and learning paradigms.
4. **Execute Basic Exploratory Loading Commands** using standard Python libraries (Pandas, NumPy, and Scikit-learn).

PART I: Workspace Setup & Environment Configuration (20 Minutes)

Data science projects rely on hundreds of external packages, each with different version requirements. If not managed carefully, library version clashes can corrupt development environments. To prevent "it worked on my machine" errors, you will install Anaconda and configure isolated **Virtual Environments**.

Step 1: Download Anaconda Navigator

1. Open your web browser and navigate to the official Anaconda Navigator download page: https://www.anaconda.com/products/navigator.
2. Download the appropriate graphical installer for your operating system (Windows, macOS, or Linux) and select your system's architecture (64-bit or 32-bit).
3. Run the installer and proceed through the prompts:
    - Read and accept the software licensing conditions by clicking - **I agree**.
    - When prompted for installation type, select **Just Me** (recommended as it allows installation without administrative privileges).
    - Select your desired destination folder path and click **Install**.

Step 2: Establish an Isolated Virtual Environment via Conda CLI

Once installation is complete, you must set up an isolated workspace for your machine learning projects.

1. Launch your command-line interface:
   Windows: Search for **Anaconda Prompt** in the Start Menu.
   macOS/Linux: Open the standard **Terminal** app.
2. Verify that the Conda package manager is installed correctly by checking its version 4:
   ```
   conda --version
   ```
3. Create a brand-new virtual environment named ml_environment with **Python 3.12** as the foundational interpreter 16:
   ```
   conda create --name ml_environment
   ```
   - *Note: When prompted Proceed ([y]/n)?, type y and press Enter.*
4. Activate your newly created environment to ensure package installations are confined to this workspace:
   ```
   conda activate ml_environment
   ```
5. Install the core Python data science and machine learning libraries:
   ```
   conda install numpy pandas scikit-learn matplotlib seaborn jupyter
   ```
   - **Pandas**: Used for tabular data manipulation, cleaning, and preprocessing.
   - **NumPy**: Powering high-performance numerical and multi-dimensional array operations.
   - **Scikit-learn**: The standard library containing built-in machine learning algorithms.
   - **Matplotlib & Seaborn**: Essential libraries for rich data visualization.

PART II: Jupyter Notebook Launch & Project Structuring (10 Minutes)

Jupyter Notebook is a web-based interactive workspace that allows data teams to write, execute, document, and visualize code in small, manageable blocks called **cells**.

Step 1: Launch Jupyter Notebook

1. Install Jupyter notebook
   ```
   conda install -c conda-forge notebook
   ```
2. In your active terminal (with ml_environment activated), run the launch command:
   ```
   jupyter notebook
   ```
3. The Jupyter interface will automatically open in your default internet web browser.

Step 2: Organize Your File Directory

To avoid writing long, absolute file paths when loading data, establish a clean folder structure:

1. In the Jupyter browser home tab, click **New** on the top-right and select **Folder**.
2. Check the box next to the new "Untitled Folder," click **Rename**, and change it to ML_For_Managers_Week1.

3. Navigate inside this folder. This is where you will save all Jupyter notebooks (.ipynb files) and associated CSV datasets.
4. Create your first live notebook: Click **New** on the top-right and select **Python 3 (ipykernel)**.
5. Click on the title "Untitled" at the top of the notebook page and rename it to Week1_Exploratory_Lab.

PART III: Strategic Dataset Profiling Matrix (15 Minutes)

Before writing code, managers must understand the shape of the data and align it with the correct learning paradigm. The following matrix profiles the standard industry benchmark datasets covered in this course.

Dataset	Dimensions & Shape	Key Features (Inputs)	Target Variable (Label)	Learning Paradigm & Task Category	Strategic Business Application
Iris Dataset	150 samples (50 per species)	Sepal length, sepal width, petal length, petal width	Species (Setosa, Virginica, Versicolor)	Supervised Learning: **Multi-Class Classification**	Introductory classification and algorithm validation.
Titanic Dataset	Variable Passenger Records	Passenger class, age, fare, sex, cabin, ticket number	Survival Status (0 = Deceased, 1 = Survived)	Supervised Learning: **Binary Classification**	Modeling churn, default, or yes/no customer behaviors.

Boston Housing Dataset	Suburb Tract Profiles	Rooms per dwelling, crime rate, tax rates, proximity to jobs	Median House Value (Continuous $)	Supervised Learning: **Continuous Regression**	Price prediction, asset valuation, and demand forecasting.
MNIST Dataset	70,000 grayscale images (28x28 pixels)	Grayscale intensity values for 784 individual pixels	Handwritten Digit Label (0 through 9)	Deep Learning / Supervised **Image Classification**	Optical character recognition, document scanning, and computer vision.
Adult Income Dataset	Census Records	Age, education level, occupation, relationship status	Income bracket (0 = ≤$50K, 1 = >$50K)	Supervised Learning: **Binary Classification**	Demographic profiling, target market selection, and risk scoring.
MovieLens Dataset	User-Movie Interactions	User IDs, Movie IDs, timestamps	User Ratings (Numerical scale)	Collaborative Filtering / **Recommendation System**	Personalization engines, cross-selling, and maximizing customer engagement.

PART IV: Hands-On Mock Python Session Script (10 Minutes)

Now, let us write your very first blocks of data exploration code in your Jupyter Notebook.

Exercise 1: Standard Imports and Environment Check

In your first cell, type the following code to import your active library toolchain and verify their installations. Click **Run** or press Shift + Enter to execute:

```
import pandas as pd
import numpy as np
import sklearn
import matplotlib.pyplot as plt

print("--- Data Toolchain Verification ---")
print(f"Pandas Version: {pd.__version__}")
print(f"NumPy Version: {np.__version__}")
print(f"Scikit-learn Version: {sklearn.__version__}")
print("Toolchain is loaded and ready for analysis!")
```

Exercise 2: Loading and Inspecting the Iris Dataset

The Iris dataset is pre-installed in Scikit-learn, making it ideal for continuous numerical regression practice. Write and execute the following code to load the data and inspect its characteristics:

```
# Import the loading utility from Scikit-learn
from sklearn.datasets import load_iris

# Load the raw dataset object
iris_data = load_iris()

# Convert the data into a structured Pandas DataFrame for
manipulation
df_iris = pd.DataFrame(iris_data.data,
columns=iris_data.feature_names)

# Display the first 5 records of the dataset
print("First 5 rows of the Iris Dataset:")
print(df_iris.head())
```

Exercise 3: Strategic Statistical Profiling

Once the data is structured, use statistical summary methods to check its distribution and identify potential data quality issues:

```
# Display a high-level summary of the dataset's variables, data
types, and non-null counts
print("\n--- Dataset Info Summary ---")
```

```
df_iris.info()

# Calculate descriptive statistics (mean, standard deviation, min,
max, quartiles)
print("\n--- Descriptive Statistics Summary ---")
print(df_iris.describe())
```

PART V: Critical-Thinking Managerial Review Questions (5 Minutes)

Review these diagnostic questions to bridge your technical environment setup with strategic decision-making 43:

1. **The "Garbage In, Garbage Out" (GIGO) Principle:**
    - *Question*: During Exercise 3, you executed df_boston.info(). If you observe that several variables have non-null counts significantly lower than the total number of rows, what does this tell you about the dataset, and what action must your data team take before training a model 42, 44, 45?
    - *Strategic Answer*: Mismatched non-null counts indicate **missing values** 42, 44. If ignored, these gaps can distort predictions or crash training runs 42, 44. The data team must perform preprocessing steps, such as removing rows with missing values or performing statistical imputation (replacing missing cells with the mean, median, or mode) 42, 44.
2. **The Compliance & Interpretability Tradeoff:**
    - *Question*: You are managing a project to predict credit risk. A data scientist suggests training a highly complex "black-box" model on the Adult Income dataset 35, 46. Why might you reject this in favor of a simpler, supervised algorithm like Logistic Regression or a single Decision Tree 31, 46, 47?
    - *Strategic Answer*: In strictly regulated industries like finance and lending, transparency is a compliance mandate 46. If a model rejects a customer's loan application, you must be able to justify why 48. Highly interpretable models (like Logistic Regression) let you audit decision boundaries and weights easily 46, 48. This reduces legal risk compared to accurate but unexplainable "black-box" systems 46, 48.