Module 2 Lab Exercise: Tools Used in Machine Learning

Learning Objectives

By the end of this lab, you will be able to:

- Set up and navigate Jupyter Notebook, Google Colab, and VS Code environments
- Install and import essential Python libraries for machine learning
- Create and format professional documentation using Markdown
- Initialize a GitHub repository for your ML projects
- Understand the basic workflow of data science tools

Prerequisites

- Basic understanding of what machine learning is (Module 1)
- Access to internet for downloading tools and datasets
- A Google account (for Colab) or local Python installation

Part 1: Environment Setup and Tool Overview

What are the main tools we'll use in this course?

Jupyter Notebook/Google Colab: Interactive computing environments where you can write code, see results immediately, and document your work with text and visualizations.

Python Libraries: Pre-written code packages that make machine learning tasks easier:

- **Pandas**: For working with data (like Excel, but more powerful)
- **NumPy**: For mathematical operations on arrays of numbers
- **Matplotlib**: For creating charts and graphs
- **Scikit-learn**: The main library for machine learning algorithms

GitHub: A platform to store, share, and collaborate on code projects

VS Code: A powerful text editor for writing and debugging code

Let's start by setting up our environment!

Environment Setup Instructions

Option 1: Google Colab (Recommended for Beginners)

1. Go to colab.research.google.com
2. Sign in with your Google account
3. Click "New Notebook"
4. You're ready to go! Libraries are pre-installed.

Option 2: Local Jupyter Notebook

1. Install Python from python.org
2. Open terminal/command prompt
3. Run `pip install jupyter pandas numpy matplotlib scikit-learn`
4. Run `jupyter notebook`
5. Create a new notebook

Option 3: VS Code

1. Download VS Code from code.visualstudio.com
2. Install Python extension
3. Install Jupyter extension
4. Create a new .ipynb file

For this lab, we recommend starting with Google Colab as it requires no installation.

```
In [28]: pip install pandas numpy matplotlib scikit-learn
Requirement already satisfied: pandas in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (2.3.2)
Requirement already satisfied: numpy in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (2.3.2)
Requirement already satisfied: matplotlib in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (3.10.6)
Requirement already satisfied: scikit-learn in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (1.7.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /Users/kendrickkaraka/Library/Python/3.11/lib/python/site-packages (from pandas
v) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from pan
das) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from pa
ndas) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from
matplotlib) (1.3.3)
Requirement already satisfied: cycler>=0.10 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from mat
plotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (fro
m matplotlib) (4.60.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (fro
m matplotlib) (1.4.9)
Requirement already satisfied: packaging>=20.0 in /Users/kendrickkaraka/Library/Python/3.11/lib/python/site-packages (from matplotlib) (2
5.0)
Requirement already satisfied: pillow>=8 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from matplo
tlib) (11.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (fro
m matplotlib) (3.2.3)
Requirement already satisfied: scipy>=1.8.0 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from sci
kit-learn) (1.16.1)
Requirement already satisfied: joblib>=1.2.0 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from s
cikit-learn) (1.5.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages
from scikit-learn) (3.6.0)
Requirement already satisfied: six>=1.5 in /Users/kendrickkaraka/Library/Python/3.11/lib/python/site-packages (from python-dateutil>=2.
8.2->pandas) (1.17.0)

[notice] A new release of pip is available: 24.0 -> 25.2
[notice] To update, run: pip3 install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

```
In [23]: # Install required libraries (uncomment if needed)
         # !pip install pandas numpy matplotlib scikit-learn

         # Import libraries with standard aliases
         import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         from sklearn import datasets
         import warnings
         warnings.filterwarnings('ignore')  # Hide warning messages for cleaner output

         print("✅ All libraries imported successfully!")
         print("Pandas version: {pd.__version__}")
         print("NumPy version: {np.__version__}")
✅ All libraries imported successfully!
Pandas version: {pd.__version__}
NumPy version: {np.__version__}
```

Part 2: Loading and Exploring Your First Dataset

We'll use the famous Iris dataset - a classic dataset for beginners. It contains measurements of Iris flowers from three different species.

```
In [24]: # Load a classic dataset (Iris flowers - a classic beginner dataset)
         from sklearn.datasets import load_iris

         # Load the data
         iris = load_iris()
         print("Dataset loaded successfully!")
         print("Dataset shape: {iris.data.shape}")
         print("Features: {iris.feature_names}")
         print("Target classes: {iris.target_names}")
Dataset loaded successfully!
Dataset shape: (150, 4)
Features: ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
Target classes: ['setosa' 'versicolor' 'virginica']
```

```
In [25]: # Convert to pandas DataFrame for easier handling
         df = pd.DataFrame(iris.data, columns=iris.feature_names)
         df['species'] = iris.target_names[iris.target]

         # Display first few rows
         print("First 5 rows of our dataset:")
         print(df.head())

         print("\nDataset info:")
         print(df.info())

First 5 rows of our dataset:
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2

  species
0  setosa
1  setosa
2  setosa
3  setosa
4  setosa

Dataset info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   sepal length (cm)  150 non-null    float64
 1   sepal width (cm)   150 non-null    float64
 2   petal length (cm)  150 non-null    float64
 3   petal width (cm)   150 non-null    float64
 4   species            150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
None
```

Part 3: Creating Your First Visualization

Data visualization is crucial in machine learning. Let's create a simple plot to understand our data.

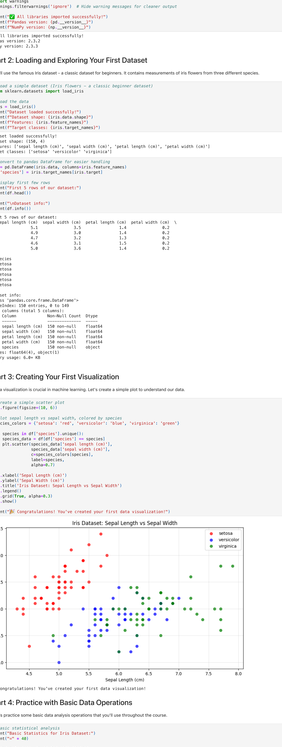
```
In [26]: # Create a simple scatter plot
         plt.figure(figsize=(10, 6))

         # Plot sepal length vs sepal width, colored by species
         species_colors = {'setosa': 'red', 'versicolor': 'blue', 'virginica': 'green'}

         for species in df['species'].unique():
             species_data = df[df['species'] == species]
             plt.scatter(species_data['sepal length (cm)'],
                         species_data['sepal width (cm)'],
                         c=species_colors[species],
                         label=species,
                         alpha=0.7)

         plt.xlabel('Sepal Length (cm)')
         plt.ylabel('Sepal Width (cm)')
         plt.title('Iris Dataset: Sepal Length vs Sepal Width')
         plt.legend()
         plt.grid(True, alpha=0.3)
         plt.show()

         print("🎉 Congratulations! You've created your first data visualization!")
```



🎉 Congratulations! You've created your first data visualization!

Part 4: Practice with Basic Data Operations

Let's practice some basic data analysis operations that you'll use throughout the course.

```
In [29]: # Basic statistical analysis
         print("Basic Statistics for Iris Dataset:")
         print("="* 40)

         # Calculate mean values for each species
         species_means = df.groupby('species').mean()
         print("\nMean values by species:")
         print(species_means)

         # Count samples per species
         species_counts = df['species'].value_counts()
         print("\nSamples per species:")
         print(species_counts)
Basic Statistics for Iris Dataset:
========================================

Mean values by species:
            sepal length (cm)  sepal width (cm)  petal length (cm)  \
species
setosa                  5.006             3.428              1.462
versicolor              5.936             2.770              4.260
virginica               6.588             2.974              5.552

            petal width (cm)
species
setosa             0.246
versicolor         1.326
virginica          2.026

Samples per species:
species
setosa        50
versicolor    50
virginica     50
Name: count, dtype: int64
```

Part 5: GitHub and Documentation Best Practices

Why GitHub for Machine Learning?

- **Version Control**: Track changes to your code and data
- **Collaboration**: Work with others on projects
- **Portfolio**: Showcase your work to potential employers
- **Backup**: Never lose your work

Basic GitHub Workflow:

1. **Create Repository**: A folder for your project
2. **Clone/Download**: Get the project on your computer
3. **Add Files**: Put your notebooks and data
4. **Commit**: Save a snapshot of your changes
5. **Push**: Upload changes to GitHub

For This Course:

- Create a repository named "ITAI-1371-ML-Labs"
- Upload each lab notebook as you complete it
- Include a README.md file describing your projects

Action Item: After this lab, create your GitHub account and repository.

Assessment: Tool Familiarity Check

Complete the following tasks to demonstrate your understanding of the tools:

```
In [30]: # Task 1: Create a simple calculation using NumPy
         # Calculate the mean and standard deviation of sepal length

         sepal_lengths = df['sepal length (cm)']

         # Your code here:
         mean_sepal = np.mean(sepal_lengths)
         std_sepal_length = np.std(sepal_lengths)

         print("Mean sepal length: {mean_sepal_length:.2f} cm")
         print("Standard deviation: {std_sepal_length:.2f} cm")

         # Verification (don't modify)
         assert isinstance(mean_sepal_length, (float, np.floating)), "Mean should be a number"
         assert isinstance(std_sepal_length, (float, np.floating)), "Std should be a number"
         print("✅ Task 1 completed successfully!")
Mean sepal length: 5.84 cm
Standard deviation: 0.83 cm
✅ Task 1 completed successfully!
```

```
In [31]: # Task 2: Create a simple bar chart showing species counts
         species_counts = df['species'].value_counts()

         plt.figure(figsize=(8, 5))
         species_counts.plot(kind='bar', species_counts.values, color=['red', 'blue', 'green'])
         plt.title('Number of Samples per Species')
         plt.xlabel('Species')
         plt.ylabel('Count')
         plt.show()

         print("Species distribution: {dict(species_counts)}")
         print("✅ Task 2 completed successfully!")
```

Species distribution: {'setosa': np.int64(50), 'versicolor': np.int64(50), 'virginica': np.int64(50)}
✅ Task 2 completed successfully!

Your Analysis and Reflection

Instructions: Complete the analysis below by editing this markdown cell.

My Observations About the Iris Dataset

Dataset Overview:

- Number of samples: [150]
- Number of features: [4]
- Number of classes: [3]

Key Findings from the Visualization:

1. [The scatter plot likely shows that the iris-setosa species forms a separate, distinct cluster with a smaller sepal length and larger sepal width compared to the other two species]
2. [The iris-versicolor and iris-virginica have more overlap in their sepal dimensions, making them harder to distinguish from each other based on sepal length and width alone]
3. [The plot may show a mild negative correlation between sepal length and sepal width across the dataset, but the relationship is heavily influenced by the species]

Questions for Further Investigation:

- [Would using sepal length and width instead of sepal measurements create a visualization where all three species are more clearly separated?]
- [Which machine learning model (like K-Nearest Neighbors) would be most effective at automatically classifying the species based on these four measurements, and how accurate would it be?]

Reflection: Using pandas made it easy to structure the data and perform quick analysis, like calculating group averages. Matplotlib then allowed me to visually confirm these numerical patterns, revealing clear relationships and clusters that would be difficult to see in a table of numbers.

Note: This is practice for documenting your machine learning projects professionally.

Lab Summary and Next Steps

What You've Accomplished:

✅ Set up your machine learning development environment
✅ Imported and used essential Python libraries
✅ Loaded and explored your first dataset
✅ Created your first data visualization
✅ Practiced professional documentation with Markdown
✅ Learned about GitHub for project management

Preparation for Module 3:

In the next lab, you'll:

- Learn about different types of machine learning
- Build your first simple classifier
- Understand the complete ML workflow
- Work with more complex datasets

Action Items:

1. **Create your GitHub account** and repository
2. **Upload this completed notebook** to your repository
3. **Experiment** with different visualizations using the Iris dataset
4. **Practice** Markdown formatting in a new notebook

Resources for Continued Learning:

- Pandas Documentation
- Matplotlib Gallery
- GitHub Guides
- Jupyter Notebook Tips

Great job completing Module 2! You're now equipped with the essential tools for machine learning. 🎉