

# 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!

```
# 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(f"Pandas version: {pd.__version__}")
print(f"NumPy version: {np.__version__}")
```

```
✅ All libraries imported successfully!
Pandas version: 2.2.2
NumPy version: 2.0.2
```

## ✓ 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.

```
# Load a simple dataset (Iris flowers - a classic beginner dataset)
from sklearn.datasets import load_iris

# Load the data
iris = load_iris()
print("Dataset loaded successfully!")
print(f"Dataset shape: {iris.data.shape}")
print(f"Features: {iris.feature_names}")
print(f"Target classes: {iris.target_names}")
```

```
Dataset loaded successfully!
Dataset shape: (150, 4)
Features: ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal
Target classes: ['setosa' 'versicolor' 'virginica']
```

```
# 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.

```

# Create a simple scatter plot
plt.figure(figsize=(10, 6))

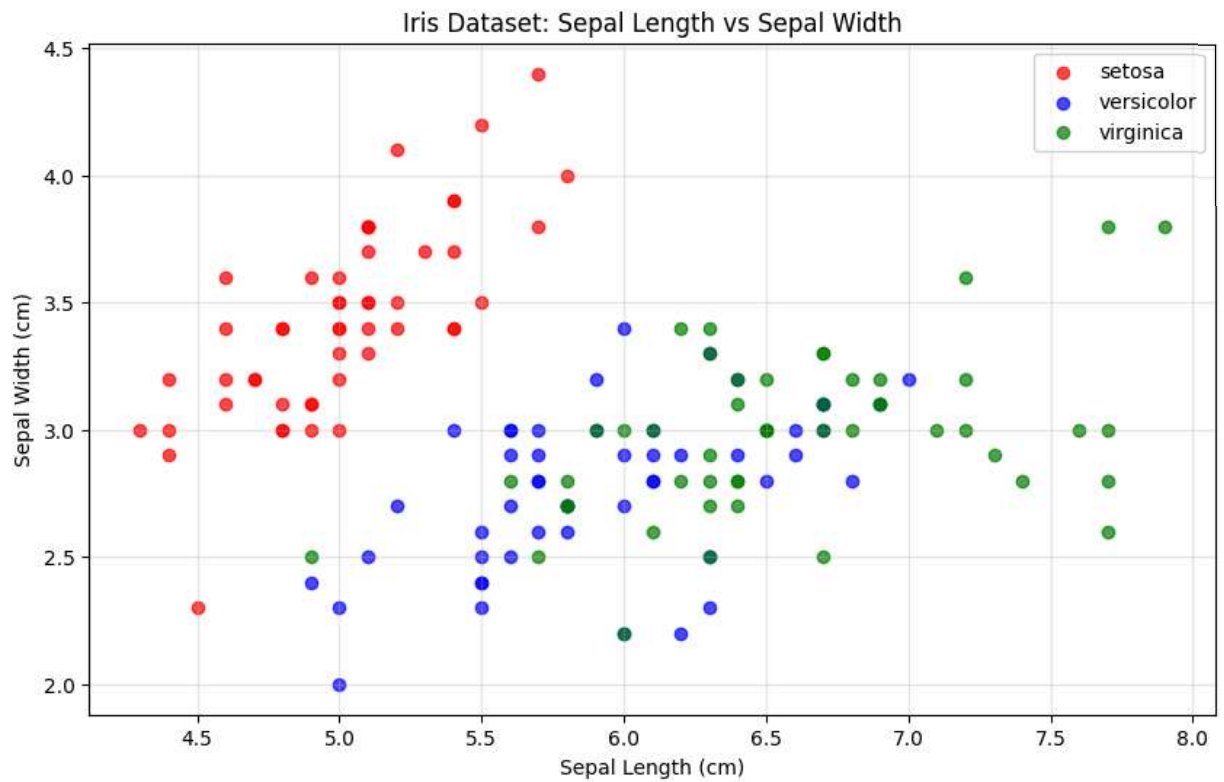
# Plot sepal length vs sepal width, colored by species
species_colors = {'setosa': 'red', 'versicolor': 'blue', 'virginica': 'green'}

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.

```
# 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:

```

# 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_length = np.mean(sepal_lengths)
std_sepal_length = np.std(sepal_lengths)

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

# Verification (don't modify)
assert isinstance(mean_sepal_length, (float, np.floating)), "Mean should be a float"
assert isinstance(std_sepal_length, (float, np.floating)), "Std should be a float"
print("✅ Task 1 completed successfully!")

```

```

Mean sepal length: 5.84 cm
Standard deviation: 0.83 cm
✅ Task 1 completed successfully!

```

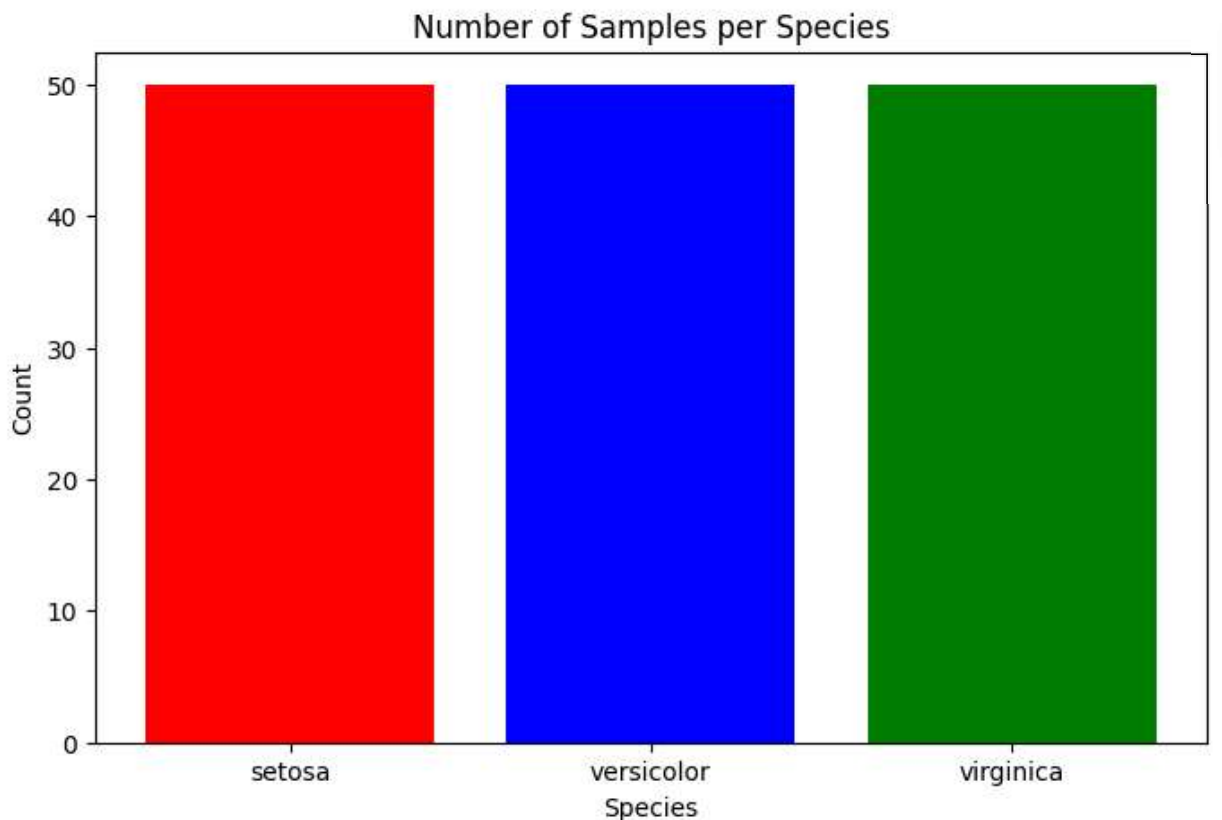
```

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

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

print(f"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 Samples.
- Number of features: 4 Features.
- Number of classes: 3 target classes.

#### Key Findings from the Visualization:

1. The Setosa species appears to be well separated from the other two species.
2. There is some overlap between the blue and green species.
3. Setosa has smaller sepal lengths and larger sepal widths compared to versicolor and virginica.

#### Questions for Further Investigation:

- Is there a relationship between sepal length and petal length.
- What species has the largest average petal.

## Reflection:

I have learned that the dataset contains 150 samples with 4 features each, belonging to 3 different species setosa, versicolor, and virginica. The number of samples per species are equal which is 50 counts. The scatter plot of sepal length vs sepal width shows that the setosa species is distinct, while there is some overlap between versicolor and virginica.

The visual analysis of the Iris dataset highlights that petal measurement, especially length, and width offer the strongest separation among the three species. Setosa forms a tight cluster with the smallest petal dimensions, while Virginica generally shows the largest values and Versicolor falls in between. Sepal width varies more widely within each species, making it a weaker discriminator compared with petal features. A noticeable positive correlation appears between petal length and petal width across all samples, reinforcing the idea that these two measurements together provide the most reliable basis for distinguishing the flowers.

---

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

◆ Gemini

## ✓ 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**