Iris Flower Classification Using Machine Learning

Name: Tanya Yadav
Roll No: 202401100400197
Course: Introduction to AI
Date: 10-03-2025

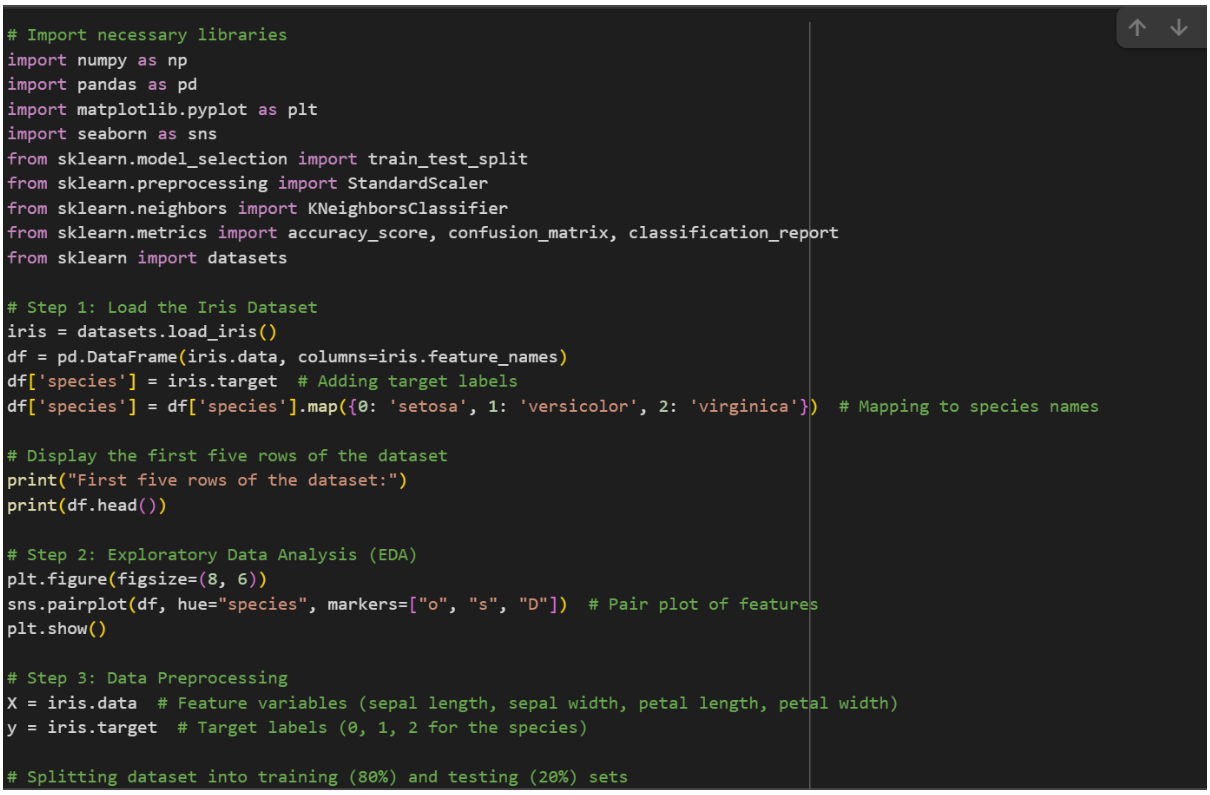
Introduction

This project is about classifying Iris flowers into three types: *Setosa, Versicolor, and Virginica* based on their petal and sepal sizes. We use a machine learning technique called **K-Nearest Neighbors (KNN)** to train a model that can identify the flower species correctly. The dataset used is the popular **Iris dataset**, which is available in the Scikit-Learn library.

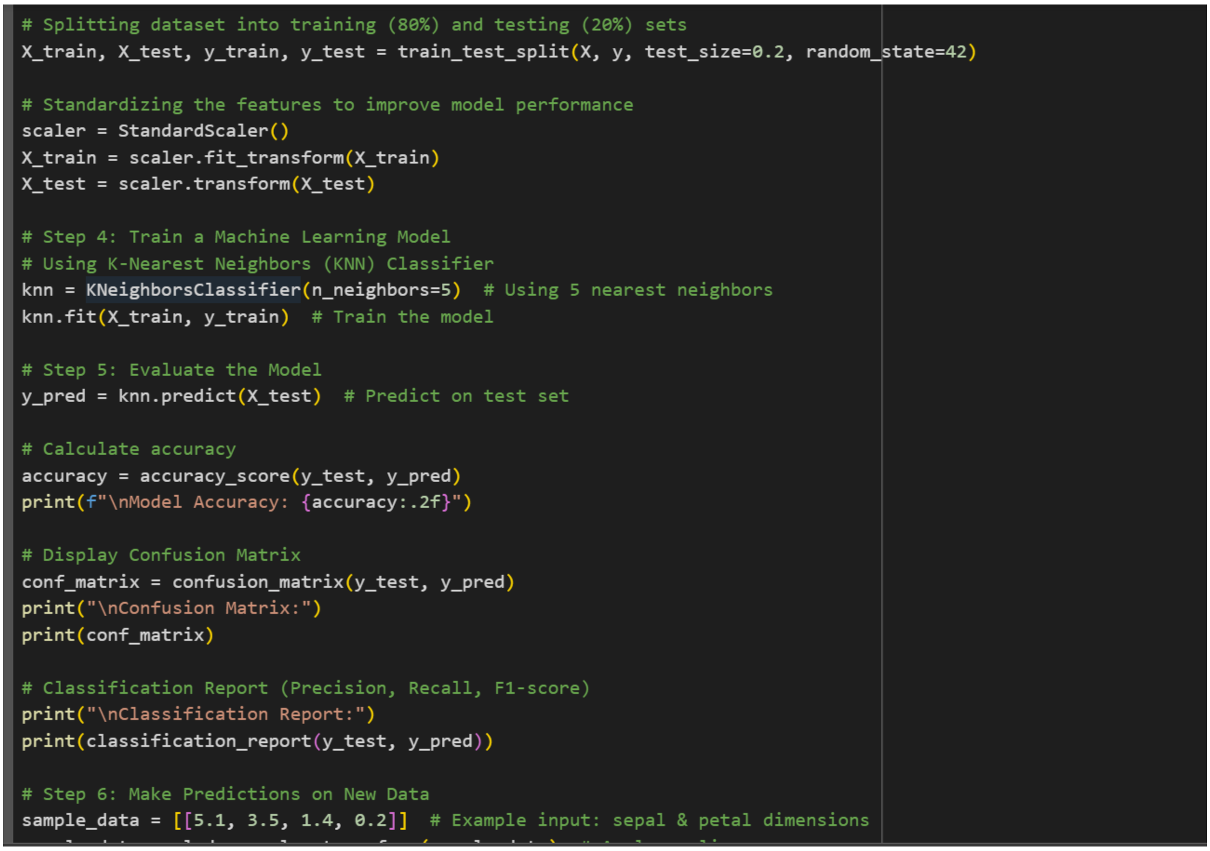
Methodology

1. **Loading the Dataset**: The Iris dataset is imported using Scikit-Learn.
2. **Data Preparation**: The data is structured into a table, and the species labels are converted to names.
3. **Visualization**: Graphs are created to understand how the features relate to each other.
4. **Splitting the Data**: The dataset is divided into **80% training data** and **20% testing data**.
5. **Standardization**: The feature values are scaled to ensure fair comparisons.
6. **Training the Model**: A KNN classifier is trained using five nearest neighbors.
7. **Model Testing**: The trained model is tested on the test data.
8. **Making Predictions**: The model predicts flower species for given inputs.

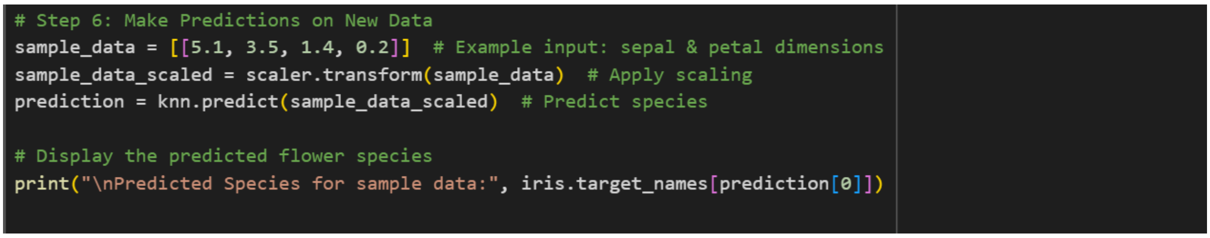
CODE



```python
# Import necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn import datasets

# Step 1: Load the Iris Dataset
iris = datasets.load_iris()
df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['species'] = iris.target  # Adding target labels
df['species'] = df['species'].map({0: 'setosa', 1: 'versicolor', 2: 'virginica'})  # Mapping to species names

# Display the first five rows of the dataset
print("First five rows of the dataset:")
print(df.head())

# Step 2: Exploratory Data Analysis (EDA)
plt.figure(figsize=(8, 6))
sns.pairplot(df, hue="species", markers=["o", "s", "D"])  # Pair plot of features
plt.show()

# Step 3: Data Preprocessing
X = iris.data  # Feature variables (sepal length, sepal width, petal length, petal width)
y = iris.target  # Target labels (0, 1, 2 for the species)

# Splitting dataset into training (80%) and testing (20%) sets
```

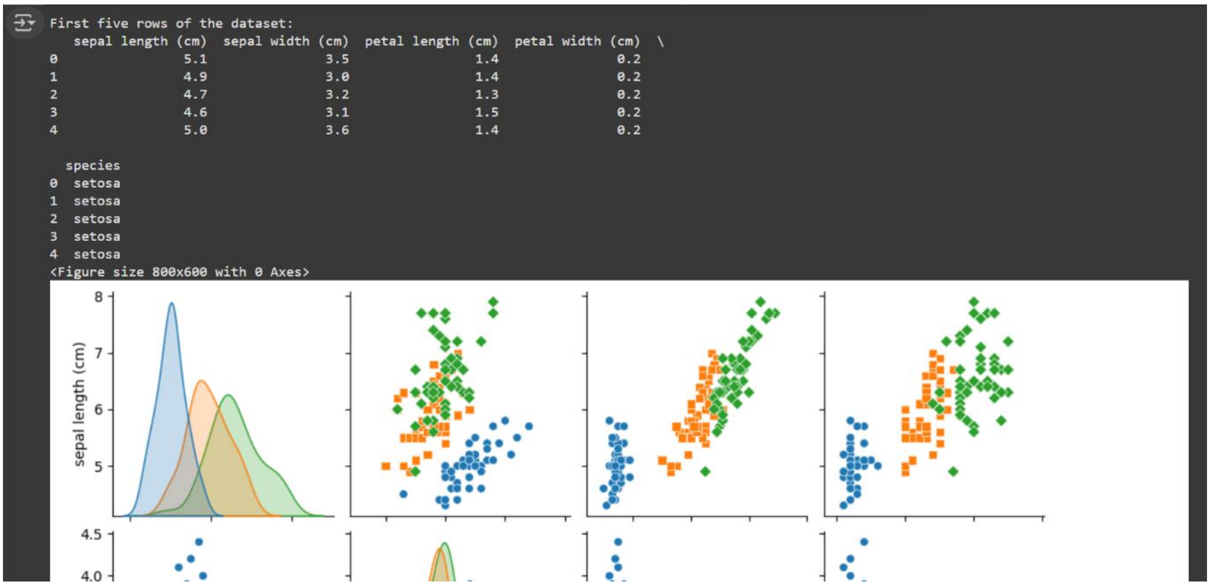


```python
# Splitting dataset into training (80%) and testing (20%) sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardizing the features to improve model performance
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Step 4: Train a Machine Learning Model
# Using K-Nearest Neighbors (KNN) Classifier
knn = KNeighborsClassifier(n_neighbors=5)  # Using 5 nearest neighbors
knn.fit(X_train, y_train)  # Train the model

# Step 5: Evaluate the Model
y_pred = knn.predict(X_test)  # Predict on test set

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"\nModel Accuracy: {accuracy:.2f}")

# Display Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)
print("\nConfusion Matrix:")
print(conf_matrix)

# Classification Report (Precision, Recall, F1-score)
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# Step 6: Make Predictions on New Data
sample_data = [[5.1, 3.5, 1.4, 0.2]]  # Example input: sepal & petal dimensions
```

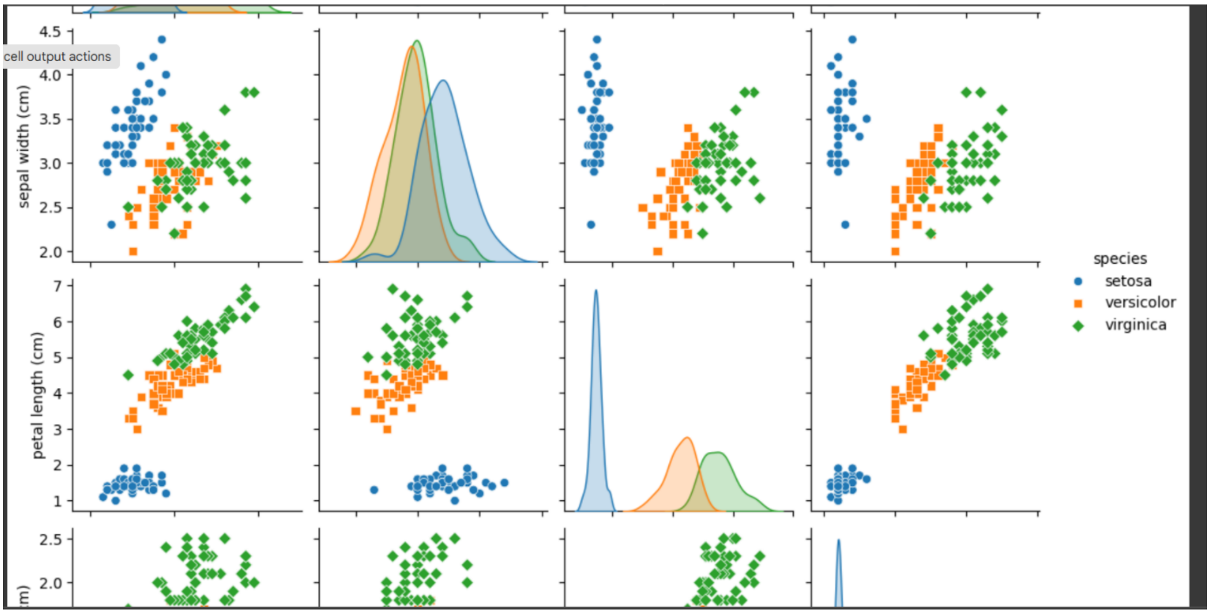


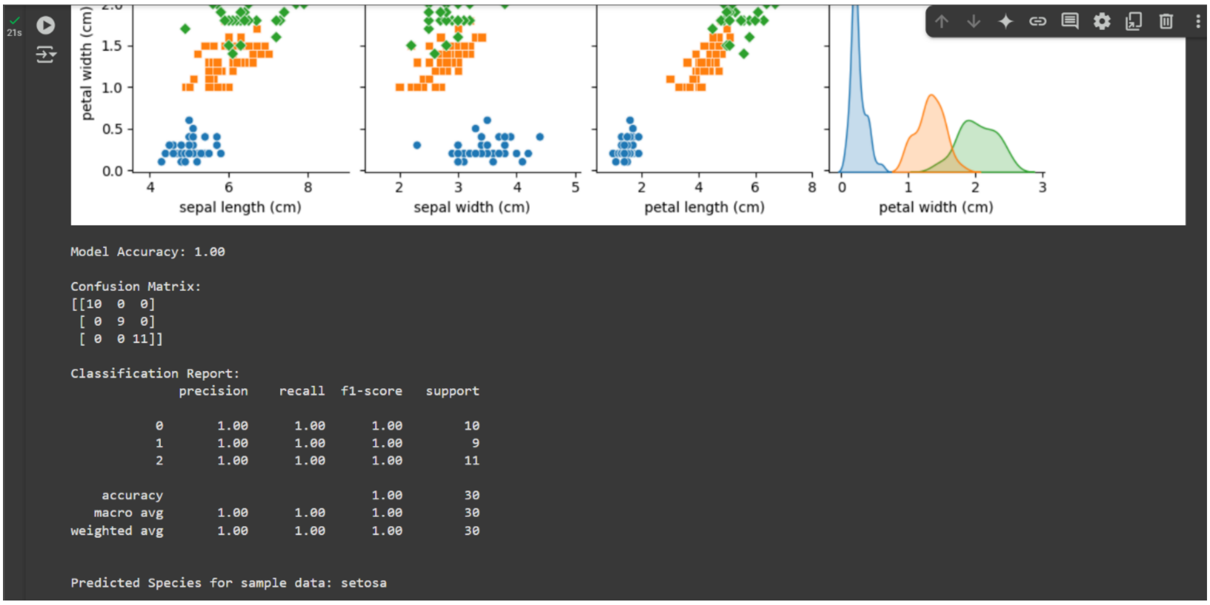
```
# Step 6: Make Predictions on New Data
sample_data = [[5.1, 3.5, 1.4, 0.2]]  # Example input: sepal & petal dimensions
sample_data_scaled = scaler.transform(sample_data)  # Apply scaling
prediction = knn.predict(sample_data_scaled)  # Predict species

# Display the predicted flower species
print("\nPredicted Species for sample data:", iris.target_names[prediction[0]])
```

OUTPUT



```
First five rows of the 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
<Figure size 800x600 with 0 Axes>
```





```
Model Accuracy: 1.00

Confusion Matrix:
[[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       1.00      1.00      1.00         9
           2       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30


Predicted Species for sample data: setosa
```

References/Credits

- **Dataset Source**: Scikit-Learn (UCI Machine Learning Repository)
- **Libraries Used**: Pandas, NumPy, Matplotlib, Seaborn, Scikit-Learn
-