**AI MSE 1ST**

**2024-25**

**2ND Semester**

Name : Nidhi

Class: CSE AI ML

Section : B

Class Roll No. -51

University Roll No.- 202401100400124

PROBLEM STATEMENT:

Iris Flower Classification-Classify flower species based on petal and sepal dimensions.

The PROBLEM EXPLAINATION

The problem at hand is a **classification problem**, where the goal is to classify different species of flowers based on their physical characteristics. In this case, we are using the **Iris dataset**, which contains measurements of various attributes of the flowers, and we want to predict the species of a flower based on those attributes.

**Problem Details:**

The **Iris dataset** is one of the most famous datasets in machine learning and is often used for demonstrating classification algorithms. It consists of three species of Iris flowers: **Setosa**, **Versicolor**, and **Virginica**. The dataset includes four features for each flower:

1. **Sepal Length (cm)**
2. **Sepal Width (cm)**
3. **Petal Length (cm)**
4. **Petal Width (cm)**

For each flower, these four features are measured, and the task is to predict the species of the flower based on these measurements.

**Objective:**

Given the measurements of the **sepal length**, **sepal width**, **petal length**, and **petal width**, the objective is to correctly classify the species of the Iris flower into one of the three categories: **Setosa**, **Versicolor**, or **Virginica**.

**Machine Learning Approach:**

The solution uses a machine learning model to perform the classification. The steps involved in solving this problem include:

1. **Loading the Dataset**: The first step is to load the dataset, typically in CSV format, using a library like **Pandas**. In this case, the data consists of flower measurements and their corresponding species.
2. **Data Preprocessing**:
   * **Checking for Missing Values**: Before applying any machine learning model, it's crucial to check for missing data. In this case, the dataset is assumed to have no missing values.
   * **Feature Selection**: The dataset contains several columns. We use the columns corresponding to the **sepal length**, **sepal width**, **petal length**, and **petal width** as **features** (input variables), and the **species** column as the **target** (output variable).
3. **Splitting the Dataset**: The data is split into two sets:
   * **Training Set**: This is the portion of the data used to train the machine learning model.
   * **Testing Set**: This is the portion of the data used to test how well the model has learned to classify the species.
4. **Model Selection**: We use a **Random Forest Classifier** in this example. This model is an ensemble method that builds multiple decision trees and combines their predictions to improve accuracy and reduce overfitting. Random Forest is a very effective algorithm for classification tasks like this one.
5. **Model Training**: The Random Forest model is trained using the training set. During training, the algorithm tries to find patterns in the data that correlate flower measurements (features) with flower species (target).
6. **Model Evaluation**: After training, the model is tested on the test set to see how accurately it can classify flowers into the correct species. The performance is evaluated using the **accuracy score**, which is the percentage of correct predictions made by the model.
7. **Feature Importance**: The Random Forest model also provides a way to understand the importance of each feature (e.g., sepal length, petal width) in making predictions. We visualize this using a bar plot to identify which features are most influential in determining the species.
8. **Making Predictions**: Finally, the trained model can predict the species of a new, unseen flower sample based on its features (e.g., 5.1, 3.5, 1.4, 0.2). This step allows the model to generalize and classify new data.

EXPLAINATION OF THE CODE

1. **Import Libraries**: The necessary libraries (Pandas, Scikit-learn, Matplotlib, and Seaborn) are imported to handle data, build the model, and visualize results.
2. **Load Dataset**: The Iris dataset is loaded from a CSV file. It contains the flower measurements and species labels.
3. **Prepare Data**: Features (flower measurements like sepal and petal dimensions) are separated from the target (species labels).
4. **Split Data**: The dataset is split into training and test sets, typically 80% for training and 20% for testing.
5. **Train Model**: A **Random Forest Classifier** is trained using the training data to learn the relationship between flower attributes and their species.
6. **Make Predictions**: The trained model predicts species for the test data based on the features.
7. **Evaluate Accuracy**: The model’s accuracy is calculated by comparing the predicted species with the actual species in the test set.
8. **Feature Importance**: The importance of each feature (like petal width, sepal length) is visualized to understand which features impact the prediction the most.
9. **Prediction for New Data**: The model is used to predict the species for a new flower sample based on its measurements.

This approach uses **Random Forest** to classify flowers and evaluate its performance using accuracy

TYPED CODE

# Import necessary libraries

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score

import matplotlib.pyplot as plt

import seaborn as sns

# Step 1: Load the dataset (assuming the file is named 'data.csv')

df = pd.read\_csv('data.csv')

# Step 2: Display the first few rows of the dataset

print("\nFirst few rows of the dataset:")

print(df)

# Step 4: Prepare the features (X) and target (y)

X = df.drop('species', axis=1)  # Features (sepal length, sepal width, etc.)

y = df['species']  # Target (species)

# Step 5: Split the data into training and testing sets (80% training, 20% testing)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Step 6: Train a Random Forest Classifier

clf = RandomForestClassifier(random\_state=42)

clf.fit(X\_train, y\_train)

# Step 7: Make predictions on the test set

y\_pred = clf.predict(X\_test)

# Step 8: Evaluate the model's accuracy

accuracy = accuracy\_score(y\_test, y\_pred)

print(f"\nAccuracy of the Random Forest Classifier: {accuracy \* 100:.2f}%")

# Step 9: Feature importance (optional)

# Visualizing feature importance to understand which features are important in the model

feature\_importances = clf.feature\_importances\_

features = X.columns

plt.figure(figsize=(10, 6))

sns.barplot(x=feature\_importances, y=features)

plt.title('Feature Importance')

plt.show()

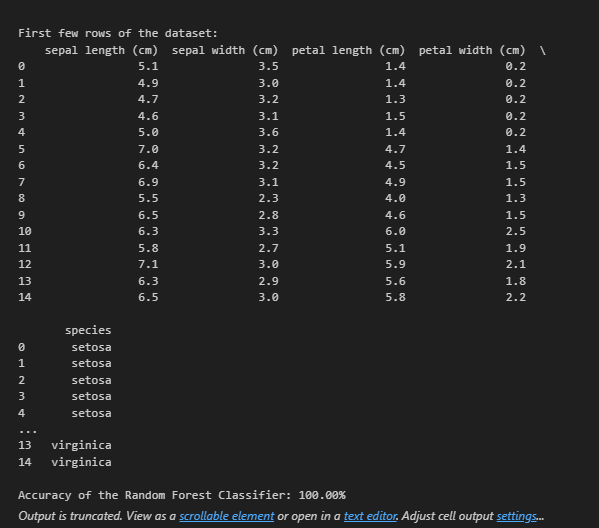
# Step 10: Predict on a new sample (example data point)

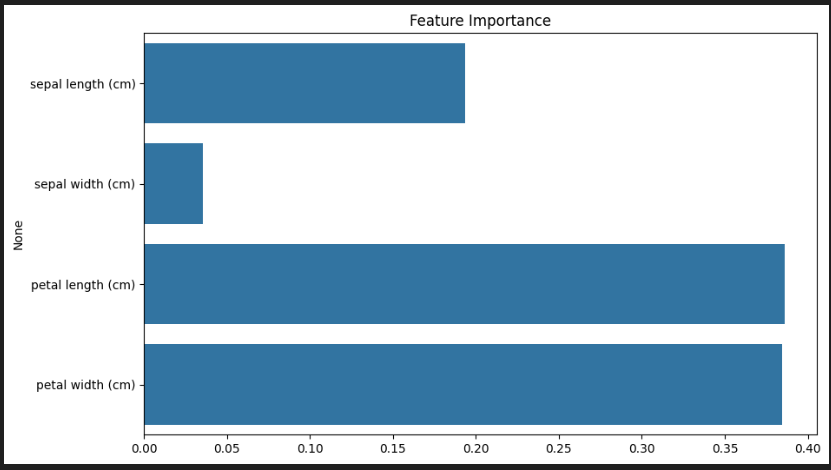
sample = [[5.1, 3.5, 1.4, 0.2]]  # Example data point (adjust based on your features)

sample\_prediction = clf.predict(sample)

print(f"\nPredicted species for the new sample: {sample\_prediction[0]}")

OUTPUT





REFERENCE/CREDITS:

1. USED panda
2. Used sklearn.model
3. Used sklearn.ensemble
4. Used sklearn.metrics
5. Used matplotlib………..libraries