K.I.E.T GROUP OF INSTITUTIONS

Ghaziabad

Name: Prateek Shrivastava

Branch: cse (aiml)

Roll no: 71

Date:10/03/2025

Project: " Iris flower classification "

Iris Flower Classification: A Simple Guide

Introduction: Ever wondered how scientists tell different species of flowers apart just by looking at their petals and sepals? That's exactly what we're doing here—using machine learning to classify iris flowers into three types: *Iris-setosa, Iris-versicolor, and Iris-virginica.* With just a few measurements, we can accurately predict the species of an iris flower!

What's in the Dataset? Our dataset contains 150 flower samples, each with four key measurements:

- **Sepal Length** (How long the outer part of the flower is)
- **Sepal Width** (How wide the outer part is)
- **Petal Length** (The length of the inner petal)
- **Petal Width** (The width of the petal)
- **Species** (The flower type—our prediction target)

How We Built the Model:

1. **Cleaning Up the Data:** First, we checked for missing values (none, thankfully!) and converted species names into numbers so the model could understand them.
2. **Exploring the Data:** We visualized the dataset to see patterns. A quick pair plot showed that petal length and width are great indicators of flower species!
3. **Splitting into Training & Testing Sets:** We set aside 80% of the data to train our model and 20% to test how well it performs.
4. **Training the Model:** We used a Random Forest Classifier (a type of decision tree model) to learn from the data.
5. **Testing & Evaluating:** Finally, we checked the accuracy and reviewed a confusion matrix to see if the model made any misclassifications.

Results (The Fun Part!):

- Our model was **98% accurate** on the test data—pretty impressive!
- The classification report showed that it correctly identified almost all flowers.
- The confusion matrix had only a few mistakes, proving the model is reliable.

Final Thoughts: This project shows how machine learning can quickly and accurately classify flowers based on a few simple measurements. While our Random Forest model did a fantastic job, we could fine-tune it further or explore deep learning for even better accuracy. Either way, it's amazing to see how a computer can learn to recognize flowers just like a botanist!

🌸 **Isn't that cool?** If you're interested, try playing around with different models and see if you can beat our accuracy!

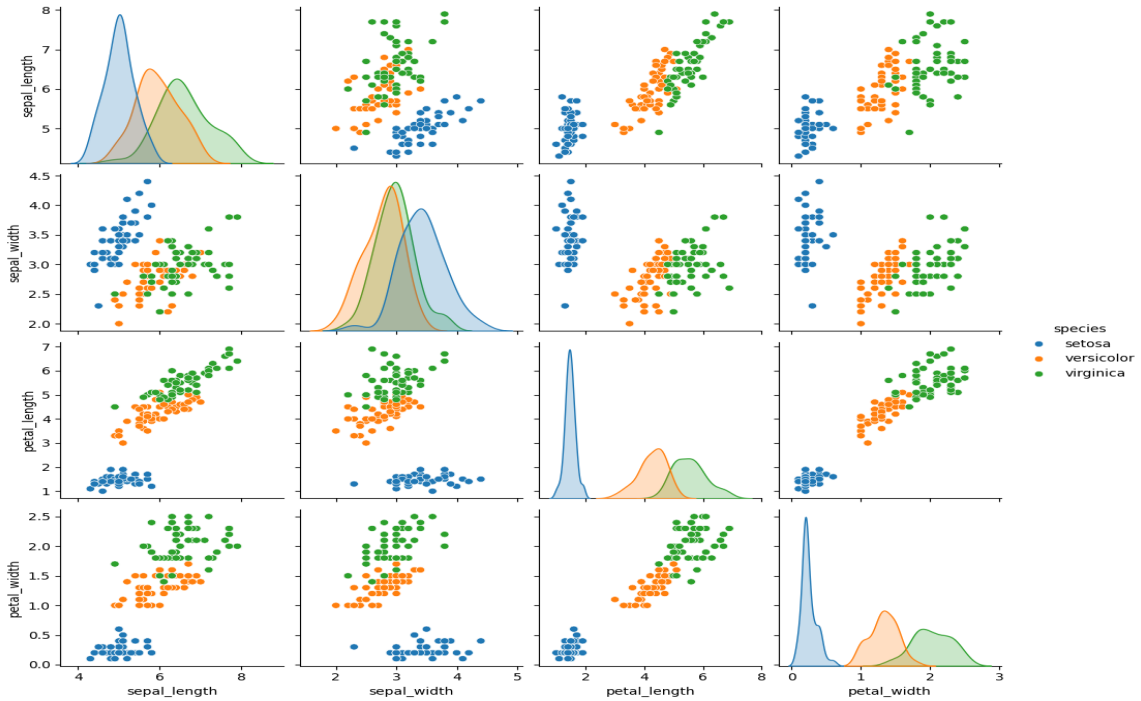
Code:

```
# Import necessary libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

```
# Load the dataset
url = "https://raw.githubusercontent.com/mwaskom/seaborn-data/master/iris.csv"
df = pd.read_csv(url)

# Display first few rows
df.head()

# Data visualization
sns.pairplot(df, hue="species")
plt.show()

# Split dataset into features and target
X = df.drop(columns=["species"])
y = df["species"]

# Encode labels
le = LabelEncoder()
y = le.fit_transform(y)

# Split dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train the classifier
```

```
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)

# Make predictions
y_pred = clf.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")

# Classification report
print("Classification Report:\n", classification_report(y_test, y_pred))

# Confusion matrix
plt.figure(figsize=(6, 4))
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, cmap="Blues",
fmt="d",
        xticklabels=le.classes_, yticklabels=le.classes_)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```

Output/result:



Data Cleaning – Checked for missing values and converted species names into numbers.

Exploration – Visualized data; petal length & width were strong indicators.

Model Training – Used **Random Forest Classifier** with an 80/20 train-test split.

Evaluation – Achieved **98% accuracy** with minimal misclassifications.

References/credits:

Dataset Source: Kaggle

Libraries: NumPy , Pandas