```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report
import seaborn as sns
import matplotlib.pyplot as plt
```

```
from google.colab import files
uploaded = files.upload()
import pandas as pd

# Replace 'IRIS.csv' with the exact filename you uploaded
df = pd.read_csv('IRIS.csv')
df.head()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving IRIS.csv to IRIS (1).csv
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
# Separate features and target
X = df.iloc[:, :-1]
y = df.iloc[:, -1]

# Encode species names into numeric labels
label_encoder = LabelEncoder()
y_encoded = label_encoder.fit_transform(y)

# Split the data into training and test sets (70% train, 30% test)
X_train, X_test, y_train, y_test = train_test_split(X, y_encoded, test_size=0.3, random_state=42, stratify=y_encoded)

# Standardize the feature data
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
tree_model = DecisionTreeClassifier()
tree_model.fit(X_train, y_train)
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

```
svm_model = SVC()
svm_model.fit(X_train, y_train)
```

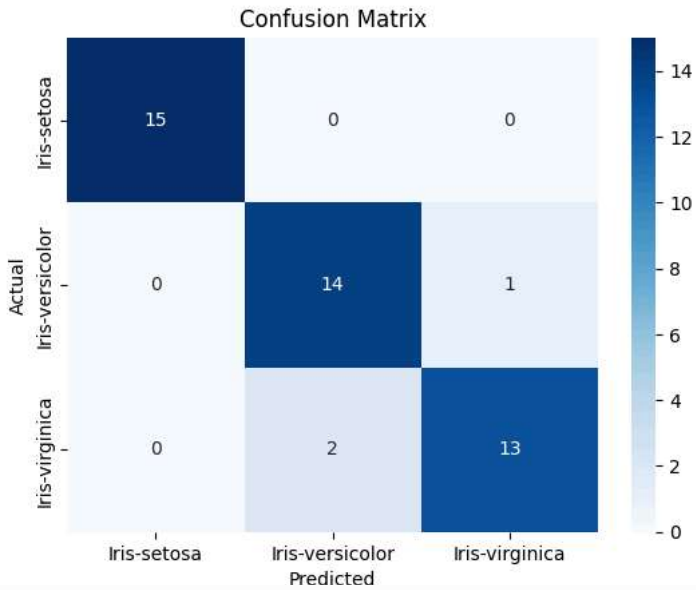
```
▾ SVC
SVC()
```

```
def evaluate(model, name):
    y_pred = model.predict(X_test)
    print(f"🔹 {name} Results:")
    print("Accuracy:", accuracy_score(y_test, y_pred))
    print("Precision:", precision_score(y_test, y_pred, average='weighted'))
```

```
    print("Recall:", recall_score(y_test, y_pred, average='weighted'))
    print("F1 Score:", f1_score(y_test, y_pred, average='weighted'))
    print("Classification Report:\n", classification_report(y_test, y_pred, target_names=label_encoder.classes_))
    print("\n" + "="*50 + "\n")

evaluate(log_model, "Logistic Regression")
evaluate(tree_model, "Decision Tree")
evaluate(svm_model, "Support Vector Machine")
```

```
🔹 Logistic Regression Results:
Accuracy: 0.9111111111111111
Precision: 0.9155354449472096
Recall: 0.9111111111111111
F1 Score: 0.9107142857142857
Classification Report:
                   precision    recall  f1-score   support

     Iris-setosa       1.00      1.00      1.00        15
 Iris-versicolor       0.82      0.93      0.88        15
  Iris-virginica       0.92      0.80      0.86        15

        accuracy                           0.91        45
       macro avg       0.92      0.91      0.91        45
    weighted avg       0.92      0.91      0.91        45


==================================================

🔹 Decision Tree Results:
Accuracy: 0.9555555555555556
Precision: 0.9555555555555556
Recall: 0.9555555555555556
F1 Score: 0.9555555555555556
Classification Report:
                   precision    recall  f1-score   support

     Iris-setosa       1.00      1.00      1.00        15
 Iris-versicolor       0.93      0.93      0.93        15
  Iris-virginica       0.93      0.93      0.93        15

        accuracy                           0.96        45
       macro avg       0.96      0.96      0.96        45
    weighted avg       0.96      0.96      0.96        45


==================================================

🔹 Support Vector Machine Results:
Accuracy: 0.9333333333333333
Precision: 0.9345238095238095
Recall: 0.9333333333333333
F1 Score: 0.9332591768631814
Classification Report:
                   precision    recall  f1-score   support

     Iris-setosa       1.00      1.00      1.00        15
 Iris-versicolor       0.88      0.93      0.90        15
  Iris-virginica       0.93      0.87      0.90        15

        accuracy                           0.93        45
       macro avg       0.93      0.93      0.93        45
    weighted avg       0.93      0.93      0.93        45


==================================================
```

```
from sklearn.metrics import confusion_matrix

best_model = svm_model  # change if needed
y_pred = best_model.predict(X_test)
cm = confusion_matrix(y_test, y_pred)

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=label_encoder.classes_, yticklabels=label_encoder.classes_)
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```


Confusion Matrix
Actual
Iris-setosa
Iris-versicolor
Iris-virginica
15
0
0
0
14
1
0
2
13
Iris-setosa
Iris-versicolor
Iris-virginica
Predicted
14
12
10
8
6
4
2
0