



```
In [ ]: # ♦ PRACTICAL 7 – Decision Tree Classifier & Evaluation
```

```
In [ ]: #Part 1: Decision Tree classifier and evaluation
```

```
In [1]: import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, classification_report, accuracy_
from sklearn.datasets import load_iris
```

```
In [3]: # Load Iris dataset
iris = load_iris()
X = iris.data
y = iris.target
target_names = iris.target_names
```

```
In [5]: # Split data
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42
)
```

```
In [7]: # Train model
dt_classifier = DecisionTreeClassifier(random_state=42)
dt_classifier.fit(X_train, y_train)
```

```
Out[7]: ▼ DecisionTreeClassifier ⓘ ?
DecisionTreeClassifier(random_state=42)
```

```
In [9]: # Predictions
y_pred = dt_classifier.predict(X_test)
```

```
In [11]: print("--- Decision Tree Classifier Evaluation ---")
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy Score: {accuracy:.4f}\n")
```

```
--- Decision Tree Classifier Evaluation ---
Accuracy Score: 1.0000
```

```
In [13]: conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(pd.DataFrame(conf_matrix, index=target_names, columns=target_names))
```

```
Confusion Matrix:
          setosa  versicolor  virginica
setosa         19           0           0
versicolor      0          13           0
virginica        0           0          13
```

```
In [15]: conf_matrix = confusion_matrix(y_test, y_pred)
```

```
print("Confusion Matrix:")
print(pd.DataFrame(conf_matrix, index=target_names, columns=target_names))
```

Confusion Matrix:

	setosa	versicolor	virginica
setosa	19	0	0
versicolor	0	13	0
virginica	0	0	13

In [17]: *#Part 2: Stratified split + tree plot*

```
In [19]: import numpy as np
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, accuracy_score
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
```

```
In [21]: iris = load_iris()
X = iris.data
y = iris.target

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42, stratify=y
)
```

```
In [23]: clf = DecisionTreeClassifier(max_depth=3, random_state=42)
clf.fit(X_train, y_train)
preds = clf.predict(X_test)
```

```
In [25]: print("Accuracy:", accuracy_score(y_test, preds))
print("\nClassification Report:\n", classification_report(y_test, preds))
```

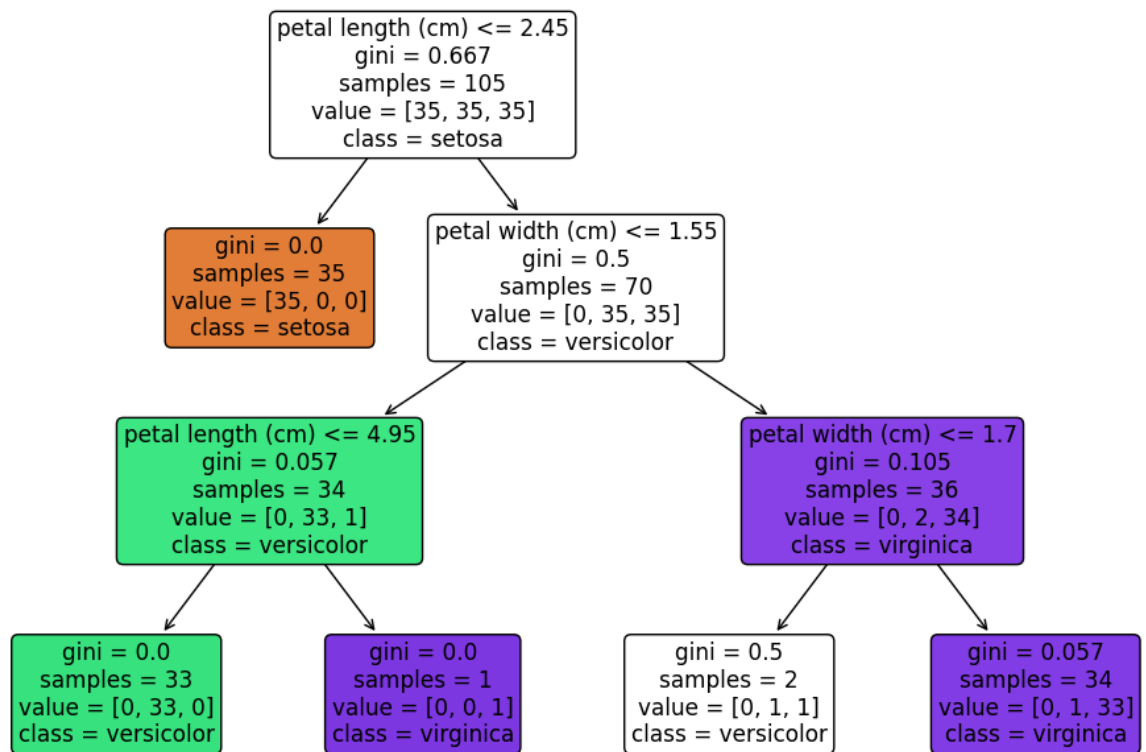
Accuracy: 0.9777777777777777

Classification Report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	15
1	1.00	0.93	0.97	15
2	0.94	1.00	0.97	15
accuracy			0.98	45
macro avg	0.98	0.98	0.98	45
weighted avg	0.98	0.98	0.98	45

```
In [27]: plt.figure(figsize=(12, 8))
plot_tree(
    clf,
    feature_names=iris.feature_names,
    class_names=iris.target_names,
```

```
filled=True,  
rounded=True  
)  
plt.show()
```



In [ ]: