Classificateur d'Arbre de Décision sur le Dataset Iris - Ewan Vidal

Introduction

Ce notebook présente l'implémentation d'un classificateur d'arbre de décision en utilisant le dataset Iris. Il couvre le chargement des données, l'entraînement du modèle, la réalisation de prédictions et la visualisation de l'arbre de décision.

In [18]:
```
# Import necessary libraries
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn import tree
import matplotlib.pyplot as plt
```

Étape 1 : Charger le Dataset Iris

Le **dataset Iris** est un jeu de données bien connu en apprentissage automatique qui contient des informations sur trois types de fleurs iris : *Setosa, Versicolor et Virginica*. Chaque échantillon est décrit par quatre caractéristiques :

- Longueur du sépale
- Largeur du sépale
- Longueur du pétale
- Largeur du pétale

Nous chargeons le dataset avec `sklearn.datasets.load_iris()`.

In [19]:
```
# Step 1: Load the Iris dataset
iris = datasets.load_iris()
```

Étape 2 : Préparer les Données

Nous extrayons :

- `X` (features) : les valeurs numériques représentant les dimensions des sépales et des pétales.
- `y` (labels) : les classes cibles correspondant aux espèces des fleurs iris.

In [20]:
```
# Step 2: Prepare the data
X = iris.data  # Features (sepal length, sepal width, petal length, petal width)
y = iris.target  # Target labels (species)
```

Étape 3 : Diviser le Dataset

Nous séparons le dataset en **ensemble d'entraînement (70%)** et **ensemble de test (30%)** à l'aide de `train_test_split()`, afin d'évaluer le modèle sur des données non vues.

In [21]:
```
# Step 3: Split the dataset into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_
```

Étape 4 : Entraîner un Classificateur d'Arbre de Décision

Nous créons et entraînons un **classificateur d'arbre de décision** en utilisant `sklearn.tree.DecisionTreeClassifier()`. Le modèle apprend ainsi à classifier les espèces d'iris à partir des données d'entraînement.

In [22]:
```
# Step 4: Train a Decision Tree Classifier
clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train, y_train)
```

Out[22]:
```
DecisionTreeClassifier(random_state=42)
```

Étape 5 : Faire des Prédictions

Une fois l'entraînement terminé, nous utilisons le classificateur pour prédire l'espèce des fleurs dans l'ensemble de test.

In [23]:
```
# Step 5: Predict on the test data
y_pred = clf.predict(X_test)
```

Étape 6 : Évaluer la Performance du Modèle

Nous calculons la **précision (accuracy)** du modèle avec `accuracy_score()`, qui mesure son efficacité sur l'ensemble de test.

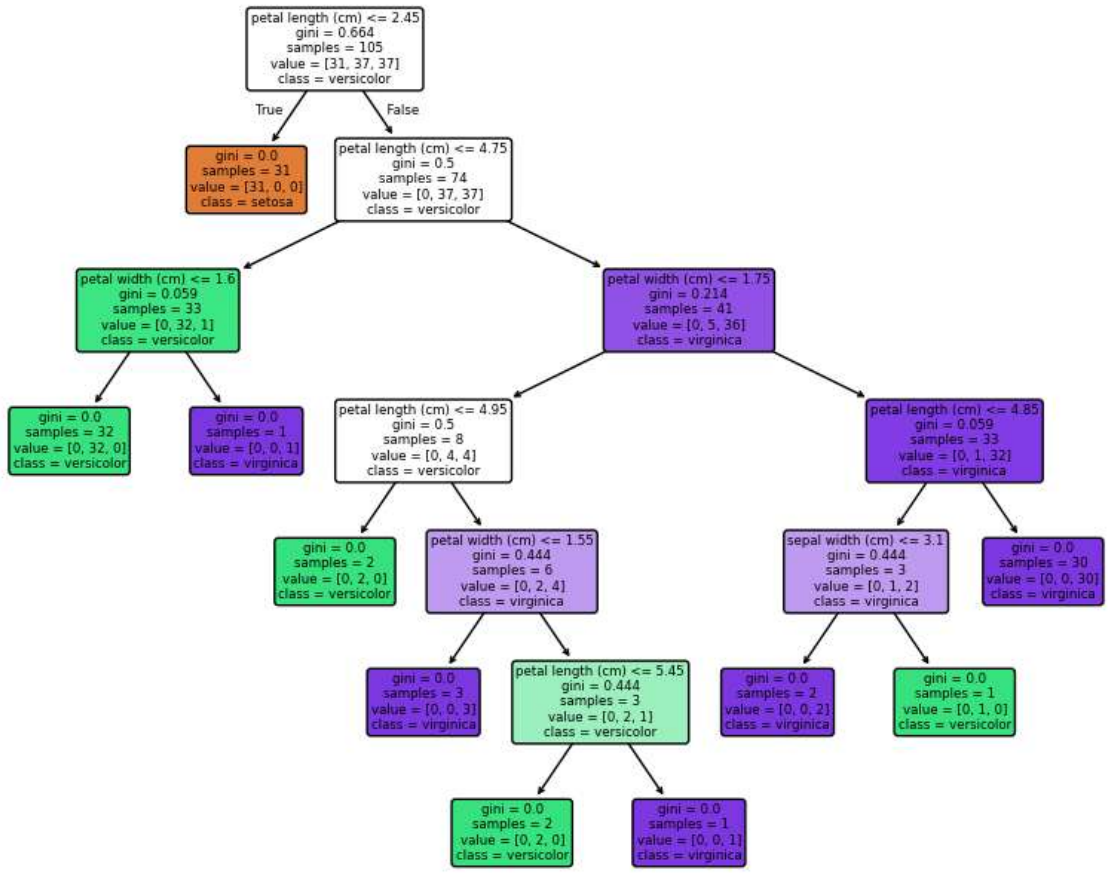
In [24]:
```
# Step 6: Evaluate the model's performance
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy of the Decision Tree Classifier: {accuracy * 100:.2f}%")
```

```
Accuracy of the Decision Tree Classifier: 100.00%
```

Étape 7 : Visualiser l'Arbre de Décision

Nous visualisons l'arbre de décision entraîné à l'aide de `tree.plot_tree()`. Cela permet d'obtenir une représentation graphique interprétable de la façon dont le modèle prend ses décisions.

In [25]:
```
# Step 7: Visualize the Decision Tree
plt.figure(figsize=(10, 8))
tree.plot_tree(clf, filled=True, feature_names=iris.feature_names, class_names=i
plt.show()
```



Étape 8 : Classifier un Nouvel Échantillon

Nous fournissons un échantillon avec des dimensions spécifiques de sépale et de pétale, puis prédisons son espèce en utilisant le modèle entraîné.

In [27]:
```
# Step 8: Classification example - Classify a new sample
# Define a new sample (e.g., new iris flower with specific features)
new_sample = [[5.5, 2.4, 3.8, 1.1]]  # Example: sepal length = 5.5, sepal width
new_sample_2 = [[6.5, 3.0, 5.2, 2.0]]  # Example: sepal length = 6.5, sepal widt

# Predict the class for the new sample
predicted_class = clf.predict(new_sample)
predicted_class_2 = clf.predict(new_sample_2)

# Print the predicted class and corresponding species name
print(f"The predicted class for the new sample is: {iris.target_names[predicted_
print(f"The predicted class for the new sample is: {iris.target_names[predicted_
```

```
The predicted class for the new sample is: versicolor
The predicted class for the new sample is: virginica
```