In [1]:

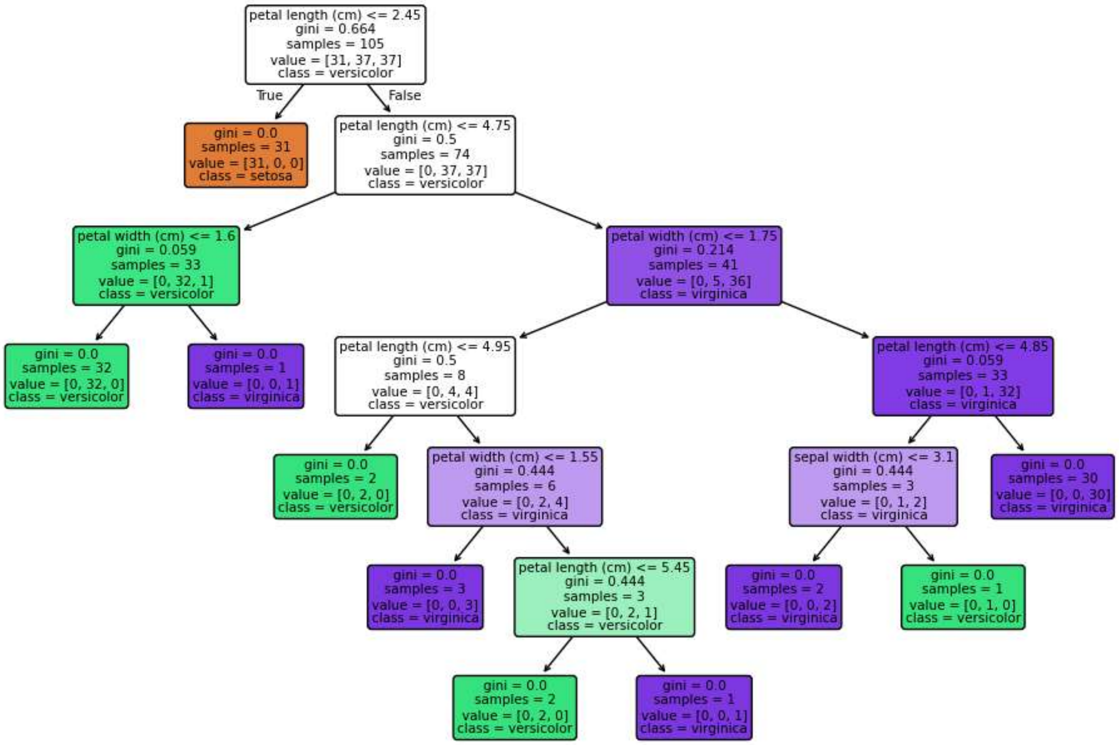
```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_m
from sklearn import tree
iris = load_iris()
X = iris.data
y = iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, rando
dt_classifier = DecisionTreeClassifier(random_state=42)
dt_classifier.fit(X_train, y_train)
y_pred = dt_classifier.predict(X_test)
print("Accuracy: ", accuracy_score(y_test, y_pred))
print("Classification Report:")
print(classification_report(y_test, y_pred))
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))
plt.figure(figsize=(12,8))
tree.plot_tree(dt_classifier, filled=True, feature_names=iris.feature_names, c
plt.show()
```

```
Accuracy:  1.0
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        19
           1       1.00      1.00      1.00        13
           2       1.00      1.00      1.00        13

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
weighted avg       1.00      1.00      1.00        45

Confusion Matrix:
[[19  0  0]
 [ 0 13  0]
 [ 0  0 13]]
```



In [3]:

```python
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error, r2_score
diabetes = load_diabetes()
X = diabetes.data
y = diabetes.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, rando
dt_regressor = DecisionTreeRegressor(random_state=42)
dt_regressor.fit(X_train, y_train)
y_pred = dt_regressor.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error: {mse:.2f}")
print(f"R-squared: {r2:.2f}")
```

```
Mean Squared Error: 5697.79
R-squared: -0.06
```

In [4]:

```
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
iris = load_iris()
X = iris.data
y = iris.target
dt_classifier = DecisionTreeClassifier(random_state=42)
param_grid = {
    'max_depth': [3, 5, 10, None],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

grid_search = GridSearchCV(estimator=dt_classifier, param_grid=param_grid, cv=
grid_search.fit(X, y)
print("Best Hyperparameters:", grid_search.best_params_)
best_model = grid_search.best_estimator_
best_model.fit(X, y)
```

```
Fitting 5 folds for each of 36 candidates, totalling 180 fits
Best Hyperparameters: {'max_depth': 3, 'min_samples_leaf': 1, 'min_samples_sp
lit': 2}
```

Out[4]:

```
▾ DecisionTreeClassifier                 i ?
                                          (https://scikit-
                                          learn.org/1.5/modules/generated/s
DecisionTreeClassifier(max_depth=3, random_state=42)
```

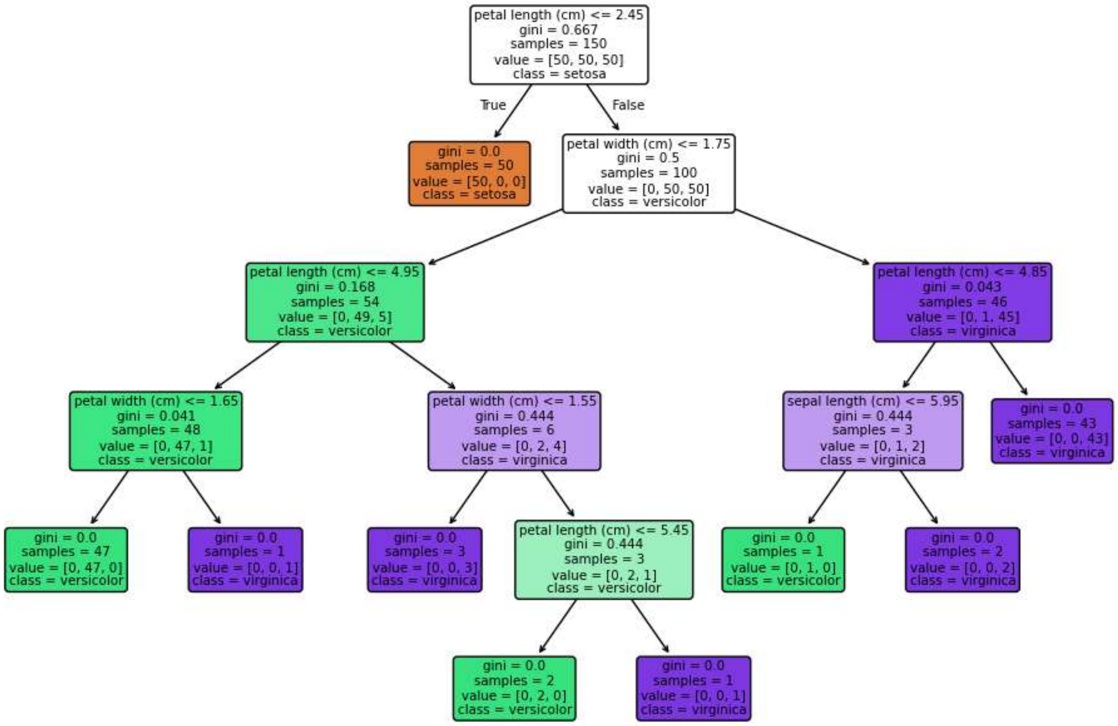
In [6]:

```
from sklearn.datasets import load_iris
from sklearn.model_selection import cross_val_score
from sklearn.tree import DecisionTreeClassifier
iris = load_iris()
X = iris.data
y = iris.target
dt_classifier = DecisionTreeClassifier(random_state=42)
cv_scores = cross_val_score(dt_classifier, X, y, cv=5)
print("Cross-validation scores: ", cv_scores)
print("Mean Cross-validation score: ", cv_scores.mean())
```

```
Cross-validation scores:  [0.96666667 0.96666667 0.9        0.93333333 1.
]
Mean Cross-validation score:  0.9533333333333334
```

In [7]:

```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
import matplotlib.pyplot as plt
iris = load_iris()
X = iris.data
y = iris.target
dt_classifier = DecisionTreeClassifier(random_state=42)
dt_classifier.fit(X, y)
plt.figure(figsize=(12, 8))
tree.plot_tree(dt_classifier, filled=True, feature_names=iris.feature_names, c
plt.show()
print("Feature Importance: ", dt_classifier.feature_importances_)
```



```
Feature Importance:  [0.01333333 0.         0.56405596 0.42261071]
```

In [ ]: