Decision Tree Post Pruning

Code	Explanation
import pandas as pd	Imports Pandas library, which is widely used for handling and analyzing structured da
import matplotlib.pyplot as plt	Imports Matplotlib's pyplot module to create plots, graphs, and charts for data visualiz
from sklearn.datasets import load_iris	Imports function to load the popular Iris dataset, which is often used for classification
iris = load_iris()	Loads the Iris dataset and stores it in the variable 'iris'.
iris.data	Provides the numerical feature values of all Iris flowers (like petal length, sepal width,
iris.target	Provides the class labels (0=setosa, 1=versicolor, 2=virginica).
import seaborn as sns	Imports Seaborn, a library built on top of Matplotlib for easier visualization and datase
df = sns.load_dataset('iris')	Loads Iris dataset as a Pandas DataFrame using Seaborn for easier manipulation.
df.head()	Displays the first five rows of the dataset for inspection.
x = df.iloc[:, :-1]	Selects all columns except the last one as feature variables (independent variables).
y = iris.target	Assigns the target labels from sklearn's Iris dataset as the dependent variable.
x, y	Displays features and target arrays together for inspection.
from sklearn.model_selection import train_test_split	Imports utility to split dataset into training and testing sets.
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.33, random_state=42)	Splits dataset into training (67%) and testing (33%) sets with a fixed random state for
x_train	Shows the training dataset features.
from sklearn.tree import DecisionTreeClassifier	Imports DecisionTreeClassifier class from sklearn for building classification trees.
treemodel = DecisionTreeClassifier(max_depth=3)	Creates a decision tree classifier with a maximum depth of 3. Limiting depth is a form
treemodel.fit(x_train, y_train)	Trains the decision tree model using the training dataset.
from sklearn import tree	Imports sklearn's tree module to visualize the decision tree.
plt.figure(figsize=(10, 8))	Creates a new figure with custom width and height for better visualization.
tree.plot_tree(treemodel, filled=True)	Plots the trained decision tree, coloring nodes by their class label.
y_pred = treemodel.predict(x_test)	Generates predictions for the test set using the trained decision tree.
y_pred	Displays the predicted labels.
from sklearn.metrics import accuracy_score, classification_report	Imports functions to evaluate model accuracy and generate a classification report.
score = accuracy_score(y_test, y_pred)	Calculates how many test samples were correctly classified by the model.
print(score)	Prints the computed accuracy score.
print(classification_report(y_pred, y_test))	Generates and prints a report with precision, recall, and F1-score for each class.
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay	Imports utilities to compute and display a confusion matrix.
cm = confusion_matrix(y_test, y_pred)	Creates a confusion matrix to compare actual vs. predicted labels.
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=iris.target_names)	Prepares display labels for the matrix, using class names.
disp.plot(cmap='Blues')	Plots the confusion matrix with a blue color map.

plt.title("Confusion Matrix")	Sets a title for the confusion matrix plot.
plt.show()	Displays the ROC curve plot.
from sklearn.metrics import roc_curve, auc, RocCurveDisplay	Imports functions to calculate ROC curve and Area Under Curve (AUC).
y_scores = treemodel.predict_proba(x_test)	Generates class probability estimates for each test sample.
fpr, tpr, thresholds = roc_curve(y_test, y_scores[:, 0], pos_label=0)	Computes False Positive Rate and True Positive Rate for class 0 to build the ROC cu
roc_auc = auc(fpr, tpr)	Calculates the Area Under Curve (AUC), summarizing ROC performance.
plt.figure()	Creates a new figure for ROC curve plotting.
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.2f})')	Plots ROC curve with AUC value in the legend.
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')	Adds a reference diagonal line for random performance.
plt.xlabel('False Positive Rate')	Labels the x-axis as False Positive Rate.
plt.ylabel('True Positive Rate')	Labels the y-axis as True Positive Rate.
plt.title('ROC Curve - Decision Tree (Class 0)')	Adds title for the ROC curve plot.
plt.legend(loc="lower right")	Adds legend to identify ROC curve.
plt.grid()	Adds a grid for easier interpretation of the ROC curve.
plt.show()	Displays the ROC curve plot.