A PYTHON PROGRAM TO IMPLEMENT

DECISION TREE

Expt no. 7

Name: Jayasuriya.j

Roll no: 241801102

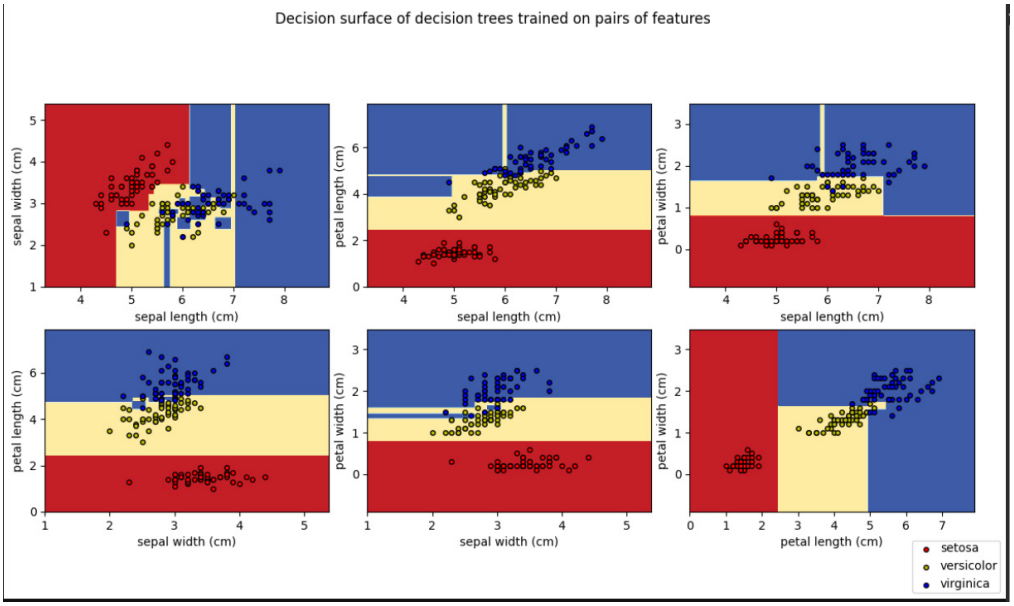
PROGRAM:

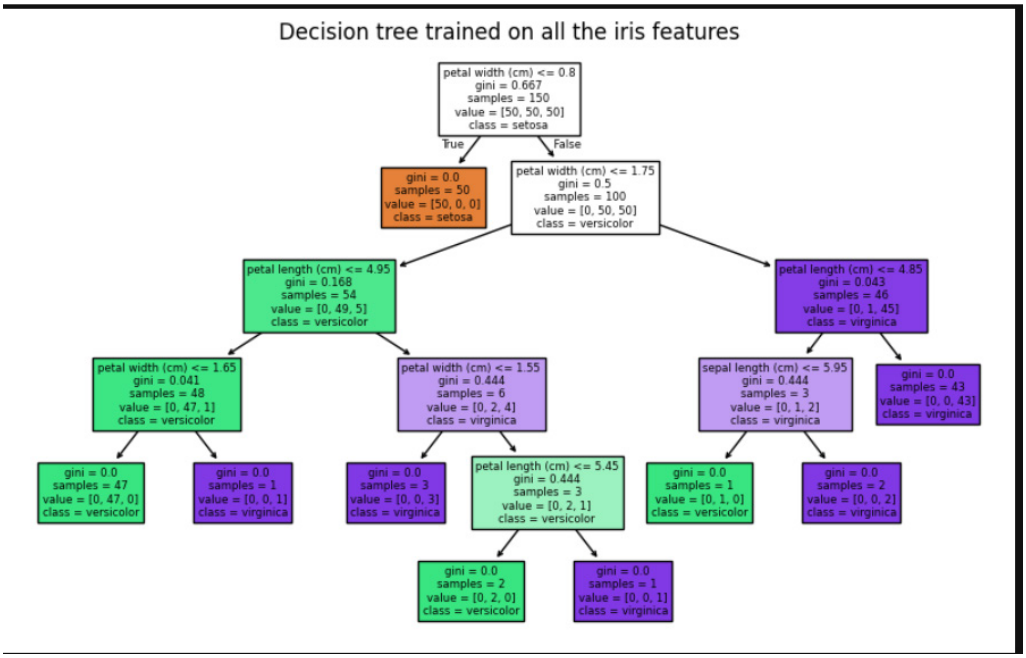
```
from sklearn.datasets import load_iris
import numpy as np
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier, plot_tree

# Data
iris = load_iris()
X_all = iris.data
y = iris.target
n_classes = len(iris.target_names)

# ---- Pairwise decision surfaces ----
pairs = [(0,1), (0,2), (0,3), (1,2), (1,3), (2,3)]
plot_step = 0.02 plot_colors = ["r", "y", "b"]

fig, axes = plt.subplots(2, 3, figsize=(12, 7))

for pairidx, pair in enumerate(pairs):


    # Two-feature slice
    X = X_all[:, pair]
```

```
    # Train
    clf = DecisionTreeClassifier().fit(X, y)

    # Grid
    x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, plot_step),
                         np.arange(y_min, y_max, plot_step))

    # Predict on grid
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()]).reshape(xx.shape)

    ax = axes[pairidx // 3, pairidx % 3]
    ax.contourf(xx, yy, Z, cmap=plt.cm.RdYlBu)

    # Training points
    for i, color in zip(range(n_classes), plot_colors):
        idx = np.where(y == i)
        ax.scatter(X[idx, 0], X[idx, 1],
                   c=color, label=iris.target_names[i],
                   edgecolor="black", s=15)

    ax.set_xlabel(iris.feature_names[pair[0]])
    ax.set_ylabel(iris.feature_names[pair[1]])
    ax.set_xlim(xx.min(), xx.max())
ax.set_ylim(yy.min(), yy.max()) fig.suptitle("Decision surface of decision

trees trained on pairs of features")
# Single legend for all subplots
```

```
handles, labels = axes[0, 0].get_legend_handles_labels() fig.legend(handles,
labels, loc="lower right", borderpad=0.2, handletextpad=0.2)

fig.tight_layout(h_pad=0.5, w_pad=0.5, pad=2.5, rect=[0, 0.03, 1, 0.95])

# ---- Full tree on all features ----
plt.figure(figsize=(10, 6))
clf_all = DecisionTreeClassifier().fit(X_all, y)
plot_tree(clf_all,
          feature_names=iris.feature_names,
          class_names=iris.target_names,
          filled=True)
plt.title("Decision tree trained on all the iris features")
plt.show()
```

OUTPUT:


Decision surface of decision trees trained on pairs of features
sepal width (cm)
sepal length (cm)
petal length (cm)
petal width (cm)
setosa
versicolor
virginica


Decision tree trained on all the iris features
petal width (cm) <= 0.8
gini = 0.667
samples = 150
value = [50, 50, 50]
class = setosa
True
False
gini = 0.0
samples = 50
value = [50, 0, 0]
class = setosa
petal width (cm) <= 1.75
gini = 0.5
samples = 100
value = [0, 50, 50]
class = versicolor
petal length (cm) <= 4.95
gini = 0.168
samples = 54
value = [0, 49, 5]
class = versicolor
petal length (cm) <= 4.85
gini = 0.043
samples = 46
value = [0, 1, 45]
class = virginica
petal width (cm) <= 1.65
gini = 0.041
samples = 48
value = [0, 47, 1]
class = versicolor
petal width (cm) <= 1.55
gini = 0.444
samples = 6
value = [0, 2, 4]
class = virginica
sepal length (cm) <= 5.95
gini = 0.444
samples = 3
value = [0, 1, 2]
class = virginica
gini = 0.0
samples = 43
value = [0, 0, 43]
class = virginica
gini = 0.0
samples = 47
value = [0, 47, 0]
class = versicolor
gini = 0.0
samples = 1
value = [0, 0, 1]
class = virginica
gini = 0.0
samples = 3
value = [0, 0, 3]
class = virginica
petal length (cm) <= 5.45
gini = 0.444
samples = 3
value = [0, 2, 1]
class = versicolor
gini = 0.0
samples = 1
value = [0, 1, 0]
class = versicolor
gini = 0.0
samples = 2
value = [0, 0, 2]
class = virginica
gini = 0.0
samples = 2
value = [0, 2, 0]
class = versicolor
gini = 0.0
samples = 1
value = [0, 0, 1]
class = virginica