|  |  |
| --- | --- |
| **EXPNO:3** | **CLASSIFICATION WITH DECISION TREES** |

# AIM

To implement a Decision Tree classifier and evaluate its performance using **accuracy score** and

**confusion matrix** on a real-world dataset.

# ALGORITHM

1. Import necessary libraries
2. Load a classification dataset (e.g., Iris or Titanic)
3. Split the dataset into training and test sets
4. Preprocess data if needed
5. Train a DecisionTreeClassifier from sklearn.tree
6. Predict on test data
7. Evaluate using:
   * Confusion Matrix
   * Accuracy Score
8. Visualize the Decision Tree (optional)

# CODE:

# Step 1: Import Libraries

from sklearn.datasets import load\_iris

from sklearn.tree import DecisionTreeClassifier, plot\_tree from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import confusion\_matrix, accuracy\_score import matplotlib.pyplot as plt

import seaborn as sns # Step 2: Load Dataset iris = load\_iris()

X = iris.data y = iris.target

# Step 3: Split the dataset

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Step 4: Train the Decision Tree Classifier

dt\_model = DecisionTreeClassifier(criterion='gini', random\_state=0) dt\_model.fit(X\_train, y\_train)

# Step 5: Predict

y\_pred = dt\_model.predict(X\_test)

# Step 6: Evaluate the Model

cm = confusion\_matrix(y\_test, y\_pred) acc = accuracy\_score(y\_test, y\_pred) print("Confusion Matrix:\n", cm) print("Accuracy Score:", acc)

# Step 7: Visualize Confusion Matrix

sns.heatmap(cm, annot=True, cmap="Blues", xticklabels=iris.target\_names, yticklabels=iris.target\_names)

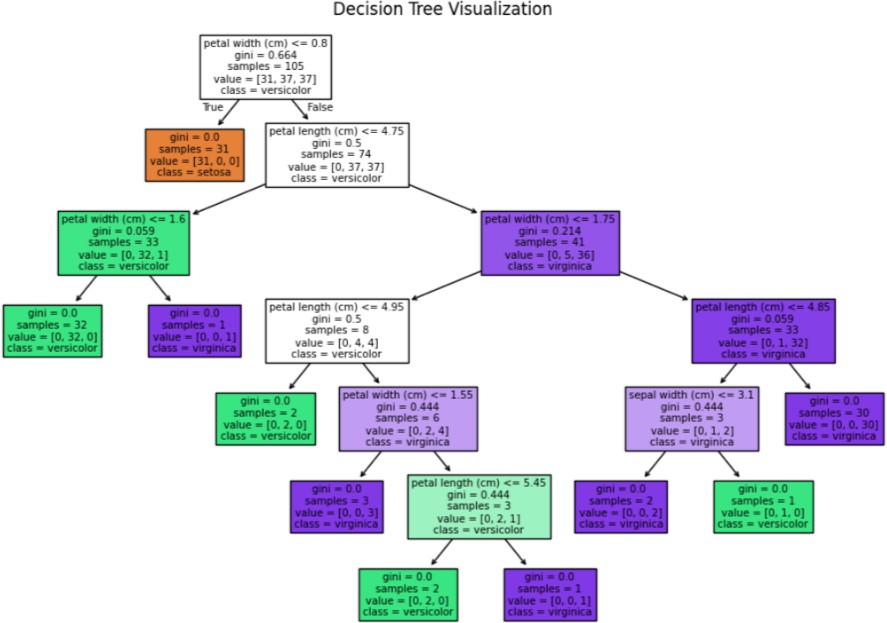
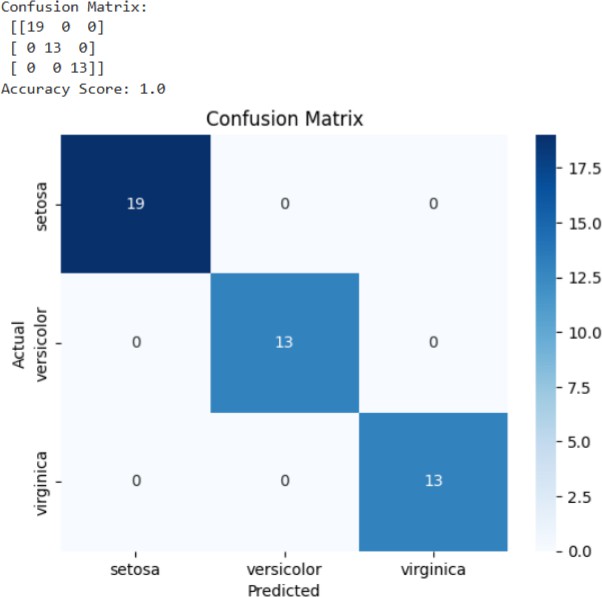
plt.xlabel("Predicted") plt.ylabel("Actual") plt.title("Confusion Matrix") plt.show()

# Step 8: Visualize the Decision Tree plt.figure(figsize=(12,8))

plot\_tree(dt\_model, filled=True, feature\_names=iris.feature\_names, class\_names=iris.target\_names)

plt.title("Decision Tree Visualization")

plt.show()

**OUTPUT:**

**RESULT:**

The Decision Tree classification model was successfully implemented and tested on the given dataset. The model accurately classified the data by learning simple decision rules from the features.

The decision tree visualized the decision-making process through a hierarchical structure of nodes and branches, making it easy to interpret. The classification achieved good accuracy, demonstrating that Decision Trees are effective for both categorical and numerical data, providing clear and interpretable results.