BE(E&C)	Data Science & Visualization Lab	
Experiment No.:08	**Implementation of decision tree classifier using python with suitable dataset.**	**Page:** /6

- **Aim:** Implementation of decision tree classifier using python with suitable dataset.
- **Software Used:** Python 3.12, Jupyter Notebook
- **Learning Objective:**

  By the end of this student should be able to:

  1. Understand the principles and functioning of Decision Trees in classification.
  2. Learn how to preprocess data for classification tasks.
  3. Implement a Decision Tree Classifier using Scikit-learn.
  4. Evaluate model performance using metrics such as accuracy, confusion matrix, and classification report.
  5. Visualize the decision tree structure and results.

- **Learning Outcomes:**

  After performing the experiment students will be able to-

  1. Ability to preprocess and manipulate datasets using Pandas and NumPy.
  2. Knowledge of the theoretical foundations and algorithms behind Decision Trees.
  3. Skills in implementing and evaluating a Decision Tree Classifier using Scikit-learn.
  4. Understanding how to visualize the decision tree for better interpretation.

- **Theory:**

  • **Decision Tree Classifier**:

  - A Decision Tree is a flowchart-like structure that splits the dataset into subsets based on feature values. Each internal node represents a feature, each branch represents a decision rule, and each leaf node represents an outcome (class label).
  - Decision Trees are intuitive and easy to interpret, making them popular for classification tasks.

  • **Working of Decision Trees**:

  - **Splitting**: The process of dividing the dataset into smaller subsets based on certain criteria. The goal is to create homogeneous subsets where instances in the same subset belong to the same class.
  - **Criteria for Splitting**: Common criteria include Gini impurity and Information Gain (Entropy).

- **Gini Impurity**: Measures the likelihood of an incorrect classification of a new instance if it was randomly labeled according to the distribution of labels in the subset.
- **Information Gain**: Measures the reduction in entropy (uncertainty) after the dataset is split.

• **Advantages of Decision Trees**:

- Easy to understand and interpret.
- Requires little data preprocessing (no need for normalization).
- Can handle both numerical and categorical data.

• **Disadvantages of Decision Trees**:

- Prone to overfitting, especially with deep trees.
- Sensitive to small changes in the data, which can lead to different splits.

• **Performance Metrics**:

- **Accuracy**: The proportion of correct predictions to total predictions.
- **Confusion Matrix**: A table that describes the performance of a classification model by showing true positives, true negatives, false positives, and false negatives.
- **Classification Report**: Provides precision, recall, and F1-score for each class.

Code:

```
import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

import matplotlib.pyplot as plt

from sklearn import tree


# Load the Iris dataset

from sklearn.datasets import load_iris

iris = load_iris()
```

BE(E&C)	Data Science & Visualization Lab	
Experiment No.:08	**Implementation of decision tree classifier using python with suitable dataset.**	**Page:** /6

```
data = pd.DataFrame(data=iris.data, columns=iris.feature_names)
data['target'] = iris.target

# Display the first few rows of the dataset
print(data.head())

# Split data into features and target
X = data.drop('target', axis=1)
y = data['target']

# Split the data into training and testing sets (80/20 split)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create and train the Decision Tree Classifier
dt_classifier = DecisionTreeClassifier(random_state=42)
dt_classifier.fit(X_train, y_train)

# Make predictions
y_pred = dt_classifier.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print(f'Accuracy: {accuracy:.2f}')
```

```
print(f'Confusion Matrix:\n{conf_matrix}')

print(f'Classification Report:\n{class_report}')


# Visualize the Decision Tree

plt.figure(figsize=(12,8))

tree.plot_tree(dt_classifier, feature_names=iris.feature_names,
class_names=iris.target_names, filled=True)

plt.title('Decision Tree Classifier')

plt.show()
```

BE(E&C)	Data Science & Visualization Lab	
Experiment No.:08	**Implementation of decision tree classifier using python with suitable dataset.**	**Page:** /6

❖ Output

The output will include:

- Accuracy of the Decision Tree Classifier.
- Confusion matrix values.
- Classification report showing precision, recall, and F1-score for each class.
- A visual representation of the decision tree structure.

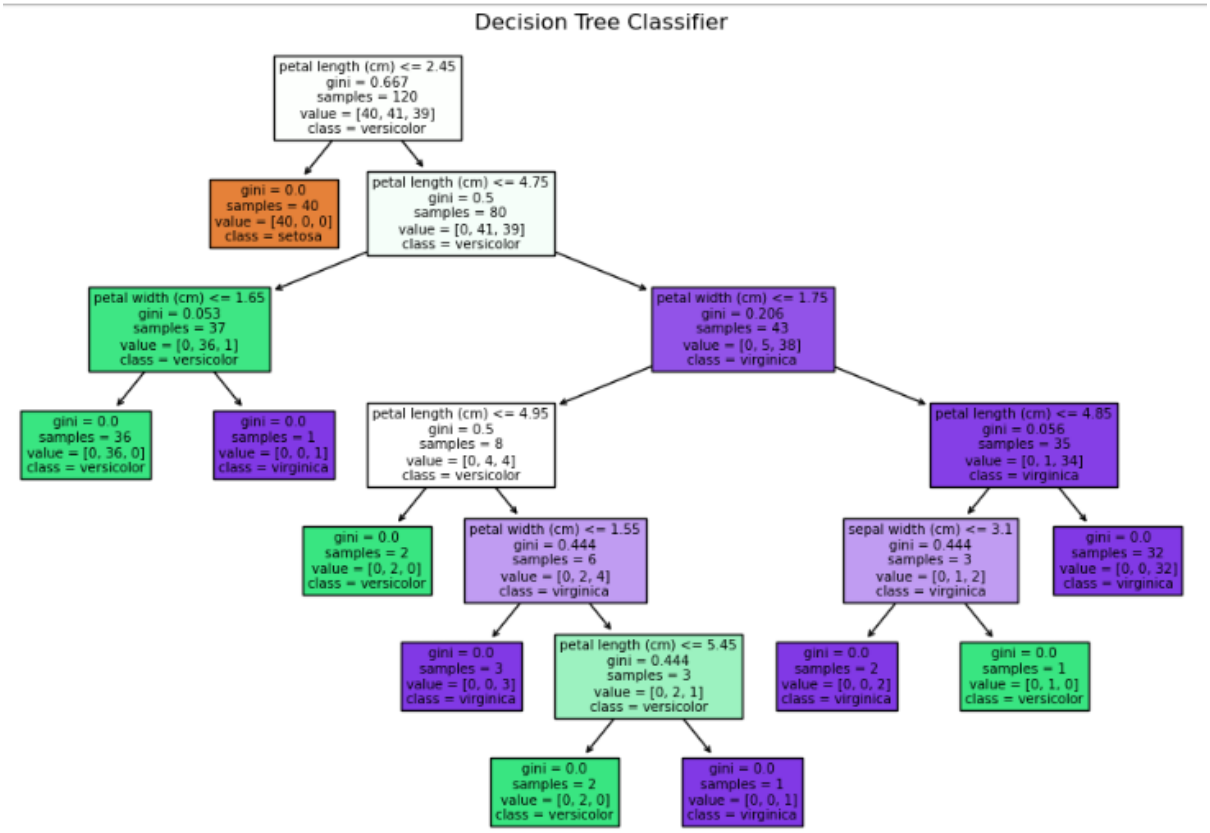
```
sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2

   target
0       0
1       0
2       0
3       0
4       0
Accuracy: 1.00
Confusion Matrix:
[[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       1.00      1.00      1.00         9
           2       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

BE(E&C)	Data Science & Visualization Lab	
Experiment No.:08	**Implementation of decision tree classifier using python with suitable dataset.**	**Page:** /6



❖ Conclusion:

_____

_____

_____

_____

❖ Questions:

1. Explain the concept of a Decision Tree and how it is used in classification tasks.
2. What are the criteria for splitting nodes in a Decision Tree, and how do they affect the model's performance?
3. Discuss the advantages and disadvantages of using Decision Trees for classification.
4. What is overfitting in the context of Decision Trees, and how can it be prevented?
5. How do metrics like accuracy, precision, recall, and F1-score help in evaluating the performance of a classification model?