Name: Devyani Khabiya

Class: TE9-B-27

Subject: DWM

```
EXPERIMENT NO. 3
```

Title: Implementation of Classification algorithm (Decision Tree/Naive Bayes)

Aim: To implement a classification algorithm using Decision Tree on the dataset and evaluate its performance.

Introduction: Classification is a supervised machine learning technique used to categorize data into predefined classes. Decision Trees are widely used for classification tasks as they provide an interpretable model by splitting data based on feature values. In this experiment, we use the Decision Tree classifier on the Iris dataset, which contains three classes of flowers: Setosa, Versicolor, and Virginica, based on four features—sepal length, sepal width, petal length, and petal width. The dataset is split into training and testing sets, and the model's performance is evaluated using various metrics such as accuracy, confusion matrix, and classification report.

Procedure:

1. **Import Required Libraries:**
   - Load necessary libraries such as NumPy, Pandas, Matplotlib, and Scikit-learn.
2. **Load the Dataset:**
   - Use the Iris dataset from sklearn.datasets.
   - Extract features (X) and target labels (y).
3. **Split Data into Training and Testing Sets:**
   - Use train_test_split() to divide data into **80% training** and **20% testing**.
4. **Train the Decision Tree Classifier:**
   - Create a Decision Tree model using DecisionTreeClassifier with a max depth of 3.
   - Fit the model using training data.
5. **Make Predictions:**
   - Use the trained model to predict the labels of the test set.
6. **Evaluate the Model:**
   - Calculate accuracy using accuracy_score().
   - Display the confusion matrix using confusion_matrix().
   - Generate a classification report with precision, recall, and F1-score.
7. **Visualize the Decision Tree:**
   - Plot the decision tree structure using plot_tree() for better interpretability.
8. **Interpret Results:**
   - Analyze **accuracy, confusion matrix, and classification report** to understand the model's performance.

Program Code:

```
# Import necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn import datasets
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.tree import plot_tree


# Load the dataset (Iris dataset as an example)
iris = datasets.load_iris()
X = iris.data  # Features
y = iris.target  # Target labels
```

```
# Split data into training and testing sets (80% train, 20% test)
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
clf = DecisionTreeClassifier(criterion='gini', max_depth=3, random_state=42)
clf.fit(X_train, y_train)

# Markdown for bold text in Colab
from IPython.display import display, Markdown
display(Markdown("**Implementation/Output snap shot:**"))

# Make predictions
y_pred = clf.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f'\nAccuracy: {accuracy:.2f}')

# Display confusion matrix
print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))

# Display classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# Visualize the Decision Tree
plt.figure(figsize=(10, 6))
plot_tree(clf, filled=True, feature_names=iris.feature_names, class_names=iris.target_names)
plt.show()
```

Implementation/Output snap shot:

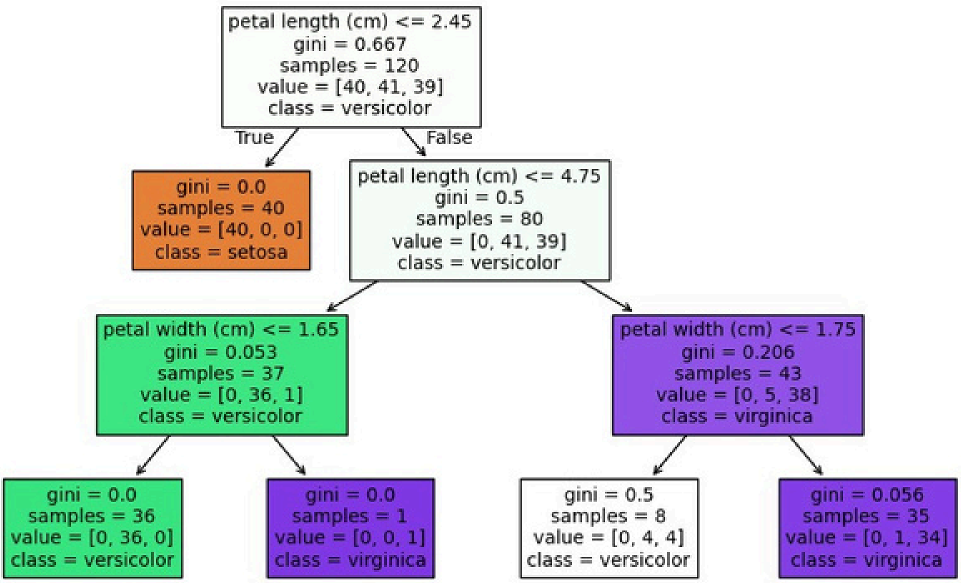
```
Accuracy: 1.00

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
```



> **Conclusion:** The Decision Tree classifier successfully classified the Iris dataset with high accuracy. The confusion matrix and classification report provided insights into model performance. The tree visualization helped in understanding decision splits.

While Decision Trees are interpretable, they can overfit if not tuned properly. Overall, this experiment demonstrated the effectiveness of Decision Trees in classification tasks.

Review Questions:

1. **What is a Decision Tree classifier, and how does it work?**

Ans. A Decision Tree classifier is a supervised learning algorithm used for classification and regression tasks. It works by recursively splitting the dataset into subsets based on feature values, forming a tree-like structure. Each internal node represents a decision based on a feature, branches indicate possible outcomes, and leaf nodes represent class labels. The algorithm selects the best feature for splitting using criteria like **Gini impurity** or **entropy**.

2. **Explain the Naive Bayes algorithm and its underlying assumptions.**

Ans. Naive Bayes is a probabilistic classifier based on **Bayes' Theorem**, assuming independence between features.
It calculates the probability of a class given the feature values and selects the class with the highest probability. The key assumptions are:

- All features are conditionally independent given the class label.
- The effect of a feature on a class is independent of other features.
- The probability distributions follow a simple distribution (e.g., **Gaussian for continuous features**).

3. **Compare the working principles of Decision Tree and Naive Bayes classifiers.**

Ans.

- **Decision Tree:**
    - Uses a hierarchical, rule-based approach, making decisions by recursively splitting data.
    - Highly interpretable but prone to overfitting.
- **Naive Bayes:**
    - Uses probability-based classification, assuming feature independence.
    - Computationally efficient, effective for large datasets, but less interpretable.
- **Comparison:**
    - **Decision Trees** are better for handling complex relationships between features.
    - **Naive Bayes** performs well when independence assumptions hold and is robust for high-dimensional data.

4. **What are the different types of Decision Tree splitting criteria?**

Ans.

- **Gini Impurity:** Measures the probability of incorrect classification by a randomly chosen feature split.
- **Entropy (Information Gain):** Measures the uncertainty in data and selects splits that maximize information gain.
- **Chi-Square:** Evaluates statistical significance for categorical features.
- **Reduction in Variance:** Used for regression tasks to minimize variance in child nodes.

Github Link: https://github.com/Pralix20/DWMexp t is a Decision Tree Classifier, and How Does it Work?

sion Tree is a supervised learning algorithm used for both classification and regression. It works by recursively splitt into smaller subsets based on feature values to form a tree-like structure.

works:

ot Node: Represents the entire dataset.

itting: The algorithm chooses the best feature and threshold to split the data based on a criterion like Gini In ormation Gain.

ernal Nodes: Represent decisions (feature-based splits).

af Nodes: Represent the predicted class labels.

e process continues until a stopping condition is met (e.g., maximum depth, minimum number of samples in a node).

n the Naïve Bayes Algorithm and Its Underlying Assumptions

Bayes is a classification algorithm based on Bayes' Theorem, which calculates the probability of a class given a s.

Theorem:

X)=P(X)P(X ∣ C)⋅P(C)Where:

C ∣ X)P(C|X)P(C ∣ X): Posterior probability of class CCC given input features XXX

X ∣ C)P(X|C)P(X ∣ C): Likelihood of features XXX given class CCC

C)P(C)P(C): Prior probability of class CCC

X)P(X)P(X): Probability of the features

ptions:

atures are independent of each other given the class label.

ssumes equal importance of each feature and no interaction between them, which is often not realistic but works actice.

pare the Working Principles of Decision Tree and Naïve Bayes Classifiers

t	Decision Tree	Naïve Bayes
	Non-probabilistic, tree-based model	Probabilistic model based on Bayes' Theorem
ng Approach	Greedy, recursive partitioning	Statistical estimation
retability	Easy to interpret and visualize	Less intuitive due to probabilities
ption	No assumptions about data distribution	Assumes conditional independence of features
ing Non-linear	Good at capturing non-linear relationships	Assumes feature independence, so less flexible
mance	Can overfit without pruning	Robust, especially with small datasets

Are the Different Types of Decision Tree Splitting Criteria?

i Impurity: Measures the frequency of a randomly chosen element being misclassified.

i=1−∑pi2Gini = 1 - \sum p_i^2Gini=1−∑pi2

rmation Gain (Entropy): Measures the reduction in entropy before and after a split.

opy=−∑pilog2(pi)Entropy = - \sum p_i \log_2(p_i)Entropy=−∑pilog2(pi)

n Ratio: A normalized version of information gain to reduce bias toward features with many levels.

-square: Statistical test to determine the relevance of a split.

uction in Variance: Used for regression trees to reduce variance in the target variable after a split.