Name – Yash Mav | **Div** – TY09 | **Roll no** – 33 |

Subject – DWM | **Experiment No** – 3

Aim :- Implementation of Classification Algorithm (Decision Tree / Naïve Bayes) using Python **Introduction** :-

- Classification is a supervised machine learning task to predict the class label of data.
- Decision Trees: Tree-like models using a series of rules based on data features for predictions. Easy to interpret and visualize.
- Naïve Bayes: Probabilistic algorithm based on Bayes' theorem. Assumes feature independence. Calculates probabilities of a data point belonging to each class and predicts the class with the highest probability.

Procedure :-

1. Import Necessary Libraries
2. Load and Prepare the Dataset
3. Split Data into Training and Testing Sets
4. Create and Train the Decision Tree Classifier
5. Make Predictions
6. Evaluate the Model
7. Visualize the Decision Tree

```
# 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)

# Make predictions
y_pred = clf.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
 print(f'Accuracy: {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()
```

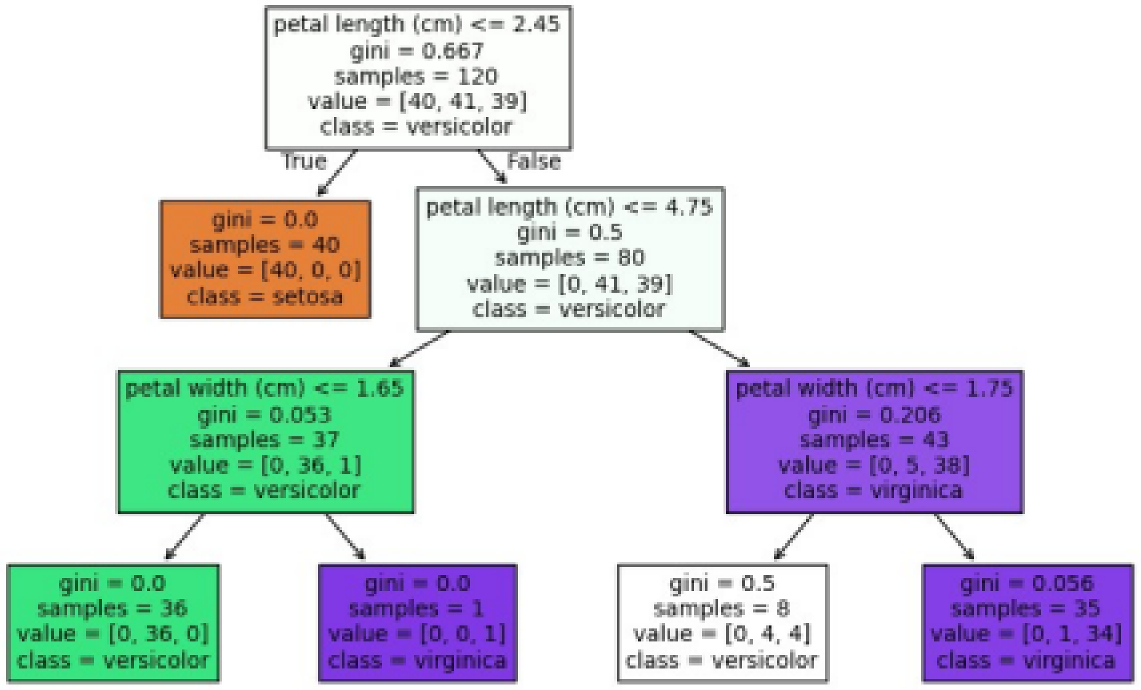
```
Accuracy: 1.00

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 :- In this experiment, we implemented Decision Tree and Naïve Bayes for classification using Python. Decision Tree provided an interpretable model but risked overfitting, while Naïve Bayes performed well with independent features. Performance evaluation showed that each algorithm excels in different scenarios. Overall, both methods effectively classify data, demonstrating their importance in DWM applications.

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 tasks. · It works by recursively splitting the dataset into branches based on feature values, forming a tree-like structure.

· At each node, the best splitting criterion is selected to maximize information gain or minimize impurity.

· The process continues until a stopping condition is met (e.g., no further splits improve classification).

· The final leaf nodes represent class labels for prediction.

1. Explain the Naïve Bayes algorithm and its underlying assumptions. Ans

- · Naïve Bayes is a probabilistic classifier based on Bayes' Theorem.

· It assumes that features are conditionally independent, given the class label (Naïve assumption).

· The algorithm calculates the probability of each class for a given input and assigns the class with the highest probability.

· It works well for text classification, spam detection, and sentiment analysis but struggles with correlated features.

1. Compare the working principles of Decision Tree and Naïve Bayes classifiers. Ans –

· Decision Tree:

1. Uses a hierarchical structure to make decisions based on feature values. 2. Works well for non-linear relationships and categorical/numerical data. 3. Can overfit but is interpretable.

· Naïve Bayes:

1. Uses probability theory and assumes feature independence.
2. Fast and works well for high-dimensional data (e.g., text classification). 3. Less interpretable and struggles with correlated features.

4. What are the different types of Decision Tree splitting criteria? Ans – · Gini Impurity – Measures how often a randomly chosen element would be incorrectly classified.

· Entropy (Information Gain) – Measures uncertainty in the dataset; higher gain means a better split.

· Chi-Square – Evaluates statistical significance of a split.

· Reduction in Variance – Used for regression trees to minimize variance within

splits.