DWM Exp. 3

Name – Sumeet Kedare
Class-Div/Roll - TE-3 / 31
Batch - B

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

Introduction:

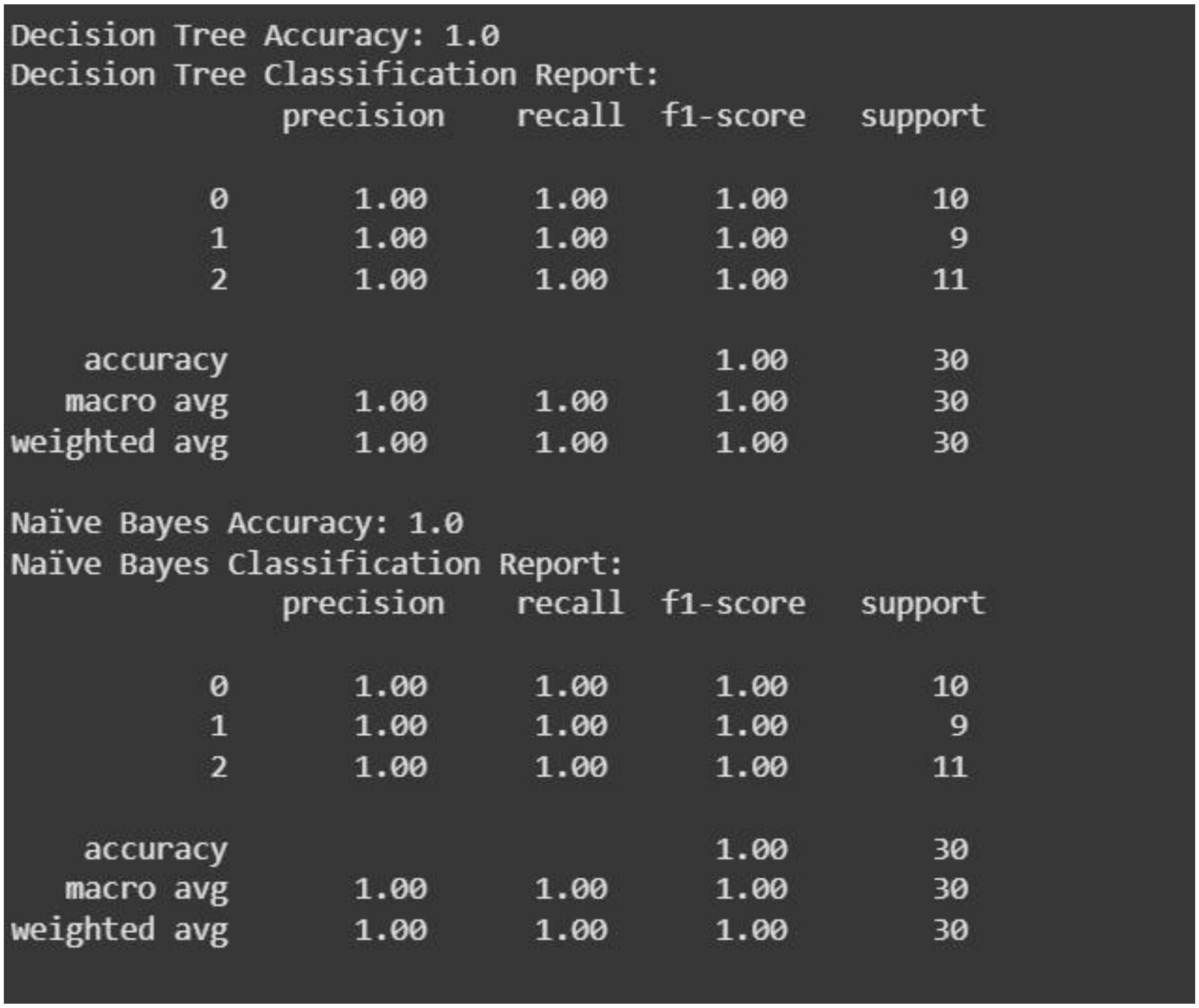
Classification algorithms are used in machine learning to categorize data into predefined classes. Decision Tree and Naïve Bayes are two popular classification techniques. Decision Trees use a hierarchical model of decisions based on feature values, while Naïve Bayes applies probabilistic principles based on Bayes' theorem assuming feature independence.

Procedure:

1. Load the dataset.
2. Preprocess and split the data into training and testing sets.
3. Standardize the features (optional for better performance).
4. Train a Decision Tree classifier and a Naïve Bayes classifier.
5. Evaluate the models using accuracy scores and classification reports.
6. Compare the results and analyze the findings.

Program Codes:

```
from sklearn.datasets import load_iris

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler  from
sklearn.tree import DecisionTreeClassifier  from
sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, classification_report


# Load dataset  data
= load_iris()

X, y = data.data, data.target


# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
# Standardize features (optional but can help some algorithms)  scaler =
StandardScaler()


X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Decision Tree Classifier  dt_model
= DecisionTreeClassifier()  dt_model.fit(X_train,
y_train)
dt_predictions = dt_model.predict(X_test)


# Naïve Bayes Classifier  nb_model =
GaussianNB()  nb_model.fit(X_train, y_train)
nb_predictions = nb_model.predict(X_test)

  # Evaluate
models
print("Decision Tree Accuracy:", accuracy_score(y_test, dt_predictions))
print("Decision Tree Classification Report:\n",
classification_report(y_test, dt_predictions))


print("Naïve Bayes Accuracy:", accuracy_score(y_test, nb_predictions))
print("Naïve Bayes Classification Report:\n",
classification_report(y_test, nb_predictions))
```



```
Decision Tree Accuracy: 1.0
Decision Tree 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

Naïve Bayes Accuracy: 1.0
Naïve Bayes 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: Both Decision Tree and Naïve Bayes classifiers are effective for classification tasks. Decision Trees provide an interpretable structure, whereas Naïve Bayes is computationally efficient and works well with small datasets. The choice between these depends on the dataset characteristics and requirements.

Review Questions:

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

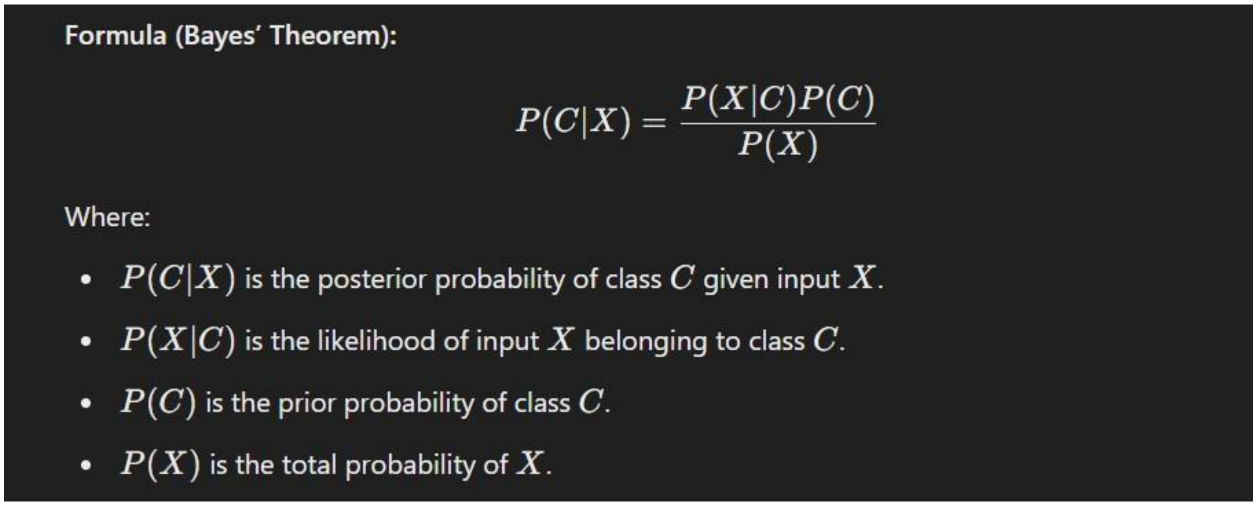
Ans. A **Decision Tree classifier** is a supervised learning algorithm used for classification and regression tasks. It works by splitting the dataset into smaller subsets based on feature values, forming a tree-like structure. The decision tree consists of:

- **Root Node**: Represents the entire dataset and the first feature split.
- **Internal Nodes**: Represent decision rules applied to split the data. ● **Leaf Nodes**: Represent the final class labels.

The tree uses **splitting criteria** such as **Gini impurity** or **Entropy (Information Gain)** to determine the best feature at each step. The algorithm continues splitting until a stopping condition is met, such as reaching a maximum depth or when further splits do not improve accuracy.

2.Explain the Naïve Bayes algorithm and its underlying assumptions.

Ans. The **Naïve Bayes classifier** is a probabilistic machine learning model based on **Bayes' theorem**, which calculates the probability of a class given a set of features. The algorithm is called "naïve" because it assumes that all features are **independent**, which is often not true in real-world data.



Formula (Bayes' Theorem):

$$P(C|X) = \frac{P(X|C)P(C)}{P(X)}$$

Where:

- $P(C|X)$ is the posterior probability of class C given input X.
- $P(X|C)$ is the likelihood of input X belonging to class C.
- $P(C)$ is the prior probability of class C.
- $P(X)$ is the total probability of X.

Types of Naïve Bayes classifiers:

- **Gaussian Naïve Bayes** (for continuous data, assumes normal distribution).
- **Multinomial Naïve Bayes** (for text classification and discrete data).
- **Bernoulli Naïve Bayes** (for binary/boolean features).

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

Aspect	**Decision Tree**	**Naïve Bayes**
Model Type	Non-parametric model	Probabilistic model
Working Principle	Recursively splits data based on attribute values	Uses Bayes' Theorem to calculate probabilities
Assumptions	No assumption about relationships between features	Assumes conditional independence among features
Structure	Tree-like structure with nodes, branches, and leaves	Probabilistic calculation using prior and likelihoods
Splitting Criterion	Gini impurity, Information Gain, Gain Ratio	Assumes independence of features
Interpretability	High - Easy to visualize trace decision-making process	Moderate - Less interpretable due to reliance on probabilities
Handling of		

Data	Can handle both numerical and categorical data; robust to outliers	Can handle both numerical and categorical data; sensitive to outliers
Prediction Method	Follows decision rules from root to leaf	Calculates posterior probability for each class
Complexity	Can become complex and overfit if not pruned	Simpler and faster to compute

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

Ans. Decision Tree algorithms use various splitting criteria to decide how to partition the data at each node. Here are the most common types of splitting criteria:

1. **Gini Impurity:** Measures the impurity of a dataset; the probability of randomly chosen elements being incorrectly classified if randomly labeled according to the distribution of labels in the dataset.
2. **Information Gain:** Measures the reduction in entropy or uncertainty in the data; calculates the difference in entropy before and after the split.
3. **Gain Ratio:** A modification of Information Gain that reduces its bias towards multi-valued attributes by normalizing the Information Gain using an intrinsic value.
4. **Chi-square:** Measures the statistical significance of the association between an attribute and the target variable using the Chi-square test.

Github link : https://github.com/Sarthak4730