Experiment No. 5

Title: Implement Naïve Bayes Classifier on data set of your choice. Test and Compare for Accuracy and Precision.

Objectives:

1. To implement a Naïve Bayes classifier for classifying the Iris dataset.
2. To evaluate the model's performance using accuracy, precision, and classification reports.
3. To understand how probabilistic classification works with Gaussian Naïve Bayes.

Problem Statement:

Classifying flowers into different species based on given attributes

Outcomes:

1. A trained Gaussian Naïve Bayes model that can accurately classify iris species.
2. Performance evaluation metrics including accuracy, precision, and a classification report.
3. A deeper understanding of Bayesian probability and how it applies to classification problems.

Tools Required: 4GB RAM, Anaconda, Notebook

Theory:

The Naïve Bayes classifier is based on Bayes' Theorem, which states:

$$P(A|B)=\frac{P(B|A)P(A)}{P(B)}$$

where:

- P(A|B) is the probability of class A given feature B.
- P(B|A) is the probability of feature BBB given class A.
- P(A) is the prior probability of class A.
- P(B) is the probability of feature B.

In Gaussian Naïve Bayes**,** features are assumed to follow a normal (Gaussian) distribution**:**

$$P(x|A) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

where:

- μ is the mean
- σ^2 is the variance of the feature values for a given class.

Algorithm:

1. Load the dataset: Import the Iris dataset.
2. Preprocess the data: Extract features (X) and labels (y).
3. Split the dataset: Use train_test_split() to divide data into training (80%) and testing (20%) sets.
4. Train the model:
    - Use GaussianNB() from sklearn.naive_bayes to train a Naïve Bayes model.
5. Make predictions:
    - Use predict() on the test set.
6. Evaluate performance:
    - Compute accuracy, precision, and a classification report.

Source Code:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, precision_score, classification_report
from sklearn.datasets import load_iris
iris = load_iris()
X = iris.data
y = iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
nb_classifier = GaussianNB()
nb_classifier.fit(X_train, y_train)
y_pred = nb_classifier.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='macro')
classification_rep = classification_report(y_test, y_pred, target_names=iris.target_names)
print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print("Classification Report:\n", classification_rep)
```

Output:

```
Accuracy: 1.00
Precision: 1.00
Classification Report:
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        10
  versicolor       1.00      1.00      1.00         9
   virginica       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:

_____

_____

_____

_____