EXPERIMENT-3

Student Name: Samuel **UID:24MAI10018**

Branch: CSE – AI & ML **Section: 24MAI - 1**

Semester: 2nd **Date of Conduct:**

Subject Name: - Machine Learning Lab **Subject Code: 24CSH-667**

AIM- Develop an application for implementing the Naive Bayes classifier.

Naïve Bayes

Naïve Bayes is a probabilistic machine learning algorithm based on Bayes' Theorem. It is commonly used for classification tasks such as spam detection, sentiment analysis, and medical diagnosis.

It assumes that features are independent (hence "naïve") and calculates the probability of a class given a set of features.

Bayes' Theorem Formula

$P(C \mid X)=P(X)P(X \mid C)P(C)$

Where:

$P(C \mid X)P(C|X)P(C \mid X)$ = Posterior probability (Probability of class CCC given data XXX)

$P(X \mid C)P(X|C)P(X \mid C)$ = Likelihood (Probability of data XXX given class CCC)

$P(C)P(C)P(C)$ = Prior probability (Probability of class CCC occurring)

$P(X)P(X)P(X)$ = Evidence (Probability of data XXX occurring)

Algorithm:

1. **Prepare the Dataset**
   - Collect and preprocess data with labeled features.
2. **Calculate Prior Probability**
   - Compute $P(C)P(C)P(C)$ for each class.
3. **Calculate Likelihood**
   - Compute $P(X \mid C)P(X|C)P(X \mid C)$ for each feature using a probability distribution (Gaussian, Bernoulli, or Multinomial).
4. **Apply Bayes' Theorem**
   - Compute $P(C \mid X)P(C|X)P(C \mid X)$ for each class and select the class with the highest probability.
5. **Make Predictions**
   - Classify new data points based on computed probabilities.
6. **Evaluate Model**
   - Use accuracy, precision, recall, and F1-score for performance assessment.

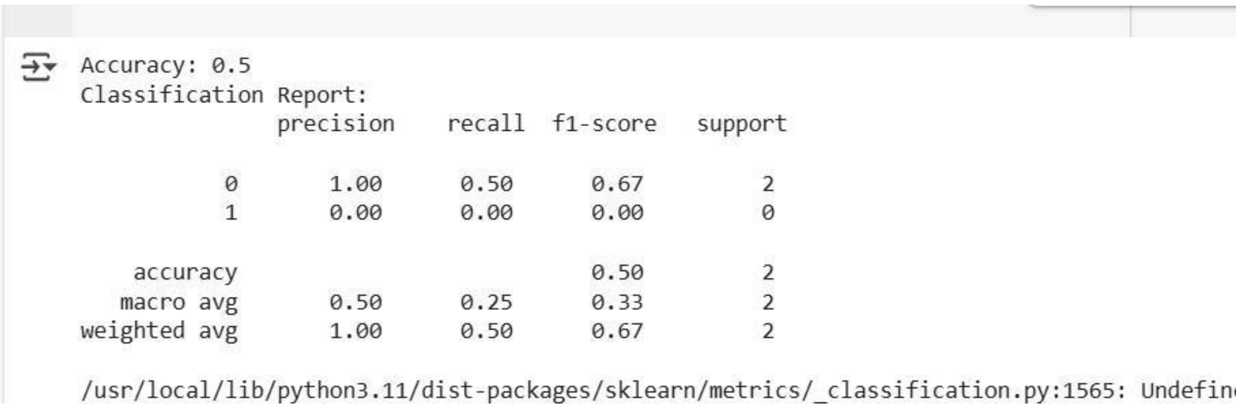
Types of Naïve Bayes Classifiers

1. Gaussian Naïve Bayes: Used for continuous data (assumes normal distribution).
2. Multinomial Naïve Bayes: Used for text classification (e.g., spam detection).
3. Bernoulli Naïve Bayes: Used for binary features (e.g., word presence in a document).

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, classification_report

# Sample
dataset data = {
   'Feature1': [2, 4, 5, 7, 9, 10, 11, 14, 16, 18],
   'Feature2': [1, 3, 5, 7, 9, 11, 13, 15, 17, 19],
   'Label': [0, 0, 0, 1, 1, 1, 1, 0, 0, 1]
}

df = pd.DataFrame(data)

# Splitting dataset into features (X) and
labels (Y) X = df[['Feature1', 'Feature2']]
Y = df['Label']

# Splitting dataset 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)

# Creating and training the Naive Bayes
model model = GaussianNB()
model.fit(X_train, Y_train)

# Making predictions
Y_pred = model.predict(X_test)

# Evaluating the model
accuracy = accuracy_score(Y_test, Y_pred)
report = classification_report(Y_test, Y_pred)

# Display results print("Accuracy:",
accuracy) print("Classification
Report:\n", report)
```

OUTPUT:



```
Accuracy: 0.5
Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.50      0.67         2
           1       0.00      0.00      0.00         0

    accuracy                           0.50         2
   macro avg       0.50      0.25      0.33         2
weighted avg       1.00      0.50      0.67         2

/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: Undefin
```

LEARNING OUTCOMES:

- Learn the fundamentals of the Naïve Bayes classifier, its assumptions, and its probabilistic nature.
- Understand Bayes' Theorem and its role in classification problems.
- Learn how to clean, transform, and prepare data for Naïve Bayes classification. Handle categorical and numerical data efficiently.
- Train a Naïve Bayes model on real-world datasets.
- Evaluate model performance using accuracy, precision, recall, and F1-score.