120A3051
Shreya Idate
Batch: E3

EXPERIMENT NO. 8

Aim : Classification modelling:

a) Choose classifier for classification problem (Naïve Bayes).
b) Evaluate the performance of classifier.

Theory :

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems. It is mainly used in text classification that includes a high-dimensional training dataset. Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions. It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.

Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:
•Naïve: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.

•Bayes: It is called Bayes because it depends on the principle of Bayes' Theorem. Bayes' Theorem:
Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.

The formula for Bayes' theorem is given as:
Naïve Bayes Classifier Algorithm where,

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

P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B

P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

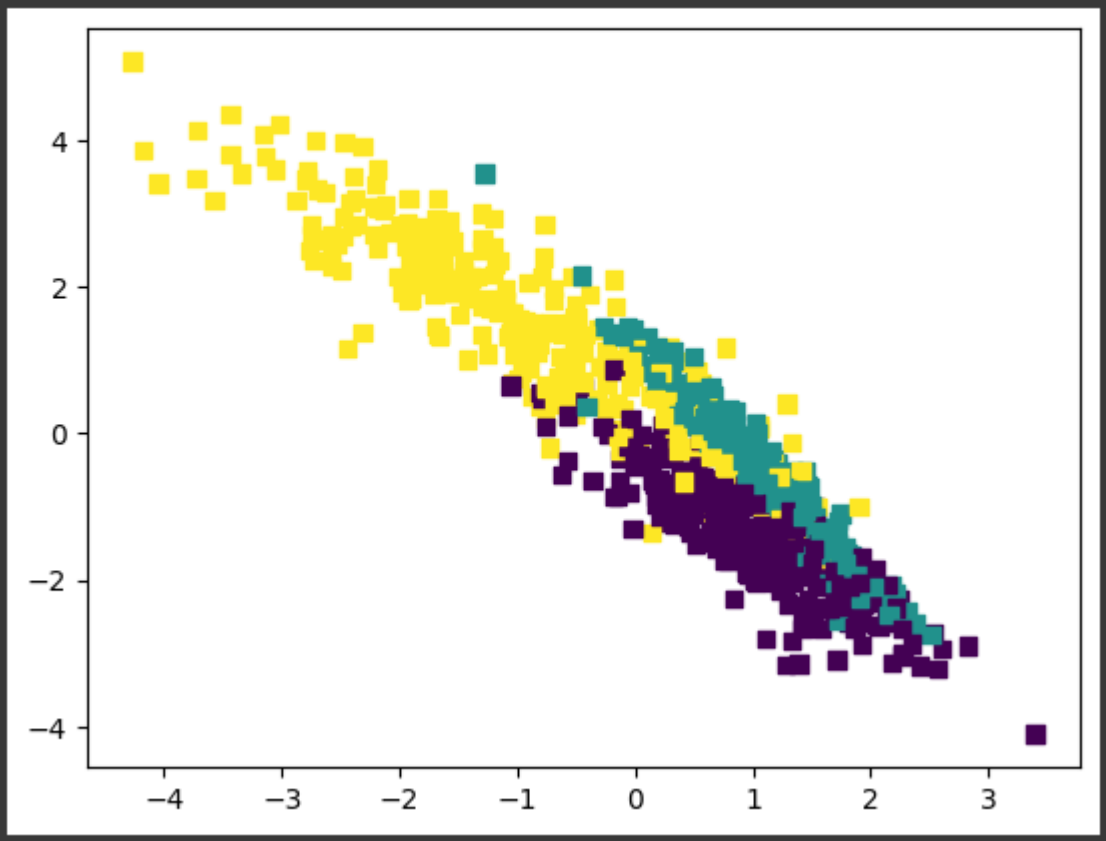
Program:

```
[1]  from sklearn.datasets import make_classification

     X, y = make_classification(
         n_features=6,
         n_classes=3,
         n_samples=800,
         n_informative=2,
         random_state=1,
         n_clusters_per_class=1,
     )
```

```
[2]  import matplotlib.pyplot as plt

     plt.scatter(X[:, 0], X[:, 1], c=y, marker=",");
```



```
[3]  from sklearn.model_selection import train_test_split

     X_train, X_test, y_train, y_test = train_test_split(
         X, y, test_size=0.33, random_state=125
     )
```

```
[8]  from sklearn.naive_bayes import GaussianNB

     # Build a Gaussian Classifier
     model = GaussianNB()

     # Model training
     model.fit(X_train, y_train)

     # Predict Output
     predicted = model.predict([X_test[2]])

     print("Actual Value:", y_test[2])
     print("Predicted Value:", predicted[0])
```

```
Actual Value: 2
Predicted Value: 2
```

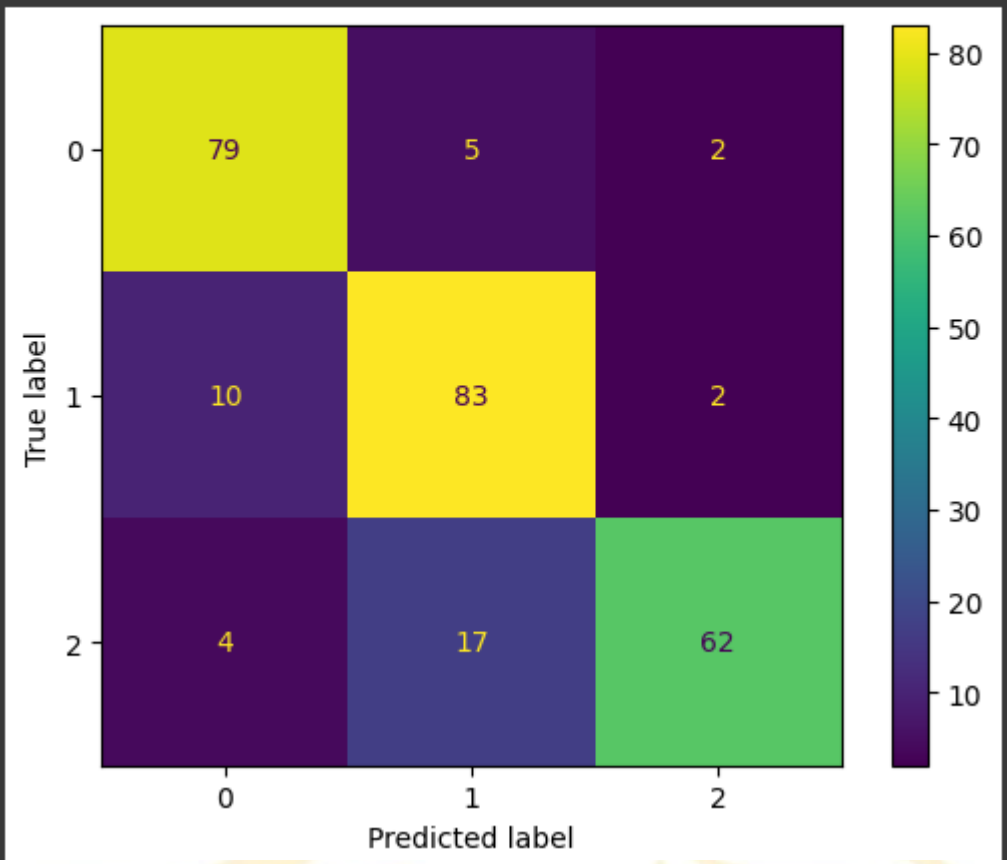
```
[9]  from sklearn.metrics import (
         accuracy_score,
         confusion_matrix,
         ConfusionMatrixDisplay,
         f1_score,
     )

     y_pred = model.predict(X_test)
     accuray = accuracy_score(y_pred, y_test)
     f1 = f1_score(y_pred, y_test, average="weighted")

     print("Accuracy:", accuray)
     print("F1 Score:", f1)
```

```
Accuracy: 0.8484848484848485
F1 Score: 0.8491119695890328
```

```
[10] labels = [0,1,2]
     cm = confusion_matrix(y_test, y_pred, labels=labels)
     disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=labels)
     disp.plot();
```




```
[11] y_pred = model.predict(X_test)
     y_pred
```

```
array([0, 1, 2, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 2, 2, 1, 1, 0, 0, 0, 0, 0,
       1, 0, 0, 2, 0, 2, 0, 2, 1, 2, 0, 0, 1, 0, 2, 2, 1, 0, 2, 1, 2, 0,
       1, 0, 1, 1, 0, 1, 2, 2, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 2, 0, 2, 1,
       0, 1, 2, 1, 0, 1, 2, 1, 1, 0, 2, 1, 0, 2, 1, 2, 0, 0, 0, 0, 0, 0,
       2, 2, 1, 1, 0, 1, 1, 2, 0, 0, 1, 2, 1, 1, 0, 1, 1, 1, 2, 1, 2, 0,
       1, 2, 2, 1, 1, 2, 1, 2, 0, 2, 2, 1, 2, 2, 1, 1, 0, 1, 0, 1, 2, 1,
       1, 2, 2, 1, 1, 0, 1, 2, 0, 0, 0, 1, 1, 1, 0, 2, 2, 0, 0, 1, 1, 1,
       0, 1, 2, 1, 2, 0, 1, 1, 1, 0, 2, 0, 2, 1, 0, 0, 1, 0, 0, 0, 0, 2,
       1, 1, 1, 2, 0, 1, 0, 0, 2, 1, 1, 2, 1, 1, 0, 0, 0, 1, 0, 2, 1, 1,
       0, 1, 1, 1, 0, 0, 1, 0, 0, 2, 1, 1, 1, 1, 2, 2, 1, 2, 0, 1, 2, 1,
       1, 0, 1, 2, 1, 2, 1, 0, 2, 1, 0, 1, 0, 2, 2, 2, 1, 0, 0, 0, 0, 2,
       0, 1, 0, 1, 1, 2, 2, 0, 1, 1, 0, 1, 0, 0, 1, 2, 2, 1, 2, 0, 1, 0])
```

```
[12] y_test

     array([0, 1, 2, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 2, 2, 1, 1, 0, 0, 0, 0, 0,
            2, 0, 2, 2, 0, 2, 0, 2, 1, 0, 0, 0, 1, 0, 2, 2, 1, 1, 2, 1, 2, 0,
            1, 0, 2, 1, 0, 1, 2, 1, 2, 1, 0, 1, 1, 1, 0, 1, 0, 1, 2, 0, 2, 1,
            0, 1, 2, 1, 0, 2, 2, 1, 1, 1, 2, 1, 0, 0, 1, 2, 0, 0, 0, 0, 0, 0,
            2, 2, 1, 1, 0, 1, 1, 2, 0, 1, 1, 2, 1, 1, 0, 1, 1, 0, 2, 2, 2, 0,
            1, 2, 2, 1, 2, 2, 1, 2, 0, 2, 2, 1, 1, 2, 0, 1, 0, 1, 2, 1, 2, 1,
            2, 2, 2, 1, 2, 1, 1, 2, 0, 0, 0, 1, 1, 1, 0, 2, 2, 0, 1, 1, 1, 1,
            0, 1, 2, 1, 2, 0, 1, 2, 1, 0, 2, 0, 2, 1, 0, 0, 0, 0, 0, 0, 0, 2,
            1, 1, 0, 2, 0, 1, 0, 0, 2, 1, 1, 2, 1, 1, 0, 0, 0, 1, 0, 2, 1, 1,
            0, 1, 1, 1, 0, 0, 1, 0, 0, 2, 2, 1, 2, 2, 2, 2, 1, 2, 0, 1, 2, 1,
            1, 1, 1, 2, 1, 2, 1, 0, 2, 1, 0, 0, 0, 2, 2, 2, 2, 1, 1, 0, 0, 2,
            0, 1, 0, 1, 2, 2, 2, 0, 2, 1, 0, 1, 2, 0, 2, 2, 2, 2, 2, 0, 2, 0])
```

Conclusion: Successfully performed Naïve Bayes classification.