In [1]:
```
# Import required libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, r
```

In [2]:
```
# Load the dataset
df = pd.read_csv('Iris.csv')
```

In [3]:
```
# Separate features and target
X = df.drop(columns=['Species'])  # Features
y = df['Species']                 # Target class
```

In [4]:
```
# Split 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_
```

In [5]:
```
# Initialize and train Naïve Bayes classifier
model = GaussianNB()
model.fit(X_train, y_train)
```

Out[5]:
```
GaussianNB()
```

In [6]:
```
# Predict using the model
y_pred = model.predict(X_test)

# Show predictions (optional)
print("Predicted Labels:\n", y_pred)
```

```
Predicted Labels:
 ['Iris-versicolor' 'Iris-setosa' 'Iris-virginica' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-setosa' 'Iris-versicolor' 'Iris-virginica'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-versicolor'
 'Iris-virginica' 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-setosa' 'Iris-setosa']
```

In [7]:
```
# Confusion Matrix
cm = confusion_matrix(y_test, y_pred, labels=model.classes_)
print("Confusion Matrix:\n", cm)
```

```
Confusion Matrix:
 [[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]
```

In [8]:
```
# Calculate evaluation metrics
accuracy = accuracy_score(y_test, y_pred)
error_rate = 1 - accuracy
precision = precision_score(y_test, y_pred, average='macro')  # average macro fo
recall = recall_score(y_test, y_pred, average='macro')
```

In [9]:
```
print(f"Accuracy      : {accuracy:.2f}")
print(f"Error Rate    : {error_rate:.2f}")
print(f"Precision     : {precision:.2f}")
print(f"Recall        : {recall:.2f}")
```

```
Accuracy      : 1.00
Error Rate    : 0.00
Precision     : 1.00
Recall        : 1.00
```

In [10]:
```
# Calculate TP, FP, FN, TN for each class
for i, label in enumerate(model.classes_):
    TP = cm[i, i]
    FP = cm[:, i].sum() - TP
    FN = cm[i, :].sum() - TP
    TN = cm.sum() - (TP + FP + FN)

    print(f"\nClass: {label}")
    print(f"True Positives (TP): {TP}")
    print(f"False Positives (FP): {FP}")
    print(f"False Negatives (FN): {FN}")
    print(f"True Negatives (TN): {TN}")
```

```
Class: Iris-setosa
True Positives (TP): 10
False Positives (FP): 0
False Negatives (FN): 0
True Negatives (TN): 20

Class: Iris-versicolor
True Positives (TP): 9
False Positives (FP): 0
False Negatives (FN): 0
True Negatives (TN): 21

Class: Iris-virginica
True Positives (TP): 11
False Positives (FP): 0
False Negatives (FN): 0
True Negatives (TN): 19
```