February 21, 2025

[1]:

*# Step 1: Import necessary 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,␣

𝗌recall\_score, f1\_score

[2]:

*# Step 2: Load the Iris dataset*

*# Using sklearn's built-in dataset for Iris*

**from sklearn.datasets import** load\_iris iris = load\_iris()

*# Convert to DataFrame for ease of use*

data = pd.DataFrame(data=iris.data, columns=iris.feature\_names) data['species'] = iris.target

*# Display first few rows of the dataset*

data.head()

1. : sepal length (cm) sepal width (cm) petal length (cm) petal width (cm) \

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 |  | 5.1 | 3.5 | 1.4 | 0.2 |
| 1 |  | 4.9 | 3.0 | 1.4 | 0.2 |
| 2 |  | 4.7 | 3.2 | 1.3 | 0.2 |
| 3 |  | 4.6 | 3.1 | 1.5 | 0.2 |
| 4 |  | 5.0 | 3.6 | 1.4 | 0.2 |
| 0 | species  0 |  | | | |
| 1 | 0 |
| 2 | 0 |
| 3 | 0 |
| 4 | 0 |

1. :

*# Step 3: Split the data into training and testing sets*

X = data[iris.feature\_names] *# Features*

y = data['species'] *# Target variable*

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3,␣

𝗌random\_state=42)

*# Display shapes of the training and test sets* print(f"Training data shape: **{**X\_train.shape**}**") print(f"Test data shape: **{**X\_test.shape**}**")

1. :

Training data shape: (105, 4)

Test data shape: (45, 4)

*# Step 4: Train the Naïve Bayes classifier* nb\_model = GaussianNB() nb\_model.fit(X\_train, y\_train)

*# Confirm model training*

print("Model trained successfully.")

Model trained successfully.

1. :

*# Step 5: Make predictions*

y\_pred = nb\_model.predict(X\_test)

*# Display first few predictions*

y\_pred[:10]

1. : array([1, 0, 2, 1, 1, 0, 1, 2, 1, 1])
2. :

*# Step 6: Compute the Confusion Matrix* cm = confusion\_matrix(y\_test, y\_pred) print("Confusion Matrix:")

print(cm)

Confusion Matrix: [[19 0 0]

[ 0 12 1]

[ 0 0 13]]

1. :

*# Step 7: Calculate additional metrics*

TP = cm[0][0] *# True Positive* FP = cm[0][1] *# False Positive* TN = cm[1][1] *# True Negative* FN = cm[1][0] *# False Negative*

*# Accuracy*

accuracy = accuracy\_score(y\_test, y\_pred)

*# Error rate*

error\_rate = 1 - accuracy

*# Precision*

precision = precision\_score(y\_test, y\_pred, average='weighted')

*# Recall*

recall = recall\_score(y\_test, y\_pred, average='weighted')

*# F1 Score (Optional but a good metric)*

f1 = f1\_score(y\_test, y\_pred, average='weighted')

*# Print metrics*

print(f"Accuracy: **{**accuracy**:**.4f**}**") print(f"Error Rate: **{**error\_rate**:**.4f**}**") print(f"Precision: **{**precision**:**.4f**}**") print(f"Recall: **{**recall**:**.4f**}**")

print(f"F1 Score: **{**f1**:**.4f**}**")

*# Output TP, FP, TN, FN*

print(f"TP: **{**TP**}**, FP: **{**FP**}**, TN: **{**TN**}**, FN: **{**FN**}**")

Accuracy: 0.9778

Error Rate: 0.0222

Precision: 0.9794

Recall: 0.9778

F1 Score: 0.9777

TP: 19, FP: 0, TN: 12, FN: 0