Naive Bayes Implementation

Author: Srikar Kalle
Student ID: C00313529

Change Log

SL No.	Change Category	Description	Duration (mins)	Difficulty (1-10)
1	Dataset Change	Changed from 'Adult Dataset' to 'Penguins Dataset'.	10	3
2	Feature Selection	Selected `bill_length_mm` and `bill_depth_mm` as features.	8	4
3	Target Encoding	Used `LabelEncoder` to encode the species column.	5	3
4	Data Preprocessing	Standardized features using `StandardScaler`.	12	5
5	Train-Test Split	Applied an **80-20 split** for training and testing.	7	4
6	Model Implementation	Implemented **Naïve Bayes (`GaussianNB`)** for classification.	15	6
7	Model Evaluation	Used **accuracy, classification report** for evaluation.	10	5

Summary

- Transitioned from Kaggle's Adult dataset to Seaborn's Penguins dataset.
- Implemented a Naïve Bayes classification model.
- Improved data preprocessing with scaling.

Performance Metrics

- Model Accuracy: `0.9254` (Generated during execution)

In [1]:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from datetime import datetime
```

In [2]:

```python
# Step 2: Data Understanding
df = sns.load_dataset("penguins").dropna()
```

In [3]:

```python
# Selecting features and target
selected_features = ["bill_length_mm", "bill_depth_mm"]
```

```
X = df[selected_features]
y = LabelEncoder().fit_transform(df["species"])  # Encoding categorical target
```

In [4]:

```
# Step 3: Data Preparation
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=
```

In [5]:

```
# Step 4: Modeling
nb_model = GaussianNB()
nb_model.fit(X_train, y_train)
```

Out[5]:

```
▾ GaussianNB

GaussianNB()
```

In [6]:

```
# Step 5: Evaluation
y_pred = nb_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Model Accuracy: {accuracy:.4f}")
print(classification_report(y_test, y_pred))
```

```
Model Accuracy: 0.9254
              precision    recall  f1-score   support

           0       0.97      1.00      0.98        31
           1       0.79      0.85      0.81        13
           2       0.95      0.87      0.91        23

    accuracy                           0.93        67
   macro avg       0.90      0.91      0.90        67
weighted avg       0.93      0.93      0.93        67
```