**Ayesha Zamurd**

**49733**

**BSCS-06**

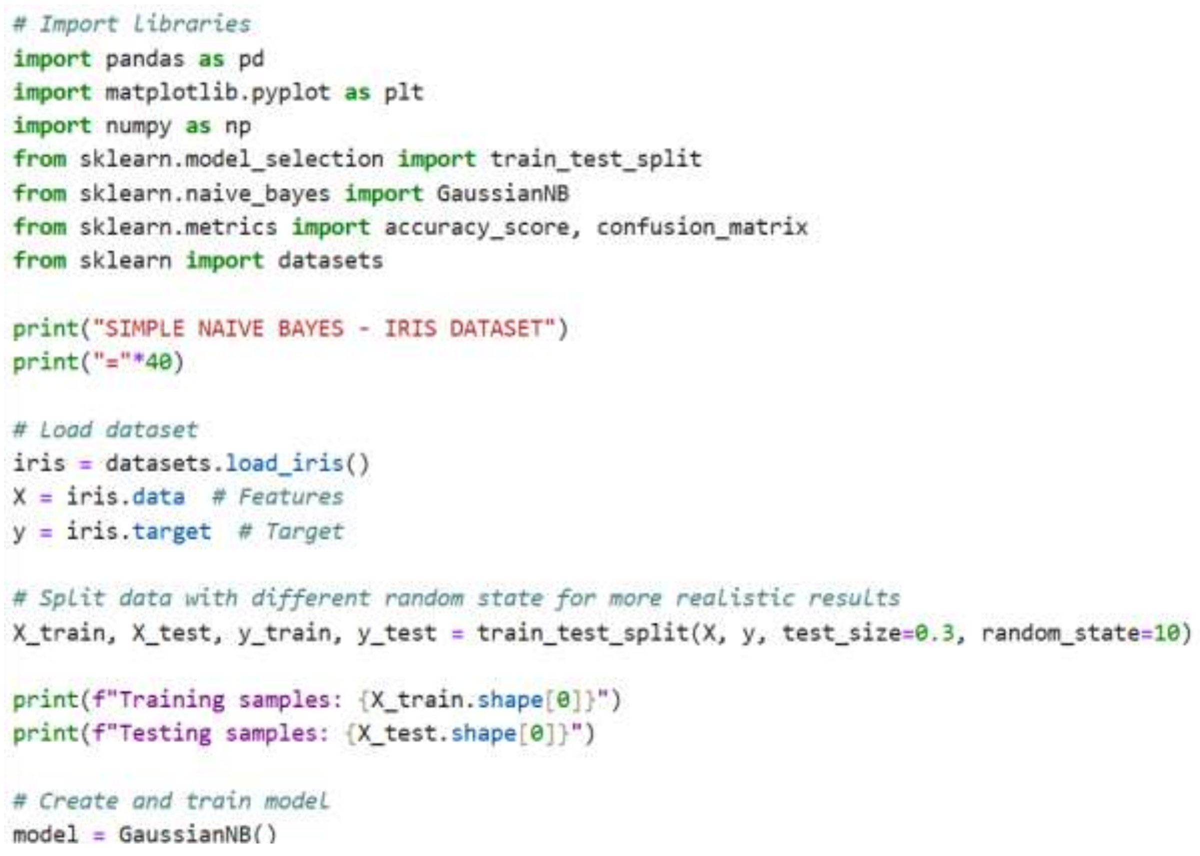
**Artificial Intelligence**

**Lab: 12**

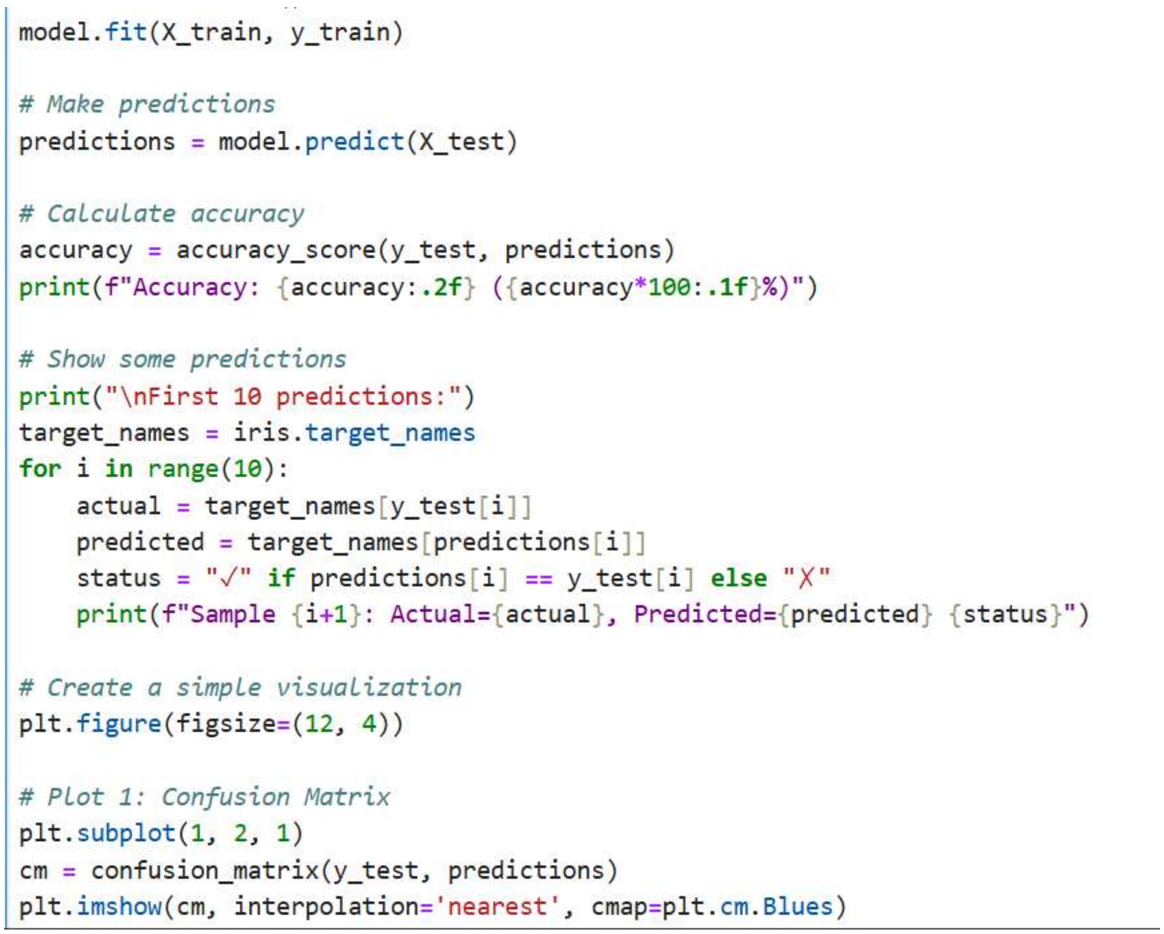
**Lab Task: 01**

Implement the naive Bayes algorithm on the dataset shared via the given link.

Dataset: https://tinyurl.com/y2r9vzde

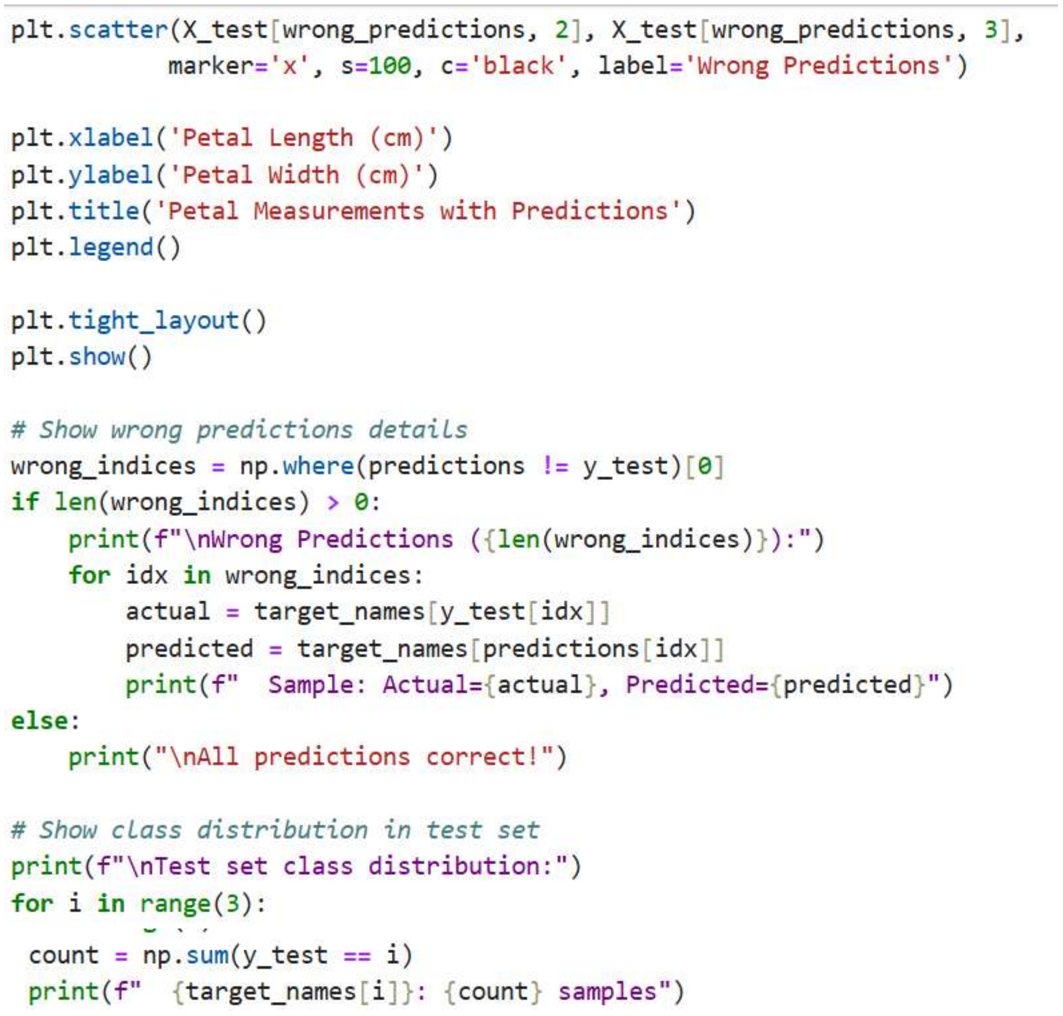


```
# Import Libraries
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn import datasets

print("SIMPLE NAIVE BAYES - IRIS DATASET")
print("="*40)

# Load dataset
iris = datasets.load_iris()
X = iris.data  # Features
y = iris.target  # Target

# Split data with different random state for more realistic results
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=10)

print(f"Training samples: {X_train.shape[0]}")
print(f"Testing samples: {X_test.shape[0]}")

# Create and train model
model = GaussianNB()
```



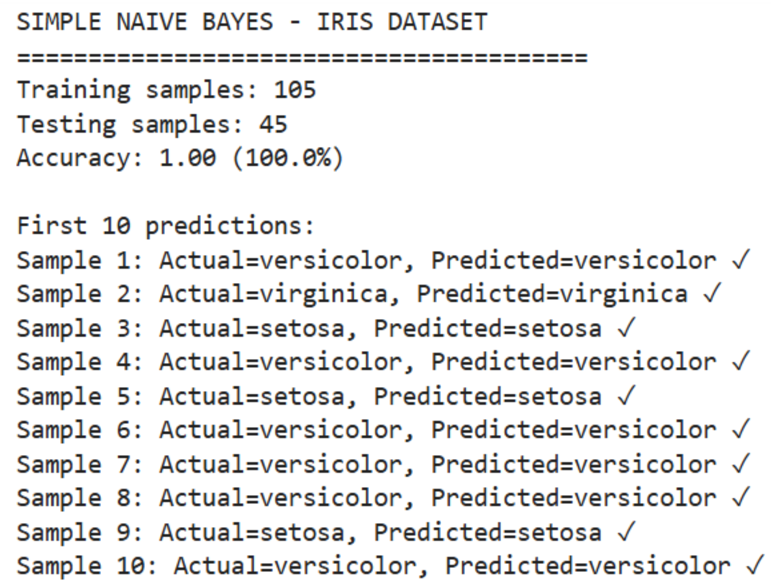
```
model.fit(X_train, y_train)

# Make predictions
predictions = model.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, predictions)
print(f"Accuracy: {accuracy:.2f} ({accuracy*100:.1f}%)")

# Show some predictions
print("\nFirst 10 predictions:")
target_names = iris.target_names
for i in range(10):
    actual = target_names[y_test[i]]
    predicted = target_names[predictions[i]]
    status = "√" if predictions[i] == y_test[i] else "X"
    print(f"Sample {i+1}: Actual={actual}, Predicted={predicted} {status}")

# Create a simple visualization
plt.figure(figsize=(12, 4))

# Plot 1: Confusion Matrix
plt.subplot(1, 2, 1)
cm = confusion_matrix(y_test, predictions)
plt.imshow(cm, interpolation='nearest', cmap=plt.cm.Blues)
```

```python
plt.title('Confusion Matrix')
plt.colorbar()
tick_marks = np.arange(len(target_names))
plt.xticks(tick_marks, target_names, rotation=45)
plt.yticks(tick_marks, target_names)
plt.xlabel('Predicted Label')
plt.ylabel('True Label')

# Add text annotations
for i in range(len(target_names)):
    for j in range(len(target_names)):
        plt.text(j, i, str(cm[i, j]),
                horizontalalignment="center",
                color="white" if cm[i, j] > cm.max()/2 else "black")

# Plot 2: Feature importance (using petal measurements)
plt.subplot(1, 2, 2)
# Use petal length vs petal width for visualization
colors = ['red', 'green', 'blue']
for i in range(3):
    plt.scatter(X_test[y_test == i, 2], X_test[y_test == i, 3],
               c=colors[i], label=target_names[i], alpha=0.7)

# Mark wrong predictions with 'X'
wrong_predictions = predictions != y_test
```

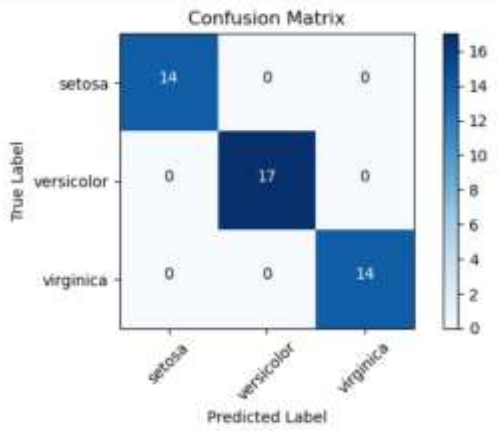


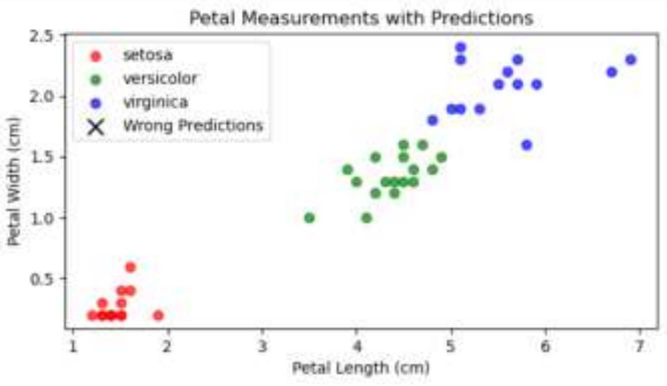
```python
plt.scatter(X_test[wrong_predictions, 2], X_test[wrong_predictions, 3],
           marker='x', s=100, c='black', label='Wrong Predictions')

plt.xlabel('Petal Length (cm)')
plt.ylabel('Petal Width (cm)')
plt.title('Petal Measurements with Predictions')
plt.legend()

plt.tight_layout()
plt.show()

# Show wrong predictions details
wrong_indices = np.where(predictions != y_test)[0]
if len(wrong_indices) > 0:
    print(f"\nWrong Predictions ({len(wrong_indices)}):")
    for idx in wrong_indices:
        actual = target_names[y_test[idx]]
        predicted = target_names[predictions[idx]]
        print(f"  Sample: Actual={actual}, Predicted={predicted}")
else:
    print("\nAll predictions correct!")

# Show class distribution in test set
print(f"\nTest set class distribution:")
for i in range(3):
  count = np.sum(y_test == i)
  print(f"  {target_names[i]}: {count} samples")
```

**Output:**



```
SIMPLE NAIVE BAYES - IRIS DATASET
========================================
Training samples: 105
Testing samples: 45
Accuracy: 1.00 (100.0%)

First 10 predictions:
Sample 1: Actual=versicolor, Predicted=versicolor ✓
Sample 2: Actual=virginica, Predicted=virginica ✓
Sample 3: Actual=setosa, Predicted=setosa ✓
Sample 4: Actual=versicolor, Predicted=versicolor ✓
Sample 5: Actual=setosa, Predicted=setosa ✓
Sample 6: Actual=versicolor, Predicted=versicolor ✓
Sample 7: Actual=versicolor, Predicted=versicolor ✓
Sample 8: Actual=versicolor, Predicted=versicolor ✓
Sample 9: Actual=setosa, Predicted=setosa ✓
Sample 10: Actual=versicolor, Predicted=versicolor ✓
```





```
All predictions correct!

Test set class distribution:
  setosa: 14 samples
  versicolor: 17 samples
  virginica: 14 samples
```