Code and Output:

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt

# Sample data for binary classification
X = np.array([[1, 2], [2, 3], [3, 4], [4, 5], [5, 6], [6, 7], [7, 8], [8, 9], [9, 10], [10, 11],
              [1, 10], [2, 9], [3, 8], [4, 7], [5, 12], [6, 11], [7, 10], [8, 13], [9, 12], [10, 14],
              [15, 2], [16, 3], [17, 4], [18, 5], [19, 6], [20, 7], [21, 8], [22, 9], [23, 10], [24, 11]])
y = np.array([0, 0, 0, 0, 1, 1, 1, 1, 1, 1,
              0, 0, 0, 0, 1, 1, 1, 1, 1, 1,
              0, 0, 0, 0, 1, 1, 1, 1, 1, 1])

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

+ Code + Text

Create Naive Bayes classifier and train it

```
# Create Naive Bayes classifier and train it
nb = GaussianNB()
nb.fit(X_train, y_train)
```

```
GaussianNB
GaussianNB()
```

Make predictions & Evaluate the classifier's performance

```
# Make predictions
predictions = nb.predict(X_test)

# Evaluate the classifier's performance
accuracy = accuracy_score(y_test, predictions)
print(f"Accuracy: {accuracy * 100:.2f}%")
```

```
Accuracy: 83.33%
```

Plotting the decision boundary,Get predictions for every point in the meshgrid,Plot decision boundary and data points

```
# Plotting the decision boundary
h = .02  # Step size in the meshgrid
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
```

```
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))

# Get predictions for every point in the meshgrid
Z = nb.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)


# Plot decision boundary and data points
plt.contourf(xx, yy, Z, alpha=0.75, cmap='coolwarm', label='Decision Boundary')  # Added label here
plt.scatter(X[:, 0], X[:, 1], c=y, edgecolors='k', marker='o', s=100, label='Data Points')  # Added label for data points

# Highlight test data points with different color
plt.scatter(X_test[:, 0], X_test[:, 1], c='black', marker='x', s=100, label='Test Data')  # Added label for test data

# Title and labels
plt.title('Naive Bayes Decision Boundary')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')

# Add legend with labels for all the plot elements
plt.legend()

plt.show()
```

```
<ipython-input-24-8292e0f0f593>:2: UserWarning: The following kwargs were not used by contour: 'label'
  plt.contourf(xx, yy, Z, alpha=0.75, cmap='coolwarm', label='Decision Boundary')  # Added label here
```