NAIVE BAYES

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from matplotlib.colors import ListedColormap

# Load Iris dataset
iris = load_iris()
X = iris.data[:, :2]  # Using only the first two features: sepal length and width
y = iris.target

# Split into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)

# Scale features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Train Naive Bayes classifier
classifier = GaussianNB()
classifier.fit(X_train, y_train)

# Function to visualize decision regions
def visualize_results(X_set, y_set, title, colors=('red', 'green', 'blue')):
    X1, X2 = np.meshgrid(np.arange(X_set[:, 0].min() - 1, X_set[:, 0].max() + 1, 0.01),
                         np.arange(X_set[:, 1].min() - 1, X_set[:, 1].max() + 1, 0.01))
    plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
                 alpha=0.75, cmap=ListedColormap(colors))
```

```
    plt.xlim(X1.min(), X1.max())
    plt.ylim(X2.min(), X2.max())
    scatter = [plt.scatter(X_set[y_set == i, 0], X_set[y_set == i, 1], c=color, label=iris.target_names[i])
               for i, color in enumerate(colors)]
    plt.title(title)
    plt.xlabel('Sepal Length')
    plt.ylabel('Sepal Width')
    plt.legend()
    plt.show()

# Visualize training results
visualize_results(X_train, y_train, 'Naive Bayes (Training Set)')

# Visualize test results
visualize_results(X_test, y_test, 'Naive Bayes (Test Set)')
```