```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score
```

```
# Set random seed for reproducibility
np.random.seed(42)

# Number of samples
num_samples = 150

# Generate random synthetic features
sepal_length = np.random.uniform(4.3, 7.9, num_samples)
sepal_width = np.random.uniform(2.0, 4.4, num_samples)
petal_length = np.random.uniform(1.0, 6.9, num_samples)
petal_width = np.random.uniform(0.1, 2.5, num_samples)

# Generate class labels (3 species)
species = np.random.choice(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], num_samples)

# Create DataFrame
df = pd.DataFrame({
    'SepalLength': sepal_length,
    'SepalWidth': sepal_width,
    'PetalLength': petal_length,
    'PetalWidth': petal_width,
    'Species': species
})

# Display first few rows
df.head()
```

	SepalLength	SepalWidth	PetalLength	PetalWidth	Species
0	5.648344	4.179838	1.304922	1.965153	Iris-setosa
1	7.722572	2.574949	4.134992	1.440170	Iris-versicolor
2	6.935178	2.347748	4.189747	1.118133	Iris-versicolor
3	6.455171	3.174687	4.760836	2.275251	Iris-setosa
4	4.861667	4.365561	5.283939	0.366874	Iris-versicolor

```
# Define features (X) and target (y)
X = df.drop(columns=['Species'])
y = df['Species']

# Split into training and testing sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Initialize the model
model = GaussianNB()

# Train the model
model.fit(X_train, y_train)
```

```
GaussianNB()
```

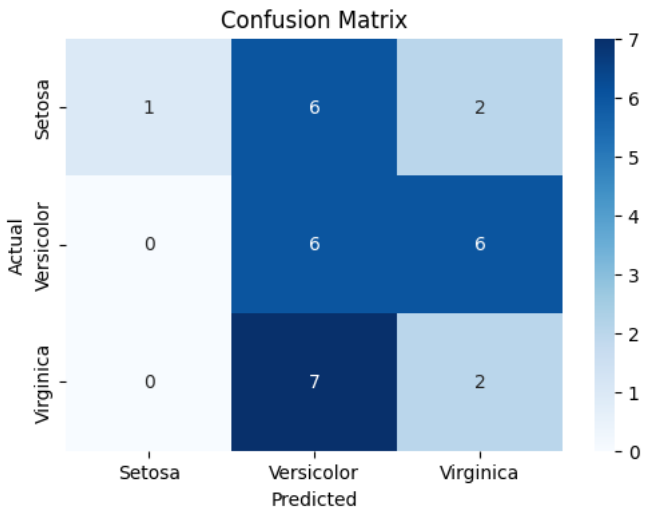
```
# Predict class labels
y_pred = model.predict(X_test)
```

```
# Compute confusion matrix
cm = confusion_matrix(y_test, y_pred, labels=['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'])

# Compute accuracy, error rate, precision, and recall
accuracy = accuracy_score(y_test, y_pred)
error_rate = 1 - accuracy
precision = precision_score(y_test, y_pred, average='macro')  # Macro because it's multi-class
```

```
recall = recall_score(y_test, y_pred, average='macro')

# Print results
print(f"Confusion Matrix:\n{cm}")
print(f"Accuracy: {accuracy:.2f}")
print(f"Error Rate: {error_rate:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
```

```
Confusion Matrix:
[[1 6 2]
 [0 6 6]
 [0 7 2]]
Accuracy: 0.30
Error Rate: 0.70
Precision: 0.51
Recall: 0.28
```

```
# Plot confusion matrix
plt.figure(figsize=(6,4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Setosa', 'Versicolor', 'Virginica'], yticklabels=['Setosa', 'Versicolor
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```




```
df.to_csv("synthetic_iris.csv", index=False)
print("Dataset saved as synthetic_iris.csv")
```

```
Dataset saved as synthetic_iris.csv
```