In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [25]:

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.metrics import precision_score, accuracy_score, confusion_matrix, clas
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import precision_score, accuracy_score, recall_score, confusio
from sklearn.naive_bayes import GaussianNB

# Read the CSV file
try:
    df = pd.read_csv("C:\\Users\\System21\\Desktop\\iris.csv", encoding='utf-8-sig'
    print("CSV file successfully loaded!")
except Exception as e:
    print(f"Error reading the CSV file: {e}")
    exit()  # Exit the script if the file can't be read

# Check column names for any spaces or hidden characters
print("Columns in the DataFrame:", df.columns)

# Strip any leading/trailing spaces in column names if needed
df.columns = df.columns.str.strip()

# Check if 'species' column exists
if 'species' in df.columns:
    print("'species' column found!")
else:
    print("'species' column not found in the DataFrame.")
    exit()  # Exit if 'species' column is missing

# Split the data into features (X) and target (y)
X = df.drop(columns=['species'])  # Features (drop the 'species' column)
y = df['species']  # Target (the 'species' column)

# Check if data is loaded properly
print("First few rows of the features (X):")
print(X.head())
print("First few rows of the target (y):")
print(y.head())

# Split the dataset into training and testing sets (80% train, 20% test)
try:
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random
    print("Data successfully split into training and testing sets!")

    # Print the shapes of the resulting datasets to verify the split
    print("Training features shape:", X_train.shape)
    print("Test features shape:", X_test.shape)
    print("Training target shape:", y_train.shape)
    print("Test target shape:", y_test.shape)
except Exception as e:
    print(f"Error splitting the data: {e}")

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

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model's performance
try:
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred, average='weighted')  # Using 'weigh
    recall = recall_score(y_test, y_pred, average='micro')  # Using 'micro' average
    conf_matrix = confusion_matrix(y_test, y_pred)
    class_report = classification_report(y_test, y_pred)

    # Print the evaluation results
    print(f"Accuracy: {accuracy}")
    print(f"Precision: {precision}")
    print(f"Recall: {recall}")
    print("Confusion Matrix:")
    print(conf_matrix)
    print("Classification Report:")
    print(class_report)
except Exception as e:
    print(f"Error in evaluation: {e}")
```

```
CSV file successfully loaded!
Columns in the DataFrame: Index(['sepal_length', 'sepal_width', 'petal_length', 'pet
al_width',
       'species'],
      dtype='object')
'species' column found!
First few rows of the features (X):
   sepal_length  sepal_width  petal_length  petal_width
0           5.1          3.5           1.4          0.2
1           4.9          3.0           1.4          0.2
2           4.7          3.2           1.3          0.2
3           4.6          3.1           1.5          0.2
4           5.0          3.6           1.4          0.2
First few rows of the target (y):
0    setosa
1    setosa
2    setosa
3    setosa
4    setosa
Name: species, dtype: object
Data successfully split into training and testing sets!
Training features shape: (120, 4)
Test features shape: (30, 4)
Training target shape: (120,)
Test target shape: (30,)
Accuracy: 1.0
Precision: 1.0
Recall: 1.0
Confusion Matrix:
[[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]
Classification Report:
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        10
  versicolor       1.00      1.00      1.00         9
   virginica       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

In [4]:

```
df = pd.read_csv("C:\\Users\\System21\\Desktop\\iris.csv", encoding='utf-8-sig')

print(df.head())
```

```
   sepal_length  sepal_width  petal_length  petal_width species
0           5.1          3.5           1.4          0.2  setosa
1           4.9          3.0           1.4          0.2  setosa
2           4.7          3.2           1.3          0.2  setosa
3           4.6          3.1           1.5          0.2  setosa
4           5.0          3.6           1.4          0.2  setosa
```

In [5]:

```
if df.select_dtypes(include=['object']).shape[1] > 0:
 df = pd.get_dummies(df, drop_first=True)
```

In [6]:

```
df.dropna(inplace=True)
```

In [8]:

```
import pandas as pd

# Read the CSV file
df = pd.read_csv("C:\\Users\\System21\\Desktop\\iris.csv", encoding='utf-8-sig')

# Check column names for any spaces or hidden characters
print("Columns in the DataFrame:", df.columns)

# Strip any leading/trailing spaces in column names
df.columns = df.columns.str.strip()

# Split the data into features (X) and target (y)
X = df.drop(columns=['species'])  # Assuming 'species' is the target variable
y = df['species']

# Print the first few rows to confirm the data
print("Features (X):")
print(X.head())
print("Target (y):")
print(y.head())
```

```
Columns in the DataFrame: Index(['sepal_length', 'sepal_width', 'petal_length', 'pet
al_width',
       'species'],
      dtype='object')
Features (X):
   sepal_length  sepal_width  petal_length  petal_width
0           5.1          3.5           1.4          0.2
1           4.9          3.0           1.4          0.2
2           4.7          3.2           1.3          0.2
3           4.6          3.1           1.5          0.2
4           5.0          3.6           1.4          0.2
Target (y):
0    setosa
1    setosa
2    setosa
3    setosa
4    setosa
Name: species, dtype: object
```

In [11]:

```
import pandas as pd
from sklearn.model_selection import train_test_split

# Read the CSV file
df = pd.read_csv("C:\\Users\\System21\\Desktop\\iris.csv", encoding='utf-8-sig')

# Check column names for any spaces or hidden characters
print("Columns in the DataFrame:", df.columns)

# Strip any leading/trailing spaces in column names
```

```
df.columns = df.columns.str.strip()

# Split the data into features (X) and target (y)
X = df.drop(columns=['species'])  # Assuming 'species' is the target variable
y = df['species']

# Split the dataset 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_sta

# Print the shapes of the resulting datasets to verify the split
print("Training features shape:", X_train.shape)
print("Test features shape:", X_test.shape)
print("Training target shape:", y_train.shape)
print("Test target shape:", y_test.shape)
```

```
Columns in the DataFrame: Index(['sepal_length', 'sepal_width', 'petal_length', 'pet
al_width',
       'species'],
      dtype='object')
Training features shape: (120, 4)
Test features shape: (30, 4)
Training target shape: (120,)
Test target shape: (30,)
```

In [12]:
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [13]:
```
gaussian = GaussianNB()
gaussian.fit(X_train, y_train)
```

Out[13]:
```
▾ GaussianNB

GaussianNB()
```

In [14]:
```
y_pred = gaussian.predict(X_test)
```

In [15]:
```
accuracy = accuracy_score(y_test, y_pred)
```

In [16]:
```
print("Accuracy:", accuracy)
```

```
Accuracy: 1.0
```

In [22]:
```
precision = precision_score(y_test, y_pred, average='weighted')  # Using 'weighted'
```

In [23]:
```
print("Precision:", precision)
```

```
Precision: 1.0
```

In [26]:
```
recall = recall_score(y_test, y_pred, average='micro')
```

In [27]:
```
print("Recall:", recall)
```

```
Recall: 1.0
```

In [28]:
```
conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", conf_matrix)
```

```
Confusion Matrix:
 [[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]
```

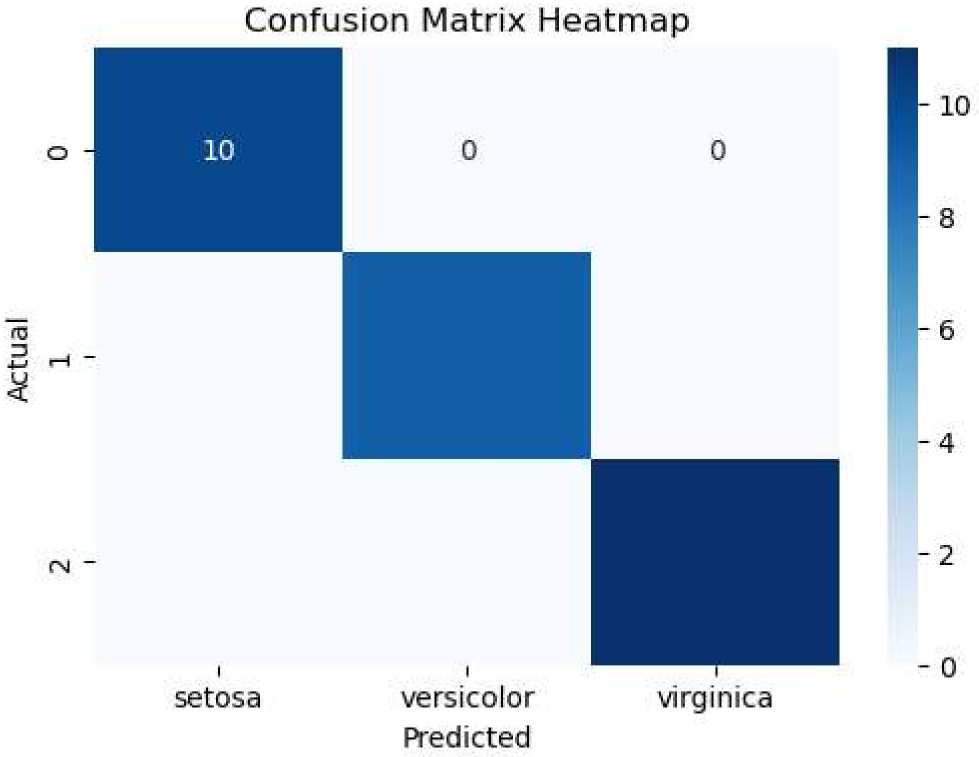
In [29]:
```
class_report = classification_report(y_test, y_pred)
print("Classification Report:\n", class_report)
```

```
Classification Report:
               precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        10
  versicolor       1.00      1.00      1.00         9
   virginica       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

In [33]:
```
plt.figure(figsize=(6,4))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=np.unique(y
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix Heatmap')
plt.show()
```



In [ ]:
```
                                Name:Sharvari Patil
                                Roll No:13265
```

```
Batch:B3
```