In [ ]:
```
                                                    Assignment No: 6
Contents for Theory:
1. Concepts used in Naïve Bayes classifier
2. Naive Bayes Example
3. Confusion Matrix Evaluation Metrics
```

In [23]:
```
#step1:Import libraries and create alias for Pandas, Numpy and Matplotlib
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [24]:
```
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, pre
```

In [25]:
```
#step2:Import the Iris dataset
df = pd.read_csv("C:\\Users\\Welcome\\Downloads\\iris dataset\\iris.csv")
```

In [26]:
```
# Step 3: Initialize the DataFrame

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 [22]:
```
# Step 4: Data Preprocessing
if df.select_dtypes(include=['object']).shape[1] > 0:
    df = pd.get_dummies(df, drop_first=True)
```

In [18]:
```
df.dropna(inplace=True)
```

In [27]:
```
X = df.drop(columns=['species'])  # Assuming 'species' is the target variable
y = df['species']
```

In [28]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42
```

In [29]:
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [30]:
```
gaussian = GaussianNB()
gaussian.fit(X_train, y_train)
```

Out[30]:
```
▾ GaussianNB ⓘ ?
(https://scikit-learn.org/1.4/modules/generated/sklearn.naive_bayes.GaussianNB.html)
GaussianNB()
```

In [31]:
```
y_pred = gaussian.predict(X_test)
```

In [32]:
```
accuracy = accuracy_score(y_test, y_pred)
```

In [33]:
```
print("Accuracy:", accuracy)
```

```
Accuracy: 1.0
```

In [34]:
```
precision = precision_score(y_test, y_pred, average='micro')
```

In [35]:
```
print("Precision:", precision)
```

```
Precision: 1.0
```

In [36]:
```
recall = recall_score(y_test, y_pred, average='micro')
```

In [37]:
```
print("Recall:", recall)
```

```
Recall: 1.0
```

In [38]:
```
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 [39]:
```
class_report = classification_report(y_test, y_pred)
print("Classification Report:\n", class_report)
```

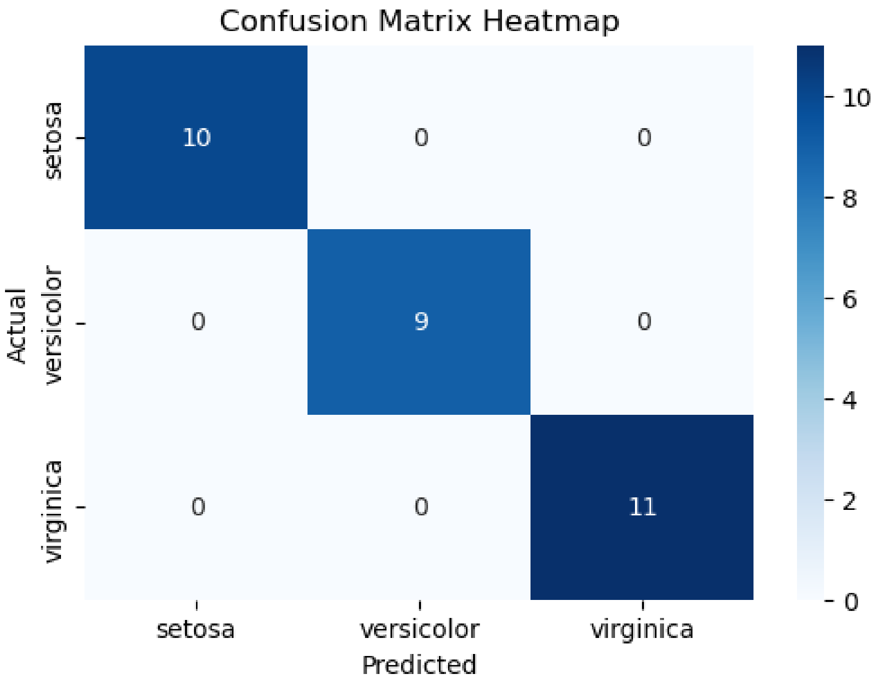
```
Classification Report:
               precision    recall  f1-score   support

      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 [40]:

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



In [ ]:

```
                                name:Neha Jahdav
                                roll no: 13247
                                batch: B3
```