

```

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

df = pd.read_csv("C:/Users/Admin/Downloads/iris1.csv")

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

```

df.tail()

```

	sepal_length	sepal_width	petal_length	petal_width	species
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

```

df.shape

(150, 5)

df.info()

```

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
#   Column          Non-Null Count  Dtype
---  -
0   sepal_length    150 non-null   float64
1   sepal_width     150 non-null   float64
2   petal_length    150 non-null   float64
3   petal_width     150 non-null   float64
4   species         150 non-null   object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB

```

```

encoder = LabelEncoder()
df['species'] = encoder.fit_transform(df['species'])

```

```
print("\nMissing values in the dataset:")
print(df.isnull().sum())
```

Missing values in the dataset:

```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

```
X = df.drop('species', axis=1)
Y = df['species']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, Y,
test_size=0.2, random_state=42)
```

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train) # Fit and transform on
training set
X_test_scaled = scaler.transform(X_test)
```

```
naive_bayes_model = GaussianNB()
naive_bayes_model.fit(X_train_scaled, y_train)
```

```
GaussianNB()
```

```
y_pred = naive_bayes_model.predict(X_test_scaled)
```

```
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='micro')
recall = recall_score(y_test, y_pred, average='micro')
conf_matrix = confusion_matrix(y_test, y_pred)
```

```
print("\nModel Evaluation:")
print(f'Accuracy: {accuracy}')
print(f'Precision: {precision}')
print(f'Recall: {recall}')
print(f'Confusion Matrix:\n{conf_matrix}')
```

Model Evaluation:

Accuracy: 1.0

Precision: 1.0

Recall: 1.0

Confusion Matrix:

```
[[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]
```

```
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=encoder.classes_, yticklabels=encoder.classes_)
plt.title('Confusion Matrix Heatmap')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.show()
```

