

Practical No 6

Data Analytics III

Implement Simple Naïve Bayes classification algorithm using Python/R on iris.csv dataset.

Compute Confusion matrix to find TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall on the given dataset.

```
In [1]: 1 import pandas as pd
        2 from sklearn.model_selection import train_test_split
        3 from sklearn.naive_bayes import GaussianNB
        4 from sklearn.metrics import confusion_matrix, accuracy_score, precision_score
```

```
In [2]: 1 df = pd.read_csv('Desktop\iris-data.csv')
        2 df
```

```
Out[2]:
```

	sepal length	sepal width	petal length	petal width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

In [21]:

```
1 df.describe()
```

Out[21]:

	sepal length	sepal width	petal length	petal width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [22]:

```
1 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   class           150 non-null   object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

```
In [24]: 1 df["sepal length"].value_counts(normalize=True)
```

```
Out[24]: sepal length
5.0    0.066667
5.1    0.060000
6.3    0.060000
5.7    0.053333
6.7    0.053333
5.8    0.046667
5.5    0.046667
6.4    0.046667
4.9    0.040000
5.4    0.040000
6.1    0.040000
6.0    0.040000
5.6    0.040000
4.8    0.033333
6.5    0.033333
6.2    0.026667
7.7    0.026667
6.9    0.026667
4.6    0.026667
5.2    0.026667
5.9    0.020000
4.4    0.020000
7.2    0.020000
6.8    0.020000
6.6    0.013333
4.7    0.013333
7.6    0.006667
7.4    0.006667
7.3    0.006667
7.0    0.006667
7.1    0.006667
5.3    0.006667
4.3    0.006667
4.5    0.006667
7.9    0.006667
Name: proportion, dtype: float64
```

```
In [25]: 1 df["petal length"].mean()
```

```
Out[25]: 3.7586666666666666
```

```
In [26]: 1 df["petal length"].mode()
```

```
Out[26]: 0    1.5
Name: petal length, dtype: float64
```

```
In [4]: 1 import pandas as pd
        2 from sklearn.model_selection import train_test_split
        3 from sklearn.naive_bayes import GaussianNB
        4 from sklearn.metrics import confusion_matrix, accuracy_score, precision_score
```

```
In [5]: 1 print("First 5 rows of the dataset:\n", df.head())
```

First 5 rows of the dataset:

	sepal length	sepal width	petal length	petal width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
In [7]: 1 X = df.drop('class', axis=1)
```

```
In [8]: 1 y = df['class']
```

```
In [9]: 1 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

```
In [10]: 1 model = GaussianNB()
        2 model.fit(X_train, y_train)
```

Out[10]: GaussianNB()

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [11]:

```
1 X_train
```

Out[11]:

	sepal length	sepal width	petal length	petal width
81	5.5	2.4	3.7	1.0
133	6.3	2.8	5.1	1.5
137	6.4	3.1	5.5	1.8
75	6.6	3.0	4.4	1.4
109	7.2	3.6	6.1	2.5
...
71	6.1	2.8	4.0	1.3
106	4.9	2.5	4.5	1.7
14	5.8	4.0	1.2	0.2
92	5.8	2.6	4.0	1.2
102	7.1	3.0	5.9	2.1

105 rows × 4 columns

In [12]:

```
1 y_train
```

Out[12]:

```
81    Iris-versicolor
133    Iris-virginica
137    Iris-virginica
75     Iris-versicolor
109    Iris-virginica
...
71     Iris-versicolor
106    Iris-virginica
14     Iris-setosa
92     Iris-versicolor
102    Iris-virginica
Name: class, Length: 105, dtype: object
```

In [13]:

```
1
2 y_pred = model.predict(X_test)
```

In [14]:

```
1 cm = confusion_matrix(y_test, y_pred)
2 print("\nConfusion Matrix:\n", cm)
```

Confusion Matrix:

```
[[19  0  0]
 [ 0 12  1]
 [ 0  0 13]]
```

```
In [15]: 1 TP = cm[1, 1]
          2 FP = cm[0, 1]
          3 TN = cm[0, 0]
          4 FN = cm[1, 0]
```

```
In [16]: 1 accuracy = accuracy_score(y_test, y_pred)
          2 print("\nAccuracy: ", accuracy)
```

Accuracy: 0.9777777777777777

```
In [17]: 1 error_rate = 1 - accuracy
          2 print("\nError Rate: ", error_rate)
```

Error Rate: 0.022222222222222254

```
In [18]: 1 precision = precision_score(y_test, y_pred, average='weighted')
          2 print("\nPrecision (weighted): ", precision)
```

Precision (weighted): 0.9793650793650793

```
In [19]: 1 recall = recall_score(y_test, y_pred, average='weighted')
          2 print("\nRecall (weighted): ", recall)
```

Recall (weighted): 0.9777777777777777

```
In [20]: 1 print(f"\nTP: {TP}, FP: {FP}, TN: {TN}, FN: {FN}")
```

TP: 12, FP: 0, TN: 19, FN: 0

```
In [29]: 1 train_predict=model.predict(X_train)
          2 test_predict=model.predict(X_test)
```

```
In [30]: 1 train_predict
```

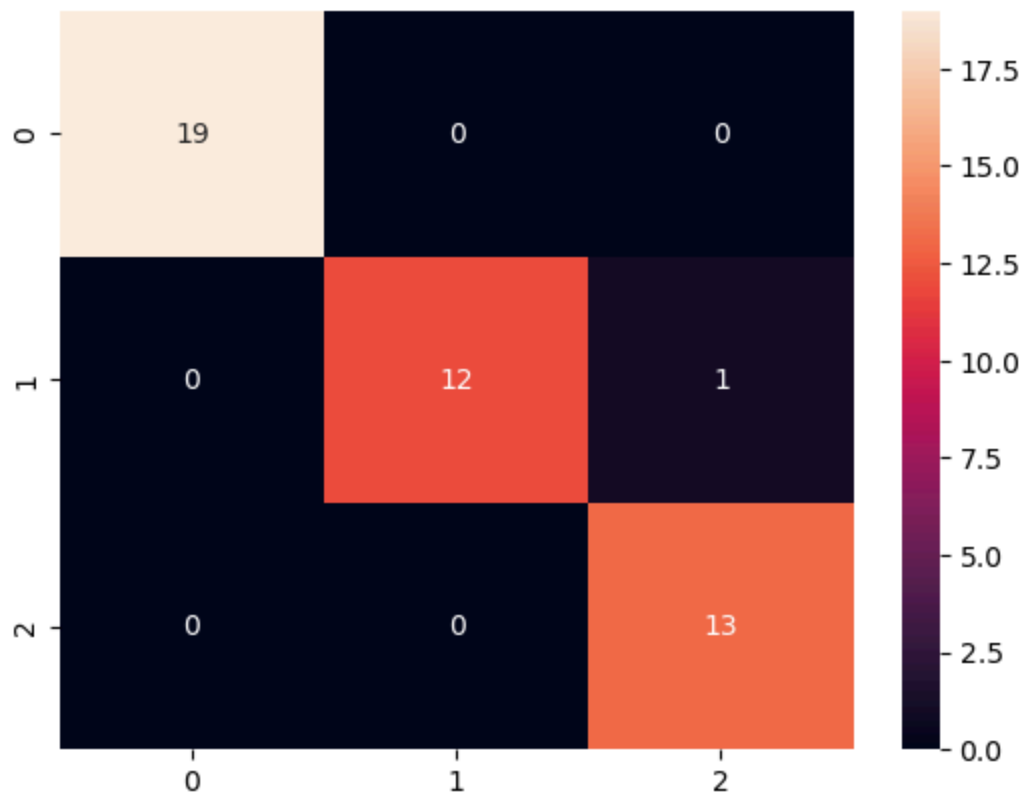
```
Out[30]: array(['Iris-versicolor', 'Iris-versicolor', 'Iris-virginica',  
                'Iris-versicolor', 'Iris-virginica', 'Iris-versicolor',  
                'Iris-virginica', 'Iris-versicolor', 'Iris-setosa',  
                'Iris-virginica', 'Iris-versicolor', 'Iris-setosa', 'Iris-setosa',  
                'Iris-setosa', 'Iris-versicolor', 'Iris-virginica', 'Iris-setosa',  
                'Iris-setosa', 'Iris-setosa', 'Iris-versicolor', 'Iris-setosa',  
                'Iris-versicolor', 'Iris-virginica', 'Iris-setosa',  
                'Iris-virginica', 'Iris-virginica', 'Iris-versicolor',  
                'Iris-versicolor', 'Iris-virginica', 'Iris-versicolor',  
                'Iris-setosa', 'Iris-versicolor', 'Iris-virginica', 'Iris-setosa',  
                'Iris-setosa', 'Iris-versicolor', 'Iris-virginica', 'Iris-setosa',  
                'Iris-virginica', 'Iris-setosa', 'Iris-setosa', 'Iris-virginica',  
                'Iris-versicolor', 'Iris-virginica', 'Iris-versicolor',  
                'Iris-virginica', 'Iris-virginica', 'Iris-versicolor',  
                'Iris-setosa', 'Iris-setosa', 'Iris-versicolor', 'Iris-virginica',  
                'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-versicolor',  
                'Iris-virginica', 'Iris-setosa', 'Iris-virginica',  
                'Iris-virginica', 'Iris-setosa', 'Iris-versicolor',  
                'Iris-versicolor', 'Iris-virginica', 'Iris-versicolor',  
                'Iris-virginica', 'Iris-setosa', 'Iris-virginica',  
                'Iris-versicolor', 'Iris-versicolor', 'Iris-setosa',  
                'Iris-versicolor', 'Iris-versicolor', 'Iris-setosa',  
                'Iris-versicolor', 'Iris-virginica', 'Iris-virginica',  
                'Iris-setosa', 'Iris-versicolor', 'Iris-virginica',  
                'Iris-virginica', 'Iris-setosa', 'Iris-virginica', 'Iris-setosa',  
                'Iris-virginica', 'Iris-virginica', 'Iris-virginica',  
                'Iris-versicolor', 'Iris-virginica', 'Iris-versicolor',  
                'Iris-versicolor', 'Iris-virginica', 'Iris-virginica',  
                'Iris-setosa', 'Iris-versicolor', 'Iris-versicolor', 'Iris-setosa',  
                'Iris-versicolor', 'Iris-virginica'], dtype='<U15')
```

```
In [31]: 1 test_predict
```

```
Out[31]: array(['Iris-versicolor', 'Iris-setosa', 'Iris-virginica',  
                'Iris-versicolor', 'Iris-versicolor', 'Iris-setosa',  
                'Iris-versicolor', 'Iris-virginica', 'Iris-versicolor',  
                'Iris-versicolor', 'Iris-virginica', 'Iris-setosa', 'Iris-setosa',  
                'Iris-setosa', 'Iris-setosa', 'Iris-virginica', 'Iris-virginica',  
                'Iris-versicolor', 'Iris-versicolor', 'Iris-virginica',  
                'Iris-setosa', 'Iris-virginica', 'Iris-setosa', 'Iris-virginica',  
                'Iris-virginica', 'Iris-virginica', 'Iris-virginica',  
                'Iris-virginica', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',  
                'Iris-setosa', 'Iris-versicolor', 'Iris-setosa', 'Iris-setosa',  
                'Iris-virginica', 'Iris-versicolor', 'Iris-setosa', 'Iris-setosa',  
                'Iris-setosa', 'Iris-virginica', 'Iris-versicolor',  
                'Iris-versicolor', 'Iris-setosa', 'Iris-setosa'], dtype='<U15')
```

```
In [38]: 1 import seaborn as sns  
2 sns.heatmap(cm,annot=True)
```

Out[38]: <Axes: >



```
In [ ]: 1
```

Name-pratik pate

Roll_No= 13258