Data Analytics III

1. Implement Simple Naïve Bayes classification algorithm using Python/R on iris.csv dataset.
2. Compute Confusion matrix to find TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall on the given dataset.

Import all Python Libraries

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

df = pd.read_csv("datasets/ass6/iris.csv")
df
```

```
      Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
0      1            5.1           3.5            1.4           0.2
1      2            4.9           3.0            1.4           0.2
2      3            4.7           3.2            1.3           0.2
3      4            4.6           3.1            1.5           0.2
4      5            5.0           3.6            1.4           0.2
..   ...            ...           ...            ...           ...
145  146            6.7           3.0            5.2           2.3
146  147            6.3           2.5            5.0           1.9
147  148            6.5           3.0            5.2           2.0
148  149            6.2           3.4            5.4           2.3
149  150            5.9           3.0            5.1           1.8

            Species
0       Iris-setosa
1       Iris-setosa
2       Iris-setosa
3       Iris-setosa
4       Iris-setosa
..              ...
145  Iris-virginica
146  Iris-virginica
147  Iris-virginica
148  Iris-virginica
149  Iris-virginica

[150 rows x 6 columns]
```

```
df.isna().sum()
```

```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

```
x = df.drop(['Id', 'Species'], axis = 1)
y = df['Species']
```

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test = train_test_split(x,y,train_size =
0.8,random_state = 22)
```

```
print(x_train.shape, x_test.shape, y_train.shape, y_test.shape)
```

```
(120, 4) (30, 4) (120,) (30,)
```

```
from sklearn.naive_bayes import GaussianNB
```

```
nb = GaussianNB()
nb.fit(x_train, y_train)
```

```
GaussianNB()
```

```
y_pred = nb.predict(x_test)
y_pred
```

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

```
from sklearn.metrics import confusion_matrix, accuracy_score,
precision_score, recall_score
```

```
cm = confusion_matrix(y_test,y_pred)
```

```
TP = cm[0, 0]
FP = cm[0, 1] + cm[0, 2]
FN = cm[1, 0] + cm[2, 0]
TN = cm[1, 1] + cm[2, 2] + cm[1,2] + cm[2,1]
```

```python
print(accuracy_score(y_test, y_pred))
print(1 - accuracy_score(y_test, y_pred))
print(precision_score(y_test, y_pred, average='macro'))
print(recall_score(y_test, y_pred, average='macro'))

print("TP:", TP)
print("FP:", FP)
print("TN:", TN)
print("FN:", FN)

print(cm)
```

```
0.9666666666666667
0.033333333333333326
0.9696969696969697
0.9761904761904763
TP: 6
FP: 0
TN: 24
FN: 0
[[ 6  0  0]
 [ 0 10  0]
 [ 0  1 13]]
```