Name :sandip gadadare.   Div:A
Roll no: CO3A09.   BATCH:B1

Assignment No: 6

Aim:- 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.

Source Code:-

```
import numpy as np

import pandas as pd

from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,shuffle=True)

from sklearn.naive_bayes import GaussianNB

from sklearn.metrics import accuracy_score, confusion_matrix,ConfusionMatrixDisplay

data=pd.read_csv('C:\\Users\\sspm\\Downloads\\iris.csv')

data
```

OUTPUT:

```
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

...    ...    ...    ...    ...    ...

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

150 rows × 5 columns
```

CODE:

```
data.isnull()
```

OUTPUT:

```
sepal_length    sepal_width    petal_length    petal_width    species

0      False    False    False    False    False

1      False    False    False    False    False

2      False    False    False    False    False

3      False    False    False    False    False

4      False    False    False    False    False

...    ...      ...      ...      ...      ...

145    False    False    False    False    False

146    False    False    False    False    False

147    False    False    False    False    False

148    False    False    False    False    False

149    False    False    False    False    False

150 rows × 5 columns
```

CODE:

```
data.isnull().sum()
```

OUTPUT:

```
sepal_length    0

sepal_width     0

petal_length    0

petal_width     0

species         0

dtype: int64
```

CODE:

```
data.columns
```

OUTPUT:

```
Index(['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species'],dtype='object')
```

CODE:

```
x=data.drop(["species"],axis=1)

y=data.drop(["sepal_length","sepal_width","petal_length","petal_width"],axis=1)

print(x)
```

OUTPUT:

```
   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

..       ...        ...        ...        ...

145        6.7        3.0        5.2        2.3

146        6.3        2.5        5.0        1.9

147        6.5        3.0        5.2        2.0

148        6.2        3.4        5.4        2.3

149        5.9        3.0        5.1        1.8

[150 rows x 4 columns]
```

CODE:

```
print(y)
```

OUTPUT:

```
     species

0     setosa

1     setosa

2     setosa
```

```
3      setosa
4      setosa
..        ...
145  virginica
146  virginica
147  virginica
148  virginica
149  virginica
[150 rows x 1 columns]
```

CODE:

```
print(x.shape)
print(y.shape)
```

OUTPUT:

```
(150, 4)
(150, 1)
```

CODE:

```
print(x_train.shape)
print(x_test.shape)
```

OUTPUT:

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

CODE:

```
print(y_train.shape)
print(y_test.shape)
```

OUTPUT:

```
(120, 1)
(30, 1)
```

CODE:

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

```
 y = column_or_1d(y, warn=True)
```

OUTPUT:

```
GaussianNB()
```

CODE:

```
y_pred = model.predict(x_test)

model.score(x_test,y_test)*100
```

OUTPUT:

```
93.33333333333333
```

CODE:

```
print(accuracy_score(y_test, y_pred)*100)
```

OUTPUT:

```
33.33333333333333
```

CODE:

```
cm = confusion_matrix(y_test, y_pred)

disp = ConfusionMatrixDisplay(confusion_matrix = cm)

print("Confusion matrix:")

print(cm)
```

OUTPUT:

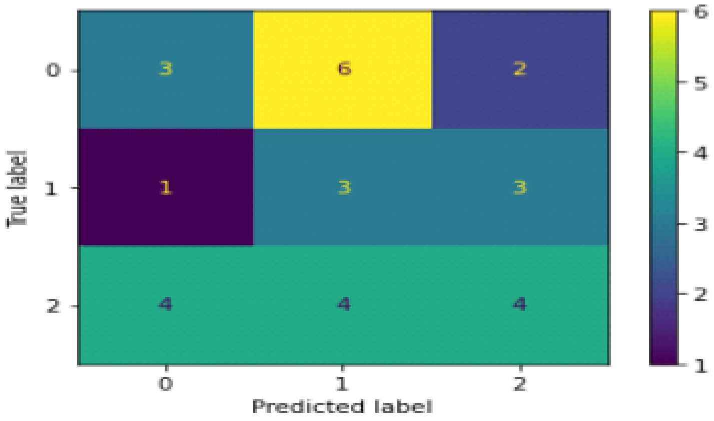
```
Confusion matrix:

[[3 6 2]

 [1 3 3]

 [4 4 4]]
```

CODE:

```
disp.plot()
```

OUTPUT:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x10aafbcb7f0>
```



CODE:

```
def get_confusion_matrix_values(y_true, y_pred):
    cm = confusion_matrix(y_true, y_pred)
    return(cm[0][0], cm[0][1], cm[1][0], cm[1][1])
TP, FP, FN, TN = get_confusion_matrix_values(y_test, y_pred)
print("TP: ", TP)
print("FP: ", FP)
print("FN: ", FN)
print("TN: ", TN)
```

OUTPUT:

```
TP:  3
FP:  6
FN:  1
TN:  3
```

CODE:

```
print("The Accuracy is ", (TP+TN)/(TP+TN+FP+FN))
print("The precision is ", TP/(TP+FP))
print("The recall is ", TP/(TP+FN))
```

OUTPUT:

```
The Accuracy is  0.46153846153846156
The precision is  0.3333333333333333
The recall is  0.75
```