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.

Naive Bayes Classification algorithm : =

1. Supervised Learning algorithm
2. Used for classification

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix , accuracy_score,precision_score,recall_score
```

```
df = pd.read_csv("/content/iris.zip")
df
```

	5.1	3.5	1.4	0.2	Iris-setosa
0	4.9	3.0	1.4	0.2	Iris-setosa
1	4.7	3.2	1.3	0.2	Iris-setosa
2	4.6	3.1	1.5	0.2	Iris-setosa
3	5.0	3.6	1.4	0.2	Iris-setosa
4	5.4	3.9	1.7	0.4	Iris-setosa
...	...	...	...	...	...
144	6.7	3.0	5.2	2.3	Iris-virginica
145	6.3	2.5	5.0	1.9	Iris-virginica
146	6.5	3.0	5.2	2.0	Iris-virginica
147	6.2	3.4	5.4	2.3	Iris-virginica
148	5.9	3.0	5.1	1.8	Iris-virginica

149 rows × 5 columns

Next steps: Generate code with df View recommended plots

```
df.columns=['sepallength','sepalwidth','petallength','petalwidth','class']
df
```

	sepallength	sepalwidth	petallength	petalwidth	class
0	4.9	3.0	1.4	0.2	Iris-setosa
1	4.7	3.2	1.3	0.2	Iris-setosa
2	4.6	3.1	1.5	0.2	Iris-setosa
3	5.0	3.6	1.4	0.2	Iris-setosa
4	5.4	3.9	1.7	0.4	Iris-setosa
...	...	...	...	...	...
144	6.7	3.0	5.2	2.3	Iris-virginica
145	6.3	2.5	5.0	1.9	Iris-virginica
146	6.5	3.0	5.2	2.0	Iris-virginica
147	6.2	3.4	5.4	2.3	Iris-virginica
148	5.9	3.0	5.1	1.8	Iris-virginica

149 rows × 5 columns

Next steps: Generate code with df View recommended plots

```
df.columns
```

```
    Index(['sepallength', 'sepalwidth', 'petallength', 'petalwidth', 'class'], dtype='object')
```

Split dataset into x and y

```
x = df[['sepallength', 'sepalwidth', 'petallength', 'petalwidth']]
y = df[['class']]
x
```

	sepallength	sepalwidth	petallength	petalwidth
0	4.9	3.0	1.4	0.2
1	4.7	3.2	1.3	0.2
2	4.6	3.1	1.5	0.2
3	5.0	3.6	1.4	0.2
4	5.4	3.9	1.7	0.4
...	...	...	...	...
144	6.7	3.0	5.2	2.3
145	6.3	2.5	5.0	1.9
146	6.5	3.0	5.2	2.0
147	6.2	3.4	5.4	2.3
148	5.9	3.0	5.1	1.8

149 rows × 4 columns

Next steps: Generate code with x View recommended plots

```
y
```

	class
0	Iris-setosa
1	Iris-setosa
2	Iris-setosa
3	Iris-setosa
4	Iris-setosa
...	...
144	Iris-virginica
145	Iris-virginica
146	Iris-virginica
147	Iris-virginica
148	Iris-virginica

149 rows × 1 columns

Next steps: Generate code with y View recommended plots

```
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size =0.2,random_state = 0)
```

```
gaussian = GaussianNB()
```

```
gaussian.fit(xtrain, ytrain)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/utils/validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d
  y = column_or_1d(y, warn=True)
▾ GaussianNB
GaussianNB()
```

```
y_pred = gaussian.predict(xtest)
```

```
y_pred
```

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

Confusion Matrix

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

```
array([[12,  0,  0],
       [ 0, 10,  0],
       [ 0,  3,  5]])
```

```
a = accuracy_score(ytest , y_pred)
a
```

```
0.9
```

```
e = 1 - a
e
```

```
0.09999999999999998
```