Name: Suraj Sawant

- 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 pandas as pd

df=pd.read\_csv('/content/Iris.csv')

df.head()

<b>→</b> *		Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
	0	1	5.1	3.5	1.4	0.2	Iris-setosa
	1	2	4.9	3.0	1.4	0.2	Iris-setosa
	2	3	4.7	3.2	1.3	0.2	Iris-setosa
	3	4	4.6	3.1	1.5	0.2	Iris-setosa
	4	5	5.0	3.6	1.4	0.2	Iris-setosa

df.tail()

₹		Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
	145	146	6.7	3.0	5.2	2.3	Iris-virginica
	146	147	6.3	2.5	5.0	1.9	Iris-virginica
	147	148	6.5	3.0	5.2	2.0	Iris-virginica
	148	149	6.2	3.4	5.4	2.3	Iris-virginica
	149	150	5.9	3.0	5.1	1.8	Iris-virginica

df.shape

**→** (150, 6)

df.info()

<</pre>
<<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):

Data	COTAMIIS (COCAT	o corumns).	
#	Column	Non-Null Count	Dtype
0	Id	150 non-null	int64
1	SepalLengthCm	150 non-null	float64
2	SepalWidthCm	150 non-null	float64
3	PetalLengthCm	150 non-null	float64
4	PetalWidthCm	150 non-null	float64
5	Species	150 non-null	object
dtype	es: float64(4),	int64(1), object	t(1)

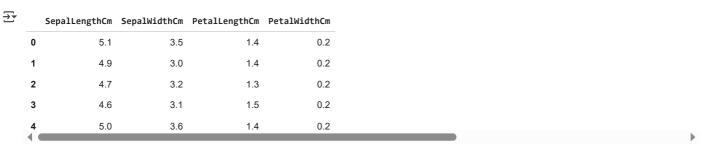
memory usage: 7.2+ KB

df.describe()

₹		Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
	count	150.000000	150.000000	150.000000	150.000000	150.000000
	mean	75.500000	5.843333	3.054000	3.758667	1.198667
	std	43.445368	0.828066	0.433594	1.764420	0.763161
	min	1.000000	4.300000	2.000000	1.000000	0.100000
	25%	38.250000	5.100000	2.800000	1.600000	0.300000
	50%	75.500000	5.800000	3.000000	4.350000	1.300000
	75%	112.750000	6.400000	3.300000	5.100000	1.800000
	max	150.000000	7.900000	4.400000	6.900000	2.500000

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

x.head()



x.shape

```
→ (150, 4)
```

 $\label{thm:conder} \begin{tabular}{ll} from $$ sklearn.preprocessing import LabelEncoder \\ label=LabelEncoder() \end{tabular}$ 

df['Species']=label.fit\_transform(df['Species'])

y=df['Species']
y.head()

₹	Species			
	0	0		
	1	0		
	2	0		
	3	0		
	4	0		

print(y.shape)

**→** (150,)

from sklearn.model\_selection import train\_test\_split
xtrain, xtest, ytrain, ytest=train\_test\_split(x,y,test\_size=0.2, random\_state=0)

xtrain.shape

**→** (120, 4)

xtest.shape

**→** (30, 4)

ytrain.shape

**→** (120,)

ytest.shape

**→** (30,)

from sklearn.naive\_bayes import GaussianNB
model=GaussianNB()

model.fit(xtrain, ytrain)



ypred=model.predict(xtest)
model.score(xtest, ytest)

**3.** 0.966666666666667

from sklearn.metrics import accuracy\_score, confusion\_matrix, classification\_report accuracy\_score(ytest, ypred)  $\frac{1}{2} \left( \frac{1}{2} + \frac{1}{2} +$ 

**→** 0.96666666666667

cm=confusion\_matrix(ytest, ypred)
print(cm)

Explaination of above 3\*3 Confusion Matrix:

Class 0: TP=11, TN=19, FP=0, FN=0

Class 1: TP=13, TN=16, FP=1, FN=0

Class 2: TP=5, TN=24, FP=0, FN=1

For Class 0:

TP: Same actual value and predicted value

TN: The sum of values of all columns and rows except the values of Class 0

FP: The sum of values of the corresponding column except for the TP value

FN: The sum of values of corresponding rows except for the TP value

Similarly for remaining classes

print(classification\_report(ytest, ypred))

₹	precision	recall	f1-score	support
0	1.00	1.00	1.00	11
1	0.93	1.00	0.96	13
2	1.00	0.83	0.91	6
accuracy			0.97	30
macro avg	0.98	0.94	0.96	30
weighted avg	0.97	0.97	0.97	30

Start coding or generate with AI.