NAME: Gayatri Gosavi

ROLL NO: TCOB13

*# Import the required libraries*

import pandas as pd

import matplotlib.pyplot as plt

data = pd.read\_csv("https://raw.githubusercontent.com/plotly/datasets/master/ iris-data.csv")

data.head()

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

data.shape (150, 5)

data.head()

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

data.tail()

sepal length sepal width petal length petal width

class 145

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 6.7 |  | 3.0 |  | 5.2 |  | 2.3 | Iris- |
| 6.3 |  | 2.5 |  | 5.0 |  | 1.9 | Iris- |
| 6.5 |  | 3.0 |  | 5.2 |  | 2.0 | Iris- |
| 6.2 |  | 3.4 |  | 5.4 |  | 2.3 | Iris- |
| 5.9 |  | 3.0 |  | 5.1 |  | 1.8 | Iris- |

virginica 146

virginica 147

virginica 148

virginica 149

virginica data.info()

X = data.drop(['class'], axis=1)

y = data.drop(['sepal length', 'sepal width', 'petal length', 'petal width'], axis=1)

print(X) print(y) print(X.shape) print(y.shape)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 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 |

Let us check if there are any Null values present

<class 'pandas.core.frame.DataFrame'> RangeIndex: 150 entries, 0 to 149 Data columns (total 5 columns):

# Column Non-Null Count Dtype

1. sepal length 150 non-null
2. sepal width 150 non-null
3. petal length 150 non-null

float64 float64 float64

3 petal width 150 non-null float64

4 class 150 non-null object dtypes: float64(4), object(1)

memory usage: 6.0+ KB

data.describe()

sepal length sepal width petal length petal width class

dtype: int64

0

0

0

0

0

data.isnull().sum()

Defining X and Y for the model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 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]

class

1. Iris-setosa
2. Iris-setosa
3. Iris-setosa
4. Iris-setosa
5. Iris-setosa

.. ...

1. Iris-virginica
2. Iris-virginica
3. Iris-virginica
4. Iris-virginica
5. Iris-virginica

[150 rows x 1 columns]

(150, 4)

(150, 1)

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)

print(X\_train.shape) print(X\_test.shape) print(y\_train.shape) print(y\_test.shape)

(120, 4)

(30, 4)

(120, 1)

(30, 1)

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

model.fit(X\_train, y\_train)

c:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\ validation.py:73: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n\_samples, ), for example using ravel().

return f(\*\*kwargs) GaussianNB()

[[ 8 0

[ 0 11

0]

1]

y\_pred = model.predict(X\_test) model.score(X\_test,y\_test)

0.9666666666666667

from sklearn.metrics import accuracy\_score, confusion\_matrix,

ConfusionMatrixDisplay print(accuracy\_score(y\_test, y\_pred))

0.9666666666666667

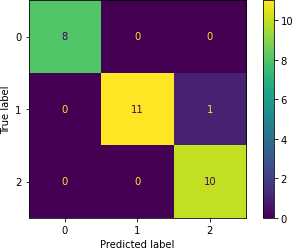
cm = confusion\_matrix(y\_test, y\_pred)

disp = ConfusionMatrixDisplay(confusion\_matrix = cm) print("Confusion matrix:")

print(cm) Confusion matrix:

[ 0 0 10]]

disp.plot() plt.show()



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)

TP: 8

FP: 0

FN: 0

TN: 11

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

The Accuracy is 1.0 The precision is 1.0 The recall is 1.0