**Practical - 8**

**ImplementationofNaïveBayesAlgorithmonJupiterNotebookusingPython.**

importnumpy as np

importmatplotlib.pyplot as plt

import pandas as pd

fromsklearn.datasets import load\_iris

iris = load\_iris()

dir(iris)

output:- ['DESCR', 'data', 'feature\_names', 'filename', 'target', 'target\_names']

iris.data

output:-array([[5.1, 3.5, 1.4, 0.2],

[4.9, 3. , 1.4, 0.2],

[4.7, 3.2, 1.3, 0.2],

[4.6, 3.1, 1.5, 0.2],

[5. , 3.6, 1.4, 0.2],

[5.4, 3.9, 1.7, 0.4],

[4.6, 3.4, 1.4, 0.3],

[5. , 3.4, 1.5, 0.2],

[4.4, 2.9, 1.4, 0.2],

[4.9, 3.1, 1.5, 0.1],

[5.4, 3.7, 1.5, 0.2],

[4.8, 3.4, 1.6, 0.2],

[4.8, 3. , 1.4, 0.1],

[4.3, 3. , 1.1, 0.1],

[5.8, 4. , 1.2, 0.2],

[5.7, 4.4, 1.5, 0.4],

[5.4, 3.9, 1.3, 0.4],

[5.1, 3.5, 1.4, 0.3],

[5.7, 3.8, 1.7, 0.3],

[5.1, 3.8, 1.5, 0.3],

[5.4, 3.4, 1.7, 0.2],

[5.1, 3.7, 1.5, 0.4],

[4.6, 3.6, 1. , 0.2],

[5.1, 3.3, 1.7, 0.5],

[4.8, 3.4, 1.9, 0.2],

[5. , 3. , 1.6, 0.2],

[5. , 3.4, 1.6, 0.4],

[5.2, 3.5, 1.5, 0.2],

[5.2, 3.4, 1.4, 0.2],

[4.7, 3.2, 1.6, 0.2],

[4.8, 3.1, 1.6, 0.2],

[5.4, 3.4, 1.5, 0.4],

[5.2, 4.1, 1.5, 0.1],

[5.5, 4.2, 1.4, 0.2],

[4.9, 3.1, 1.5, 0.2],

[5. , 3.2, 1.2, 0.2],

[5.5, 3.5, 1.3, 0.2],

[4.9, 3.6, 1.4, 0.1],

[4.4, 3. , 1.3, 0.2],

[5.1, 3.4, 1.5, 0.2],

[5. , 3.5, 1.3, 0.3],

[4.5, 2.3, 1.3, 0.3],

[4.4, 3.2, 1.3, 0.2],

[5. , 3.5, 1.6, 0.6],

[5.1, 3.8, 1.9, 0.4],

[4.8, 3. , 1.4, 0.3],

[5.1, 3.8, 1.6, 0.2],

[4.6, 3.2, 1.4, 0.2],

[5.3, 3.7, 1.5, 0.2],

[5. , 3.3, 1.4, 0.2],

[7. , 3.2, 4.7, 1.4],

[6.4, 3.2, 4.5, 1.5],

[6.9, 3.1, 4.9, 1.5],

[5.5, 2.3, 4. , 1.3],

[6.5, 2.8, 4.6, 1.5],

[5.7, 2.8, 4.5, 1.3],

[6.3, 3.3, 4.7, 1.6],

[4.9, 2.4, 3.3, 1. ],

[6.6, 2.9, 4.6, 1.3],

[5.2, 2.7, 3.9, 1.4],

[5. , 2. , 3.5, 1. ],

[5.9, 3. , 4.2, 1.5],

[6. , 2.2, 4. , 1. ],

[6.1, 2.9, 4.7, 1.4],

[5.6, 2.9, 3.6, 1.3],

[6.7, 3.1, 4.4, 1.4],

[5.6, 3. , 4.5, 1.5],

[5.8, 2.7, 4.1, 1. ],

[6.2, 2.2, 4.5, 1.5],

[5.6, 2.5, 3.9, 1.1],

[5.9, 3.2, 4.8, 1.8],

[6.1, 2.8, 4. , 1.3],

[6.3, 2.5, 4.9, 1.5],

[6.1, 2.8, 4.7, 1.2],

[6.4, 2.9, 4.3, 1.3],

[6.6, 3. , 4.4, 1.4],

[6.8, 2.8, 4.8, 1.4],

[6.7, 3. , 5. , 1.7],

[6. , 2.9, 4.5, 1.5],

[5.7, 2.6, 3.5, 1. ],

[5.5, 2.4, 3.8, 1.1],

[5.5, 2.4, 3.7, 1. ],

[5.8, 2.7, 3.9, 1.2],

[6. , 2.7, 5.1, 1.6],

[5.4, 3. , 4.5, 1.5],

[6. , 3.4, 4.5, 1.6],

[6.7, 3.1, 4.7, 1.5],

[6.3, 2.3, 4.4, 1.3],

[5.6, 3. , 4.1, 1.3],

[5.5, 2.5, 4. , 1.3],

[5.5, 2.6, 4.4, 1.2],

[6.1, 3. , 4.6, 1.4],

[5.8, 2.6, 4. , 1.2],

[5. , 2.3, 3.3, 1. ],

[5.6, 2.7, 4.2, 1.3],

[5.7, 3. , 4.2, 1.2],

[5.7, 2.9, 4.2, 1.3],

[6.2, 2.9, 4.3, 1.3],

[5.1, 2.5, 3. , 1.1],

[5.7, 2.8, 4.1, 1.3],

[6.3, 3.3, 6. , 2.5],

[5.8, 2.7, 5.1, 1.9],

[7.1, 3. , 5.9, 2.1],

[6.3, 2.9, 5.6, 1.8],

[6.5, 3. , 5.8, 2.2],

[7.6, 3. , 6.6, 2.1],

[4.9, 2.5, 4.5, 1.7],

[7.3, 2.9, 6.3, 1.8],

[6.7, 2.5, 5.8, 1.8],

[7.2, 3.6, 6.1, 2.5],

[6.5, 3.2, 5.1, 2. ],

[6.4, 2.7, 5.3, 1.9],

[6.8, 3. , 5.5, 2.1],

[5.7, 2.5, 5. , 2. ],

[5.8, 2.8, 5.1, 2.4],

[6.4, 3.2, 5.3, 2.3],

[6.5, 3. , 5.5, 1.8],

[7.7, 3.8, 6.7, 2.2],

[7.7, 2.6, 6.9, 2.3],

[6. , 2.2, 5. , 1.5],

[6.9, 3.2, 5.7, 2.3],

[5.6, 2.8, 4.9, 2. ],

[7.7, 2.8, 6.7, 2. ],

[6.3, 2.7, 4.9, 1.8],

[6.7, 3.3, 5.7, 2.1],

[7.2, 3.2, 6. , 1.8],

[6.2, 2.8, 4.8, 1.8],

[6.1, 3. , 4.9, 1.8],

[6.4, 2.8, 5.6, 2.1],

[7.2, 3. , 5.8, 1.6],

[7.4, 2.8, 6.1, 1.9],

[7.9, 3.8, 6.4, 2. ],

[6.4, 2.8, 5.6, 2.2],

[6.3, 2.8, 5.1, 1.5],

[6.1, 2.6, 5.6, 1.4],

[7.7, 3. , 6.1, 2.3],

[6.3, 3.4, 5.6, 2.4],

[6.4, 3.1, 5.5, 1.8],

[6. , 3. , 4.8, 1.8],

[6.9, 3.1, 5.4, 2.1],

[6.7, 3.1, 5.6, 2.4],

[6.9, 3.1, 5.1, 2.3],

[5.8, 2.7, 5.1, 1.9],

[6.8, 3.2, 5.9, 2.3],

[6.7, 3.3, 5.7, 2.5],

[6.7, 3. , 5.2, 2.3],

[6.3, 2.5, 5. , 1.9],

[6.5, 3. , 5.2, 2. ],

[6.2, 3.4, 5.4, 2.3],

[5.9, 3. , 5.1, 1.8]]”

iris.filename

output:- 'C:\\Users\\Admin\\anaconda3\\lib\\site-packages\\sklearn\\datasets\\data\\iris.csv'

df = pd.read\_csv('C:\\Users\\Admin\\anaconda3\\lib\\site-packages\\sklearn\\datasets\\data\\iris.csv')

fromsklearn.datasets import load\_iris

iris = load\_iris()

# store the feature matrix (X) and response vector (y)

X = iris.data

y = iris.target

# splitting X and y into training and testing sets

fromsklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.4, random\_state=1)

# training the model on training set

fromsklearn.naive\_bayes import GaussianNB

model = GaussianNB()

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

output:- GaussianNB(priors=None, var\_smoothing=1e-09)

# making predictions on the testing set

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

# comparing actual response values (y\_test) with predicted response values (y\_pred)

fromsklearn.metrics import accuracy\_score

print(f'Gaussian Naive Bayes model accuracy(in %):={accuracy\_score(y\_test, y\_pred)\*100} %')

res = model.predict([[6.5,3.0,5.2,2.0]])

print(f'Result = {iris.target\_names[res[0]]}')

output:- Gaussian Naive Bayes model accuracy(in %):=95.0 %

Result = virginica