**Name:** Jaydeep Solanki

**Roll:** 22BEC059

**Practical 6:** Write a program for classifying iris images using a KNN classifier. Impement accuracy, precision, recall and f1-measure.

1. Using SkLearn Library.

import pandas as pd

from sklearn.metrics import accuracy\_score, classification\_report

from sklearn.model\_selection import train\_test\_split

from sklearn.neighbors import KNeighborsClassifier

data = pd.read\_csv("Iris.csv", header='infer').values

x = data[:, 0:-1]

y = data[:, -1]

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.3, stratify=y)

k = int(input("Enter the nearest neighbor number(k) : "))

model = KNeighborsClassifier(n\_neighbors=k, weights="distance")

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

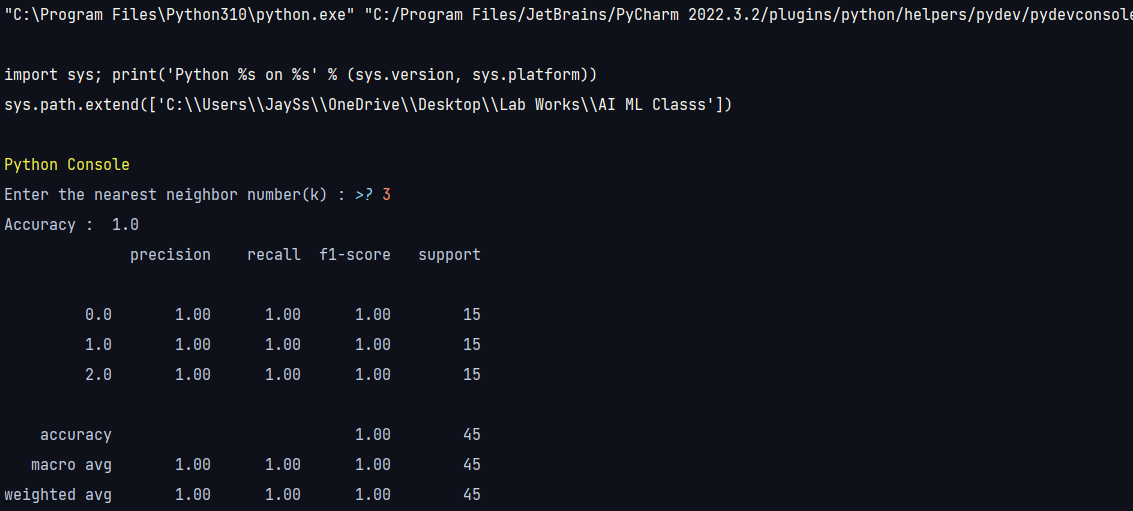
pred = model.predict(x\_test)

accuracy = accuracy\_score(y\_test, pred)

print("Accuracy : ", accuracy)

print(classification\_report(y\_test, pred))

Output:



1. Without using SkLearn Library

import numpy as np

import pandas as pd

from sklearn.model\_selection import train\_test\_split

data = pd.read\_csv("Iris.csv", header='infer').values

x = data[:, 0:-1]

y = data[:, -1]

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.3, stratify=y)

nClasses = np.unique(y\_train).shape[0]

distance = np.zeros(shape=x\_train.shape[0])

pred = np.zeros(shape=x\_test.shape[0])

classVotes = np.zeros(shape=nClasses)

k = int(input("Enter the nearest neighbor number(k) : "))

for i in range(x\_test.shape[0]):

    distance = np.sqrt(np.sum((x\_train - x\_test[i]) \*\* 2, axis=1))

    kMinIndex = np.argpartition(distance, k)[0:k]

    invDist = 1 / (distance + 10e-20)

    Denom = sum(invDist[kMinIndex])

    for j in range(k):

        classVotes[int(y\_train[kMinIndex[j]])] += invDist[kMinIndex[j]]

    classVotes /= Denom

    pred[i] = np.argmax(classVotes)

print(f"""

1. Pred : {pred}\n

2. Class Votes : {classVotes}\n

3. nClasses :{nClasses}\n

5. Distance : {distance}\n

6. Classification : C{classVotes.tolist().index(max(classVotes.tolist()))}

""")

Output:

