**IMPLEMENTING KNN ALGORITHM**

**-22pt27**

import pandas as pd

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

import math

from sklearn.preprocessing import StandardScaler

import statistics

import matplotlib.pyplot as plt

from matplotlib.colors import ListedColormap

df = pd.read\_csv('/content/Implementing KNN Algorithm.csv',sep = '\t',engine = 'python')

y=df['target']

**#normalize**

sc = StandardScaler()

df = pd.DataFrame(sc.fit\_transform(df), columns = df.columns)

x1=df['sepal length (cm)']

x2=df['sepal width (cm)']

x3=df['petal length (cm)']

x4=df['petal width (cm)']

X1\_test,X1\_train, X2\_test,X2\_train,X3\_test,X3\_train,X4\_test,X4\_train,y\_test,y\_train = train\_test\_split(x1,x2,x3,x4,y, test\_size=0.05, random\_state=0)

**def visvualization(x1,x2,x3,x4,y):**

    # Create a figure for the plots

    plt.figure(figsize=(5, 5))

    # 2D Scatter plot for Sepal Dimensions

    plt.subplot(2, 1, 1)

    plt.scatter(X1\_train, X2\_train, c=y\_train, cmap=ListedColormap(["green","blue","yellow"]), s=50)

    plt.scatter(x1, x2, c=y, s=50)

    plt.xlabel('Sepal Length (cm)')

    plt.ylabel('Sepal Width (cm)')

    plt.title('2D Scatter Plot of Sepal Dimensions')

    # 2D Scatter plot for Petal Dimensions

    plt.subplot(2, 1, 2)

    plt.scatter(X3\_train, X4\_train, c=y\_train, cmap=ListedColormap(["green","blue","yellow"]), s=50)

    plt.scatter(x3, x4, c=y, s=50)

    plt.xlabel('Petal Length (cm)')

    plt.ylabel('Petal Width (cm)')

    plt.title('2D Scatter Plot of Petal Dimensions')

    plt.tight\_layout()

    plt.show()

**k=3 # k nearest neighbors to take can be any odd value**

**def distance(x1,x2,x3,x4):**

  d={}

  for X1,X2,X3,X4,y in zip(X1\_train,X2\_train,X3\_train,X4\_train,y\_train):

    d[math.sqrt((x1-X1)\*\*2 + (x2-X2)\*\*2 +(x3-X3)\*\*2 +(x4-X4)\*\*2)]=y

  return d

for x1,x2,x3,x4 in zip(X1\_test,X2\_test,X3\_test,X4\_test):

  d=distance(x1,x2,x3,x4)

  myKeys = list(d.keys())

  myKeys.sort()

  myKeys = myKeys[:k]

  sorted\_dict = {i: d[i] for i in myKeys}

  #print(sorted\_dict)

  y="red"

  visvualization(x1,x2,x3,x4,y)

  if statistics.mode(sorted\_dict.values()) == 0:

    y="green"

  elif statistics.mode(sorted\_dict.values()) == 1:

    y="blue"

  elif statistics.mode(sorted\_dict.values()) == 2:

    y="yellow"

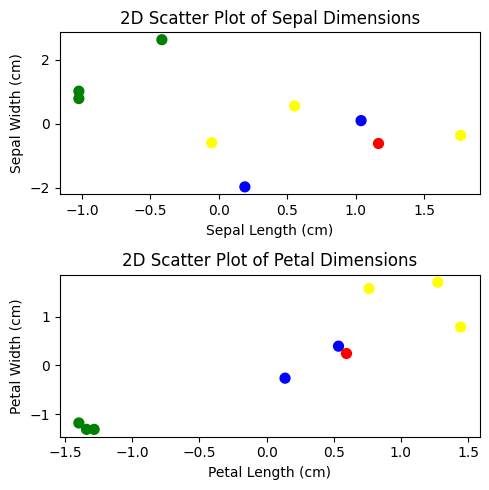
  else:

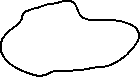
    y="red"

  visvualization(x1,x2,x3,x4,y)

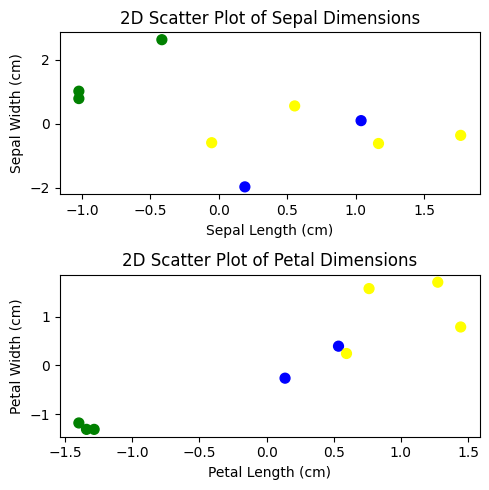
***OUTPUT***

***Before classification:-***



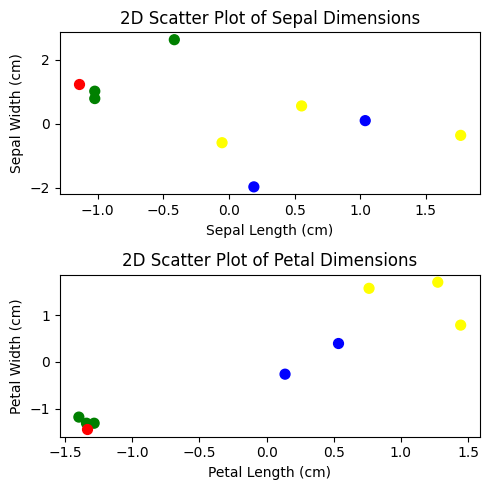


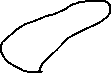
***After classification:-***



***Before classification:-***







***After classification:-***

