Write a program to implement k-Nearest Neighbour algorithm to classify the data set.

Aim:-

To write a program to implement k-Nearest Neighbour algorithm to classify the data set.

Program:-

```
import csv
import random
import math
import operator
def loadDataset(filename, split, trainingSet=[], testSet=[]):
    with open(filename,'rb') as csvfile:
        lines = csv.reader(csvfile)
        dataset = list(lines)
        for x in range(len(dataset)-1):
            for y in range(4):
                dataset[x][y] = float(dataset[x][y])
            if random.random() < split:
                trainingSet.append(dataset[x])
            else:
                testSet.append(dataset[x])
def euclideanDistance(instance1, instance2, length):
    distance = 0
    for x in range(length):
        distance += pow((instance1[x] - instance2[x]), 2)
    return math.sqrt(distance)
def getNeighbors(trainingSet, testInstance, k):
    distances = []
    length = len(testInstance) - 1
    for x in range(len(trainingSet)):
        dist = euclideanDistance(testInstance, trainingSet[x], length)
        distances.append((trainingSet[x], dist))
    distances.sort(key=operator.itemgetter(1))
    neighbors = []
    for x in range(k):
        neighbors.append(distances[x][0])
    return neighbors
def getResponse(neighbors):
    classVotes = {}
    for x in range(len(neighbors)):
        response = neighbors[x][-1]
        if response in classVotes:
            classVotes[response] += 1
        else:
            classVotes[response] = 1
    sortedVotes = sorted(classVotes.iteritems(), reverse = True)
    return sortedVotes[0][0]
def getAccuracy(testSet, predictions):
    correct = 0
    for x in range(len(testSet)):
        key= operator.itemgetter(1),
        if testSet[x][-1] == predictions[x]:
            correct +=1
    return(correct/float(len(testSet))) * 100.0
def main():
    trainingSet = [] testSet=[]
    split=0.67
    loadDataset('knndat.data', split, trainingSet, testSet)
    print("Train set:'+ repr(len(trainingSet)))
    print ("Test set:'+ repr(len(testSet)))
    predictions=[]
    k=3
    for x in range(len(testSet)):
        neighbors = getNeighbors(trainingSet, testSet[x];k)
        result = getResponse(neighbors)
        predictions.append(result)
        print('> predicted=' + repr(result) +', actual='+ repr(testSet[x]
        [-1])) accuracy = getAccuracy(testSet, predictions)
        print('Accuracy:' + repr(accuracy) + '%') main()
```

Result:-

Thus the program to implement k-Nearest Neighbour algorithm to classify the data set has been executed successfully.

Output:-

```
confusion matrix is as follows
        [[11 0 0]
          [0 9 1]
          [0 1 8]]
        Accuracy metrics0
        1.00 1.00 1.00 11
        1 0.90 0.90 0.90 10
        2 0.89 0.89 0.89 9
                 Avg/Total 0.93 0.93 0.93 30
```