LAB CYCLE 2

Experiment No:2
Date : 22/12/2021
AIM:

Problem Statement
        Write a program to use a K-nearest neighbor to predict class labels of test data. Euclidean distance should be used as the distance metric. Consider K=5. The learned classifier should be tested on test instances with unknown class labels, and the predicted class labels for the test instances should be printed as output.

Data Set Description:
        Training Data Filename: data4.csv, Training Data File Format: Boolean input attributes (x1, x2, ... , x8) in first 8 columns. The last (9th) column represents the Boolean class label (y). Each row is a training instance. There are 20 training instances.

Test Data Filename:
        test4.csv, Test Data File Format: Boolean input attributes (x1, x2, ... , x8) in each of the 8 columns. Note that, there is no class label column. Each row is a test instance. There are 4 test instances. The row number corresponds to the instance number of the test instances.

Input Format:
        Assume the data files data4.csv and test4.csv are present in the same directory and contain the training and test data. Thus, your program should not require any input from the user and should read from these files.

Output Format:
        Predicted class labels (0/1) for the test data exactly in the order in which the test instances are present in the test file. e.g., output 0 0 1 1 can give class labels for Test Instance 1: 0, Test Instance 2: 0, Test Instance 3: 1, Test Instance 4: 0

SOURCE CODE

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [1]:

In [3]:
```python
dataset=pd.read_csv("Downloads/data4.csv",header=None)
dataset
```

Out[3]:

	0	1	2	3	4	5	6	7	8
0	1	1	1	1	1	1	0	1	1
1	1	1	1	1	1	1	0	0	1
2	1	1	1	1	1	1	1	1	0
3	1	1	1	1	1	0	0	1	1
4	1	1	1	1	1	0	0	0	1
5	1	1	1	0	1	1	0	1	1
6	1	1	0	1	1	1	0	1	0
7	1	1	1	0	1	1	0	0	1
8	1	1	1	0	1	0	0	1	1
9	1	1	1	0	1	0	0	0	1
10	0	1	1	1	1	1	0	1	1
11	0	1	1	1	1	1	0	0	1
12	1	0	1	1	1	1	0	1	0
13	0	1	1	1	1	0	0	1	1
14	1	1	0	1	0	1	0	1	0
15	1	0	0	1	1	1	0	1	0
16	1	0	0	1	0	1	1	1	0
17	0	1	1	1	1	0	0	0	1
18	1	0	1	1	1	1	1	1	0
19	0	1	1	0	1	1	0	1	1

In [4]:
```python
testset=pd.read_csv("Downloads/test4.csv",header=None)
testset
```

Out[4]:

	0	1	2	3	4	5	6	7
0	0	1	1	1	1	1	1	1
1	1	0	0	0	0	0	0	0
2	0	1	1	0	1	0	0	0
3	0	1	1	1	1	0	0	0

In [6]:
```python
d=dataset.to_numpy()
t=testset.to_numpy()
k=5
```

In [13]:
```python
e=np.zeros((d.shape[0],t.shape[0]))
```

In [14]:
```python
for i in range(0,t.shape[0]):
    for j in range(0,d.shape[0]):
```

```
            e[j,i]=np.sum(np.square(d[j,0:8] - t[i,:]))

        e
```

Out[14]:
```
array([[2., 6., 4., 3.],
       [3., 5., 3., 2.],
       [1., 7., 5., 4.],
       [3., 5., 3., 2.],
       [4., 4., 2., 1.],
       [3., 5., 3., 4.],
       [3., 5., 5., 4.],
       [4., 4., 2., 3.],
       [4., 4., 2., 3.],
       [5., 3., 1., 2.],
       [1., 7., 3., 2.],
       [2., 6., 2., 1.],
       [3., 5., 5., 4.],
       [2., 6., 2., 1.],
       [4., 4., 6., 5.],
       [4., 4., 6., 5.],
       [4., 4., 8., 7.],
       [3., 5., 1., 0.],
       [2., 6., 6., 5.],
       [2., 6., 2., 3.]])
```

In [15]:
```
s=np.argsort(e,axis=0)
s
```

Out[15]:
```
array([[ 2,  9,  9, 17],
       [10,  4, 17,  4],
       [ 0,  7, 13, 13],
       [13,  8, 11, 11],
       [11, 16,  8,  9],
       [18, 15,  7,  1],
       [19, 14, 19,  3],
       [ 5,  1,  4, 10],
       [ 3, 17,  5,  0],
       [12,  3, 10,  7],
       [ 1,  5,  3,  8],
       [17,  6,  1, 19],
       [ 6, 12,  0,  6],
       [ 7,  0, 12, 12],
       [ 8, 11,  2,  5],
       [ 4, 18,  6,  2],
       [14, 13, 18, 18],
       [15, 19, 14, 14],
       [16, 10, 15, 15],
       [ 9,  2, 16, 16]], dtype=int64)
```

In [17]:
```
knn=np.zeros((k,t.shape[0]))
```

In [18]:
```
for i in range(0,t.shape[0]):
    for j in range(0,k):
        knn[j,i]=d[s[j,i],8]

knn
```

Out[18]:
```
array([[0., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 0., 1., 1.]])
```

In [20]:
```
print("The class labels are : \n ",np.max(knn,axis=0))
```

The class labels are :

    [1. 1. 1. 1.]

In [ ]: