**----------------------------performance metrics----------------EXERCISE#1-----------------------------------------**

import numpy as np

x=np.array([[0, 0, 0],[0, 0, 1],[0, 1 ,0],[0, 1, 1],[1, 0, 0],[1, 0, 1],[1, 1, 0],[1, 1, 1]])

y=np.array([0,0,0,0,1,1,1,1])

print(x)

print(y)

w=np.array([1,-1,-1])

b=np.array([-0.5])

def perceptron(x):

if x>=0:

return 1

else:

return 0

net=np.zeros(8)

out=np.zeros(8,dtype=np.int16)

def net\_input(x,w,b):

for i in range(8):

net[i]=(x[i]\*w).sum()+b

out[i]=perceptron(net[i])

return net,out

[n,o]=net\_input(x,w,b)

print(n)

print(o)

error=y-o

print(error)

**----------------------------Data Division----------------EXERCISE#2-----------------------------------------**

**i**mport numpy as np

x=np.arange(1,101,dtype=np.int16)

print(x)

print(x.size)

#

def train\_test\_split(x,test\_size):

train\_size=100-test\_size

x\_train=x[0:train\_size]

x\_test=x[train\_size:]

return x\_train,x\_test

[x\_train,x\_test]=train\_test\_split(x,20)

print(x\_train)

print(x\_train.size)

print(x\_test)

print(x\_test.size)

**----------------------------Data Division----------------EXERCISE#3-----------------------------------------**

import numpy as np

np.set\_printoptions(threshold=np.inf)

x=np.arange(1,101,dtype=np.int16)

print(x)

print(x.size)

#

import random

def train\_test\_split(x,test\_size):

train\_size=100-test\_size

x\_train=[]

x\_test=[]

while(len(x\_train)<train\_size):

x1=random.choice(x)

if x1 not in x\_train:

x\_train.append(x1)

while(len(x\_test)<test\_size):

x1=random.choice(x)

if ((x1 not in x\_train) and (x1 not in x\_test)):

x\_test.append(x1)

return x\_train,x\_test

[x\_train,x\_test]=train\_test\_split(x,20)

x\_train=np.array(x\_train)

x\_test=np.array(x\_test)

print(x\_train)

print(x\_train.size)

print(x\_test)

print(x\_test.size)