import numpy as np

sigmoid = lambda x: 1/(1+np.exp(-x))

dsigmoid = lambda x: x\*(1-x)

class NN:

def \_\_init\_\_(self, inp, hid, out, lr=0.1):

self.W1, self.b1 = np.random.randn(inp,hid), np.zeros((1,hid))

self.W2, self.b2 = np.random.randn(hid,out), np.zeros((1,out))

self.lr = lr

def forward(self, X):

self.h = sigmoid(X@self.W1 + self.b1)

self.o = sigmoid(self.h@self.W2 + self.b2)

return self.o

def train(self, X, y, epochs=10000):

for \_ in range(epochs):

o = self.forward(X)

d2 = (y-o)\*dsigmoid(o)

d1 = (d2@self.W2.T)\*dsigmoid(self.h)

self.W2 += self.h.T@d2\*self.lr; self.b2 += d2.sum(0,keepdims=True)\*self.lr

self.W1 += X.T@d1\*self.lr; self.b1 += d1.sum(0,keepdims=True)\*self.lr

# XOR Example

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

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

nn = NN(2,2,1); nn.train(X,y)

print(nn.forward(X))

