Assignment3

June 17, 2022

1 Implement Backpropagation algorithm to train majority function with 3-bits. Use network of configuration 3x2x2x1.

[1]:
```
# Importing the Required Libaries
import numpy as np
```

1.1 Sigmoid Activation Function

[2]:
```
def sigmoid(x):
    return 1/(1+np.exp(-x))
```

1.2 Derivative of Sigmoid

[3]:
```
def sigmoid_derivative(x):
    return x * (1 - x)
```

1.3 Creating Training Dataset

[4]:
```
x = np.array([[0,0,0],[0,0,1],[0,1,1],[0,1,0],[1,0,0],[1,0,1],[1,1,0],[1,1,1]])
t = np.array([[0],[0],[1],[0],[0],[1],[1],[1]])
```

1.4 Setting Up Hyper Parameters

[5]:
```
epochs = 10000
lr = 0.1
```

1.5 Setting up Neural Network Structure

[6]:
```
ILNeurons, HLNeurons1,HLNeurons2, OLNeurons = 3,2,2,1
```

1.6 Weight and Bias Initialization

[7]:
```
wh1 = np.random.uniform(size=(ILNeurons,HLNeurons1))
bh1 =np.random.uniform(size=(1,HLNeurons1))

wh2 = np.random.uniform(size=(HLNeurons1,HLNeurons2))
bh2 =np.random.uniform(size=(1,HLNeurons2))

wo = np.random.uniform(size=(HLNeurons2,OLNeurons))
bo = np.random.uniform(size=(1,OLNeurons))
```

1.7 Training the Model

[8]:
```
for i in range(epochs):
    # Forward Propagation
    vh1 = np.dot(x,wh1)
    vh1+=bh1
    yh1 = sigmoid(vh1)

    vh2 = np.dot(yh1,wh2)
    vh2+=bh2
    yh2 = sigmoid(vh2)

    vo = np.dot(yh2,wo)
    vo+=bo
    yo = sigmoid(vo)

    # Back Propagation
    error = t-yo
    deltao = error+sigmoid_derivative(yo)

    hidden_err2 = deltao.dot(wo.T)
    deltah2 = hidden_err2*sigmoid_derivative(yh2)

    hidden_err1 = deltah2.dot(wh2.T)
    deltah1 = hidden_err1*sigmoid_derivative(yh1)

    # Udating the Weights and Bias
    wo+=yh2.T.dot(deltao)*lr
    bo+=np.sum(deltao,axis=0,keepdims=True)*lr

    wh2+=yh1.T.dot(deltah2)*lr
    bh2+=np.sum(deltah2,axis=0,keepdims=True)*lr

    wh1+=x.T.dot(deltah1)*lr
    bh1+=np.sum(deltah1,axis=0,keepdims=True)*lr
```

```
print("Final hidden Layer 1 weights: ",end='')
print(*wh1)
print("Final hidden Layer 2 weights: ",end='')
print(*wh2)
print("Final hidden Layer 1 bias: ",end='')
print(*bh1)
print("Final hidden Layer 2 bias: ",end='')
print(*bh2)
print("Final output weights: ",end='')
print(*wo)
print("Final output bias: ",end='')
print(*bo)

print("\nOutput from neural network after 10,000 epochs: ",end='')
print(*yo)
```

```
Final hidden Layer 1 weights: [3.3311879 1.8427214] [3.30723517 1.88264993]
[3.30582667 1.88495384]
Final hidden Layer 2 weights: [7.21834025 2.48352736] [3.17622517 1.56641908]
Final hidden Layer 1 bias: [-5.04352362 -2.90221059]
Final hidden Layer 2 bias: [-5.09232956 -1.94198313]
Final output weights: [11.91501417] [2.85712987]
Final output bias: [-4.92681678]

Output from neural network after 10,000 epochs: [0.01157858] [0.02272764]
[0.99983127] [0.0227233] [0.022649] [0.99983069] [0.99983066] [0.99991943]
```