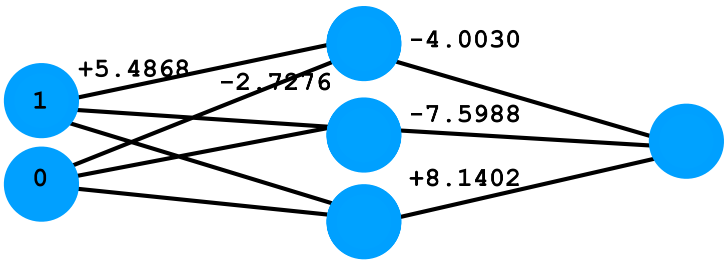
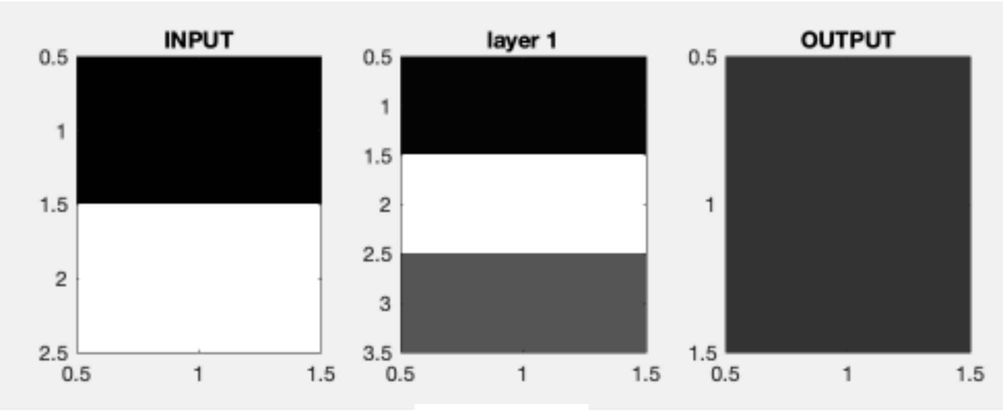
Neural Network

Simulates XOR logic - exclusive or

2 inputs, 1 output,
1 hidden layer
with 3 neurons &
9 synapses





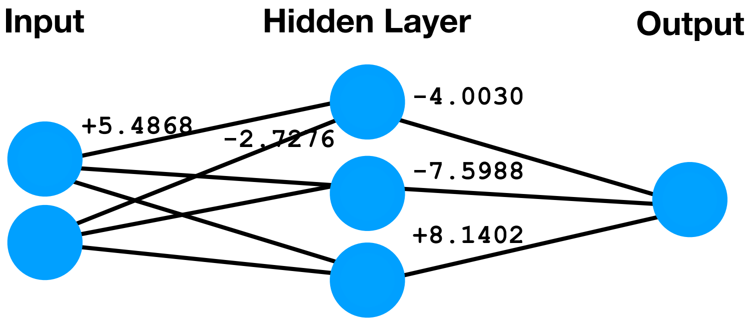
```
1
0
```

```
0.9959
0.0153
0.6719
```

```
0.7969
```

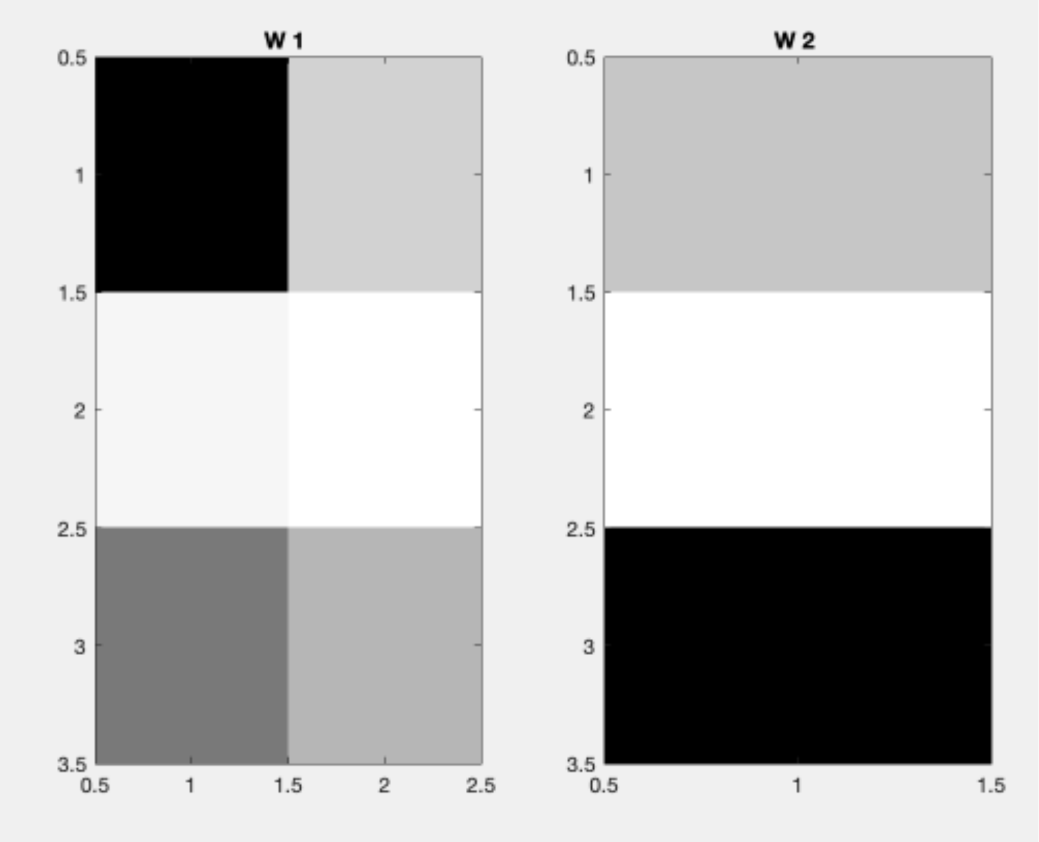
True

```
EXAMPLE for input of  1
                      0

  node value = sigmaFunc( sum of (node activation * connection weight) )

  where sigmaFunc(x) = 1 / (1 + exp(-x))  >> converts all input x values into range 0 to 1

INPUT > HIDDEN LAYER

  sigmaFunc( 1*5.4868 + 0*(-2.7276) ) = 0.9959 = hidden node 1 activation

  sigmaFunc( 1*(-4.1663) + 0*(-4.4991) ) = 0.0153 = hidden node 2 activation

  sigmaFunc( 1*0.7169 + 0*(-1.5748) ) = 0.6719 = hidden node 3 activation

HIDDEN LAYER > OUTPUT

  sigmaFunc( 0.9959*(-4.0030) + 0.0153*(-7.5988) + 0.6719*8.1402 ) = 0.7969 = output node
```



Synapse weights to hidden layer and to output, min = -7.60 (white), max = +8.14 (black)

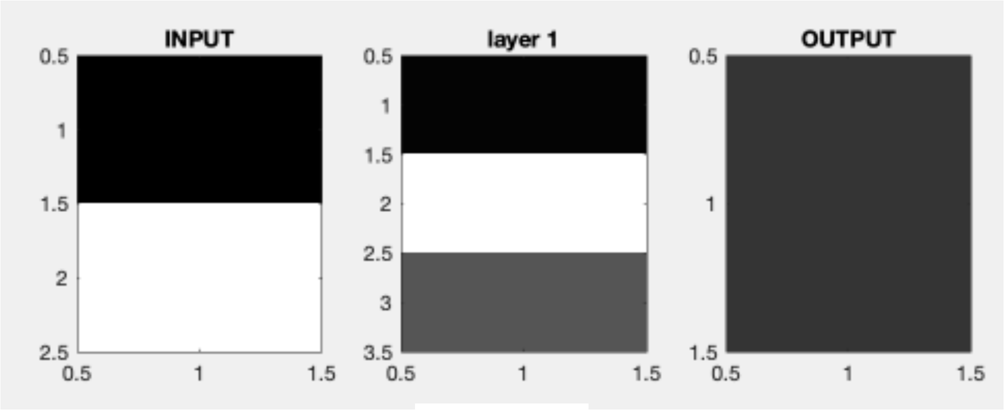
```
 5.4868   -2.7276
-4.1663   -4.4991
 0.7169   -1.5748
```



```
-4.0030
-7.5988
 8.1402
```

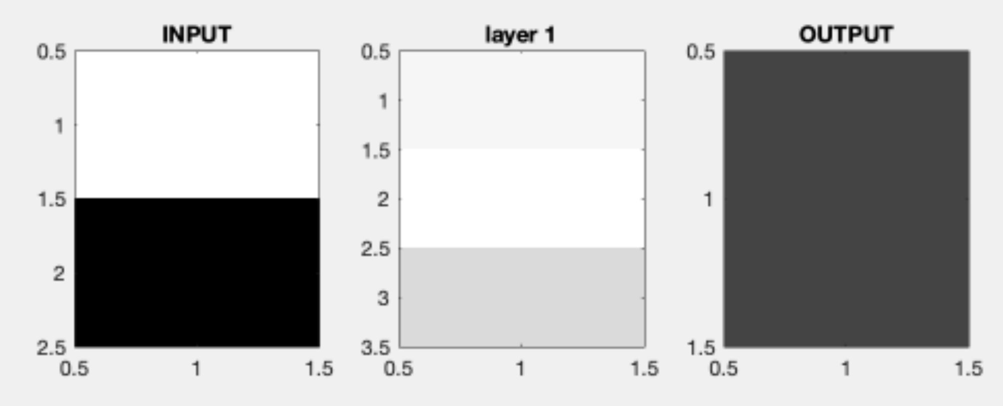
Below are node activations: input > hidden layer > output

Simulates XOR logic - exclusive or



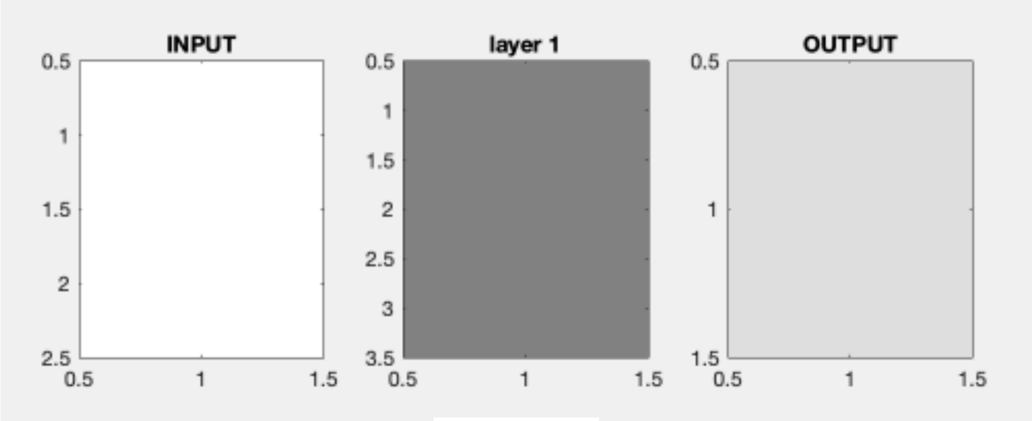
```
1        0.9959        0.7969
0        0.0153
         0.6719
```

True



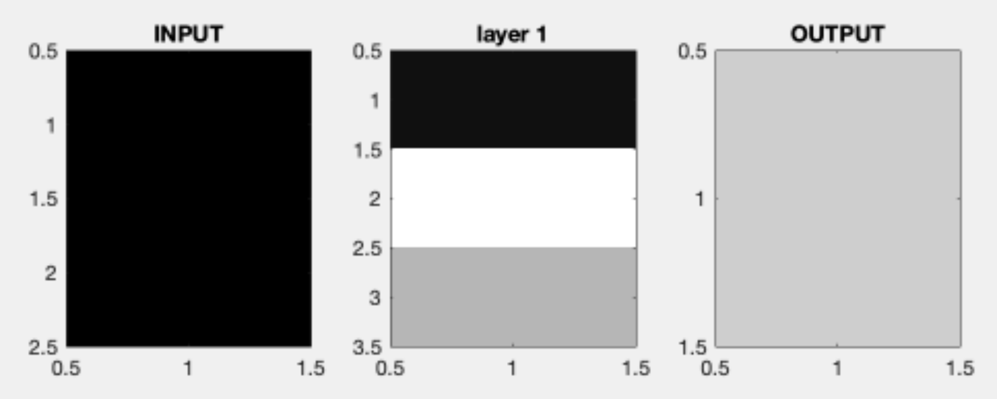
```
0        0.0614        0.7440
1        0.0110
         0.1715
```

True



```
0        0.5000        0.1505
0        0.5000
         0.5000
```

False



```
1        0.9404        0.2072
1        0.0002
         0.2978
```

False