Assignment -7

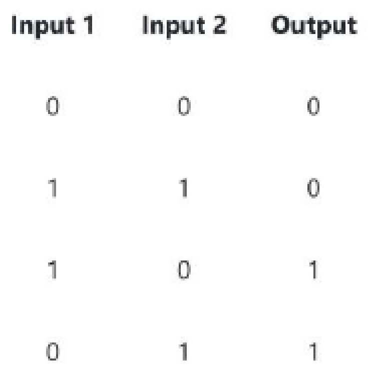
Title: python program to show back propagation network for XOR function with Binary Input and Output

Aim: Write a python program to show back propagation network for XOR function with Binary Input and Output.

Theory:

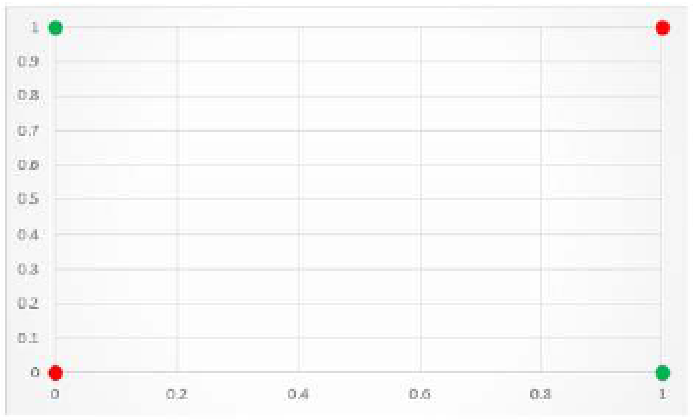
Solving XOR problem with Back Propagation Algorithm

XOR or Exclusive OR is a classic problem in Artificial Neural Network Research. An XOR function takes two binary inputs (0 or 1) & returns True if both inputs are different & False if both inputs are same.



Input 1	Input 2	Output
0	0	0
1	1	0
1	0	1
0	1	1

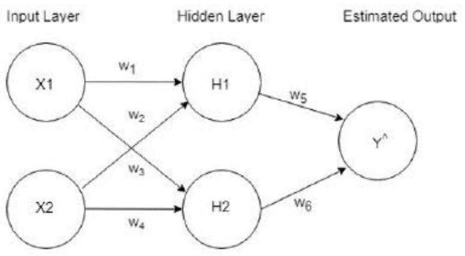
On the surface, XOR appears to be a very simple problem, however, Minksy and Papert (1969) showed that this was a big problem for neural network architectures of the 1960s, known as perceptrons. A limitation of this architecture is that it is only capable of separating data points with a single line. This is unfortunate because the XOR inputs are not linearly separable. This is particularly visible if you plot the XOR input values to a graph. As shown in the figure, there is no way to separate the 1 and O predictions with a single classification line.



Solution

The back propagation algorithm begins by comparing the actual value output by the forward propagation process to the expected value and then moves backward through the network, slightly adjusting each of the weights in a direction that reduces the size of the error by a small degree. Both forward and back propagation are re-run thousands of times on each input combination until the network can accurately predict the expected output of the possible inputs using forward propagation.

Model



Inputs

$$x_1 = [1,1]^T, \quad y_1 = +1$$
$$x_2 = [0,0]^T, \quad y_2 = +1$$
$$x_3 = [1,0]^T, \quad y_3 = -1$$
$$x_4 = [0,1]^T, \quad y_4 = -1$$

2 hidden neurons are used, each takes two inputs with different weights. After each forward pass, the error is back propagated. I have used sigmoid as the activation function at the hidden layer.

At hidden layer:

$$H_1 = x_1 w_1 + x_2 w_2$$
$$H_2 = x_1 w_3 + x_2 w_4$$

At output layer:

$$Y^{\wedge} = \sigma(H_1) w_5 + \sigma(H_2) w_6$$

here, σ represents sigmoid function.

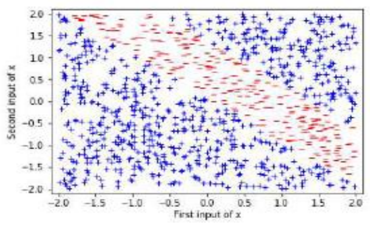
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Loss function:

$$\frac{1}{2}(Y - Y^{\wedge})^2$$

Results

Classification WITHOUT gaussian noise



```
(731, 269)
```

Observation : To classify, I generated 1000 x 2 random floats in range -2 to 2. Using weights from trained model, I classified each input & plotted it on 2-D space. Out of 1000, 731 points were classified as"+1" and 269 points were classified as **"-1"** It is clearly seen, classification region is not a single line, rather the 2-D region is separated by "-1" class. I did the same for classifications with Gaussian noise.

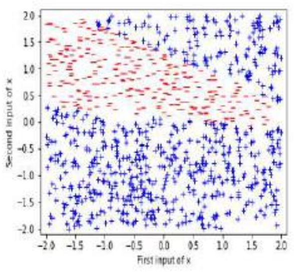
Classification WITH gaussian noise

In real applications, we almost never work with data without noise. Now instead of using the above points generate Gaussian random noise centered on these locations.

$$
\begin{aligned}
x_1 &\sim \mu_1 = [1,1]^T, \Sigma_1 = \Sigma \quad y_1 = +1 \\
x_2 &\sim \mu_1 = [0,0]^T, \Sigma_2 = \Sigma \quad y_1 = +1 \\
x_3 &\sim \mu_1 = [1,0]^T, \Sigma_3 = \Sigma \quad y_1 = -1 \\
x_4 &\sim \mu_1 = [0,1]^T, \Sigma_4 = \Sigma \quad y_1 = -1
\end{aligned}
$$

$$\Sigma = \begin{bmatrix} \sigma & 0 \\ 0 & \sigma \end{bmatrix}$$

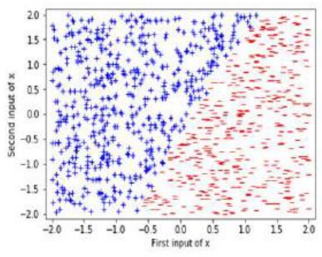
σ = 0.5



```
(782, 298)
```

Observation : We see a shift in classification regions. Here, classification is still not separated by a line

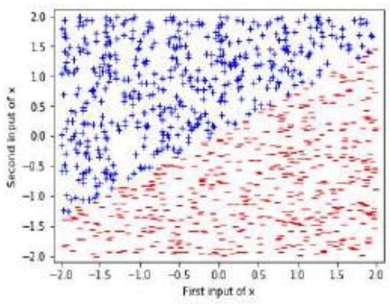
σ = 1.0



```
(538, 462)
```

Observation : We observe classifications being divided into two distinct regions.

σ = 2.0



```
(467, 533)
```

Observation : We see a shift in classification regions (compared to of σ = 1). Here also, we observe two distinct regions of classification.

Conclusion:

We have successfully implemented XOR problem with Back Propagation Algorithm.

Questions:

1. Explain two basic features of mapping.
2. Explain Feature Mapping Models.
3. Explain Self Organization Map.
4. Explain SOM Algorithm.
5. Explain properties of Feature Map.