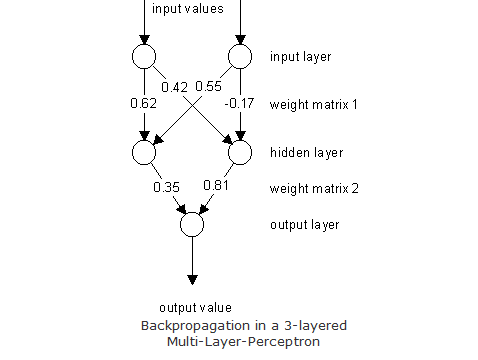
Suppose you have the following 3-layered Multi-Layer-Perceptron:



Patterns to be learned:

|  |  |  |
| --- | --- | --- |
| **Input1** | **Input2** | **Target** |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

First, the weight values are set to random values:

**V1** =0.62;

**V2**=0.42, [for weight matrix 1, weights between the input layer to hidden layer with name V1, V2, V3 and V4]

**V3**= 0.55,

**V4**= -0.17

**W5**= 0.35,

**W6**=0.81 [for weight matrix 2, weights between the hidden layer to output layer with name W5 and W6]

**The learning rate** of the net is set to **0.25**.

Next, the values of the **first input pattern (X1=0 and X2=1)** are set to the neurons of the input layer (the output of the input layer is the same as its input). And **target** for this pattern is Zero **(0)**

**The neurons in the hidden layer are activated:**

**Input of hidden neuron 1:**

**Zin1=X1\*V1+X2\*V3**=0 \* 0.62 + 1 \* 0.55 = **0.55**

**Input of hidden neuron 2:**

**Zin2=X1\*V2+X2\*V4**=0 \* 0.42 + 1 \* (-0.17) = **-0.17**

**Output of hidden neuron 1:**

**Z1=1 / ( 1 + exp(-Zin1) )** =**1 / ( 1 + exp(-0.55) )** = **0.634135591**

**Output of hidden neuron 2**:

**Z2=1 / ( 1 + exp(-Zin2) )** =**1 / ( 1 + exp(+0.17) )** = **0.457602059**

**The neurons in the output layer are activated:**

**Input of output neuron:**

**yin=Z1\*W5+Z2\*W6**=0.634135591 \* 0.35 + 0.457602059 \* 0.81 = **0.592605124**

**Output of output neuron:**

**y=1 / ( 1 + exp(-Yin ) )** =1 / ( 1 + exp(-0.592605124) ) = **0.643962658**

**Having calculated y, forward pass is completed.**

**(1)Back Propagation to the output layer:**

Next step is to calculate error of output node with name **“y”**. Predicted (computed) value of node (**y**) in our example is **0.643962658**. Error calculation is done the following way:

**Error of the output Node =**

**(Output node (y) error) δ y** = **e \***

**Output node ( y) Error** = **(target-computed) \***

= [(0 - **0.643962658**)] \* [**0.643962658**\*(1 - **0.643962658**) ]

= **-0.643962658**\* [**-0.643962658**\*(1 - **0.643962658**)]

**=0.14764437941**

Once error is known, it will be used for (i) backward propagation and (ii) weights adjustment. It is two step processes.

Error is propagated from output layer to the hidden layer first. This is where learning rate are brought to equation. So weights W1,0(**W5**) and W1,1(**W6**) will be updated first.

Before weights can be updated, rate of change needs to be found. This is done by multiplication of the learning rate, error value and node N1,0 value

**We start with changing the weights in weight matrix 2:**

[Here only two weights **W5 and W6**that are connected between the hidden and output layer]

**Value for changing weight W5:**

**W5=α\*δY \*Z1**

[ α = 0.25 and Z1 = **0.634135591**

**(Output error) Y= e \*= (target-computed) \***

**=0.14764437941** (we compute it at above)

Where = y\*(1-y) [first derivative can be found be y\*(1-y) ]

W5 =α\*δY \*Z1

Putting the values of all three **α, δN2, 0and Z1**

=0.25 \* **0.14764437941** \* **0.634135591**

**W5** = **0.0234066389**

**Value for changing weight W6:**

**W6=α\*δY \*Z2**

[  **α** = 0.25 and **Z2** = **0.457602059**and

**(Output error) δN2, 0= e \*= (target-computed) \***

**=0.14764437941** (we compute it at above)

Where = y\*(1-y) [first derivative can be found be y\*(1-y) ]

W6 =α\*δY \*Z2

Putting the values of all three **α, δN2, 0and Z2**

=0.25 \* **0.14764437941** \* **0.457602059**

**W6** = **0.0168905930**

**;**

**[W5 and W6 are updated that are connected between the hidden and output layer]**

**W5New= W5old+ΔW5**

**W5New**= 0.35+ 0.0234066389

**W5New = 0.3734066389**

**W6New= W6old+ΔW6**

W6New = 0.81 + 0.0168905930

**W6New = 0.826890593**

**(2) Back propagation to the hidden layer**

Now errors have to be propagated from hidden layer down to the input layer.

This is bit more complicated than propagating error from output to hidden layer. In previous case, output from node”**Y”** was know beforehand.

Output of nodes Z1 and Z2 was unknown. Let's start with finding Z1 error first. This will be calculated multiplying new weight **(W5)** value with error for the output node **“Y”** value. Same way error for Z2 node will be found.

**(Error of first hidden Neuron Z1) δZ1= error of output node δY \* W5New**

**=** 0.14764437941 \* 0.3734066389

**δZ1**= **0.05513139146**

**(Error of second hidden Neuron Z2) δZ2= error of output node δY \* W6New**

**=** 0.14764437941 \* 0.826890593

**δZ2**= **0.1220857484**

Once error for hidden layer nodes is known, weights between input and hidden layer can be updated. Rate of change first needs to be calculated for every weight:

**Now we will change the weights in weight matrix 1:**

We start with changing the weights in weight matrix 2: [here only four weights **V1, V2, V3 and V4**  that are connected between the input layer and hidden layer]

1. **Value for changing weight V1:**

V1 =α\* **δZ1**\*X1

[ X1=1,  **α** = 0.25 and **δZ1** = 0.05513139146 **]**

V1 =α\* **δZ1**\*X1

**V1  =** 0.25 \*0.05513139146 \*1

**V1**  = **0.013782847865**

1. **Value for changing weight V2:**

V2 =α\* **δZ2**\*X2

[ X2=0,  **α** = 0.25 and **δZ2** = 0.1220857484**]**

V2 =α\* **δZ2**\*X2

**V2  =** 0.45 \*0.012965\*0

**V2**  = 0

1. **Value for changing weight V3:**

V3 =α\* **δZ1**\*X1

[ X1=1,  **α** = 0.25 and **δZ1** = 0.05513139146  **]**

V3 =α\* **δZ1**\*X1

**V3  =** 0.25 \* 0.05513139146 \*1

**V3 = 0.013782847865**

1. **Value for changing weight V4:**

V4 =α\* **δZ2**\*X1

[ X2=0,  **α** = 0.25 and **δZ2** = 0.1220857484**]**

V4 =α\* **δZ2**\*X2

**V4  =** 0.45 \* 0.1220857484\*0

**V4 = 0**

**[And Now V1, V2, V3 and V4are updated that are connected between the input and output layer]**

**V1New= V1old+**ΔV1

**V1New** = 0.62 + **0.013782847865**

**V1New =0.633782847865**

**V2New= V2old+**ΔV2

**V2New** =0.42 + 0.0

**V2New =0.42**

**V3New= V3old+**ΔV3

**V3New** = 0.55+ **0.013782847865**

**V3New = 0.563782847865**

**V4New= V4old+**ΔV4

**V4New** = (-0.17) + (0.0)

**V4New =** -0.17