Backpropagation algorithm

In [1]:
```
import numpy as np
x = np.array(([2,9],[1,5],[3,6]),dtype=float)
y = np.array(([92],[86],[89]),dtype=float)
x = x/np.amax(x,axis=0)
y = y/100
```

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

In [3]:
```
epoch=5000
lr=0.1
inputlayer_neurons = 2
hiddenlayer_neurons = 3
output_neurons = 1

wh=np.random.uniform(size=(inputlayer_neurons,hiddenlayer_neurons))
bh=np.random.uniform(size=(1,hiddenlayer_neurons))
wout=np.random.uniform(size=(hiddenlayer_neurons,output_neurons))
bout=np.random.uniform(size=(1,output_neurons))
```

In [4]:
```
for i in range(epoch):
    hinp1=np.dot(x,wh)
hinp=hinp1 + bh
hlayer_act = sigmoid(hinp)
outinp1=np.dot(hlayer_act,wout)
outinp= outinp1+ bout
output = sigmoid(outinp)
```

In [5]:
```
E0 = y-output
outgrad =derivatives_sigmoid(output)
d_output = E0* outgrad
EH = d_output.dot(wout.T)
hiddengrad = derivatives_sigmoid(hlayer_act)
d_hiddenlayer = EH * hiddengrad
wout += hlayer_act.T.dot(d_output) *lr
wh += x.T.dot(d_hiddenlayer) *lr
```

In [6]:
```
print("Input:\n" + str(x))
```

```
Input:
[[0.66666667 1.        ]
 [0.33333333 0.55555556]
 [1.         0.66666667]]
```

In [7]:
```
print("Actual Output: \n" + str(y))
```

```
Actual Output:
[[0.92]
 [0.86]
 [0.89]]
```

In [8]:

```
print("PredictedOutput: \n",output)
```

```
PredictedOutput: 
 [[0.8528972 ]
 [0.8434814 ]
 [0.85410258]]
```