In [77]:
```
import numpy as np
```

In [78]:
```
def sigmoid(x,deriv=False):
    if deriv:
        return x*(1-x)
    return 1/(1+np.exp(-x))
```

In [91]:
```
ones = np.ones((3,1))
X = np.array([[2,9],[1,5],[3,6]],dtype=float)
X = np.column_stack((ones,X))
y = np.array([[92],[86],[89]],dtype=float)/100

w1 = np.random.random((3,3))
w2 = np.random.random((3,1))

epoch = 100000
lr = 0.1

for i in range(epoch):
    z2 = X.dot(w1)
    a2 = sigmoid(z2)
    a2[:,0] = 1.0 # bias

    z3 = a2.dot(w2)
    a3 = sigmoid(z3)

    delta3 = a3-y
    delta2 = w2.T*delta3*sigmoid(a2,True)

    w2 -= lr * a2.T.dot(delta3)
    w1 -= lr * X.T.dot(delta2)

print("Input: \n{0}".format(X[:,1:]))
print("Actual: \n{0}".format(y))
print("Predicted: \n{0}".format(a3))


```

```
Input:
[[2. 9.]
 [1. 5.]
 [3. 6.]]
Actual:
[[0.92]
 [0.86]
 [0.89]]
Predicted:
[[0.92]
 [0.86]
 [0.89]]
```