AIM: Write a program to implement DeltaRule

DeltaRule

$$w(new) = w(old) + l.r * (desired - actual)$$

In [1]:
```
import numpy as np
```

In [2]:
```
x = np.zeros((3,))
```

In [3]:
```
weights = np.zeros((3,))
```

In [4]:
```
desired = np.zeros((3,))
```

In [5]:
```
actual = np.zeros((3,))
```

In [6]:
```
for i in range(0,3):
    x[i] = float(input("Initial Inputs: "))
```

```
Initial Inputs: 1
Initial Inputs: 1
Initial Inputs: 1
```

In [7]:
```
for i in range(0,3):
    weights[i] = float(input("Initial Weights: "))
```

```
Initial Weights: 1
Initial Weights: 1
Initial Weights: 1
```

In [8]:
```
for i in range(0,3):
    desired[i] = float(input("Desired Output: "))
```

```
Desired Output: 2
Desired Output: 3
Desired Output: 4
```

In [9]:
```
a = float(input("Enter learning rate = "))
```

```
Enter learning rate = 1
```

In [10]:
```
x
```

Out[10]:
```
array([1., 1., 1.])
```

In [11]:
```
weights
```

Out[11]:
```
array([1., 1., 1.])
```

In [12]:
```
desired
```

Out[12]:
```
array([2., 3., 4.])
```

In [13]:
```
actual = x*weights
```

In [14]:
```
print('Actual output',actual)
print('Desired output',desired)
```

```
Actual output [1. 1. 1.]
Desired output [2. 3. 4.]
```

In [15]:
```
while True:
    if np.array_equal(desired,actual):
        break
    else:
        for i in range(0,3):
            weights[i] = weights[i]+a*(desired[i]-actual[i])
            actual = x*weights
```

In [16]:
```
print('Final Output')
print('Corrected Weights',weights)
print('Actual output',actual)
print('Desired output',desired)
```

```
Final Output
Corrected Weights [2. 3. 4.]
Actual output [2. 3. 4.]
Desired output [2. 3. 4.]
```

In [ ]:
```

```