AND using MP Neuron

In [1]:

```
import numpy as np
x=np.array([[1,1],[1,0],[0,1],[0,0]])
t=np.array([[1],[0],[0],[0]])
w=np.array([[0],[0]])
theta=1
yin=np.zeros(shape=(4,1))
y=np.zeros(shape=(4,1))
yin=np.dot(x,w)
i=0
found=0
while(found==0):
    i=0
    yin=np.dot(x,w)
    #print(yin)
    while(i<4):
        if yin[i]>=theta:
            y[i]=1
            i=i+1
        else:
            y[i]=0
            i=i+1
    #print("y",y)
    #print("t",t)
    if (y==t).all():
        print("MODEL IS TRAINED ")
        print("\nOutput : \n",y)
        print("\nweights : ",w,"\n")
        print("theta : ",theta)
        found=1
    else:
        print("MODEL IS NOT TRAINED")
        w=np.zeros(shape=(0,0))
        theta=int(input("Enter New Theta : "))
        for k in range(int(2)):
            w1=int(input("Enter Weight : "))
            w=np.append(w,w1)
```

```
MODEL IS NOT TRAINED
Enter New Theta : 2
Enter Weight : 1
Enter Weight : 1
MODEL IS TRAINED

Output :
 [[1.]
 [0.]
 [0.]
 [0.]]

weights :  [1. 1.]

theta :  2
```

OR using MP Neuron

In [2]:

```
import numpy as np
x=np.array([[1,1],[1,0],[0,1],[0,0]])
t=np.array([[1],[1],[1],[0]])
w=np.array([[0],[0]])
theta=1
```

```
yin=np.zeros(shape=(4,1))
y=np.zeros(shape=(4,1))
yin=np.dot(x,w)
i=0
found=0
while(found==0):
    i=0
    yin=np.dot(x,w)
    #print(yin)
    while(i<4):
        if yin[i]>=theta:
            y[i]=1
            i=i+1
        else:
            y[i]=0
            i=i+1
    #print("y",y)
    #print("t",t)
    if (y==t).all():
        print("MODEL IS TRAINED ")
        print("\nOutput : \n",y)
        print("\nweights : ",w,"\n")
        print("theta : ",theta)
        found=1
    else:
        print("MODEL IS NOT TRAINED")
        w=np.zeros(shape=(0,0))
        theta=int(input("Enter New Theta : "))
        for k in range(int(2)):
            w1=int(input("Enter Weight : "))
            w=np.append(w,w1)
```

```
MODEL IS NOT TRAINED
Enter New Theta : 1
Enter Weight : 1
Enter Weight : 1
MODEL IS TRAINED

Output :
 [[1.]
 [1.]
 [1.]
 [0.]]

weights :  [1. 1.]

theta :  1
```

AND-NOT using MP Neuron

In [4]:

```
import numpy as np
x=np.array([[1,1],[1,0],[0,1],[0,0]])
t=np.array([[0],[1],[0],[0]])
w=np.array([[0],[0]])
theta=1
yin=np.zeros(shape=(4,1))
y=np.zeros(shape=(4,1))
yin=np.dot(x,w)
i=0
found=0
while(found==0):
    i=0
    yin=np.dot(x,w)
    #print(yin)
    while(i<4):
        if yin[i]>=theta:
            y[i]=1
            i=i+1
```

```
        else:
            y[i]=0
            i=i+1
    #print("y",y)
    #print("t",t)
    if (y==t).all():
        print("MODEL IS TRAINED ")
        print("\nOutput : \n",y)
        print("\nweights : ",w,"\n")
        print("theta : ",theta)
        found=1
    else:
        print("MODEL IS NOT TRAINED")
        w=np.zeros(shape=(0,0))
        theta=int(input("Enter New Theta : "))
        for k in range(int(2)):
            w1=int(input("Enter Weight : "))
            w=np.append(w,w1)
```

```
MODEL IS NOT TRAINED
Enter New Theta : 1
Enter Weight : 1
Enter Weight : -1
MODEL IS TRAINED

Output :
 [[0.]
 [1.]
 [0.]
 [0.]]

weights :  [ 1. -1.]

theta :  1
```

NOT using MP Neuron

In [2]:

```
import numpy as np
x=np.array([[0],[1]])
t=np.array([[1],[0]])
w=np.array([0])
theta=1
yin=np.zeros(shape=(2,1))
y=np.zeros(shape=(2,1))
yin=np.dot(x,w)
i=0
found=0
while(found==0):
    i=0
    yin=np.dot(x,w)
    #print(yin)
    while(i<2):
        if yin[i]>=theta:
            y[i]=1
            i=i+1

            #if(i==4):
                #break
        else:
            y[i]=0
            i=i+1
    #print("y",y)
    #print("t",t)
    if (y==t).all():
        print("MODEL IS TRAINED ")
        print("\nOutput : \n",y)
        print("\nweights : ",w,"\n")
        print("theta : ",theta)
```

```
        found=1
    else:
        print("MODEL IS NOT TRAINED")
        w=np.zeros(shape=(0,0))
        theta=int(input("Enter New Theta : "))
        for k in range(int(1)):
            w=int(input("Enter Weight : "))
```

```
MODEL IS NOT TRAINED
Enter New Theta : 0
Enter Weight : -1
MODEL IS TRAINED

Output :
 [[1.]
 [0.]]

weights :  -1

theta :  0
```

In [ ]: