In [1]:

```
"""
Write a program to train a two input ADLAINE network that uses gradient
descent algorithm to minimise sum squared output error by updating the
weights in each epoch.
a. Train the ADALINE to implement OR and AND Gate. In both the cases
train until there is no output error. Check for convergence.
b. Plot the error for different epochs (error vs. epoch curve).
c. Train the ADALINE to implement X-OR gate and check for its
accuracy.
d. In each of the above cases plot the resulting decision boundary along
with the samples.
"""
import matplotlib.pyplot as plt
import numpy as np
def function(arr,d,w1,w2,b,alpha,th):
    error=[]
    a=[]
    def f(error):
        if(len(error)==4):
            if(all(e == 0 for e in error)):
                return 1
            return 0
        return 0
    epoch=1
    s=[0,0,0,0]
    t=[0,0,0,0]
    while(f(error)==0):
        for i in range(len(arr)):
            s[i]=b+arr[i][0]*w1+arr[i][1]*w2
            if(s[i]>=0):
                y=1
            else:
                y=0
            t[i]=d[i]-y
            if(t[i]!=0):
                w1n=w1+alpha*t[i]*arr[i][0]
                w2n=w2+alpha*t[i]*arr[i][1]
                bn=b+alpha*t[i]
                print("|\t",epoch,"\t",i+1,"\t",w1,"\t",w2,"\t",b,"\t",d[i],"\t",y,"\t",t[i],"\t",w1n,"\t",w2n,"\t",bn,"\t|")
                w1=w1n
                w2=w2n
                b=bn
```

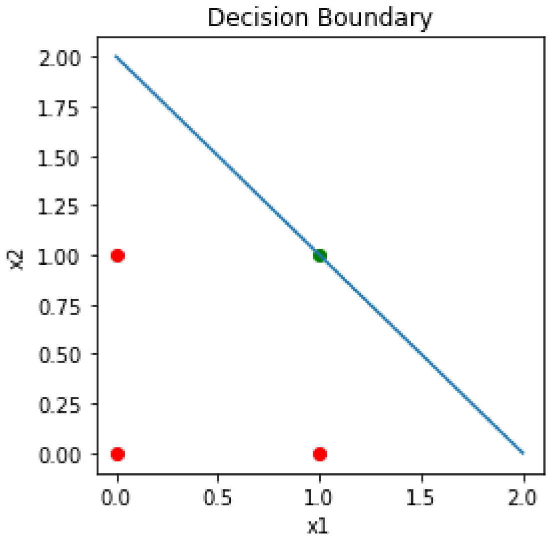
```
        else:
            print("|\t",epoch,"\t",i+1,"\t",w1,"\t",w2,"\t",b,"\t",d[i],"\t",y,"\t",t[i],"\t",w1,"\t",w2,"\t",b,"\t|")
            error.append(0)
    mse=[x*x for x in t]
    r=sum(mse)
    a.append(r)
    if(d==[0,1,1,0]):
        if(epoch==5):
            break
        else:
            epoch+=1
            print("------------------------------------------------------------------------------------------------")
            error.clear()
    else:
        if(f(error)):
            break
        else:
            epoch+=1
            print("------------------------------------------------------------------------------------------------")
            error.clear()
if(f(error)):
    print("converges")
else:
    print("diverges")
if(d==[0,1,1,0]):
    c=error.count(0)
    accuracy=c*100/epoch
    print(accuracy)
x1=[]
x2=[]
for i in range(len(arr)):
    x1.append(arr[i][0])
    x2.append(arr[i][1])
l=len(arr)
def decisionboundary(l,x1,x2,d,w1,w2):
    plt.figure(figsize=(4,4))
    plt.title("Decision Boundary")
    for i in range(4):
        if d[i]==1:
            color="g"
        if d[i]==0:
            color="r"
        plt.scatter(x1[i],x2[i],c=color)
    x=np.linspace(0,2,4)
```

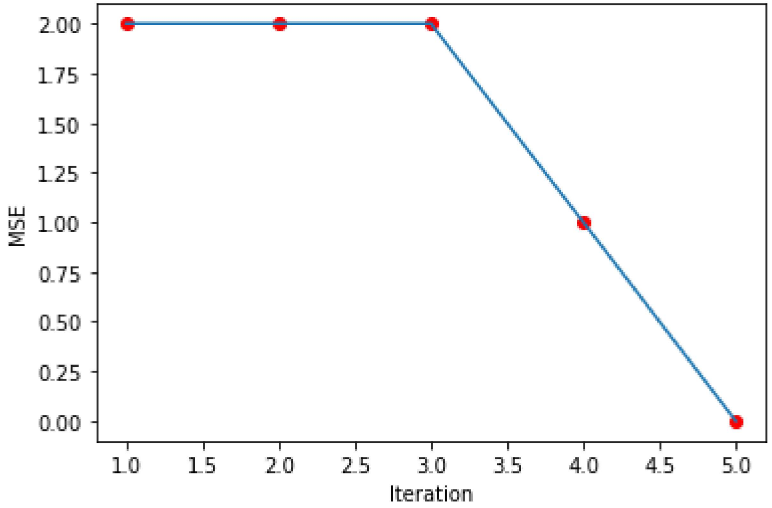
```
        y=-x+th
        plt.plot(x,y)
        plt.xlabel('x1')
        plt.ylabel('x2')
        plt.show()
    decisionboundary(l,x1,x2,d,w1,w2)
    def mse(a):
        plt.plot(list(range(1,len(a)+1)),a, 'ro')
        plt.plot(list(range(1,len(a)+1)),a)
        plt.ylabel('MSE')
        plt.xlabel('Iteration')
        plt.show()
    mse(a)
```

In [2]:

```
arr=[[0,0],[0,1],[1,0],[1,1]]
d=[0,0,0,1]
w1,w2,b,alpha,th=-1,1,-0.5,0.5,2
function(arr,d,w1,w2,b,alpha,th)
```

```
|        1       1       -1      1       -0.5    0       0       0       -1      1       -0.5    |
|        1       2       -1      1       -0.5    0       1       -1      -1.0    0.5     -1.0    |
|        1       3       -1.0    0.5     -1.0    0       0       0       -1.0    0.5     -1.0    |
|        1       4       -1.0    0.5     -1.0    1       0       1       -0.5    1.0     -0.5    |
-------------------------------------------------------------------------------------------------
|        2       1       -0.5    1.0     -0.5    0       0       0       -0.5    1.0     -0.5    |
|        2       2       -0.5    1.0     -0.5    0       1       -1      -0.5    0.5     -1.0    |
|        2       3       -0.5    0.5     -1.0    0       0       0       -0.5    0.5     -1.0    |
|        2       4       -0.5    0.5     -1.0    1       0       1       0.0     1.0     -0.5    |
-------------------------------------------------------------------------------------------------
|        3       1       0.0     1.0     -0.5    0       0       0       0.0     1.0     -0.5    |
|        3       2       0.0     1.0     -0.5    0       1       -1      0.0     0.5     -1.0    |
|        3       3       0.0     0.5     -1.0    0       0       0       0.0     0.5     -1.0    |
|        3       4       0.0     0.5     -1.0    1       0       1       0.5     1.0     -0.5    |
-------------------------------------------------------------------------------------------------
|        4       1       0.5     1.0     -0.5    0       0       0       0.5     1.0     -0.5    |
|        4       2       0.5     1.0     -0.5    0       1       -1      0.5     0.5     -1.0    |
|        4       3       0.5     0.5     -1.0    0       0       0       0.5     0.5     -1.0    |
|        4       4       0.5     0.5     -1.0    1       1       0       0.5     0.5     -1.0    |
-------------------------------------------------------------------------------------------------
|        5       1       0.5     0.5     -1.0    0       0       0       0.5     0.5     -1.0    |
|        5       2       0.5     0.5     -1.0    0       0       0       0.5     0.5     -1.0    |
|        5       3       0.5     0.5     -1.0    0       0       0       0.5     0.5     -1.0    |
|        5       4       0.5     0.5     -1.0    1       1       0       0.5     0.5     -1.0    |
converges
```



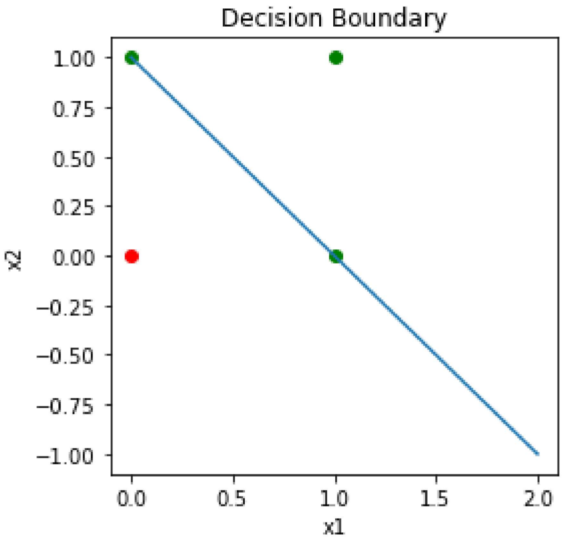


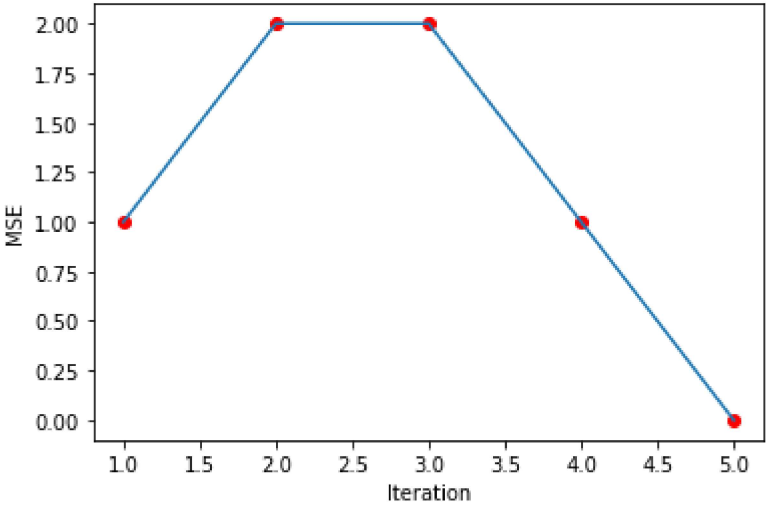
In [3]:

```
arr=[[0,0],[0,1],[1,0],[1,1]]
d=[0,1,1,1]
w1,w2,b,alpha,th=-0.2,0.2,-0.1,0.1,1
function(arr,d,w1,w2,b,alpha,th)
```

```
|        1        1        -0.2     0.2      -0.1     0        0        0        -0.2     0.2      -0.1     |
|        1        2        -0.2     0.2      -0.1     1        1        0        -0.2     0.2      -0.1     |
|        1        3        -0.2     0.2      -0.1     1        0        1        -0.1     0.2      0.0      |
|        1        4        -0.1     0.2      0.0      1        1        0        -0.1     0.2      0.0      |
```

```
---------------------------------------------------------------------------------------------
|        2       1       -0.1    0.2     0.0     0       1       -1      -0.1    0.2     -0.1   |
|        2       2       -0.1    0.2     -0.1    1       1       0       -0.1    0.2     -0.1   |
|        2       3       -0.1    0.2     -0.1    1       0       1       0.0     0.2     0.0    |
|        2       4       0.0     0.2     0.0     1       1       0       0.0     0.2     0.0    |
---------------------------------------------------------------------------------------------
|        3       1       0.0     0.2     0.0     0       1       -1      0.0     0.2     -0.1   |
|        3       2       0.0     0.2     -0.1    1       1       0       0.0     0.2     -0.1   |
|        3       3       0.0     0.2     -0.1    1       0       1       0.1     0.2     0.0    |
|        3       4       0.1     0.2     0.0     1       1       0       0.1     0.2     0.0    |
---------------------------------------------------------------------------------------------
|        4       1       0.1     0.2     0.0     0       1       -1      0.1     0.2     -0.1   |
|        4       2       0.1     0.2     -0.1    1       1       0       0.1     0.2     -0.1   |
|        4       3       0.1     0.2     -0.1    1       1       0       0.1     0.2     -0.1   |
|        4       4       0.1     0.2     -0.1    1       1       0       0.1     0.2     -0.1   |
---------------------------------------------------------------------------------------------
|        5       1       0.1     0.2     -0.1    0       0       0       0.1     0.2     -0.1   |
|        5       2       0.1     0.2     -0.1    1       1       0       0.1     0.2     -0.1   |
|        5       3       0.1     0.2     -0.1    1       1       0       0.1     0.2     -0.1   |
|        5       4       0.1     0.2     -0.1    1       1       0       0.1     0.2     -0.1   |
converges
```



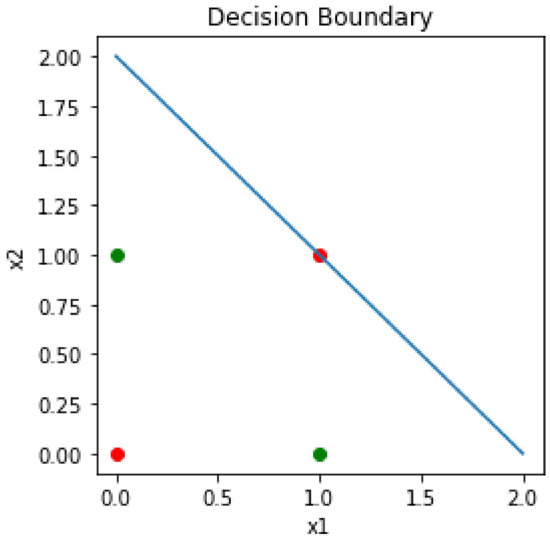


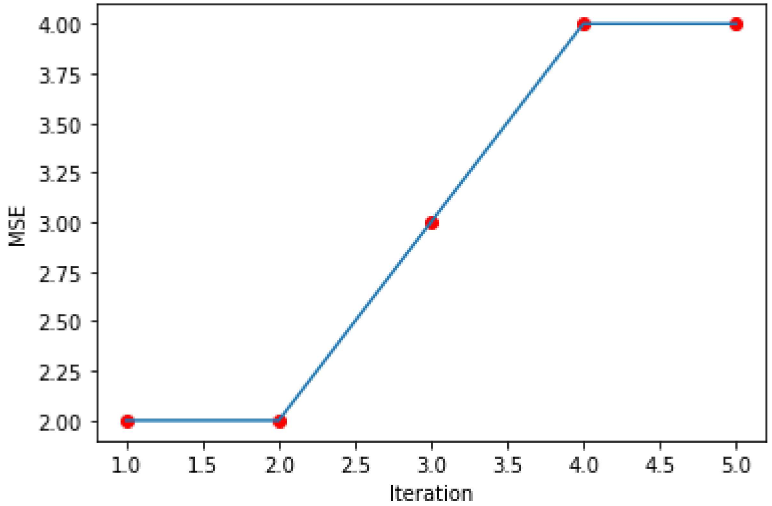
In [4]:

```
arr=[[0,0],[0,1],[1,0],[1,1]]
d=[0,1,1,0]
w1,w2,b,alpha,th=-1,1,-0.5,0.5,2
function(arr,d,w1,w2,b,alpha,th)
```

```
|       1       1       -1      1       -0.5    0       0       0       -1      1       -0.5    |
|       1       2       -1      1       -0.5    1       1       0       -1      1       -0.5    |
|       1       3       -1      1       -0.5    1       0       1       -0.5    1.0     0.0     |
|       1       4       -0.5    1.0     0.0     0       1       -1      -1.0    0.5     -0.5    |
-------------------------------------------------------------------------------------------------
|       2       1       -1.0    0.5     -0.5    0       0       0       -1.0    0.5     -0.5    |
|       2       2       -1.0    0.5     -0.5    1       1       0       -1.0    0.5     -0.5    |
|       2       3       -1.0    0.5     -0.5    1       0       1       -0.5    0.5     0.0     |
|       2       4       -0.5    0.5     0.0     0       1       -1      -1.0    0.0     -0.5    |
-------------------------------------------------------------------------------------------------
|       3       1       -1.0    0.0     -0.5    0       0       0       -1.0    0.0     -0.5    |
|       3       2       -1.0    0.0     -0.5    1       0       1       -1.0    0.5     0.0     |
|       3       3       -1.0    0.5     0.0     1       0       1       -0.5    0.5     0.5     |
|       3       4       -0.5    0.5     0.5     0       1       -1      -1.0    0.0     0.0     |
-------------------------------------------------------------------------------------------------
|       4       1       -1.0    0.0     0.0     0       1       -1      -1.0    0.0     -0.5    |
|       4       2       -1.0    0.0     -0.5    1       0       1       -1.0    0.5     0.0     |
|       4       3       -1.0    0.5     0.0     1       0       1       -0.5    0.5     0.5     |
|       4       4       -0.5    0.5     0.5     0       1       -1      -1.0    0.0     0.0     |
-------------------------------------------------------------------------------------------------
|       5       1       -1.0    0.0     0.0     0       1       -1      -1.0    0.0     -0.5    |
|       5       2       -1.0    0.0     -0.5    1       0       1       -1.0    0.5     0.0     |
```

```
|        5       3       -1.0    0.5     0.0     1       0       1       -0.5    0.5     0.5     |
|        5       4       -0.5    0.5     0.5     0       1       -1      -1.0    0.0     0.0     |
diverges
0.0
```






```
In [ ]:
```