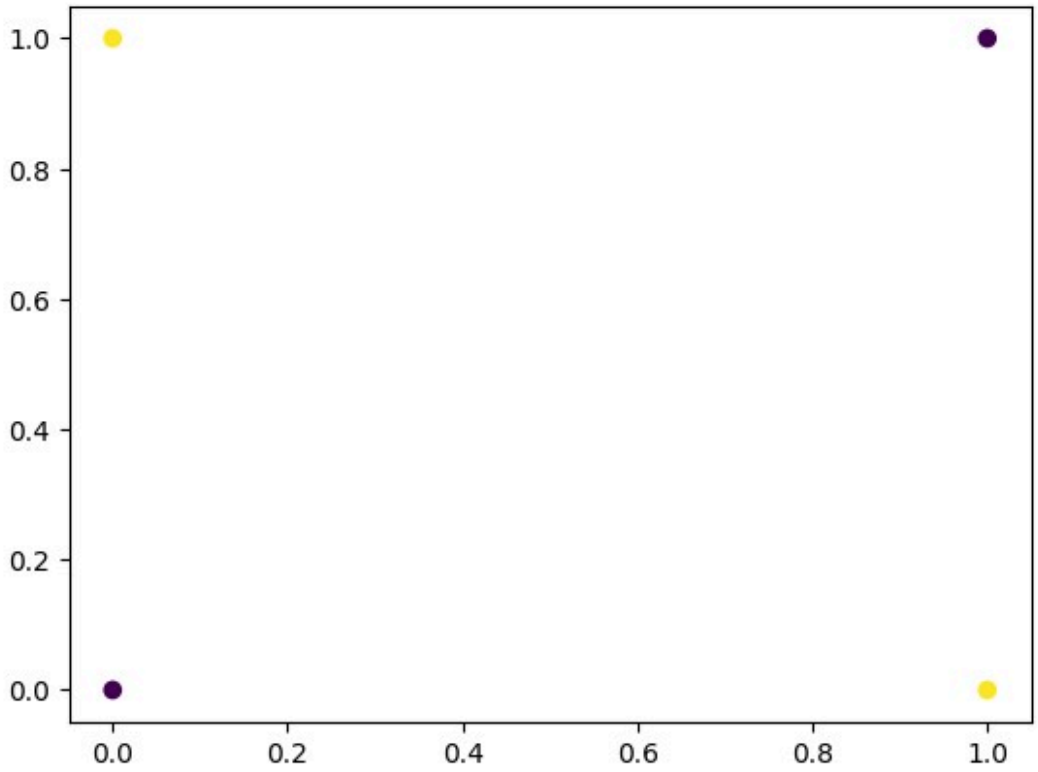
Experiment 10

Name : Shrey Chaudhari

Roll no: 24bee094

```
import numpy as np
from matplotlib import pyplot as plt
#XOR truth table
X=np.array([[0,0],[0,1],[1,0],[1,1]])
Y=np.array([0,1,1,0])
#scatter plot. Notice data points are not linearly separable
#and therefore a network without a hidden layer, can't learn to
separate them
plt.scatter(x=X[:,0],y=X[:,1],c=Y)
plt.show()
#still we will try and demonstrate
n_samples=X.shape[0]
n_features=X.shape[1]
#initial weights and bias are random
w=np.random.uniform(0,1,size=n_features)
b=np.random.uniform(0,1,1)
#Scan number of epochs
n_epoch=int(input("Enter Number of epochs:"))
#learning rate
lr=0.01
for e in range(n_epoch): #epoch loop
      for s in range(n_samples): #Update after every presentation, so
can't do away with this loop
            net=np.dot(X[s,:],w)+b
            if net >=0:
            a=1
            else:
                  a=0
            error=Y[s]-a
            w=w+lr*error*X[s,:]
            b=b+lr*error
#now calculate the slope and the intercept of the decision boundary
#AX+BY+C=0 => slope=-A/B and intercept=-C/B (from y=mx+c)
m=-w[0]/w[1]
c=-b/w[1]
#function to plot decison boundary along with train points
#notice that the line is not able to separate data points
def plot_decision_boundary(X):
      for x in np.linspace(np.min(X[:,0]),np.max(X[:,0])): #default
      #50 points are generated by linspace
            y=m*x+c
            plt.plot(x,y,linestyle='-', color='k', marker='.')
      plt.scatter(X[:,0], X[:,1], c=Y)
      plt.show()
plot_decision_boundary(X)
```




```
Enter Number of epochs:5
```