```
import pandas as pd
import numpy as np

def activation(A):
        for i in range(A.shape[0]):
                if(A[i]>0):
                        A[i]=1
                else:
                        A[i]=0
        return A
def activation2(A):
        for i in range(A.shape[0]):
                if(A[i][0]>0):
                        A[i][0]=1
                else:
                        A[i][0]=0
        return A

def main():

        files=['IRIS.csv','SPECT.csv','SPECTF.csv']

        for file in files:
                print("Using ",file," dataset")
                df1=pd.read_csv(file)
        #print(df1)
                df1["class"]=df1["class"].astype('category')
                df1["class_cat"]=df1["class"].cat.codes
                df1=df1.drop(columns=['class'])

                df1=df1.values#convert dataframe to numpy matrix
                np.random.shuffle(df1)
                X=df1[:,:-1]
                Y=df1[:,-1]
                I=np.ones((X.shape[0],1))
                X=np.concatenate((I,X),axis=1)
                m=X.shape[0]
                Y=Y.reshape((m,1))

                sz_test=(int)(m/10)
                sz_train=m-sz_test

                alpha=0.1
                while(alpha<=1.0):
                        print("Using learning rate : ",alpha)
                        precision=0
                        recall=0
                        accuracy=0
                        fp,fn,tp,tn=0,0,0,0
                        for i in range(10):
                                i1=(i*sz_test)
                                i2=((i+1)*sz_test)
                                X_test,X_train = X[i1:i2,:],np.concatenate((X[0:i1,:],X[i2:
m,:]),axis=0)
                                Y_test,Y_train = Y[(i*sz_test):((i+1)*sz_test),:],np.concat
enate((Y[0:i1,:],Y[i2:m,:]),axis=0)
        #X_train.shape

                                W=np.random.rand(X_train.shape[1],1)
                #threshold=0

                                ex_no=0
                                for i in range(500):
                                        prod=np.dot(X_train[ex_no,:],W)
                                        a=activation(np.dot(X_train[ex_no,:],W))
                                        J=Y_train[ex_no]-a[0]
                                        change=(alpha*J)*X_train[ex_no,:]
                                        change=change.reshape(X.shape[1],1)
```

```
                                        W=np.add(W,change)
                                        ex_no=(ex_no+1)%sz_train

                                prod=np.dot(X_test,W)
                        #print("Shape is ",prod.shape)

                                a=activation2(prod)

                                for i in range(a.shape[0]):
                                #print(a[i]," ",Y_test[i],"\n")
                                        if(a[i][0]==0 and Y_test[i]==0):
                                                tn+=1
                                        elif(a[i][0]==0 and Y_test[i]==1):
                                                fn+=1
                                        elif(a[i][0]==1 and Y_test[i]==0):
                                                fp+=1
                                        elif(a[i][0]==1 and Y_test[i]==1):
                                                tp+=1

                        #print("FP, TP, FN, TN ",fp,tp,fn,tn)

                                precision=tp/(tp+fp)
                                recall=tp/(tp+fn)
                                accuracy=(tp+tn)/(tp+tn+fp+fn)

                                print("Accuracy, Precision and Recall : ",accuracy," ",precision,"
",recall,"\n\n")
                                alpha+=0.1


if __name__=='__main__':
        main()
```