# Assignment_1

# Batch Perceptron and Online Training Algorithms

**Name: Omar Hesham Abdelfatah Muhammed**

**Code: 1170065**

**Submitted to: TA, Peter**

# I. Batch Perceptron Algorithm:

a) In this algorithm we have given data x,y where each element in x corresponds to a (-1 or 1) in y , so we start weights and deltas of zeros of the same length to start looping on data to reach zero error which is delta.
b) In this algorithm delta is updated in each invalid condition in for loop while weight is updated after finishing each loop (ephoch).
c) When delta reaches zero the function ends and returns final weights , delta array which is change in weights , weights array and finally epochs count.
d) Following figure shows code of this algorithm.

```
def batch_perceptron(x,y,lr=1):
    n = x.shape[1]
    epochs_count = 0
    weights_array = []                    #initializing weights array
    delta_array = []                      #initializing delta_array which carries weights changes
                    #initializing weights with random number from -1 to 1
    weights = np.zeros(n)
    for i in range(0,n):
        weights[i] = random.random()*random.randint(-1,1)

    delta = np.ones(n)                   #initializing delta with ones to enter while loop
    e = sys.float_info.epsilon
    while (norm(delta,1) > e):
        delta = np.zeros(n)              #giving detla zero values at the begining of each loop
        for i in range(0,len(y)):
            if(y[i]*(weights.dot(x[i])) <= 0):          #check condition if they have the same sign or not
                delta = delta - y[i]*x[i]
        delta = delta / len(y)
        delta_array.append(delta)
        weights = weights - (lr * delta)                #updating weights by delta
        weights_array.append(weights)                   #filling weight array with weights updated
        epochs_count = epochs_count + 1                 #counting ephocs "complete loops"
    return weights,delta_array,weights_array,epochs_count
```

# II. Online Training Algorithm:

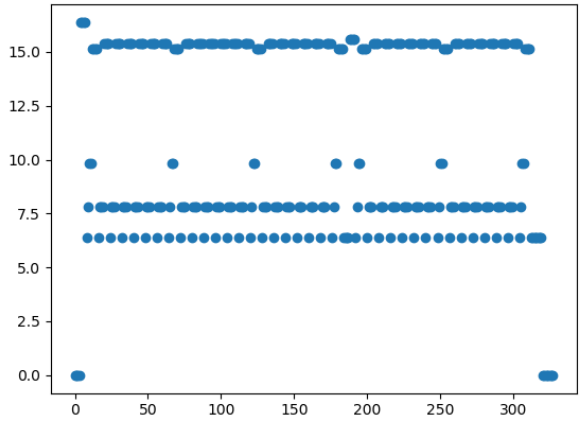
a) In this algorithm we have given data x,y where each element in x corresponds to a (-1 or 1) in y , so we start weights and deltas of zeros of the same length to start looping on data to reach zero error which is delta.
b) In this algorithm delta and weight are updated in each invalid condition in for loop.
c) When delta reaches zero the function ends and returns final weights , delta array which is change in weights , weights array and finally epochs count.
d) Following figure shows code of this algorithm.

```
def online_training(x,y,lr=1):
    n = x.shape[1]
    epochs_count = 0
    weights_array = []          #initializing weights array
    delta_array = []            #initializing delta_array which carries weights changes
    #initializing weights with random number from -1 to 1
    weights = np.zeros(n)
    for i in range(0,n):
        weights[i] = random.random()*random.randint(-1,1)

    delta = np.ones(n)         #initializing delta with ones to enter while loop
    e = sys.float_info.epsilon
    while (norm(delta,1) > e):
        delta = np.zeros(n)    #giving detla zero values at the begining of each loop
        for i in range(0,len(y)):
            if(y[i]*(weights.dot(x[i])) <= 0):          #check condition if they have the same sign or not
                delta = delta - y[i]*x[i]
                delta = delta / len(y)

                weights = weights - (lr * delta)     #updating weights by delta
                weights_array.append(weights)        #filling weight array with weights updated
            delta_array.append(norm(delta,1))

        epochs_count = epochs_count + 1             #counting ephocs "complete loops"
    return weights,delta_array,weights_array,epochs_count
```

## Problem No.1 output parameters and plots:

### a) Online Training:

Delta norm ouput:

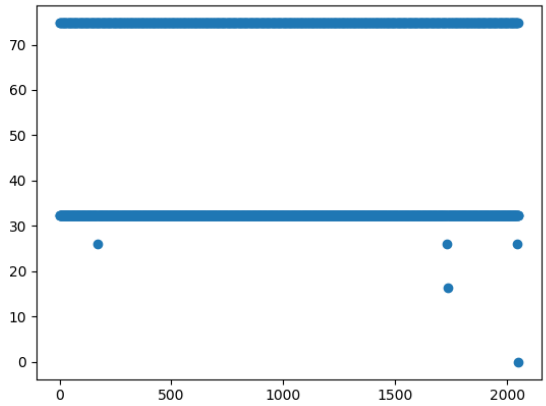



other parameters:

```
Parameters for online_training method for problem No.1
Number of epochs = 41
Number of times model weights are updated = 122
```

### b) Batch Perceptron:

Delta norm output:



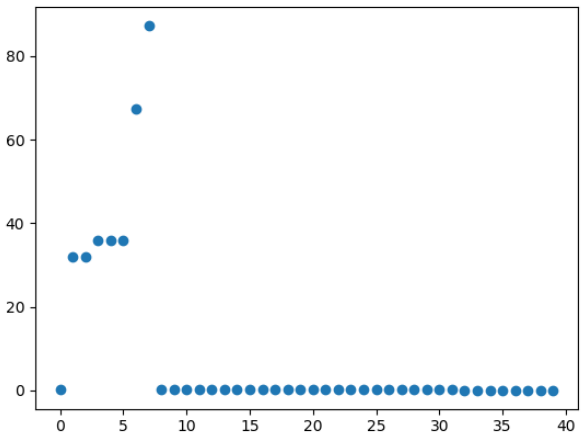

other parameters:

```
Parameters for batch_perceptron method for problem No.1
Number of epochs = 2053
Number of times model weights are updated = 2053
```

## Problem No.4 output parameters and plots:

### a) Online Training:

Delta norm output

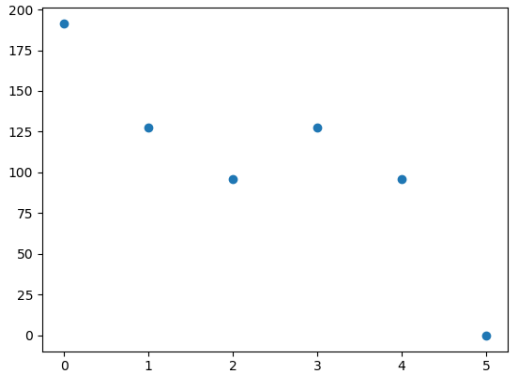



other parameters:

```
Parameters for online_training method for problem No.4
Number of epochs = 5
Number of times model weights are updated = 8
```

b) Batch Perceptron

Delta output:




other paramters:

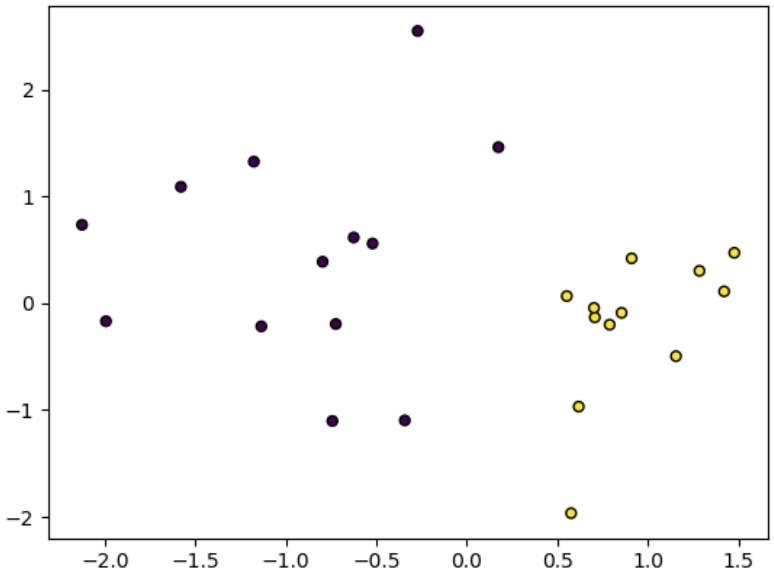
```
Parameters for online_training method for problem No.4
Number of epochs = 6
Number of times model weights are updated = 6
```

## Make_Classification function:

- This function generates random values for x,y with given number of features for x and (1,-1) values for y to test these random inputs for online training and batch perceptron algorithms

```
x,y = make_classification(25,n_features=2,n_redundant = 0,n_informative=1,n_clusters_per_class=1)
mask_for_y = y == 0
y[mask_for_y] = -1
plt.scatter(x[:,0],x[:,1],marker='o',c=y,s=25,edgecolor='k')
plt.show()
```

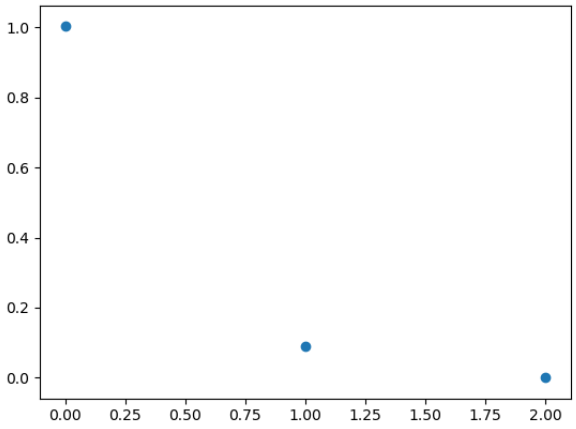
This is the output of plotting these random values



## Mask Classifications output parameters and plots:

### a) Batch Preceptron:

Delta norm ouput:

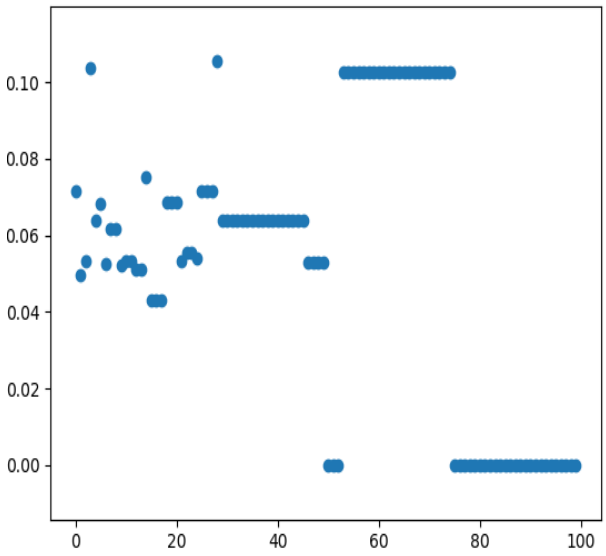



other parameters:

```
Parameters for batch preceptron method for mask classification
Number of epochs = 3
Number of times model weights are updated = 3
```

### b) Online Training:

Delta norm output:

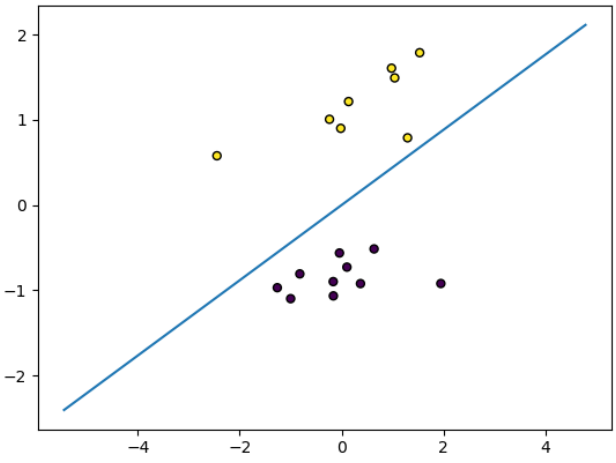



other parameters:

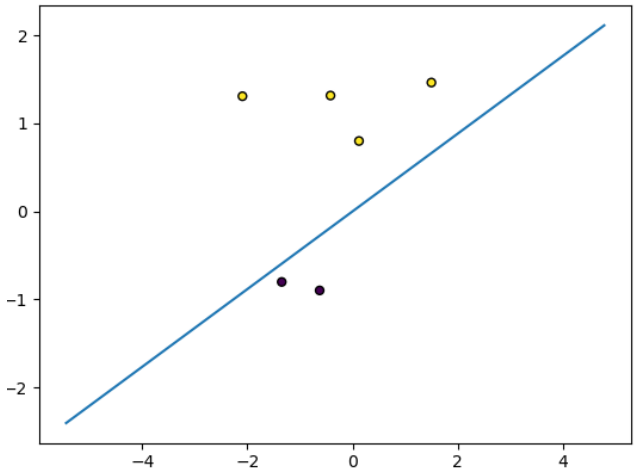
```
Parameters for online training method for mask classification
Number of epochs = 4
Number of times model weights are updated = 22
```

## Visualizing the model on the plot:

Training Data




Testing Data




Calculating accuracy:

-Following code is how accuracy is calculated:

```
count = 0
for i in range(0,len(x_test)):

    if(y[i+training_len]*(w.dot(x_test[i])) >=0):
        count=count+1
accuracy = count/len(x_test)
print("accuracy = " + str(accuracy))
```

- Accuracy in this example was : 50%