EEE 591 Machine Learning with deployment to FPGA

HOMEWORK 2
ASU ID: 1225713099
Praveen Paidi

1.

Compact notation for perceptron is:

g(b,w) = $\frac{1}{P}\sum_{p=1}^{p}\max\left(0, y_p\left(X_p^T w\right) - y_p\right)^n$

The Perceptron loss is not differentiable, but it can be used for the gradient descent.
Perceptron is linear which means that hyperplane is separating the classes.

For a differentiable convex function $g(x)$ its gradients are linear underestimators

If a convex function is not differentiable at a point x, it can have many sub gradients

ReLU is not a complete linear function, it is piecewise linear function , which means that, below zero it is differentiable and above 0 it is not differentiable but not at 0.

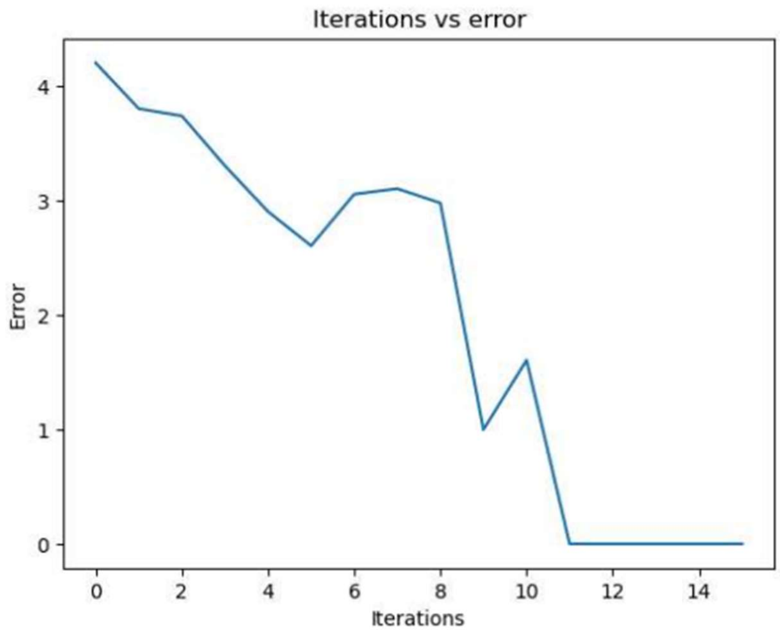
g ' (w) = 0 if $X_p^T w < 0$,
1 if $X_p^T w > 0$,

ReLU is actually not differentiable at x = 0, but it has subdifferential [0,1]. Any value in that interval can be taken as a sub derivative

2. Perceptron code.

Results:

```
Iteration 1
 weights  [0.61627887 0.18248128 0.50961002 0.28188752]
 error  -2.4484990856987876
prediction [1. 1. 1. 1. 1.]

Iteration 2
 weights  [0.51627887 0.08248128 0.60961002 0.38188752]
 error  -1.8484990856987877
prediction [1. 1. 1. 1. 1.]

Iteration 3
 weights  [0.41627887 0.08248128 0.50961002 0.48188752]
 error  -1.448499085698788
prediction [1. 1. 1. 1. 1.]

Iteration 4
 weights  [ 0.31627887 -0.01751872  0.60961002  0.58188752]
 error  -0.8484990856987882
prediction [1. 1. 1. 1. 1.]

Iteration 5
 weights  [ 0.21627887 -0.11751872  0.70961002  0.68188752]
 error  -0.2484990856987883
prediction [ 1. -1.  1.  1.  1.]

Iteration 6
 weights  [0.11627887 0.08248128 0.30961002 0.78188752]
 error  -0.4484990856987884
prediction [ 1. -1.  1.  1.  1.]

Iteration 7
 weights  [0.11627887 0.58248128 0.60961002 0.78188752]
 error  0.1515009143012116
prediction [1. 1. 1. 1. 1.]

Iteration 8
 weights  [0.01627887 0.48248128 0.70961002 0.88188752]
 error  -3.4484990856987894
prediction [1. 1. 1. 1. 1.]

Iteration 9
 weights  [-0.18372113  0.28248128  0.20961002  1.08188752]
 error  -0.8484990856987887
prediction [-1.  1.  1.  1.  1.]

Iteration 10
 weights  [-0.18372113 -0.01751872  0.70961002  1.08188752]
 error  -0.0484990856987888
prediction [ 1. -1.  1.  1.  1.]

Iteration 11
 weights  [-0.28372113  0.18248128  0.30961002  1.18188752]
 error  -0.2484990856987886
prediction [-1. -1.  1.  1.  1.]
```



3.

Perceptron Softmax cost function:

Cost function :

g(b,w) = $\frac{1}{P}\sum_{p=1}^{p}\max\left(0, y_p\left(X_p^T w\right) - y_p\right)^n$

softmax function:

$$\sigma_{(z)_i} = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_i}}$$

softmax (s1, s2) = log ($e^{s1} + e^{s2}$)

Applying softmax function to cost function results in :

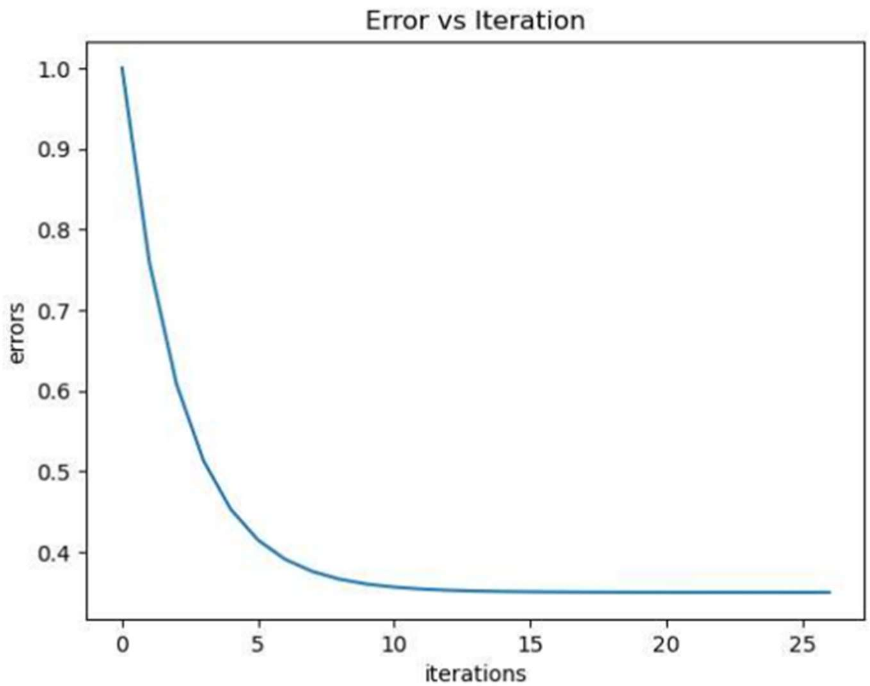
$$\frac{\partial g(w)}{\partial w} = \sum \frac{-1}{1 + e^{(-y_p(X_p^T W)}} \cdot y_p X_p^T$$

If we define the sigmoid function :

$$\frac{1}{1 + e^{-z}}$$

V(g(w)) = -$\sum sigmoid\ (-y_p\left(X_p^T W\right). y_p X_p^T$

4. Results of Adaline function:




```
Q4. DATASET1
 Weights  [-0.03919539  0.3551497  -0.67420292  0.21692546]
 No. of Iterations: 27

 For training data
 Misclassified observations: 8
 Total Error  32.0
 Accuracy:  0.98

 For test data
 Misclassified observations: 6
 Total Error  24.0
 Accuracy:  0.96
```

5.

Directly using the sklearn function:

Results from the sklearn function:

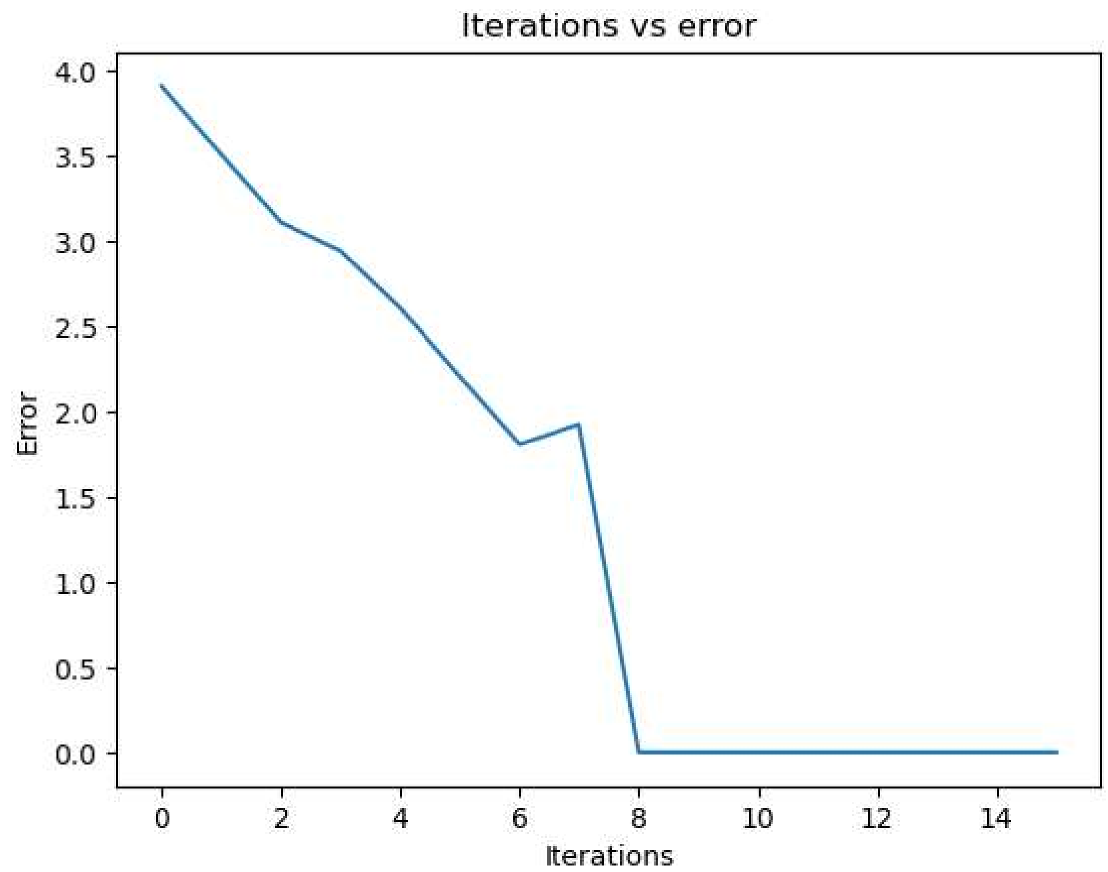
```
FROM SKLEARN

Q5 Data set 1

 Training Data
Misclassified samples: 5
 Train Accuracy: 0.99

 Testing data
Misclassified samples: 4
Accuracy: 0.98

 Data set 2

 Training data
Misclassified samples: 3
Train Accuracy: 0.99

 testing data
Misclassified samples: 1
Test Accuracy: 0.99
```

In [3]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score
import pandas as pd

X = np.array([[-2,4,-1],[4,1,-1],[1,6,-1],[2,4,-1],[6,2,-1]])
Y = np.array([-1,-1,1,1,1])
alpha = 0.1
errors = []
converged= False
count=0
misclassification=0
S=len(X[:,0])
ones= np.ones(S,dtype=float).reshape(S,1)
A=np.hstack((ones, X))
w = np.random.rand(len(A[0]))

print('Q2')
while not converged:
    count+=1
    total_error = 0.0

    for i in range(len(Y)):
        error = -np.dot(A[i], w) * Y[i]
        if (error >= 0):
            w = w + alpha*A[i]*Y[i]
            total_error += error
    ypred = np.dot(A,w)
    errors.append(total_error)

    for i in range (len(ypred)):
        if ypred[i]<0:
            ypred[i]=-1
        else:
            ypred[i]=1

    if count>15:
        converged=True
    print( '\nIteration',count,'\n weights ', w, '\n error ', error, '\nprediction' ,ypred)

plt.plot(errors)
plt.xlabel('Iterations')
plt.ylabel('Error')
plt.title('Iterations vs error')
plt.show()
print('Accuracy of model: %.2f' % accuracy_score(Y, ypred))
for i in range(len(Y)):
    if ypred[i]!=Y[i]:
        misclassification+=1
print('Number of misclassification predictions are' ,misclassification)


from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split

df_adaline = pd.read_csv(r'D:\Spring 23\EEE 591\HW2\Dataset_1.csv')
X_train1 = df_adaline.iloc[:,0:3].values
Y_train1 = df_adaline['y'].values
stdslr = StandardScaler()
X_trainstd1 = stdslr.fit_transform(X_train1[:,0:3])
one = np.ones(len(X_trainstd1[:,0]),dtype=float)
# compact notation
A_a1 = np.column_stack((one,X_trainstd1))

X_train1, X_test1, Y_train1, Y_test1 = train_test_split(A_a1,Y_train1,test_size=0.33,random_state=0)
no_of_obs = len(Y_train1)

def adaline(X, w, Y, eta):
    errors = []
    total_error = 1.0
    last_total_error = 0
    A = X
    count=0
    converged= False
    while not converged:
        y_pred = np.dot(A,w)
        Error = y_pred - Y
        w = w - eta *(2/no_of_obs)*np.dot(A.T,Error)
        total_error = np.dot(Error,Error)
        errors.append(total_error/len(Y))
        count += 1
        if count>50 or abs(total_error - last_total_error) < 0.001:
            converged=True
        last_total_error = total_error
    return w,errors,count

w = np.zeros(len(X_train1[0]))
w,errors,t = adaline(X_train1,w,Y_train1,0.1)

print('Q4')
plt.plot(errors)
plt.title('Error vs Iteration')
plt.xlabel('iterations')
plt.ylabel('errors')
```

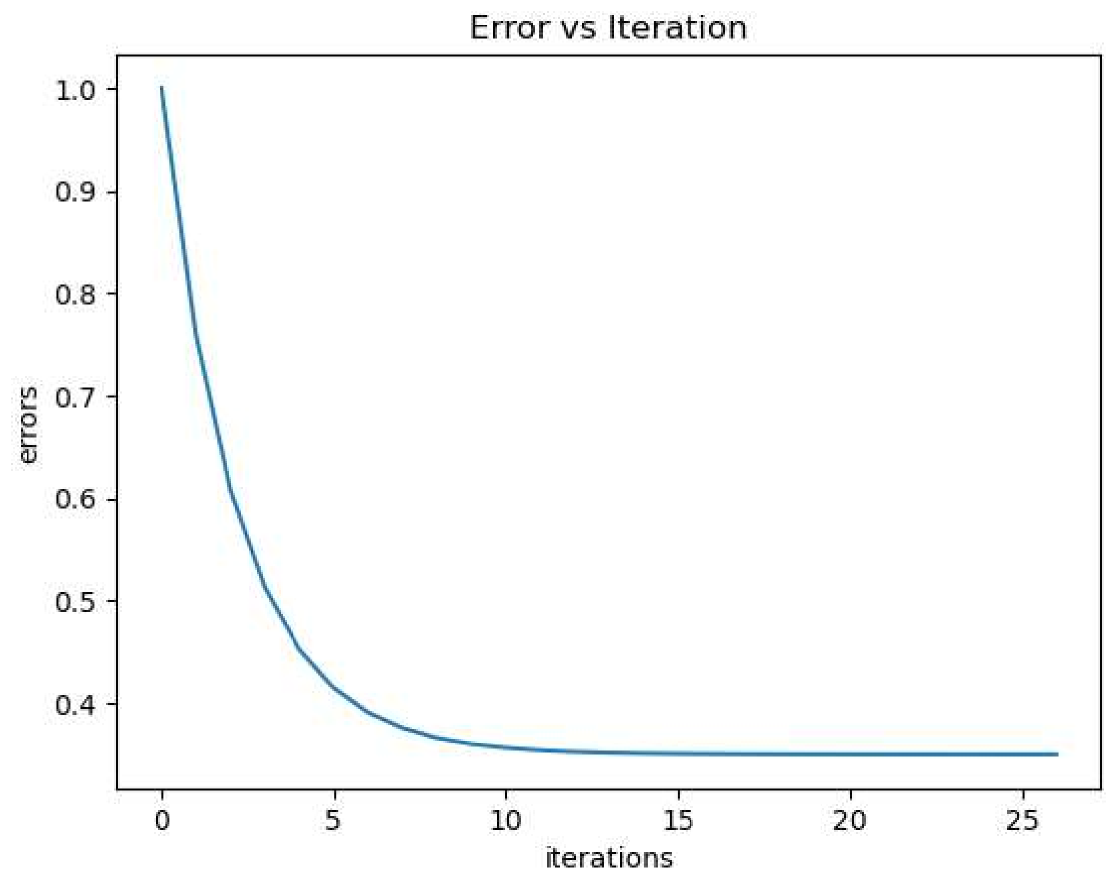
```
plt.show()
print('Q4. DATASET1')
print(' Weights ', w)
print(' No. of Iterations:', t)
def squared_loss(X,w,y):
    Sq_error = 0.0
    misclassified=0
    ypred = np.dot(X,w)
    for i in range(len(y)):
        if (ypred[i] >= 0.0):
            ypred[i]=1
        else:
            ypred[i]=-1
        error = y[i]-ypred[i]
        Sq_error += error*error
        if y[i]!=ypred[i]:
            misclassified+=1

    print(' Misclassified observations:', misclassified)
    print(' Total Error ', Sq_error)
    print(' Accuracy:  %.2f' % accuracy_score(y, ypred))
    return ypred,Sq_error

print('\n For training data')
ypred_train,sq_error = squared_loss(X_train1,w,Y_train1)
print('\n For test data')
ypred_test,sq_error = squared_loss(X_test1,w,Y_test1)

print('Q4. DATASET2')
df_adaline2 = pd.read_csv(r'D:\Spring 23\EEE 591\HW2\Dataset_2.csv')
X_train2 = df_adaline2.iloc[:,0:3].values
Y_train2 = df_adaline2['y'].values
X_trainstd2 = stdslr.fit_transform(X_train2[:,0:3])
one = np.ones(len(X_trainstd2[:,0]),dtype=float)
# compact notation
A_a2 = np.column_stack((one,X_trainstd2))

X_train2, X_test2, Y_train2, Y_test2 = train_test_split(A_a2,Y_train2,test_size=0.33,random_state=0)
w_init = np.ones(len(X_train2[0]))*0.1
w_2,errors_tot,t = adaline(X_train2,w_init,Y_train2,0.1)

print('\n For training data')
yp_train,total_error = squared_loss(X_train2,w_2,Y_train2)
print('\n For test data')
yp_test,tot_err = squared_loss(X_test2,w_2,Y_test2)

print(' FROM SKLEARN')
print('\nQ5 Data set 1')


from sklearn.linear_model import Perceptron
P = Perceptron(max_iter=1500, tol=0.000001, eta0=0.001, random_state=0)
P.fit(X_train1, Y_train1)
y_pred = P.predict(X_train1)
print(' \n Training Data'' \nMisclassified samples: %d' % (Y_train1 != y_pred).sum())
print(' Train Accuracy: %.2f' % accuracy_score(Y_train1, y_pred))

y_pred = P.predict(X_test1)
print(' \n Testing data ''\nMisclassified samples: %d' % (Y_test1 != y_pred).sum())
print('Accuracy: %.2f' % accuracy_score(Y_test1, y_pred))


print('\n Data set 2')
P2 = Perceptron(max_iter=1500, tol=0.0000001, eta0=0.001, random_state=0)
P2.fit(X_train2, Y_train2)
y_pred2 = P2.predict(X_train2)
print(' \n Training data ''\nMisclassified samples: %d' % (Y_train2 != y_pred2).sum())
print('Train Accuracy: %.2f' % accuracy_score(Y_train2, y_pred2))

y_pred2 = P2.predict(X_test2)
print('\n testing data''\nMisclassified samples: %d' % (Y_test2 != y_pred2).sum())
print('Test Accuracy: %.2f' % accuracy_score(Y_test2, y_pred2))
```

Q2

```
Iteration 1
 weights  [-0.01813055  0.54280608  0.48894682  0.32988374]
 error  -3.886715807571962
prediction [1. 1. 1. 1. 1.]

Iteration 2
 weights  [-0.11813055  0.44280608  0.58894682  0.42988374]
 error  -3.2867158075719614
prediction [1. 1. 1. 1. 1.]

Iteration 3
 weights  [-0.21813055  0.34280608  0.68894682  0.52988374]
 error  -2.6867158075719613
prediction [1. 1. 1. 1. 1.]

Iteration 4
 weights  [-0.31813055  0.34280608  0.58894682  0.62988374]
 error  -2.286715807571961
prediction [1. 1. 1. 1. 1.]

Iteration 5
 weights  [-0.41813055  0.24280608  0.68894682  0.72988374]
 error  -1.6867158075719613
prediction [1. 1. 1. 1. 1.]

Iteration 6
 weights  [-0.51813055  0.14280608  0.78894682  0.82988374]
 error  -1.0867158075719616
prediction [1. 1. 1. 1. 1.]

Iteration 7
 weights  [-0.61813055  0.04280608  0.88894682  0.92988374]
 error  -0.48671580757196176
prediction [ 1. -1.  1.  1.  1.]

Iteration 8
 weights  [-0.71813055  0.24280608  0.48894682  1.02988374]
 error  -0.6867158075719617
prediction [-1. -1.  1.  1.  1.]

Iteration 9
 weights  [-0.71813055  0.24280608  0.48894682  1.02988374]
 error  -0.6867158075719617
prediction [-1. -1.  1.  1.  1.]

Iteration 10
 weights  [-0.71813055  0.24280608  0.48894682  1.02988374]
 error  -0.6867158075719617
prediction [-1. -1.  1.  1.  1.]

Iteration 11
 weights  [-0.71813055  0.24280608  0.48894682  1.02988374]
 error  -0.6867158075719617
prediction [-1. -1.  1.  1.  1.]

Iteration 12
 weights  [-0.71813055  0.24280608  0.48894682  1.02988374]
 error  -0.6867158075719617
prediction [-1. -1.  1.  1.  1.]

Iteration 13
 weights  [-0.71813055  0.24280608  0.48894682  1.02988374]
 error  -0.6867158075719617
prediction [-1. -1.  1.  1.  1.]

Iteration 14
 weights  [-0.71813055  0.24280608  0.48894682  1.02988374]
 error  -0.6867158075719617
prediction [-1. -1.  1.  1.  1.]

Iteration 15
 weights  [-0.71813055  0.24280608  0.48894682  1.02988374]
 error  -0.6867158075719617
prediction [-1. -1.  1.  1.  1.]

Iteration 16
 weights  [-0.71813055  0.24280608  0.48894682  1.02988374]
 error  -0.6867158075719617
prediction [-1. -1.  1.  1.  1.]
```




```
Accuracy of model: 1.00
Number of misclassification predictions are 0
Q4
```



```
Q4. DATASET1
 Weights  [-0.03919539  0.3551497  -0.67420292  0.21692546]
 No. of Iterations: 27

 For training data
 Misclassified observations: 8
 Total Error  32.0
 Accuracy:  0.98

 For test data
 Misclassified observations: 6
 Total Error  24.0
 Accuracy:  0.96
Q4. DATASET2

 For training data
 Misclassified observations: 19
 Total Error  76.0
 Accuracy:  0.94

 For test data
 Misclassified observations: 10
 Total Error  40.0
 Accuracy:  0.94
 FROM SKLEARN

Q5 Data set 1

 Training Data
Misclassified samples: 5
 Train Accuracy: 0.99

 Testing data
Misclassified samples: 4
Accuracy: 0.98

 Data set 2

 Training data
Misclassified samples: 3
Train Accuracy: 0.99

 testing data
Misclassified samples: 1
Test Accuracy: 0.99
```

In [ ]: