import numpy as np

import numpy

from matplotlib import pyplot as plt

%matplotlib inline

import pandas as pd

dt=pd.read\_csv('data.csv',sep=',',encoding='cp1252')

outing=pd.read\_csv('out.csv',sep=',',header=0,encoding='cp1252')

arr=outing.values

array=dt.values

X=dt.drop('clas',axis=1)

x=array[:,1:]

t=array[:,0]

X = np.array([

x[0],x[1],x[2],x[3],x[4],x[5],x[6],x[7],x[8],x[9]

])

pred=arr[:,:]

y = np.array(t)

print(y) ###class labels

print(x)

print(pred)

def svm\_sgd\_plot(X, Y):

#Initialize our SVMs weight vector with zeros (3 values)

w = np.zeros(len(X[0]))

#The learning rate

eta = 1

#how many iterations to train for

epochs = 100000

#store misclassifications so we can plot how they change over time

errors = []

#training part, gradient descent part

for epoch in range(1,epochs):

error = 0

for i, x in enumerate(X):

#misclassification

if (Y[i]\*np.dot(X[i], w)) < 1:

#misclassified update for ours weights

w = w + eta \* ( (X[i] \* Y[i]) + (-2 \*(1/epoch)\* w) )

error = 1

else:

#correct classification, update our weights

w = w + eta \* (-2 \*(1/epoch)\* w)

errors.append(error)

#lets plot the rate of classification errors during training for our SVM

plt.plot(errors, '|')

plt.ylim(0.5,1.5)

plt.axes().set\_yticklabels([])

plt.xlabel('Epoch')

plt.ylabel('Misclassified')

plt.show()

return w

def prediction(X):

for i, x in enumerate(X):

#misclassification

print(i,x)

print((y[i]\*np.dot(X[i], w)))

if (y[i]\*np.dot(X[i], w)) <0:

print("A")

else:

print("B")

w = svm\_sgd\_plot(X,y)

X=pred

prediction(X)

print(w)