Question_4_Complete

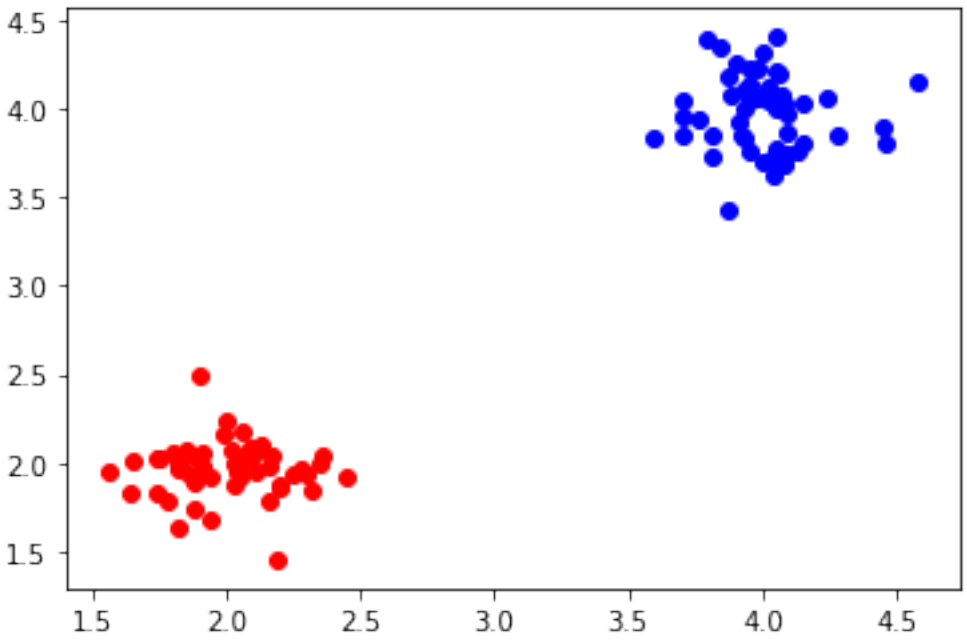
September 28, 2020

[1]:
```
#importing necessary libraries
import numpy as np
from matplotlib import pyplot as plt
import random
random.seed(101)
```

[2]:
```
# user chooses the length of the data
length_dataset = 100
length_test_dataset = 10000
```

[3]:
```
#creating the linearly separable dataset
x1 = np.random.normal(4, 0.2, (int(length_dataset/2),2))
x2 = np.random.normal(2, 0.2, (int(length_dataset/2),2))
all_input = np.concatenate((x1, x2)) #creating a combined dataset of
```

[4]:
```
#Visualizing the linearly separable dataset
plt.scatter(x1[:,0], x1[:,1], color='blue')
plt.scatter(x2[:,0], x2[:,1], color='red')
#separating the blue and the red points and categorizing the same
d1 = -1 * (np.ones(int(length_dataset/2)))
d2 = np.ones(int(length_dataset/2))
all_combined_targets = np.concatenate((d1,d2))
plt.show()
```



[5]:
```
def Y_predict(x_vector,w):
    x_new = [1]
    for i in x_vector:
        x_new.append(i)
    x_new = np.array((x_new))
    res = (np.dot(x_new,w))
    if res > 0:
        Y = 1
        return Y
    elif res < 0:
        Y = -1
        return Y
    elif res ==0:
        Y = 0
        return Y


def train(X,iterations,eta):
    global count
    global w
    global all_combined_targets
    for y_idx in range (len(X)):
        ran_num = random.randint(0,len(X)-1)
        x_train = X[ran_num]
        y_t = Y_predict(x_train,w)
        misrepresented_list = []
```

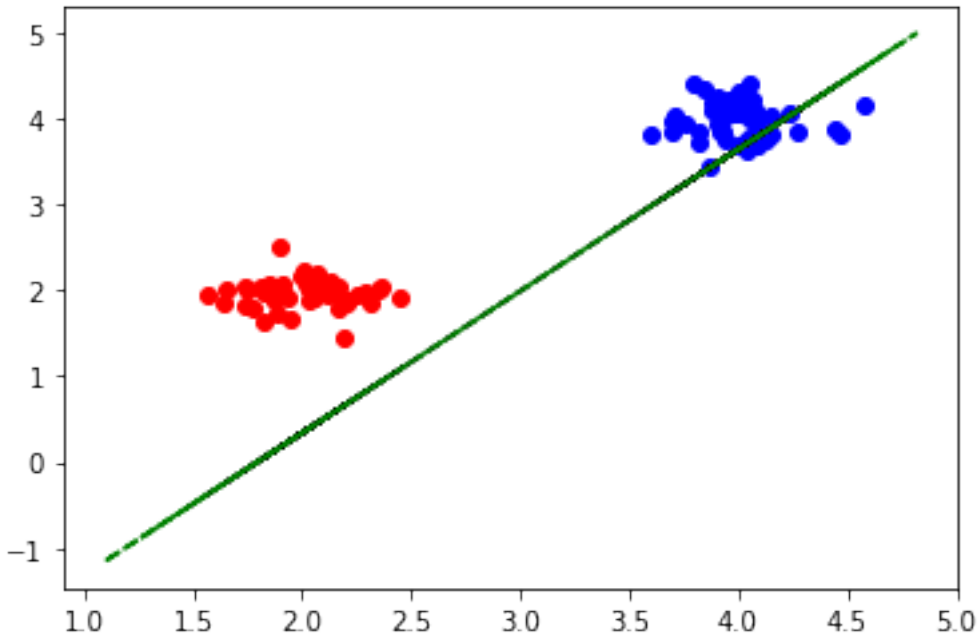
```
        for i,j in enumerate(all_combined_targets):
            if j!=y_t:
                misrepresented_list.append(i)
        if len(misrepresented_list)==0:
            print('Full accuracy achieved')
            break
        random_selection = random.randint(0,len(misrepresented_list)-1)
        random_index = misrepresented_list[random_selection]
        x_selected = X[random_index]
        y_selected = all_combined_targets[random_index]
#         print(x_selected,y_selected)
        x_with1 = [1]
        for i in x_selected:
            x_with1.append(i)
        x_with1 = np.array((x_with1))
#         print('old w - > ',w)
        s_t = np.matmul(w,x_with1)
#         print('x_with1',x_with1)
#         print('s_t',s_t)
#         print('y_selected',y_selected)
#         print('y_selected*s_t',y_selected*s_t)
        if (y_selected*s_t)<=1:
            w = w+(eta*(y_selected-s_t)*x_with1)
#             print('w - > ' , w)
#             print(' ')
#             print(' ')
#             print(' ')
        if (count==iterations):
            print('maximum iterations reached in the training block')
            break
            count+=1
```

0.1 eta = 100

[6]:
```
#initializing all parameters
count = 0
w0 = random.randint(1,4)
w1 = random.randint(1,4)
w2 = random.randint(1,4)
w = np.array((w0,w1,w2))
weight= 0
iterations = 20
eta = 0.001
#calling the function
train(all_input,iterations,eta)
```

[7]:
```
#Visualizing the linearly separable dataset
plt.scatter(x1[:,0], x1[:,1], color='blue')
plt.scatter(x2[:,0], x2[:,1], color='red')
m = -(w[1]/w[2])
c = -(w[0]/w[2])
plt.plot(all_input, m*all_input + c,'k--')
#test plotting
#creating the linearly separable dataset
xtest = np.random.normal(4, 0.2, (int(length_test_dataset/2),2))
x2test = np.random.normal(2, 0.2, (int(length_test_dataset/2),2))
all_input_test = np.concatenate((xtest, x2test)) #creating a combined dataset of
plt.plot(all_input_test, m*all_input_test + c,'g--')
print('w:',w)
```

```
w: [ 1.08709874 -0.60569865  0.36709108]
```



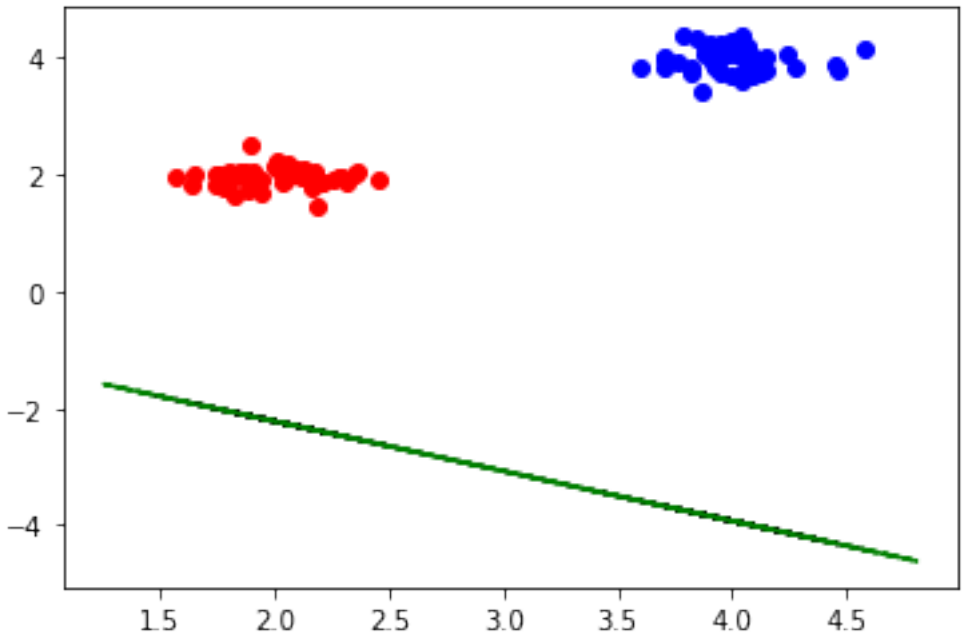
0.2 eta = 1

[8]:
```
#initializing all parameters
count = 0
w0 = random.randint(1,4)
w1 = random.randint(1,4)
w2 = random.randint(1,4)
w = np.array((w0,w1,w2))
weight= 0
iterations = 20
```

```
eta = 1
#calling the function
train(all_input,iterations,eta)
```

[9]:
```
#Visualizing the linearly separable dataset
plt.scatter(x1[:,0], x1[:,1], color='blue')
plt.scatter(x2[:,0], x2[:,1], color='red')
m = -(w[1]/w[2])
c = -(w[0]/w[2])
plt.plot(all_input, m*all_input + c,'k--')
#test plotting
#creating the linearly separable dataset
xtest = np.random.normal(4, 0.2, (int(length_test_dataset/2),2))
x2test = np.random.normal(2, 0.2, (int(length_test_dataset/2),2))
all_input_test = np.concatenate((xtest, x2test)) #creating a combined dataset of
plt.plot(all_input_test, m*all_input_test + c,'g--')
print('w:',w)
```

```
w: [3.92614577e+118 6.48064357e+118 7.60059729e+118]
```



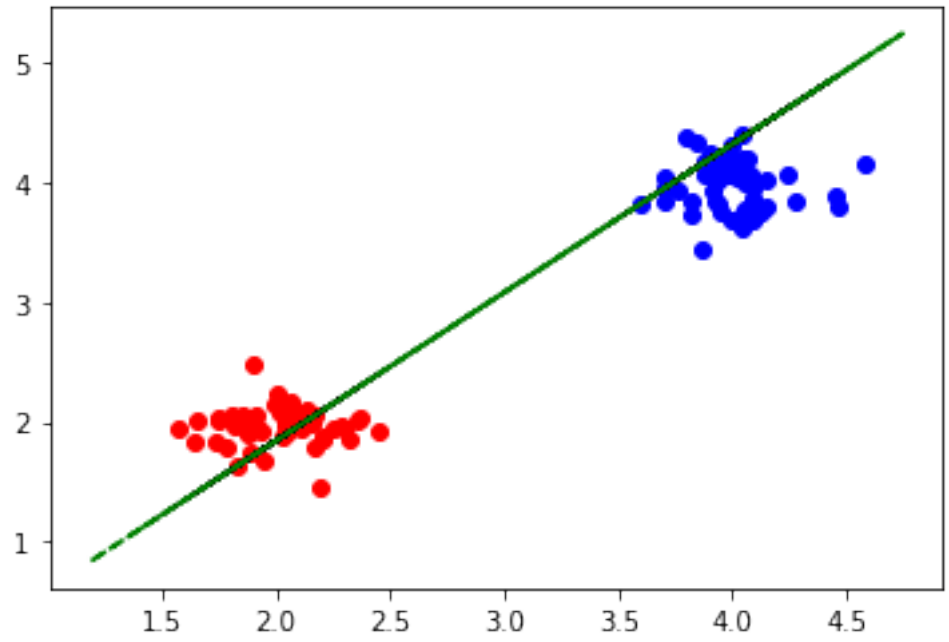
0.3 eta = 0.01

[10]:
```
#initializing all parameters
count = 0
w0 = random.randint(1,4)
w1 = random.randint(1,4)
```

```
w2 = random.randint(1,4)
w = np.array((w0,w1,w2))
weight= 0
iterations = 20
eta = 0.01
#calling the function
train(all_input,iterations,eta)
```

[11]:
```
#Visualizing the linearly separable dataset
plt.scatter(x1[:,0], x1[:,1], color='blue')
plt.scatter(x2[:,0], x2[:,1], color='red')
m = -(w[1]/w[2])
c = -(w[0]/w[2])
plt.plot(all_input, m*all_input + c,'k--')
#test plotting
#creating the linearly separable dataset
xtest = np.random.normal(4, 0.2, (int(length_test_dataset/2),2))
x2test = np.random.normal(2, 0.2, (int(length_test_dataset/2),2))
all_input_test = np.concatenate((xtest, x2test)) #creating a combined dataset of
plt.plot(all_input_test, m*all_input_test + c,'g--')
print('w:',w)
```

```
w: [ 0.51616892 -1.01058716  0.81523726]
```

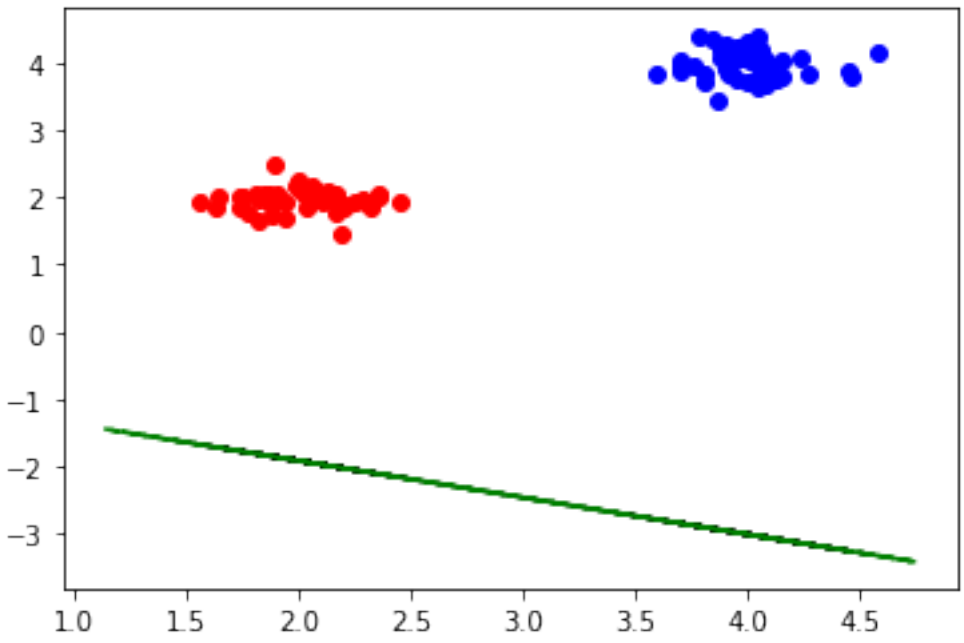


0.4 eta = 0.0001

[12]:
```
#initializing all parameters
count = 0
w0 = random.randint(1,4)
w1 = random.randint(1,4)
w2 = random.randint(1,4)
w = np.array((w0,w1,w2))
weight= 0
iterations = 20
eta = 0.0001
#calling the function
train(all_input,iterations,eta)
```

[13]:
```
#Visualizing the linearly separable dataset
plt.scatter(x1[:,0], x1[:,1], color='blue')
plt.scatter(x2[:,0], x2[:,1], color='red')
m = -(w[1]/w[2])
c = -(w[0]/w[2])
plt.plot(all_input, m*all_input + c,'k--')
#test plotting
#creating the linearly separable dataset
xtest = np.random.normal(4, 0.2, (int(length_test_dataset/2),2))
x2test = np.random.normal(2, 0.2, (int(length_test_dataset/2),2))
all_input_test = np.concatenate((xtest, x2test)) #creating a combined dataset of
plt.plot(all_input_test, m*all_input_test + c,'g--')
print('w:',w)
```

```
w: [1.80332769 1.20856561 2.21148356]
```



1 Comparison of Results

[15]:
```
# We can see that in case a) the results are not too bad but the line cuts
# through the linearly separable points
```

[16]:
```
# In case b) The line is very off
```

[17]:
```
# In case c) It almost verifies the linear separability
```

[18]:
```
# In case d) The line goes off the path again
```

[19]:
```
# Conclusion the eta value that resulted in the minimu classification error␣
↪wass when eta = 0.01
```

[ ]:
```

```