ENGR421

HW3

Doğa Demirtürk
68859

In this homework, to implement a discrimination by regression algorithm, we use sigmoid function for estimation and the sum squared errors to minimize error function.

First, I realized that the given mean values, covariances and sizes are the same as the first homework, so I took the data generation and plotting part from my first homework. Then I saved data to a file and read from the file to estimate regression parameters. I used one hot encoding in y values.

I defined a sigmoid function from my lecture notes and lab 3.

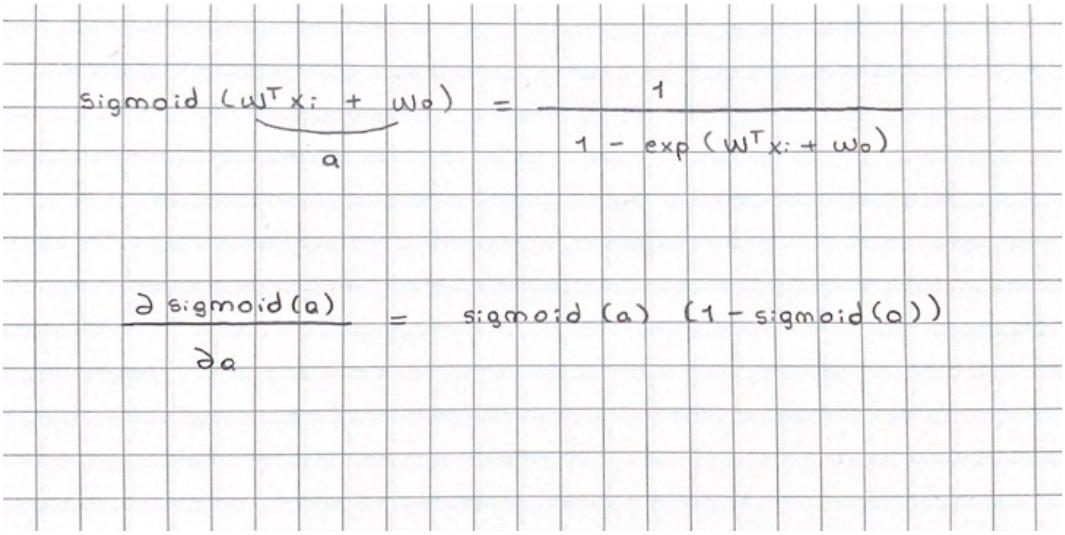

Figure 1: Sigmoid function

Since we use sum squared error for error function, I calculated the gradients accordingly and implemented gradient functions. Then I started iteration with help pf lab 4 and finally I found the expected values. I found the exact random seed used in the description file so I could compare my results with the example results.

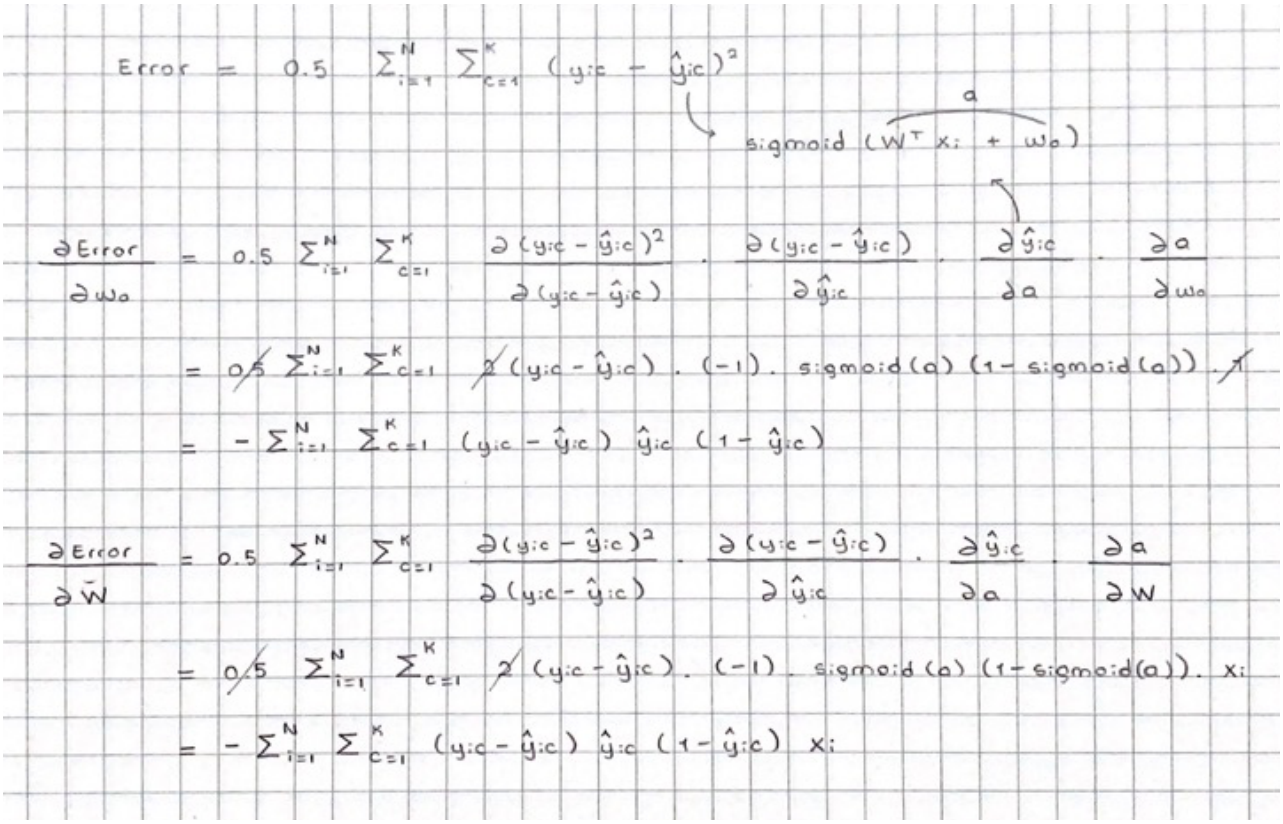

Figure 2: Calculations on gradients for update equations

Here are the final results I get by implementing discrimination by regression algorithm:

In [11]:
```
# learn W and w0 using gradient descent
iteration = 1
objective_values = []
while 1:
    Y_predicted = sigmoid(X, W, w0)

    objective_values = np.append(objective_values, 0.5 * np.sum(np.square(Y_truth - Y_predicted)))

    W_old = W
    w0_old = w0

    W = W - eta * gradient_W(X, Y_truth, Y_predicted)
    w0 = w0 - eta * gradient_w0(Y_truth, Y_predicted)

    if np.sqrt(np.sum((w0 - w0_old))**2 + np.sum((W - W_old)**2)) < epsilon:
        break

    iteration = iteration + 1
print(W)
print(w0)
```

```
[[ 0.02528161 -2.23729285  2.44057305]
 [ 4.60790215 -2.46097419 -2.26881565]]
[[-1.13651539 -4.28418412 -3.6081088 ]]
```

Figure 3: W and w0 values

In [13]:
```
# calculate confusion matrix
y_predicted = np.argmax(Y_predicted, axis = 1) + 1
confusion_matrix = pd.crosstab(y_predicted, y_truth, rownames = ['y_pred'], colnames = ['y_truth'])
print(confusion_matrix)
```

```
y_truth    1   2   3
y_pred
1        117   1   3
2          2  78   0
3          1   1  97
```

Figure 5: Confusion matrix

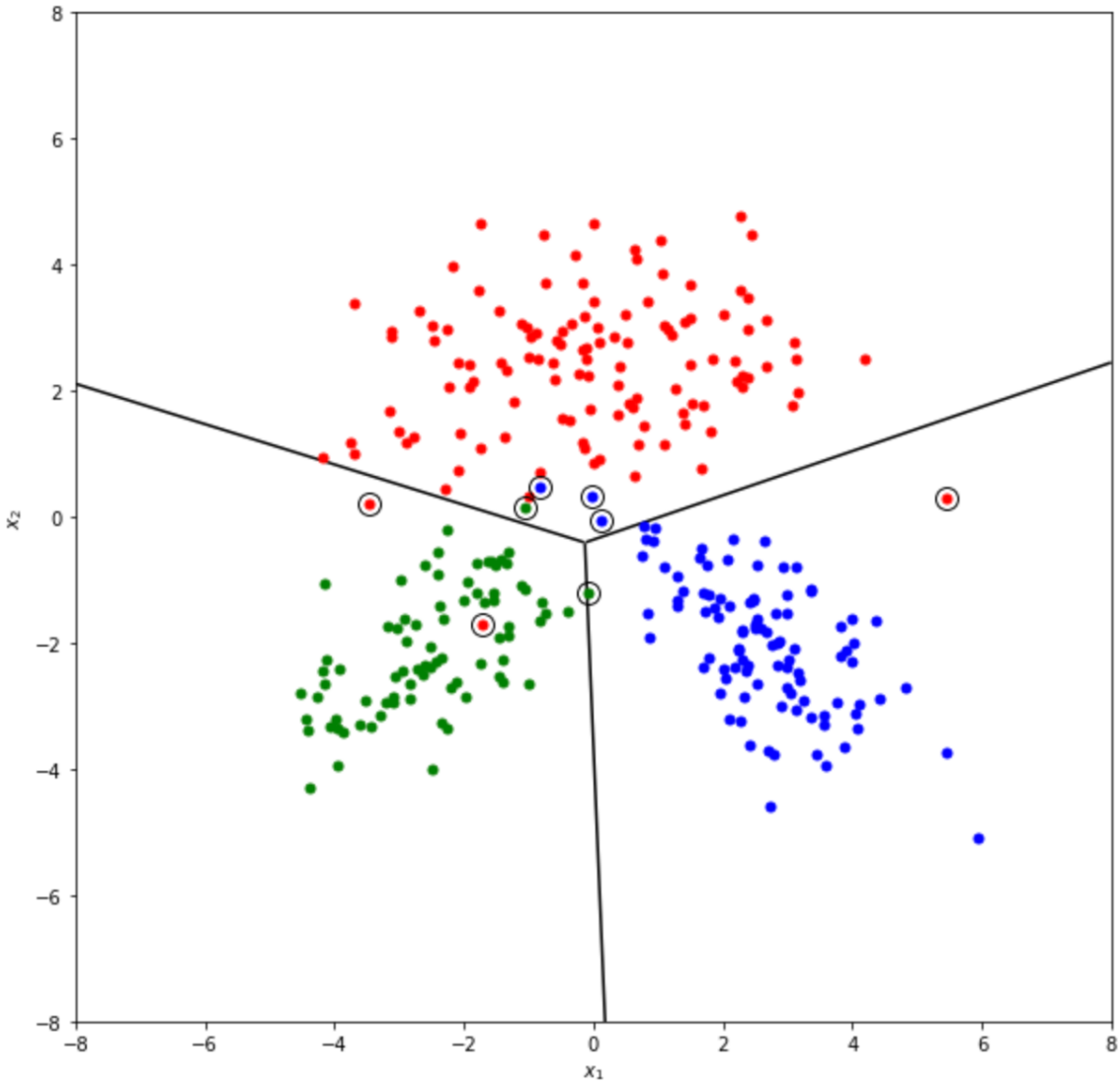

Figure 4: Decision boundaries