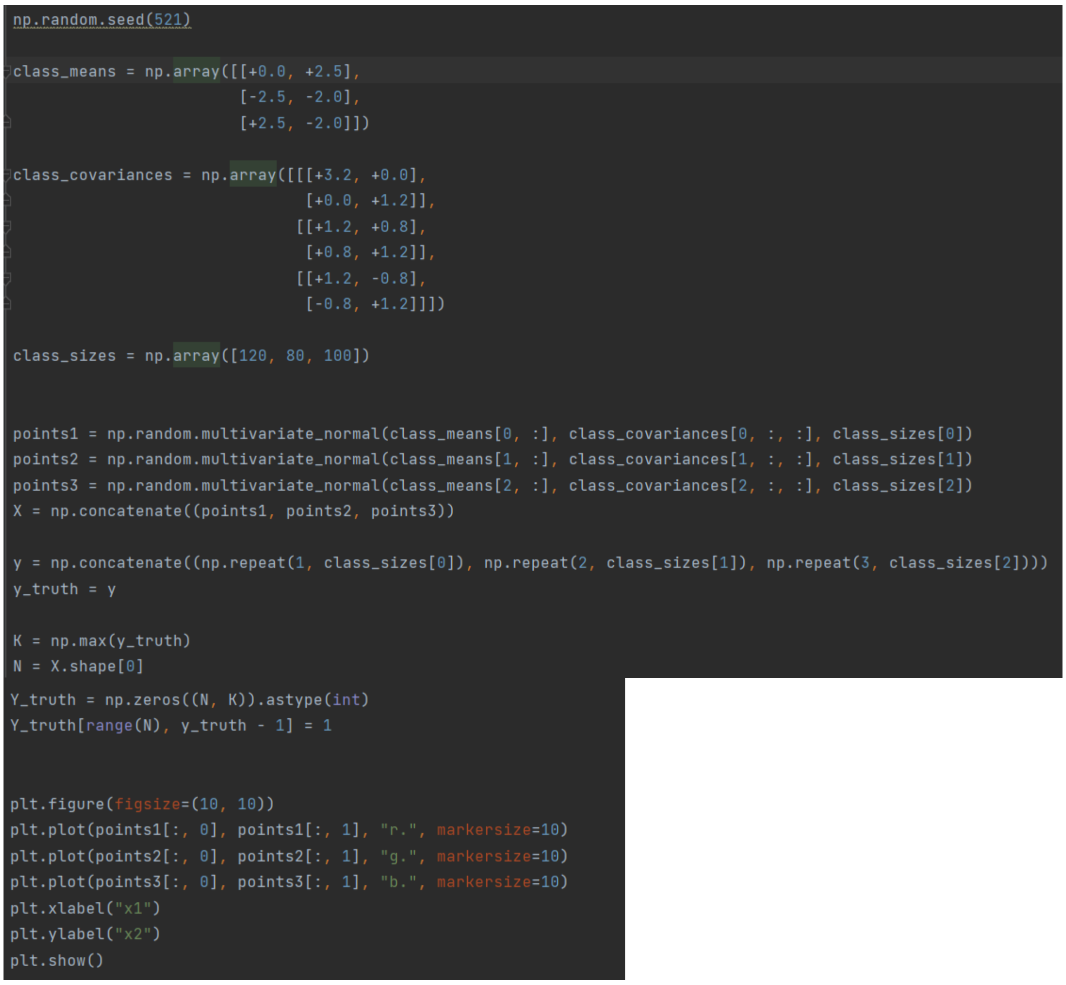
Eren Barış Bostancı
68770

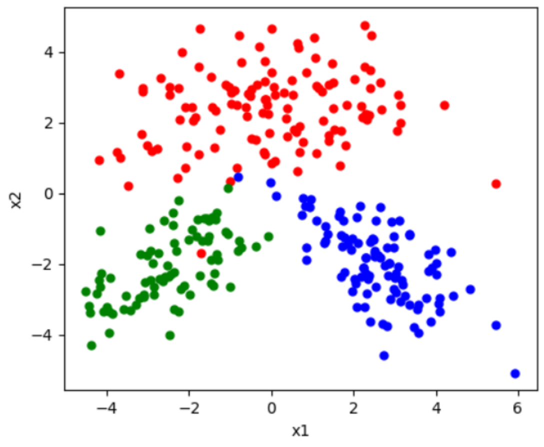
ENGR 421 – HW3

Data Generation:

First, I generate the data with the same way I in HW1. I tried 421 and 521(course codes) seeds and I saw that 521 gives same data with the HW pdf. Thus, I decide to use 521.



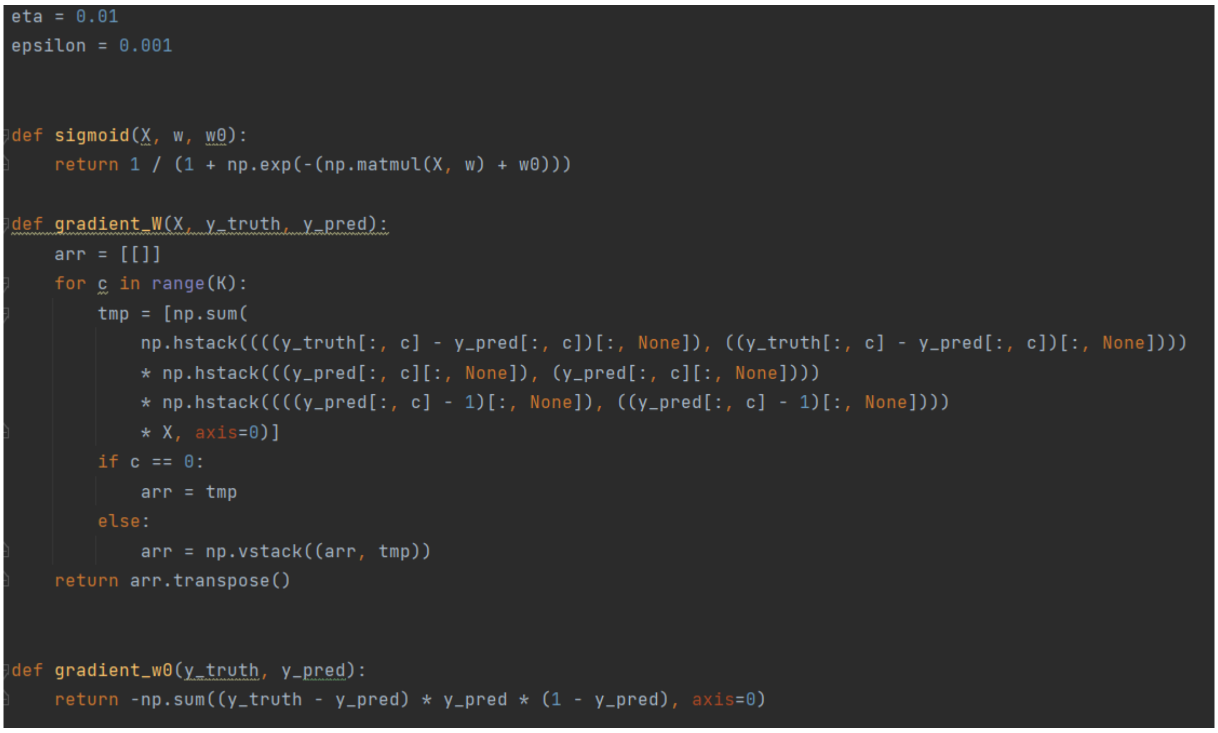
```python
np.random.seed(521)

class_means = np.array([[+0.0, +2.5],
                        [-2.5, -2.0],
                        [+2.5, -2.0]])

class_covariances = np.array([[[+3.2, +0.0],
                               [+0.0, +1.2]],
                              [[+1.2, +0.8],
                               [+0.8, +1.2]],
                              [[+1.2, -0.8],
                               [-0.8, +1.2]]])

class_sizes = np.array([120, 80, 100])


points1 = np.random.multivariate_normal(class_means[0, :], class_covariances[0, :, :], class_sizes[0])
points2 = np.random.multivariate_normal(class_means[1, :], class_covariances[1, :, :], class_sizes[1])
points3 = np.random.multivariate_normal(class_means[2, :], class_covariances[2, :, :], class_sizes[2])
X = np.concatenate((points1, points2, points3))

y = np.concatenate((np.repeat(1, class_sizes[0]), np.repeat(2, class_sizes[1]), np.repeat(3, class_sizes[2])))
y_truth = y

K = np.max(y_truth)
N = X.shape[0]

Y_truth = np.zeros((N, K)).astype(int)
Y_truth[range(N), y_truth - 1] = 1


plt.figure(figsize=(10, 10))
plt.plot(points1[:, 0], points1[:, 1], "r.", markersize=10)
plt.plot(points2[:, 0], points2[:, 1], "g.", markersize=10)
plt.plot(points3[:, 0], points3[:, 1], "b.", markersize=10)
plt.xlabel("x1")
plt.ylabel("x2")
plt.show()
```



Eren Barış Bostancı
68770

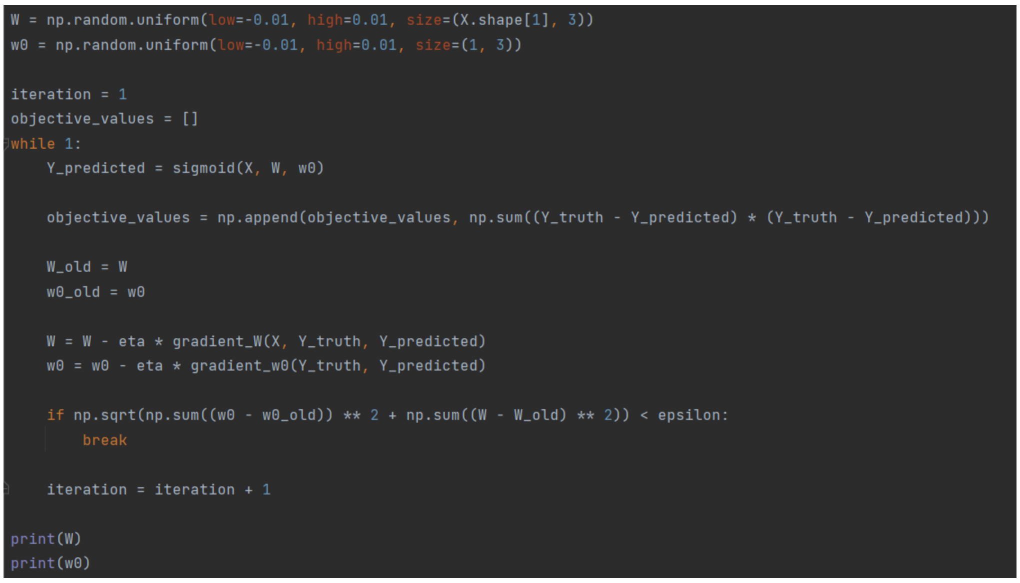
Parameter Methods:

I used the same sigmoid function from the lab3, and I modified the gradient function according to the hw’s pdf.

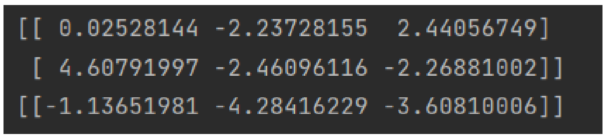


```
eta = 0.01
epsilon = 0.001


def sigmoid(X, w, w0):
    return 1 / (1 + np.exp(-(np.matmul(X, w) + w0)))

def gradient_W(X, y_truth, y_pred):
    arr = [[]]
    for c in range(K):
        tmp = [np.sum(
            np.hstack((((y_truth[:, c] - y_pred[:, c])[:, None]), ((y_truth[:, c] - y_pred[:, c])[:, None])))
            * np.hstack(((y_pred[:, c][:, None]), (y_pred[:, c][:, None])))
            * np.hstack((((y_pred[:, c] - 1)[:, None]), ((y_pred[:, c] - 1)[:, None])))
            * X, axis=0)]
        if c == 0:
            arr = tmp
        else:
            arr = np.vstack((arr, tmp))
    return arr.transpose()


def gradient_w0(y_truth, y_pred):
    return -np.sum((y_truth - y_pred) * y_pred * (1 - y_pred), axis=0)
```

And made the iteration to determine the W w0 sets.



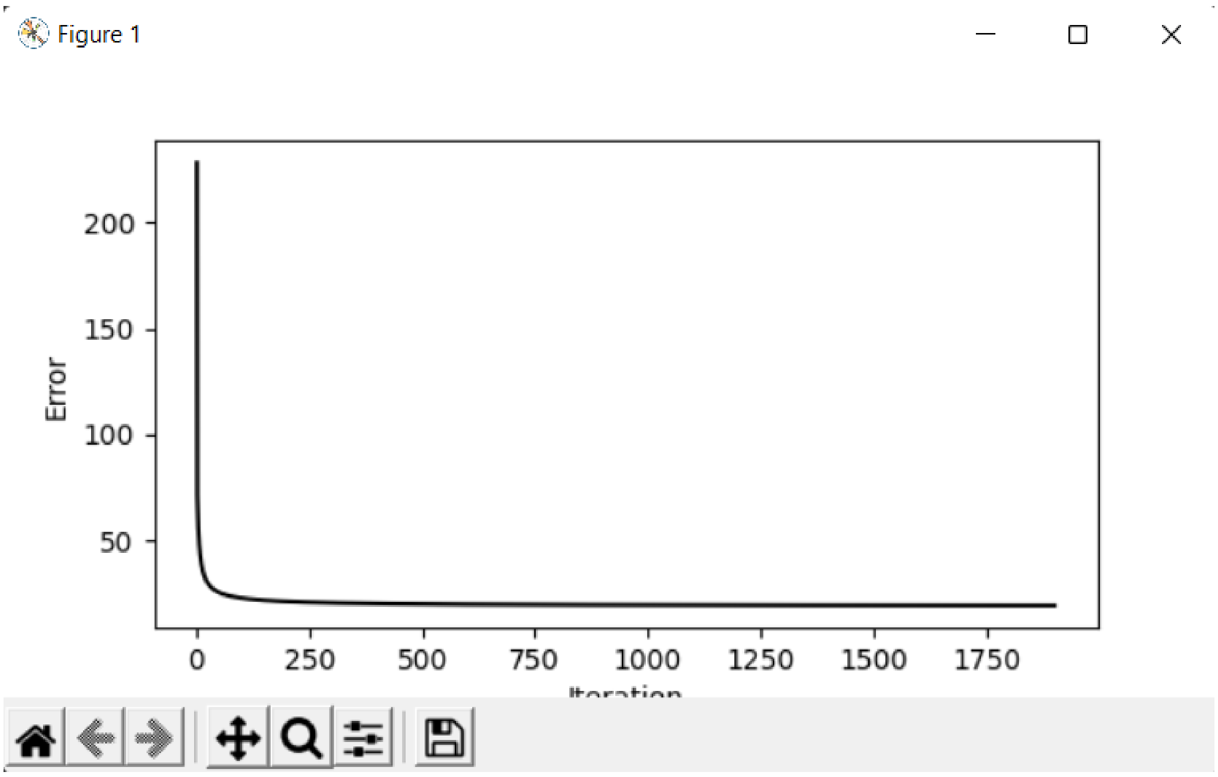
```
W = np.random.uniform(low=-0.01, high=0.01, size=(X.shape[1], 3))
w0 = np.random.uniform(low=-0.01, high=0.01, size=(1, 3))

iteration = 1
objective_values = []
while 1:
    Y_predicted = sigmoid(X, W, w0)

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



```
[[ 0.02528144 -2.23728155  2.44056749]
 [ 4.60791997 -2.46096116 -2.26881002]]
[[-1.13651981 -4.28416229 -3.60810006]]
```

Then I print the Convergence and it fits the hw's pdf.

```
plt.figure(figsize=(10, 6))
plt.plot(range(1, iteration + 1), objective_values, "k-")
plt.xlabel("Iteration")
plt.ylabel("Error")
plt.show()
```



Confusion Matrix

Then I print the confusion Matrix which is also same with the hw's pdf.

```
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
```

Eren Barış Bostancı
68770

Decision Boundaries

For plotting the decision boundaries, I used the same code from the labs and plot is same with the HW’s pdf.

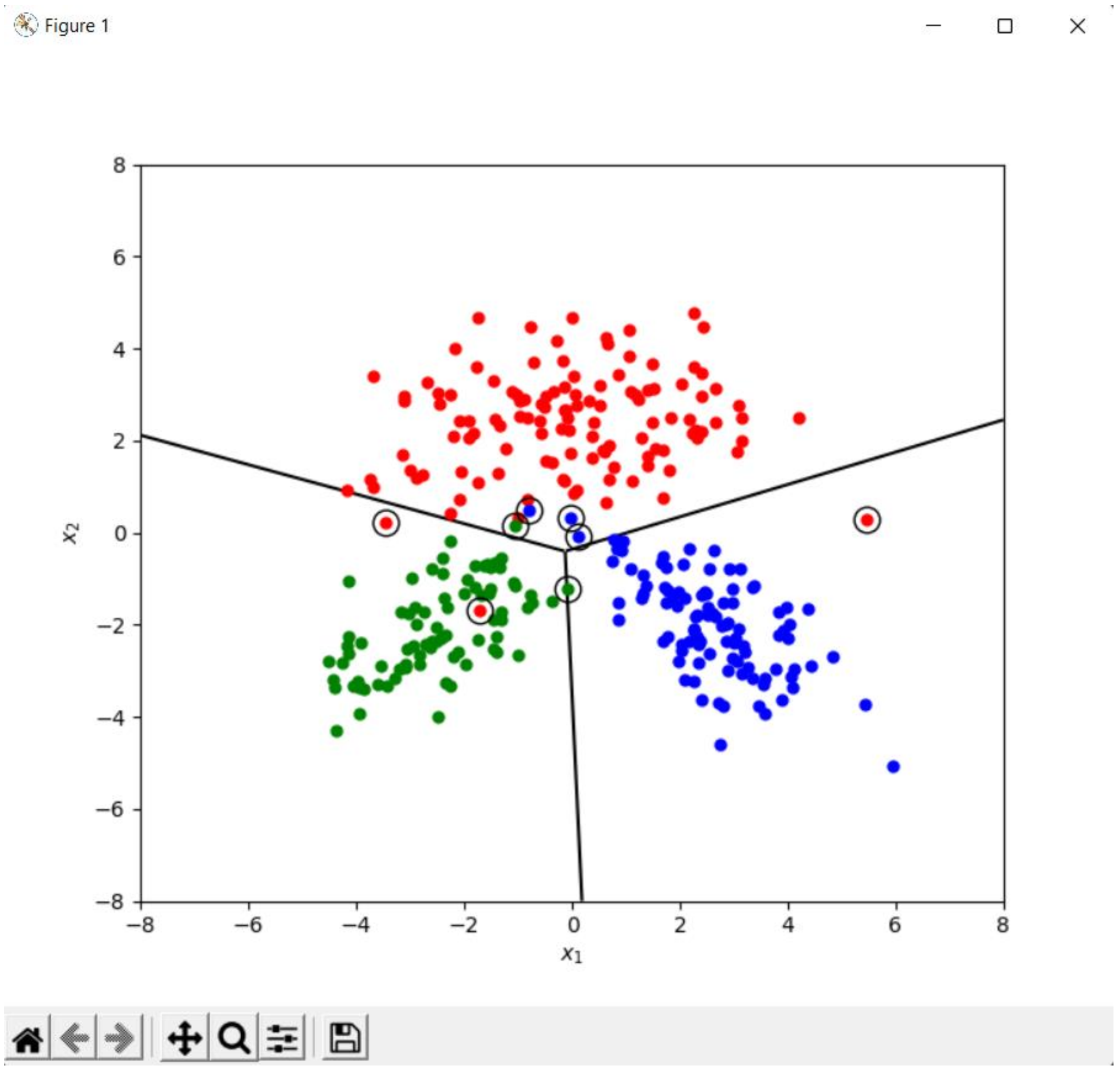


Figure 1
x_1
x_2