**ENGR-421 HW-3 REPORT**

**Name-Surname: Barış KAPLAN**

Initially, I have imported the necessary libraries. These libraries are as follows:

**import matplotlib.pyplot as plt**

**import numpy as np**

**import pandas as pd**

**import math**

Then, I have generated the mean, covariance matrix, and size parameters.

After that, by using the **np.random.multivariate\_normal** function of the numpy library, I have generated the random data points.

By using the **np.vstack** function of numpy library, I have vertically stacked the created random data points.

After that, I have generated the class labels by utilizing the **np.concatenate** function of the numpy library.

Next, by using the **plt.plot** function, I have generated the plot of the created random data points. By utilizing **plt.xlim** and **plt.ylim** functions, I have determined the limits of the x values and the limits of the y values in the plot. You can see the generated plot in Figure 1.

Chart, scatter chart

Description automatically generated

Figure 1: The plot containing the created random data points

I have used sigmoid function to generate the predicted values. While writing the sigmoid function, I have utilized the formula in Figure 2.

**Text, whiteboard

Description automatically generated**

Figure 2: The sigmoid function formula I have used

Subsequently, I have defined the gradient functions for W and w0. While defining the gradient functions, I have utilized the formula in Figure 3 for W and the formula in Figure 4 for w0.

Text

Description automatically generated

Text

Description automatically generated with medium confidence

Figure 4: The gradient function for w0

Figure 3: The gradient function for W

**By using Figure 5 and Figure 6, I have found the gradients in Figure 3 & Figure 4.**

Diagram, text

Description automatically generated

Figure 5: Predicted y (y hat)

Text, letter

Description automatically generated

Figure 6: The derivation of gradients

In this step, by using the np.random.uniform() function of numpy library, I have randomly initialized W and w0.

Then, I have applied the gradient descent algorithm to W and w0. In this algorithm, I have generated the predicted values by using the sigmoid function (sigmoid\_calculation\_function). Furthermore, to update W and w0 values, I have subtracted the update amount delta W from W and delta w0 from w0 (you can see the update amounts for W and w0 in Figure 7).

Text, letter

Description automatically generated

Figure 7: The update amount formulas for W and w0

In the gradient descent algorithm, I have updated W and w0 until the condition in Figure 8 is met. When the condition in Figure 8 is met, I have terminated the algorithm.

Text, letter

Description automatically generatedText

Description automatically generated with low confidenceAs the error function, I have utilized the sum squared errors in the gradient descent algorithm. I have utilized the formula in Figure 9 for obtaining the sum squared errors. You can see the learned values of W and w0 (values after applying the gradient descent algorithm) in Figure 10.

Figure 10: The values of W and w0 after the gradient descent algorithm finishes

Figure 8: The break condition of the gradient descent algorithm

Figure 9: The formula for the sum squared errors

After that, I have plotted the “Iteration vs Error” function. We can observe from the “Iteration vs Error” function (see Figure 11) that the gradient descent algorithm is completed in approximately 1200 steps.

Then, by using the pd.crosstab function of the pandas library, I have created the confusion matrix. We can see from the confusion matrix (see Figure 12) that there is 1 misclassified data point for class 1 (having the red color) , there are 4 misclassified data points for class 2 (having the green color), and there are 3 misclassified data points for class 3 (having the blue color). In total, there are 8 (1+4+3) misclassified data points. You can see the decision boundaries between the classes and the misclassified data points from the Figure 13. While drawing the decision boundaries, I have used plt.contour function of the matplotlib.pyplot library.

Graphical user interface

Description automatically generated with low confidence

Chart, scatter chart

Description automatically generatedCalendar

Description automatically generated with low confidence

Figure 13: The plot showing the decision boundaries and the misclassified data points

Figure 12: The confusion matrix for the data points in my training set

Figure 11: The "Iteration vs Error" graph