Part B Report

Name: Gordon McCulloh

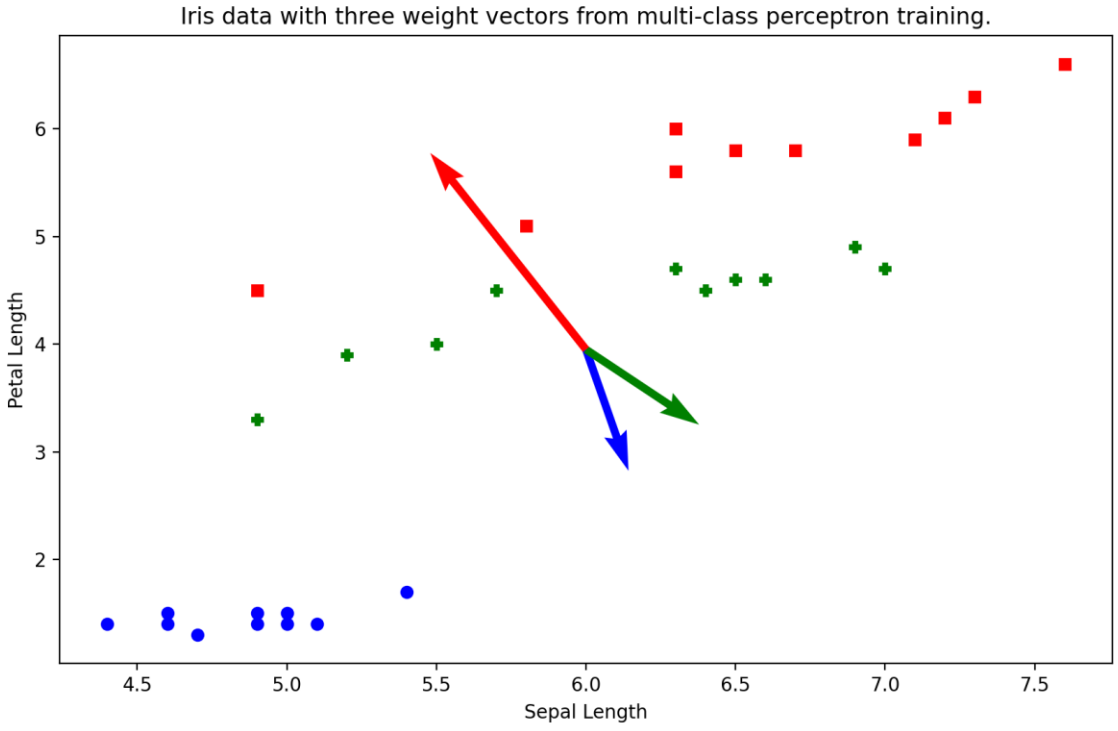
Assignment 6: Perceptron Classification and Training

CSE 415 Introduction to Artificial Intelligence, Spring 2021, University of Washington

B1. How many epochs were required to train your perceptron on the 3-class Iris data having 4 features (the given training file, with 30 examples)? How many of the test data examples (out of 120) were mis-classified? Determine the percentage error rate and write that here.

The perceptron required 85 epochs to converge for the given data set. It produced 14 errors on the test data out of 120 items. This is an 11.67% error rate.

B2. Capture the plot that is produced by the program showing the training data and the weight vectors when projected onto the 2-D subspace spanned by sepal length and petal length (which is the starter-code default in run_3_class_4_feature_iris_data.py. Paste it here, reduced to fit in the remaining space on this page.



B3. In the file run_3_class_4_feature_iris_data.py, adjust the commenting near lines 23-25 so you can see the data in the plot projects to features 2 and 3 (petal length and petal width). Describe the how the data seems to be distributed in this view. Describe how the weight vectors seem to be pointing. Finally, describe the relationship between the weight vectors and the distribution of the data.

The data is distributed such that the red data set has the highest average petal length (feature 3), the green data set is evenly distributed across many sepal lengths beneath the petal lengths of the red data set, and the blue class is tightly bunched nearer to the origin with lower sepal lengths and petal lengths. The weight vectors generally point in the direction of their respective color-coded class, but green appears to point away from the green data. This is likely to delineate the relative position between the red and green classes because the red weight vector also has much greater magnitude.

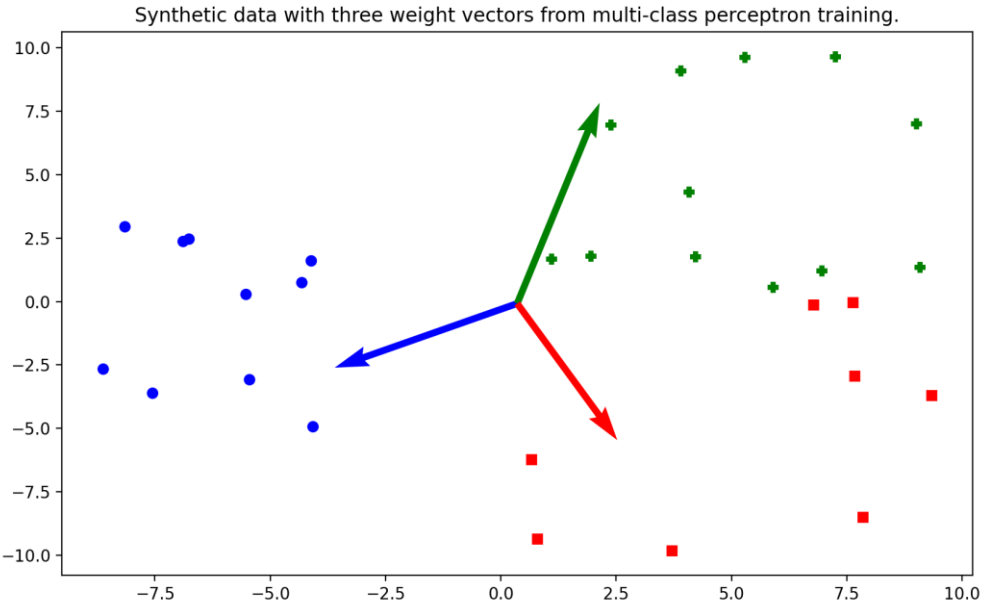
B4. In the file run_3_class_4_feature_iris_data.py, instead of using all zero weight, let

W = [[1, 1, 1, 1, 1], [-1, -1, -1, -1, -1], [0, 0, 0, 0, 0]]. Now, for learning rates starting from 1e-3 to 1e+3 (all powers of 10), investigate how many epochs it takes for the model to converge. (You may similarly investigate the model for other initializations of W if you wish). Also, find the number of errors on the test set for each binary perceptron. What kinds of trends do you observe?

All cases tested for W = [[1, 1, 1, 1, 1], [-1, -1, -1, -1, -1], [0, 0, 0, 0, 0]]. For learning rates (α) of 1e-3, 1e-2, 1e-1. 1, 1e2, and 1e3 all runs of the code required 50 epochs to converge with 10 errors out of 120 items. For lower learning rates, the program ran faster. This is because higher learning rates are inefficient. The graph appears identical to the output seen in B2.

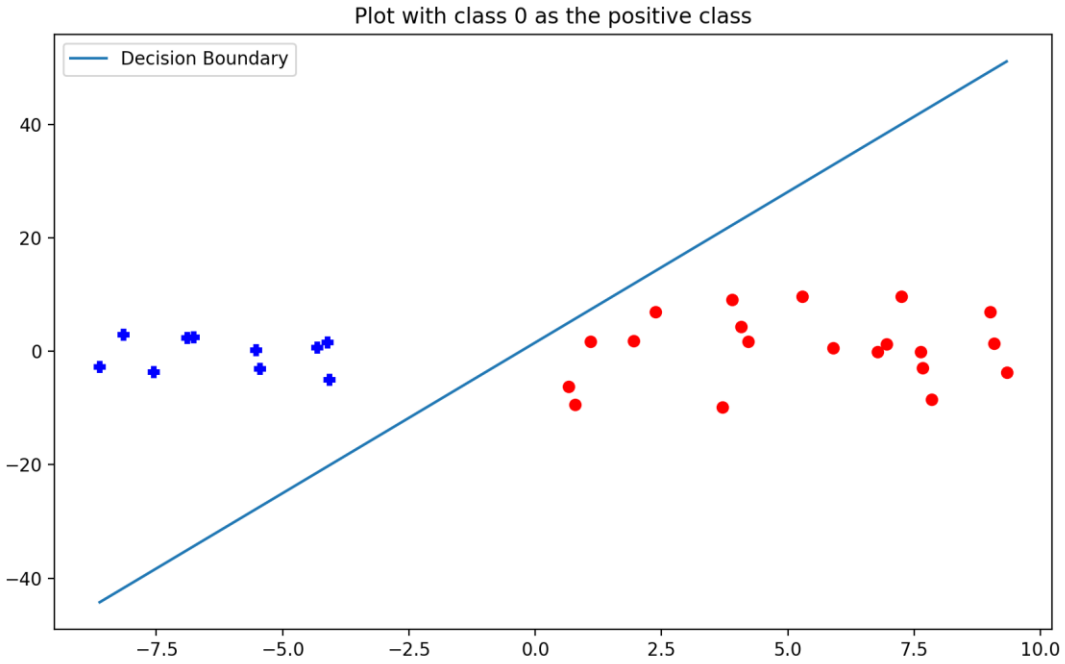
B5. Using the file run_synth_data_ternary.py, capture the plot of the ternary perceptron for the synthetic dataset and paste it here. (Let the maximum number of epochs be 50 and learning rate 0.5).

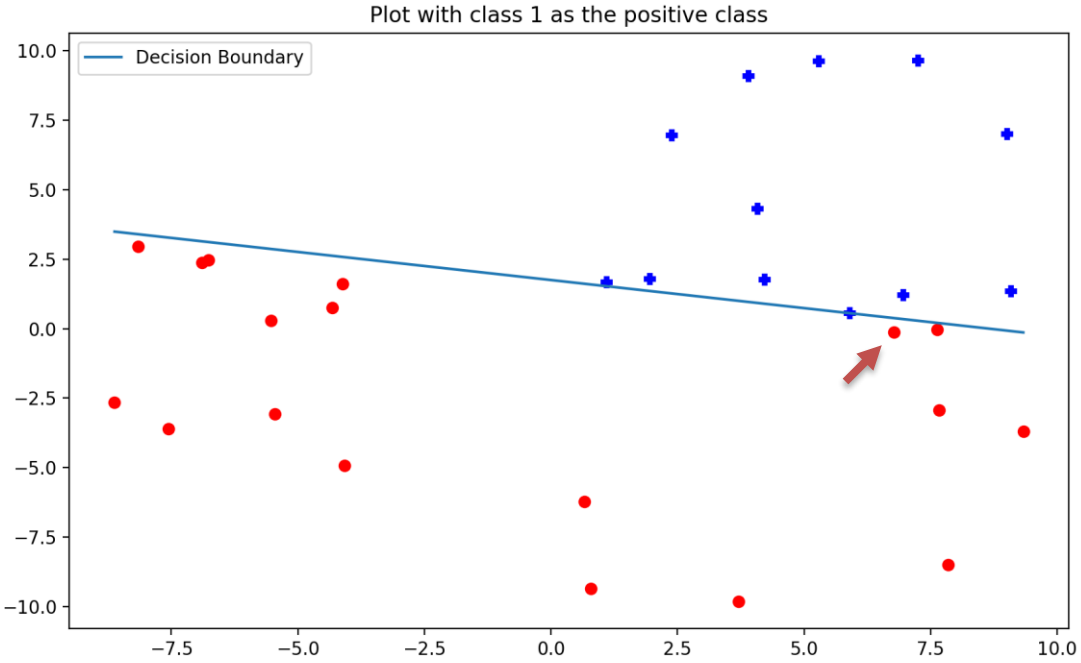
The perceptron converged in 5 epochs with 0 errors out of 30 items in the synthetic data set.

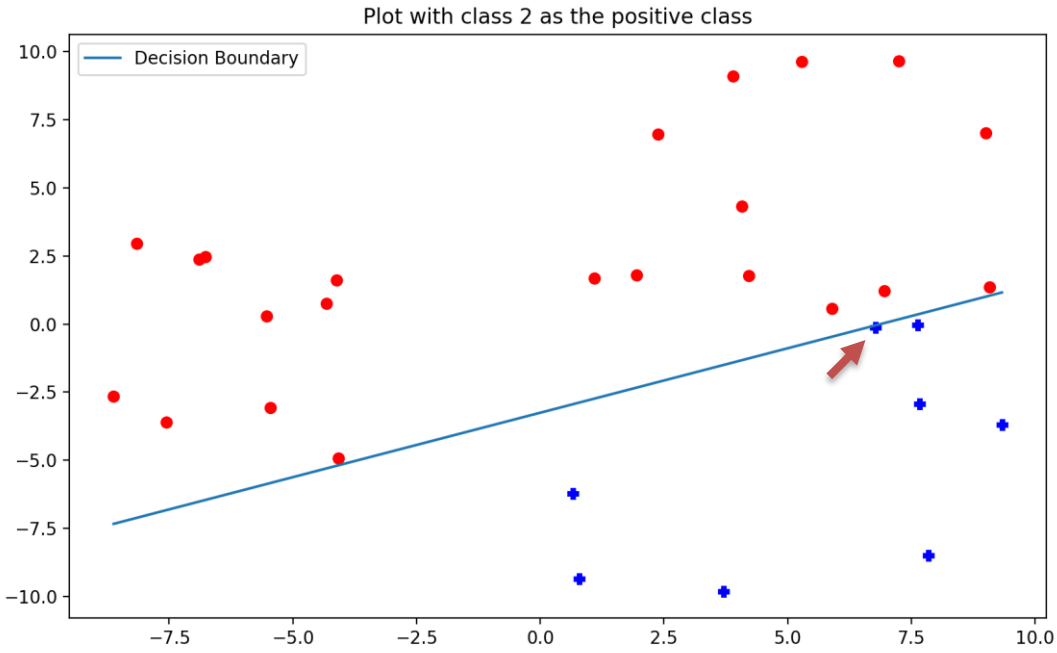


B6. Using the file run_synth_data_1_vs_all.py, capture the plots of all the One-Vs-All classifiers for the synthetic data and paste them here. (Let the maximum number of epochs be 50 and learning rate 0.5).

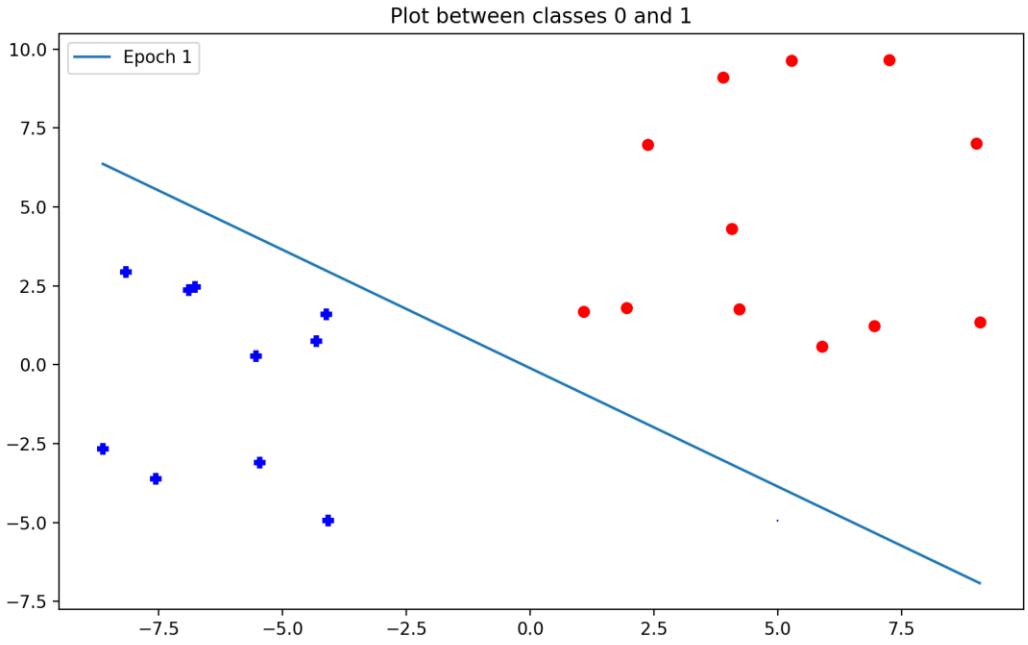
POSITIVE = 0 converged in 2 epochs, POSITIVE = 1 converged in 10, and POSITIVE = 2 converged in 49 as depicted by the busy plot below. The final separator in the first plot matches the assignment notes.

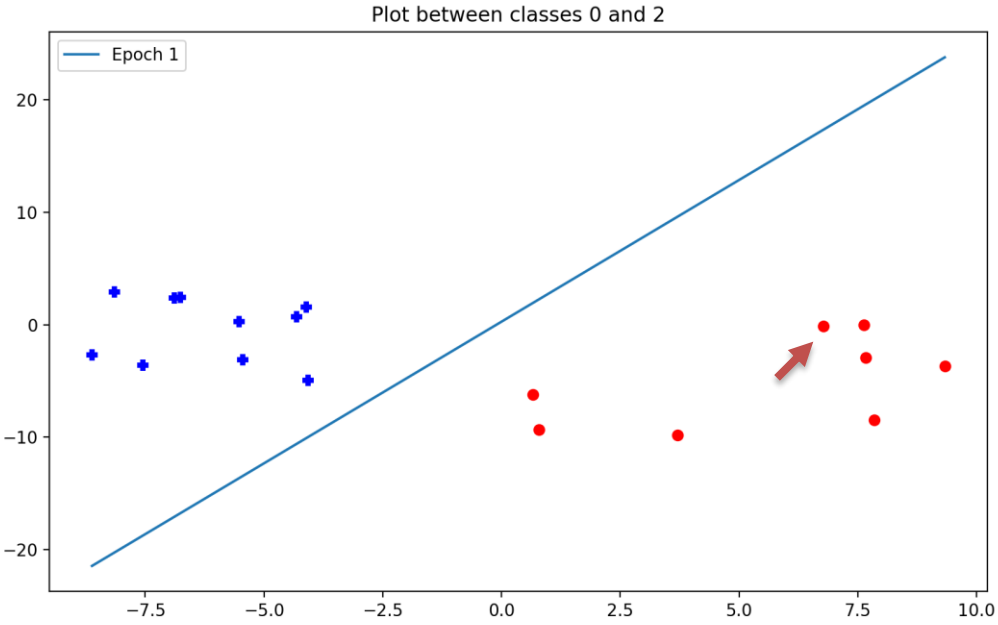


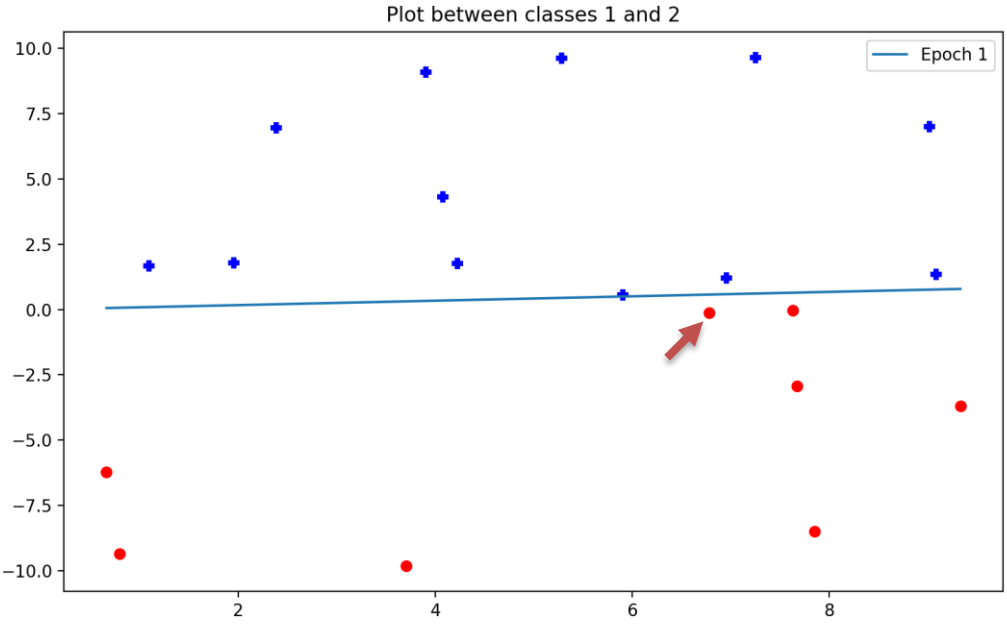




B7. Using the file run_synth_data_1_vs_1.py, capture the plots of all the One-Vs-One classifiers for the synthetic dataset and paste them here. (Let the max number of epochs be 50, and learning rate 0.5)







B8. Using the One-Vs-All classifier, classify the point [6.78, -0.12] as either in class 0, 1, or 2. Briefly explain how you got that class using the individual classifiers. Repeat the same process for One-Vs-One.

In all plots, the positive class is shown in blue. The One-Vs-All shows us in the third plot of B6 with 2 as the positive class that [6.78, -0.12] is in class 2. Since the data point is close to the separation manifold between classes 1 and 2, we can confirm this with the second plot, which shows the point in red to denote that it is not in class 1. One-Vs-One again allows us to look at multiple plots, the most helpful of which is the plot between classes 1 and 2, which shows [6.78, -0.12] in red denoting class 2. I have identified the data point with red arrows in the plots above.