Tyler Arndt
COMP 479
Homework 2
9/21/2019

For the linearly separable dataset, I created 15 different samples, each with two features ranging from 0.0 – 5.0 (Figure 01). After implementing my own version of the perceptron algorithm, I found that it was able to converge by the fourth epoch, as the algorithm no longer needed to update its weights to correctly predict the output label of the data (Figure 02). This provided evidence that the algorithm had found a decision boundary.

The second dataset I created was not linearly separable, but also consisted of 15 samples, each with two features ranging from 0.0 to 5.0 (Figure 03). While training this algorithm, it was discovered that I could expect 4 or 5 weight updates for any given epoch (Figure 04). Thus, this is evidence that the algorithm was unable to converge. Running the main.py file in the submission can print both data sets and their results.

Next, I downloaded the Titanic training set and randomly split it into 70% training data and 30% testing data. From there I configured the data in a few different ways:

- Removal of features believed to be irrelevant to the algorithm
  - “PassengerId”, “Name”, “Ticket”
- Removal of features with very little data
  - “Cabin”
- Transform features from strings to integers
  - “Sex”: male = 1, female = 2
  - “Embarked”: C = 1, Q = 2, S = 3
- Determine best approach to fill in empty data cells
  - “Pclass”: use most common class
  - “Sex”: use most common sex
  - “Age”: use average age
  - “SibSp”: use average number of siblings / spouses on board
  - “Parch”: use average number of parents / children on board
  - “Fare”: use average fare
  - “Embarked”: use most common location

After scaling the features, I adjusted the hyperparameters of the Adaline function until I found a combination that reduced the errors over the number of epochs (Figure 05 and Figure 06).

Lastly, I tested this trained algorithm on the testing data where I was able to correctly predict the labels of 270 out of 270 samples (Figure 07). By taking the absolute value of the resulting weights of trained Adaline function, I can assume that the most predictive features are “Age” (-9.6 weight), “Pclass” (7.5 weight) and “Embarked” (6.4 weight) (Figure 07). Running the Adaline.py file will print the adjusted weights as well as the accuracy of the test data predictions.

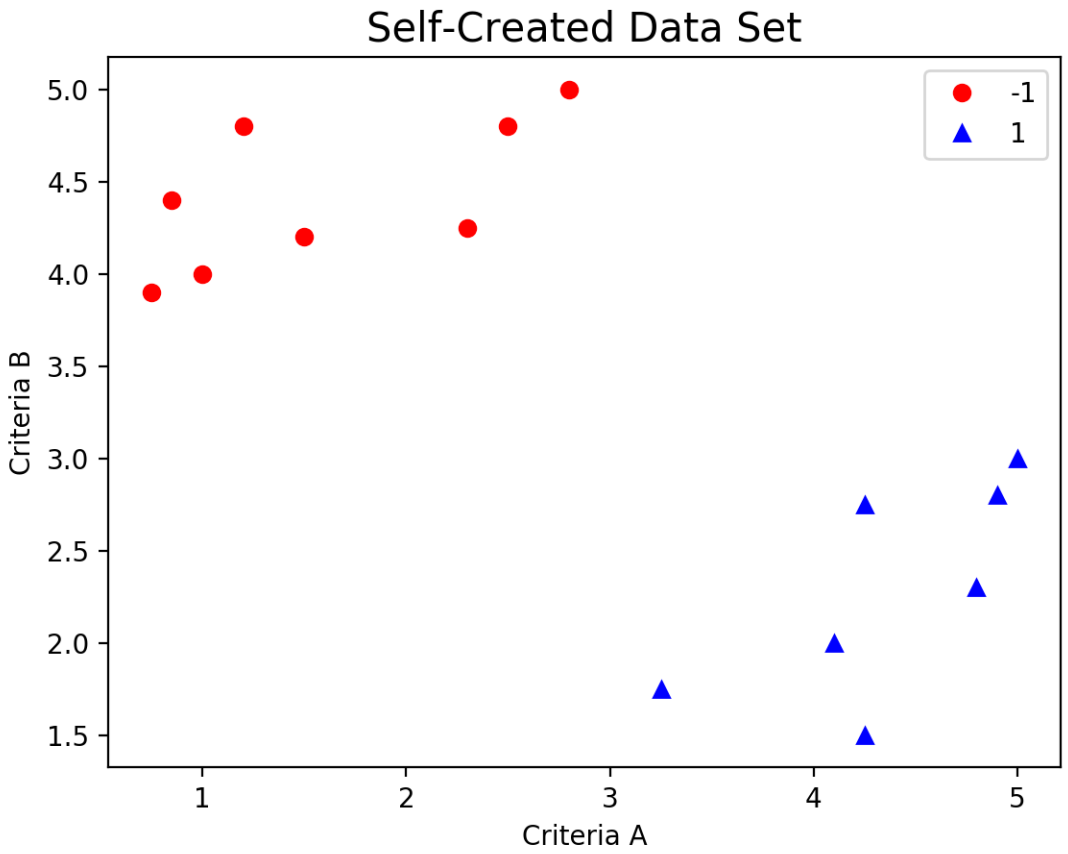

Figure 01

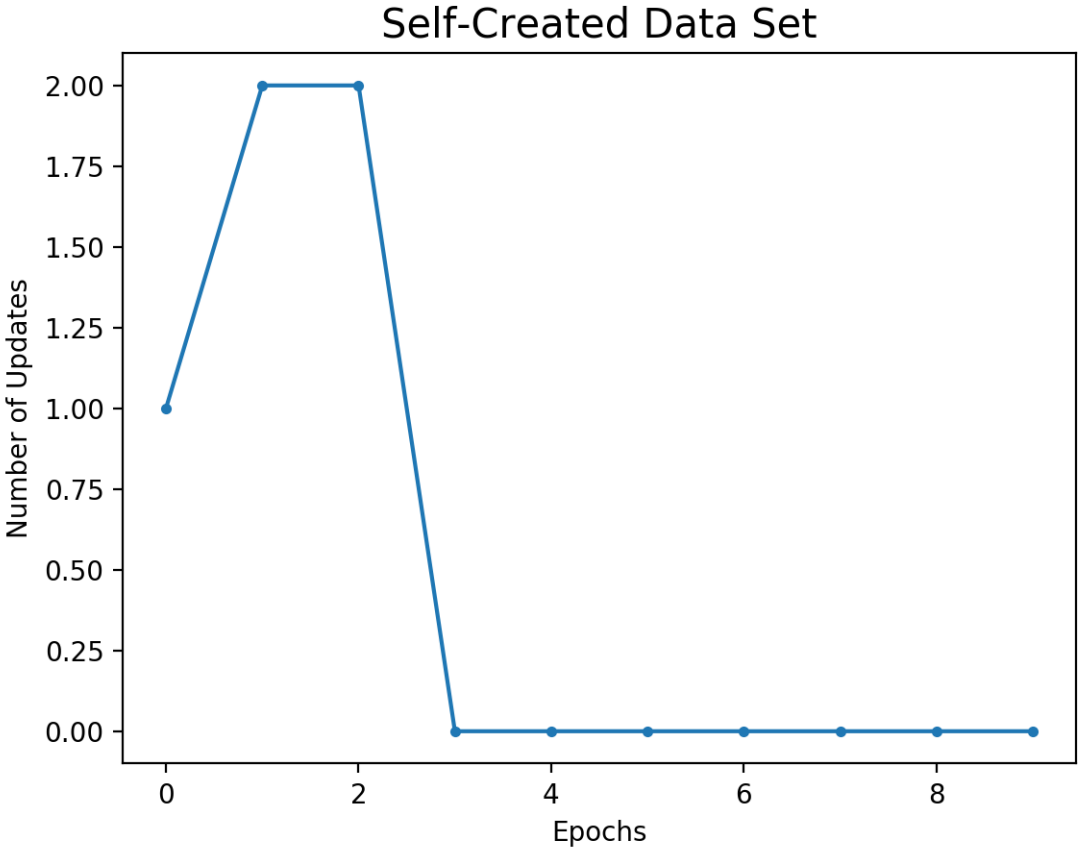

Figure 02

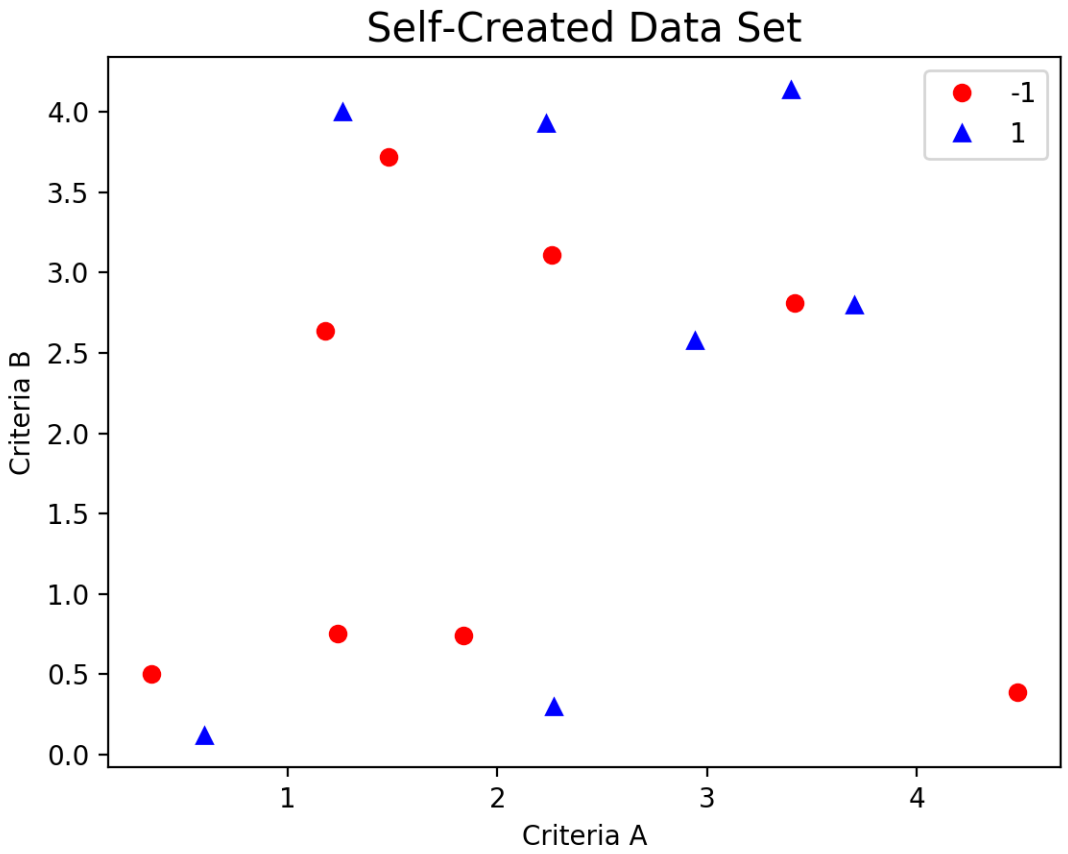

Figure 03

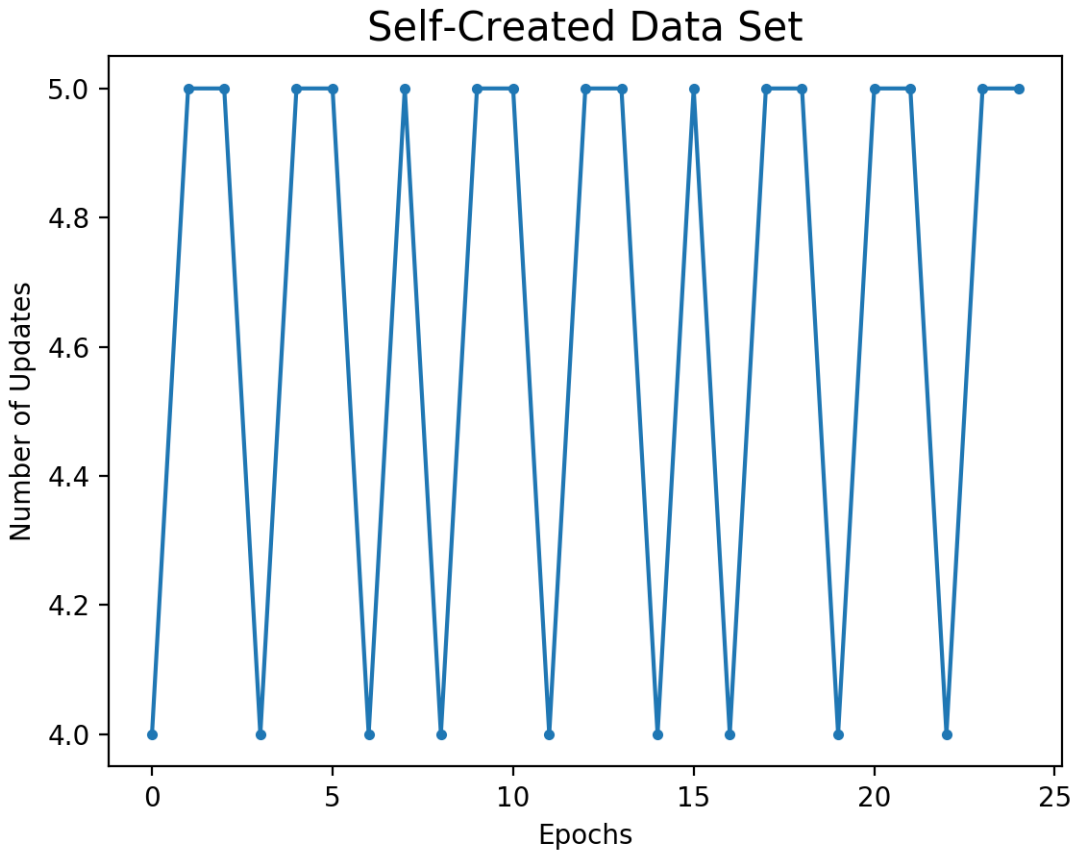

Figure 04

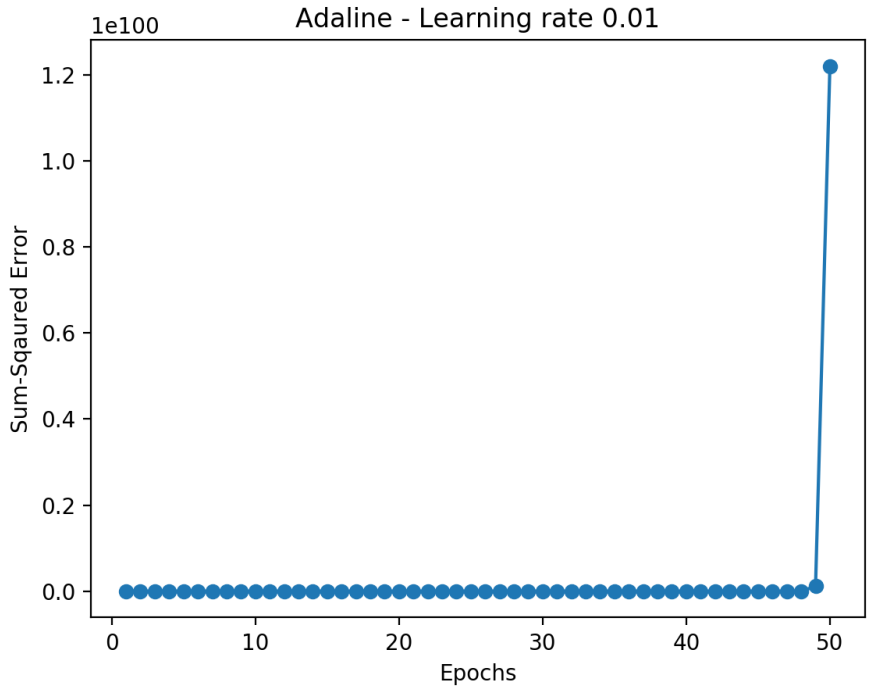

Figure 05

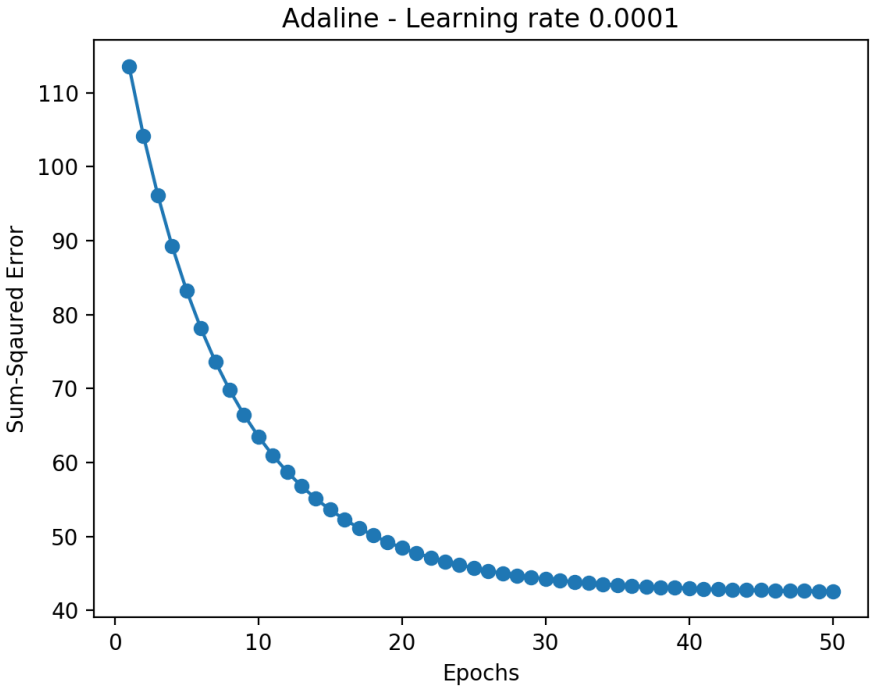

Figure 06

```
[-9.58806893e-01  7.54822408e-04  1.66066657e-04 -9.59700224e-04
  1.71353891e-03 -3.30467633e-03  2.08788782e-03  6.37699369e-04]
The algorithm made 0 mistakes out of 270 predictions for a total accuracy of 100.0%
```

Figure 07