Ananya Srinivasan, Javier Piraneque, Chris Weeden, Kevin Ackerman
CSC 470 - Machine Learning
Project 4
5/10/19

Based on your results, are any of the LPs linearly separable? If so, which one?
Based on the results, we determined that the setosa examples are linearly separable. This is because the setosa examples were able to gradually be classified with no error over time. However, the versicolor and virginica examples are not linearly separable.

T5.1: Create written report, including main points and analysis of results from T2.
The learning algorithm had different levels of effectiveness on the three different learning conditions based on the type of flower. The first being Setosa, was learned rather quickly and through a relatively low number of epochs learned weights which resulted in zero errors. The other two learning conditions aka the other flowers however were not as successful. This is believed to be due to the fact that only the Setosa is linearly separable, and therefore was the only data set which was able to resolve to 0 error.

T5.2: Create written report, including main points and analysis of results from T3.
In the case where the initial weights are all set to 1, the setosa examples were quickly learned within 4 epochs. The versicolor and virginica, much like in the case where the initial weights were all 0, display varied errors. The versicolor examples are particularly erratic, although there was a period between epochs 5 and 16 where the errors stabilize to 2. The virginica ultimately stabilizes at 2 errors. The behavior of the versicolor and virginica examples demonstrates that examples that are not separable are not assured to converge under the perceptron training rule. *Please refer to the plots and results in the weight1 folder.*

When we randomly set the weights for training, we can see behavior similar to the previous case. The setosa examples, however, takes 6 epochs to learn and converge. The versicolor examples, much like in the previous case, have periods of stabilized error values (between epochs 2 to 13, 14 to 15, 16 to 18, and 19 to 20). However, the errors do not stabilize to 0. The virginica examples, much like in the previous case, stabilizes to 2 errors over epochs. *Please refer to the plots and results in the weightrand0 folder.*

We trained the examples again with a new set of random values, and found similar patterns between these plots and those of the previous randomized weight task. This time, the setosa takes 5 epochs to learn. The virginica and versicolor, much like in the previous task, tend to stabilize their learning with 2 errors.

T5.3: Create written report, including main points and analysis of results from T4.
With our first shuffled dataset, the setosa training examples were learned within two epochs. The errors for the versicolor are drastically higher compared to previous training tasks. For instance, the highest amount of training errors in an epoch for the versicolor was 64 errors. The remaining errors range between 47 to 61. While in previous tasks, there is a drastic reduction of errors between epochs for the virginica examples, the decline in errors for the

virginica examples is more gradual in this task. In fact, the plot of errors for virginica appears to be oscillating beginning from epoch 29. The previous tasks, on the other hand, showed that the errors for the virginica examples stabilized around 2 errors. *Please refer to the plots and results in the shuff0 folder.*

When we conducted the learning problems for another randomized dataset, we found similar trends between these plots and those of the previous randomized task. The setosa, like that of the other randomized training set, converges within 2 epochs. The error range for the versicolor examples has widened to range between 36 to 60. The virginica plot has a similar gradual decline in errors to the other shuffled dataset plot, with minor oscillations. *Please refer to the plots and results in the shuff1 folder.*

The results of the random dataset compared to the previous task results demonstrate that the order of the training dataset has a correlation with the number of errors found in the perceptron's classification of the example. This is probably due to the fact that the weights in the unshuffled dataset tasks are quickly adjusted, since the examples for each learning problem are grouped together, which allows for faster learning. However, this is not the case with the shuffled dataset. The weights are adjusted based on the given data, so they do not adjust as quickly to the specific learning problem. This means that the errors will be much higher, as the weights need more time to be adjusted based on the shuffled dataset. For instance, in virginicashuff1.txt shows the first weight gradually reducing from -0.5 by increments of -0.1 to -0.3. The other weights are also incremented. The virginica1.txt, on the other hand, shows how the first weight remains relatively constant, while the other weights are incremented. More weights had to be changed for the shuffled dataset tasks, as a result.

(1) anything positive you enjoyed or learned from this assignment:
We enjoyed implementing the perceptron, as it allowed us to notice the main setback of the perceptron training rule first hand (inability to effectively handle non-linearly separable examples) . It was interesting to see how the order of training examples and the value of the initial training weights had an effect on the overall learning process.

(2) anything negative you didn't like about this assignment:
None

(3) any parts of this assignment you found easy:
Since we were familiar with the general algorithm/formulas for the perceptron learning rule, it was easy to implement.

(4) any parts of this assignment you found challenging or couldn't get working correctly:
We initially had trouble stopping the training process at a point where the number of errors stabilized to a non-zero value.

(5) how your team functioned, including details such as what each team member contributed, how the team communicated with each other, and how team software development & design was accomplished:

We regularly contacted each other over GroupMe throughout the project. Most of us met in person two times to discuss logic and group code the project. If any of us had doubts about the algorithm (for instance, about updating the weights), the rest of us would clarify and fix the issue in the code. Version control of our project was maintained through GitHub.

(6) any other remarks you want to make:
None