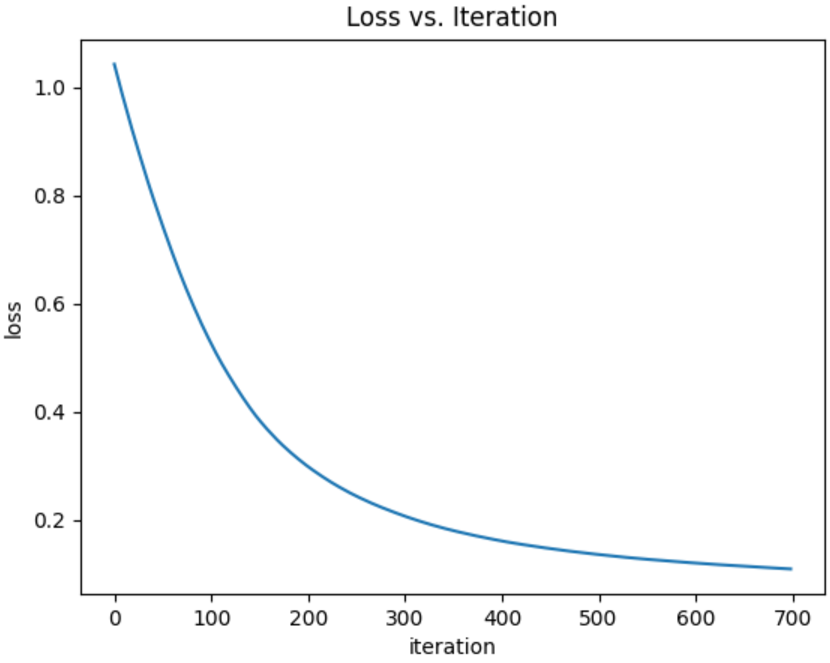
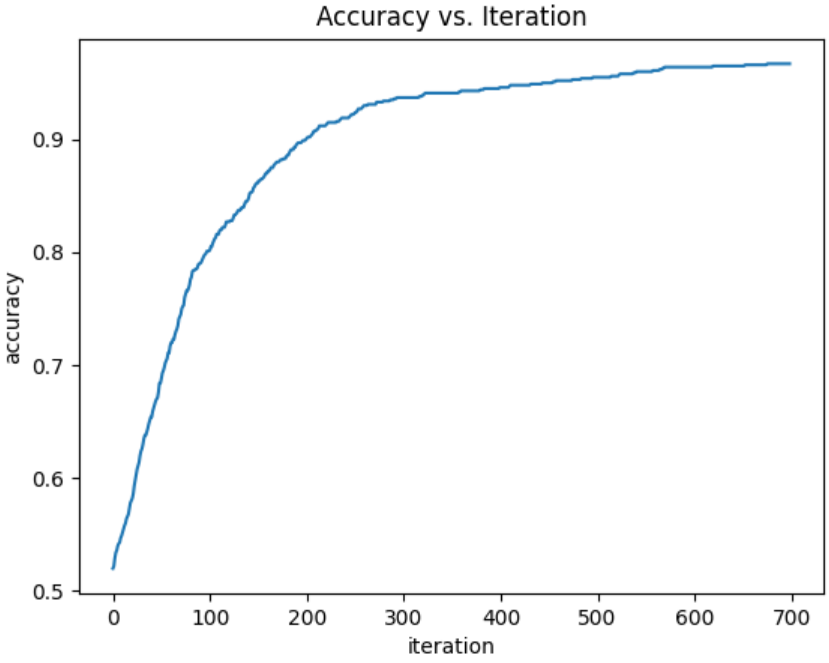
Christopher Tsai

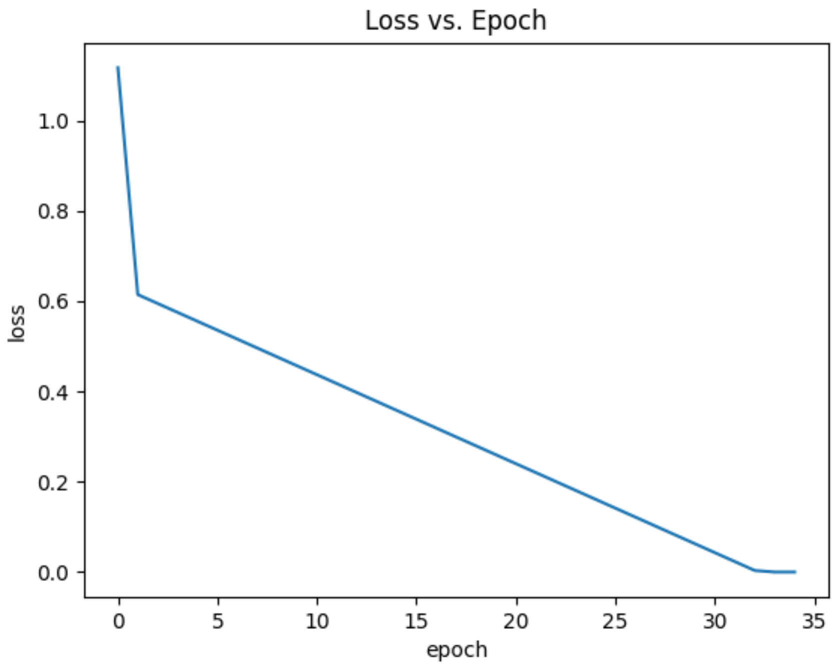
CS 349 HW #5 Free Response Questions

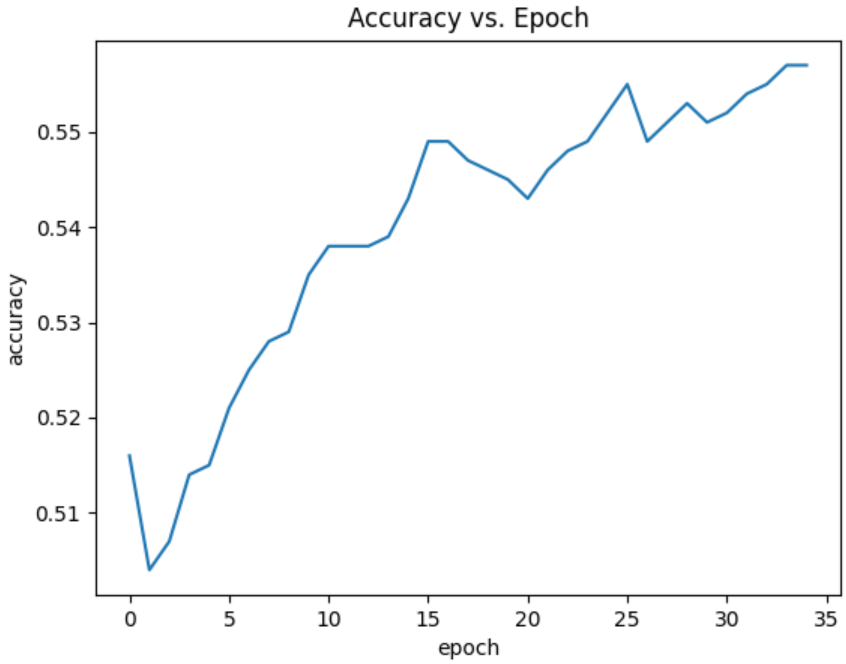
1a:





1b:

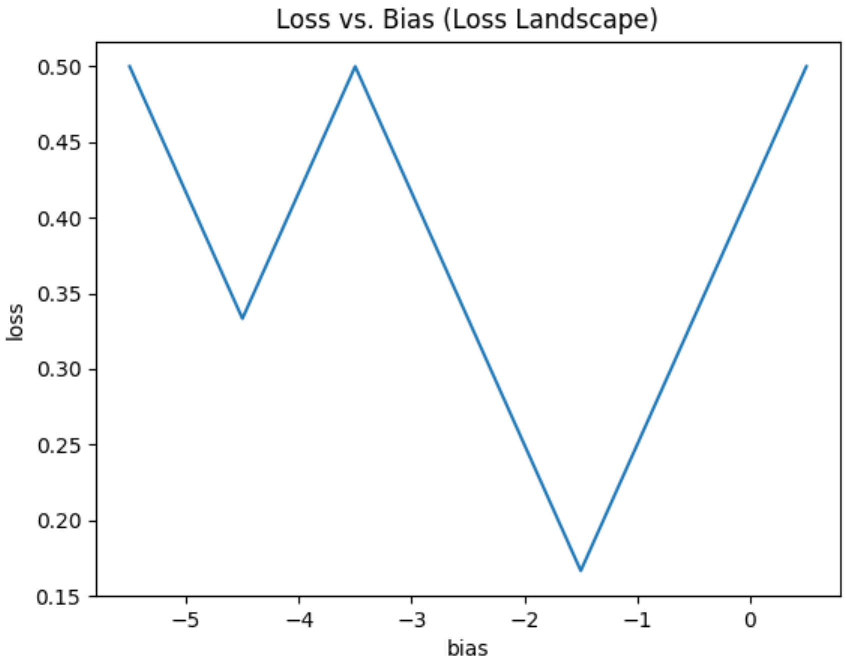

Loss vs. Epoch
loss
1.0
0.8
0.6
0.4
0.2
0.0
0
5
10
15
20
25
30
35
epoch


Accuracy vs. Epoch
accuracy
0.55
0.54
0.53
0.52
0.51
0
5
10
15
20
25
30
35
epoch

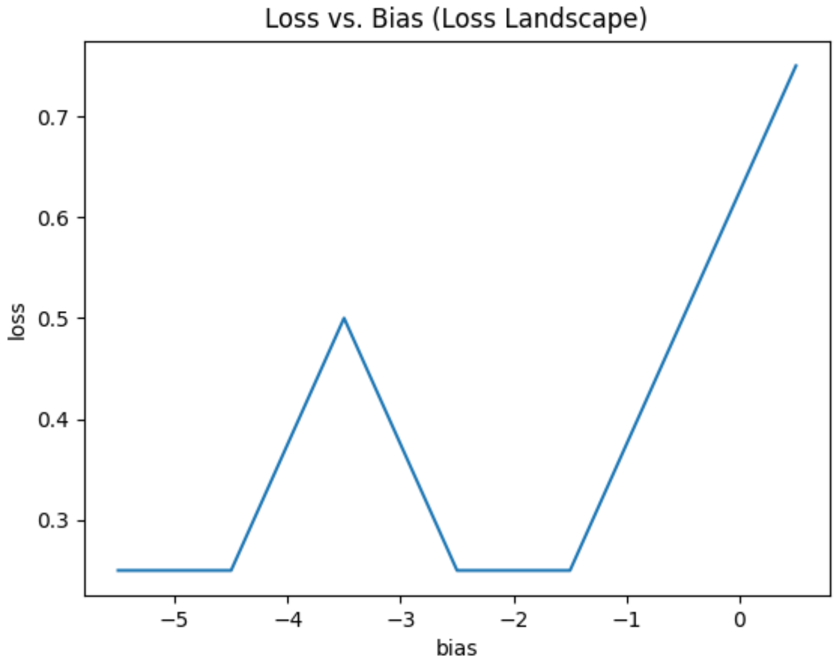
1c:

Stochastic gradient descent is faster to train, normal gradient descent converges to a lower loss better, but difference is not too large in the above case. Stochastic gradient descent has a random element to it, which reduces its reliability.

2a:



2b:

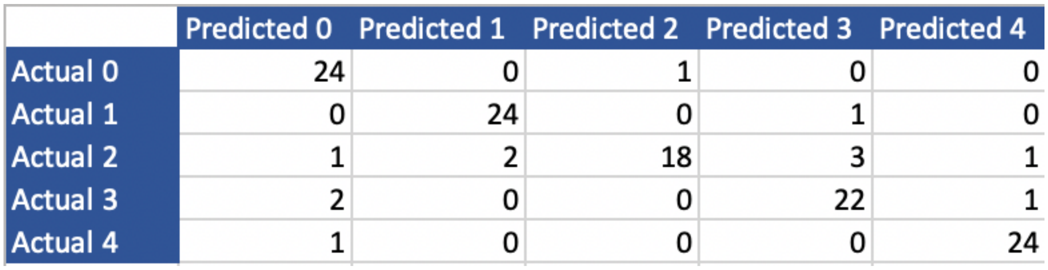


2c:

Batching can introduce variation in the loss landscape and we typically won't know how the landscape changes. Even small changes in data selection can introduce large changes in loss landscape. Thus, the learner could converge to different parameter values if one isn't lucky.

It did take me a few tries of changing the batch for this question before the minimum changed, though. I assume batching is usually safe to do but one just doesn't know when changes like the one above can happen.

3a:



| | Predicted 0 | Predicted 1 | Predicted 2 | Predicted 3 | Predicted 4 |
|---|---|---|---|---|---|
| Actual 0 | 24 | 0 | 1 | 0 | 0 |
| Actual 1 | 0 | 24 | 0 | 1 | 0 |
| Actual 2 | 1 | 2 | 18 | 3 | 1 |
| Actual 3 | 2 | 0 | 0 | 22 | 1 |
| Actual 4 | 1 | 0 | 0 | 0 | 24 |

Clearly, most numbers are predicted correctly most of the time (as indicated by the high diagonal values). However, my model struggled with mistaking an actual 2 with other numbers, especially a 3. This makes sense as 2 and 3 are arguably the numbers that are most similar to each other. My model also mistook an actual 3 with a 0 twice, and this could be due to the oval nature of both numbers. 0, 1, and 4 are the easiest to distinguish, as only 1 misprediction happened per number.

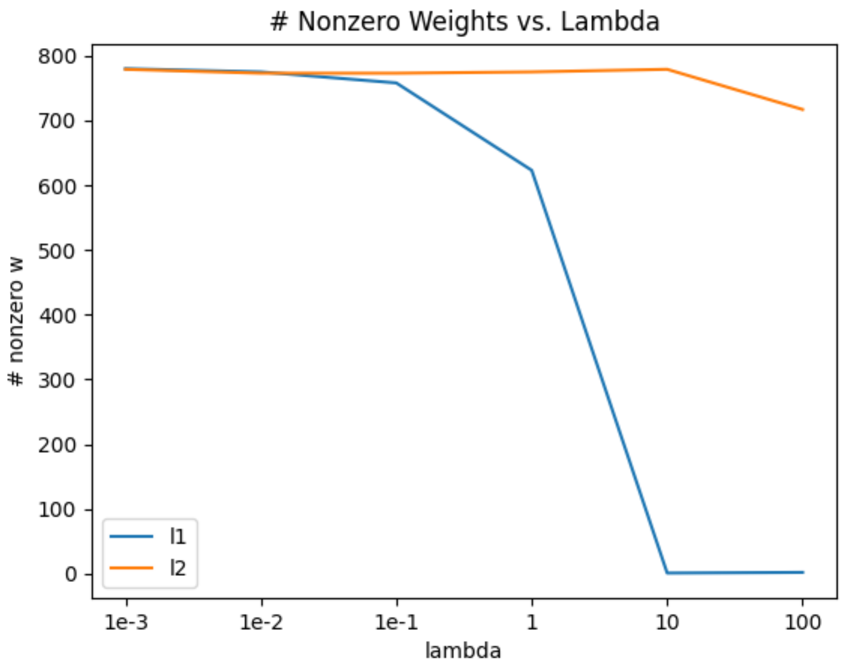
3b:

Only c binary classifiers are needed for OVA, one per class (as OVA just compares each class to everything else). Space complexity is O(n).

3c:

c*d binary classifiers are needed for OVO. Space complexity is O(n^2).

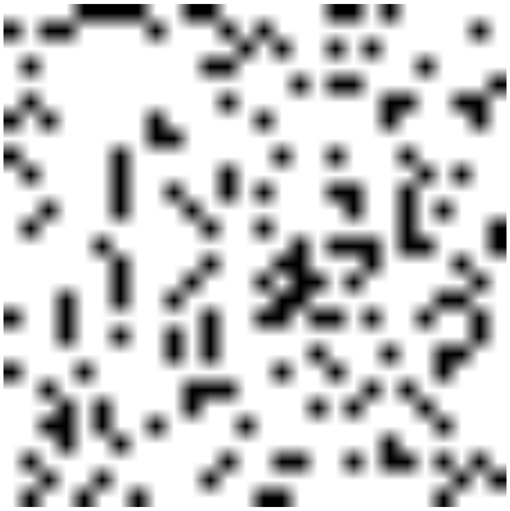
4a:



4b:

The L1 regularizer produces more sparsity because the its gradient (which is a constant positive or negative number) moves model parameters towards 0 at a constant rate rather than L2's gradient, which is a constantly decreasing function near. This sparse property allows L1 regularization to be more computationally efficient (as we don't have to compute predictions whose coefficient is 0), and this can matter a lot with large datasets.

4c:



Concentrated in small batches in random spots.