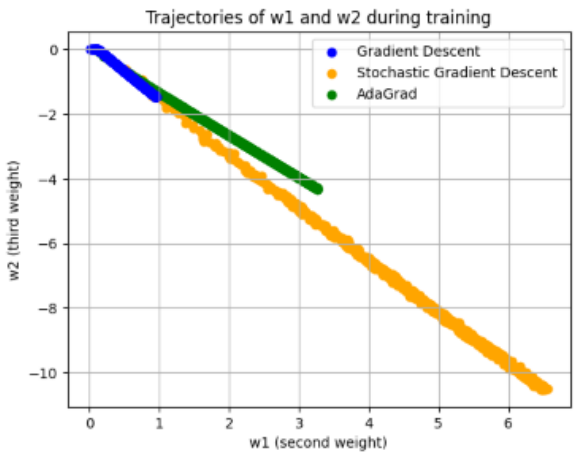
2 Unconstrained Loss Optimization for ML (Programming) (15 points)

In this question, you will implement *Gradient Descent*, *Stochastic Gradient Descent*, and *AdaGrad*. Please code the update rules for each of these optimization techniques in the `loss_optimization.py` file provided to you. After you are done, you will submit your code to Gradescope, where we wil run your code on a full suite of tests and the autograder will assign your code points based on whether it passes the tests. After implementing the update rules, please answer the following questions:

1. (3 points) Create a plot for the 2D parameter space w_1, w_2 corresponding to the non-bias weight parameters. These are the second and third elements in the weight vector since the first element w_0 corresponds to the bias term. In this plot, show the trajectory of the weight parameters captured at the end of each epoch as they are updated during the training process. The plot should include three trajectories, one for each of the three parameter update rules. Include a legend indicating which trajectory corresponds to which optimizer.

Plot



Correction attempt made

1.1 (no title) 1.5 / 3 pts

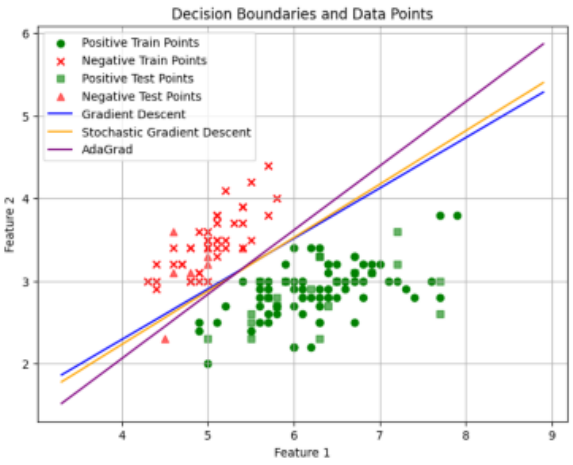
− 0 pts Correct

− 1.5 pts Error

The plot should consist of 3 trajectories, not just points.

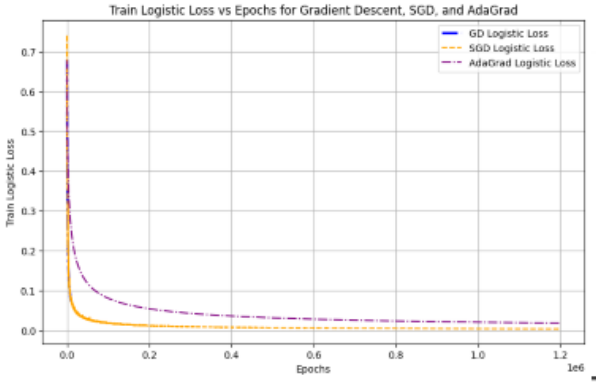
2. (3 points) Plot the two-dimensional data with the positive datapoints (label=1) in green and negative datapoints (label=0) in red. Display the train and test datapoints using different market shapes. Also include in the plot the classifier decision boundary learned by the three different optimizers. Include a legend indicating which decision boundary corresponds to which optimizer.

Plot



3. (3 points) Plot the curve for train logistic loss versus train epoch for each of the three optimizers in a single plot. Include a legend indicating which curve corresponds to which optimizer.

Plot



Just so you know the GD Logistics loss is depicted here, it's just hard to see because of its similarity to the SGD Logistics loss.

Correction attempt made

1.3 (no title) 1.5 / 3 pts

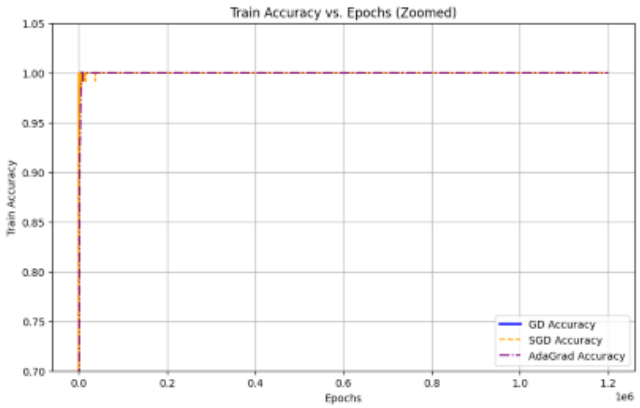
– 0 pts Correct

– 1.5 pts Error

Missing a line and the loss curves are incorrect

4. (3 points) Plot the curve for train accuracy versus train epoch for each of the three optimizers in a single plot. Include a legend indicating which curve corresponds to which optimizer.

Plot



Just so you know the GD Accuracy is depicted here, it's just hard to see because of its similarity to the SGD Accuracy.

Correction attempt made

1.4 (no title) 1.5 / 3 pts

– 0 pts Correct

– 1.5 pts Error

The plot should have 3 curves and shouldn't just be at 1.

5. (1 point) What is the accuracy obtained by the gradient descent optimizer on the test data?

Solution

0.967

```
Test Accuracy (Gradient Descent): 0.9666666666666667
Test Accuracy (Stochastic Gradient Descent): 0.9666666666666667
Test Accuracy (AdaGrad): 0.9666666666666667
```

6. (1 point) What is the accuracy obtained by the stochastic gradient descent optimizer on the test data?

Solution

0.967

```
Test Accuracy (Gradient Descent): 0.9666666666666667
Test Accuracy (Stochastic Gradient Descent): 0.9666666666666667
Test Accuracy (AdaGrad): 0.9666666666666667
```

7. (1 point) What is the accuracy obtained by the Adagrad optimizer on the test data?

Solution

0.967

1 Search (5 points)

For each statement, state which search algorithm it applies to. Multiple choices may be selected.

1. (1 point) Commonly used to decode a vector during machine translation

   - [x] Beam Search
   - [ ] A* Search
   - [ ] Greedy Search
   - [ ] None of the above

   Extra Credit? 0-0"

2. (1 point) Requires use of heuristics in its algorithm

   - [ ] Beam Search
   - [x] A* Search
   - [x] Greedy Search
   - [ ] None of the above

3. (1 point) Involves using BFS to build a search tree

   - [ ] Beam Search
   - [ ] A* Search
   - [ ] Greedy Search
   - [x] None of the above

   Could be this Beam search... I guess but we'll go with none of the above

4. (1 point) Guaranteed to find shortest path on a weighted graph regardless of the heuristic used

   - [ ] Beam Search
   - [ ] A* Search
   - [ ] Greedy Search
   - [x] None of the above

5. (1 point) Guaranteed to find shortest path on a weighted graph given an admissible and consistent heuristic

   - [ ] Beam Search
   - [x] A* Search
   - [ ] Greedy Search
   - [ ] None of the above