EE 559 **Homework 3** Posted: Feb. 3, 2023
Chugg /Jenkins Due: Feb 10, 2023, 11:59 PM

Significant updates in blue

1. For a linear two-class classifier, show that $\underline{w}$ is orthogonal to the decision boundary H. **Hint:** if $\underline{x}_1$ and $\underline{x}_2$ are on the decision boundary, $g(\underline{x}_1) = g(\underline{x}_2) = 0$.

2. Let $p(\underline{x})$ be a scalar function of a D-dimensional vector $\underline{x}$, and $f(p)$ be a scalar function of p. Prove that:

$$\nabla_{\underline{x}} f\left[p(\underline{x})\right] = \left[\frac{d}{dp} f(p)\right] \nabla_{\underline{x}} p(\underline{x})$$

i.e., prove that the chain rule applies in this way. [**Hint:** you can show it for the i^{th} component of the gradient vector, for any i. It can be done in a couple lines.]

3. Find the following gradients:

   a. $\nabla_{\underline{x}} \left(\underline{x}^T \underline{x}\right)$

   b. $\nabla_{\underline{x}} \left[\left(\underline{x}^T \underline{x}\right)^3\right]$

   c. $\nabla_{\underline{w}} ||\underline{w}||_2^2$

   d. $\nabla_{\underline{w}} \left\|\underline{w}\right\|_2$

   e. $\nabla_{\underline{w}} ||\underline{\underline{M}}\, \underline{w}||_2^2$

   f. $\nabla_{\underline{w}} \left\|\underline{\underline{M}}\underline{w} - \underline{b}\right\|_2$

4. Code up a 2-class perceptron learning algorithm and classifier. For this problem, you may use only python built-in functions, numpy, matplotlib; you may use the PlotDecBoundaries.py function provided with Homework 1; and you may use pandas only for reading and/or writing csv files.

Please observe the following:

(i) For the optimization, use basic sequential GD. Also use SGD variant 1, which is the same as basic sequential GD, except a shuffle is done at the start of every epoch whereas in basic sequential GD a shuffle is done only before the first epoch and the order is held fixed thereafter.

(ii) For the initial weight vector, use $\underline{w}(0) = a\underline{1}$, in which $a = 0.1$.

(iii) For the learning rate parameter, use $\eta(i) = 1 \ \forall_i$.

(iv) For the halting condition, use 2 conditions, such that it halts when either one is met:

i.1 When all the training data points are correctly classified. In this case, it also outputs the statement "data is linearly separable".

i.2 When 10,000 iterations have been performed. In this case, choose for the final weight vector $\underline{\hat{w}}$ as the weight vector corresponding to the lowest $J(\underline{w})$ over all iterations.

(v) You will also need a function that classifies any given data point, using the optimal $\underline{\hat{w}}$ from the learning algorithm.

Please answer the questions, or proceed as instructed, below:

(a) For the synthetic dataset1 given with Homework 1, implement the following:

(i) Run the perceptron learning algorithm to find $\underline{\hat{w}}$.

Give the resulting $\underline{\hat{w}}$ vector; state whether the algorithm converged (i.1 reached) or halted without convergence (i.2 reached); and give the final criterion function value $J(\underline{\hat{w}})$.

(ii) Produce a learning curve, which is a plot of the values of the criterion function during the training process. If the training goes for more than 10 epochs, plot the criterion function vs. epochs. If training ends before 10 epochs, plot the criterion function vs. iterations. Recall that one epoch is one pass through the entire training set while one iteration is one weight vector update. You may also want to include the misclassification rate vs. epoch (or iterations) on this plot as well.

(iii) Run the perceptron classifier on the training set and the test set using the final $\underline{\hat{w}}$. Give the classification error of each.

(iv) Plot in feature space the training data points, decision boundaries, and decision regions. The decision boundaries and regions should use the final $\underline{\hat{w}}$.

(b) Repeat part (a) except use the synthetic dataset2 datasets given with Homework 1

(c) Repeat part (a) except use the synthetic dataset3 datasets given with Homework 1.

(d) Repeat part (a) except use the breast cancer dataset. The data is provided in a .npy file for both train and test. The training data comprises 480 data points and the test set is 31 points. The feature vector has dimension D=30. The data has 31 dimensions, with the first component being the label: 1 = Malignant, 2 = Benign. More details and resources will be posted with the data. When working with the breast cancer data, you will find that the features vary significantly in their dynamic range. You should normalize this data. You can use the following to normalize the BC data. Specifically, a recommended normalization that works well with the weight vector initialization is:

```
Get the L1 norms of the train data and nromalize the train data:
## x_train.shape is (N, D)
x_train_L1_norms = np.linalg.norm(x_train, ord=1, axis=0)  ## this is a
length D array
x_train_normalized = 100 * x_train / x_train_L1_norms

## now use these parameters to normalize the test data
# x_test.shape is (N_test, D)
x_test_normalized = 100 * x_test / x_train_L1_norms
```

It is a good idea to visually inspect (i.e., do some plotting) of your data before and after normalization. Also answer the following:

(iv) Is the 2-class data linearly separable? Answer yes, no, or don't know. Briefly justify your answer. This dataset has a 30 dimensional feature vector, so you cannot produce the plot requested in part (a)-(iv). Instead produce a histogram of the distance from the decision boundary ($g(\underline{x})/||\underline{w}||$). On the same plot, produce a histogram of this quantity for all class 1 training data and another for all class 2 training data. Put these on the same plot with different colors for the class1 and class 2 histograms as demonstrated in the example notebooks.

5. In lecture, we defined the criterion function for a 2-class Perceptron Learning problem (in augmented space) as:

$$J(\underline{w}) = -\sum_{n=1}^{N} [\![\underline{w}^T z_n \, \underline{x}_n \leq 0]\!] \, \underline{w}^T z_n \, \underline{x}_n$$

(a) Rewrite the criterion function using $max\{\cdot\}$ function and the rectified linear unit (ReLu) function defined by $ReLu(x) = max(0, x)$

(b) Consider replacing the ReLu(.) function with the soft(.) function defined by $soft(x) = ln(1 + exp(x))$ in this criterion function to produce a new criterion function $J_L(\underline{w})$. This results in logistic regression.

a. Plot ReLu(x) function with the soft(x) vs x. Discuss the change in the criterion function when this soft(.) function is used. Specifically, for perceptron learning, the criterion function penalizes only errors with a loss proportional to the distance from the decision boundary. Is this still the case?

b. Find the gradient of $J_L(\underline{w})$. You may utilize the sigmoid function $\sigma(v) = e^v/(1 + e^v) = 1/(1 + e^{-v})$ in expressing the gradient.

(c) Repeat problem 4(d) (i.e., the breast cancer data) using this Logistic regression and compare the results to that obtained with perceptron learning.