CEE 554: Data Mining in Transportation

Homework Assignment 1
Due date: 01/28, 11:59pm on Canvas

- **You can either type your answers or handwrite them and then scan the document. If you choose to handwrite, ensure your writing is legible and the scan quality is clear. Illegible submissions will not be awarded points.**
- **Add Appendix A at the end of each homework assignment, specifying whether you used Generative AI in developing your solution, the Generative AI engine you used, and the command you used for each problem.**
- **Add Appendix B at the end of the assignment, including the code you used for each problem.**

1. In this problem, we will explore a basic perceptron learning algorithm on linearly separable and non-separable data. Provide your code in an appendix at the end of your homework.

   (a) Prepare your dataset:
   - Load the iris dataset
   - Use the first two columns as your input/attribute/feature vector
   - The last column is the label of each data point. As you can see, there are a total of three labels: 0, 1, and 2. We would like to create two datasets out of the iris dataset as follows:
     - Dataset A: Take all the data points with labels 0 and 1. Then convert the labels to -1 and 1 using the formula $y_i \leftarrow 2y_i - 1$
     - Dataset B: Take all the data point with labels 1 and 2. Then covert the labels to -1 and 1 using the formula $y_i \leftarrow 2y_i - 3$

   (b) Show the two datasets in two scatter plots and verify that one is linearly separable. **(5 points)**

(c) Use the Perceptron Learning Algorithm (PLA) on dataset A. Plot both the data points and the final decision boundary. Plot the value of the misclassification rate as a function of iteration. **(15 points)**

(d) Use the perceptron classifier on dataset A again, this time shuffling the data first (shuffling refers to randomly changing the order of points in your dataset). Repeat this process 10 times, each time recording the number of iterations it takes for the algorithm to stop. Does the ordering of data points in your dataset play a role on the behavior of PLA? **(5 points)**

(e) Use the Perceptron Learning Algorithm (PLA) on dataset B. Plot both the data points and the final decision boundary. Plot the value of the misclassification rate as a function of iteration. Specify what stopping criteria you are using. **(20 points)**

(f) Modify the code so that you can always keep the best perceptron found in your "pocket". Show the last perceptron, and the best perceptron found by the pocket algorithm in two figures side by side. **(10 points)**

2. Consider a linear classifier with a logistic-MSE cost function (with a linear signal) as we saw in class. When our target values have +1 and -1 labels (i.e., $y_i \in \{-1, +1\}, i = 1, \cdots, N$), The MSE loss function looks like the following:

$$C(W) = \frac{1}{N} \sum_{i=1}^{N} \left(\rho(W^T X_i) - y_i\right)^2$$

Where $\rho(z) = 2\sigma(z) - 1$, and $\sigma(z) = \frac{1}{1+e^{-z}}$.

(a) Use the above equations to derive the **stochastic gradient descent** update rule. **(5 points)**

(b) Use this linear classifier on dataset A: **(20 points)**

- Draw the final classifier and the training data points in one figure
- Report the changes in the cost function and error rate per iteration.
- Show the final perceptron classifier from your previous homework and the logistic-MSE classifier side by side, and draw comparisons between their performances.

(c) Use this linear classifier on dataset B: **(20 points)**

- Draw the final classifier and the training data points in one figure
- Report the changes in the cost function and error rate per iteration.
- Show the final perceptron classifier from your previous homework and the logistic-MSE classifier side by side, and draw comparisons between their performances.

You might have to play with parameters in your code, such as the max number of iterations, tolerance, and step size. Additionally, we know that this method only provides a local minimum. Therefore, you have to run the code multiple times, each time starting with a random weight vector W, to obtain an acceptable classifier.