Homework 1

Please upload your assignments on or before September 18, 2020.

- You are encouraged to discuss ideas with each other; but
- you **must acknowledge** your collaborator, and
- you **must compose your own** writeup and/or code independently.
- We **require** answers to theory questions to be written in LaTeX, and answers to coding questions in Python (Jupyter notebooks)
- Upload your answers in the form of a single PDF on Gradescope.

---

0. **(0.5 points)** Introduce yourself on the HW1Q0 thread on the class Piazza! Mention a little bit about your background, interests, and why you wish to learn about neural nets and deep learning.

1. **(1.5 points)** *Fun with vector calculus.* This question has two parts.

   a. Assume we have n real-valued scalar data points $x_1, x_2, \dots, x_n$. Analytically derive a constant μ for which
   $$\sum_{i=1}^{n} (x_i - \mu)^2$$
   is minimized.

   b. Now assume that the n data points are real d-dimensional vectors. Analytically derive a constant vector μ for which
   $$\sum_{i=1}^{n} \|x_i - \mu\|_2^2$$
   is minimized.

2. **(2 points)** *Linear regression with non-standard losses.* In class we derived an analytical expression for the optimal linear regression model using the least squares loss for a finite dataset. If X is the matrix of training data points (stacked row-wise) and y is the vector of labels, then:

   a. Using matrix/vector notation, write down a loss function that measures the training error in terms of the ℓ_1-norm.

   b. Can you write down the optimal linear model in closed form? If not, why not?

   c. If the answer to b is no, can you think of an alternative algorithm to optimize the loss function? Comment on its pros and cons.

3. **(2 points)** *Hard coding a multi-layer perceptron.* In class, we derived the functional form for a single perceptron: $y = \text{sign}(\langle w, x\rangle + b)$, where x is the data point. Suppose your data is 5-dimensional (i.e., $x = (x_1, x_2, x_3, x_4, x_5)$) and real-valued. Find a simple 2-layer network of perceptrons that implements the *Increasing-Order* function, i.e., it returns +1 if
   $$x_1 < x_2 < x_3 < x_4 < x_5$$

and -1 otherwise. Your network should have 2 layers: the input nodes, feeding into 4 hidden perceptrons, which in turn feed into 1 output perceptron.

4. **(4 points)** This exercise is meant to introduce you to neural network training using Pytorch. Open the (incomplete) Jupyter notebook provided as an attachment to this homework in Google Colab (or other cloud service of your choice) and complete the missing items. Save your finished notebook in PDF format and upload along with your answers to the above theory questions in a single PDF.