ECE/CS 559: Neural Networks **Fall 2024**

Homework 4

Due: **Tuesday** September 24, 2024 (by 9pm, on Gradescope)

Note: *You may discuss these problems in groups. However, you must write up your own solutions and mention the names of the people in your group. Also, please do mention any solution manuals, online material, or other sources you used in a major way.*

Submit your code as **single** `.py` **file** in Gradescope. Make sure that it **runs properly** and generates all the plots that you report on in the main submission.

1. **[Linear Regression for MNIST]**

   Let's recast a multi-label classification in the MNIST dataset as a regression problem.

   (a) Write code to load the MNIST data in its entirety (you can use any particular way to do this, such as via the `sklearn` library). This will give you `Xraw` and `Yraw` matrices, with 70,000 rows each. Each row in `Xraw` is a 784-dimensional feature (representing a 28×28 image, each pixel ranging in $0, 1, \cdots, 255$). Each row in `Yraw` is an integer representing one of 10 handwritten digits (0 through 9). Plot one example of each digit as an image using `imshow`, after reshaping the 784-dimensional feature into a 28×28 array. (Report both code and plots.)

   (b) Instead of using the raw features and raw labels, pre-process them as follows.
   - Create a $d \times 784$ matrix M by making a matrix with i.i.d. Uniform$([0, 1])$ entries then dividing it by $255d$ (to make sure feature values are not too large.) This will be your (random) feature extractor.
   - Create a $d \times 70,000$ X matrix, by multiplying M with (the transpose of) each row in `Xraw`, reshaped to be a column vector.
   - Create a $10 \times 70,000$ Y matrix, representing the *one-hot encoding* of each row of `Yraw`.

   Then, using the `linalg` library in `numpy` to calculate the Moore-Penrose pseudoinverse, find the best weights W for the linear predictor $f(x; W) = Wx$. (Only report code for this question.)

   (c) Set d at $10, 50, 100, 200, 500$, and find and report two numbers in each case:
   - the MSE of your predictor, i.e., $\sum_{(x,y)} \|y - f(x; W\|^2$, and
   - the number of mistakes if you choose the label to be the coordinate where your predictor is maximal, i.e., $\sum_{(x,y)} \mathbb{1}\{\arg\max y \neq \arg\max f(x; W\}$. (Since y is a one-hot encoding, $\arg\max y$ is the digit label.)

How do your numbers of mistakes compare to the expected number of mistakes if we were to randomly guess the digit? What is a good choice of d and why?

(d) Implement the Widrow-Hoff LMS algorithm instead to find W. Fix $d = 100$. Initialize your weight vector at the origin and use $\eta = 0.001$. Run it for 10 epochs and, at the end of each epoch, calculate and save the MSE of your estimator. Plot the MSE vs. the number of epochs. Also report the number of mistakes you get if you label as in the second point in (c) with the newly obtained W. How does this compare to the exact solution with $d = 100$ in (c)? Suggest changes to get closer to the exact solution and implement your suggestion. Describe any obstacles you run into.