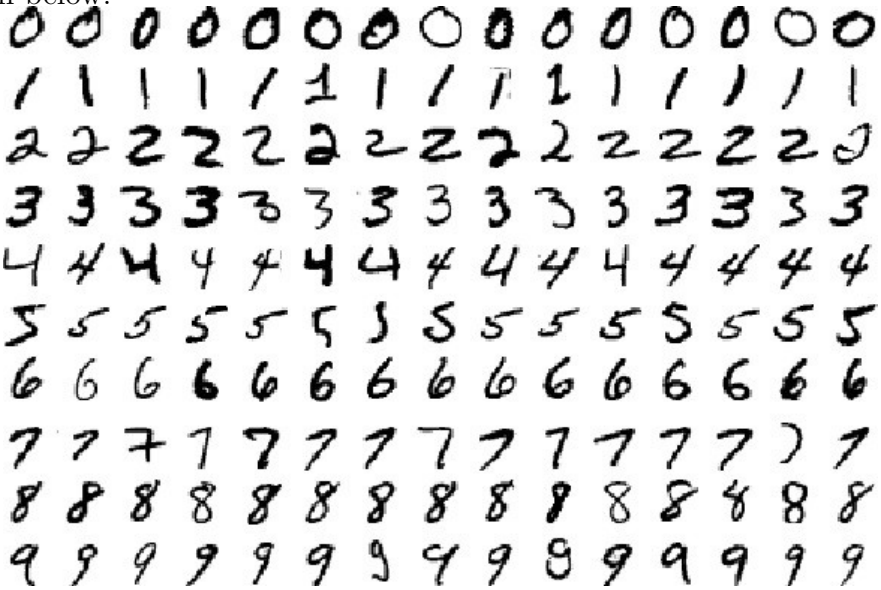
Machine Learning	**F2020**
Problem Set 3	
Handed Out: Oct 9^{th}*,* 2020	*Due: Oct* 23^{rd}*,* 2020

In this assignment you will work with the infamous MNIST dataset! The MNIST dataset contains examples images of the handwritten numbers 0-9. This is a benchmark dataset that is frequently used in machine vision. It just so happens that it is also a dataset that multilayer perceptrons are well equipped to handle! An example of these handwritten numbers can be seen below.



In this assignment we won't be using the full MNIST dataset. I've provided 4 datasets, a training and test set for the numbers 0 and 1 (mnist_train_0_1.csv and mnist_test_0_1.csv) and a training and test set for the numbers 0 through 4 (mnist_train_0_4.csv and mnist_test_0_1.csv). Each training example consists of a comma separated list of 785 values. The first element is the numeric label, and the remaining 784 elements are values for each pixel in a 28x28 image of a handwritten character. The goal is to successfully use these pixel values to successfully classify what number the image corresponds to!

You will submit a writeup in Word or PDF that summarizes your results and all code as a zip file. Submit the writeup (with attached source code) to the Canvas submission locker before 11:59pm on the due date.

Binary Classification Using Multilayer Perceptrons (80 points)

Only two parts to this assignment. For this first part, implement a multilayer perceptron to predict the label of images of handwritten 0's and 1's. To train this network, you will be using

backpropagation. For this code, you need only report the overall accuracy of your technique on the test set as well as a description of your network's architecture. In other words, just report the percentage of the test set that your neural network is able to accurately predict as well as information on the number of hidden layers/nodes used, the activation function, etc.

For this part of the assignment, you will only need to use the following files: mnist_train_0_1.csv for training and mnist_test_0_1.csv for testing.

Details:

- You may not use a high-level function for implementing your neural network or back-propagation. This includes whatever activation function you end up using. Functions that perform any matrix operations such as addition or multiplication, however, are perfectly fine!
- All code should be written in Python3.
- Since this is a binary classification problem, your network can have a single output node, or two output nodes. It's up to you which you choose to use.
- Your network must have at least one hidden layer. I'll leave it up to you how many hidden layers or hidden nodes you'll use. The more hidden layers and hidden nodes you use, the more accurate your network will be; however, it will take longer to train as well.
- You can also use whatever activation function you'd like for your network as long as you specify it in the final report. Two example activation functions we have talked about in class are the sigmoid function and the rectifier function. I will say that using sigmoid at the output makes more sense than using rectifier at the output.
- Don't forget the bias term! I typically ignore the bias term out of simplicity, but it's an important part of the neural network architecture.
- Explain any implementation choices you had to make in the final report.
- Include the accuracy obtained by your network on the provided dataset in your writeup.

Presentation (20 points)

Your report must be complete and clear. A few key points to remember:

- Complete: the report does not need to be long, but should include everything that was requested.
- Clear: your grammar should be correct and it should be easy to follow what you're saying.

- Concise: I sometimes print out reports to ease grading, don't make figures larger than they need to be. Graphics and text should be large enough to get the point across, but not much larger.
- Credit (partial): if you are not able to get something working, or unable to generate a particular figure, explain why in your report. If you don't explain, I can't give partial credit.

Bonus: Multiclass Classification using Multilayer Perceptrons (10 Points, required for graduate students)

In this part of the assignment, you will extend your neural network to perform multiclass classification on images of handwritten numbers ranging from 0 to 4. I have provided a training and test set that you will use: mnist_train_0_4.csv and mnist_test_0_4.csv respectively. As with the previous assignment, please report the accuracy of your network as well as details about the architecture used.

From a coding standpoint, it shouldn't require much to extend your code to handle multiclass classification; however, your model will take longer to train. Be sure to take this into account when testing your code!

Details:

- As always, don't use any high level functions to extend your code.
- Same rules apply as in Part 1. You are free to use any activation function or network architecture you want. Just be sure to mention it in the final report.
- Explain any implementation choices you had to make in the final report.
- Remember to include accuracy values along with a discussion of the architecture that you used!