Due date: Nov. 4, 2020, 11:59 PM (Arlington time). You have **two** late days — use it at as you wish. Once you run out of this quota, the penalty for late submission for each subsequent 24 hour period will be applied. You can either use your late days quota (or let the penalty be applied). Clearly indicate in your submission if you seek to use the quota.

What to turn in:

1. Your submission should include your complete code base in an archive file (`zip, tar.gz`) and `q1/, q2/,` and so on), and a very very clear README describing how to run it.
2. report (typed up with LaTeX, submit as a PDF

   le, NO handwritten scanned copies) describing what you solved and implemented and perhaps known failure cases. The report is important since we will be evaluating the grades mostly based on the report.
3. Submit your entire code report to Canvas.

Notes from instructor:

- Start early!
- You may ask the TA or instructor for suggestions, and discuss the problem with others (minimally). But **all parts of the submitted code must be your own**.
- Use Matlab or Python for your implementation.

Problem 1

(Logistic regression, **40pts**) Implement your own logistic regression. For your training data, generate 1000 training instances in two sets of random data points (500 in each) from multi-variate normal distribution with

$$\mu_1 = [1, 0],\ \mu_2 = [0, 1],\ \Sigma_1 = \begin{bmatrix} 1 & 0.75 \\ 0.75 & 1 \end{bmatrix},\ \Sigma_2 = \begin{bmatrix} 1 & -0.5 \\ 0.5 & 1 \end{bmatrix} \tag{1}$$

for each group and label them 0 and 1. Generate testing data in the same manner but sample 500 instances for each class, i.e., 1000 in total. Use sigmoid function for your activation function and cross entropy for your objective function, and perform batch training. The stopping criteria would be 1) when the gradient vanishes (i.e., norm of the gradient is too small) and 2) when the loss function converges. Set your maximum number of iterations to 10000 so that you don't wait forever for the model to converge. Plot an ROC curve and compute Area Under the Curve (AUC) in the end to evaluate your implementation, and write up a brief summary in your report. All the code base for the training, testing and plotting the ROC curve should be on your own. How many iterations does it require to traine when you change your learning rate as 0.0001, 0.001, 0.01, 0.1, and 1? Plot the number with respect to the learning rate.

Problem 2

(Multi-class classification, **30pts**) Implement multi-class logistic regression using a soft-max function and cross entropy. Use MNIST data for your training and testing data. The dataset has 60000 training and 10000 test samples that contain images of hand-written numbers in 28x28 pixels. There are various sources you can find online to fetch the data.

Please use the images within class 0 to 4 (i.e., 5 classes) only, you can do this by checking the labels given to each image. Set the maximum number of iterations to 10000. Print out accuracy, precision and recall at the end to evaluate your implementation, and write up a brief summary in your report.

Problem 3

(Artificial Neural Network, **30pts**) For this part, you may use existing libraries (Yeah!). Implement an artificial neural network (ANN) by stacking up your perceptron. You may use one hidden layer and the number of hidden nodes are up to you. Use the same MNIST data as in P2 for training and testing. Try out both sigmoid and ReLU for activation functions. Print out accuracy, precision and recall at the end. Did the performance get better or worse compared to the result from P2? It can be either so justify your answer in your report.