CSE 417T: Homework 2

Due: February 22, 2019 at 11:59PM

Notes:

- Please check the submission instructions for Gradescope provided on the course website. You must follow those instructions exactly.
- Please complete and submit the following two stub Matlab files for Problem 1:
  `https://www.cse.wustl.edu/~sanmay/teaching/cse417-spring19/logistic_reg.m`
  `https://www.cse.wustl.edu/~sanmay/teaching/cse417-spring19/find_test_error.m`
- The datasets are available here:
  `https://www.cse.wustl.edu/~sanmay/teaching/cse417-spring19/cleveland-train.csv`
  `https://www.cse.wustl.edu/~sanmay/teaching/cse417-spring19/cleveland-test.csv`
- Homework is due **by 11:59 PM on the due date.** Remember that you may not use more than 2 late days on any one homework, and you only have a budget of 5 in total.
- Please keep in mind the collaboration policy as specified in the course syllabus. If you discuss questions with others you **must** write their names on your submission, and if you use any outside resources you **must** reference them. **Do not look at each others' writeups, including code.**
- There are 6 problems on 2 pages in this homework.
- **Keep in mind that problems and exercises are distinct in LFD.**

Problems:

1. (50 points) Implement and test logistic regression using the two files provided to you as function headers. `find_test_error` should encode a function that, given as inputs a weight vector w, a data matrix X and a vector of true labels y (in the formats defined in the header), returns the **classification error** of w on the data (assuming that the classifier applies a threshold at 0 to the dot product of w and a feature vector x (augmented with a 1 in the first position in the vector to allow for a constant or bias term). `logistic_reg` should encode a gradient descent algorithm for learning a logistic regression model. It should return a weight vector w and the training set error E_{in} (not the classification error, the negative log likelihood function) as defined in class. Use a learning rate $\eta = 10^{-5}$ and automatically terminate the algorithm if the magnitude of each term in the gradient is below 10^{-3} at any step.

   - Implement the functions in the two files. Remember to submit the final version of your code for these two files on Gradescope.

- Read more about the "Cleveland" dataset we'll be using here: `https://archive.ics.uci.edu/ml/datasets/Heart+Disease`
- Learn a logistic regression model on the data in `cleveland.train` (be careful about the fact that the classes are $0/1$ – you should convert them to $-1/+1$ so that everything we've done in class is still valid). Apply the model to classify the data (using a probability of 0.5 as the threshold) in `cleveland.test`. In your writeup, report E_{in} as well as the classification error on both the training and test data when using three different bounds on the maximum number of iterations: ten thousand, one hundred thousand, and one million. What can you say about the generalization properties of the model?
- Now train and test a logistic regression model using the inbuilt matlab function `glmfit` (learn about and use the "binomial" option, and check the label format). Compare the results with the best ones you achieved and also compare the time taken to achieve the results.
- Now scale the features by subtracting the mean and dividing by the standard deviation for each of the features in advance of calling the learning algorithm (you may find the matlab function `zscore` useful). Experiment with the learning rate η (you may want to start by trying different orders of magnitude), this time using a tolerance (how close to zero you need each element of the gradient to be in order to terminate) of 10^{-6}. Report the results in terms of number of iterations until the algorithm terminates, and also the final E_{in}.

2. (10 points) LFD Problem 2.22
3. (10 points) LFD Problem 2.24
4. (10 points) LFD **Exercise** (not Problem) 3.4 parts (a) through (c) only
5. (10 points) LFD Problem 3.4
6. (10 points) LFD Problem 3.19