Homework 1, ECE 6254, Spring 2014

Due: Wednesday Jan 29, at the beginning of class

Problems:

1. Suppose that we have some number m of coins. Each coin has some probability of landing on heads, denoted p_i. Suppose that we pick up each of the m coins in turn and for each coin do n independent coin tosses. Note that the probability of obtaining exactly k heads out of n tosses for the i^{th} coin is given by the binomial distribution:

$$\mathbb{P}[k|n, p_i] = \binom{n}{k} p_i^k (1-p_i)^{n-k}.$$

For each series of coin tosses, we will record the results via the empirical estimates of the p_i given by

$$\widehat{p}_i = \frac{\text{number of times coin } i \text{ lands on heads}}{n}.$$

   (a) Assume that $n = 10$. If all the coins have $p_i = 0.05$, compute the probability that at least one coin will have $\widehat{p}_i = 0$ for the cases of $m = 1$, $m = 1,000$, and $m = 1,000,000$. Repeat for $p_i = 0.75$.

   (b) Assume that $n = 10$, $m = 2$, and $p_i = 0.5$ for both coins. Compute and then plot/sketch

$$\mathbb{P}\left[\max_i |\widehat{p}_i - p_i| > \epsilon\right]$$

as a function of $\epsilon \in [0, 1]$. On the same plot, show the bound that results from applying the Hoeffding inequality together with the union bound (see pages 11-12 of the notes from lecture 2).

2. Consider a binary classification problem involving a single (scalar) feature x and suppose that $X|Y = 0$ and $X|Y = 1$ are continuous random variables with densities given by

$$g_0(x) = \frac{1}{\sqrt{2}} e^{-\sqrt{2}|x|}$$
$$g_1(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$

respectively.

   (a) Plot g_0 and g_1.

   (b) Suppose that $\pi_0 = \mathbb{P}[Y = 0] = \frac{1}{2}$ and hence $\pi_1 = \mathbb{P}[Y = 1] = \frac{1}{2}$. Derive the optimal classification rule in terms of minimizing the probability of error. Relate this rule to the plot of g_0 and g_1.

(c) Calculate the Bayes risk for this classification problem (i.e., calculate the probability of error for the classification rule derived above). You can use the MATLAB function `erf` to compute integrals of the Gaussian density.

3. In this problem we are going to prove that the perceptron learning algorithm (PLA) will eventually converge to a linear separator for a separable data set. We will analyze the algorithm assuming for simplicity that the starting point for the algorithm is given by $\boldsymbol{w}(0) = 0$. Recall that for iterations $t \geq 1$, the algorithm proceeds by setting

$$\boldsymbol{w}(t) = \boldsymbol{w}(t-1) + y(t-1)\boldsymbol{x}(t-1),$$

where $(\boldsymbol{x}(t-1), y(t-1))$ is any input/output pair in the training data that is mislabeled by the classifier defined by $\boldsymbol{w}(t-1)$. The general approach of the proof will be to argue that $\boldsymbol{w}(t)$ and $\boldsymbol{w}^*$ get more "aligned" as t grows, and that this ultimately yields an upper bound on how many iterations the algorithm can take.

(a) Let $\delta = \min_i \ y_i \boldsymbol{w}^{*T} \boldsymbol{x}_i$. Argue that $\delta > 0$.

(b) Show that $\boldsymbol{w}^T(t)\boldsymbol{w}^* \geq \boldsymbol{w}^T(t-1)\boldsymbol{w}^* + \delta$, and conclude that $\boldsymbol{w}^T(t)\boldsymbol{w}^* \geq t\delta$. [Hint: Use induction.]

(c) Show that $\|\boldsymbol{w}(t)\|^2 \leq \|\boldsymbol{w}(t-1)\|^2 + \|\boldsymbol{x}(t-1)\|^2$. [Hint: Use the fact that $\boldsymbol{x}(t-1)$ was misclassified by $\boldsymbol{w}(t-1)$.]

(d) Show by induction that $\|\boldsymbol{w}(t)\|^2 \leq tR^2$, where $R = \max_i \ \|\boldsymbol{x}_i\|$.

(e) Show that (b) and (d) imply that

$$t \leq \frac{R^2 \|\boldsymbol{w}^*\|^2}{\delta^2}.$$

[Hint: Use the Cauchy-Schwartz inequality, i.e., $|\boldsymbol{x}_1^T \boldsymbol{x}_2| \leq \|\boldsymbol{x}_1\| \|\boldsymbol{x}_2\|$.]

4. In this problem you will implement several of the algorithms we have discussed in class and compare them on a few example datasets. There is some starter code along with the files containing the example datasets in the file `assignment1.zip` which is available for download on the course website.

For each of the algorithms we implement below, you will complete a MATLAB function which takes training data as input and gives the parameters $\boldsymbol{w}$ and b associated with the hyperplane learned by the algorithm. Be careful to make sure you get the signs correct on $\boldsymbol{w}$ and b so that the correct decision rule is given by:

$$\widehat{f}(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{w}^T\boldsymbol{x} + b \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

You will also plot the results for four synthetic datasets included in "assignment1.zip". Finally, you will then try applying each of the algorithms to a real-world dataset where you will try to predict the presence or absence of coronary heart disease based on a number of risk factors.

(a) We will begin by implementing LDA. Using the notes from lecture 3, complete the code in the file `LDAFit.m`.

(b) Next we will implement a solver for logistic regression using the Newton-Raphson algorithm for performing the maximum likelihood estimation step. This algorithm requires computing both the gradient and the Hessian at each iteration. We computed the gradient in class (lecture 4). The Hessian is given by

$$\frac{\partial^2 \ell(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = -\sum_{i=1}^{n} \widetilde{\boldsymbol{x}}_i \widetilde{\boldsymbol{x}}^T g(\boldsymbol{\theta}^T \widetilde{\boldsymbol{x}}_i)(1 - g(\boldsymbol{\theta}^T \widetilde{\boldsymbol{x}}_i)),$$

where g is defined as in the notes. Using this fact, complete the code in the file `LRFit.m`. In order to establish working code, you will need to define some kind of stopping criterion. A natural choice would be to stop when $\|\boldsymbol{\theta}^{\text{new}} - \boldsymbol{\theta}^{\text{old}}\|/\|\boldsymbol{\theta}^{\text{old}}\|$ becomes sufficiently small, but you should feel free to experiment. The Newton-Raphson algorithm should converge very quickly, so if it is taking you more than 5-10 or so iterations, you should check your code to make sure you haven't made a mistake somewhere.

(c) Finally, we will also implement the PLA. Using whatever method you like for selecting $(\boldsymbol{x}(t), y(t))$ at each iteration, implement the PLA as described in the notes and in problem 3 using the template in `PLAFit.m`. You will again need to decide on some kind of stopping rule since the data might not be linearly separable, in which case the algorithm could potentially run forever.

(d) Next try running each of these algorithms on the synthetic datasets included in the zip file, plotting the results using the file `syntheticTests.m`. Print out or save a copy of your results and include this with your writeup. Comment on the performance of each algorithm on the various datasets. Which algorithm seems to do best?

(e) Finally, try training each of these algorithms using the training data in `heartTrain.mat`. For each classifier that you learn, then try applying it to the data in the file `heartTest.mat`. Do not use the "testing" data during training, keep it separate and then use it to estimate the probability of error for each classifier that you have trained. Report the testing error for each algorithm.

Along with your written solutions, you should also submit a .zip file containing all of your completed code. The file should be named `LastnameFirstnameAssignment1.zip` and should contain all of the core files in the main directory. Email this file to `mdav@gatech.edu` with the subject line: "ECE 6254: Assignment 1".