CIS 520, Machine Learning, Fall 2014: Assignment 4B
Due: Thursday, October 16th, 11:59pm (via turnin)

Instructions. This is a MATLAB programming assignment. This assignment consists of multiple parts. Portions of each part will be graded automatically, and you can submit your code to be automatically checked for correctness to receive feedback ahead of time.

We are providing you with codebase / templates / dataset that you will require for this assignment. Download the file `hw4_kit.zip` from Canvas **before** beginning the assignment. **Please read through the documentation provided in ALL Matlab files before starting the assignment.** The instructions for submitting your homeworks and receiving automatic feedback are online on the wiki:

http://alliance.seas.upenn.edu/~cis520/wiki/index.php?n=Resources.HomeworkSubmission

Collaboration. For this programming assignment, you are allowed to work in pairs, but no more than 2. Each group is required to submit **ONCE** only. **The LOWEST grade will be recorded for multiple submissions.** Be sure to include your pennkeys and names in the *Group.txt* file. Failure to do so will result in a failure in the grading process. **We will be using automatic checking software to detect blatant copying of other student's assignments, so, please, don't do it.**

Learning Curves of Naive Bayes and Logistic Regression [60 points]

Description. A common debate in machine learning has been over generative versus discriminative models for classification. In this question we will explore this issue by considering Naive Bayes and logistic regression classification algorithms.

In this assignment you will briefly compare the two approaches on the Breast Cancer dataset included in `hw4_kit.zip`. In this problem you will obtain obtain the learning curves comparing the error rate of logistic regression vs. naive bayes as they are allowed to train on more data. Note that for this assignment we will use $Y \in \{0, 1\}$ instead of $Y \in \{-1, +1\}$ as in the last assignment.

Your task. You have five smaller programming tasks for this assignment, besides the programming required to generate the report. You simply need to fill in relatively few lines of missing code in five template files we have given you in order to estimate parameters for Gaussian Naive Bayes and run gradient ascent for logistic regression.

(i) **[6 points]** `nb_train.m` - In this file, complete the lines estimating the Naive Bayes parameters. Recall that according to the shared variances form of Gaussian Naive Bayes, the likelihood is

$$P(\mathbf{x}, y) = P(Y = y) \prod_i P(X_i = x_i \mid \mu_{yi}, \sigma_i^2),$$

where $P(X_i \mid \mu_{yi}, \sigma_i^2)$ is a Gaussian distribution with mean μ_{yi} and variance σ_i^2. Therefore, you need to estimate parameters:

$$P(Y = 1), \quad \mu_{yi}, \quad \sigma_i^2, \qquad \forall y, i$$

where i ranges over features and $y \in \{0, 1\}$ are the possible values of Y on this dataset.

(ii) **[6 points]** `nb_test.m` - In this file, complete the lines to compute *log generative probability*:

$$\log P(\mathbf{x}, y) = \log P(Y = y) + \sum_{i=1}^{m} \log P(x_i \mid Y = y),$$

where x_i is the value of the i'th feature and $y \in \{0, 1\}$. These probabilities are necessary to make predictions using Naive Bayes.

(iii) **[6 points]** `lr_gradient.m` - In this file, complete the lines to compute the gradient of the regularized logistic regression objective L for a given $\mathbf{X}, \mathbf{Y}$, and $\mathbf{w}$:

$$L(\mathbf{w}) = \log P(\mathbf{Y} \mid \mathbf{X}, \mathbf{w}) - \frac{C}{2}||\mathbf{w}||_2^2$$

Note: this is a slightly different form than the one given in class. It is slightly easier to work with this in practice. (In lecture, we used $C = \frac{1}{\lambda^2}$.)

(iv) **[14 points]** `lr_train.m` - In this file, implement gradient ascent as described in class and according to the specifications given in the m-file. This will involve repeatedly applying the update:

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \nabla_{\mathbf{w}} L(\mathbf{w})$$

where η is a stepsize that is given as an input. *In addition*, your training routine must also compute the conditional log likelihood objective,

$$\log P(\mathbf{Y} \mid \mathbf{X}, \mathbf{w}) = -\sum_i \log(1 + \exp\{-y_i \mathbf{x}_i \cdot \mathbf{w}\}),$$

for each iteration of gradient ascent. This provides a trace of the progress of the optimization. Gradient ascent should halt whenever a maximum number of rounds T is reached, or if the average norm of the gradient $||\nabla_{\mathbf{w}} L(\mathbf{w})||_2/n$ (where n is the number of datapoints) is smaller than a threshold given as an argument to your function.

(v) **[6 points]** `lr_test.m` - In this file, complete the lines to compute the posterior probability:

$$P(Y = 1 \mid \mathbf{x}, \mathbf{w}) = \frac{1}{1 + \exp\{-\mathbf{w}^\top \mathbf{x}\}},$$

which is necessary to make predictions using logistic regression.

Report - Step sizes. For this written/analysis portion of the assignment, refer to the script `generate_lr_plots.m` and make sure to follow the instructions there. Your first task is to analyze how the step size η influences the gradient ascent process.

1. **[8 points]** Using the *entire dataset*, train logistic regression using $C = 10^{-3}$ and a range of step sizes (specified in the m-file) and 5000 iterations (set the convergence test threshold to zero so it never triggers). On a single graph, plot the objectives of each run using a logarithmic scale for the Y-axis, with the iteration number as the X axis.

   How does the progress of the optimization different from the largest step size to the smallest?

2. **[4 points]** Compute the training error of each of the returned weight vectors from each run of gradient ascent, as well as the norm of the gradient at the returned solution. Which learning rate has the smallest gradient? Which is the best in terms of training error? In general, does optimizing the objective correlate with optimizing training error?

Report - Learning curves. For this written/analysis portion of the assignment, refer to the script `generate_lr_plots.m` and make sure to follow the instructions there. Your next task is to analyze the rate at which logistic regression and Naive Bayes learn.

1. **[10 points]** Generate *learning curves* for Naive Bayes and Logistic Regression. A learning curve shows how test error changes as more training data becomes available; given a fixed training and test set, we simulate more or less data by subsampling the training set and assessing performance on the test set.

   To generate learning curves, you will need to do the following:

   (a) Randomly separate the dataset into 80% training, 20% test.

   (b) Further subdivide the training set into 8 partitions (10% of the data in each).

   (c) Train NB and LR using partition 1 as training and record the test error.

   (d) Train NB and LR using partitions 1 and 2 as training and record the test error.

   (e) Train NB and LR using partitions 1, 2, and 3 as training and record th etest error.

   (f) Repeat for partitions (1,2,3,4), (1,2,3,4,5), and so forth.

   (g) Train NB and LR using all the train data and record the test error.

   Repeat this process 100 times and average the test errors across random separations of the data into training and test. For LR, use parameters $C = 10^{-3}$, $\eta = 10^{-3}$, a stopping tolerance of 10^{-5}, and at most 1000 iterations of training (these parameters lead to faster convergence, but not necessarily the best accuracy for LR).

   Finally, on a single plot, plot the test error as a function of the number of datapoints given to each algorithm. Does LR or NB converge to peak test set performance faster? Which method works better with the most data?