Statistical Machine Learning (W4400)
Spring 2016
https://courseworks.columbia.edu

John P. Cunningham
jpc2181
Ben Reddy, Phyllis Wan, Ashutosh Nanda
bmr2136, pw2348, an2655

Homework 3

Due: Tuesday 08 March 2016

Students are encouraged to work together, but homework write-ups must be done individually and must be entirely the author's own work. Homework is due at the **beginning** of the class for which it is due. **Late homework will not be accepted under any circumstances.** To receive full credit, students must thoroughly explain how they arrived at their solutions and include the following information on their homeworks: name, UNI, homework number (e.g., HW03), and class (STAT W4400). All homework must be turned in online through Courseworks in PDF format, have a .pdf extension (not zip or other archive!), and be less than 4MB. If programming is part of the assignment, the code must be turned in in one or more .R files. Homeworks not adhering to these requirements will receive no credit. For your convenience (not required), a tex template for producing nice PDF files can be found on courseworks.

1. **Boosting** (70 points)

   The objective of this problem is to implement the AdaBoost algorithm. We will test the algorithm on handwritten digits from the USPS data set.

   AdaBoost: Assume we are given a training sample $(\mathbf{x}^{(i)}, y_i), i = 1, ..., n$, where $\mathbf{x}^{(i)}$ are data values in $\mathbb{R}^d$ and $y_i \in \{-1, +1\}$ are class labels. Along with the training data, we provide the algorithm with a training routine for some classifier c (the "weak learner"). Here is the AdaBoost algorithm for the two-class problem:

   1. Initialize weights: $w_i = \frac{1}{n}$
   2. for $b = 1, ..., B$
      (a) Train a weak learner c_b on the weighted training data.
      (b) Compute error: $\epsilon_b := \frac{\sum_{i=1}^n w_i \mathbb{I}\{y_i \neq c_b(\mathbf{x}^{(i)})\}}{\sum_{i=1}^n w_i}$
      (c) Compute voting weights: $\alpha_b = \log\left(\frac{1-\epsilon_b}{\epsilon_b}\right)$
      (d) Recompute weights: $w_i = w_i \exp\left(\alpha_b \mathbb{I}\{y_i \neq c_b(\mathbf{x}^{(i)})\}\right)$
   3. Return classifier $\hat{c}_B(\mathbf{x}^{(i)}) = \operatorname{sgn}\left(\sum_{b=1}^B \alpha_b c_b(\mathbf{x}^{(i)})\right)$

   Decision stumps: Recall that a stump classifier c is defined by

   $$c(\mathbf{x}|j, \theta, m) := \begin{cases} +m & x_j > \theta \\ -m & \text{otherwise.} \end{cases} \tag{1}$$

   Since the stump ignores all entries of $\mathbf{x}$ except x_j, it is equivalent to a linear classifier defined by an affine hyperplane. The plane is orthogonal to the jth axis, with which it intersects at $x_j = \theta$. The orientation of the hyperplane is determined by $m \in \{-1, +1\}$. We will employ stumps as weak learners in our boosting algorithm. To train stumps on weighted data, use the learning rule

   $$(j^*, \theta^*) := \arg\min_{j,\theta} \frac{\sum_{i=1}^n w_i \mathbb{I}\{y_i \neq c(\mathbf{x}^{(i)}|j, \theta, m)\}}{\sum_{i=1}^n w_i} \ . \tag{2}$$

In the implementation of your training routine, first determine an optimal parameter θ_j^* for each dimension $j = 1, ..., d$, and then select the j^* for which the cost term in (2) is minimal.

Homework problems:

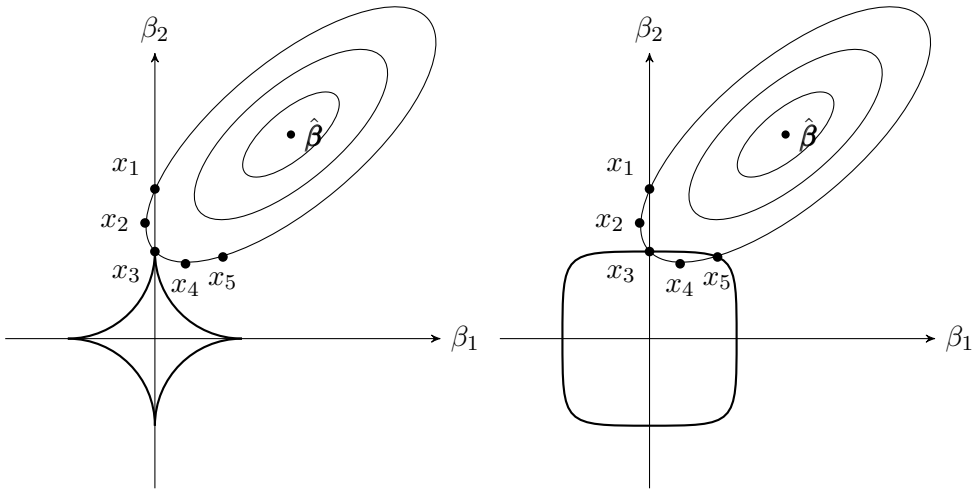
1. (30 points) Implement the AdaBoost algorithm in R. The algorithm requires two auxiliary functions, to train and evaluate the weak learner. We also need a third function which implements the resulting boosting classifier. We will use decision stumps as weak learners, but a good implementation of the boosting algorithm should permit you to easily plug in arbitrary weak learners. To make sure that is possible, please use function calls of the following form:
   - `pars <- train(X, w, y)` for the weak learner training routine, where `X` is a matrix the columns of which are the training vectors $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$, and `w` and `y` are vectors containing the weights and class labels. The output `pars` is a list which contains the parameters specifying the resulting classifier. (In our case, `pars` will be the triplet (j, θ, m) which specifies the decision stump).
   - `label <- classify(X, pars)` for the classification routine, which evaluates the weak learner on `X` using the parametrization `pars`.
   - A function `c_hat <- agg_class(X, alpha, allPars)` which evaluates the boosting classifier ("aggregated classifier") on `X`. The argument `alpha` denotes the vector of voting weights and `allPars` contains the parameters of all weak learners.
2. (15 points) Implement the functions `train` and `classify` for decision stumps.
3. (20 points) Run your algorithm on the USPS data (the digit data we used in Homework 2) and evaluate your results using cross validation.

   More precisely: Your AdaBoost algorithm returns a classifier that is a combination of B weak learners. Since it is an incremental algorithm, we can evaluate the AdaBoost at every iteration b by considering the sum up to the b-th weak learner. At each iteration, perform 5-fold cross validation to estimate the training and test error of the current classifier (that is, the errors measured on the cross validation training and test sets, respectively).
4. (5 points) Plot the training error and the test error as a function of b.

Submission. Please make sure your solution contains the following:

- Your implementation for `train`, `classify` and `agg_class`.
- Your implementation of `AdaBoost`.
- Plots of your results (training error and cross-validated test error).

2. ℓ_q **regression** (30 points)



The figures show the cost function components of the ℓ_q-regression problems with $q = 0.5$ (left) and $q = 4$ (right).

1. (15 points) Does one/none/both of the cost functions encourage sparse estimates? If so, which one? Explain your answer.
2. (15 points) Which of the points $x_1, \ldots, x_5$ would achieve the smallest cost under the ℓ_q-constrained least squares cost function? For each of the two cases, name the respective point and give a brief explanation for your answer.