STAT 538 Homework 3
Out January 30, 2012
Due February 9, 2012
©Marina Meilă
mmp@stat.washington.edu

Problem 1 – Online linear regression by Stochastic gradient

Consider the linear regression problem with Least Square loss

$$\min_{\beta} E[(y - \beta^T x)^2] \;=\; \min_{\beta} \; L_{LS} \tag{1}$$

where $y \in \mathbb{R}$, $x \in \mathbb{R}^n$, $\beta \in \mathbb{R}^n$. For simplicity we consider the infinite sample version of the problem, but if you want a variation (ungraded) try also the finite sample version, where we optimize $\hat{L}_{LS}$ instead.

The function in (1) is a quadratic function that has a closed form solution, but we will pretend that we don't know this and investigate the use of (stochastic) gradient descent for this problem.

a. Find the expression of the gradient and Hessian of this problem, i.e $\nabla L_{LS}(\beta)$, $\nabla^2 L_{LS}(\beta)$. Express the Hessian as a function of some well known statistical descriptor(s) of the data distribution.

b. Assume that the covariates x are sampled from a Normal distribution with mean 0 and covariance Σ. Describe and motivate a reasonable way to find the λ parameter of the Stochastic Gradient algorithm based on this assumption.

c. Write the expression of $d = \frac{\partial L_{LS}(y,\beta^T x)}{\partial \beta}$. Show that the direction of descent d is collinear with x. What does the scaling of x represent from a statistical modeling point of view?

e. Write the Stochastic Gradient Descent algorithm to optimize this problem. Assume that λ is known.

For practice, ungraded Consider now Neural Network regression, with LS loss, and solve the same problem. Consider also Neural Network with logistic output and L_{logit} loss, and solve the same problem. Consider any of these with a regularization term $\frac{C}{2}||\beta||^2$.

Problem 2 – Boosting algorithms on stumps

Read the whole problem carefully before you start working. You need to hand out the required plots (described at the end) and your comments, including the answers to the questions scattered through the text.

Code: You are required to write your own code for all questions, with the following exception: if you want to replace the stump classifier with another classifier, whose code you have available, you can discuss this with me BEFOREHAND. Exception:note that you can modify the code for the linear classifier you wrote in Homework 1 to do question **a.***; this is allowed and even recommended ex officio.*

This homework will make use of the following (two-dimensional) data sets: `hw3-linear-train.dat, hw3-circle-train.dat, hw3-linear-test.dat`,`hw3-circle-test.dat` having each 200 examples. The first two are training sets, which you will use to construct your classifiers. The last two are test sets, on which you will evaluate the peformance of the classifiers you obtained.

1 - Fitting a "stump" A stump is a linear classifier that takes a single coordinate into account, i.e. $f(x) = \pm\text{sign}(x_i - b)$. Its decision surface is a hyperplane perpendicular to one of the coordinate axes. The name stump recalls that this classifier is a decision tree with just 1 "branch".

Write code that implements stumps:

Input d-dimensional data $x^1, \dots x^N$
It is assumed that $\{x_j^1, \dots x^N\} \subseteq [A_j, B_j]$ with $R_j = B_j - A_j$. In other words, the range of coordinate j has value R_j, given. (For both data sets in this assignment, all the ranges R_j are 1.)
the index of a coordinate $j \in \{1, 2, \dots d\}$
weights w_i, $i = 1, \dots N$ for the datapoints
(the weights are non-negative and normalized)

1. Choose a coordinate $j = 1 : d$
2. Sort the data in increasing order by coordinate x_j^i; denote by $x_j^{(i)}$ the order statistics; reorder the labels $\{y_i\}$ and the weights $\{w_i\}$ accordingly
3. for $i = 0 : N$
   Calculate $\epsilon_i = -\sum_{i' \le i} y^{(i')} w^{(i')} + \sum_{i' > i} y^{(i')} w^{(i')}$. This represents the fraction of correctly classified points - fraction of errors for the stump $f(x) = x_j - a_i$ with $a_i \in (x_j^{(i)}, x_j^{(i+1)})$ along direction j.
4. Let $E_+ = \max_i \epsilon_i$, $l_+ = \underset{i}{\text{argmax}}\, \epsilon_i$, $E_- = \max_i(-\epsilon_i)$, $l_- = \underset{i}{\text{argmax}}\,(-\epsilon_i)$.

The "max" here must be computed ONLY over the i's that are "local" peaks, i.e. for which $\epsilon_{i-1} < \epsilon_i > \epsilon_{i+1}$ or $-\epsilon_{i-1} < -\epsilon_i > -\epsilon_{i+1}$. **Extra credit:** If $x_{i,j} \neq x_{i',j}$ for all i, i' and there is at least one label of each class, then such a "maximum" exists. True or false?
Note the $(1-\epsilon_i)/2$ values represents the classification error of a classifier that labels $x_{i:n}$ with $+1$, and $x_{1:i-1}$ with -1. And $1+\epsilon_i)/2$ is the error rate of a classifier that labels the point to it's left with $+1$, and the ones to its right with -1.

5. Set

$$f(x) = \begin{cases} (x_j - \frac{x_j^{(l_+)} + x_j^{(l_+ +1)}}{2})/R_j & \text{if } E_+ > E_- \\ -(x_j - \frac{x_j^{(l_-)} + x_j^{(l_- +1)}}{2})/R_j & \text{if } E_+ \leq E_- \end{cases}$$

In other words, the stump classifier is a hyperplane perpendicular on coordinate axis j passing halfway between two data points. The normalization by R_j ensures that the values of f are bounded in $[-1, 1]$ for all input data.

Let $\mathcal{B}_j$ be the family of stumps for a fixed j and let $\tilde{\mathcal{B}}_j = \{\, \mathrm{sign} f,\ b \in \mathcal{B}_j\}$ be the corresponding family of integer-valued stumps. (In the homework, because we use `DiscreteAdaBoost`, we use only integer-valued stumps $\tilde{\mathcal{B}}_j$.)

Note that for some data sets, the same value of $E_\pm$ can be attained for several different stumps along the same direction. In that case, choose one of them arbitrarily.

To test that your stump classifier works well, two one-dimensional data sets are provided: `hw3-1D-clean.dat, hw3-1D-noisy-labels.dat`. The first is separable by a stump; in the second, 10% of the labels have changed sign so the data cannot be separated with 0 error. These data are only provided to assist you debug; you are not required to show any plots about them

b. Run the DISCRETE ADABOOST algorithm on the `hw3-linear-train.dat, hw3-circle-train.dat` for M iterations, using the families $\tilde{\mathcal{B}}_1, \tilde{\mathcal{B}}_2$ as base classifiers. For simplicity, take $j = 1$ at iterations $k = 1, 3, 5, \ldots$ and $j = 2$ on the even iterations. Choose M somewhere between 50 and 200; in fact, you are encouraged to try various values of M before chosing one to plot.

c. Plots and comments (Suggestion: make the plots as you progress through the homework, so that you can use them for debugging)

- Plot the two training sets in 2D.
- **If you implement stumps in part a.** Plot ϵ_i vs i for $k = 1$ or 2, for each of the two data sets (2 plots, one for each data set).

- For each of the classification problems, let $f_k = \sum_{m=1}^{k} \beta_m f_m$ (the current boosted classifier). Let $E_k = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}_{f_k(x^i) \neq y^i}$ the training error of f_k. Define E_k^{test} similarly, as the average number of errors of f_k on the test set. Plot the E_k and E_k^{test} vs k on the same plot. Do you see any evidence of overfitting?
- Plot the decision regions or decision boundary on the two data sets (overimposed on the training data if possible)

d. What would happen if RealAdaBoost with stumps (as described in the lecture notes) was run on the `hw3-circle-train.dat`? More precisely, would the training error of E_k decrease to 0? Explain why or why not.[OK to find the answer by experiment, i.e. by actually running RealAdaBoost on the the circle data, but not required.]

e. For extra credit: Prove or disprove that the stumps are "weak learners", i.e that for every n there is a $\delta < 1/2$ so that for every data set of size n the classificaition error of the best stump is $\leq \delta$.