COMS 4721: Machine Learning for Data Science

Columbia University, Spring 2015

Homework 3: Due March 31, 2015

Submit the written portion of your homework as a single PDF file through Courseworks (less than 5MB). In addition to your PDF write-up, submit all code written by you in their original extensions through Courseworks. Do not submit in .rar, .zip, .tar, .doc, or other file types. Your grade will be based on the contents of one PDF file and original source code. Everything resulting from the problems on this homework other than the raw code should be put in the PDF file.

Show all work for full credit. Late homeworks will not be accepted – i.e., homework submitted to Courseworks after midnight on the due date.

Problem 1 (boosting) – 100 points total

This homework will focus on boosting. You will boost two classifiers, the Bayes classifier with *shared* covariance and the logistic regression classifier learned "online" similar to the Perceptron. The version of AdaBoost you will implement involves sampling a bootstrap data set using the distribution on data at the current iteration. This will allow the classifier to account for the evolving distribution on the training data set. We point out that these samples are only used to learn the classifier f_t. All other computations, such as for ϵ_t, are calculated using the weights on each point in the training set. The general form of boosting you should implement is given below.

Algorithm: AdaBoost (with sampling)

Given $(x_1, y_1), \ldots, (x_n, y_n)$, $x \in \mathcal{X}$, $y \in \{-1, +1\}$, set $p_1(i) = \frac{1}{n}$

- For $t = 1, \ldots, T$

  1. Sample a bootstrap set $\mathcal{B}_t$ of size n from $D_t = \sum_{i=1}^n p_t(i)\delta_{x_i}$ $\leftarrow$ distribution on training set
  2. Learn a classifier f_t using data in $\mathcal{B}_t$.
  3. Set $\epsilon_t = \sum_{i=1}^n p_t(i)\mathbb{1}\{y_i \neq f_t(x_i)\}$ and $\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$.
  4. Update $\tilde{p}_{t+1}(i) = p_t(i)\exp\{-\alpha_t y_i\, f_t(x_i)\}$.
  5. Normalize $p_{t+1}(i) = \tilde{p}_{t+1}(i)/\sum_j \tilde{p}_{t+1}(j)$.

- Set the classification rule to be

$$f_{boost}(x_0) = \text{sign}\left(\sum_{t=1}^T \alpha_t\, f_t(x_0)\right).$$

Data

The data for the experiments is the breast cancer data set from the University of Wisconsin Hospitals located on the UCI Machine Learning Repository.[1] The data has been preprocessed, so you must use the version posted on Courseworks and the class website. The data consists of 10-dimensional observations (including a dimension fixed to 1), and a label, $+1$ indicating cancer and -1 indicating no cancer. There are 683 observations in total.

For experiments in Parts 2 and 3 below, set aside the first 183 observations as a testing set and use the remaining 500 observations for training the boosted classifier.

- Part 1 (10 points)

  Write a function that samples discrete random variables. You will use this function to implement Step 1 of the boosting algorithm given above. The function should take in a positive integer n and a discrete, k-dimensional probability distribution w, and return a $1 \times n$ vector c, where $c_i \in \{1, \ldots, k\}$, $\text{Prob}(c_i = j|w) = w(j)$ and the entries of c are independent. For a distribution $w = [0.1,\ 0.2,\ 0.3,\ 0.4]$, show the histogram of a sample vector c when $n = 100, 200, 300, 400, 500$.

  Hint: The cumulative distribution function (CDF) of w, and n uniform random variables may be useful.

- Part 2 (45 points)

  In this part you will boost the Bayes classifier with a shared covariance. Recall that we can write this as a linear classifier, where the label prediction for x is $y = f(x) = \text{sign}(w_0 + x^T w)$. For the Bayes classifier, $f(x)$ is equal to

$$\ln \frac{p(x|y=1)p(y=1)}{p(x|y=0)p(y=0)} = \underbrace{\ln \frac{\pi_1}{\pi_0} - \frac{1}{2}(\mu_1 + \mu_0)^T \Sigma^{-1}(\mu_1 - \mu_0)}_{= w_0} + x^T \underbrace{\Sigma^{-1}(\mu_1 - \mu_0)}_{= w}.$$

  To make the notation easier to read we've written the -1 class as a 0 class, but the data is labeled ± 1.

  For $t = 1, \ldots, 1000$ iterations of boosting, do the following:

  1. Implement a boosted version of this Bayes classifier, where class-specific π and μ, and shared Σ are learned on the bootstrap set $\mathcal{B}_t$. Notice that you only need to store w_0 and w for this problem, as indicated in the equation above. Since the data already contains a bias dimension, you can store a single "augmented" vector where w_0 and w are combined.
  2. On a single plot, show the training and testing error as a function of iteration t.
  3. Indicate the testing accuracy by learning the Bayes classifier on the training set without boosting.
  4. Plot α_t and ϵ_t as a function of t.
  5. Pick 3 data points and plot their corresponding $p_t(i)$ as a function of t. Select the points such that there is some variation in these values.

[1] https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Original)

- Part 3 (45 points)

  In this part you will perform essentially the same experiments as in Part 2, but with a different classifier. We will focus on an online version of the logistic regression classifier:

Algorithm: Logistic regression (online)

Input: A bootstrapped set $\mathcal{B}_t$ and step size $\eta \in (0, 1]$ (e.g., $\eta = 0.1$)

1. **Set** $w^{(1)} = \vec{0}$
2. Randomly order the data set.
3. **For step** $i = 1, \ldots, n$ **do**
   - Update $w^{(i+1)} = w^{(i)} + \eta\{1 - \sigma_i(y_i \cdot w)\}y_i x_i$, where $\sigma_i(y_i \cdot w) = 1/(1 + e^{-y_i x_i^T w})$

For $t = 1, \ldots, 1000$ iterations of boosting, do the following:

1. Implement the online logistic classifier described above.
2. On a single plot, show the training and testing error as a function of iteration t.
3. Indicate the testing accuracy by learning logistic regression model on the training set *without* boosting. You can use the two-class version of your softmax logistic regression code from Homework 2 to do this, or your own implementation of binary logistic regression.
4. Plot α_t and ϵ_t as a function of t.
5. Pick 3 data points and plot their corresponding $p_t(i)$ as a function of t. Select the points such that there is some variation in these values.