Multi-class boosting

COS424: Assignment # 2

Due : Friday, March 14, 2008

In class, we have discussed *binary* classification, which is classification into one of two classes. In this programming assignment, we will consider *multi-class* classification, where a data point can be in one of L classes. Specifically, we will compare multi-class versions of boosting, support vector machines, and naive Bayes for text classification.

In the following sections, our data are $\{x_n, y_n\}_{n=1}^N$, where y_n takes on one of L labels. We will look at two text data sets. The first is a collection of articles from usenet newsgroups. The second are wire articles from Reuters. In addition to studying classification algorithms, this assignment will teach you how to set up and organize careful data analysis.

You will implement multi-class boosting for text classification. The AdaBoost.MH algorithm maintains a distribution over the data and labels at each round, denoted $D_t(n, \ell)$, and uses a weak hypothesis that maps data and labels to the reals, $X \times Y \rightarrow R$. We summarize the algorithm here; to read more see [1].

- Initialize $D_1(n, \ell) = \frac{1}{NL}$.
- For each round $t = \{1, \ldots, T\}$:
  - Pass distribution D_t to a weak learner.
  - Obtain the weak hypothesis h_t.
  - Choose $\alpha_t \in R$.
  - Update
$$D_{t+1}(n, \ell) = \frac{1}{Z_t} D_t(n, \ell) \exp\{-\alpha_t y_n[\ell] h_t(x_n, \ell))\} \tag{1}$$
    where Z_t is the normalizer that ensures that this distribution sums to one, and $y_n[\ell]$ is $+1$ when $y_n = \ell$ and -1 when $y_n \neq \ell$.
- Output the final hypothesis:
$$f(x, \ell) = \sum_{t=1}^{T} \alpha_i h_t(x, \ell). \tag{2}$$
  We use f to predict the class of a new document x with
$$\hat{\ell} = \arg\max_{\ell} f(x, \ell). \tag{3}$$

In the variant that you will implement, the weak hypotheses have the form

$$h^v(x,\ell) = \begin{array}{ll} c_{\ell,v} & \text{if } v \in x \\ c_{\ell,\bar{v}} & \text{if } v \notin x \end{array} \tag{4}$$

where

$$c_{\ell,\bar{v}} = \frac{1}{2}\log\frac{W_+^{\ell,\bar{v}}}{W_-^{\ell,\bar{v}}} \tag{5}$$

$$c_{\ell,v} = \frac{1}{2}\log\frac{W_+^{\ell,v}}{W_-^{\ell,v}}. \tag{6}$$

These expressions comprise the following four terms,

$$W_+^{\ell,v} = \sum_{n=1}^{N} D_t(n,\ell)\,1[v \in x_n]\,1[y_n = \ell] \tag{7}$$

$$W_+^{\ell,\bar{v}} = \sum_{n=1}^{N} D_t(n,\ell)\,1[v \notin x_n]\,1[y_n = \ell] \tag{8}$$

$$W_-^{\ell,v} = \sum_{n=1}^{N} D_t(n,\ell)\,1[v \in x_n]\,1[y_n \neq \ell] \tag{9}$$

$$W_-^{\ell,\bar{v}} = \sum_{n=1}^{N} D_t(n,\ell)\,1[v \notin x_n]\,1[y_n \neq \ell]. \tag{10}$$

Given that we choose a particular hypothesis h^v, the normalizing constant for the resulting next round's distribution over data is

$$Z_t(v) = 2\sum_{\ell=1}^{L}\left(\sqrt{W_+^{\ell,v}W_-^{\ell,v}} + \sqrt{W_+^{\ell,\bar{v}}W_-^{\ell,\bar{v}}}\right) \tag{11}$$

To choose a weak hypothesis at each round, we choose the v such that $Z_t(v)$ is *minimized*.

Written problems

1. Normalizing constant. Using the definition of $D_{t+1}(n,\ell)$ in Equation 1, show that Equation 11 holds.

2. Intuitions. Describe the intuition behind how this algorithm works. What are the intuitions behind the weak hypothesis in Equation 4? How would you describe the c quantities and the W quantities computed in Equations 5 through 10?

3. Smoothed variant. In the smoothed variant, the weak hypothesis is changed to

$$c^{\text{sm}}_{\ell,\bar{v}} = \frac{1}{2}\log\frac{W^{\ell,\bar{v}}_{+}+\epsilon}{W^{\ell,\bar{v}}_{-}+\epsilon} \tag{12}$$

$$c^{\text{sm}}_{\ell,v} = \frac{1}{2}\log\frac{W^{\ell,v}_{+}+\epsilon}{W^{\ell,v}_{-}+\epsilon}. \tag{13}$$

What is the normalizing constant for the Boosting.MH algorithm with this learner?

R programming

In the next several problems, you will implement, evaluate, and explore Boosting.MH. Two data sets have been provided on-line: the reuters data are news articles that are classified into 10 categories; the newsgroups data are Usenet posts classified into ten newsgroups. Each data set contains: (a) an Rdat file that has the term occurrence matrix, class vector, and fold assignments (see below); (b) the original documents, one per line, in a file; and (c) a file of labels, which name each of the classes.

4. Implementation. Implement the smoothed variant of Boosting.MH as described above. This should be a function that takes a term count matrix, fold assignment vector, fold number, vocabulary, and a number of rounds as arguments. It should return two objects: first, a matrix or dataframe that includes the selected vocabulary word, within-fold accuracy, and out-of-fold accuracy at each round of boosting; second, a function which takes a document row as input and returns a predicted class.

5. Weak hypotheses. For each fold, which vocabulary words were selected at each round of boosting? Try at least 20 rounds. Does this vary across folds?

6. Accuracy. *Cross validation* is a way of estimating generalization error. We first divide the data into K folds. For each fold, we train a classifier on the data that are *not* in that fold, and then test on the data that are in that fold. Note that with cross-validation, we compute out-of-fold predictions for each data point once.

For each fold, the *out-of-fold accuracy* is defined as

$$\frac{\sum_{\{n:\text{fold}(n)=i\}} 1[f_i(x_n) = y_n]}{|\{n : \text{fold}(n) = i|}, \tag{14}$$

where $f_i(x_n)$ is the classifier trained on the data that are not assigned to fold i, and fold(n) maps the nth data point to its assigned fold. This is the proportion of correctly classified documents in the out-of-fold data.

Note that this quantity can apply to any classification algorithm. One trains the classifier on the out-of-fold data, and tests the classifier on the within-fold data.

Plot the mean 5-fold cross validated within-fold accuracy and out-of-fold accuracy as a function of numbers of rounds of boosting. (Compute this for at least up to 500 rounds.) Make a plot for each data set.

We have provided `nbayes.train`, which fits a naive Bayes classifier to the same inputs (with Laplace smoothing). Plot the accuracy of naive Bayes as a straight line across the plot. Be sure to include a legend. (See the function `legend`.)

[**Extra credit**: explore different smoothing parameters in naive Bayes.]

7. Error analysis. Choose one of the two analyses. (Or, for extra credit, choose both.)

(a) **Confusion matrix.** The *confusion matrix* is a table of values, where C_{ij} is the percentage of time a document of class i was classified as class j. Note, the per-class accuracy is along the diagonal of this matrix. What is the 5-fold cross-validated confusion matrix for the reuters and newsgroups corpora after 500 rounds of boosting? Does it make intuitive sense? Why?

(b) **Going back to the data.** After 500 rounds of boosting, examine 5 within-fold errors in each data set (for any of the folds). What happened?

References

[1] Schapire, R. and Singer, Y. (2000). BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39:135–168.