CS534 Implementation Assignment 3: Bagging and AdaBoost

Amit Bawaskar, Michael Lam
EECS, Oregon State University

May 27, 2013

Abstract

In this assignment we implemented and evaluated bagging and AdaBoost using the decision stump as the base learner.

1 Introduction

Ensemble methods such as bagging and AdaBoost work by taking a base learner and generating a set (ensemble) of hypotheses by varying the training set. The final hypothesis for classification is a majority vote of these hypotheses. The decision stump is a suitable base learner for bagging and especially AdaBoost because it is a weak learner, meaning it classifies slightly better than random.

In this assignment we implemented the decision stump, bagging and AdaBoost. For each ensemble method we evaluated the training and test errors as a function of the ensemble size.

2 Decision Stump

The decision stump is a one level decision tree. In this assignment all features and labels are binary. Therefore the decision stump is a one level binary tree with a binary feature test at the root node connected to two leaf nodes each containing a label.

Learning is done by going through every feature in the training data and computing the maximum information gain (minimum entropy). Once the feature with the maximum information gain is determined, the two leaf nodes are given the label of the majority class from splitting the training data on that feature. Inference is done by simply testing the feature and assigning the label following the appropriate branch.

For AdaBoost, decision stump learning also accepts a distribution of the data as input. Therefore each training example is weighted differently based on the distribution, which affects the information gain computation and leaf node labels in every iteration of AdaBoost.

3 Bagging

Bootstrap aggregating (bagging) is a machine learning ensemble algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It also reduces variance and helps to avoid over fitting.

Bagging uses different training sets for the learning algorithm and learns a machine learning algorithm which combines the hypothesis learned from the different training sets. To make the training sets, we draw random instances of the whole training set and use the newly generated training sets to learn different hypothesis. Then we combine these different hypothesis by majority voting to come up with the best possible hypothesis for the training set and test the testing data using this hypothesis. Since the random samples of the training data will vary, the hypothesis that we generate will account for most of the variations in the training data and the final combining of the hypothesis will eventually reduce the variance of the classifier.

4 AdaBoost

Boosting is an ensemble method that considers errors from previous hypotheses to decide how to sample the training data for the next iteration. This is in contrast to bagging, which individual classifiers were independent. Boosting works when the base learner is weak learner, i.e. its classification accuracy is slightly better than random.

AdaBoost is a particular boosting algorithm that maintains and updates

a distribution over the training data based on the hypothesis error at each iteration. Thus each training data point is given more or less weight according to the distribution D. At each iteration t AdaBoost records the current hypothesis and a weight α_t computed from the weighted error ϵ_t. The distribution D is updated using the computed α_t.

These equations are shown below, for m training examples x_i and their labels y_i, distribution $D_t(i)$ at time t for ith example, hypothesis h_t from weak base learner (i.e. decision stump) at time t and Z is a normalization factor to make D_{t+1} a distribution:

$$\epsilon_t = \sum_{i=1}^{m} D_t(i) I(y_i \neq h_t(x_i)) \tag{1}$$

$$\alpha_t = \frac{1}{2} log\left(\frac{1 - \epsilon_t}{\epsilon_t}\right) \tag{2}$$

$$D_{t+1}(i) = \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z} \qquad \forall i = 1, ..., m \tag{3}$$

$$\tag{4}$$

The final hypothesis is just the sign of the weighted sum of each hypothesis and its corresponding weight α.

5   Results

For AdaBoost, figure **??** plots the training errors versus the ensemble size, and figure **??** plots the test errors versus the ensemble size.

6   Discussion

For AdaBoost, the training error appears to decrease as the ensemble size increases but then flattens out at ensemble size 15 and greater for the given training data. This indicates that while AdaBoost continues to select new hypotheses and weights (otherwise it would terminate), the training error does not improve. Interestingly, from observing the ensemble of hypotheses for ensemble size 30, there is a brief period after iteration 15 where the decision stump uses feature 17 iteration after iteration with more or less
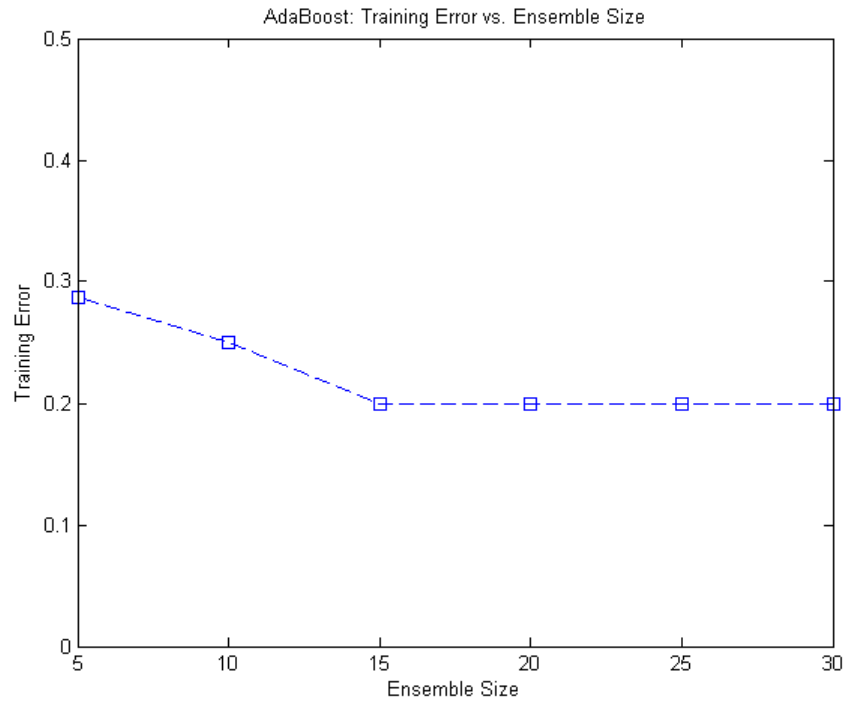
Figure 1: AdaBoost training errors on different ensemble sizes.

similar weights but with different leaf node labels. This may explain why the training error doesn't improve; the hypotheses and weights after iteration 15 have negligible impact on the final classifier.

The AdaBoost test (generalization) error decreases and performs the best at around ensemble size 20, but then increases afterward. This trend is typical of test curves in machine learning; it indicates that test error decreases with the training error but eventually increases due to over fitting.
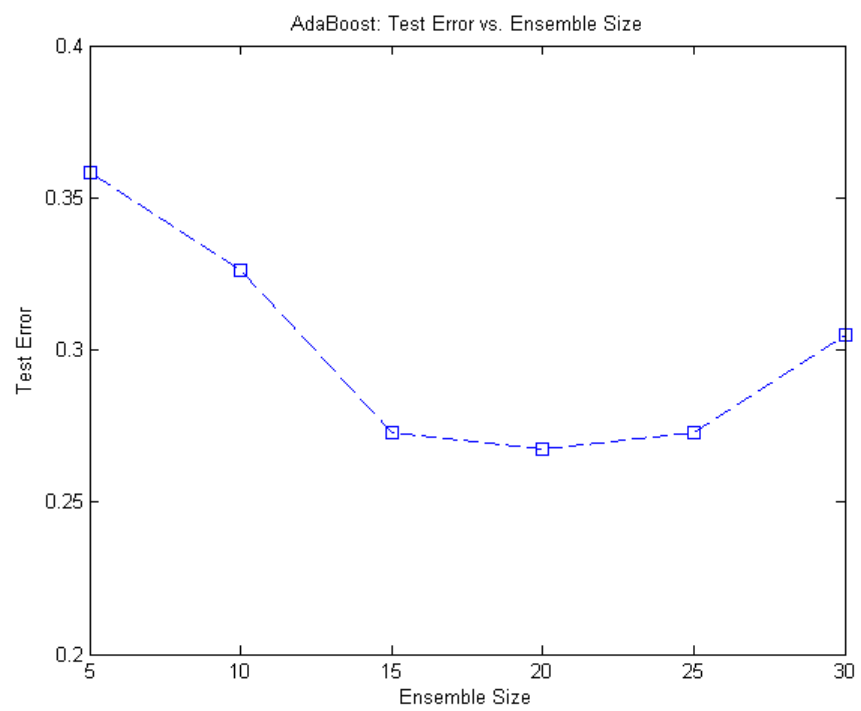
Figure 2: AdaBoost test errors on different ensemble sizes.