

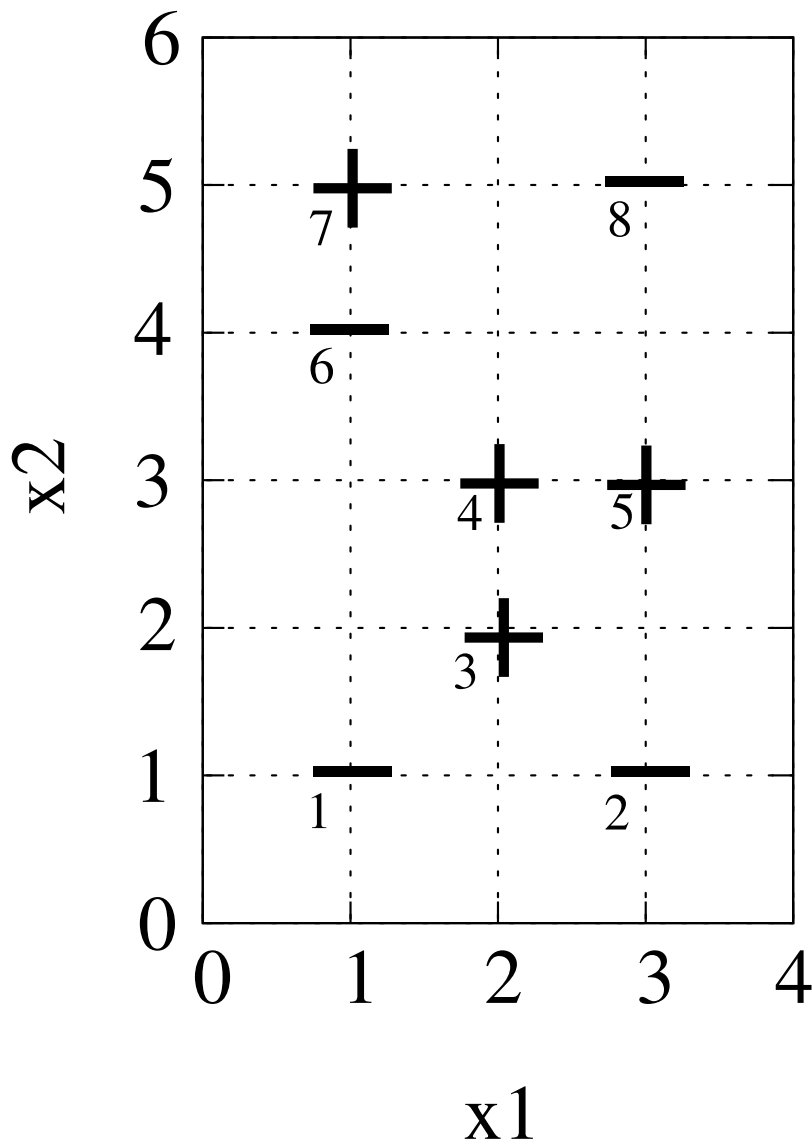
Boosting: Review Problem

(Luis E. Ortiz, 12/17/05)

Important Note: Boosting in general, and AdaBoost in particular, treats the weak learner as a black box: by the definition of *weak learner*, anything that guarantees to return a classifier that is strictly better than random-guessing will work (*i.e.*, anything with guaranteed error strictly smaller than $1/2$ will work).

Thus, we need to define what the weak learner is when using boosting. We will tell you exactly what the weak learner is in the exam.

The training data for this review problem is as given in the plot below, where $+$ corresponds to an output of 1 and $-$ to -1 . For convenience, we are assigning a number to each data point; the number is drawn near the data point in the plot. Our objective is to obtain a classifier by running AdaBoost for $T = 3$ iterations.



For this practice problem we will use the following weak learner, which outputs decision stumps. The weak learner considers each of the decision stump classifiers g_i , given in the next three pages, and outputs the one with the smallest (weighted) error. In case of ties, it outputs the classifier with the smallest index: g_1 is the **most preferred** while g_{14} is the **least preferred**. Both a formal mathematical definition and a drawing of the decision stump of each classifier are given in the next three pages. Note that classifier g_1 **always** classifies points as positive (1), while g_2 always classifies points as negative (-1). More generally, for each classifier, there is also its “inverse” in the sense that they use the same test but have their labels inverted: the decision-stump branches have their labels inverted. For each odd number i , classifier g_{i+1} is the “inverse” of g_i .

NOTE again that a different weak learner could have been used! The weak learner described here was just our choice for this particular problem.

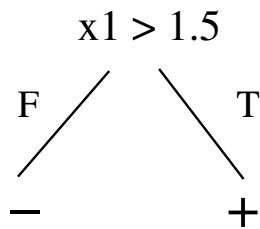
$$g_1(x) = 1$$

Always +

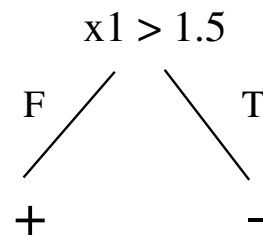
$$g_2(x) = -g_1(x) = -1$$

Always -

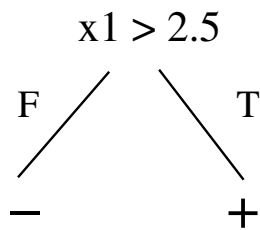
$$g_3(x) = \begin{cases} 1 & \text{if } x_1 > 1.5, \\ -1 & \text{otherwise.} \end{cases}$$



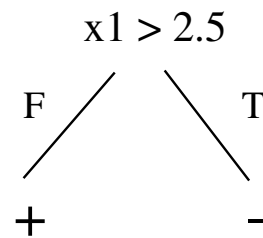
$$g_4(x) = -g_3(x)$$



$$g_5(x) = \begin{cases} 1 & \text{if } x_1 > 2.5, \\ -1 & \text{otherwise.} \end{cases}$$

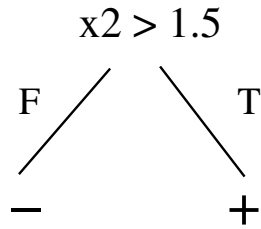


$$g_6(x) = -g_5(x)$$

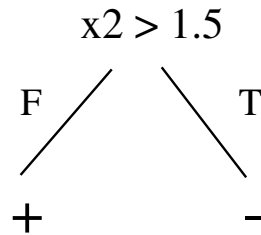


(CONTINUE ON NEXT PAGE...)

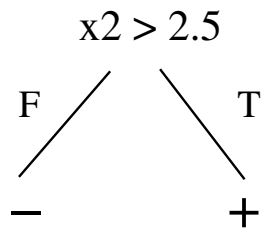
$$g_7(x) = \begin{cases} 1 & \text{if } x_2 > 1.5, \\ -1 & \text{otherwise.} \end{cases}$$



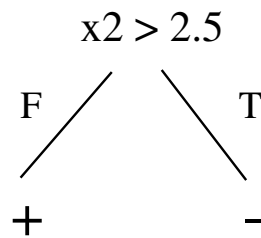
$$g_8(x) = -g_7(x)$$



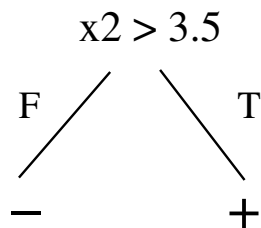
$$g_9(x) = \begin{cases} 1 & \text{if } x_2 > 2.5, \\ -1 & \text{otherwise.} \end{cases}$$



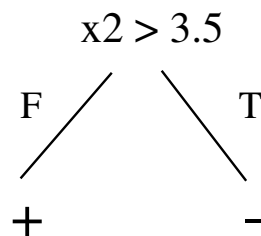
$$g_{10}(x) = -g_9(x)$$



$$g_{11}(x) = \begin{cases} 1 & \text{if } x_2 > 3.5, \\ -1 & \text{otherwise.} \end{cases}$$

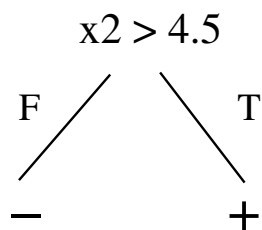


$$g_{12}(x) = -g_{11}(x)$$

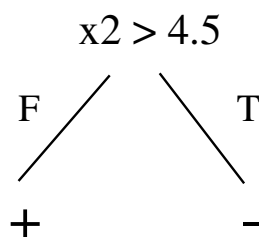


(CONTINUE ON NEXT PAGE...)

$$g_{13}(x) = \begin{cases} 1 & \text{if } x_2 > 4.5, \\ -1 & \text{otherwise.} \end{cases}$$



$$g_{13}(x) = -g_{14}(x)$$



(CONTINUE PROBLEM ON NEXT PAGE...)

After $T = 3$ iterations, AdaBoost outputs the classifier

$$H(x) = \text{sign}(\alpha_1 h_1(x) + \alpha_2 h_2(x) + \alpha_3 h_3(x)),$$

where

$$\text{sign}(z) = \begin{cases} 1 & \text{if } z > 0, \\ -1 & \text{otherwise,} \end{cases}$$

and each h_t corresponds to the output of the weak learner at iteration t (in this case, the g_i with the smallest (weighted) error for the weighted data at iteration t).

We will assume a uniform weight over the data points during the first iteration of AdaBoost: each data point initially has the same weight $1/8$.

Tips

Here are some tips to simplify computation of the AdaBoost classifier.

1. The way in which the data is split by the g_i 's (more generally, each possible classifier that the weak learner uses) is the same at every iteration. This can save time in computing the errors since the data points that the classifier labels incorrectly are the same at every iteration, only the *weights* associated with those data points, and hence the classifier error, change with the iteration.
2. If a given classifier g has error ϵ , then its “inverse” has error $1 - \epsilon$. Naturally, we would pick the one with the smallest error. Thus, if a classifier has error $> 1/2$ then we know we can ignore it during that iteration since it cannot be output by the weak learner; in fact, its inverse will have error $< 1/2$ and therefore better.
3. Since the weights need to sum to one, if we have a set of data points all with the same weight and all changing in the same direction (decreasing or increasing), then their new weight will be the same. This often allows us to avoid manually computing the weights of all the data points: we could infer them from the values we found for the new weights of other data points.

Iteration 1

Since all the weights are the same, only the number of errors determines the best weak classifier g_i out of the set of all possible classifiers $\{g_i\}$. Also, the mistakes made by each classifier are the same for all iterations, what changes between iterations is the weights of the data points and therefore the classifier error (see tip 1 above). Thus, it is convenient to first write a table listing the data points that the weak classifier labels incorrectly. Naturally, a classifier and its inverse will have complementary sets of such points, so you might want to just write the list for one of them and not both, if you are comfortable with that. The table follows.

Data points where classifier errs			
g_1	1, 2, 6, 8	g_2	3, 4, 5, 7
g_3	2, 7, 8	g_4	1, 3, 4, 5, 6
g_5	2, 3, 4, 7, 8	g_6	1, 5, 6
g_7	6, 8	g_8	1, 2, 3, 4, 5, 7
g_9	3, 6, 8	g_{10}	1, 2, 4, 5, 7
g_{11}	3, 4, 5, 6, 8	g_{12}	1, 2, 7
g_{13}	3, 4, 5, 8	g_{14}	1, 2, 6, 7

From this, it is clear that the best classifier is g_7 . So, we have

$$h_1 = g_7,$$

$$\epsilon_1 = 2/8 = 1/4,$$

and

$$\alpha_1 = \frac{1}{2} \ln(3).$$

Let us denote the weights of data point i at iteration t as $D_t(i)$. Thus, at iteration 1, we had $D_1(i) = 1/8$. To obtain the new weights, we just need to realize that only the weights for data points 6, 8 increase. Some useful properties of the natural-logarithm function \ln and the natural-exponential function e are that (1) $c \ln(x) = \ln(x^c)$, and (2) $e^{\ln(x)} = x$. Using (a) those properties, (b) the formula that AdaBoost uses to update the weights, and (c) some simplifications, we obtain the following new set of weights: for $i = 6, 8$, we have

$$D_2(i) = 1/4.$$

Again, to get the weights for the other data points all we need to realize is that they will all have the same value, since they all started with the same weight, they all must decrease by the same factor, and the sum of all the weights must equal 1. This simplifies their calculation (see tip 3 above). More specifically, for $i \neq 6, 8$, we have

$$D_2(i) = (1 - 2(1/4))(1/6) = 1/12.$$

Iteration 2

Now all we need to do is use the table of mistakes above to calculate the *weighted* error of the classifiers, then select the one with the smallest error. The fact that errors of a classifier and its inverse are deterministically related simplifies the calculation of those values (see tip 2 above). It is also convenient to express all the weight as fractions with the same denominator in order to speed up the calculation of the weights and determine the best classifier. Also, any classifier that makes mistakes in both 6 and 8 will have classification error at least $1/2$. Realizing that also speeds things up, since we do not have to calculate the error of such a classifier (that classifier cannot be selected as the best). The table of errors at iteration 2 follows.

Classifiers errors			
g_1	$> 1/2$	g_2	$4(1/12) = 4/12$
g_3	$2(1/12) + (3/12) = 5/12$	g_4	$7/12$
g_5	$4(1/12) + (3/12) = 7/12$	g_6	$5/12$
g_7	$1/2$	g_8	$1/2$
g_9	$> 1/2$	g_{10}	$5(1/12) = 5/12$
g_{11}	$> 1/2$	g_{12}	$3(1/12) = 3/12$
g_{13}	$3(1/12) + (3/12) = 6/12$	g_{14}	$6/12$

From this we can see that g_{12} achieves the smallest error. Therefore, we have

$$h_2 = g_{12},$$

$$\epsilon_2 = 3/12 = 1/4,$$

and

$$\alpha_2 = \frac{1}{2} \ln(3).$$

Note that if we were to stop right now, the output of the final classifier when h_1 and h_2 disagree depends only on how the sign function is defined (since both classifiers have the same weight, the value of the function inside the sign function in the final classifier will evaluate to 0 when the classifiers disagree). However, we will continue and perform a third iteration of AdaBoost, in which the weak learner will output a tie-breaking classifier.

To compute the new weights, it helps to realize that the sets of data points that will have the same new weight are $\{3, 4, 5\}$, $\{6, 8\}$, and $\{1, 2, 7\}$. Thus, for $i = 3, 4, 5$, we have

$$D_3(i) = \frac{\frac{1}{12} \frac{1}{\sqrt{3}}}{3 \frac{1}{12} \frac{1}{\sqrt{3}} + 2 \frac{3}{12} \frac{1}{\sqrt{3}} + 3 \frac{1}{12} \sqrt{3}} = 1/18.$$

The first term in the denominator (the normalizing constant/factor) is associated with data points 3, 4, 5, the second with data point 6, 8 and the third with 1, 2, 7 (*i.e.*, the mistakes of h_2). For $i = 6, 8$, we have

$$D_3(i) = \frac{\frac{3}{12} \frac{1}{\sqrt{3}}}{3 \frac{1}{12} \frac{1}{\sqrt{3}} + 2 \frac{3}{12} \frac{1}{\sqrt{3}} + 3 \frac{1}{12} \sqrt{3}} = 3/18 = 1/6$$

Finally, noting that the weights need to sum to 1, we get that for $i = 1, 2, 7$,

$$D_3(i) = (1/3)(1 - (3(1/18) + 2(3/18))) = (1/3)(9/18) = 3/18 = 1/6.$$

Iteration 3

Using the same tricks as before, we can calculate the errors for the classifiers under the new weights on the data points. Note that any classifier that errs on any subset of size 3 of the set of data points $\{1, 2, 6, 7, 8\}$ will have error at least 1/2. That makes sense since, intuitively, AdaBoost is trying to learn something new and those points are the set of data points where h_1 and h_2 disagree (the “hardest” points, which as intuitively expected, have the largest weights). The table of errors follows.

Classifiers errors			
g_1	$> 1/2$	g_2	$3(1/18) + (3/18) = 6/18$
g_3	$1/2$	g_4	$1/2$
g_5	$\geq 1/2$	g_6	$(1/18) + 2(3/18) = 7/18$
g_7	$2(3/18) = 6/18$	g_8	$> 1/2$
g_9	$(1/18) + 2(3/18) = 7/18$	g_{10}	$> 1/2$
g_{11}	$1/2$	g_{12}	$1/2$
g_{13}	$3(1/18) + (3/18) = 6/18$	g_{14}	$> 1/2$

From the table, we can see that g_2 , g_7 and g_{13} all achieve the smallest error. But, by our order of preference, the weak learner outputs g_2 . Thus, we have

$$h_3 = g_2,$$

$$\epsilon_3 = 6/18 = 1/3,$$

and

$$\alpha_3 = \frac{1}{2} \ln(2).$$

Putting Everything Together

The final classifier of AdaBoost for this example can be written as

$$H(x) = \text{sign}((1/2) \ln(3)g_7(x) + (1/2) \ln(3)g_{12}(x) + (1/2) \ln(2)g_2(x)).$$

From this we can see that the places where h_1 and h_2 disagree ($x_2 < 1.5, x_2 > 3.5$) will be classified as negative (-1). Thus, the only data point that H classifies incorrectly is 7. The following plot shows the decision boundary of H .

