2021/10/27

Ensemble Methods: Adaptive Boosting (AdaBoost)

Boosting refers to the ensemble methods that train a weak classifier **sequentially**, each trying to correct its predecessor. In other words, boosting uses the misclassifications of prior iterations to influence the training of the next iterative classifier. Boosting advocates an approach in which the weak classifier is forced to focus its attention on the "hardest" examples, that is, the ones for which the previous classifiers were most apt to give incorrect predictions.

There are many boosting methods available, but by far the most popular are **AdaBoost** (short for **Adaptive Boosting**) and **Gradient Boosting**. Let's start with AdaBoost.

One way for a new predictor to correct its predecessor is to pay a bit more attention to **the training instances that the predecessor underfitted**. This results in new predictors focusing more and more on the hard cases. This is the technique used by **AdaBoost**.

j: boosting round

i: train example

1. Theoretical Minimum for AdaBoost

Let $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$ denote a set of N training samples. In AdaBoost algorithm, the importance of a base classifier h_j depends on its weighted error rate[1], which is defined as:

$$\varepsilon_j = \frac{\sum_{i=1}^{N} w_i \delta\big(h_j(\mathbf{x}_i) \neq y_i\big)}{\sum_{i=1}^{N} w_i \longrightarrow 1.0}$$

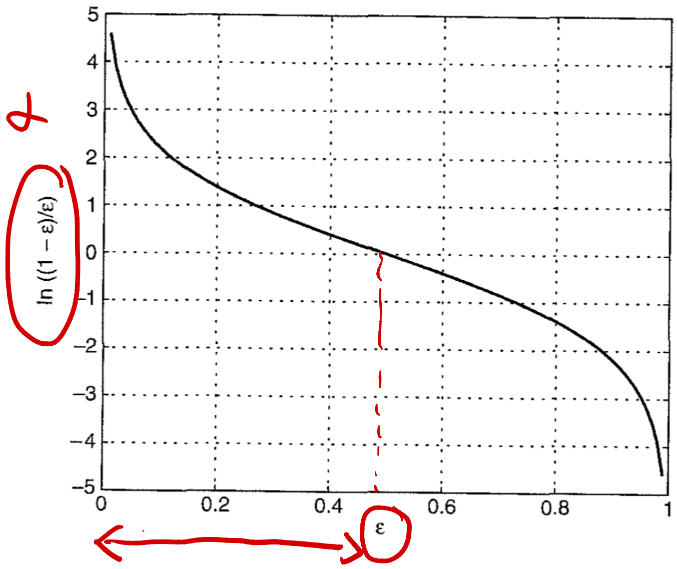
where $\delta(p) = 1$ if the predicate p is true, and 0 otherwise. The importance of a classifier h_j is given by the following parameter:

(1) $\alpha_j = \frac{1}{2}\ln(\frac{1-\varepsilon_j}{\varepsilon_j})$

Figure below plots the relationship between α_j and ε_j. What have you observed?

A: α_j has a large positive value if the error rate is close to 0 and a large negative value if the error is close to 1.

[1] For equally weighted examples, error rate reduces to $e = \frac{\text{Number of wrong prediction}}{\text{Total Number of prediction}}$ and accuracy $= 1 - e$. The weighted error rate can also be interpreted as the weighted zero-one training loss.



Remark: As the base classifier should do better than a classifier that performs random guessing ($\varepsilon < 0.5$), α parameter in general should be greater than 0.

The α_j parameter is also used to **update the weight** of the training examples. Let $w_i^{(j)}$ denote the weight assigned to example $(\mathbf{x}_i, y_i)$ during **the j^{th} boosting round**. The weight update mechanism for AdaBoost is given by:

*

$$(2) \quad w_i^{(j+1)} = \frac{w_i^{(j)}}{Z_j} \times \begin{cases} e^{-\alpha_j} & \text{if } h_j(\mathbf{x}_i) = y_i \\ e^{\alpha_j} & \text{if } h_j(\mathbf{x}_i) \neq y_i \end{cases}$$

← 錯的 training example

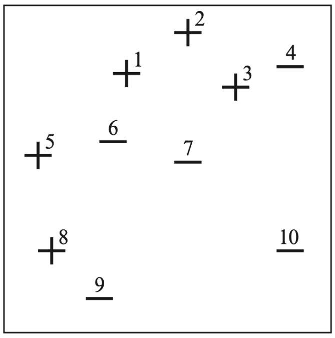
where Z_j is the normalization factor used to ensure $\sum_i w_i^{(j+1)} = 1$.

Q: What does the weight update formula imply?
A: The weight increases for the incorrectly classified examples and decreases for the correctly classified examples.

2. AdaBoost Toy Example

To illustrate how AdaBoost works, let us look at the tiny toy learning problem shown in figure below. Here, the instances are points in the plane which are labeled + or −. Let the label + be +1 and – be –1. In this case, there are 10 training examples, as shown in the figure; five are positive ($y_i = +1$) and five are negative ($y_i = -1$). The weight updating formula (Eq. (2)) can then be simplified as:

(3) $w_i^{(j+1)} = \frac{w_i^{(j)}}{Z_j} \times e^{-\alpha_j y_i h_j(x_i)}$



假設

$+ : 1 \longrightarrow -1$

$- : -1 \longrightarrow +1$

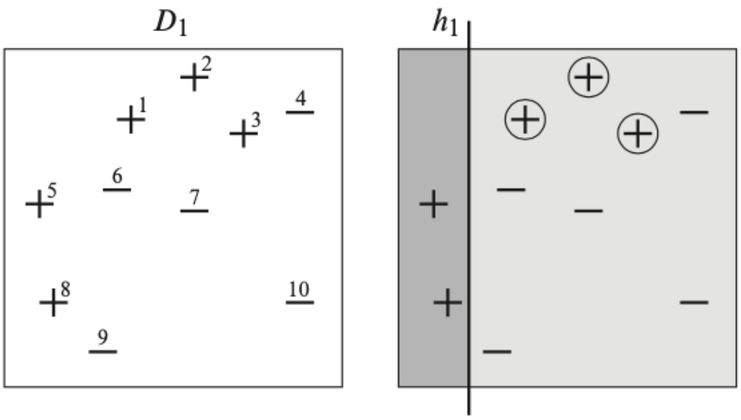
$h_j(x_i) = y_i$

| —

Let us suppose that our base learner finds classifiers defined by **vertical or horizontal lines** through the plane. For instance, such a base classifier defined by a vertical line might classify all points to the right of the line as positive, and all points to the left as negative.

It can be checked that no base classifier of this form correctly classifies more than seven of the ten training examples, meaning that none has an unweighted training error below 30%. On each round t, we suppose that the base learner always finds the base hypothesis of this form (miss three points) that has minimum weighted error with respect to the distribution D_t.

Round 1



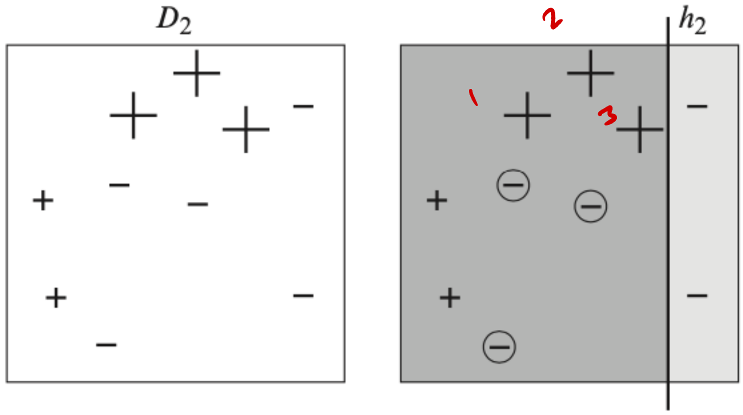
On round 1, AdaBoost assigns equal weight to all of the examples, as is indicated in the figure by drawing all examples in the box marked D_1 to be of the same size. Given examples with these weights, the base learner chooses the base hypothesis indicated by h_1 in the figure, which classifies points as positive if and only if they lie to the left of this line. This hypothesis incorrectly classifies three points—

namely, the three circled positive points—so its error ε_1 is 0.30. Plugging into the formula of algorithm (1) gives $\alpha_1 \approx 0.42$.

	1	2	3	4	5	6	7	8	9	10	
$D_1(i)$ $w_i^{(1)}$	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	$\epsilon_1 = 0.30, \alpha_1 \approx 0.42$
$e^{-\alpha_1 y_i h_1(x_i)}$	1.53	1.53	1.53	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
$D_1(i)\ e^{-\alpha_1 y_i h_1(x_i)}$	0.15	0.15	0.15	0.07	0.07	0.07	0.07	0.07	0.07	0.07	$Z_1 \approx 0.92$
$D_2(i)$	0.17	0.17	0.17	0.07	0.07	0.07	0.07	0.07	0.07	0.07	$\epsilon_2 \approx 0.21, \alpha_2 \approx 0.65$

Round 2

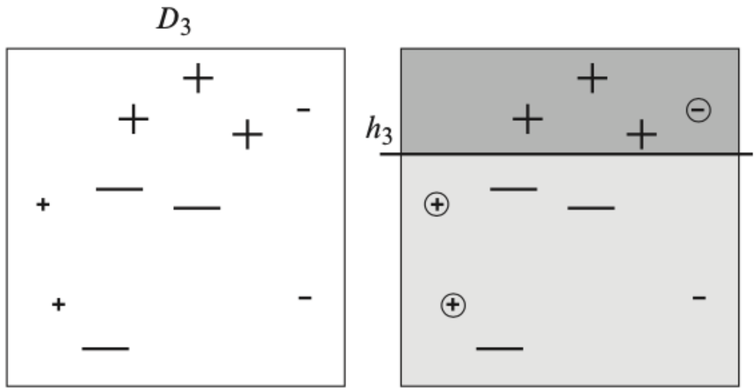
On round 2, the base learner **pays special attention** on the three relatively high-weight points missed by h_1 and chooses the line marked h_2. This base classifier correctly classifies the three relatively high-weight points at the expense of missing three other comparatively low-weight points which were correctly classified by h_1. Under distribution D_2, these three points have weight only around 0.07, so the error of h_2 with respect to D_2, $\varepsilon_2 \approx 0.21$ and $\alpha_2 \approx 0.65$.



	1	2	3	4	5	6	7	8	9	10	
$D_1(i)$	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	$\epsilon_1 = 0.30, \alpha_1 \approx 0.42$
$e^{-\alpha_1 y_i h_1(x_i)}$	1.53	1.53	1.53	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
$D_1(i)\ e^{-\alpha_1 y_i h_1(x_i)}$	0.15	0.15	0.15	0.07	0.07	0.07	0.07	0.07	0.07	0.07	$Z_1 \approx 0.92$
$D_2(i)$	0.17	0.17	0.17	0.07	0.07	0.07	0.07	0.07	0.07	0.07	$\epsilon_2 \approx 0.21, \alpha_2 \approx 0.65$
$e^{-\alpha_2 y_i h_2(x_i)}$	0.52	0.52	0.52	0.52	0.52	1.91	1.91	0.52	1.91	0.52	
$D_2(i)\ e^{-\alpha_2 y_i h_2(x_i)}$	0.09	0.09	0.09	0.04	0.04	0.14	0.14	0.04	0.14	0.04	$Z_2 \approx 0.82$
$D_3(i)$	0.11	0.11	0.11	0.05	0.05	0.17	0.17	0.05	0.17	0.05	$\epsilon_3 \approx 0.14, \alpha_3 \approx 0.92$

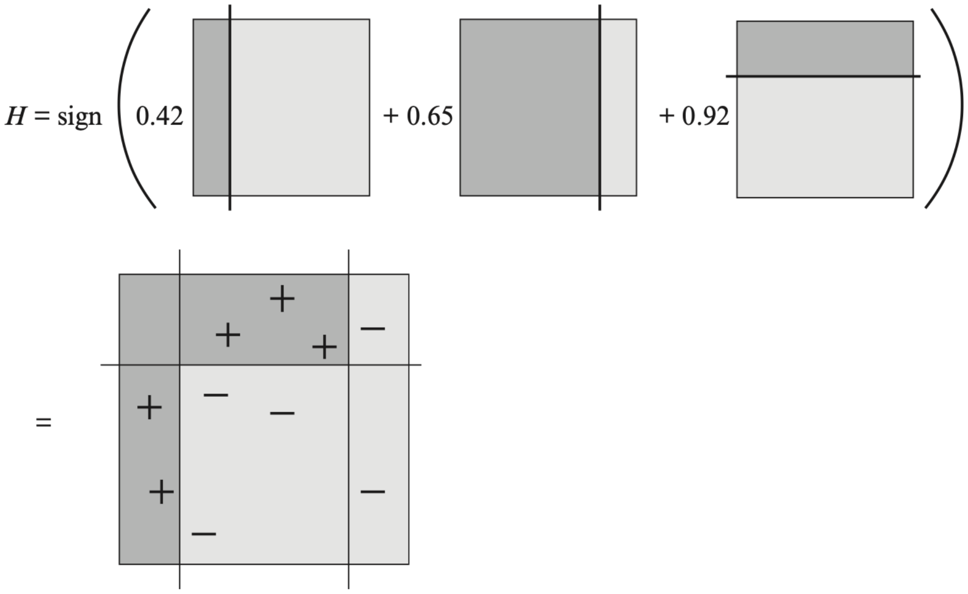
Round 3

On round 3, classifier h_3 is chosen. This classifier misses none of the points misclassified by h_1 and h_2 since these points have relatively high weight under D_3. Instead, it misclassifies three points which, because they were not misclassified by h_1 or h_2, are of very low weight under D_3. On round 3, $\varepsilon_3 \approx 0.14$ and $\alpha_3 \approx 0.92$.



	1	2	3	4	5	6	7	8	9	10	
$D_1(i)$	$\underline{0.10}$	$\underline{0.10}$	$\underline{0.10}$	0.10	0.10	0.10	0.10	0.10	0.10	0.10	$\epsilon_1 = 0.30, \alpha_1 \approx 0.42$
$e^{-\alpha_1 y_i h_1(x_i)}$	1.53	1.53	1.53	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
$D_1(i)\ e^{-\alpha_1 y_i h_1(x_i)}$	0.15	0.15	0.15	0.07	0.07	0.07	0.07	0.07	0.07	0.07	$Z_1 \approx 0.92$
$D_2(i)$	0.17	0.17	0.17	0.07	0.07	$\underline{0.07}$	$\underline{0.07}$	0.07	$\underline{0.07}$	0.07	$\epsilon_2 \approx 0.21, \alpha_2 \approx 0.65$
$e^{-\alpha_2 y_i h_2(x_i)}$	0.52	0.52	0.52	0.52	0.52	1.91	1.91	0.52	1.91	0.52	
$D_2(i)\ e^{-\alpha_2 y_i h_2(x_i)}$	0.09	0.09	0.09	0.04	0.04	0.14	0.14	0.04	0.14	0.04	$Z_2 \approx 0.82$
$D_3(i)$	0.11	0.11	0.11	$\underline{0.05}$	$\underline{0.05}$	0.17	0.17	$\underline{0.05}$	0.17	0.05	$\epsilon_3 \approx 0.14, \alpha_3 \approx 0.92$
$e^{-\alpha_3 y_i h_3(x_i)}$	0.40	0.40	0.40	2.52	2.52	0.40	0.40	2.52	0.40	0.40	
$D_3(i)\ e^{-\alpha_3 y_i h_3(x_i)}$	0.04	0.04	0.04	0.11	0.11	0.07	0.07	0.11	0.07	0.02	$Z_3 \approx 0.69$

The combined classifier H is **a weighted vote** of h_1, h_2 and h_3 as shown in figure below, where the weights on the respective classifiers are α_1, α_2, and α_3. Although each of the composite weak classifiers misclassifies three of the ten examples, **the combined classifier, correctly classifies all of the training examples!**



3. Summary

AdaBoost proceeds in iterative call to the base learner. Initially, all weights are set equally, but on each round, the weights of incorrectly classified examples are increased so that, effectively, hard examples get successively higher weight, forcing the base learner to focus its attention on them.