Adaboost

@S S Roy,19th Sept

---

1. **Goal.** Combine several weak learners $M_1, \dots, M_k$ into a strong classifier by reweighting training examples each round so that later learners focus on previously misclassified examples. Final prediction is a weighted vote of weak learners.
2. **Initialization.** Give each of the d training tuples equal weight $w_j^{(0)} = 1/d$.
3. **For each round $i = 1 \dots k$:**
   - Sample a training set D_i according to the current weights.
   - Train weak learner M_i on D_i.
   - Compute weighted error:

$$\text{error}(M_i) = \sum_{j: M_i(x_j) \neq y_j} w_j$$

- If $\text{error}(M_i) > 0.5$, reject M_i and try another weak learner.
- Compute classifier vote weight:

$$\alpha_i = \log \frac{1 - \text{error}(M_i)}{\text{error}(M_i)}$$

- Update sample weights: for each sample

$$w_j \leftarrow \begin{cases} w_j & \text{if misclassified by } M_i \\ w_j \times \dfrac{\text{error}(M_i)}{1 - \text{error}(M_i)} & \text{if correctly classified} \end{cases}$$

  then normalize all w_j so $\sum_j w_j = 1$.

4. **Final classification of a new X:**
   - For each class c, sum $\sum_i \alpha_i \cdot \mathbf{1}[M_i(x) = c]$.
   - Return class with largest total weight. (Equivalently: compute $S(x) = \sum_i \alpha_i M_i(x)$ if labels are ± 1, then $\text{sign}(S)$.)

Part B — Worked example

Sample	X_1	X_2	**True class** Y
1	1	2	+1
2	2	1	+1
3	2	3	+1
4	3	2	+1
5	3	3	+1
6	4	1	−1
7	4	2	−1

So classes are imbalanced: **5 positives (+1)** and **2 negatives (−1)**.

We will run AdaBoost for **k = 2** rounds (two weak learners). Weak learners are **decision stumps** (one feature threshold).

Round 1

Step 1 — Initialize weights

There are $d = 7$ examples, so initial weight for each:

$$w_j^{(0)} = \frac{1}{7} \approx 0.1428571429.$$

(We'll display decimals to 6 places when helpful: 0.142857.)

Step 2 — Choose a weak learner

Pick a decision stump:

- Rule M_1: if $X_1 < 3$ predict $+1$, else predict -1.

Apply M_1 to all samples:

Sample	X_1	True Y	M_1 prediction	Correct?
1	1	+1	+1	yes
2	2	+1	+1	yes
3	2	+1	+1	yes
4	3	+1	—(3 is not <3) → -1	**no**
5	3	+1	-1	**no**
6	4	-1	-1	yes
7	4	-1	-1	yes

Misclassified: samples 4 and 5 (both are positives that stump got wrong).

Step 3 — Weighted error of M_1

$$\text{error}(M_1) = w_4^{(0)} + w_5^{(0)} = \frac{1}{7} + \frac{1}{7} = \frac{2}{7} \approx 0.285714.$$

(This is ≈ 0.285714, less than 0.5, so M_1 is accepted.)

Step 4 — Classifier vote weight

$$\alpha_1 = \log \frac{1 - \text{error}}{\text{error}} = \log \frac{1 - 2/7}{2/7} = \log \frac{5/7}{2/7} = \log \frac{5}{2} = \log(2.5).$$

Numeric value (natural log):

$$\alpha_1 \approx \ln(2.5) \approx 0.916291.$$

Step 5 — Update sample weights (before normalization)

Compute ratio used for correctly classified samples:

$$r = \frac{\text{error}}{1 - \text{error}} = \frac{2/7}{5/7} = \frac{2}{5} = 0.4.$$

- For **misclassified** samples (4,5): keep weight = 0.142857.
- For **correctly classified** samples (1,2,3,6,7): multiply weight by $r = 0.4$: new weight = $0.142857 \times 0.4 = 0.0571428$.

So **unnormalized** weights after update:

Sample	unnorm. weight
1	0.0571429
2	0.0571429
3	0.0571429
4	0.1428571
5	0.1428571
6	0.0571429
7	0.0571429

Sum of unnormalized weights:

$$S = 5 \times 0.0571429 + 2 \times 0.1428571 = 0.2857145 + 0.2857142 \approx 0.5714287$$

(Exact rational value: $S = \frac{5}{7} \cdot 0.4 + \frac{2}{7} = \frac{2}{7} + \frac{2}{7} = \frac{4}{7}$ — but we keep decimals; normalized result follows.)

Step 6 — Normalize (divide each by S)

Compute normalized weights $w_j^{(1)}$ = (unnorm weight)/S.

Because numbers are symmetric, we can compute:

- For misclassified samples (4 & 5):

$$w_4^{(1)} = w_5^{(1)} = \frac{0.1428571}{0.5714287} \approx 0.25.$$

- For correctly classified samples (1,2,3,6,7):

$$w_j^{(1)} = \frac{0.0571429}{0.5714287} \approx 0.10.$$

Check sum:

$$5 \times 0.10 + 2 \times 0.25 = 0.50 + 0.50 = 1.00.$$

So after Round 1 final weights:

Sample	$w_j^{(1)}$
1	0.10
2	0.10
3	0.10
4	0.25
5	0.25
6	0.10
7	0.10

Observation: the two samples misclassified (4 & 5) now have larger weight (0.25 each); the remaining five samples have reduced weight (0.10 each). The algorithm has focused attention on the previously hard (positive) examples.

Round 2

Step 1 — Train a new weak learner using the new weights

We try a decision stump that splits on X_2. Consider the rule:

- M_2: if $X_2 \geqslant 2.5$ predict $+1$, else predict -1.

Apply M_2 to all samples:

Sample	X_2	True Y	M_2 pred	Correct?
1	2	+1	(2 >=2.5?) → -1	**no**
2	1	+1	-1	**no**
3	3	+1	+1	yes
4	2	+1	-1	**no**
5	3	+1	+1	yes
6	1	-1	-1	yes
7	2	-1	-1	yes

So with this stump, **misclassified** samples are 1, 2, 4 (all positives with $X_2 < 2.5$). Samples 3 & 5 (positives with $X_2 \geqslant 2.5$) are correctly classified; negatives 6 & 7 also correct.

Step 2 — Compute weighted error

Use weights $w_j^{(1)}$ from after Round 1:

$$\text{error}(M_2) = w_1^{(1)} + w_2^{(1)} + w_4^{(1)} = 0.10 + 0.10 + 0.25 = 0.45.$$

This error is $0.45 < 0.5$, so M_2 is acceptable.

Step 3 — Compute classifier vote weight α_2

$$\alpha_2 = \log \frac{1 - 0.45}{0.45} = \log \frac{0.55}{0.45} = \log\left(\frac{11}{9}\right).$$

Numeric:

$$\alpha_2 \approx \ln(1.222222\ldots) \approx 0.2006707.$$

(We'll keep 6 dp: $\alpha_2 \approx 0.200671$.)

Step 4 — Update sample weights (before normalization)

Compute ratio for correctly classified samples:

$$r_2 = \frac{\text{error}(M_2)}{1 - \text{error}(M_2)} = \frac{0.45}{0.55} \approx 0.8181818.$$

Update rule: if correctly classified → multiply weight by r_2. If misclassified → keep weight.

List samples and their unnormalized updated weights:

- Misclassified (keep same):
  - sample1: unnorm = 0.10
  - sample2: unnorm = 0.10
  - sample4: unnorm = 0.25
- Correctly classified (multiply by $r_2 \approx 0.8181818$):
  - sample3: $0.10 \times 0.8181818 \approx 0.0818182$
  - sample5: $0.25 \times 0.8181818 \approx 0.2045455$
  - sample6: $0.10 \times 0.8181818 \approx 0.0818182$
  - sample7: $0.10 \times 0.8181818 \approx 0.0818182$

Now compute sum of unnormalized weights S_2:

$$S_2 = (0.10 + 0.10 + 0.25) + (0.0818182 + 0.2045455 + 0.0818182 + 0.0818182).$$

Compute each group:

- Misclassified sum = 0.45.
- Correctly-classified sum $\approx 0.0818182 + 0.2045455 + 0.0818182 + 0.0818182 = 0.4499999$ (rounding gives 0.45).

So total $S_2 \approx 0.45 + 0.45 = 0.90$.

Step 5 — Normalize to get final weights $w_j^{(2)}$

Divide each unnormalized weight by $S_2 \approx 0.90$:

- For misclassified:
  - $w_1^{(2)} = 0.10/0.90 \approx 0.1111111$
  - $w_2^{(2)} = 0.10/0.90 \approx 0.1111111$
  - $w_4^{(2)} = 0.25/0.90 \approx 0.2777778$
- For correctly classified:
  - $w_3^{(2)} \approx 0.0818182/0.90 \approx 0.0909091$
  - $w_5^{(2)} \approx 0.2045455/0.90 \approx 0.2272728$

- $w_6^{(2)} \approx 0.0818182/0.90 \approx 0.0909091$
- $w_7^{(2)} \approx 0.0818182/0.90 \approx 0.0909091$

Check sum:

$$2 \times 0.1111111 + 1 \times 0.2777778 + 3 \times 0.0909091 + 1 \times 0.2272728$$

$$= 0.2222222 + 0.2777778 + 0.2727273 + 0.2272728 \approx 1.0000001 \text{ (rounding er}$$

So final weights after Round 2 (rounded to 6 dp):

Sample	$w_j^{(2)}$ (approx)
1	0.111111
2	0.111111
3	0.090909
4	0.277778
5	0.227273
6	0.090909
7	0.090909

Observation: sample 4 (a positive that was misclassified in both rounds) now has the highest weight (~0.2778). Sample 5 is next (~0.2273). The negatives (6 & 7) are relatively low weight (~0.0909 each).

Final Ensemble Model (after 2 rounds)

We have two accepted classifiers with weights:

- M_1: rule $X_1 < 3 \Rightarrow +1$ else -1, weight $\alpha_1 \approx 0.916291$.
- M_2: rule $X_2 \geqslant 2.5 \Rightarrow +1$ else -1, weight $\alpha_2 \approx 0.200671$.

For a new input x, compute the weighted sum:

$$S(x) = \alpha_1 \cdot M_1(x) + \alpha_2 \cdot M_2(x).$$

Return $\text{sign}(S(x))$ (ties break arbitrarily, e.g., +1).

Example: classify training sample 4 (as a check)

Sample 4: $(X_1, X_2) = (3, 2)$, true $Y = +1$.

- $M_1(4)$: since $X_1 = 3$ not <3 → predict -1.
- $M_2(4)$: since $X_2 = 2 < 2.5$ → predict -1.

Weighted sum:

$$S(4) = 0.916291 \times (-1) + 0.200671 \times (-1) = -(0.916291 + 0.200671) = -1.116962.$$

Sign is negative → the ensemble predicts -1, so the ensemble **misclassifies sample 4** (true label +1). That reflects that sample 4 remained hard for both stumps in our chosen stump set.

Points to remember-->

- We used an **imbalanced training set** (5 positives vs 2 negatives). AdaBoost correctly shifted attention (weights) to the positive examples that were difficult (samples 4 and 5).
- After Round 1, misclassified positives (4 & 5) had higher weight (0.25 each) while the rest had 0.10.
- In Round 2 we chose M_2 to reduce error on some of these; final weights show sample 4 is now the hardest example (~0.278).
- Final ensemble weights α_1 and α_2 reflect relative strengths: $\alpha_1 \approx 0.9163$, $\alpha_2 \approx 0.2007$. So M_1 has more influence.
- Even after boosting, some hard examples may remain misclassified if weak learners cannot capture their pattern; AdaBoost will continue to focus on them in subsequent rounds.
