Machine Learning: Principles and Techniques

Bagging, Boosting
IE 506

March 24 & 28, 2023

1. Classification Algorithms
   - Bagging
   - Boosting

Classification Algorithms: Bagging

Bagging

Bagging (or Bootstrap aggregating)

Create different bootstrap data sets

1. Let D denote the original data set of N samples, $\mathcal{Y}$ denote the label set and let k denote the number of bootstrap data sets.
2. For $j = 1, 2, \ldots k$ do:
   2.1. Create a data set D_j of size N by sampling uniformly at random with replacement from D.
   2.2. Build a base classifier C_j using D_j.
3. Inference for test sample $\hat{x}$ is done as: $\hat{y} = \arg\max_{y \in \mathcal{Y}} \sum_{j=1}^{k} \mathbb{I}(C_j(\hat{x}) == y)$

Each sample has a probability of $1 - (1 - 1/N)^N$ of getting selected in each data set D_j. For large N, this quantity can be approximated as $1 - 1/e \approx 0.632$.

Thus on an average each bootstrap data set D_j has 63% of samples in the original data set D.

Bagging

Table Example of data set used to construct an ensemble of bagging classifiers.

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1	1	1	−1	−1	−1	−1	1	1	1

Bagging Round 1:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Bagging Round 2:

x	0.1	0.2	0.3	0.4	0.5	0.8	0.9	1	1	1
y	1	1	1	-1	-1	1	1	1	1	1

x <= 0.65 ==> y = 1
x > 0.65 ==> y = 1

Bagging Round 3:

x	0.1	0.2	0.3	0.4	0.4	0.5	0.7	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Bagging Round 4:

x	0.1	0.1	0.2	0.4	0.4	0.5	0.5	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x <= 0.3 ==> y = 1
x > 0.3 ==> y = -1

Bagging Round 5:

x	0.1	0.1	0.2	0.5	0.6	0.6	0.6	1	1	1
y	1	1	1	-1	-1	-1	-1	1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Bagging Round 6:

x	0.2	0.4	0.5	0.6	0.7	0.7	0.7	0.8	0.9	1
y	1	-1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 7:

x	0.1	0.4	0.4	0.6	0.7	0.8	0.9	0.9	0.9	1
y	1	-1	-1	-1	-1	1	1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 8:

x	0.1	0.2	0.5	0.5	0.5	0.7	0.7	0.8	0.9	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 9:

x	0.1	0.3	0.4	0.4	0.6	0.7	0.7	0.8	1	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 10:

x	0.1	0.1	0.1	0.1	0.3	0.3	0.8	0.8	0.9	0.9
y	1	1	1	1	1	1	1	1	1	1

x <= 0.05 ==> y = -1
x > 0.05 ==> y = 1

Example from Introduction to Data Mining book by Tan et al.

Bagging

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	1	1	1	-1	-1	-1	-1	-1	-1	-1
2	1	1	1	1	1	1	1	1	1	1
3	1	1	1	-1	-1	-1	-1	-1	-1	-1
4	1	1	1	-1	-1	-1	-1	-1	-1	-1
5	1	1	1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	1	1	1
7	-1	-1	-1	-1	-1	-1	-1	1	1	1
8	-1	-1	-1	-1	-1	-1	-1	1	1	1
9	-1	-1	-1	-1	-1	-1	-1	1	1	1
10	1	1	1	1	1	1	1	1	1	1
Sum	2	2	2	-6	-6	-6	-6	2	2	2
Sign	1	1	1	-1	-1	-1	-1	1	1	1
True Class	1	1	1	-1	-1	-1	-1	1	1	1

Figure 5.36. Example of combining classifiers constructed using the bagging approach.

Classification Algorithms: Boosting

Boosting: AdaBoost

Algorithm AdaBoost

Input: sequence of N labeled examples $\langle (x_1, y_1), \ldots, (x_N, y_N) \rangle$
distribution D over the N examples
weak learning algorithm **WeakLearn**
integer T specifying number of iterations

Initialize the weight vector: $w_i^1 = D(i)$ for $i = 1, \ldots, N$.

Do for $t = 1, 2, \ldots, T$

1. Set
$$\mathbf{p}^t = \frac{\mathbf{w}^t}{\sum_{i=1}^N w_i^t}$$
2. Call **WeakLearn**, providing it with the distribution $\mathbf{p}^t$; get back a hypothesis $h_t : X \rightarrow [0, 1]$.
3. Calculate the error of h_t: $\epsilon_t = \sum_{i=1}^N p_i^t |h_t(x_i) - y_i|$.
4. Set $\beta_t = \epsilon_t / (1 - \epsilon_t)$.
5. Set the new weights vector to be
$$w_i^{t+1} = w_i^t \beta_t^{1 - |h_t(x_i) - y_i|}$$

Output the hypothesis

$$h_f(x) = \begin{cases} 1 & \text{if } \sum_{t=1}^T \left(\log \frac{1}{\beta_t}\right) h_t(x) \geq \frac{1}{2} \sum_{t=1}^T \log \frac{1}{\beta_t} \\ 0 & \text{otherwise} \end{cases}.$$

Y. Freund, R. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, Journal of Computer and System Sciences. Vol 55-1, pp. 119-139, 1997.

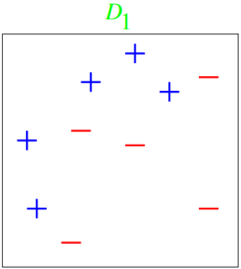
AdaBoost - a loss perspective†

- Input: N samples $\{(x^i, y^i)\}_{i=1}^N$, $x^i \in \mathbb{R}^d$, $y^i \in \{+1, -1\}, \forall i \in \{1, 2, \ldots, N\}$.
- Initialize weights $w_i^1 = 1/N, \forall i \in \{1, 2, \ldots, N\}$.
- For $t = 1, 2, \ldots, T$ do:
  - Train a weak classifier with examples weighed using current weights w_i^t by minimizing: $\epsilon_t = \sum_{i=1}^N w_i^t \mathbb{I}(h_t(x^i) \neq y^i)$.
  - Compute $\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$
  - Update weights as: $w_i^{t+1} = w_i^t e^{-\alpha_t y^i h(x^i)}$
  - Normalize $w_i^{t+1} = w_i^{t+1} / \sum_{i=1}^N w_i^{t+1}$.
- Output: Final classifier $h(x) = \text{sign}(\sum_{t=1}^T \alpha_t h_t(x))$.

†:J. Friedman, T. Hastie and R. Tibshirani. Additive logistic regression: A statistical view of Boosting, Annals of Statistics, 2000, Vol. 28, no. 2, pp. 337–407.

Bagging: AdaBoost

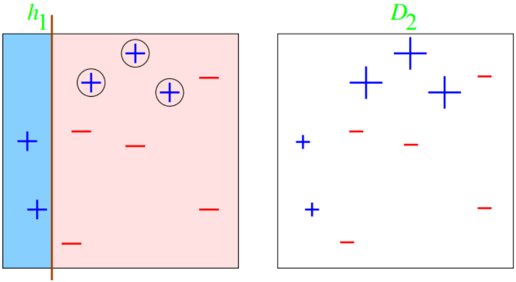
10 data points and 2 features



Example from Ameet Talwalkar's slides on AdaBoost

Bagging: AdaBoost

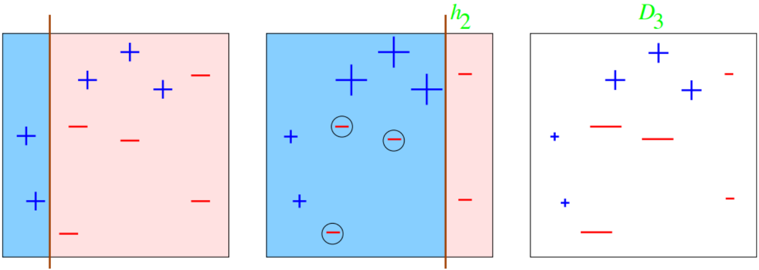
Round 1: $t = 1$



- 3 misclassified (with circles): $\epsilon_1 = 0.3 \rightarrow \alpha_1 = 0.42$.
- Weights recomputed; the 3 misclassified data points receive larger weights

Bagging: AdaBoost

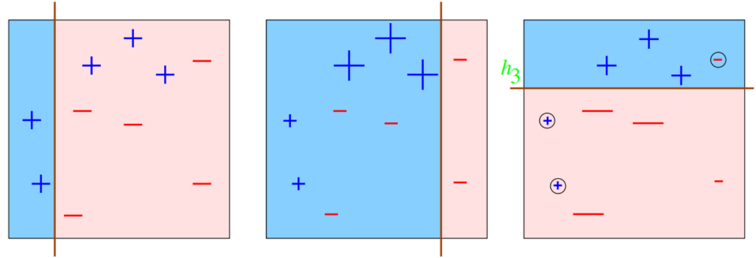
Round 2: $t = 2$



- 3 misclassified (with circles): $\epsilon_2 = 0.21 \rightarrow \alpha_2 = 0.65$.
  Note that $\epsilon_2 \neq 0.3$ as those 3 data points have weights less than $1/10$
- 3 misclassified data points get larger weights
- Data points classified correctly in both rounds have small weights

Bagging: AdaBoost

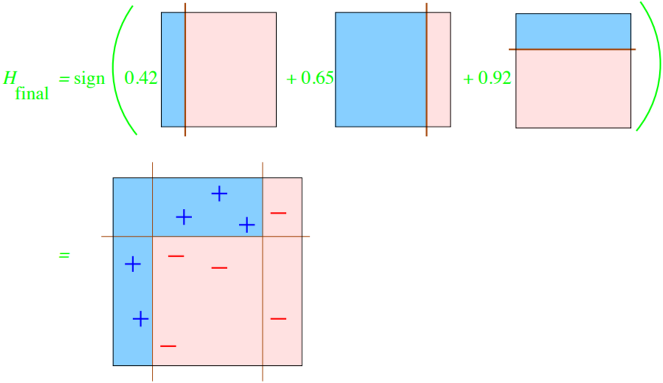
Round 3: $t = 3$



- 3 misclassified (with circles): $\epsilon_3 = 0.14 \rightarrow \alpha_3 = 0.92$.
- Previously correctly classified data points are now misclassified, hence our error is low; what's the intuition?
  - Since they have been consistently classified correctly, this round's mistake will hopefully not have a huge impact on the overall prediction

Bagging: AdaBoost

Final classifier: combining 3 classifiers



- All data points are now classified correctly!