Ensemble Model

Bùi Tiến Lên

2023

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Contents

Notation

symbol	meaning
$a, b, c, N \ldots$	scalar number
$\boldsymbol{w}, \boldsymbol{v}, \boldsymbol{x}, \boldsymbol{y} \ldots$	column vector
$\boldsymbol{X}, \boldsymbol{Y} \ldots$	matrix
$\mathbb{R}$	set of real numbers
$\mathbb{Z}$	set of integer numbers
$\mathbb{N}$	set of natural numbers
$\mathbb{R}^D$	set of vectors
$\mathcal{X}, \mathcal{Y}, \ldots$	set
$\mathcal{A}$	algorithm

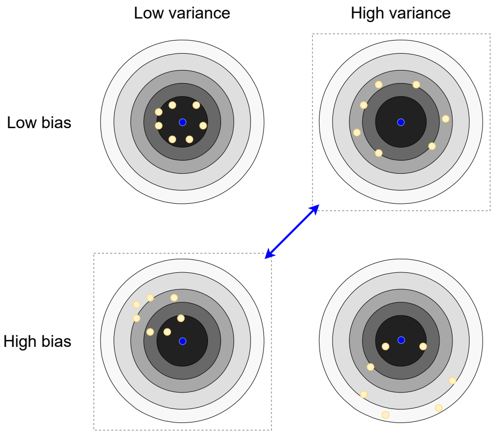
operator	meaning
$\boldsymbol{w}^\mathsf{T}$	transpose
$\boldsymbol{X}\boldsymbol{Y}$	matrix multiplication
$\boldsymbol{X}^{-1}$	inverse

Big Picture

Ensemble Model

Bias vs. Variance

- Low-bias models tend to have high variance, and vice versa.



Basics of Ensembles

Concept 1

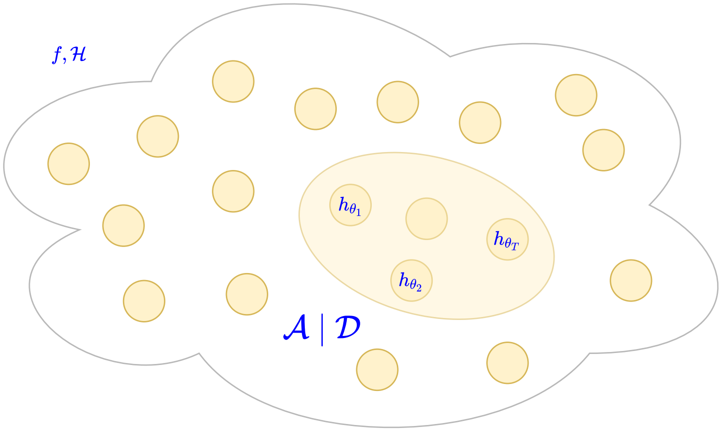
Instead of providing **one model,** an **ensemble** approach proposes **many models** to the same problem, and **combine** them

- The simplest ensemble H over models $\{h_i \in \mathcal{H}, i = 1 \ldots T\}$

$$H(\boldsymbol{x}) = \sum_{i=1}^{T} \alpha_i h_i(\boldsymbol{x}) \text{ with } \sum_{i=1}^{T} \alpha_i = 1 \quad (1)$$

- Why should this be a good idea?
  - combine models $\rightarrow$ reduce the variance $\rightarrow$ enhance expected performance.
- However, increase the performance cost

Basics of Ensembles (cont.)



Why Does it Work?

It has been shown that the expected risk of the average of a set of models is better than the average of the expected risk of these models

- Let us consider the simplest ensemble H over models h_i

$$H(\boldsymbol{x}) = \sum_{i=1}^{T} \alpha_i h_i(\boldsymbol{x}) \text{ with } \sum_{i=1}^{T} \alpha_i = 1 \tag{2}$$

- The MSE risk of h_i at $\boldsymbol{x}$ is

$$e_i(\boldsymbol{x}) = \mathbb{E}_y[(y - h_i(\boldsymbol{x}))^2] \tag{3}$$

Why Does it Work? (cont.)

- The average risk $\bar{e}(\boldsymbol{x})$ of a model is

$$\bar{e}(\boldsymbol{x}) = \sum_i \alpha_i e_i(\boldsymbol{x}) \tag{4}$$

- The average risk $e(\boldsymbol{x})$ of the ensemble is

$$e(\boldsymbol{x}) = \mathbb{E}_y[(y - H(\boldsymbol{x}))^2] \tag{5}$$

- Let us define diversity

$$d_i(\boldsymbol{x}) = (h_i(\boldsymbol{x}) - H(\boldsymbol{x}))^2 \tag{6}$$

- The average diversity is

$$\bar{d}(\boldsymbol{x}) = \sum_i \alpha_i d_i(\boldsymbol{x}) \tag{7}$$

Why Does it Work? (cont.)

- It can then be shown that

$$e(\boldsymbol{x}) = \bar{e}(\boldsymbol{x}) - \bar{d}(\boldsymbol{x}) \quad (8)$$

$$e(\boldsymbol{x}) < \bar{e}(\boldsymbol{x}) \quad (9)$$

Bagging

- Bootstrap
- Algorithm
- Random Forests

Bagging

Underlying idea

A part of the **variance** is due to the specific choice of the training data set

- Let us create many **similar** training data sets,
- For each of them, let us train a new function f_i
- The final function will be the *average* of each function outputs.

- How similar? using **bootstrap aggregating**

Bootstrap

Concept 2

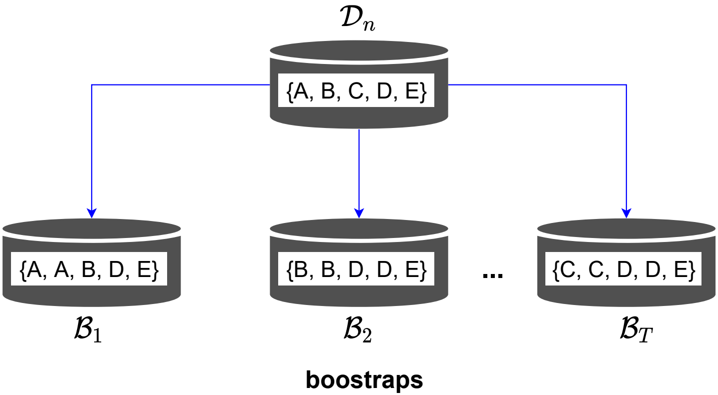
Given a data set $\mathcal{D}_n$ with n examples drawn from $p(\mathcal{Z}) = p(\mathcal{X}, \mathcal{Y})$, a bootstrap $\mathcal{B}_i, i = 1 \ldots T$ of $\mathcal{D}_n$ also contains n examples:

for $j = 1 \to n$

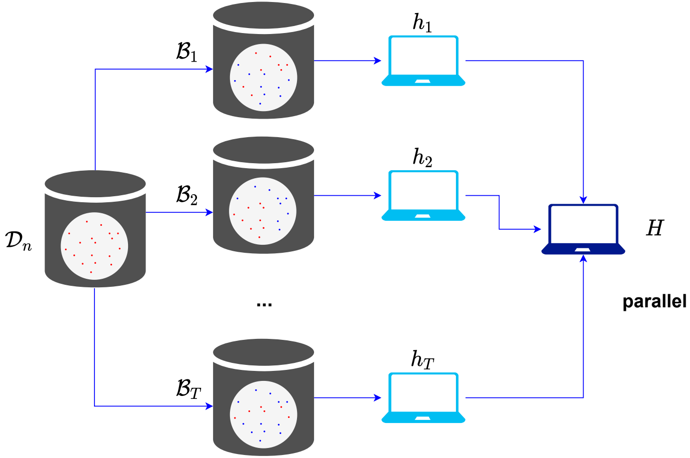
the j-th example of B_i is drawn independently with replacement from $\mathcal{D}_n$

- Some examples from $\mathcal{D}_n$ are in multiple copies in $\mathcal{B}_i$
- Some examples from $\mathcal{D}_n$ are not in $\mathcal{B}_i$
- The examples were i.i.d. drawn from $p(Z) \to$ the datasets $\mathcal{B}_i$ are as plausible as $\mathcal{D}_n$, but drawn from $\mathcal{D}_n$ instead of $p(Z)$.

Example


$\mathcal{D}_n$
{A, B, C, D, E}
{A, A, B, D, E}
{B, B, D, D, E}
...
{C, C, D, D, E}
$\mathcal{B}_1$
$\mathcal{B}_2$
$\mathcal{B}_T$
boostraps

Diagram


$\mathcal{D}_n$
$\mathcal{B}_1$
$\mathcal{B}_2$
...
$\mathcal{B}_T$
h_1
h_2
h_T
H
parallel

Algorithm

Training:

- Given a training set $\mathcal{D}_n$, create T bootstraps $\mathcal{B}_i$ of $\mathcal{D}_n$
- For each bootstrap $\mathcal{B}_i$, select

$$h_i = \arg\min_{h \in \mathcal{H}} E(h \mid \mathcal{B}_i) \tag{10}$$

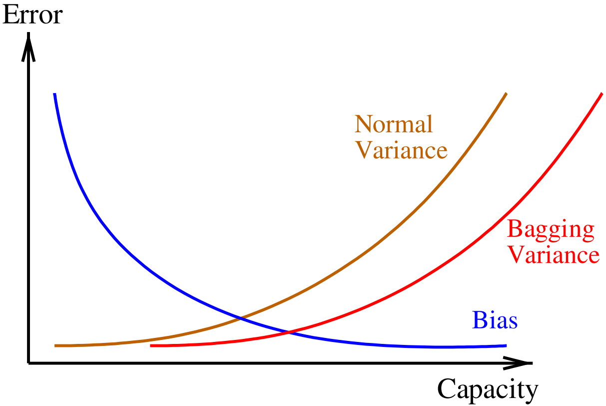
Running:

- Given an input $\boldsymbol{x}$, the corresponding output $\hat{y}$ is:

$$\hat{y} = H(\boldsymbol{x}) = \frac{1}{T} \sum_{i=1}^{T} h_i(\boldsymbol{x}) \tag{11}$$

Bias + Variance

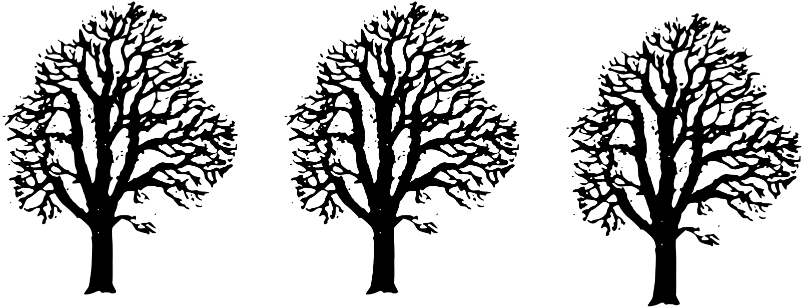
- **Analysis**: if generalization error is decomposed into **bias** and **variance** terms then **bagging reduces variance**.



Random Forests

Concept 3

A **random forest** is an ensemble of decision trees.



Random Forests (cont.)

Each decision tree h_i is trained as follows:

- Create a **bootstrap** of the training set
- Select a subset $m \ll d$ input variables as **potential** split nodes (m is constant over all trees)
- No pruning of the trees

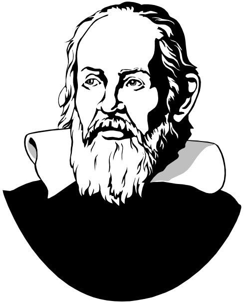
A decision is taken by **voting** amongst the trees

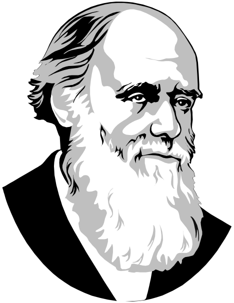
- Somehow, m controls the capacity.

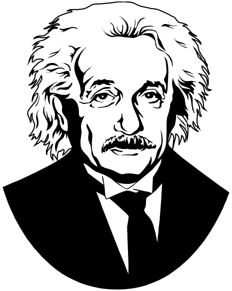
Boosting

- AdaBoost
- Face Detection

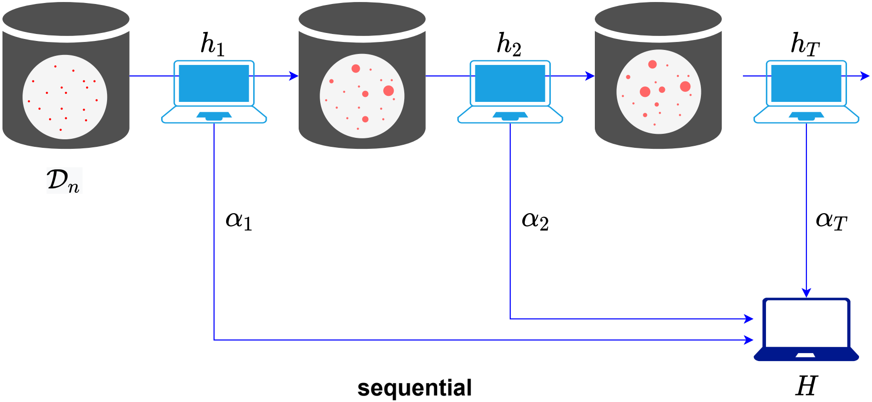
Big Picture







Diagram



Weak vs. Strong Learning Model

Concept 4

A learning model is **strong** iff every hypothesis h has low error

Concept 5

A learning model is **weak** iff every hypothesis h has high error

Examples of weak classifiers:

- Simple decision trees such as **stumps**
- Simple neural networks such as **perceptrons**
- Haar-like features

Boosting

Boosting involves three elements:

- A loss function to be optimized $\ell(.,.)$
- A set of weak learners $\{h_t(x)\}$
- An additive model $H(x)$ to add weak learners to minimize the loss function

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \tag{12}$$

Loss function

- Square loss
$$\ell(\hat{y}, y) = (\hat{y} - y)^2 \tag{13}$$
- Absolute loss
$$\ell(\hat{y}, y) = |\hat{y} - y| \tag{14}$$
- Huber loss
$$\ell(\hat{y}, y) = \begin{cases} \frac{1}{2}(\hat{y} - y)^2 & (\hat{y} - y) \leq \delta \\ \delta(|\hat{y} - y| - \delta/2) & (\hat{y} - y) > \delta \end{cases} \tag{15}$$
- Exponential loss
$$\ell(\hat{y}, y) = e^{-\hat{y}\, y} \tag{16}$$

Algorithm

- **Initialize** $H_0 \leftarrow \emptyset$ or α_0
- At each time step t,
  - **Choose** h_t given the performance obtained by previous H_{t-1}.
  - **Train** a new weak classifier
  - **Find** the new weight α_t by minimizing the loss function

$$H_t \leftarrow H_{t-1} + \alpha_t h_t \tag{17}$$

AdaBoost

Concept 6

AdaBoost, short for Adaptive Boosting, is the most popular algorithm in the family of **boosting** algorithms

- **Simplest framework**: binary classification $H(x)$
- **Simplest requirement**: each weak classifier $y = h_t(x)$, $y \in \{-1, +1\}$ should perform better than chance
- **Loss function**:

$$\ell((H(x), y)) = e^{-yH(x)} \tag{18}$$

Algorithm

Inputs: training data $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_N, y_N)\}$ and a set of weak binary classifiers $\{h_i \in \mathcal{H}\}$

Initialize the weights' distribution of training data

$$(w_1^{(1)}, w_2^{(1)}, \cdots, w_N^{(1)}) = \left(\frac{1}{N}, \frac{1}{N}, \cdots, \frac{1}{N}\right) \tag{19}$$

Iterate over $t = 1, 2, \ldots, T$, use training data with current weights' distribution

1. **Find** a weak classifier $h_t(\boldsymbol{x})$ that that minimizes the error rate e_t of over the training data

$$e_t = P(h_t(\boldsymbol{x}_i) \neq y_i) = \sum_{i=1}^{N} w_i^{(t)} \mathbb{I}(h_t(\boldsymbol{x}_i) \neq y_i) \tag{20}$$

Algorithm (cont.)

2. **Compute** the weight of classifier $h_t(\boldsymbol{x})$

$$\alpha_t = \frac{1}{2}\log\frac{1-e_t}{e_t} \tag{21}$$

3. **Update** the weights' distribution of training data

$$w_i^{(t+1)} = w_i^{(t)}\exp(-\alpha_t y_i h_t(\boldsymbol{x}_i)) \tag{22}$$

4. **Normalize** the weights of data points

$$w_i = \frac{w_i}{\sum_i w_i} \tag{23}$$

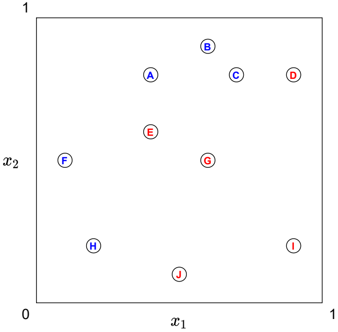
Ensemble T weak classifiers

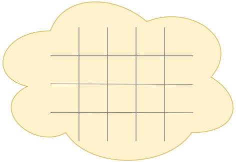
$$\text{sign}\left[H(\boldsymbol{x})\right] = \text{sign}\left[\sum_{t=1}^{T}\alpha_t h_t(\boldsymbol{x})\right] \tag{24}$$

Example

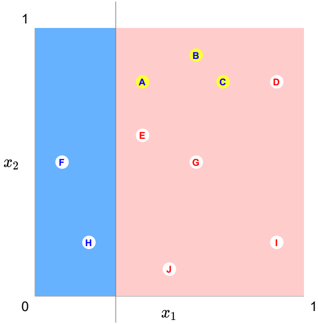
- Given a training data set $\mathcal{D} = \{$A, B, C, D, E, F, G, H, I, J$\}$, find a strong classifier from weak classifiers (vertical or horizontal lines)

#	x_1	x_2	label
A	0.4	0.8	1
B	0.6	0.9	1
C	0.7	0.8	1
D	0.9	0.8	-1
E	0.4	0.6	-1
F	0.1	0.5	1
G	0.6	0.5	-1
H	0.2	0.2	1
I	0.9	0.2	-1
J	0.5	0.1	-1



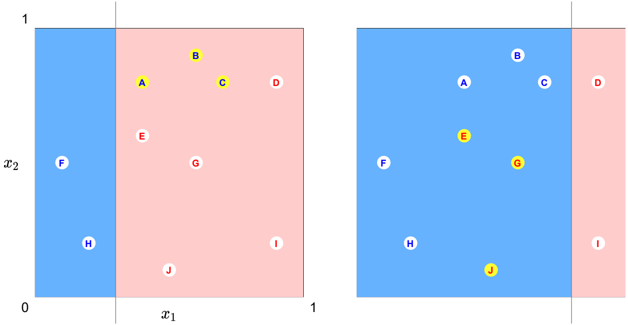


Round 1



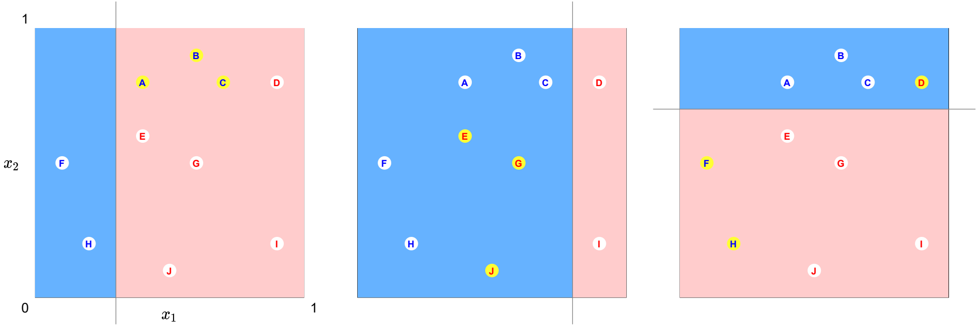
$\epsilon_1 = 0.30, \alpha_1 = 0.42$

Round 2



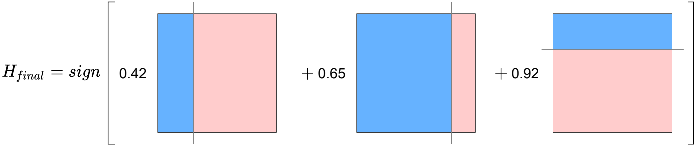
$\epsilon_2 = 0.21, \alpha_2 = 0.65$

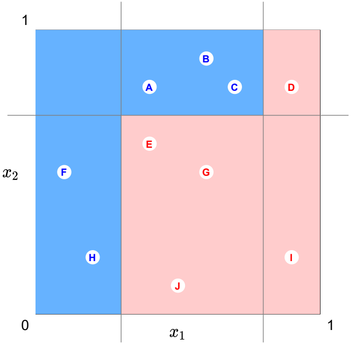
Round 3



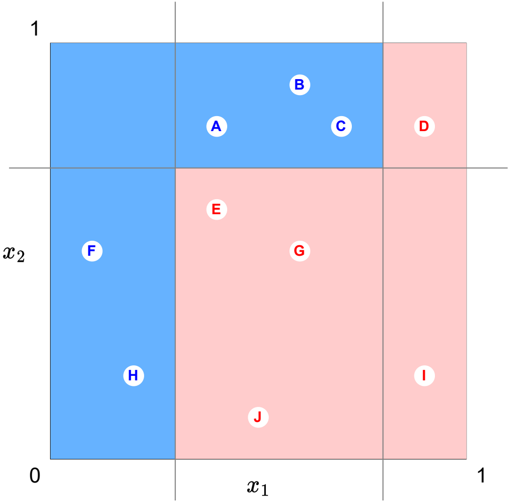
$\epsilon_3 = 0.14, \alpha_3 = 0.92$

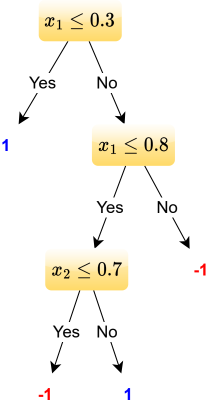
The combined classifier





Tree based classifier





Analysis

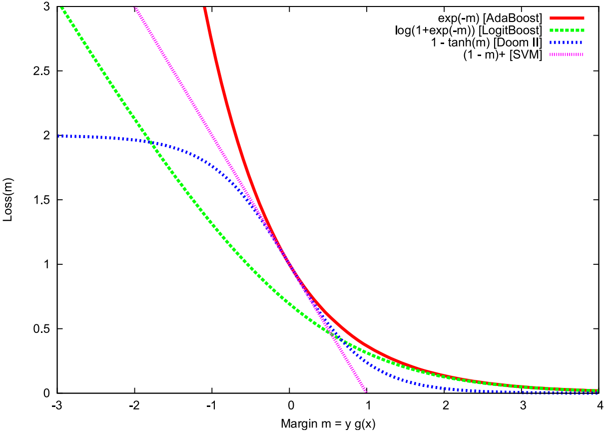
- Selection of α_t comes from minimizing

$$\arg\min_{\alpha_t} \sum_{i=1}^{N} \exp\left(-y_i \left[H_{t-1}(\boldsymbol{x}_i) + \alpha_t h_t(\boldsymbol{x}_i)\right]\right) \tag{25}$$

- If each weak classifier is always better than chance, then AdaBoost can be proven to **converge to 0 training error**
- Even after training error is 0, generalization error continues to improve: the **margin** continues to grow
- **Sampling** can often be replaced by **weighting**

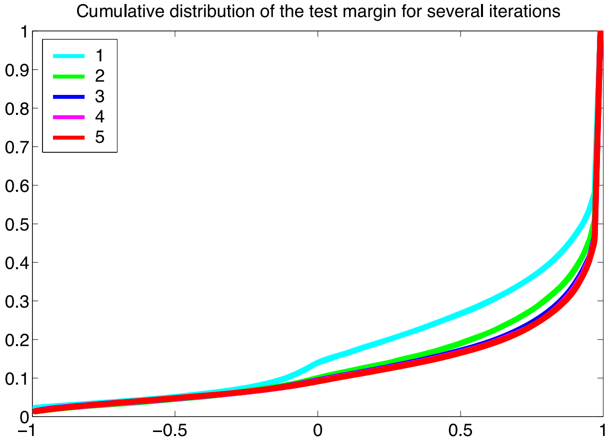
Cost Functions

- Comparison of various cost functions related to AdaBoost



Margin

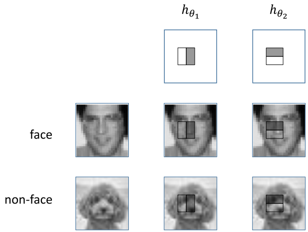
- The AdaBoost margin is defined as the distribution of $y \cdot h(x)$



Face Detection

Face detection framework was proposed in 2001 by Paul Viola and Michael Jones using AdaBoost

- Some hypotheses h_θ



- Haar-like features for each hypothesis

$$h_\theta = \sum_{(x,y)\in \text{dark area}} image(x, y) - \sum_{(x,y)\in \text{white area}} image(x, y) \tag{26}$$

Gradient Boosting

What is Gradient Boosting

$$\textit{\textbf{Gradient Boosting}} = \textit{\textbf{Gradient Descent}} + \textit{\textbf{Boosting}}$$

- In Adaboost, “losses” are identified by high-weight data points
- In Gradient Boosting, “losses” are identified by gradients
- Gradient Boosting can be applied for different problems

$$\text{Regression} \rightarrow \text{Classification} \rightarrow \text{Ranking}$$

Algorithm

Input: training set $\{(x_i, y_i)\}_{i=1}^{n}$ and a differentiable loss function $\ell(y, H(x))$, number of iterations M.

1. Initialize model with a constant value:

$$H_0(x) = \gamma_0 = \arg\min_{\gamma} \sum_{i=1}^{n} \ell(y_i, \gamma). \tag{27}$$

2. For $m = 1$ to M:

    2.1 Compute so-called pseudo-residuals:

$$r_{im} = -\left[\frac{\partial \ell(y_i, H(x_i))}{\partial H(x_i)}\right]_{H(x)=H_{m-1}(x)}, \quad \text{for } i = 1, \ldots, m \tag{28}$$

Algorithm (cont.)

2.2 Fit a base learner (or weak learner, e.g. tree) closed under scaling $h_m(x)$ to pseudo-residuals, i.e. train it using the training set $\{(x_i, r_{im})\}_{i=1}^n$.

2.3 Compute multiplier by solving the following one-dimensional optimization problem:

$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} \ell\left(y_i, H_{m-1}(x_i) + \gamma h_m(x_i)\right). \tag{29}$$

2.4 Update the model:

$$H_m(x) = H_{m-1}(x) + \gamma_m h_m(x) \tag{30}$$

3. Output $H_M(x)$

Important points to remember

- Bagging is predominantly a variance-reduction technique, while boosting is primarily a bias-reduction technique.
- This explains why bagging is often used in combination with high-variance models such as tree models, whereas boosting is typically used with high-bias models such as linear classifiers.

References

Goodfellow, I., Bengio, Y., and Courville, A. (2016).
Deep learning.
MIT press.

Lê, B. and Tô, V. (2014).
Cở sở trí tuệ nhân tạo.
Nhà xuất bản Khoa học và Kỹ thuật.

Russell, S. and Norvig, P. (2021).
Artificial intelligence: a modern approach.
Pearson Education Limited.