

# Boosting

Statistical Machine Learning (ENGG\*6600\*02)

School of Engineering,  
University of Guelph, ON, Canada

Course Instructor: Benyamin Ghogh  
Summer 2023

## Definition

# Definition

- Boosting is a meta algorithm which can be used with any model (classifier, regression, etc).
- For binary classification, for example, if we use boosting with a classifier even slightly better than flipping a coin, we will have a strong classifier (we will explain the reason later). Thus, we can say boosting makes the estimation or classification very strong. In other words, boosting addresses the question whether a strong classifier can be obtained from a set of weak classifiers [1, 2].

# Definition

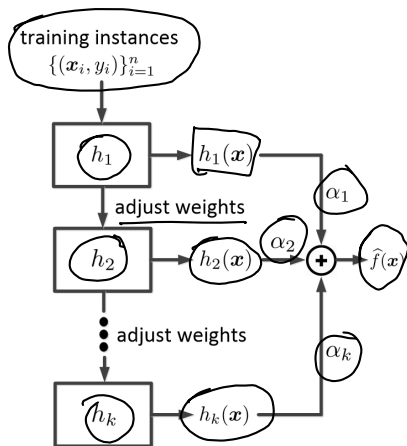
- The idea of boosting is to learn  $k$  models in a hierarchy where every model gives more attention (larger weight) to the instances misclassified (or estimated very badly) by the previous model.
- Finally, the overall estimation or classification is a weighted summation (average) of the  $k$  estimations. For an instance  $x$ , we have:

$$\hat{f}(x) = \sum_{j=1}^k \alpha_j h_j(x). \quad (1)$$

If the model is classifier, we should probably use sign function:

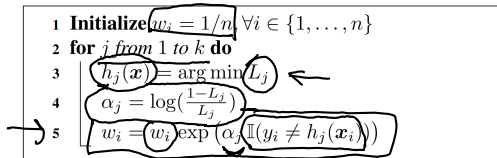
$$\rightarrow \hat{f}(x) = \text{sign}\left(\sum_{j=1}^k \alpha_j h_j(x)\right), \quad (2)$$

which is equivalent to majority voting among the trained classifiers.



# Definition

- Different methods have been proposed for boosting, one of the most well-known ones is AdaBoost (Adaptive Boosting) (1996) [3].
- The algorithm of AdaBoost for binary classification is shown below.



**Algorithm :** The AdaBoost Algorithm

- In this algorithm,  $L_j$  is the cost function minimized in the  $j$ -th model  $h_j$ , the  $\mathbb{I}(\cdot)$  is the indicator function which is one and zero if its condition is and is not satisfied, respectively, and  $w_i$  is the weight associated to the  $i$ -th instance for weighting it as the input to the next layer of boosting.
- Note that the cost in AdaBoost is:

$$L_j = \frac{\sum_{i=1}^n w_i \mathbb{I}(y_i \neq h_j(x_i))}{\sum_{i=1}^n w_i}, \quad (3)$$

which makes sense because it gets larger if the observations of more instances are estimated incorrectly.

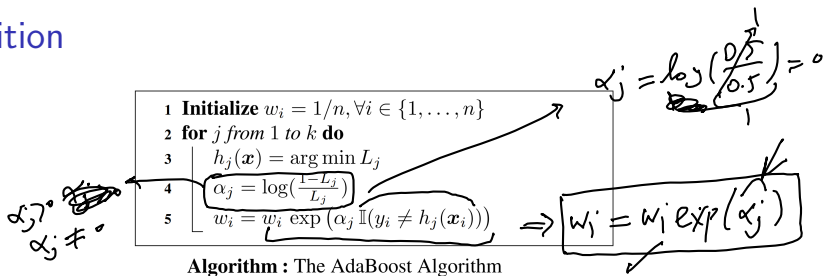
# Definition

```
1 Initialize  $w_i = 1/n, \forall i \in \{1, \dots, n\}$ 
2 for  $j$  from 1 to  $k$  do
3    $h_j(\mathbf{x}) = \arg \min L_j$ 
4    $\alpha_j = \log(\frac{1-L_j}{L_j})$ 
5    $w_i = w_i \exp(\alpha_j \mathbb{I}(y_i \neq h_j(\mathbf{x}_i)))$  ★
```

**Algorithm :** The AdaBoost Algorithm

- Here, we can have several cases which help us understand the interpretation of the AdaBoost algorithm:
- If an instance is correctly classified, the  $\mathbb{I}(y_i \neq h_j(\mathbf{x}_i))$  is zero and thus the  $w_i$  will be still  $w_i$  without any change. This makes sense because the correctly classified instance should not gain a significant weight in the next layer of boosting.

# Definition



- If an instance is misclassified, the  $\mathbb{I}(y_i \neq h_j(x_i))$  is one. In this case, we can have two sub-cases:
  - ★ ▶ If the classifier which classified that instance was a bad classifier, its cost would be like flipping a coin, i.e.,  $L_j = 0.5$ . Therefore, we will have  $\alpha_j = \log(1) = 0$  and again the  $w_i$  will still be  $w_i$  without any change. This makes sense because we cannot trust the bad classifier whether the instance is correctly or incorrectly classified and thus we should not make any decision based on that.
  - ★ ▶ If the classifier which classified that instance was a good classifier, then we have  $L_j \neq 0.5$ , and as we also have  $\mathbb{I}(y_i \neq h_j(x_i)) = 1$ , the weight will change as  $w_i := w_i \exp(\alpha_j)$ . This also is intuitive because the previous model in the boosting was a good classifier and we can trust it and that good classifier could not classify the instance correctly. Therefore, we should notice that instance more in the next model in the boosting hierarchy.

## **Theory Based on Additive Models**



# Theory Based on Additive Models

- Additive models [4] can be used to explain why boosting works [5, 6]. In an additive model, we map the data as  $\mathbf{x} \mapsto \phi_j(\mathbf{x}), \forall j \in \{1, \dots, k\}$  and then add them using some weights  $\beta_j$ 's:

$$\star \quad \phi(\mathbf{x}) = \sum_{j=1}^k \beta_j \phi_j(\mathbf{x}). \quad (4)$$

- A well-known example of the additive model is Radial Basis Function (RBF) neural network (here with  $k$  hidden nodes) which uses Gaussian mappings [7, 8].
- Now, consider a cost function for an instance as:

$$\star \quad L(y, h(\mathbf{x})) := \exp(-y h(\mathbf{x})), \quad (5)$$

where  $y$  is the observation or label for  $\mathbf{x}$  and  $h(\mathbf{x})$  is the model's estimation of  $y$ . This cost is intuitive because when the instance is misclassified, the signs of  $y$  and  $h(\mathbf{x})$  will be different (so  $y h(\mathbf{x}) < 0$ ) and the cost will be large, while in case of correct classification, the signs are similar (so  $y h(\mathbf{x}) > 0$ ) and the cost is small.

- If we add up the cost over the  $n$  training instances, we have:

$$\star \quad L_t(y, h(\mathbf{x})) := \sum_{i=1}^n \exp(-y_i h(\mathbf{x}_i)), \quad (6)$$

where  $L_t$  denotes the total cost.

# Theory Based on Additive Models

- In Eq. (4),  $\phi(\mathbf{x}) = \sum_{j=1}^k \beta_j \phi_j(\mathbf{x})$ , if we rename the mapping to  $\underline{h(\mathbf{x})}$ , which is the model used in boosting, we will have:

$$\star \quad \overbrace{h(\mathbf{x})} = \sum_{j=1}^k \overbrace{\beta_j h_j(\mathbf{x})}. \quad (7)$$

- We can write this expression as a **forward stage-wise additive model** [5, 6] in which we work with the models one by one where we add up the previously worked models:

$$\star \quad \underbrace{f_{q-1}(\mathbf{x})}_{\star} = \sum_{j=1}^{q-1} \beta_j h_j(\mathbf{x}), \quad (8)$$

$$f_q(\mathbf{x}) = \underbrace{f_{q-1}(\mathbf{x})}_{\star} + \boxed{\beta_q h_q(\mathbf{x})}, \quad \underline{q \leq k}, \quad (9)$$

where  $\boxed{h(\mathbf{x})} = \boxed{f_k(\mathbf{x})}$ .

- Therefore, minimizing the cost, i.e., Eq. (6),  $\boxed{L_t(y, h(\mathbf{x})) := \sum_{i=1}^n \exp(-y_i h(\mathbf{x}_i))}$ , for the  $j$ -th model in the additive manner is:

$$\min_{\beta_j, h_j} \sum_{i=1}^n \exp(-y_i [f_{j-1}(\mathbf{x}_i) + \beta_j h_j(\mathbf{x}_i)]) = \min_{\beta_j, h_j} \sum_{i=1}^n \underbrace{\exp(-y_i f_{j-1}(\mathbf{x}_i))}_{\star} \underbrace{(\exp(-y_i \beta_j h_j(\mathbf{x}_i)))}_{\star}.$$

$e^{a+b} = e^a e^b$

# Theory Based on Additive Models

- We had:

$$\star \min_{\beta_j, h_j} \sum_{i=1}^n \exp(-y_i [f_{j-1}(\mathbf{x}_i) + \beta_j h_j(\mathbf{x}_i)]) = \min_{\beta_j, h_j} \sum_{i=1}^n \exp(-y_i f_{j-1}(\mathbf{x}_i)) \exp(-y_i \beta_j h_j(\mathbf{x}_i)).$$

- The first term is a constant with respect to  $\beta_j$  and  $h_j$  so we name it by  $w_i$ :

$$w_i := \exp(-y_i f_{j-1}(\mathbf{x}_i)). \quad (10)$$

Thus:

$$\star \min_{\beta_j, h_j} \sum_{i=1}^n w_i \exp(-y_i \beta_j h_j(\mathbf{x}_i)).$$

- As in binary AdaBoost, we have  $\pm 1$  for  $y_i$  and  $h_j$ , we can say:

$$\begin{aligned} \star \min_{\beta_j, h_j} & \exp(-\beta_j) \sum_{i=1}^n w_i \mathbb{I}(y_i = h_j(\mathbf{x}_i)) + \exp(\beta_j) \sum_{i=1}^n w_i \mathbb{I}(y_i \neq h_j(\mathbf{x}_i)) \\ \stackrel{(a)}{=} & \min_{\beta_j, h_j} \underbrace{\exp(-\beta_j) \sum_{i=1}^n w_i - \exp(-\beta_j) \sum_{i=1}^n w_i \mathbb{I}(y_i \neq h_j(\mathbf{x}_i))}_{\text{first term}} + \underbrace{\exp(\beta_j) \sum_{i=1}^n w_i \mathbb{I}(y_i \neq h_j(\mathbf{x}_i))}_{\text{second term}}, \end{aligned}$$

where (a) is because:

$$\sum_{i=1}^n w_i \mathbb{I}(y_i = h_j(\mathbf{x}_i)) = \sum_{i=1}^n w_i - \sum_{i=1}^n w_i \mathbb{I}(y_i \neq h_j(\mathbf{x}_i)).$$

# Theory Based on Additive Models

- We had:

$$\star \min_{\beta_j, h_j} \exp(-\beta_j) \sum_{i=1}^n w_i \ominus \exp(\ominus \beta_j) \sum_{i=1}^n w_i \mathbb{I}(y_i \neq h_j(\mathbf{x}_i)) + \exp(\beta_j) \sum_{i=1}^n w_i \mathbb{I}(y_i \neq h_j(\mathbf{x}_i)).$$

- For the sake of minimization, we take the derivative:

$$\star \frac{\partial L_t}{\partial \beta_j} = \ominus \exp(-\beta_j) \sum_{i=1}^n w_i \oplus \exp(-\beta_j) \sum_{i=1}^n w_i \mathbb{I}(y_i \neq h_j(\mathbf{x}_i)) \\ + \exp(\beta_j) \sum_{i=1}^n w_i \mathbb{I}(y_i \neq h_j(\mathbf{x}_i)) \stackrel{\text{set}}{=} 0,$$

which gives:

$$\begin{aligned} \Rightarrow & (\exp(-\beta_j) + \exp(\beta_j)) \times \frac{\sum_{i=1}^n w_i \mathbb{I}(y_i \neq h_j(\mathbf{x}_i))}{\sum_{i=1}^n w_i} = \exp(-\beta_j) \\ \stackrel{(3)}{\Rightarrow} & (\exp(-\beta_j) + \exp(\beta_j)) L_j = \exp(-\beta_j) \\ \Rightarrow & L_j = \frac{\exp(-\beta_j)}{\exp(-\beta_j) + \exp(\beta_j)} \Rightarrow \exp(2\beta_j) = \frac{1 - L_j}{L_j} \Rightarrow 2\beta = \log\left(\frac{1 - L_j}{L_j}\right) \\ \stackrel{(a)}{\Rightarrow} & \boxed{\alpha_j = 2\beta_j}, \star \end{aligned} \tag{11}$$

where (a) is because of the formula of  $\alpha_j$  in the Algorithm.

# Theory Based on Additive Models

- According to Eqs. (8), (9), and (10),

$$\left\{ \begin{array}{l} \rightarrow f_{q-1}(\mathbf{x}) = \sum_{j=1}^{q-1} \beta_j h_j(\mathbf{x}), \\ \rightarrow f_q(\mathbf{x}) = f_{q-1}(\mathbf{x}) + \beta_q h_q(\mathbf{x}), \quad q \leq k, \\ \rightarrow w_i := \exp(-y_i f_{q-1}(\mathbf{x}_i)). \end{array} \right.$$

we have:

$$w_i := w_i \exp(-y_i \beta_j h_j(\mathbf{x}_i)). \quad (12)$$

??  $\exp(\beta_j (2) I(y_i \neq h_j(\mathbf{x}_i)))$

As we have  $y_i h_j(\mathbf{x}_i) = \pm 1$ , we can say:

$$* \quad -y_i h_j(\mathbf{x}_i) = 2 \mathbb{I}(y_i \neq h_j(\mathbf{x}_i)) - 1 \quad \begin{array}{l} y_i = h_j(\mathbf{x}_i) \Rightarrow -1 \\ y_i \neq h_j(\mathbf{x}_i) \Rightarrow 1 \end{array} \quad (13)$$

According Eqs. (11),  $\alpha_j = 2\beta_j$ , (12), and (13), we have:

$$\rightarrow w_i := w_i \exp(\alpha_j \mathbb{I}(y_i \neq h(\mathbf{x}_i))) \exp(-\beta_j), \quad e^{a+b} = e^a e^b \quad (14)$$

which is equivalent to the formula of  $w_i$  in the Algorithm with a factor of  $\exp(-\beta_j)$ . This factor does not have impact on whether the instance is correctly classified or not.

**Upper Bound on the  
Generalization Error of  
Boosting**

# Upper Bound on the Generalization Error of Boosting

- There is an upper bound on the generalization error of boosting [9]. In binary boosting, we have  $\pm 1$  for  $y_i$  and also the sign of  $\hat{f}(\mathbf{x}_i)$  is important; therefore,  $y_i \hat{f}(\mathbf{x}_i) < 0$  means that we have error for estimating the  $i$ -th instance. Thus, for an error, we have:

$$\star \quad y_i \hat{f}(\mathbf{x}_i) \leq -\theta, \quad (15)$$

for a  $\theta > 0$ . Recall the Eq. (1):

$$\star \quad \hat{f}(\mathbf{x}) = \sum_{j=1}^k \alpha_j h_j(\mathbf{x}).$$

- We can normalize this equation because the sign of it is important:

$$\star \quad \hat{f}(\mathbf{x}_i) = \frac{\sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i)}{\sum_{j=1}^k \alpha_j}. \quad (16)$$

- According to Eqs. (15) and (16), we have:

$$\begin{aligned} \star \quad y_i \hat{f}(\mathbf{x}_i) \leq -\theta &\iff y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i) \leq -\theta \sum_{j=1}^k \alpha_j \quad \star \\ &\iff -y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i) + \theta \sum_{j=1}^k \alpha_j \geq 0 \iff \exp\left(-y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i) + \theta \sum_{j=1}^k \alpha_j\right) \geq 1. \end{aligned}$$

# Upper Bound on the Generalization Error of Boosting

- We found:

$$\star \quad y_i \hat{f}(\mathbf{x}_i) \leq \theta \iff \exp\left(-y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i) + \theta \sum_{j=1}^k \alpha_j\right) \geq 1.$$

- Therefore, in terms of probability, we have:

$$\star \quad \mathbb{P}(y_i \hat{f}(\mathbf{x}_i) \leq \theta) = \mathbb{P}\left(\exp\left(-y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i) + \theta \sum_{j=1}^k \alpha_j\right) \geq 1\right). \quad (17)$$

- According to the Markov's inequality which is (for  $a > 0$  and a random variable  $X$ ):

$$\mathbb{P}(X \geq a) \leq \frac{\mathbb{E}(X)}{a}, \rightarrow \mathbb{P}(X \geq 1) \leq \mathbb{E}(X) \quad (18)$$

and Eq. (17), we have (take  $a = 1$  and the exponential term as  $X$  in Markov's inequality):

$$\begin{aligned} \mathbb{P}(y_i \hat{f}(\mathbf{x}_i) \leq \theta) &\stackrel{(17)}{=} \mathbb{P}\left(\exp\left(-y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i) + \theta \sum_{j=1}^k \alpha_j\right) \geq 1\right) \\ &\stackrel{(18)}{\leq} \mathbb{E}\left(\exp\left(-y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i) + \theta \sum_{j=1}^k \alpha_j\right)\right) \end{aligned}$$



# Upper Bound on the Generalization Error of Boosting

- Continuing:

$$\begin{aligned}
 & \star \left\{ \begin{aligned} & \mathbb{P}(y_i \hat{f}(\mathbf{x}_i) \leq \theta) = \mathbb{P}\left(\exp\left(-y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i) + \theta \sum_{j=1}^k \alpha_j\right) \geq 1\right) \star \\ & \stackrel{(18)}{\leq} \mathbb{E}\left(\exp\left(-y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i) + \theta \sum_{j=1}^k \alpha_j\right)\right) \end{aligned} \right. \quad e^{a+b} = e^a e^b \\
 & \stackrel{(a)}{=} \exp\left(\theta \sum_{j=1}^k \alpha_j\right) \mathbb{E}\left(\exp\left(-y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i)\right)\right) \\
 & \star \stackrel{(b)}{=} \frac{1}{n} \exp\left(\theta \sum_{j=1}^k \alpha_j\right) \sum_{i=1}^n \exp\left(-y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i)\right), \quad (19)
 \end{aligned}$$

where (a) is because the expectation is with respect to the data, i.e.,  $\mathbf{x}_i$  and  $y_i$  and (b) is according to the definition of expectation.

# Upper Bound on the Generalization Error of Boosting

- Recall the formula of  $w_i$  in the Algorithm:

$$\star \quad w_i^{(j+1)} = w_i^{(j)} \exp \left( \underbrace{\alpha_j \mathbb{I}(y_i \neq h_j(\mathbf{x}_i))}_{\text{handwritten}} \right), \quad \star$$

where  $j$  denotes the iteration index. It can be restated as:

$$w_i^{(j+1)} = w_i^{(j)} \exp \left( \underbrace{-y_i \alpha_j h_j(\mathbf{x}_i)}_{\text{handwritten}} \right),$$

because  $y_i = \pm 1$  and  $h_j(\mathbf{x}_i) = \pm 1$ .

- It is not harmful to AdaBoost if we use the normalized weights:

$$w_i^{(j+1)} = \frac{w_i^{(j)} \exp(-y_i \alpha_j h_j(\mathbf{x}_i))}{\underbrace{(z_j)}_{\text{handwritten}}}, \quad (20)$$

where:

$$\star \quad \underbrace{(z_j) := \sum_{i=1}^n w_i^{(j)} \exp(-y_i \alpha_j h_j(\mathbf{x}_i))}_{\text{handwritten}}. \quad (21)$$

# Upper Bound on the Generalization Error of Boosting

- We had:

$$\star w_i^{(j+1)} = \frac{w_i^{(j)} \exp(-y_i \alpha_j h_j(\mathbf{x}_i))}{z_j}, \star$$

- Considering that  $w_i^{(1)} = 1/n$ , we can have recursive expression for the weights:

$$\begin{aligned} \star w_i^{(k+1)} &= \frac{w_i^{(k)} \exp(-y_i \alpha_k h_k(\mathbf{x}_i))}{z_k} \\ &= w_i^{(1)} \times \frac{1}{z_k \times \dots \times z_1} \times \exp(-y_i \alpha_k h_k(\mathbf{x}_i)) \times \dots \times \exp(-y_i \alpha_1 h_1(\mathbf{x}_i)) \\ &= \frac{1}{n} \times \frac{1}{\prod_{j=1}^k z_j} \times \prod_{j=1}^k \exp(-y_i \alpha_j h_j(\mathbf{x}_i)) = \frac{1}{n} \times \frac{1}{\prod_{j=1}^k z_j} \times \exp\left(-y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i)\right) \star \\ \star \Rightarrow \underbrace{\frac{1}{n} \exp\left(-y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i)\right)}_{\star} &= \underbrace{\left(\prod_{j=1}^k z_j\right)}_{\star} \sum_{i=1}^n w_i^{(k+1)}. \end{aligned}$$

$e^a e^b = e^{a+b} \quad (22)$

# Upper Bound on the Generalization Error of Boosting

- We found (Eq. (22)):

$$\frac{1}{n} \exp \left( - y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i) \right) = \left( \prod_{j=1}^k z_j \right) \sum_{i=1}^n w_i^{(k+1)}.$$

- We continue the Eq. (19):

$$\begin{aligned} \mathbb{P}(y_i \hat{f}(\mathbf{x}_i) \leq \theta) &\leq \left( \frac{1}{n} \exp \left( \theta \sum_{j=1}^k \alpha_j \right) \right) \sum_{i=1}^n \exp \left( - y_i \sum_{j=1}^k \alpha_j h_j(\mathbf{x}_i) \right) \\ &\stackrel{(22)}{=} \exp \left( \theta \sum_{j=1}^k \alpha_j \right) \left( \prod_{j=1}^k z_j \right) \sum_{i=1}^n w_i^{(k+1)}. \end{aligned}$$

# Upper Bound on the Generalization Error of Boosting

- We found:

$$\star \quad \mathbb{P}(y_i \hat{f}(\mathbf{x}_i) \leq \theta) \leq \exp\left(\theta \sum_{j=1}^k \alpha_j\right) \left(\prod_{j=1}^k z_j\right) \sum_{i=1}^n w_i^{(k+1)}.$$

- According to Eqs. (20) and (21),

$$(20) \rightarrow w_i^{(j+1)} = \frac{w_i^{(j)} \exp(-y_i \alpha_j h_j(\mathbf{x}_i))}{z_j},$$

$$(21) \rightarrow z_j := \sum_{i=1}^n w_i^{(j)} \exp(-y_i \alpha_j h_j(\mathbf{x}_i)),$$

we have:

$$\left(\sum_{i=1}^n w_i^{(j+1)}\right) = \frac{\sum_{i=1}^n w_i^{(j)} \exp(-y_i \alpha_j h_j(\mathbf{x}_i))}{\sum_{i=1}^n w_i^{(j)} \exp(-y_i \alpha_j h_j(\mathbf{x}_i))} = 1.$$

Therefore:

therefore

$$\star \quad \mathbb{P}(y_i \hat{f}(\mathbf{x}_i) \leq \theta) \leq \exp\left(\theta \sum_{j=1}^k \alpha_j\right) \left(\prod_{j=1}^k z_j\right). \quad (23)$$

# Upper Bound on the Generalization Error of Boosting

- On the other hand, according to Eq. (21),  $z_j := \sum_{i=1}^n w_i^{(j)} \exp(-y_i \alpha_j h_j(\mathbf{x}_i))$ , we have:

$$\begin{aligned}
 \star \quad z_j &= \sum_{i=1}^n w_i^{(j)} \exp(-y_i \alpha_j h_j(\mathbf{x}_i)) \\
 &= \sum_{i=1}^n w_i^{(j)} \exp(-\alpha_j) \mathbb{I}(y_i = h_j(\mathbf{x}_i)) + \sum_{i=1}^n w_i^{(j)} \exp(\alpha_j) \mathbb{I}(y_i \neq h_j(\mathbf{x}_i)) \\
 &= \exp(-\alpha_j) \sum_{i=1}^n w_i^{(j)} \mathbb{I}(y_i = h_j(\mathbf{x}_i)) + \exp(\alpha_j) \sum_{i=1}^n w_i^{(j)} \mathbb{I}(y_i \neq h_j(\mathbf{x}_i)). \quad \star \quad (24)
 \end{aligned}$$

- Recall Eq. (20) for  $w_i^{(j+1)}$ , i.e.,  $w_i^{(j+1)} = \frac{w_i^{(j)} \exp(-y_i \alpha_j h_j(\mathbf{x}_i))}{z_j}$ . This is in the range  $[0, 1]$  and its summation over error cases can be considered as the probability of error:

$$\sum_{i=1}^n w_i^{(j)} \mathbb{I}(y_i \neq h_j(\mathbf{x}_i)) = \mathbb{P}(y_i \neq h_j(\mathbf{x}_i)) \stackrel{(a)}{=} L_j \quad (25)$$

where (a) is because the Eq. (3),  $L_j = \frac{\sum_{i=1}^n w_i \mathbb{I}(y_i \neq h_j(\mathbf{x}_i))}{\sum_{i=1}^n w_i}$ , is the cost which is the probability of error. Therefore, the Eq. (24) becomes:

$$\star \quad z_j = \exp(-\alpha_j) (1 - L_j) + \exp(\alpha_j) L_j.$$

# Upper Bound on the Generalization Error of Boosting

- Recall the formula of  $\alpha_j$  in the Algorithm, i.e.,  $\alpha_j = \log\left(\frac{1-L_j}{L_j}\right)$ . Scaling it is not harmful to AdaBoost:

$$\alpha_j = \left(\frac{1}{2}\right) \log\left(\frac{1-L_j}{L_j}\right). \quad (26)$$

- Therefore, we can have:

$$\begin{aligned} \star \quad z_j &= \exp(-\alpha_j) (1-L_j) + \exp(\alpha_j) L_j \\ &= \exp\left(-\frac{1}{2} \log\left(\frac{1-L_j}{L_j}\right)\right) (1-L_j) + \exp\left(\frac{1}{2} \log\left(\frac{1-L_j}{L_j}\right)\right) L_j \\ &= \exp\left(\log\left(\sqrt{\frac{L_j}{1-L_j}}\right)\right) (1-L_j) + \exp\left(\log\left(\sqrt{\frac{1-L_j}{L_j}}\right)\right) L_j \\ &= \sqrt{\frac{L_j}{1-L_j}} (1-L_j) + \sqrt{\frac{1-L_j}{L_j}} L_j = \sqrt{L_j(1-L_j)} + \sqrt{L_j(1-L_j)} \\ &\Rightarrow z_j = 2\sqrt{L_j(1-L_j)}. \end{aligned} \quad (27)$$

# Upper Bound on the Generalization Error of Boosting

- We found Eqs. (26), (27), and (23):

$$\alpha_j = \left(\frac{1}{2}\right) \log\left(\frac{1 - L_j}{L_j}\right), \quad z_j = 2\sqrt{L_j(1 - L_j)}, \quad \mathbb{P}(y_i \hat{f}(\mathbf{x}_i) \leq \theta) \leq \exp\left(\theta \sum_{j=1}^k \alpha_j\right) \left(\prod_{j=1}^k z_j\right).$$

- Plugging Eqs. (26) and (27) in Eq. (23) gives:

$$\begin{aligned} \mathbb{P}(y_i \hat{f}(\mathbf{x}_i) \leq \theta) &\leq \exp\left(\left(\frac{1}{2}\theta\right) \sum_{j=1}^k \log\left(\frac{1 - L_j}{L_j}\right)\right) (2^k) \prod_{j=1}^k \sqrt{L_j(1 - L_j)} \\ &= (2^k) \exp\left(\sum_{j=1}^k \log\left(\left(\frac{1 - L_j}{L_j}\right)^{\theta/2}\right)\right) \prod_{j=1}^k \sqrt{L_j(1 - L_j)} \\ &= 2^k \prod_{j=1}^k \exp\left(\log\left(\left(\frac{1 - L_j}{L_j}\right)^{\theta/2}\right)\right) \prod_{j=1}^k \sqrt{L_j(1 - L_j)} \\ &= 2^k \prod_{j=1}^k \left(\frac{1 - L_j}{L_j}\right)^{\theta/2} \prod_{j=1}^k \sqrt{L_j(1 - L_j)} = 2^k \prod_{j=1}^k \sqrt{\left(\frac{1 - L_j}{L_j}\right)^\theta L_j(1 - L_j)}, \end{aligned}$$

Handwritten notes on the right side of the derivation:

- Diagram showing the simplification of the product term:  $\prod_{j=1}^k \sqrt{L_j(1 - L_j)} = \sqrt{\prod_{j=1}^k L_j(1 - L_j)}$ . It further shows that  $\prod_{j=1}^k L_j = L$  and  $\prod_{j=1}^k (1 - L_j) = 1 - L$ , leading to  $\sqrt{L(1 - L)}$ .
- Diagram showing the simplification of the exponential term:  $\exp\left(\sum_{j=1}^k \log\left(\left(\frac{1 - L_j}{L_j}\right)^{\theta/2}\right)\right) = \exp\left(\frac{\theta}{2} \sum_{j=1}^k \log\left(\frac{1 - L_j}{L_j}\right)\right) = \left(\prod_{j=1}^k \frac{1 - L_j}{L_j}\right)^{\theta/2} = \frac{1 - L}{L}.$

which simplifies to the upper bound on the generalization error of AdaBoost [9]:

$$\mathbb{P}(y_i \hat{f}(\mathbf{x}_i) \leq \theta) \leq 2^k \prod_{j=1}^k \sqrt{L_j^{1-\theta} (1 - L_j)^{1+\theta}}. \quad (28)$$

$$e^{a+b} = e^a e^b$$



# Upper Bound on the Generalization Error of Boosting

- We found the upper bound on the generalization error of AdaBoost [9]:

$$\star \mathbb{P}(y_i \hat{f}(\mathbf{x}_i) \leq \theta) \leq 2^k \prod_{j=1}^k \sqrt{L_j^{1-\theta} (1-L_j)^{1+\theta}}, \quad \leftarrow$$

where  $\mathbb{P}(y_i \hat{f}(\mathbf{x}_i) \leq \theta)$  is the probability that the generalization (true) error for the  $i$ -th instance is less than  $\theta > 0$ .

- According to Eq. (3),  $L_j = \frac{\sum_{i=1}^n w_i \mathbb{I}(y_i \neq h_j(\mathbf{x}_i))}{\sum_{i=1}^n w_i}$ , we have  $L_j \in [0, 1]$ . If we have:

$$L_j \leq 0.5 - \xi, \quad \star \star \quad (29)$$

where  $\xi \in (0, 0.5)$ , the Eq. (28) becomes:

$$\begin{aligned} \mathbb{P}(y_i \hat{f}(\mathbf{x}_i) \leq \theta) &\leq 2^k \prod_{j=1}^k \sqrt{L_j^{1-\theta} (1-L_j)^{1+\theta}} = \prod_{j=1}^k (2 \sqrt{L_j^{1-\theta} (1-L_j)^{1+\theta}}) \\ &= \prod_{j=1}^k \sqrt{2^2 L_j^{1-\theta} (1-L_j)^{1+\theta}} = \prod_{j=1}^k \sqrt{2^{1-\theta} L_j^{1-\theta} 2^{1+\theta} (1-L_j)^{1+\theta}} \\ &= \prod_{j=1}^k \sqrt{(2L_j)^{1-\theta} (2(1-L_j))^{1+\theta}} \stackrel{(29)}{\leq} \left( \sqrt{(1-2\xi)^{1-\theta} (1+2\xi)^{1+\theta}} \right)^k. \end{aligned} \quad (30)$$

# Upper Bound on the Generalization Error of Boosting

- We found:

$$P(y_i \hat{f}(x_i) \leq \theta) \\ \star \cancel{P(y_i \hat{f}(x_i) \leq \theta)} \leq \left( \sqrt{(1 - 2\xi)^{1-\theta} (1 + 2\xi)^{1+\theta}} \right)^k \quad \leftarrow$$

which is a very good upper bound because if  $\theta < \xi$  we have  $\sqrt{(1 - 2\xi)^{1-\theta} (1 + 2\xi)^{1+\theta}} < 1$ ; thus, the probability of error,  $P(y_i \hat{f}(x_i) \leq \theta)$ , decreases exponentially with  $k$  which is the number of models used in boosting.

- This shows that boosting helps us reduce the generalization error and thus helps us avoid overfitting. In other words, because of the bound on generalization error, boosting overfits very hardly.

- ● If  $\xi$  is a very small positive number, the  $L_j \leq 0.5 - \xi$  is a little smaller than 0.5, i.e.,  $L_j \lesssim 0.5$ . As we are discussing binary classification in boosting,  $L_j = 0.5$  means random classification by flipping a coin. Therefore, for having the great bound of Eq. (30), having weak base models (a little better than random decision) suffices. This shows the effectiveness of boosting.

- Note that a very small  $\xi$  means a very small  $\theta$  because of  $\theta < \xi$ ; therefore, it means a very small probability of error because of  $P(y_i \hat{f}(x_i) \leq \theta)$ .
- It is noteworthy that both boosting and bagging can be seen as ensemble learning [10] (or majority voting) methods which use model averaging [11, 12] and are very effective in learning theory.
- Moreover, both boosting and bagging reduce the variance of estimation [13, 9], especially for the models with high variance of estimation such as trees [14].
- In the above, we analyzed boosting for binary classification. A similar discussion can be done for multi-class classification in boosting and find an upper bound on the generalization error (see the appendix in [9] for more details).

## Boosting as Maximum Margin Classifier

# Boosting as Maximum Margin Classifier

- In another perspective, the found upper bound for boosting shows that boosting can be seen as a method to increase (maximize) the margins of training error which results in a good generalization error [15]. This phenomenon is the base for the theory of Support Vector Machines (SVM) [16, 17].
- In the following, we analyze the analogy between maximum margin classifier (i.e., SVM) and boosting [9]. In addition to [9], some more discussions exist for upper bound and margin of boosting [18, 19] to which we refer the interested readers.
- Assume we have training instances  $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$  where  $y_i \in \{-1, +1\}$  for binary classification. The two classes may not be linearly separable. In order to handle this case, we map the data to higher dimensional feature space using kernels [20, 21], hoping that they become linearly separable in the feature space. Assume  $\mathbf{h}(\mathbf{x})$  is a vector which non-linearly maps data to the feature space. Considering  $\boldsymbol{\alpha}$  as the vector of optimization variables, the optimization problem [22] in SVM is [17, 9]:

$$\left[ \begin{array}{ll} \underset{\boldsymbol{\alpha}}{\text{maximize}} & \underset{\{(\mathbf{x}_i, y_i)\}_{i=1}^n}{\text{minimize}} \quad \frac{y_i (\boldsymbol{\alpha}^\top \mathbf{h}(\mathbf{x}_i))}{\|\boldsymbol{\alpha}\|_2} \end{array} \right. \quad (31)$$

Note that  $y_i = \pm 1$  and  $\boldsymbol{\alpha}^\top \mathbf{h}(\mathbf{x}_i) \geq 0$ ; therefore, the sign of  $y_i (\boldsymbol{\alpha}^\top \mathbf{h}(\mathbf{x}_i))$  determines the class of the  $i$ -th instance.

# Boosting as Maximum Margin Classifier

- Eq. (31) was:

$$\star \quad \underset{\alpha}{\text{maximize}} \quad \underset{\{(x_i, y_i)\}_{i=1}^n}{\text{minimize}} \quad \frac{y_i (\alpha^\top \mathbf{h}(x_i))}{\|\alpha\|_2}$$

- On the other hand, the Eq. (16),  $\hat{f}(x_i) = \frac{\sum_{j=1}^k \alpha_j h_j(x_i)}{\sum_{j=1}^k \alpha_j}$ , can be written in a vector form:

$$\hat{f}(x_i) = \frac{\sum_{j=1}^k \alpha_j h_j(x_i)}{\sum_{j=1}^k \alpha_j} = \frac{\alpha^\top \mathbf{h}(x_i)}{\|\alpha\|_1}, \quad (32)$$

where  $\mathbf{h}(x_i) = [h_1(x_i), \dots, h_k(x_i)]^\top$  and  $\alpha = [\alpha_1, \dots, \alpha_k]^\top$ . Note that here,  $h(x_i) = \pm 1$  and  $\alpha_j$  is obtained from Eq. (26),  $\alpha_j = \frac{1}{2} \log(\frac{1-L_j}{L_j})$ , or the formula of  $\alpha_j$  in the Algorithm.

- The similarity between the Eq. (32) and the cost function in Eq. (31) shows that boosting can be seen as maximizing the margin of classification resulting in a good generalization error [9].
- In other words, finding a linear combination in the high dimensional feature space having a large margin between the training instances of the classes is performed in the two methods.
- Note that a slight difference is the type of norm which is interpretable because the mapping to feature space in boosting is only to  $h(x_i) = \pm 1$  while in SVM, it can be any number where the sign is important. Therefore,  $\ell_1$  and  $\ell_2$  norms are suitable in boosting and SVM, respectively [9].

# Boosting as Maximum Margin Classifier

- Another connection between SVM (maximum margin classifier) and boosting is that some of the training instances are found to be most important instances, called **support vectors** [17]. In boosting, also, weighting the training instances can be seen as selecting some **informative** models [23] which can be analogous to support vectors.

# Acknowledgment

- For more information on boosting in machine learning, see the book: Trevor Hastie, Robert Tibshirani, Jerome H. Friedman, Jerome H. Friedman. "The elements of statistical learning: data mining, inference, and prediction". Vol. 2. New York: springer, 2009 [24].
- Some slides are based on our tutorial paper: "The Theory Behind Overfitting, Cross Validation, Regularization, Bagging, and Boosting: Tutorial" [25]
- Some slides of this slide deck are inspired by teachings of Prof. Ali Ghodsi at University of Waterloo, Department of Statistics and Prof. Hoda Mohammadzade at Sharif University of Technology, Department of Electrical Engineering.

# References

- [1] M. Kearns, "Thoughts on hypothesis boosting," *Technical Report, Machine Learning class project*, pp. 1–9, 1988.
- [2] M. Kearns and L. Valiant, "Cryptographic limitations on learning boolean formulae and finite automata," *Journal of the ACM (JACM)*, vol. 41, no. 1, pp. 67–95, 1994.
- [3] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proceedings of the Thirteenth International Conference on Machine Learning*, pp. 148–156, Morgan Kaufman, San Francisco, 1996.
- [4] T. J. Hastie and R. Tibshirani, "Generalized additive models," *Statistical Science*, vol. 1, no. 3, pp. 297–318, 1986.
- [5] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting," *The annals of statistics*, vol. 28, no. 2, pp. 337–407, 2000.
- [6] R. Rojas, "Adaboost and the super bowl of classifiers: a tutorial introduction to adaptive boosting," *Freie University, Berlin, Technical Report*, 2009.
- [7] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," *Complex Systems*, vol. 2, pp. 321–355, 1988.
- [8] F. Schwenker, H. A. Kestler, and G. Palm, "Three learning phases for radial-basis-function networks," *Neural networks*, vol. 14, no. 4-5, pp. 439–458, 2001.



## References (cont.)

- [9] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, "Boosting the margin: A new explanation for the effectiveness of voting methods," *The annals of statistics*, vol. 26, no. 5, pp. 1651–1686, 1998.
- [10] R. Polikar, "Ensemble learning," in *Ensemble machine learning*, pp. 1–34, Springer, 2012.
- [11] J. A. Hoeting, D. Madigan, A. E. Raftery, and C. T. Volinsky, "Bayesian model averaging: a tutorial," *Statistical science*, pp. 382–401, 1999.
- [12] G. Claeskens and N. L. Hjort, *Model selection and model averaging*. Cambridge Books, Cambridge University Press, 2008.
- [13] L. Breiman, "Arcing classifier (with discussion and a rejoinder by the author)," *The annals of statistics*, vol. 26, no. 3, pp. 801–849, 1998.
- [14] J. R. Quinlan, "Bagging, boosting, and c4.5," in *AAAI/IAAI Conference*, vol. 1, pp. 725–730, 1996.
- [15] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, ACM, 1992.
- [16] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

## References (cont.)

- [17] C. J. Burges, “A tutorial on support vector machines for pattern recognition,” *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.
- [18] L. Wang, M. Sugiyama, C. Yang, Z.-H. Zhou, and J. Feng, “On the margin explanation of boosting algorithms,” in *COLT*, pp. 479–490, Citeseer, 2008.
- [19] W. Gao and Z.-H. Zhou, “On the doubt about margin explanation of boosting,” *Artificial Intelligence*, vol. 203, pp. 1–18, 2013.
- [20] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.
- [21] T. Hofmann, B. Schölkopf, and A. J. Smola, “Kernel methods in machine learning,” *The annals of statistics*, pp. 1171–1220, 2008.
- [22] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.
- [23] Y. Freund, “Boosting a weak learning algorithm by majority,” *Information and computation*, vol. 121, no. 2, pp. 256–285, 1995.
- [24] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, vol. 2. Springer, 2009.

## References (cont.)

- [25] B. Ghojogh and M. Crowley, “The theory behind overfitting, cross validation, regularization, bagging, and boosting: tutorial,” *arXiv preprint arXiv:1905.12787*, 2019.