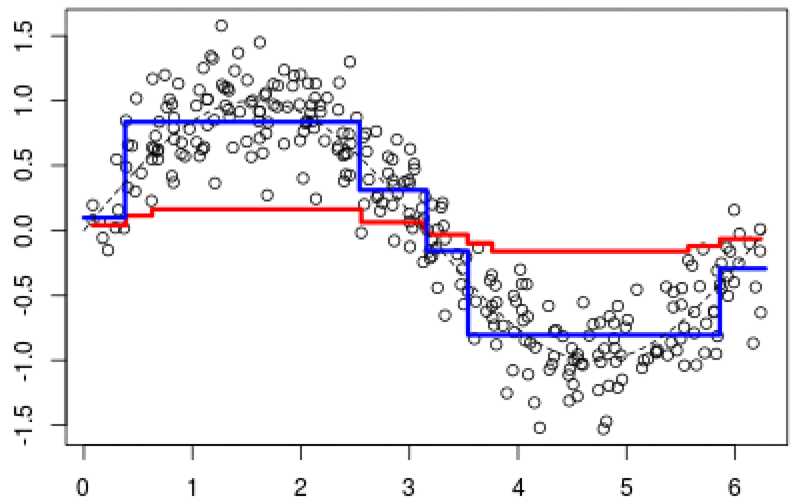


GBM → XGBoost, LightGBM, CatBoost

Ensemble Learning: Gradient Boosting Machine (GBM)

Pilsung Kang
School of Industrial Management Engineering
Korea University

Gradient Boosting Machine: GBM

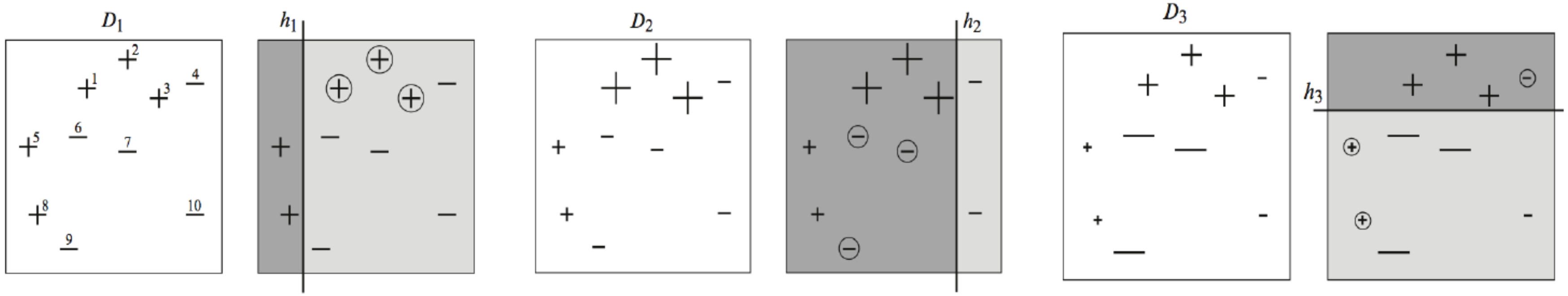
Friedman (2001), Natekin and Knoll (2013)

Gradient Boosting = Gradient Descent + Boosting

①

②

- Adaboost



α_t

✓ Fit an additive model (ensemble) $\sum_t \rho_t h_t(x)$ in a forward stage-wise manner.

✓ In each stage, introduce a weak leaner to compensate the shortcomings of existing weak leaners.

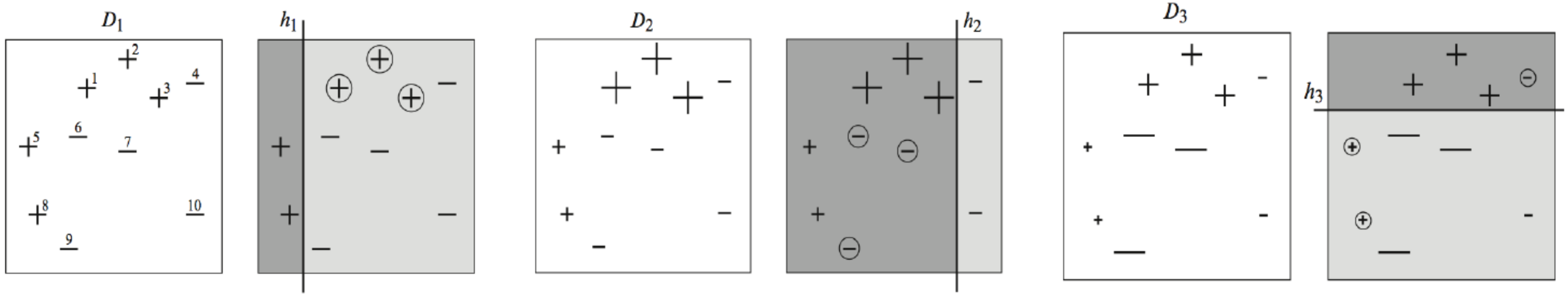
✓ In Adaboost, "shortcomings" are identified by high-weight data points.

KOREA UNIVERSITY

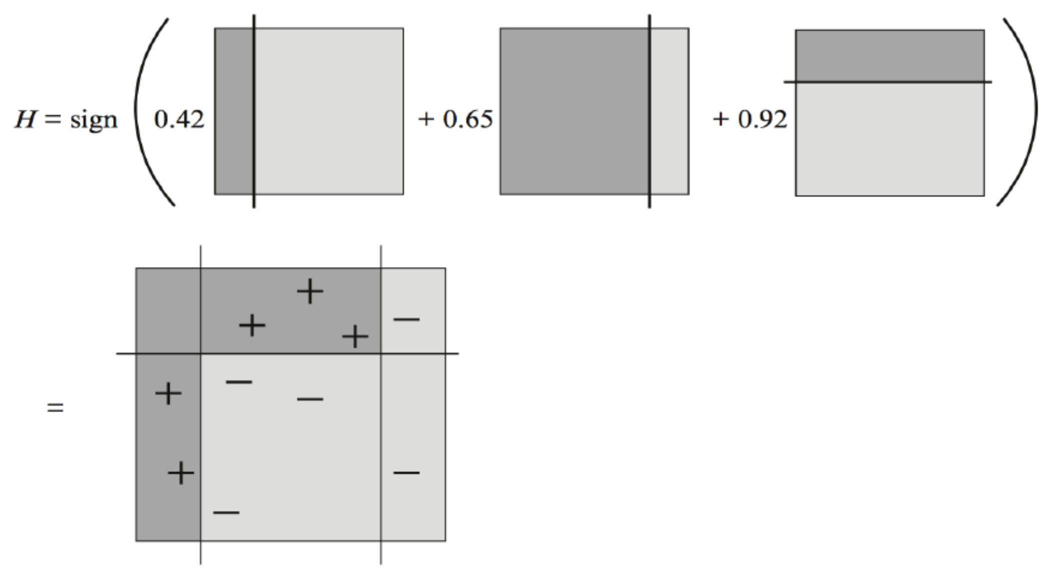
DSBA Data Science & Business Analytics

Gradient Boosting Machine: GBM

- Adaboost



$$H(x) = \sum_t \rho_t h_t(x)$$



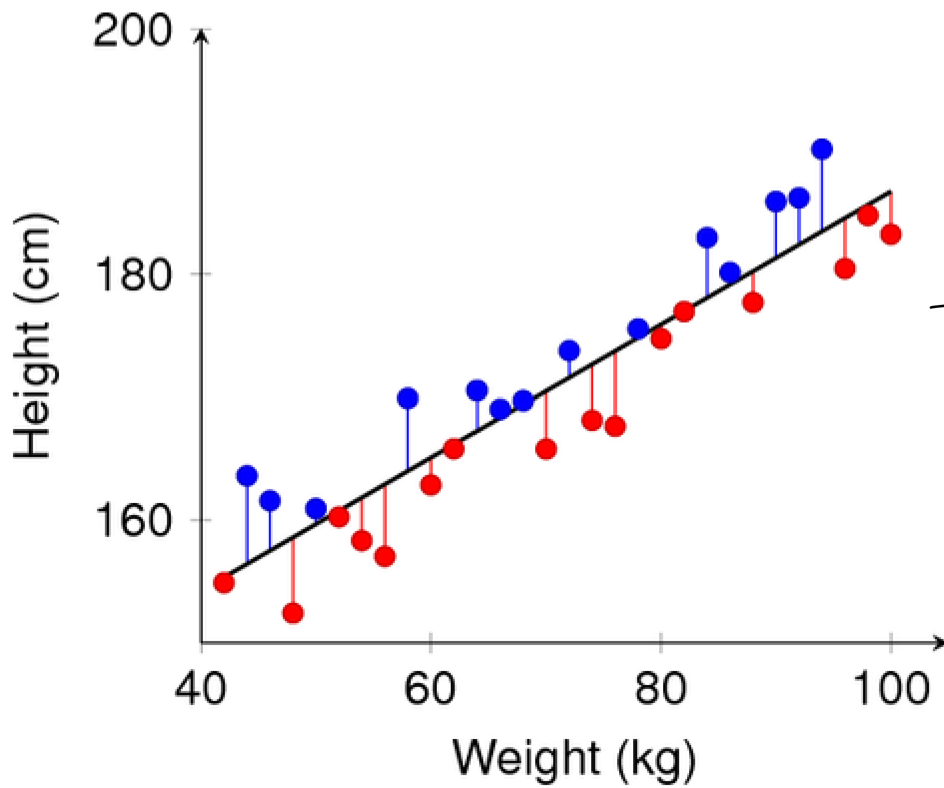
Gradient Boosting Machine: GBM

- Gradient Boosting
  - ✓ Fit an additive model (ensemble) $\sum_t \rho_t h_t(x)$ in a forward stage-wise manner.
  - ✓ In each stage, introduce a weak leaner to compensate the shortcomings of existing weak leaners.
  - ✓ In Gradient Boosting, "shortcomings" are identified by gradients.
  - ✓ Both high-weight data points and gradients tell us how to improve our model.

- Gradient Boosting for Different Problems
  - ✓ Difficulty: Regression < Classification < Ranking → 손실함수만 다름
    - Associated with the complexity of the derivative of a loss function

Gradient Boosting Machine: GBM

- Motivation (for regression problem)
  - ✓ What if we attempt to predict the residuals with the additional regression model?



$\hat{y} = f_1(x)$

$\underbrace{y - f_1(x)}_{\text{residual}} = f_2(x)$

$f_1(x) + f_2(x) = y$

Gradient Boosting Machine: GBM

- Main idea

다변량 데이터

Original Dataset

x^1	y^1
x^2	y^2
x^3	y^3
x^4	y^4
x^5	y^5
x^6	y^6
x^7	y^7
x^8	y^8
x^9	y^9
x^{10}	y^{10}

Modified Dataset 1

x^1	$y^1-f_1(x^1)$
x^2	$y^2-f_1(x^2)$
x^3	$y^3-f_1(x^3)$
x^4	$y^4-f_1(x^4)$
x^5	$y^5-f_1(x^5)$
x^6	$y^6-f_1(x^6)$
x^7	$y^7-f_1(x^7)$
x^8	$y^8-f_1(x^8)$
x^9	$y^9-f_1(x^9)$
x^{10}	$y^{10}-f_1(x^{10})$

Modified Dataset 2

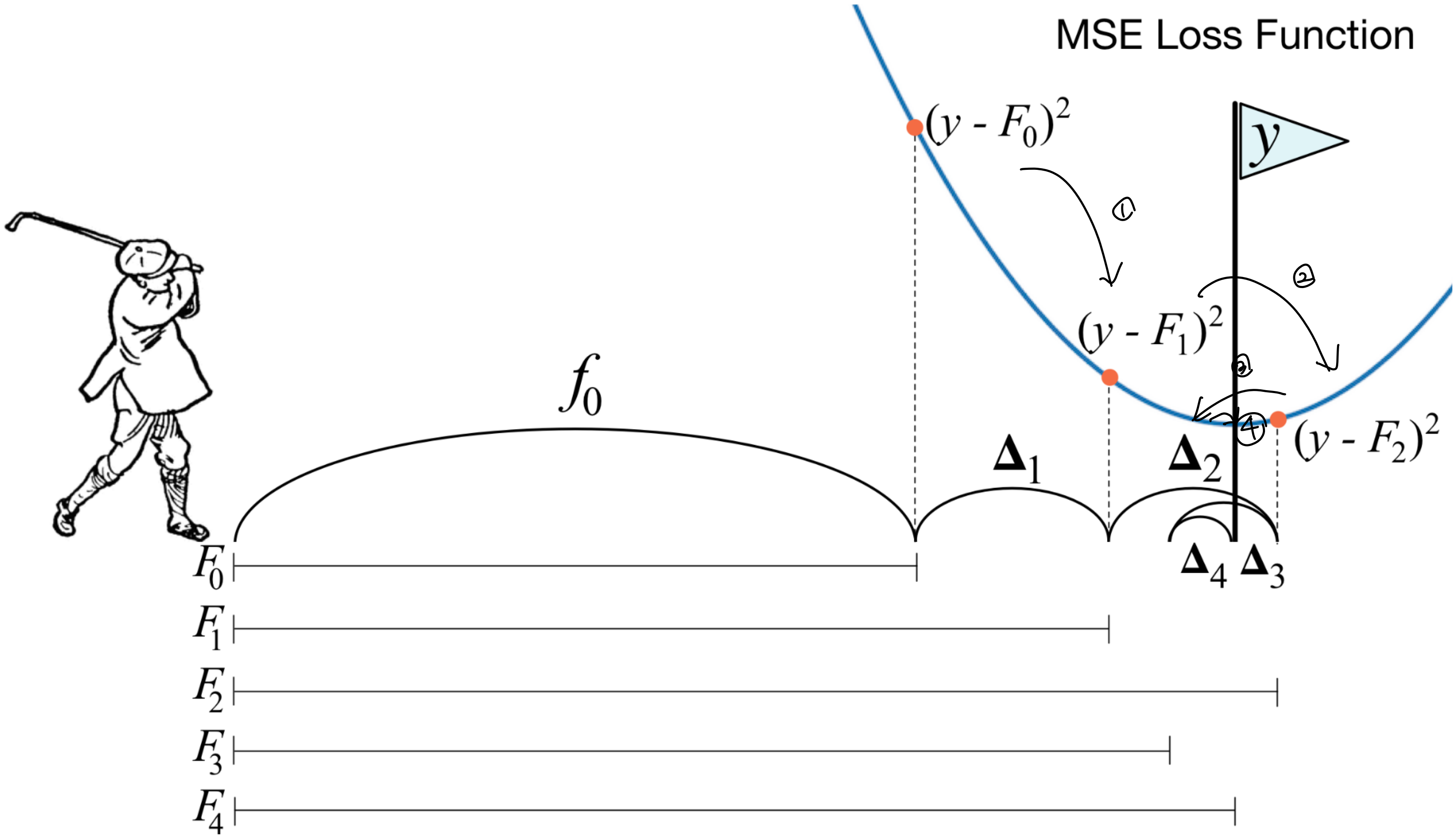
x^1	$y^1-f_1(x^1)-f_2(x^1)$
x^2	$y^2-f_1(x^2)-f_2(x^2)$
x^3	$y^3-f_1(x^3)-f_2(x^3)$
x^4	$y^4-f_1(x^4)-f_2(x^4)$
x^5	$y^5-f_1(x^5)-f_2(x^5)$
x^6	$y^6-f_1(x^6)-f_2(x^6)$
x^7	$y^7-f_1(x^7)-f_2(x^7)$
x^8	$y^8-f_1(x^8)-f_2(x^8)$
x^9	$y^9-f_1(x^9)-f_2(x^9)$
x^{10}	$y^{10}-f_1(x^{10})-f_2(x^{10})$

• • •

$$\hat{y} = f_1(\mathbf{x}) \qquad y - f_1(\mathbf{x}) = f_2(\mathbf{x}) \qquad y - f_1(\mathbf{x}) - f_2(\mathbf{x}) = f_3(\mathbf{x})$$

Gradient Boosting Machine: GBM

- Illustrative Example



https://github.com/topics/gradient-boosting-machine

Gradient Boosting Machine: GBM

- How is this idea related to the gradient?

최소자승법

✓ Loss function of the ordinary least square (OLS)

$$\min L = \frac{1}{2}\sum_{i=1}^{n}(y_i - f(\mathbf{x}_i))^2$$

✓ Gradient of the Loss function

$$\frac{\partial L}{\partial f(\mathbf{x}_i)} = f(\mathbf{x}_i) - y_i$$

✓ Residuals are the negative gradient of the loss function

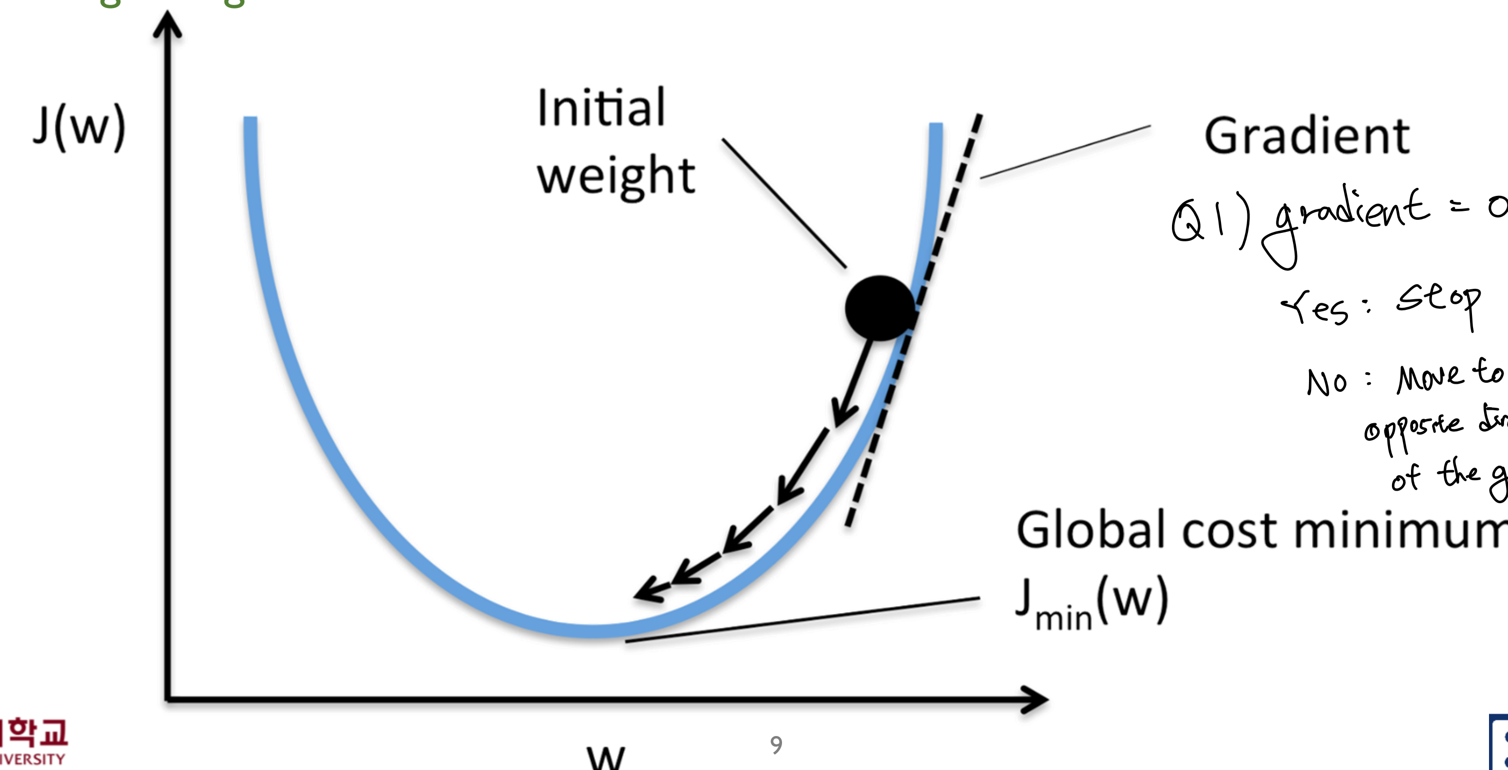
$$y_i - f(\mathbf{x}_i) = -\frac{\partial L}{\partial f(\mathbf{x}_i)}$$

gradien와 반대방향으로 움직여라 (방향)

이만큼 움직여라 (크기)

Gradient Boosting Machine: GBM

- Gradient Descent Algorithm
  - ✓ Blue line: value of loss function with a given parameter
  - ✓ Black point: current state
  - ✓ Arrows: the direction that the parameter should follow to minimize the loss function
    = negative gradient of the loss function



Q1) gradient = 0 ?

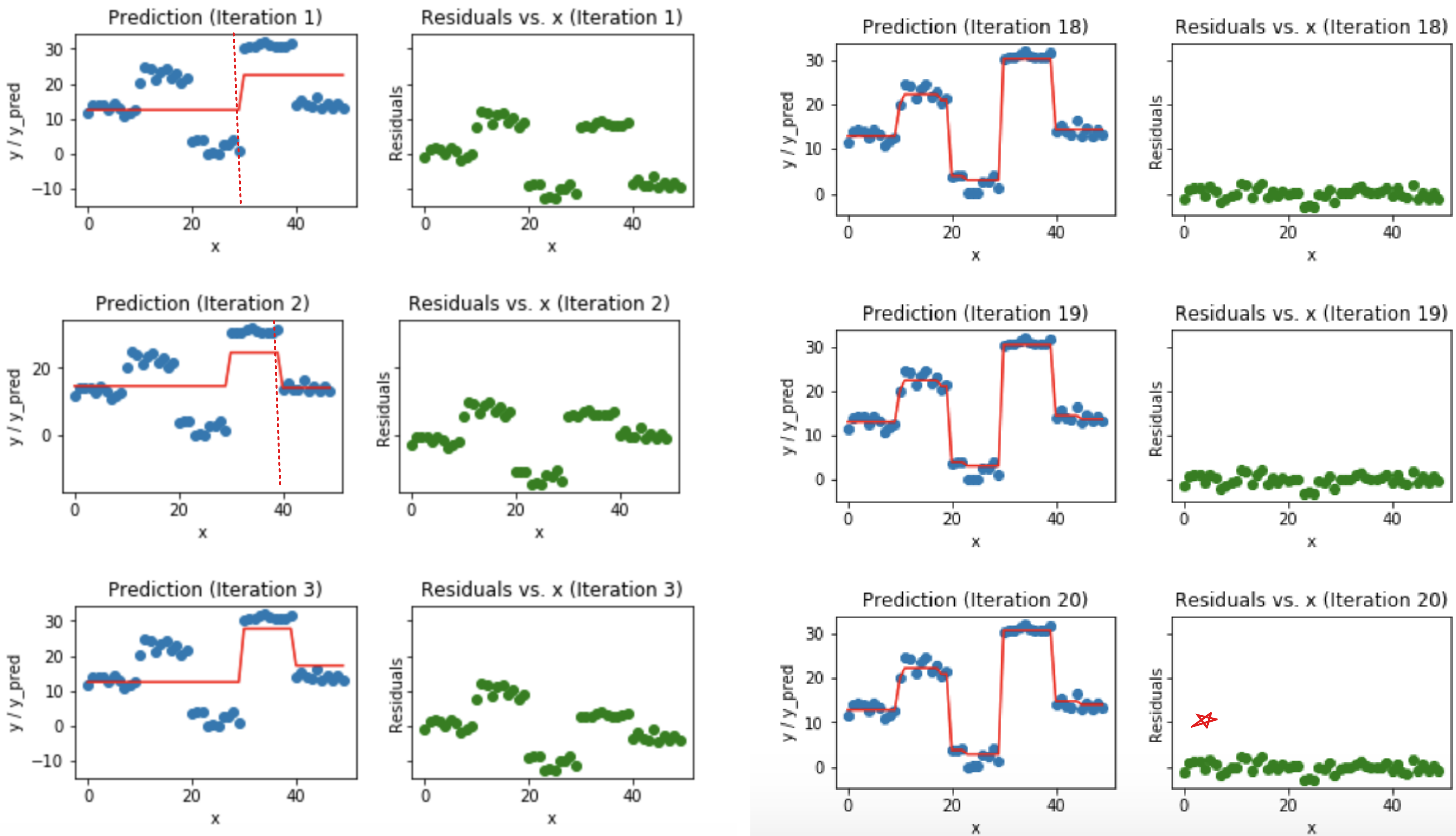
Yes : stop

No : Move to the opposite direction of the gradient

Gradient Boosting Machine: GBM

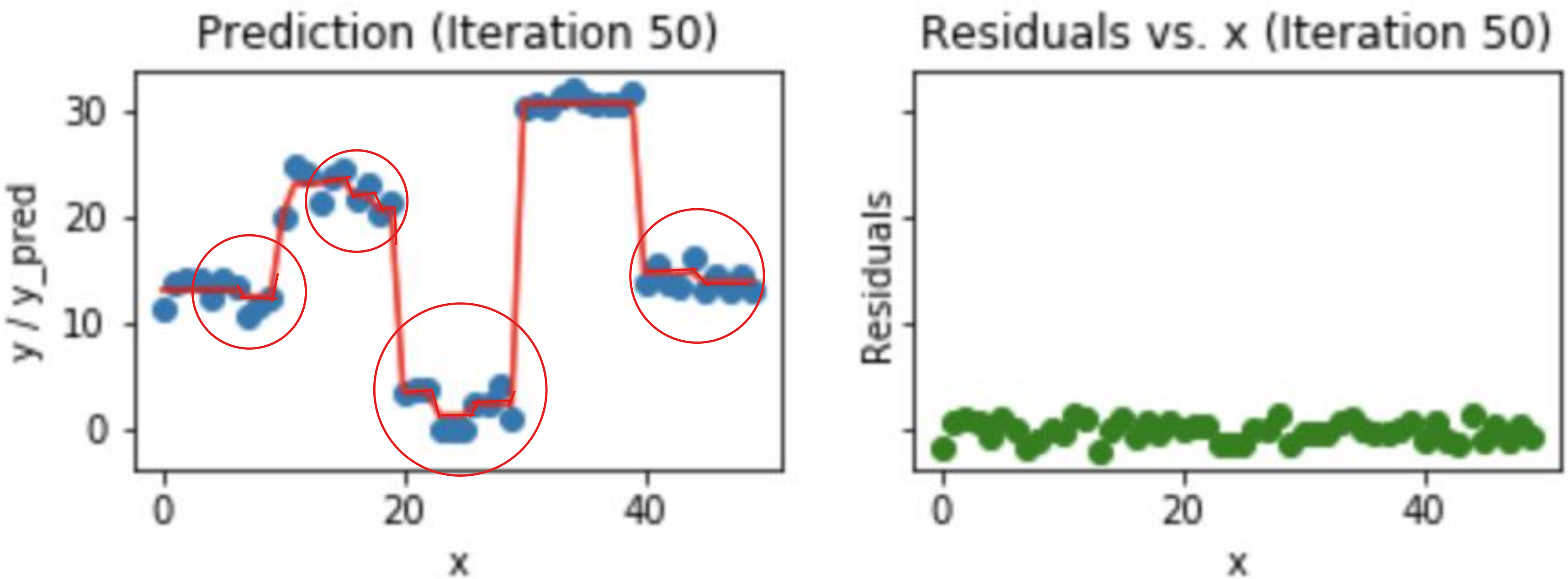
Split 1회 → 계단식의 회귀선 1회 생성

- GBM Regression Example 1



Gradient Boosting Machine: GBM

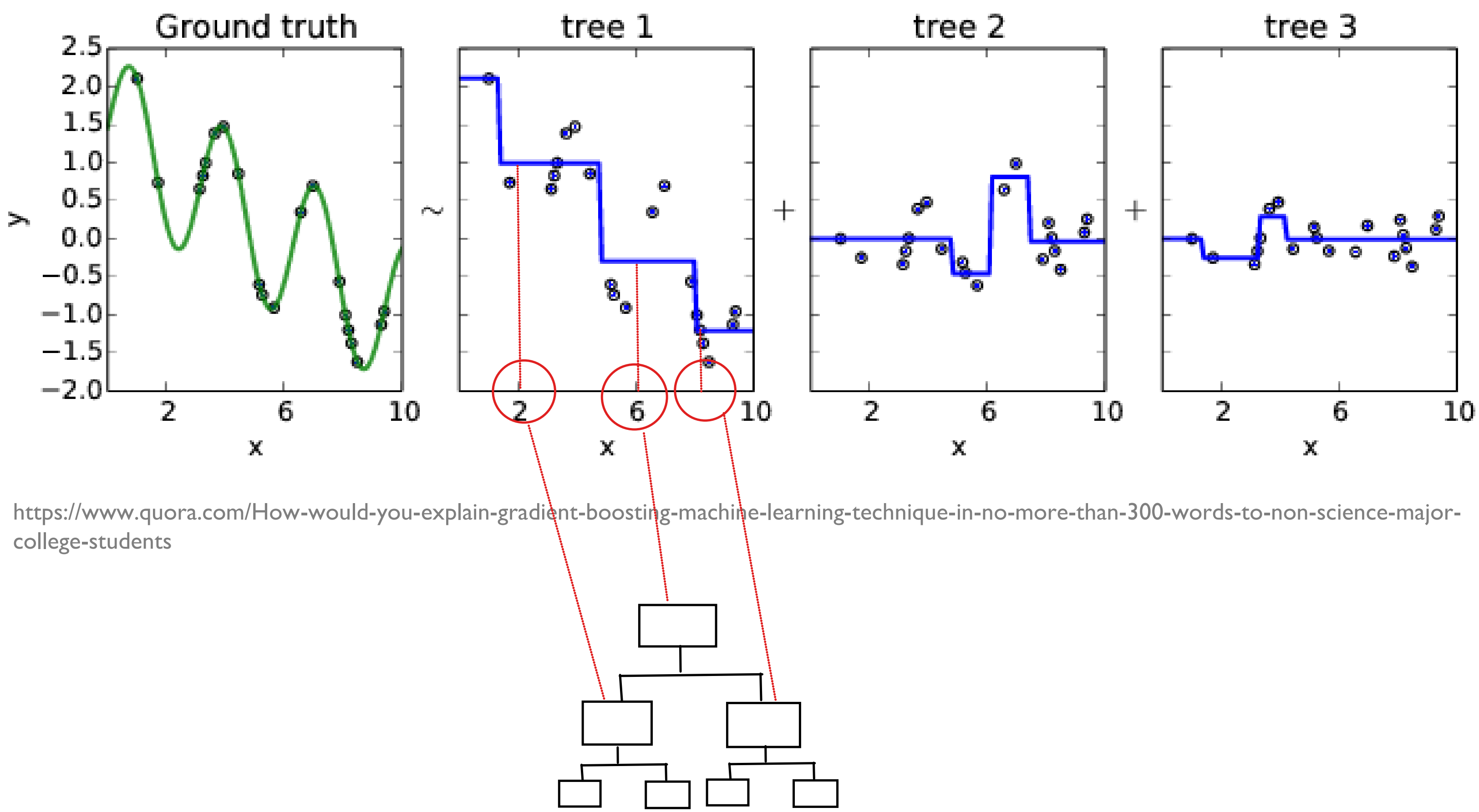
- GBM Regression Example 1



https://medium.com/mlreview/gradient-boosting-from-scratch-1e317ae4587d

Gradient Boosting Machine: GBM

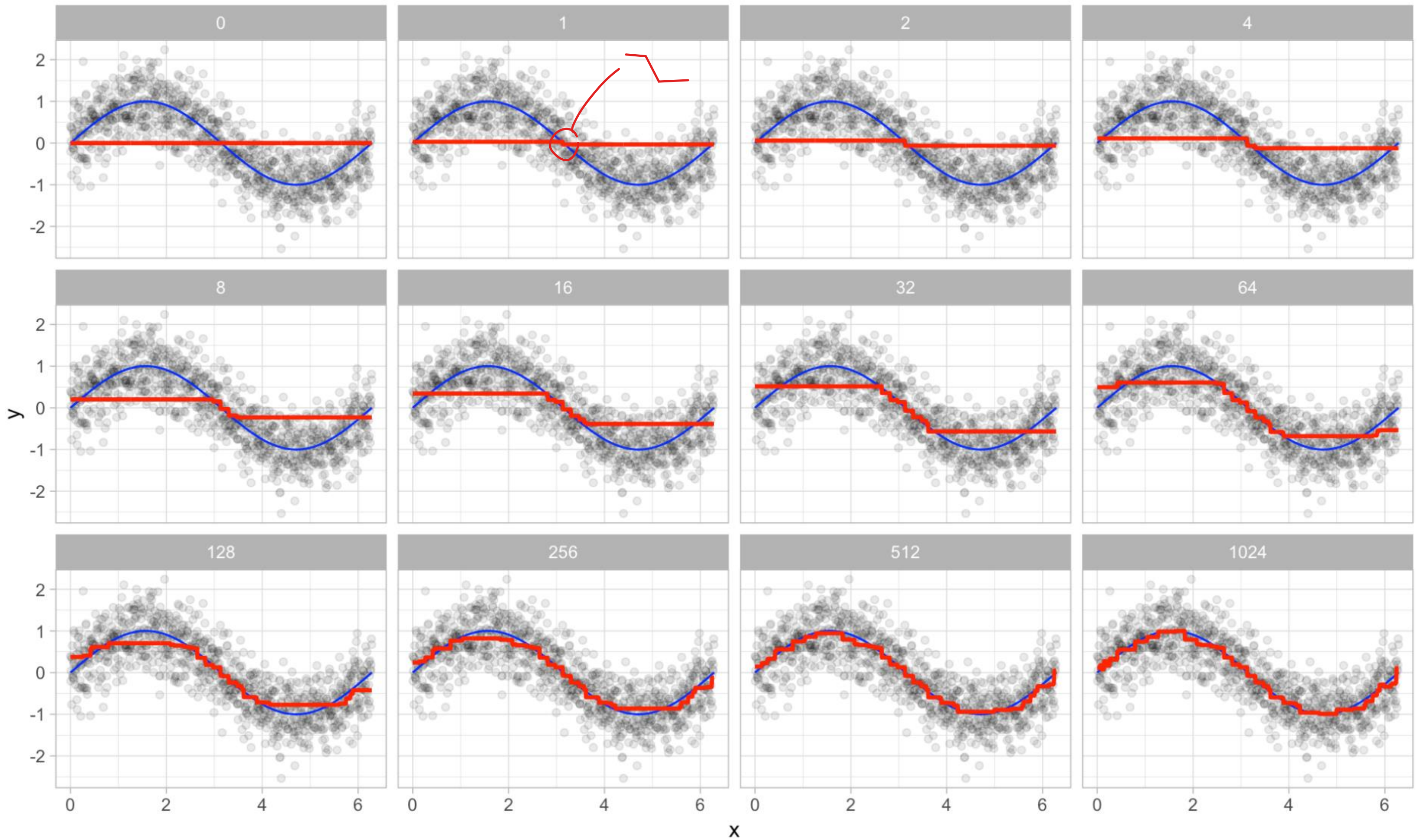
- GBM Regression Example 2



https://www.quora.com/How-would-you-explain-gradient-boosting-machine-learning-technique-in-no-more-than-300-words-to-non-science-major-college-students

Gradient Boosting Machine: GBM

- GBM Regression Example 3



https://docs.paperspace.com/machine-learning/wiki/gradient-boosting

Gradient Boosting Machine: GBM

- Gradient Boosting: Algorithm

원데이터

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:

   2.1 For $i = 1, \ldots, N$ compute

   $$g_{im} = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x_i)=f_{m-1}(x_i)}$$

   gradient

   2.2 Fit a regression tree to the targets g_{im} giving terminal regions $R_{jm}, j = 1, \ldots, J_m$.

   2.3 For $j = 1, \ldots, J_m$ compute

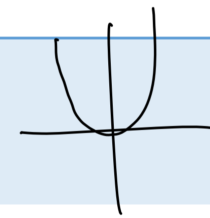
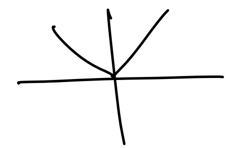
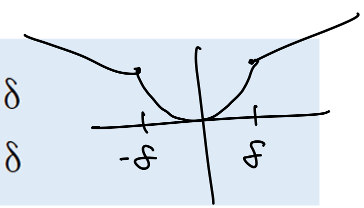
   $$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma)$$

   2.4 Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$

3. Output $\hat{f}(x) = f_M(x)$.

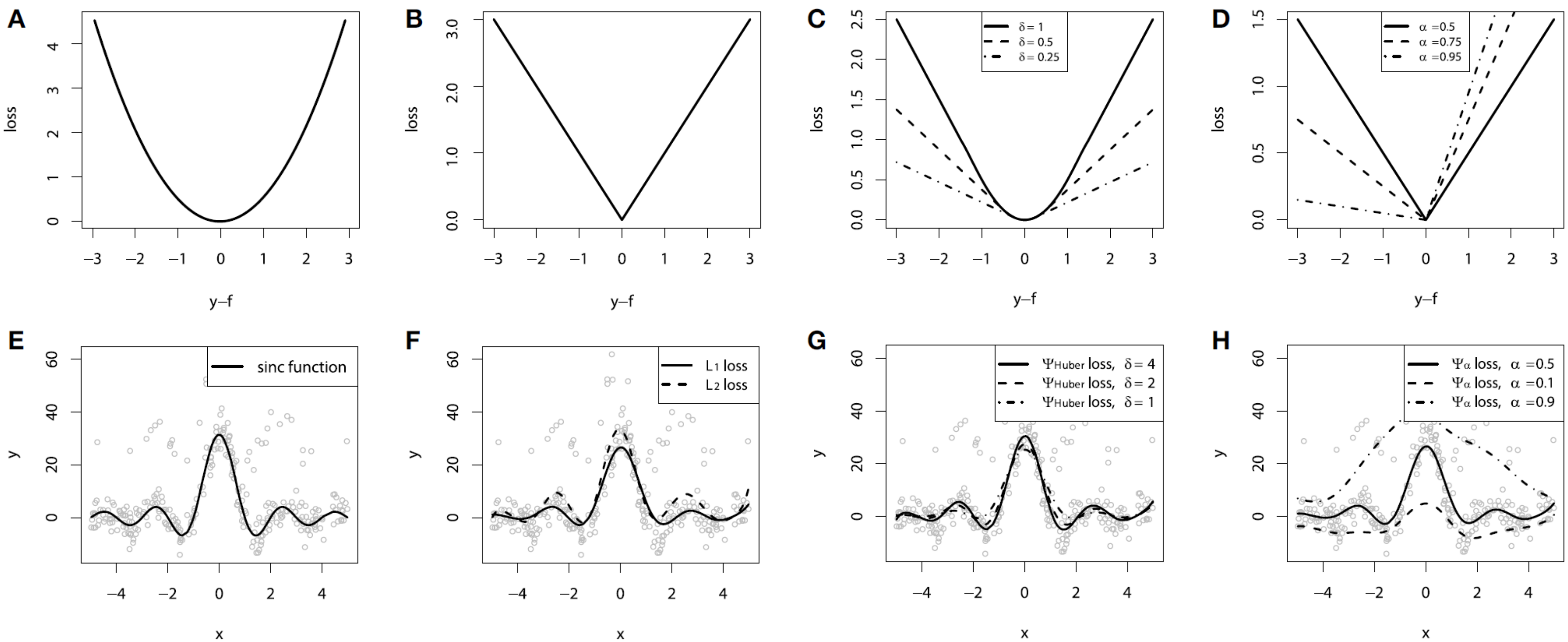
Gradient Boosting Machine: GBM

- Loss Functions for Regression

Loss Function	Formula
Squared loss (L_2)	$\Psi(y,f)_{L_2} = \frac{1}{2}(y-f)^2$ 
Absolute loss (L_1)	$\Psi(y,f)_{L_1} = \lvert y-f \rvert$ 
Huber loss	$\Psi(y,f)_{\text{Huber},\,\delta} = \begin{cases} \frac{1}{2}(y-f)^2 & \lvert y-f\rvert \le \delta \\ \delta(\lvert y-f\rvert - \delta/2) & \lvert y-f\rvert > \delta \end{cases}$ 
Quantile loss	$\Psi(y,f)_{\alpha} = \begin{cases} (1-\alpha)\lvert y-f\rvert & y-f \le 0 \\ \alpha\lvert y-f\rvert & y-f > 0 \end{cases}$

Gradient Boosting Machine: GBM

- Loss Functions for Regression



Gradient Boosting Machine: GBM

- Loss Functions for Classification

Loss Function	Formula
Bernoulli loss	$\Psi(y,f)_{\text{Bern}} = \log(1+\exp(-2\bar{y}f))$
Adaboost loss	$\Psi(y,f)_{\text{Ada}} = \exp(-\bar{y}f)$

$e^{-\ast} < 1$

$e^{+\ast} > 1$

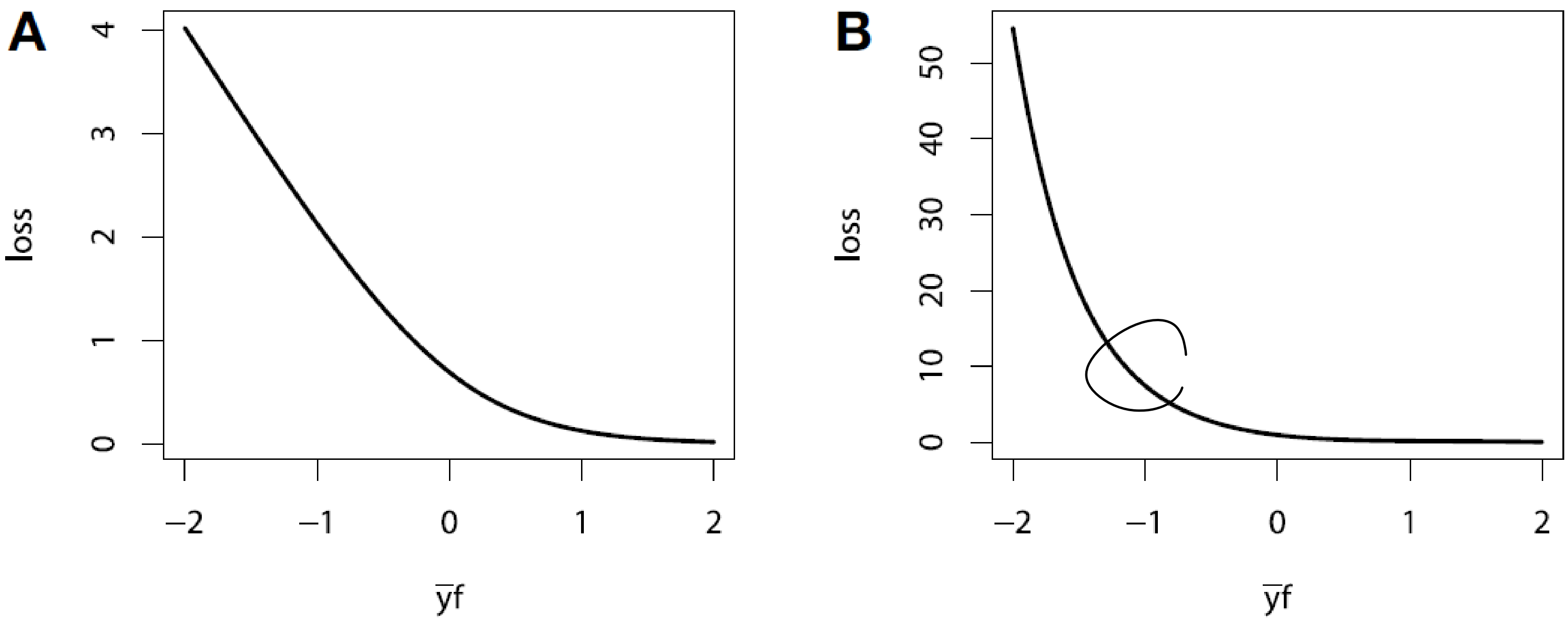
(Note)

In binary classification, the target is usually defined by $y \in \{0,1\}$, but here we define $\bar{y} = 2y - 1$ so that $\bar{y} \in \{-1,1\}$

$\rightarrow y$

Gradient Boosting Machine: GBM

- Loss Functions for Classification



(A) Bernoulli loss function. **(B)** Adaboost loss function.

Gradient Boosting Machine: GBM

- Overfitting problem in GBM

$y = f(x) + \varepsilon$

"noise"

→ 현재 noise까지 학습되버린다

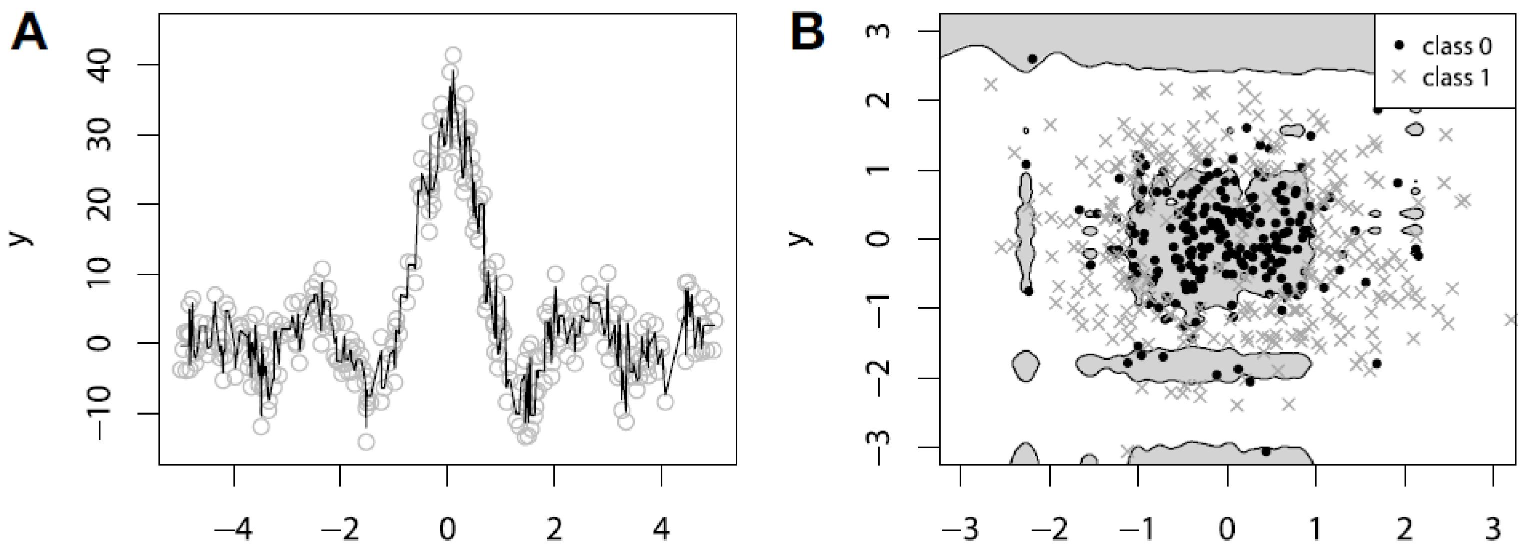
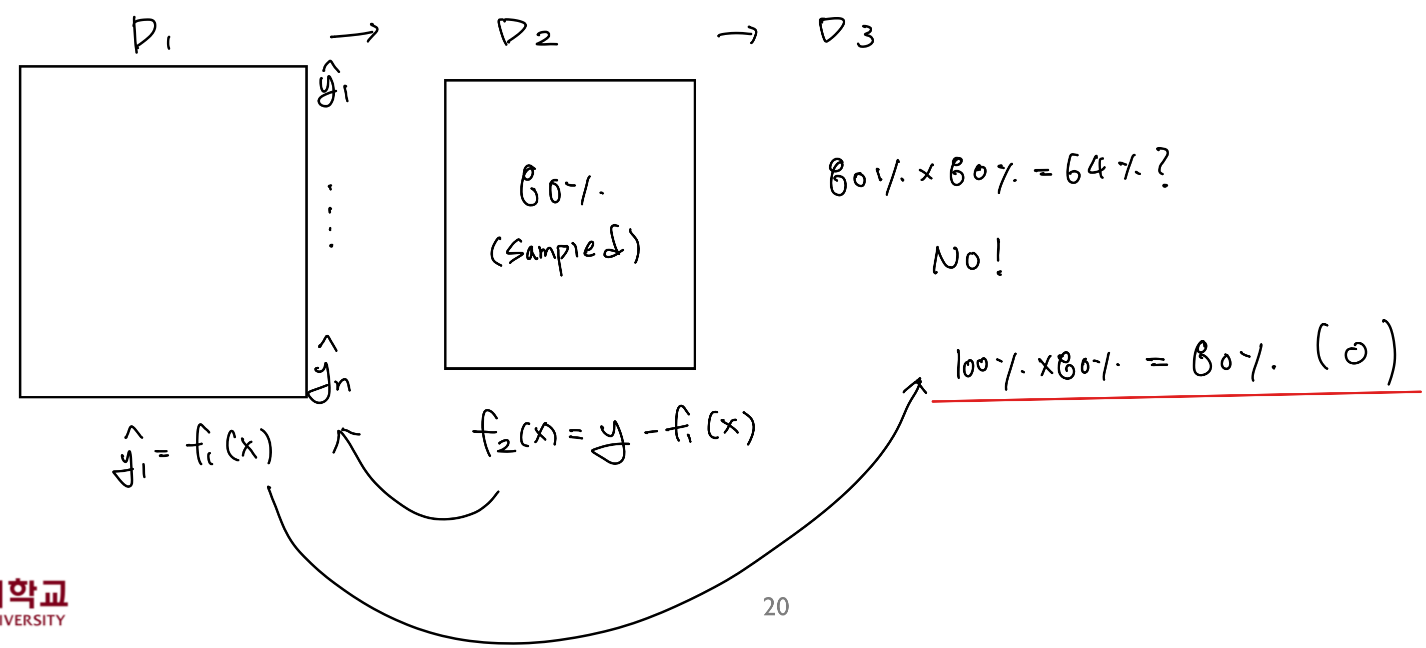


FIGURE 4 | Examples of overfitting in GBMs on: (A) regression task; (B) classification task. Demonstration of fitting a decision-tree GBM to a noisy sinc(x) data: **(C)** $M = 100$, $\lambda = 1$; **(D)** $M = 1000$, $\lambda = 1$; **(E)** $M = 100$, $\lambda = 0.1$; **(F)** $M = 1000$, $\lambda = 0.1$.

Gradient Boosting Machine: GBM

- Regularization
  - ✓ Subsampling
    - At each learning iteration, only a random part of the training data is used to fit a consecutive base-learner.
    - The training data is typically sampled without replacement, but bagging can be also acceptable.



Gradient Boosting Machine: GBM

- Regularization
  - ✓ Shrinkage
    - Used for reducing/shrinking the impact of each additional fitted base-leaners.
    - Better to improve a model by taking many small steps than by taking fewer large steps.

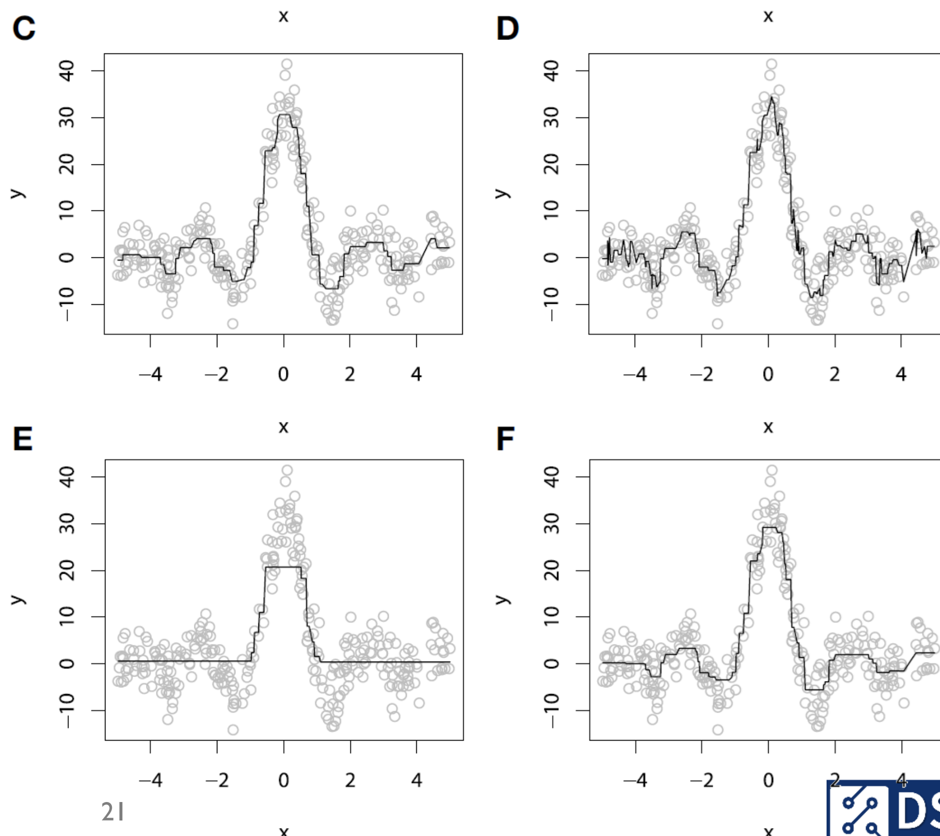
$$\widehat{f}_t \leftarrow \widehat{f}_{t-1} + \lambda \rho_t h(x, \theta_t)$$

$\hat{y} = f_1(x) + f_2(x) + \cdots + f_n(x)$

$\downarrow$

$\hat{y} = 1 \cdot f_1(x) + 0.9 f_2(x) + \cdots + 0.1 \cdot f_n(x)$

⇒ 뒤에 생긴 모델의 가중치의 영향력을
줄인다!



고려대학교 KOREA UNIVERSITY

DSBA Data Science & Business Analytics

Gradient Boosting Machine: GBM

- Regularization
  - ✓ Early Stopping
    - Use the validation error

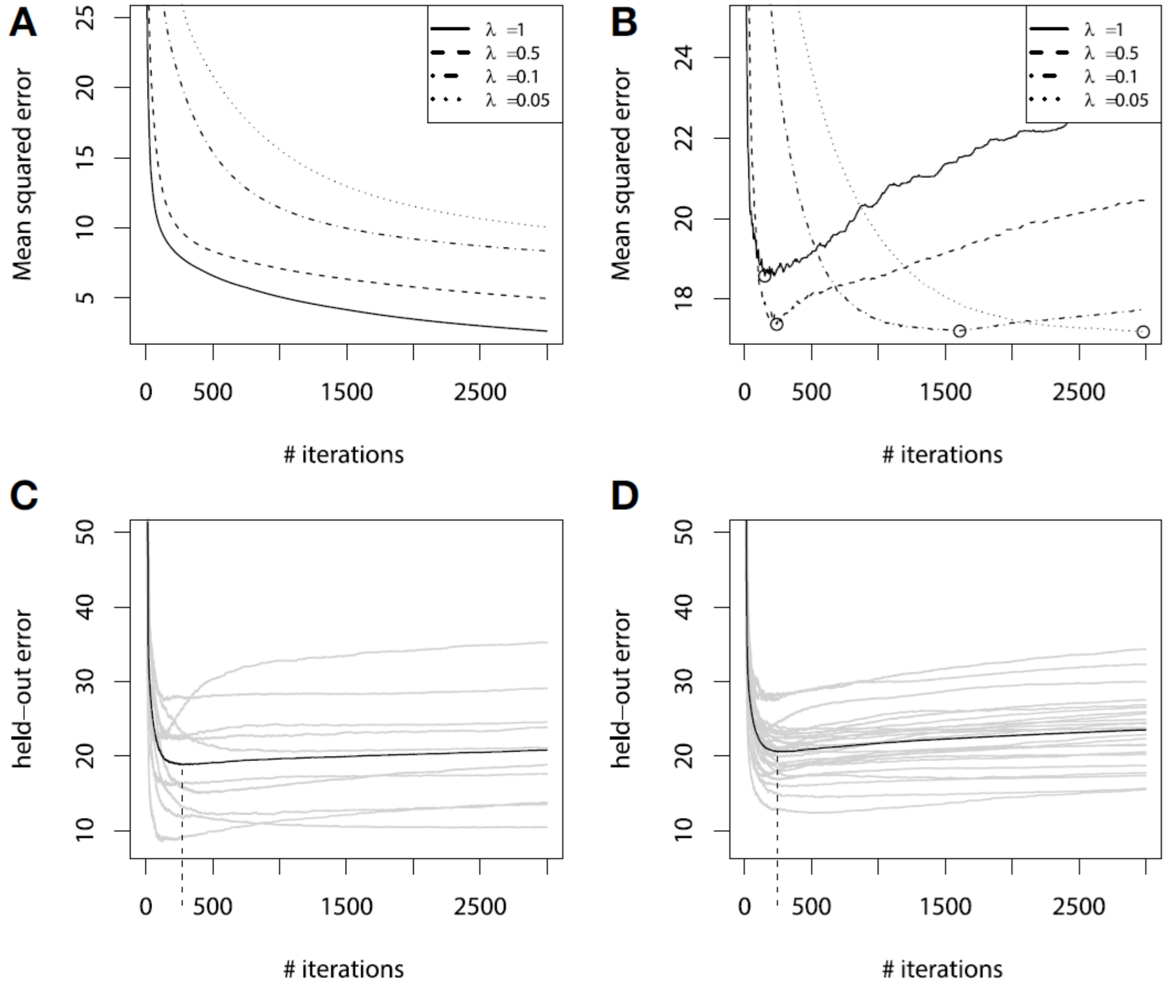


FIGURE 5 | Error curves for GBM fitting on sinc(x) data: (A) training set error; (B) validation set error. Error curves for learning simulations and number of base-learners *M* estimation: **(C)** error curves for cross-validation; **(D)** error curves for bootstrap estimates.

Gradient Boosting Machine: GBM

- Variable Importance in Tree-based Gradient Boosting
  - ✓ $Influence_j(T)$: importance of the variable j in a single tree T.
  - ✓ Assume that there are L terminal nodes → $L-1$ splits.

$$Influence_j(T) = \sum_{i=1}^{L-1} \left(\underbrace{IG_i}_{\text{Information Gain (혼잡도)}} \times \underline{\mathbf{1}(S_i = j)}\right)$$

i번째 split에서
j변수 사용 O → 1
× → 0

  - ✓ Variable importance of Gradient boosting

$$Influence_j = \frac{1}{M}\sum_{k=1}^{M} Influence_j(T_k)$$

RF와 다르게
Random Permutation을 하지 않아
상대적으로 수식이 단순함

	Performance	Variable Importance
GBM	good	Yes (Implicit)
RF	good	Yes


Q & A