データマイニング

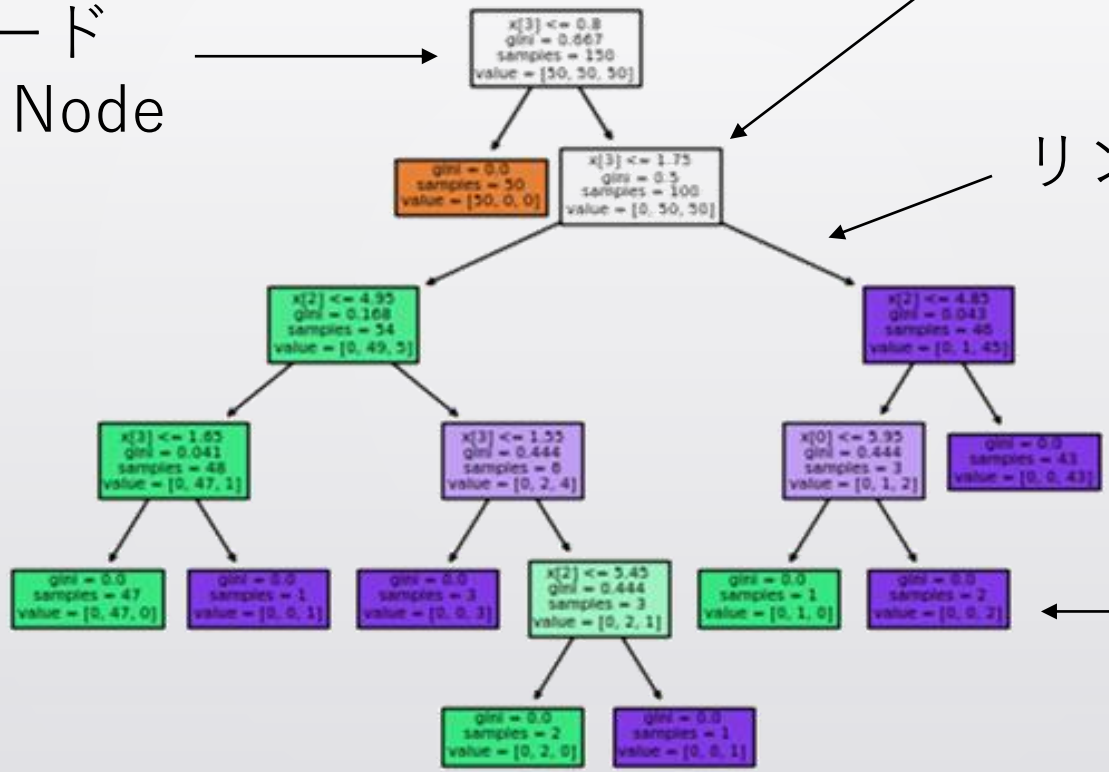
Data Mining

10: 分類⑤ Classification

土居 裕和 Hirokazu Doi

長岡技術科学大学 Nagaoka University of Technology

決定木 Decision Tree



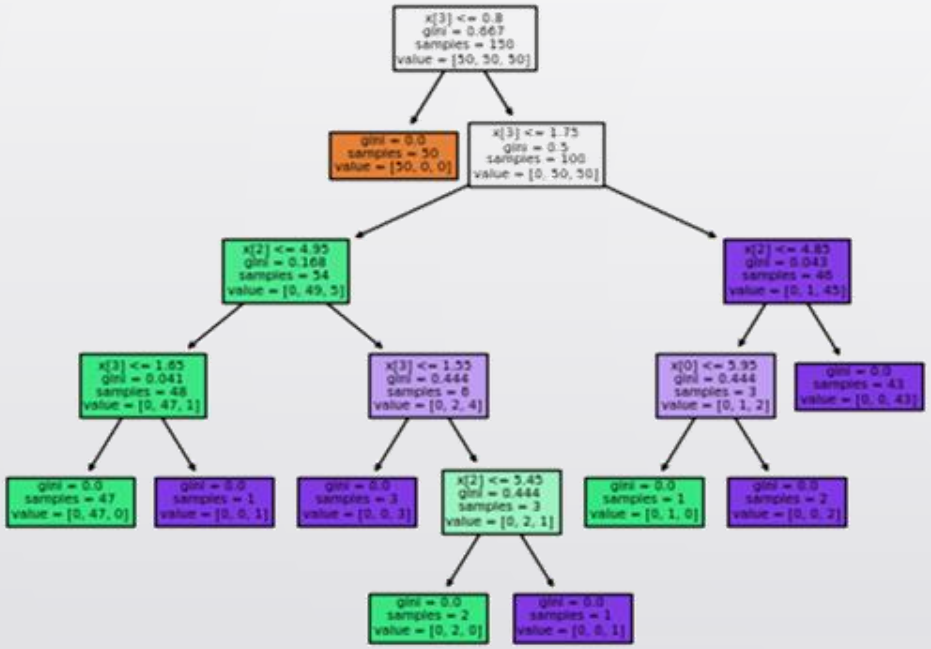
根ノード
Root Node

ノード Node

リンク Link

葉ノード Leaf Node
終端ノード
Terminal Node

分割統治法 Divide and Conquer Induction



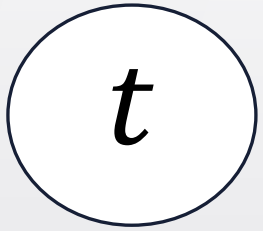
出来るだけ誤りなくデータを分類できる2分割基準でデータを分類する

Classify data by dichotomous criteria that minimizes false classification rate

この手続きを繰り返すことで決定木を成長させる

Grow decision tree by repeating this procedure

ノードのクラスの決定 Decision of the Class of a Node



$\{(x_i, t_i)\}$:学習データx_iとクラスラベルt_iの集合

C_j :クラスj

N_j :クラスjに属するデータの数

$N(t)$:ノードtに属するデータの数

$N_j(t)$:ノードtに属するクラスjのデータの数

ベイズの定理 Bayes Theorem

$$P(H \cap C) = P(H|C)P(C)$$

$$P(H \cap C) = P(C|H)P(H)$$

$$P(C|H)P(H) = P(H|C)P(C)$$

$$P(C|H) = \frac{P(H|C)P(C)}{P(H)}$$

ノードのクラスの決定 Decision of the Class of a Node

ノードtに属するデータが、クラスC_jに属する事後確率を最大化するクラスC_jを、ノードtのクラスとする

Designate class C_j as the class of node t so that posterior probability of data in node t belonging to class C_j is maximized

$$P(C_j|t) = \frac{P(t|C_j)P(C_j)}{P(t)}$$

ノードtのクラス$= argmax_j\ P(C_j|t)$

Class of node t

ノードのクラスの決定 Decision of the Class of a Node

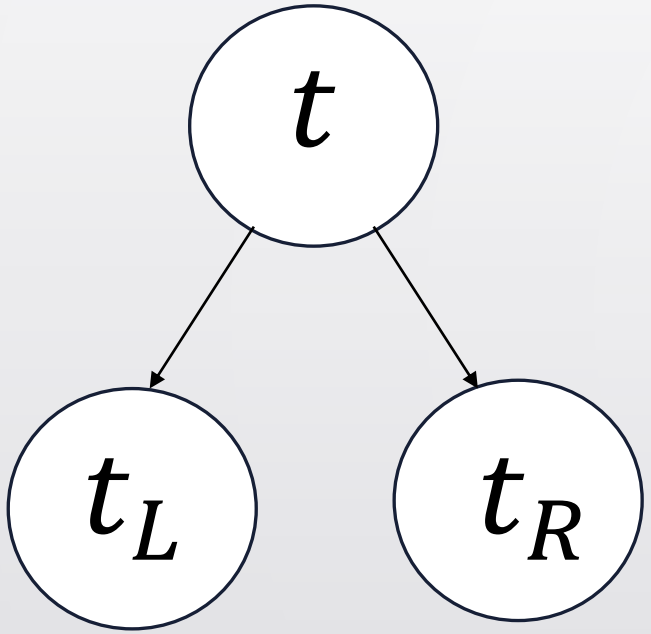
$$P(t|C_j)P(C_j) = \frac{N_j(t)}{N_j} \times \frac{N_j}{N} = \frac{N_j(t)}{N} \qquad P(t) = \frac{N(t)}{N}$$

$$P(C_j|t) = \frac{P(t|C_j)P(C_j)}{P(t)} = \frac{N_j(t)}{N} \times \frac{N}{N(t)} = \frac{N_j(t)}{N(t)}$$

$$\text{ノード}t\text{のクラス} = argmax_j\ P(C_j|t) = argmax_j\ \frac{N_j(t)}{N(t)}$$

Class of node t

ノードの不純度 Impurity of a Node



$$\Delta Impurity = Impurity_{before} - Impurity_{after}$$

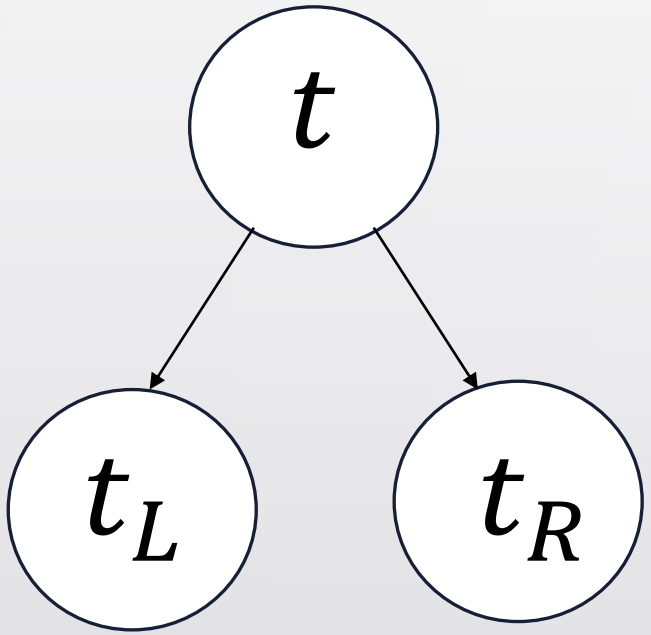
$$Impurity_{before} = Impurity(t)$$

$$Impurity_{after} = P(t_L)Impurity(t_L) + P(t_R)Impurity(t_R)$$

分割前後の不純度の減少が最大になるようにデータを2分割する

Divide data so as to maximize the decrease of impurity $\Delta Impurity$ after division

ジニ係数 Gini Index

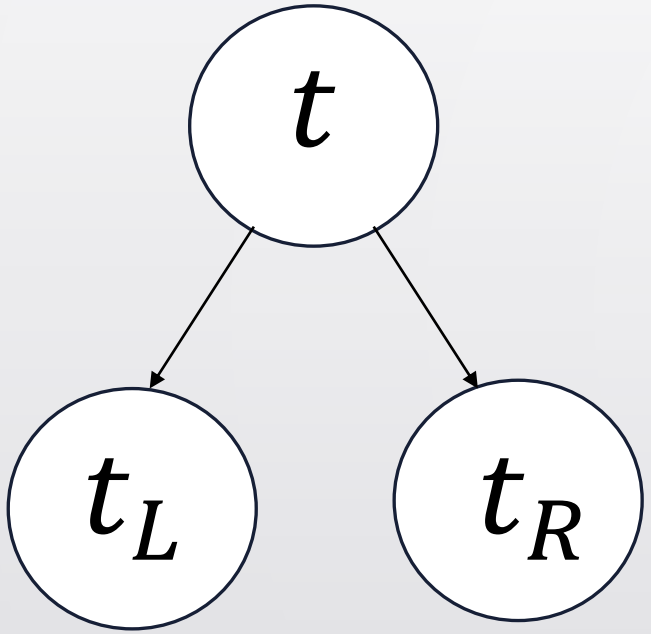


$$Gini\ Index = 1 - \sum_{1}^{K} P(C_j|t)^2$$

$P(C_j|t)$ のクラス間の違いが二乗により強調される

Inter-class difference in $P(C_j|t)$ is pronounced when squared

エントロピー Entropy



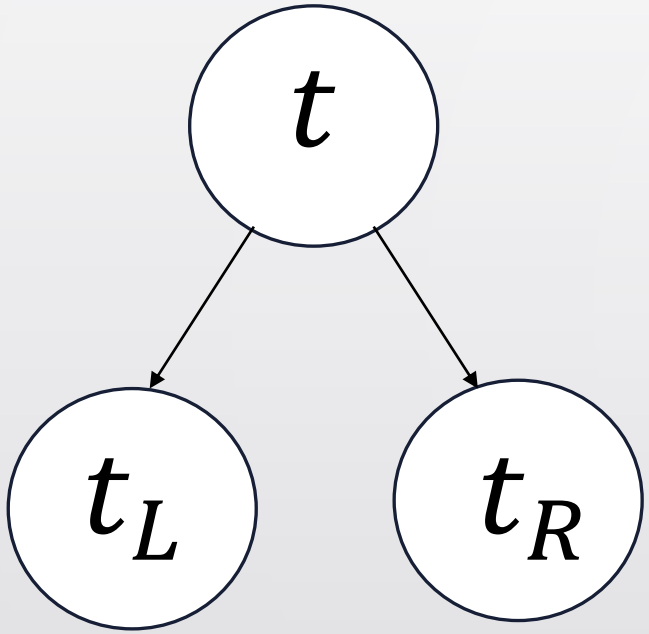
ノードtに含まれるクラスがばらついている

Class of data belonging to node t is not uniform

ノードtの不純度が高いと$P(C_j|t)$が小さくなる

$P(C_j|t)$ gets small when impurity of node t is large

エントロピー　Entropy

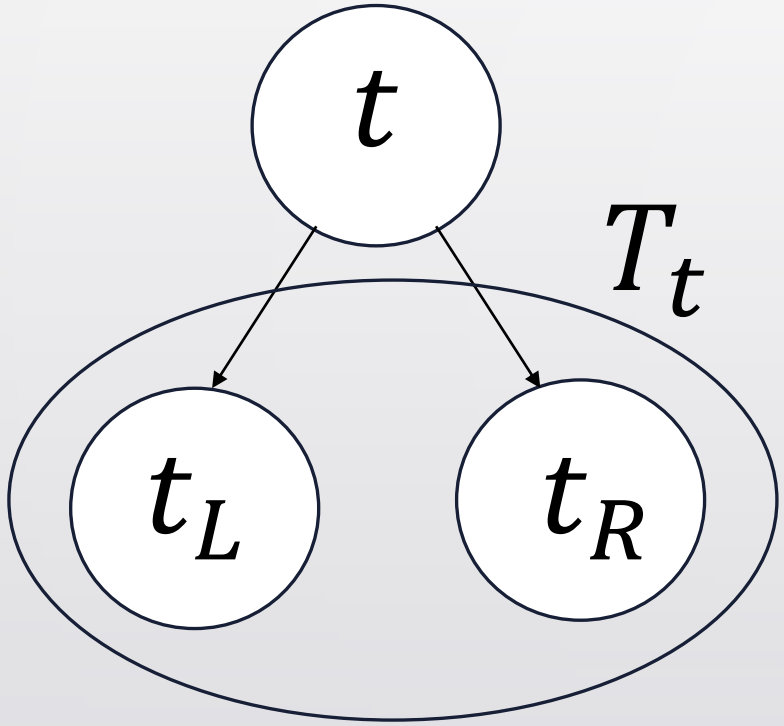


$$Entropy = -\int plog(p) = E\left[log\frac{1}{p}\right]$$

確率が低い事象が起こると大きくなる
Increases when an event with low-probability occurs

$$Impurity = -\sum_{1}^{K} P(C_j|t) log\, P(C_j|t)$$

木の剪定 Pruning Decision Tree

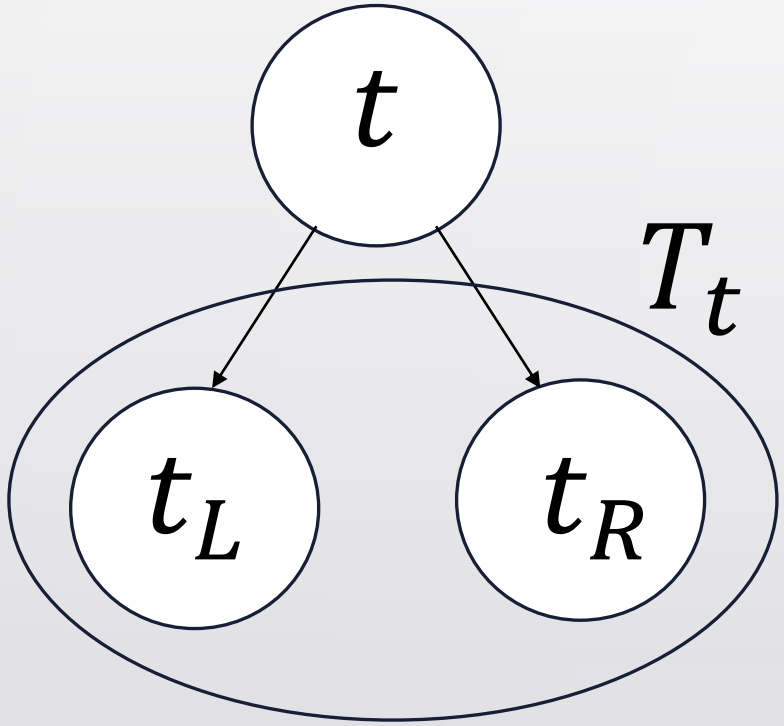


ノードtの分岐 Branch of node t

ノードtを終端ノードとすべきかどうか
＝ノードtの分岐T_tを削除すべきかどうか

The problem of whether node t should be determined as a terminal node equals to whether branch T_t of node t should be removed or not.

木のコスト Cost of Tree



ノードtの分岐 Branch of node t

$$R_\alpha(t) = R(t) + \alpha$$

ノードtにおける誤り率 Error rate at node t

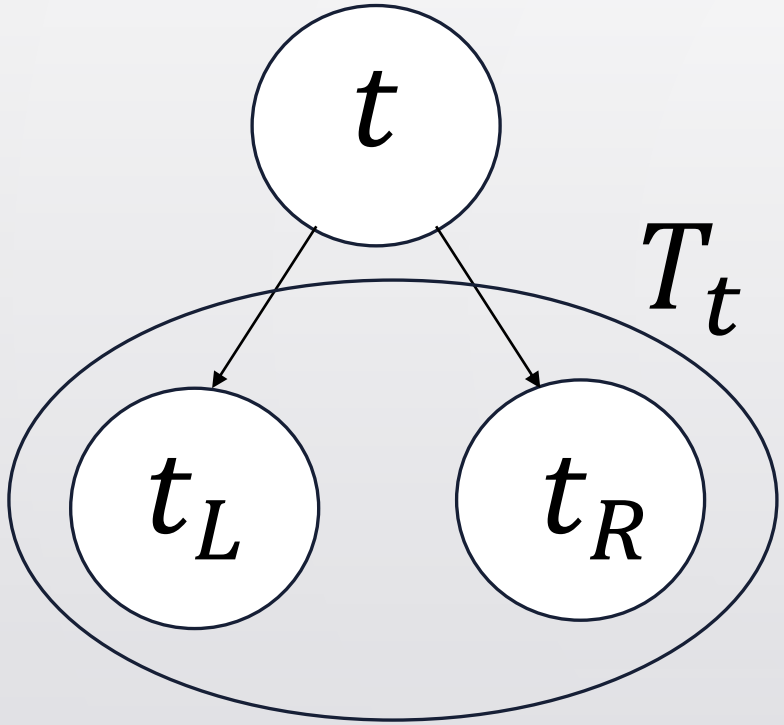
$$R_\alpha(T) = \sum_{t \in T} R_\alpha(t) = R(T) + \alpha\left|\tilde{T}\right|$$

$\left|\tilde{T}\right|$:終端ノードの数 Number of terminal node

小さい木で高い正答率を達成したい

Aims to achieve high accuracy by decision tree with smaller size

木のコスト Cost of Tree



ノードtの分岐 Branch of node t

$$R_\alpha(t) = R(t) + \alpha$$

$$R_\alpha(T_t) = R(T_t) + \alpha\left|\widetilde{T}_t\right|$$

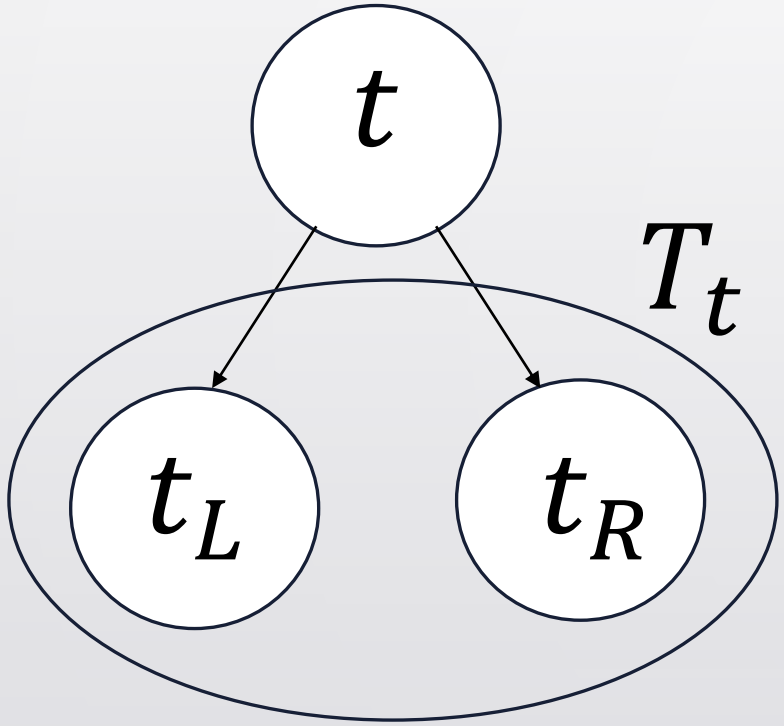
T_tは複数のノードを含むことに注意

Note that T_t includes multiple nodes

$R_\alpha(T_t) \geq R_\alpha(t)$なら$T_t$を削除する

Remove T_t if $R_\alpha(T_t) \geq R_\alpha(t)$

木のコスト Cost of Tree



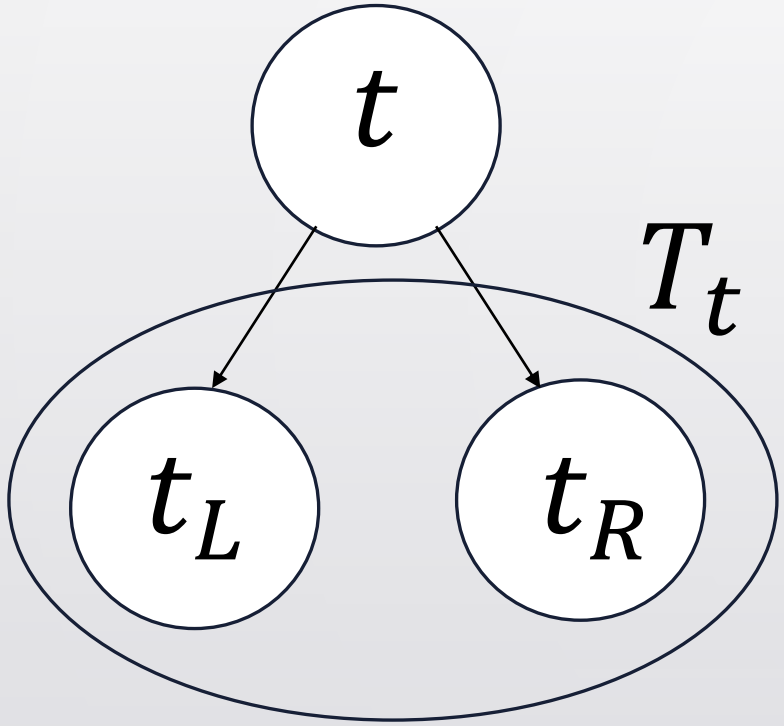
ノードtの分岐 Branch of node t

$$R(T_t) + \alpha\left|\widetilde{T}_t\right| \geq R(t) + \alpha$$

$$\frac{R(t) - R(T_t)}{|\widetilde{T}_t| - 1} \leq \alpha \text{ なら} T_t \text{を削除する}$$

Remove T_t if $\frac{R(t) - R(T_t)}{|\widetilde{T}_t| - 1} \leq \alpha$

木のコスト Cost of Tree



ノードtの分岐 Branch of node t

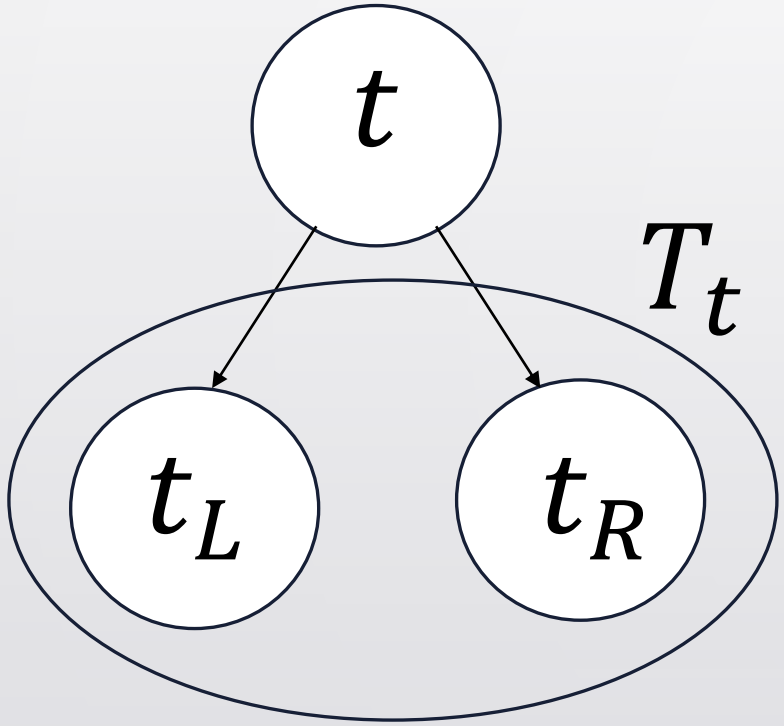
$\frac{R(t)-R(T_t)}{|\tilde{T}_t|-1} \leq \alpha$ ならT_tを削除する

$$g(t) = \frac{R(t) - R(T_t)}{|\tilde{T}_t| - 1}$$

$g(t)$は剪定するかどうかの判断基準を与える

$g(t)$ gives a criteira to decide whether to prune branch

剪定のアルゴリズム Algorithm of Pruning



ノードtの分岐 Branch of node t

- 全ての内部ノードについて$g(t)$を計算する
  Compute $g(t)$ for all the internal nodes
- $g(t)$が最小のノードの分岐を削除する
  Remove branch of the node with minimum $g(t)$
- この手続きを繰り返す
  Repeat the procedure above until certain criteria is met

アンサンブル学習 Ensemble Learning

ノーフリーランチ定理 No Free Lunch Theorem

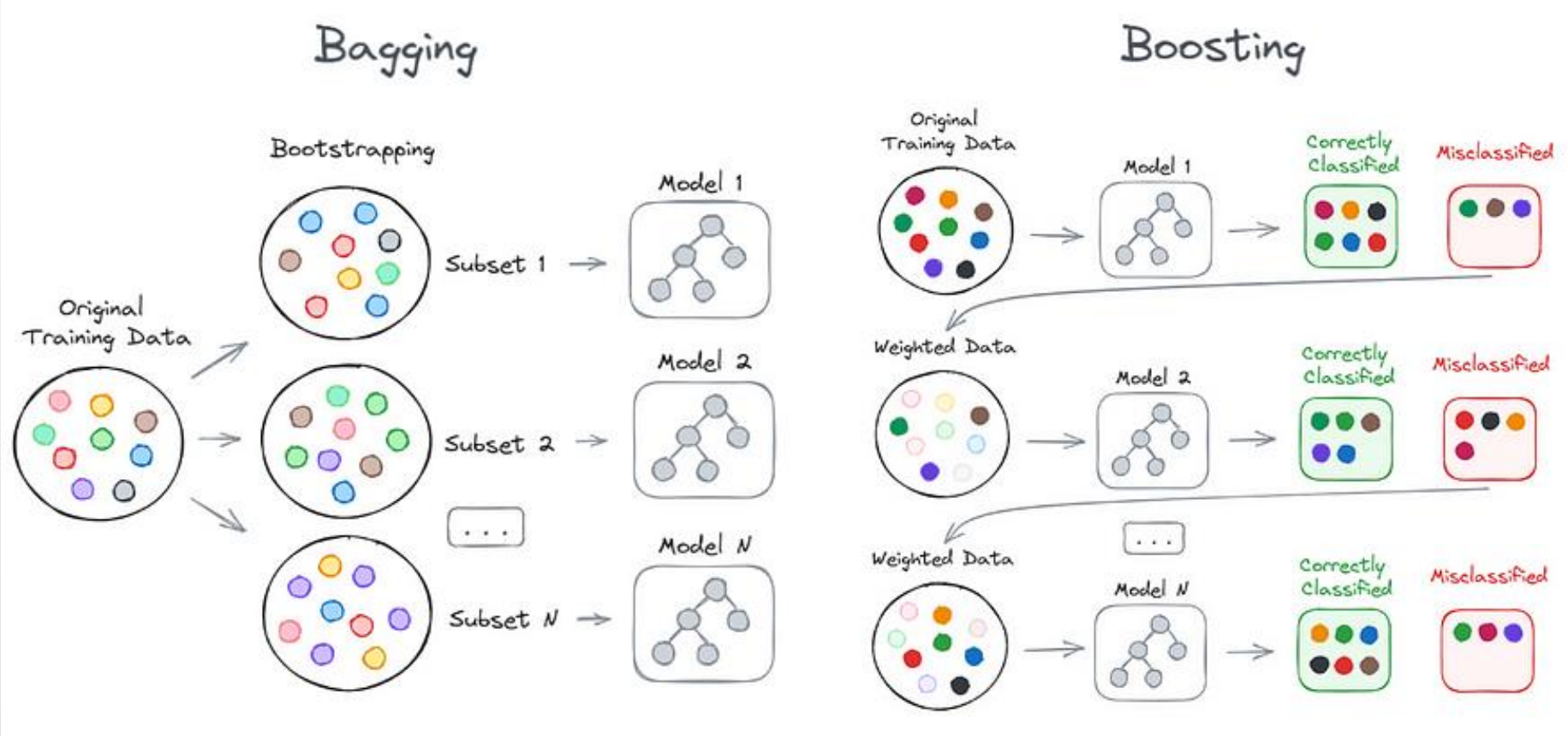
あらゆる問題に対して最適な分類器は存在しない

There is no classifier that shows best performance for every classification problem

複数の弱識別器を組み合わせることで、精度のよい分類器を作る

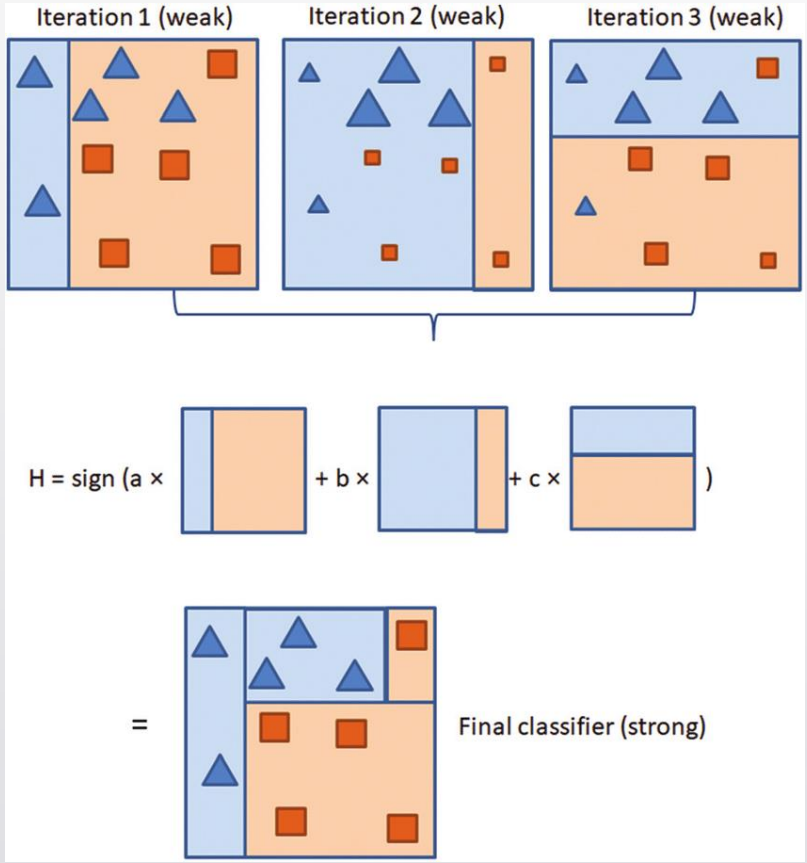
Create superior classifier by combining multiple weak classifiers

バギングとブースティング Bagging and Boosting



https://pub.towardsai.net/bagging-vs-boosting-the-power-of-ensemble-methods-in-machine-learning-6404e33524e6

アダブースト Ada(ptive)Boost(ing)



・ブースティングアルゴリズムの一つ

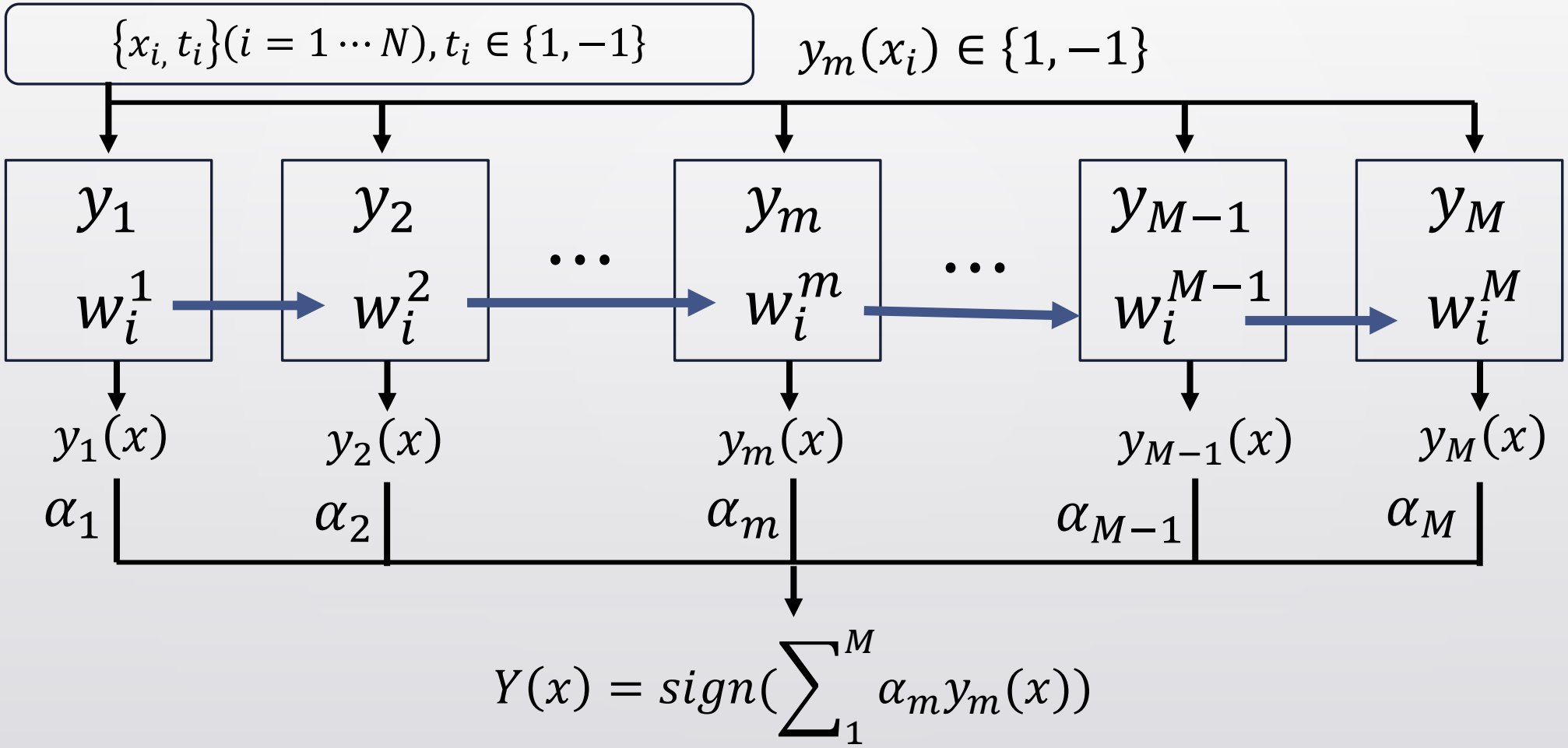
One of boosting algorithms

・t個めの弱学習器が誤識別したデータの重みを大きくして、$t+1$個めの弱学習器をトレーニングする

Train $t+1$–th weak learner by giving large weight to data points that t–th weak learner misclassified

Tang et al, 2021

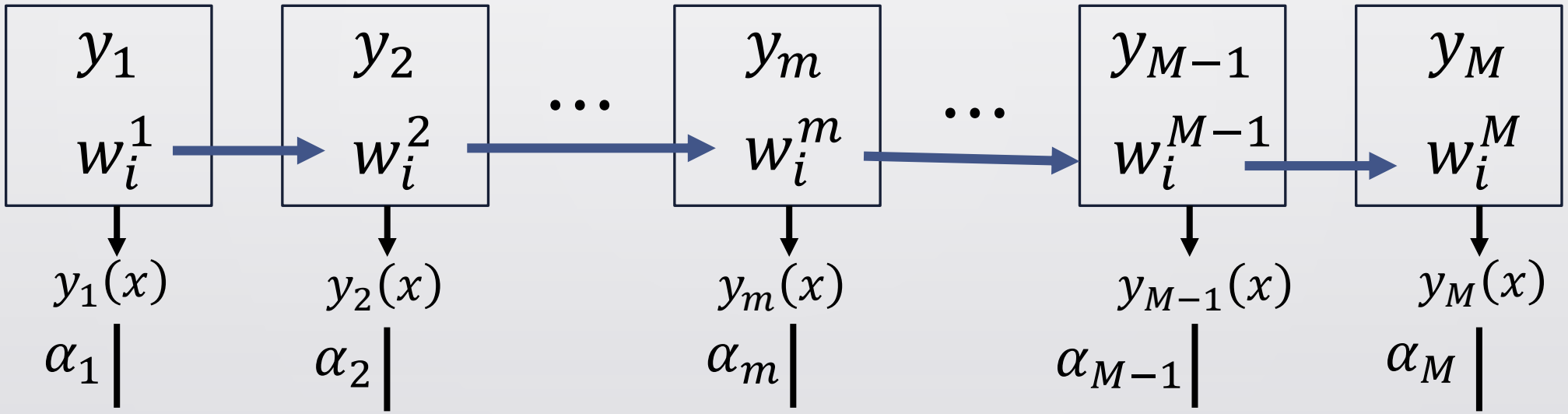
アダブースト AdaBoost



アダブースト AdaBoost

弱識別器y_mを逐次的に訓練する

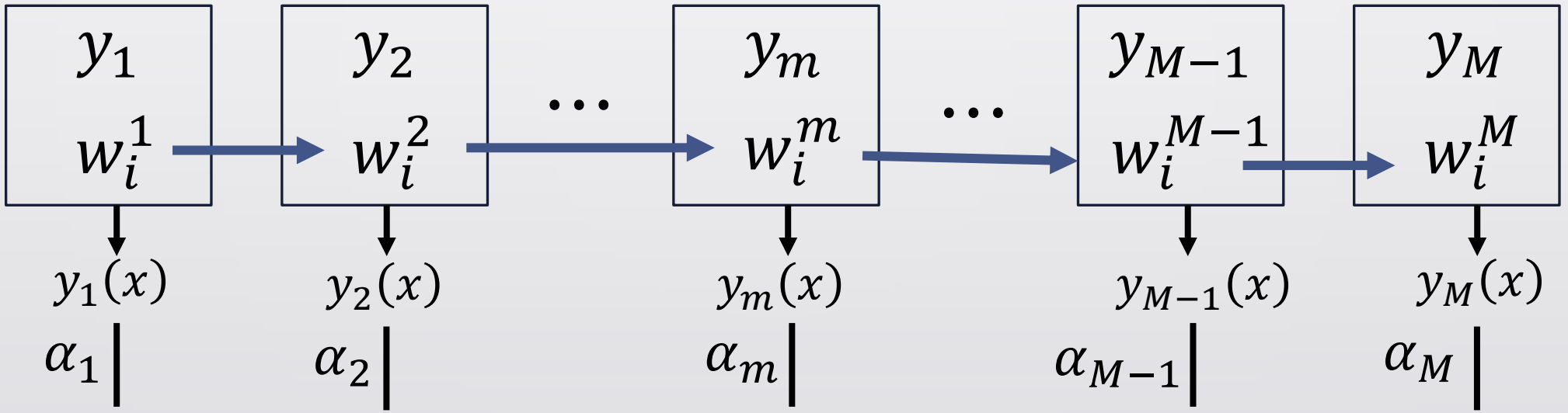
Train weak learners y_m sequentially



アダブースト AdaBoost

データの重みw_i^mと弱学習器の重みα_mが更新される

Update weight of data w_i^m and weight of weak learner α_m



アダブースト AdaBoost

E_mが最小になるよう弱識別器y_mを訓練する

Train weak learners y_m so that E_m is minimized

$$E_m = \frac{\sum_1^N w_i^m I\left(y_m(x_i)\right)}{\sum_1^N w_i^m}$$

$$I\left(y_m(x_i)\right) = \begin{cases} 1 & (y_m(x_i) \neq t_i) \\ 0 & (y_m(x_i) = t_i) \end{cases}$$

アダブースト AdaBoost

データの重みw_i^mと弱学習器の重みα_mが更新される

Update weight of data w_i^m and weight of weak learner α_m

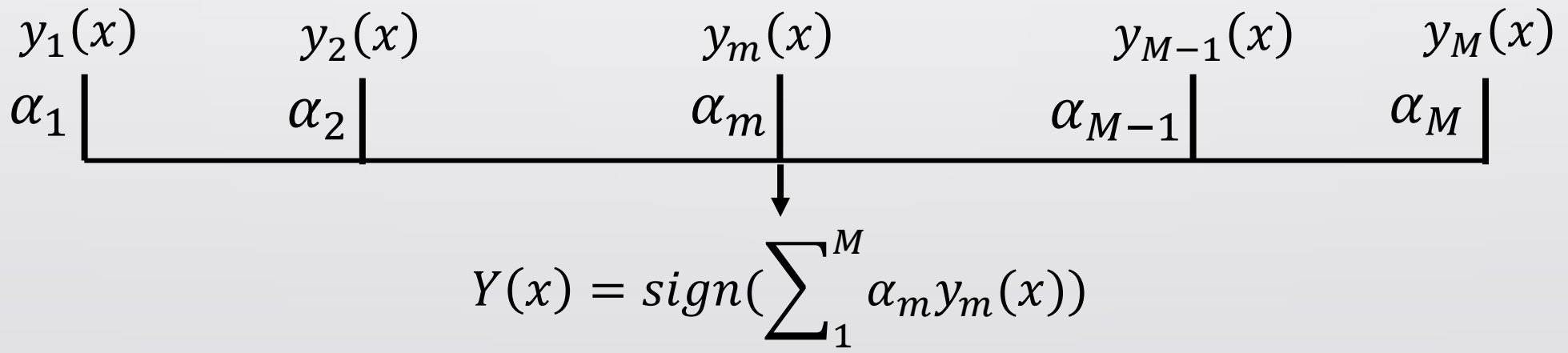
$$\alpha_m = ln\left(\frac{1}{E_m} - 1\right) \geq 0$$

$$w_i^{m+1} = \begin{cases} w_i^m exp(\alpha_m) \geq w_i^m \ (y_m(x_i) \neq t_i) \\ w_i^m \quad (y_m(x_i) = t_i) \end{cases}$$

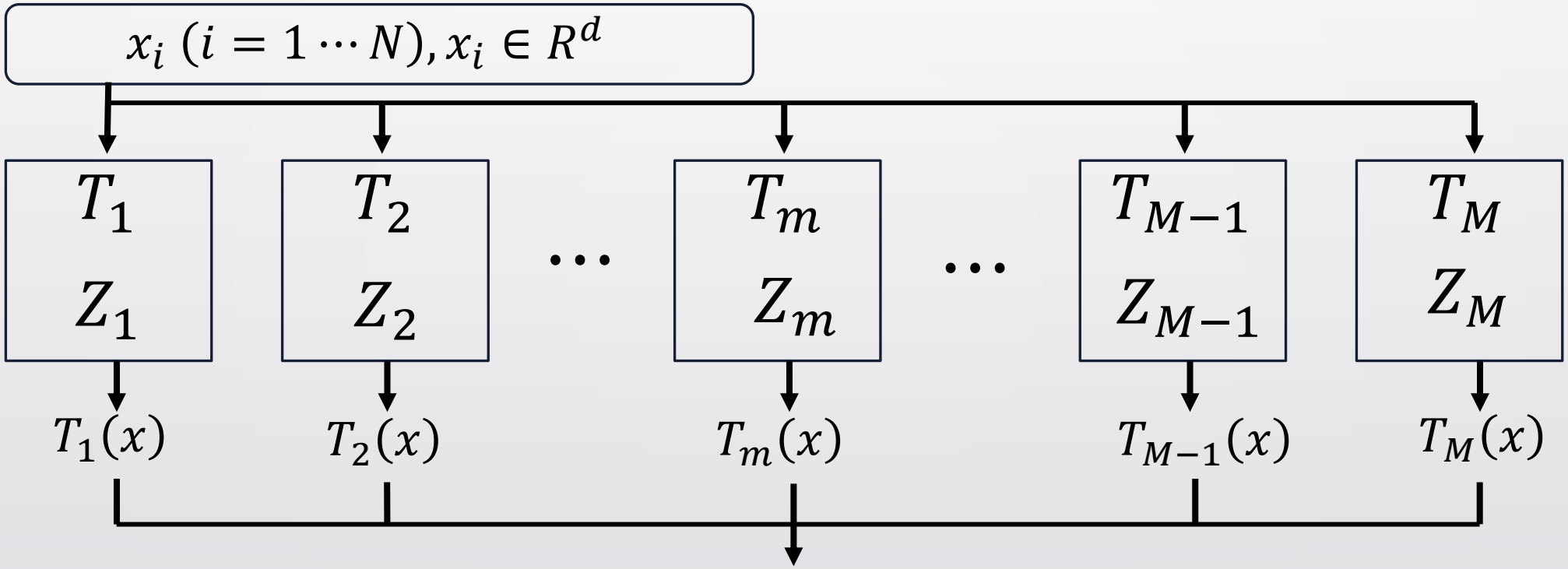
アダブースト AdaBoost

最終的な出力は、弱学習器の出力の重み付き和によって決まる

Output of AdaBoost classifier is determined by weighted sum of output of each weak classifier



ランダムフォレスト Random Forest



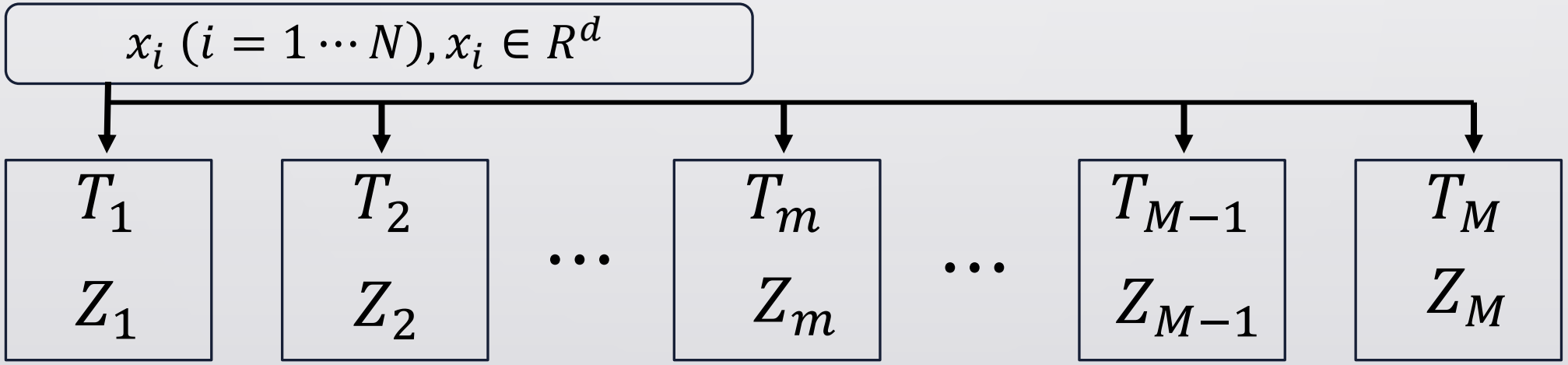
ランダムフォレスト Random Forest

データからブートストラップサンプルZ_mを抽出する

Extract bootstrapped samples Z_m from dataset

ランダムに選択したd'次元の特徴量を使って決定木T_mを構成する

Grow decision tree T_m based on randomly-selected d'–dimensional features



ランダムフォレスト Random Forest