DECISION TREE

Week10

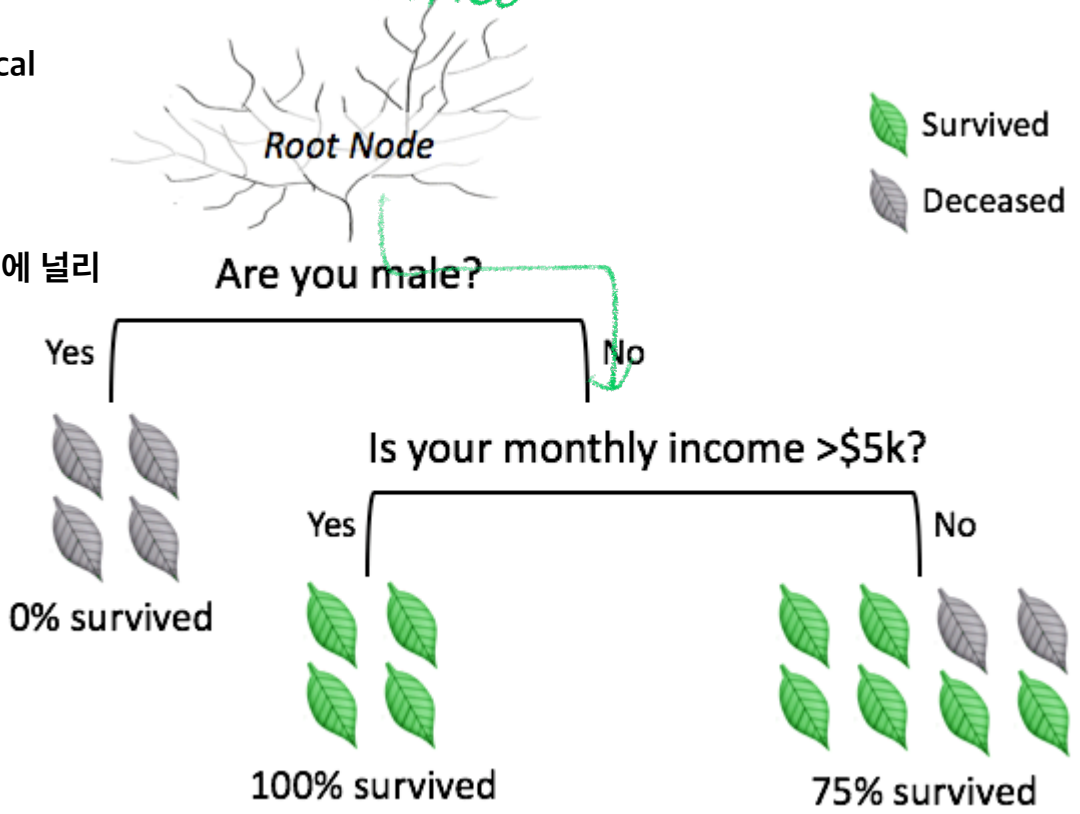
Decision Tree

Introduction to Decision Trees

- Decision tree
  - A decision tree is a hierarchical model used for classification and regression
  - It consists of nodes representing decisions, branches representing choices, and leaves representing final outcomes
  - Commonly used due to its interpretability and ease of implementation

의사결정 나무는 classification와 regression에 사용되는 hierarchical model 계층적 모델이다.
- **node: 결정을 나타냄**
- **branch: 선택을 나타냄**
- **leaf: 최종 결과를 나타냄**

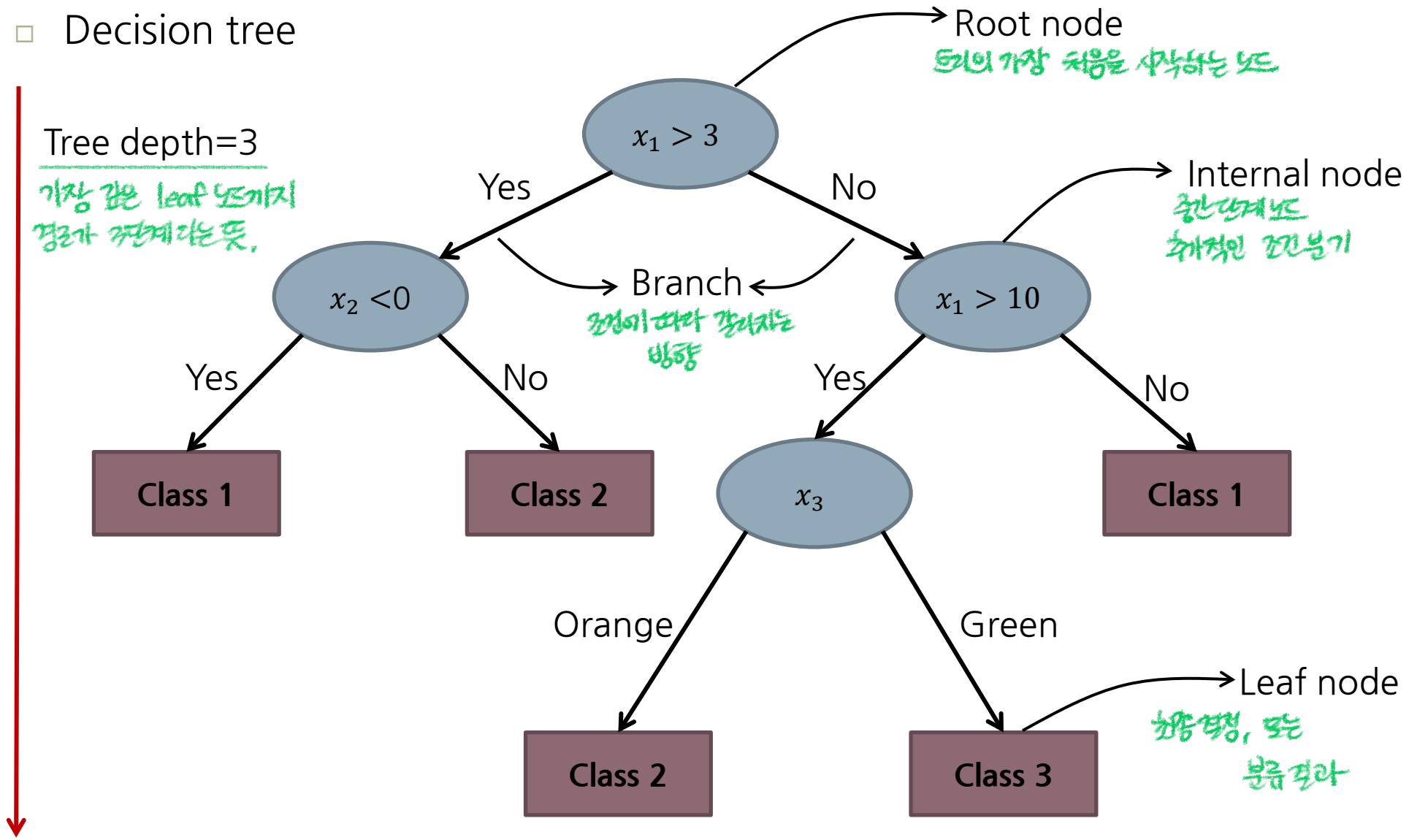
해석이 쉬우며 구현이 간단하기 때문에 널리 사용됨



Decision Tree

조건 기반 분류를 통해 최종 클래스 예측

- Decision tree



Decision Tree

- Each root node and internal node represent a specific input variable
  - Root and internal node tests each attribute
    - $x_1 > 1$
    - x_3 is orange

각 루트 노드와 내부 노드는 특정 입력 변수를 나타냄

루트 및 내부 노드는 각 속성을 테스트함

- Each branch corresponds to the result of the test of node
  - Yes/No
  - Values of attribute
    - Orang/Green
    - Long/Short

각 branch는 노드의 테스트 결과에 따라 나뉨

- Each leaf node assigns a class

각 leaf node는 하나의 클래스를 할당

각 노드에서 어떻게 속성을 선택하고 branch 기준을 정할까?

In each node, how to choose attribute?

how to split branches?

Decision Tree Algorithm

How to determine which one is the most effective?

⬇

Need some criteria for measure of effectiveness

효율성의 척도

Splitting Criteria

- Categorical target - Classification
  - Entropy 데이터의 불확실성(uncertainty)또는 혼잡도를 측정하는 지표
    엔트로피를 최대한 줄이는 방향으로 노드를 분기
  - Gini impurity 지니 불순도, 노드 안에 클래스가 섞여있는 정도를 측정
    Gini 값이 작아지는 방향으로 분기

- Continuous target - Regression
  - MSE 각 예측값과 실제값의 차이를 제곱한 평균, 분할 후 MSE가 가장작아지는 분기 선택
  - Friedman MSE 좌/우 자식 노드의 평균값 차이 제곱에 기반한 간접 오차 척도
    응집력 있는 평균값 분리 극대화
  - MAE 예측값과 실제값의 차이의 절대값 평균, outlier에 덜 민감

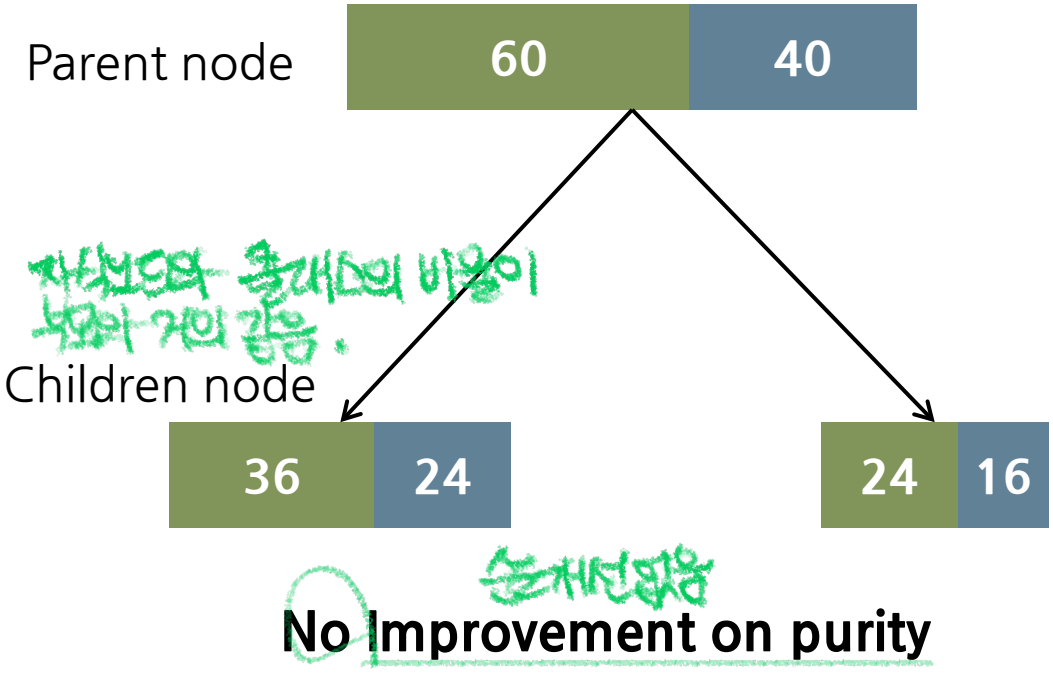
$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$$

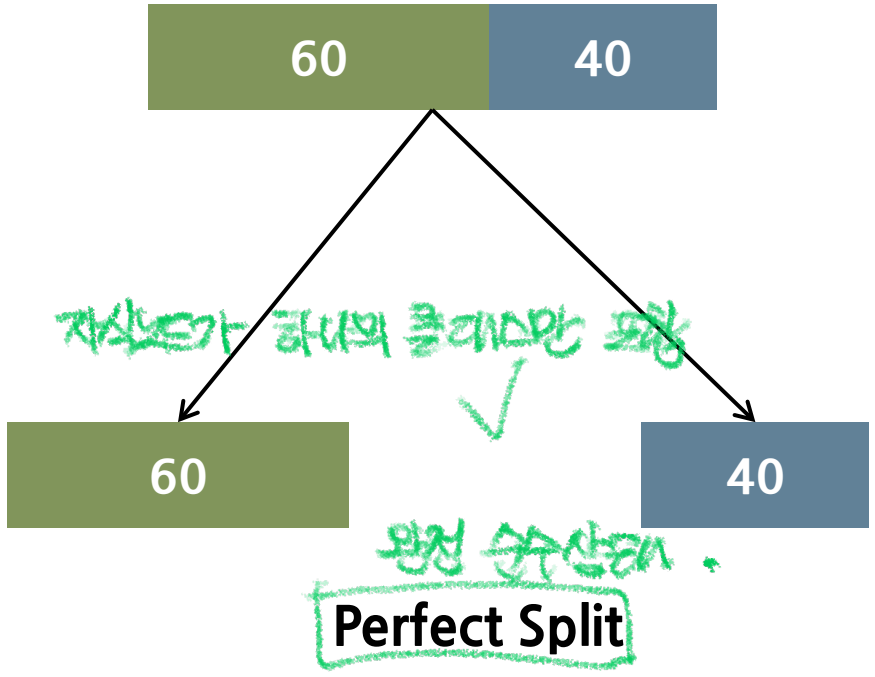
$$\text{Friedman MSE} = \frac{N_L N_R}{(N_L + N_R)^2}(\bar{y}_L - \bar{y}_R)^2$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|$$

Splitting Criteria for Classification: Purity

- Select each split of a node so that in each of the child nodes are purer or less impure than that in the parent node





Entropy와 Gini impurity는 순도의 반대 개념인 불순도를 측정하는 개념

- Entropy and Gini impurity are measures of impurity
  - Split a node toward decreasing impurity → Maximize reduction in impurity

불순도가 가장 많이 감소하는 방향으로 분할

Splitting Criteria for Classification: Entropy

- Expected value of the information
  - The less likely an event is, the more information it provides when it occurs.
  - Information ↔ uncertainty

어떤 사건이 일어날 가능성이 낮을수록, 실제로 그것이 발생했을때 더 많은 정보를 제공한다.
정보량과 불확실성은 서로 밀접하게 연관되어 있음

- Entropy H of event X

$$H(X) = E[I(X)] = E[-\ln P(X)]$$

Information content of X

Expectation value

Probability function

$I(X) = -\ln P(X)$

확률이 낮을 수록 정보량이 크고 엔트로피도 증가

- For a finite sample

전체 엔트로피는 각 클래스의 확률과 정보량을 곱해 모두 더한것

$$H(X) = \sum_i P(x_i) I(x_i) = -\sum_i P(x_i) \log_b P(x_i)$$

  - X with possible values of $\{x_1, x_2, \ldots, x_n\}$
  - Commonly b is 2 (10, e are also used)

$I(x_i) = \log_b P(x_i)$

Example: Entropy

- When you flip one coin(X)
  - Possible output of X: (H), (T)
  - $P(H) = 0.5, P(T) = 0.5$

$$\begin{aligned} H(X) &= -P(H)\log_2 P(H) - P(T)\log_2 P(T) \\ &= -0.5\log_2 0.5 - 0.5\log_2 0.5 \\ &= 1 \end{aligned}$$

- When you flip two coins (X)
  - Possible output of X: (H,H), (H,T), (T,H), (T,T)
  - $P(H,H) = P(H,T) = P(T,H) = P(T,T) = 0.25$

$$\begin{aligned} H(X) &= -P(H,H)\log_2 P(H,H) - P(H,T)\log_2 P(H,T) - P(T,H)\log_2 P(T,H) - P(T,T)\log_2 P(T,T) \\ &= -4 \times 0.25\log_2 0.25 \\ &= 2 \end{aligned}$$

How to Define Effectiveness of Split

- If split is effective, information gain is large
  - Information gain=reduction of uncertainty

$$IG(T, a) = H(T) - H(T|a)$$

Information gain with the split on attribute a
속성 a로 분할했을 때의 정보 이득

Entropy of original set
전체 데이터 T의 엔트로피 (분할 전)

Entropy of new state after split
분할 후의 엔트로피

  - Entropy of new state after split=normalized sum of entropy of split sets

$$H(T|a) = \sum_{i}^{n} \frac{|T_i'|}{|T|} \cdot H(T_i')$$

    - T is split to $T_1', T_2', \ldots, T_n'$

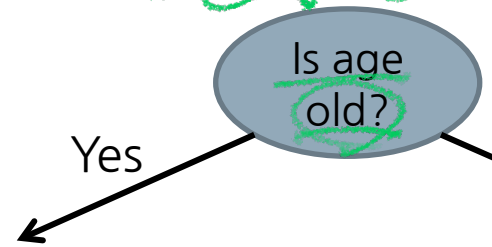
Example: Calculate Information Gain through Entropy

- The node is split by age to predict profit of company

Age(x)	old	old	old	mid	mid	mid	mid	new	new	new
Profit(y)	down	down	down	down	down	up	up	up	up	up

$$H(T) = -P(down)\log_2 P(down) - P(up)\log_2 P(up)$$
$$= -2 \times 0.5 \log_2 P(0.5) = 1$$

P(down) = P(up) = 0.5



P(down) = 1

Age(x)	old	old	old
Profit(y)	down	down	down

$$H(T_1') = -1\log_2 1 = 0$$

P(down) = 2/7
P(up) = 5/7

Age(x)	mid	mid	mid	mid	new	new	new
Profit(y)	down	down	up	up	up	up	up

$$H(T_2') = -\frac{2}{7}\log_2\frac{2}{7} - \frac{5}{7}\log_2\frac{5}{7} \approx 0.86$$

$$IG = H(\text{before}) - H(\text{after}) = 1 - 0.86 \times \frac{7}{10} = 1 - 0.602 = 0.398$$

Information gain

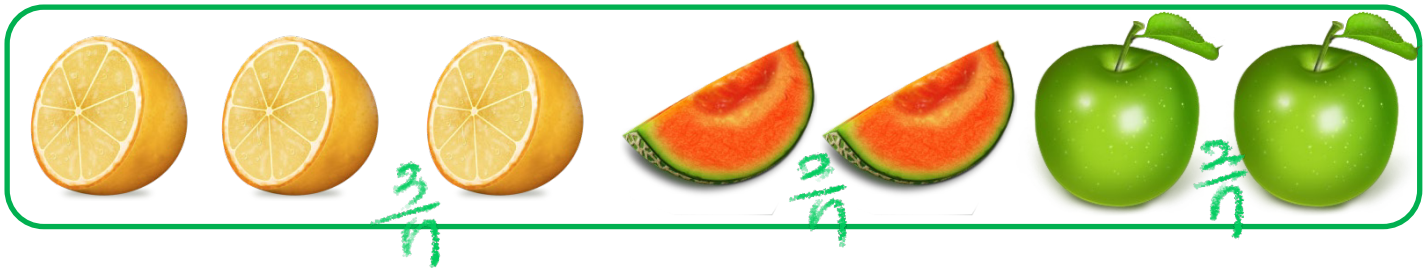
Splitting Criteria for Classification: Gini Impurity

- Gini Impurity: measure of impurity 데이터가 얼마나 섞여 있는지를 측정하는 지표

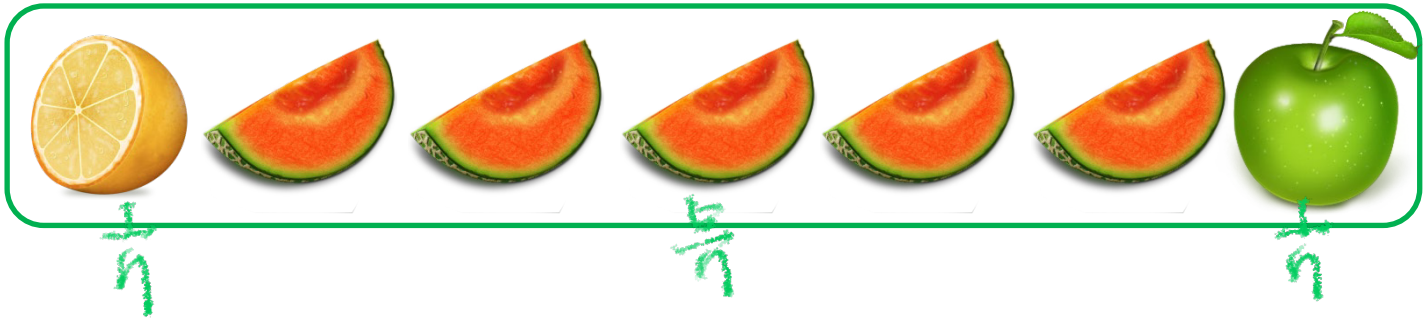
$$G(T) = \sum_{i \neq j} P(i|T)P(j|T) = 1 - \sum_{j} P(j|T)^2 = 1 - \sum_{j} \left(\frac{n_j(T)}{n(T)}\right)^2$$

  - $P(j|t)$ is the probability of output j in node T
  - $n(t)$ is the total number of samples in node T
  - $n_j(t)$ is the number of samples with output j in node T

값이 작을 수록 더 순숫한 상태 즉 하나의 클래스에 가깝다는 뜻
지니 불순도가 가장 작아지는 방향으로 노드를 분할



$$G = 1 - \left(\frac{3}{7}\right)^2 - \left(\frac{2}{7}\right)^2 - \left(\frac{2}{7}\right)^2 = \frac{32}{49}$$



$$G = 1 - \left(\frac{1}{7}\right)^2 - \left(\frac{5}{7}\right)^2 - \left(\frac{1}{7}\right)^2 = \frac{22}{49}$$

Example: Calculate Information Gain through Gini Impurity

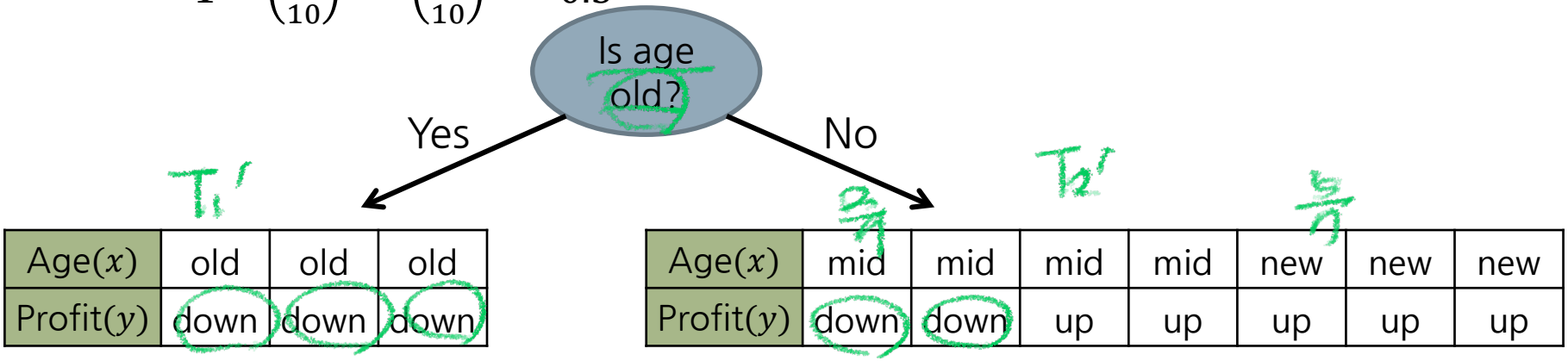
- The node is split by age to predict profit of company

Age(x)	old	old	old	mid	mid	mid	mid	new	new	new
Profit(y)	down	down	down	down	down	up	up	up	up	up

$$G(T) = 1 - P^2(down) - P^2(up)$$

$$= 1 - \left(\frac{5}{10}\right)^2 - \left(\frac{5}{10}\right)^2 = 0.5$$

P(down) = P(up) = 0.5



T_1'

Age(x)	old	old	old
Profit(y)	down	down	down

$$G(T_1') = 1 - \left(\frac{3}{3}\right)^2 = 0$$

T_2'

Age(x)	mid	mid	mid	mid	new	new	new
Profit(y)	down	down	up	up	up	up	up

$$G(T_2') = 1 - \left(\frac{2}{7}\right)^2 - \left(\frac{5}{7}\right)^2 \approx 0.41$$

$$IG = G(\text{before}) - G(\text{after}) = 0.5 - 0.41 \times \frac{7}{10} = 0.5 - 0.287 = 0.213$$

Information gain

Question

- Predict profit of companies based on age of company, type of company, and competition status

Age	Competition	Type	Profit
old	yes	S/W	down
old	no	S/W	down
old	no	H/W	down
mid	yes	S/W	down
mid	yes	H/W	down
mid	no	H/W	up
mid	no	S/W	up
new	yes	S/W	up
new	no	H/W	up
new	no	S/W	up

1) How much information gain based on Entropy is obtained with the splitting on competition?

2) How much information gain based on Gini impurity is obtained with the splitting on type of company?

1)

전체 $H(T) = -0.5\log_2 0.5 - 0.5\log_2 0.5 = 1$

$H(T_{yes}) = -0.75\log_2 0.75 - 0.25\log 0.25 \simeq 0.811$

$H(T_{no}) = -\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3} \simeq 0.918$

$H(T|a) = \frac{4}{10}\cdot 0.811 + \frac{6}{10}\times 0.918 \simeq 0.875$

$IG = H(T) - H(T|a) = 1 - 0.875 = 0.125$

2)

$G(T) = 1 - (0.5)^2 - (0.5)^2 = 0.5$

$G(after) = 1 - (0.5)^2 - (0.5)^2 = 0.5$

$IG = G(T) - G(after) = 0.5 - 0.5 = 0$

SW < down 3, up 3

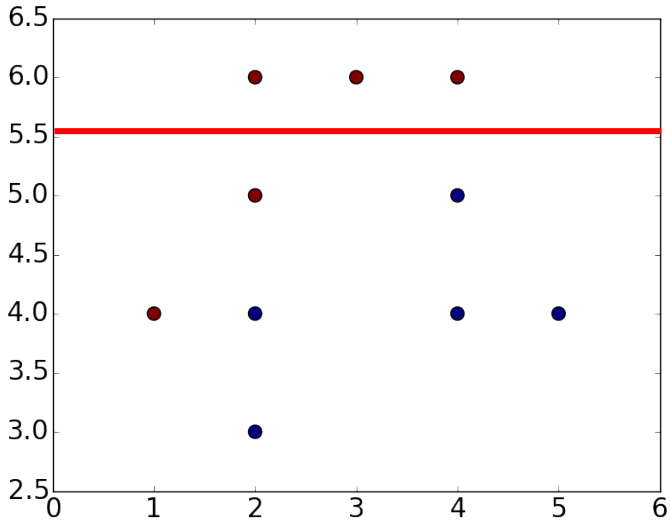
HW < down 2, up 2

비록 분할 수행 각 그룹의 gini impurity가 그대로이기 때문에 예측력이 전혀 향상되지 X

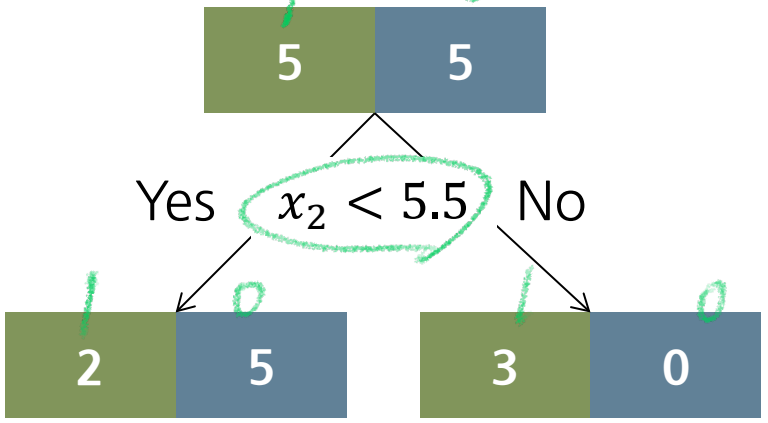
Simple Example for Tree

Decision Tree에서 지니 불순도 기준으로 분할할 때정보이득을 계산하는과정

Class	x_1	x_2
1	1	4
1	2	6
1	2	5
0	2	4
0	2	3
1	3	6
1	4	6
0	4	5
0	4	4
0	5	4



$$G = 1 - \left(\frac{5}{10}\right)^2 - \left(\frac{5}{10}\right)^2 = 0.5$$



$$G = 1 - \left(\frac{2}{7}\right)^2 - \left(\frac{5}{7}\right)^2 \approx 0.4083 \qquad G = 1 - \left(\frac{3}{3}\right)^2 - \left(\frac{0}{3}\right)^2 = 0$$

$$IG = 0.5 - 0.408 \times \frac{7}{10} - 0 \times \frac{3}{10} = 0.2144$$

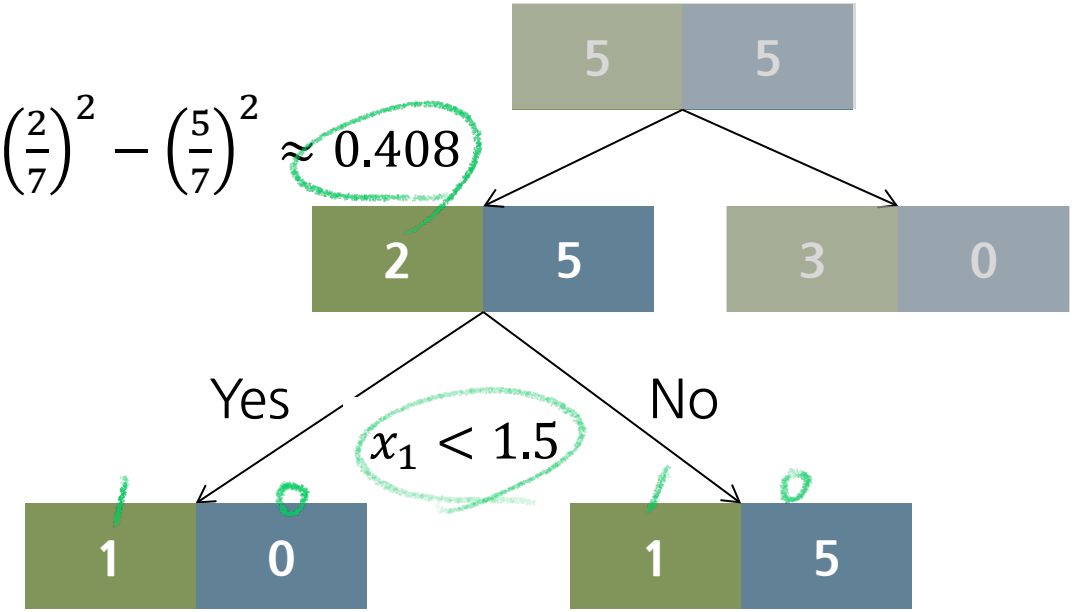
before

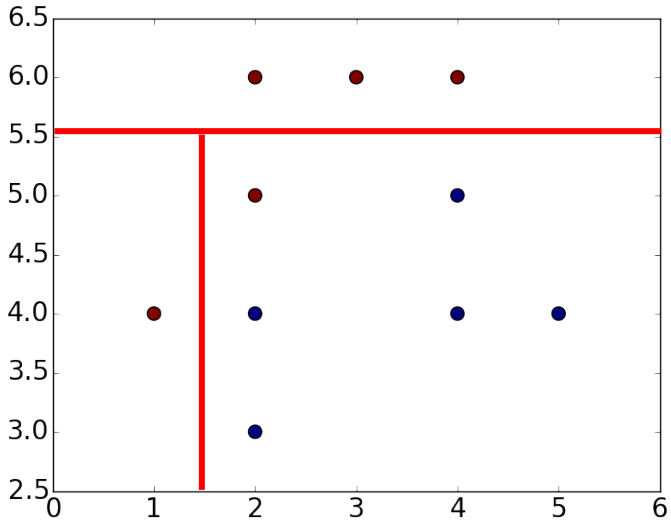
after

Simple Example for Tree

Class	x_1	x_2
1	1	4
1	2	6
1	2	5
0	2	4
0	2	3
1	3	6
1	4	6
0	4	5
0	4	4
0	5	4

$$G = 1 - \left(\frac{2}{7}\right)^2 - \left(\frac{5}{7}\right)^2 \approx 0.408$$



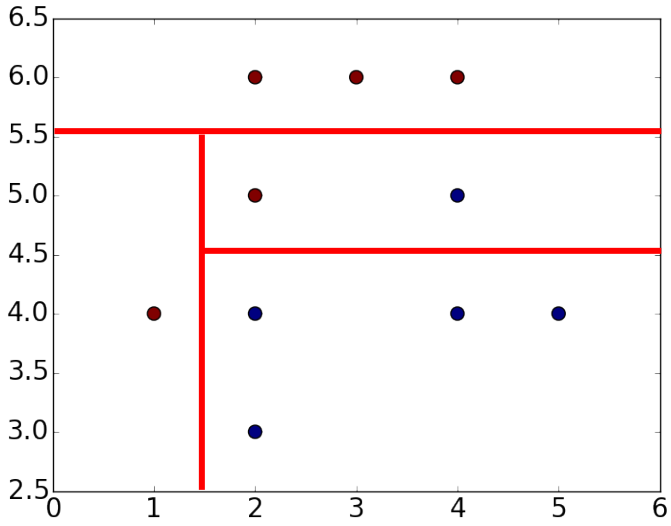


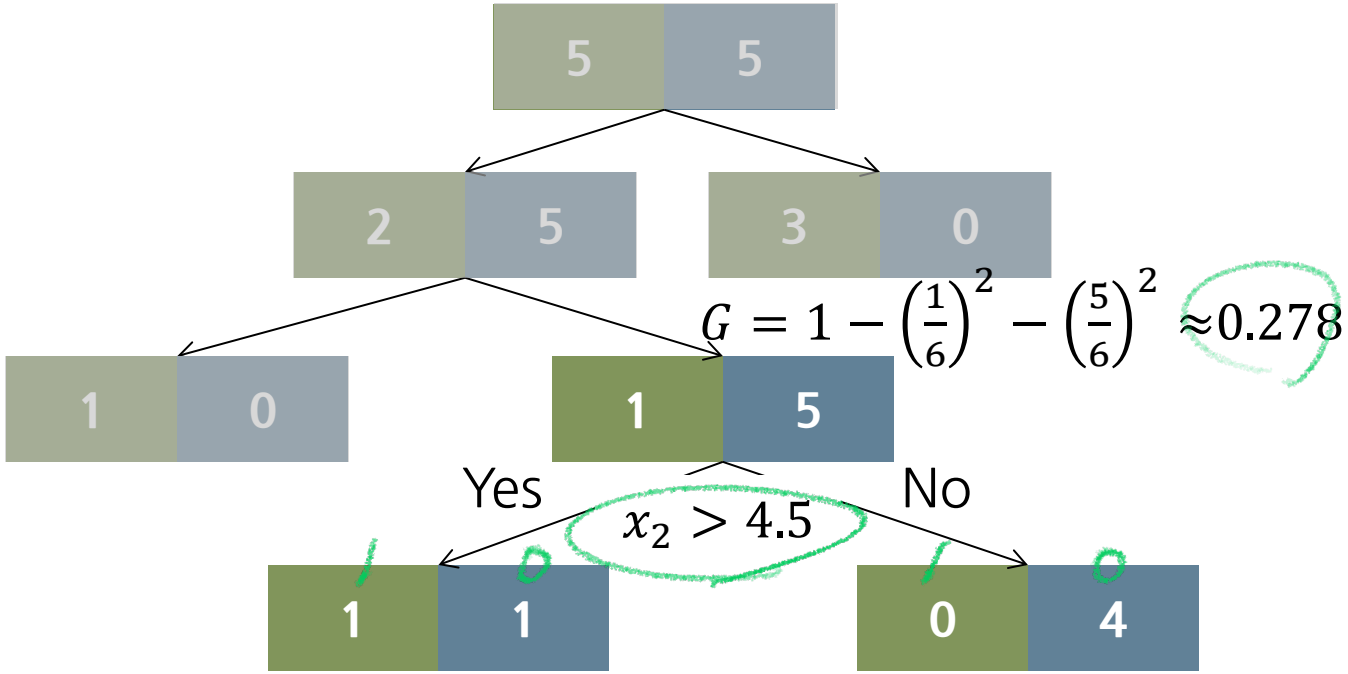
$$G = 1 - \left(\frac{1}{1}\right)^2 - \left(\frac{0}{1}\right)^2 = 0 \qquad G = 1 - \left(\frac{1}{6}\right)^2 - \left(\frac{5}{6}\right)^2 \approx 0.278$$

$$IG = \left(0.408 - 0 \times \frac{1}{7} - 0.278 \times \frac{6}{7}\right) \times \frac{7}{10} \approx 0.11$$

Simple Example for Tree

Class	x_1	x_2
1	1	4
1	2	6
1	2	5
0	2	4
0	2	3
1	3	6
1	4	6
0	4	5
0	4	4
0	5	4





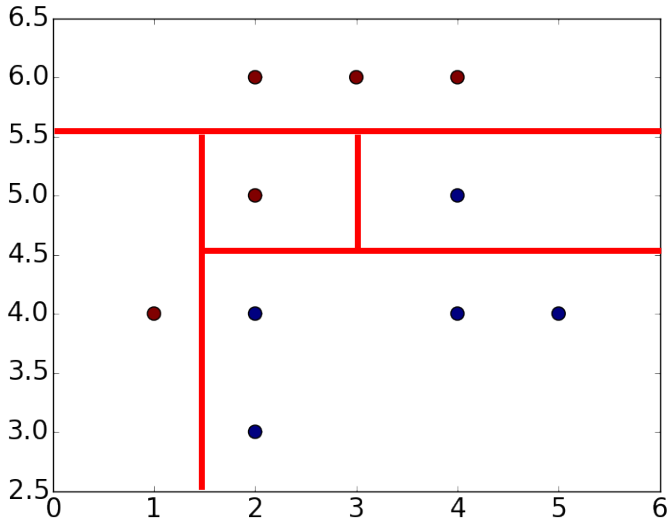
$$G = 1 - \left(\frac{1}{6}\right)^2 - \left(\frac{5}{6}\right)^2 \approx 0.278$$

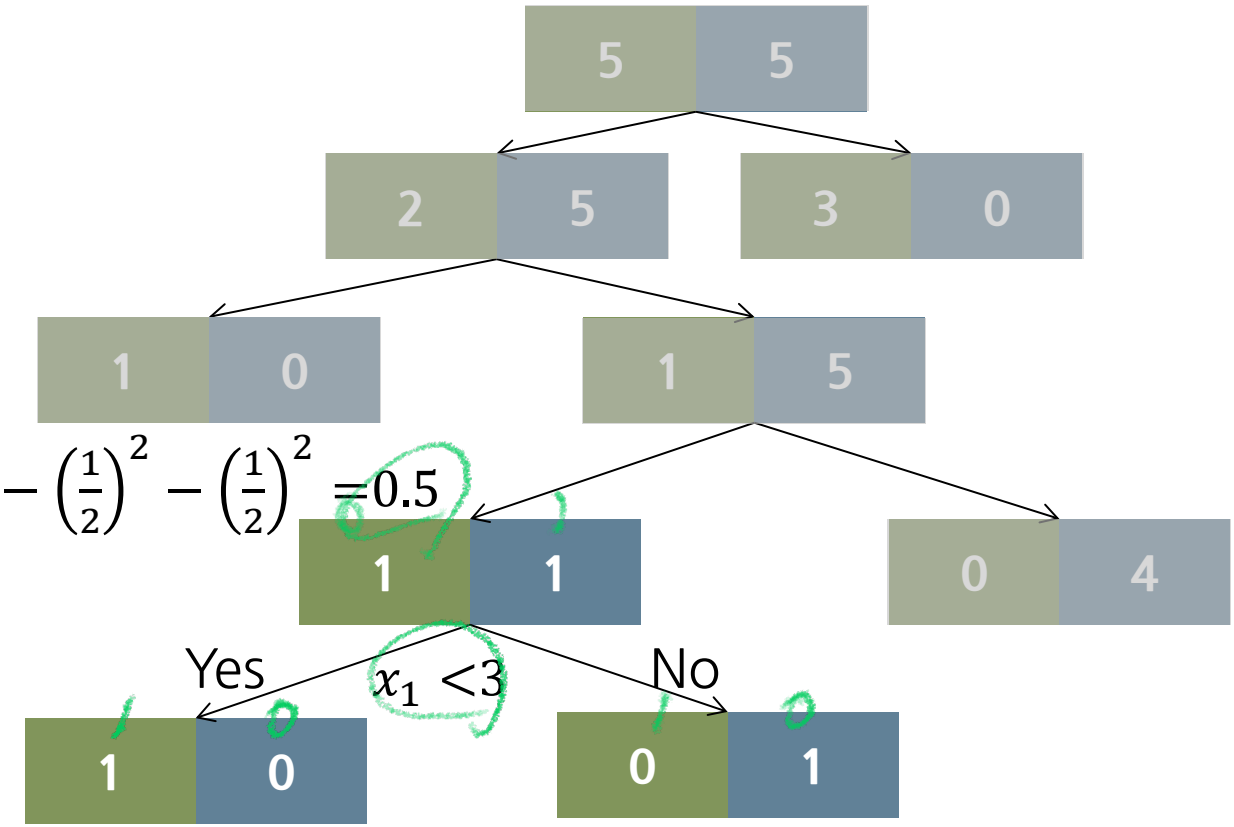
$$G = 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2 = 0.5 \qquad G = 1 - \left(\frac{0}{4}\right)^2 - \left(\frac{4}{4}\right)^2 = 0$$

$$IG = \left(0.278 - 0.5 \times \frac{2}{6} - 0 \times \frac{4}{6}\right) \times \frac{6}{10} \approx 0.067$$

Simple Example for Tree

Class	x_1	x_2
1	1	4
1	2	6
1	2	5
0	2	4
0	2	3
1	3	6
1	4	6
0	4	5
0	4	4
0	5	4



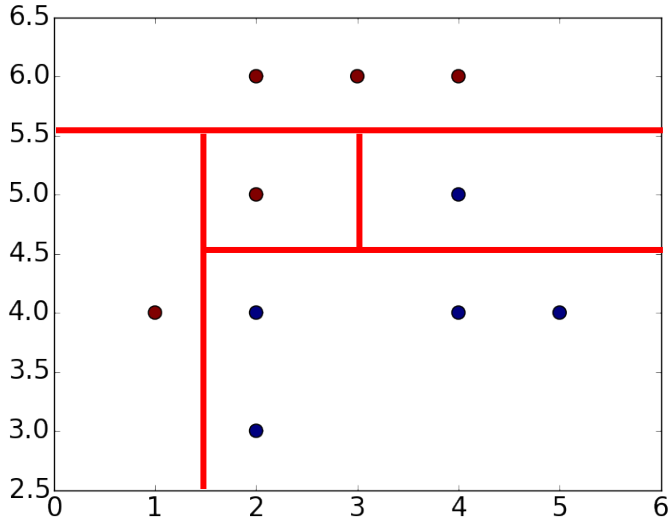


$$G = 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2 = 0.5$$

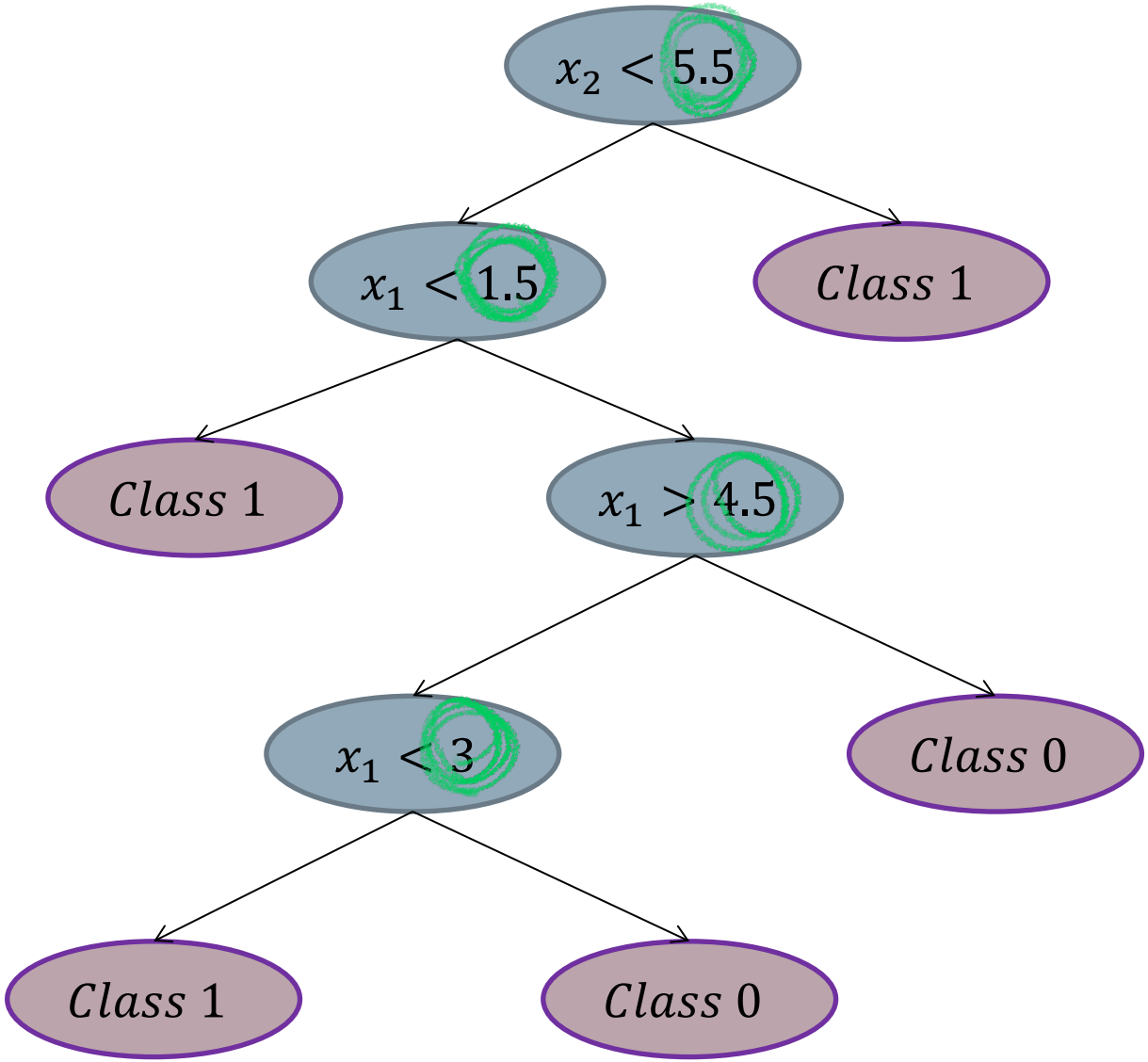
$$G = 1 - \left(\frac{1}{1}\right)^2 - \left(\frac{0}{1}\right)^2 = 0 \qquad G = 1 - \left(\frac{0}{1}\right)^2 - \left(\frac{1}{1}\right)^2 = 0$$

$$IG = \left(0.5 - 0 \times \frac{1}{2} - 0 \times \frac{1}{2}\right) \times \frac{2}{10} = 0.1$$

Simple Example for Tree



Step	Impurity Change	Total impurity
0	0.000	0.500
1	0.214	0.286
2	0.119	0.167
3	0.067	0.100
4	0.100	0.000



Pros and Cons of Decision Tree

- Pros
  - Easy interpretation
  - Non-parametric approach
  - Inherently non-linear
  - Easy to handle categorical variables
  - Implicitly perform feature selection

- Cons
  - Large computing cost
  - Lack of linearity or main effects
  - Each node only considers single variable
    - Many algorithms has been proposed to overcome this problem

When Does Tree Stop Growing?

- Growing full-size tree can cause overfitting
  - Low classification accuracy on test set
- Introduce pruning step after growing tree
  - Pruning simplifies the tree by trimming some branches of the fully grown tree
  - Generate several pruned trees and select best tree
- Pruning techniques
  - Pre-Pruning (Early Stopping): Stop growing the tree when certain conditions are met (e.g., max depth, min samples per leaf)
  - Post-Pruning: Grow the tree fully, then remove branches that do not improve accuracy significantly

Cost Complexity Pruning

- Cost complexity pruning
  - A post-pruning method that reduces overfitting by controlling tree complexity
  - Uses a penalty term to balance model complexity and accuracy

- Cost complexity pruning algorithm
  1. Grow a fully expanded tree
  2. Compute the cost complexity for each subtree
  3. Remove nodes that do not significantly improve performance
  4. Select the optimal subtree using cross-validation.

Cost Complexity Pruning

- Cost complexity measure

Complexity parameter

$$R_\alpha(T) = R(T) + \alpha|T|$$

  - $R(T)$ is misclassification cost of T
  - $|T|$ is tree complexity=the number of terminal nodes

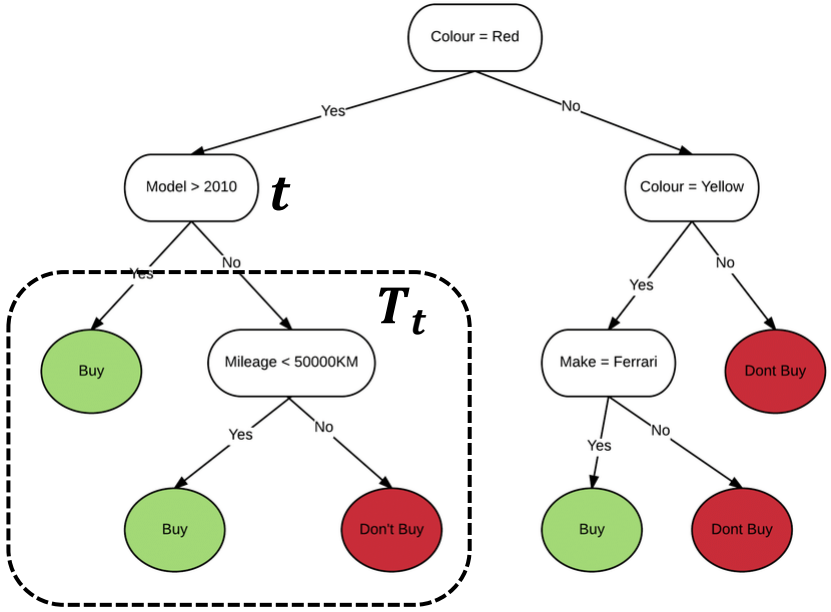
Cost Complexity Pruning

- Calculate values of cost complexity measure for internal node t before and after removing subtree T_t
  - For a subtree at t

$$R_\alpha(T_t) = R(T_t) + \alpha|T_t|$$

  - For a terminal node t

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



Cost Complexity Pruning

- Misclassification cost at node t

$$r(t) = \min_i \sum_{k=1}^{K} C(i|k)p(k|t) \quad \rightarrow \quad r(t) = 1 - \max_k p(k|t)$$

  - $C(i|k) = \begin{cases} 1, & if\ i \neq k \\ 0, & if\ i = k \end{cases}$
  - $p(k|t)$ is probability that class of data point is k given that it is in node t

- Misclassification cost of tree T

$$R(T) = \sum_{t \in \tilde{T}} r(t)p(t) = \sum_{t \in \tilde{T}} R(t)$$

  - $\tilde{T}$ is set of terminal nodes of tree T
  - $p(t)$ is probability that data point is in node t
  - Set $R(t) = r(t)p(t)$

Cost Complexity Pruning

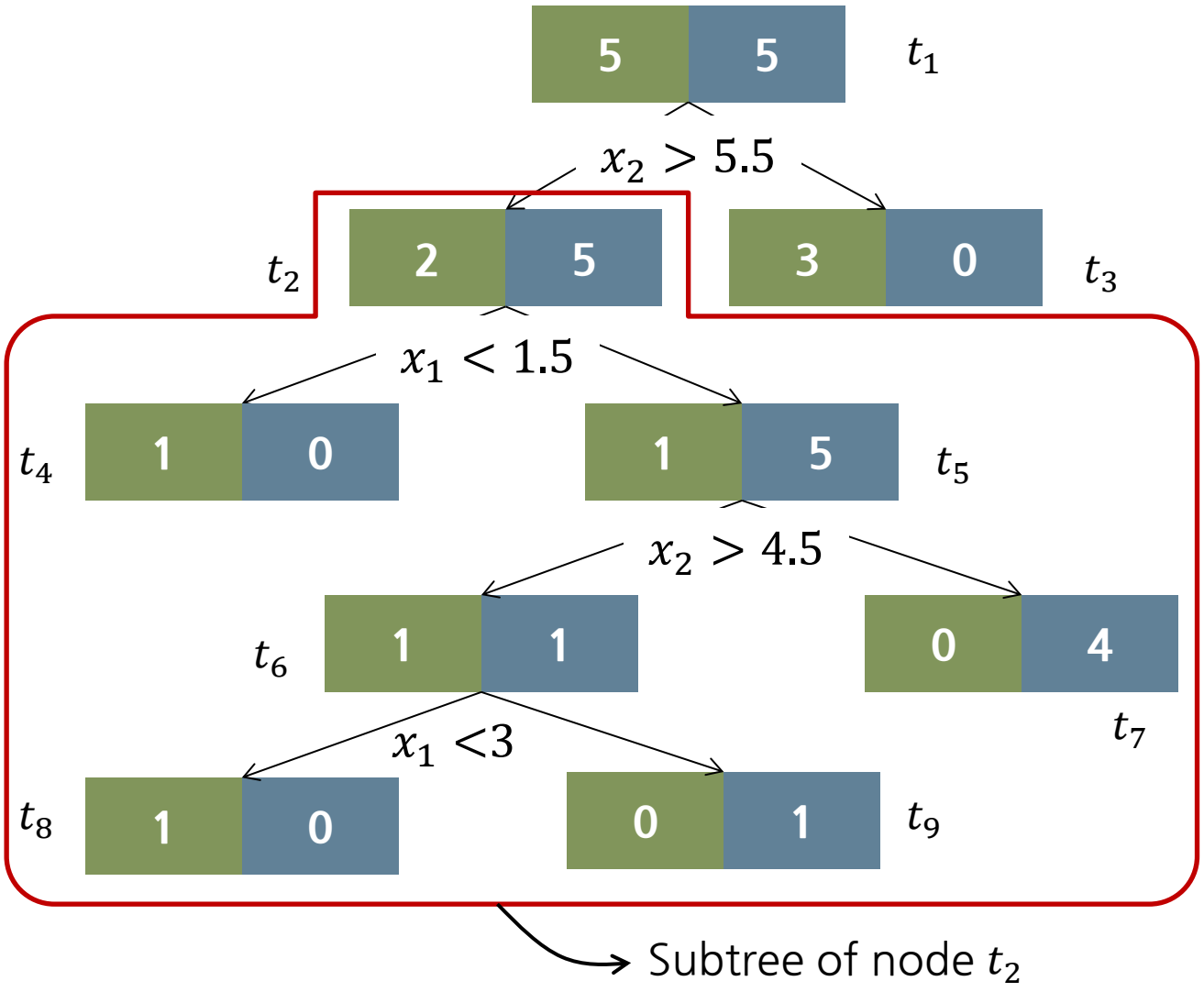
- Cost complexity pruning prunes subtree at t by comparing $R_\alpha(T_t)$ and $R_\alpha(t)$

$$R(T_t) + \alpha|T_t| = R(t) + \alpha$$

$$\alpha(t) = \frac{R(t) - R(T_t)}{|T_t| - 1}$$

  - **A small alpha value** at a node means that removing (pruning) the subtree rooted at that node results in only a small increase in the error relative to the reduction in tree complexity
    - This suggests the subtree is not very important and can be pruned early with little cost in accuracy.
  - **A large alpha value** at a node indicates that pruning the subtree would significantly increase the error compared to the gain from simplifying the tree
    - This means the subtree contains useful structure and should be retained longer during pruning.

Cost Complexity Pruning



At node t_2

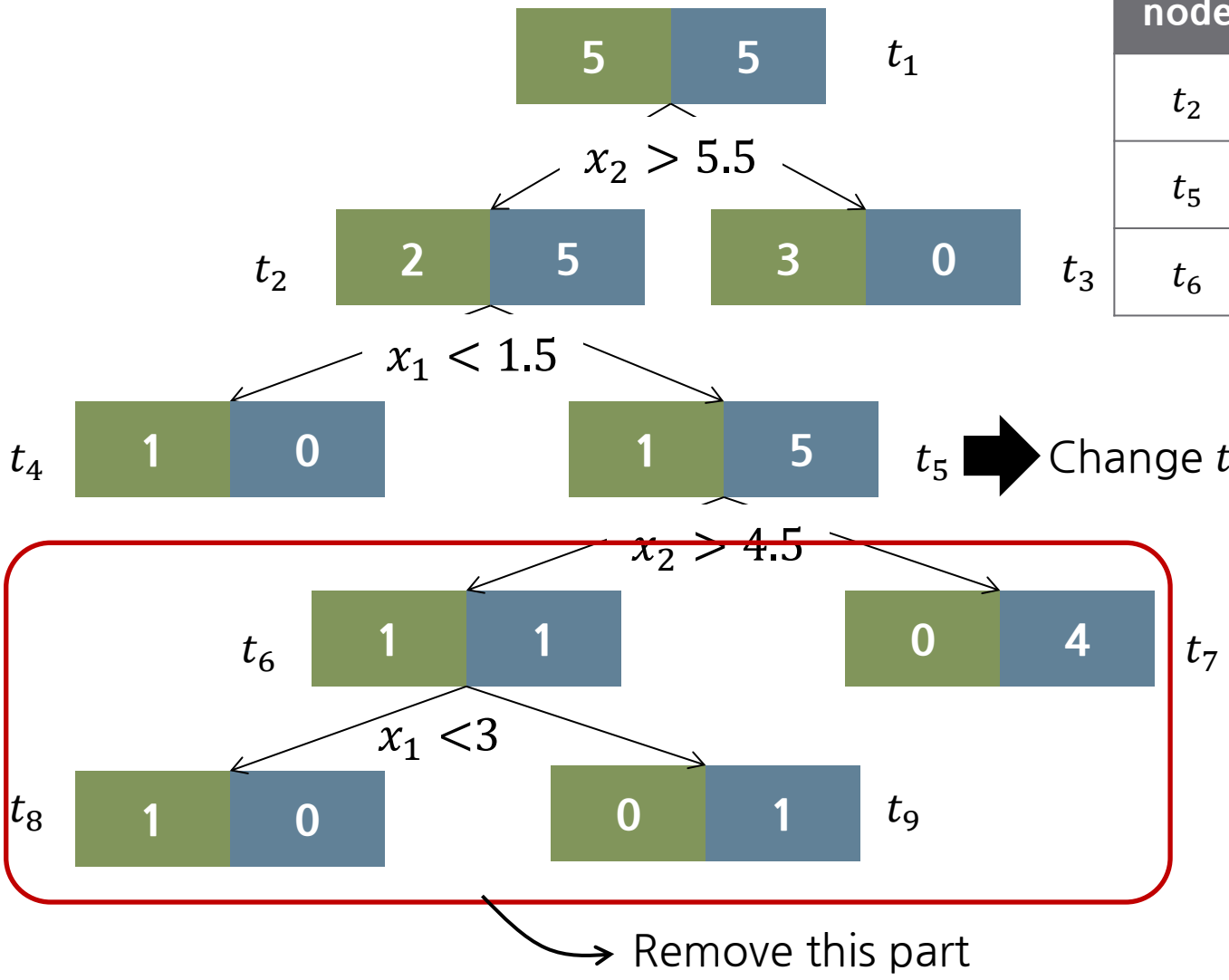
$$R(t_2) = r(t_2)p(t_2)$$
$$= \left(1 - \frac{5}{7}\right) \times \frac{7}{10} = 0.2$$

$$|T_{t_2}| = 4$$

$$R(T_{t_2})$$
$$= (1-1) \times \frac{1}{10} + (1-1) \times \frac{1}{10}$$
$$+ (1-1) \times \frac{1}{10} + (1-1) \times \frac{4}{10}$$
$$= 0$$

$$\alpha(t_2) = \frac{0.2 - 0}{4 - 1} \approx 0.067$$

Cost Complexity Pruning



node	$R(t)$	$R(T_t)$	$\lvert T_t \rvert$	$\alpha(t)$
t_2	0.2	0	4	0.067
t_5	0.1	0	3	**0.050**
t_6	0.1	0	2	0.100

Change t_5 to leaf node to classify class 0

Regression Tree: Splitting Criteria

- Splitting criteria for regression problems
  - Entropy and Gini impurity are not appropriate split measure for regression analysis
  - MSE (Mean Squared Error)
    - The split that most decreases the MSE is selected

$$\hat{y}_i = \frac{\sum_{j \in t_i} y_j}{|t_i|}$$

$$R(t_i) = \frac{1}{N_{t_i}} \sum_{j \in t_i} \left(y_j - \hat{y}_i\right)^2$$

$$IG = p(t_p)R\left(t_p\right) - p(t_r)R(t_r) - p(t_l)R(t_l)$$

  - Friedman MSE

$$\hat{y}_i = \frac{\sum_{j \in t_i} y_j}{|t_i|}$$

$$IG = \frac{N_{t_r} N_{t_l}}{N_{t_r} + N_{t_l}} \left(\hat{y}_{t_r} - \hat{y}_{t_l}\right)^2$$

Regression Tree

- Splitting criteria for regression problems
  - MAE (Mean Absolute Error)
    - The split that minimizes the L1 loss using the median of each terminal node is selected

$$\hat{y}_i = the\ median\ of\ each\ terminal\ node$$

$$R(t_i) = \frac{1}{N_{t_i}} \sum_{j \in t_i} |y_j - \hat{y}_i|$$

$$IG = p(t_p)R(t_p) - p(t_r)R(t_r) - p(t_l)R(t_l)$$

Regression Tree

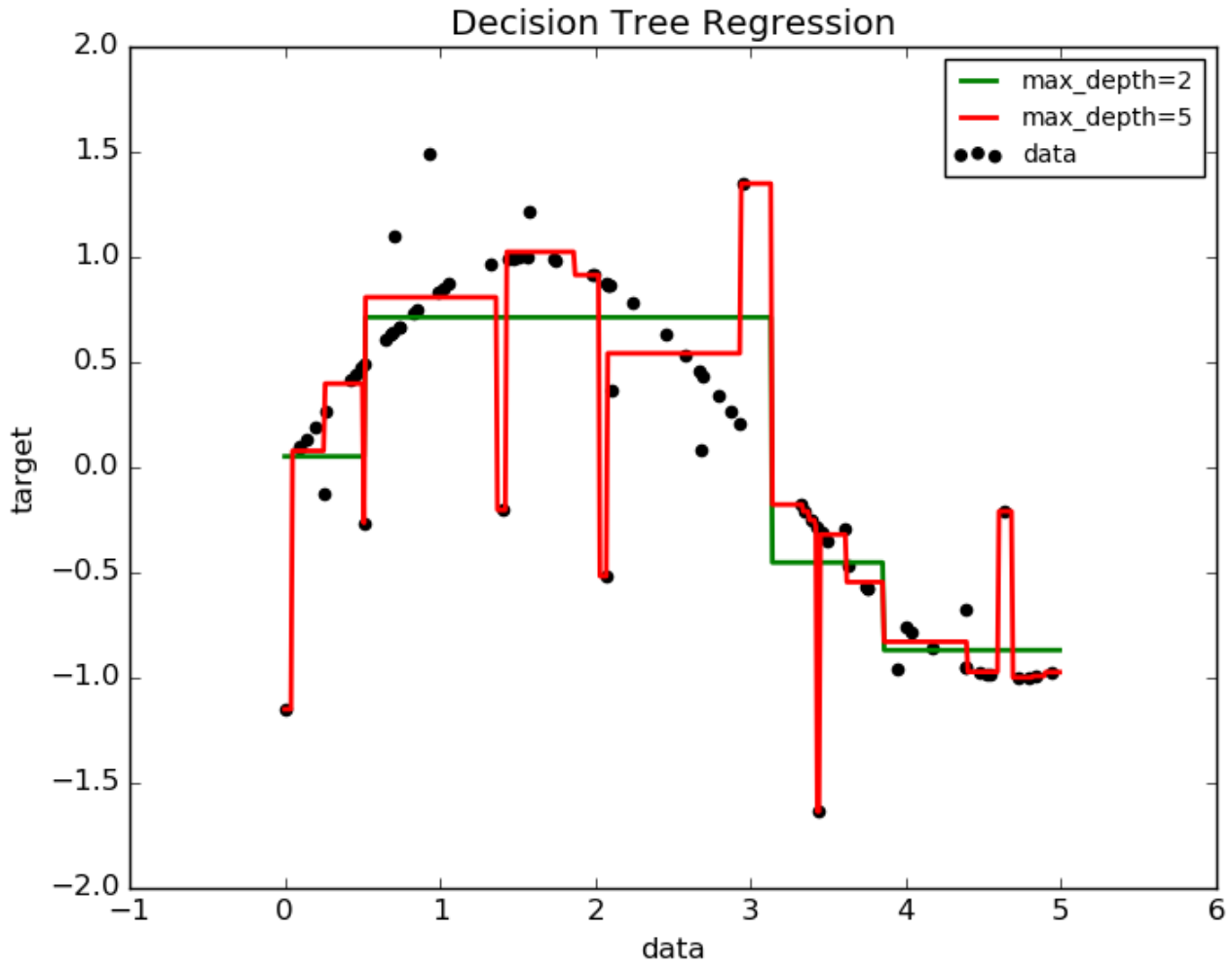
- Predicting a target value
  - The predicted value for each leaf node is the mean of the target values in that node

$$\hat{y}_t = \frac{1}{N_t} \sum_{i \in node_t} y_i$$

    - $node_t$: t-th leaf node
    - N_t: the number of training samples assigned to $node_t$

Regression Tree

- Drawback of regression tree



Random Forest

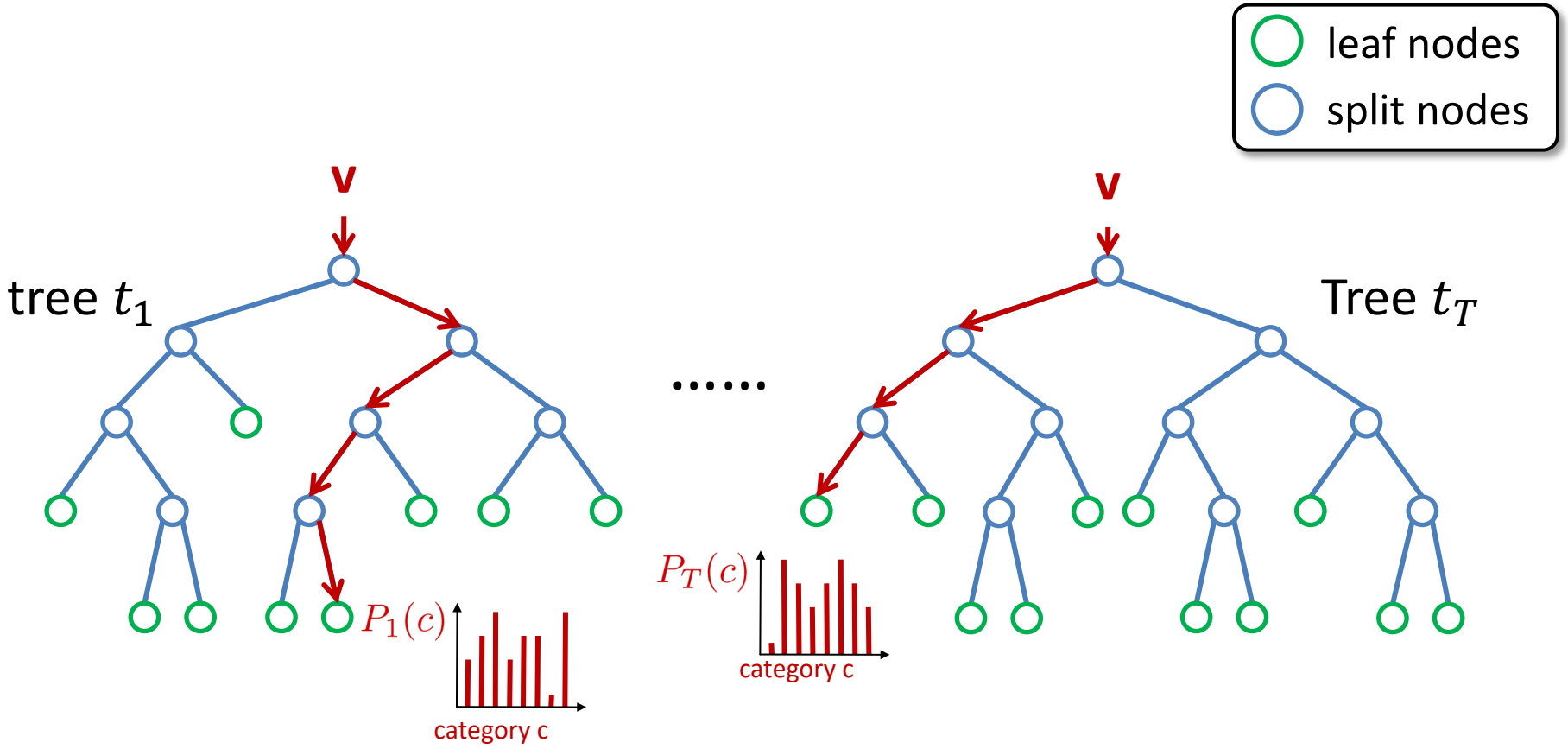
- Random Forest is an ensemble learning algorithm that combines multiple decision trees to improve the predictive performance and reduce overfitting
  - Combination of bagging idea and the random selection of features
  - T number of trees are training from T bootstrap samples
    - For regression trees, output is obtained by averaging the predictions from all trees

$$\hat{f}(\mathbf{x}) = \frac{1}{T}\sum_{i=1}^{T} f_i(\mathbf{x})$$

    - For classification tree, output is determined by taking the majority vote in the case of decision trees
    - Through bootstrapping, model performance increases over single tree (single tree is highly sensitive to noise in its training set)

Random Forest

- Random forest create T different trees on each bootstrap samples



Random Forest

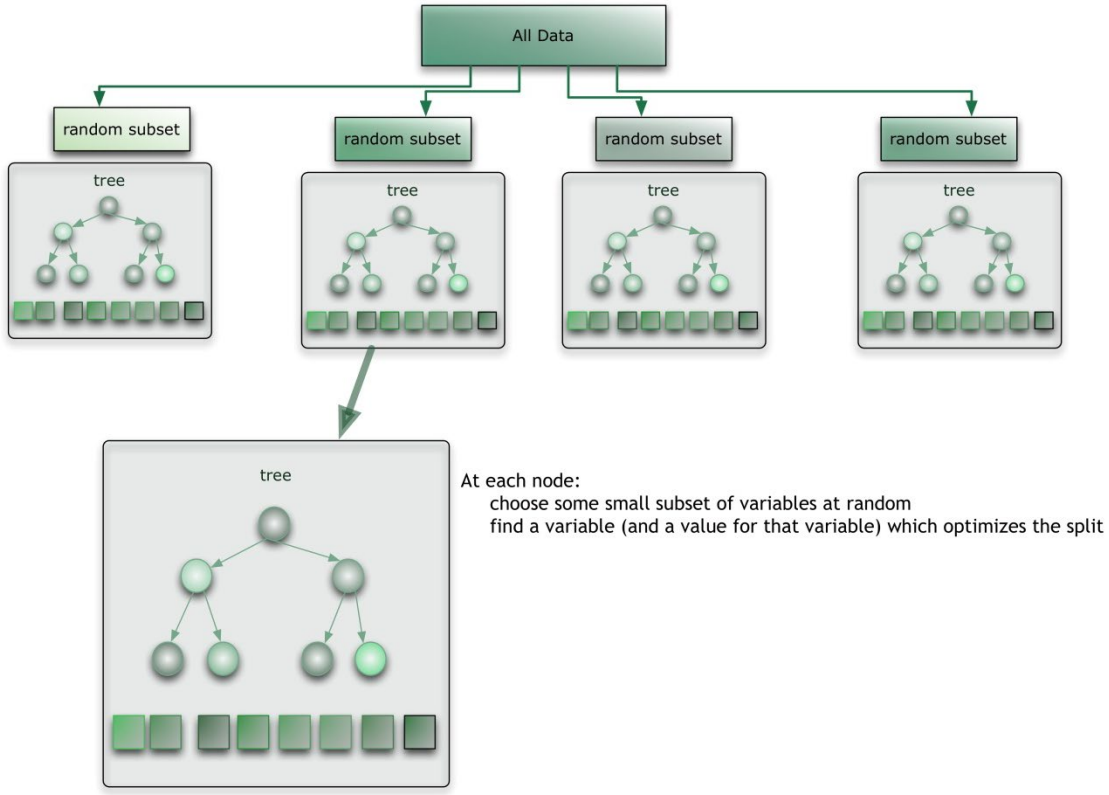
- Random forest differs from bagging in the way that this algorithm selects a random subset of the features at each candidate split
  - It is sometimes called ‘feature bagging’
  - The reason for doing this is to reduce the correlation of the trees
    - If one or a few features are very strong predictors for the response variable, these features will be selected in many trees → Trees become correlated

- Detailed splitting process for each tree

  At each node
  - For some number m, m predictor variables are selected at random from all the predictor variables
  - The predictor variable that provides the best split, according to some objective function is used to do a binary split on that node
  - At the next node, choose another m variables at random from all predictor variables and do the same

Random Forest

- Three different types of ensemble methods on tree
  - Random splitter selection: $m = 1$
  - Breiman's bagger: m =total number of predictor variables
  - **Random forest**: m ≪number of predictor variables
    - Literature suggests three possible values for m: $\frac{1}{2}\sqrt{m}, \sqrt{m}$ and $2\sqrt{m}$



Random Forest

- Pros
  - High Accuracy: Random Forest generally provides high accuracy in both classification and regression tasks
  - Robust to Overfitting: Since it uses multiple trees and random subsets of features and data, it is less likely to overfit than a single decision tree
  - Feature Importance: Random Forest can be used to determine the importance of each feature in making predictions
- Cons
  - Computationally Intensive: Since it builds multiple decision trees, Random Forest can be slow to train and require more memory, especially on large datasets
  - Less Interpretable: While individual decision trees are interpretable, the Random Forest model as a whole is more of a “black box,” making it harder to interpret the relationships between features and predictions
  - Predictions Are Slower: Predicting with a large number of trees can be slower compared to a single decision tree, especially in real-time applications