Decision Tree

Data Mining

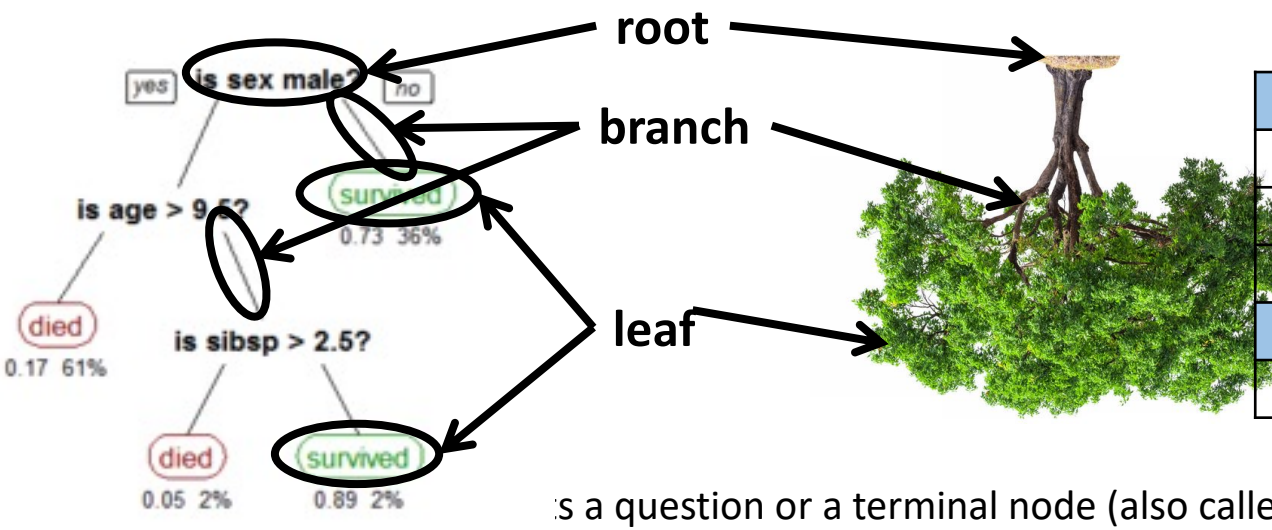
Prof. Sujee Lee

Department of Systems Management Engineering

Sungkyunkwan University

Decision Tree

- **A decision tree is a hierarchy of if/else questions leading to a decision.**
  - It follows a structure of tree.



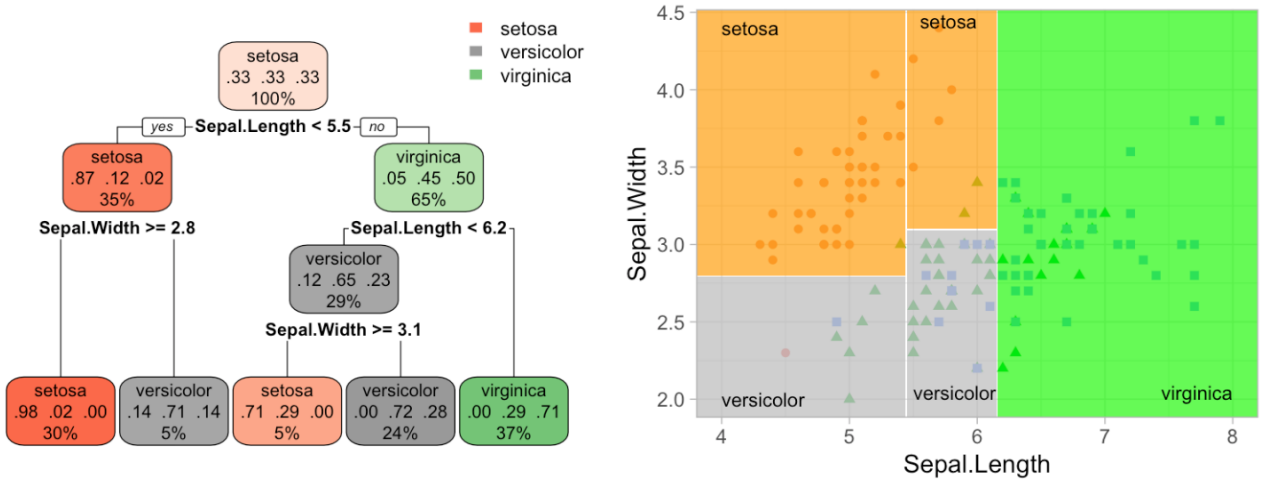
Feature	Description
X1	Sex
X2	Age
X3	# of spouse or siblings
Outcome	Description
Y	Survival

s a question or a terminal node (also called a leaf) that contains the answer.

  - The edges(branch) connect the answers to a question with the next question you would ask.

Decision Tree

- **Decision Tree**
  - ***Purpose*** : Make a decision(prediction) for a record at a leaf node
    - Each internal note tests one feature
    - Each branch from an internal node represents one outcome of the test (e.g. Yes or No)
    - Each leaf predicts Y or P(Y|X)



- The extracted knowledge can be easily understood, interpreted, and controlled by humans in the form of a readable decision tree

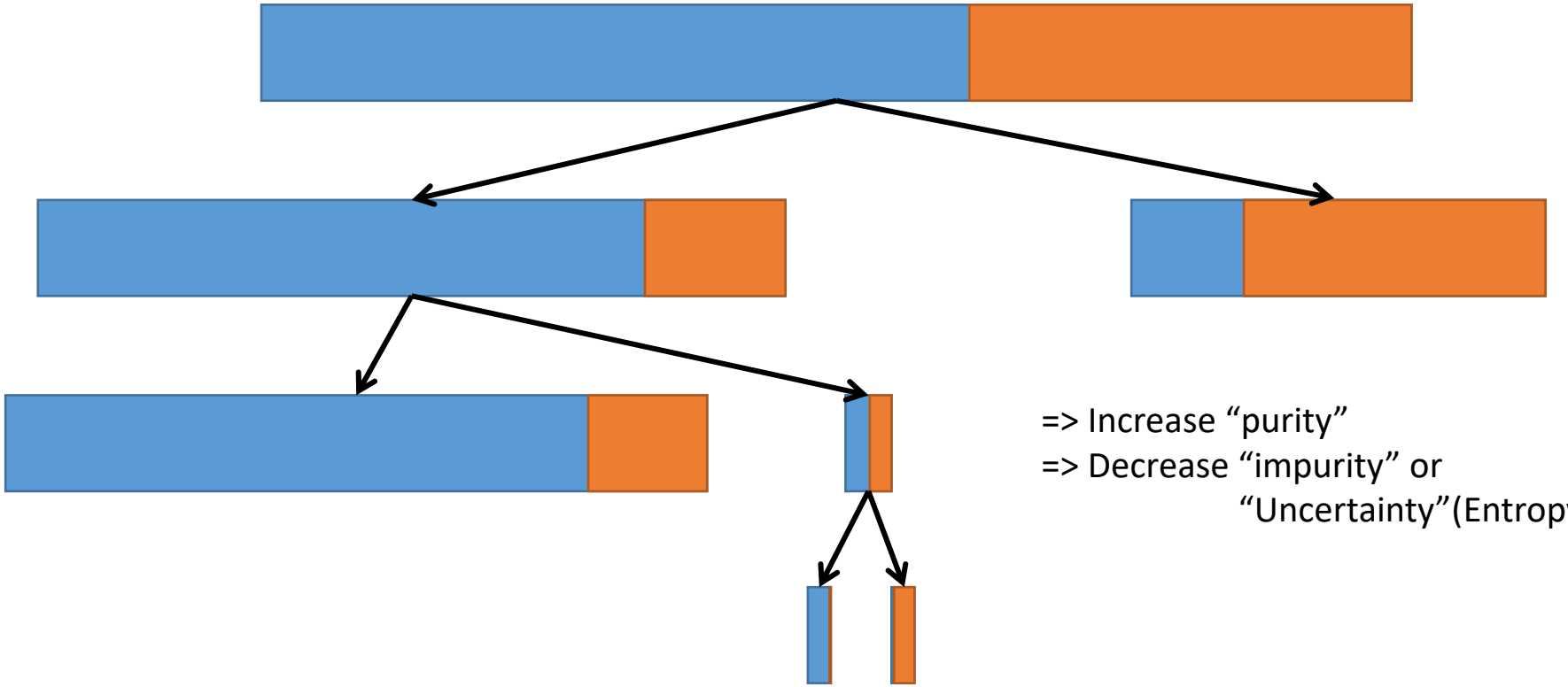
Learning Decision Trees

- **Training Process of Decision Trees**
  - A decision tree recursively partitions the feature space such that the samples with the same labels or similar target values are grouped together.

1. **Starting from the Root Node:**
   - The algorithm begins with the entire dataset and treats it as the root node.
2. **Choosing the Best Split:**
   - At each node, the algorithm evaluates each feature to determine **the best splitting** point that **maximizes data separation**.
3. **Splitting the Node:**
   - Based on the chosen feature and threshold, the node is split into two child nodes (binary split).
4. **Recursive Splitting:**
   - The algorithm recursively applies the splitting process for each child node.

Learning Decision Trees

- **How to determine the best split (for classification)**
  - Nodes with purer class distribution are preferred



=> Increase “purity”
=> Decrease “impurity” or
“Uncertainty”(Entropy)

Learning Decision Trees (Classification)

- **Measures of node impurity**
  - Let the data at node *m* be represented by Q_m with n_m samples
  - Let p_{mk} be the proportion of class *k* observations in node *m*

$$p_{mk} = \frac{1}{n_m} \sum_{y \in Q_m} I(y = k)$$

  - Common measures of impurity function H are:
    - **Gini Index** : the likelihood of incorrect classification by randomly selecting a data point

$$H(Q_m) = \sum_k p_{mk} (1 - p_{mk}) = 1 - \sum_k {p_{mk}}^2$$

    - **Entropy** : the degree of uncertainty representing how mixed the classes are within a node

$$H(Q_m) = -\sum_k p_{mk} \log p_{mk}$$

  - Select the best split that minimize the impurity among the possible split candidates

Learning Decision Trees

- **Mathematical formulation**
  - Let the data at node m be represented by Q_m with n_m samples
  - For each candidate split $\theta = (j, t_m)$ consisting of a feature j and threshold t_m, partition the data into $Q_m^{left}(\theta)$ and $Q_m^{right}(\theta)$ subsets

$$Q_m^{left}(\theta) = \{(x, y) | x_j \leq t_m\}$$

$$Q_m^{right}(\theta) = Q_m \backslash Q_m^{left}(\theta)$$

  - The quality of a candidate spilt θ of node m is :

$$G(Q_m, \theta) = \frac{n_m^{left}}{n_m} H(Q_m^{left}(\theta)) + \frac{n_m^{right}}{n_m} H(Q_m^{right}(\theta))$$

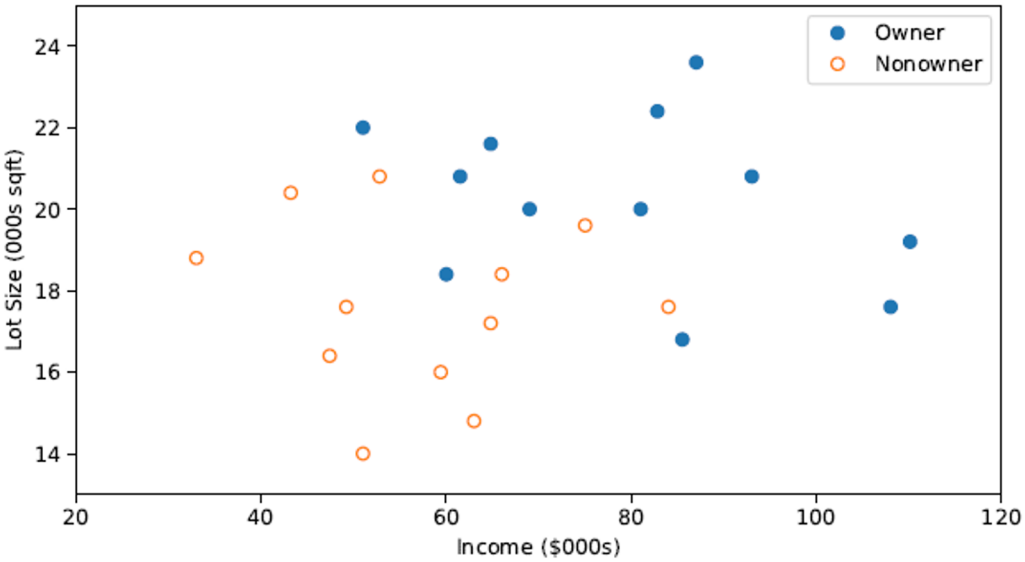
  - Select the split that minimizes the impurity

$$\theta^* = \text{argmin}_\theta \, G(Q_m, \theta)$$

  - Recurse for subsets $Q_m^{left}(\theta^*)$ and $Q_m^{right}(\theta^*)$

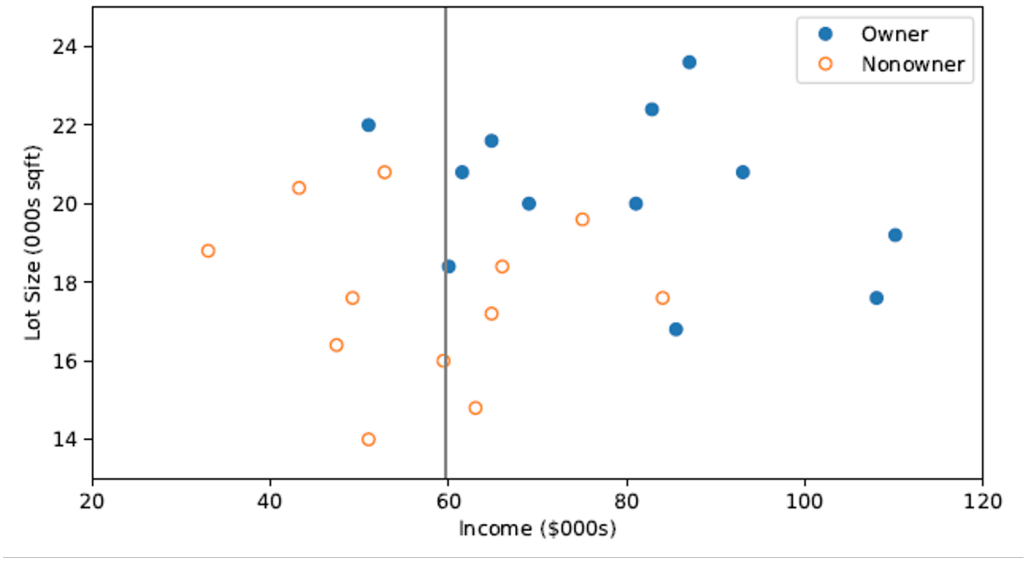
Example: Riding Mowers

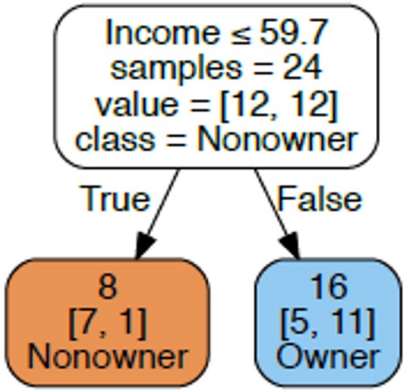
Income	Lot_Size	Ownership
60.0	18.4	owner
85.5	16.8	owner
64.8	21.6	owner
61.5	20.8	owner
87.0	23.6	owner
110.1	19.2	owner
108.0	17.6	owner
82.8	22.4	owner
69.0	20.0	owner
93.0	20.8	owner
51.0	22.0	owner
81.0	20.0	owner
75.0	19.6	non-owner
52.8	20.8	non-owner
64.8	17.2	non-owner
43.2	20.4	non-owner
84.0	17.6	non-owner
49.2	17.6	non-owner
59.4	16.0	non-owner
66.0	18.4	non-owner
47.4	16.4	non-owner
33.0	18.8	non-owner
51.0	14.0	non-owner
63.0	14.8	non-owner



- **Split Candidates & Find the best**
  - Order records according to one variable, say income
  - Take a predictor value and divide records into two parts
  - Measure resulting purity of class in each resulting portion
  - Try all other split values
  - Repeat for other variable(s)
  - Select the one variable & split that yields the most purity

Example: Riding Mowers





$$H(Q_m) = \sum_k p_{mk}\,(1 - p_{mk}) = 1 - \sum_k {p_{mk}}^2$$

$$H(Q_m) = -\sum_k p_{mk} \log p_{mk}$$

$$G(Q_m, \theta) = \frac{n_m^{left}}{n_m} H(Q_m^{left}(\theta)) + \frac{n_m^{right}}{n_m} H(Q_m^{right}(\theta))$$

$$\text{gini\_left} = 1 - (7/8)^2 - (1/8)^2 = 0.219$$

$$\text{entropy\_left} = -(7/8)\log_2(7/8) - (1/8)\log_2(1/8) = 0.544$$

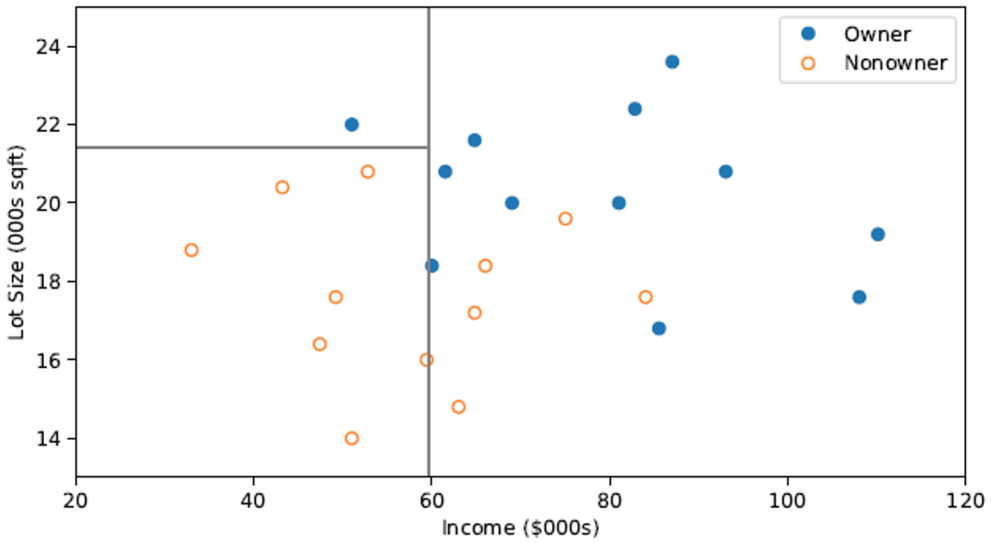
$$\text{gini\_right} = 1 - (11/16)^2 - (5/16)^2 = 0.430$$

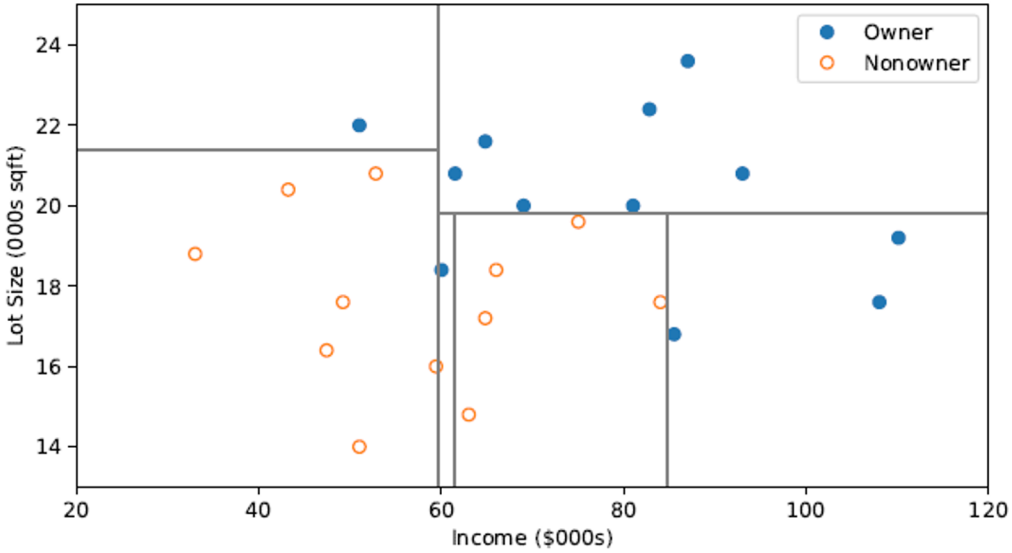
$$\text{entropy\_right} = -(11/16)\log_2(11/16) - (5/16)\log_2(5/16) = 0.896$$

$$\text{gini} = (8/24)(0.219) + (16/24)(0.430) = 0.359$$

$$\text{entropy} = (8/24)(0.544) + (16/24)(0.896) = 0.779$$

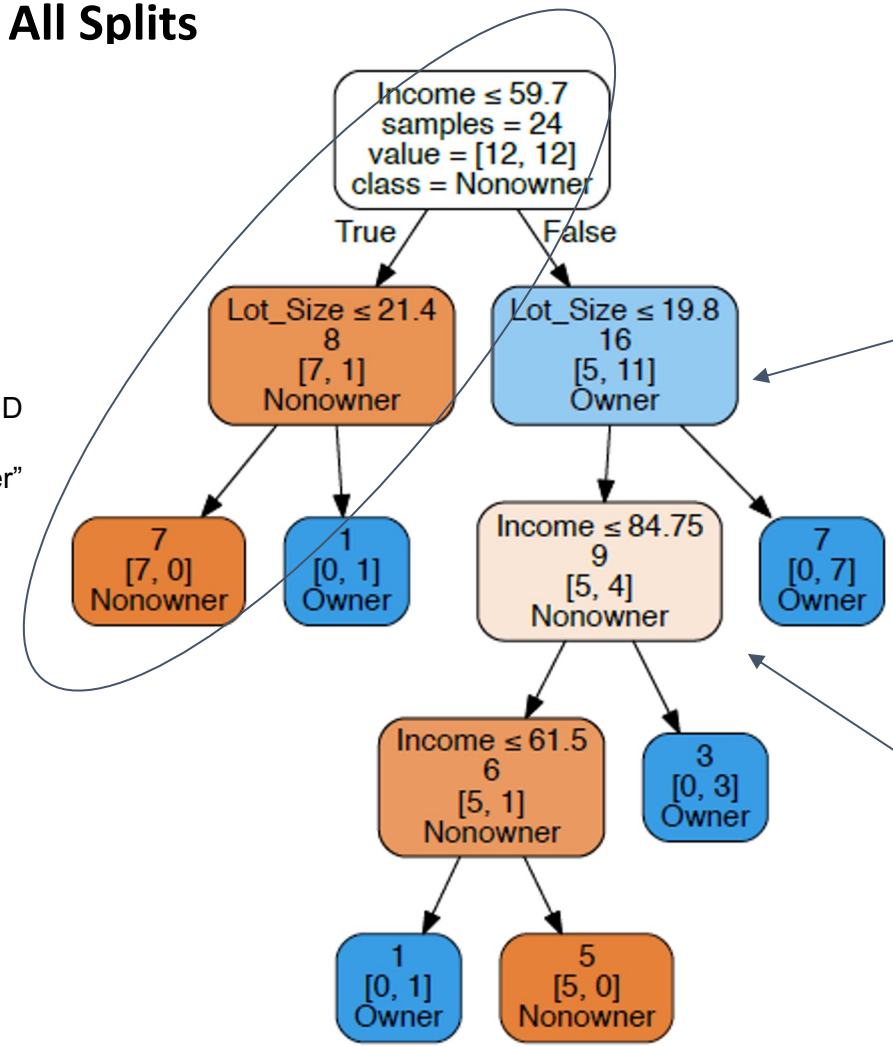
Example: Riding Mowers





Example: Riding Mowers

- **Tree After All Splits**



If Income <= 59.7 AND Lot Size <= 21.4, classify as "Nonowner"

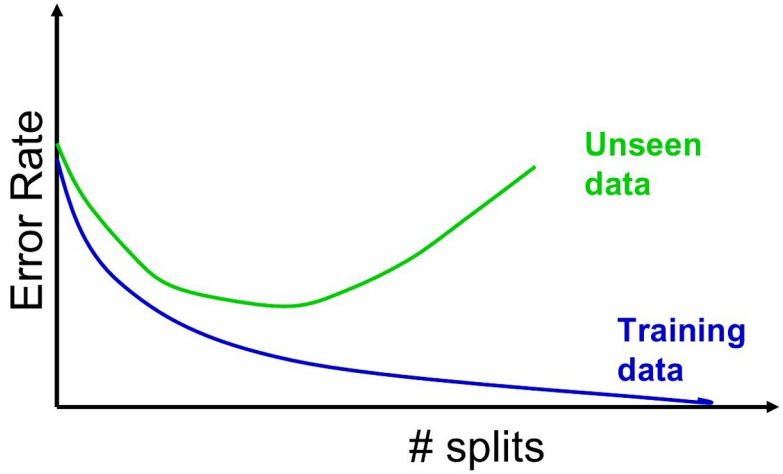
The dominant class in this portion of the first split (those with income >= 59.7) is "owner" – 11 owners and 5 non-owners

The next split for this group of 9 will be on the basis of Income, splitting at 84.75

The Overfitting Problem

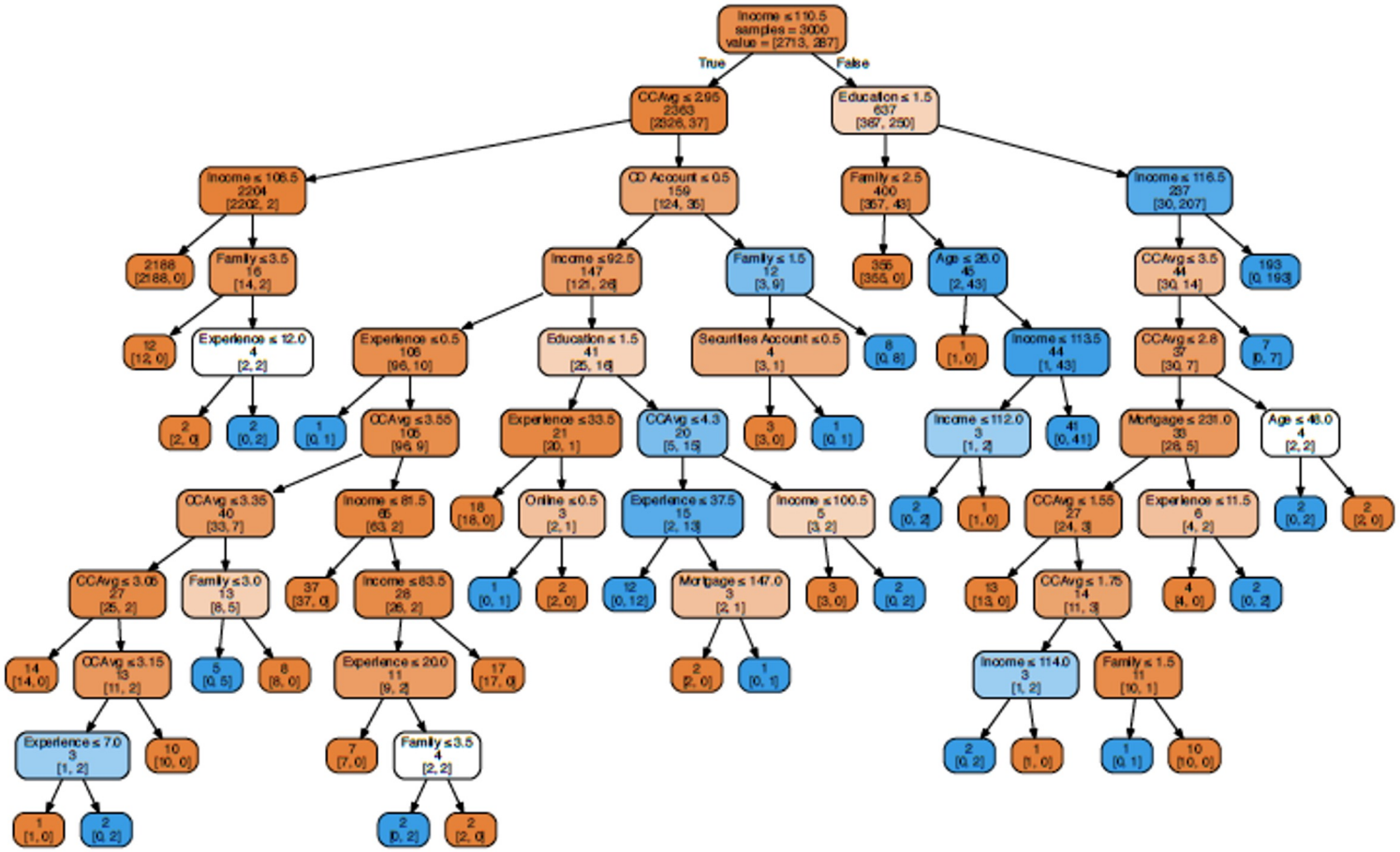
- **Naïve Stopping Criteria**
  - All nodes have one class instances (e.g., 100% purity)
  - Unable to reduce conditional entropy further
  - Exhausted all of the candidate splits

- **Issue**
  - Building a tree typically leads to models that are very *complex* and highly *overfit* to the training data.
  - Continuing until all leaves are pure means that a tree is 100% accurate on the training set, but fails to generalize on new data



The Overfitting Problem

- **Example: Loan Acceptance**
  - Full trees are too complex – they end up fitting noise, overfitting the data



Strategies to Avoid Overfitting

- **Early stopping**
  - Stopping the creation of the tree early by controlling
    - Tree depth
    - Minimum number of records in split nodes
    - Minimum number of records in terminal node
    - Minimum impurity increase needed, etc.
  - But what are optimal values for these parameters?
    - By validation data / cross-validation

- **Post-Pruning**
  - Building the tree but then removing or collapsing nodes that contain little information

Post-Pruning

- **Minimal Cost-Complexity Pruning**
  - **Cost-Complexity Measure**: $R_\alpha(T) = R(T) + \alpha|T|$
    - $|T|$: Number of terminal nodes
    - $R(T)$: Total misclassification rate of the terminal nodes / Total weighted impurity
  - Minimal cost-complexity pruning finds the subtree of T that minimizes $R_\alpha(T)$.
  - **Process:**
    - For each non-terminal node t, calculate its effective complexity parameter,

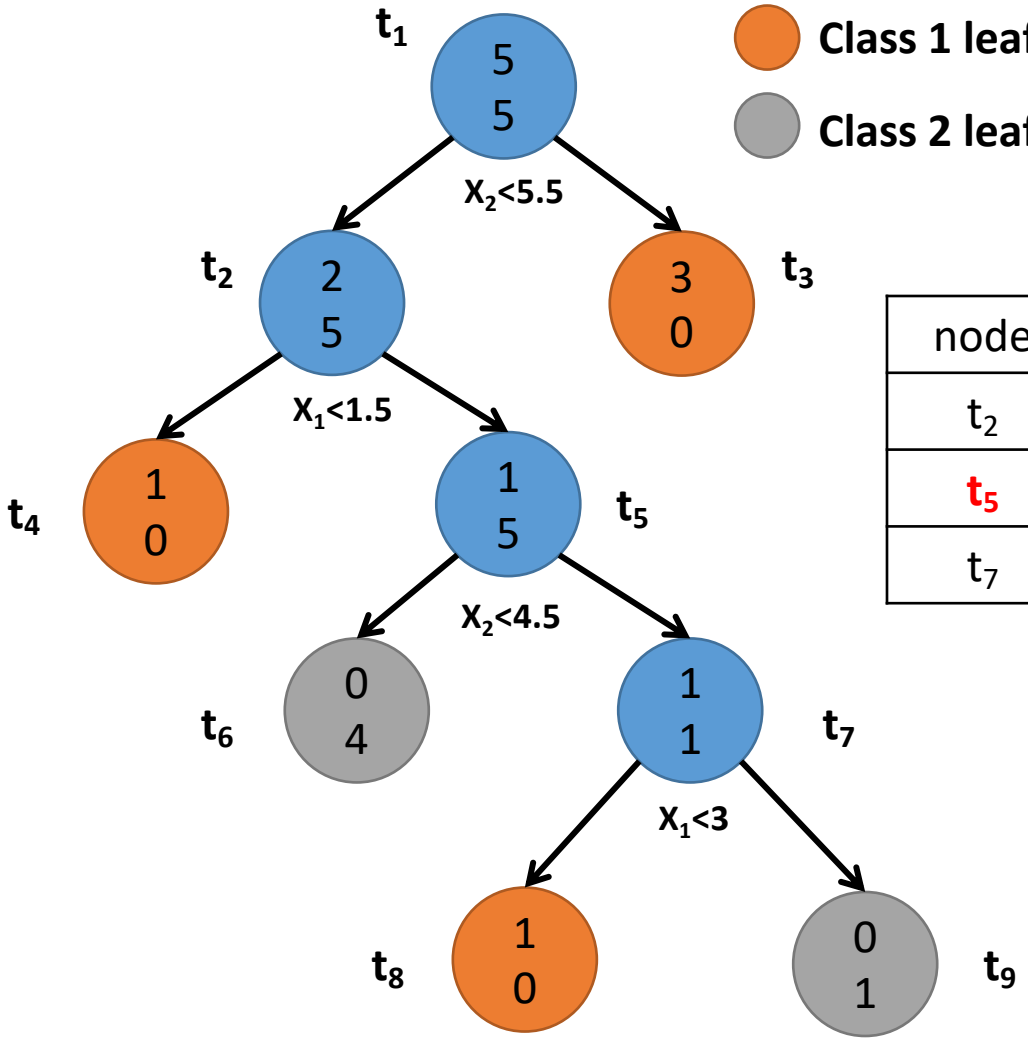
$$\alpha_t = \frac{R(t) - R(T_t)}{|T_t| - 1}$$

where T_t is the branch under t. The node with the smallest α_t is selected for pruning.

- Continue pruning until the minimum α_t exceeds some threshold.

Post-Pruning

- **Minimal Cost-Complexity Pruning Example**



Class 1 leaf node

Class 2 leaf node

node	$R(t)$	$R(T_t)$	$\|T_t\|$	$\alpha(t)$
t_2	0.2	0	4	0.067
t_5	0.1	0	3	**0.05**
t_7	0.1	0	2	0.1

Regression Tree

- **Regression Tree**
  - Regression tree is similar, except:
    - Prediction is computed as the average of numerical target variable in the node

$$\bar{y}_m = \frac{1}{n_m} \sum_{y \in Q_m} y$$

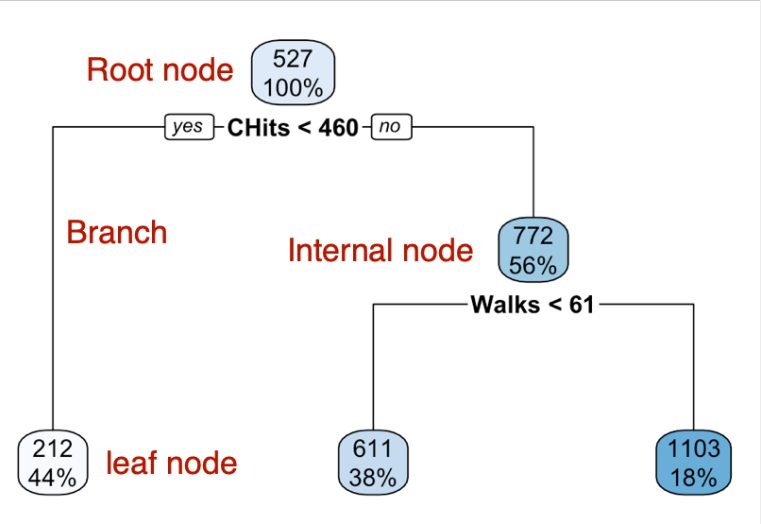
    - Impurity measured by mean squared error from leaf mean

$$H(Q_m) = \frac{1}{n_m} \sum_{y \in Q_m} (y - \bar{y}_m)^2$$

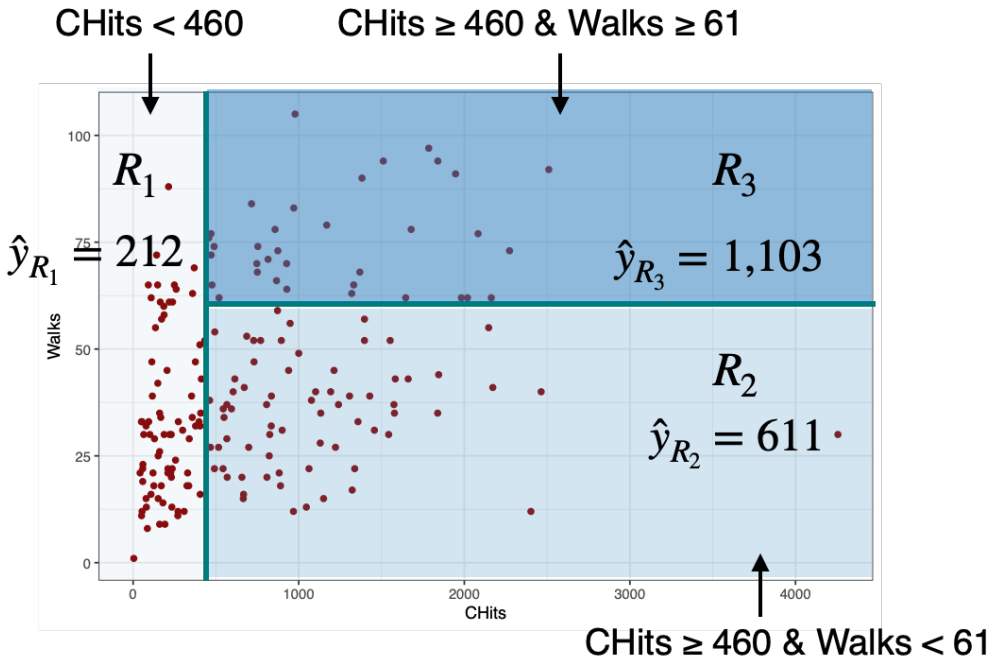
    - Performance measured by RMSE (root mean squared error)

Regression Tree

- **Regression Tree Example**



Regression Tree



Dataset Partition

Decision Tree – Pros and Cons

- **What is good about decision tree**
  - Easy to understand (good interpretability)
  - Selects important variables
  - Robust (insensitive) to noise
  - Easy to handle missing values
  - No assumption about data and parameters

- **What is bad about decision tree**
  - High computational complexity: The search space is huge.
  - Interaction terms not considered
  - Linearity not considered