

# Decision Tree

ML Instruction Team, Fall 2022

CE Department  
Sharif University of Technology

# Decision Tree

- **Decision Tree** algorithms can be considered as an iterative, top-down construction method for either classifier or regressor.
- Considering only binary (or Boolean) features, at each node, there are  $2^m$  potential splits to be evaluated given that the dataset has  $m$  features.

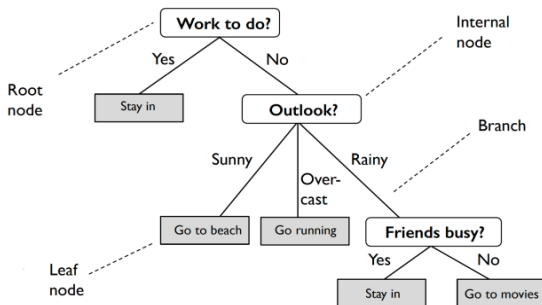


Figure: Decision Tree, Source

# Terminology

- **Root Node**: No incoming edge, zero, or more outgoing edges.
- **Internal Node**: One incoming edge, two (or more) outgoing edges.
- **Leaf Node**: Each leaf node is assigned a class label if nodes are pure; otherwise, the class label is determined by majority vote.
- **Parent & Child Node**: If a node is split, we refer to that given node as the parent node, and the resulting nodes are called child nodes.

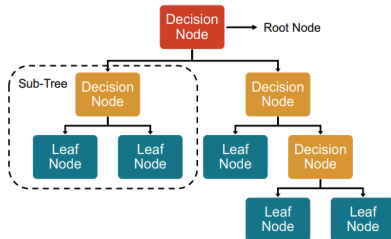


Figure: Terminology, [Source](#)

# Information Gain

- **Entropy**: Let  $X$  be a random variable over a discrete space  $\Omega$  with probability mass function  $\mathbb{P}$ . Then the (Shannon) entropy of  $X$  is:

$$H(X) = \mathbb{E}_X[\log \frac{1}{\mathbb{P}(X)}] = \sum_{x \in X} \mathbb{P}(x) \log \frac{1}{\mathbb{P}(x)} = - \sum_{x \in X} \mathbb{P}(x) \log \mathbb{P}(x)$$

- **Gini Impurity**: Let  $X$  be a random variable over a discrete space  $\Omega$  with probability mass function  $\mathbb{P}$ . Then the gini coefficient of  $X$  is:

$$G(X) = \sum_{x \in X} \mathbb{P}(x)(1 - \mathbb{P}(x)) = 1 - \sum_{x \in X} \mathbb{P}(x)^2$$

# Information Gain

$IG$	Information Gain
$X$	Discrete random variable $\{x_1, \dots, x_n\}$
$P(X)$	Probability mass function
$H(X)$	Entropy

## Information Gain on Split

$$H(X) = - \sum_{i=1}^n P(x_i) \log_b P(x_i)$$

units of entropy

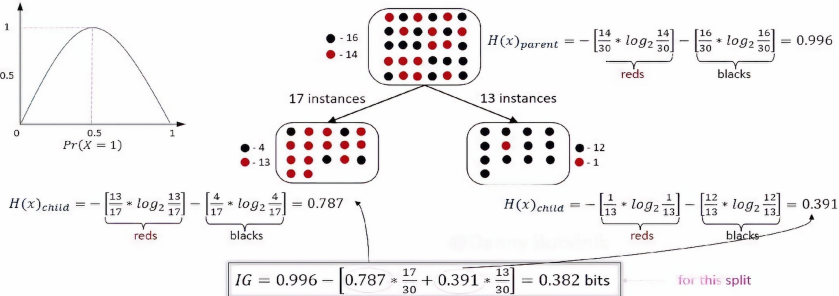
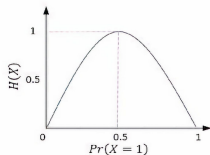


Figure: Information Gain, [Source](#)

# CART Algorithm

- The algorithm first splits the training set in **two subsets** using a **single feature**  $k$  and a **threshold**  $t_k$ .
- How does it choose  $k$  and  $t_k$ ?  
It searches for the pair  $(k, t_k)$  producing the **purest subsets**, which are weighted by their size.
- Once it has successfully split the training set in two, it splits the **subsets** using the same logic, then the **sub-subsets** and so on, recursively.
- **CART** algorithm is a greedy algorithm: it greedily searches for an optimum split at the top level, then repeats the process at each level.
- It does not check whether or not the split will lead to the lowest possible impurity several levels down.

# Decision Trees in Practice



Training / test data

Features				Labels
Sepal length	Sepal width	Petal length	Petal width	Species
5.1	3.5	1.4	0.2	Iris setosa
4.9	3.0	1.4	0.2	Iris setosa
7.0	3.2	4.7	1.4	Iris versicolor
6.4	3.2	4.5	1.5	Iris versicolor
6.3	3.3	6.0	2.5	Iris virginica
5.8	3.3	6.0	2.5	Iris virginica

Figure: Iris Dataset, [Source](#)

# Decision Trees in Practice

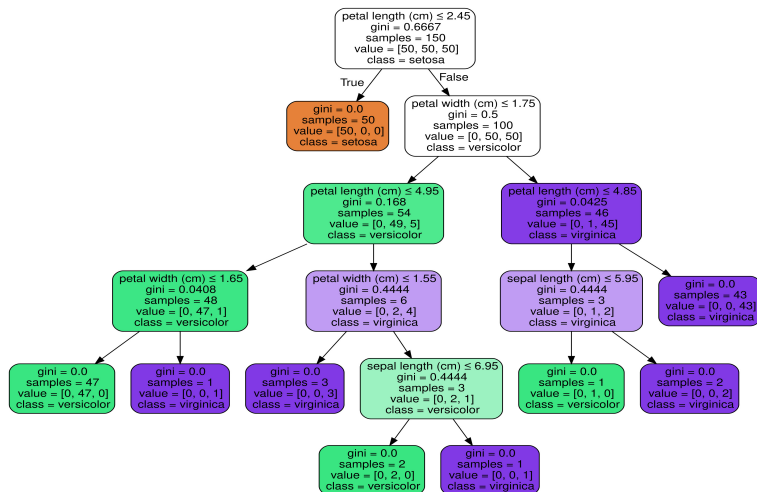


Figure: Decision Trees on Iris Dataset, [Source](#)



# Misclassifications Error

- **Misclassification** error seems to be another reasonable choice, where:

$$ERR(\mathcal{D}) = \frac{1}{n} \sum_{i=1}^n L(\hat{y}^{[i]}, y^{[i]})$$

with the 0-1 Loss,

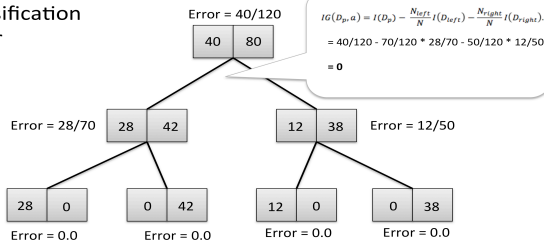
$$L(\hat{y}, y) = \begin{cases} 0 & \text{if } \hat{y} = y, \\ 1 & \text{if } \hat{y} \neq y \end{cases}$$

- This, in case of the training set, is equal to

$$ERR(p) = 1 - \max_i (p(i | x_j))$$

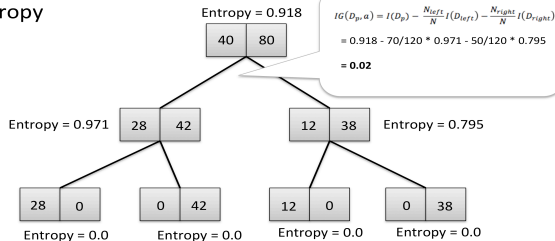
- Can we apply this measure as the criteria for splitting?

## Classification Error



(a) Misclassification Error, Source

## Entropy



(b) Entropy, Source

# Information Gain

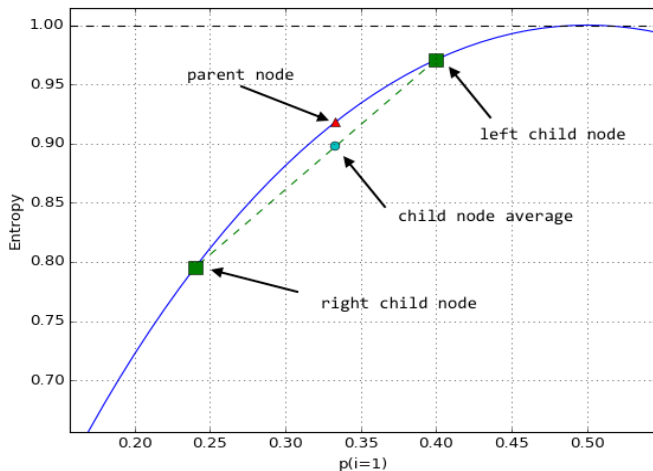


Figure: Concavity of Entropy, [Source](#)

# Overfitting

- Decision Trees make very few assumptions about the training data!
- If left unconstrained, their structure will adapt itself to the training data, fitting it very closely, and most likely overfitting it!
- To avoid overfitting the training data, you need to restrict the Decision Tree's freedom during training.
- Which parameters can be restricted?
  - ▷ **Max Tree Depth**: Maximum depth of the tree.
  - ▷ **Max Leaf Nodes**: Maximum number of leaf nodes.
  - ▷ **Min Sample Leaf**: Minimum number of samples a leaf node must have.
  - ▷ **Min Sample Split**: Minimum number of samples a node must have before it can be split.
  - ▷ **Max Features**: Maximum number of features that are evaluated for splitting at each node)

# Other Algorithms

## ■ ID3: Iterative Dichotomizer 3

- ▷ It can only be used for Discrete features, and cannot handle numeric features!
- ▷ It supports multi-category splits.
- ▷ It does not apply any pruning, and is prone to overfitting.
- ▷ It results in short and wide trees (compared to CART).
- ▷ It maximizes information gain/minimizes entropy.

## ■ C4.5:

- ▷ It supports Continuous and discrete features (continuous feature splitting is very expensive because must consider all possible ranges).
- ▷ It can handles missing attributes (ignores them in information gain computation)
- ▷ It performs post-pruning

# Decision Trees for Regression

- It is the same as classification with a few subtle modifications:
  - ▷ The main difference is that instead of predicting a class in each node, it predicts a value.
  - ▷ The prediction for each node is simply the average target value of the all training instances associated to this leaf node.
  - ▷ Instead of trying to split the training set in a way that minimizes impurity, CART algorithm now tries to split the training set in a way that minimizes the **MSE**.
  - ▷ Likewise to classification tasks, decision trees are prone to overfitting when dealing with regression tasks.

**Thank You!**

**Any Question?**