CMPSC 448: Machine Learning

Lecture 9. Decision Trees

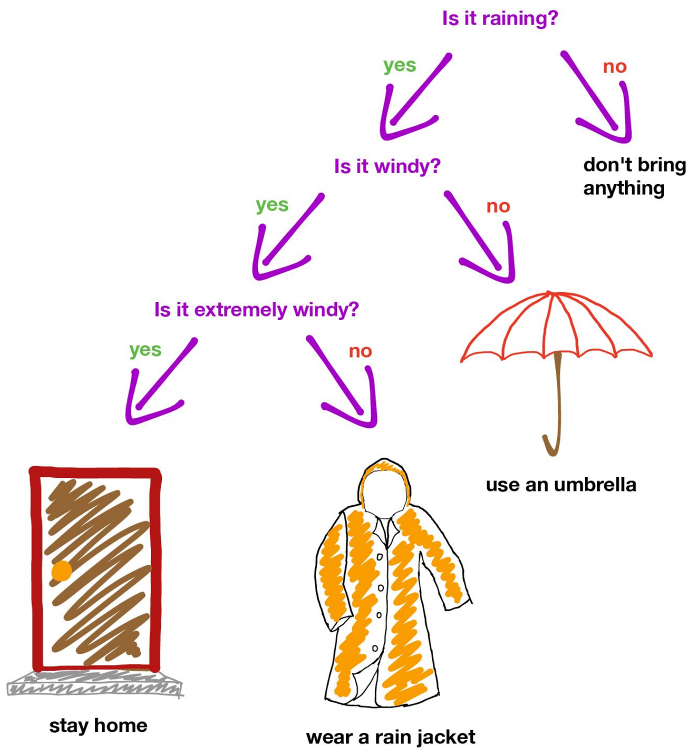
Rui Zhang
Fall 2021

PennState

Outline

- Decision Trees
- Learning Decision Trees using Entropy and Information gain
- Overfitting and Pruning (regularization) in Decision Trees

Decision trees

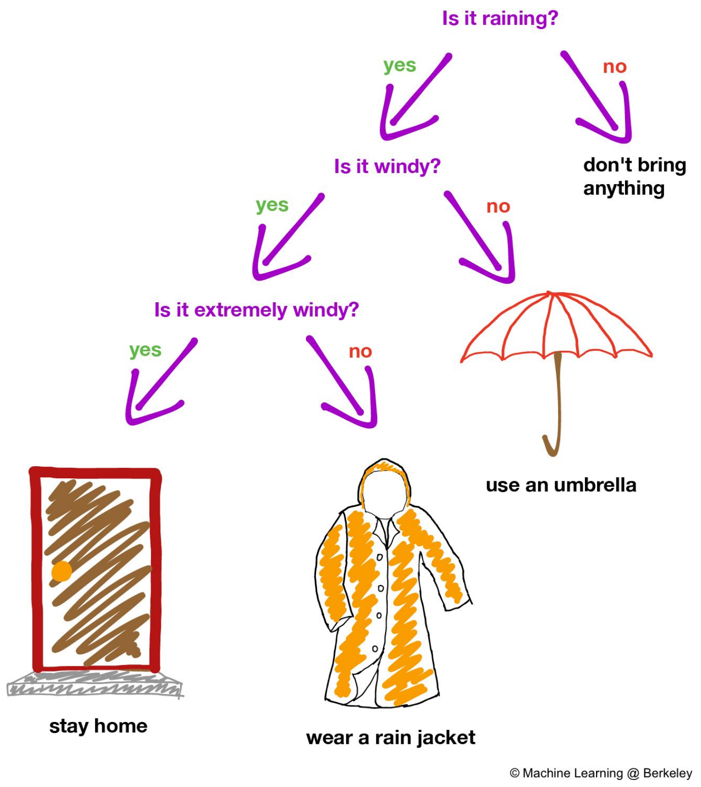


A simple decision tree for deciding what to wear!

Decision trees

A decision tree is a prediction rule, represented by a tree (usually binary) in which:

- Each internal node is associated with a splitting rule on a feature
- Each leaf node is associated with a label



A simple decision tree for deciding what to wear!

Decision trees

A decision tree is a prediction rule, represented by a tree (usually binary) in which:

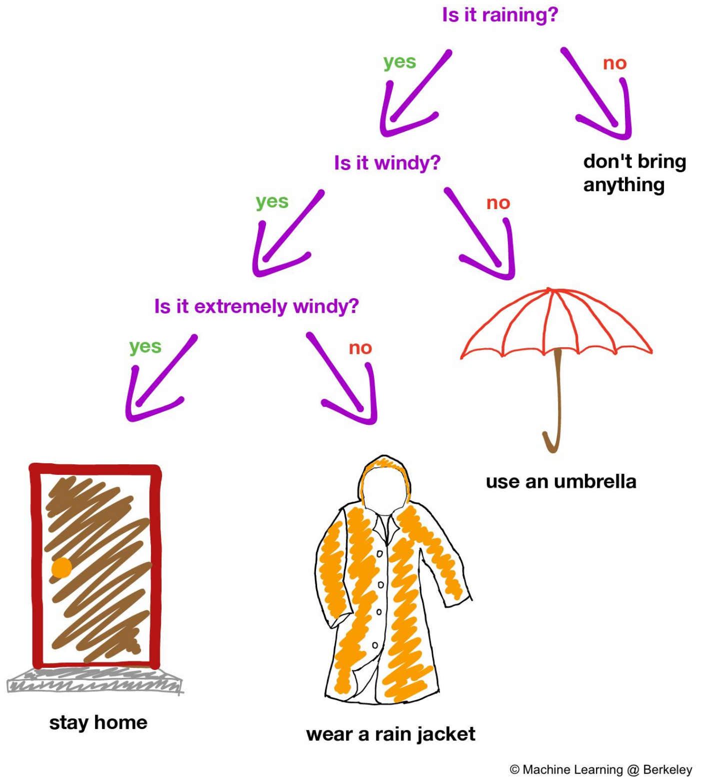
- Each internal node is associated with a splitting rule on a feature
- Each leaf node is associated with a label

A nested sequence of “if-else” decisions based on the splitting rules on features.

Mirrors human decision making more closely than other approaches.

Simple to understand and interpret.

Able to handle both **numerical** and **categorical** data



A simple decision tree for deciding what to wear!

Example

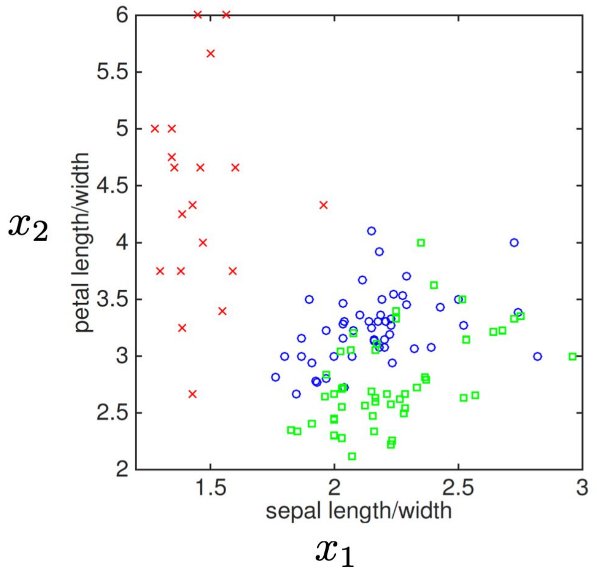
Classifying irises flower images into three types by sepal and petal measurements

https://archive.ics.uci.edu/ml/datasets/iris

- $\mathcal{X} = \mathbb{R}^2, \mathcal{Y} = \{1, 2, 3\}$
- x_1 : ratio of sepal length to width
- x_2 : ratio of petal length to width

Example

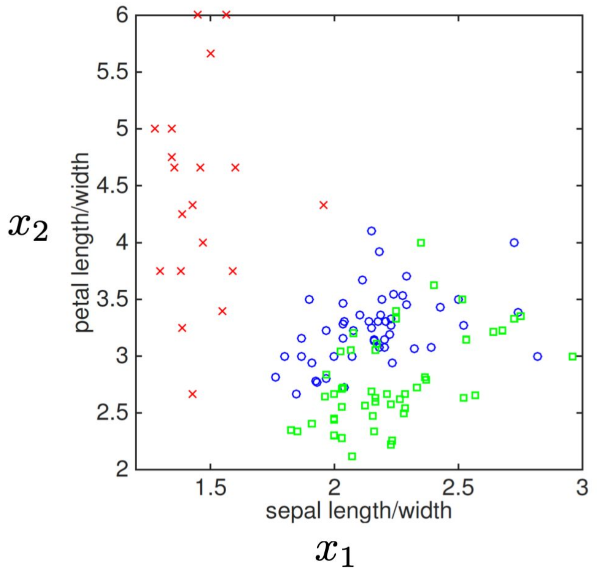
Classifying irises flower images into three types by sepal and petal measurements



- $\mathcal{X} = \mathbb{R}^2, \mathcal{Y} = \{1, 2, 3\}$
- x_1 : ratio of sepal length to width
- x_2 : ratio of petal length to width

Example

Classifying irises flower images into three types by sepal and petal measurements



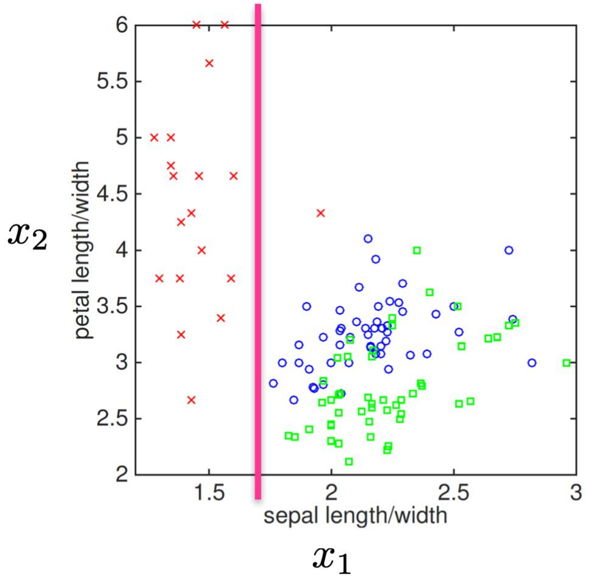
- $\mathcal{X} = \mathbb{R}^2, \mathcal{Y} = \{1, 2, 3\}$
- x_1 : ratio of sepal length to width
- x_2 : ratio of petal length to width

$\boxed{\widehat{y} = 2}$

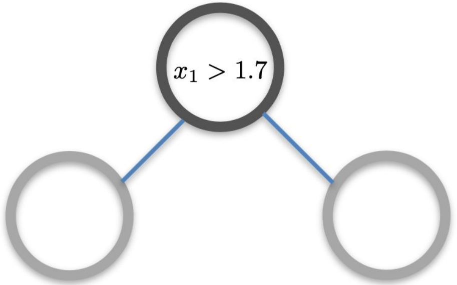
At the very beginning, we have a single node with all training data. The label of node is set to the **majority** of labels which is 2 (green) here.

Example

Classifying irises flower images into three types by sepal and petal measurements



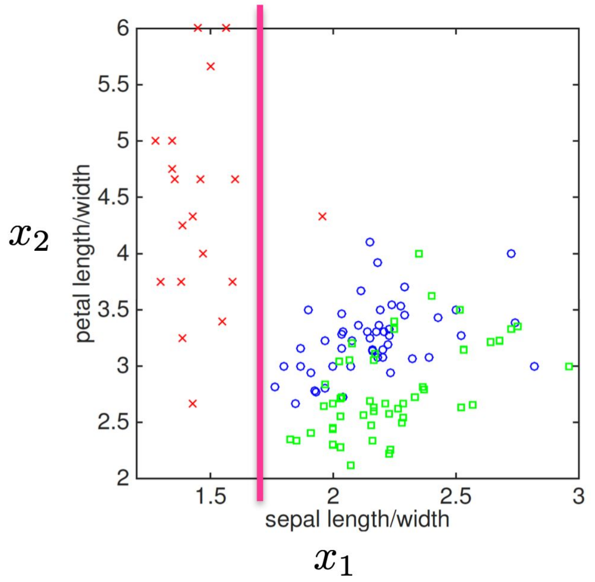
- $\mathcal{X} = \mathbb{R}^2, \mathcal{Y} = \{1, 2, 3\}$
- x_1 : ratio of sepal length to width
- x_2 : ratio of petal length to width



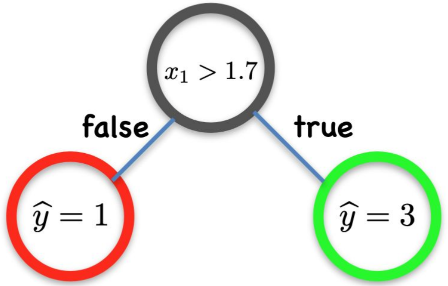
We choose the feature x_1 as the splitting feature with value 1.7 and split training data: all the training samples with value of x_1 larger than 1.7 go to left and the rest go to right.

Example

Classifying irises flower images into three types by sepal and petal measurements



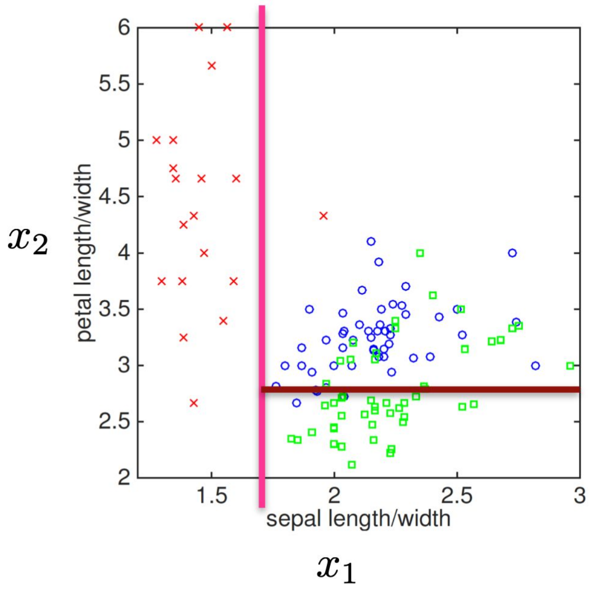
- $\mathcal{X} = \mathbb{R}^2, \mathcal{Y} = \{1, 2, 3\}$
- x_1 : ratio of sepal length to width
- x_2 : ratio of petal length to width



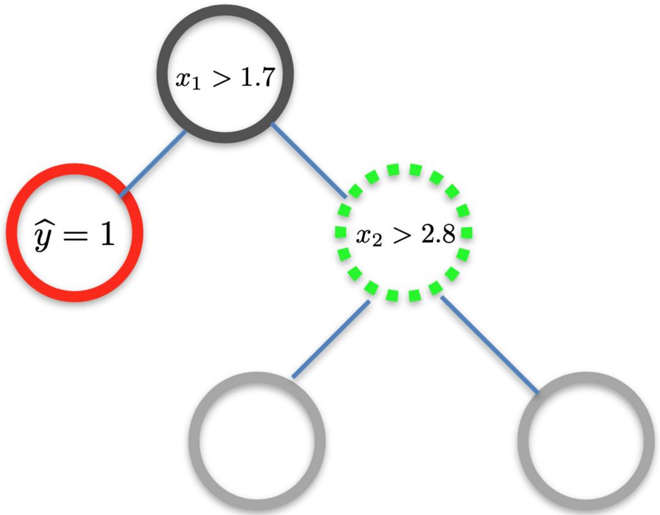
For two new leaf nodes, we decide the label by the **majority** of labels (red (1) and green (3))

Decision trees

Classifying irises flower images into three types by sepal and petal measurements



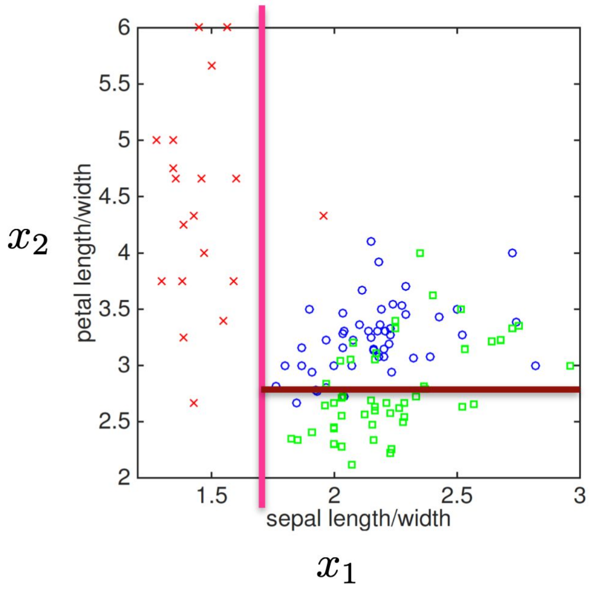
- $\mathcal{X} = \mathbb{R}^2, \mathcal{Y} = \{1, 2, 3\}$
- x_1 : ratio of sepal length to width
- x_2 : ratio of petal length to width



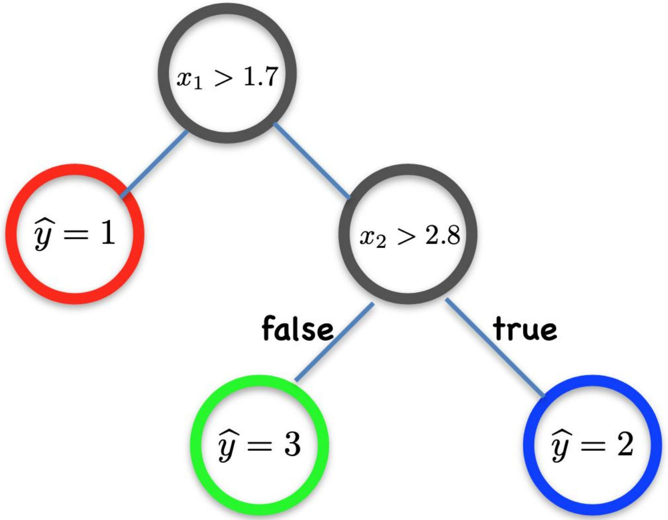
No need to split left node (already **pure** by all red labels). For the right leaf node, we choose x_2 with threshold 2.8 as the splitting node and split the training data in this node accordingly.

Decision trees

Classifying irises flower images into three types by sepal and petal measurements

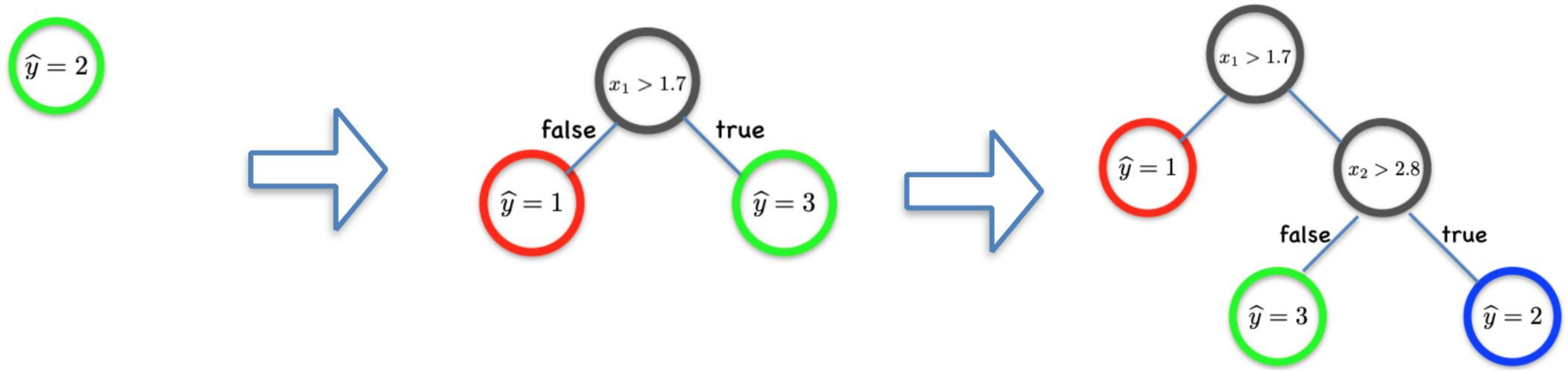


- $\mathcal{X} = \mathbb{R}^2, \mathcal{Y} = \{1, 2, 3\}$
- x_1 : ratio of sepal length to width
- x_2 : ratio of petal length to width



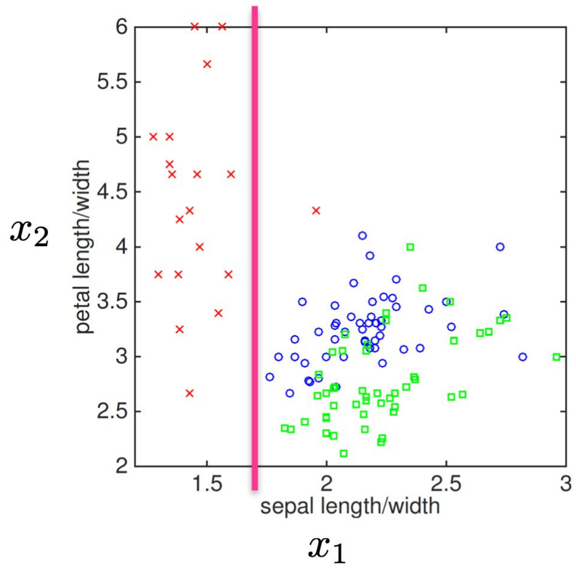
For two newly generated leaf nodes, we decide the label by **majority voting**. These two nodes are not pure, but pure enough to **stop training**.

Top-down construction



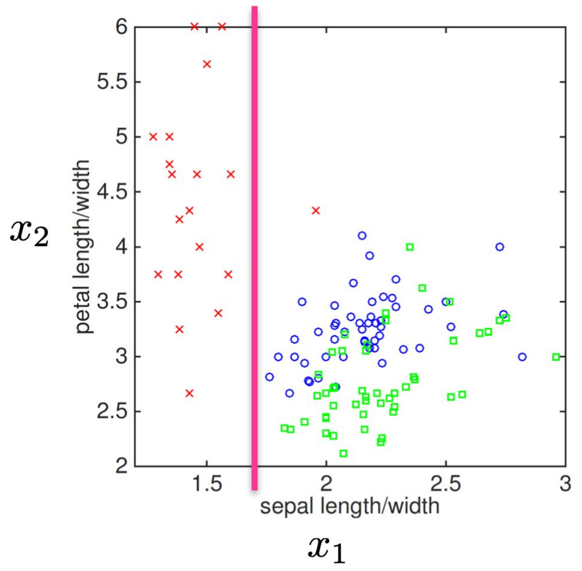
Key question to learn decision trees: How to split a node?

Question: **What feature** and **what values** we should split the data?



Key question to learn decision trees: How to split a node?

Question: **What feature** and **what values** we should split the data?



Answer: A good attribute should "**reduce uncertainty**" and result in "**gain in information**".

How much information do we gain if we disclose the value of some attribute?

Uncertainty before splitting - Uncertainty after splitting

Entropy: measuring uncertainty

Entropy is a measure of **uncertainty** (**impurity**) associated with a random variable

$$H(X) = -\sum_{x \,\in\, \text{values}(X)} p(x)\log_2(p(x))$$

Entropy: measuring uncertainty

Entropy is a measure of **uncertainty** (**impurity**) associated with a random variable

$$H(X) = -\sum_{x \in \text{values}(X)} p(x)\log_2(p(x))$$

For a Bernoulli random variable we have:
[a coin with probability p of Head]

$$H(X) = -\left[p\log_2 p + (1-p)\log_2(1-p)\right]$$

Entropy: measuring uncertainty

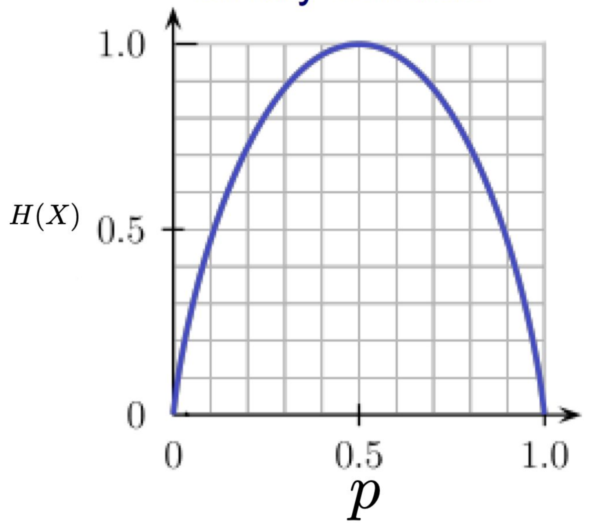
Entropy is a measure of **uncertainty** (**impurity**) associated with a random variable

$$H(X) = - \sum_{x \,\in\, \text{values}(X)} p(x) \log_2(p(x))$$

For a Bernoulli random variable we have:
[a coin with probability p of Head]

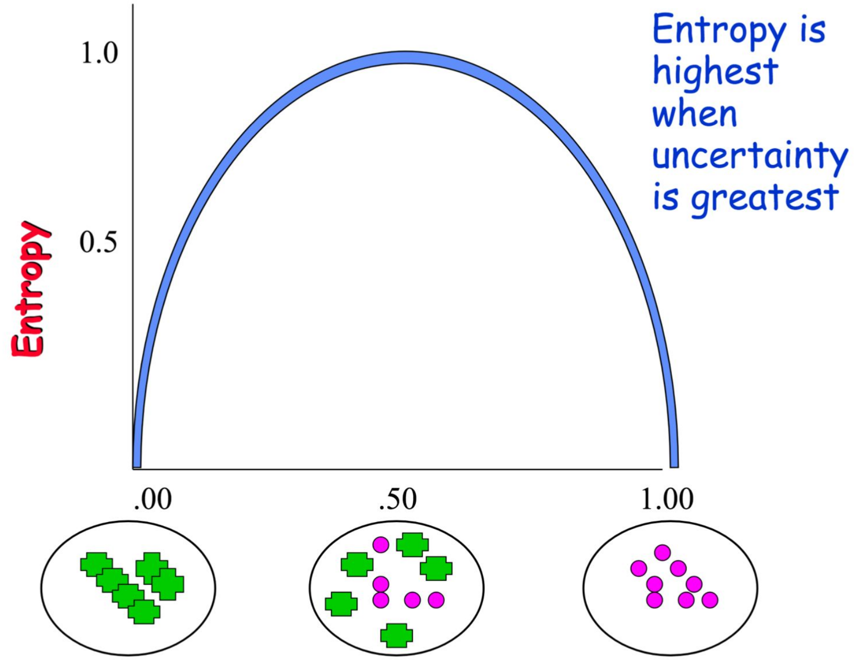
$$H(X) = -\left[p \log_2 p + (1-p) \log_2(1-p)\right]$$

entropy function for binary variable



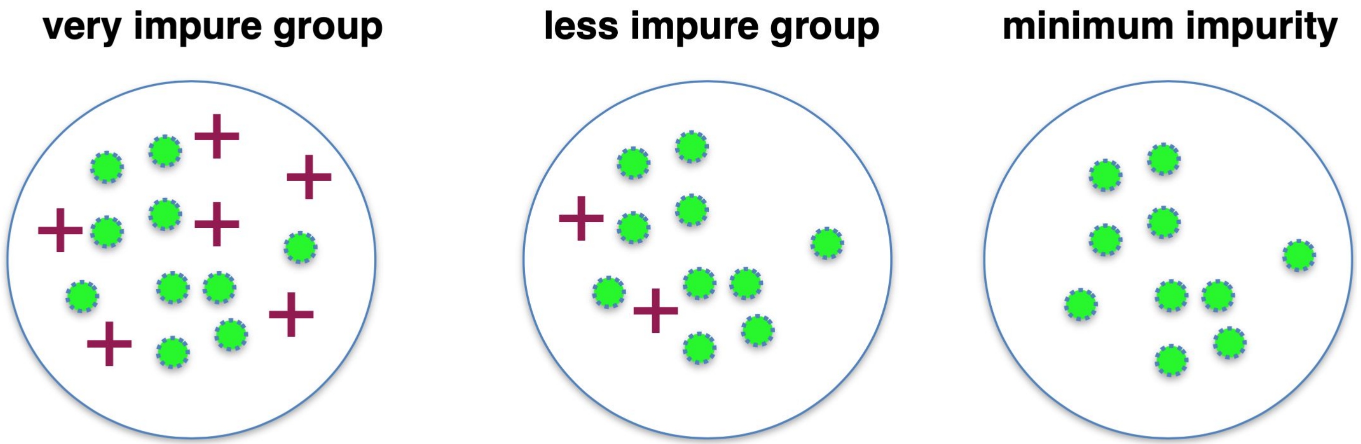
Entropy of a sample binary data

- When all examples belong to the same class entropy = 0 (minimum entropy)
- Entropy of a sample with 50% in each class is 1 (maximum entropy)



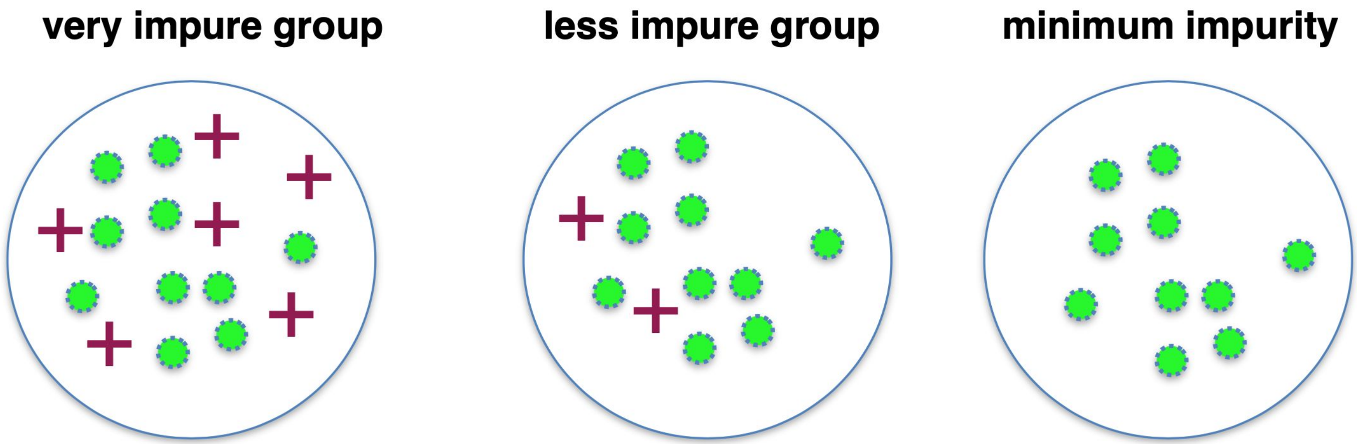
Example

Entropy comes from information theory. The higher the entropy, the more the information content!



Example

Entropy comes from information theory. The higher the entropy, the more the information content!



We have two type of data (classes)

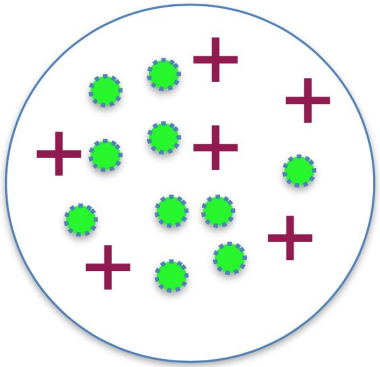
Let $p_i, i = 1, 2$ be the probability of each class

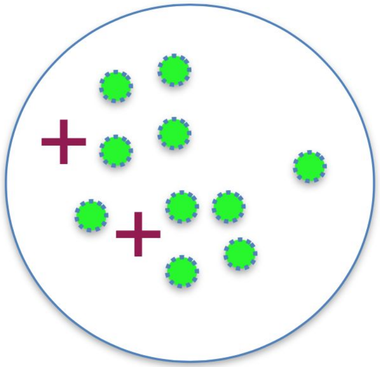
We can compute the entropy of each group $-\sum p_i \log p_i$

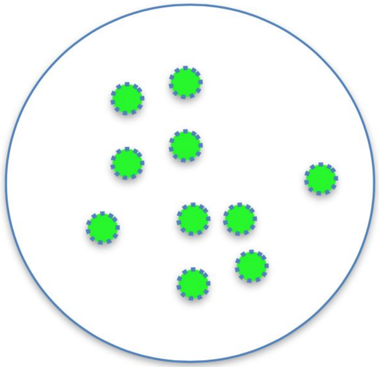
Example

Entropy comes from information theory. The higher the entropy, the more the information content!

very impure group **less impure group** **minimum impurity**







$$-\left[\frac{10}{16}\log_2(\frac{10}{16}) + \frac{6}{16}\log_2(\frac{6}{16})\right] = 0.924$$

$$-\left[\frac{10}{12}\log_2(\frac{10}{12}) + \frac{2}{12}\log_2(\frac{2}{12})\right] = 0.650$$

$$-1\log_2 1 = 0$$

We have two type of data (classes)

Let $p_i, i = 1, 2$ be the probability of each class

We can compute the entropy of each group $-\sum p_i \log p_i$

Information gain

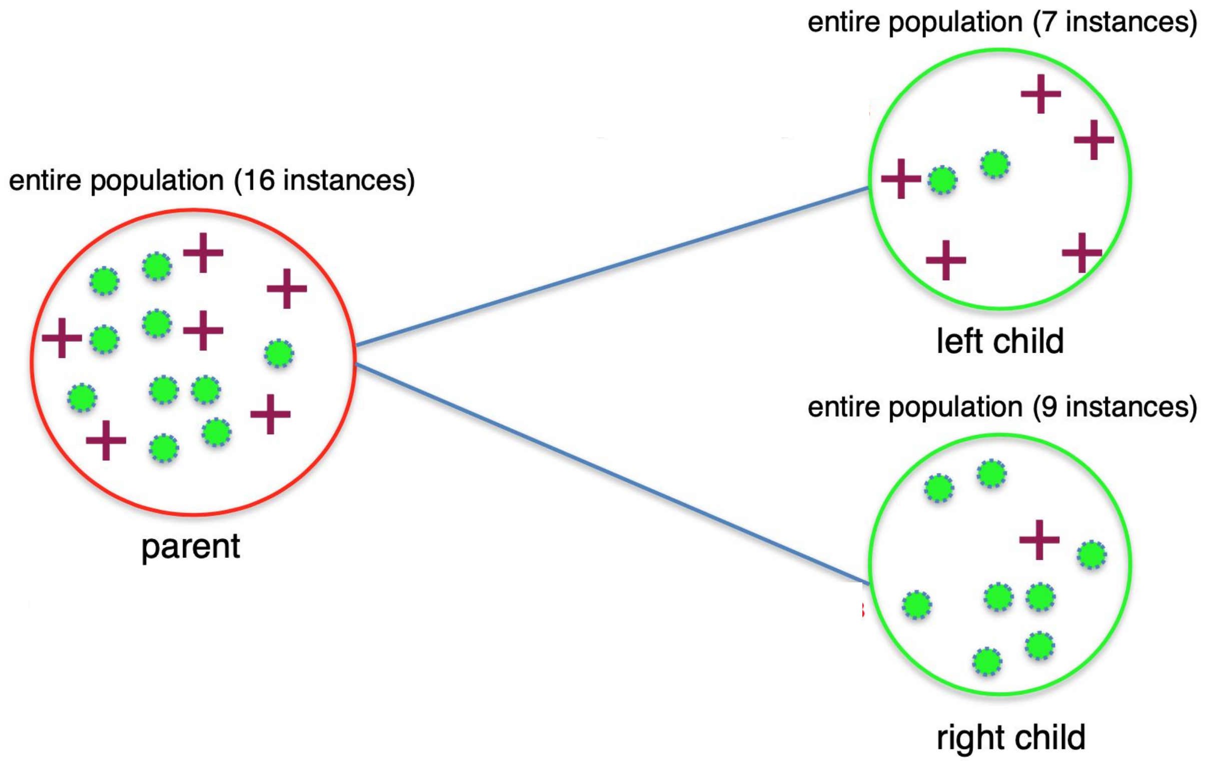
Recall, we would like to determine which attribute (feature) in a given set of features is most useful for discriminating between the classes to be learned

Information gain = uncertainty before splitting - uncertainty after splitting

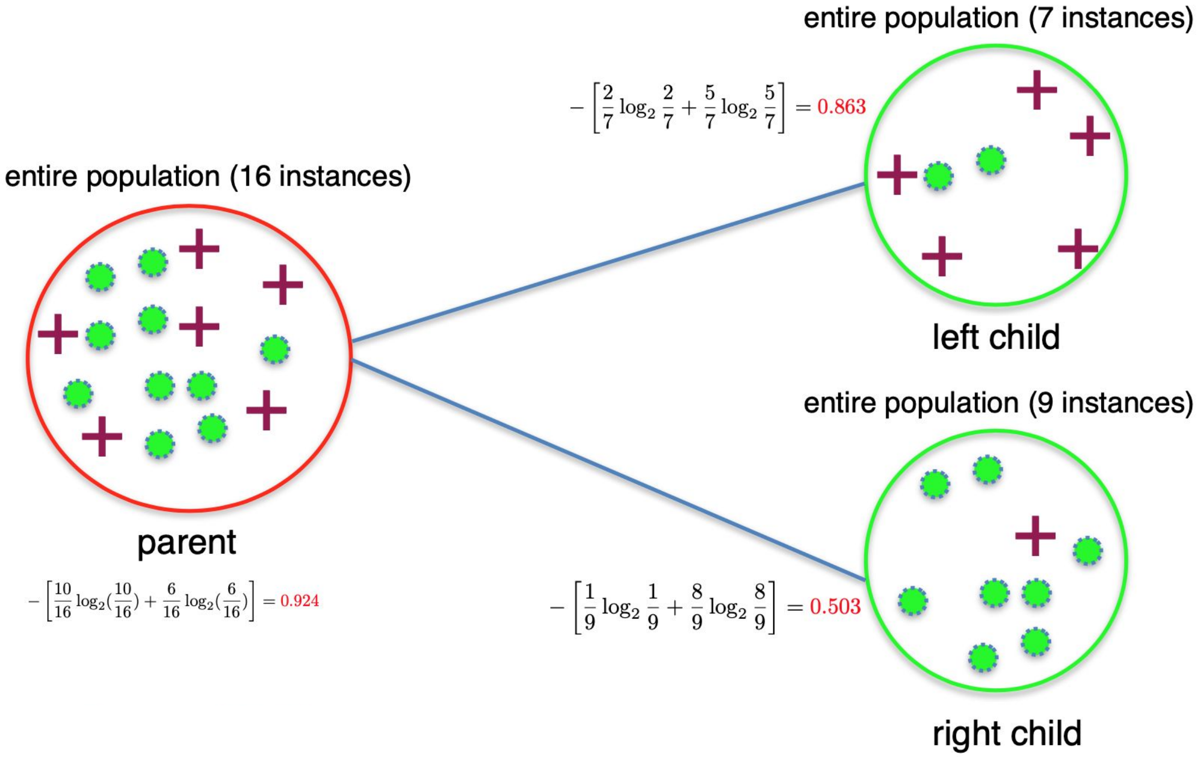
This tells us how important a given feature is!

Compute the information gain for all features, and pick the one as final splitter that maximizes information gain!

How to compute information gain


entire population (16 instances)
parent
entire population (7 instances)
left child
entire population (9 instances)
right child

How to compute information gain



entire population (16 instances)
parent
$-\left[\frac{10}{16}\log_2(\frac{10}{16}) + \frac{6}{16}\log_2(\frac{6}{16})\right] = 0.924$
entire population (7 instances)
$-\left[\frac{2}{7}\log_2\frac{2}{7} + \frac{5}{7}\log_2\frac{5}{7}\right] = 0.863$
left child
entire population (9 instances)
$-\left[\frac{1}{9}\log_2\frac{1}{9} + \frac{8}{9}\log_2\frac{8}{9}\right] = 0.503$
right child

Conditional entropy for calculating entropy after splitting

What’s the entropy of Y if we condition on some other variable X ?

Here, Y is the label, X is the feature to split!

Conditional entropy for calculating entropy after splitting

What's the entropy of *Y* if we condition on some other variable *X* ?
Here, Y is the label, X is the feature to split!

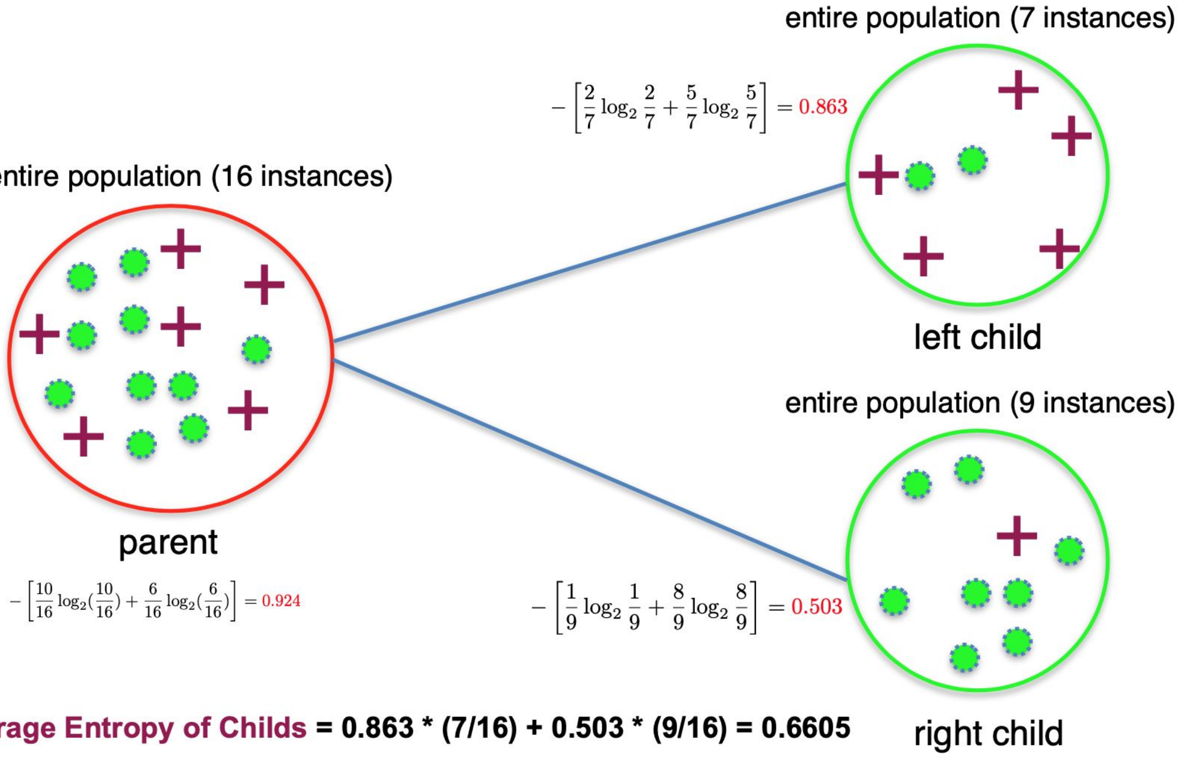
$$H(Y|X) = \sum_{x\ \in \text{values}(X)} p(X=x) H(Y|X=x)$$

where

$$H(Y|X=x) = -\sum_{y\ \in \text{values}(Y)} p(Y=y|X=x) \log_2 p(Y=y|X=x)$$

How to compute information gain

weighted average entropy of childs


entire population (7 instances)
left child
entire population (16 instances)
parent
entire population (9 instances)
right child

$$-\left[\frac{2}{7}\log_2\frac{2}{7}+\frac{5}{7}\log_2\frac{5}{7}\right]=0.863$$

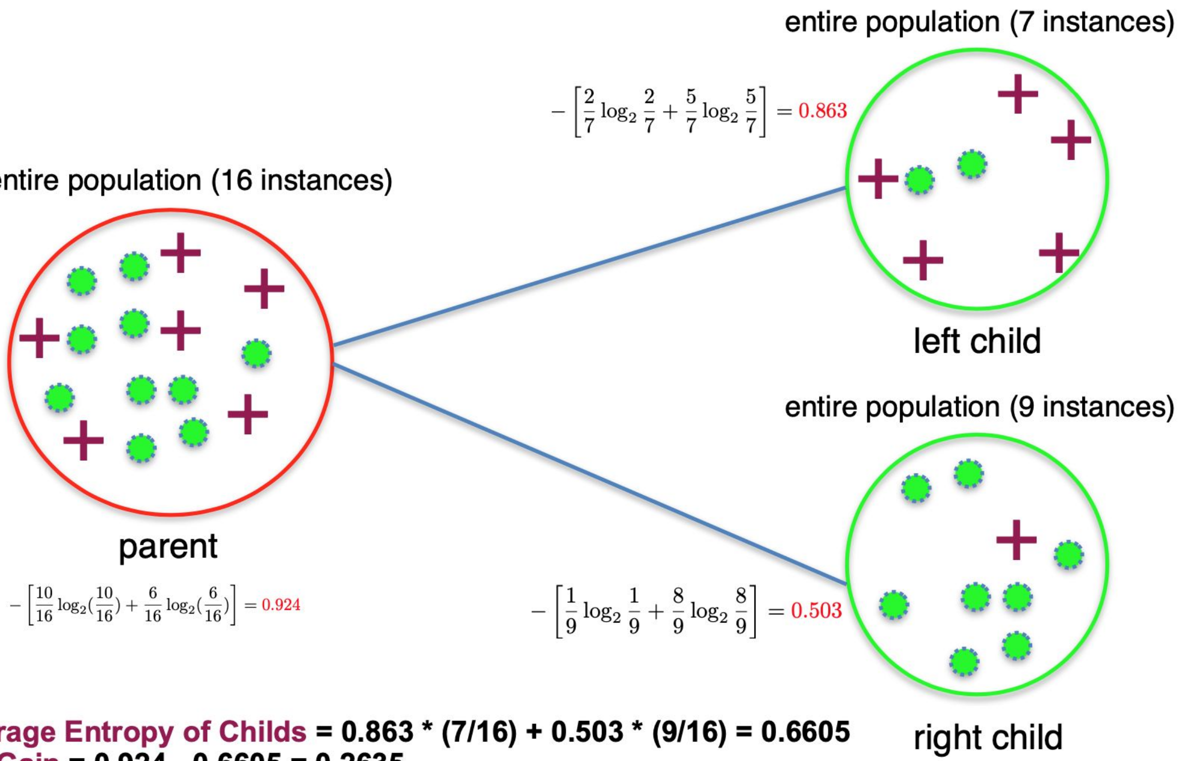
$$-\left[\frac{10}{16}\log_2(\frac{10}{16})+\frac{6}{16}\log_2(\frac{6}{16})\right]=0.924$$

$$-\left[\frac{1}{9}\log_2\frac{1}{9}+\frac{8}{9}\log_2\frac{8}{9}\right]=0.503$$

Average Entropy of Childs = 0.863 * (7/16) + 0.503 * (9/16) = 0.6605

How to compute information gain

Information Gain = entropy of parent - weighted average entropy of childs


entire population (16 instances)
parent
entire population (7 instances)
left child
entire population (9 instances)
right child

$$-\left[\frac{10}{16}\log_2(\frac{10}{16})+\frac{6}{16}\log_2(\frac{6}{16})\right]=0.924$$

$$-\left[\frac{2}{7}\log_2\frac{2}{7}+\frac{5}{7}\log_2\frac{5}{7}\right]=0.863$$

$$-\left[\frac{1}{9}\log_2\frac{1}{9}+\frac{8}{9}\log_2\frac{8}{9}\right]=0.503$$

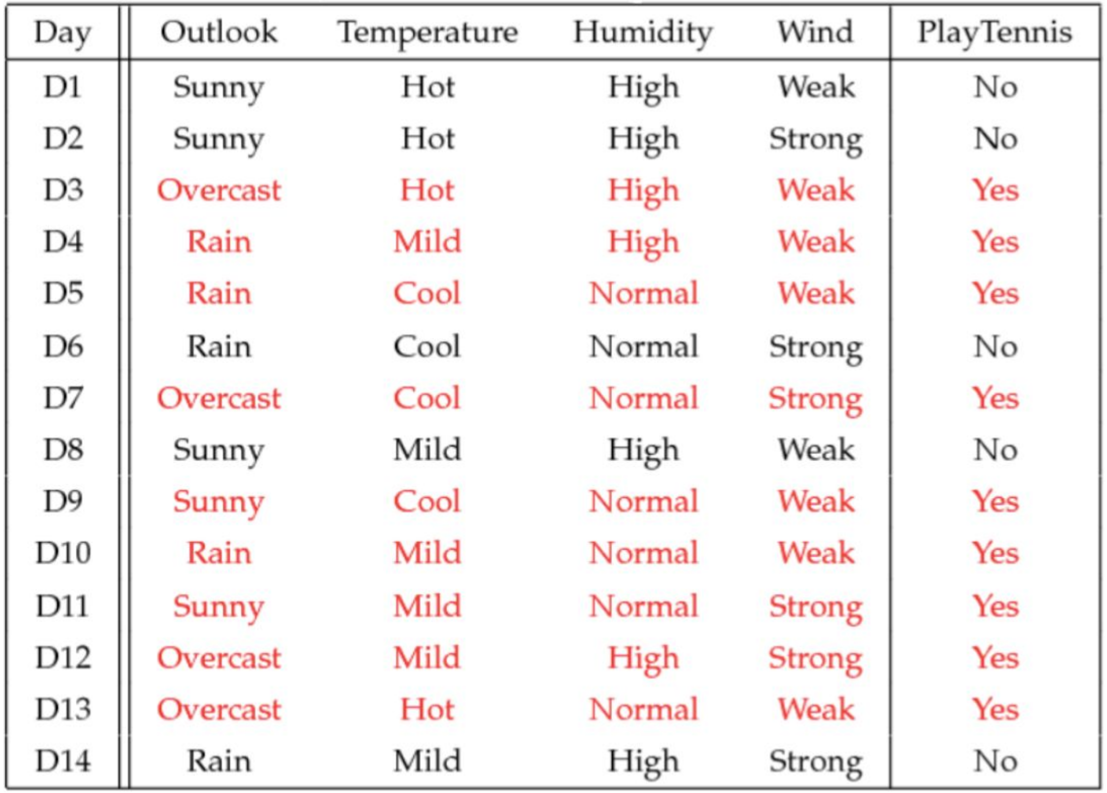
Average Entropy of Childs = 0.863 * (7/16) + 0.503 * (9/16) = 0.6605
InfoGain = 0.924 - 0.6605 = 0.2635

Example: Tennis Data

Attributes and their values:

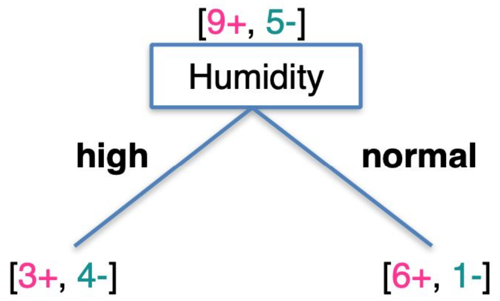
- Outlook: Sunny, Overcast, Rain
- Humidity: High, Normal
- Wind: Strong, Weak
- Temperature: Hot, Mild, Cool

Target label- Play Tennis: {Yes, No}



Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Splitting for tennis data



$$H(Y) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940$$

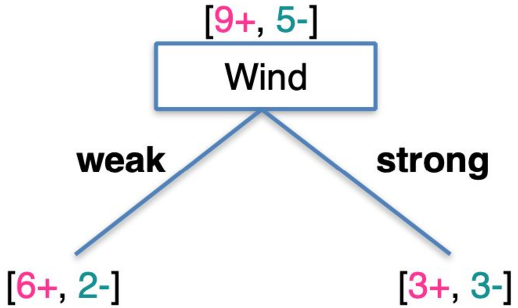
$$H(Y \mid \text{high}) = -\frac{3}{7}\log_2\left(\frac{3}{7}\right) - \frac{4}{7}\log_2\left(\frac{4}{7}\right) = 0.985$$

$$H(Y \mid \text{normal}) = -\frac{6}{7}\log_2\left(\frac{6}{7}\right) - \frac{1}{7}\log_2\left(\frac{1}{7}\right) = 0.592$$

Y is the label as a random variable

$$\begin{aligned}\text{InfoGain}(\text{Humidity}) &= H(Y) - H(Y \mid \text{Humidity}) \\ &= H(Y) - \left[\frac{7}{14}H(Y \mid \text{high}) + \frac{7}{14}H(Y \mid \text{normal})\right] \\ &= 0.940 - \left[\frac{7}{14}(0.985) + \frac{7}{14}(0.592)\right] \\ &= 0.151\end{aligned}$$

Splitting for tennis data

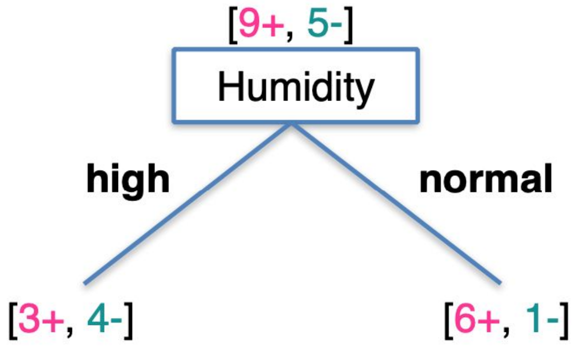


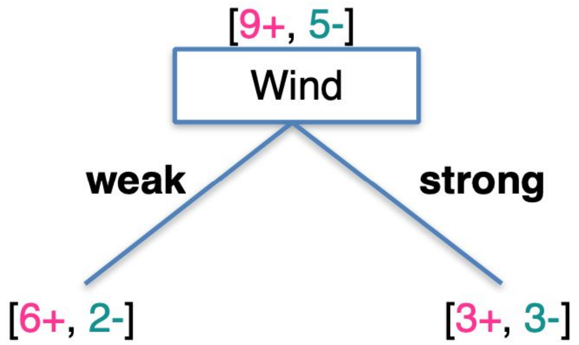
$$H(Y \mid \text{weak}) = 0.811$$

$$H(Y \mid \text{strong}) = 1$$

$$\begin{aligned} \text{InfoGain}(\text{Wind}) &= H(Y) - H(Y \mid \text{Wind}) \\ &= H(Y) - \left[\frac{8}{14} H(Y \mid \text{weak}) + \frac{6}{14} H(Y \mid \text{strong})\right] \\ &= 0.940 - \left[\frac{8}{14}(0.811) + \frac{6}{14}(1.0)\right] \\ &= 0.048 \end{aligned}$$

Example splitting

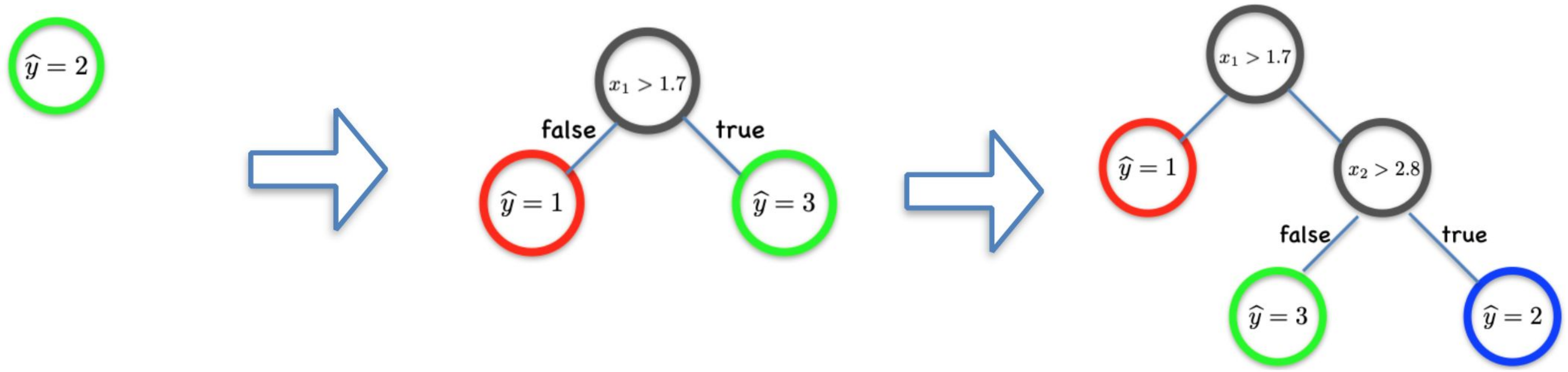




✓ $$\text{InfoGain}(\text{Humidity}) = H(Y) - H(Y \mid \text{Humidity}) = 0.151$$

$$\text{InfoGain}(\text{Wind}) = H(Y) - H(Y \mid \text{Wind}) = 0.048$$

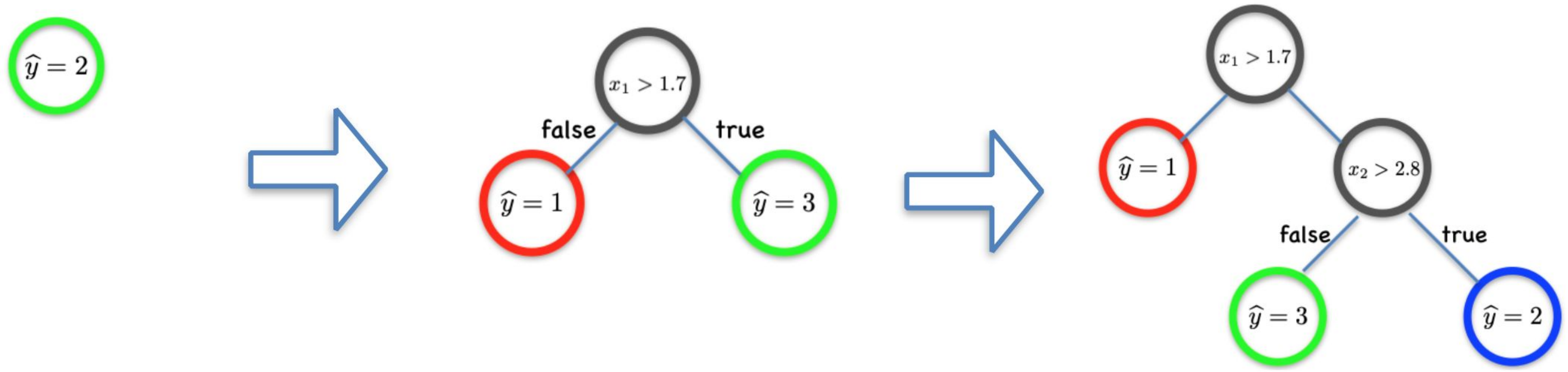
Top-down construction



Goal: Find a decision tree that achieves minimum misclassification error on the training data and avoids overfitting.

In fact, we can achieve **zero training error** trivially: how?

Top-down construction



Goal: Find a decision tree that achieves minimum misclassification error on the training data and avoids overfitting.

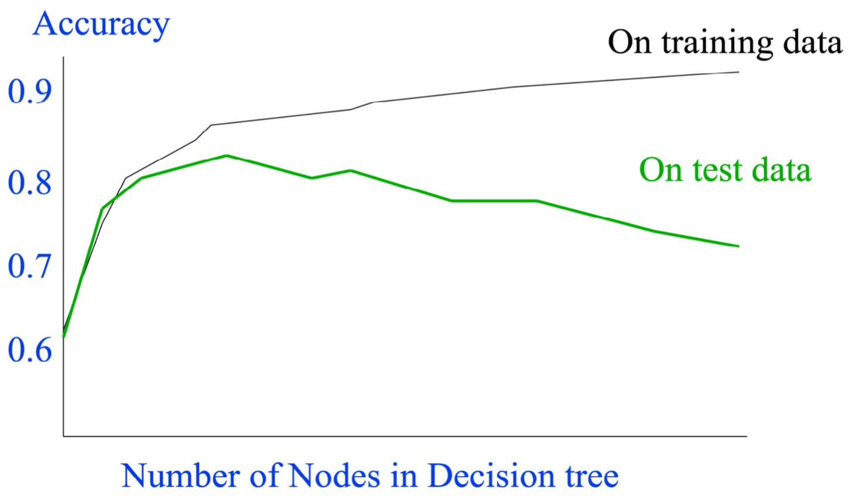
In fact, we can achieve **zero training error** trivially: how?

just create a decision tree with one path from root to leaf for each training example!

We will get back to **overfitting** issue later.

Overfitting in decision trees

How deep we should do splitting (bias-variance)

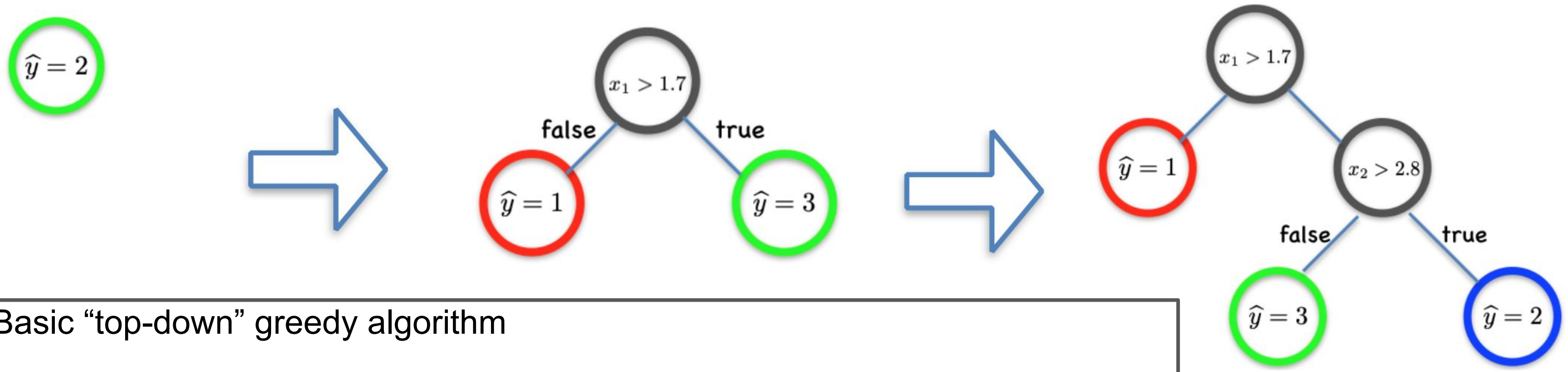


Training error goes to zero as the number of nodes in the tree increases.

Test error decreases initially, but eventually increases due to overfitting

Among all trees, find the smallest tree that minimizes training error!

Top-down construction



Basic “top-down” greedy algorithm

Initially, tree is a single leaf node containing all (training) data.
Loop:
- Pick a leaf node and splitting rule *h* that maximally reduces uncertainty.
- Split data in node using *h*, and grow tree accordingly.

Until some stopping criterion is satisfied.

Rules to stop splitting

How deep we should do splitting (bias-variance)

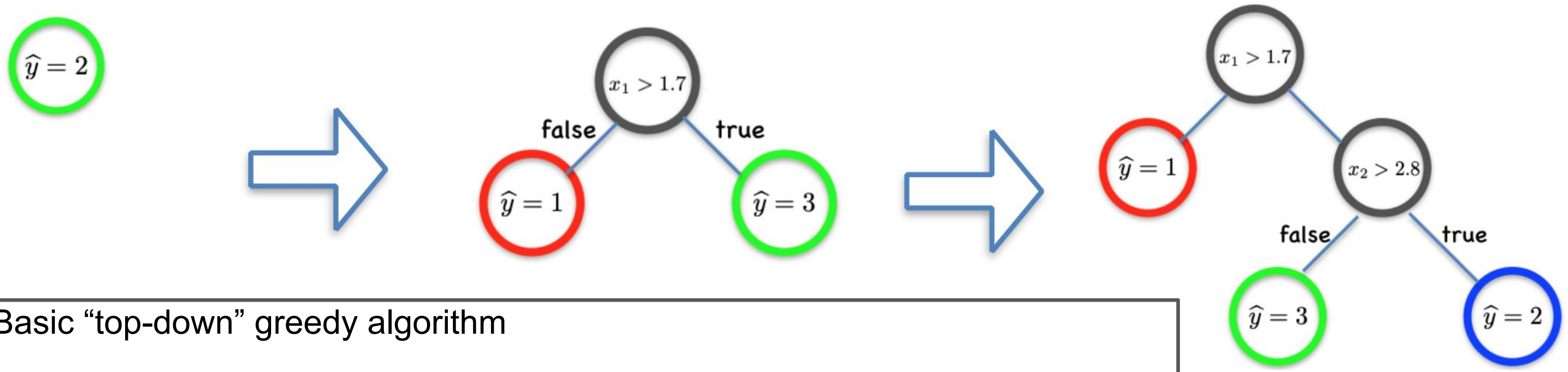
We might continue splitting until:

- The leaves each have only one label
- We reach a user-defined maximum depth

There are more complicated rules for deciding when to stop split.

- Don’t split any nodes where the number of objects is less than some
- Don’t split any nodes that create children with less than ‘m’ objects.
- Don’t split the node if it decreases an approximation of test accuracy (using a validation set)

Top-down construction



```
Basic "top-down" greedy algorithm

Initially, tree is a single leaf node containing all (training) data.
Loop:
        - Pick a leaf node and splitting rule h that maximally reduces uncertainty.
        - Split data in node using h, and grow tree accordingly.
Until some stopping criterion is satisfied.
```

Label of a leaf is the plurality label among the data contained in the leaf.

- For **classification**: the **majority** of labels in the training data of node
- For **regression**, e.g., the **average** of labels

Advantages and disadvantages

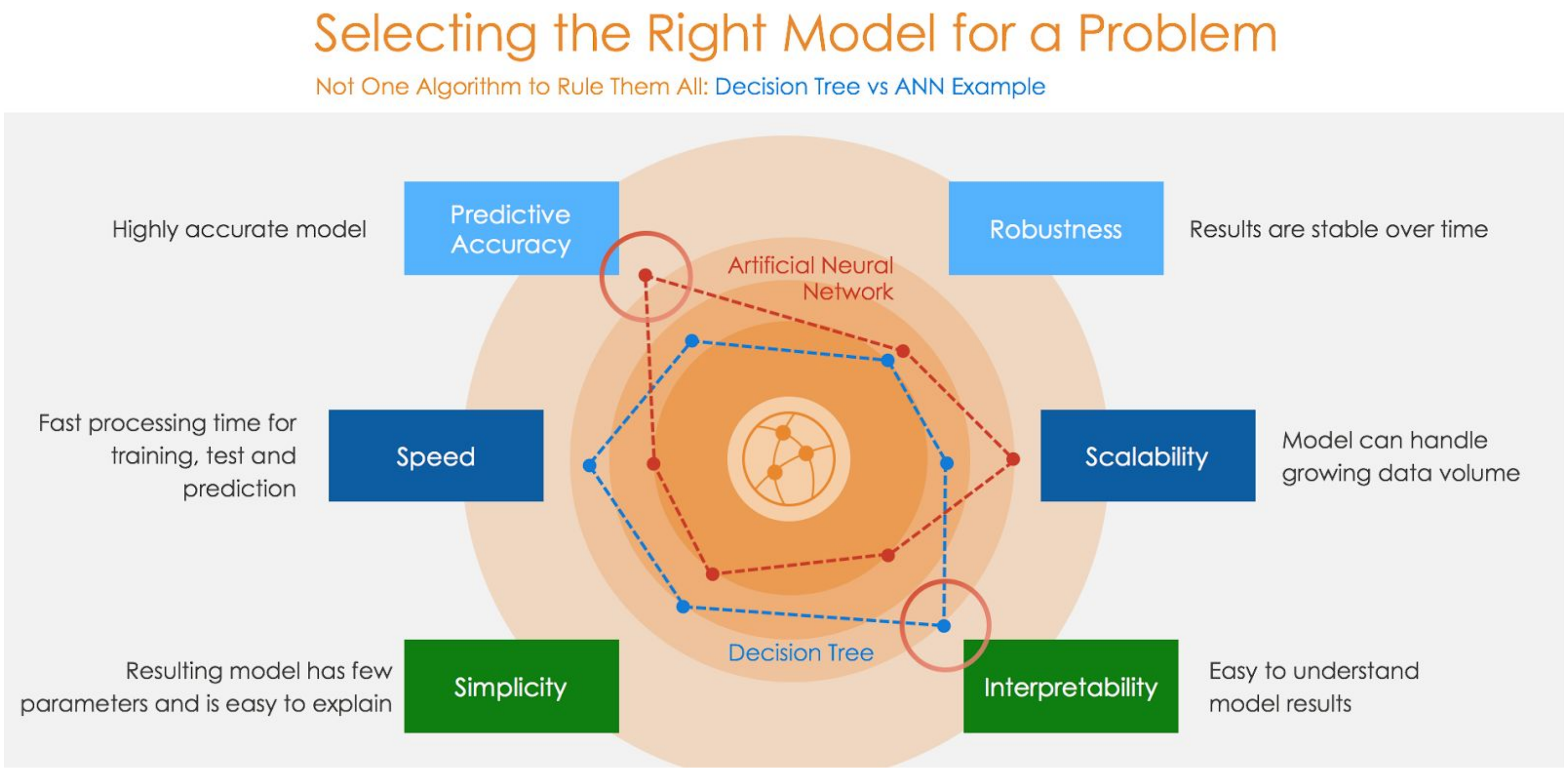
Advantages:

- Interpretable
- Fast to learn
- Very fast to classify
- Can handle missing values (create a branch for NAs)

Disadvantages:

- Hard to find optimal set of rules
- Greedy splitting often not accurate, requires very deep trees
- Limitations of uncertainty notions

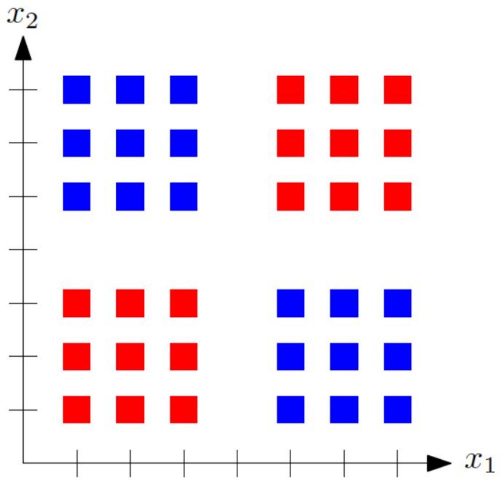
Decision trees versus NNs


Selecting the Right Model for a Problem
Not One Algorithm to Rule Them All: Decision Tree vs ANN Example
Highly accurate model
Predictive Accuracy
Robustness
Results are stable over time
Artificial Neural Network
Fast processing time for training, test and prediction
Speed
Scalability
Model can handle growing data volume
Decision Tree
Resulting model has few parameters and is easy to explain
Simplicity
Interpretability
Easy to understand model results

Source: https://www.teradata.com/Blogs/Occam%E2%80%99s-razor-and-machine-learning

Limitations of uncertainty notions

Suppose Y = {red, blue} , and the data is as follows:

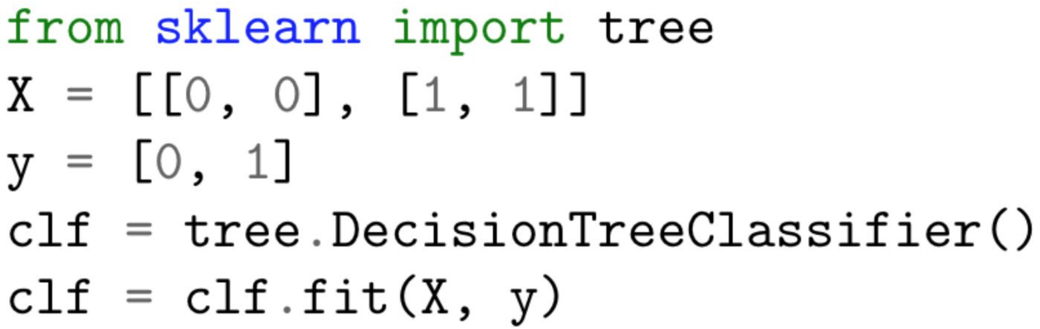


Every split of the form of 'values large than a threshold' provides no reduction in uncertainty

Take away: Zero reduction in uncertainty may not be a good stopping condition

Decision trees in scikit-learn

The **`DecisionTreeClassifier`** takes as input two arrays: an array **X**, sparse or dense, of size `[n_samples, n_features]` holding the training samples, and an array **y** of integer values, size `[n_samples]`, holding the class labels for the training samples.



```python
from sklearn import tree
X = [[0, 0], [1, 1]]
y = [0, 1]
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, y)
```

Decision trees for iris data in scikit-learn

```
from sklearn.datasets import load_iris
iris = load_iris()
X = iris.data
y = iris.target

from sklearn import tree
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, y)
```