CS M146: Introduction to Machine Learning
Decision Trees

Aditya Grover

The instructor gratefully acknowledges Sriram Sankararaman (UCLA) for some of the materials and organization used in these slides, and many others who made their course materials freely available online.

https://aditya-grover.github.io/ @adityagrover_

Sample Dataset

Columns denote features $\boldsymbol{x}^{(i)}$ and labels $y^{(i)}$

Rows denote labelled training instance $(\boldsymbol{x}^{(i)}, y^{(i)})$

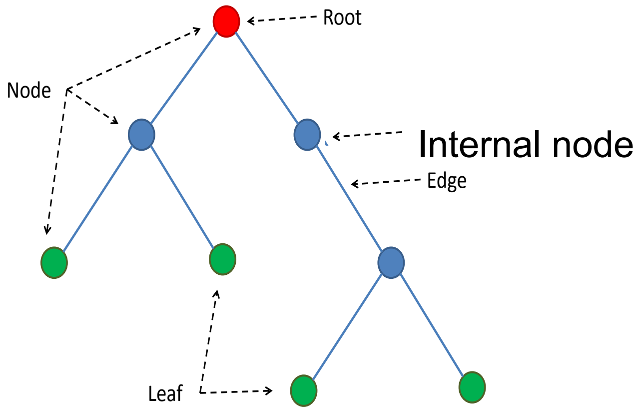
Predictors				Response
Outlook	**Temperature**	**Humidity**	**Wind**	**Class**
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

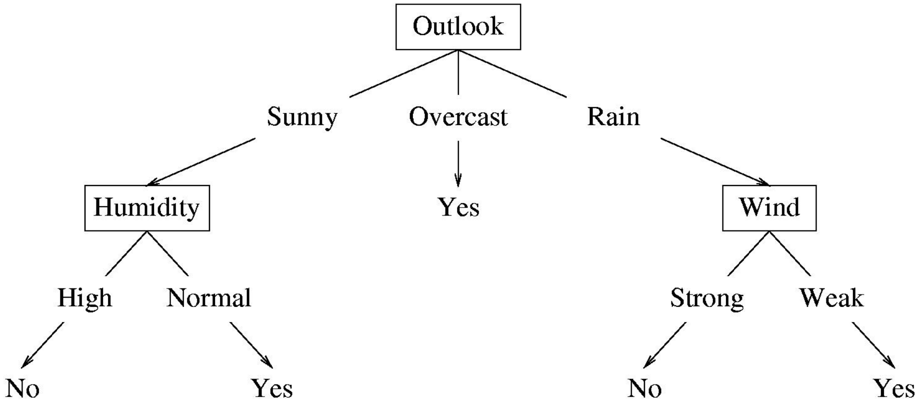
$(\boldsymbol{x}^{(i)}, y^{(i)})$

Can you describe a ML model that could be used to decide whether to play tennis given weather?

Decision Tree

Think of predictions as making a decision by walking down a tree

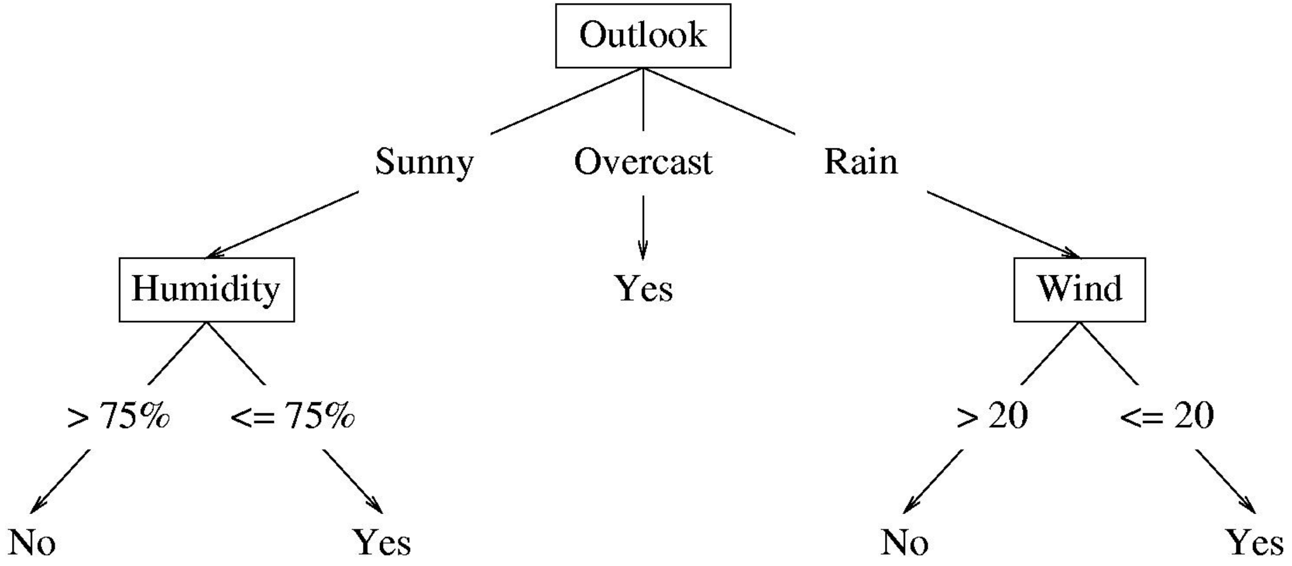




- Each **internal node**: test one feature $x_j^{(i)}$
- Each **branch from node**: selects one value for $x_j^{(i)}$
- Each **leaf node**: predict $y^{(i)}$

Decision Tree

- If features are continuous, internal nodes can test value of feature against threshold



Slide credit: Eric Eaton

Decision Trees

Given

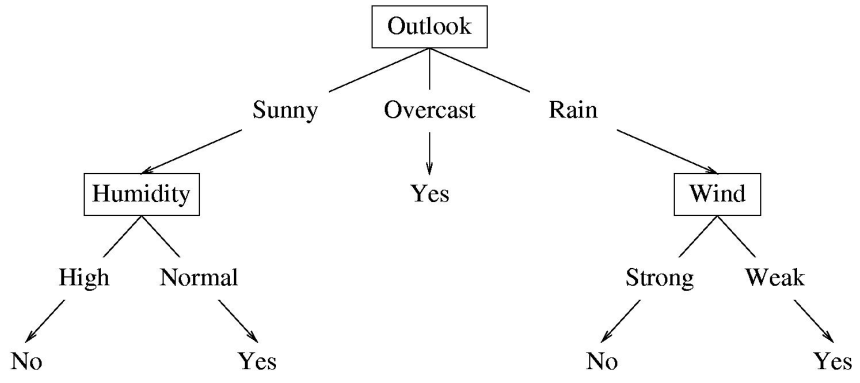
A training dataset D consisting of n labelled training examples

- Input **features** $\mathbf{X} = \{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n)}\}$ where $\boldsymbol{x}^{(i)} \in \mathcal{X}$
- Corresponding **labels** $\boldsymbol{y} = \{y^{(1)}, \dots, y^{(n)}\}$ where $y^{(i)} \in \mathcal{Y}$

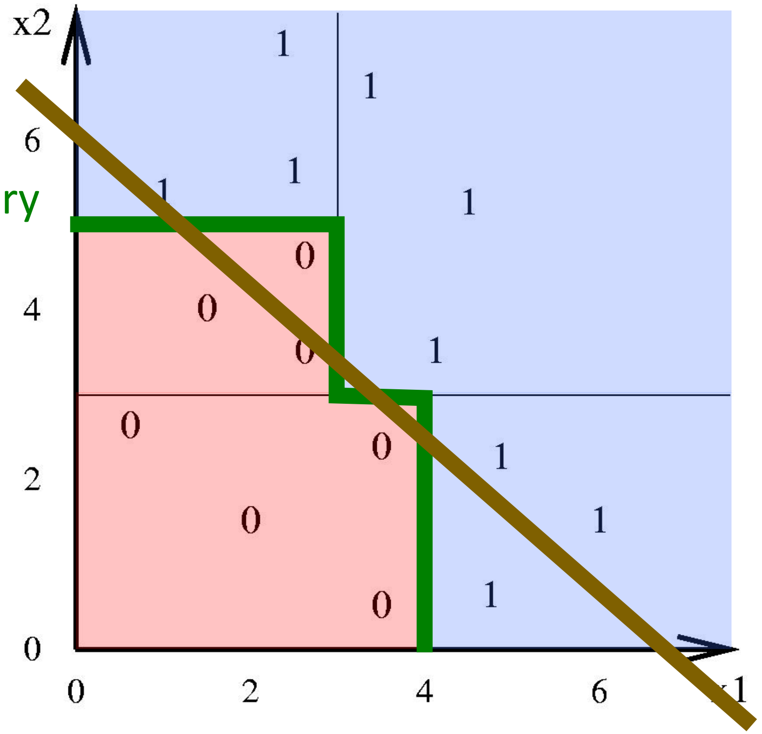
Output

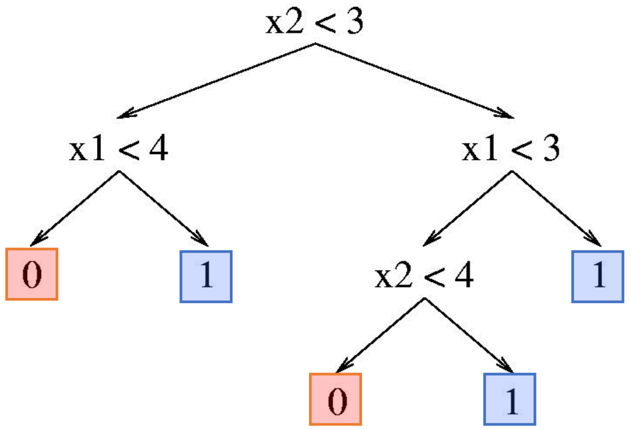
Hypothesis function $h: \mathcal{X} \to \mathcal{Y}$ such that $h(\boldsymbol{x}) \approx y$

- each hypothesis h is a **decision tree**
- trees sort $\boldsymbol{x}$ to leaf, which assigns y



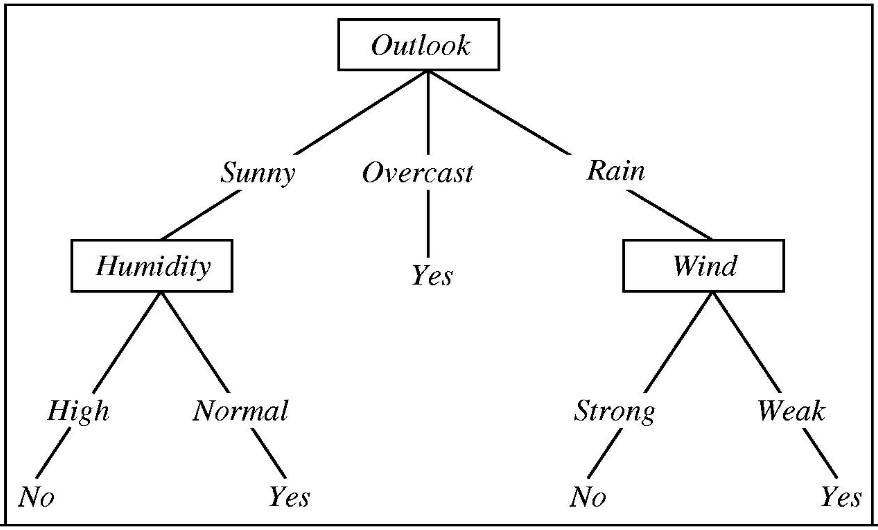
Decision Tree – Decision Boundary





- The decision boundary of a decision trees (w.r.t. default features) can be specified as axis-parallel (hyper) rectangles
- Each rectangular region is labeled with one label

Interpretability of Decision Trees



Is it good weather for playing Tennis?

IF (*Outlook* = *Sunny*) AND (*Humidity* = *High*)
THEN *PlayTennis* = *No*

IF (*Outlook* = *Sunny*) AND (*Humidity* = *Normal*)
THEN *PlayTennis* = *Yes*

. . .

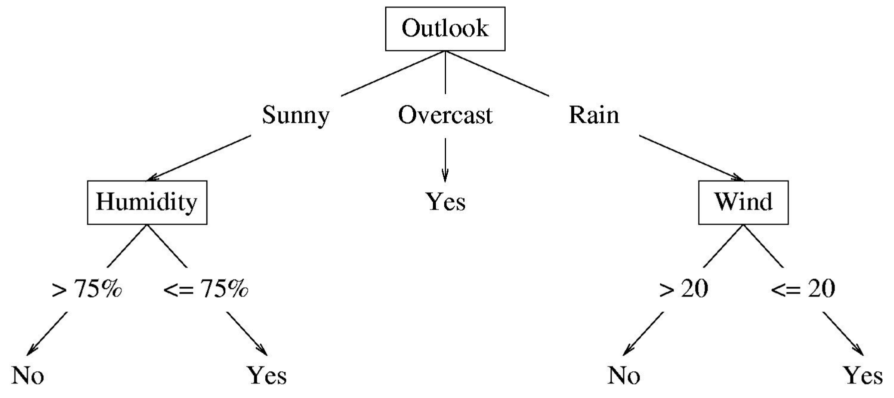
- Easy to convert decision trees to if-else rules
- Very widely used in scenarios where interpretability of ML algorithms is important e.g., healthcare

Slide credit: Eric Eaton
Based on slide by Pedro Domingos

Learning Decision Trees

For any decision tree, we need to learn 3 things:

1. Structure i.e., which nodes appear where
2. Labels for the leaves
3. (if applicable) Threshold values for continuous features



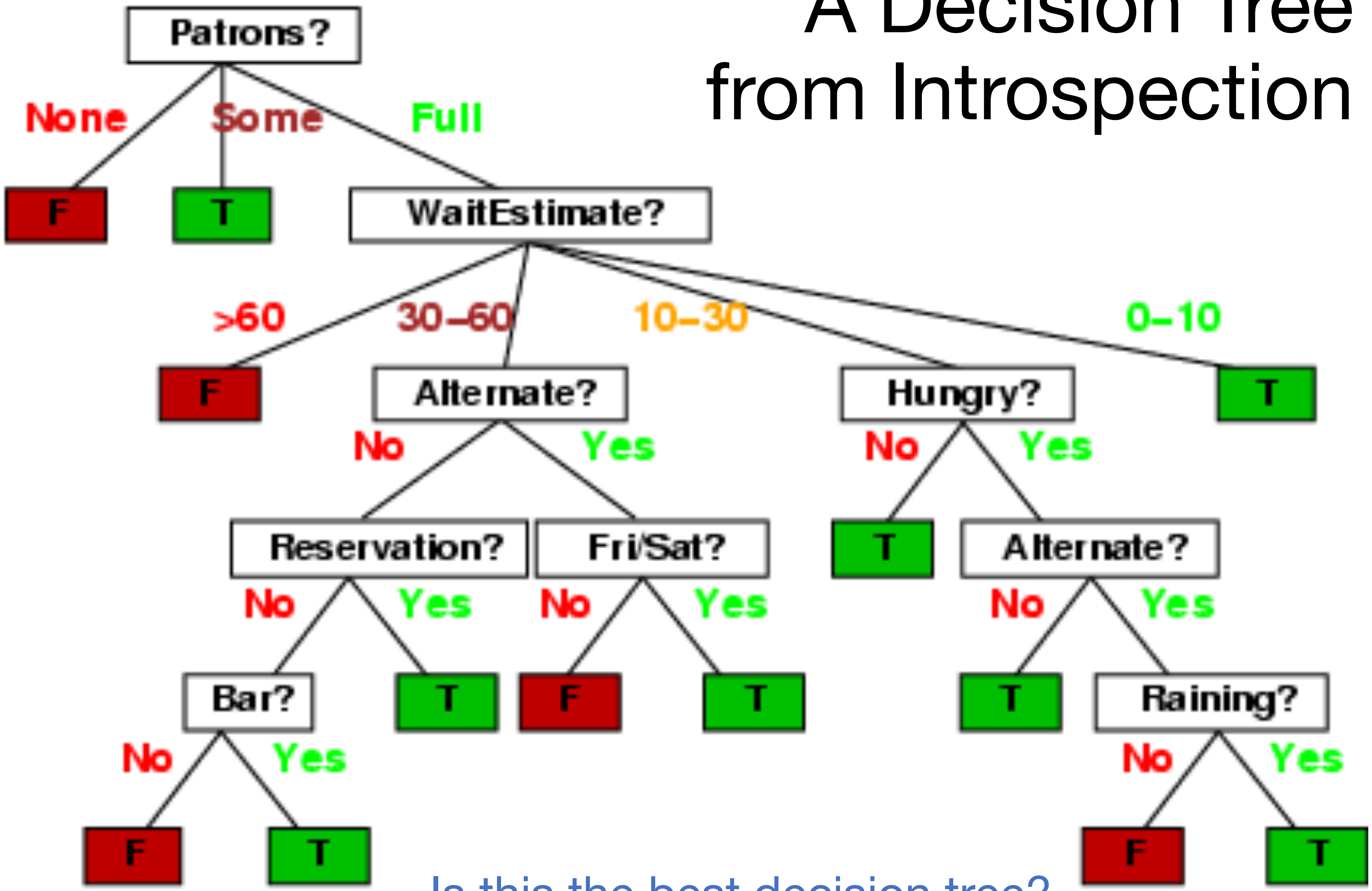
Example: Restaurant Domain (Russell & Norvig)

Model patron's decision of whether to wait for table at restaurant

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

Slide credit: Eric Eaton

A Decision Tree from Introspection



Is this the best decision tree?

Slide credit: Eric Eaton

Preference bias: Occam's Razor

Principle stated by William of Ockham (1285-1347)

- AKA **Occam's Razor**, Law of Economy, or Law of Parsimony

Idea: The **simplest consistent** explanation is the best

Therefore, the smallest decision tree that correctly classifies all of training examples is best

- finding provably smallest decision tree is NP-hard
- ... so instead of constructing the absolute smallest tree consistent with training examples, construct one that is pretty small

Slide credit: Eric Eaton

Iterative Dichotomiser (ID3) Algorithm

[Ross Quinlan]

Initialize *current node* ← root and S ← all input attributes

Main **loop**:

1. Compute A ← "best" decision feature in S.
2. Assign A as decision feature for *current node*.
3. For each value of A, create a branch for a new *child node*.
4. Sort training examples along each branch.
5. If training examples are perfectly classified, assign *child node* as leaf node and stop. Else, assign *child node* as an internal node and recurse **loop** over these internal nodes and features $S - A$.

How do we choose which attribute is best?

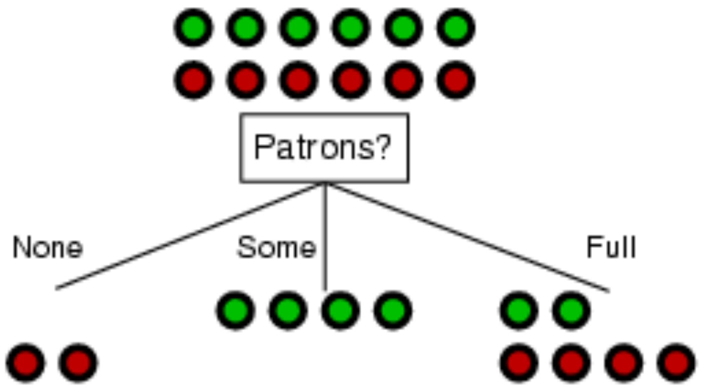
Choosing the **Best** Attribute

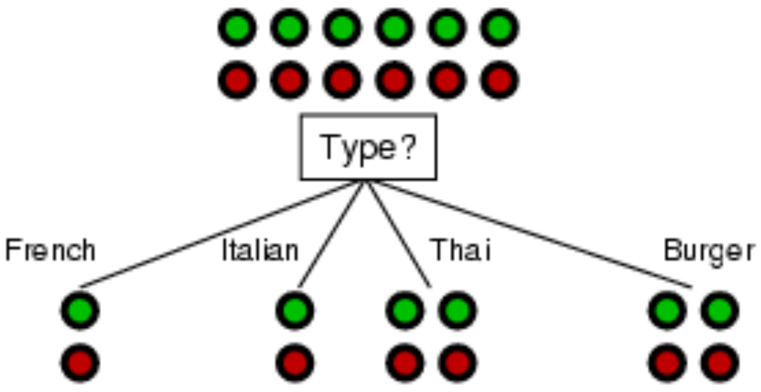
Key problem: choosing which attribute to split given set of examples

- Some possibilities are:
  - **Random:** select any attribute at random
  - **Least-Values:** choose the attribute with the smallest number of possible values
  - **Highest accuracy:** choose the attribute with the largest accuracy
- **ID3 algorithm**
  - Uses **Max-Gain** to select the best attribute: one that has the largest expected **information gain**

Slide credit: Eric Eaton

Choosing an Attribute





Which split is more informative: *Patrons?* or *Type?*

Use **Information Gain** to choose which attribute to split

Idea: a good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative"

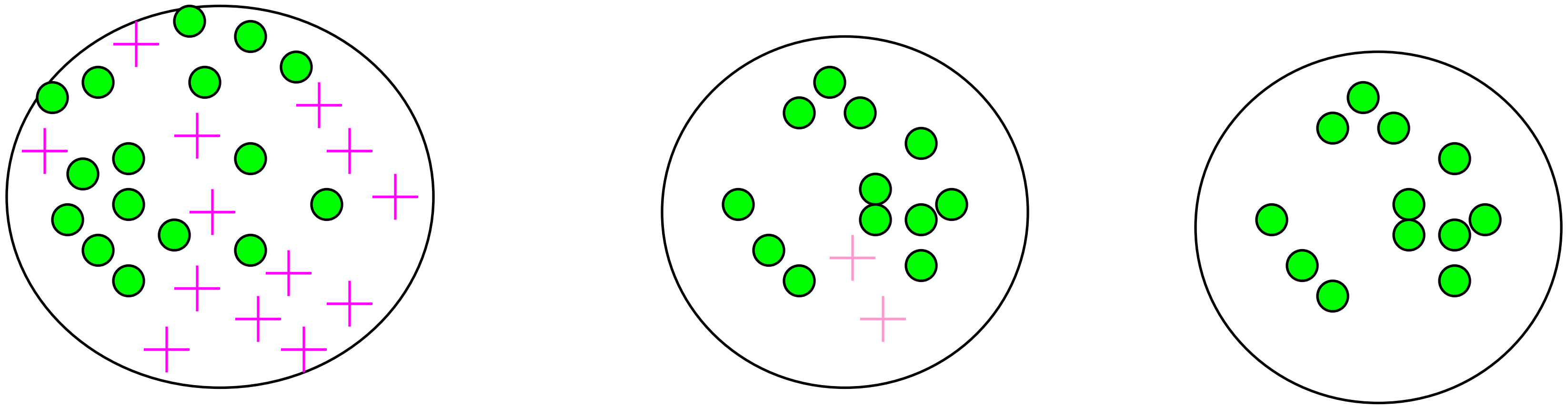
Slide credit: Eric Eaton
Based on slide by M. desJardins & T. Finin

How to measure information gain?

Entropy (informal)

- Measures the level of **impurity** in a group of examples

Very impure group | **Less impure** | **Minimum impurity**



Based on slide by Pedro Domingos

How to measure information gain?

Idea: Gaining information reduces uncertainty

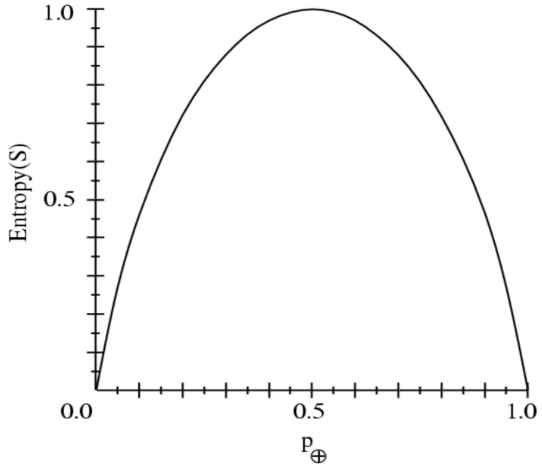
- Use **entropy** to measure **uncertainty**
- Entropy of a random variable A that takes K different values $a_1, a_2, \cdots, a_K$

$$H(A) = -\sum_{k=1}^{K} P(A = a_k) \log P(A = a_k)$$

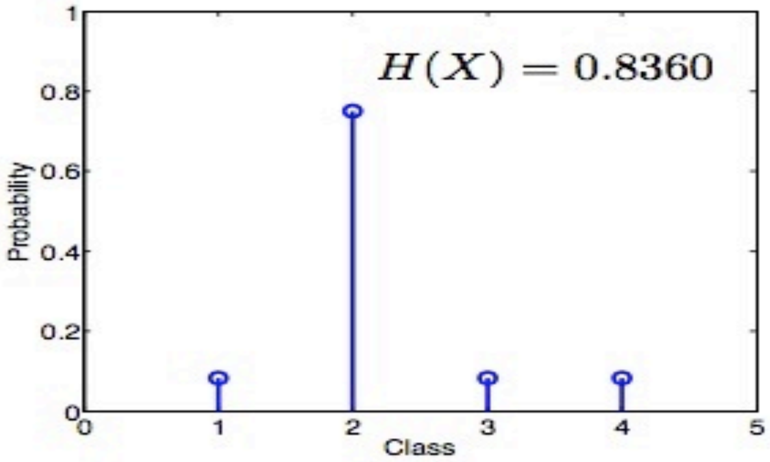
The base can be 2 though this is not essential.
If the base is 2, the unit of entropy is called "bit"
If the base is e, the unit of entropy is called "nat"

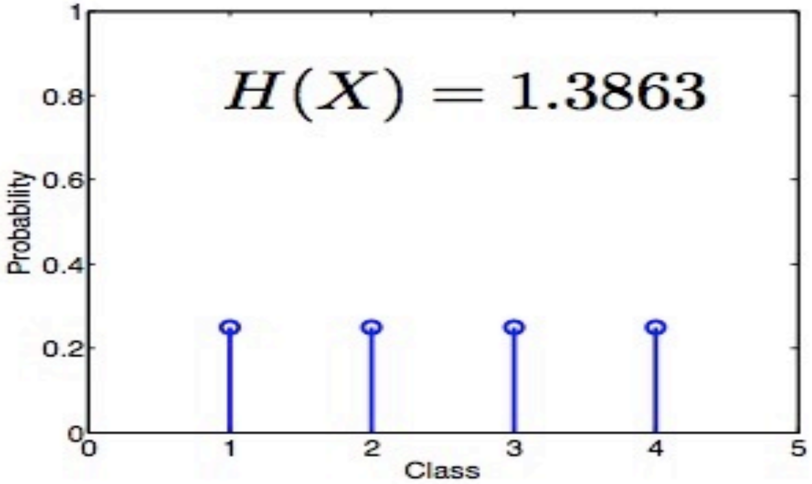
Entropy

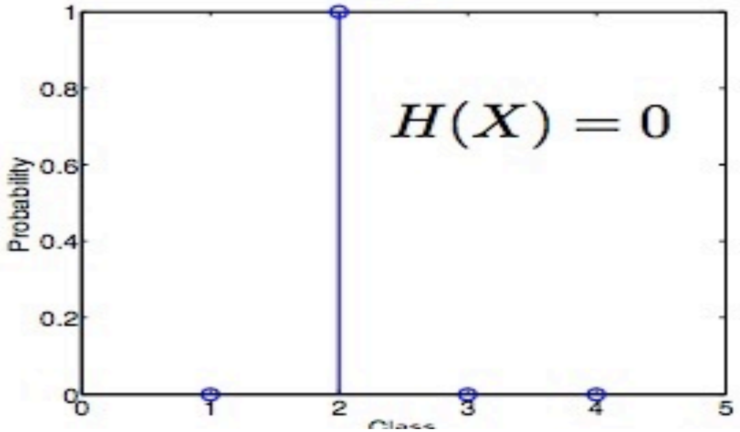
- Measures amount of uncertainty of random variable with specific probability distribution
- Higher the entropy, less confident we are in its outcome
- Example
  - Coin flip ($n = 2$)
    $H(X) = - P(X = 0)\log_2 P(X = 0) - P(X = 1)\log_2 P(X = 1)$
  - if $P(X = 1) = 1$
    $H(X) = - 1 \cdot \log_2(1) - 0 \cdot \log_2(0) = 0$ (no uncertainty)
  - if $P(X = 1) = 0.5$
    $H(X) = - 0.5 \cdot \log_2(0.5) - 0.5 \cdot \log_2(0.5) = -\log_2(0.5) = 1$



Examples of multi-class entropy







Conditional Entropy

Definition: Given two random variables A and Y, **conditional entropy** of Y given X is:

$$H(Y|A) = \sum_{k=1}^{K} P(A = a_k)\, H(Y \mid A = a_k)$$

Intuition:

- Conditional entropy measures how much uncertainty (entropy) in Y if we know A
- Can also be thought of as a weighted average of $H(Y|A = a_k)$ with weights given by $P(A = a_k)$

Information Gain

- How much do we gain from knowing one of the attributes?
- In other words, what is the reduction in entropy from this knowledge?

Definition

Information Gain (a.k.a. **mutual information**) of A and Y

$$I(A, Y) = H(Y) - H(Y|A)$$

Properties

- Symmetricity: $I(A, Y) = I(Y, A) = H(A) - H(A|Y)$
- Non-negativity: $I(A, Y) \geq 0$

Information Gain in Decision Trees

In decision trees:

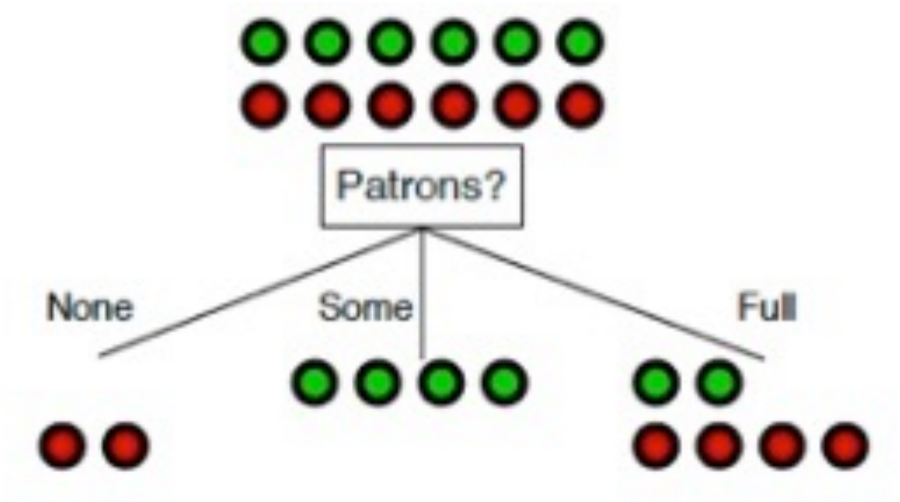
A is an input attribute (e.g., Patron, Type, etc)

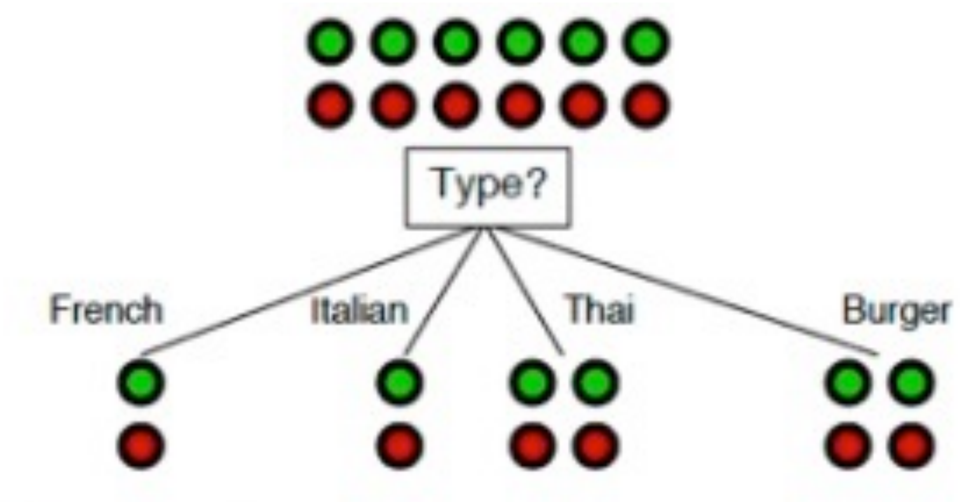
Y is a target variable (e.g., should you wait for table?)

Information Gain $I(A, Y)$

- is the mutual information between input attribute A and target variable Y
- is the expected reduction in entropy of target variable Y, due to sorting on attribute A
- tells us how important a given attribute of the feature vector is

Choosing an attribute





Should we pick *Patrons* or *Type* as root node for decision tree?

- Compute $I(\text{Patrons}, Y)$ and $I(\text{Type}, Y)$. Pick whichever is higher
- Note $H(Y)$ is constant. So we only need to compare $H(Y|A)$

Conditional Entropy for “Patrons”

For “None” branch

$$-\left(\frac{0}{0+2}\log\frac{0}{0+2}+\frac{2}{0+2}\log\frac{2}{0+2}\right)=0$$

For “Some” branch

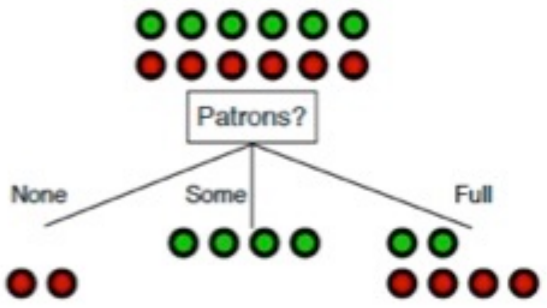
$$-\left(\frac{4}{4+0}\log\frac{4}{4+0}+\frac{0}{4+0}\log\frac{4}{4+0}\right)=0$$

For “Full” branch

$$-\left(\frac{2}{2+4}\log\frac{2}{2+4}+\frac{4}{2+4}\log\frac{4}{2+4}\right)\approx 0.9$$

For choosing “Patrons” (weighted average)

$$\frac{2}{12}*0+\frac{4}{12}*0+\frac{6}{12}*0.9=0.45$$



Conditional entropy for “Type”

For “French” branch

$$-\left(\frac{1}{1+1}\log\frac{1}{1+1}+\frac{1}{1+1}\log\frac{1}{1+1}\right)=1$$

For “Italian” branch

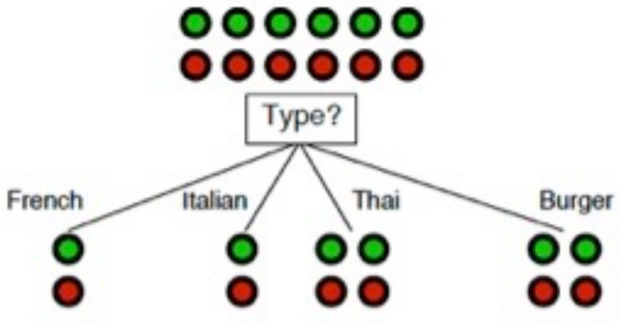
$$-\left(\frac{1}{1+1}\log\frac{1}{1+1}+\frac{1}{1+1}\log\frac{1}{1+1}\right)=1$$

For “Thai” and “Burger” branches

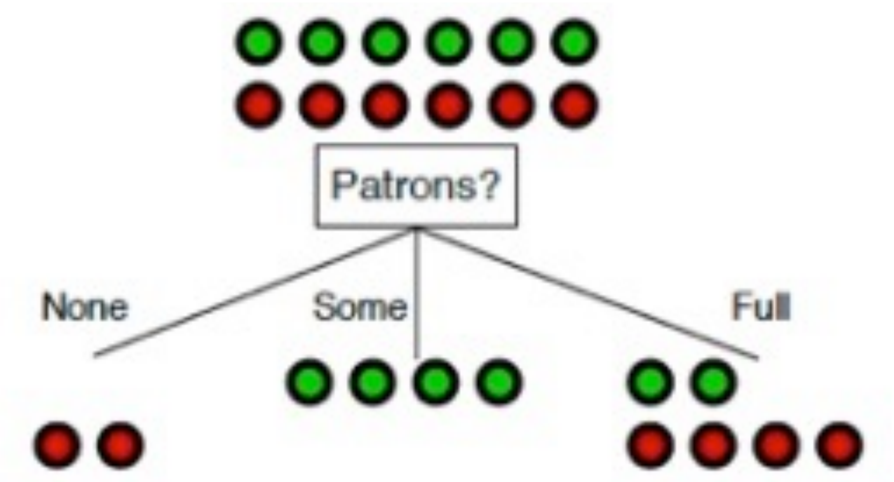
$$-\left(\frac{2}{2+2}\log\frac{2}{2+2}+\frac{2}{2+2}\log\frac{2}{2+2}\right)=1$$

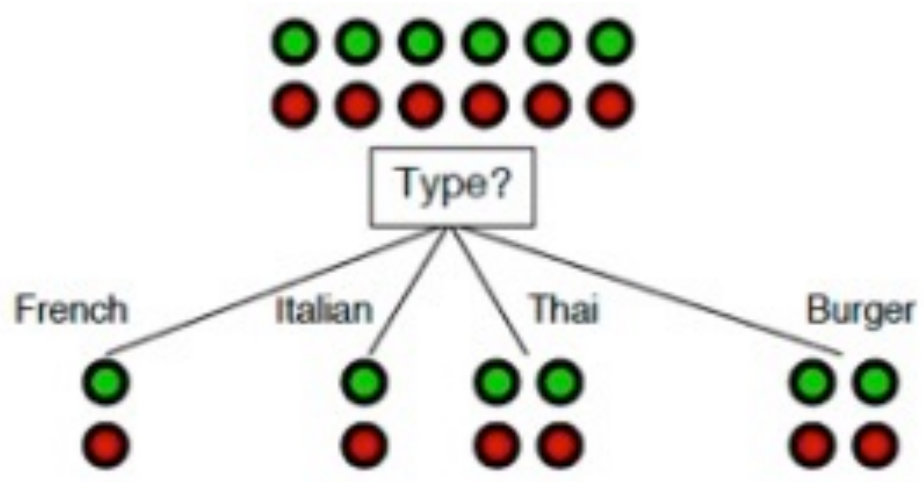
For choosing “Type”

$$\frac{2}{12}*1+\frac{2}{12}*1+\frac{4}{12}*1+\frac{4}{12}*1=1$$



Choosing an attribute

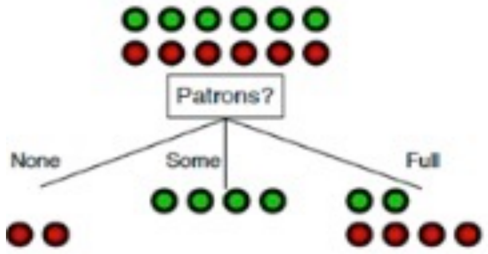




Should we pick *Patrons* or *Type* as root node for decision tree?

- By choosing *Patrons*, we end up with a partition (3 branches) with uncertainty (i.e., conditional entropy) of 0.45 bits.
- By choosing *Type*, we end up with uncertainty of 1 bit.
- Thus, we choose *Patrons* over *Type* to lower our uncertainty.

Next split?



- We will only look at the instances with Patrons=Full

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

Full ID3 Algorithm

see any problems?

function BuildTree(D, S, Y) // D: training set, S: input attributes, Y: class attribute

A ← "best" decision attribute among S using D

tree ← new tree with root node assigned attribute A

for each value a_k of A

D_k ← examples from D with value a_k for attribute A

subtree ← BuildTree($D_k, S - A, Y$)

add *subtree* as child of *tree* with branch labeled $A = a_k$

return *tree*

what if A is empty or all examples have same values for attributes?

what if D is empty?

recursion missing base case?

Full ID3 Algorithm

function BuildTree(D, A, Y) // D: training set, S: input attributes, Y: class attribute

    if D is empty

        return leaf node with failure

    else if all samples in D have same label label for Y

        return leaf node with label

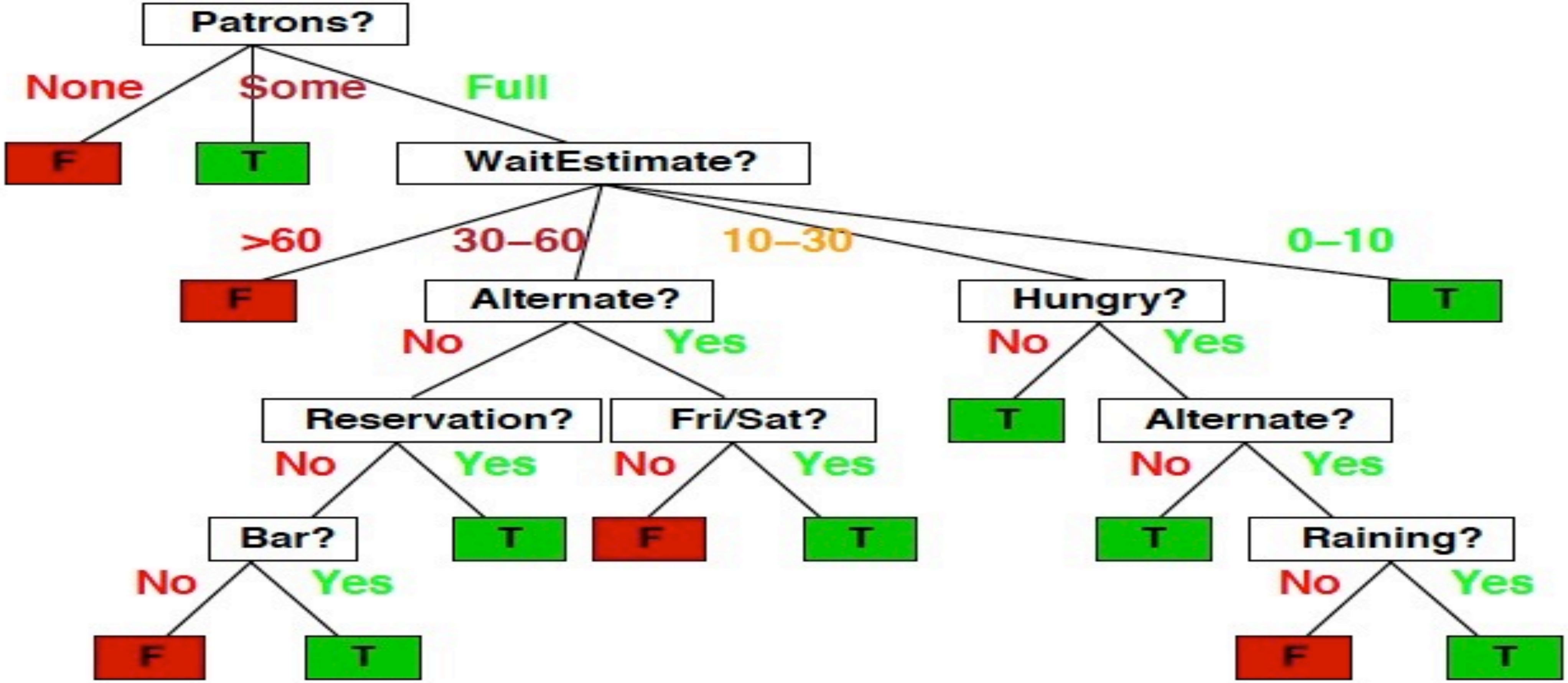
    else if A is empty or all D have same feature values

        return leaf node with majority vote of values of Y in D

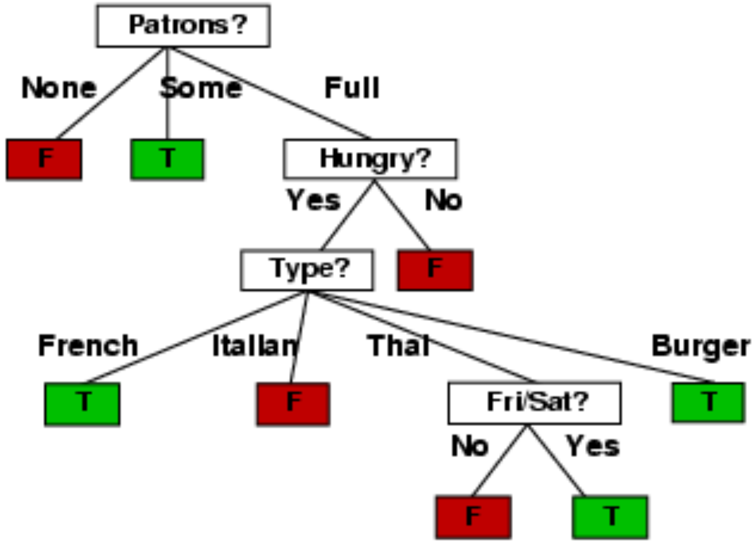
    else

        A ← "best" decision attribute among S using D
        tree ← new tree with root node assigned attribute A
        for each value a_k of A
                D_k← examples from D with value a_k for attribute A
                subtree ← BuildTree($D_k, S - A, Y$)
                add *subtree* as child of *tree* with branch labeled $A = a_k$
        return *tree*

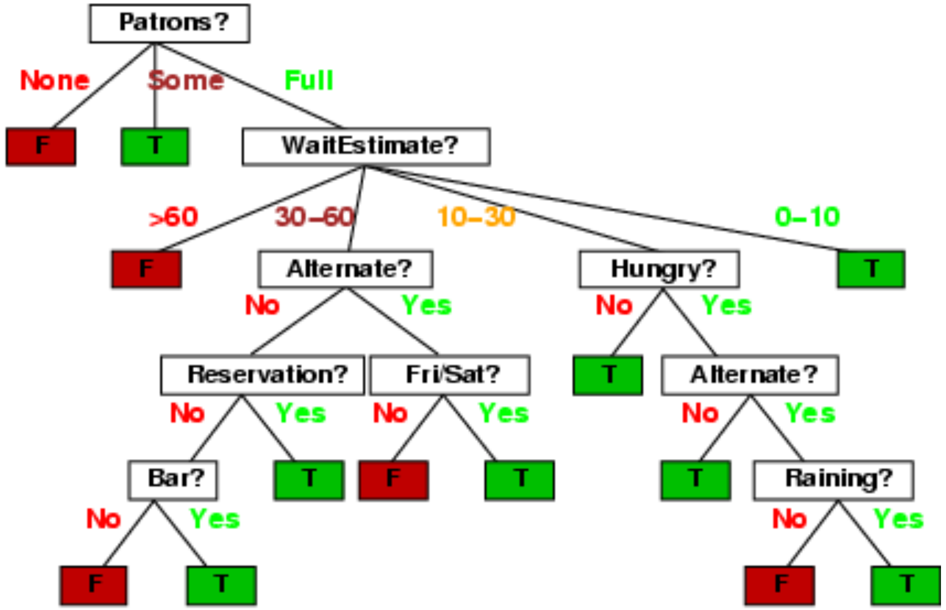
Greedy Tree Building With Introspection



ID3-induced Decision Tree



Compare to introspection



Is this the smallest tree?

Based on slide by Eric Eaton
(originally by M. desJardins & T. Finin)

Pruning Decision Trees

- Tree depth is an important hyperparameter for generalization of decision trees
- Need to carefully pick an appropriate depth
  - Too deep: overfit
  - Too shallow: underfit
- Decision Tree Pruning
  - **Pre-pruning:** Prune while building tree
  - **Post-pruning:** Prune after building tree

Pre-pruning Decision Trees

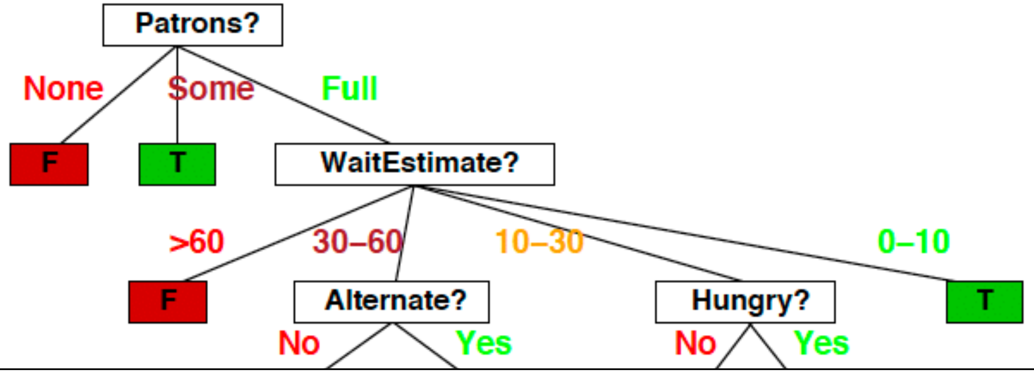
- Common regularization techniques control size of tree
  - Minimum leaf size
    - Do not split A if its cardinality falls below a **fixed threshold**
  - Maximum depth
    - Do not split A if more than a **fixed threshold** of splits were already taken to reach A
  - Maximum number of nodes
    - Stop if a tree has more than a **fixed threshold** of leaf nodes
- How to set threshold(s) to select "best" tree?
  - **Early stopping:** measure performance over separate validation set – stop when validation accuracy starts decreasing

Decision Tree Pruning

- Tree depth is a hyperparameter
- Need to carefully pick an appropriate depth
  - Too deep: overfit
  - Too shallow: underfit
- Decision Tree Pruning
  - Prune while building tree (**early stopping**)
  - Prune after building tree (**post-pruning**)

Pruning Decision Trees

Prune to a smaller tree

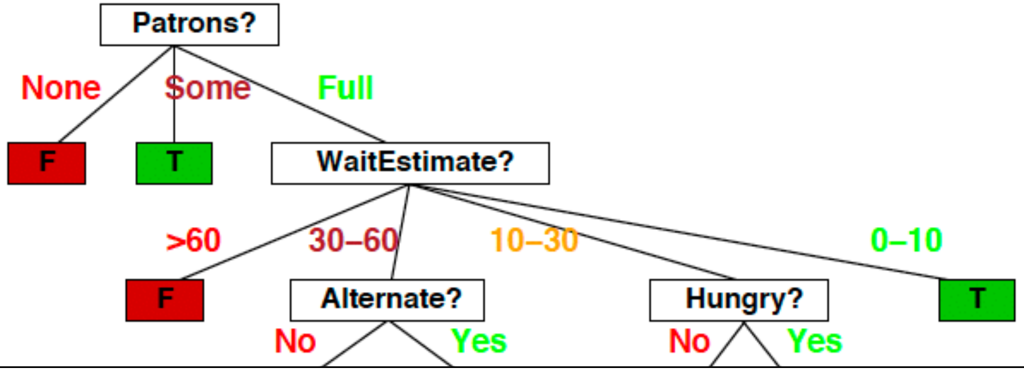


If we stop here, not all training sample would be classified correctly.

More importantly, how do we classify a new instance?

Pruning Decision Trees

Prune to a smaller tree



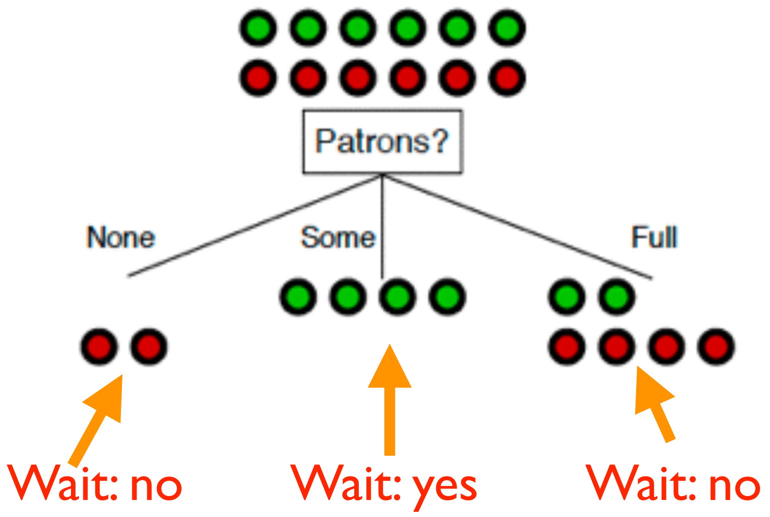
If we stop here, not all training sample would be classified correctly.

More importantly, how do we classify a new instance?

We label the leaves of this smaller tree with the majority of training samples' labels

Example

We stop after the root (first node)



Reduced-Error Pruning

- An example of a **post-pruning strategy**
- Classify examples in validation set – some might be errors
- For each node:
  - Sum the errors over entire subtree
  - Calculate error on same example if converted to a leaf with majority class label
- Prune node with highest reduction in error
- Repeat until error no longer reduced

Summary: Decision Trees

- Representation

  Trees
- Loss function

  Not explicitly defined, 0-1 loss at the leaves
- Search/optimization algorithm

  ID3 algorithm
- Generalization and Regularization

  Limiting tree size via pre-pruning and post-pruning strategies