

Introduction to Machine Learning

Decision Trees

Ashley Gao

William & Mary

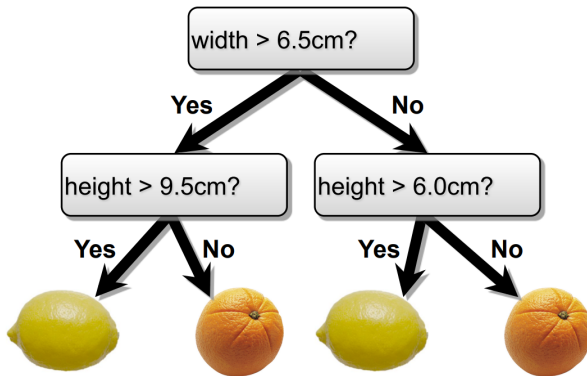
October 11, 2023

Overview

- Decision Tree
 - Simple but powerful learning algorithm
 - Used widely in Kaggle competitions
 - Lets us motivate concepts from information theory (entropy, mutual information, etc.)
- Bias-variance decomposition
 - Lets us motivate methods for combining different classifiers.

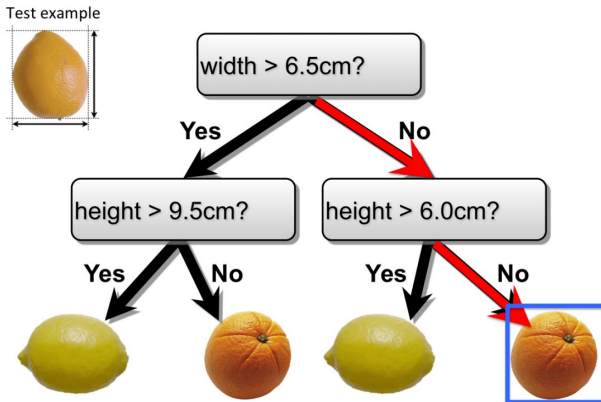
Decision Tree

- Make predictions by splitting features according to a tree structure.



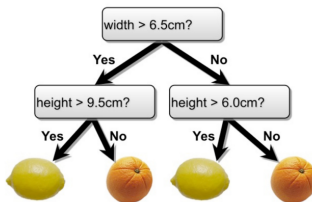
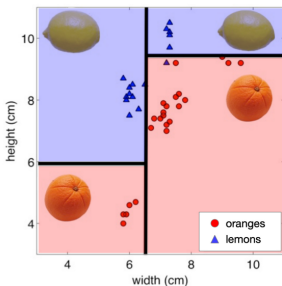
Decision Tree

- Make predictions by splitting features according to a tree structure.



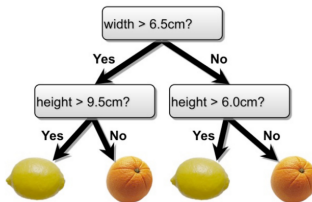
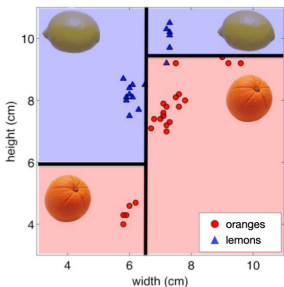
Decision Trees — Continuous Features

- Split continuous features by checking whether that feature is greater than or less than some threshold.
- Decision boundary is made up of axis-aligned planes.



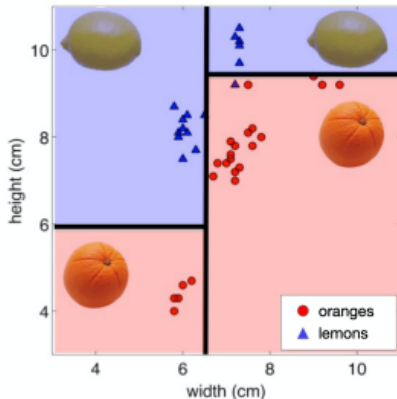
Decision Trees — Continuous Features

- Internal nodes test a feature
- Branching is determined by the feature value
- Leaf nodes are outputs (predictions)



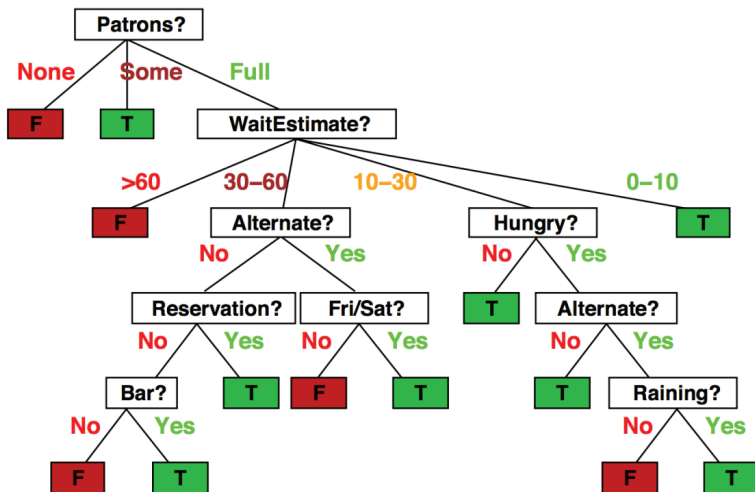
Decision Trees — Continuous Features

- Each path from the root to a leaf defines a region R_m of input space
- Let $\{(x^{(m_1)}, t^{(m_1)}), \dots, (x^{(m_k)}, t^{(m_k)})\}$ be the training examples that fall into R_m
- Classification tree (we will focus on this):
 - Discrete output
 - Leaf value y^m typically set to the most common value in $\{t^{(m_1)}, \dots, t^{(m_k)}\}$



Decision Trees — Discrete Features

- Will I eat at this restaurant?



Decision Trees — Discrete Features

- Split discrete features into a partition of possible values.

Example	Input Attributes										Goal WillWait
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	
x_1	Yes	No	No	Yes	Some	\$\$\$	No	Yes	French	0-10	$y_1 = \text{Yes}$
x_2	Yes	No	No	Yes	Full	\$	No	No	Thai	30-60	$y_2 = \text{No}$
x_3	No	Yes	No	No	Some	\$	No	No	Burger	0-10	$y_3 = \text{Yes}$
x_4	Yes	No	Yes	Yes	Full	\$	Yes	No	Thai	10-30	$y_4 = \text{Yes}$
x_5	Yes	No	Yes	No	Full	\$\$\$	No	Yes	French	>60	$y_5 = \text{No}$
x_6	No	Yes	No	Yes	Some	\$\$	Yes	Yes	Italian	0-10	$y_6 = \text{Yes}$
x_7	No	Yes	No	No	None	\$	Yes	No	Burger	0-10	$y_7 = \text{No}$
x_8	No	No	No	Yes	Some	\$\$	Yes	Yes	Thai	0-10	$y_8 = \text{Yes}$
x_9	No	Yes	Yes	No	Full	\$	Yes	No	Burger	>60	$y_9 = \text{No}$
x_{10}	Yes	Yes	Yes	Yes	Full	\$\$\$	No	Yes	Italian	10-30	$y_{10} = \text{No}$
x_{11}	No	No	No	No	None	\$	No	No	Thai	0-10	$y_{11} = \text{No}$
x_{12}	Yes	Yes	Yes	Yes	Full	\$	No	No	Burger	30-60	$y_{12} = \text{Yes}$

1.	Alternate: whether there is a suitable alternative restaurant nearby.
2.	Bar: whether the restaurant has a comfortable bar area to wait in.
3.	Fri/Sat: true on Fridays and Saturdays.
4.	Hungry: whether we are hungry.
5.	Patrons: how many people are in the restaurant (values are None, Some, and Full).
6.	Price: the restaurant's price range (\$, \$\$, \$\$\$).
7.	Raining: whether it is raining outside.
8.	Reservation: whether we made a reservation.
9.	Type: the kind of restaurant (French, Italian, Thai or Burger).
10.	WaitEstimate: the wait estimated by the host (0-10 minutes, 10-30, 30-60, >60).

Features:

Learning Decision Trees

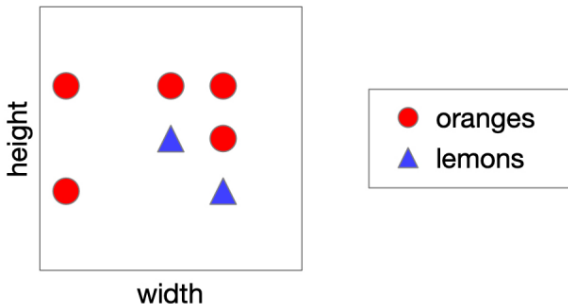
- For any training set we can construct a decision tree that has exactly one leaf for every training point, but it probably won't generalize.
 - Decision trees are universal function approximators.
- But, finding the smallest decision tree that correctly classifies a training set is NP-complete.
 - If you are interested, check: Hyafil & Rivest'76.
- So, how do we construct a useful decision tree?

Learning Decision Trees

- Resort to a greedy heuristic:
 - Start with the whole training set and an empty decision tree
 - Pick a feature and candidate split that would most reduce the loss
 - Split on that feature and recurse on subpartitions
- Which loss should we use?
 - Let's see if the misclassification rate is a good loss.

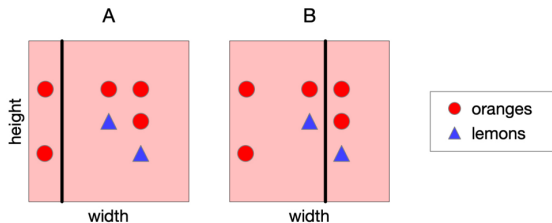
Choosing a Good Split

- Consider the following data. Let's split on width



Choosing a Good Split

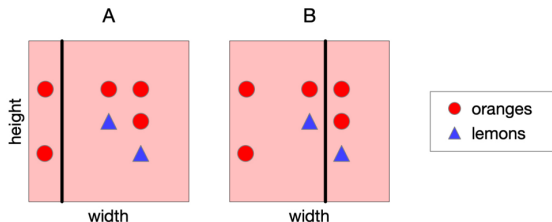
- Recall: classify by majority.



- A and B have the same misclassification rate, so which is the best split?
Vote!

Choosing a Good Split

- A feels like a better split, because the left-hand region is very certain about whether the fruit is an orange.



- Can we quantify this?

Choosing a Good Split

- How can we quantify uncertainty in prediction for a given leaf node?
 - If all examples in the leaf have the same class: good, low uncertainty
 - If each class has the same amount of examples in leaf: bad, high uncertainty
- Idea: Use counts at leaves to define probability distributions; use a probabilistic notion of uncertainty to decide splits.
- A brief detour through information theory...

Choosing a Good Split

- The entropy of a discrete random variable is a number that quantifies the uncertainty inherent in its possible outcomes.
- The mathematical definition of entropy that we give in a few slides may seem arbitrary, but it can be motivated.
 - If you're interested, check: Information Theory by Robert Ash.
- To explain entropy, consider flipping two different coins...

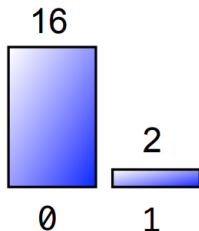
We Flip Two Different Coins

Sequence 1:

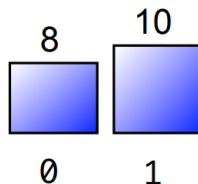
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 ... ?

Sequence 2:

0 1 0 1 0 1 1 1 0 1 0 0 1 1 0 1 0 1 ... ?



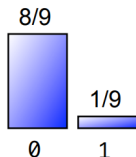
versus



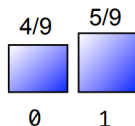
Quantifying Uncertainty

- The entropy of a loaded coin with probability p of heads is given by:

$$-p\log_2(p) - (1 - p)\log_2(1 - p) \quad (1)$$



$$-\frac{8}{9}\log_2\frac{8}{9} - \frac{1}{9}\log_2\frac{1}{9} \approx \frac{1}{2}$$

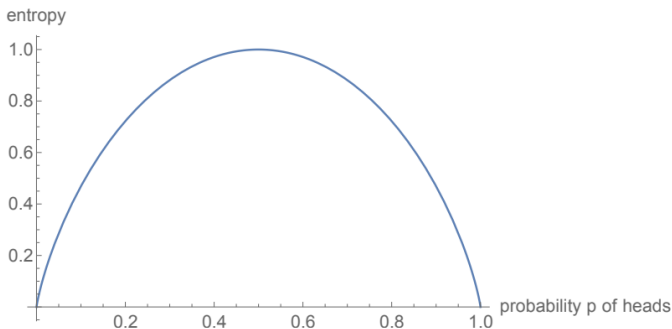


$$-\frac{4}{9}\log_2\frac{4}{9} - \frac{5}{9}\log_2\frac{5}{9} \approx 0.99$$

- Notice: the coin whose outcomes are more certain has a lower entropy.
- In the extreme case $p = 0$ or $p = 1$, we were certain of the outcome before observing. So, we gained no certainty by observing it, i.e., entropy is 0.

Quantifying Uncertainty

- Can also think of entropy as the expected information content of a random draw from a probability distribution.



- Units of entropy are bits; a fair coin flip has 1 bit of entropy

Choosing a Good Split

- More generally, the entropy of a discrete random variable Y is given by:

$$H(Y) = - \sum_{y \in Y} p(y) \log_2 p(y) \quad (2)$$

- “High Entropy”:
 - Variable has a uniform like distribution over many outcomes
 - Flat histogram
 - Values sampled from it are less predictable
- “Low Entropy”:
 - Distribution is concentrated on only a few outcomes
 - Histogram is concentrated in a few areas
 - Values sampled from it are more predictable

Entropy

- Suppose we observe partial information X about a random variable Y
 - For example, $X = \text{sign}(Y)$
- We want to work towards a definition of the expected amount of information that will be conveyed about Y by observing X .
 - Or equivalently, the expected reduction in our uncertainty about Y after observing X .

Entropy of a Joint Distribution

- Example: $X = \{\text{Raining, Not raining}\}$, $Y = \{\text{Cloudy, Not cloudy}\}$

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

$$H(X, Y) = - \sum_{x \in X} \sum_{y \in Y} p(x, y) \log_2 p(x, y) \quad (3)$$

$$H(X, Y) = -\frac{24}{100} \log_2 \frac{24}{100} - \frac{1}{100} \log_2 \frac{1}{100} - \frac{25}{100} \log_2 \frac{25}{100} - \frac{50}{100} \log_2 \frac{50}{100} \quad (4)$$

$$H(X, Y) \approx 1.56 \text{bits} \quad (5)$$

Specific Conditional Entropy

- Example: $X = \{\text{Raining, Not raining}\}$, $Y = \{\text{Cloudy, Not cloudy}\}$

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

- What is the entropy of cloudiness Y , given that it is raining ($X = \text{raining}$)?

$$H(Y|X = x) = - \sum_{y \in Y} p(y|x) \log_2 p(y|x) \quad (6)$$

$$H(Y|X = x) = -\frac{24}{25} \log_2 \frac{24}{25} - \frac{1}{25} \log_2 \frac{1}{25} \approx 0.24 \text{bits} \quad (7)$$

- We used $p(y|x) = \frac{p(x,y)}{p(x)}$ and $p(x) = \sum_y p(x, y)$ (sum in a row)

Conditional Entropy

- Example: $X = \{\text{Raining, Not raining}\}$, $Y = \{\text{Cloudy, Not cloudy}\}$

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

- What is the entropy of cloudiness Y , given the variable X ?
- The expected conditional entropy:

$$H(Y|X) = \sum_{x \in X} p(x) H(Y|X = x) \quad (8)$$

$$H(Y|X) = - \sum_{x \in X} \sum_{y \in Y} p(x, y) \log_2 p(y|x) \quad (9)$$

Conditional Entropy

- Example: $X = \{\text{Raining}, \text{Not raining}\}$, $Y = \{\text{Cloudy}, \text{Not cloudy}\}$

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

- What is the entropy of cloudiness Y , given the variable X ?

$$H(Y|X) = \sum_{x \in X} p(x) H(Y|X = x) \quad (10)$$

$$H(Y|X) = \frac{1}{4} H(\text{cloudy}|\text{raining}) + \frac{3}{4} H(\text{cloudy}|\text{not raining}) \approx 0.75 \text{bits} \quad (11)$$

Conditional Entropy

- Some useful properties:
 - H is always non-negative
 - Chain rule: $H(X, Y) = H(X|Y) + H(Y) = H(Y|X) + H(X)$
 - if X and Y are independent, then X does not affect our uncertainty about Y :
 $H(Y|X) = H(Y)$
 - By knowing Y makes our knowledge of Y certain: $H(Y|Y) = 0$
 - By knowing X , we can decrease the uncertainty about Y : $H(Y|X) \leq H(Y)$

Information Gain

- Example: $X = \{\text{Raining, Not raining}\}$, $Y = \{\text{Cloudy, Not cloudy}\}$

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

- How much more certain am I about whether it's cloudy if I'm told whether it is raining? My uncertainty in Y minus my expected uncertainty that would remain in Y after seeing X .
- This is called the information gain $IG(Y|X)$ in Y due to X , or the mutual information of Y and X

$$IG(Y|X) = H(Y) - H(Y|X) \quad (12)$$

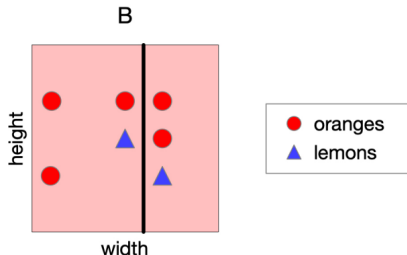
- if X is completely uninformative about Y : $IG(Y|X) = 0$
- if X is completely informative about Y : $IG(Y|X) = H(Y)$

Revisiting Our Original Example

- Information gain measures the informativeness of a variable, which is exactly what we desire in a decision tree split!
- The information gain of a split: how much information (over the training set) about the class label Y is gained by knowing which side of a split you're on.

Revisiting Our Original Example

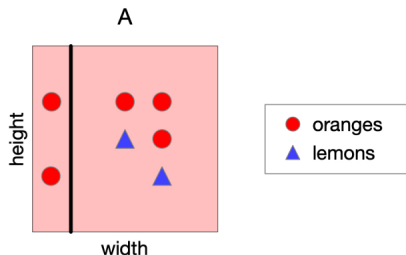
- What is the information gain of split B? Not terribly informative...



- Root entropy of class outcome: $H(Y) = -\frac{2}{7}\log_2\frac{2}{7} - \frac{5}{7}\log_2\frac{5}{7} \approx 0.86$
- Leaf conditional entropy of class outcome: $H(Y|\text{left}) \approx 0.81$,
 $H(Y|\text{right}) \approx 0.92$
- $IG(\text{split}) \approx 0.86 - (\frac{4}{7} \cdot 0.81 + \frac{3}{7} \cdot 0.92) \approx 0.006$

Revisiting Our Original Example

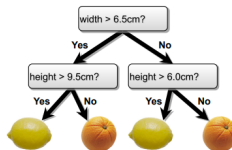
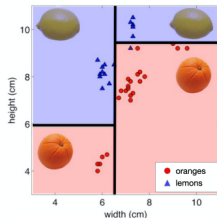
- What is the information gain of split A? Very informative!



- Root entropy of class outcome: $H(Y) = -\frac{2}{7}\log_2\frac{2}{7} - \frac{5}{7}\log_2\frac{5}{7} \approx 0.86$
- Leaf conditional entropy of class outcome: $H(Y|\text{left}) = 0$,
 $h(Y|\text{right}) \approx 0.97$
- $IG(\text{split}) \approx 0.86 - (\frac{2}{7} \cdot 0 + \frac{5}{7} \cdot 0.97) \approx 0.17!$

Constructing Decision Trees

- At each level, one must choose:
 - Which particular feature to split
 - Possibly where to split it
- Choose them based on how much information we would gain from the decision! (choose the feature that gives the highest gain)



Decision Tree Construction Algorithm

- Simple, greedy, recursive approach, builds up tree node-by-node
 - pick a feature to split at a non-terminal node
 - split examples into groups based on a feature value
 - for each group:
 - if all examples in the same class – return class
 - else loop to step 1
- Terminates when all leaves contain only examples in the same class or are empty

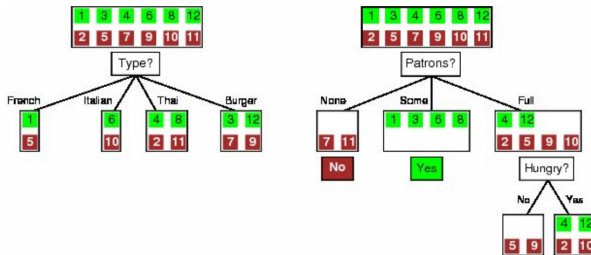
Back to Our Example

Example	Input Attributes										Goal
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>WillWait</i>
x ₁	Yes	No	No	Yes	Some	\$\$\$	No	Yes	French	0-10	y ₁ = Yes
x ₂	Yes	No	No	Yes	Full	\$	No	No	Thai	30-60	y ₂ = No
x ₃	No	Yes	No	No	Some	\$	No	No	Burger	0-10	y ₃ = Yes
x ₄	Yes	No	Yes	Yes	Full	\$	Yes	No	Thai	10-30	y ₄ = Yes
x ₅	Yes	No	Yes	No	Full	\$\$\$	No	Yes	French	>60	y ₅ = No
x ₆	No	Yes	No	Yes	Some	\$\$	Yes	Yes	Italian	0-10	y ₆ = Yes
x ₇	No	Yes	No	No	None	\$	Yes	No	Burger	0-10	y ₇ = No
x ₈	No	No	No	Yes	Some	\$\$	Yes	Yes	Thai	0-10	y ₈ = Yes
x ₉	No	Yes	Yes	No	Full	\$	Yes	No	Burger	>60	y ₉ = No
x ₁₀	Yes	Yes	Yes	Yes	Full	\$\$\$	No	Yes	Italian	10-30	y ₁₀ = No
x ₁₁	No	No	No	No	None	\$	No	No	Thai	0-10	y ₁₁ = No
x ₁₂	Yes	Yes	Yes	Yes	Full	\$	No	No	Burger	30-60	y ₁₂ = Yes

1. Alternate: whether there is a suitable alternative restaurant nearby.
2. Bar: whether the restaurant has a comfortable bar area to wait in.
3. Fri/Sat: true on Fridays and Saturdays.
4. Hungry: whether we are hungry.
5. Patrons: how many people are in the restaurant (values are None, Some, and Full).
6. Price: the restaurant's price range (\$, \$\$, \$\$\$).
7. Raining: whether it is raining outside.
8. Reservation: whether we made a reservation.
9. Type: the kind of restaurant (French, Italian, Thai or Burger).
10. WaitEstimate: the wait estimated by the host (0-10 minutes, 10-30, 30-60, >60).

Features:

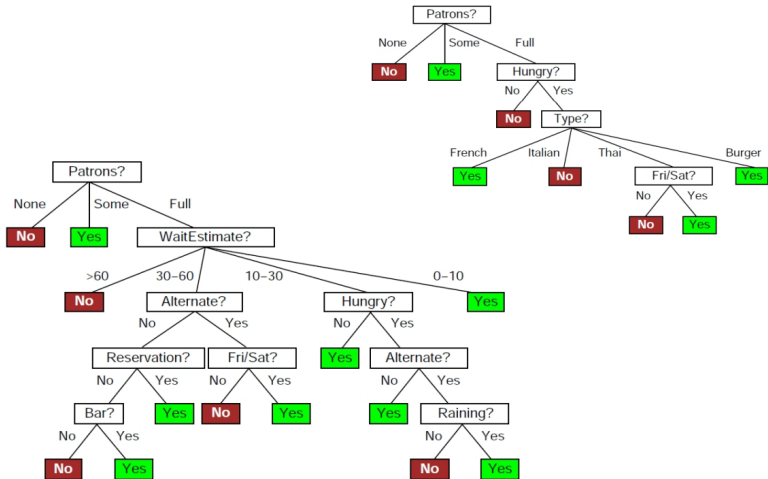
Feature Selection



$$IG(\text{Type}) = 1 - \left[\frac{2}{12}H(Y|\text{Fr.}) + \frac{2}{12}H(Y|\text{It.}) + \frac{4}{12}H(Y|\text{Thai}) + \frac{4}{12}H(Y|\text{Bur.}) \right] = 0 \quad (13)$$

$$IG(\text{Patron}) = 1 - \left[\frac{2}{12}H(Y|\text{None}) + \frac{4}{12}H(Y|\text{Some}) + \frac{6}{12}H(Y|\text{Full}) \right] \approx 0.541 \quad (14)$$

Which Tree is Better? Vote!



What Makes a Good Tree?

- Not too small: need to handle important but possibly subtle distinctions in data
- Not too big:
 - Computational efficiency (avoid redundant, spurious attributes)
 - Avoid over-fitting training examples
 - Human interpretability
- Occam's Razor": find the simplest hypothesis that fits the observations
 - Useful principle, but hard to formalize (how to define simplicity?)
 - See Domingos, 1999, "The role of Occam's razor in knowledge discovery"
- We desire small trees with informative nodes near the root

What Makes a Good Tree?

- Problems:
 - You have exponentially less data at lower levels
 - Too big of a tree can overfit the data
 - Greedy algorithms don't necessarily yield the global optimum
- Handling continuous attributes
 - Split based on a threshold, chosen to maximize information gain
- Decision trees can also be used for regression on real-valued outputs. Choose splits to minimize squared error, rather than maximize information gain.