CS 188: Artificial Intelligence

Decision Trees

Instructor: Oliver Grillmeyer --- University of California, Berkeley

Announcements

- HW9 is due **Tuesday, August 5**, 11:59 PM PT
- HW10 is due **Thursday, August 7**, 11:59 PM PT
- Project 5 is due **Friday, August 8**, 11:59 PM PT
- Ignore assessment on HWs part B, but please show your work
- Final Exam is **Wednesday, August 13**, 7-10 PM PT

Today

- Neural Nets -- wrap

- Enhanced Training

- Formalizing Learning
  - Consistency
  - Simplicity

- Decision Trees
  - Expressiveness
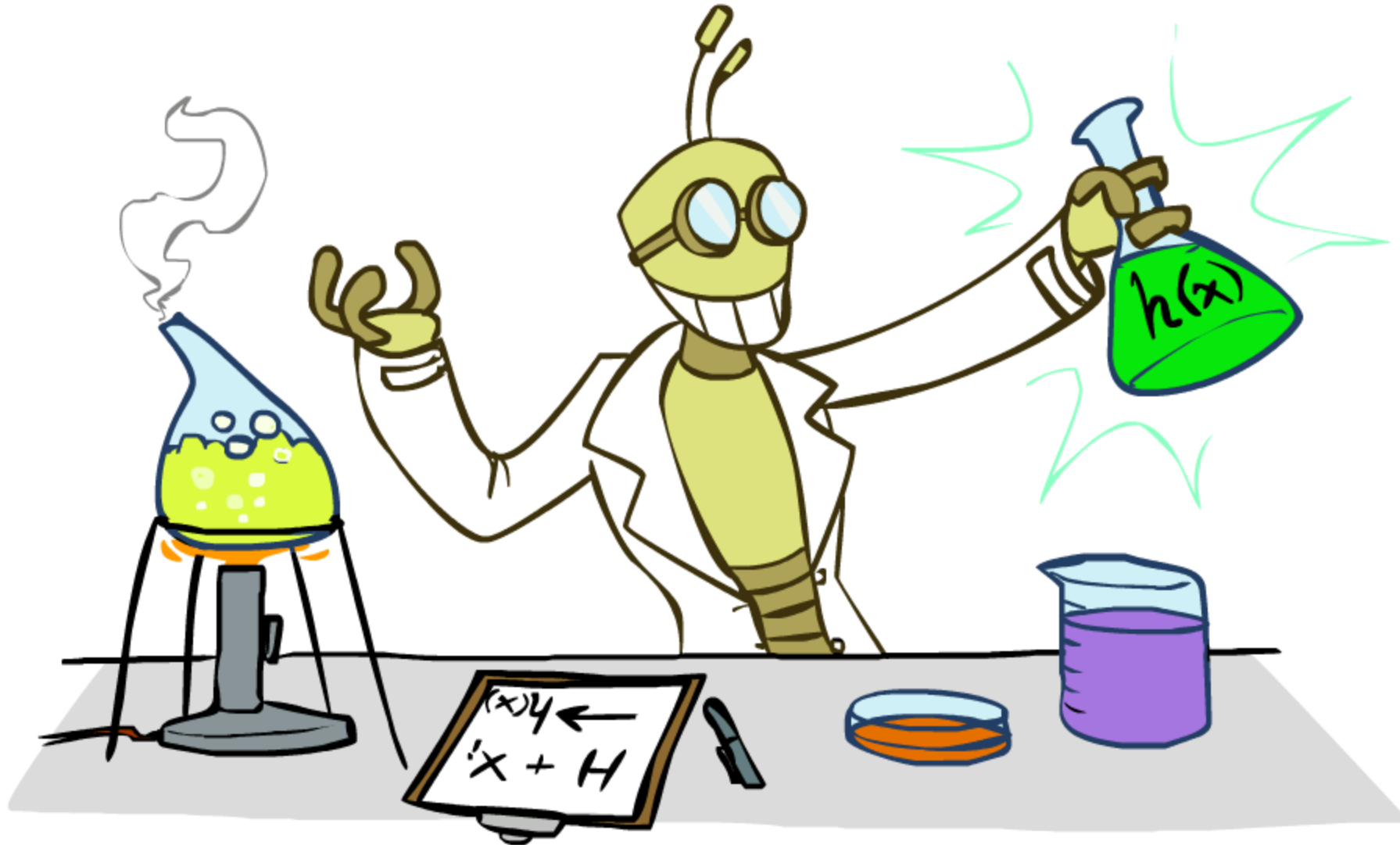 - Information Gain
  - Overfitting

Refresh: Deep Neural Network
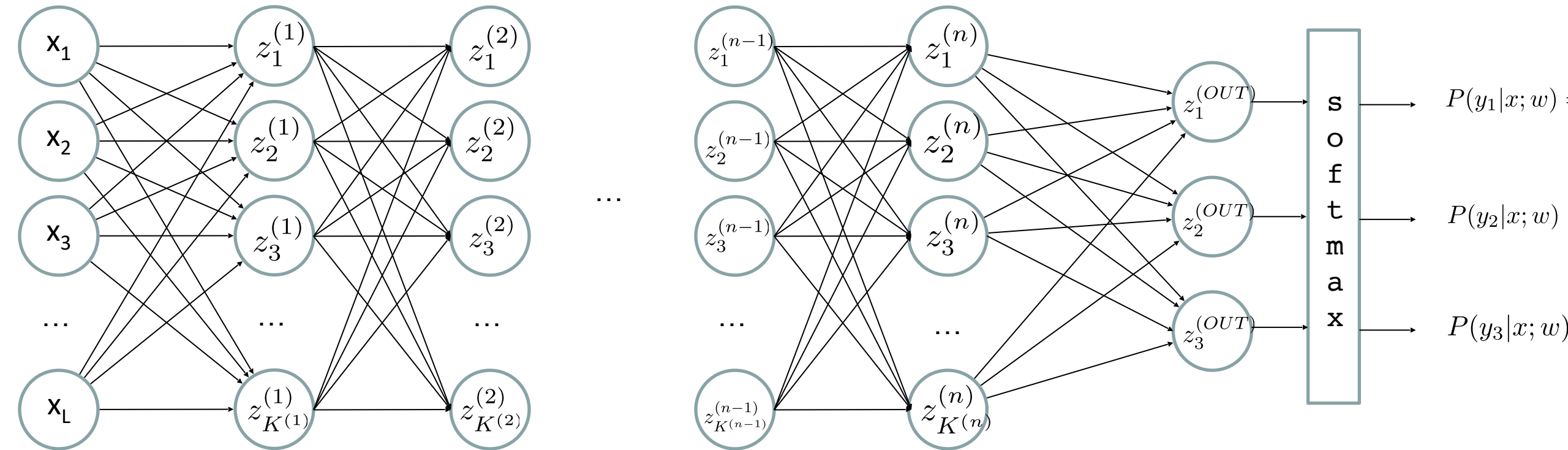
$$z_i^{(k)} = g(\sum_j W_{i,j}^{(k-1,k)} z_j^{(k-1)})$$

g = nonlinear activation function

# Formalizing Learning: Inductive Learning

Inductive Learning (Science)

- Simplest form: learn a function from examples
  - A target function: g
  - Examples: input-output pairs $(x, g(x))$
  - E.g. x is an email and $g(x)$ is spam / ham
  - E.g. x is a house and $g(x)$ is its selling price

- Problem:
  - Given a hypothesis space H
  - Given a training set of examples x_i
  - Find a hypothesis $h(x)$ such that $h \sim g$

- Includes:
  - Classification (outputs = class labels)
  - Regression (outputs = real numbers)

- How do perceptron and naïve Bayes fit in? $(H, h, g, \text{etc.})$

g

h

H
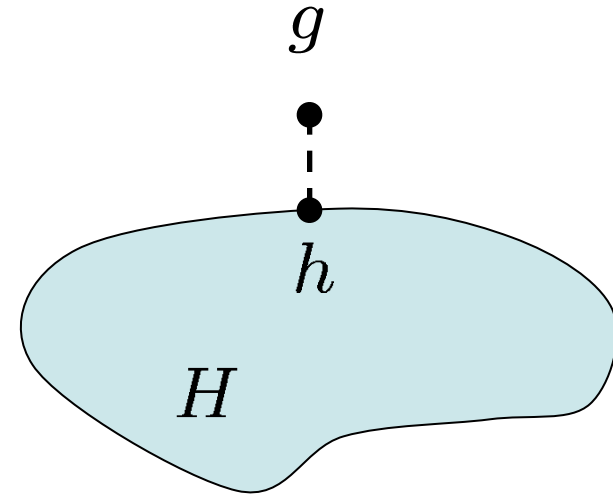
Inductive Learning

- Curve fitting (regression, function approximation):
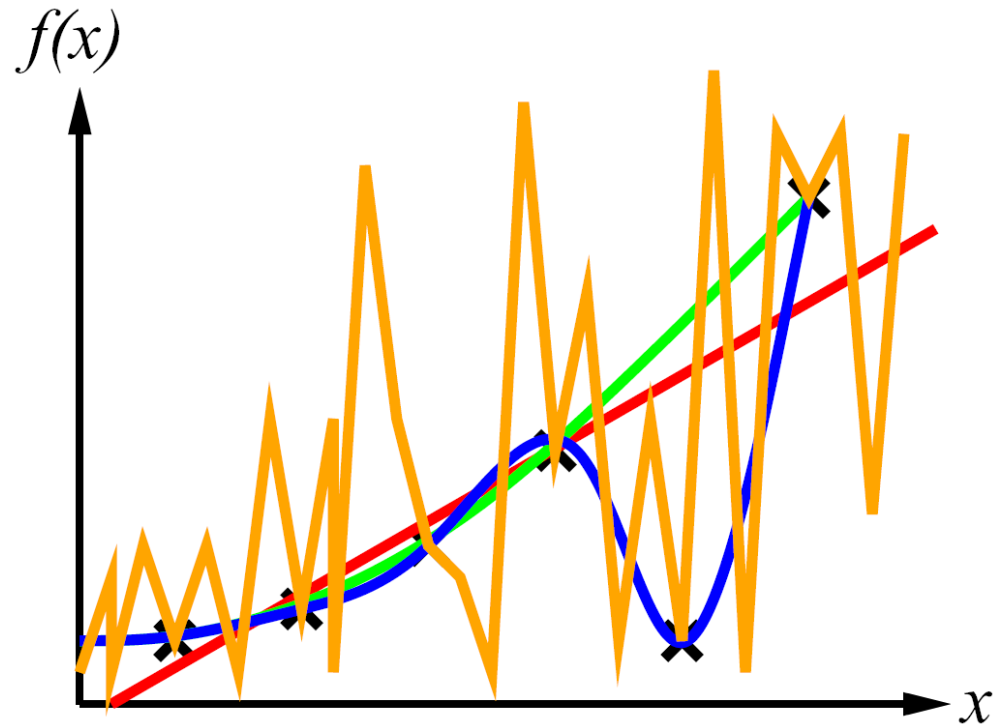
- Consistency vs. simplicity
- Ockham's razor

Consistency vs. Simplicity

- Fundamental tradeoff: bias vs. variance

- Usually algorithms prefer consistency by default (why?)

- Several ways to operationalize "simplicity"
  - Reduce the hypothesis space
    - Assume more: e.g. independence assumptions, as in naïve Bayes
    - Have fewer, better features / attributes: feature selection
    - Other structural limitations (decision lists vs trees)
  - Regularization
    - Smoothing: cautious use of small counts
    - Many other generalization parameters (pruning cutoffs today)
    - Hypothesis space stays big, but harder to get to the outskirts

Decision Trees

Features

- Features, aka attributes
  - Sometimes: TYPE=French
  - Sometimes: $f_{\text{TYPE=French}}(x) = 1$

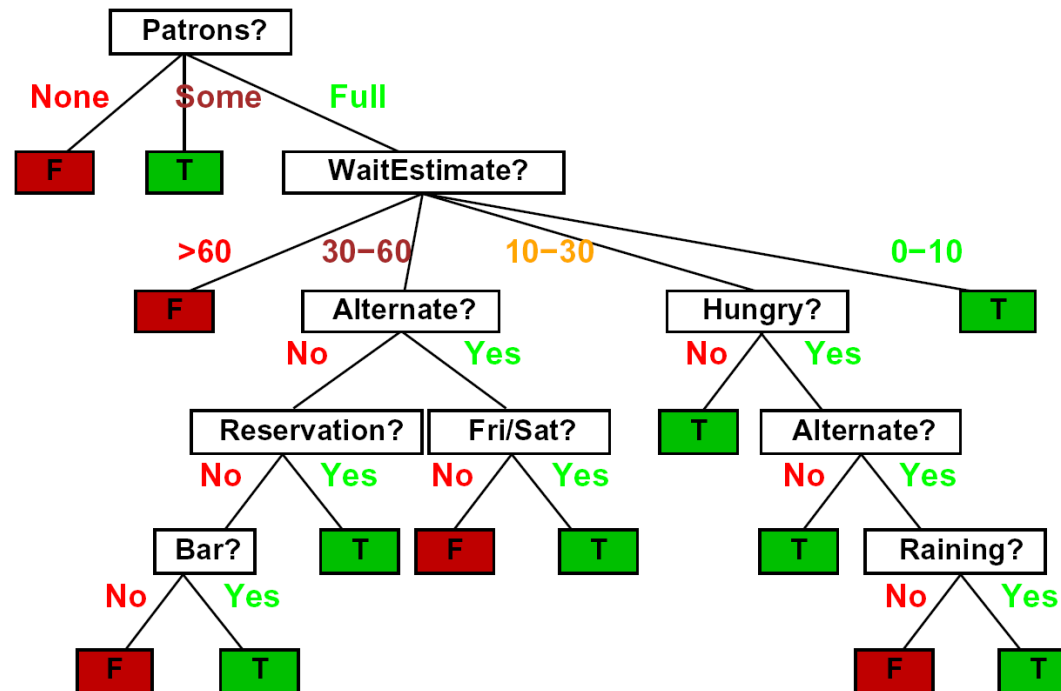| Example	Attributes										Target
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	WillWait
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T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

Decision Trees

- Compact representation of a function:
  - Truth table
  - Conditional probability table
  - Regression values

- True function
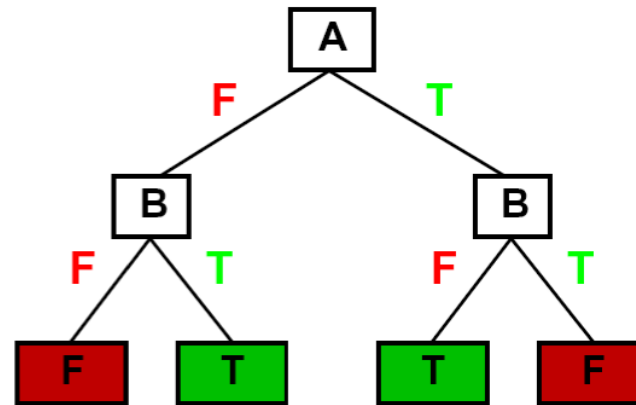 - Realizable: in H

Expressiveness of DTs

- Can express any function of the features

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F

$$P(C|A, B)$$

- However, we hope for compact trees

Comparison: Perceptrons

- What is the expressiveness of a perceptron over these features?

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

- For a perceptron, a feature's contribution is either positive or negative
  - If you want one feature's effect to depend on another, you have to add a new conjunction feature
  - E.g. adding "PATRONS=full ∧ WAIT = 60" allows a perceptron to model the interaction between the two atomic features

- DTs automatically conjoin features / attributes
  - Features can have different effects in different branches of the tree!

- Difference between modeling relative evidence weighting (NB) and complex evidence interaction (DTs)
  - Though if the interactions are too complex, may not find the DT greedily

Hypothesis Spaces

- **How many distinct decision trees with n Boolean attributes?**
  - = number of Boolean functions over n attributes
  - = number of distinct truth tables with 2^n rows
  - = $2^{(2^n)}$
    - E.g., with 6 Boolean attributes, there are
      18,446,744,073,709,551,616 trees

- **How many trees of depth 1 (decision stumps)?**
  - = number of Boolean functions over 1 attribute
  - = number of truth tables with 2 rows, times n
  - = 4n
    - E.g. with 6 Boolean attributes, there are 24 decision stumps

- **More expressive hypothesis space:**
  - Increases chance that target function can be expressed (good)
  - Increases number of hypotheses consistent with training set (bad, why?)
  - Means we can get better predictions (lower bias)
  - But we may get worse predictions (higher variance)
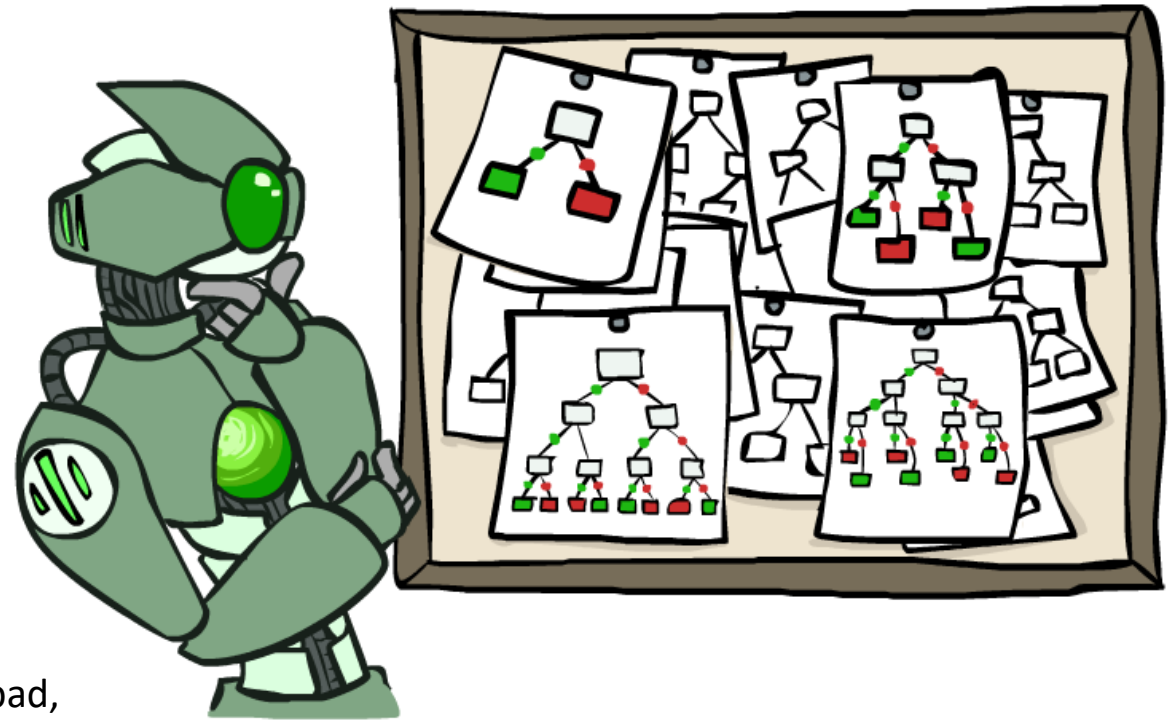
Decision Tree Learning

- Aim: find a small tree consistent with the training examples
- Idea: (recursively) choose "most significant" attribute as root of (sub)tree

function DTL($examples, attributes, default$) returns a decision tree

   if $examples$ is empty then return $default$
   else if all $examples$ have the same classification then return the classification
   else if $attributes$ is empty then return MODE($examples$)
   else
      $best \leftarrow$ CHOOSE-ATTRIBUTE($attributes, examples$)
      $tree \leftarrow$ a new decision tree with root test $best$
      for each value v_i of $best$ do
         $examples_i \leftarrow \{$elements of $examples$ with $best = v_i\}$
         $subtree \leftarrow$ DTL($examples_i, attributes - best$, MODE($examples$))
         add a branch to $tree$ with label v_i and subtree $subtree$
      return $tree$

Choosing an Attribute

- Idea: a good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative"
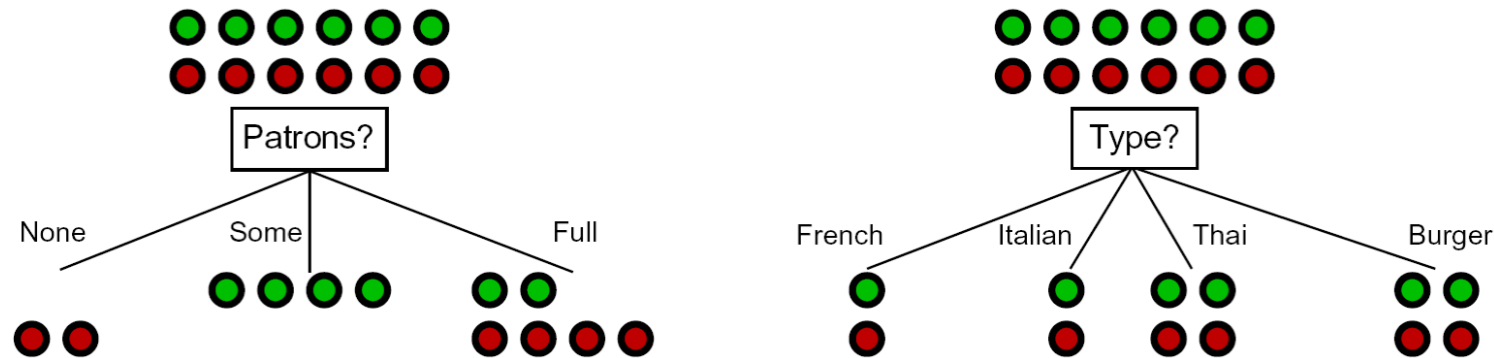
- So: we need a measure of how "good" a split is, even if the results aren't perfectly separated out

Entropy and Information

- **Information** answers questions
  - The more uncertain about the answer initially, the more information in the answer
  - Scale: bits
    - Answer to Boolean question with prior <1/2, 1/2>?
    - Answer to 4-way question with prior <1/4, 1/4, 1/4, 1/4>?
    - Answer to 4-way question with prior <0, 0, 0, 1>?
    - Answer to 3-way question with prior <1/2, 1/4, 1/4>?

- A probability p is typical of:
  - A uniform distribution of size 1/p
  - A code of length log 1/p

Entropy

- General answer: if prior is $\langle p_1, ..., p_n \rangle$:

  - Information is the expected code length

$$H(\langle p_1, \ldots, p_n \rangle) = E_p \log_2 1/p_i$$

$$= \sum_{i=1}^{n} -p_i \log_2 p_i$$

1 bit

0 bits

- Also called the entropy of the distribution

  - More uniform = higher entropy
  - More values = higher entropy
  - More peaked = lower entropy
  - Rare values almost "don't count"
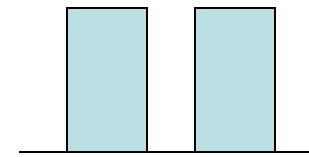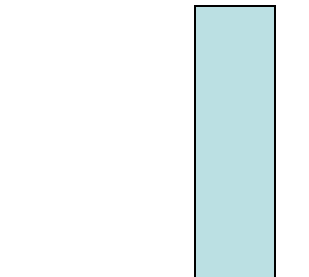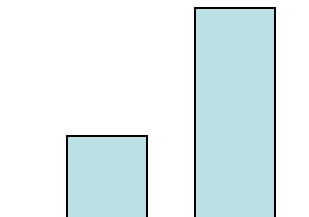
0.5 bit

Information Gain

- Back to decision trees!
- For each split, compare entropy before and after
    - Difference is the information gain
    - Problem: there's more than one distribution after split!
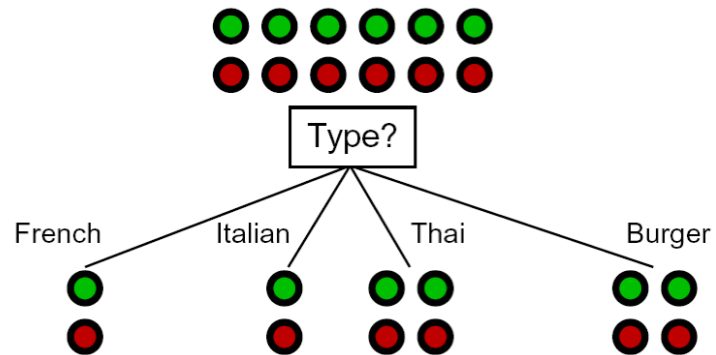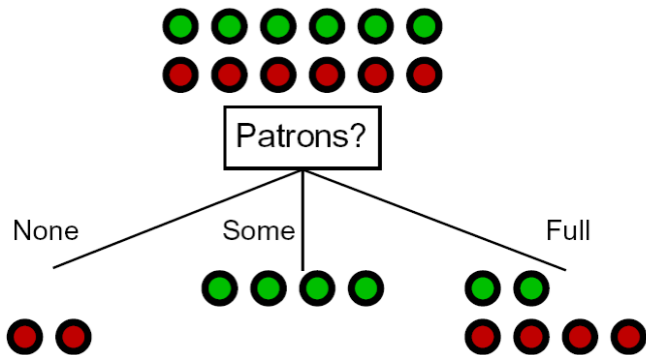
- Solution: use expected entropy, weighted by the number of examples
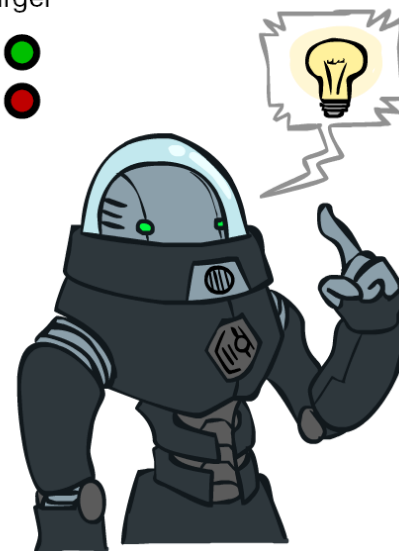
Next Step: Recurse

- Now we need to keep growing the tree!

- Two branches are done (why?)

- What to do under "full"?

  - See what examples are there…

| 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

Example: Learned Tree

- Decision tree learned from these 12 examples:

- Substantially simpler than "true" tree
  - A more complex hypothesis isn't justified by data
- Also: it's reasonable, but wrong

Example: Miles Per Gallon

40 Examples

mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	75to78	asia
bad	6	medium	medium	medium	medium	70to74	america
bad	4	medium	medium	medium	low	75to78	europe
bad	8	high	high	high	low	70to74	america
bad	6	medium	medium	medium	medium	70to74	america
bad	4	low	medium	low	medium	70to74	asia
bad	4	low	medium	low	low	70to74	asia
bad	8	high	high	high	low	75to78	america
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
bad	8	high	high	high	low	70to74	america
good	8	high	medium	high	high	79to83	america
bad	8	high	high	high	low	75to78	america
good	4	low	low	low	low	79to83	america
bad	6	medium	medium	medium	high	75to78	america
good	4	medium	low	low	low	79to83	america
good	4	low	low	medium	high	79to83	america
bad	8	high	high	high	low	70to74	america
good	4	low	medium	low	medium	75to78	europe
bad	5	medium	medium	medium	medium	75to78	europe

Find the First Split

- Look at information gain for each attribute

- Note that each attribute is correlated with the target!

- What do we split on?

Information gains using the training set (40 records)

mpg values: bad good

Input	Value	Distribution	Info Gain
cylinders	3		0.506731
	4		
	5		
	6		
	8		
displacement	low		0.223144
	medium		
	high		
horsepower	low		0.387605
	medium		
	high		
weight	low		0.304018
	medium		
	high		
acceleration	low		0.0642088
	medium		
	high		
modelyear	70to74		0.267964
	75to78		
	79to83		
maker	america		0.0437265
	asia		

Result: Decision Stump

mpg values: bad good

root

22 18

pchance = 0.001

cylinders = 3	cylinders = 4	cylinders = 5	cylinders = 6	cylinders = 8
0 0	4 17	1 0	8 0	9 1
Predict bad	Predict good	Predict bad	Predict bad	Predict bad

Second Level

mpg values: bad good

root
22 18
pchance = 0.001

cylinders = 3	cylinders = 4	cylinders = 5	cylinders = 6	cylinders = 8
0 0	4 17	1 0	8 0	9 1
Predict bad	pchance = 0.135	Predict bad	Predict bad	pchance = 0.085

maker = america	maker = asia	maker = europe	horsepower = low	horsepower = medium	horsepower = high
0 10	2 5	2 2	0 0	0 1	9 0
Predict good	Predict good	Predict bad	Predict bad	Predict good	Predict bad

Final Tree

root

22 18

pchance = 0.001

---

cylinders = 3	cylinders = 4	cylinders = 5	cylinders = 6	cylinders = 8
0 0	4 17	1 0	8 0	9 1
Predict bad	pchance = 0.135	Predict bad	Predict bad	pchance = 0.085

---

maker = america	maker = asia	maker = europe	horsepower = low	horsepower = medium	horsepower = high
0 10	2 5	2 2	0 0	0 1	9 0
Predict good	pchance = 0.317	pchance = 0.717	Predict bad	Predict good	Predict bad

---

horsepower = low	horsepower = medium	horsepower = high	acceleration = low	acceleration = medium	acceleration = high
0 4	2 1	0 0	1 0	0 1	1
Predict good	pchance = 0.894	Predict bad	Predict bad	Predict good	

---

acceleration = low	acceleration = medium	acceleration = high		
1 0	1 1	0		0 0
Predict bad	(unexpandable)	Predict bad	Predict good	Predict bad
	Predict bad			

Information gains using the training set (2 records)

mpg values: bad   good

Input	Value	Distribution	Info Gain
cylinders	3		0
	4	▇▇	
	5		
	6		
	8		
displacement	low	▇▇	0
	medium		
	high		
horsepower	low		0
	medium	▇▇	
	high		
weight	low	▇▇	0
	medium		
	high		
acceleration	low		0
	medium	▇▇	
	high		
modelyear	70to74	▇▇	0
	75to78		
	79to83		
maker	america		0
	asia	▇▇	
	europe		

mpg values: bad   good

root

22 18

pchance = 0.001

	Num Errors	Set Size	Percent Wrong
Training Set	1	40	2.50
Test Set	74	352	21.02

epower = high

ict bad

horsepower = low | horsepower = medium | horsepower = high | acceleration = low | acceleration = medium | acceleration = high

= 0.717

The test set error is much worse than the training set error...

...why?

= 79to83

Predict bad | (unexpandable) | Predict bad | Predict good | Predict bad | Predict bad
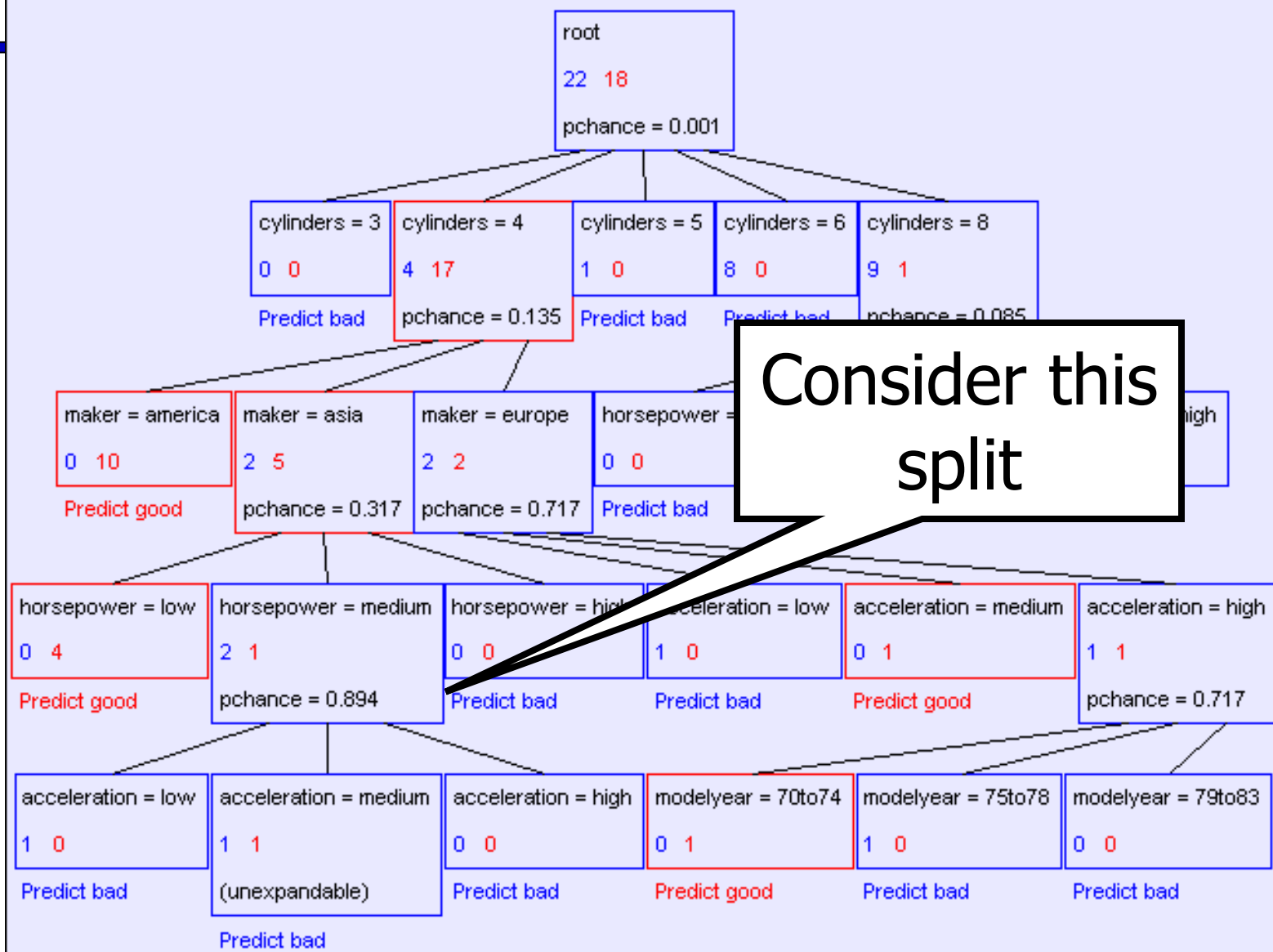
Predict bad

Reminder: Overfitting

- Overfitting:
  - When you stop modeling the patterns in the training data (which generalize)
  - And start modeling the noise (which doesn't)

- We had this before:
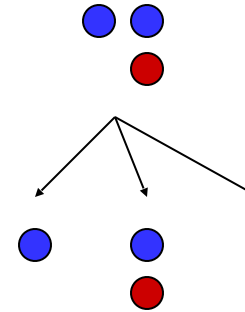 - Naïve Bayes: needed to smooth
  - Perceptron: early stopping

Significance of a Split

- Starting with:
    - Three cars with 4 cylinders, from Asia, with medium HP
    - 2 bad MPG
    - 1 good MPG

- What do we expect from a three-way split?
    - Maybe each example in its own subset?
    - Maybe just what we saw in the last slide?

- Probably shouldn't split if the counts are so small they could be due to chance

- A chi-squared test can tell us how likely it is that deviations from a perfect split are due to chance*

- Each split will have a significance value, p_{CHANCE}

Keeping it General

■ Pruning:

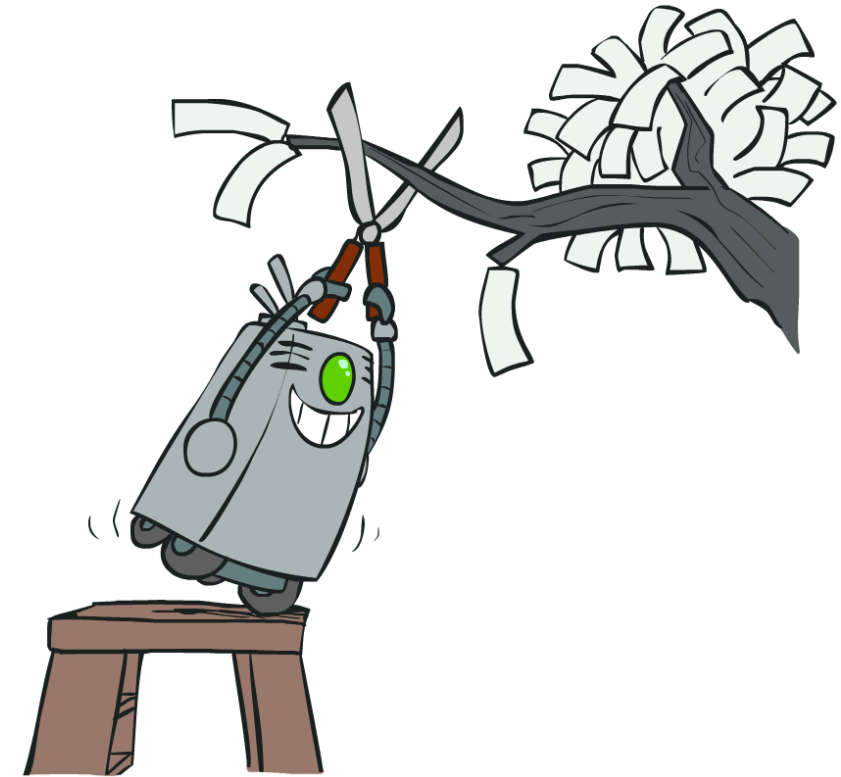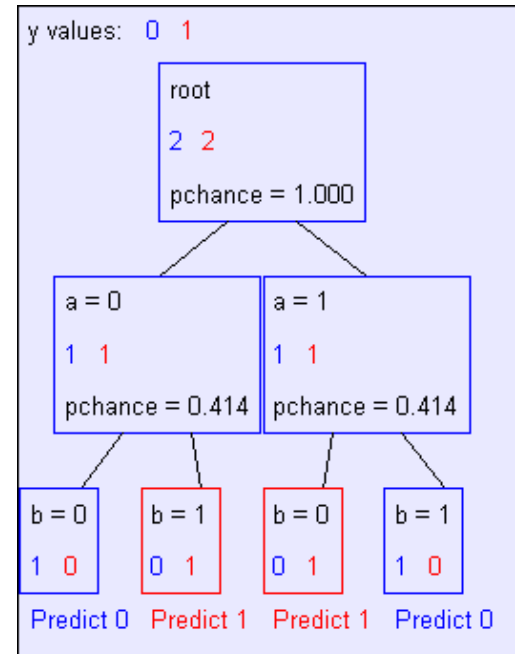   ■ Build the full decision tree

   ■ Begin at the bottom of the tree

   ■ Delete splits in which

$$p_{CHANCE} > MaxP_{CHANCE}$$

   ■ Continue working upward until there are no more prunable nodes

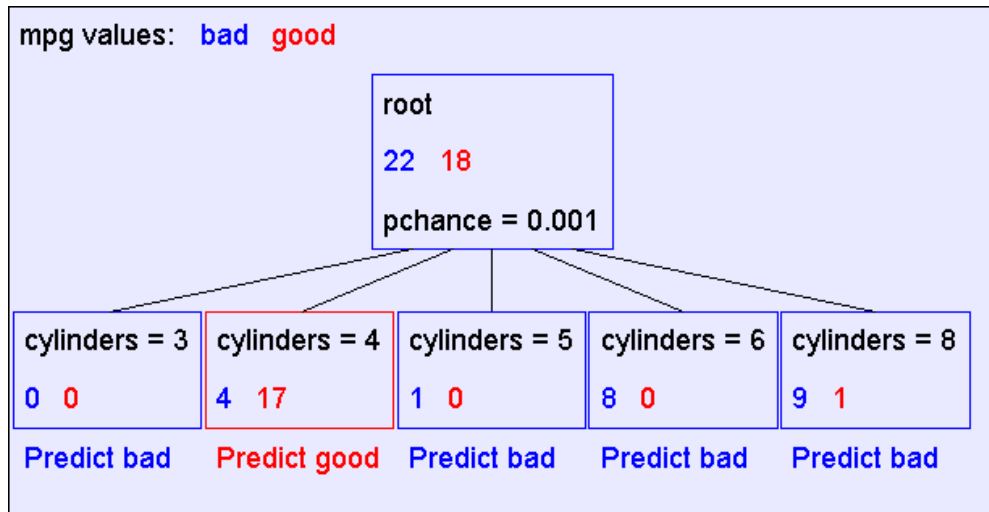   ■ Note: some chance nodes may not get pruned because they were "redeemed" later

y = a XOR b

a	b	y
0	0	0
0	1	1
1	0	1
1	1	0

Pruning example

- With MaxP$_{CHANGE}$ = 0.1:

mpg values: bad good

root
22 18
pchance = 0.001

cylinders = 3	cylinders = 4	cylinders = 5	cylinders = 6	cylinders = 8
0 0	4 17	1 0	8 0	9 1
Predict bad	Predict good	Predict bad	Predict bad	Predict bad

Note the improved test set accuracy compared with the unpruned tree

	Num Errors	Set Size	Percent Wrong
Training Set	5	40	12.50
Test Set	56	352	15.91

Regularization

- MaxP$_{CHANCE}$ is a regularization parameter

- Generally, set it using held-out data (as usual)

Training

Held-out / Test

Accuracy

Decreasing

Increasing

MaxP$_{CHANCE}$

Small Trees
High Bias

Large Trees
High Variance
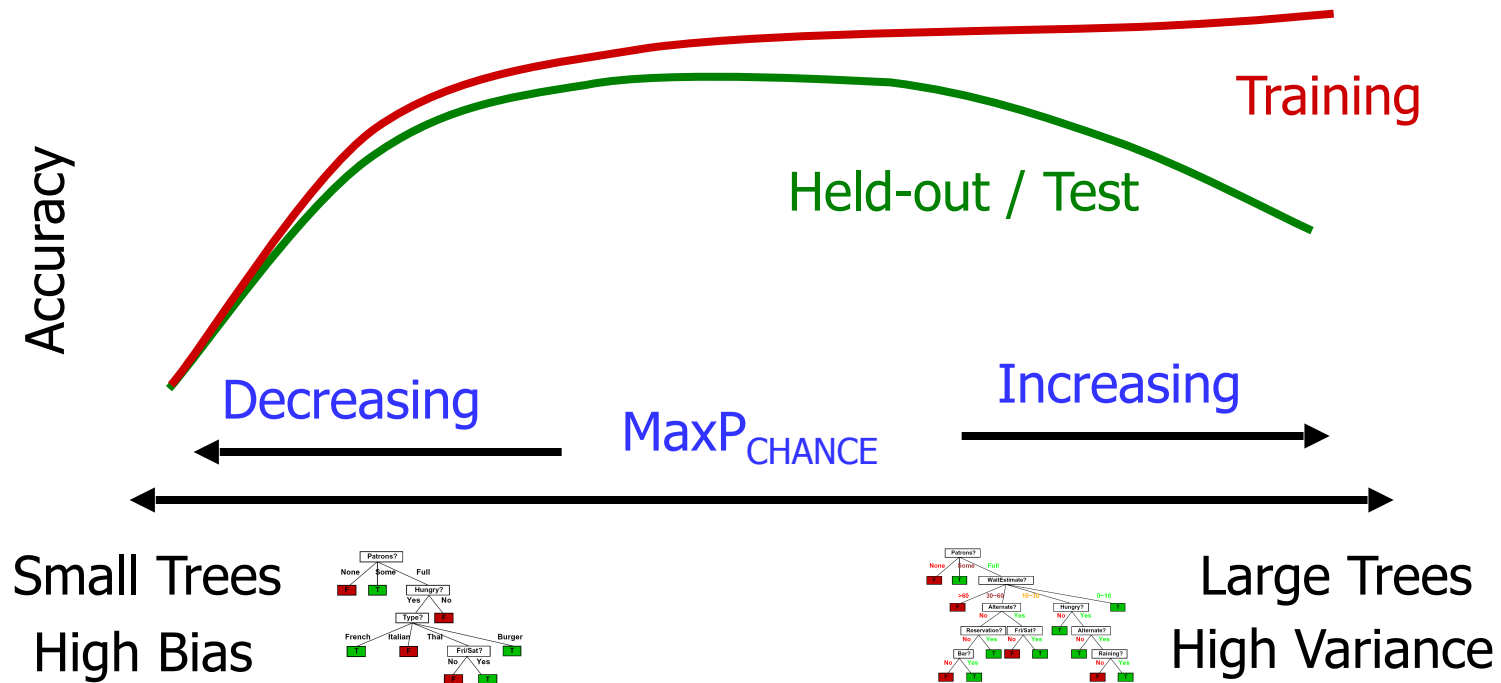
Two Ways of Controlling Overfitting

- Limit the hypothesis space
  - E.g. limit the max depth of trees
  - Easier to analyze

- Regularize the hypothesis selection
  - E.g. chance cutoff
  - Skip most of the hypotheses unless data is clear
  - Usually done in practice

Expert Systems

- Expert Systems are collections of if-then rules that capture the knowledge of a human expert to carry out a specific task

- Rules have certainty factors that give a probability of their importance or likelihood

- Rules are independent of one another but can add information that other rules use

- Inference Engine decides which rule to apply

- Similar to Decision Trees which can be thought of as a large collection of if-then-else statements, but no tree structure is used

Expert System Rules

- Sample rules for car mileage
  - if model-year is "79to83" then mileage <— "good"
  - if model-year is "70to74" then mileage <— "bad"
  - if model-year is "75to78" and displacement <— "low" then mileage <— "good"
  - if model-year is "75to78" and displacement is not "low" then mileage <— "bad"
  - if weight is "low" then mpg_trending_low <— true

Next Lecture: Large Language Models & Transformers

- Large Langauge Models
  - Transformers