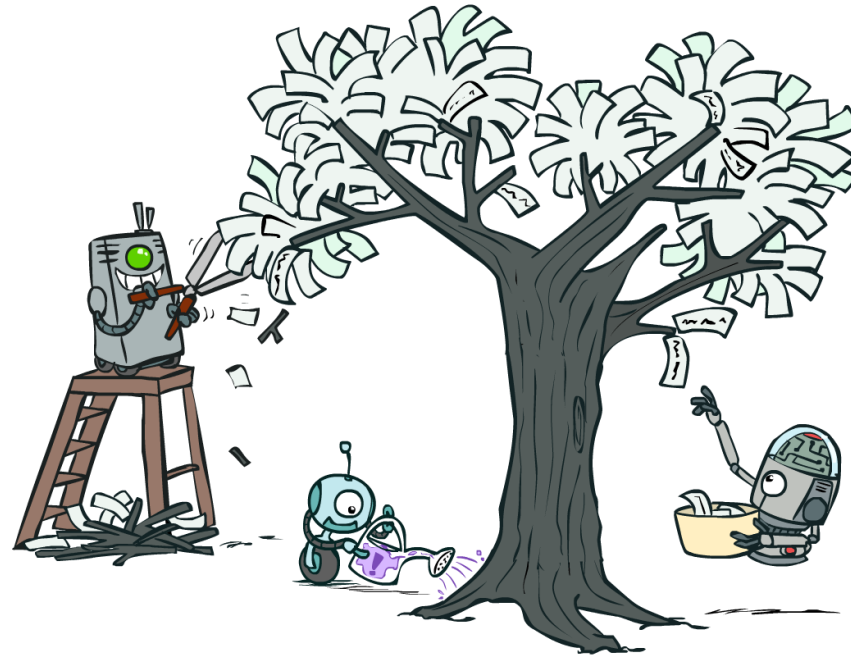


CS 188: Artificial Intelligence

Neural Nets (wrap-up) and Decision Trees



Instructors: Michele Van Dyne,

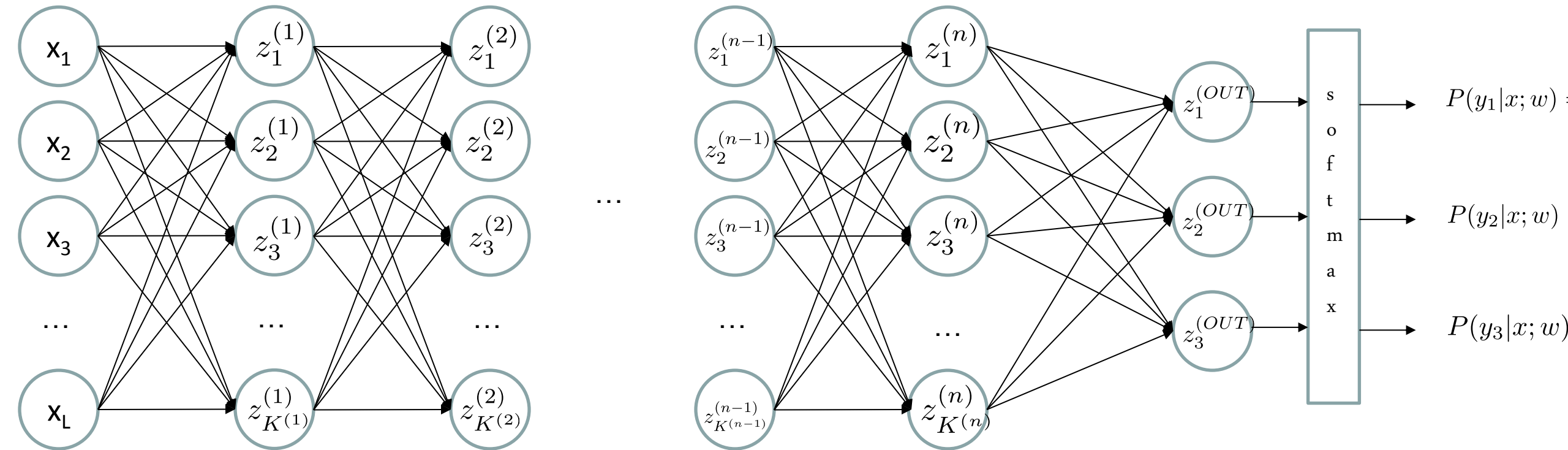
Adapted from Pieter Abbeel and Dan Klein --- University of California, Berkeley

[These slides were created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley. All CS188 materials are available at <http://ai.berkeley.edu>.]

Today

- Neural Nets -- wrap
- Formalizing Learning
 - Consistency
 - Simplicity
- Decision Trees
 - Expressiveness
 - Information Gain
 - Overfitting

Deep Neural Network



$$z_i^{(k)} = g\left(\sum_j W_{i,j}^{(k-1,k)} z_j^{(k-1)}\right)$$

g = nonlinear activation function

Deep Neural Network: Also Learn the Features!

- Training the deep neural network is just like logistic regression:

$$\max_w ll(w) = \max_w \sum_i \log P(y^{(i)} | x^{(i)}; w)$$

just w tends to be a much, much larger vector 😊

→ just run gradient ascent

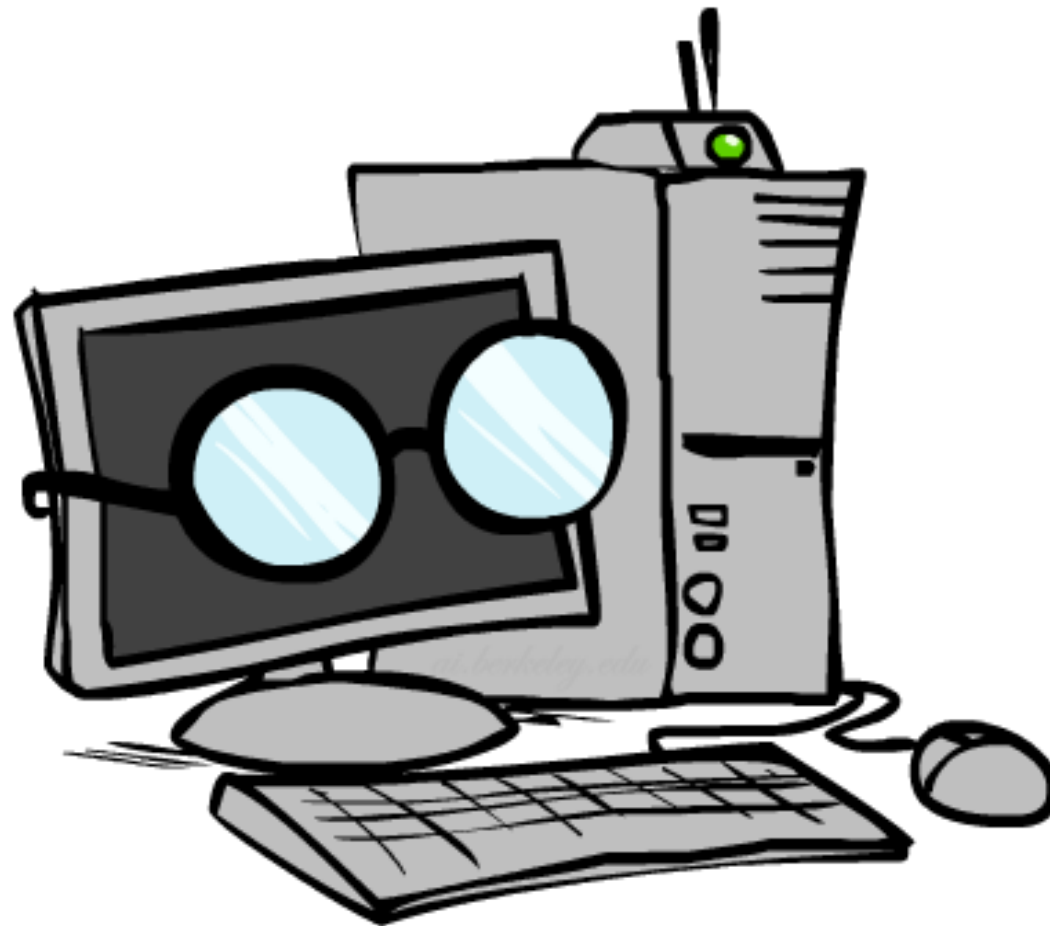
+ stop when log likelihood of hold-out data starts to decrease

Neural Networks Properties

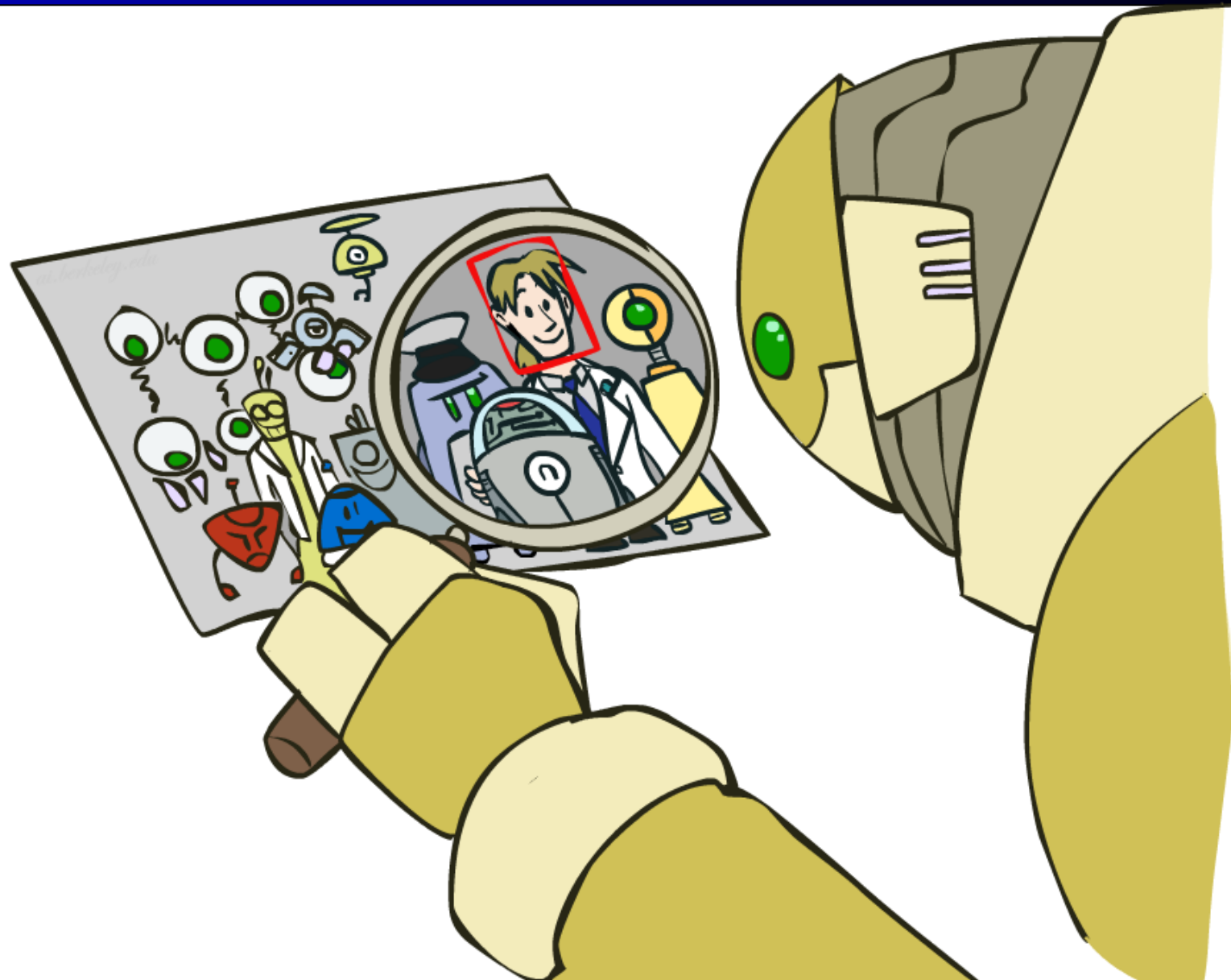
- Theorem (Universal Function Approximators). A two-layer neural network with a sufficient number of neurons can approximate any continuous function to any desired accuracy.
- Practical considerations
 - Can be seen as learning the features
 - Large number of neurons
 - Danger for overfitting
 - (hence early stopping!)

How well does it work?

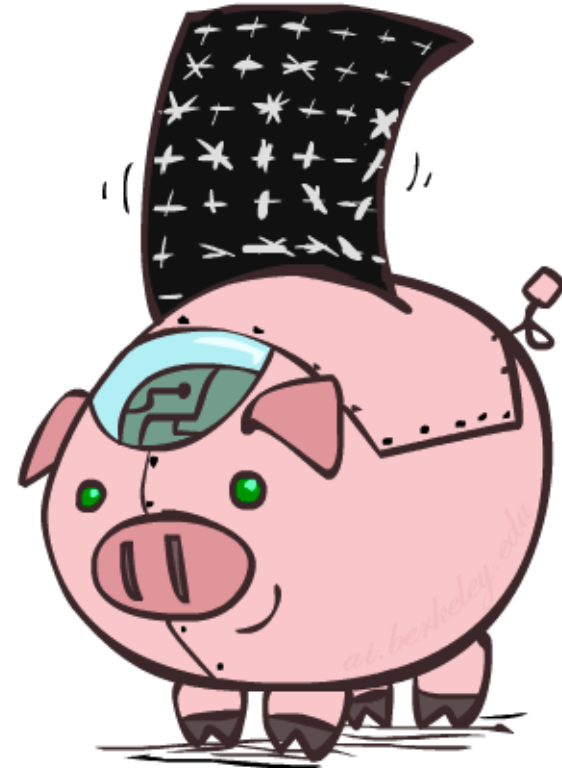
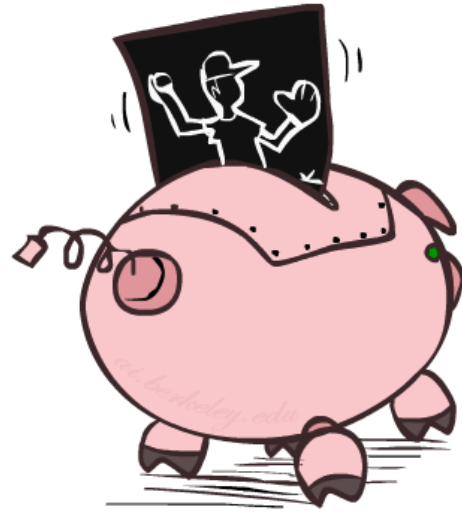
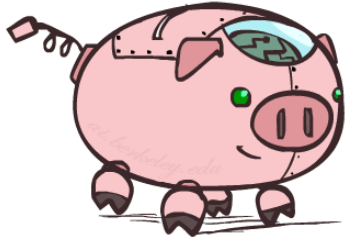
Computer Vision



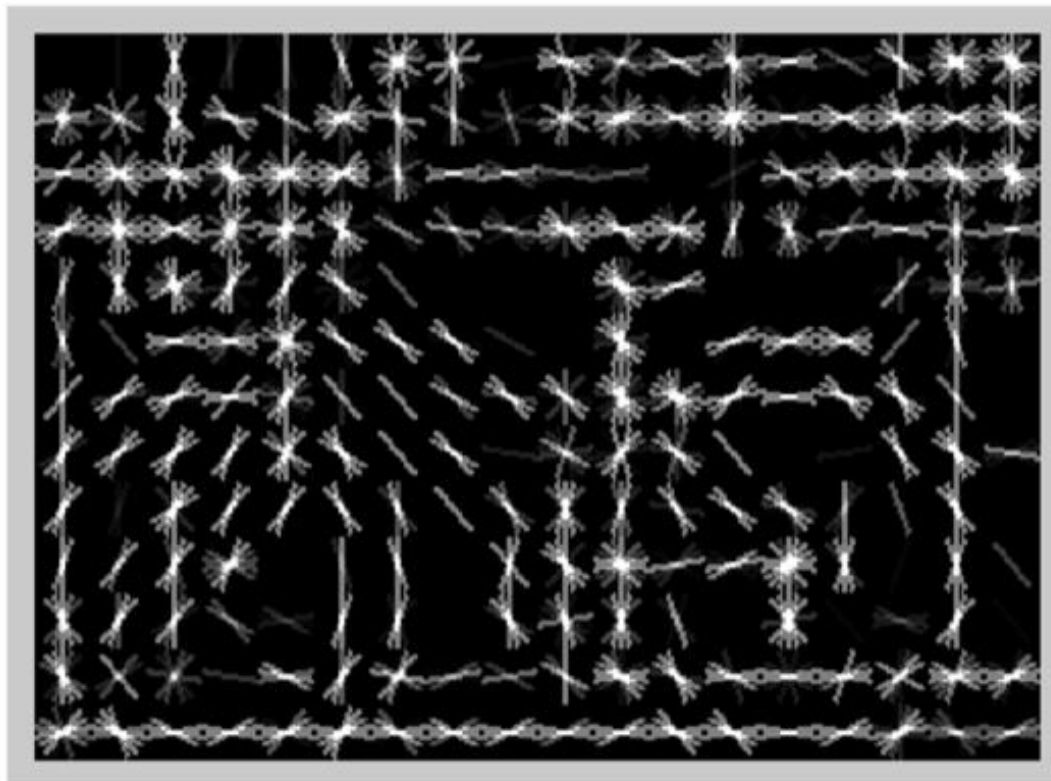
Object Detection



Manual Feature Design



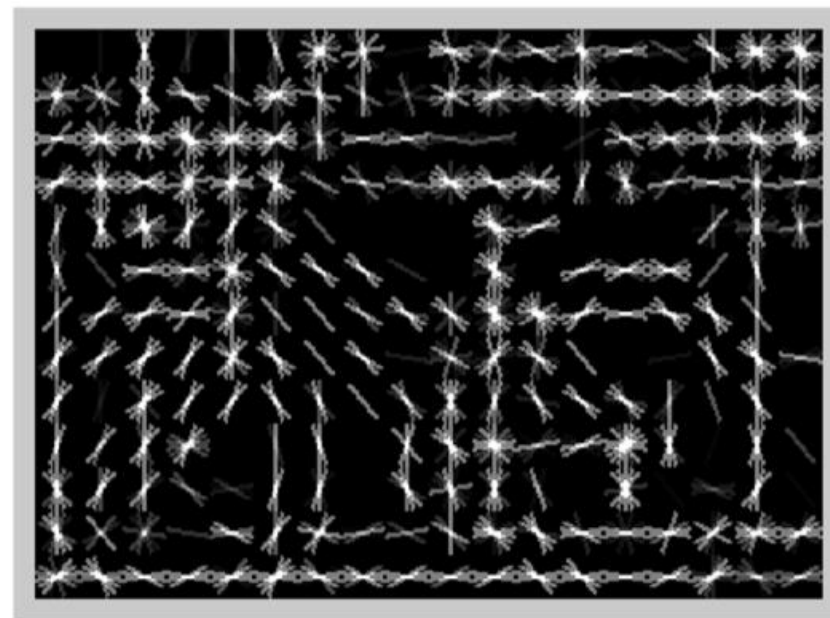
Features and Generalization



Features and Generalization



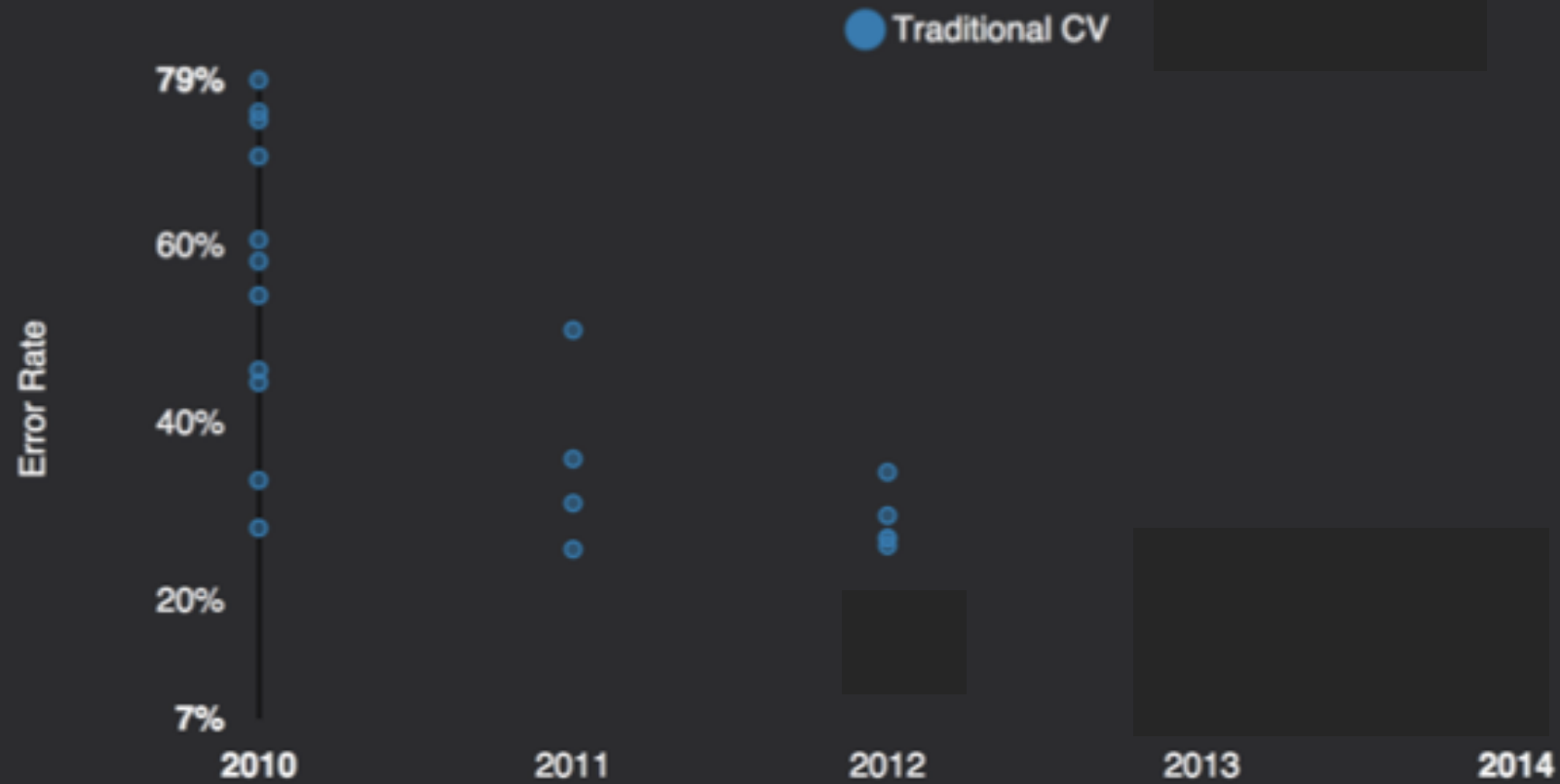
Image



HoG

Performance

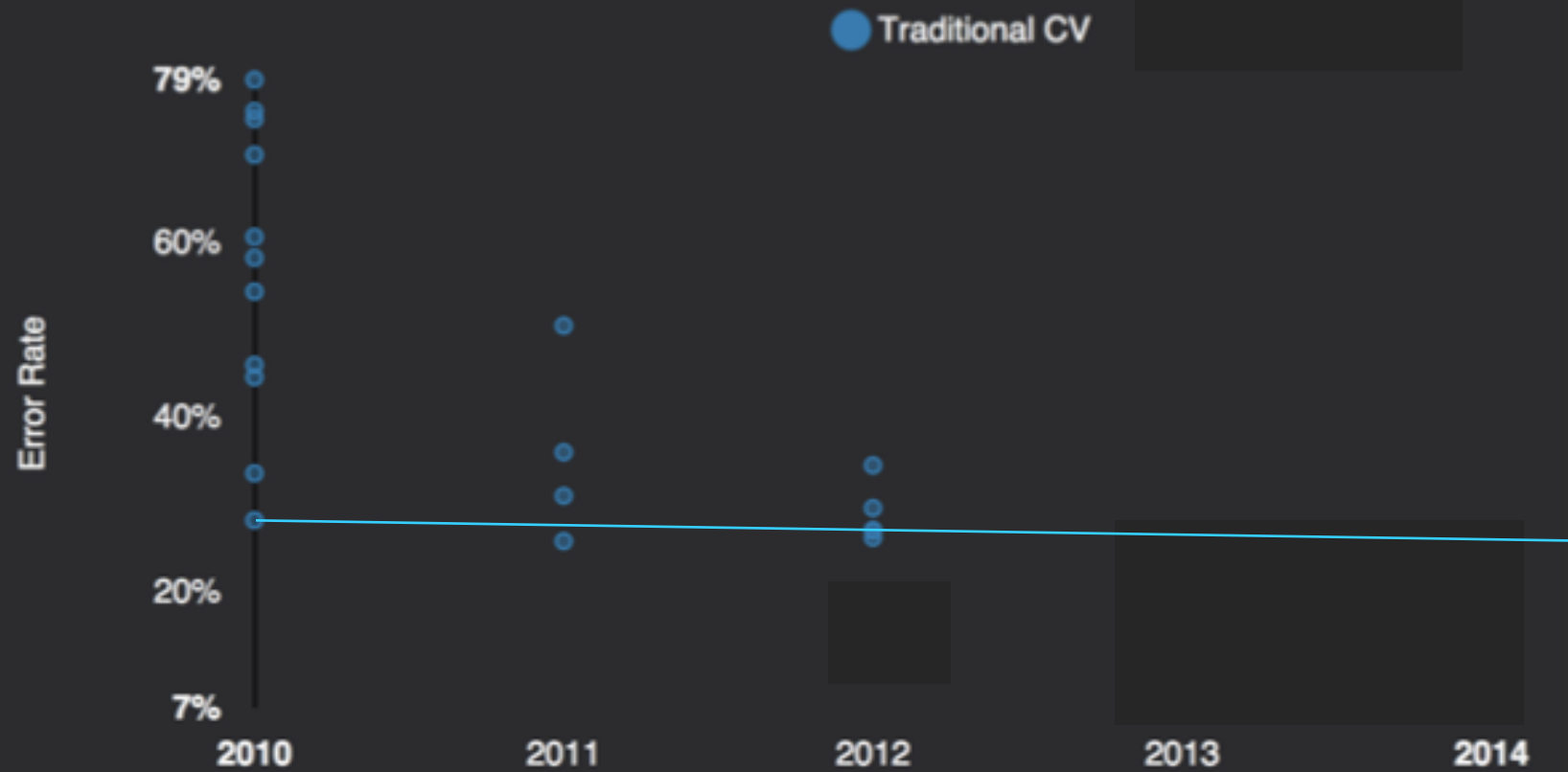
ImageNet Error Rate 2010-2014



graph credit Matt
Zeiler, Clarifai

Performance

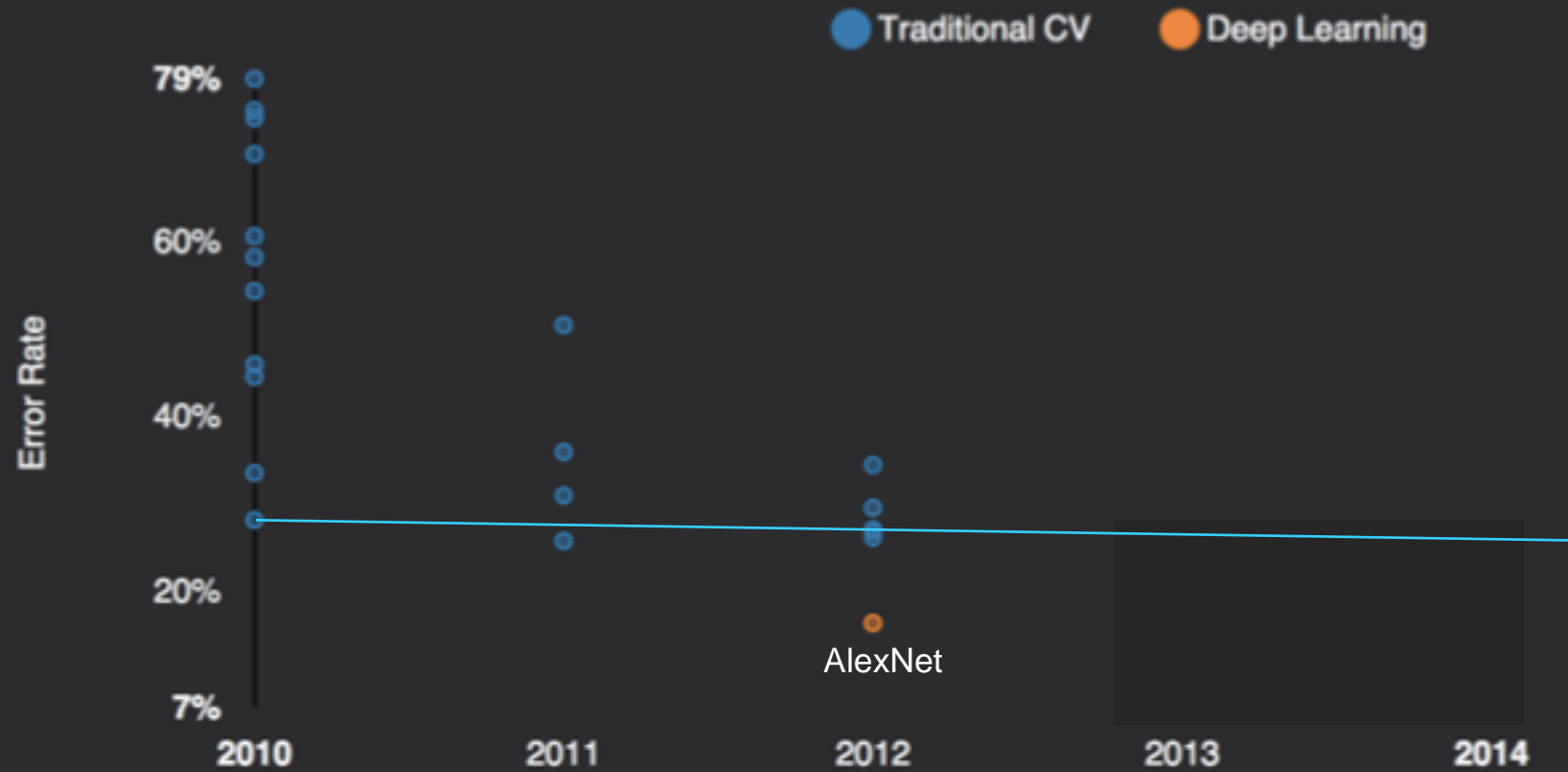
ImageNet Error Rate 2010-2014



graph credit Matt
Zeiler, Clarifai

Performance

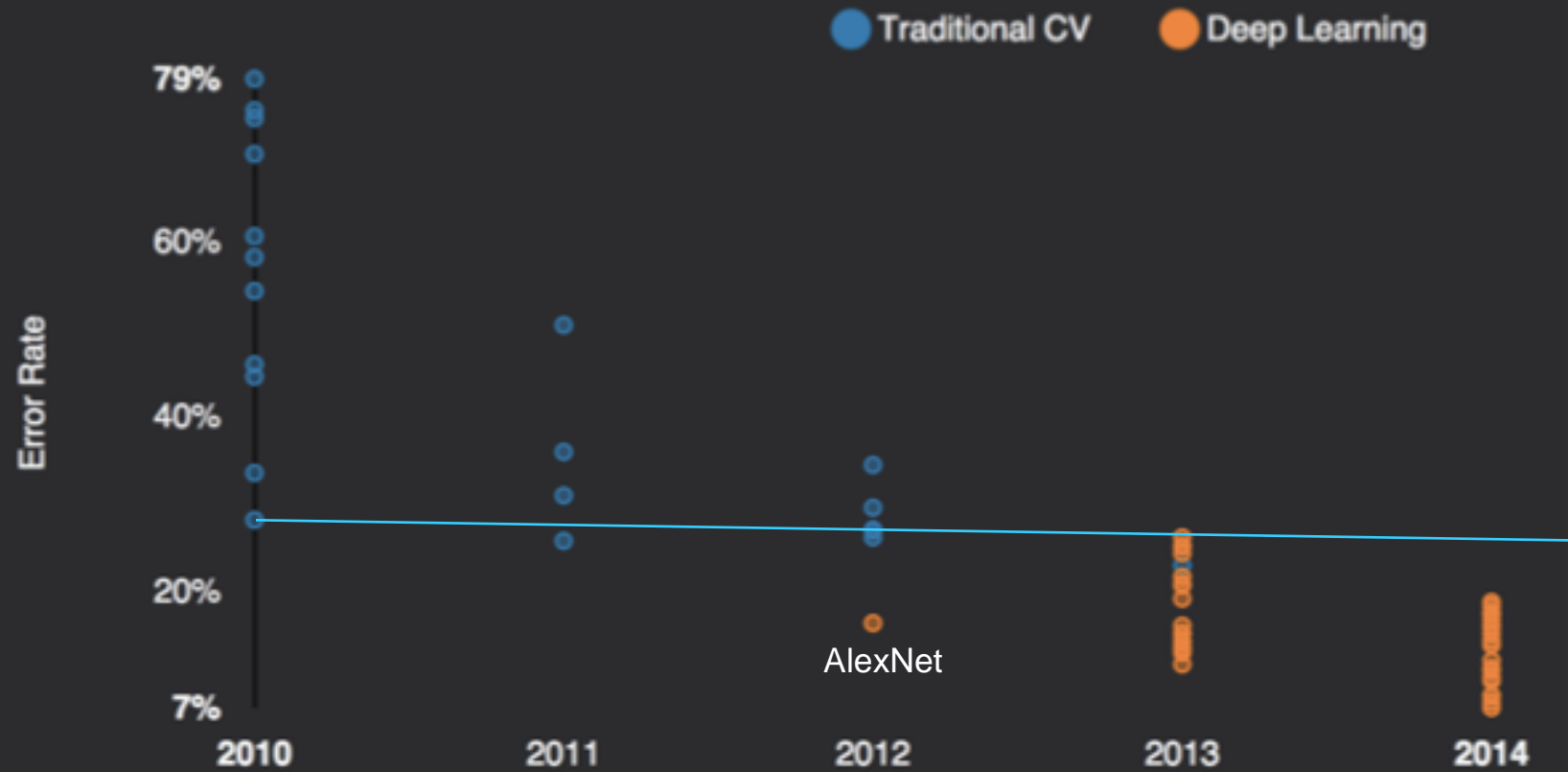
ImageNet Error Rate 2010-2014



graph credit Matt
Zeiler, Clarifai

Performance

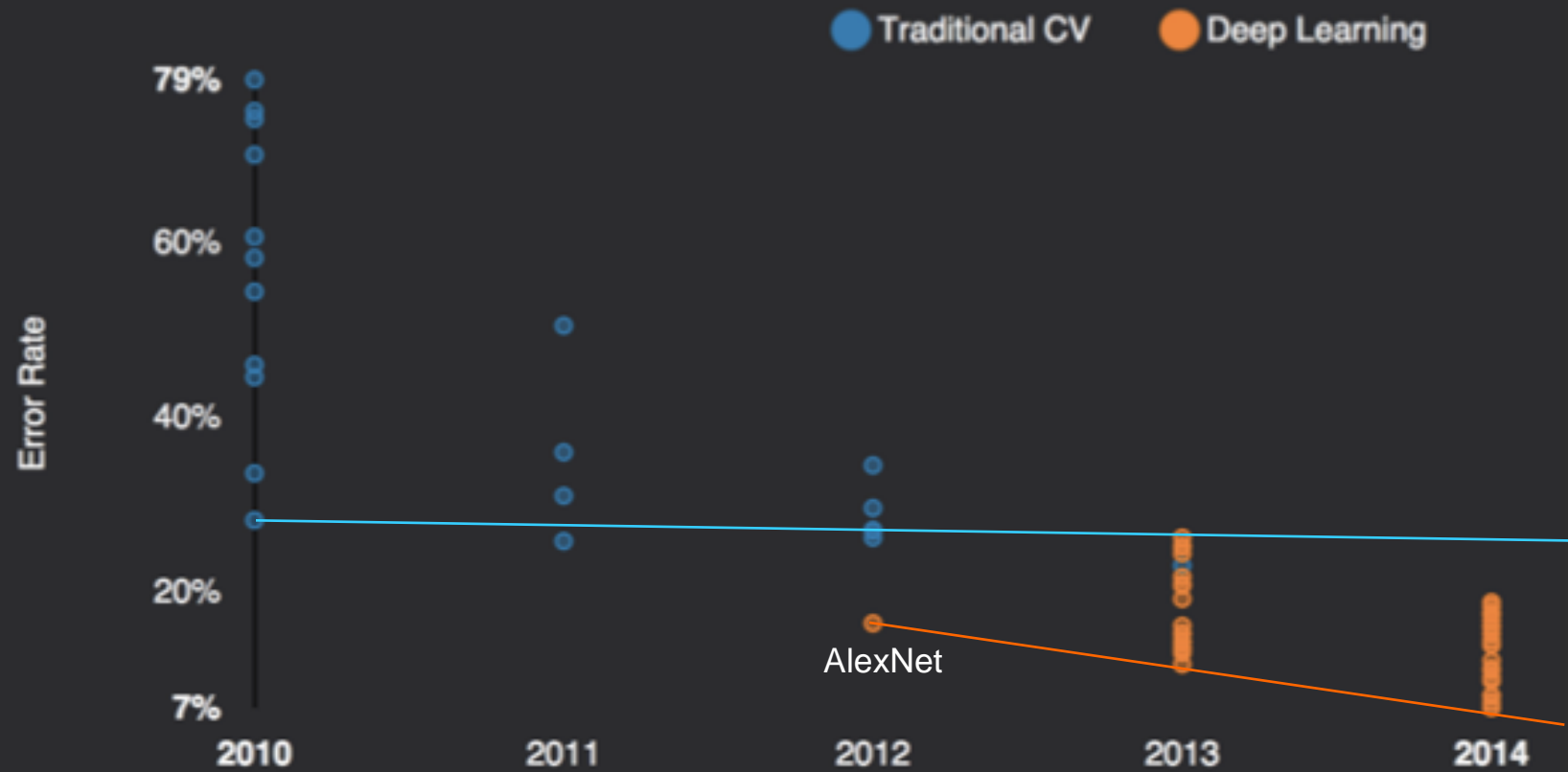
ImageNet Error Rate 2010-2014



graph credit Matt
Zeiler, Clarifai

Performance

ImageNet Error Rate 2010-2014



graph credit Matt Zeiler, Clarifai

MS COCO Image Captioning Challenge



"man in black shirt is playing guitar."



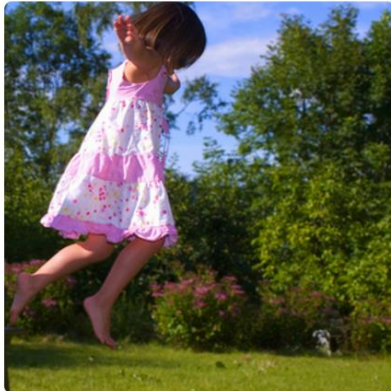
"construction worker in orange safety vest is working on road."



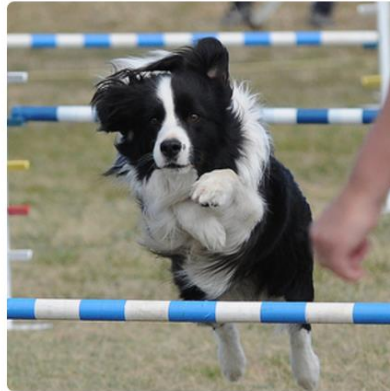
"two young girls are playing with lego toy."



"boy is doing backflip on wakeboard."



"girl in pink dress is jumping in air."



"black and white dog jumps over bar."



"young girl in pink shirt is swinging on swing."



"man in blue wetsuit is surfing on wave."

Karpathy & Fei-Fei, 2015; Donahue et al., 2015; Xu et al, 2015; many more

Visual QA Challenge

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh



What vegetable is on the plate?

Neural Net: broccoli

Ground Truth: broccoli



What color are the shoes on the person's feet ?

Neural Net: brown

Ground Truth: brown



How many school busses are there?

Neural Net: 2

Ground Truth: 2



What sport is this?

Neural Net: baseball

Ground Truth: baseball



What is on top of the refrigerator?

Neural Net: magnets

Ground Truth: cereal



What uniform is she wearing?

Neural Net: shorts

Ground Truth: girl scout



What is the table number?

Neural Net: 4

Ground Truth: 40



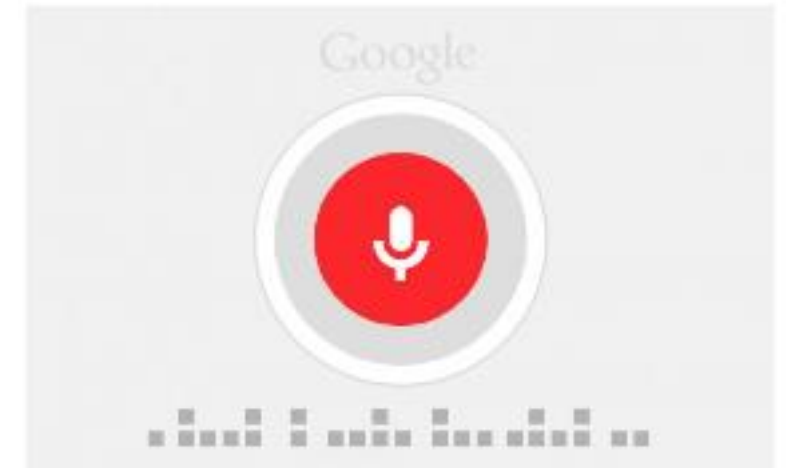
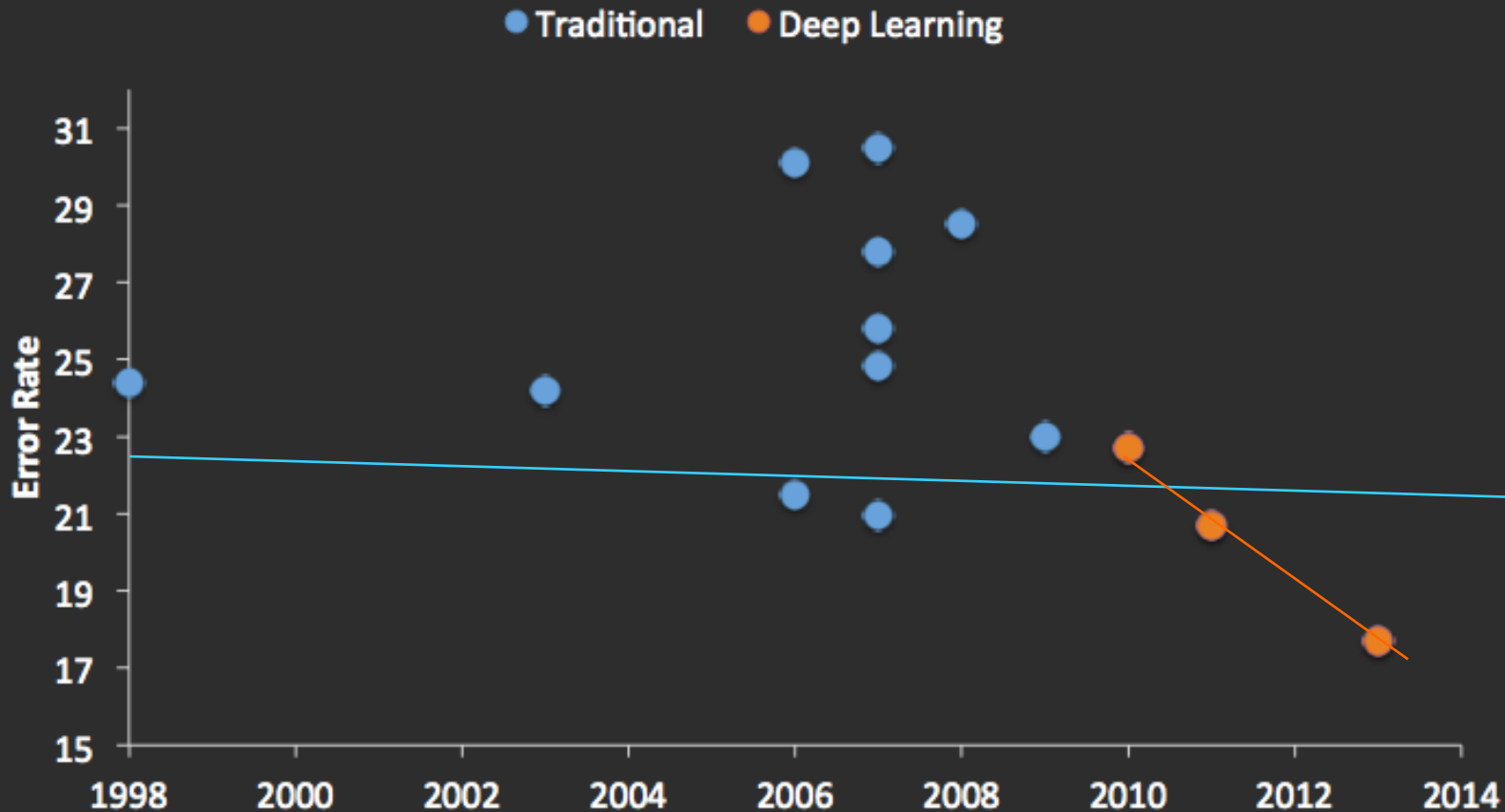
What are people sitting under in the back?

Neural Net: bench

Ground Truth: tent

Speech Recognition

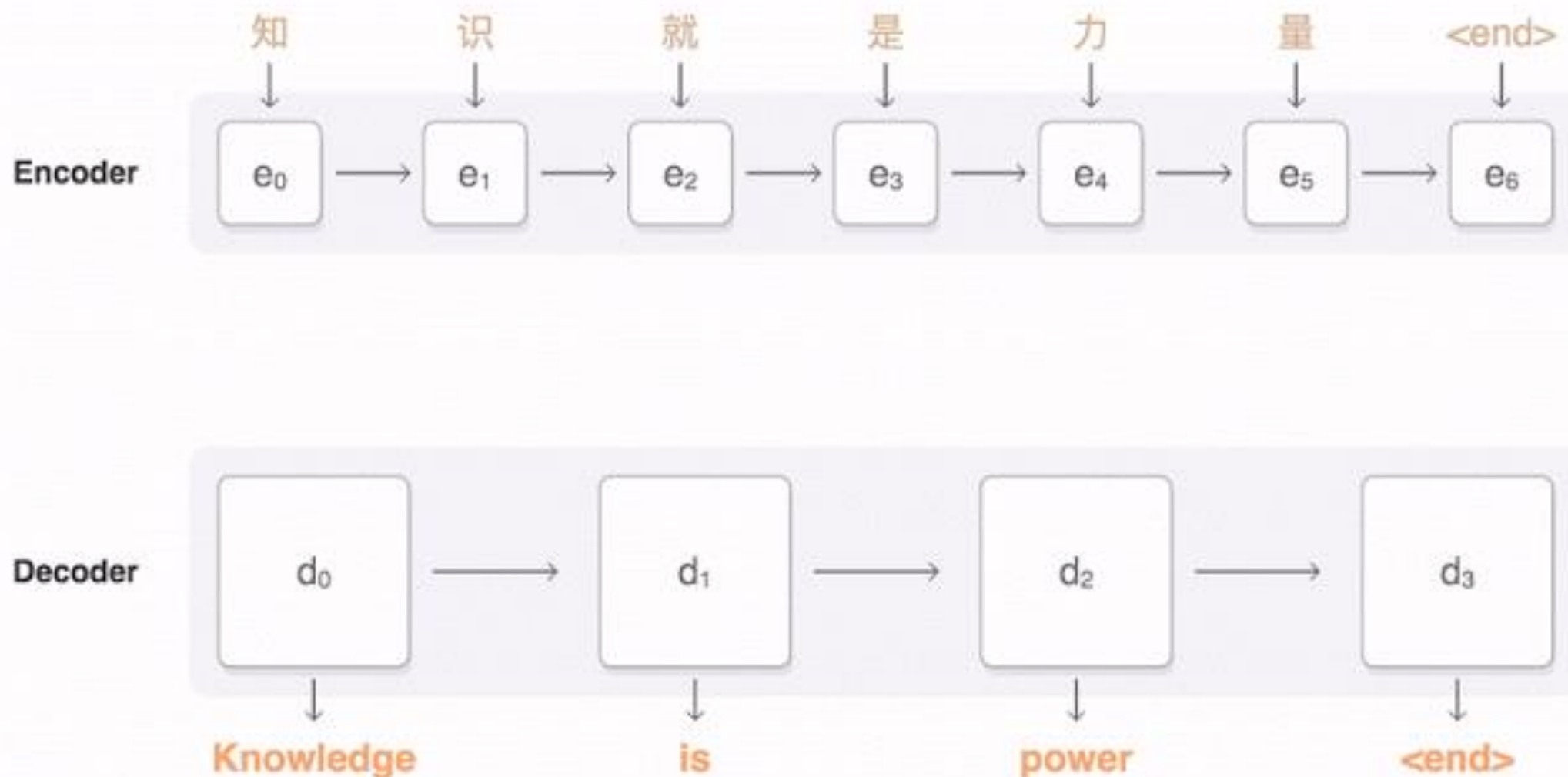
TIMIT Speech Recognition



graph credit Matt Zeiler, Clarifai

Machine Translation

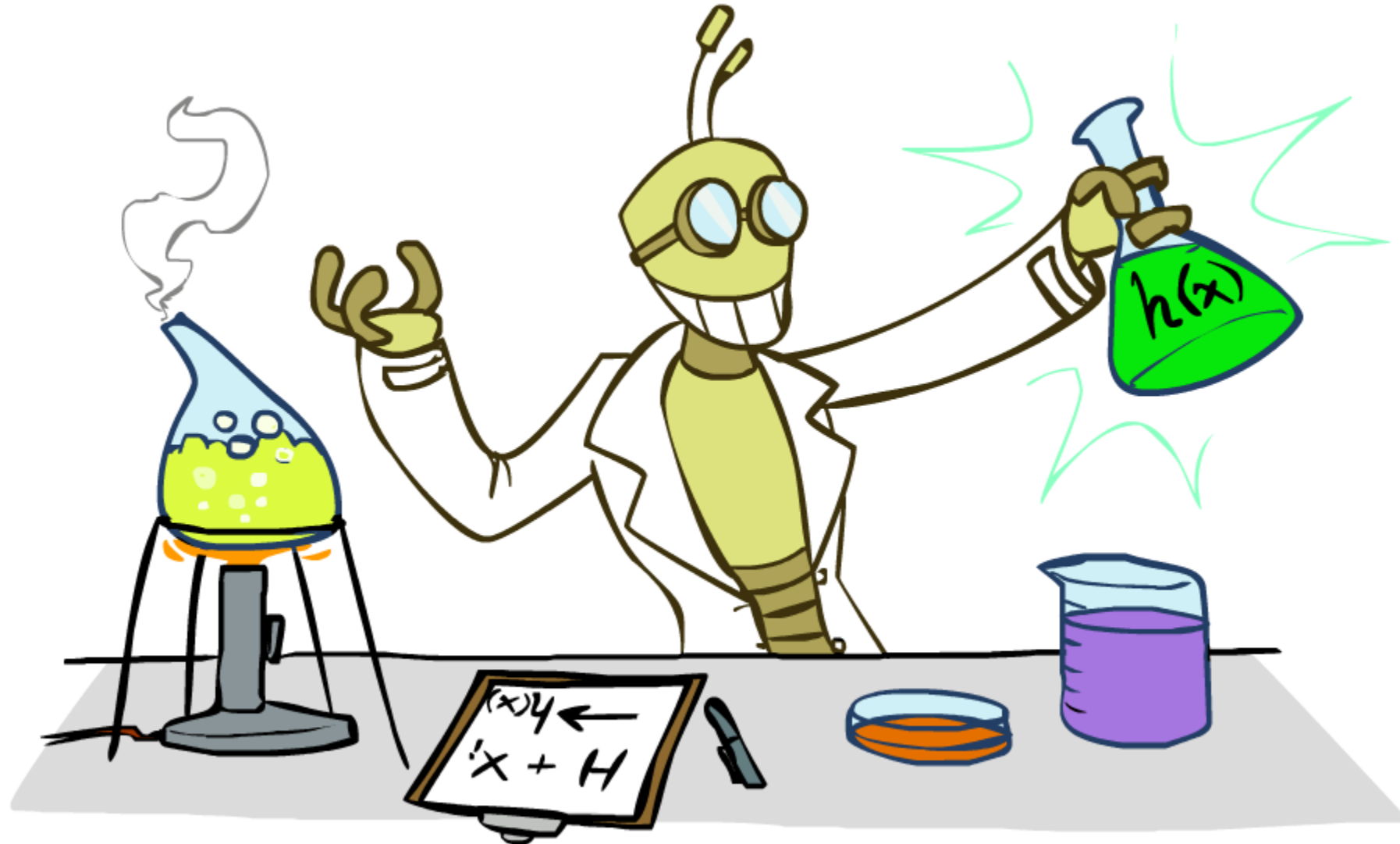
Google Neural Machine Translation (in production)



Today

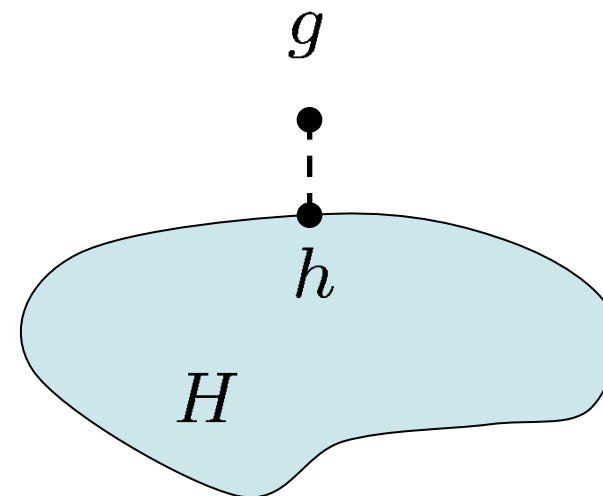
- Neural Nets -- wrap
- **Formalizing Learning**
 - **Consistency**
 - **Simplicity**
- **Decision Trees**
 - Expressiveness
 - Information Gain
 - Overfitting
- Clustering

Inductive Learning



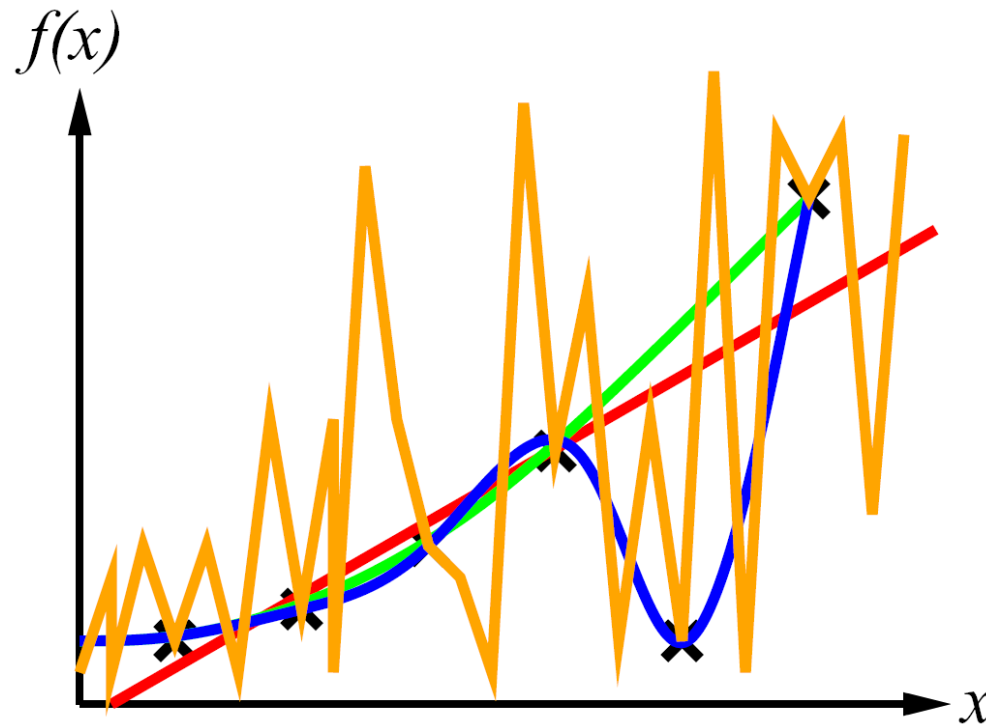
Inductive Learning (Science)

- 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 , etc.)



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



Reminder: Features

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

Example	Attributes										Target
	<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	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>0–10</i>	<i>T</i>
X_2	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>30–60</i>	<i>F</i>
X_3	<i>F</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>Some</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Burger</i>	<i>0–10</i>	<i>T</i>
X_4	<i>T</i>	<i>F</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>10–30</i>	<i>T</i>
X_5	<i>T</i>	<i>F</i>	<i>T</i>	<i>F</i>	<i>Full</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>>60</i>	<i>F</i>
X_6	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$</i>	<i>T</i>	<i>T</i>	<i>Italian</i>	<i>0–10</i>	<i>T</i>
X_7	<i>F</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>None</i>	<i>\$</i>	<i>T</i>	<i>F</i>	<i>Burger</i>	<i>0–10</i>	<i>F</i>
X_8	<i>F</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$</i>	<i>T</i>	<i>T</i>	<i>Thai</i>	<i>0–10</i>	<i>T</i>
X_9	<i>F</i>	<i>T</i>	<i>T</i>	<i>F</i>	<i>Full</i>	<i>\$</i>	<i>T</i>	<i>F</i>	<i>Burger</i>	<i>>60</i>	<i>F</i>
X_{10}	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>Italian</i>	<i>10–30</i>	<i>F</i>
X_{11}	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>None</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>0–10</i>	<i>F</i>
X_{12}	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Burger</i>	<i>30–60</i>	<i>T</i>

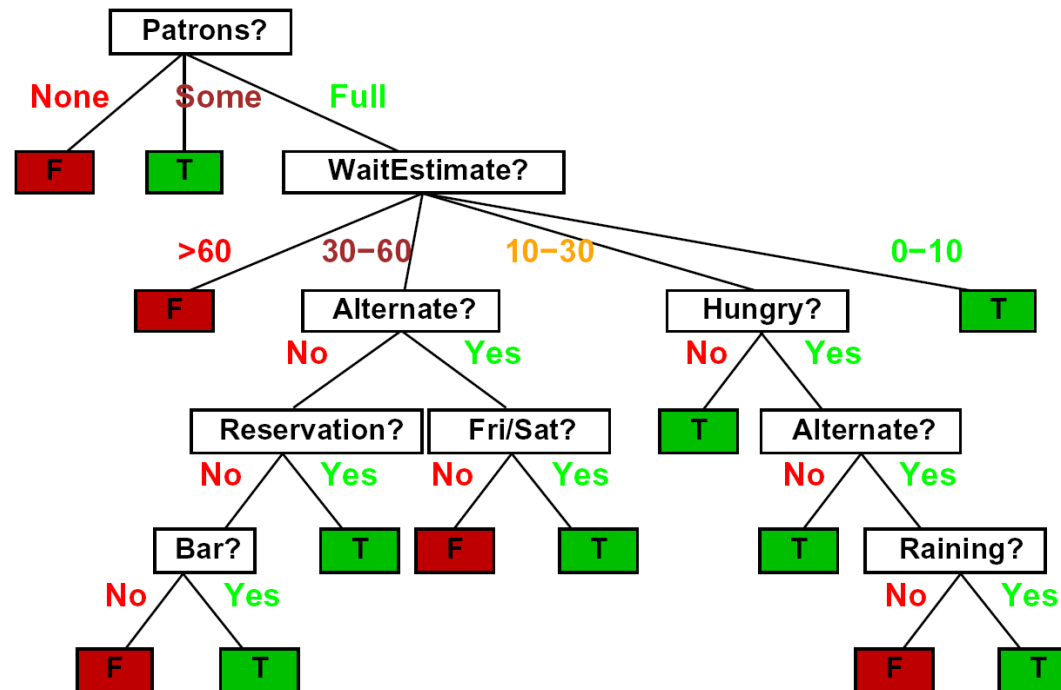
Decision Trees

- Compact representation of a function:

- Truth table
- Conditional probability table
- Regression values

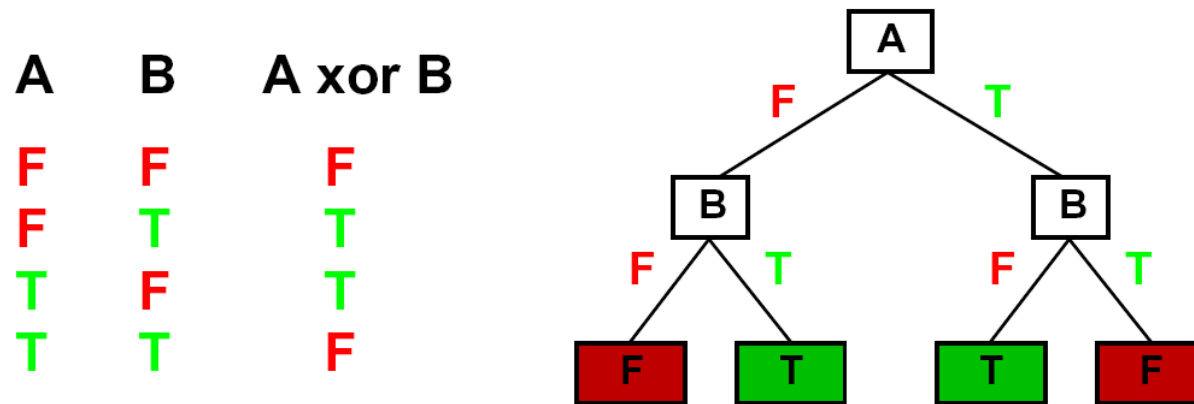
- True function

- Realizable: in H



Expressiveness of DTs

- Can express any function of the features



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

- However, we hope for compact trees

Comparison: Perceptrons

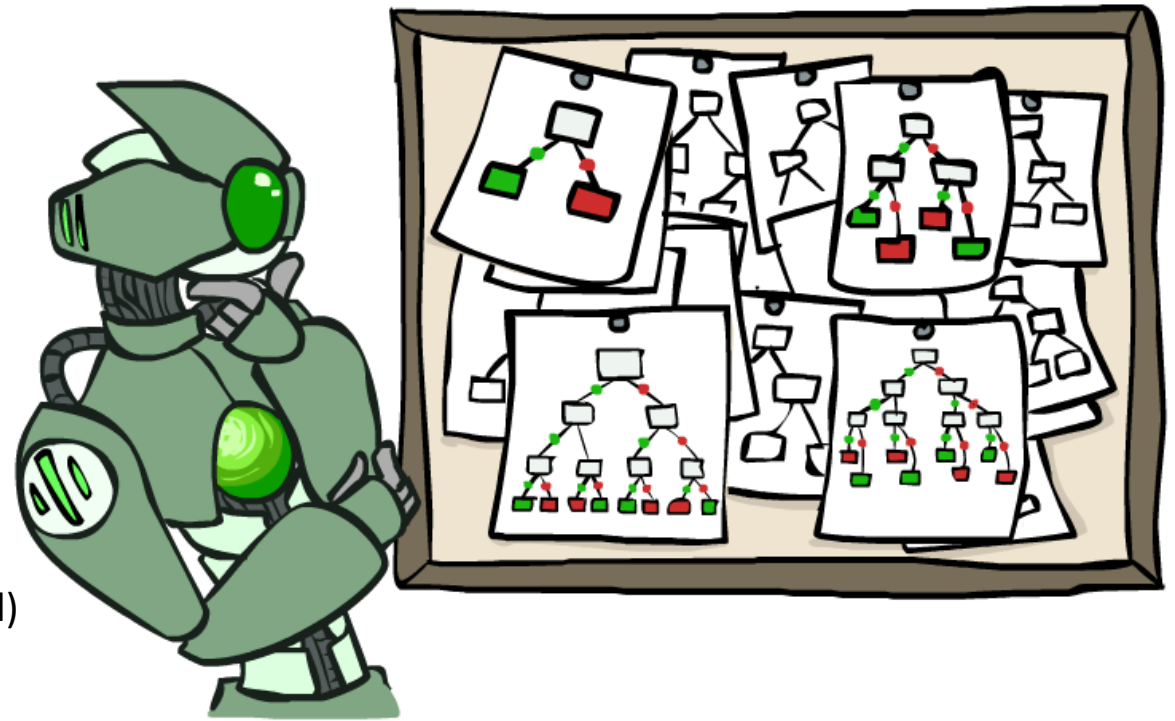
- What is the expressiveness of a perceptron over these features?

Example	Attributes										Target
	<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	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>0-10</i>	<i>T</i>
X_2	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>30-60</i>	<i>F</i>

- 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 \wedge 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**)



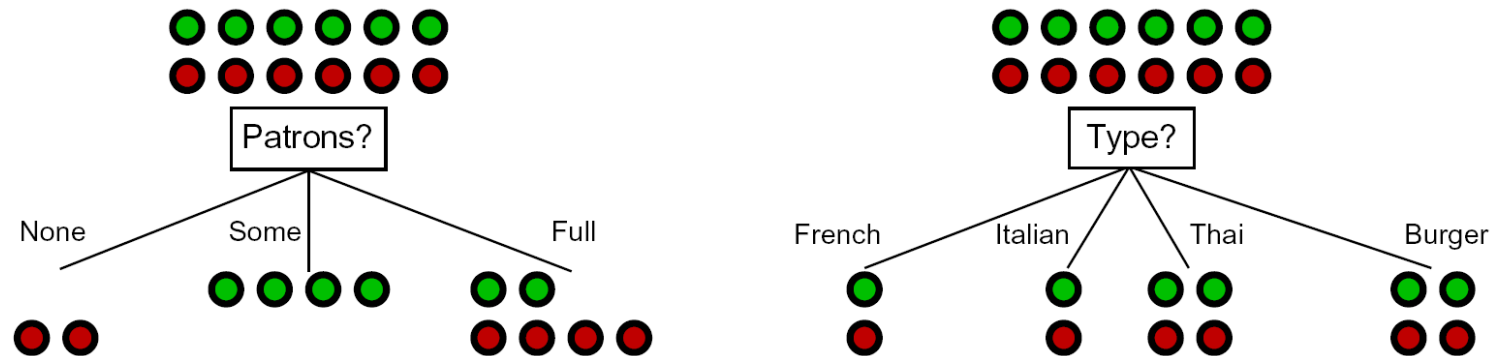
Decision Tree Learning

- 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 ← CHOOSE-ATTRIBUTE(attributes, examples)
    tree ← a new decision tree with root test best
    for each value  $v_i$  of best do
      examplesi ← {elements of examples with best =  $v_i$ }
      subtree ← DTL(examplesi, 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

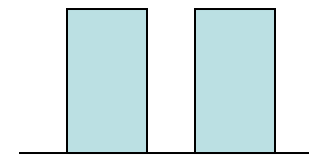
- **Information** answers questions
 - The more uncertain about the answer initially, the more information in the answer
 - Scale: bits
 - Answer to Boolean question with prior $\langle 1/2, 1/2 \rangle$?
 - Answer to 4-way question with prior $\langle 1/4, 1/4, 1/4, 1/4 \rangle$?
 - Answer to 4-way question with prior $\langle 0, 0, 0, 1 \rangle$?
 - Answer to 3-way question with prior $\langle 1/2, 1/4, 1/4 \rangle$?
- A probability p is typical of:
 - A uniform distribution of size $1/p$
 - A code of length $\log 1/p$

Entropy

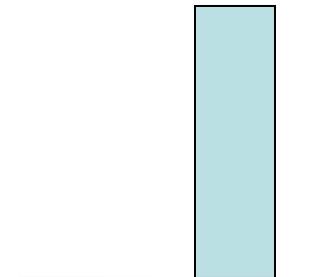
- General answer: if prior is $\langle p_1, \dots, p_n \rangle$:
 - Information is the expected code length

$$\begin{aligned} H(\langle p_1, \dots, p_n \rangle) &= E_p \log_2 1/p_i \\ &= \sum_{i=1}^n -p_i \log_2 p_i \end{aligned}$$

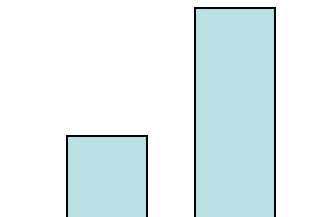
- Also called the **entropy** of the distribution
 - More uniform = higher entropy
 - More values = higher entropy
 - More peaked = lower entropy
 - Rare values almost “don’t count”



1 bit



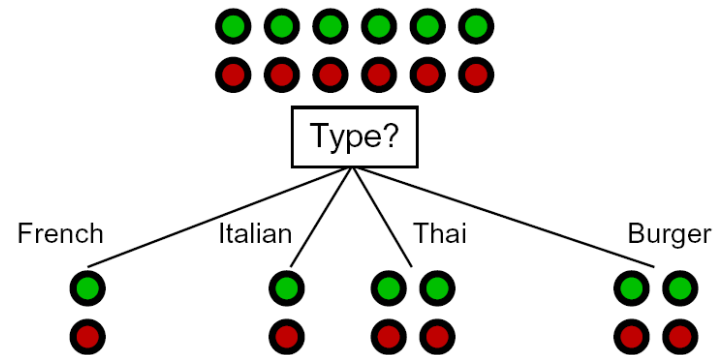
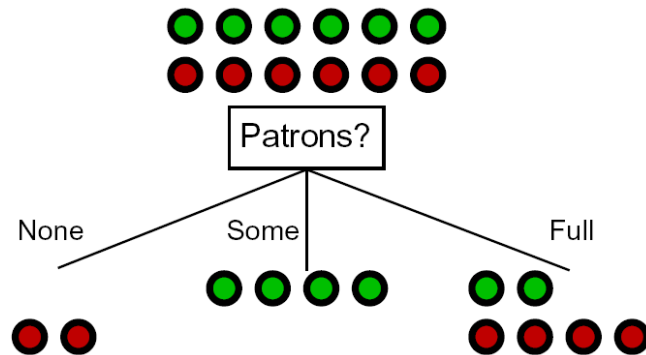
0 bits



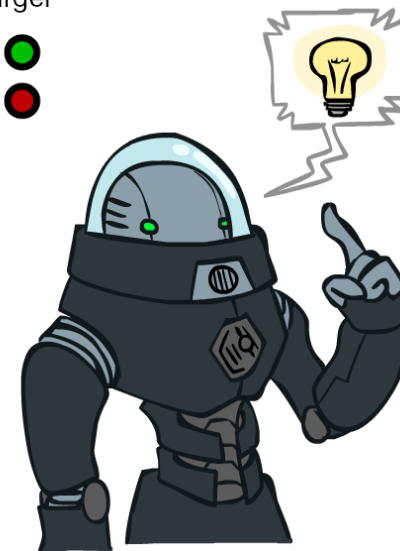
0.5 bit

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!

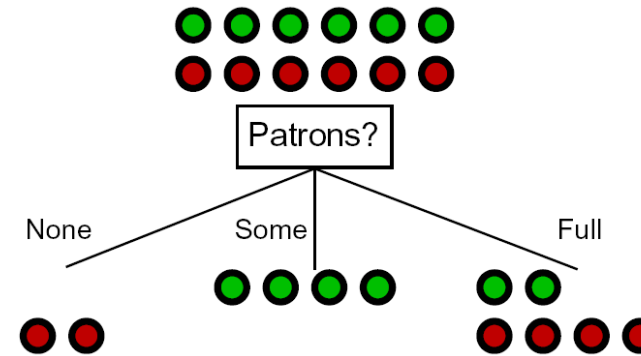


- Solution: use **expected entropy**, weighted by the number of examples



Next Step: Recurse

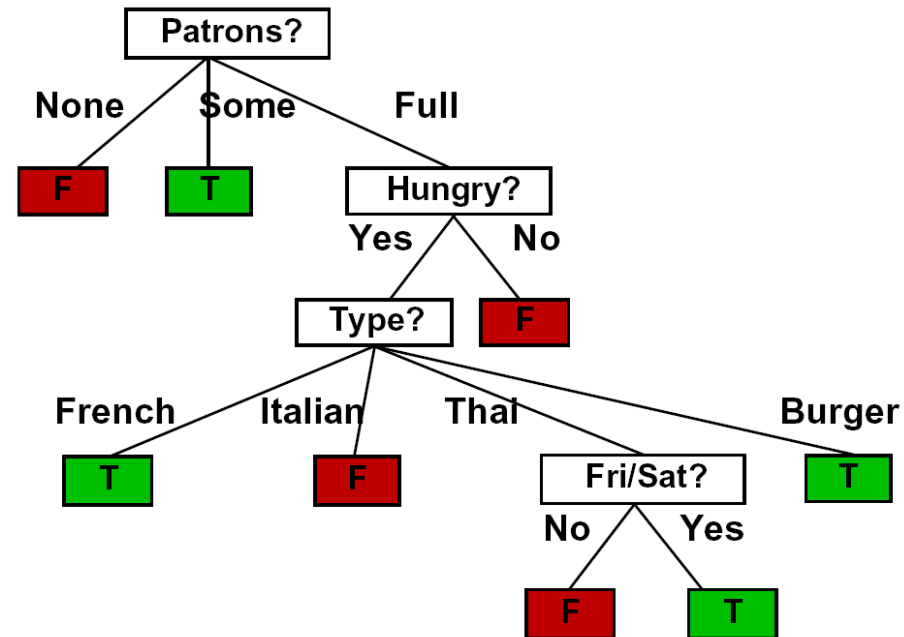
- Now we need to keep growing the tree!
- Two branches are done (why?)
- What to do under “full”?
 - See what examples are there...



Example	Attributes											Target
	<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	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>0-10</i>		<i>T</i>
X_2	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>30-60</i>		<i>F</i>
X_3	<i>F</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>Some</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Burger</i>	<i>0-10</i>		<i>T</i>
X_4	<i>T</i>	<i>F</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>10-30</i>		<i>T</i>
X_5	<i>T</i>	<i>F</i>	<i>T</i>	<i>F</i>	<i>Full</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>>60</i>		<i>F</i>
X_6	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$</i>	<i>T</i>	<i>T</i>	<i>Italian</i>	<i>0-10</i>		<i>T</i>
X_7	<i>F</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>None</i>	<i>\$</i>	<i>T</i>	<i>F</i>	<i>Burger</i>	<i>0-10</i>		<i>F</i>
X_8	<i>F</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$</i>	<i>T</i>	<i>T</i>	<i>Thai</i>	<i>0-10</i>		<i>T</i>
X_9	<i>F</i>	<i>T</i>	<i>T</i>	<i>F</i>	<i>Full</i>	<i>\$</i>	<i>T</i>	<i>F</i>	<i>Burger</i>	<i>>60</i>		<i>F</i>
X_{10}	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>Italian</i>	<i>10-30</i>		<i>F</i>
X_{11}	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>None</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>0-10</i>		<i>F</i>
X_{12}	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Burger</i>	<i>30-60</i>		<i>T</i>

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

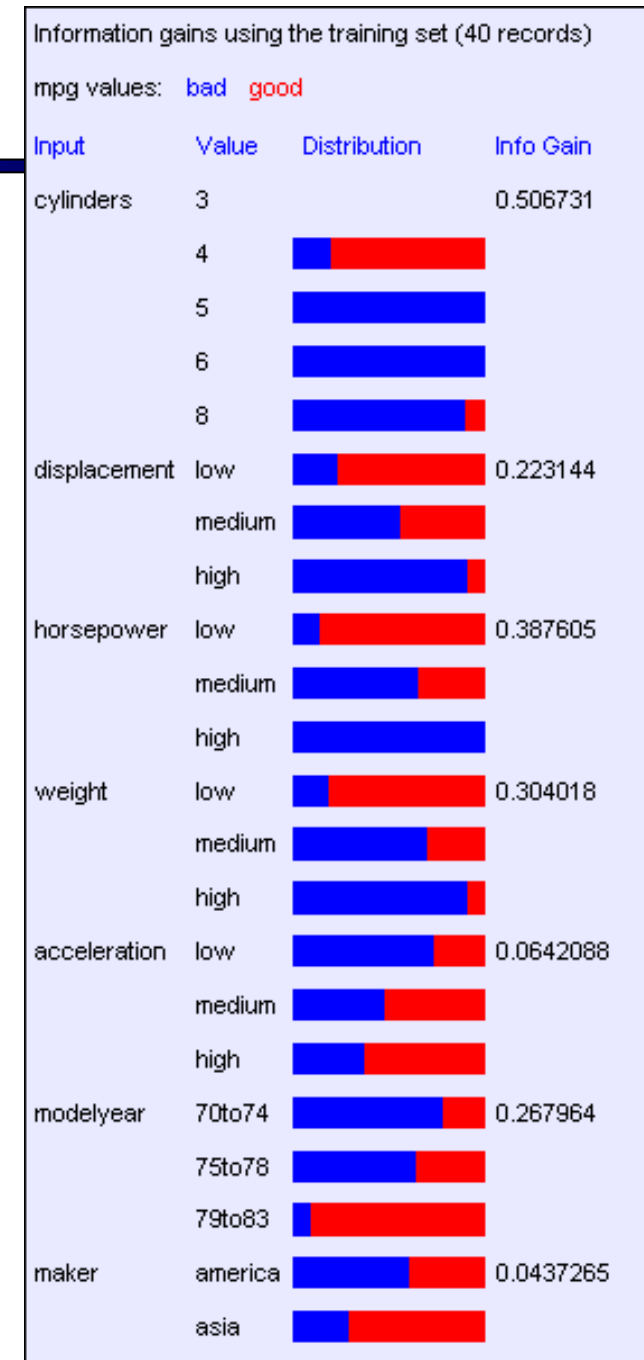
Example: Miles Per Gallon

40 Examples

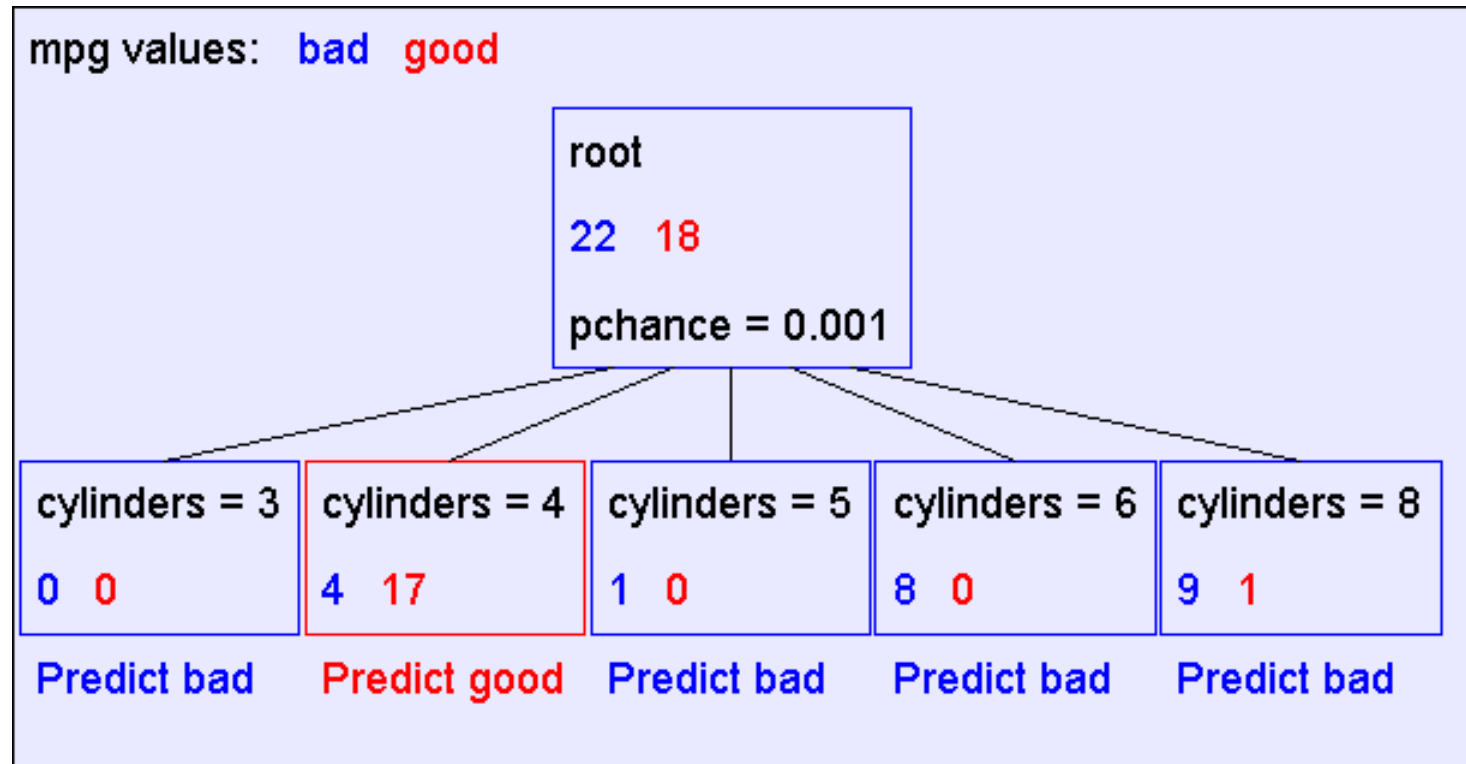
mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	75to78	asia
bad	6	medium	medium	medium	medium	70to74	america
bad	4	medium	medium	medium	low	75to78	europa
bad	8	high	high	high	low	70to74	america
bad	6	medium	medium	medium	medium	70to74	america
bad	4	low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

Find the First Split

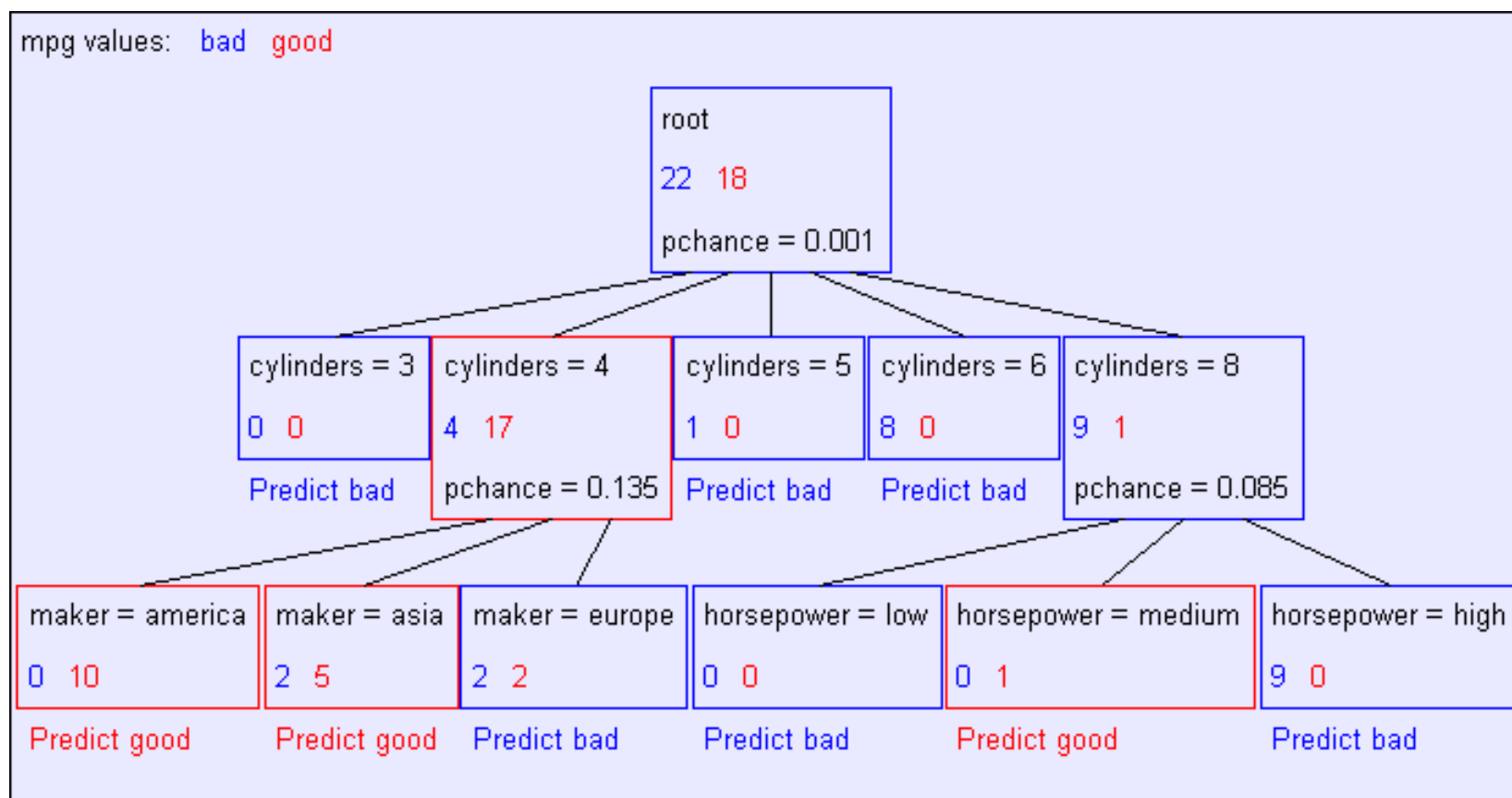
- Look at information gain for each attribute
- Note that each attribute is correlated with the target!
- What do we split on?



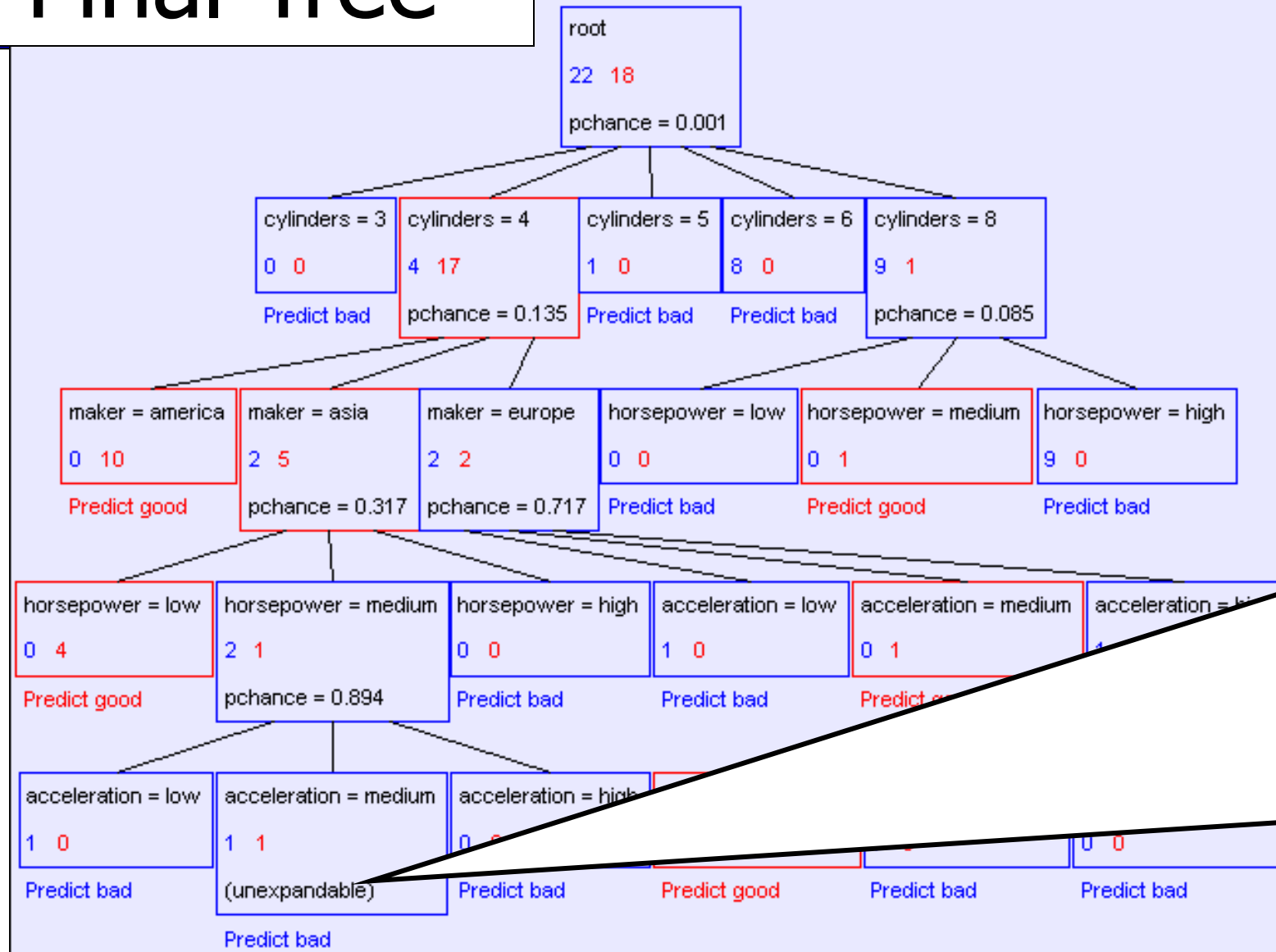
Result: Decision Stump



Second Level



Final Tree



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		

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

MPG Training Error

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

Pr

ad

1

Predict bad

(unexpandable)

Predict bad

Predict good

Predict bad

Predict bad

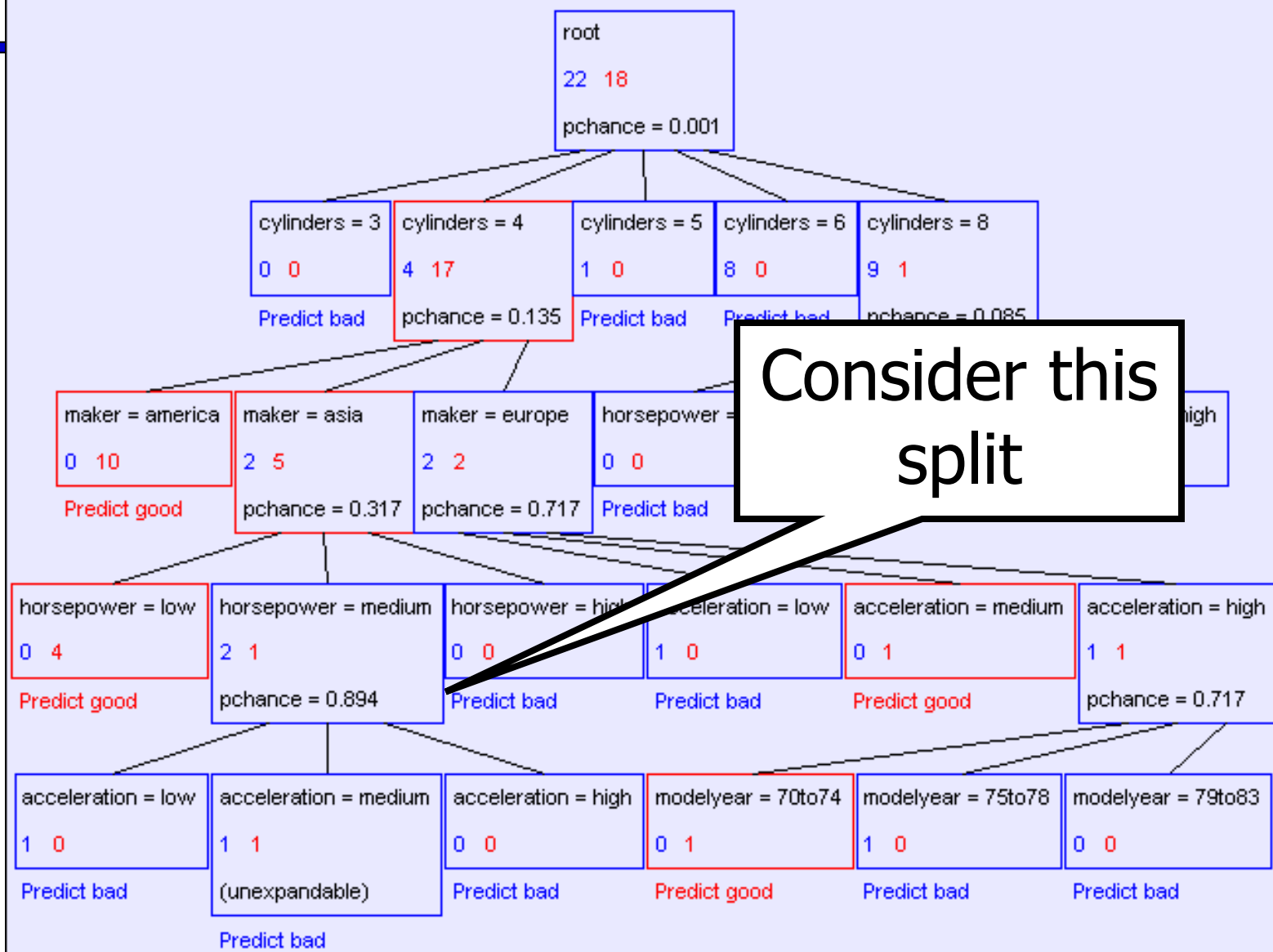
= 0.717

= 79to83

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

...why?

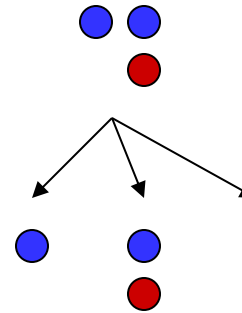
mpg values: bad good



Consider this
split

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:

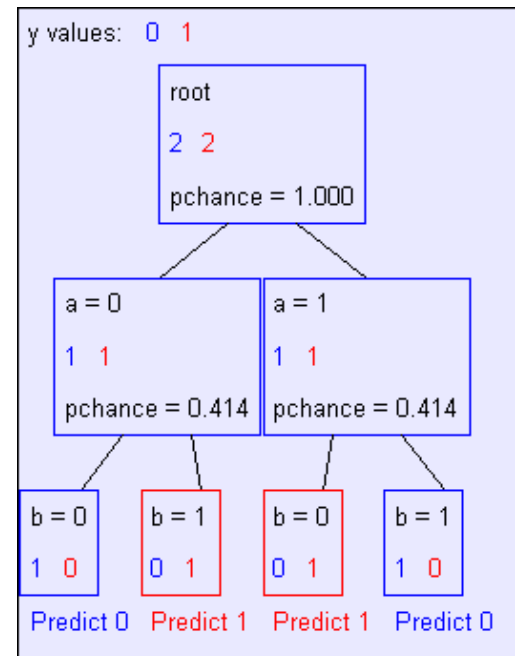
- Build the full decision tree
- Begin at the bottom of the tree
- Delete splits in which

$$p_{\text{CHANCE}} > \text{Max} p_{\text{CHANCE}}$$

- Continue working upward until there are no more prunable nodes
- Note: some chance nodes may not get pruned because they were “redeemed” later

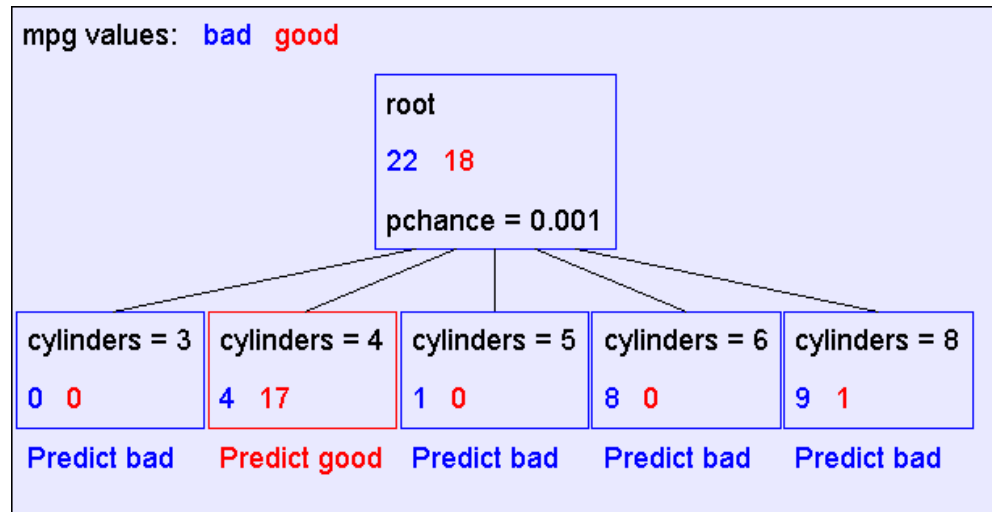
$$y = a \text{ XOR } b$$

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



Pruning example

- With $\text{MaxP}_{\text{CHANCE}} = 0.1$:

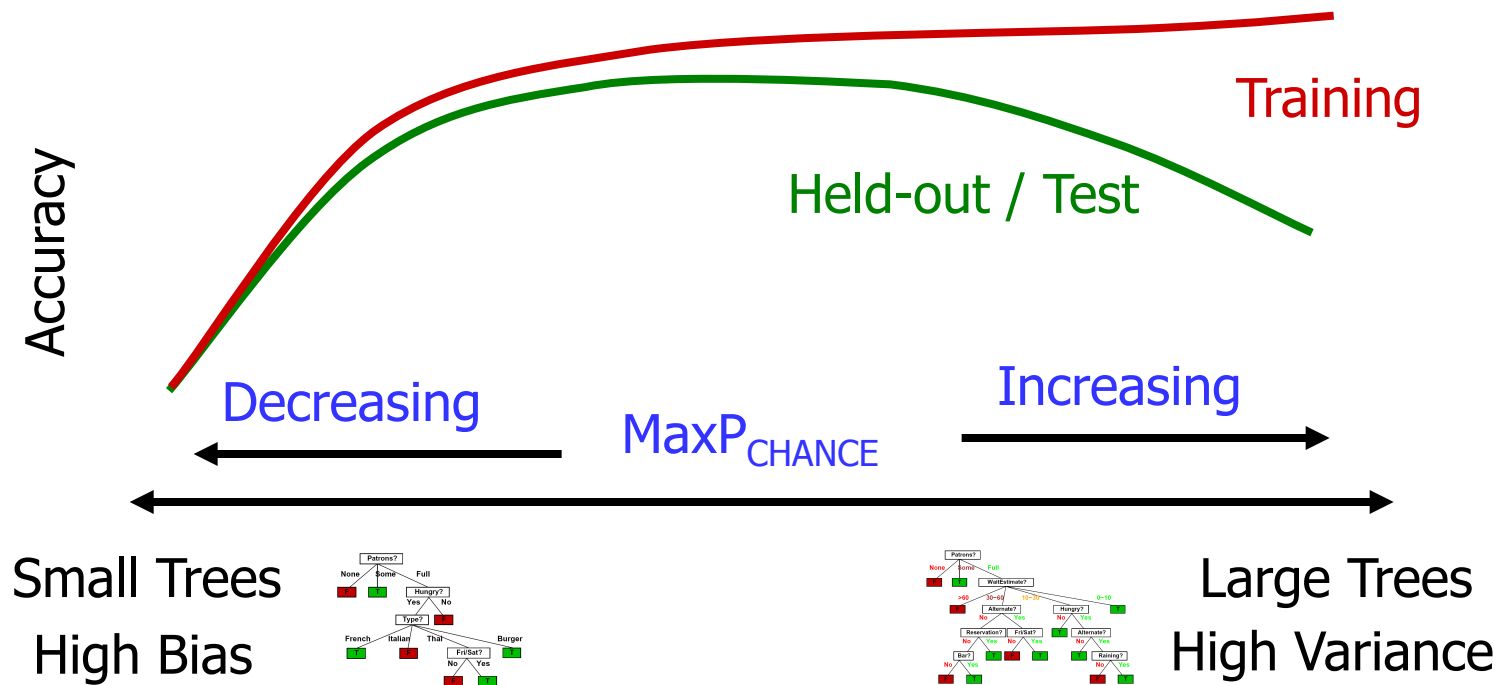


Note the improved test set accuracy compared with the unpruned tree

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

Regularization

- $\text{MaxP}_{\text{CHANCE}}$ is a regularization parameter
- Generally, set it using held-out data (as usual)



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
 - Disprefer most of the hypotheses unless data is clear
 - Usually done in practice