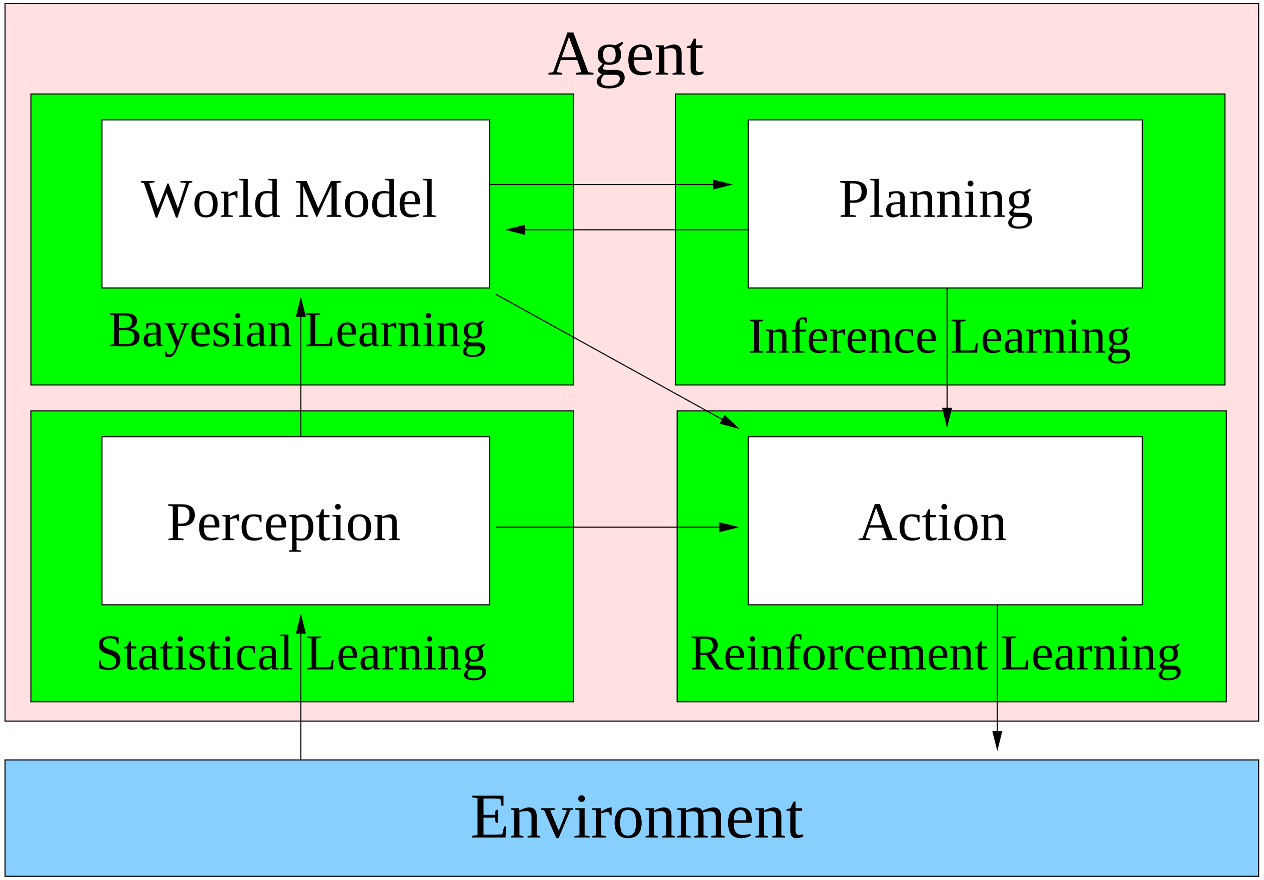
COMP9414/9814/3411: Artificial Intelligence

Week 6: Learning and Decision Trees

Russell & Norvig: 18.1, 18.2, 18.3

Learning Agents



Types of Learning

- Supervised Learning
  - agent is presented with examples of inputs and their target outputs
- Reinforcement Learning
  - agent is not presented with target outputs, but is given a reward signal, which it aims to maximize
- Unsupervised Learning
  - agent is only presented with the inputs themselves, and aims to find structure in these inputs

Supervised Learning

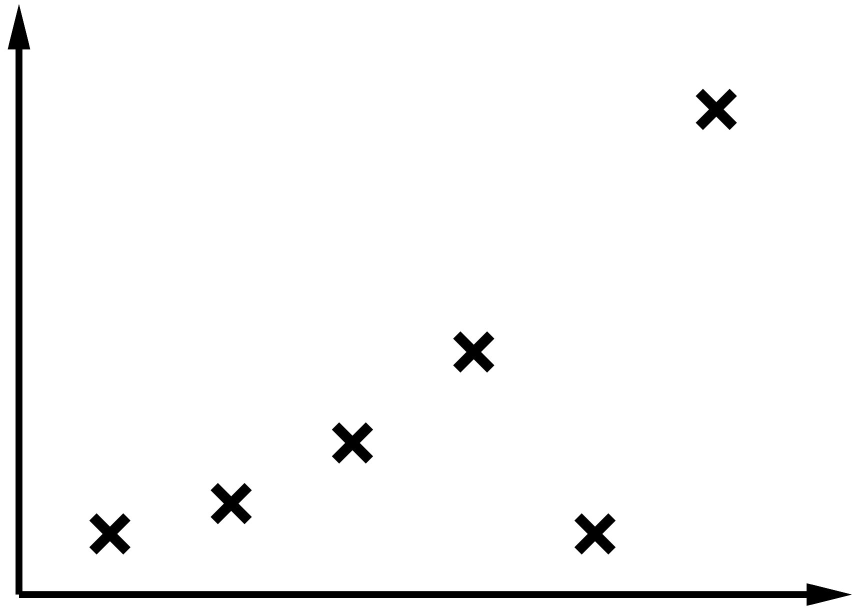
- we have a training set and a test set, each consisting of a set of items; for each item, a number of input attributes and a target value are specified.
- the aim is to predict the target value, based on the input attributes.
- agent is presented with the input and target output for each item in the training set; it must then predict the output for each item in the test set
- various learning paradigms are available:
  - Decision Tree
  - Neural Network
  - Support Vector Machine, etc.

Supervised Learning – Issues

- framework (decision tree, neural network, SVM, etc.)
- representation (of inputs and outputs)
- pre-processing / post-processing
- training method (perceptron learning, backpropagation, etc.)
- generalization (avoid over-fitting)
- evaluation (separate training and testing sets)

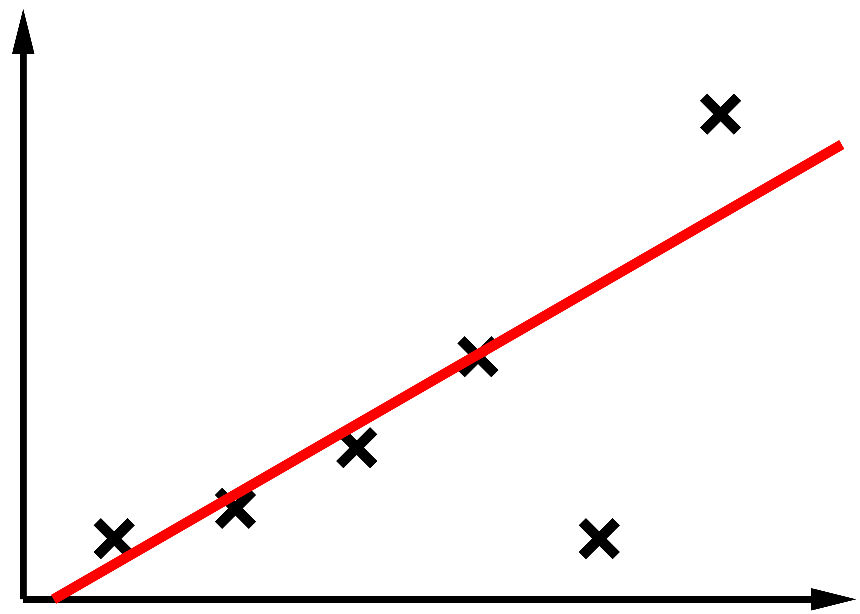
Curve Fitting

Which curve gives the “best fit” to these data?



Curve Fitting

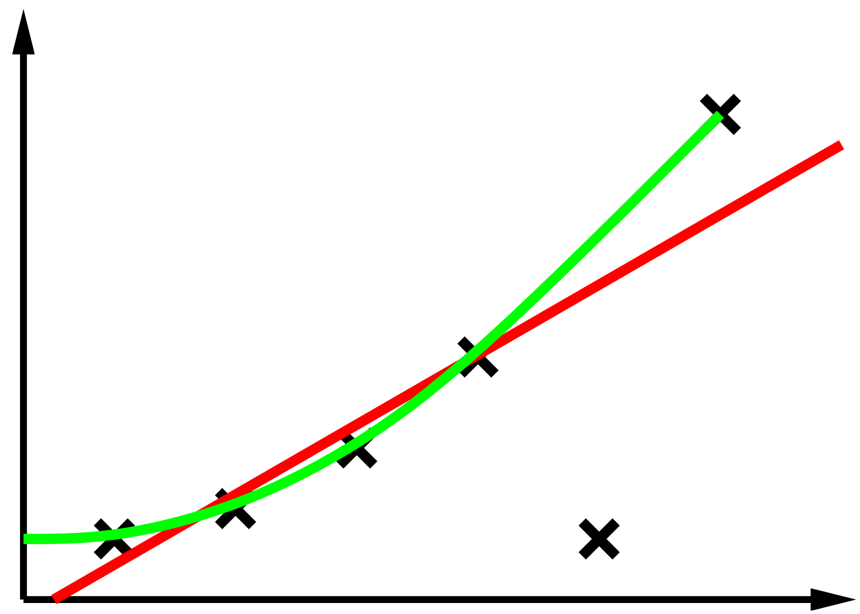
Which curve gives the “best fit” to these data?



straight line?

Curve Fitting

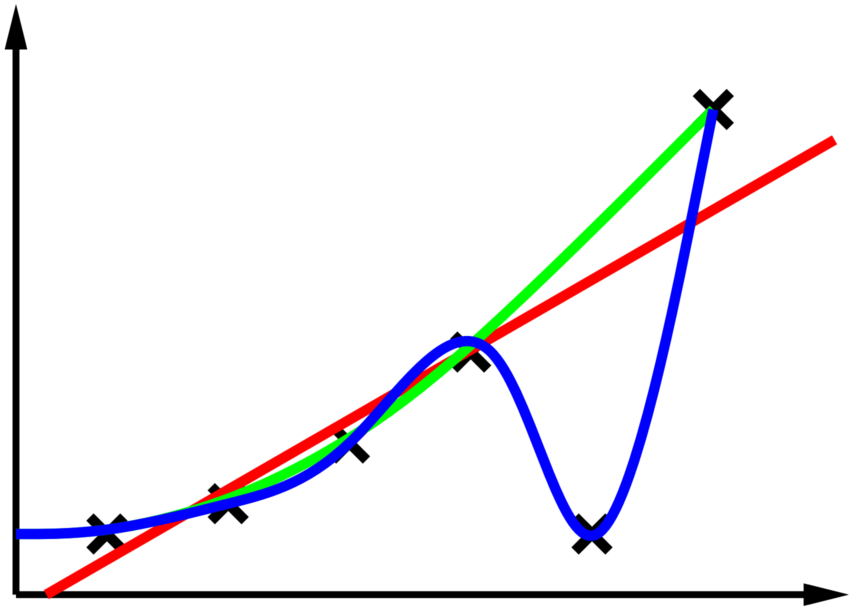
Which curve gives the “best fit” to these data?



parabola?

Curve Fitting

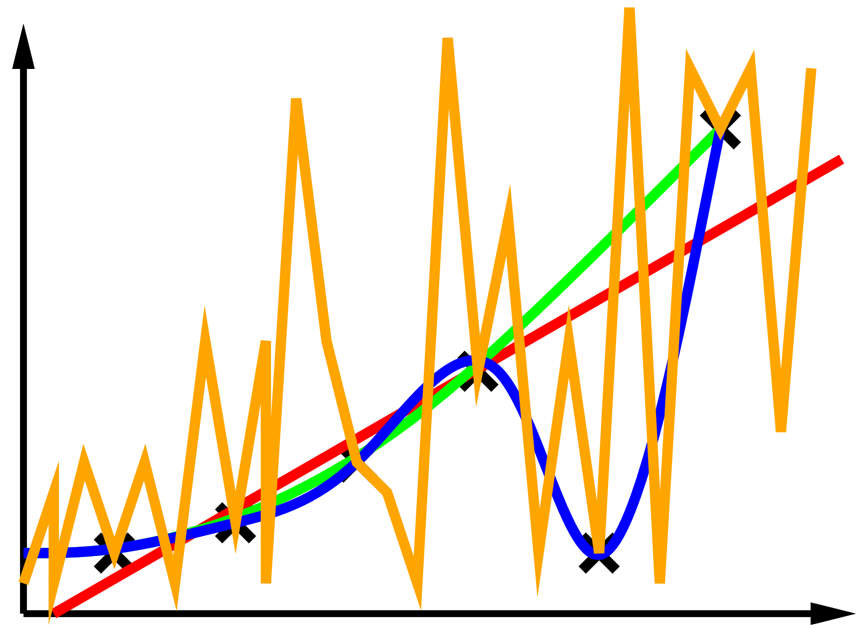
Which curve gives the “best fit” to these data?



4th order polynomial?

Curve Fitting

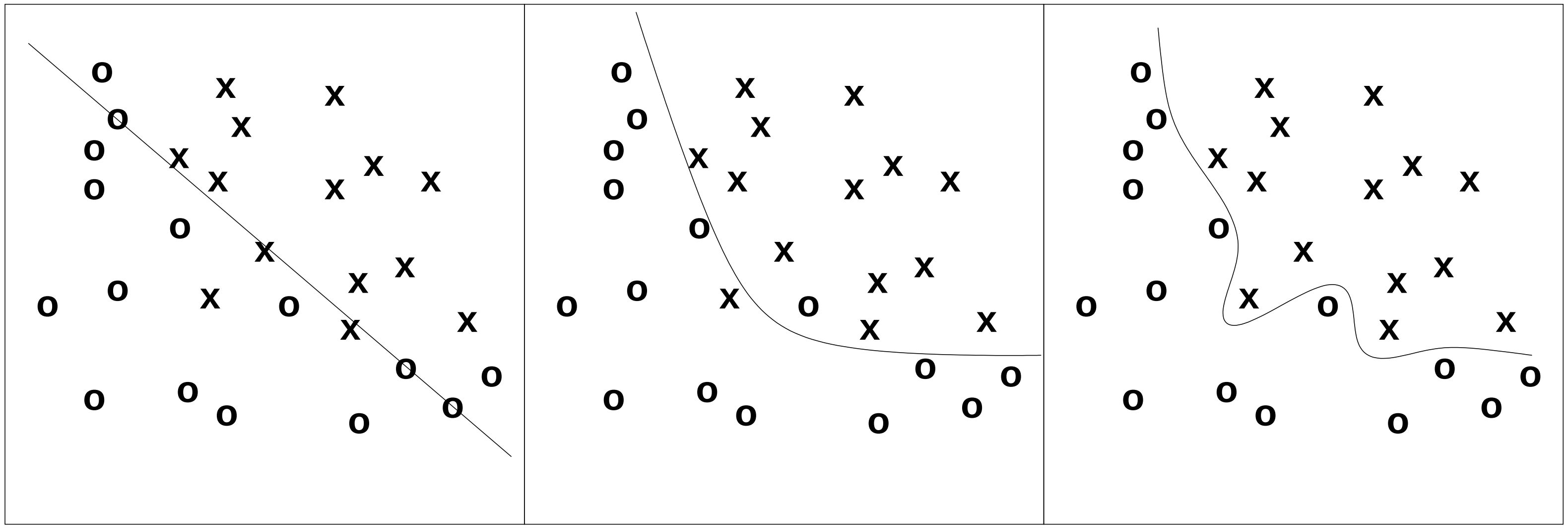
Which curve gives the "best fit" to these data?



Something else?

Ockham's Razor

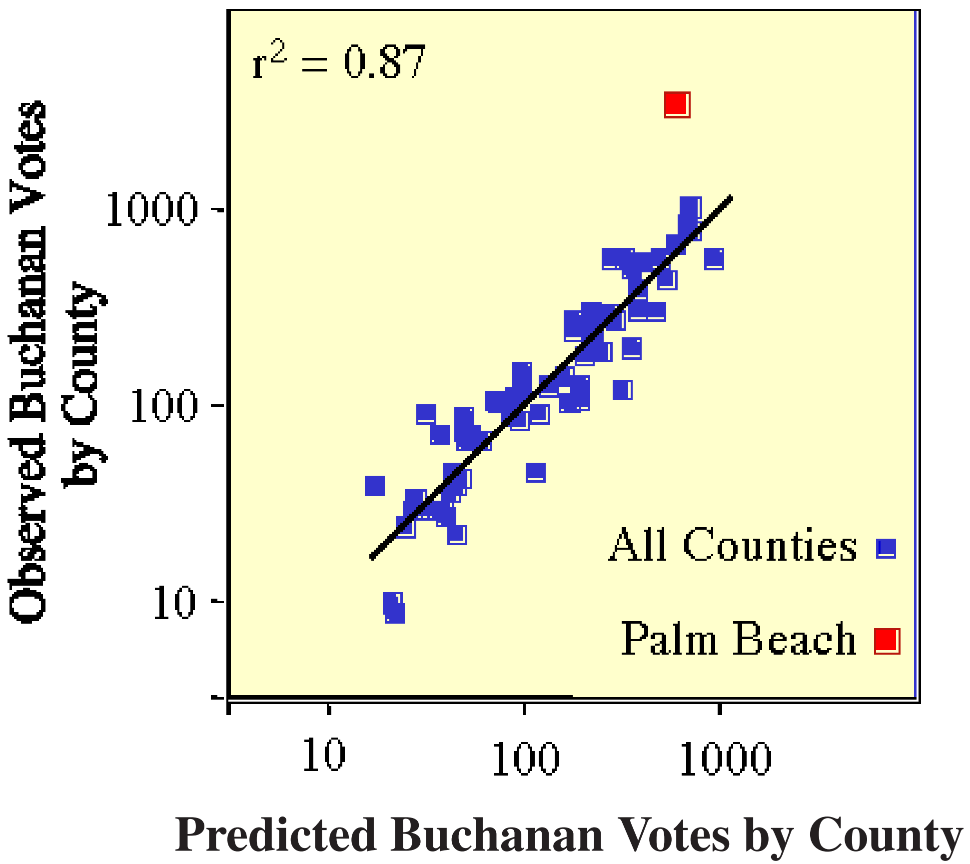
"The most likely hypothesis is the simplest one consistent with the data."



inadequate good compromise over-fitting

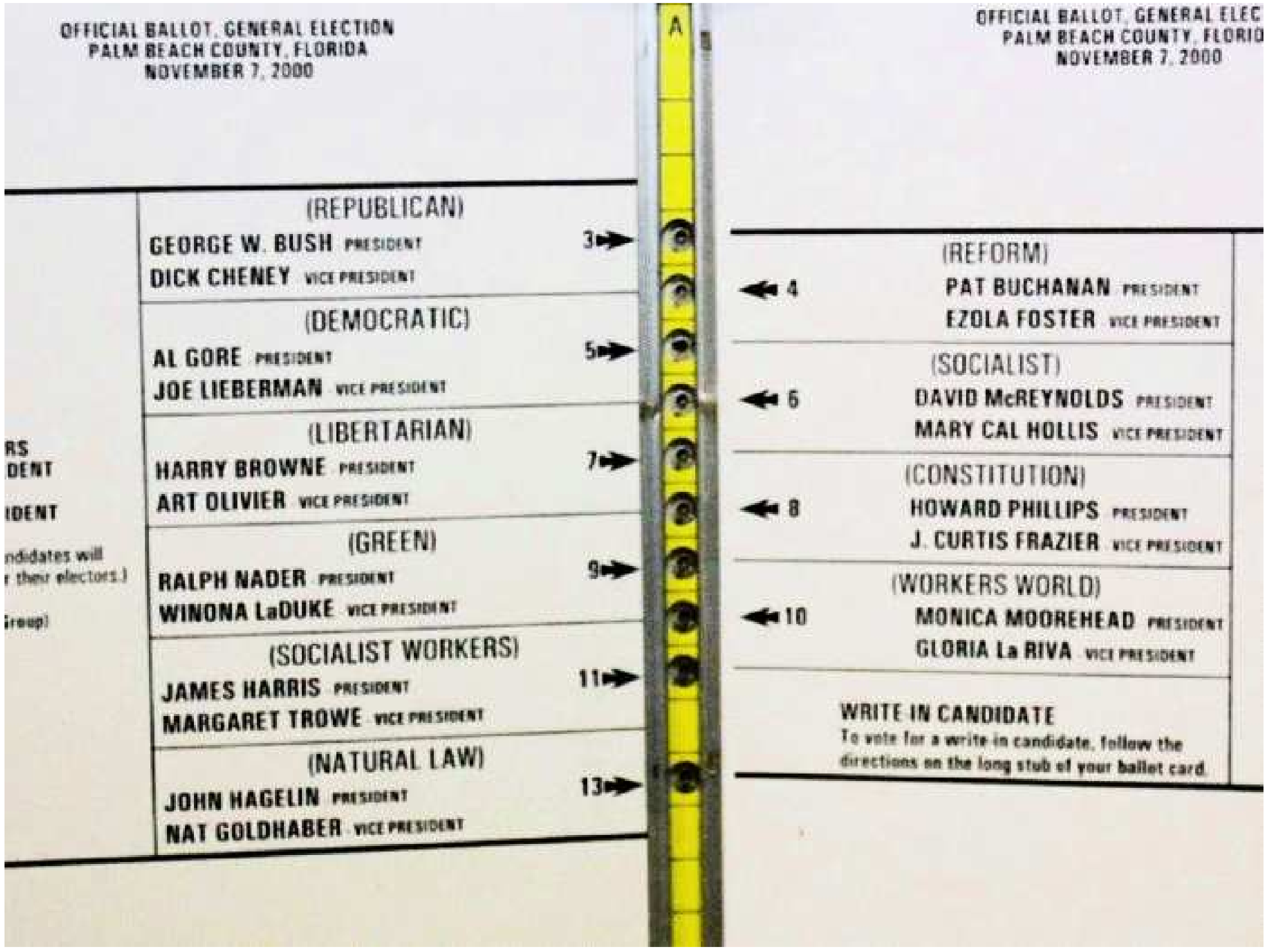
Since there can be noise in the measurements, in practice need to make a tradeoff between simplicity of the hypothesis and how well it fits the data.

Outliers



[faculty.washington.edu/mtbrett]

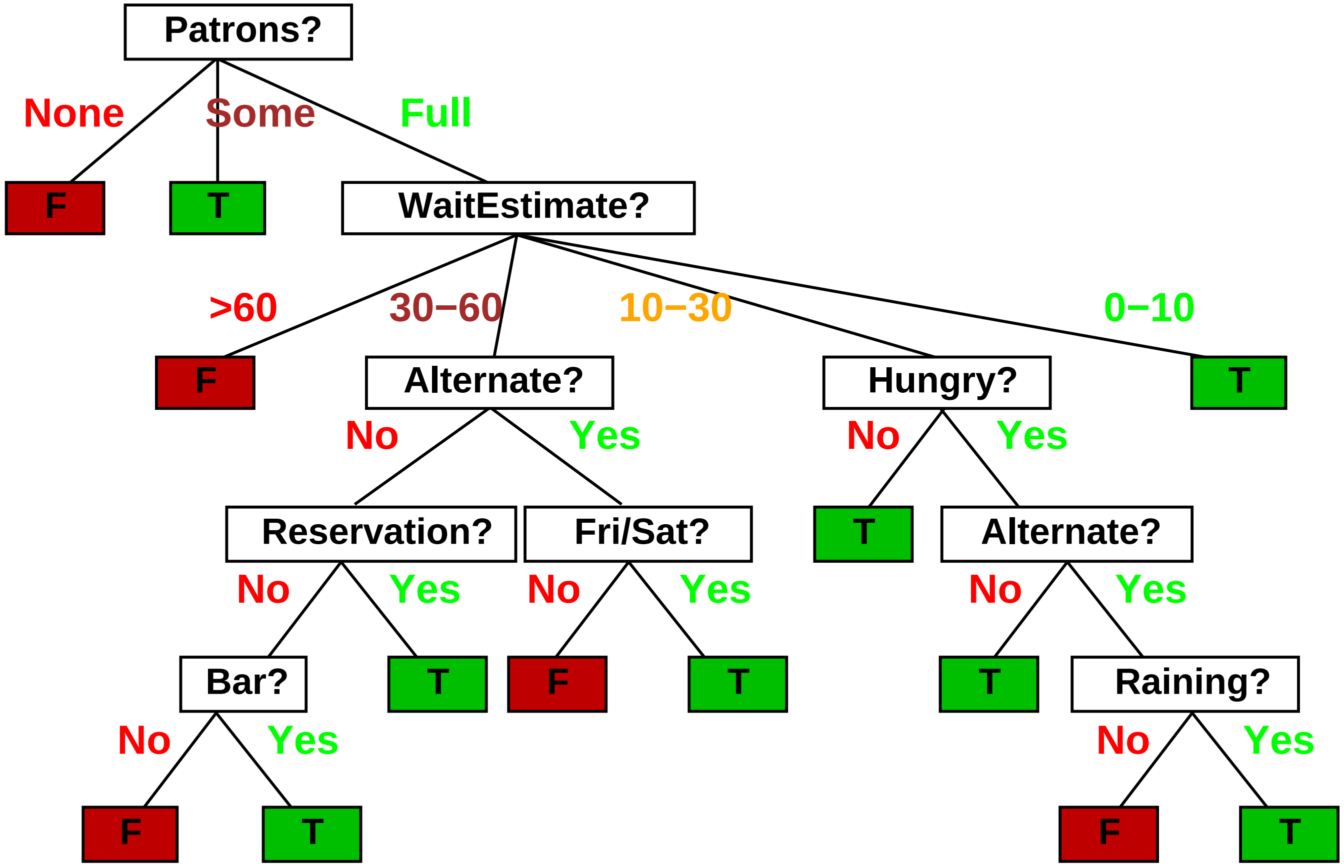
Butterfly Ballot



Restaurant Training Data

	Alt	Bar	F/S	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

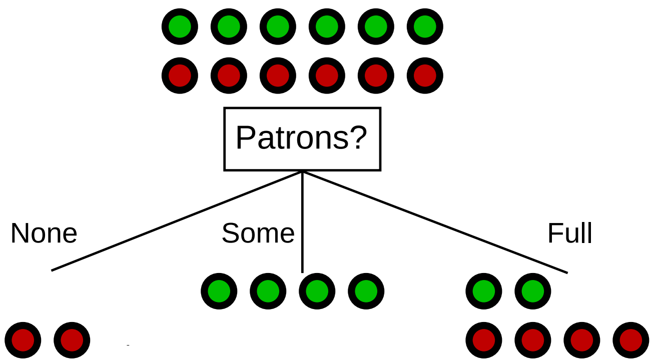
Decision Tree



Generalization

- Provided the training data are not inconsistent, we can split the attributes in any order and still produce a tree that correctly classifies all examples in the training set.
- However, we really want a tree which is likely to generalize to correctly classify the (unseen) examples in the test set.
- In view of Ochkam's Razor, we prefer a simpler hypothesis, i.e. a smaller tree.
- But how can we choose attributes in order to produce a small tree?

Choosing an Attribute



Patrons is a "more informative" attribute than Type, because it splits the examples more nearly into sets that are "all positive" or "all negative".

This notion of "informativeness" can be quantified using the mathematical concept of "entropy".

A parsimonious tree can be built by minimizing the entropy at each step.

Entropy

Entropy is a measure of how much information we gain when the target attribute is revealed to us. In other words, it is not a measure of how much we know, but of how much we don't know.

If the prior probabilities of the n target attribute values are $p_1, \ldots, p_n$ then the entropy is

$$H(\langle p_1, \ldots, p_n \rangle) = \sum_{i=1}^{n} -p_i \log_2 p_i$$

Entropy and Huffmann Coding

Entropy is the number of bits per symbol achieved by a (block) Huffman Coding scheme.

Example 1: $H(\langle 0.5, 0.5 \rangle) = 1$ bit.

Suppose we want to encode, in zeroes and ones, a long message composed of the two letters `A` and `B`, which occur with equal frequency. This can be done efficiently by assigning `A=0`, `B=1`. In other words, one bit is needed to encode each letter.

Entropy and Huffmann Coding

Example 2: $H(\langle 0.5, 0.25, 0.25 \rangle) = 1.5$ bits.

Suppose we need to encode a message consisting of the letters `A`, `B` and `C`, and that `B` and `C` occur equally often but `A` occurs twice as often as the other two letters. In this case, the most efficient code would be `A=0`, `B=10`, `C=11`. The average number of bits needed to encode each letter is 1.5 .

If the letters occur in some other proportion, we would need to "block" them together in order to encode them efficiently. But, the average number of bits required by the most efficient coding scheme is given by

$$H(\langle p_1, \ldots, p_n \rangle) = \sum_{i=1}^{n} -p_i \log_2 p_i$$

Entropy

Suppose we have p positive and n negative examples at a node.
$\rightarrow H(\langle p/(p+n), n/(p+n)\rangle)$ bits needed to classify a new example.
e.g. for 12 restaurant examples, $p = n = 6$ so we need 1 bit.

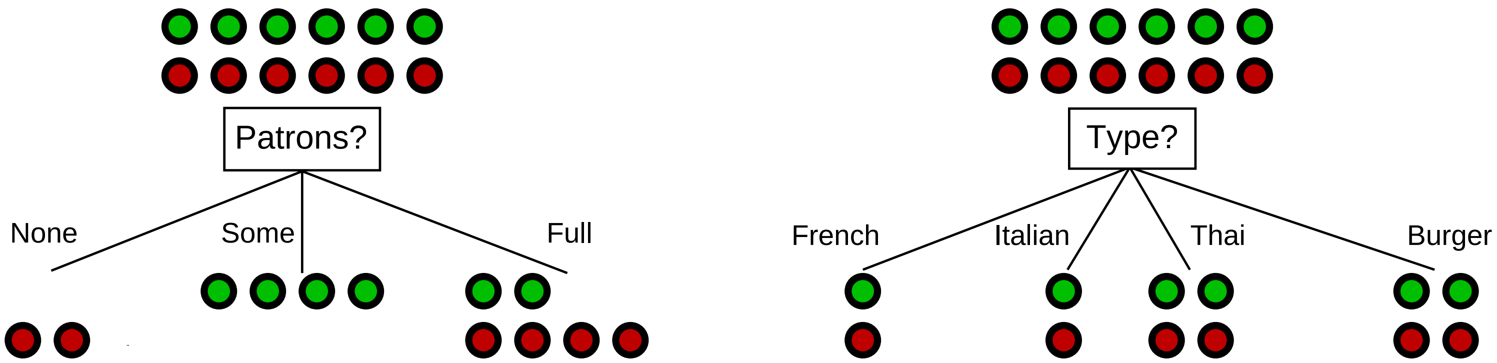
An attribute splits the examples E into subsets E_i, each of which (we hope) needs less information to complete the classification.

Let E_i have p_i positive and n_i negative examples
$\rightarrow H(\langle p_i/(p_i+n_i), n_i/(p_i+n_i)\rangle)$ bits needed to classify a new example
$\rightarrow$ expected number of bits per example over all branches is

$$\sum_i \frac{p_i+n_i}{p+n} H(\langle \frac{p_i}{p_i+n_i}, \frac{n_i}{p_i+n_i}\rangle)$$

For Patrons, this is 0.459 bits, for Type this is (still) 1 bit.

Choosing an Attribute

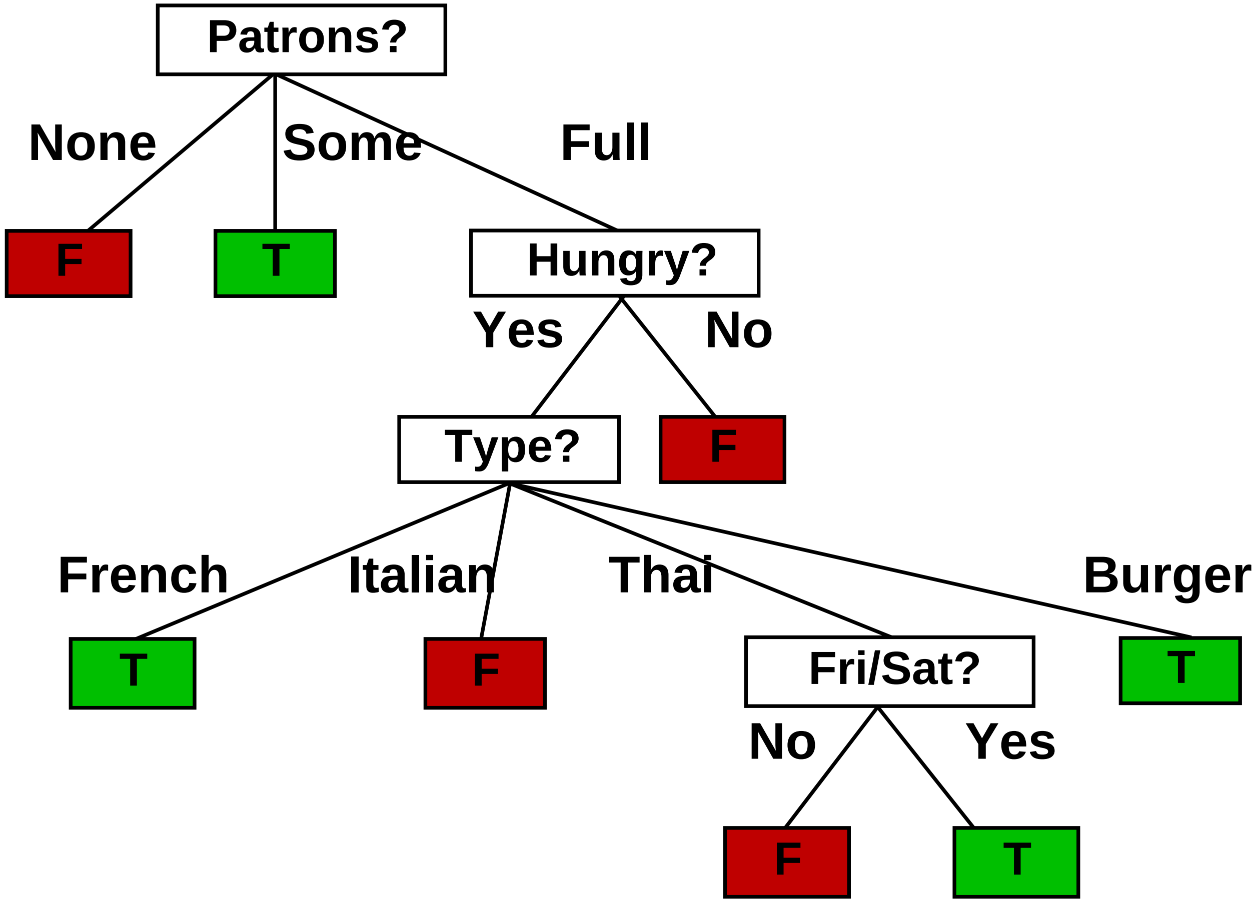


$$\text{For Patrons, Entropy} = \frac{1}{6}(0) + \frac{1}{3}(0) + \frac{1}{2}\left[-\frac{1}{3}\log(\frac{1}{3}) - \frac{2}{3}\log(\frac{2}{3})\right]$$

$$= 0 + 0 + \frac{1}{2}\left[\frac{1}{3}(1.585) + \frac{2}{3}(0.585)\right] = 0.459$$

$$\text{For Type, Entropy} = \frac{1}{6}(1) + \frac{1}{6}(1) + \frac{1}{3}(1) + \frac{1}{3}(1) = 1$$

Induced Tree



Laplace Error and Pruning

According to Ockham's Razor, we may wish to prune off branches that do not provide much benefit in classifying the items.

When a node becomes a leaf, all items will be assigned to the majority class at that node. We can estimate the error rate on the (unseen) test items using the Laplace error:

$$E = 1 - \frac{n+1}{N+k}$$

$$\begin{aligned} N &= \text{total number of (training) items at the node} \\ n &= \text{number of (training) items in the majority class} \\ k &= \text{number of classes} \end{aligned}$$

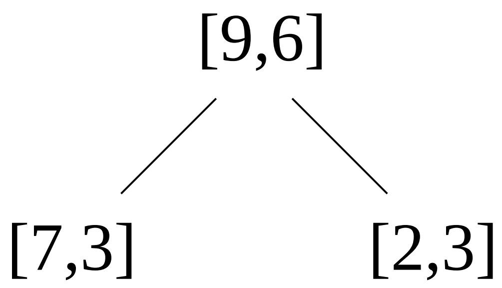
If the average Laplace error of the children exceeds that of the parent node, we prune off the children.

Minimal Error Pruning

Should the children of this node be pruned or not?

Left child has class frequencies [7,3]

$$E = 1 - \frac{n+1}{N+k} = 1 - \frac{7+1}{10+2} = 0.333$$



Right child has $E = 0.429$

Parent node has $E = 0.412$

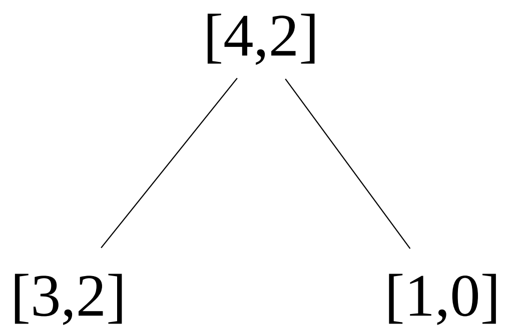
Average for Left and Right child is

$$E = \frac{10}{15}(0.333) + \frac{5}{15}(0.429) = 0.365$$

Since $0.365 < 0.412$, children should NOT be pruned.

Minimal Error Pruning

Should the children of this node be pruned or not?



Left child has class frequencies [3,2]

$$E = 1 - \frac{n+1}{N+k} = 1 - \frac{3+1}{5+2} = 0.429$$

Right child has $E = 0.333$

Parent node has $E = 0.375$

Average for Left and Right child is

$$E = \frac{5}{6}(0.429) + \frac{1}{6}(0.333) = 0.413$$

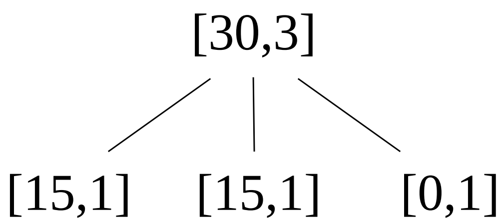
Since $0.413 > 0.375$, children should be pruned.

Minimal Error Pruning

Should the children of this node be pruned or not?

Left and Middle child have class frequencies [15,1]

$$E = 1 - \frac{n+1}{N+k} = 1 - \frac{15+1}{16+2} = 0.111$$



Right child has $E = 0.333$
Parent node has $E = \frac{4}{35} = 0.114$
Average for Left, Middle and Right child is

$$E = \frac{16}{33}(0.111) + \frac{16}{33}(0.111) + \frac{1}{33}(0.333) = 0.118$$

Since $0.118 > 0.114$, children should be pruned.

Summary

- Supervised Learning
  - training set and test set
  - try to predict target value, based on input attributes
- Ockham's Razor
  - tradeoff between simplicity and accuracy
- Decision Trees
  - improve generalisation by building a smaller tree (using entropy)
  - prune nodes based on Laplace error