Artificial Intelligence:
Representation and Problem Solving

15-381

April 10, 2007

# Introduction to Learning & Decision Trees

## Learning and Decision Trees to learning

*aka:*
- *regression*
- *pattern recognition*
- *machine learning*
- *data mining*

- What is learning?
  - more than just memorizing facts
  - learning the underlying *structure* of the problem or data

- A fundamental aspect of learning is *generalization:*
  - given a few examples, can you *generalize* to others?

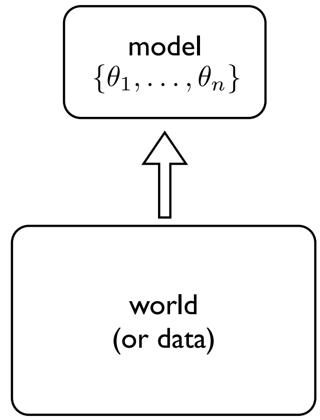
- Learning is ubiquitous:
  - *medical diagnosis*: identify new disorders from observations
  - *loan applications*: predict risk of default
  - *prediction: (*climate, stocks, etc.) predict future from current and past data
  - *speech/object recognition*: from examples, generalize to others

## Representation

- How do we model or represent the world?
- All learning requires some form of representation.

- Learning:

  *adjust model parameters to match data*






## The complexity of learning

- Fundamental trade-off in learning:

  *complexity of model*

  *vs*

  *amount of data required to learn parameters*

- The more complex the model, the more it can describe,

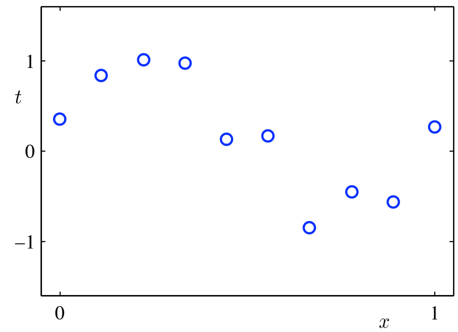
  but the more data it requires to constrain the parameters.

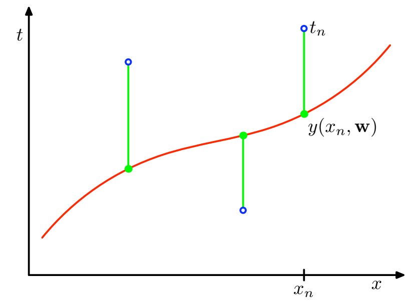
- Consider a hypothesis space of N models:
  - How many bits would it take to identify which of the N models is 'correct'?
  - $\log_2(N)$ in the worst case

- Want simple models to explain examples and generalize to others
  - Ockham's (some say Occam) razor

# Complex learning example: curve fitting

$$t = \sin(2\pi x) + \text{noise}$$




***How do we model the data?***

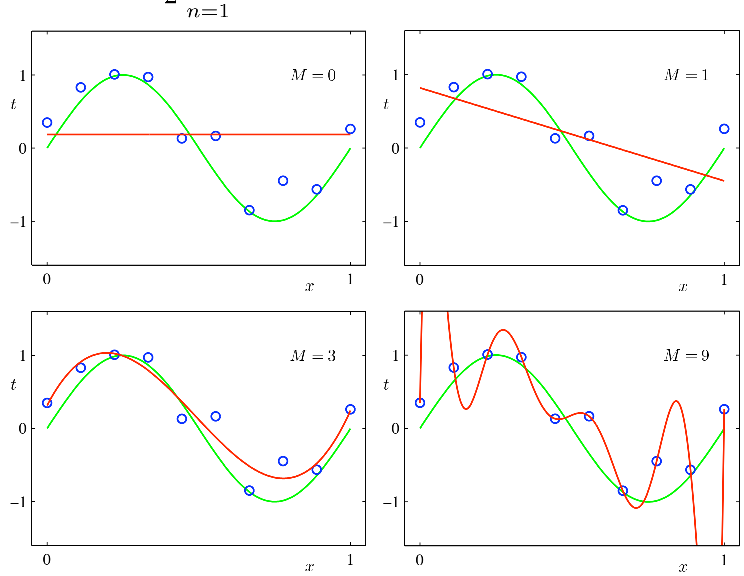
example from Bishop (2006), *Pattern Recognition and Machine Learning*



# Polynomial curve fitting

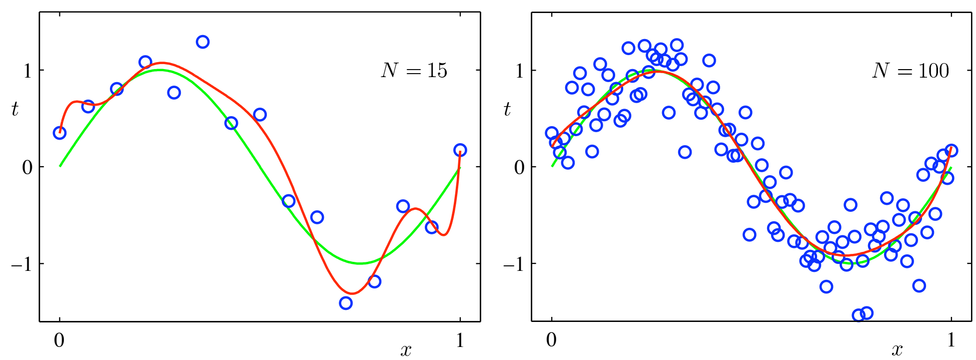
$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} [y(x_n, \mathbf{w}) - t_n]^2$$




example from Bishop (2006), *Pattern Recognition and Machine Learning*

## More data are needed to learn correct model

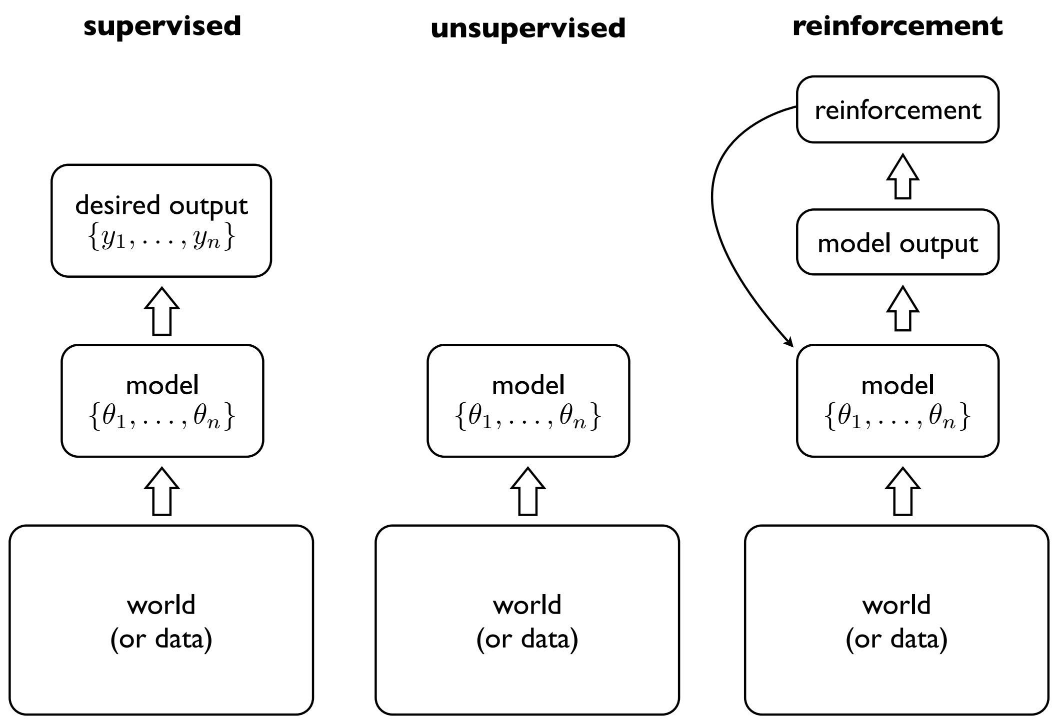



This is *overfitting.*

example from Bishop (2006), *Pattern Recognition and Machine Learning*



## Types of learning

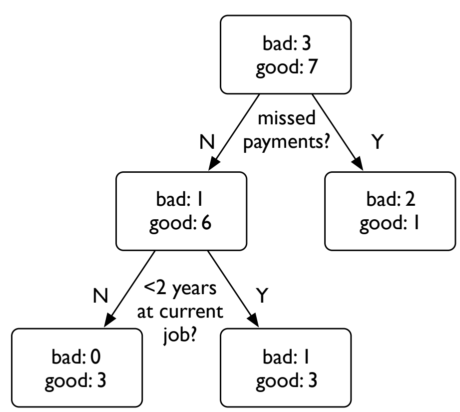


*Decision Trees*

## Decision trees: classifying from a set of attributes

Predicting credit risk

| <2 years at current job? | missed payments? | defaulted? |
|---|---|---|
| N | N | N |
| Y | N | Y |
| N | N | N |
| N | N | N |
| N | Y | Y |
| Y | N | N |
| N | Y | N |
| N | Y | Y |
| Y | N | N |
| Y | N | N |




- each level splits the data according to different attributes

- **goal**: achieve perfect classification with minimal number of decisions
  - not always possible due to noise or inconsistencies in the data

# Observations

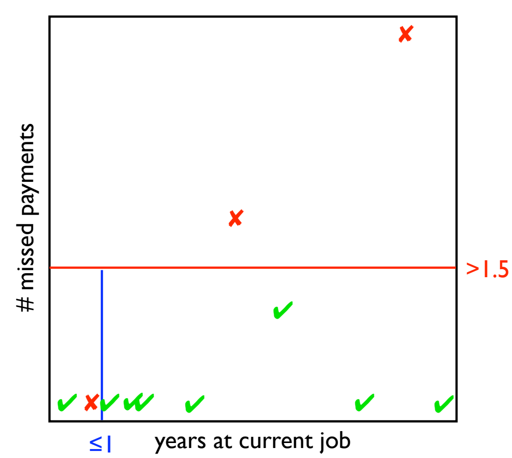
- Any boolean function can be represented by a decision tree.
- not good for all functions, e.g.:
  - parity function: return 1 iff an even number of inputs are 1
  - majority function: return 1 if more than half inputs are 1
- best when a small number of attributes provide a lot of information
- Note: finding optimal tree for arbitrary data is NP-hard.



# Decision trees with continuous values

Predicting credit risk

| years at current job | # missed payments | defaulted? |
|---|---|---|
| 7 | 0 | N |
| 0.75 | 0 | Y |
| 3 | 0 | N |
| 9 | 0 | N |
| 4 | 2 | Y |
| 0.25 | 0 | N |
| 5 | 1 | N |
| 8 | 4 | Y |
| 1.0 | 0 | N |
| 1.75 | 0 | N |




- Now tree corresponds to order and placement of boundaries
- General case:
  - arbitrary number of attributes: binary, multi-valued, or continuous
  - output: binary, multi-valued (*decision or axis-aligned classification trees*), or continuous (*regression trees*)

## Examples

- loan applications
- medical diagnosis
- movie preferences (Netflix contest)
- spam filters
- security screening
- many real-word systems, and AI success

- In each case, we want
  - *accurate* classification, i.e. minimize error
  - *efficient* decision making, i.e. fewest # of decisions/tests

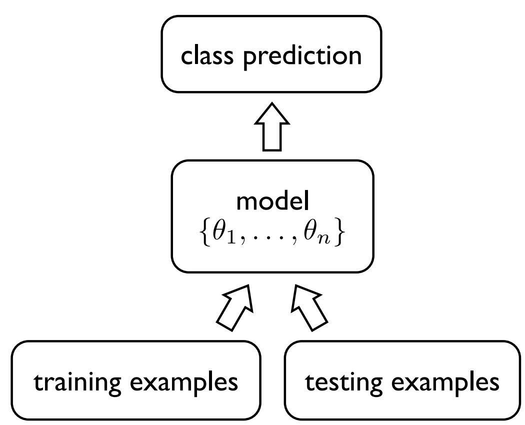
- decision sequence could be further complicated
  - want to minimize false negatives in medical diagnosis or minimize cost of test sequence
  - don't want to miss important email



## Decision Trees

- simple example of inductive learning
  1. *learn* decision tree from training examples
  2. *predict* classes for novel testing examples

- Generalization is how well we do on the testing examples.
- Only works if we can learn the underlying structure of the data.



## Choosing the attributes

- How do we find a decision tree that agrees with the training data?
- Could just choose a tree that has one path to a leaf for each example
  - but this just memorizes the observations (assuming data are consistent)
  - we want it to *generalize* to new examples
- Ideally, best attribute would partition the data into positive and negative examples
- Strategy (greedy):
  - choose attributes that give the best partition first
- Want correct classification with fewest number of tests

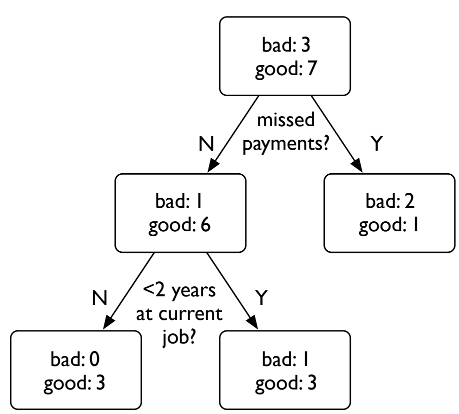


## Problems

- How do we which attribute or value to split on?
- When should we stop splitting?
- What do we do when we can't achieve perfect classification?
- What if tree is too large? Can we approximate with a smaller tree?

Predicting credit risk

| <2 years at current job? | missed payments? | defaulted? |
|---|---|---|
| N | N | N |
| Y | N | Y |
| N | N | N |
| N | N | N |
| N | Y | Y |
| Y | N | N |
| N | Y | N |
| N | Y | Y |
| Y | N | N |
| Y | N | N |



## Basic algorithm for learning decision trees

1. starting with whole training data
2. select attribute or value along dimension that gives "best" split
3. create child nodes based on split
4. recurse on each child using child data until a stopping criterion is reached
   - all examples have same class
   - amount of data is too small
   - tree too large

- Central problem: How do we choose the "best" attribute?



## Measuring information

- A convenient measure to use is based on information theory.
- How much "information" does an attribute give us about the class?
  - attributes that perfectly partition should given maximal information
  - unrelated attributes should give no information

- Information of symbol w:

$$I(w) \equiv -\log_2 P(w)$$

$$\begin{aligned} P(w) &= 1/2 \\ \Rightarrow I(w) &= -\log_2 1/2 = 1 \text{ bit} \\ P(w) &= 1/4 \\ \Rightarrow I(w) &= -\log_2 1/4 = 2 \text{ bits} \end{aligned}$$

## Information and Entropy

$$I(w) \equiv -\log_2 P(w)$$

- For a random variable *X* with probability *P(x)*, the entropy is the average (or expected) amount of information obtained by observing *x*:

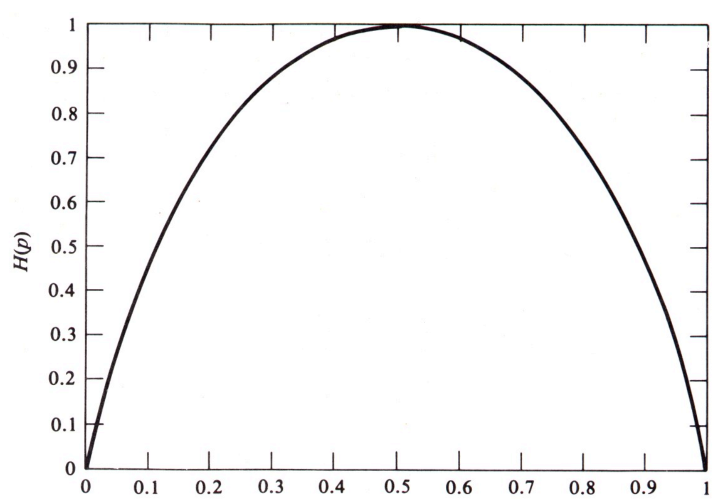
$$H(X) = \sum_x P(x)I(x) = -\sum_x P(x)\log_2 P(x)$$

- Note: H(X) depends only on the probability, not the value.
- H(X) quantifies the uncertainty in the data in terms of bits
- H(X) gives a lower bound on cost (in bits) of coding (or describing) X

$$\begin{aligned} H(X) &= -\sum_x P(x)\log_2 P(x) \\ P(\text{heads}) = 1/2 \quad &\Rightarrow \quad -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = 1 \text{ bit} \\ P(\text{heads}) = 1/3 \quad &\Rightarrow \quad -\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3} = 0.9183 \text{ bits} \end{aligned}$$



## Entropy of a binary random variable




- Entropy is maximum at p=0.5
- Entropy is zero and p=0 or p=1.

## English character strings revisited: A-Z and space

$H_1$ = 4.76 bits/char

1. *Zero-order approximation.* (The symbols are independent and equiprobable.)

   XFOML RXKHRJFFJUJ ZLPWCFWKCYJ

   FFJEYVKCQSGXYD QPAAMKBZAACIBZLHJQD

$H_2$ = 4.03 bits/char

2. *First-order approximation.* (The symbols are independent. Frequency of letters matches English text.)

   OCRO HLI RGWR NMIELWIS EU LL NBNESEBYA TH EEI

   ALHENHTTPA OOBTTVA NAH BRL

   •
   •
   •

$H_2$ = 2.8 bits/char

5. *Fourth-order approximation.* (The frequency of quadruplets of letters matches English text. Each letter depends on the previous three letters. This sentence is from Lucky's book, *Silicon Dreams* [183].)

   THE GENERATED JOB PROVIDUAL BETTER TRAND THE

   DISPLAYED CODE, ABOVERY UPONDULTS WELL THE

   CODERST IN THESTICAL IT DO HOCK BOTHE MERG.

The entropy *increases* as the data become less ordered.



## Credit risk revisited

- How many bits does it take to specify the attribute of 'defaulted?'
  - P(defaulted = Y) = 3/10
  - P(defaulted = N) = 7/10

$$
\begin{aligned}
H(Y) &= -\sum_{i=\mathrm{Y,N}} P(Y=y_i)\log_2 P(Y=y_i) \\
&= -0.3\log_2 0.3 - 0.7\log_2 0.7 \\
&= 0.8813
\end{aligned}
$$

- How much can we *reduce* the entropy (or uncertainty) of 'defaulted' by knowing the other attributes?
- Ideally, we could reduce it to zero, in which case we classify perfectly.

Predicting credit risk

| <2 years at current job? | missed payments? | defaulted? |
|---|---|---|
| N | N | N |
| Y | N | Y |
| N | N | N |
| N | N | N |
| N | Y | Y |
| Y | N | N |
| N | Y | N |
| N | Y | Y |
| Y | N | N |
| Y | N | N |

## Conditional entropy

- H(Y|X) is the remaining entropy of Y given X

  *or*

  The expected (or average) entropy of P(y|x)

$$
\begin{aligned}
H(Y|X) &\equiv -\sum_x P(x) \sum_y P(y|x) \log_2 P(y|x) \\
&= -\sum_x P(x) \sum_y P(Y=y|X=x) \log_2 P(Y=y|X=x) \\
&= -\sum_x P(x) \sum_y H(Y|X=x)
\end{aligned}
$$

- H(Y|X=x) is the *specific conditional entropy*, i.e. the entropy of Y knowing the value of a specific attribute x.



## Back to the credit risk example

$$
\begin{aligned}
H(Y|X) &\equiv -\sum_x P(x) \sum_y P(y|x) \log_2 P(y|x) \\
&= -\sum_x P(x) \sum_y P(Y=y|X=x) \log_2 P(Y=y|X=x) \\
&= -\sum_x P(x) \sum_y H(Y|X=x)
\end{aligned}
$$

$$
\begin{aligned}
H(\text{defaulted}|< 2\text{years} = \text{N}) &= -\frac{4}{4+2}\log_2\frac{4}{4+2} - \frac{2}{6}\log_2\frac{2}{6} = 0.9183 \\
H(\text{defaulted}|< 2\text{years} = \text{Y}) &= -\frac{3}{4}\log_2\frac{3}{4} - \frac{1}{4}\log_2\frac{1}{4} = 0.8133 \\
H(\text{defaulted}|\text{missed}) &= \frac{6}{10}0.9183 + \frac{4}{10}0.8133 = 0.8763
\end{aligned}
$$

$$
\begin{aligned}
H(\text{defaulted}|\text{missed} = \text{N}) &= -\frac{6}{7}\log_2\frac{6}{7} - \frac{1}{7}\log_2\frac{1}{7} = 0.5917 \\
H(\text{defaulted}|\text{missed} = \text{Y}) &= -\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3} = 0.9183 \\
H(\text{defaulted}|\text{missed}) &= \frac{7}{10}0.5917 + \frac{3}{10}0.9183 = 0.6897
\end{aligned}
$$

Predicting credit risk

| <2 yrs | missed | def? |
|---|---|---|
| N | N | N |
| Y | N | Y |
| N | N | N |
| N | N | N |
| N | Y | Y |
| Y | N | N |
| N | Y | N |
| N | Y | Y |
| Y | N | N |
| Y | N | N |

## Mutual information

- We now have the entropy - the minimal number of bits required to specify the  target attribute:

$$H(Y) \quad = \quad \sum_y P(y) \log_2 P(y)$$

- The conditional entropy - the remaining entropy of Y knowing X

$$H(Y|X) \quad = \quad -\sum_x P(x) \sum_y P(y|x) \log_2 P(y|x)$$

- So we can now define the reduction of the entropy after learning Y.
- This is known as the *mutual information* between Y and X

$$I(Y;X) = H(Y) - H(Y|X)$$



## Properties of mutual information

- Mutual information is symmetric

$$I(Y;X) \quad = \quad I(X;Y)$$

- In terms of probability distributions, it is written as

$$I(X;Y) \quad = \quad -\sum_{x,y} P(x,y) \log_2 \frac{P(x,y)}{P(x)P(y)}$$

- It is zero, if Y provides no information about X:

$I(X;Y) \quad = \quad 0 \quad \Leftrightarrow \quad P(x)$ and $P(y)$ are independent

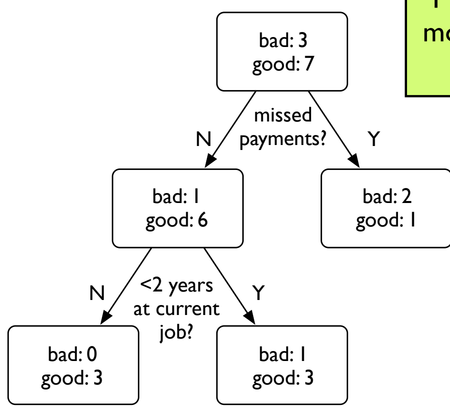
- If Y = X then

$$I(X;X) \quad = \quad H(X) - H(X|X) = H(X)$$

## Information gain

$$
\begin{aligned}
H(\text{defaulted}) \quad &- \quad H(\text{defaulted}|<2 \text{ years}) \\
0.8813 \quad &- \quad 0.8763 = 0.0050 \\
H(\text{defaulted}) \quad &- \quad H(\text{defaulted}|\text{missed}) \\
0.8813 \quad &- \quad 0.6897 = 0.1916
\end{aligned}
$$

Missed payments are the most informative attribute about defaulting.





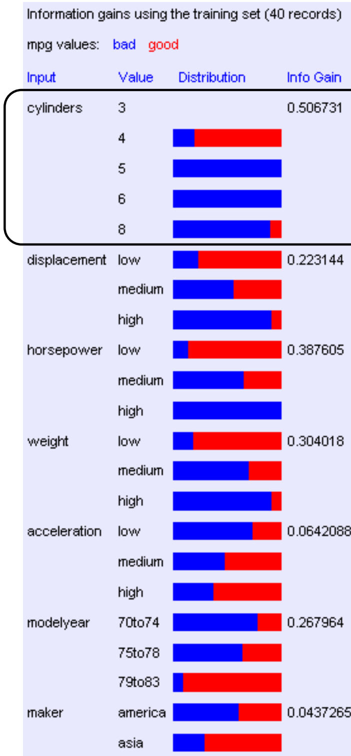

## Example (from Andrew Moore): Predicting miles per gallon

http://www.autonlab.org/tutorials/dtree.html

| 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 |

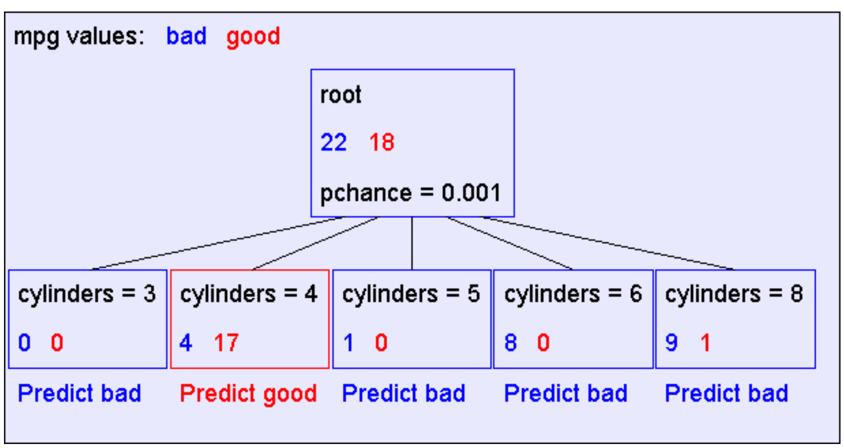
## First step: calculate information gains

- Compute for information gain for each attribute
- In this case, *cylinders* provides the most gain, because it nearly partitions the data.



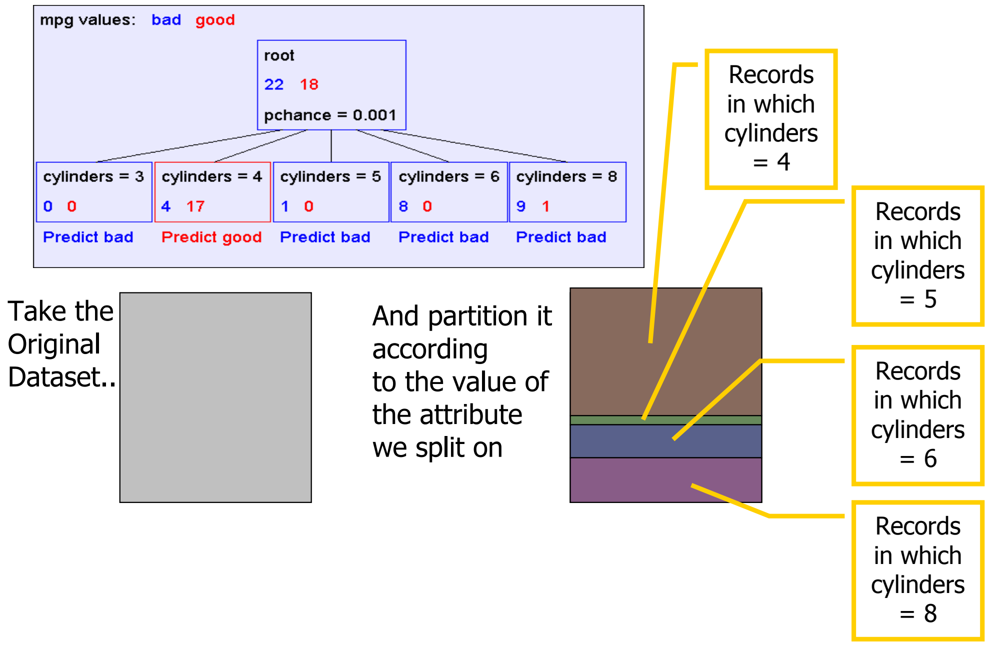



## First decision: partition on cylinders



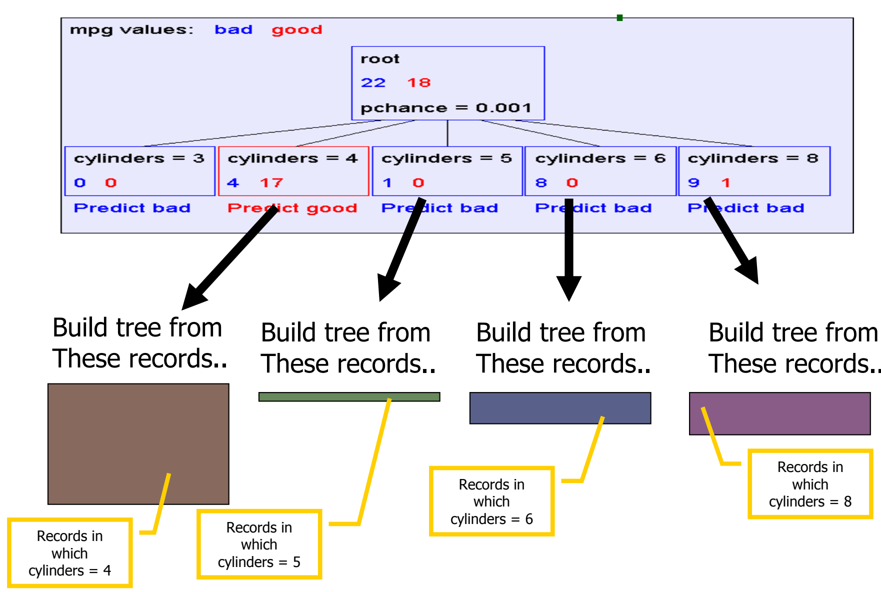

Note the lopsided mpg class distribution.

## Recurse on child nodes to expand tree



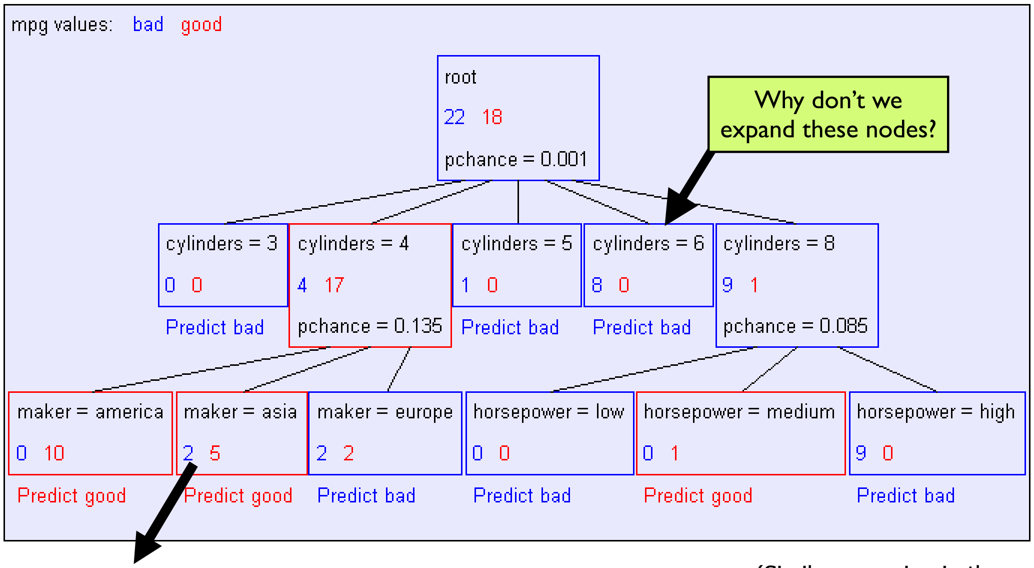



## Expanding the tree: data is partitioned for each child




Exactly the same, but with a smaller, conditioned datasets.

## Second level of decisions




Recursively build a tree from the seven records in which there are four cylinders and the maker was based in Asia

(Similar recursion in the other cases)



## The final tree