



# Decision Trees

Machine Learning – 10701/15781

Carlos Guestrin

Carnegie Mellon University

February 5<sup>th</sup>, 2007

©2005-2007 Carlos Guestrin

# Linear separability

- A dataset is **linearly separable** iff  $\exists$  a **separating hyperplane**:
  - $\exists \mathbf{w}$ , such that:
    - $w_0 + \sum_i w_i x_i > 0$ ; if  $\mathbf{x}=\{x_1, \dots, x_n\}$  is a positive example
    - $w_0 + \sum_i w_i x_i < 0$ ; if  $\mathbf{x}=\{x_1, \dots, x_n\}$  is a negative example

# Not linearly separable data

- Some datasets are **not linearly separable!**

# Addressing non-linearly separable data – Option 1, non-linear features

- Choose non-linear features, e.g.,
  - Typical linear features:  $w_0 + \sum_i w_i x_i$
  - Example of non-linear features:
    - Degree 2 polynomials,  $w_0 + \sum_i w_i x_i + \sum_{ij} w_{ij} x_i x_j$
- Classifier  $h_{\mathbf{w}}(\mathbf{x})$  still linear in parameters  $\mathbf{w}$ 
  - As easy to learn
  - Data is linearly separable in higher dimensional spaces
  - More discussion later this semester

# Addressing non-linearly separable data – Option 2, non-linear classifier

- Choose a classifier  $h_{\mathbf{w}}(\mathbf{x})$  that is non-linear in parameters  $\mathbf{w}$ , e.g.,
  - Decision trees, neural networks, nearest neighbor,...
- More general than linear classifiers
- But, can often be harder to learn (non-convex/concave optimization required)
- But, but, often very useful
- (BTW. Later this semester, we'll see that these options are not that different)

# A small dataset: Miles Per Gallon

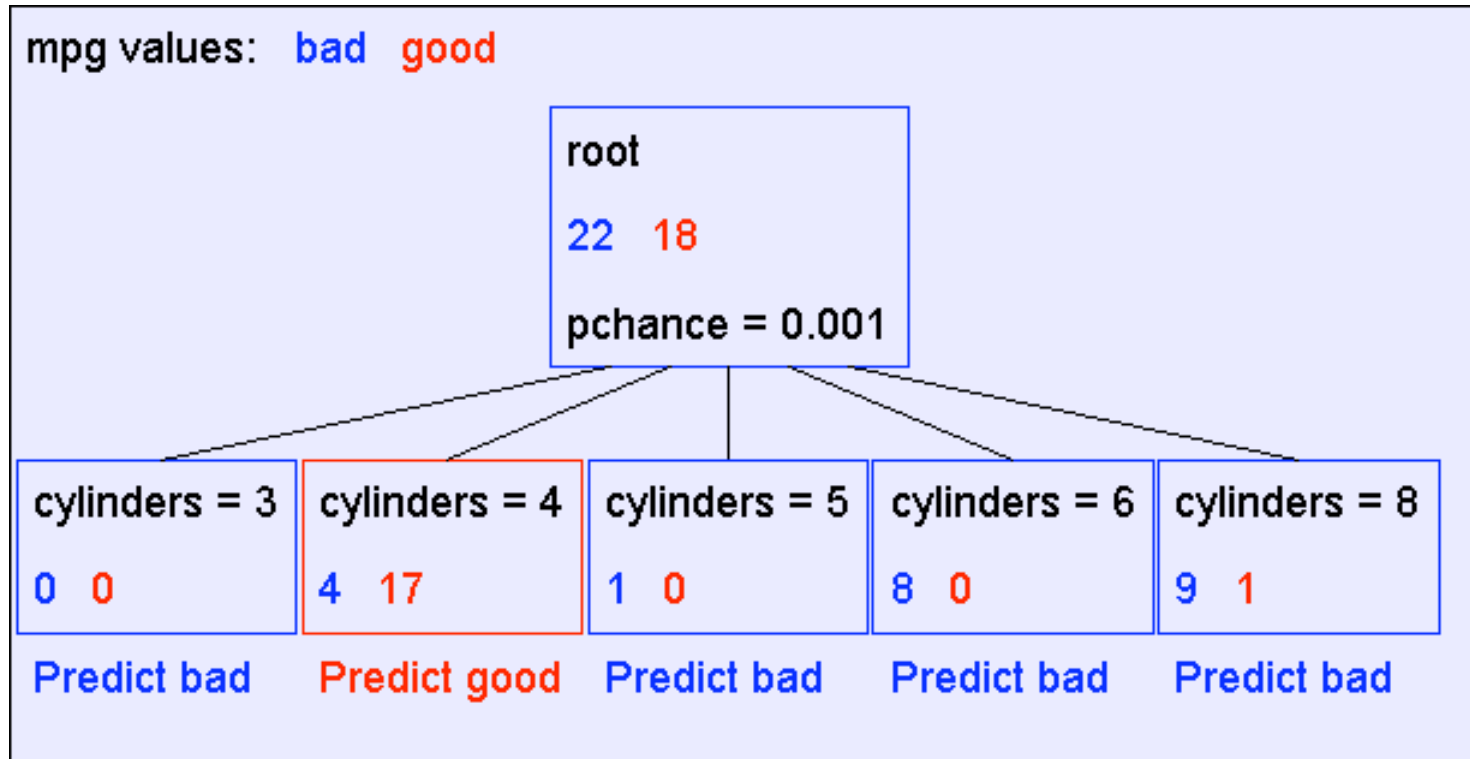
Suppose we want to predict MPG

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

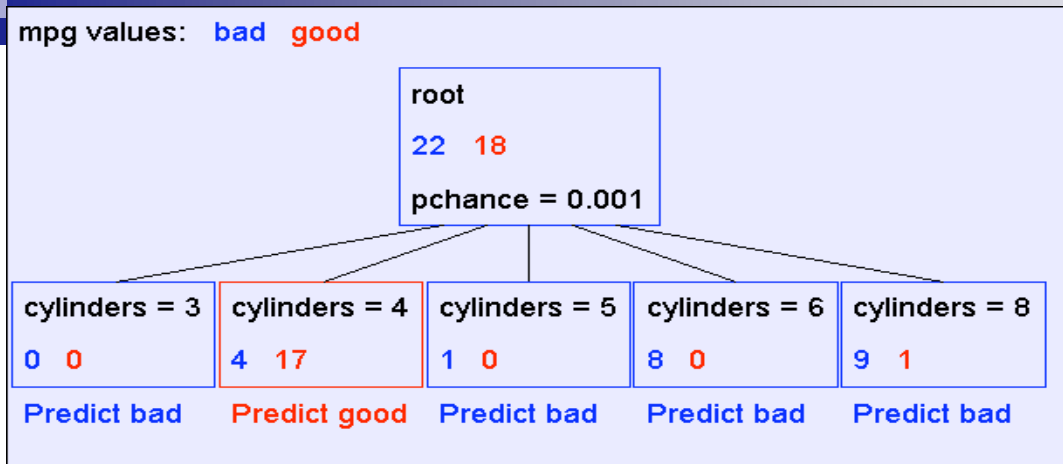
40 Records

From the UCI repository (thanks to Ross Quinlan)

# A Decision Stump



# Recursion Step



Records in which cylinders = 4

Records in which cylinders = 5

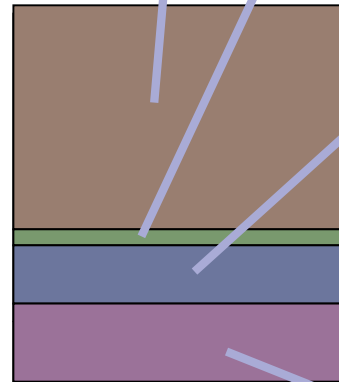
Records in which cylinders = 6

Records in which cylinders = 8

Take the Original Dataset..

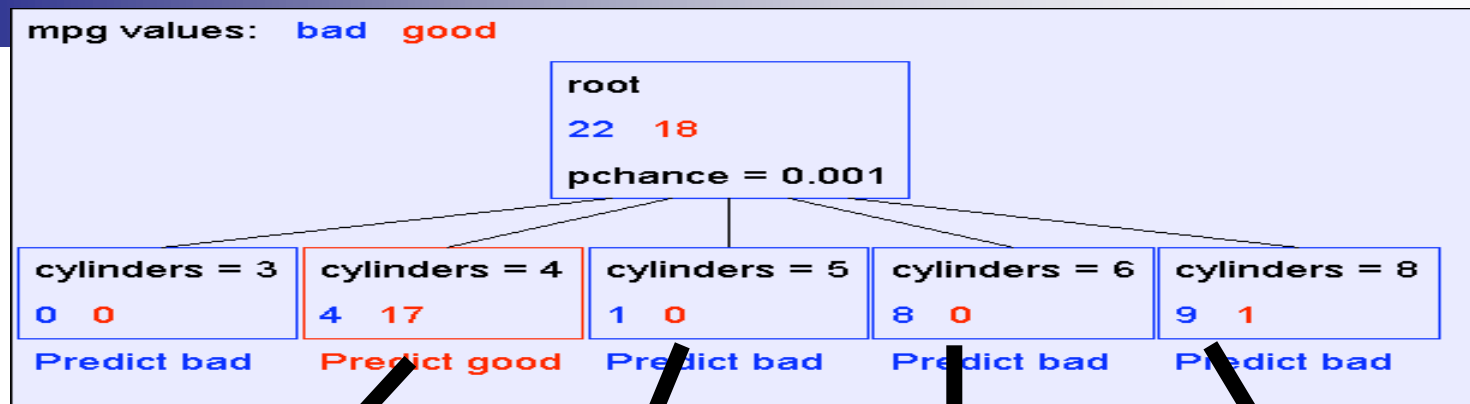


And partition it according to the value of the attribute we split on

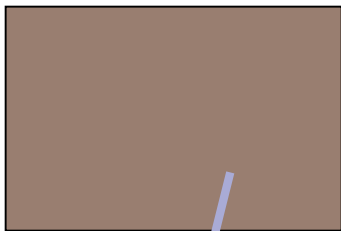




# Recursion Step



Build tree from  
These records..



Records in  
which  
cylinders = 4

Build tree from  
These records..



Records in  
which  
cylinders = 5

Build tree from  
These records..



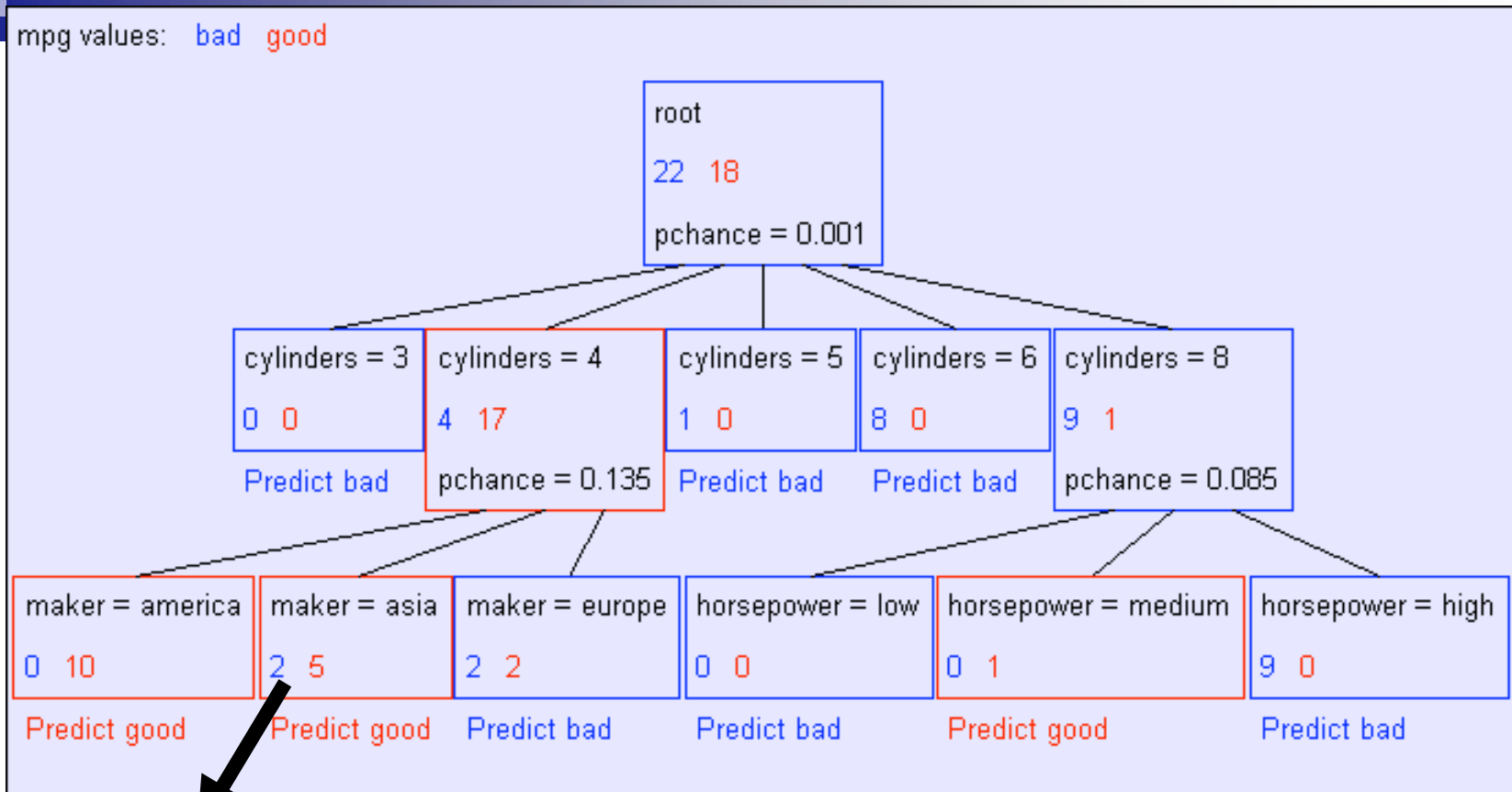
Records in  
which  
cylinders = 6

Build tree from  
These records..



Records in  
which  
cylinders = 8

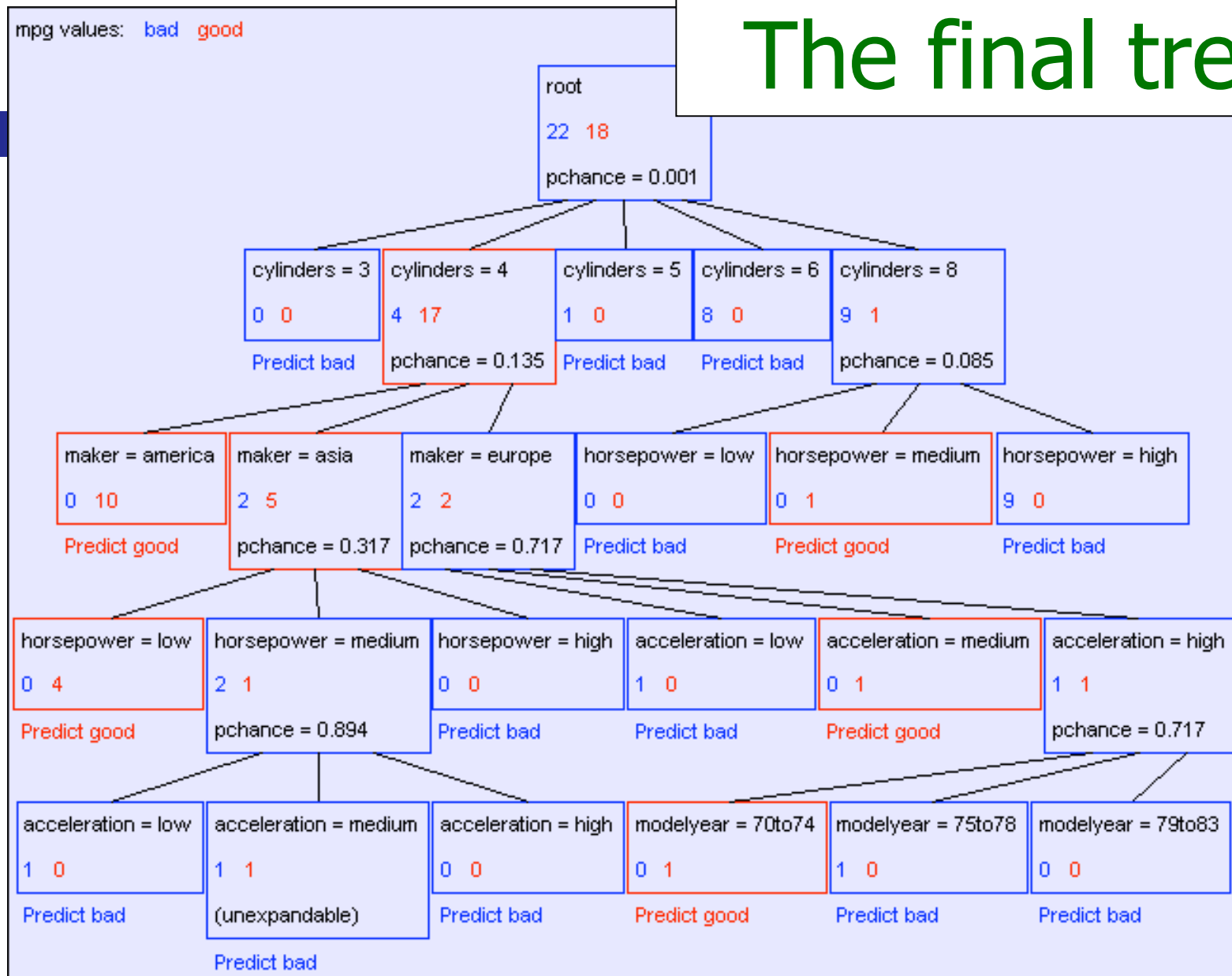
# Second level of tree



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

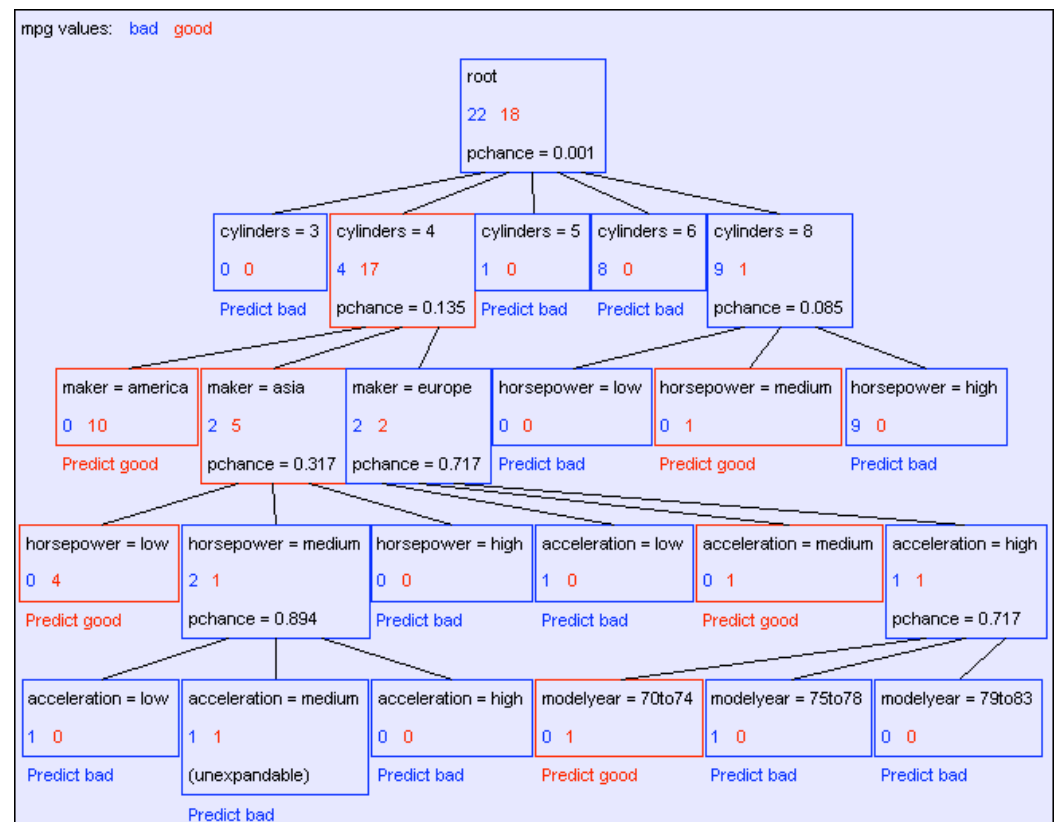
(Similar recursion in the other cases)

# The final tree



# Classification of a new example

- Classifying a test example – traverse tree and report leaf label



# Are all decision trees equal?

- Many trees can represent the same concept
- But, not all trees will have the same size!
  - e.g.,  $\phi = A \wedge B \vee \neg A \wedge C$  ((A and B) or (not A and C))

# Learning decision trees is hard!!!

- Learning the simplest (smallest) decision tree is an NP-complete problem [Hyafil & Rivest '76]
- Resort to a greedy heuristic:
  - Start from empty decision tree
  - Split on **next best attribute (feature)**
  - Recurse

# Choosing a good attribute



$X_1$	$X_2$	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F

# Measuring uncertainty

- Good split if we are more certain about classification after split
  - Deterministic good (all true or all false)
  - Uniform distribution bad

$P(Y=A) = 1/2$	$P(Y=B) = 1/4$	$P(Y=C) = 1/8$	$P(Y=D) = 1/8$
----------------	----------------	----------------	----------------

$P(Y=A) = 1/4$	$P(Y=B) = 1/4$	$P(Y=C) = 1/4$	$P(Y=D) = 1/4$
----------------	----------------	----------------	----------------



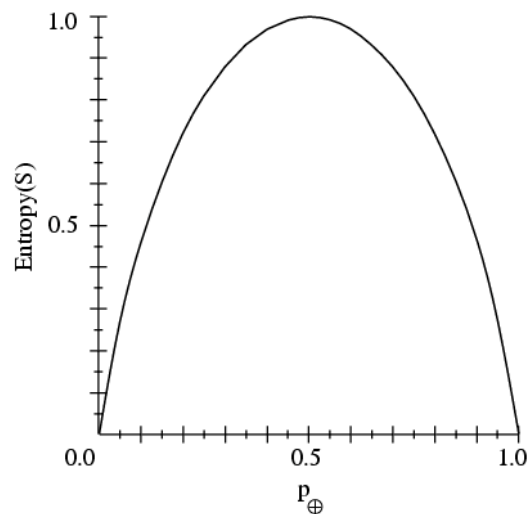
# Entropy

Entropy  $H(X)$  of a random variable  $Y$

$$H(Y) = - \sum_{i=1}^k P(Y = y_i) \log_2 P(Y = y_i)$$

**More uncertainty, more entropy!**

*Information Theory interpretation:*  $H(Y)$  is the expected number of bits needed to encode a randomly drawn value of  $Y$  (under most efficient code)



# Andrew Moore's Entropy in a nutshell



Low Entropy



High Entropy



# Andrew Moore's Entropy in a nutshell



Low Entropy

..the values (locations of soup) sampled entirely from within the soup bowl



High Entropy

..the values (locations of soup) unpredictable... almost uniformly sampled throughout our dining room

# Information gain

$X_1$	$X_2$	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F

- Advantage of attribute – decrease in uncertainty
  - Entropy of Y before you split
  - Entropy after split
    - Weight by probability of following each branch, i.e., normalized number of records

$$H(Y | X) = - \sum_{j=1}^v P(X = x_j) \sum_{i=1}^k P(Y = y_i | X = x_j) \log_2 P(Y = y_i | X = x_j)$$

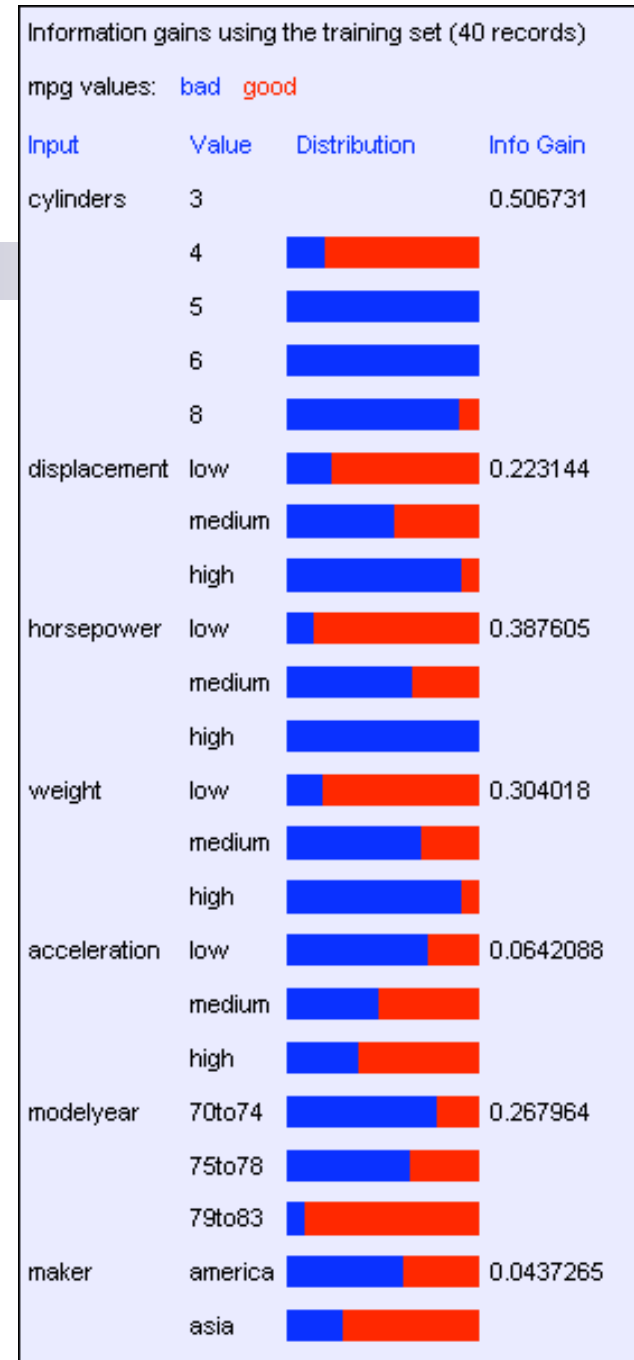
- Information gain is difference  $IG(X) = H(Y) - H(Y | X)$

# Learning decision trees

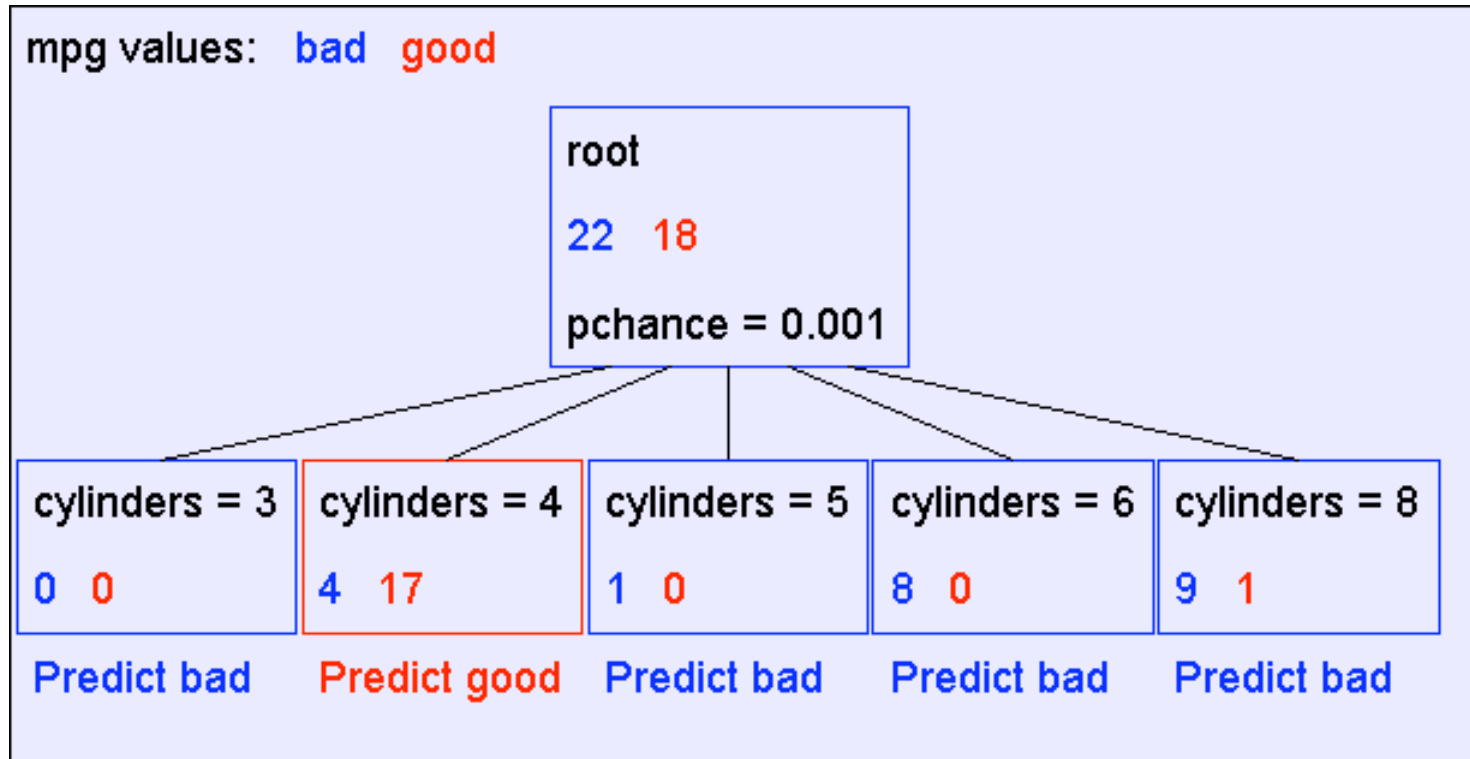
- Start from empty decision tree
- Split on **next best attribute (feature)**
  - Use, for example, information gain to select attribute
  - Split on  $\arg \max_i IG(X_i) = \arg \max_i H(Y) - H(Y | X_i)$
- Recurse

Suppose we want to predict MPG

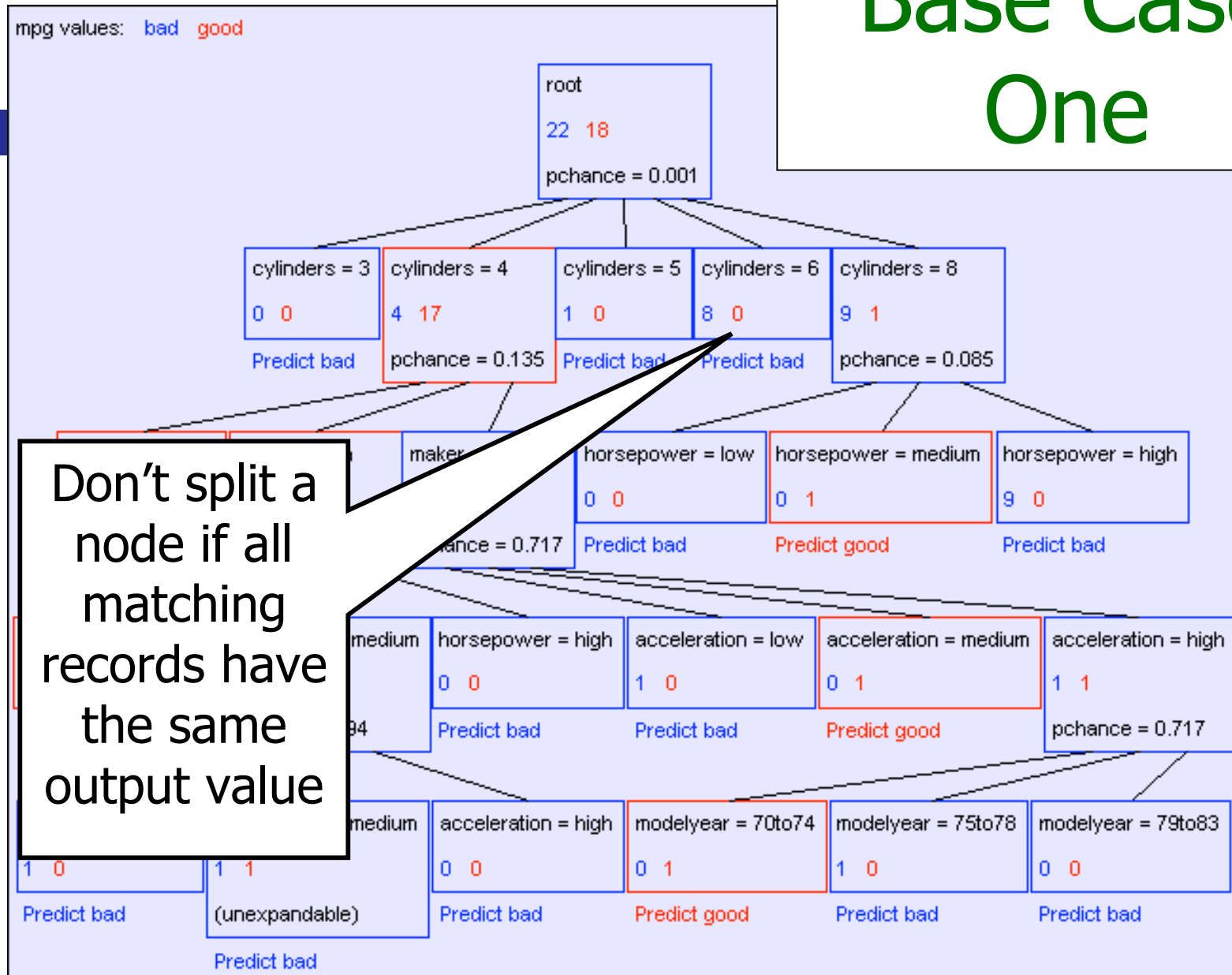
Look at all the information gains...



# A Decision Stump

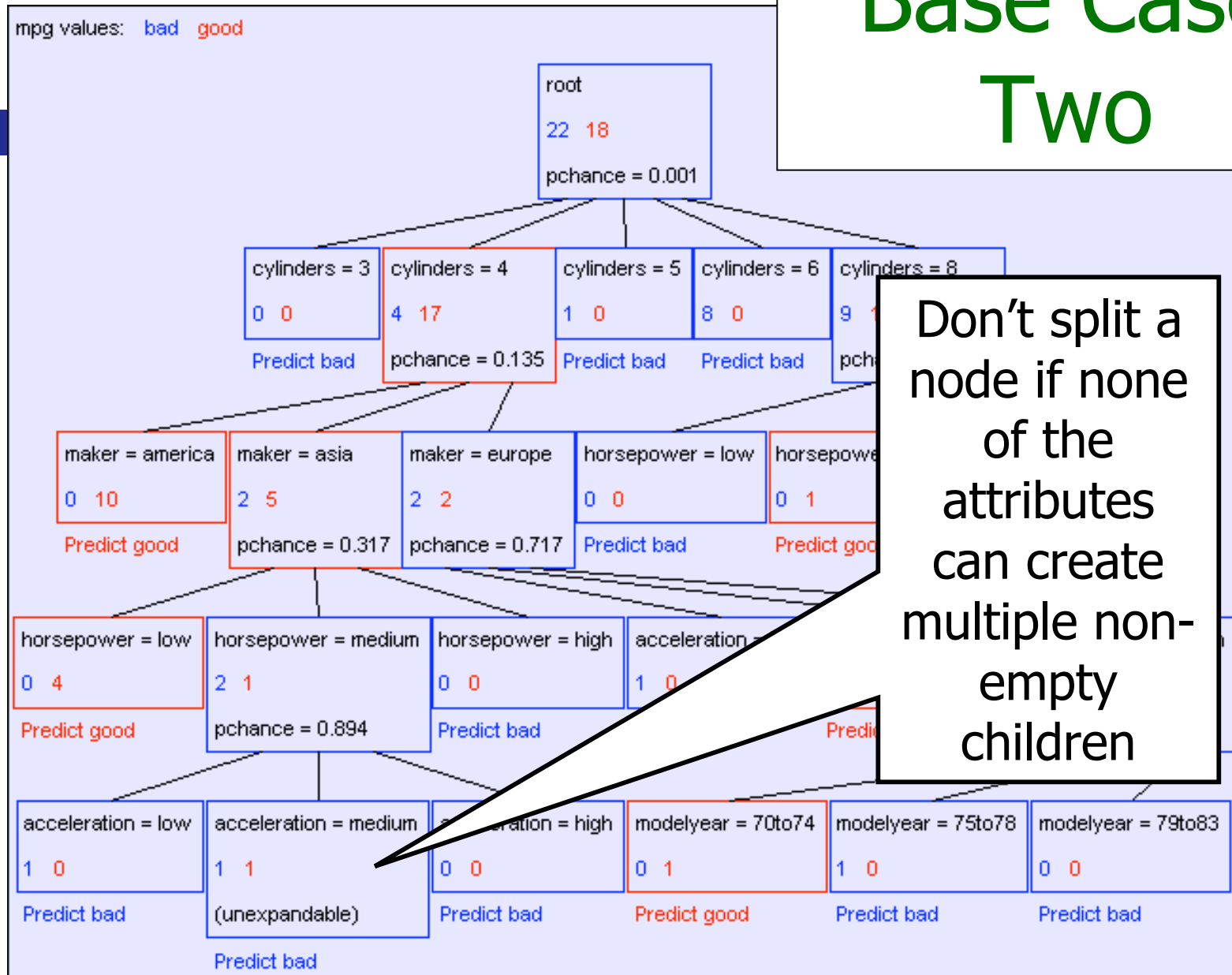


# Base Case One



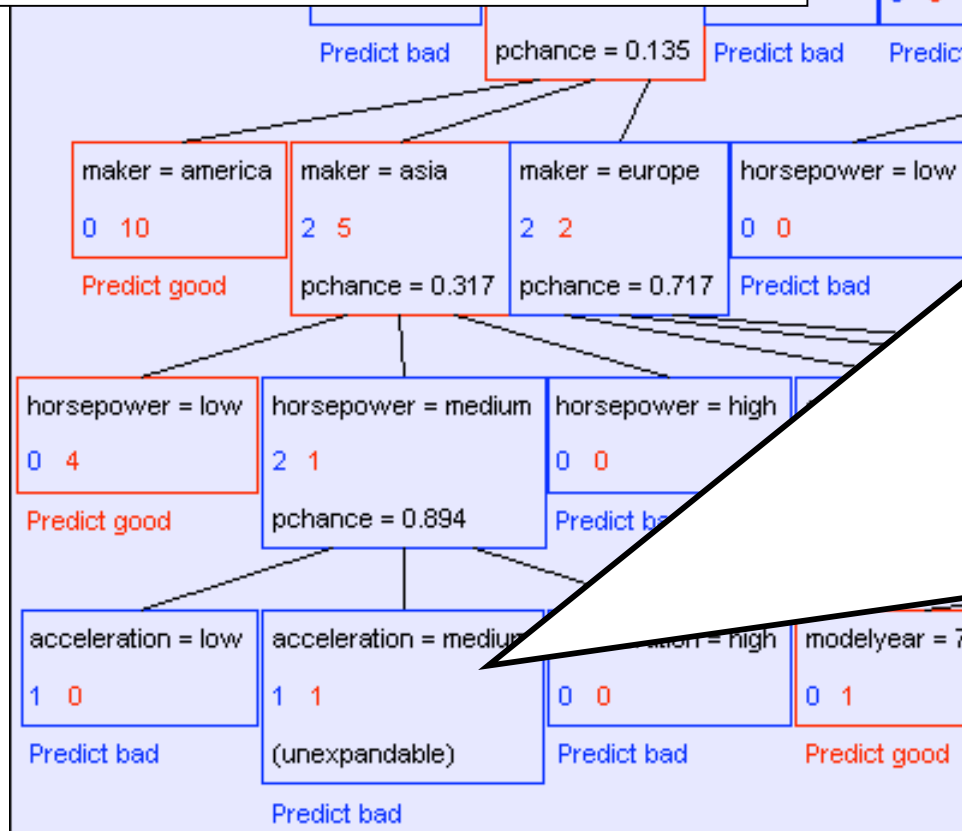


# Base Case Two



Don't split a node if none of the attributes can create multiple non-empty children

# Base Case Two: No attributes can distinguish



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		

# Base Cases



- Base Case One: If all records in current data subset have the same output then **don't recurse**
- Base Case Two: If all records have exactly the same set of input attributes then **don't recurse**

# Base Cases: An idea

- Base Case One: If all records in current data subset have the same output then **don't recurse**
- Base Case Two: If all records have exactly the same set of input attributes then **don't recurse**

Proposed Base Case 3:

If all attributes have zero information gain then **don't recurse**

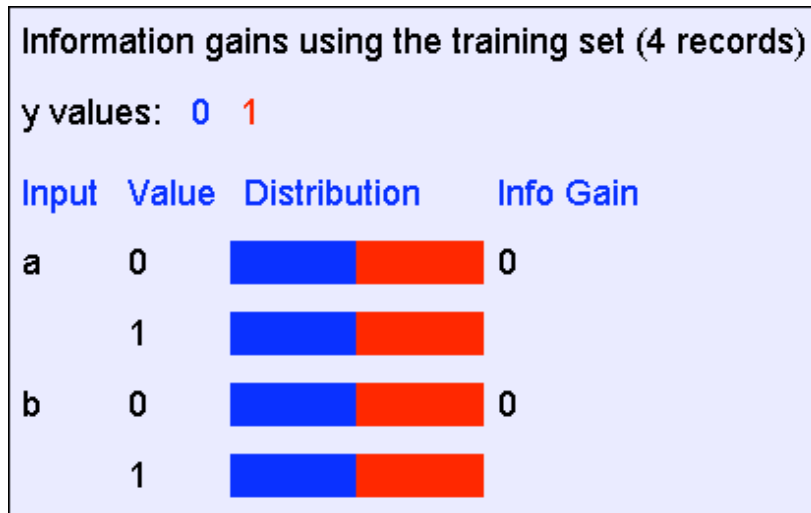
• *Is this a good idea?*

# The problem with Base Case 3

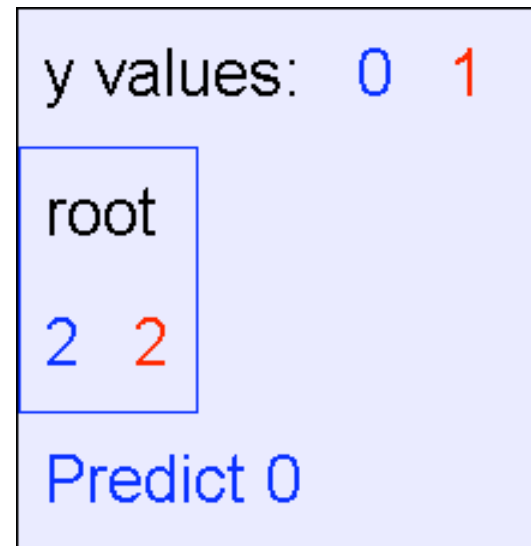
a	b	y
0	0	0
0	1	1
1	0	1
1	1	0

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

The information gains:



The resulting decision tree:

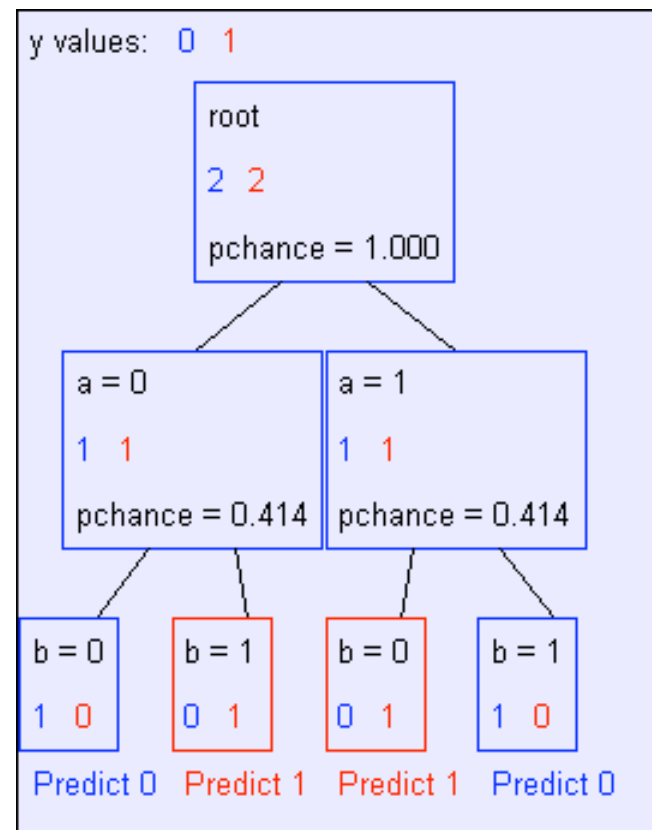


# If we omit Base Case 3:

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

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

The resulting decision tree:



# Basic Decision Tree Building

## Summarized

BuildTree(*DataSet*, *Output*)

- If all output values are the same in *DataSet*, return a leaf node that says “predict this unique output”
- If all input values are the same, return a leaf node that says “predict the majority output”
- Else find attribute  $X$  with highest Info Gain
- Suppose  $X$  has  $n_X$  distinct values (i.e.  $X$  has arity  $n_X$ ).
  - Create and return a non-leaf node with  $n_X$  children.
  - The  $i$ 'th child should be built by calling  
BuildTree( $DS_i$ , *Output*)

Where  $DS_i$  built consists of all those records in *DataSet* for which  $X = i$ th distinct value of  $X$ .

# Announcements



- **Pittsburgh won the Super Bowl !!**
  - Last year...

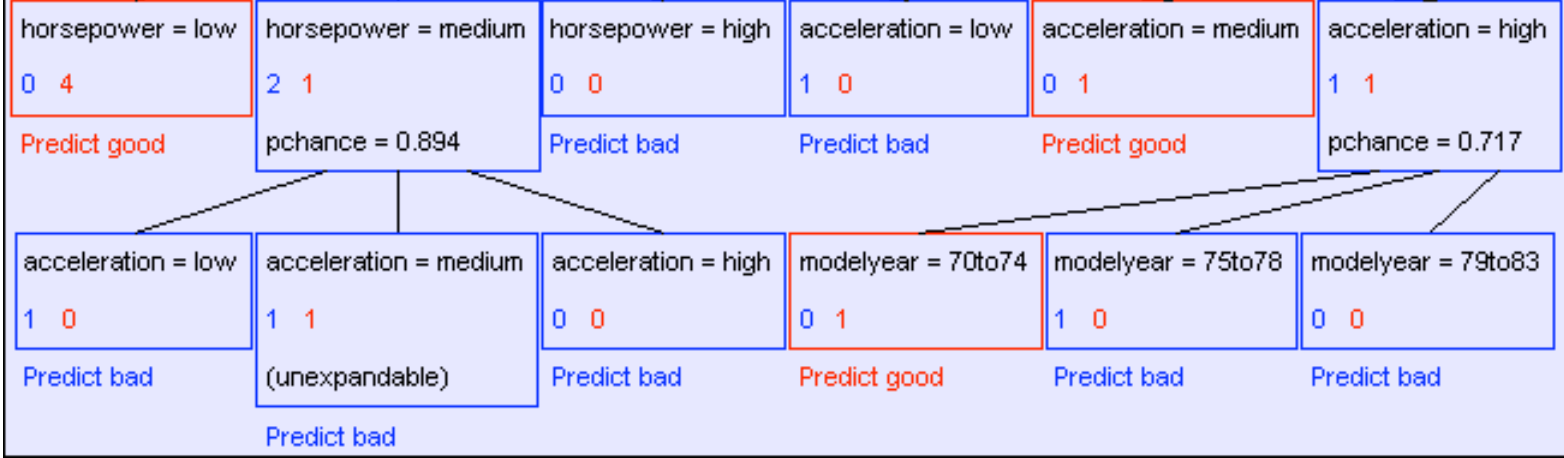


# MPG Test set error

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



# MPG Test set 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

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

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

...why?

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

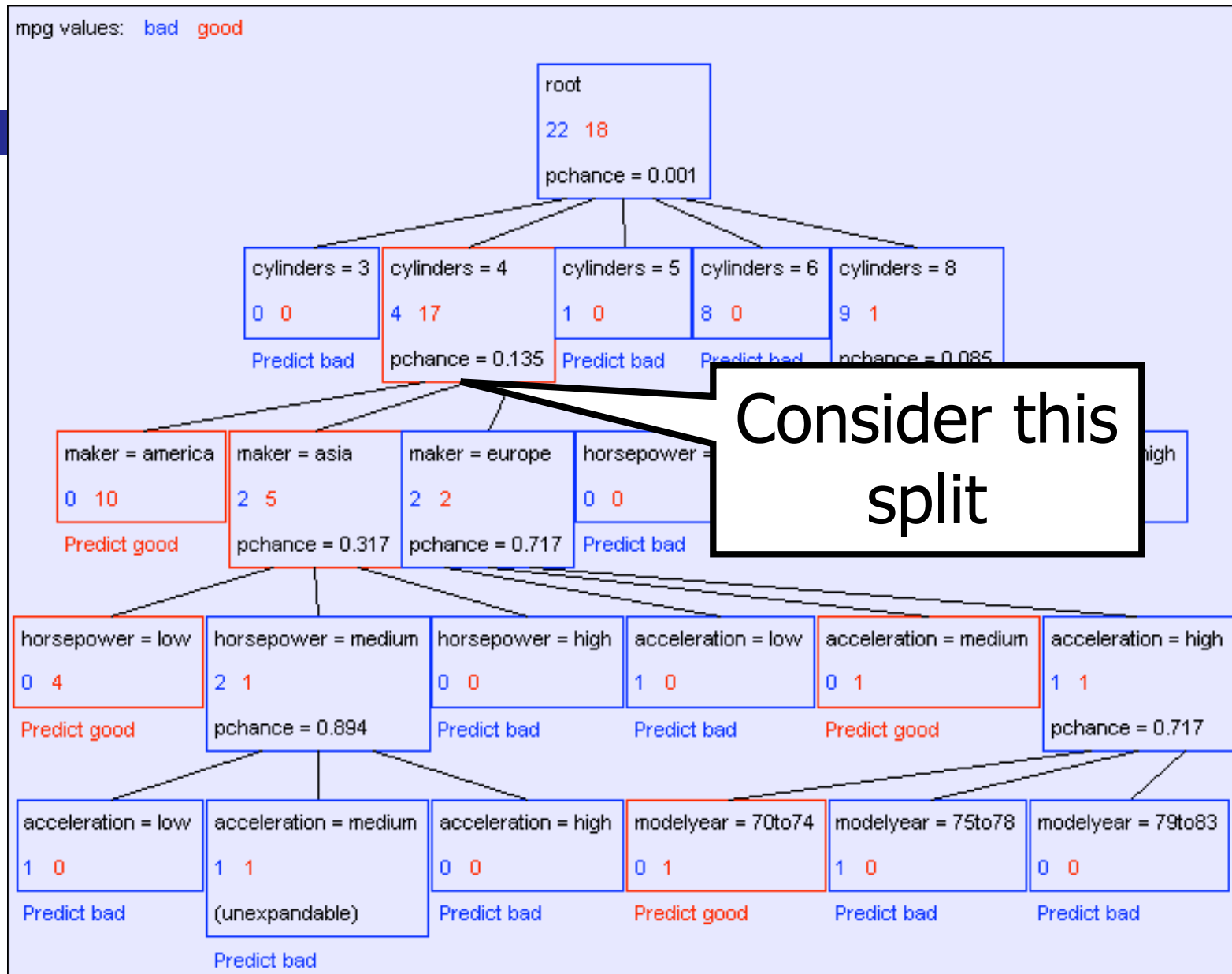
# Decision trees & Learning Bias



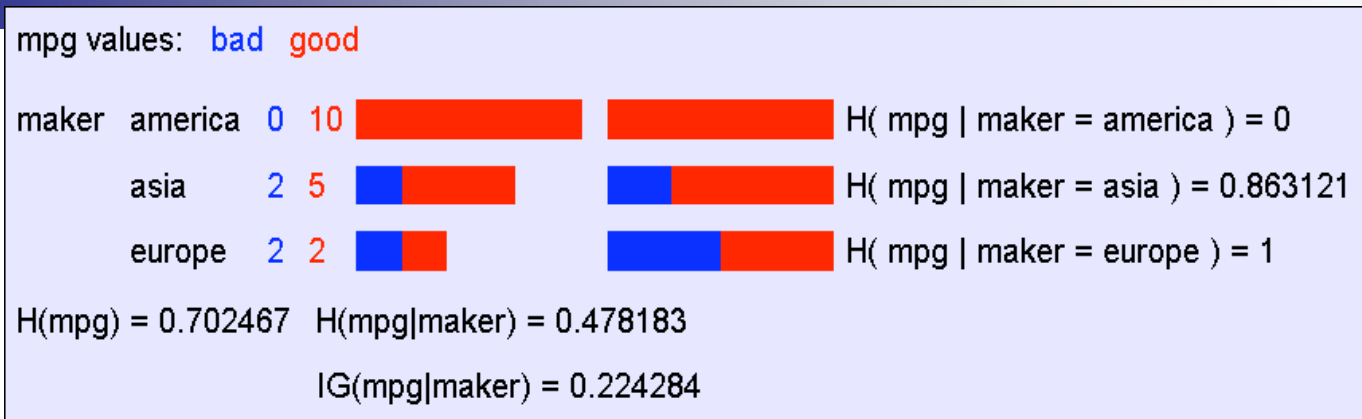
mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	75to78	asia
bad	6	medium	medium	medium	medium	70to74	america
bad	4	medium	medium	medium	low	75to78	europa
bad	8	high	high	high	low	70to74	america
bad	6	medium	medium	medium	medium	70to74	america
bad	4	low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

# Decision trees will overfit

- Standard decision trees are have no learning biased
  - Training set error is always zero!
    - (If there is no label noise)
  - Lots of variance
  - Will definitely overfit!!!
  - Must bias towards simpler trees
- Many strategies for picking simpler trees:
  - Fixed depth
  - Fixed number of leaves
  - Or something smarter...



# A chi-square test



- Suppose that mpg was completely uncorrelated with maker.
- What is the chance we'd have seen data of at least this apparent level of association anyway?

# A chi-square test

mpg values: bad good

maker	america	0	10			$H(\text{mpg} \mid \text{maker} = \text{america}) = 0$
	asia	2	5			$H(\text{mpg} \mid \text{maker} = \text{asia}) = 0.863121$
	europa	2	2			$H(\text{mpg} \mid \text{maker} = \text{europa}) = 1$

$H(\text{mpg}) = 0.702467$   $H(\text{mpg} \mid \text{maker}) = 0.478183$   
 $IG(\text{mpg} \mid \text{maker}) = 0.224284$

- Suppose that mpg was completely uncorrelated with maker.
- What is the chance we'd have seen data of at least this apparent level of association anyway?

By using a particular kind of chi-square test, the answer is 7.2%

(Such simple hypothesis tests are very easy to compute, unfortunately, not enough time to cover in the lecture, but in your homework, you'll have fun! :))

# Using Chi-squared to avoid overfitting

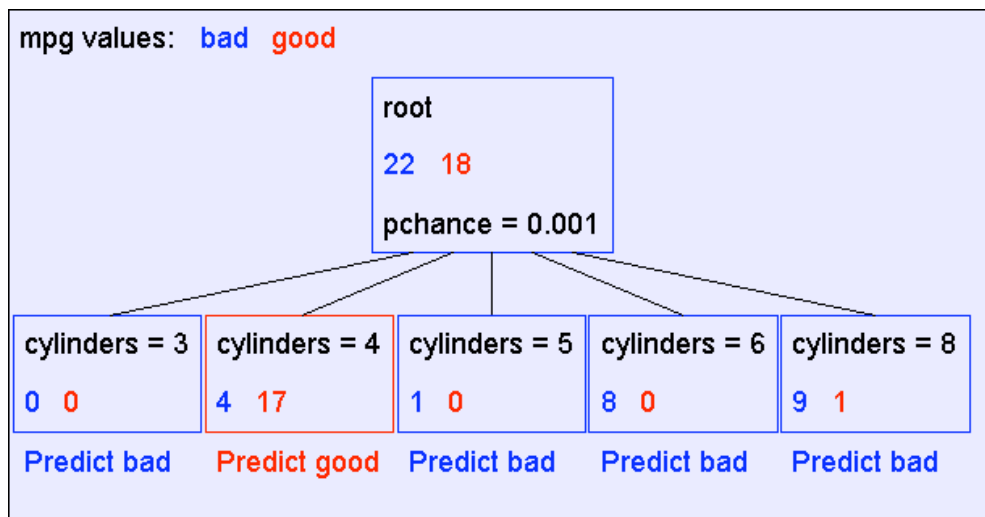
- Build the full decision tree as before
- But when you can grow it no more, start to prune:
  - Beginning at the bottom of the tree, delete splits in which  $p_{chance} > MaxPchance$
  - Continue working your way up until there are no more prunable nodes

*MaxPchance* is a magic parameter you must specify to the decision tree, indicating your willingness to risk fitting noise



# Pruning example

- With MaxPchance = 0.1, you will see the following MPG decision tree:

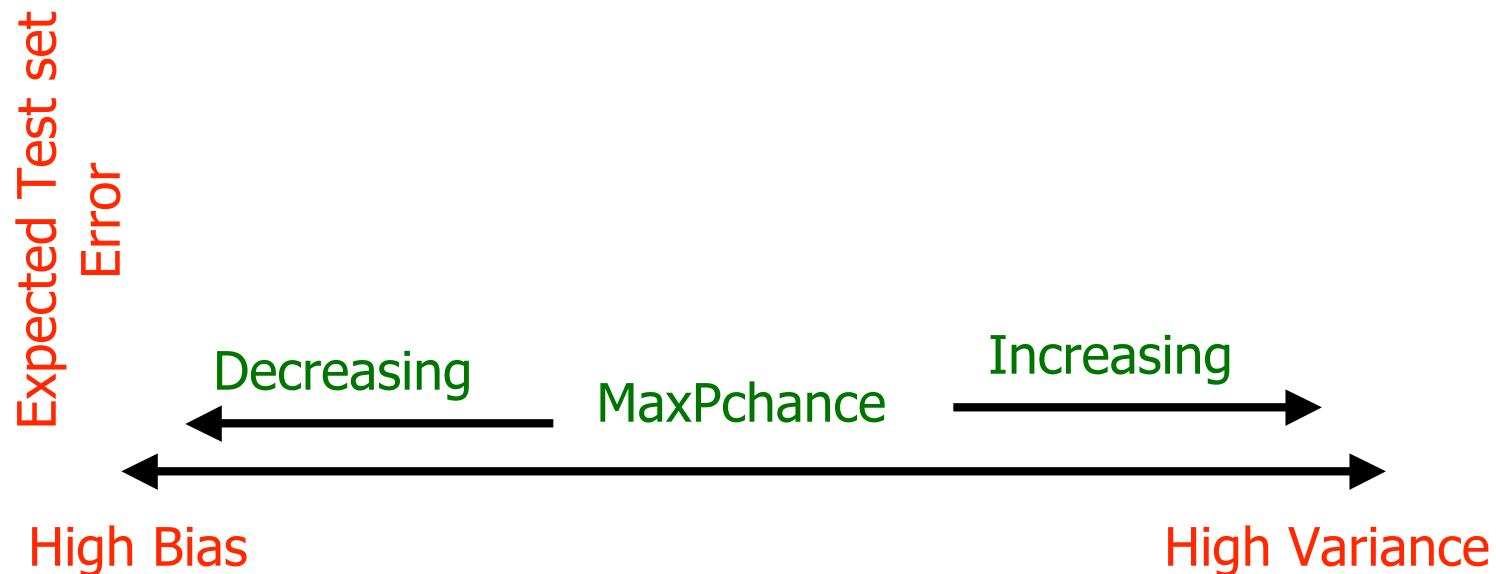


Note the improved test set accuracy compared with the unpruned tree

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

# MaxPchance

- Technical note MaxPchance is a regularization parameter that helps us bias towards simpler models



- We'll learn to choose the value of these magic parameters soon!

# Real-Valued inputs

- What should we do if some of the inputs are real-valued?

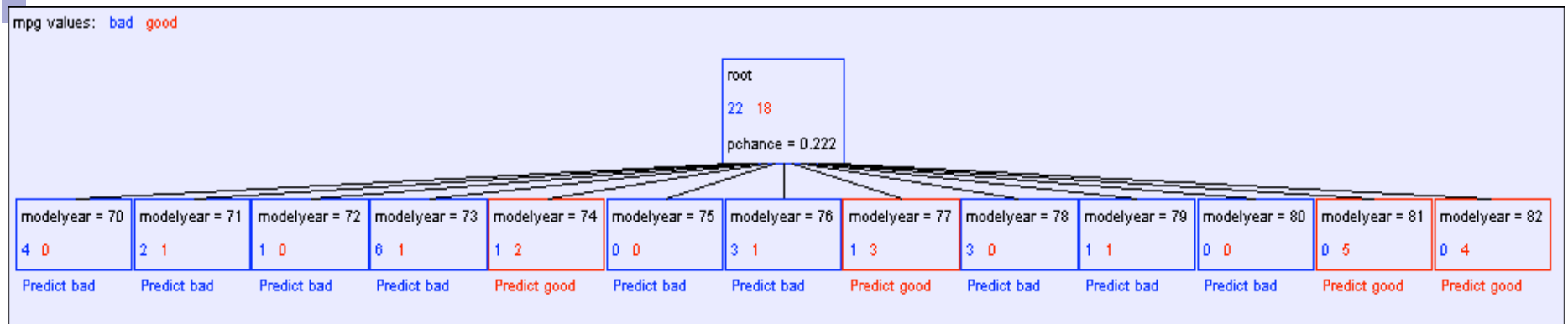
mpg	cylinders	displacemen	horsepower	weight	acceleration	modelyear	maker
good	4	97	75	2265	18.2	77	asia
bad	6	199	90	2648	15	70	america
bad	4	121	110	2600	12.8	77	europa
bad	8	350	175	4100	13	73	america
bad	6	198	95	3102	16.5	74	america
bad	4	108	94	2379	16.5	73	asia
bad	4	113	95	2228	14	71	asia
bad	8	302	139	3570	12.8	78	america
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
good	4	120	79	2625	18.6	82	america
bad	8	455	225	4425	10	70	america
good	4	107	86	2464	15.5	76	europa
bad	5	131	103	2830	15.9	78	europa

Infinite number of possible split values!!!

Finite dataset, only finite number of relevant splits!

Idea One: Branch on each possible real value

# “One branch for each numeric value” idea:



Hopeless: with such high branching factor will shatter the dataset and overfit

# Threshold splits



- Binary tree, split on attribute  $X$ 
  - One branch:  $X < t$
  - Other branch:  $X \geq t$

# Choosing threshold split

- Binary tree, split on attribute  $X$ 
  - One branch:  $X < t$
  - Other branch:  $X \geq t$
- Search through possible values of  $t$ 
  - Seems hard!!!
- But only finite number of  $t$ 's are important
  - Sort data according to  $X$  into  $\{x_1, \dots, x_m\}$
  - Consider split points of the form  $x_i + (x_{i+1} - x_i)/2$
















# A better idea: thresholded splits

- Suppose  $X$  is real valued
- Define  $IG(Y|X:t)$  as  $H(Y) - H(Y|X:t)$
- Define  $H(Y|X:t) =$   
$$H(Y|X < t) P(X < t) + H(Y|X \geq t) P(X \geq t)$$
  - $IG(Y|X:t)$  is the information gain for predicting  $Y$  if all you know is whether  $X$  is greater than or less than  $t$
- Then define  $IG^*(Y|X) = \max_t IG(Y|X:t)$
- For each real-valued attribute, use  $IG^*(Y|X)$  for assessing its suitability as a split

# Example with MPG

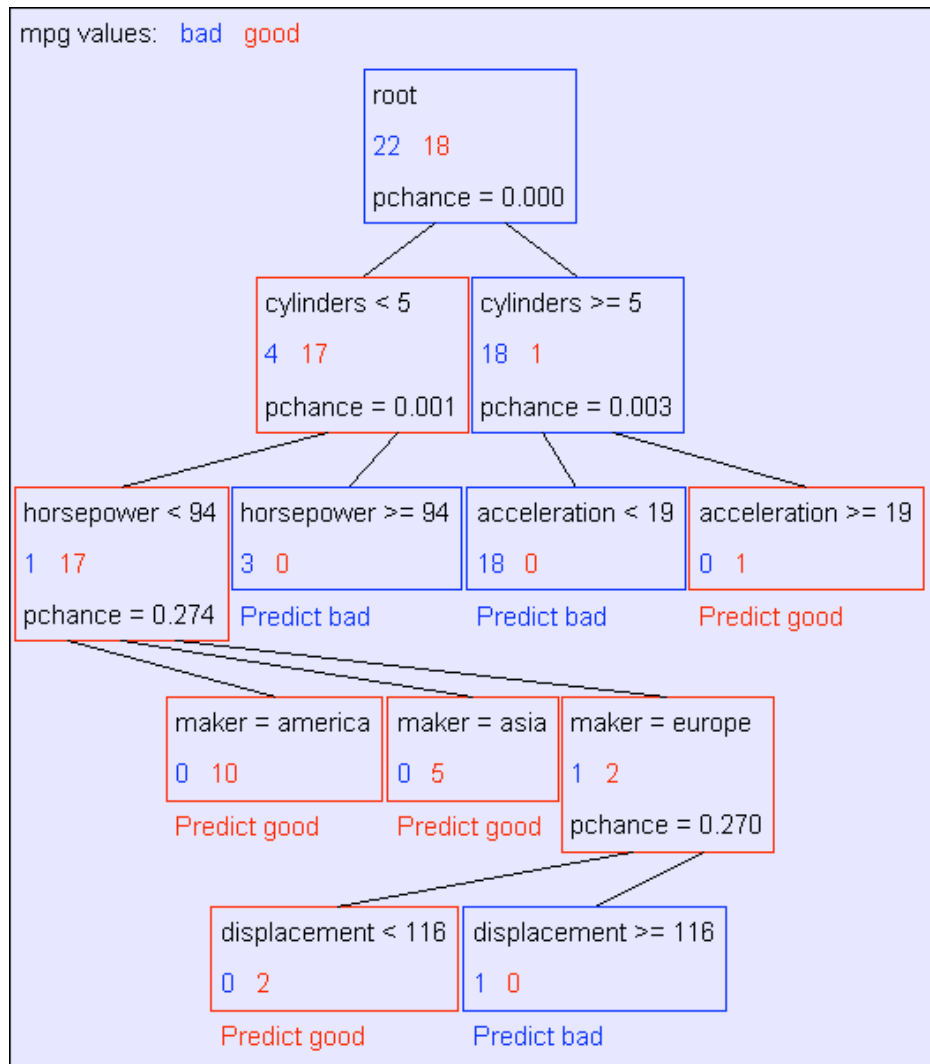
Information gains using the training set (40 records)

mpg values: bad good

Input	Value	Distribution	Info Gain
cylinders	< 5		0.48268
	>= 5		
displacement	< 198		0.428205
	>= 198		
horsepower	< 94		0.48268
	>= 94		
weight	< 2789		0.379471
	>= 2789		
acceleration	< 18.2		0.159982
	>= 18.2		
modelyear	< 81		0.319193
	>= 81		
maker	america		0.0437265
	asia		
	europa		



# Example tree using reals



# What you need to know about decision trees

- Decision trees are one of the most popular data mining tools
  - Easy to understand
  - Easy to implement
  - Easy to use
  - Computationally cheap (to solve heuristically)
- Information gain to select attributes (ID3, C4.5,...)
- Presented for classification, can be used for regression and density estimation too
- Decision trees will overfit!!!
  - Zero bias classifier → Lots of variance
  - Must use tricks to find “simple trees”, e.g.,
    - Fixed depth/Early stopping
    - Pruning
    - Hypothesis testing

# Acknowledgements



- Some of the material in the presentation is courtesy of Andrew Moore, from his excellent collection of ML tutorials:

- <http://www.cs.cmu.edu/~awm/tutorials>