Logistic Regressio, cont.

Decision Trees

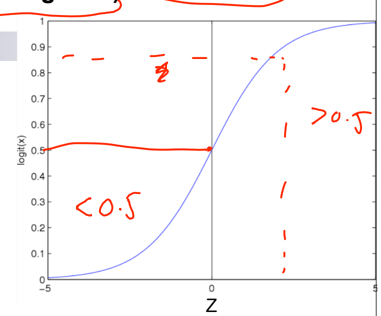
Machine Learning – 10701/15781
Carlos Guestrin
Carnegie Mellon University

September 26th, 2007

©Carlos Guestrin 2005-2007

Logistic Regression

Logistic function (or Sigmoid): $\frac{1}{1 + exp(-z)}$



- Learn P(Y|**X**) directly!
  - Assume a particular functional form
  - Sigmoid applied to a linear function of the data:

$$P(Y = 1|X) = \frac{1}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i)}$$

Features can be discrete or continuous!

©Carlos Guestrin 2005-2007

Loss functions: Likelihood v. Conditional Likelihood

- Generative (Naïve Bayes) Loss function:
  Data likelihood

$$\ln P(\mathcal{D} \mid \mathbf{w}) = \sum_{j=1}^{N} \ln P(\mathbf{x}^j, y^j \mid \mathbf{w})$$
$$= \sum_{j=1}^{N} \ln P(y^j \mid \mathbf{x}^j, \mathbf{w}) + \sum_{j=1}^{N} \ln P(\mathbf{x}^j \mid \mathbf{w})$$

- Discriminative models cannot compute P(**x**^{j}|**w**)!
- But, discriminative (logistic regression) loss function:
  Conditional Data Likelihood

$$\ln P(\mathcal{D}_Y \mid \mathcal{D}_{\mathbf{X}}, \mathbf{w}) = \sum_{j=1}^{N} \ln P(y^j \mid \mathbf{x}^j, \mathbf{w})$$

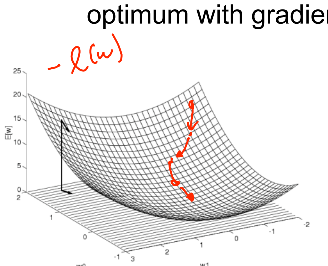
  - Doesn't waste effort learning P(X) – focuses on P(Y|**X**) all that matters for classification

3

©Carlos Guestrin 2005-2007

Optimizing concave function – Gradient ascent

- Conditional likelihood for Logistic Regression is concave → Find optimum with gradient ascent



Gradient: $\nabla_{\mathbf{w}} l(\mathbf{w}) = [\frac{\partial l(\mathbf{w})}{\partial w_0}, \ldots, \frac{\partial l(\mathbf{w})}{\partial w_n}]'$

Learning rate, η>0

Update rule: $\Delta \mathbf{w} = \eta \nabla_{\mathbf{w}} l(\mathbf{w})$

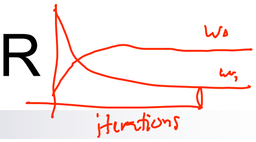
$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \frac{\partial l(\mathbf{w})}{\partial w_i}$$

- Gradient ascent is simplest of optimization approaches
  - e.g., Conjugate gradient ascent much better (see reading)

4

©Carlos Guestrin 2005-2007

Gradient Descent for LR



Gradient ascent algorithm: iterate until change < ε

$$w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \sum_j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w})]$$

For *i* = *1… n*,

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w})]$$

repeat

©Carlos Guestrin 2005-2007

That's all M(C)LE. How about MAP?

$$p(\mathbf{w} \mid Y, \mathbf{X}) \propto P(Y \mid \mathbf{X}, \mathbf{w}) p(\mathbf{w})$$

- One common approach is to define priors on **w**
  - Normal distribution, zero mean, identity covariance
  - "Pushes" parameters towards zero
- Corresponds to ***Regularization***
  - Helps avoid very large weights and overfitting
  - More on this later in the semester
- MAP estimate

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \ln \left[p(\mathbf{w}) \prod_{j=1}^{N} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]$$

©Carlos Guestrin 2005-2007

M(C)AP as Regularization

$$\ln \left[p(\mathbf{w}) \prod_{j=1}^{N} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]$$

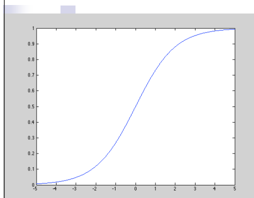
$$p(\mathbf{w}) = \prod_i \frac{1}{\kappa\sqrt{2\pi}} \; e^{\frac{-w_i^2}{2\kappa^2}}$$

Penalizes high weights, also applicable in linear regression

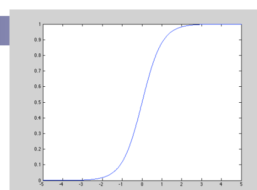
7

©Carlos Guestrin 2005-2007

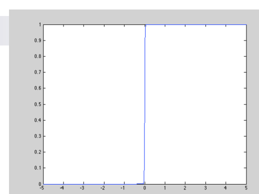
Large parameters → Overfitting



$$\frac{1}{1+e^{-x}}$$



$$\frac{1}{1+e^{-2x}}$$



$$\frac{1}{1+e^{-100x}}$$

- If data is linearly separable, weights go to infinity
- Leads to overfitting:
- Penalizing high weights can prevent overfitting…
  - again, more on this later in the semester

8

©Carlos Guestrin 2005-2007

Gradient of M(C)AP

$$\frac{\partial}{\partial w_i} \ln \left[p(\mathbf{w}) \prod_{j=1}^{N} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]$$

$$p(\mathbf{w}) = \prod_i \frac{1}{\kappa\sqrt{2\pi}} \; e^{\frac{-w_i^2}{2\kappa^2}}$$

©Carlos Guestrin 2005-2007

MLE vs MAP

- Maximum conditional likelihood estimate

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \ln \left[\prod_{j=1}^{N} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]$$

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w})]$$

- Maximum conditional a posteriori estimate

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \ln \left[p(\mathbf{w}) \prod_{j=1}^{N} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]$$

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \left\{ -\lambda w_i^{(t)} + \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w})] \right\}$$

©Carlos Guestrin 2005-2007

G. Naïve Bayes vs. Logistic Regression 1

[Ng & Jordan, 2002]

- Generative and Discriminative classifiers
  - focuses on setting when GNB leads to linear classifier
    - variance σ_i (depends on feature i, not on class k)
- Asymptotic comparison (# training examples → infinity)
  - when GNB model correct
    - GNB, LR produce identical classifiers
  - when model incorrect
    - LR is less biased – does not assume conditional independence
      - **therefore LR expected to outperform GNB**

11

©Carlos Guestrin 2005-2007

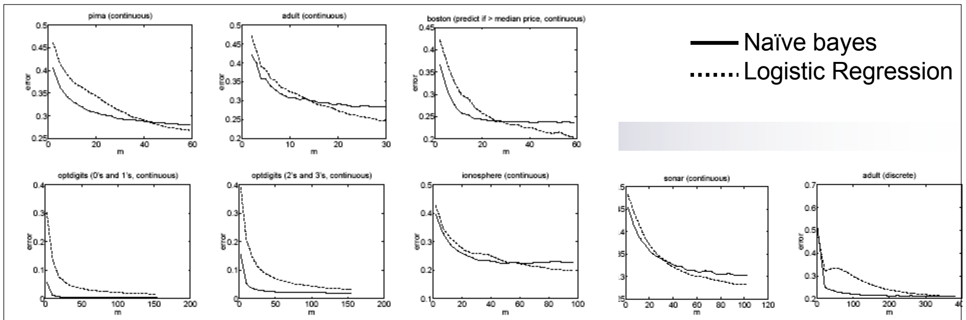
G. Naïve Bayes vs. Logistic Regression 2

[Ng & Jordan, 2002]

- Generative and Discriminative classifiers
  - focuses on setting when GNB leads to linear classifier
- Non-asymptotic analysis
  - convergence rate of parameter estimates, n = # of attributes in X
    - Size of training data to get close to infinite data solution
    - GNB needs *O*(log n) samples
    - LR needs *O*(n) samples
  - **GNB converges more quickly to its (perhaps less helpful) asymptotic estimates**

12

©Carlos Guestrin 2005-2007



Some experiments from UCI data sets

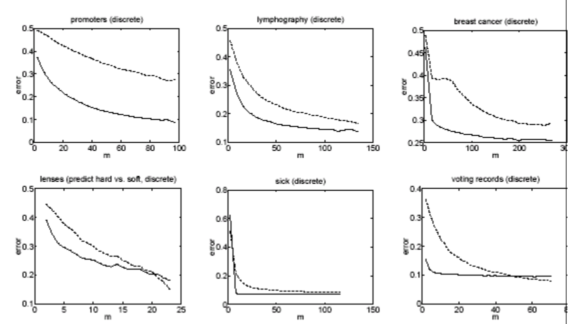


Figure 1: Results of 15 experiments on datasets from the UCI Machine Learnin repository. Plots are of generalization error vs. m (averaged over 1000 rando train/test splits). Dashed line is logistic regression; solid line is naive Bayes.

What you should know about Logistic Regression (LR)

- Gaussian Naïve Bayes with class-independent variances representationally equivalent to LR
  - Solution differs because of objective (loss) function
- In general, NB and LR make different assumptions
  - NB: Features independent given class → assumption on P(**X**|Y)
  - LR: Functional form of P(Y|**X**), no assumption on P(**X**|Y)
- LR is a linear classifier
  - decision rule is a hyperplane
- LR optimized by conditional likelihood
  - no closed-form solution
  - concave → global optimum with gradient ascent
  - Maximum conditional a posteriori corresponds to regularization
- Convergence rates
  - GNB (usually) needs less data
  - LR (usually) gets to better solutions in the limit

14

©Carlos Guestrin 2005-2007

Linear separability

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

©Carlos Guestrin 2005-2007

Not linearly separable data

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

©Carlos Guestrin 2005-2007

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 **w**
  - As easy to learn
  - Data is linearly separable in higher dimensional spaces
  - More discussion later this semester

©Carlos Guestrin 2005-2007

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

- Choose a classifier $h_{\mathbf{w}}(\mathbf{x})$ that is non-linear in parameters **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)

©Carlos Guestrin 2005-2007

A small dataset: Miles Per Gallon

Suppose we want to predict MPG

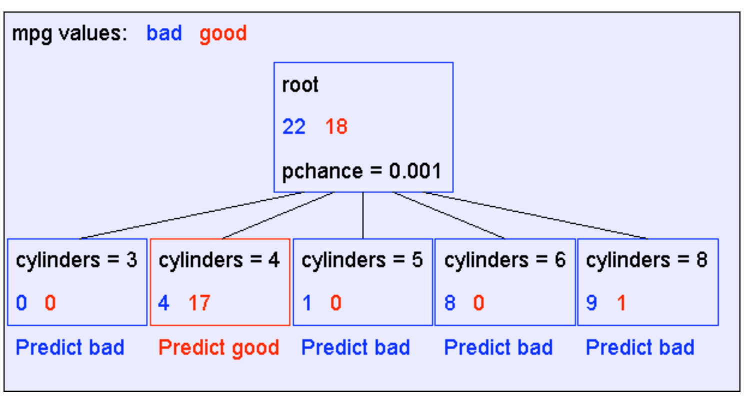
mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	75to78	asia
bad	6	medium	medium	medium	medium	70to74	america
bad	4	medium	medium	medium	low	75to78	europe
bad	8	high	high	high	low	70to74	america
bad	6	medium	medium	medium	medium	70to74	america
bad	4	low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

40 Records

From the UCI repository (thanks to Ross Quinlan)

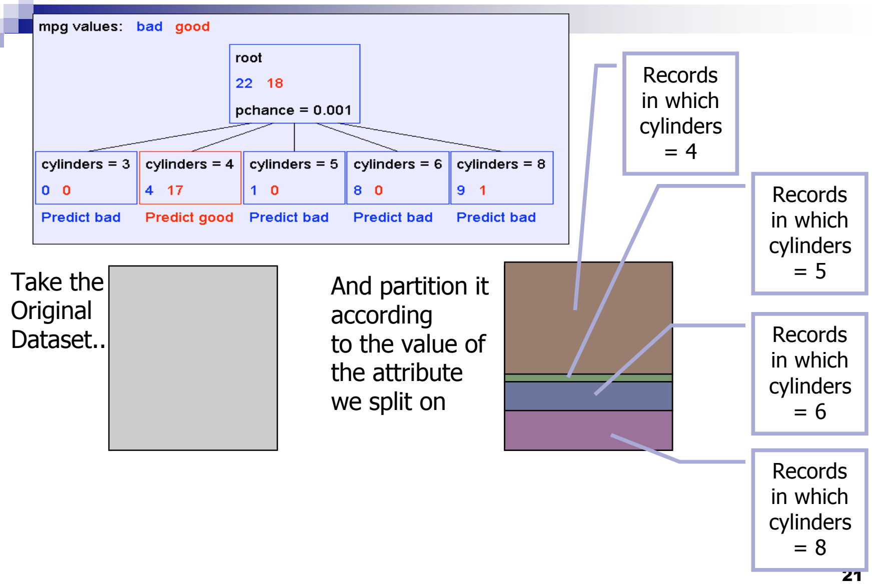
©Carlos Guestrin 2005-2007

A Decision Stump



©Carlos Guestrin 2005-2007

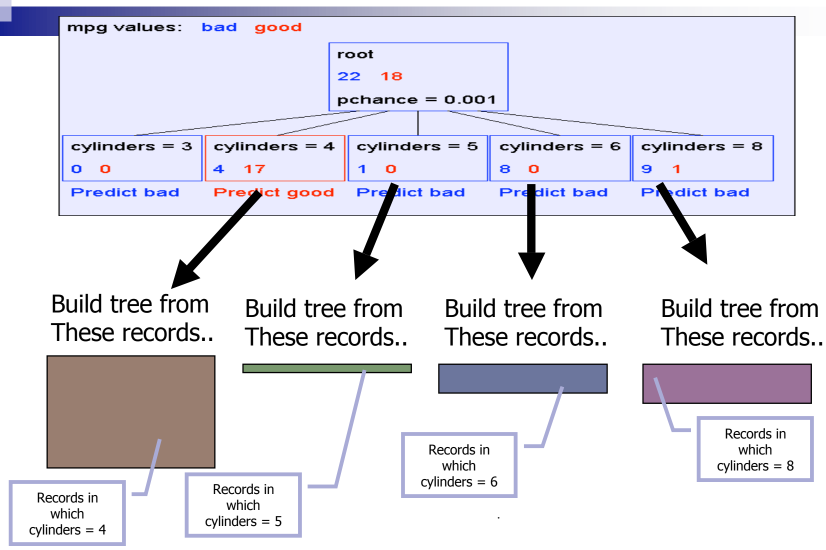
Recursion Step



21

©Carlos Guestrin 2005-2007

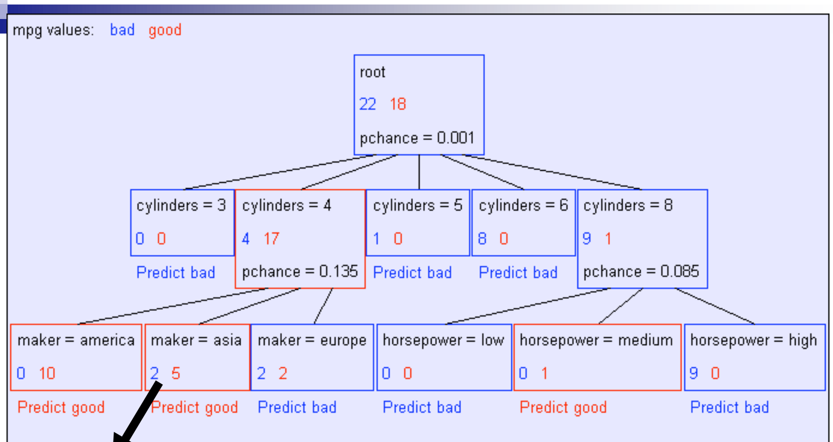
Recursion Step



22

©Carlos Guestrin 2005-2007

Second level of tree

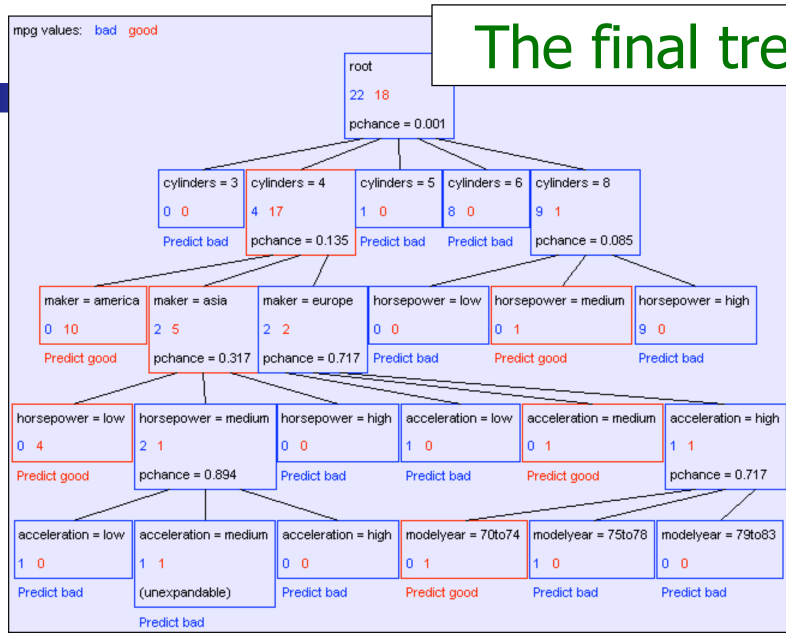


Recursively build a tree from the seven records in which there are four cylinders and the maker was based in Asia

(Similar recursion in the other cases)

©Carlos Guestrin 2005-2007

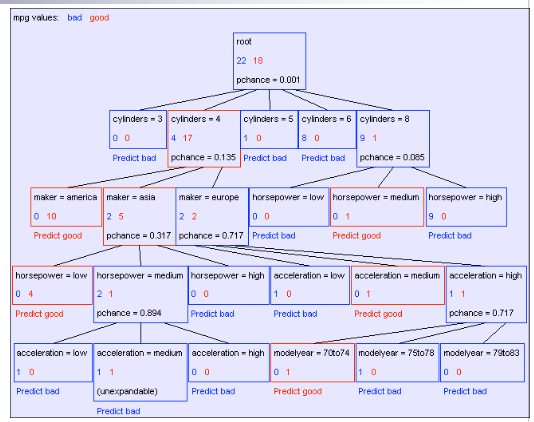
The final tree



©Carlos Guestrin 2005-2007

Classification of a new example

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



©Carlos Guestrin 2005-2007

Announcements

- **Pittsburgh won the Super Bowl !!**
  - Two years ago…
- Recitation this Thursday
  - Logistic regression, discriminative v. generative

©Carlos Guestrin 2005-2007

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))

©Carlos Guestrin 2005-2007

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

©Carlos Guestrin 2005-2007

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

©Carlos Guestrin 2005-2007

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

©Carlos Guestrin 2005-2007

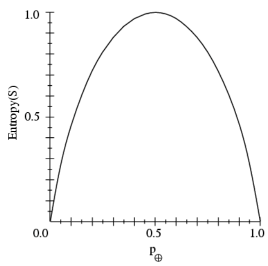
Entropy

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)



31

©Carlos Guestrin 2005-2007

Andrew Moore's Entropy in a nutshell

Low Entropy

High Entropy

32

©Carlos Guestrin 2005-2007

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

33

©Carlos Guestrin 2005-2007

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

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

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

34

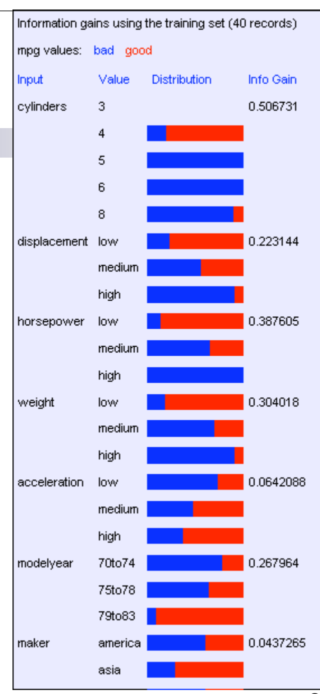
©Carlos Guestrin 2005-2007

Learning decision trees

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

©Carlos Guestrin 2005-2007

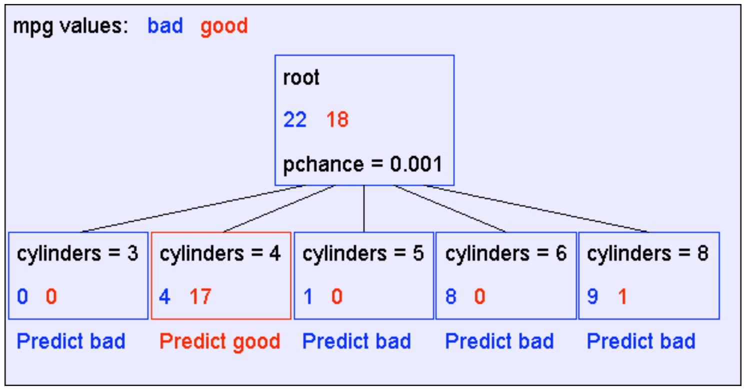
Suppose we want to predict MPG



Look at all the information gains…

©Carlos Guestrin 2005-2007

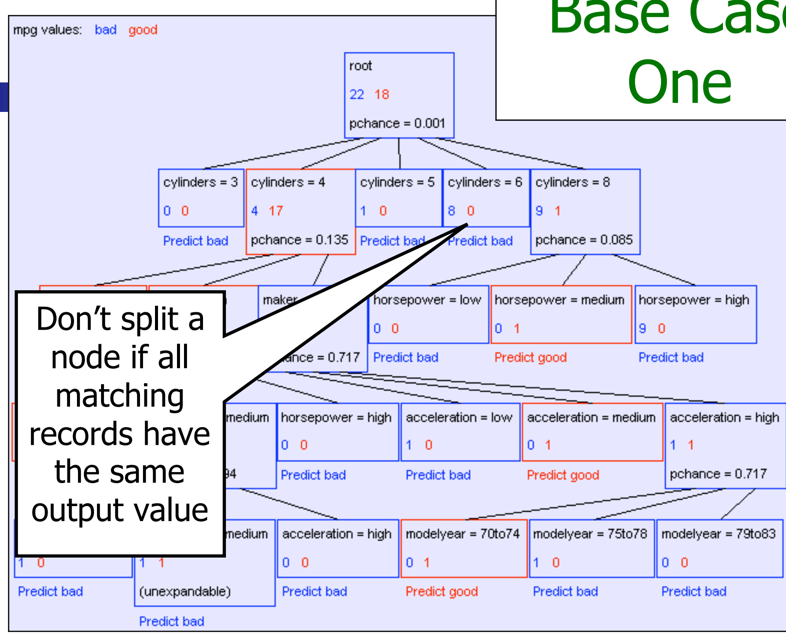
A Decision Stump



37

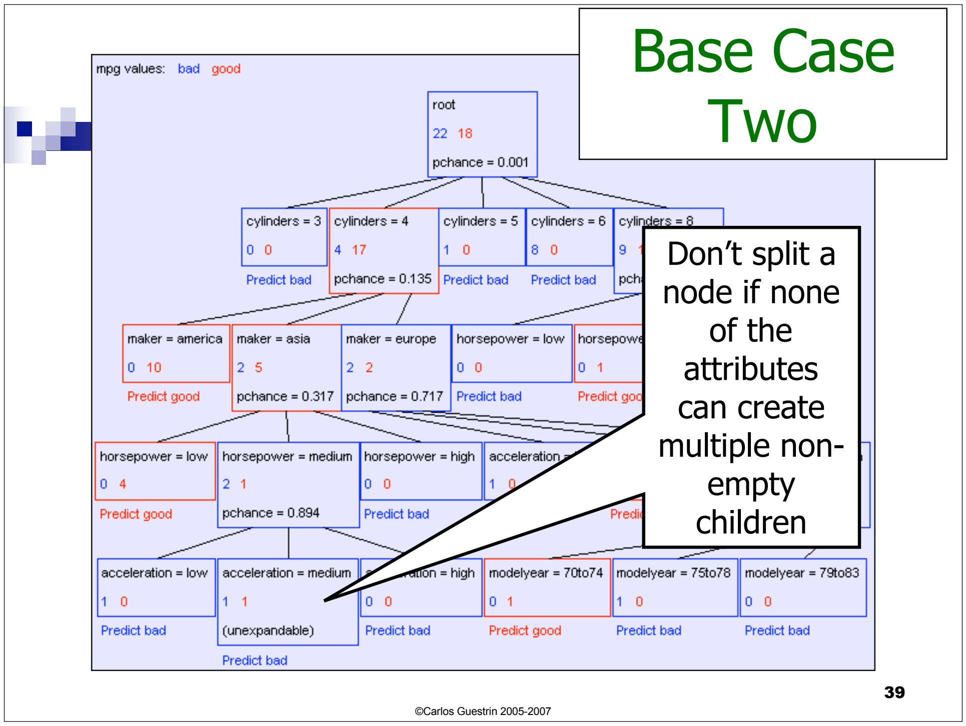
©Carlos Guestrin 2005-2007

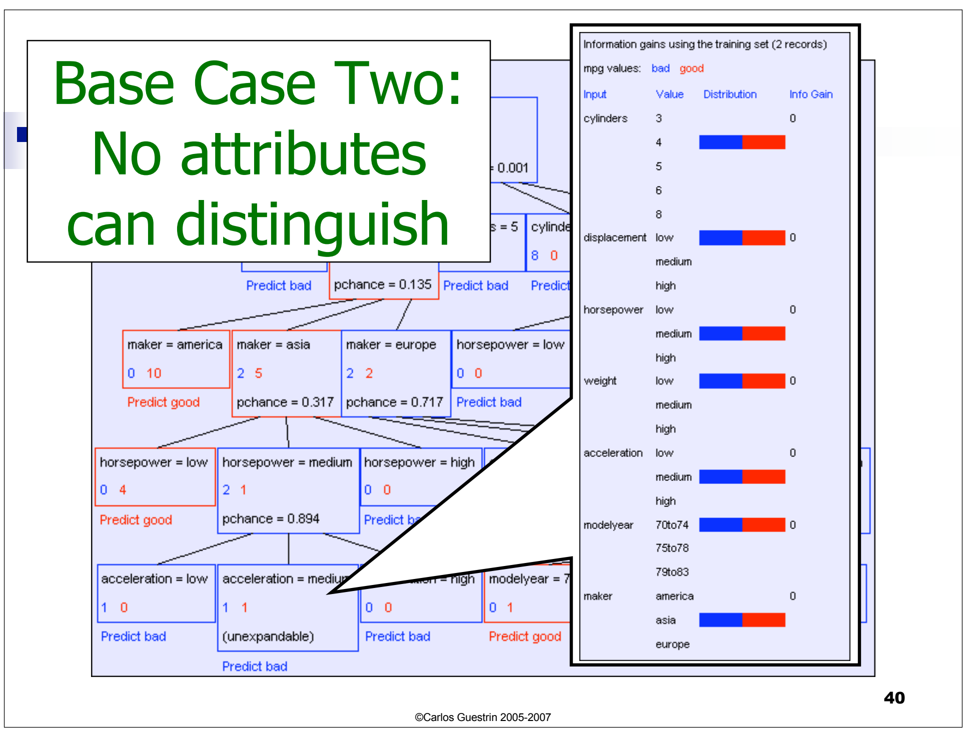
Base Case One



38

©Carlos Guestrin 2005-2007


Base Case Two
Don't split a node if none of the attributes can create multiple non-empty children
mpg values: bad good
root
22 18
pchance = 0.001
cylinders = 3
0 0
Predict bad
cylinders = 4
4 17
pchance = 0.135
cylinders = 5
1 0
Predict bad
cylinders = 6
8 0
Predict bad
cylinders = 8
maker = america
0 10
Predict good
maker = asia
2 5
pchance = 0.317
maker = europe
2 2
pchance = 0.717
horsepower = low
0 0
Predict bad
horsepower = low
0 4
Predict good
horsepower = medium
2 1
pchance = 0.894
horsepower = high
0 0
Predict bad
acceleration = low
1 0
Predict bad
acceleration = medium
1 1
(unexpandable)
Predict bad
0 0
Predict bad
modelyear = 70to74
0 1
Predict good
modelyear = 75to78
1 0
Predict bad
modelyear = 79to83
0 0
Predict bad
39
©Carlos Guestrin 2005-2007


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
pchance = 0.135
Predict bad
maker = america
0 10
Predict good
maker = asia
2 5
pchance = 0.317
maker = europe
2 2
pchance = 0.717
horsepower = low
0 0
Predict bad
horsepower = low
0 4
Predict good
horsepower = medium
2 1
pchance = 0.894
horsepower = high
0 0
acceleration = low
1 0
Predict bad
1 1
(unexpandable)
Predict bad
0 0
Predict bad
0 1
Predict good
40
©Carlos Guestrin 2005-2007

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

41

©Carlos Guestrin 2005-2007

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?*

42

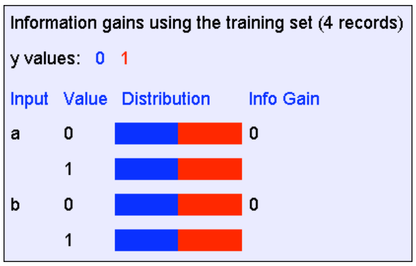
©Carlos Guestrin 2005-2007

The problem with Base Case 3

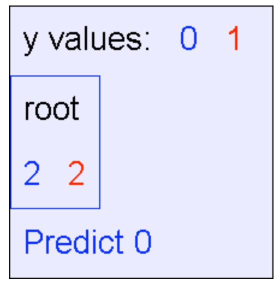
a	b	y
0	0	0
0	1	1
1	0	1
1	1	0

y = a XOR b

The information gains:



The resulting decision tree:



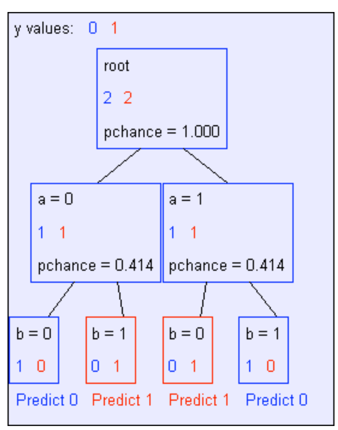
©Carlos Guestrin 2005-2007

If we omit Base Case 3:

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

y = a XOR b

The resulting decision tree:



©Carlos Guestrin 2005-2007

Basic Decision Tree Building Summarized

BuildTree(*DataSet,Output*)

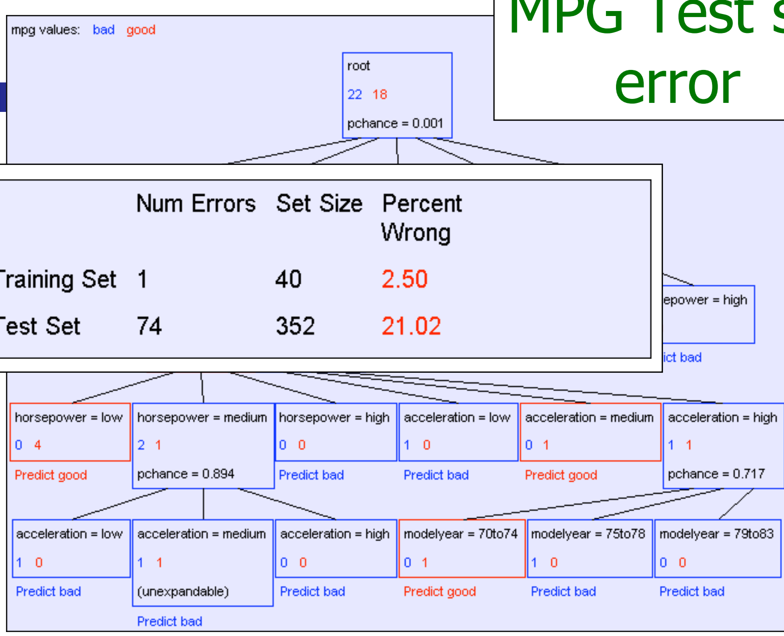
- If all output values are the same in *DataSet*, return a leaf node that says "predict this unique output"
- If all input values are the same, return a leaf node that says "predict the majority output"
- Else find attribute *X* with highest Info Gain
- Suppose *X* has n_X distinct values (i.e. X has 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.

©Carlos Guestrin 2005-2007

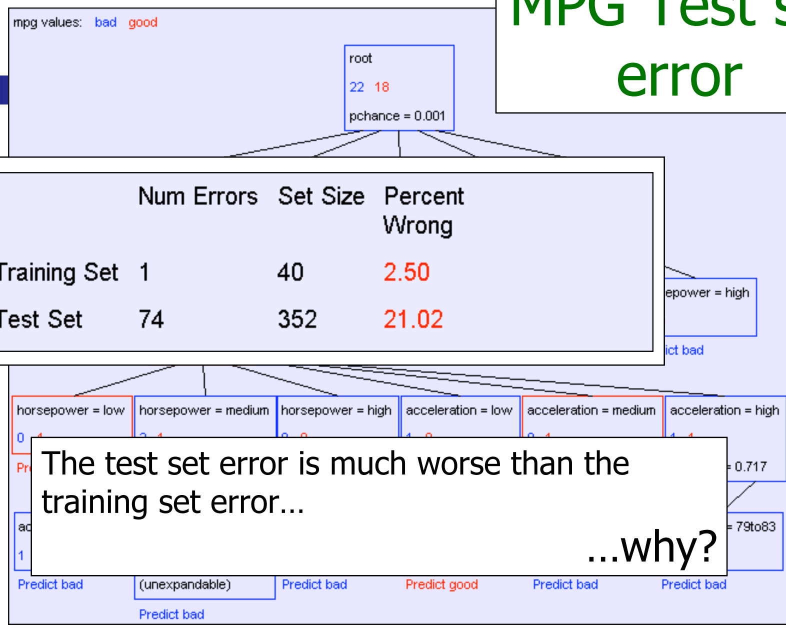
MPG Test set error



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

©Carlos Guestrin 2005-2007

MPG Test set error



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

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

…why?

©Carlos Guestrin 2005-2007

Decision trees & Learning Bias

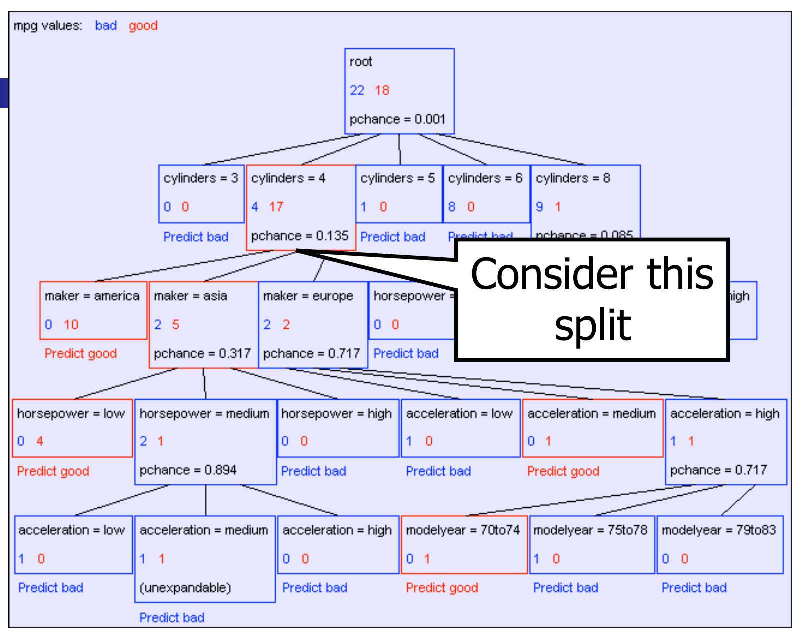
mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	75to78	asia
bad	6	medium	medium	medium	medium	70to74	america
bad	4	medium	medium	medium	low	75to78	europe
bad	8	high	high	high	low	70to74	america
bad	6	medium	medium	medium	medium	70to74	america
bad	4	low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

©Carlos Guestrin 2005-2007

Decision trees will overfit

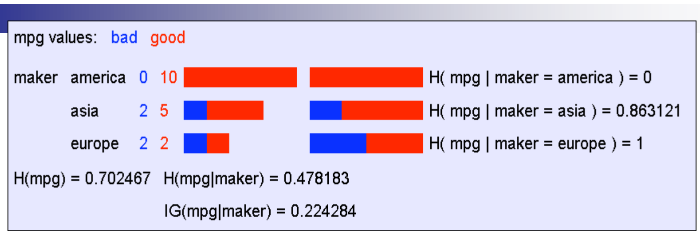
- Standard decision trees are have no learning biased
  - Training set error is always zero!
    - (If there is no label noise)
  - Lots of variance
  - Will definitely overfit!!!
  - Must bias towards simpler trees
- Many strategies for picking simpler trees:
  - Fixed depth
  - Fixed number of leaves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

©Carlos Guestrin 2005-2007



©Carlos Guestrin 2005-2007

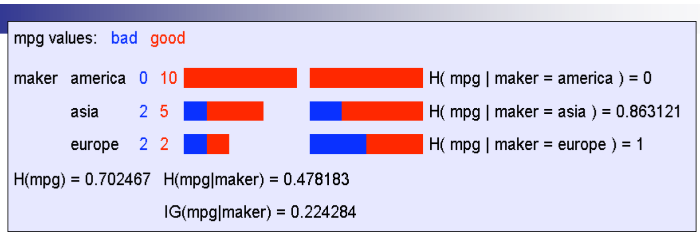
A chi-square test



- 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?

©Carlos Guestrin 2005-2007

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?

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! :))

©Carlos Guestrin 2005-2007

Using Chi-squared to avoid overfitting

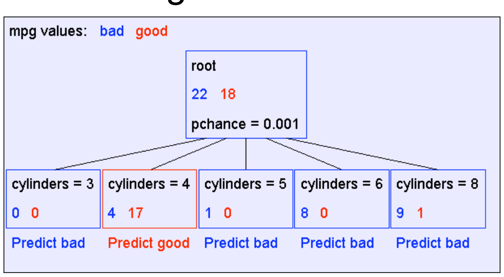
- Build the full decision tree as before
- But when you can grow it no more, start to prune:
  - Beginning at the bottom of the tree, delete splits in which p_{chance} > *MaxPchance*
  - Continue working you way up until there are no more prunable nodes

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

©Carlos Guestrin 2005-2007

Pruning example

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



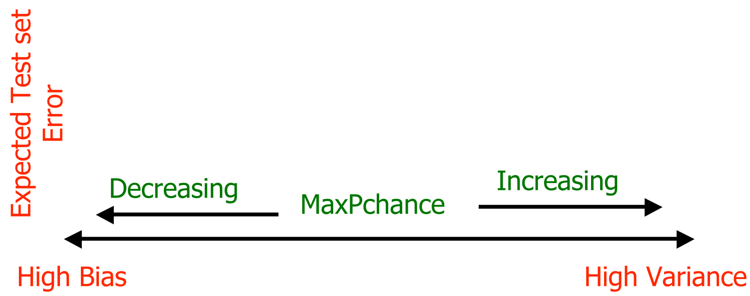
Note the improved test set accuracy compared with the unpruned tree

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

©Carlos Guestrin 2005-2007

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!

55

©Carlos Guestrin 2005-2007

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	europe
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	europe
bad	5	131	103	2830	15.9	78	europe

Infinite number of possible split values!!!

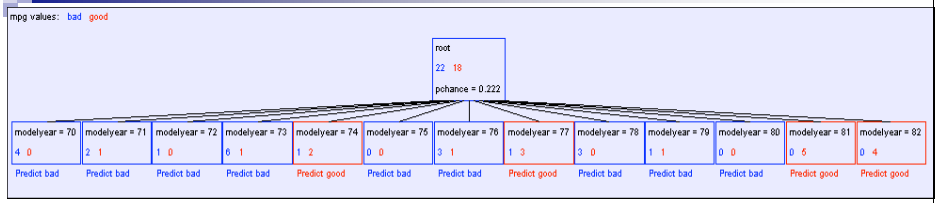
Finite dataset, only finite number of relevant splits!

Idea One: Branch on each possible real value

56

©Carlos Guestrin 2005-2007

“One branch for each numeric value” idea:



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

©Carlos Guestrin 2005-2007

Threshold splits

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

©Carlos Guestrin 2005-2007

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 $\{x_1,\ldots,x_m\}$
  - Consider split points of the form $x_i + (x_{i+1} - x_i)/2$

©Carlos Guestrin 2005-2007

A better idea: thresholded splits

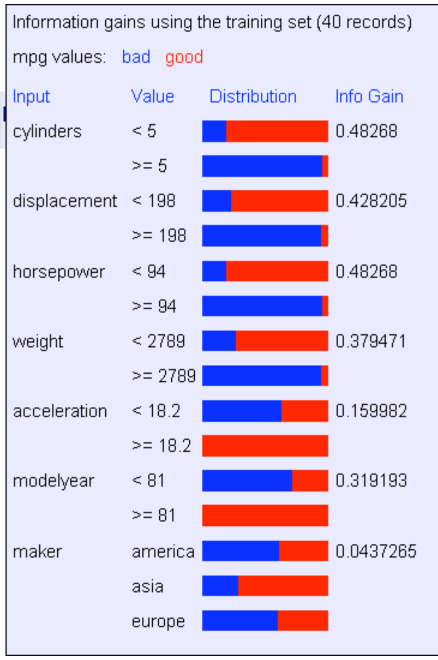
- Suppose X is real valued
- Define *IG(Y|X:t)* as *H(Y) - H(Y|X:t)*
- Define *H(Y|X:t)* =

  $H(Y|X < t)\ P(X < t) + H(Y|X >= t)\ P(X >= t)$

  - *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
- Note, may split on an attribute multiple times, with different thresholds

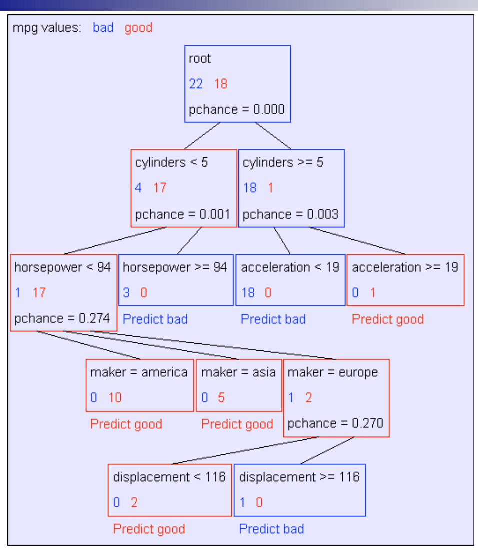
©Carlos Guestrin 2005-2007

Example with MPG



©Carlos Guestrin 2005-2007

Example tree using reals



©Carlos Guestrin 2005-2007

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

©Carlos Guestrin 2005-2007

Acknowledgements

- Some of the material in the decision trees presentation is courtesy of Andrew Moore, from his excellent collection of ML tutorials:
  - http://www.cs.cmu.edu/~awm/tutorials

©Carlos Guestrin 2005-2007