

# Logistic Regression (Continued) Generative v. Discriminative Decision Trees

Machine Learning – 10701/15781  
Carlos Guestrin  
Carnegie Mellon University

January 31<sup>st</sup>, 2007

©2005-2007 Carlos Guestrin

1

## Generative v. Discriminative classifiers – Intuition

- **Want to Learn:**  $h: X \mapsto Y$

- **X** – features
- **Y** – target classes

- **Bayes optimal classifier** –  $P(Y|X)$

- **Generative classifier**, e.g., Naïve Bayes:

- Assume some **functional form for  $P(X|Y), P(Y)$**
- Estimate parameters of  $P(X|Y), P(Y)$  directly from training data
- Use **Bayes rule** to calculate  $P(Y|X=x) \propto P(Y) \cdot P(X=x|Y)$
- This is a **'generative' model**

$P(Y)$   
 $P(X|Y)$   
sample image  
first sample  
clear  
more sample  
pixel values

- **Indirect** computation of  $P(Y|X)$  through Bayes rule
- But, **can generate a sample of the data**,  $P(X) = \sum_y P(y) P(X|y)$

- **Discriminative classifiers**, e.g., Logistic Regression:

- Assume some **functional form for  $P(Y|X)$**
- Estimate parameters of  $P(Y|X)$  directly from training data
- This is the **'discriminative' model**
- Directly learn  $P(Y|X)$
- But **cannot obtain a sample of the data**, because  $P(X)$  is not available

if you have  
 $h_{\text{Bayes}}(x) = \arg \max_y P(Y|X=x)$

MB ← table

if you want to classify → have use

directly model  $p(y|x)$

discriminate classes, e.g.,  
persons v. animals

©2005-2007 Carlos Guestrin

2

# Logistic Regression

Logistic function (or Sigmoid):

$$\frac{1}{1 + \exp(-z)}$$

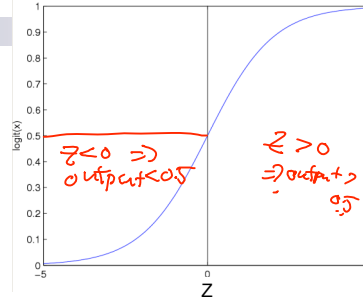
## Learn $P(Y|X)$ directly!

- Assume a particular functional form
- Sigmoid applied to a linear function of the data:

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

linear function of  $X$ , with parameters  $w$

$$P(Y=0|X) = 1 - P(Y=1|X)$$

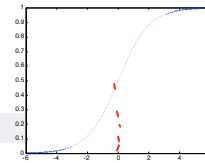


**Features can be discrete or continuous!**

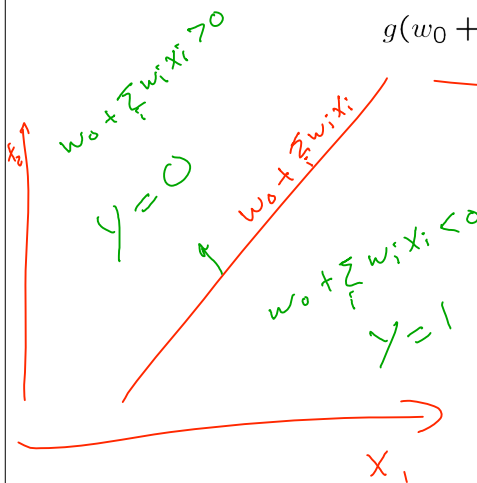
©2005-2007 Carlos Guestrin

3

# Logistic Regression – a Linear classifier



$$g(w_0 + \sum_i w_i x_i) = \frac{1}{1 + e^{-(w_0 + \sum_i w_i x_i)}}$$



$$P(Y=1|X) > 0.5$$

when

$$-(w_0 + \sum_i w_i x_i) > 0$$

$$\Leftrightarrow w_0 + \sum_i w_i x_i < 0$$

$$P(Y=0|X) > 0.5$$

•  $w_0 + \sum_i w_i x_i > 0$

©2005-2007 Carlos Guestrin

4

## Very convenient!

$$P(Y = 1|X = \langle X_1, \dots, X_n \rangle) = \frac{1}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

implies

$$P(Y = 0|X = \langle X_1, \dots, X_n \rangle) = \frac{\exp(w_0 + \sum_i w_i X_i)}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

implies

$$\frac{P(Y = 0|X)}{P(Y = 1|X)} = \exp(w_0 + \sum_i w_i X_i)$$

implies

$$\ln \frac{P(Y = 0|X)}{P(Y = 1|X)} = w_0 + \sum_i w_i X_i$$

linear  
classification  
rule!

©2005-2007 Carlos Guestrin

5

## Logistic regression v. Naïve Bayes

- Consider learning  $f: X \rightarrow Y$ , where
  - $X$  is a vector of real-valued features,  $\langle X_1 \dots X_n \rangle$
  - $Y$  is boolean
- Could use a Gaussian Naïve Bayes classifier
  - assume all  $X_i$  are conditionally independent given  $Y$
  - model  $P(X_i | Y = y_k)$  as Gaussian  $N(\mu_{ik}, \sigma_i)$
  - model  $P(Y)$  as Bernoulli( $\theta, 1-\theta$ )
- What does that imply about the form of  $P(Y|X)$ ?

$$P(Y = 1|X = \langle X_1, \dots, X_n \rangle) = \frac{1}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

Cool!!!!

©2005-2007 Carlos Guestrin

6

## Derive form for $P(Y|X)$ for continuous $X_i$

$$\begin{aligned} P(Y = 1|X) &= \frac{P(Y = 1)P(X|Y = 1)}{P(Y = 1)P(X|Y = 1) + P(Y = 0)P(X|Y = 0)} \\ &= \frac{1}{1 + \frac{P(Y=0)P(X|Y=0)}{P(Y=1)P(X|Y=1)}} \\ &= \frac{1}{1 + \exp(\ln \frac{P(Y=0)P(X|Y=0)}{P(Y=1)P(X|Y=1)})} \\ &= \frac{1}{1 + \exp(\ln \frac{1-\theta}{\theta} + \sum_i \ln \frac{P(X_i|Y=0)}{P(X_i|Y=1)})} \end{aligned}$$

©2005-2007 Carlos Guestrin

7

## Ratio of class-conditional probabilities

$$\ln \frac{P(X_i|Y = 0)}{P(X_i|Y = 1)}$$

$$P(X_i = x | Y = y_k) = \frac{1}{\sigma_i \sqrt{2\pi}} e^{-\frac{(x - \mu_{ik})^2}{2\sigma_i^2}}$$

©2005-2007 Carlos Guestrin

8

## Derive form for $P(Y|X)$ for continuous $X_i$

$$\begin{aligned}
 P(Y = 1|X) &= \frac{P(Y = 1)P(X|Y = 1)}{P(Y = 1)P(X|Y = 1) + P(Y = 0)P(X|Y = 0)} \\
 &= \frac{1}{1 + \exp\left(\ln \frac{1-\theta}{\theta} + \sum_i \ln \frac{P(X_i|Y=0)}{P(X_i|Y=1)}\right)} \\
 &= \frac{1}{1 + \exp\left(w_0 + \sum_{i=1}^n w_i X_i\right)}
 \end{aligned}$$

©2005-2007 Carlos Guestrin

9

## Gaussian Naïve Bayes v. Logistic Regression

**Set of Gaussian Naïve Bayes parameters (feature variance independent of class label)**

**Set of Logistic Regression parameters**

- Representation equivalence
  - **But only in a special case!!!** (GNB with class-independent variances)
- But what's the difference???
- **LR makes no assumptions about  $P(X|Y)$  in learning!!!**
- **Loss function!!!**
  - Optimize different functions → Obtain different solutions

©2005-2007 Carlos Guestrin

10

## Logistic regression for more than 2 classes

- Logistic regression in more general case, where  $Y \in \{Y_1 \dots Y_R\}$  : learn  $R-1$  sets of weights

## Logistic regression more generally

- Logistic regression in more general case, where  $Y \in \{Y_1 \dots Y_R\}$  : learn  $R-1$  sets of weights

for  $k < R$

$$P(Y = y_k | X) = \frac{\exp(w_{k0} + \sum_{i=1}^n w_{ki} X_i)}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^n w_{ji} X_i)}$$

for  $k=R$  (normalization, so no weights for this class)

$$P(Y = y_R | X) = \frac{1}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^n w_{ji} X_i)}$$

**Features can be discrete or continuous!**

# Announcements

- Don't forget recitation tomorrow
- And start the homework early

# Loss functions: Likelihood v. Conditional Likelihood

- Generative (Naïve Bayes) Loss function:

## Data likelihood

$$\begin{aligned}\ln P(\mathcal{D} | \mathbf{w}) &= \sum_{j=1}^N \ln P(\mathbf{x}^j, y^j | \mathbf{w}) \\ &= \sum_{j=1}^N \ln P(y^j | \mathbf{x}^j, \mathbf{w}) + \sum_{j=1}^N \ln P(\mathbf{x}^j | \mathbf{w})\end{aligned}$$

- Discriminative models cannot compute  $P(\mathbf{x} | \mathbf{w})!$
- But, discriminative (logistic regression) loss function:

## Conditional Data Likelihood

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

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

## Expressing Conditional Log Likelihood

$$l(\mathbf{w}) \equiv \sum_j \ln P(y^j | \mathbf{x}^j, \mathbf{w})$$

$$P(Y = 0 | \mathbf{X}, \mathbf{w}) = \frac{1}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

$$P(Y = 1 | \mathbf{X}, \mathbf{w}) = \frac{\exp(w_0 + \sum_i w_i X_i)}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

$$l(\mathbf{w}) = \sum_j y^j \ln P(y^j = 1 | \mathbf{x}^j, \mathbf{w}) + (1 - y^j) \ln P(y^j = 0 | \mathbf{x}^j, \mathbf{w})$$

©2005-2007 Carlos Guestrin

15

## Maximizing Conditional Log Likelihood

$$l(\mathbf{w}) \equiv \ln \prod_j P(y^j | \mathbf{x}^j, \mathbf{w})$$

$$P(Y = 0 | X, W) = \frac{1}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

$$P(Y = 1 | X, W) = \frac{\exp(w_0 + \sum_i w_i X_i)}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

$$= \sum_j y^j (w_0 + \sum_i w_i x_i^j) - \ln(1 + \exp(w_0 + \sum_i w_i x_i^j))$$

**Good news:**  $l(\mathbf{w})$  is concave function of  $\mathbf{w}$  → no locally optimal solutions

**Bad news:** no closed-form solution to maximize  $l(\mathbf{w})$

**Good news:** concave functions easy to optimize

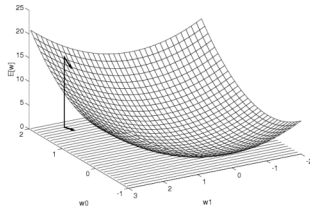
©2005-2007 Carlos Guestrin

16



## Optimizing concave function – Gradient ascent

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



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

Learning rate,  $\eta > 0$

**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)

©2005-2007 Carlos Guestrin

17

## Maximize Conditional Log Likelihood: Gradient ascent

$$l(\mathbf{w}) = \sum_j y^j (w_0 + \sum_i w_i x_i^j) - \ln(1 + \exp(w_0 + \sum_i w_i x_i^j))$$

©2005-2007 Carlos Guestrin

18

# Gradient Descent for LR

Gradient ascent algorithm: iterate until change  $< \epsilon$

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

For  $i = 1 \dots n$ ,

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

repeat

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

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

- One common approach is to define priors on  $\mathbf{w}$ 
  - 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

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

## M(C)AP as Regularization

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

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

Penalizes high weights, also applicable in linear regression

©2005-2007 Carlos Guestrin

21

## Gradient of M(C)AP

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

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

©2005-2007 Carlos Guestrin

22

## MLE vs MAP

- Maximum conditional likelihood estimate

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

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

- Maximum conditional a posteriori estimate

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

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

©2005-2007 Carlos Guestrin

23

## Naïve Bayes vs Logistic Regression

Consider  $Y$  boolean,  $X_i$  continuous,  $\mathbf{X} = \langle X_1 \dots X_n \rangle$

Number of parameters:

- NB:  $4n + 1$
- LR:  $n + 1$

Estimation method:

- NB parameter estimates are uncoupled
- LR parameter estimates are coupled

©2005-2007 Carlos Guestrin

24

## G. Naïve Bayes vs. Logistic Regression 1

[Ng & Jordan, 2002]

- Generative and Discriminative classifiers
  
- Asymptotic comparison (# training examples  $\rightarrow$  infinity)
  - when 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**

©2005-2007 Carlos Guestrin

25

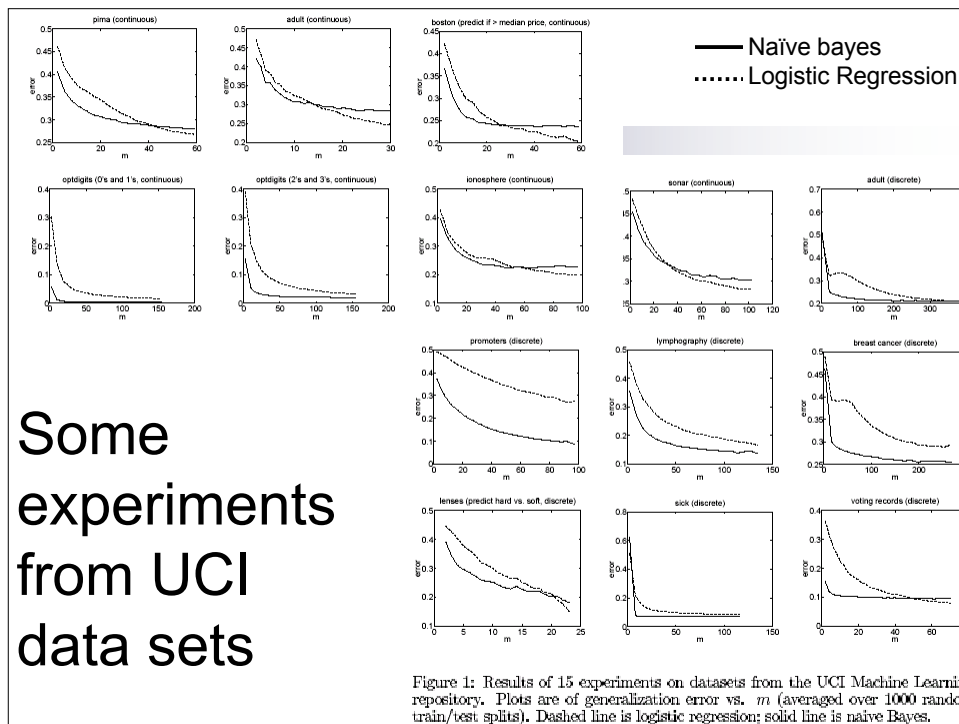
## G. Naïve Bayes vs. Logistic Regression 2

[Ng & Jordan, 2002]

- Generative and Discriminative classifiers
  
- 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**

©2005-2007 Carlos Guestrin

26



## 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  $\rightarrow$  assumption on  $P(\mathbf{X}|Y)$
  - LR: Functional form of  $P(Y|\mathbf{X})$ , no assumption on  $P(\mathbf{X}|Y)$
- LR is a linear classifier
  - decision rule is a hyperplane
- LR optimized by conditional likelihood
  - no closed-form solution
  - concave  $\rightarrow$  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

## 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  $\mathbf{w}$ 
  - Usually easy to learn (closed-form or convex/concave optimization)
  - Data is linearly separable in higher dimensional spaces
  - More discussion later this semester

©2005-2007 Carlos Guestrin

31

## 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)

©2005-2007 Carlos Guestrin

32



# 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	78to83	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

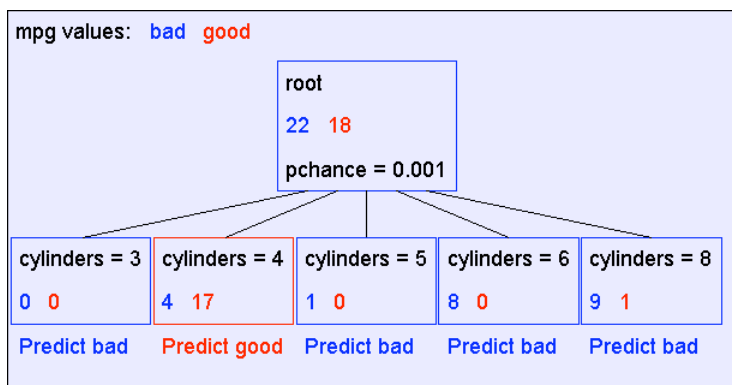
40 Records

From the UCI repository (thanks to Ross Quinlan)

©2005-2007 Carlos Guestrin

33

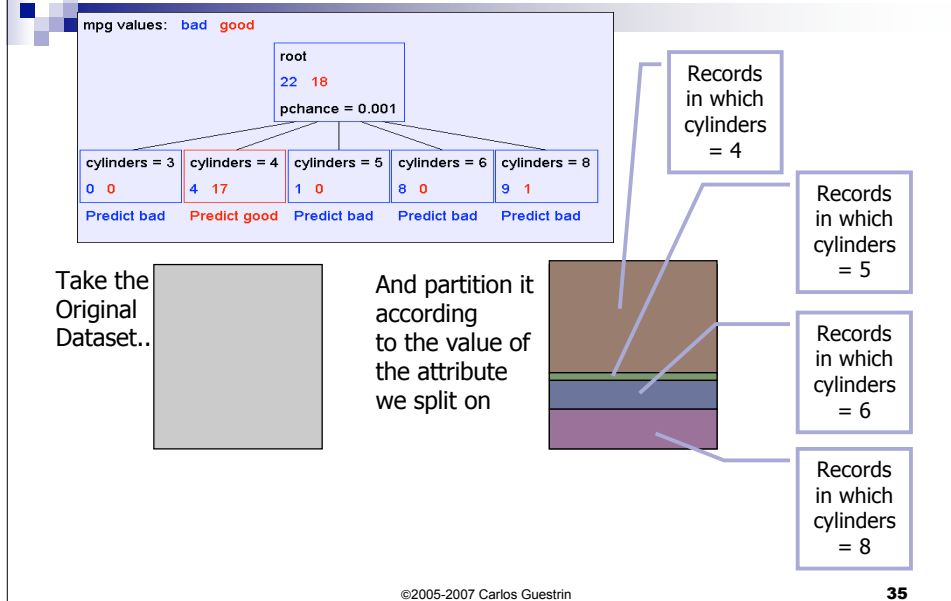
# A Decision Stump



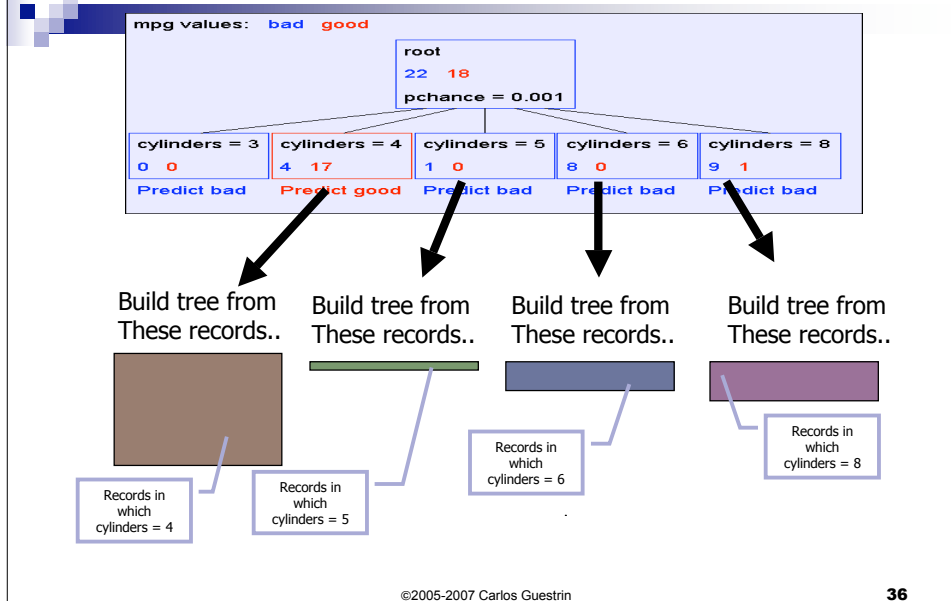
©2005-2007 Carlos Guestrin

34

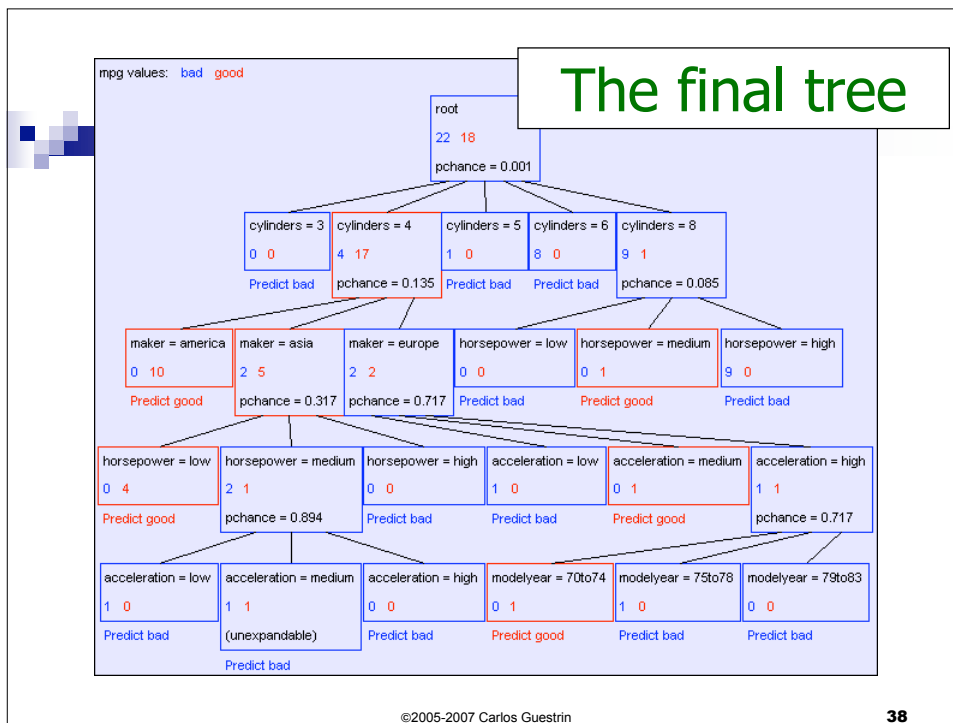
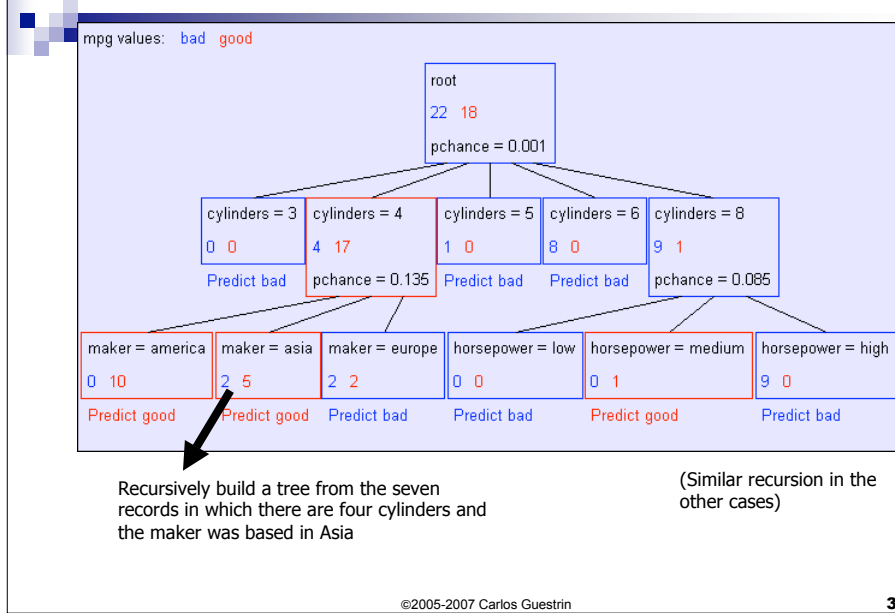
# Recursion Step



# Recursion Step

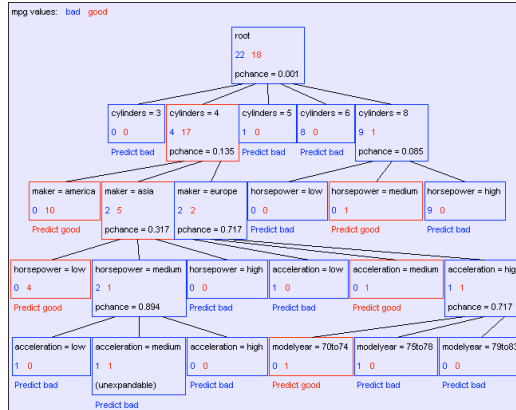


# Second level of tree



# Classification of a new example

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



©2005-2007 Carlos Guestrin

39

# 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))

©2005-2007 Carlos Guestrin

40

# 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

©2005-2007 Carlos Guestrin

41

## 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

©2005-2007 Carlos Guestrin

42

# Measuring uncertainty

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

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

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

©2005-2007 Carlos Guestrin

43

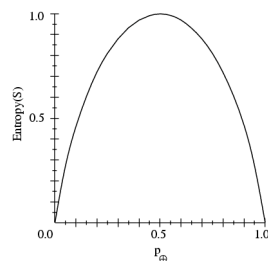
# Entropy

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)



©2005-2007 Carlos Guestrin

44

# Andrew Moore's Entropy in a nutshell



Low Entropy



High Entropy

©2005-2007 Carlos Guestrin

45

# Andrew Moore's Entropy in a nutshell



Low Entropy



High Entropy

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

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

©2005-2007 Carlos Guestrin

46

# Information gain

X <sub>1</sub>	X <sub>2</sub>	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)**

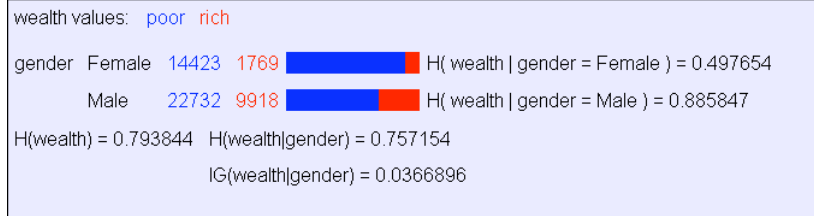
- Use, for example, information gain to select attribute

- Split on  $\arg \max_i IG(X_i) = \arg \max_i H(Y) - H(Y | X_i)$

- Recurse

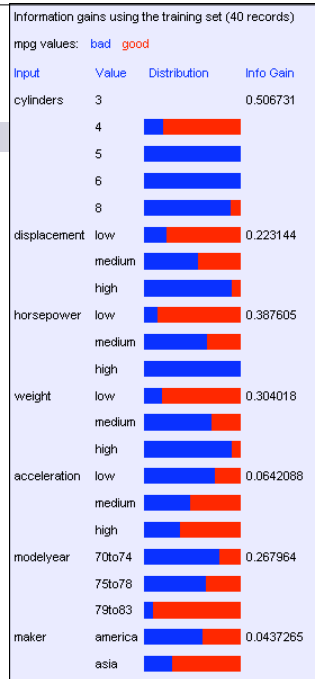


# Information Gain Example

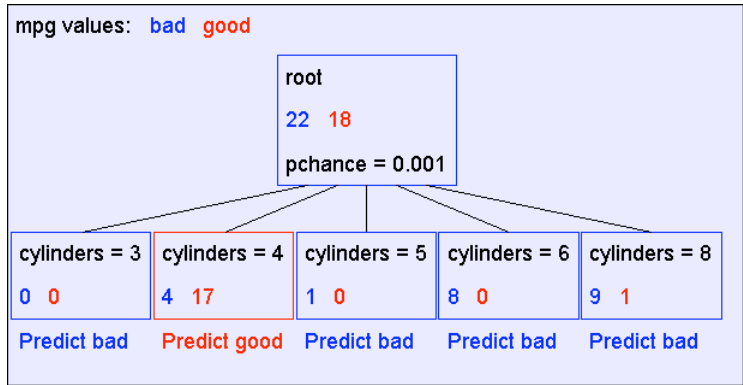


Suppose we want to predict MPG

Look at all the information gains...

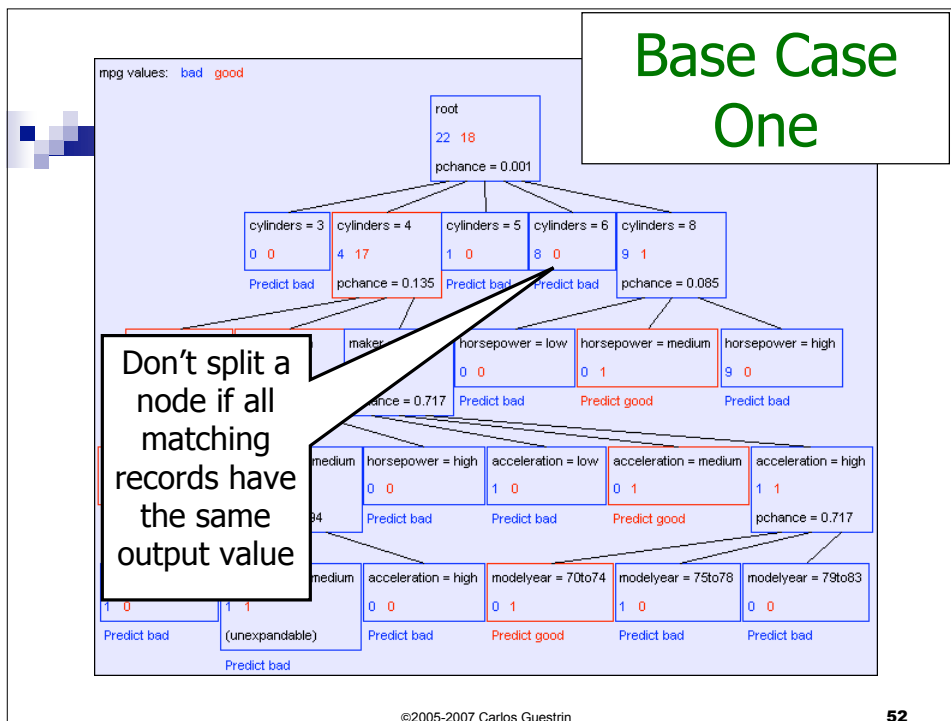


# A Decision Stump



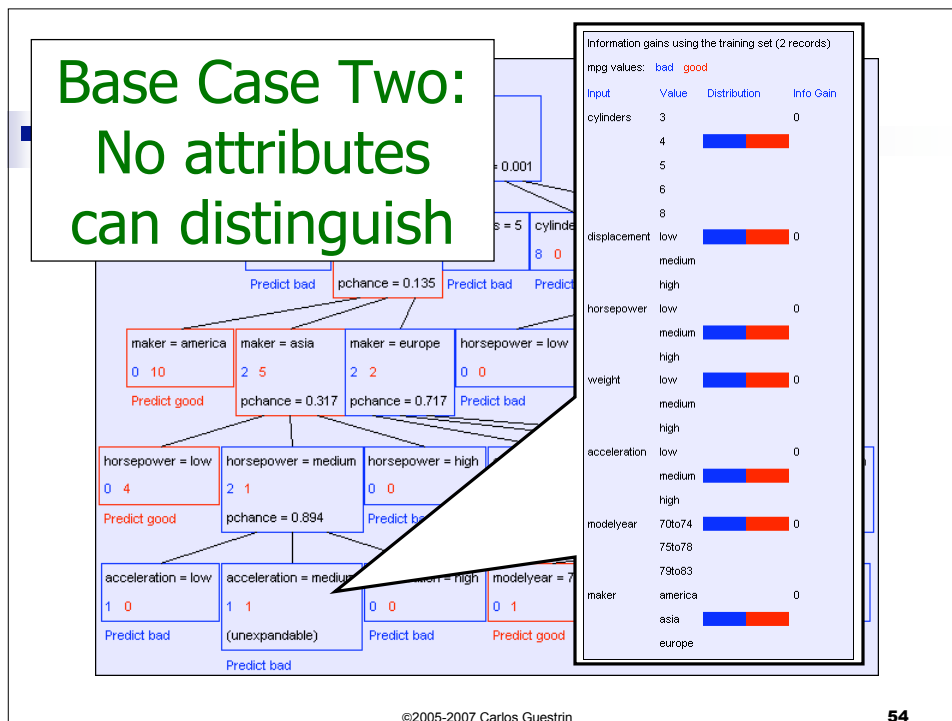
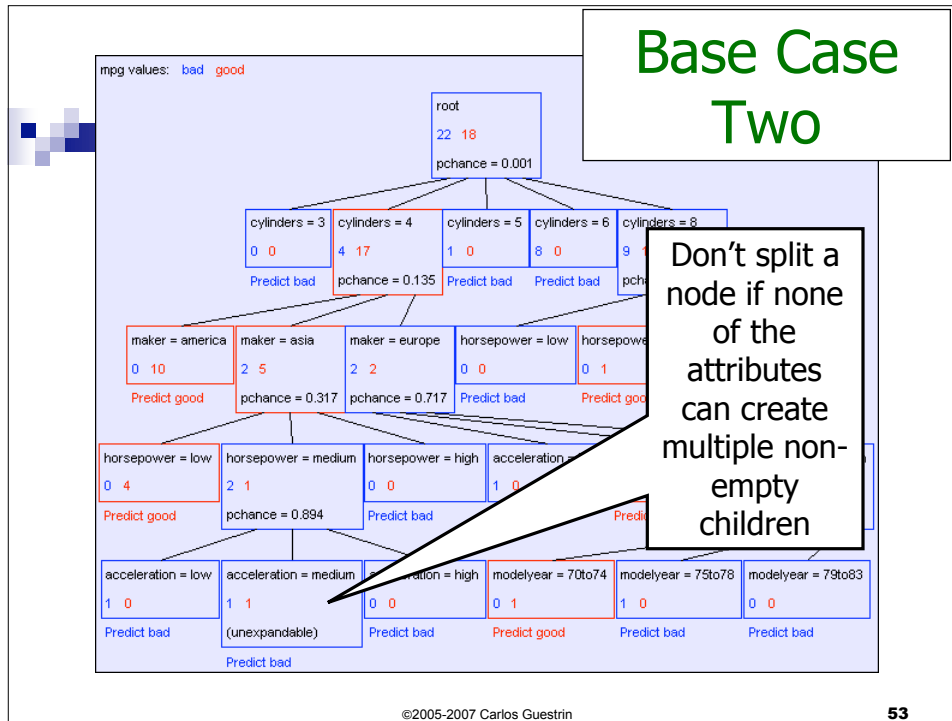
©2005-2007 Carlos Guestrin

51



©2005-2007 Carlos Guestrin

52

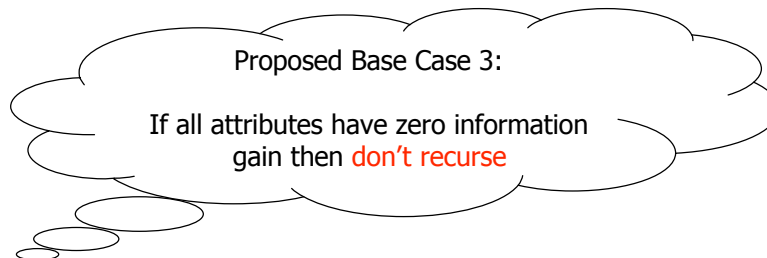


# Base Cases

- 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**



• *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

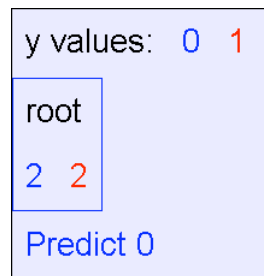
$$y = a \text{ XOR } b$$

The information gains:

Information gains using the training set (4 records)  
y values: 0 1

Input	Value	Distribution	Info Gain
a	0		0
	1		
b	0		0
	1		

The resulting decision tree:



©2005-2007 Carlos Guestrin

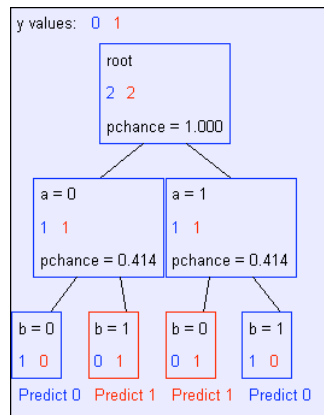
57

# 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:



©2005-2007 Carlos Guestrin

58

# 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.

# Real-Valued inputs

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

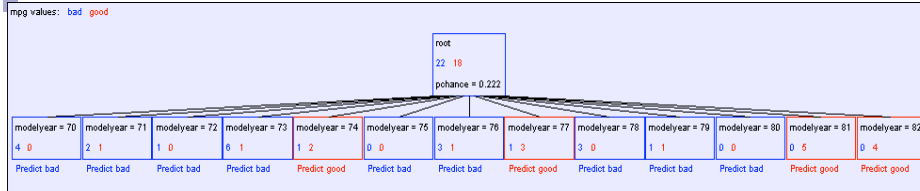
mpg	cylinders	displacemen	horsepower	weight	acceleration	modelyear	maker
good	4	97	75	2265	18.2	77	asia
bad	6	199	90	2648	15	70	america
bad	4	121	110	2600	12.8	77	europa
bad	8	350	175	4100	13	73	america
bad	6	198	95	3102	16.5	74	america
bad	4	108	94	2379	16.5	73	asia
bad	4	113	95	2228	14	71	asia
bad	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

©2005-2007 Carlos Guestrin

61

## Threshold splits

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

©2005-2007 Carlos Guestrin

62

## 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  $x_i + (x_{i+1} - x_i)/2$

©2005-2007 Carlos Guestrin

63

## 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

©2005-2007 Carlos Guestrin

64



# Example with MPG

Information gains using the training set (40 records)

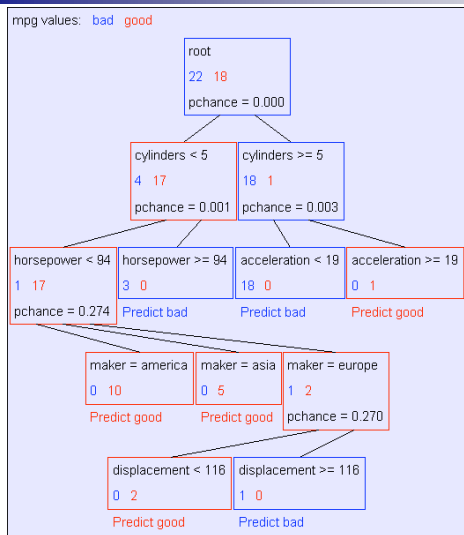
mpg values: bad good

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		

©2005-2007 Carlos Guestrin

65

# Example tree using reals



©2005-2007 Carlos Guestrin

66

## 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
- It's possible to get in trouble with overfitting (more next lecture)

©2005-2007 Carlos Guestrin

67

## Acknowledgements

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

©2005-2007 Carlos Guestrin

68