DS 4400

Machine Learning and Data Mining I
Spring 2021

Alina Oprea
Associate Professor
Khoury College of Computer Science
Northeastern University

March 4 2021

Announcements

- Project proposal is due today
  - One page
  - Team of 2
  - Problem statement
  - Dataset description
  - ML Models and metrics for evaluation
  - Related work
- Homework 3 is due on Monday, March 8

Outline

- Decision Trees
  - Entropy, information gain
  - Learning decision trees
  - Stopping conditions
- Ensemble models
  - Majority vote over multiple models

Sample Dataset

- Columns denote features X_i
- Rows denote labeled instances $< x_i, y_i >$
- Class label denotes whether a tennis game was played

$< x_i, y_i >$

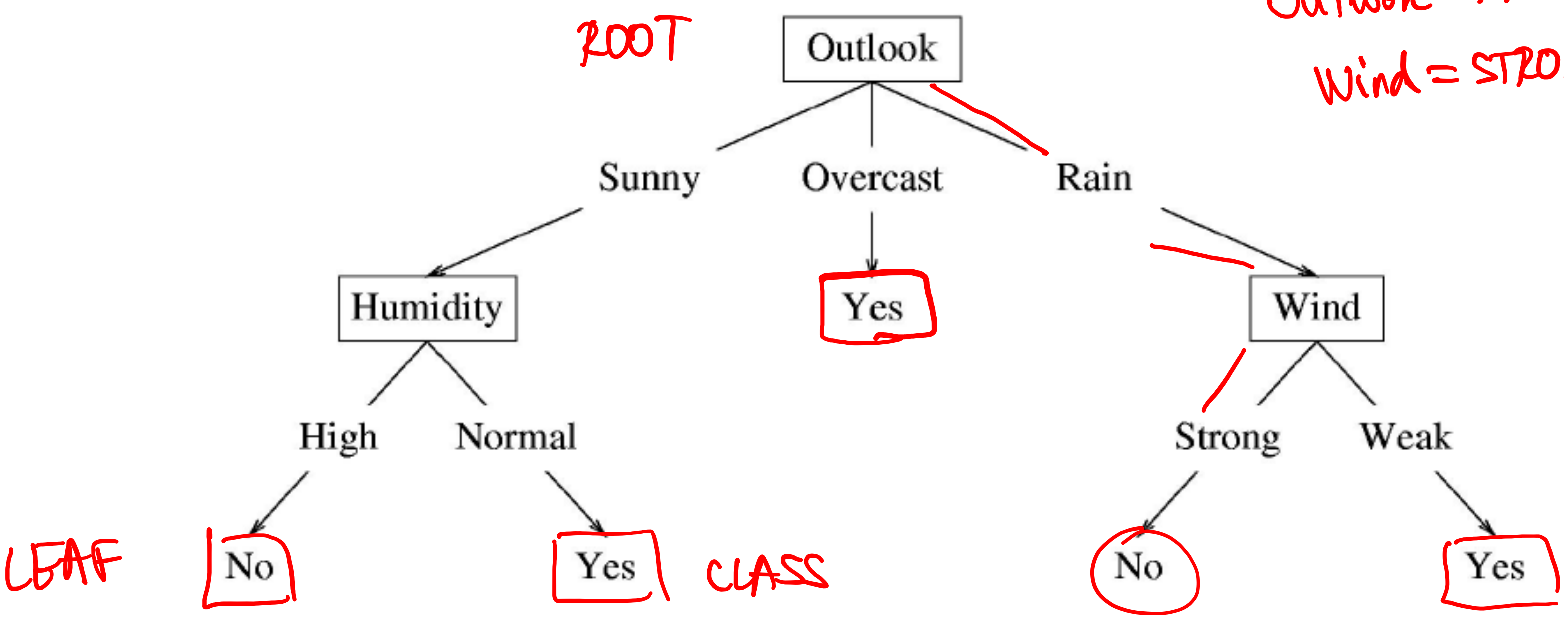
Categorical data

Predictors				Response
Outlook	**Temperature**	**Humidity**	**Wind**	**Class**
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

Decision Tree

- A possible decision tree for the data:

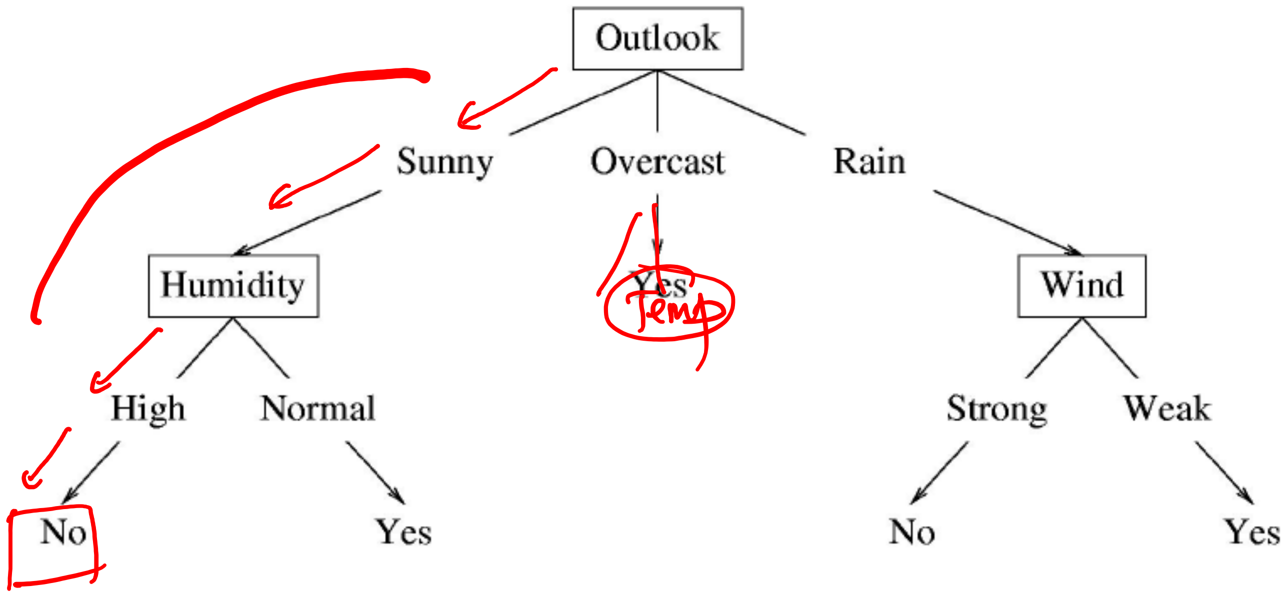
Outlook = RAIN AND
Wind = STRONG ⇒ NO



- Each internal node: test one attribute X_i
- Each branch from a node: selects one value for X_i
- Each leaf node: predict Y (or $p(Y \mid x \in \text{leaf})$)

Decision Tree MODEL

- A possible decision tree for the data:

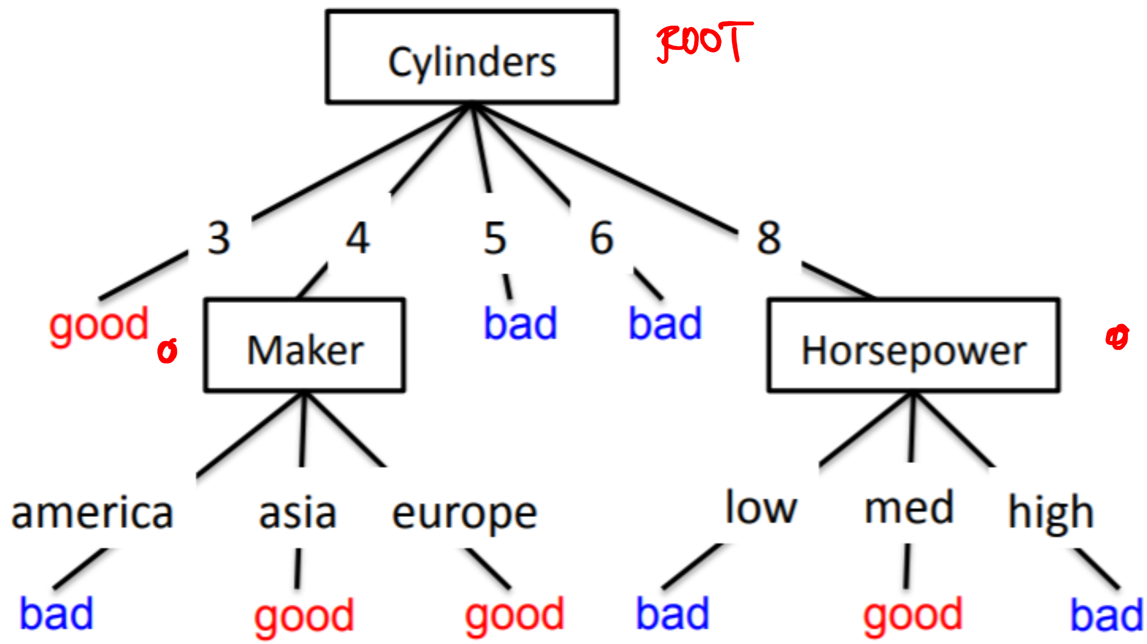


- What prediction would we make for

TESTING <outlook=sunny, temperature=hot, humidity=high, wind=weak> ?

Interpretability

- Each internal node tests an attribute x_i
- One branch for each possible attribute value $x_i = v$
- Each leaf assigns a class y
- To classify input x: traverse the tree from root to leaf, output the labeled y



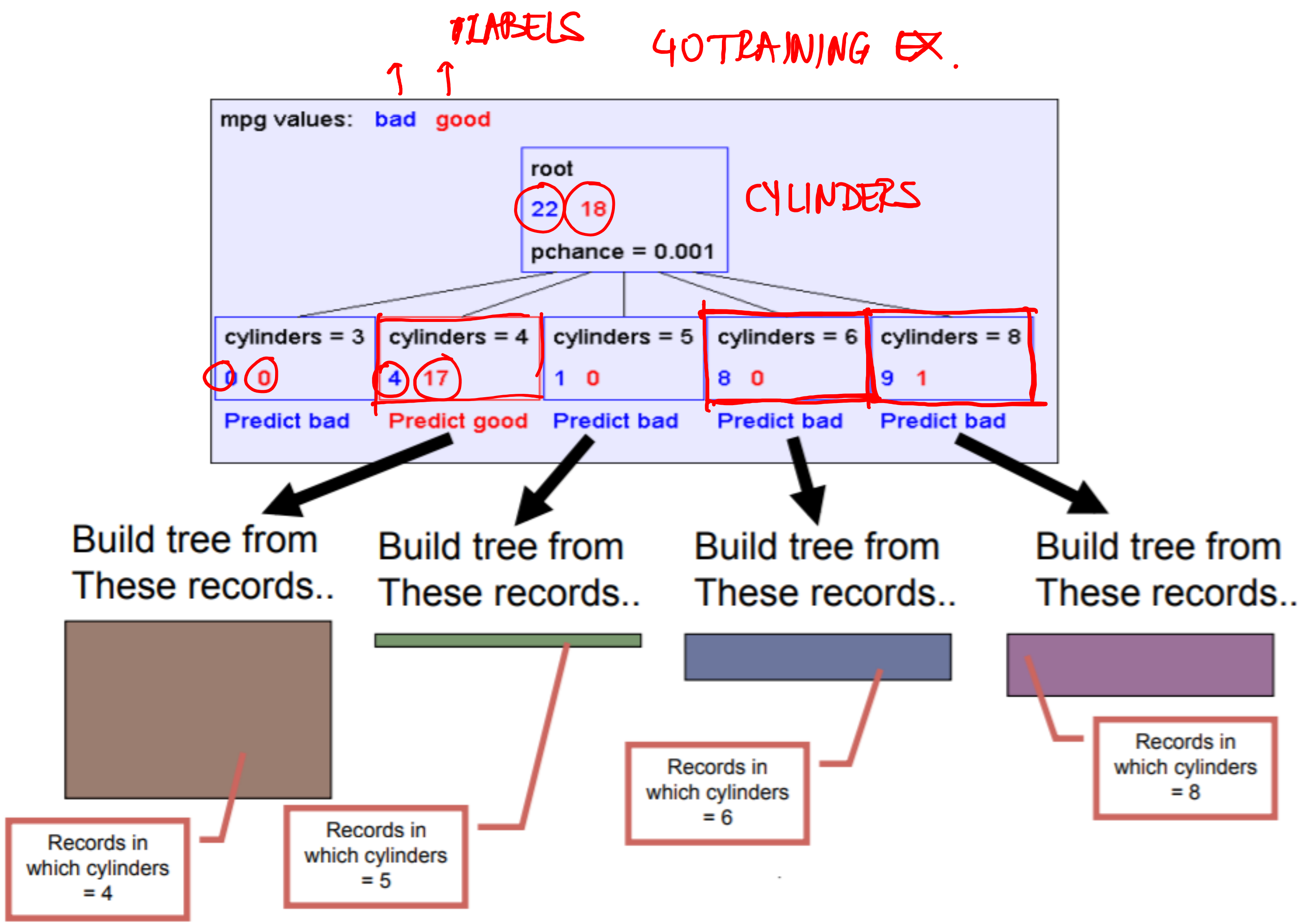
Human interpretable!

Learning Decision Trees

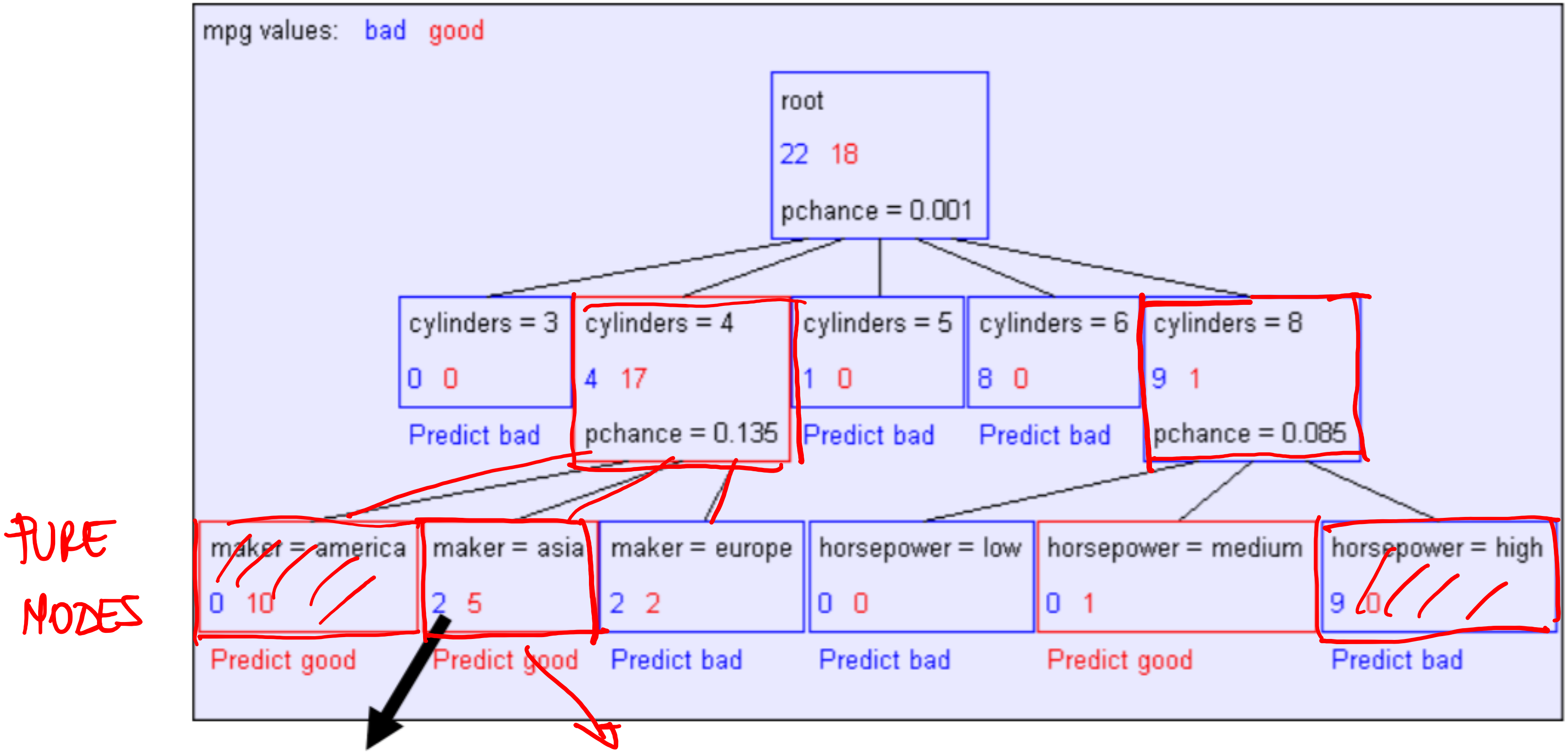
- Learning the simplest (smallest) decision tree is an NP-complete problem [Hyafil & Rivest ’76]
- Resort to a greedy heuristic:
  - Start from empty decision tree
  - Split on **next best attribute (feature)**
  - Recurse

STOPPING CONDITION.

Key Idea: Use Recursion Greedily



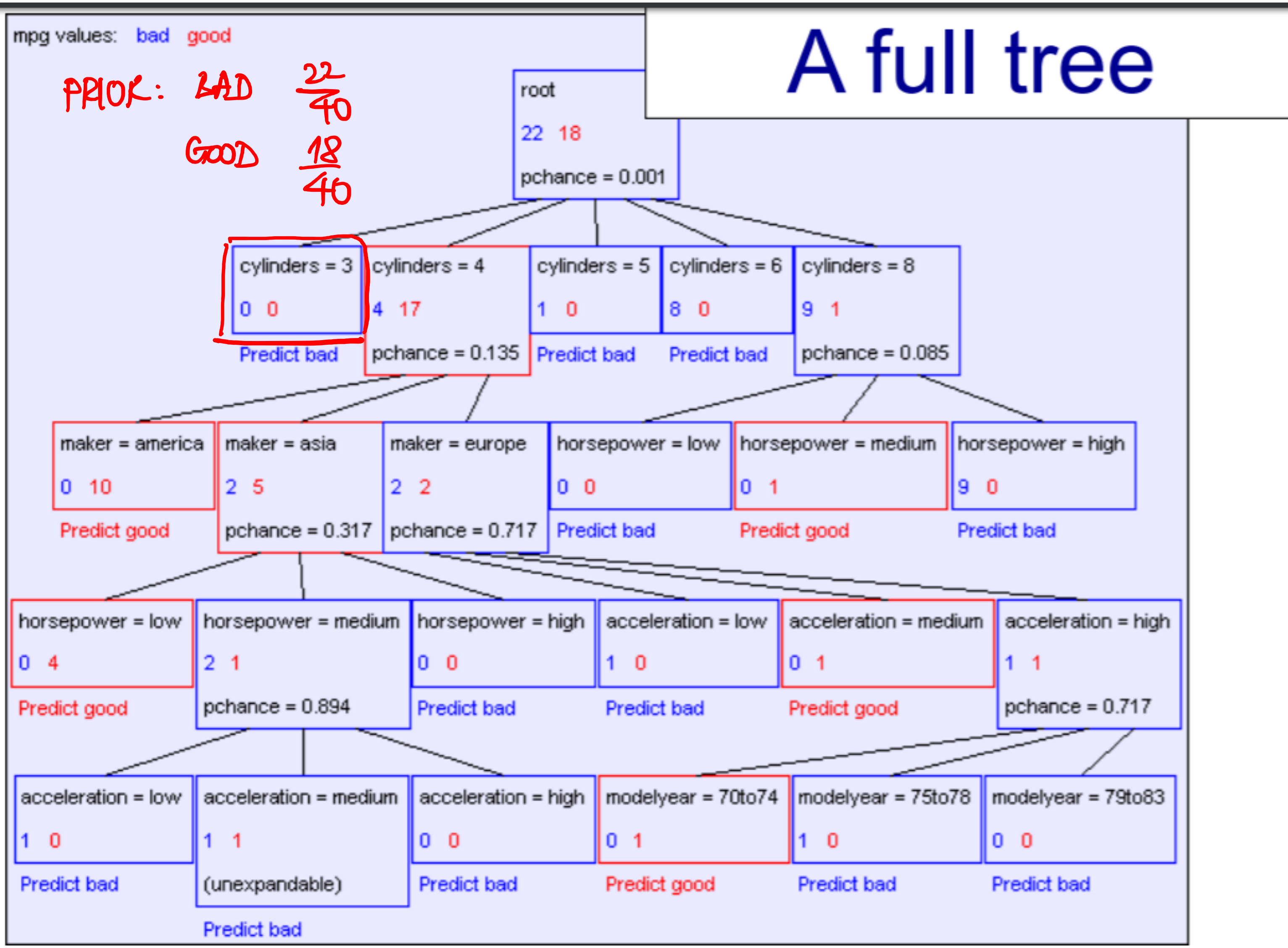
Second Level



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)

Full Tree

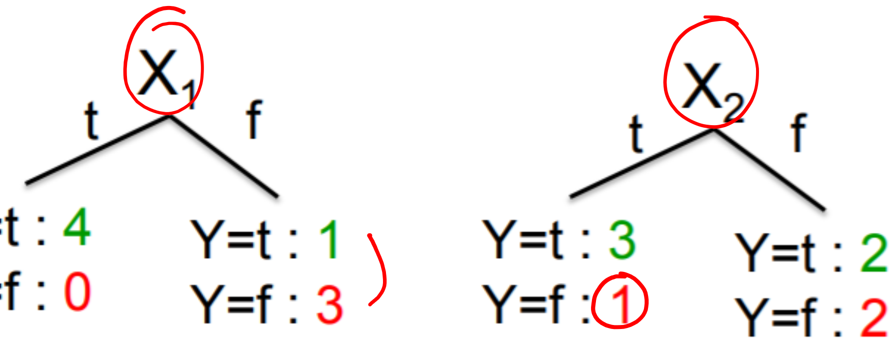


A full tree
mpg values: bad good
PRIOR: BAD 22/40
GOOD 18/40
root
22 18
pchance = 0.001
cylinders = 3
0 0
Predict bad
cylinders = 4
4 17
pchance = 0.135
cylinders = 5
1 0
Predict bad
cylinders = 6
8 0
Predict bad
cylinders = 8
9 1
pchance = 0.085
maker = america
0 10
Predict good
maker = asia
2 5
pchance = 0.317
maker = europe
2 2
pchance = 0.717
horsepower = low
0 0
Predict bad
horsepower = medium
0 1
Predict good
horsepower = high
9 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
0 1
Predict good
acceleration = high
1 1
pchance = 0.717
acceleration = low
1 0
Predict bad
acceleration = medium
1 1
(unexpandable)
Predict bad
acceleration = high
0 0
Predict bad
modelyear = 70to74
0 1
Predict good
modelyear = 75to78
1 0
Predict bad
modelyear = 79to83
0 0
Predict bad

Splitting

TRAINING DATA

Would we prefer to split on X_1 or X_2?



PURE

TRUE FALSE TRUE T | F

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

Idea: use counts at leaves to define probability distributions, so we can measure uncertainty!

ERRORS: $\begin{cases} X_1 : 1 \\ X_2 : 3 \end{cases}$

Use entropy-based measure (Information Gain)

Entropy

RV

Suppose X can have one of m values… $V_1, V_2, \ldots V_m$

$\sum_{i=1}^{m} p_i = 1$

$P(X=V_1) = p_1$	$P(X=V_2) = p_2$	….	$P(X=V_m) = p_m$

What's the smallest possible number of bits, on average, per symbol, needed to transmit a stream of symbols drawn from X's distribution? It's

$$H(X) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \ldots - p_m \log_2 p_m$$

$$= \boxed{-\sum_{j=1}^{m} p_j \log_2 p_j}$$

Symbols: A: 90% $p_1 = 0.9$ → BIT 0

B: 5% $p_2 = 0.05$ → 10

C: 5% $p_3 = 0.05$ → 11

Avg # bits: $0.9 \cdot 1 + 0.1 \cdot 2$

$= 1.1$

H(X) = The entropy of X

- "High Entropy" means X is from a uniform (boring) distribution
- "Low Entropy" means X is from varied (peaks and valleys) distribution

Entropy Examples

① UNIFORM : MAX

$$X \sim \begin{pmatrix} A & B \\ 0 & 1 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} \begin{array}{l} \to \text{values} \\ \to \text{prob} \end{array}, \quad H(X) = -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = -\log_2\frac{1}{2} = \log_2 2 = 1$$

$$X \sim \begin{pmatrix} 1 & 2 & \dots & n \\ \frac{1}{n} & \frac{1}{n} & & \frac{1}{n} \end{pmatrix}, \quad H(X) = -\frac{1}{n}\log_2\frac{1}{n} - \dots - \frac{1}{n}\log_2\frac{1}{n} = -\log_2\frac{1}{n} = \log_2 n$$

② MIN

$$Y \sim \begin{pmatrix} A & B \\ 0 & 1 \end{pmatrix}, \quad H(Y) = 0$$

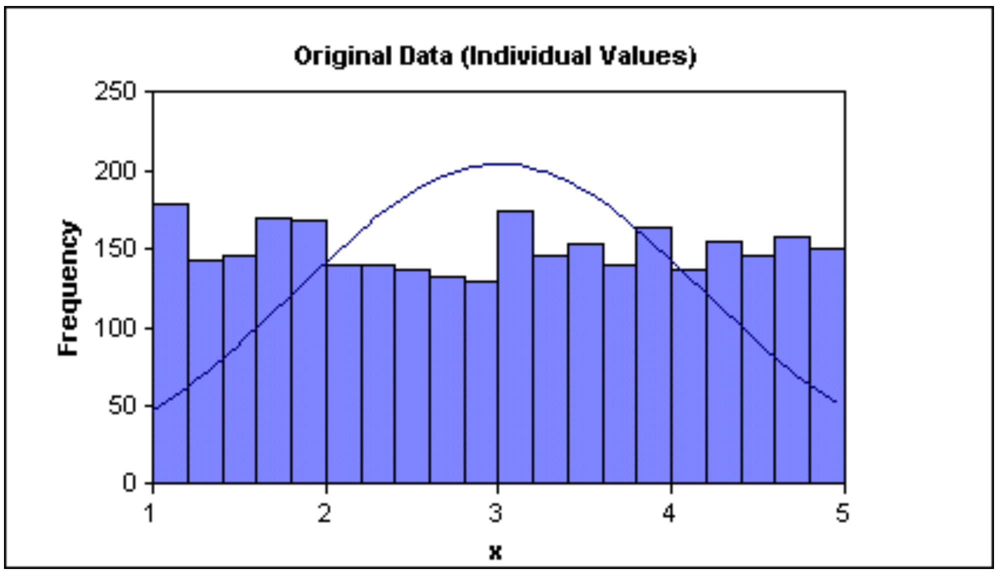
③

$$Z \sim \begin{pmatrix} A & B \\ \frac{1}{3} & \frac{2}{3} \end{pmatrix}, \quad H(Z) = -\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3} = 0.52$$

High/Low Entropy

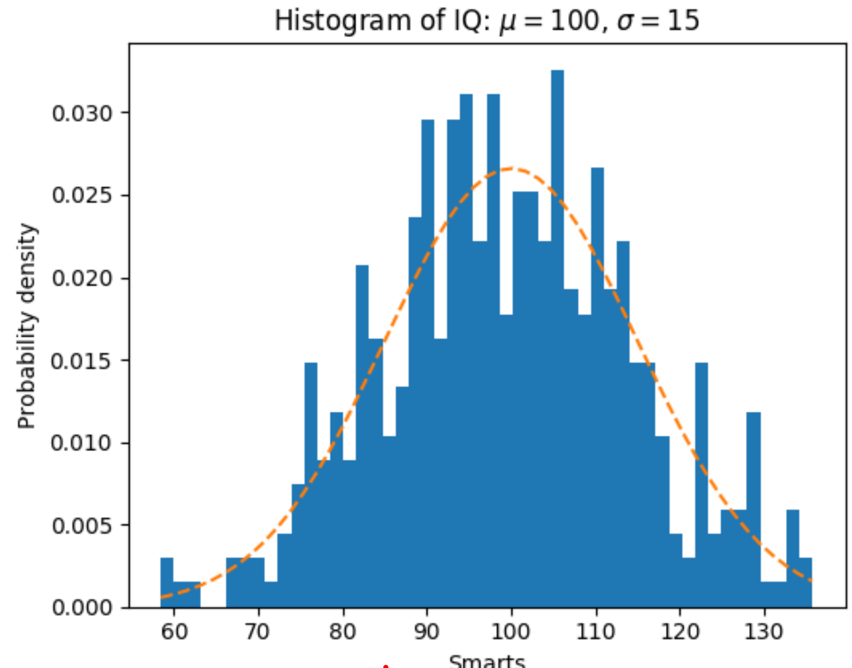
Which distribution has high entropy?

~ UNIFORM



HIGH ENT.

HIGH VAR



LOW ENT.

ENT = 0

Conditional Entropy

Suppose I'm trying to predict output Y and I have input X

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Let's assume this reflects the true probabilities

E.G. From this data we estimate

- *P(LikeG = Yes) =* 1/2 PRIOR
- *P(Major = Math & LikeG = No) =* 1/4
- *P(Major = Math) =* 1/2
- *P(LikeG = Yes | Major = History) =* 0

$X \sim \begin{pmatrix} M & CS & H \\ 1/2 & 1/4 & 1/4 \end{pmatrix}$

Note:

- *H(X) =* 1.5
- *H(Y) =* 1

$H(X) = -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{4}\log_2\frac{1}{4} - \frac{1}{4}\log_2\frac{1}{4} = 1.5$

Conditional Entropy

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

$H(Y)=1$

Definition of Specific Conditional Entropy:

$H(Y|X=v)$ = **The entropy of** Y **among only those records in which** X **has value** v

Example:

- $H(Y|X=Math) = 1$
- $H(Y|X=History) = 0$
- $H(Y|X=CS) = 0$

Conditional Entropy

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Conditional Entropy:

$\boxed{H(Y|X)}$ = The average specific conditional entropy of Y

= if you choose a record at random what will be the conditional entropy of Y, conditioned on that row's value of X

= Expected number of bits to transmit Y if both sides will know the value of X

$$= \Sigma_j \underbrace{Prob(X=v_j)}\ \underbrace{H(Y \mid X = v_j)}$$

Conditional Entropy

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Conditional Entropy:

$H(Y|X)$ = The average conditional entropy of Y

$= \boxed{\Sigma_j Prob(X=v_j)\, H(Y \mid X = v_j)}$

Example:

v_j	$Prob(X=v_j)$	$H(Y \mid X = v_j)$
Math	1/2	1
History	1/4	0
CS	1/4	0

$H(Y|X) = \frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 0 + \frac{1}{4} \cdot 0 = \frac{1}{2}$

Information Gain

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Information Gain:

$IG(Y|X)$ = **I must transmit** Y. **How many bits on average would it save me if both ends of the line knew** X**?**

$$\boxed{IG(Y|X) = H(Y) - H(Y \mid X)}$$

Example:

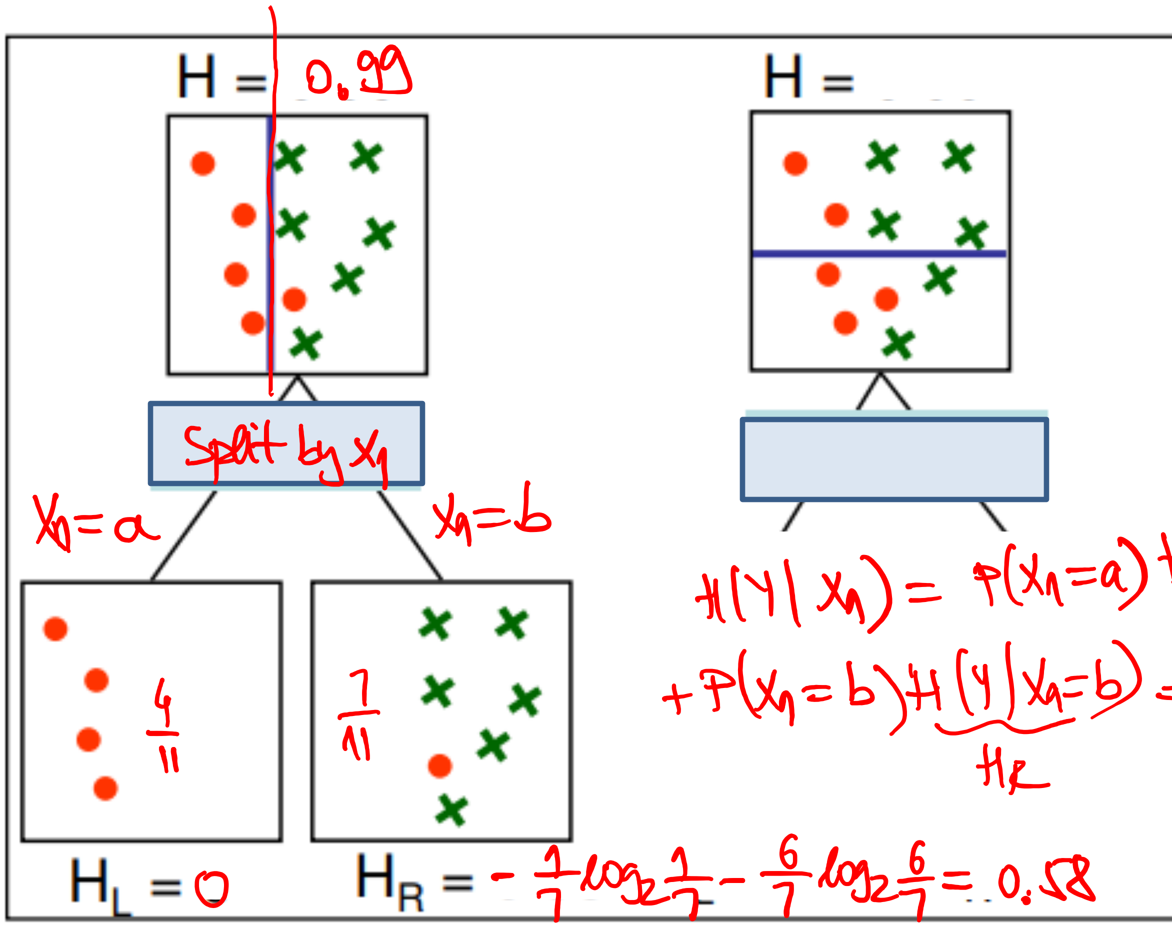
- **H(Y) =** 1
- **H(Y|X) =** 1/2
- **Thus IG(Y|X) =** $\frac{1}{2}$

$X_1, \dots, X_{10}$, Y. COMPUTE $IG(Y|X_i)$

PICK X_i TO MAX $IG(Y|X_i)$

Example Information Gain

$$Y \sim \begin{pmatrix} R & G \\ \frac{5}{11} & \frac{6}{11} \end{pmatrix}; \quad H(Y) = -\frac{5}{11}\log_2\frac{5}{11} - \frac{6}{11}\log_2\frac{6}{11}$$

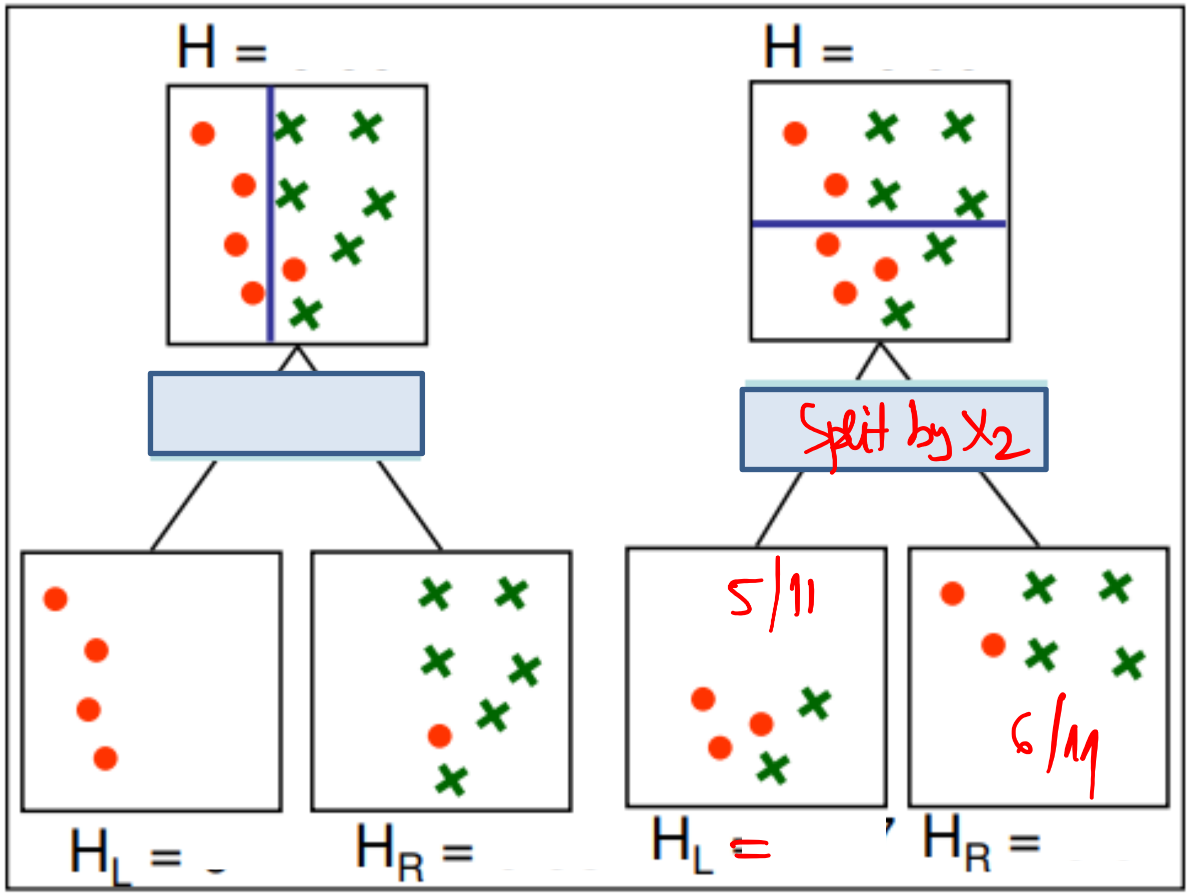


PURE

$$H(Y|X_1) = P(X_1=a)\underbrace{H(Y|X_1=a)}_{H_L} + P(X_1=b)\underbrace{H(Y|X_1=b)}_{H_R} = \frac{4}{11}\cdot H_L + \frac{7}{11}\cdot H_R = 0.37$$

$$IG(Y|X_1) = H(Y) - H(Y|X_1) = 0.99 - 0.37 = 0.62$$

Example Information Gain

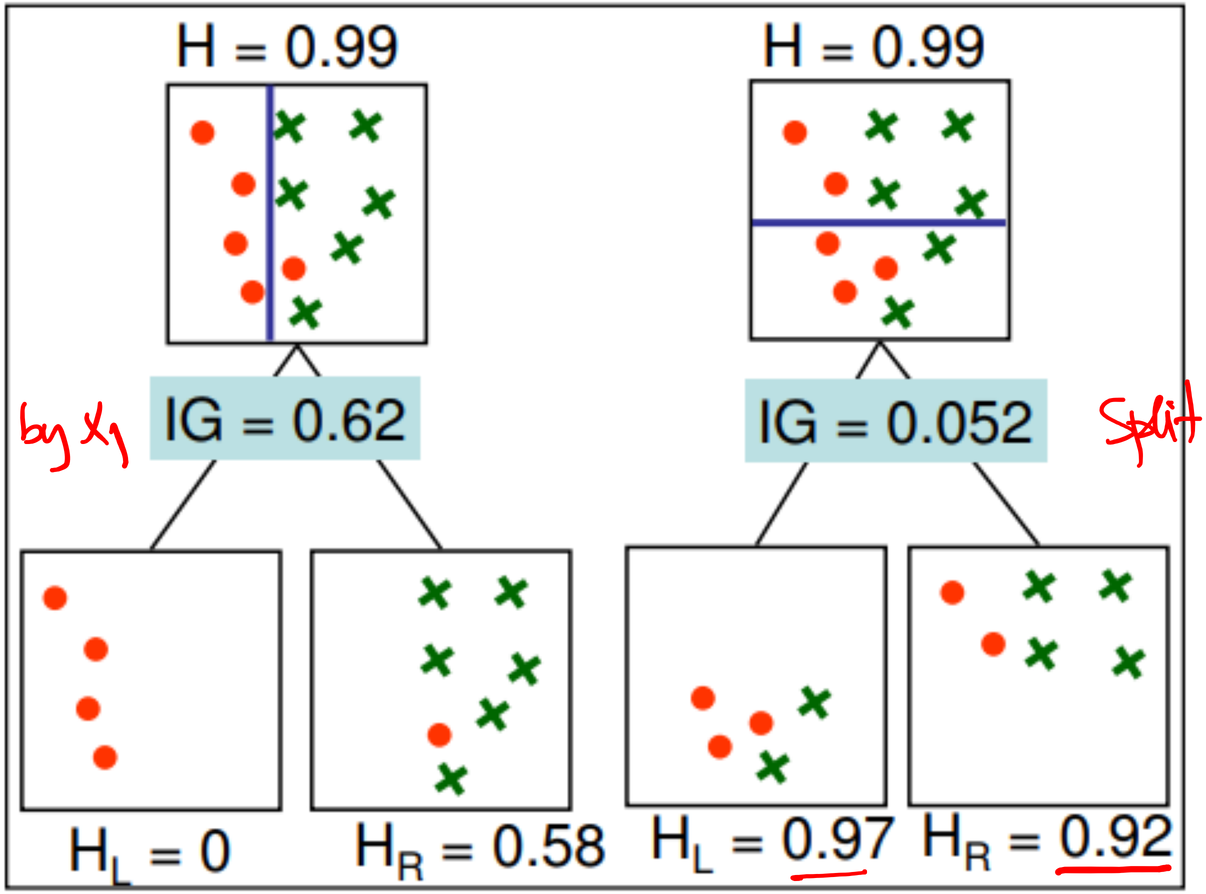


$= -\frac{3}{5}\log\frac{3}{5} - \frac{2}{5}\log\frac{2}{5}$

$IG(Y|X_2)$

Example Information Gain

Max Information Gain



Split by x_1

Split by x_2

Pure node

Learning Decision Trees

ROOT: ALL FEATURES

- Start from empty decision tree
- Split on **next best attribute (feature)**
  - Use, for example, information gain to select attribute:

$$\arg\max_i IG(X_i) = \arg\max_i H(Y) - H(Y \mid X_i)$$

IN EACH ITERATION: SUBSET OF FEATURES

- Recurse

ID3 algorithm uses Information Gain
Information Gain reduces uncertainty on Y

Impurity Metrics

Split a node according to max reduction of impurity

1. Entropy | IG
2. Gini Index

UNIFORM: $p_0 = p_1 = \frac{1}{2}$

$I(p_0, p_1) = \frac{1}{2}$

   - For binary case with prob p_0, p_1:

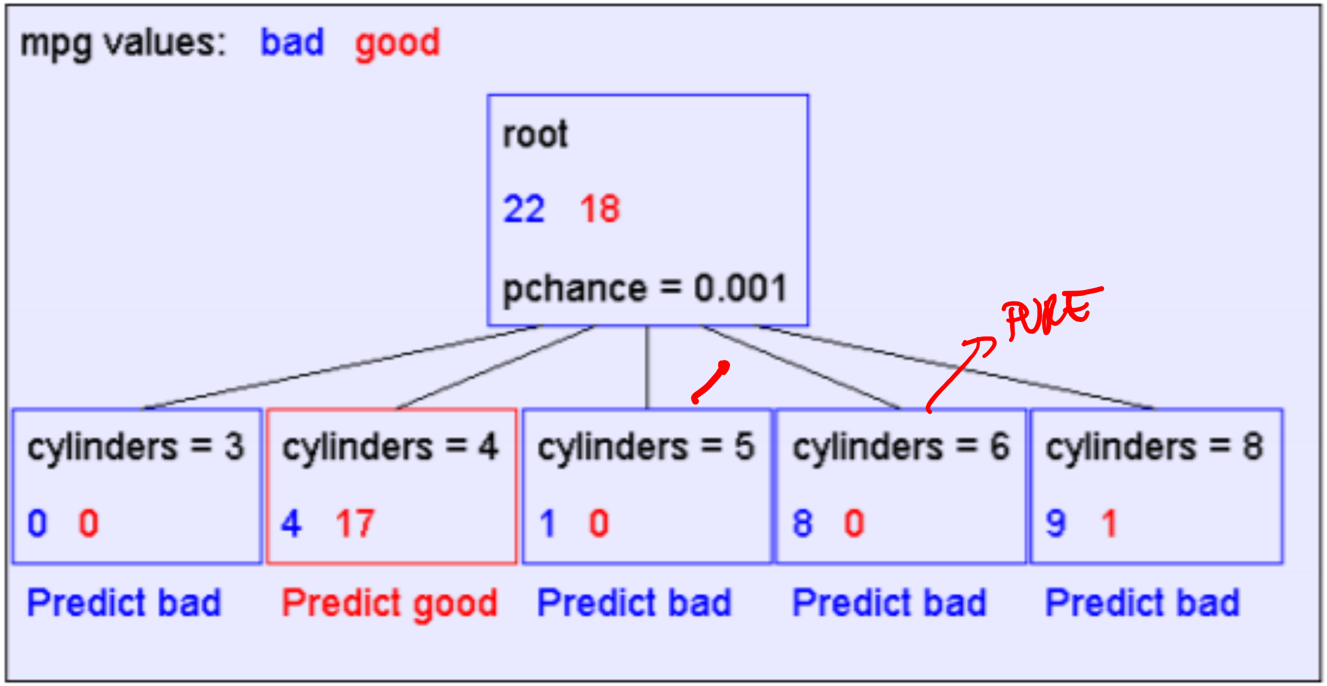
$$I(p_0, p_1) = 2p_0 p_1 = 2p_0(1 - p_0)$$

   - For multi-class with prob $p_1, \ldots, p_K$:

$$I(p_1, \ldots p_K) = 1 - \sum_{k=1}^{K} p_k^2$$

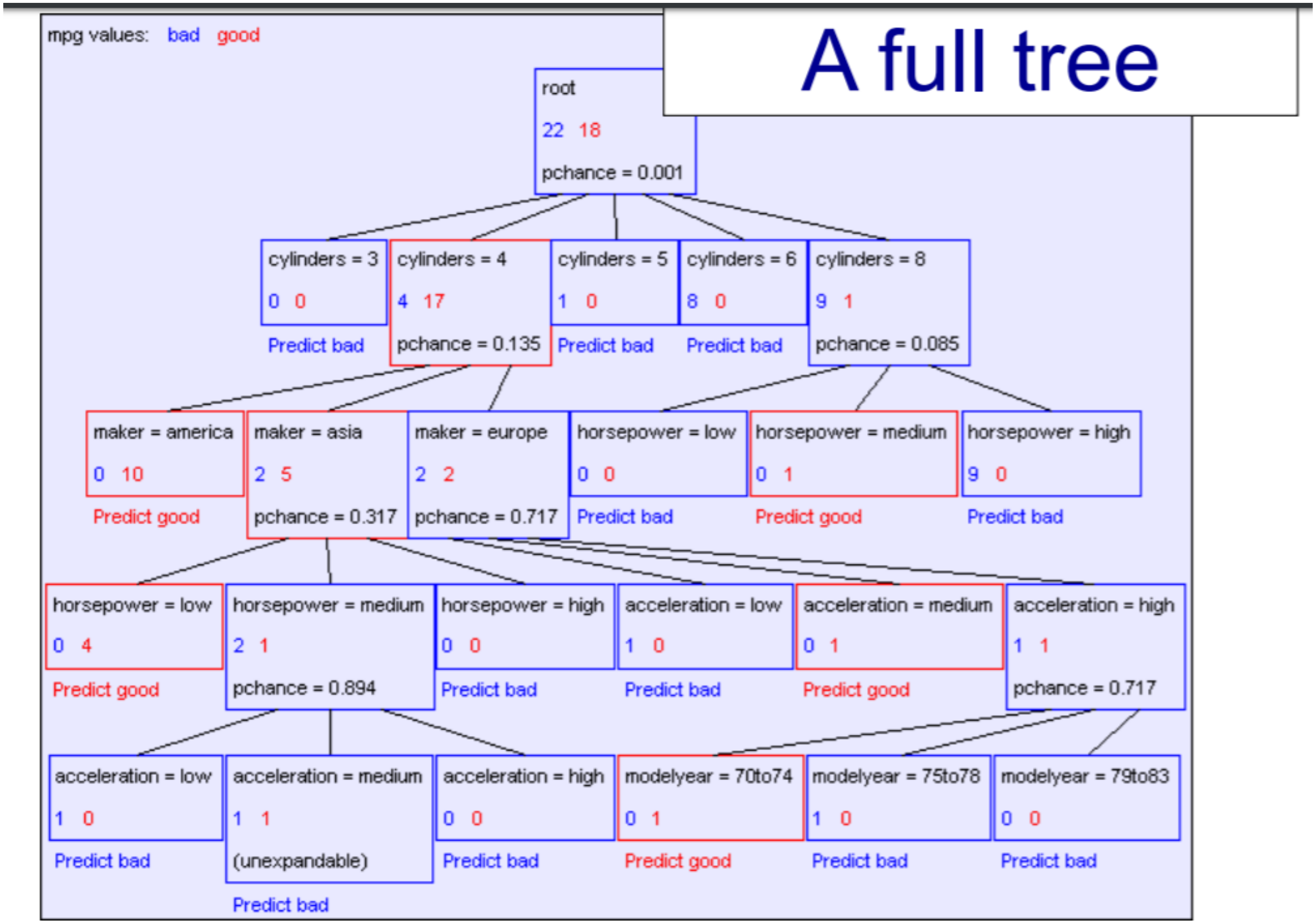
- Properties
  - Impurity metrics have value 0 for pure nodes
  - Impurity metrics are maximized for uniform distribution (nodes with most uncertainty)

When to stop?

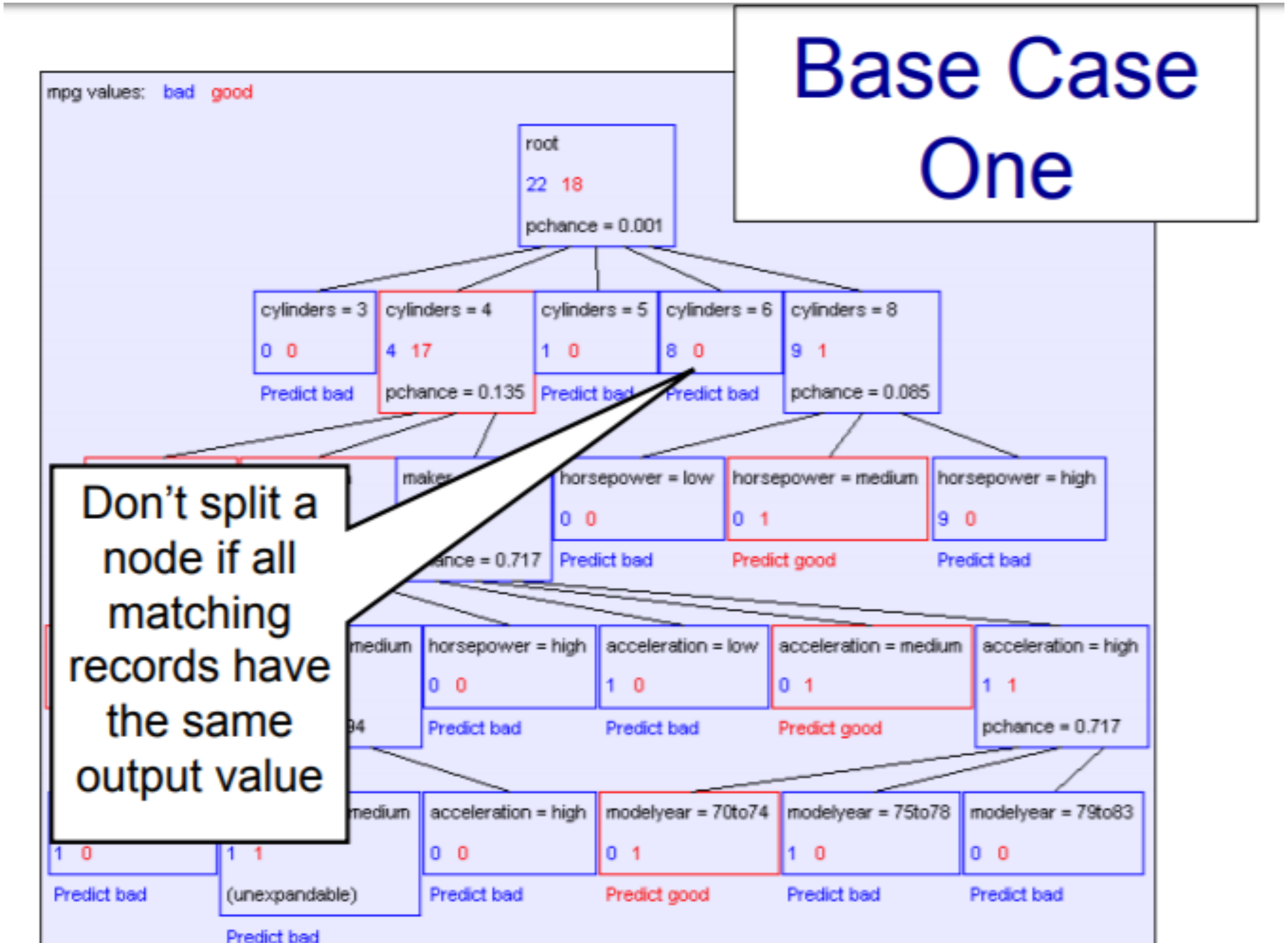


First split looks good! But, when do we stop?

Full Tree

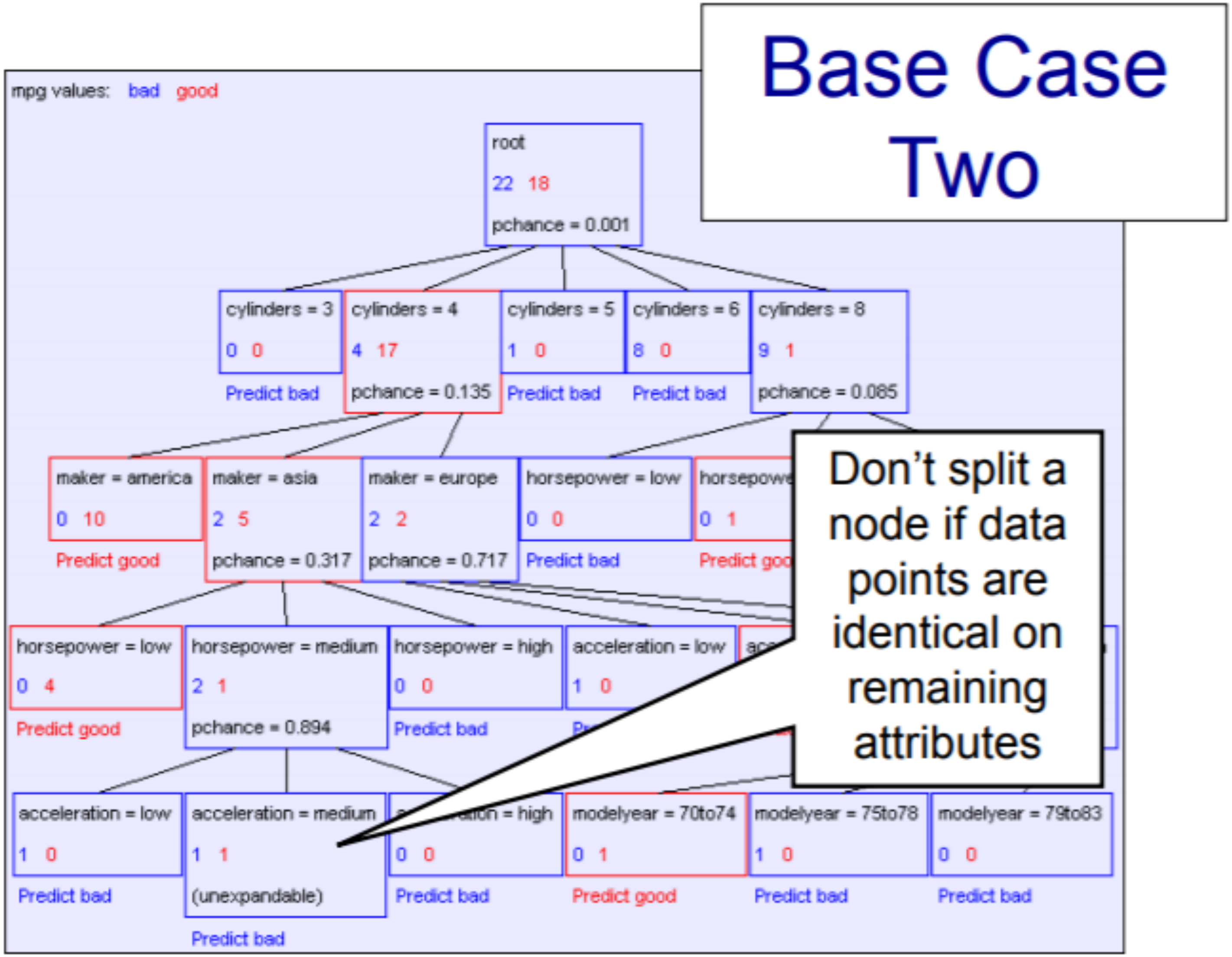

A full tree
mpg values: bad good
root
22 18
pchance = 0.001
cylinders = 3
0 0
Predict bad
cylinders = 4
4 17
pchance = 0.135
cylinders = 5
1 0
Predict bad
cylinders = 6
8 0
Predict bad
cylinders = 8
9 1
pchance = 0.085
maker = america
0 10
Predict good
maker = asia
2 5
pchance = 0.317
maker = europe
2 2
pchance = 0.717
horsepower = low
0 0
Predict bad
horsepower = medium
0 1
Predict good
horsepower = high
9 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
0 1
Predict good
acceleration = high
1 1
pchance = 0.717
acceleration = low
1 0
Predict bad
acceleration = medium
1 1
(unexpandable)
Predict bad
acceleration = high
0 0
Predict bad
modelyear = 70to74
0 1
Predict good
modelyear = 75to78
1 0
Predict bad
modelyear = 79to83
0 0
Predict bad

Case 1



Base Case One
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
9 1
pchance = 0.085
horsepower = low
0 0
Predict bad
horsepower = medium
0 1
Predict good
horsepower = high
9 0
Predict bad
horsepower = high
0 0
Predict bad
acceleration = low
1 0
Predict bad
acceleration = medium
0 1
Predict good
acceleration = high
1 1
pchance = 0.717
acceleration = high
0 0
Predict bad
modelyear = 70to74
0 1
Predict good
modelyear = 75to78
1 0
Predict bad
modelyear = 79to83
0 0
Predict bad
(unexpandable)
Predict bad
Don't split a node if all matching records have the same output value

Case 2



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
9 1
pchance = 0.085
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
0 1
horsepower = low
0 4
Predict good
horsepower = medium
2 1
pchance = 0.894
horsepower = high
0 0
Predict bad
acceleration = low
1 0
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
Base Case Two
Don't split a node if data points are identical on remaining attributes

Decision Trees

TRAINING LABELS

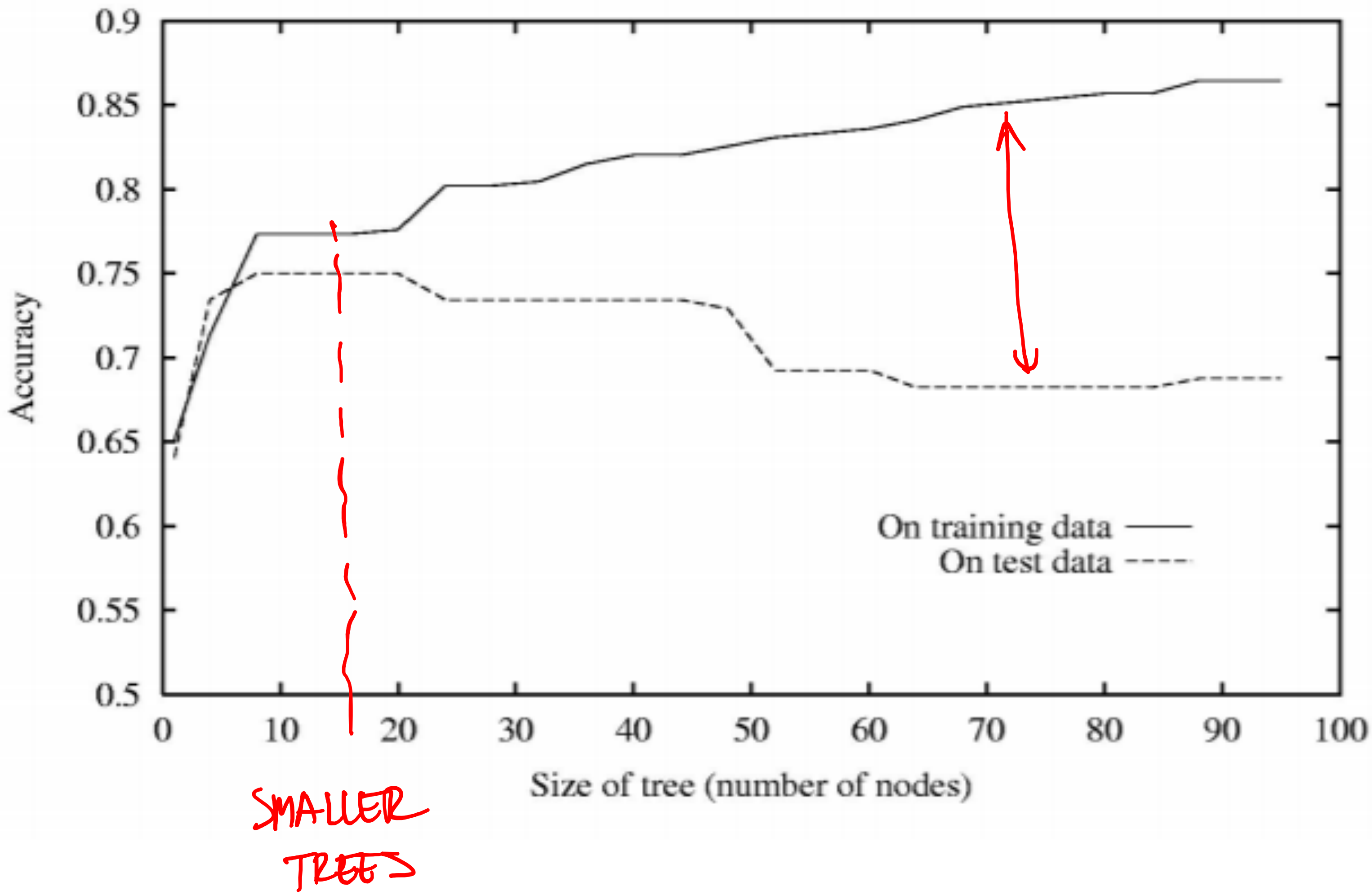
BuildTree(*DataSet*,*Output*)

- If all output values are the same in *DataSet*, return a leaf node that says "predict this unique output" CASE 1 | PURE
- If all input values are the same, return a leaf node that says "predict the majority output" CASE 2
- Else find attribute *X* with highest Info Gain ON SUBSET OF FEATURES
- Suppose *X* has n_X distinct values (i.e. X has arity n_X).
  - Create a non-leaf node with n_X children.
  - The *i*'th child should be built by calling

    RECURSE BuildTree(DS_i,*Output*) SUBSET OF TRAINING

    Where DS_i contains the records in DataSet where X = *i*th value of X.

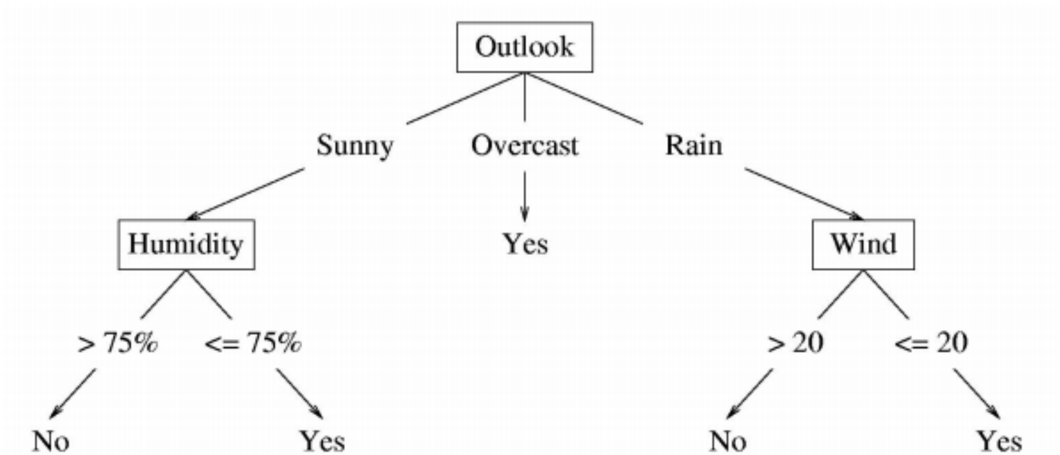
Overfitting



Solutions against Overfitting

- Standard decision trees have no learning bias
  - Training set error is always zero!
    - (If there is no label noise)
  - Lots of variance
  - Must introduce some bias towards simpler trees
- Many strategies for picking simpler trees
  - Fixed depth
  - Minimum number of samples per leaf
- Pruning
  - Remove branches of the tree that increase error using cross-validation

Real-valued Features



- Change to binary splits by choosing a threshold
- One method:
  – Sort instances by value, identify adjacencies with different classes

Humidity	40	48	60	72	80	90
PlayTennis:	No	No	Yes	Yes	Yes	No

candidate splits

  – Choose among splits by InfoGain()

Acknowledgements

- Slides made using resources from:
  - Andrew Ng
  - Eric Eaton
  - David Sontag
  - Andrew Moore
- Thanks!