ECE 5984: Introduction to Machine Learning

Topics:

- Decision/Classification Trees
- Ensemble Methods: Bagging, Boosting

Readings: Murphy 16.1-16.2; Hastie 9.2; Murphy 16.4

Dhruv Batra

Virginia Tech

Administrativia

- HW3
  - Due: April 14, 11:55pm
  - You will implement primal & dual SVMs
  - Kaggle competition: Higgs Boson Signal vs Background classification
  - https://inclass.kaggle.com/c/2015-Spring-vt-ece-machine-learning-hw3
  - https://www.kaggle.com/c/higgs-boson

Administrativia

- Project Mid-Sem Spotlight Presentations
  - 9 remaining
  - Resume in class on April 20th

  - Format
    - 5 slides (recommended)
    - 4 minute time (STRICT) + 1-2 min Q&A

  - Content
    - Tell the class what you're working on
    - Any results yet?
    - Problems faced?
  - Upload slides on Scholar

Decision Trees

Pose Estimation

- Random Forests!
  - Multiple decision trees
  - http://youtu.be/HNkbG3KsY84

Learning Decision Trees

Decision trees provide a very popular and efficient hypothesis space.

- **Variable Size**. Any boolean function can be represented.
- **Deterministic**.
- **Discrete and Continuous Parameters**.

A small dataset: Miles Per Gallon

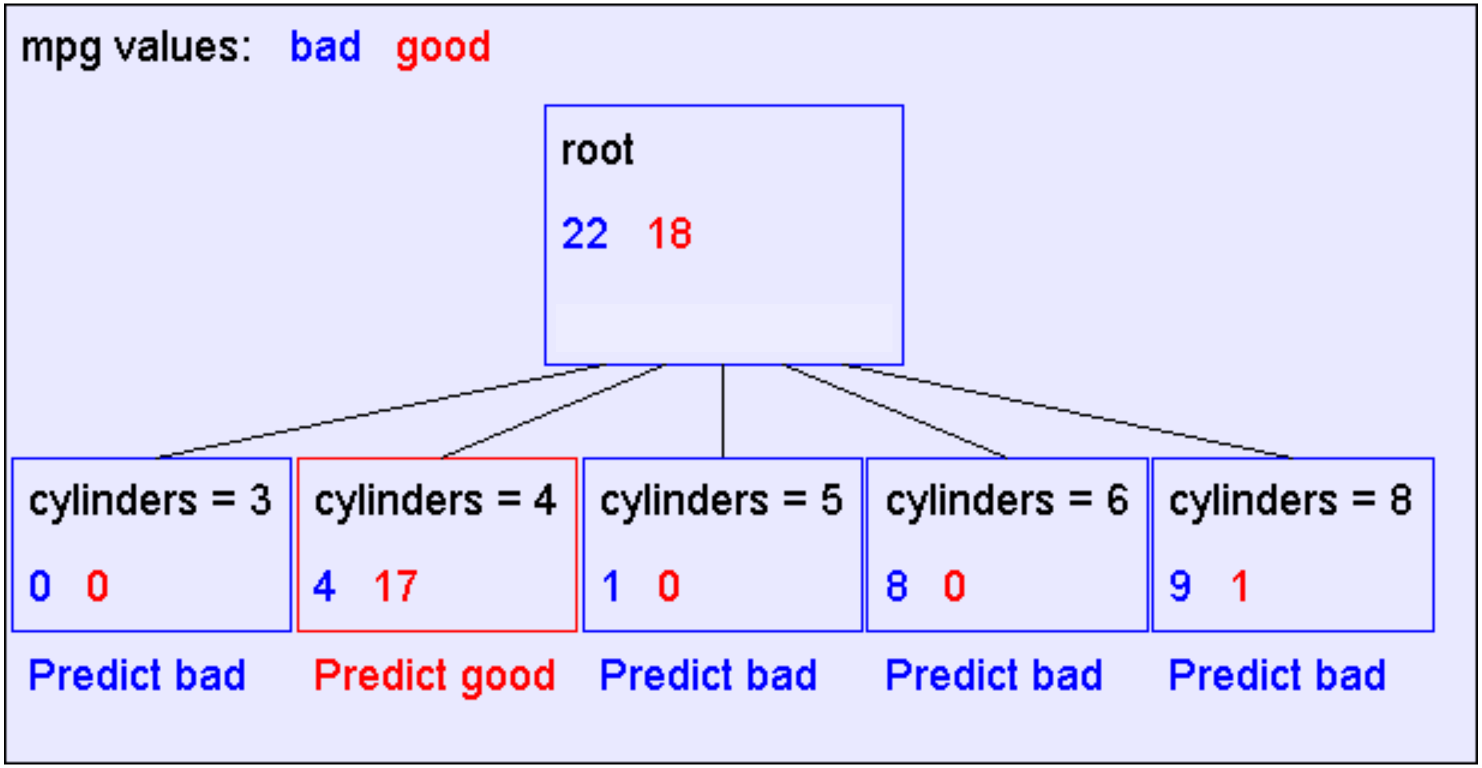
Suppose we want to predict MPG

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

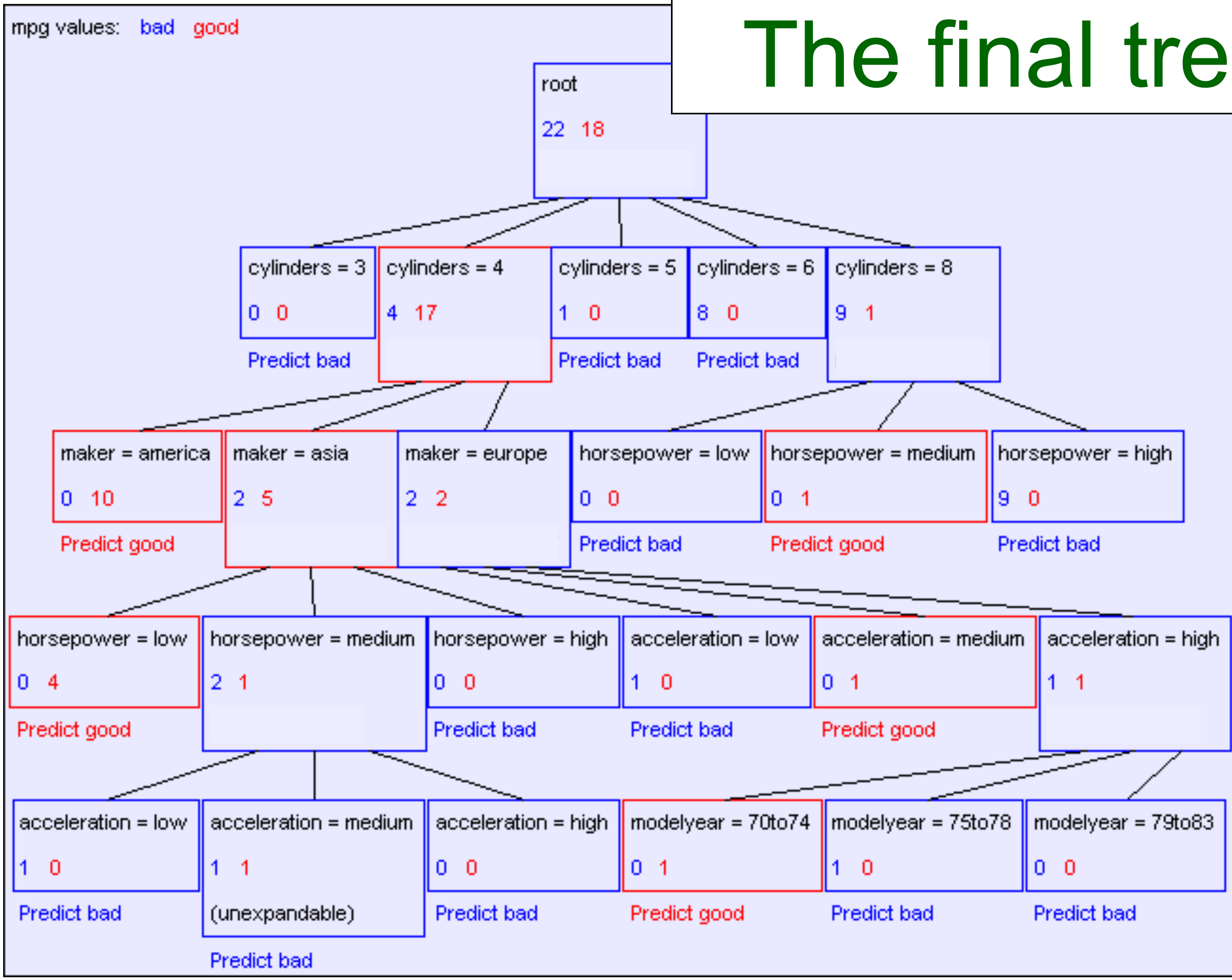
40 Records

From the UCI repository (thanks to Ross Quinlan)

A Decision Stump



The final tree



Comments

- Not all features/attributes need to appear in the tree.

- A features/attribute X_i may appear in multiple branches.

- On a path, no feature may appear more than once.
  - Not true for continuous features. We'll see later.

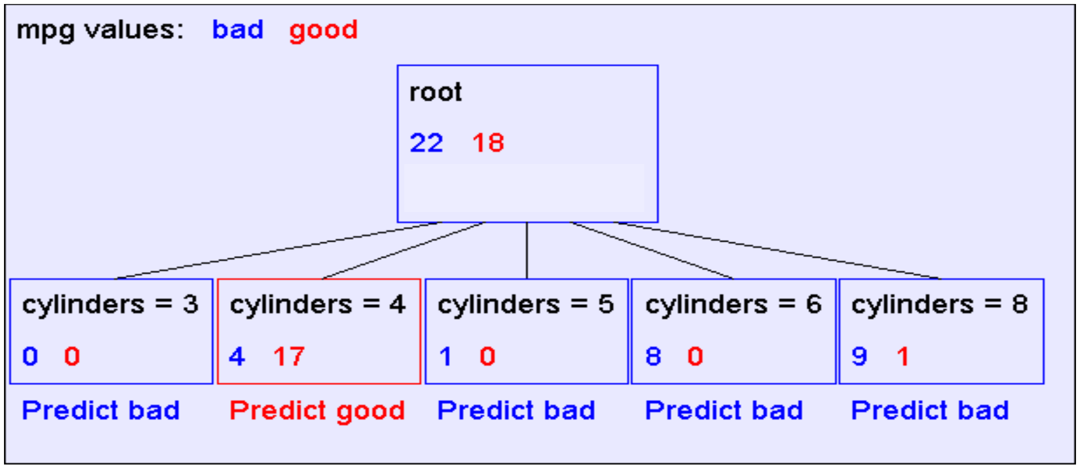
- Many trees can represent the same concept
- But, not all trees will have the same size!
  - e.g., $Y = (A \wedge B) \vee (\neg A \wedge C)$ (A and B) or (not A and C)

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
    - "Iterative Dichotomizer" (ID3)
    - C4.5 (ID3+improvements)

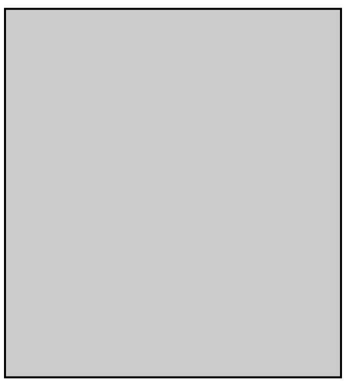
Recursion Step



Take the

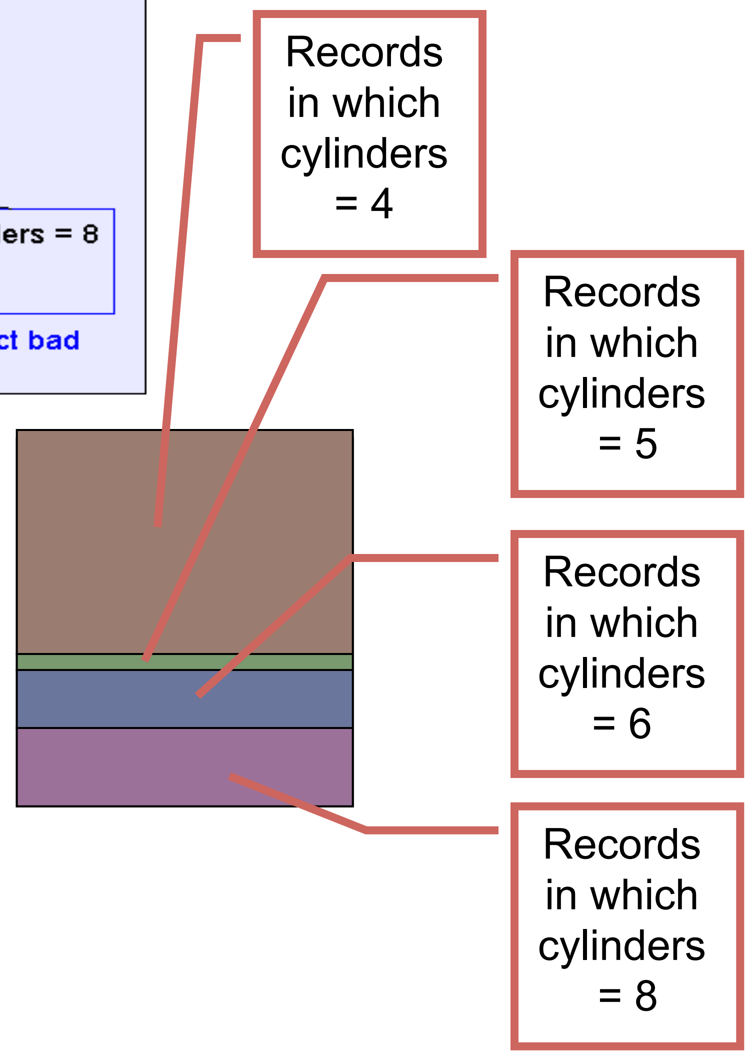
Original

Dataset..

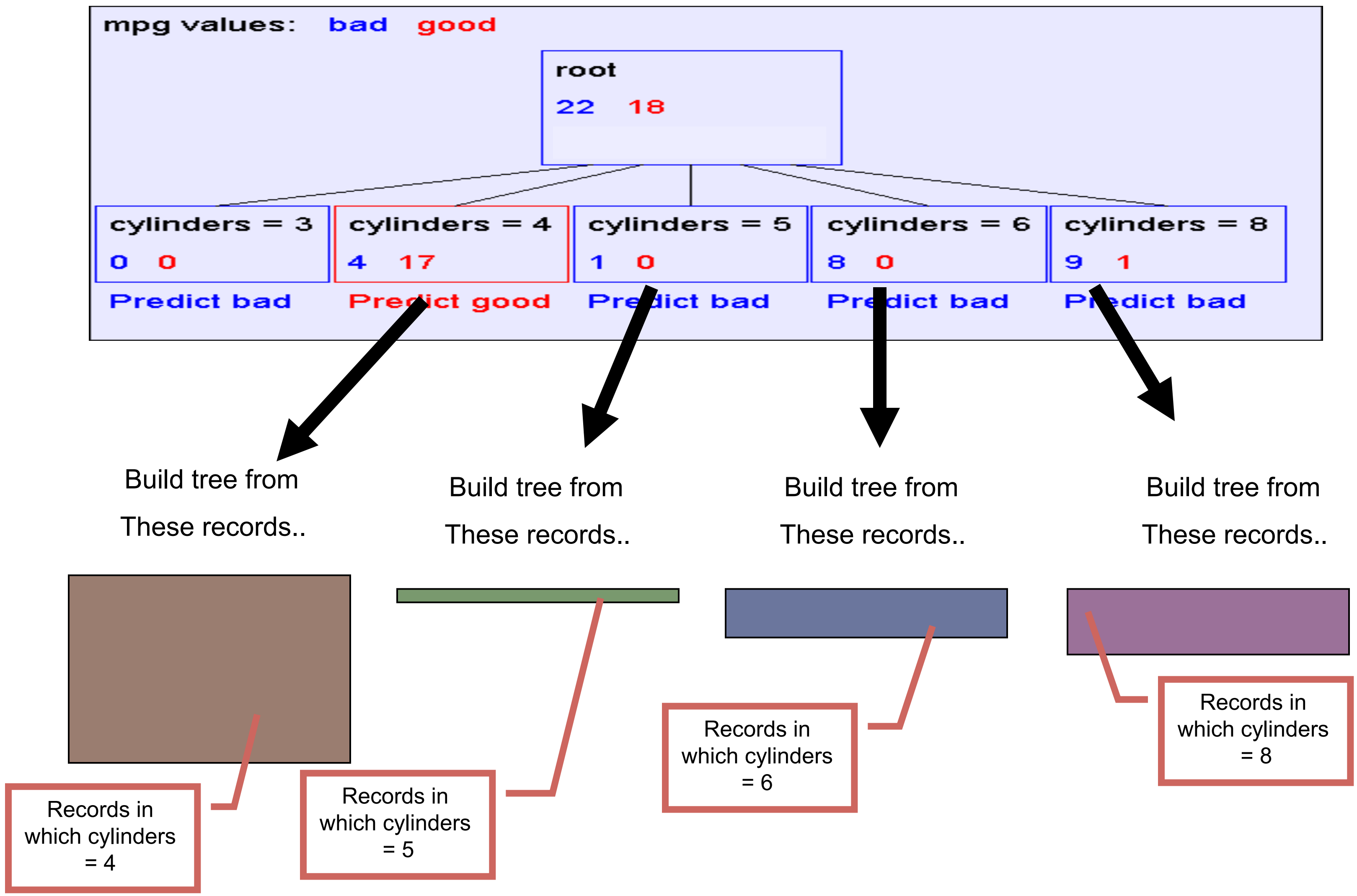


And partition it according

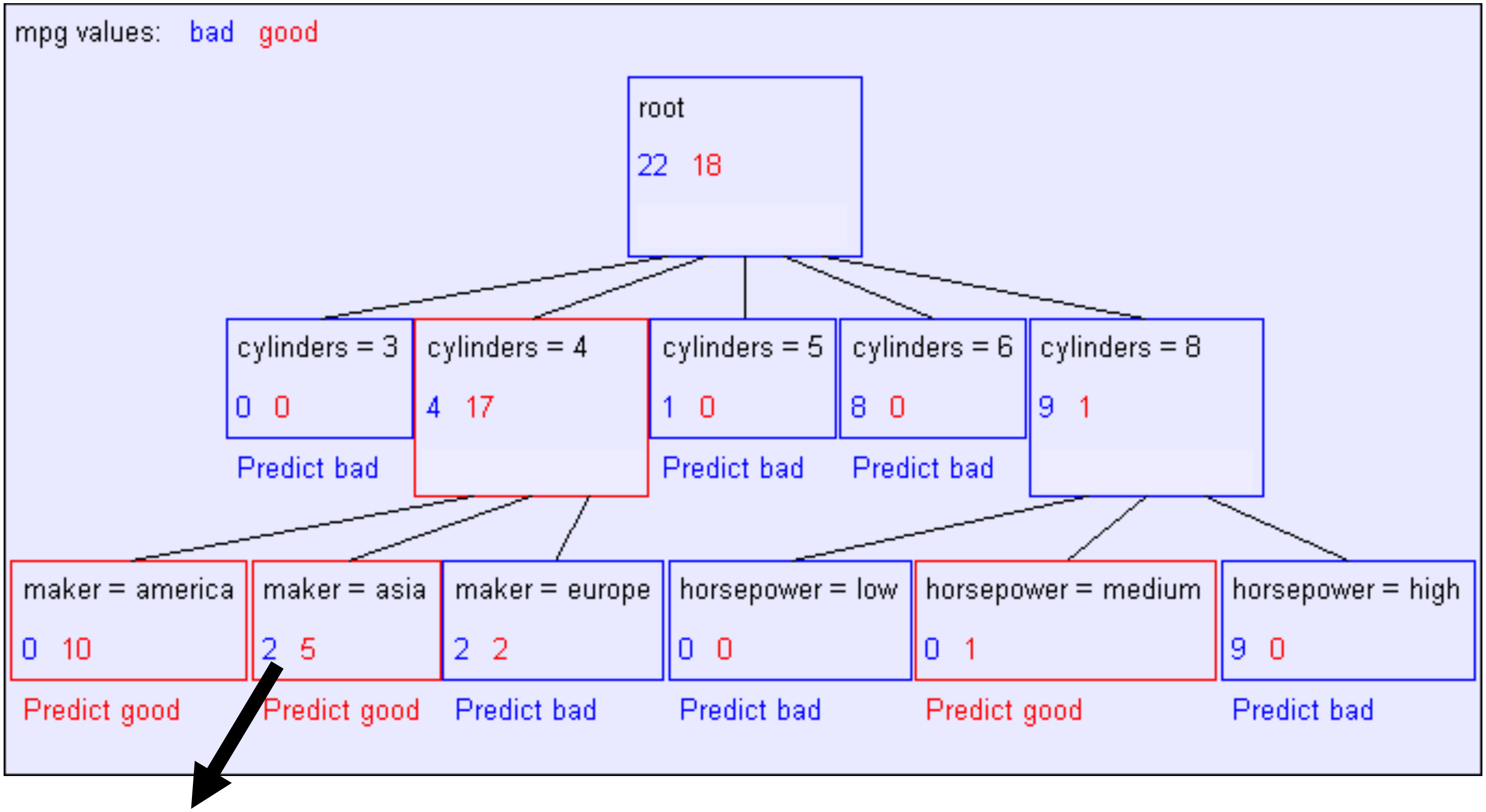
to the value of the attribute we split on



Recursion Step



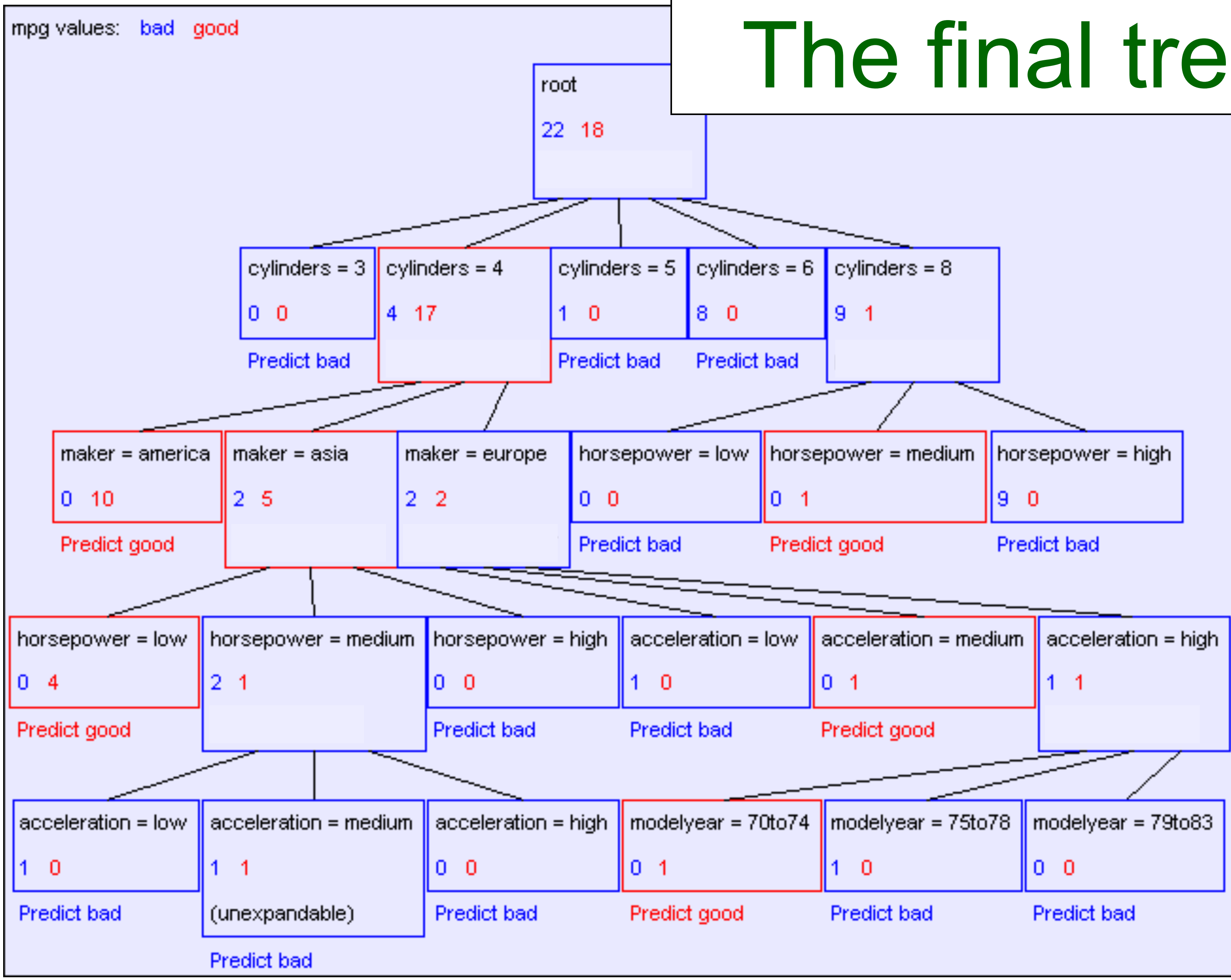
Second level of tree



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



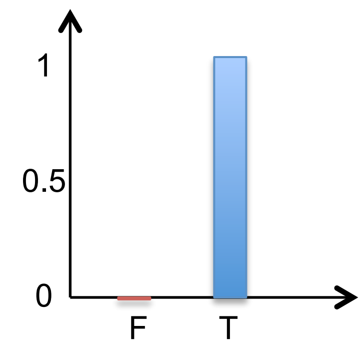
Choosing a good attribute

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

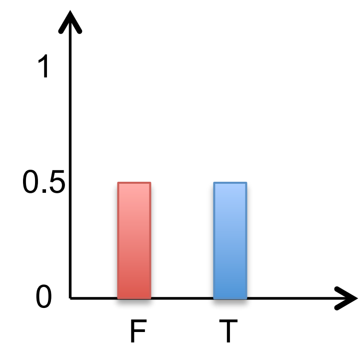
Measuring uncertainty

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

$P(Y=F \mid X_1= T) = 0$	$P(Y=T \mid X_1= T) = 1$



$P(Y=F \mid X_2=F) = 1/2$	$P(Y=T \mid X_2=F) = 1/2$



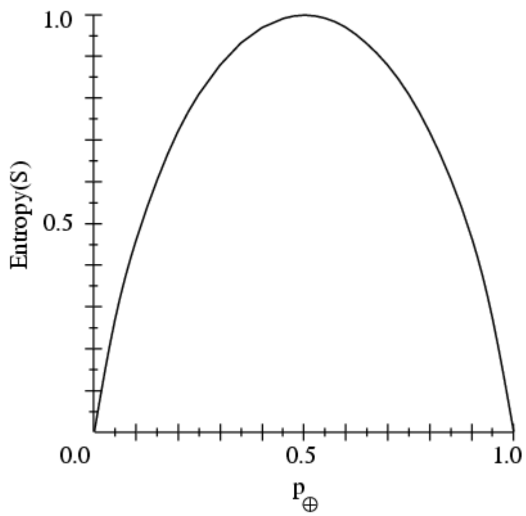
Entropy

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)



Information gain

- 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

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

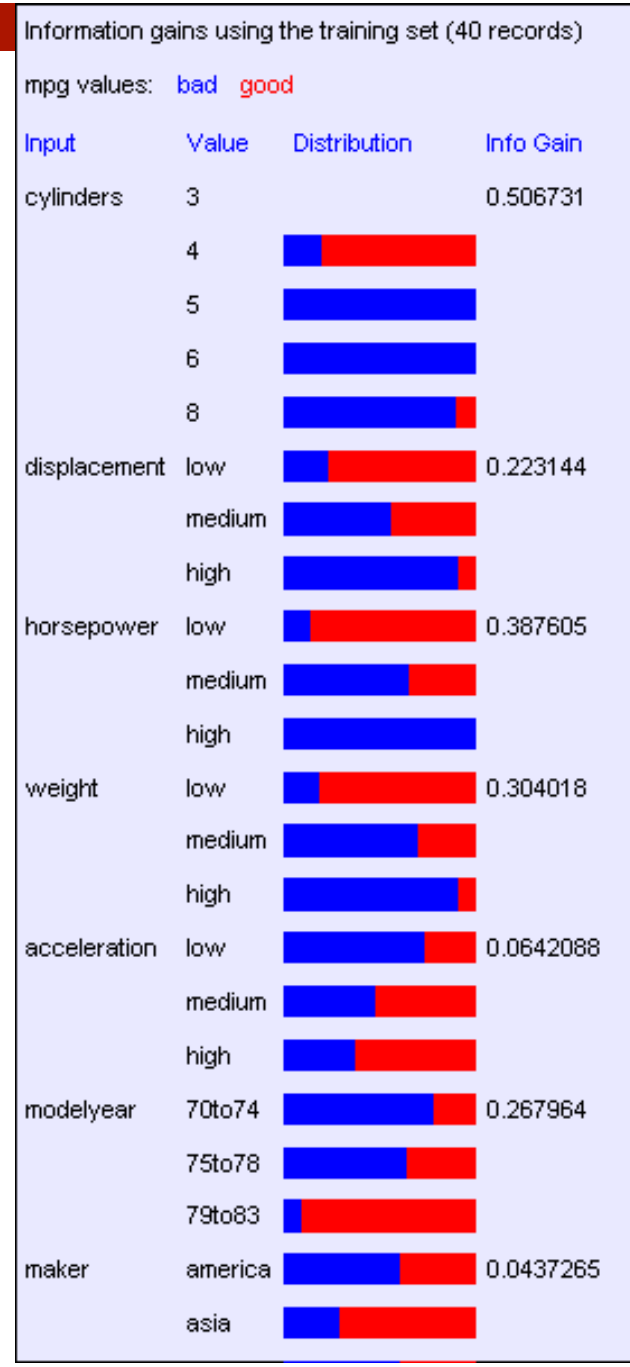
- Information gain is difference $IG(X) = H(Y) - H(Y \mid X)$
  - (Technically it's mutual information; but in this context also referred to as information gain)

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

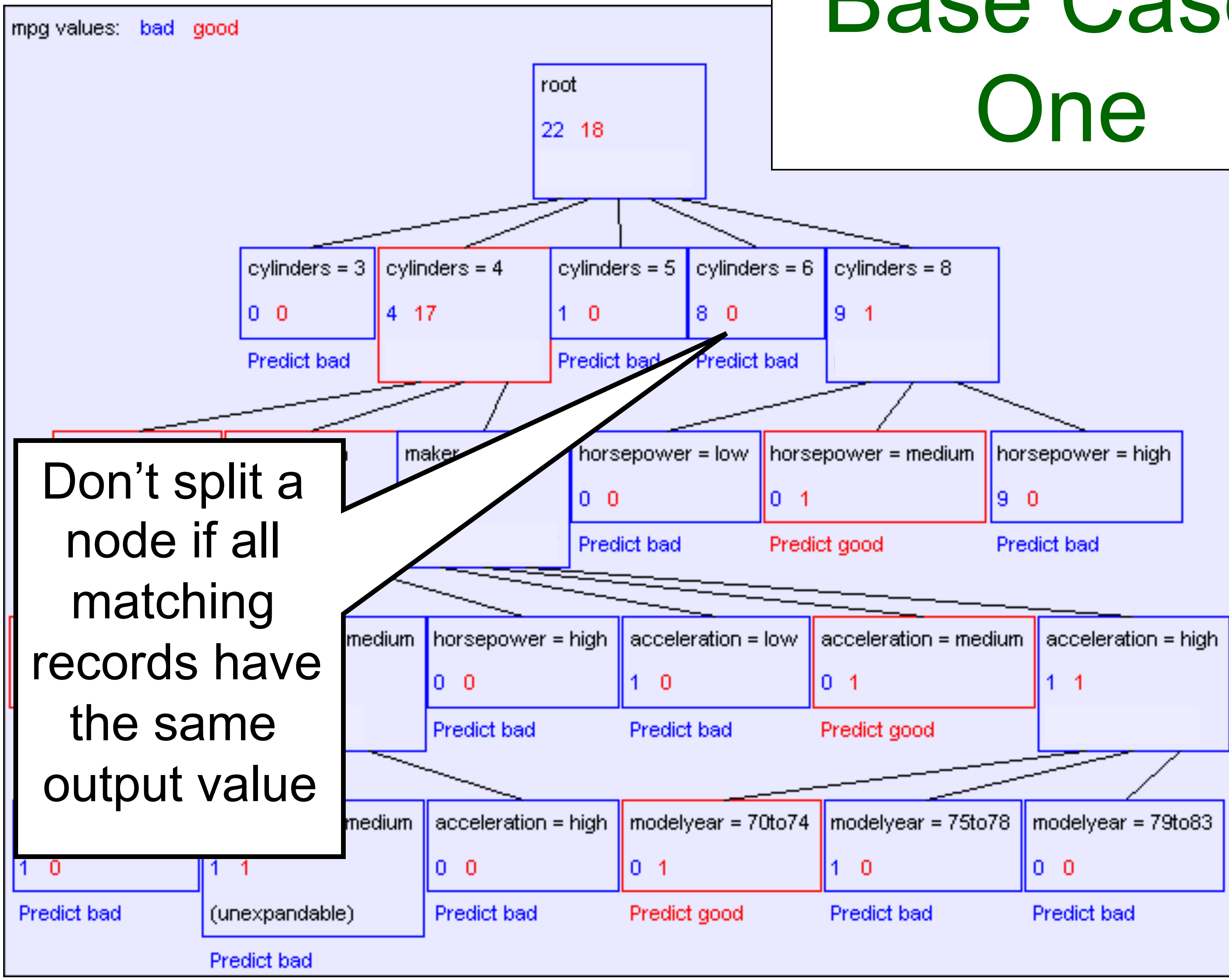
Suppose we want to predict MPG

Look at all the information gains…



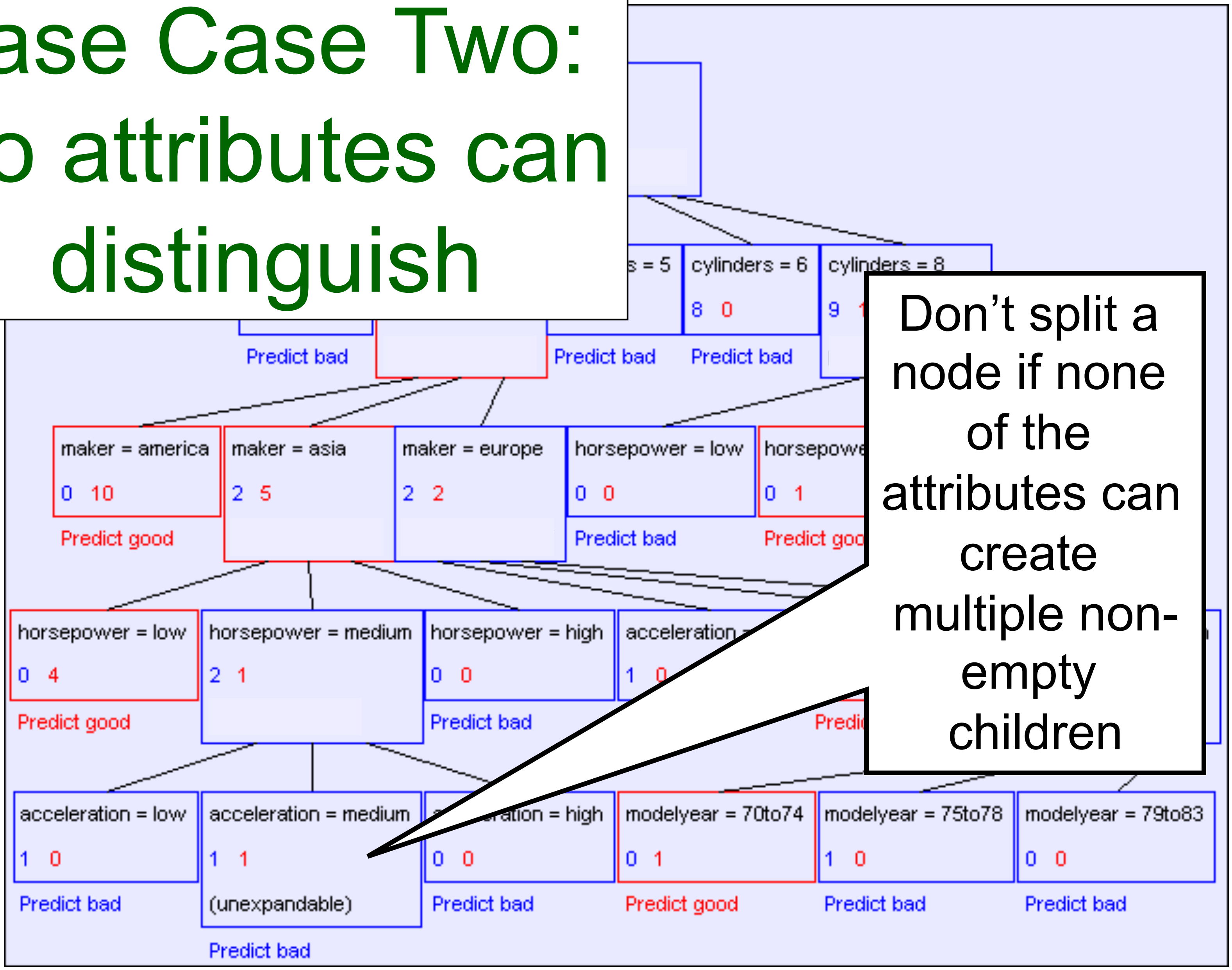
When do we stop?

Base Case One



Don’t split a node if all matching records have the same output value

Base Case Two: No attributes can distinguish



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

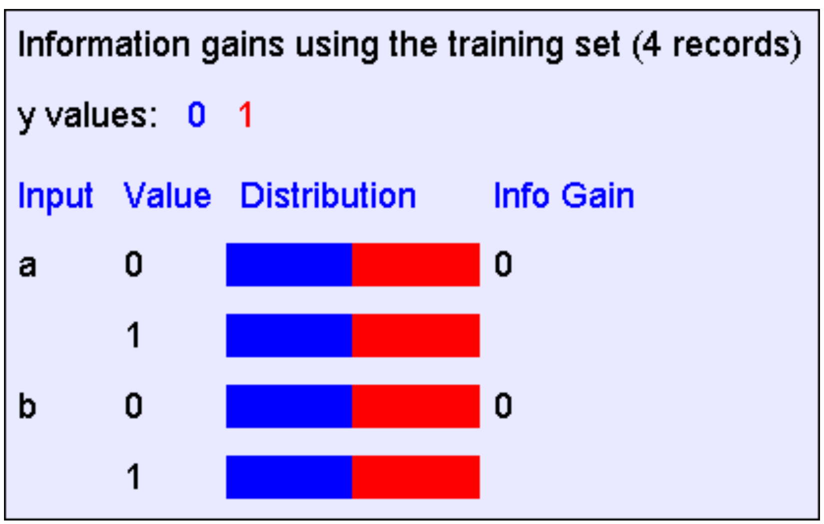
- *Is this a good idea?*

The problem with Base Case 3

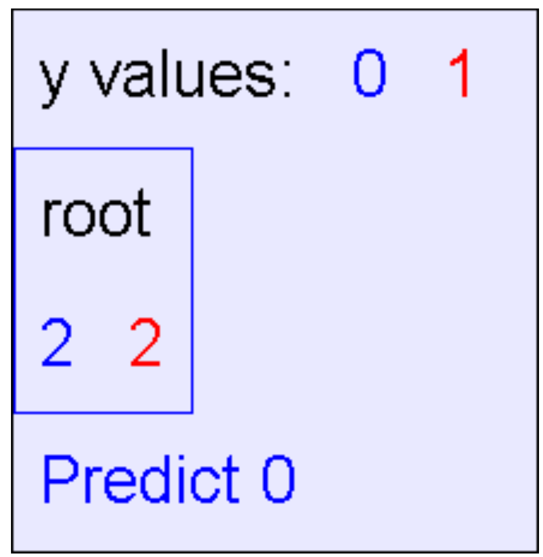
a	b	y
0	0	0
0	1	1
1	0	1
1	1	0

y = a XOR b

The information gains:



The resulting decision tree:

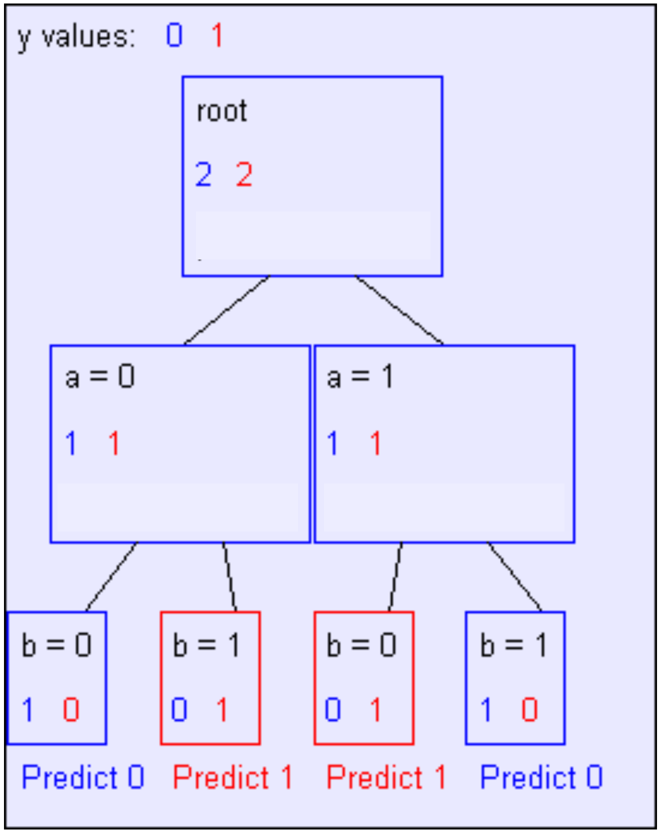


If we omit Base Case 3:

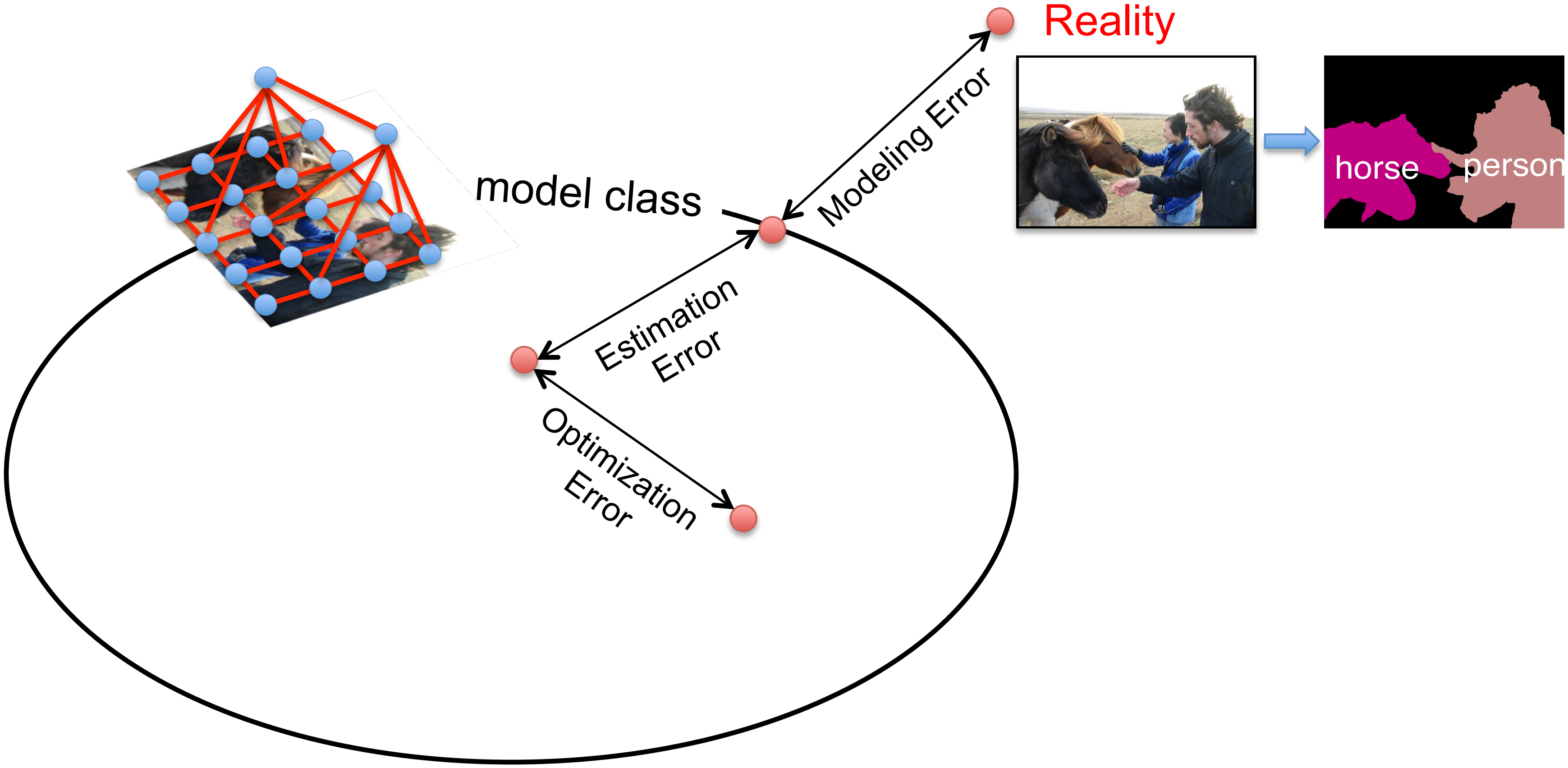
a	b	y
0	0	0
0	1	1
1	0	1
1	1	0

y = a XOR b

The resulting decision tree:



Remember: Error Decomposition



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*)

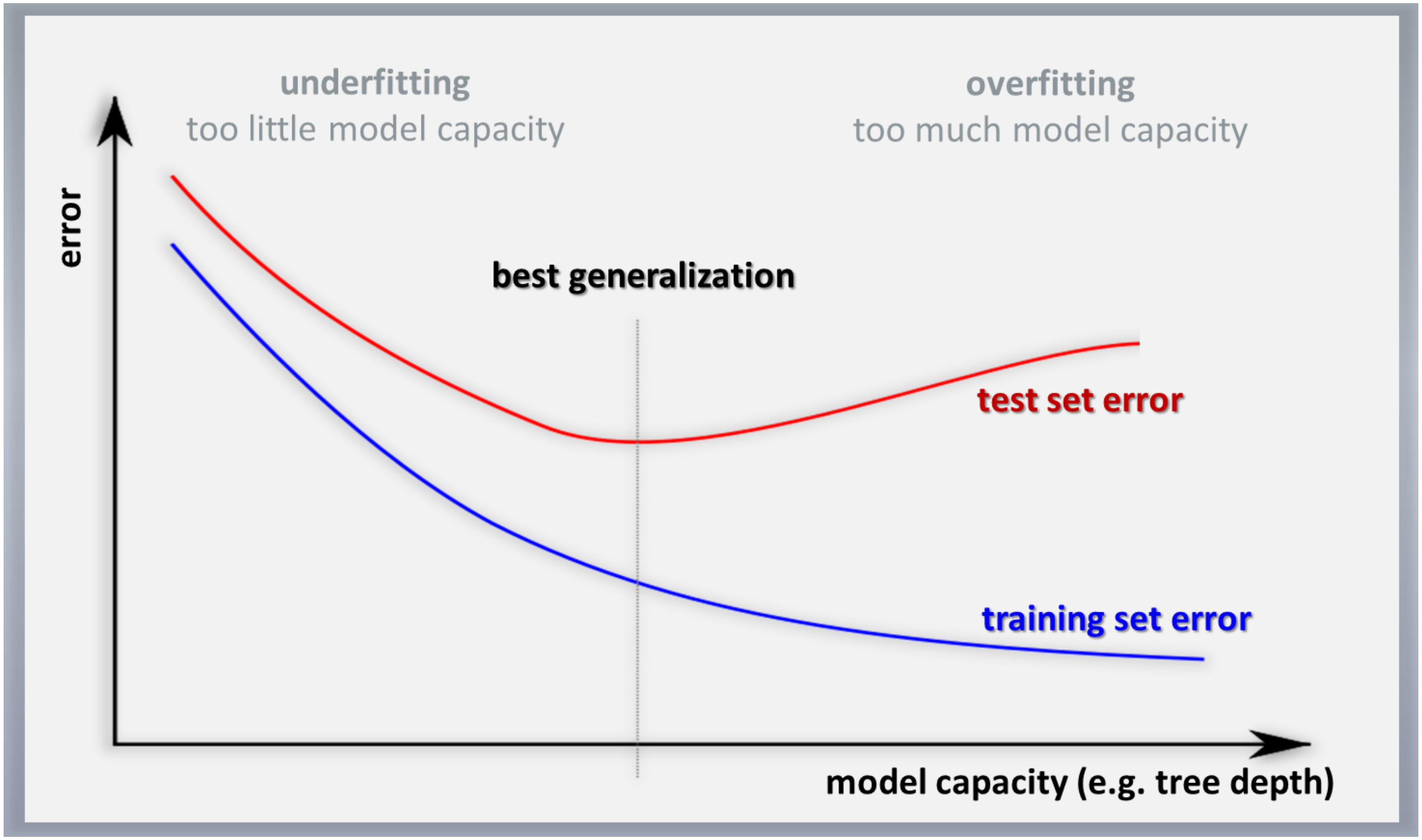
    Where DS_i built consists of all those records in DataSet for which X = ith distinct value of X.

Decision trees will overfit

- Standard decision trees have no prior
  - 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… (chi2 tests)

Decision trees will overfit



Avoiding Overfitting

How can we avoid overfitting?

- Stop growing when data split not statistically significant
- Grow full tree, then post-prune

How to select "best" tree:

- Measure performance over training data
- Measure performance over separate validation data set
- Add complexity penalty to performance measure

Reduced-Error Pruning

Split data into *training* and *validation* set

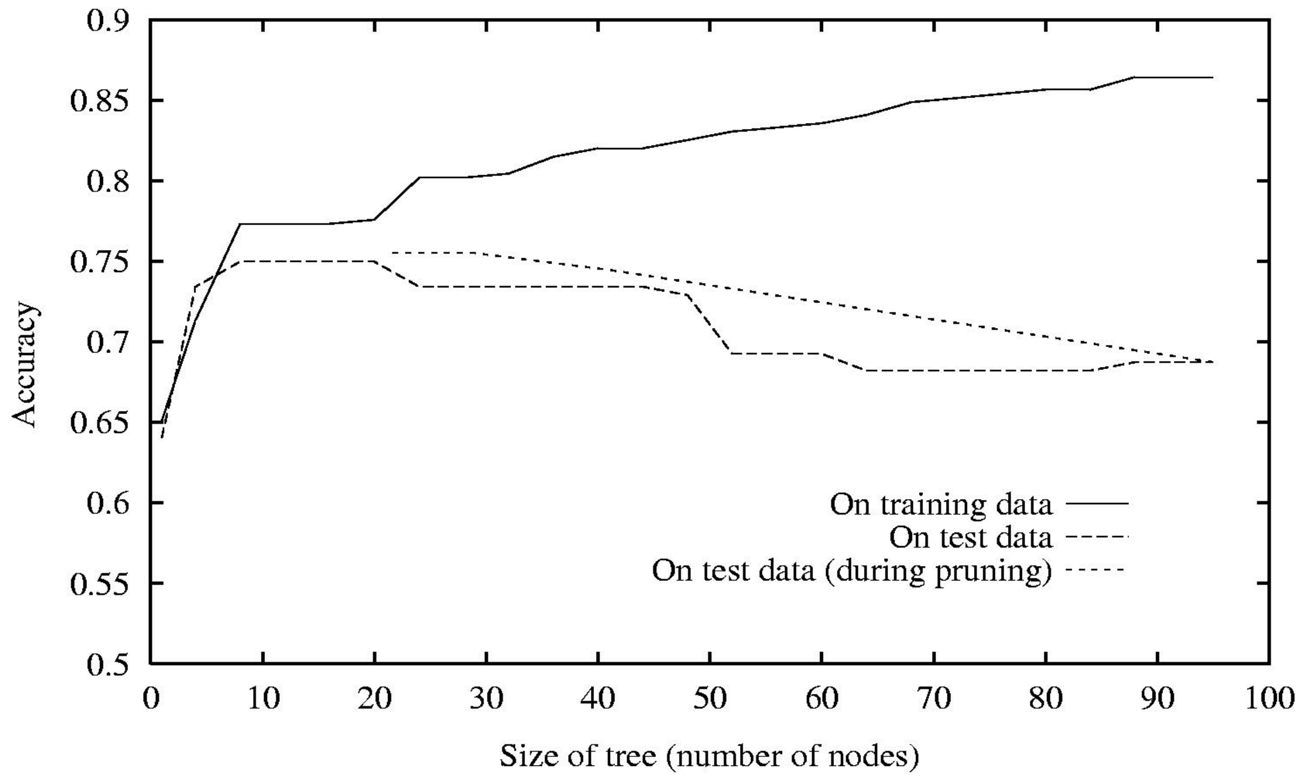
Do until further pruning is harmful:

1. Evaluate impact on *validation* set of pruning each possible node (plus those below it)
2. Greedily remove the one that most improves *validation* set accuracy

Pruning Decision Trees

- Demo
  - http://webdocs.cs.ualberta.ca/~aixplore/learning/DecisionTrees/Applet/DecisionTreeApplet.html

Effect of Reduced-Error Pruning



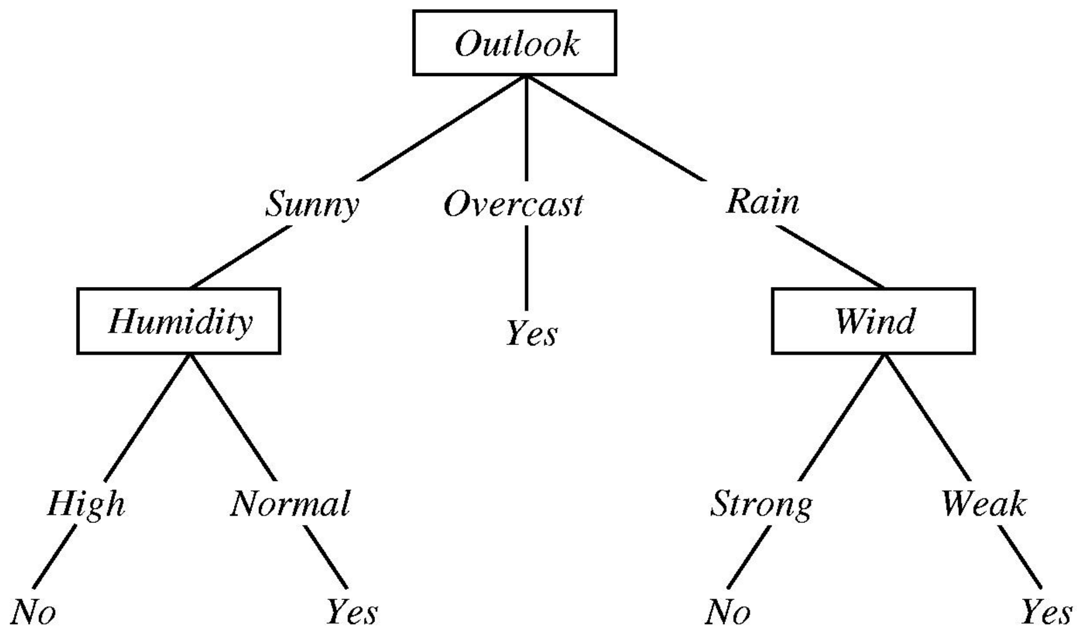
Slide Credit: Pedro Domingos, Tom Mitchel, Tom Dietterich

Rule Post-Pruning

1. Convert tree to equivalent set of rules
2. Prune each rule independently of others
3. Sort final rules into desired sequence for use

Perhaps most frequently used method (e.g., C4.5)

Converting A Tree to Rules



Slide Credit: Pedro Domingos, Tom Mitchel, Tom Dietterich

IF $(Outlook = Sunny)\ AND\ (Humidity = High)$
THEN $PlayTennis = No$

IF $(Outlook = Sunny)\ AND\ (Humidity = Normal)$
THEN $PlayTennis = Yes$

...

Real-Valued inputs

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

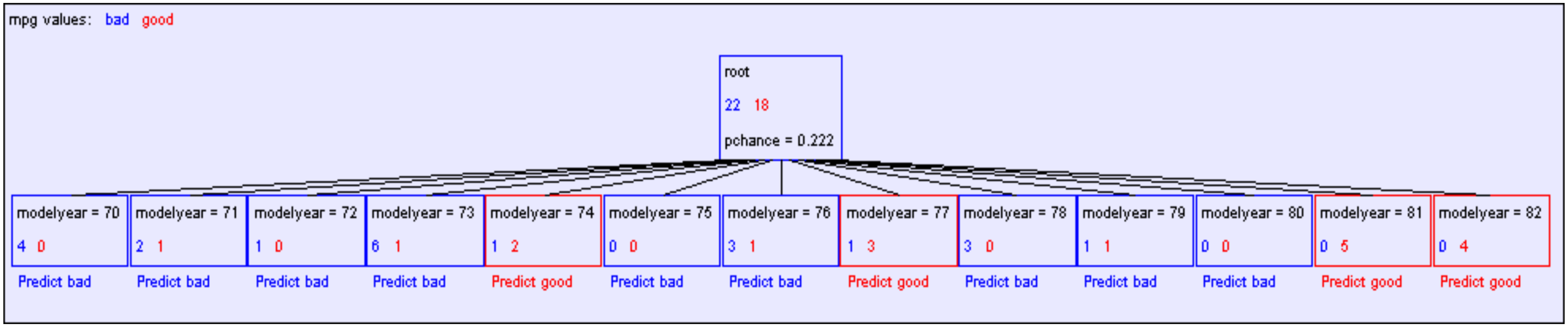
mpg	cylinders	displacemen	horsepower	weight	acceleration	modelyear	maker
good	4	97	75	2265	18.2	77	asia
bad	6	199	90	2648	15	70	america
bad	4	121	110	2600	12.8	77	europe
bad	8	350	175	4100	13	73	america
bad	6	198	95	3102	16.5	74	america
bad	4	108	94	2379	16.5	73	asia
bad	4	113	95	2228	14	71	asia
bad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

“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 >= t

Choosing threshold split

- Binary tree, split on attribute X
  - One branch: X < t
  - Other branch: X >= t

- Search through possible values of *t*
  - Seems hard!!!

- But only finite number of *t*'s are important
  - Sort data according to X into $\{x_1,\ldots,x_n\}$
  - 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)* =

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

  - *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

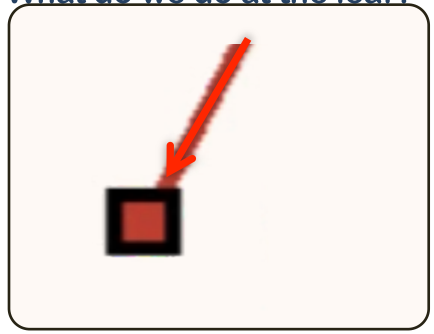
- Note, may split on an attribute multiple times, with different thresholds

Decision Trees

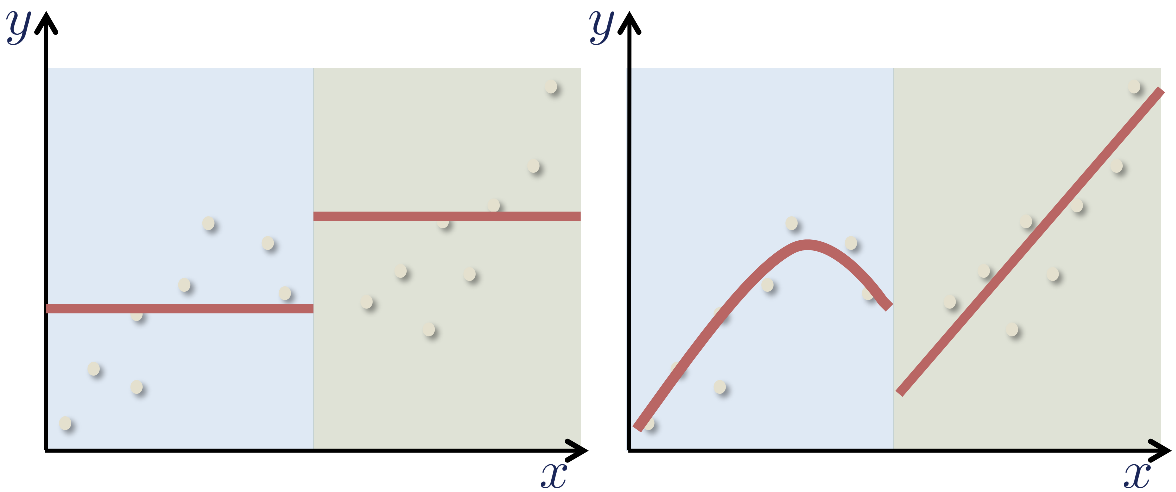
- Demo
  - http://www.cs.technion.ac.il/~rani/LocBoost/

Regression Trees

What do we do at the leaf?



Examples of leaf (predictor) models



Predictor model: constant

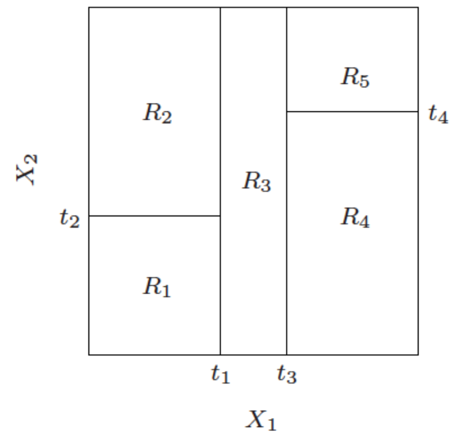
$$y = const$$

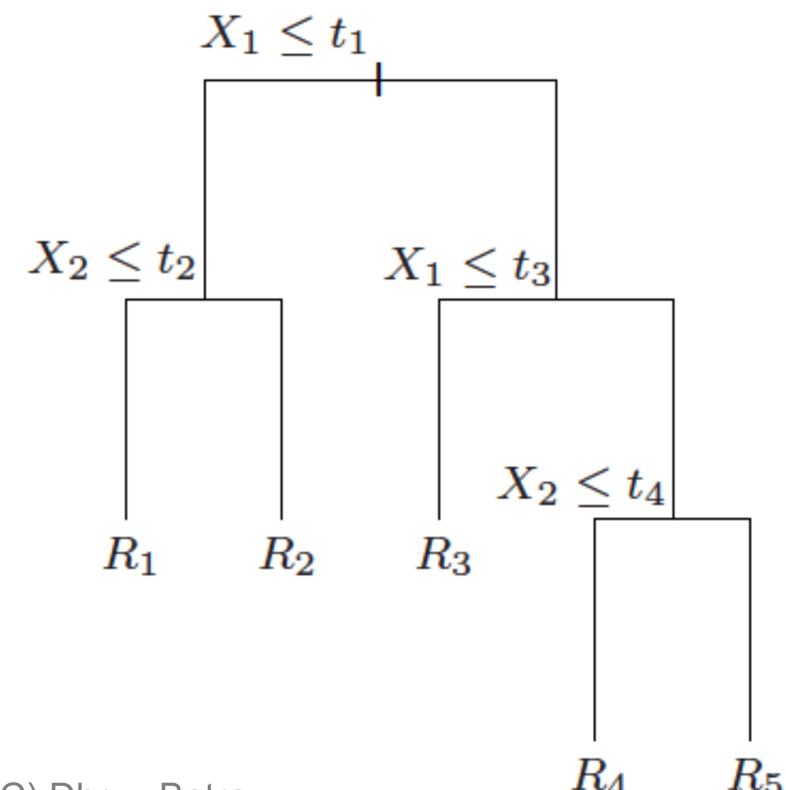
Predictor model: polynomial

$$y = \sum_{i=0}^{n} w_i x^i$$

(note: linear for n=1, constant for n=0)

Regression Trees





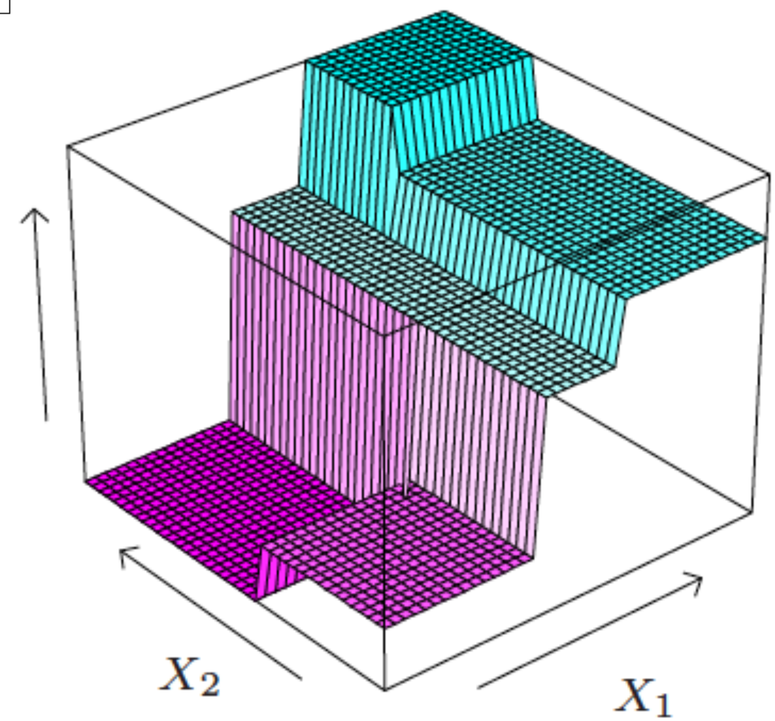
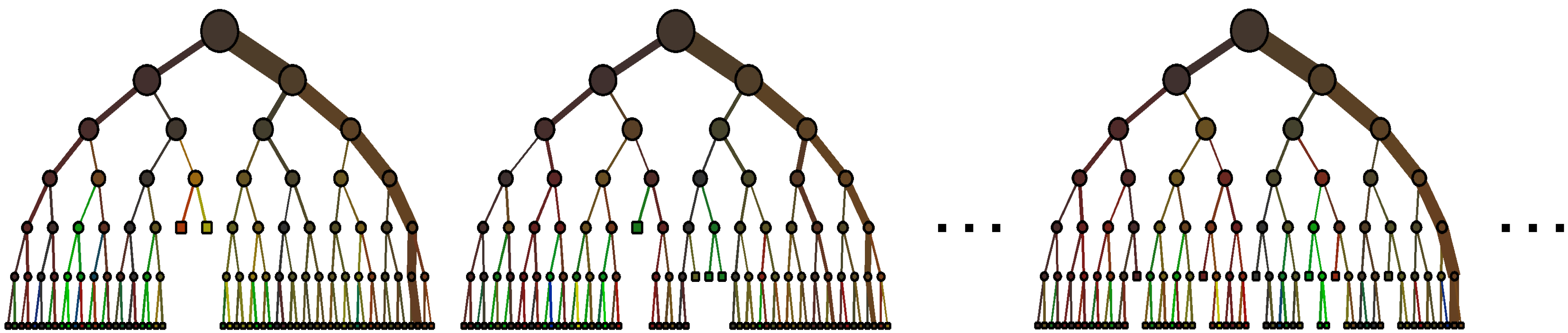


Image Credit: ESL Book

Decision Forests



Learn many trees & Average Outputs

Will formally visit this in Bagging lecture

Image Credit: Jamie Shotton

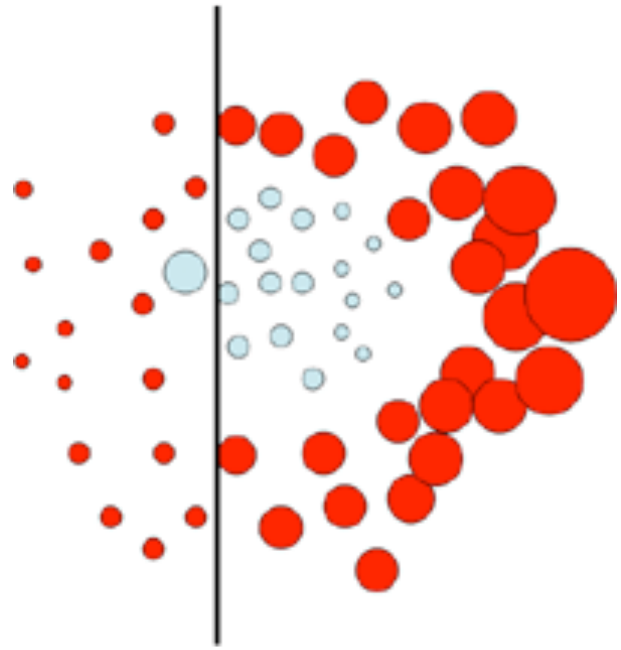
What you need to know about decision trees

- 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

New Topic: Ensemble Methods

Bagging



Boosting

Image Credit: Antonio Torralba

Synonyms

- Ensemble Methods
- Learning Mixture of Experts/Committees
- Boosting types
  - AdaBoost
  - L2Boost
  - LogitBoost
  - <Your-Favorite-keyword>Boost

A quick look back

- So far you have learnt

- Regression
  - Least Squares
  - Robust Least Squares

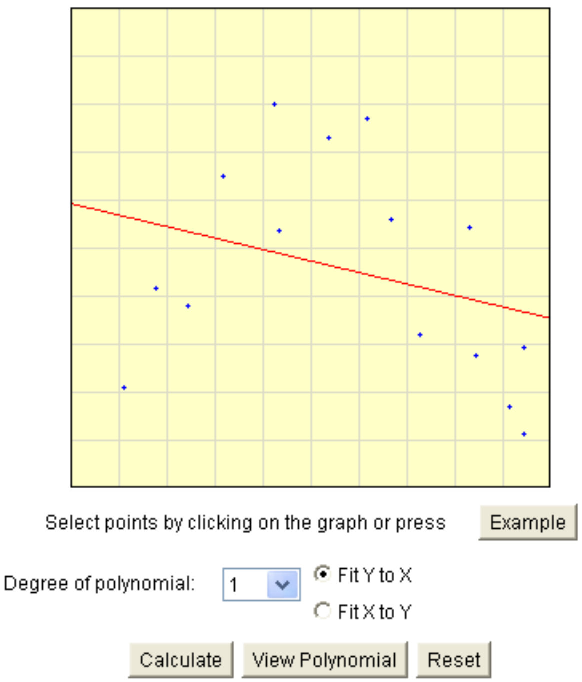
- Classification
  - Linear
    - Naïve Bayes
    - Logistic Regression
    - SVMs
  - Non-linear
    - Decision Trees
    - Neural Networks
    - K-NNs

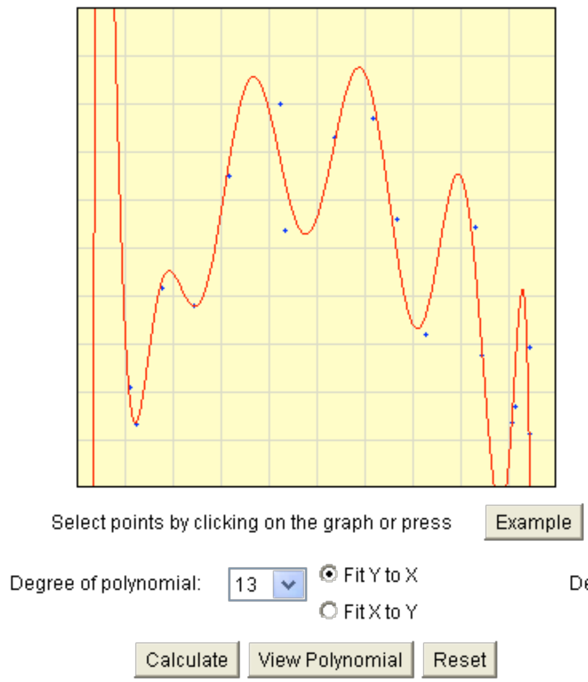
Recall Bias-Variance Tradeoff

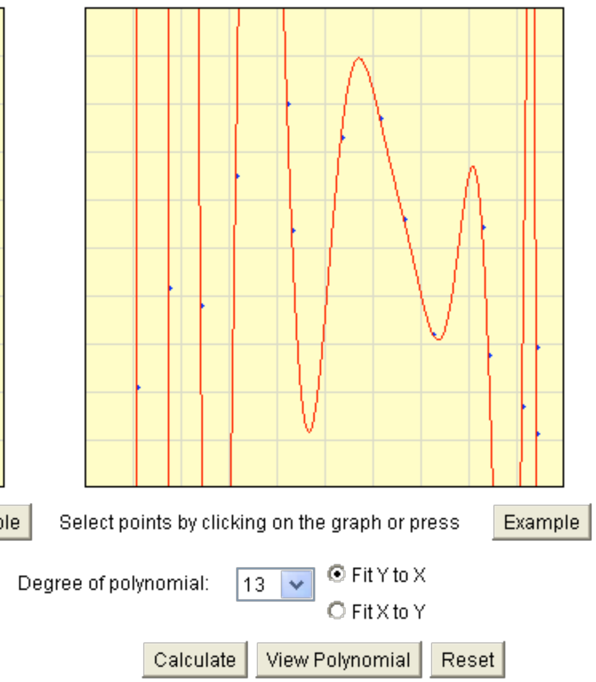
- Demo
  - http://www.princeton.edu/~rkatzwer/PolynomialRegression/

Bias-Variance Tradeoff

- Choice of hypothesis class introduces learning bias
  - More complex class → less bias
  - More complex class → more variance







Fighting the bias-variance tradeoff

- **Simple (a.k.a. weak) learners**
  - e.g., naïve Bayes, logistic regression, decision stumps (or shallow decision trees)
  - **Good:** Low variance, don't usually overfit
  - **Bad:** High bias, can't solve hard learning problems

- **Sophisticated learners**
  - Kernel SVMs, Deep Neural Nets, Deep Decision Trees
  - **Good:** Low bias, have the potential to learn with Big Data
  - **Bad:** High variance, difficult to generalize

- Can we make combine these properties
  - **In general, No!!**
  - **But often yes…**

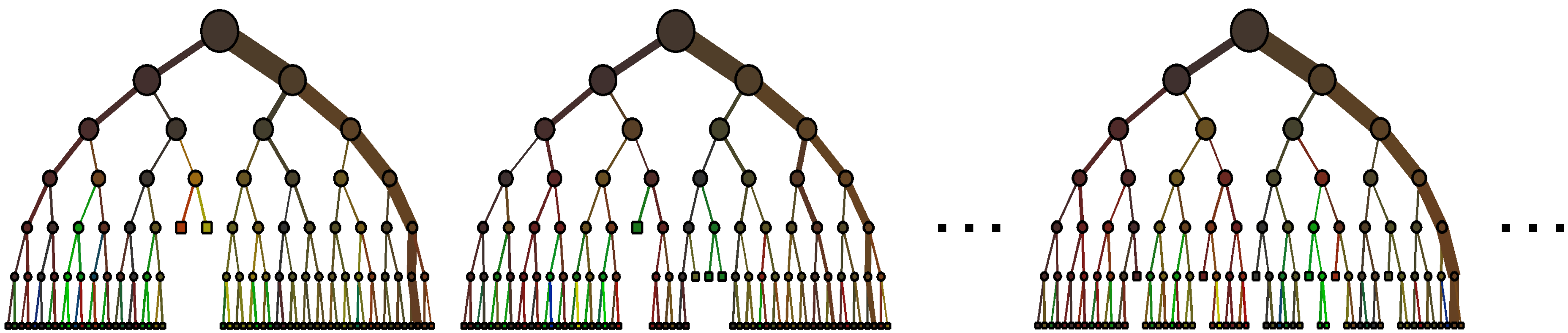
Voting (Ensemble Methods)

- Instead of learning a single classifier, learn **many classifiers**
- **Output class:** (Weighted) vote of each classifier
  - Classifiers that are most "sure" will vote with more conviction
- With sophisticated learners
  - Uncorrelated errors → expected error goes down
  - On average, do better than single classifier!
  - Bagging
- With weak learners
  - each one good at different parts of the input space
  - On average, do better than single classifier!
  - Boosting

Bagging

- Bagging = Bootstrap Averaging
  - On board
  - Bootstrap Demo
    - http://wise.cgu.edu/bootstrap/

Decision Forests



Learn many trees & Average Outputs

Will formally visit this in Bagging lecture