CSE 446: Decision Trees
Winter 2012

Slides adapted from Carlos Guestrin and Andrew Moore by Luke Zettlemoyer & Dan Weld

---

Overview of Learning

Type of Supervision
(eg, Experience, Feedback)

What is Being Learned?

	Labeled Examples	Reward	Nothing
Discrete Function	Classification		Clustering
Continuous Function	Regression		
Policy	Apprenticeship Learning	Reinforcement Learning	

5

A learning problem: predict fuel efficiency

- 40 Records
- Discrete data (for now)
- Predict MPG

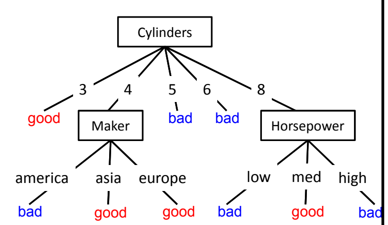
mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	75to78	asia
bad	6	medium	medium	medium	medium	70to74	america
bad	4	medium	medium	medium	low	75to78	europe
bad	8	high	high	high	low	70to74	america
bad	6	medium	medium	medium	medium	70to74	america
bad	4	low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

From the UCI repository (thanks to Ross Quinlan)

Hypotheses: decision trees $f : X \rightarrow Y$

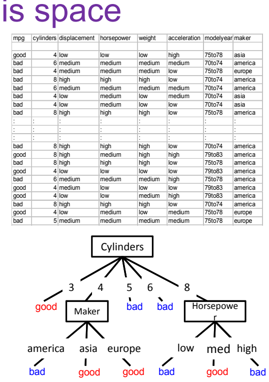
- Each internal node tests an attribute x_i
- Each branch assigns an attribute value $x_i = v$
- Each leaf assigns a class y
- To classify input x?

  traverse the tree from root to leaf, output the labeled y



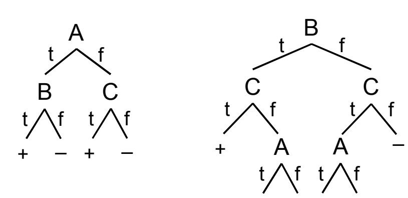
Hypothesis space

- How many possible hypotheses?
- What functions can be represented?
- How many will be consistent with a given dataset?
- How will we choose the best one?



Are all decision trees equal?

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

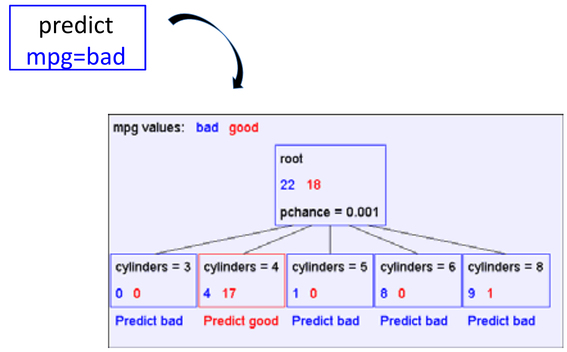


- Which tree do we prefer?

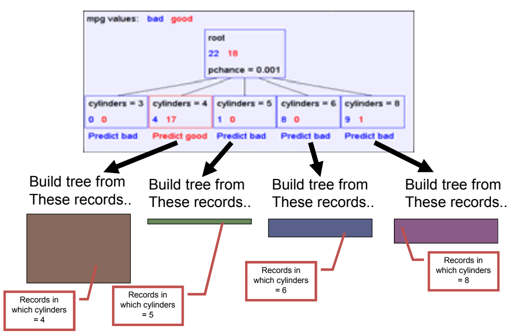
Learning decision trees is hard!!!

- Finding 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

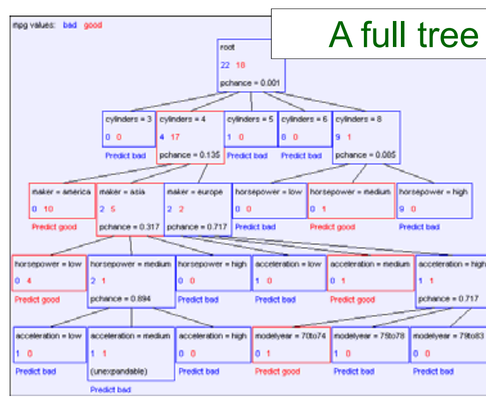
Improving Our Tree



Recursive Step



A full tree



Two Questions

Greedy Algorithm:
- Start from empty decision tree
- Split on the **best attribute (feature)**
- Recurse

1. Which attribute gives the best split?
2. When to stop recursion?

Which attribute gives the best split?

A_1: The one with the highest ***information gain***
Defined in terms of ***entropy***

A_2: Actually many alternatives, eg, ***accuracy***
Seeks to reduce the ***misclassification rate***

30

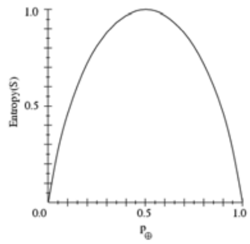
Entropy

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

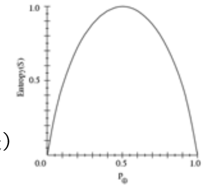
$$H(Y) = -\sum_{i=1}^{k} P(Y = y_i) \log_2 P(Y = y_i)$$

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)



Entropy Example



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

P(Y=t) = 5/6

P(Y=f) = 1/6

H(Y) = - 5/6 $\log_2$ 5/6 - 1/6 $\log_2$ 1/6

= 0.65

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

Conditional Entropy

Conditional Entropy $H(Y|X)$ of a random variable Y conditioned on a random variable X

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

Example:

X_1: t → Y=t : 4, Y=f : 0; f → Y=t : 1, Y=f : 1

$P(X_1$=t) = 4/6

$P(X_1$=f) = 2/6

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

$H(Y|X_1)$ = - 4/6 (1 $\log_2$ 1 + 0 $\log_2$ 0)
- 2/6 (1/2 $\log_2$ 1/2 + 1/2 $\log_2$ 1/2)

= 2/6

= 0.33

Information Gain

Advantage of attribute – decrease in entropy (uncertainty) after splitting

$$IG(X) = H(Y) - H(Y \mid X)$$

In our running example:

$IG(X_1) = H(Y) – H(Y|X_1)$
= 0.65 – 0.33

$IG(X_1) > 0$ → we prefer the split!

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

Alternate Splitting Criteria

- Misclassification Impurity

  Minimum probability that a training pattern will be misclassified

  $M(Y) = 1-\max_i P(Y=y_i)$

- Misclassification Gain

  $IG_M(X) = [1-\max_i P(Y=y_i)] - [1-(\max_j \max_i P(Y=y_i | x=x_j))]$

37

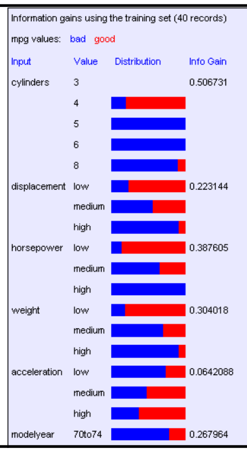
Learning Decision Trees

- Start from empty decision tree
- Split on **next best attribute (feature)**
  - Use information gain (or…?) to select attribute:

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

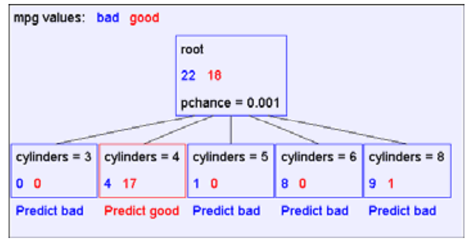
Suppose we want to predict MPG

predict mpg=bad



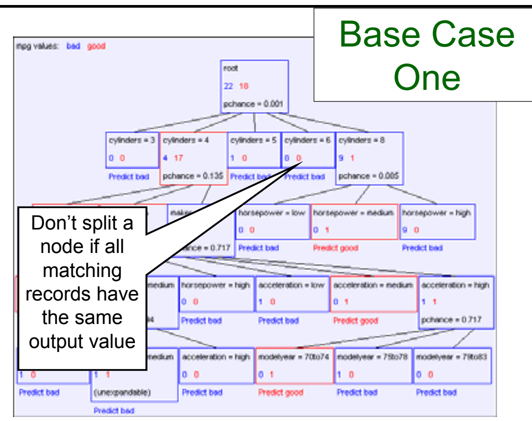
Now, Look at all the information gains…

Tree After One Iteration

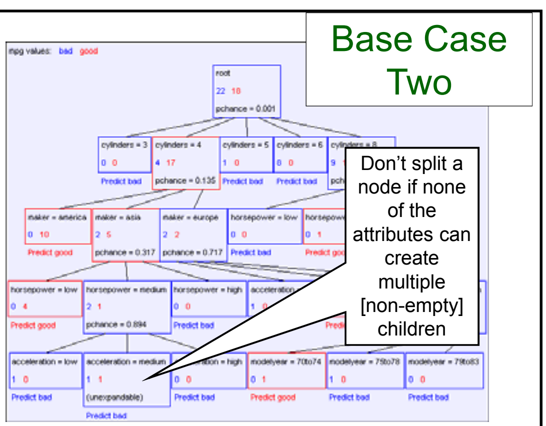


When to Terminate?

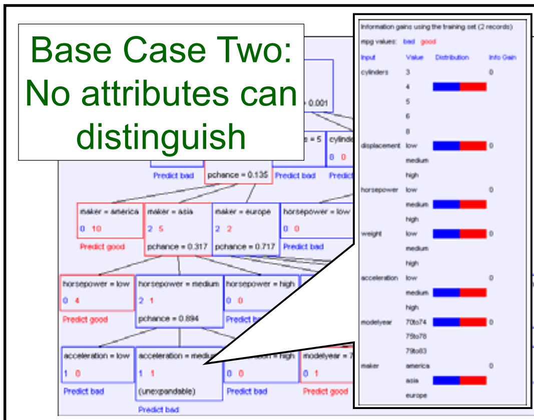
Base Case One



Base Case Two



Base Case Two: No attributes can distinguish



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

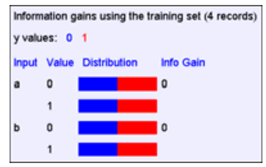
Is this a good idea?

The problem with Base Case 3

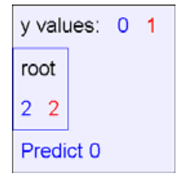
y = a XOR b

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

The information gains:



The resulting decision tree:

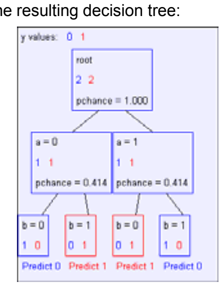


But *Without* Base Case 3:

y = a XOR b

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

The resulting decision tree:



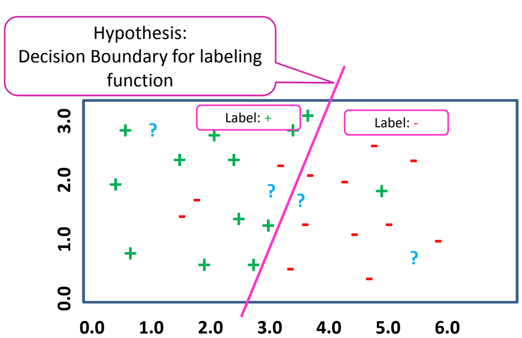
So: Base Case 3? Include or Omit?

Building Decision Trees

BuildTree(*DataSet,Output*)

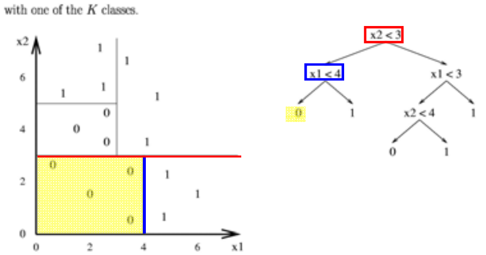
If all output values are the same in *DataSet*,
  Then return a leaf node that says "predict this unique output"
If all input values are the same,
  Then return a leaf node that says "predict the majority output"
Else find attribute *X* with highest Info Gain
  Suppose *X* has n_X distinct values (i.e. X has arity n_X).
  Create and return a non-leaf node with n_X children.
  The i^{th} child is built by calling **BuildTree(*DS_i,Output*)**
    Where DS_i consists of all those records in DataSet
    for which X = i^{th} distinct value of X.

General View of a Classifier



Decision Tree Decision Boundaries

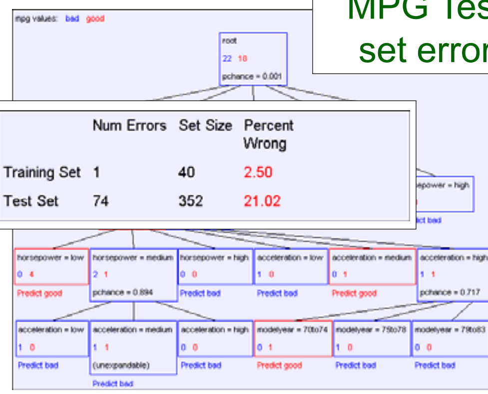
Decision trees divide the feature space into axis-parallel rectangles, and label each rectangle with one of the K classes.



Ok, so how does it perform?

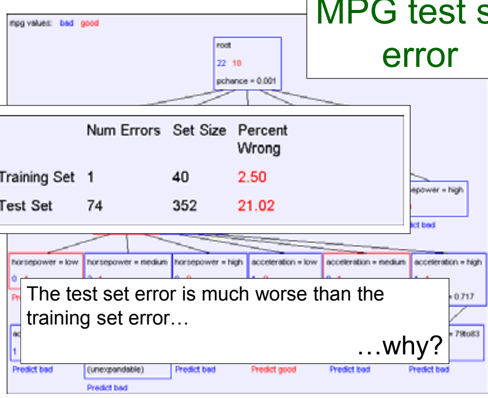
51

MPG Test set error



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

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?

Decision trees will overfit

- Our decision trees have no learning bias
  - Training set error is always zero!
    - (If there is no label noise)
  - Lots of variance
  - Will definitely overfit!!!
  - Must introduce some bias towards *simpler* trees
- Why might one pick simpler trees?

Occam's Razor

- Why Favor Short Hypotheses?
- Arguments for:
  - Fewer short hypotheses than long ones
    - →A short hyp. less likely to fit data by coincidence
    - →Longer hyp. that fit data may might be coincidence
- Arguments against:
  - Argument above on really uses the fact that hypothesis ***space*** is small!!!
  - What is so special about small sets based on the ***complexity*** of each ***hypothesis***?

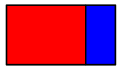
How to Build Small Trees

Several reasonable approaches:

- **Stop growing tree before overfit**
  - Bound depth or # leaves
  - Base Case 3
  - *Doesn't work well in practice*
- **Grow full tree; then prune**
  - **Optimize on a held-out (development set)**
    - If growing the tree hurts performance, then cut back
    - Con: Requires a larger amount of data…
  - **Use statistical significance testing**
    - Test if the improvement for any split is likely due to noise
    - If so, then prune the split!
  - **Convert to logical rules**
    - Then simplify rules

Reduced Error Pruning

Split data into ***training*** & ***validation*** sets (10-33%)

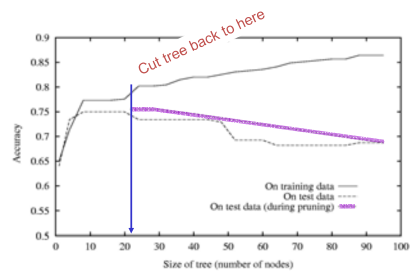


Train on training set (overfitting)

Do until further pruning is harmful:

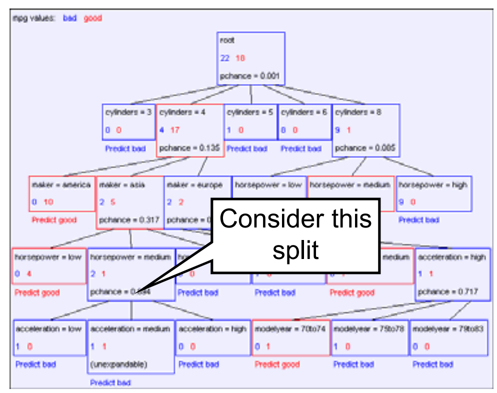
1) Evaluate effect on validation set of pruning ***each*** possible node (and tree below it)
2) Greedily remove the node that ***most improves accuracy of validation set***

Effect of Reduced-Error Pruning

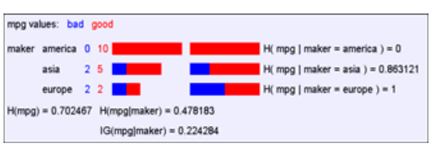


Alternatively

- Chi-squared pruning
  - Grow tree fully
  - Consider leaves in turn
    - Is parent split worth it?
- Compared to Base-Case 3?



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 13.5%

Such hypothesis tests are relatively easy to compute, but involved

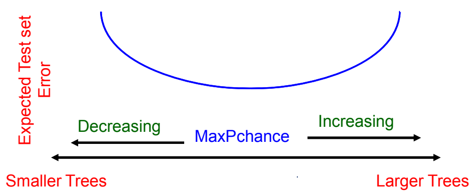
Using Chi-squared to avoid overfitting

- 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

Regularization

- Note for Future: MaxPchance is a regularization parameter that helps us bias towards simpler models



We'll learn to choose the value of magic parameters like this one later!

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
- Improved by
  - Carlos Guestrin &
  - Luke Zettlemoyer