Machine Learning as Search & as Continuous Optimization

CSE 573

DANIEL WELD

1

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, Dan Weld

Logistics

PS2 due Sat 2/2

3

Machine Learning

Study of algorithms that

improve their performance

at some task

with experience

Space of ML Problems

Type of Supervision
(eg, Experience, Feedback)

What is Being Learned?

	Labeled Examples	Reward	Examples w/o labels
Discrete Function	Classification		Clustering
Continuous Function	Regression		
Policy	Apprenticeship Learning	Reinforcement Learning	

5

Classification

from data to discrete classes

Task: Predicting class membership (eg spam or not?)

Output = F: messages → labels {spam, not}

Performance: Accuracy of prediction

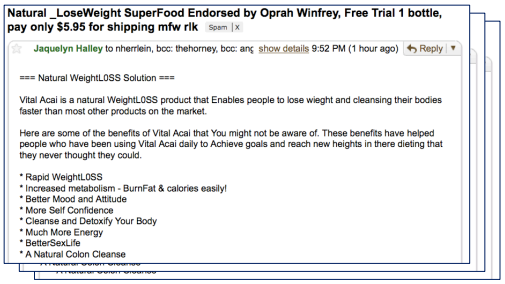
Learning as Function Approximation

Experience: Labeled examples

{ … <message$_i$, label$_i$>… }

6

Training Data for Spam Filtering



"Features"

a	…	homework	…	viagra	…	label
5		0		2		spam
7		1		0		ham

7

Weather prediction

8

Object Recognition

{Cat, Not-Cat}

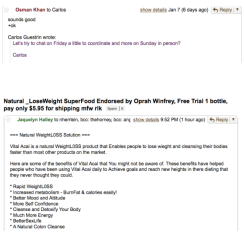
P(Cat)

Multiple type of objects?
Bounding boxes

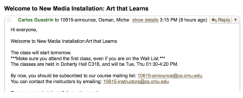
9

The classification pipeline

Training

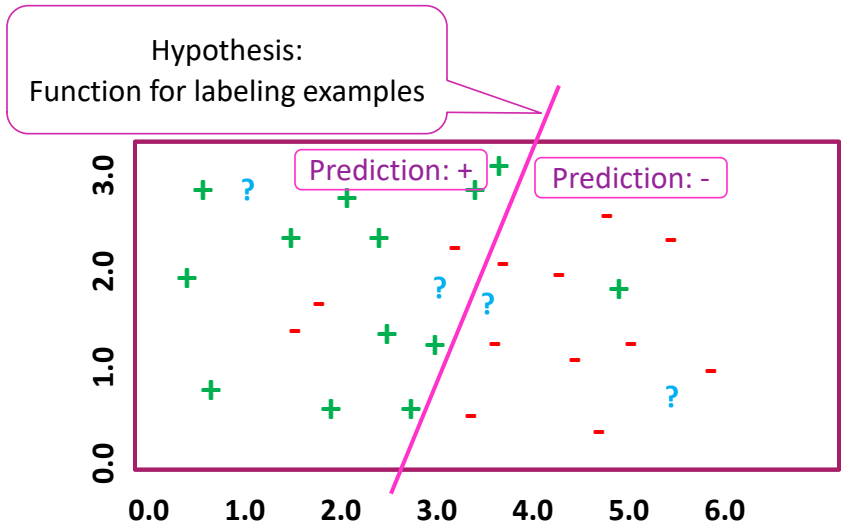


Testing



Measure accuracy here

Classifier



Key Concepts

13

Generalization

Hypotheses must ***generalize*** to correctly classify instances not in the training data.

Simply memorizing training examples is a consistent hypothesis ***that does not generalize***.

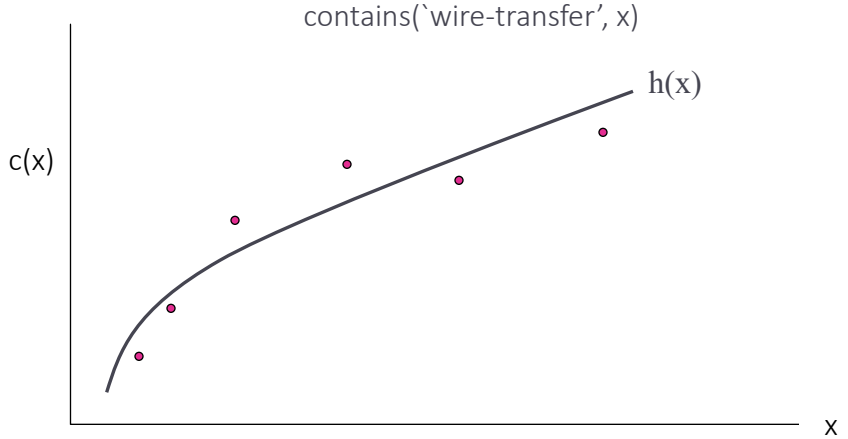
14

ML = Function Approximation

May not be any perfect fit

Classification ~ discrete functions

h(x) = contains(`nigeria', x) ∧
contains(`wire-transfer', x)



15

Why is Learning Possible?

Experience alone never justifies any conclusion about any unseen instance.

Learning occurs when

PREJUDICE meets DATA!

16

Bias

The nice word for prejudice is "bias".
- Different from "Bias" in statistics

What kind of hypotheses will you ***consider***?
- What is allowable ***range*** of functions you use when approximating?
- E.g., pure conjunctions, linear separators, …

What kind of hypotheses do you ***prefer***?
- E.g., simple with few parameters

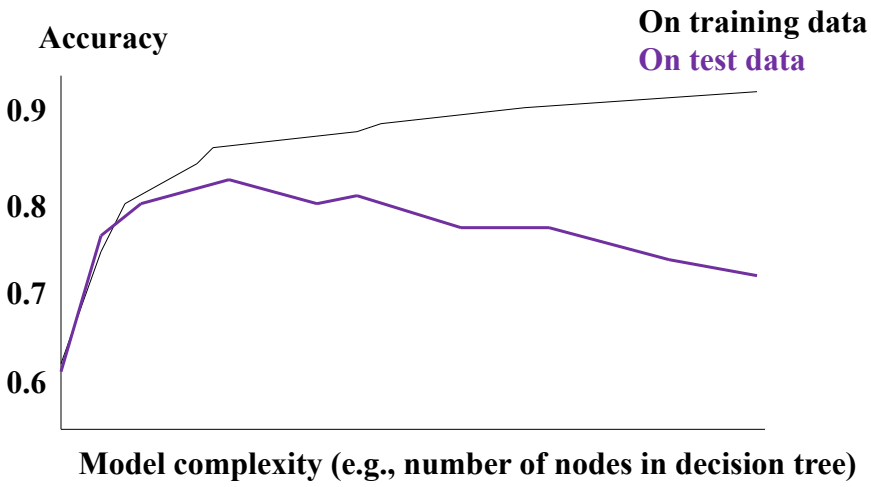
"It is needless to do more when less will suffice"
– William of Occam,
died 1349 of the Black plague

17

Overfitting

Hypothesis H is *overfit* when ∃ H' and
- H has ***smaller*** error on training examples, but
- H has ***bigger*** error on test examples

Overfitting



Overfitting

Hypothesis H is *overfit* when ∃ H' and
- H has ***smaller*** error on training examples, but
- H has ***bigger*** error on test examples

Causes of overfitting
- Training set is too small
- Large number of features

Big Problem for ML

Some solutions
- Validation set
- Regularization

A learning problem: predict fuel efficiency

From the UCI repository (thanks to Ross Quinlan)

- 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

Y X

Need to find "Hypothesis": $f : X \rightarrow Y$

How Represent Function?

f (

cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
4	low	low	low	high	75to78	asia

) → mpg: good

General Propositional Logic?

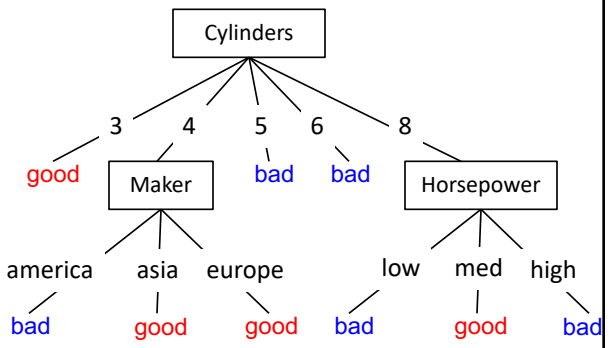
maker=asia $\vee$ weight=low

Need to find "Hypothesis": $f : X \rightarrow Y$

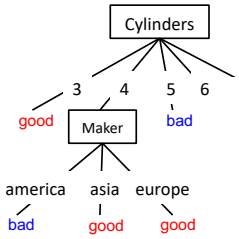
Hypotheses: decision trees $f : X \rightarrow Y$

- 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



What functions can be represented by D.T.s?



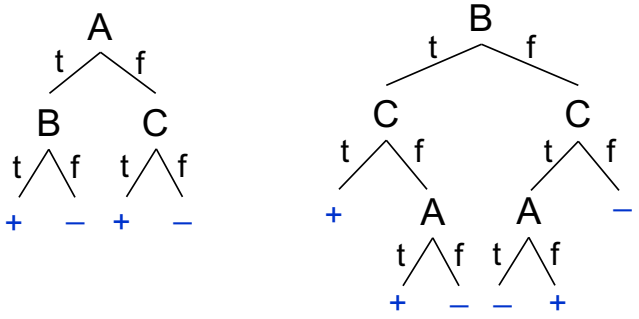
cyl=3 ∨ (cyl=4 ∧ (maker=asia ∨ maker=europe)) ∨ …

Are all decision trees equal?

Many trees can represent the same concept

But, not all trees will have the same size!

e.g., ϕ = (A ∧ B) ∨ (¬A ∧ C)



How to find the best tree?

Learning decision trees is hard!!!

Finding the simplest (smallest) decision tree is an NP-complete problem [Hyafil & Rivest '76]

What to do?

Learning as Search

Learning as Search

Nodes?

Operators?

Start State?

Goal?

Search Algorithm?

Heuristic?

36

The Starting Node: What is the Simplest Tree?

predict
mpg=bad

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

Is this a good tree?

[22+, 18-] ⟵ Means:
correct on 22 examples
incorrect on 18 examples

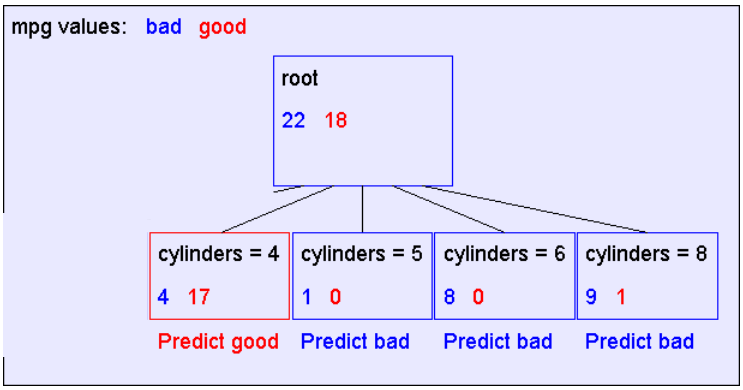
What search algorithm?

- Depth-First Search
- Breadth-First Search
- Iterative Deepening Search
- Uniform-Cost Search
- A*
- IDA*
- ***Hill climbing***
- Local search

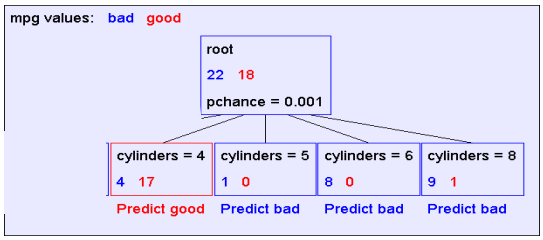
38

Operators: Improving the Tree

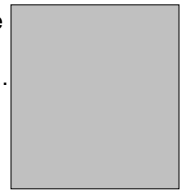
predict
mpg=bad



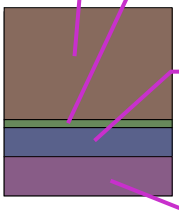
Improving the Tree



Take the Original Dataset..



And partition it according to the value of the attribute we split on



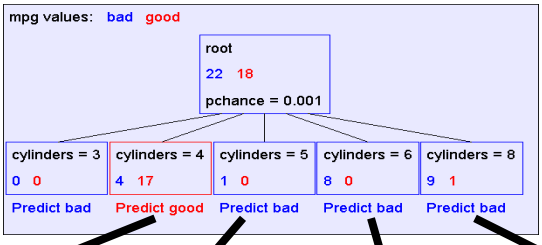
Records in which cylinders = 4

Records in which cylinders = 5

Records in which cylinders = 6

Records in which cylinders = 8

Keep Hill Climbing



Build tree from These records..

Records in which cylinders = 4

Build tree from These records..

Records in which cylinders = 5

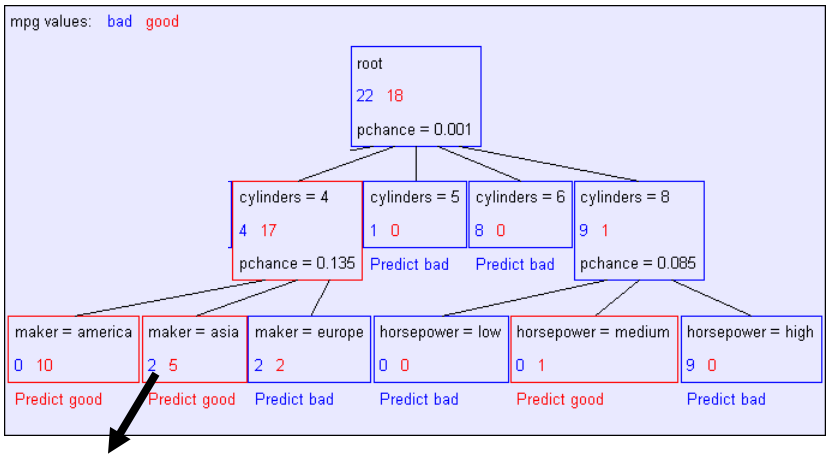
Build tree from These records..

Records in which cylinders = 6

Build tree from These records..

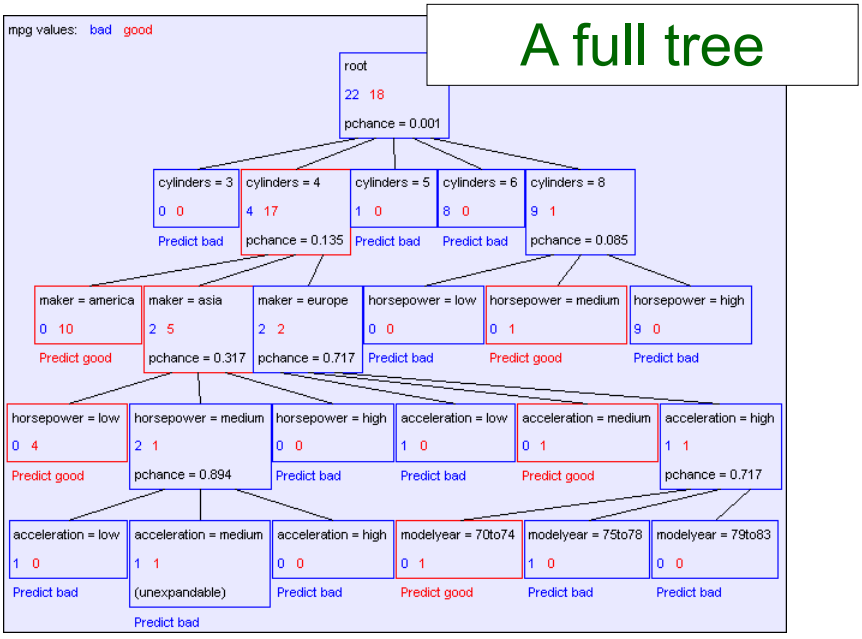
Records in which cylinders = 8

Eventually



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

A full tree



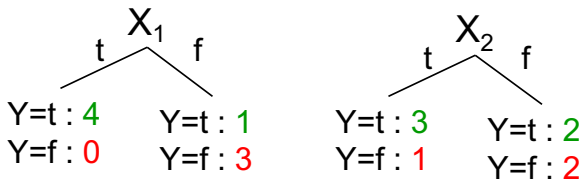
Two Questions

Hill Climbing 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?

Splitting: choosing a good attribute

Would we prefer to split on X_1 or X_2?



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

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
- What about distributions in between?

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

P(Y=A) = 1/3	P(Y=B) = 1/4	P(Y=C) = 1/4	P(Y=D) = 1/6

Measuring uncertainty

Good split if we are more certain about classification after split

- Deterministic good (all true or all false)
- Uniform distribution? Bad
- What about distributions in between?
- What about distributions over more values?

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

P(Y=A) = 1/3	P(Y=B) = 1/4	P(Y=C) = 1/4	P(Y=D) = 1/6

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***

50

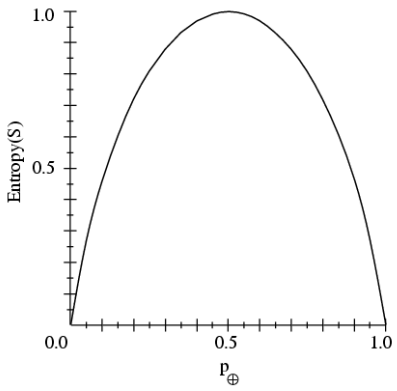
Entropy

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

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

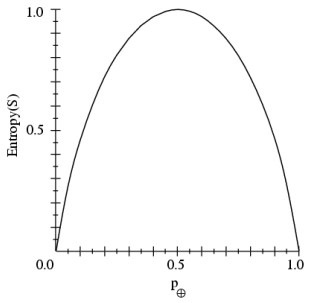
More uncertainty, more entropy!

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

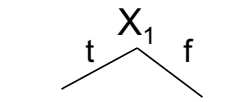
X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F

Conditional Entropy

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



Y=t : 4
Y=f : 0

Y=t : 1
Y=f : 1

P(X_1=t) = 4/6
P(X_1=f) = 2/6

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

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

Information Gain

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

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

In our running example:

$IG(X_1) = H(Y) - H(Y|X_1)$
$= 0.65 - 0.33 = .32$

$IG(X_2) = H(Y) - H(Y|X_2)$
$= 0.65 - 0.5 = .15$

$IG(X_1) > IG(X_2)$ → better to split on X_1

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

Learning Decision Trees

Start from empty decision tree

Choose a good leaf to split

Split on **best attribute (feature)**

- Use information gain (or…?) to select attribute:

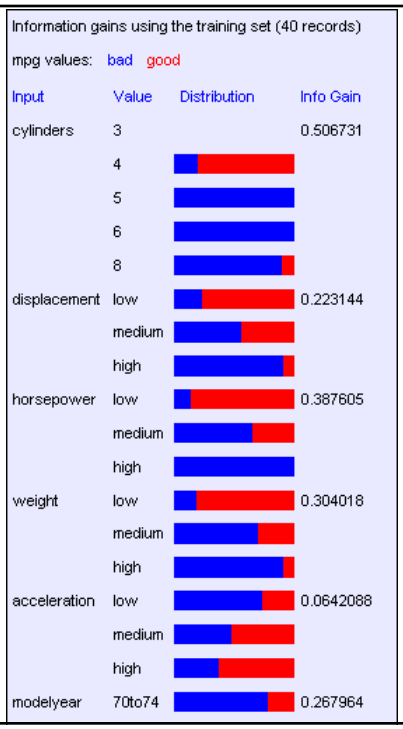
$$\arg\max_i IG(X_i) = \arg\max_i H(Y) - H(Y \mid X_i)$$

Recurse (keep hill climbing)

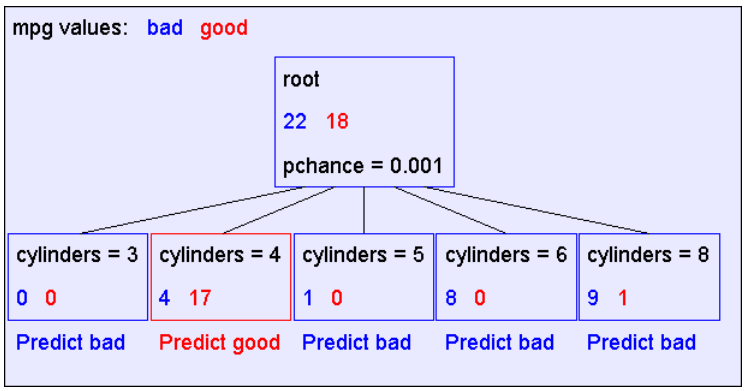
Suppose we want to predict MPG

predict mpg=bad

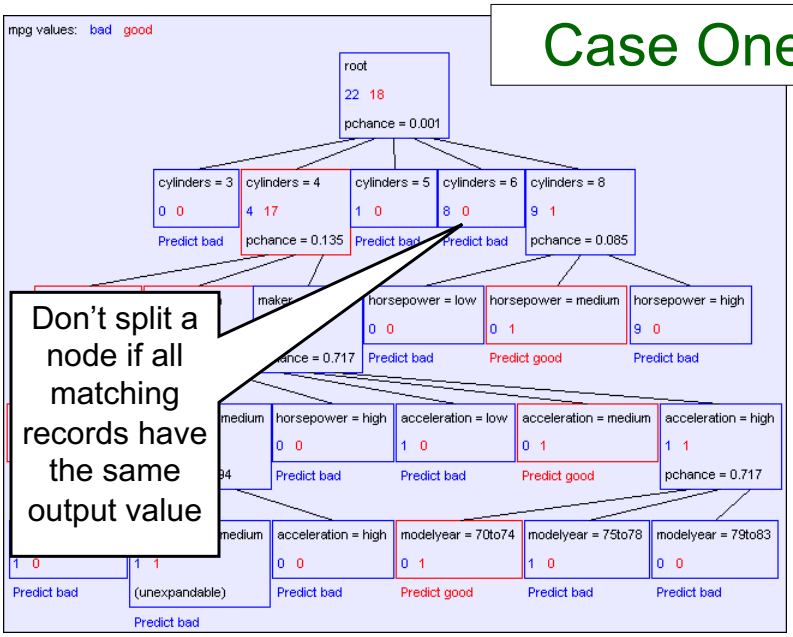
Now, Look at all the information gains…



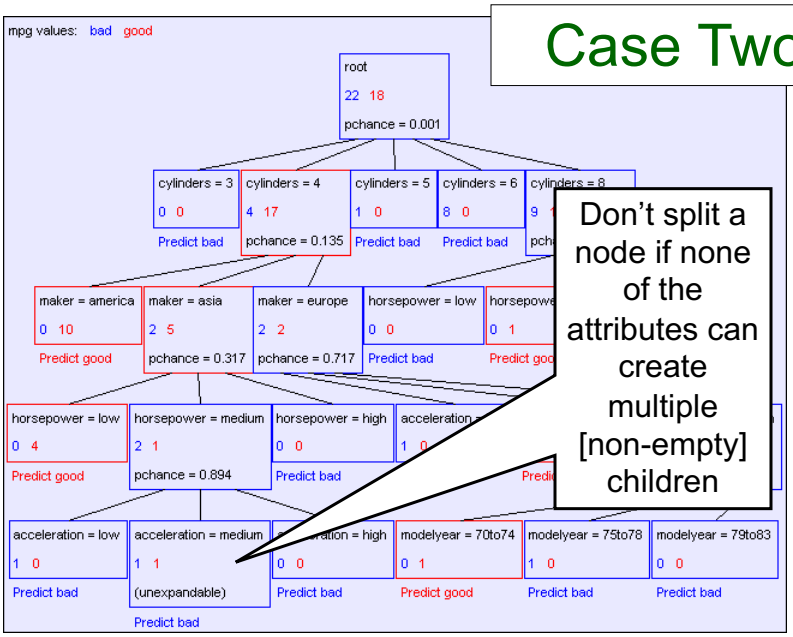
Tree After One Iteration



When to Terminate?



Case Two



Generalized Termination Criterion?

Case One: If all records in current data subset have the same output then don't recurse

Case Two: If all records have exactly the same set of input attributes then don't recurse

Proposed Base Case 3:
If no attribute has positive information gain then don't recurse

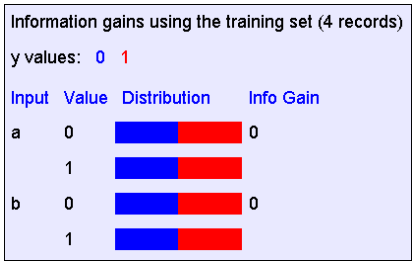
Is this a good idea?

The problem with strict hill climbing

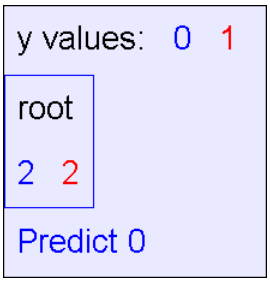
y = a XOR b

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

The information gains:



The resulting decision tree:



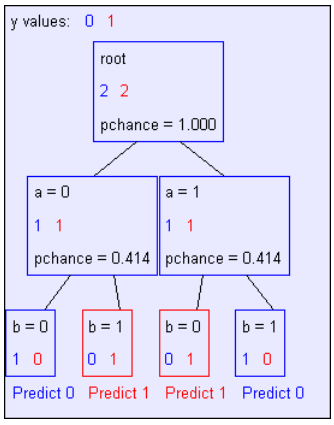
But *if we keep searching…*

y = a XOR b

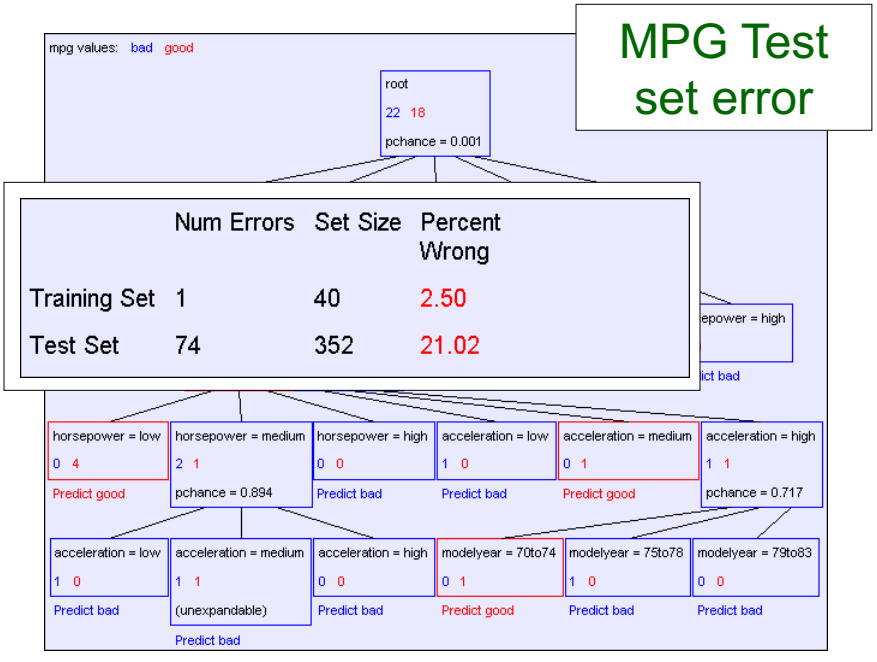
a	b	y
0	0	0
0	1	1
1	0	1
1	1	0

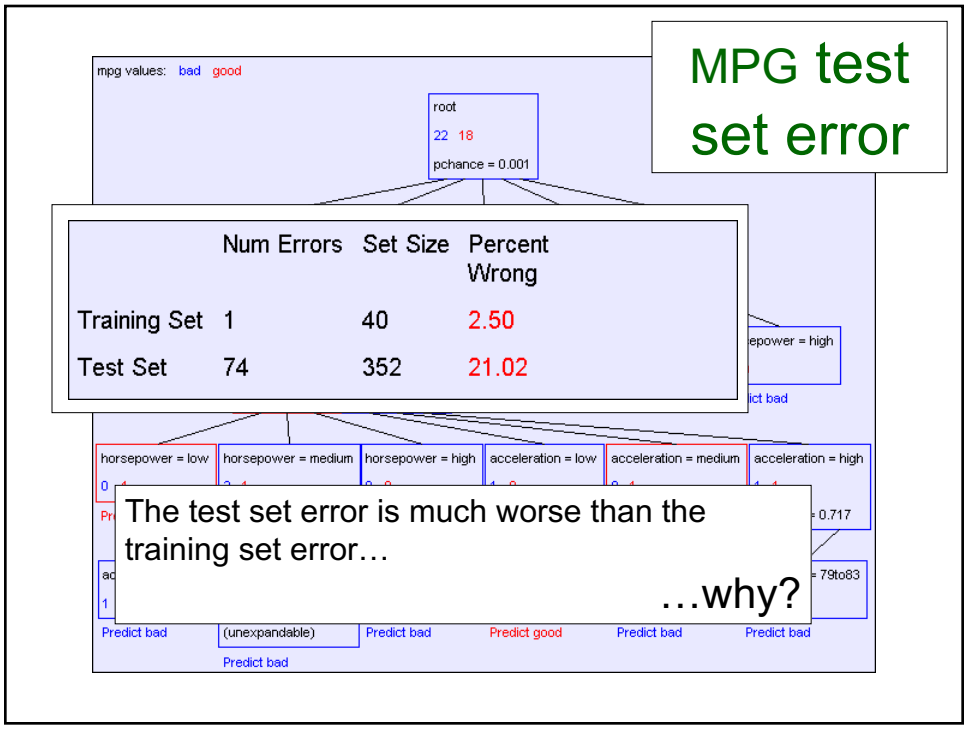
So…
what to do if no way to split with positive info gain?

The resulting decision tree:



Ok, so how does it perform?





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 uses 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

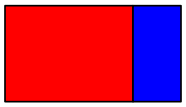
- Bound depth or # leaves
- Base Case 3
- *Doesn't work well in practice*

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***

75

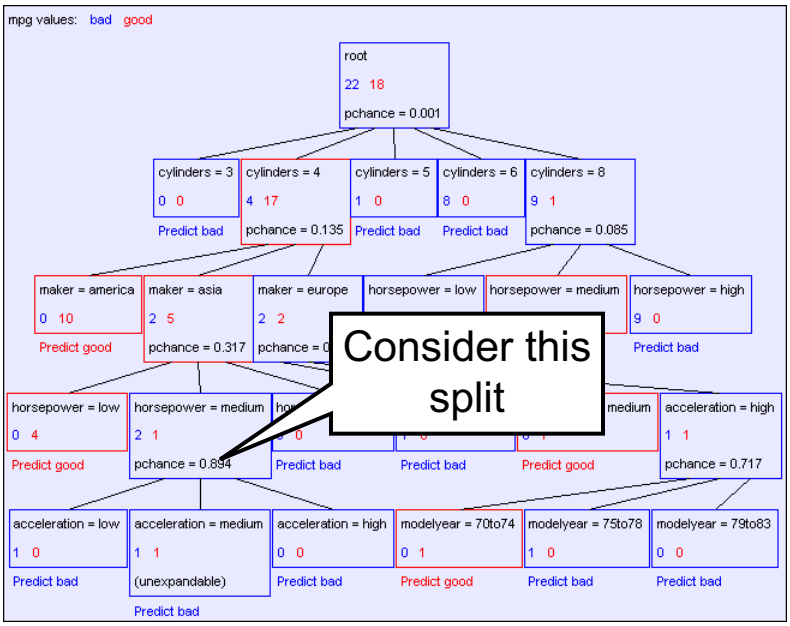
Alternatively

Chi-squared pruning

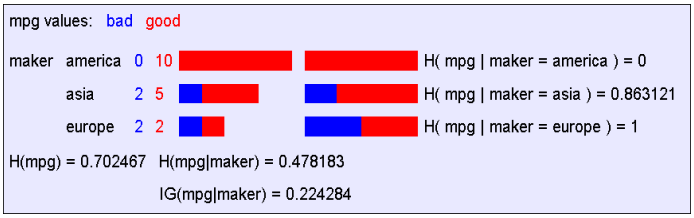
- Grow tree fully
- Consider leaves in turn
  - Is parent split worth it?

Compared to Base-Case 3?

76



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

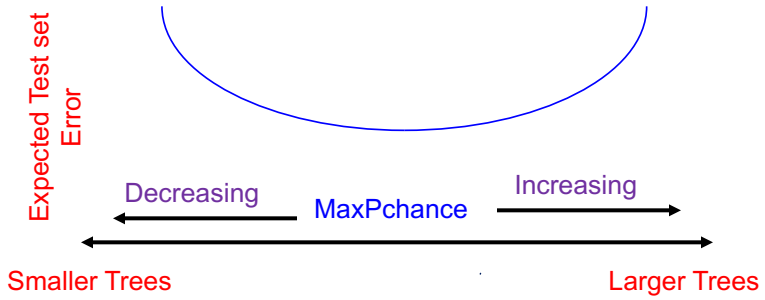
But when you can grow it no more, start to prune:

- 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!

ML as Optimization

Greedy search for best ***scoring*** hypothesis

Where ***score*** =

- Fits training data most accurately?
- Sum: ***training accuracy – complexity penalty***

regularization

82

Advanced Decision Trees

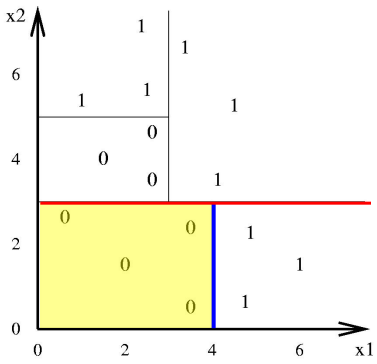
Attributes with:

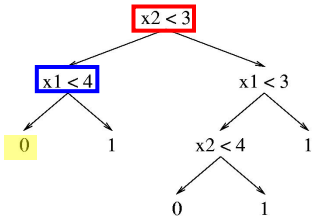
- Numerous Possible Values
- Continuous (Ordered) Values
- Missing Values

83

Decision Tree Decision Boundaries

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





decision tree summary

Decision trees are one of the most popular ML tools

- Another example of search
- Easy to understand, implement, and use
- Computationally cheap (to solve heuristically)

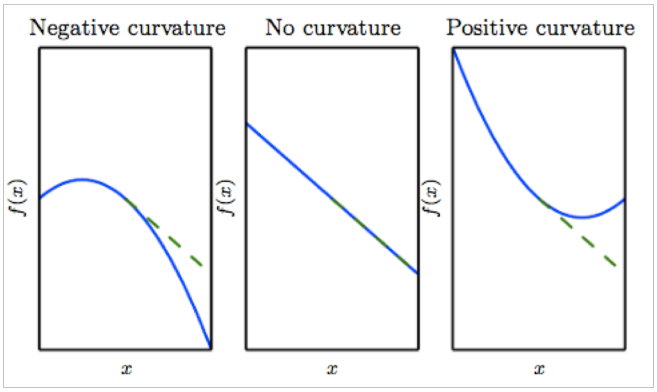
Information gain to select attributes (ID3, C4.5,…)

Presented for classification, can be used for regression and density estimation too

Decision trees will overfit!!!

- Must use tricks to find “simple trees”, e.g.,
  - Fixed depth/Early stopping
  - Pruning
  - Hypothesis testing

Higher Order Derivatives

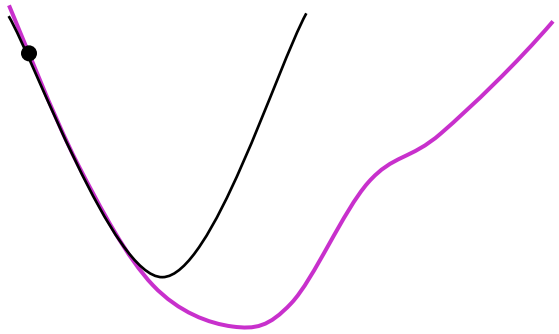


139

Newton's Method

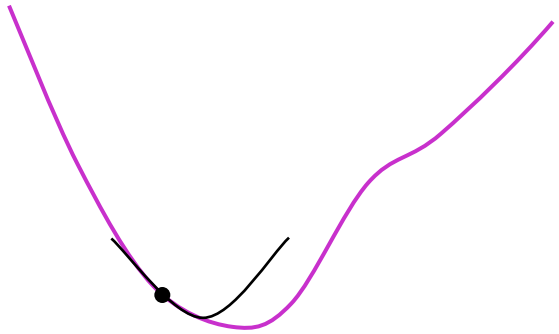
Assume function can be locally approximated with quadratic

Use both first & second derivatives



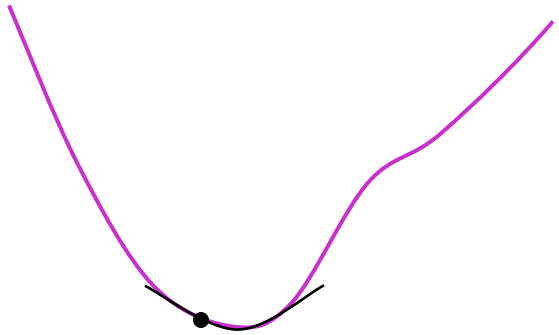
Slide from Princeton COS323 / Szymon Rusinkiewicz

Newton's Method



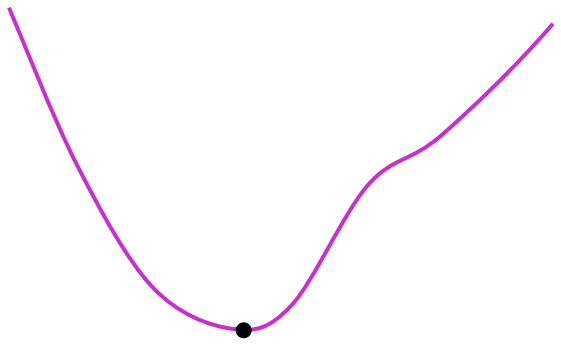
Slide from Princeton COS323 / Szymon Rusinkiewicz

Newton's Method



Slide from Princeton COS323 / Szymon Rusinkiewicz

Newton's Method



Slide from Princeton COS323 / Szymon Rusinkiewicz

Newton's Method

At each step:

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$$

Requires 1st and 2nd derivatives

Quadratic convergence

Slide from Princeton COS323 / Szymon Rusinkiewicz

Newton's Method in Multiple Dimensions

Replace 1st derivative with gradient,
2nd derivative with Hessian

$$f(x,y)$$

$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix}$$

$$H = \begin{pmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{pmatrix}$$

Slide from Princeton COS323 / Szymon Rusinkiewicz

Newton's Method in Multiple Dimensions

Replace 1st derivative with gradient,
2nd derivative with Hessian

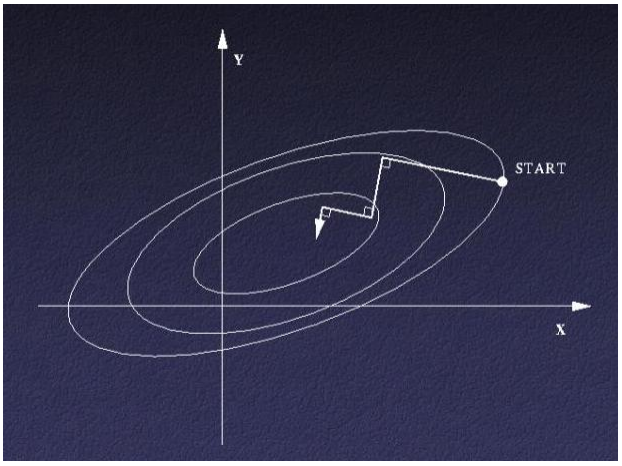
So,

$$\vec{x}_{k+1} = \vec{x}_k - H^{-1}(\vec{x}_k)\nabla f(\vec{x}_k)$$

Tends to be extremely fragile unless function very smooth and starting close to minimum

Slide from Princeton COS323 / Szymon Rusinkiewicz

Problem With Steepest Descent



Slide from Princeton COS323 / Szymon Rusinkiewicz

Conjugate Gradient Methods

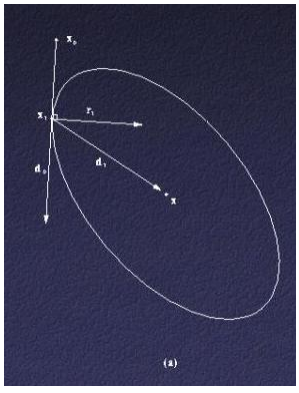
Idea: avoid "undoing" minimization that's already been done

Walk along direction

$$d_{k+1} = -g_{k+1} + \beta_k d_k$$

Polak and Ribiere formula:

$$\beta_k = \frac{g_{k+1}^{\mathrm{T}}(g_{k+1} - g_k)}{g_k^{\mathrm{T}} g_k}$$



Slide from Princeton COS323 / Szymon Rusinkiewicz

Conjugate Gradient Methods

Conjugate gradient implicitly obtains information about Hessian

For quadratic function in *n* dimensions, gets *exact* solution in *n* steps (ignoring roundoff error)

Works well in practice…

Slide from Princeton COS323 / Szymon Rusinkiewicz