

Decision Trees, cont.

Boosting

Machine Learning – 10701/15781

Carlos Guestrin

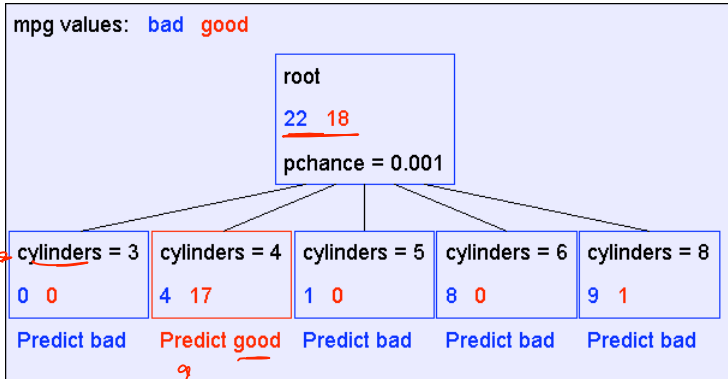
Carnegie Mellon University

October 1st, 2007

1

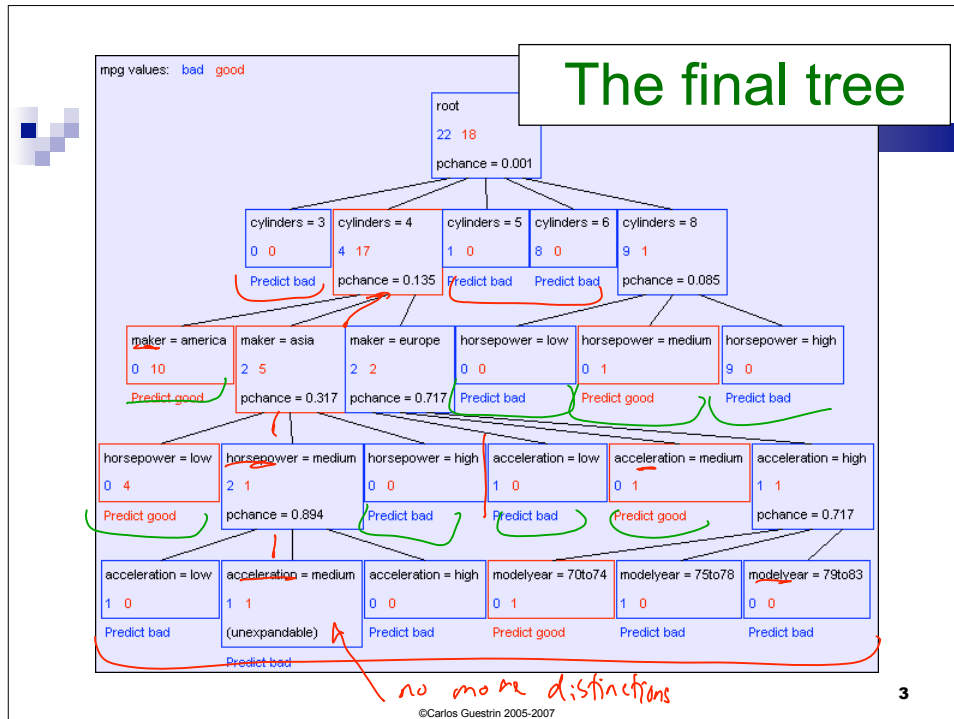
©Carlos Guestrin 2005-2007

A Decision Stump



2

©Carlos Guestrin 2005-2007



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 $\text{BuildTree}(DS_i, \text{Output})$

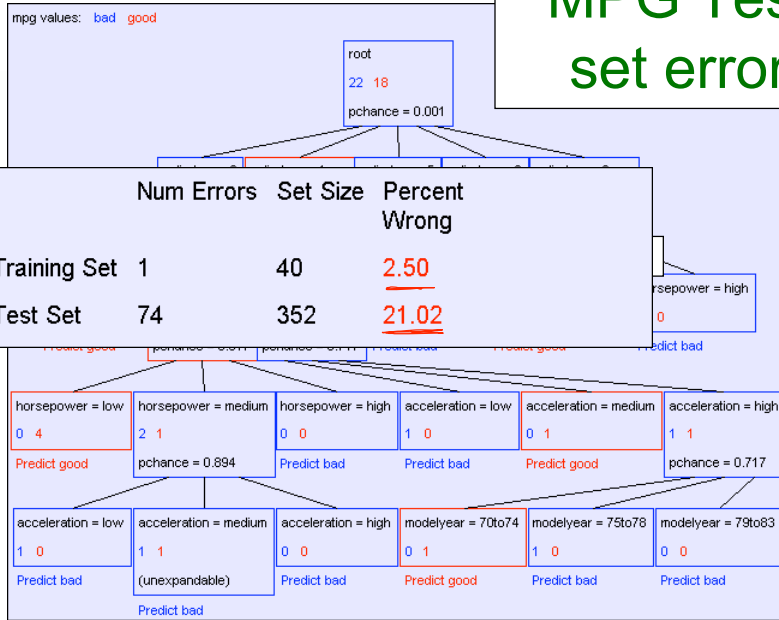
Where DS_i built consists of all those records in DataSet for which $X = i$ th distinct value of X.

Handwritten notes:

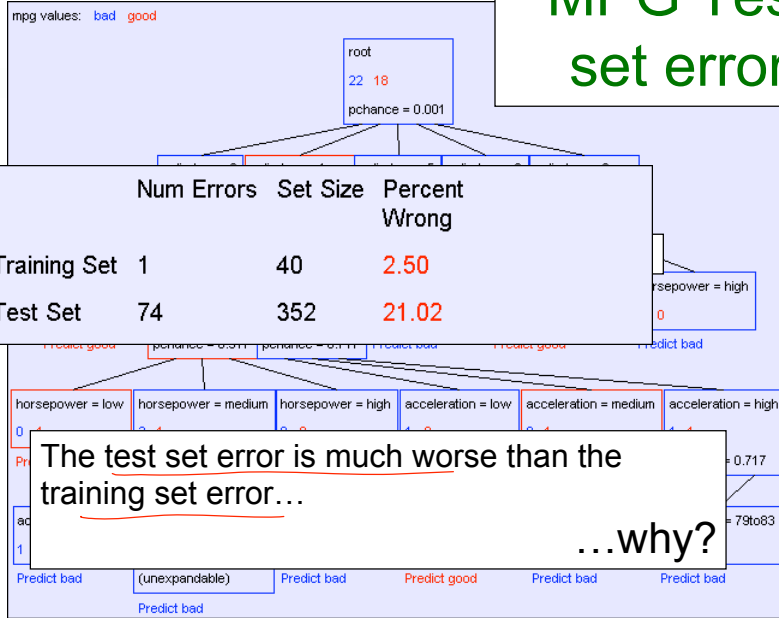
- recurse on children (pointing to the recursive call)
- recurse (pointing to the list of children)
- base cases (pointing to the first two bullet points)
- pick part of dataset consistent with child (pointing to the definition of DS_i)

©Carlos Guestrin 2005-2007

MPG Test set error



MPG Test set error



Decision trees & Learning Bias

mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	78to78	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	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	europa
bad	5	medium	medium	medium	medium	75to78	europa

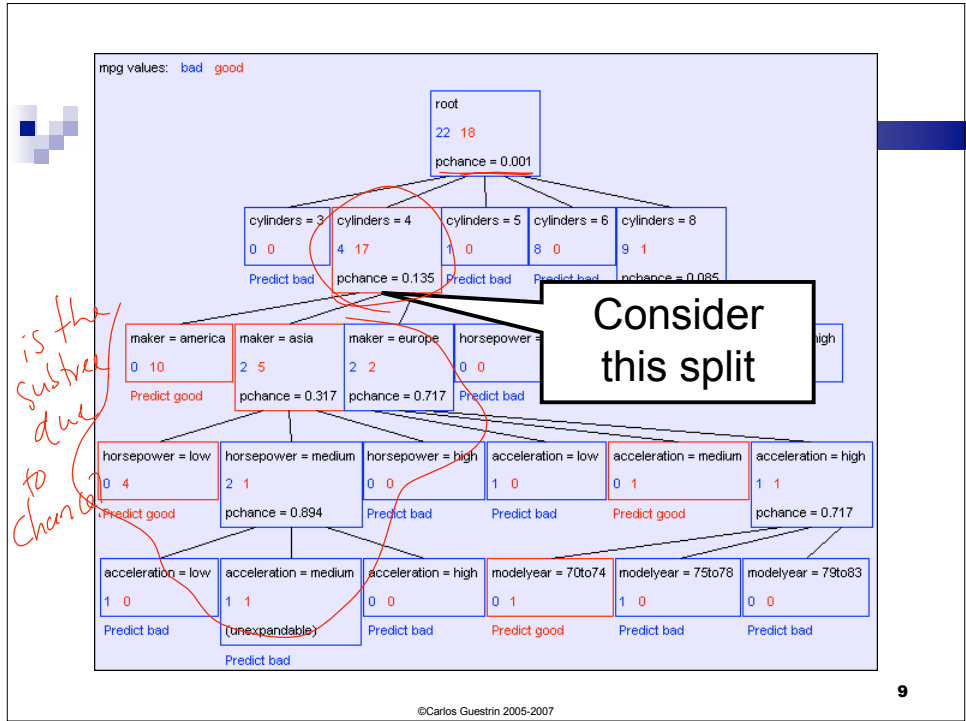
Can fit training set perfectly
 i.e., $R_{train} = 0$
 Very complex tree

no label noise!
 if $x^1 \neq x^2$
 $\therefore x^1 = x^2 \Rightarrow y^1 = y^2$
 agree on features
 agree on labels

©Carlos Guestrin 2005-2007 7

Decision trees will overfit

- Standard decision trees are have no learning bias
 - 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...



A chi-square test

mpg values: bad good

maker = america	0	10			$H(\text{mpg} \mid \text{maker} = \text{america}) = 0$
asia	2	5			$H(\text{mpg} \mid \text{maker} = \text{asia}) = 0.863121$
europe	2	2			$H(\text{mpg} \mid \text{maker} = \text{europe}) = 1$

$H(\text{mpg}) = 0.702467$ $H(\text{mpg} \mid \text{maker}) = 0.478183$
 $IG(\text{mpg} \mid \text{maker}) = 0.224284$

- 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?

A chi-square test

mpg values: bad good

maker	america	0	10		$H(\text{mpg} \mid \text{maker} = \text{america}) = 0$
	asia	2	5		$H(\text{mpg} \mid \text{maker} = \text{asia}) = 0.863121$
	europa	2	2		$H(\text{mpg} \mid \text{maker} = \text{europa}) = 1$

$H(\text{mpg}) = 0.702467$ $H(\text{mpg} \mid \text{maker}) = 0.478183$
 $IG(\text{mpg} \mid \text{maker}) = 0.224284$

- 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 7.2%

(Such simple hypothesis tests are very easy to compute, unfortunately, not enough time to cover in the lecture, but in your homework, you'll have fun! :))

11

©Carlos Guestrin 2005-2007

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 $p_{\text{chance}} > \text{MaxPchance}$
 - Continue working your way up until there are no more prunable nodes

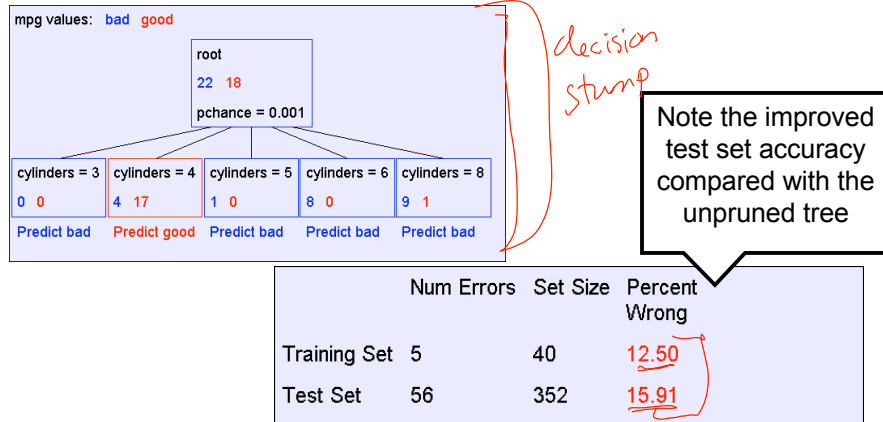
MaxPchance is a magic parameter you must specify to the decision tree, indicating your willingness to risk fitting noise

12

©Carlos Guestrin 2005-2007

Pruning example

- With $\text{MaxPchance} = 0.1$, you will see the following MPG decision tree:

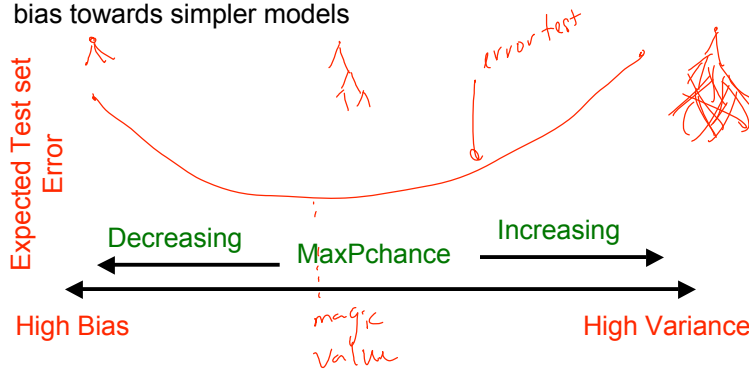


©Carlos Guestrin 2005-2007

13

MaxPchance

- Technical note MaxPchance is a regularization parameter that helps us bias towards simpler models



We'll learn to choose the value of these magic parameters soon!

©Carlos Guestrin 2005-2007

14

Real-Valued inputs

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

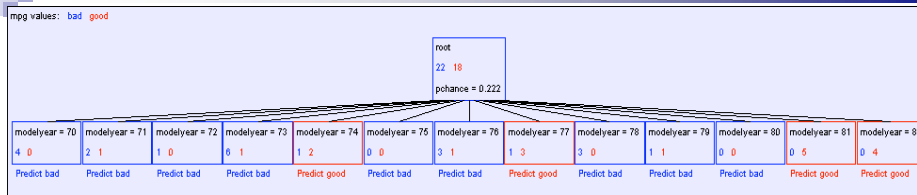
mpg	cylinders	displacemen	horsepower	weight	acceleration	modelyear	maker
good	4	97	75	2265	18.2	77	asia
bad	6	199	90	2648	15	70	america
bad	4	121	110	2600	12.8	77	europa
bad	8	350	175	4100	13	73	america
bad	6	198	95	3102	16.5	74	america
bad	4	108	94	2379	16.5	73	asia
bad	4	113	95	2228	14	71	asia
bad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

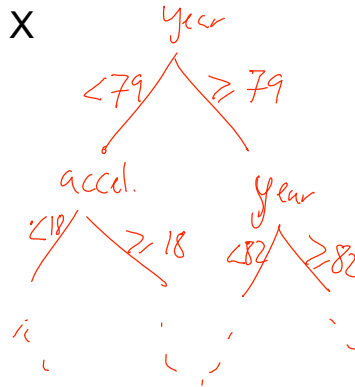


Hopeless: with such high branching factor will shatter the dataset and overfit

Threshold splits

- Binary tree, split on attribute X

- One branch: $X < t$
- Other branch: $X \geq t$



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

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
 - then pick best $x_i = \operatorname{argmax}_i IG^*(Y|x_i)$
- Note, may split on an attribute multiple times, with different thresholds

naive implementation $O(m^2)$ for m data points
 Dynamic program $O(m)$

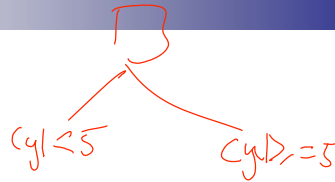
Information gains using the training set (40 records)

mpg values: bad good

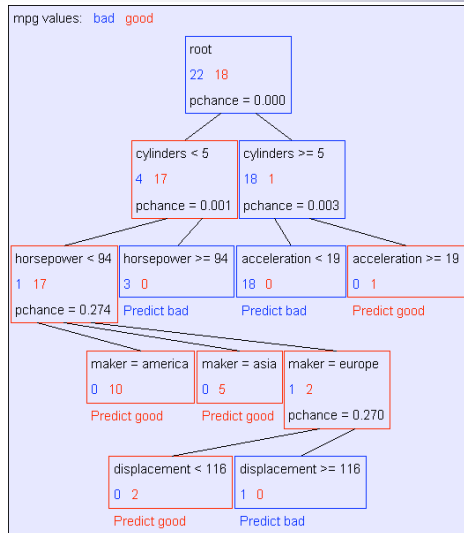
Input	Value	Distribution	Info Gain
cylinders	< 5		0.48268
	>= 5		
displacement	< 198		0.428205
	>= 198		
horsepower	< 94		0.48268
	>= 94		
weight	< 2789		0.379471
	>= 2789		
acceleration	< 18.2		0.159982
	>= 18.2		
modelyear	< 81		0.319193
	>= 81		
maker	america		0.0437265
	asia		
	europa		

best values

Example with MPG



Example tree using reals



21

©Carlos Guestrin 2005-2007

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

22

©Carlos Guestrin 2005-2007

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>

23

©Carlos Guestrin 2005-2007

Announcements

- Homework 1 due Wednesday beginning of class

- started early, started early, started early, started early,
started early, started early, started early, started early

- Exam dates set:

- Midterm: Thursday, Oct. 25th, 5-6:30pm, MM A14

- Final: Tuesday, Dec. 11, 05:30PM-08:30PM

this room

↳ Somewhere

24

©Carlos Guestrin 2005-2007

Fighting the bias-variance tradeoff

- **Simple (a.k.a. weak) learners are good**
 - e.g., naïve Bayes, logistic regression, decision stumps (or shallow decision trees)
 - Low variance, don't usually overfit
- **Simple (a.k.a. weak) learners are bad**
 - High bias, can't solve hard learning problems
- Can we make weak learners always good???
 - No!!!
 - But often yes...

25

©Carlos Guestrin 2005-2007

Voting (Ensemble Methods)

- Instead of learning a single (weak) classifier, learn **many weak classifiers** that are **good at different parts of the input space**
- **Output class:** (Weighted) vote of each classifier

$$H(x) = \text{Sign} \left\{ \sum_{t=1}^T \alpha_t h_t(x) \right\}$$

weight

$$H(x): X \mapsto Y$$

$$Y \in \{-1, +1\}$$

Simple learners:

$$h_t(x): X \rightarrow \{-1, +1\}$$

$\rightarrow [-1, +1]$

- **But how do you ???**
 - □ force classifiers to learn about different parts of the input space?
 - □ weigh the votes of different classifiers?

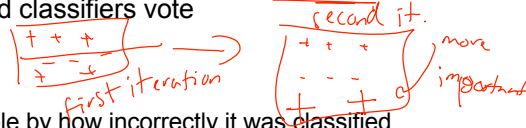
26

©Carlos Guestrin 2005-2007

Boosting [Schapire, 1989]

- Idea: given a weak learner, run it multiple times on (reweighted) training data, then let learned classifiers vote

- On each iteration t :
 - weight each training example by how incorrectly it was classified
 - Learn a hypothesis $- h_t$
 - A strength for this hypothesis $- \alpha_t$



- Final classifier:

$$H(x) = \text{sign} \left\{ \sum_{t=1}^T \alpha_t h_t(x) \right\}$$

- Practically useful
- Theoretically interesting

Learning from weighted data

- Sometimes not all data points are equal**
 - Some data points are more equal than others
- Consider a weighted dataset**
 - $D(i)$ – weight of i th training example (x^i, y^i)
 - Interpretations:
 - i th training example counts as $D(i)$ examples
 - If I were to “resample” data, I would get more samples of “heavier” data points
- Now, in all calculations, whenever used, i th training example counts as $D(i)$ “examples”**

Prior

$$\hat{P}(Y=y) = \frac{\text{Count}(Y=y)}{m}$$

weighted:

$$\hat{P}(Y=y) = \frac{\sum_{i=1}^m D(i) \mathbb{I}(y^i=y)}{\sum_{i=1}^m D(i)}$$

indicator

AdaBoost

Given: $(x_1, y_1), \dots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$ data

Initialize $D_1(i) = 1/m$. ← uniform

For $t = 1, \dots, T$: ← iteration

- Train base learner using distribution D_t .
- Get base classifier $h_t : X \rightarrow \mathbb{R}$.
- Choose $\alpha_t \in \mathbb{R}$.
- Update:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where Z_t is a normalization factor

$$Z_t = \sum_{i=1}^m D_t(i) \exp(-\alpha_t y_i h_t(x_i))$$

Output the final classifier:

$$H(x) = \text{sign} \left(\sum_{t=1}^T \alpha_t h_t(x) \right)$$

if $y_i = +1$
 $\alpha_t > 0$
 if $h_t(x_i)$ is correct:
 $\Rightarrow h_t(x_i) > 0$
 $\Rightarrow -\alpha_t y_i h_t(x_i) < 0$
 $\Rightarrow D_{t+1}(i)$ reduced

if $h_t(x_i)$ is incorrect:
 $\Rightarrow -\alpha_t y_i h_t(x_i) > 0$
 $\Rightarrow D_{t+1}(i)$ increase

if normalized $Z_t = 1$

Figure 1: The boosting algorithm AdaBoost.

©Carlos Guestrin 2005-2007

Given: $(x_1, y_1), \dots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$

Initialize $D_1(i) = 1/m$.

For $t = 1, \dots, T$:

- Train base learner using distribution D_t .
- Get base classifier $h_t : X \rightarrow \mathbb{R}$.
- Choose $\alpha_t \in \mathbb{R}$.
- Update:

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right)$$

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

ϵ_t is weighted error of $h_t(x)$
 iteration t :
 really trust

$$\epsilon_t = P_{i \sim D_t} [x^i \neq y^i]$$

as $\epsilon_t \rightarrow 0, \alpha_t \rightarrow +\infty$

$$\epsilon_t = \frac{1}{\sum_{i=1}^m D_t(i)} \sum_{i=1}^m D_t(i) \delta(h_t(x_i) \neq y_i)$$

as $\epsilon_t \rightarrow 1, \alpha_t \rightarrow -\infty$ really trust opposite

if $\epsilon_t = 0.5, \alpha_t = 0$, random classifiers are bad \Rightarrow zero weight

©Carlos Guestrin 2005-2007

What α_t to choose for hypothesis h_t ?

[Schapire, 1989]

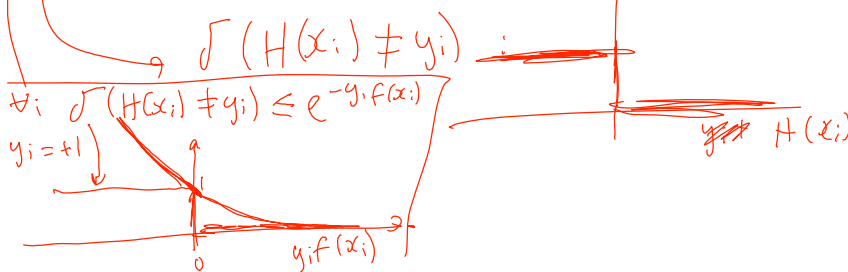
Training error of final classifier is bounded by:

bound (count # mistakes in training)

$$\text{error}_{\text{train}}(H) = \frac{1}{m} \sum_{i=1}^m \delta(H(x_i) \neq y_i) \leq \frac{1}{m} \sum_{i=1}^m \exp(-y_i f(x_i))$$

Where $f(x) = \sum_t \alpha_t h_t(x); H(x) = \text{sign}(f(x))$

if $y_i = +1$



What α_t to choose for hypothesis h_t ?

[Schapire, 1989]

Training error of final classifier is bounded by:

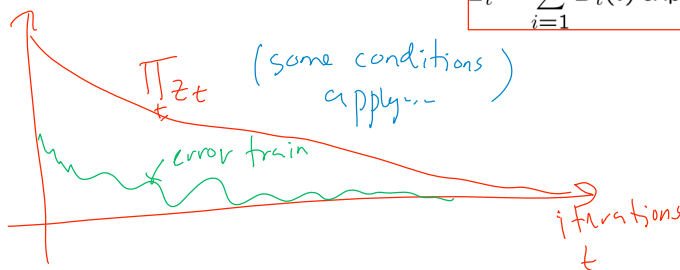
[magic of telescoping sums]
Your homework answer

$$\frac{1}{m} \sum_{i=1}^m \delta(H(x_i) \neq y_i) \leq \frac{1}{m} \sum_{i=1}^m \exp(-y_i f(x_i)) = \prod_{t=1}^T Z_t$$

Where $f(x) = \sum_{t=1}^T \alpha_t h_t(x); H(x) = \text{sign}(f(x))$

upper bound on the error

$$Z_t = \sum_{i=1}^m D_t(i) \exp(-\alpha_t y_i h_t(x_i))$$



What α_t to choose for hypothesis h_t ?

[Schapire, 1989]

Training error of final classifier is bounded by:

$$\frac{1}{m} \sum_{i=1}^m \delta(H(x_i) \neq y_i) \leq \frac{1}{m} \sum_i \exp(-y_i f(x_i)) = \prod_t Z_t$$

Where $f(x) = \sum_t \alpha_t h_t(x)$; $H(x) = \text{sign}(f(x))$

If we minimize $\prod_t Z_t$, we minimize our training error

Z_{t-1} doesn't depend on α_t, h_t

We can tighten this bound greedily, by choosing α_t and h_t on each iteration to minimize Z_t .

$$Z_t = \sum_{i=1}^m D_t(i) \exp(-\alpha_t y_i h_t(x_i))$$

33

©Carlos Guestrin 2005-2007

What α_t to choose for hypothesis h_t ?

[Schapire, 1989]

We can minimize this bound by choosing α_t on each iteration to minimize Z_t .

$$Z_t = \sum_{i=1}^m D_t(i) \exp(-\alpha_t y_i h_t(x_i))$$

For boolean target function, this is accomplished by [Freund & Schapire '97]:

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right)$$

You'll prove this in your homework! ☺

34

©Carlos Guestrin 2005-2007

Strong, weak classifiers

- If each classifier is (at least slightly) better than random
 - $\epsilon_t < 0.5$

- AdaBoost will achieve zero training error (exponentially fast):

$$\frac{1}{m} \sum_{i=1}^m \delta(H(x_i) \neq y_i) \leq \prod_t Z_t \leq \exp \left(-2 \sum_{t=1}^T (1/2 - \epsilon_t)^2 \right) \ll e^{-2T\gamma^2}$$

exponentially fast
 $e^{-2T\gamma^2}$
 exponentially fast

$(1/2 - \epsilon_t)^2$ ← how much better is ϵ_t than random

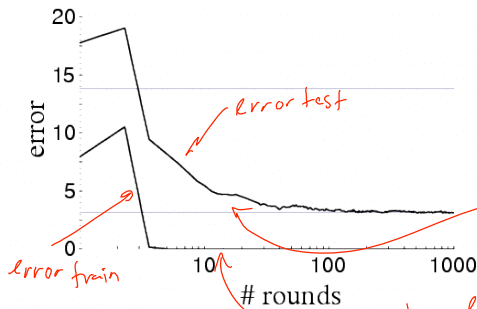
- Is it hard to achieve better than random training error?

$$|1/2 - \epsilon_t| \geq \gamma$$

~~Is it hard to achieve better than random training error?~~
 T

Boosting results – Digit recognition

[Schapire, 1989]



- Boosting often
 - Robust to overfitting
 - Test set error decreases even after training error is zero